# PREDIKSI HARGA TELEVISI DENGAN MENGGUNAKAN PENERAPAN METODE *RANDOM FOREST* DAN *FRAMEWORK FLASK*

**TUGAS AKHIR**


**Disusun Oleh:**

**RISMITA WAHYU WIDYASTUTI**

**16611088**

**PROGRAM STUDI STATISTIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS ISLAM INDONESIA**

**YOGYAKARTA**

**2020**

**HALAMAN PERSETUJUAN PEMBIMBING**

**TUGAS AKHIR**

| | | |
|---|---|---|
| Judul | : | Prediksi Harga Televisi dengan Menggunakan Penerapan Metode *Random Forest* dan *Framework Flask.* |
| Nama Mahasiswa | : | Rismita Wahyu Widyastuti |
| Nomor Mahasiswa | : | 16611088 |

**TUGAS AKHIR INI TELAH DIPERIKSA DAN DISETUJUI UNTUK DIAJUKAN**

Yogyakarta, 29 April 2020

Menyetujui,

**Pembimbing**

**(Dr. Raden Bagus Fajriya Hakim, S.Si., M.Si.)**

**HALAMAN PENGESAHAN**

**TUGAS AKHIR**

**PREDIKSI HARGA TELEVISI DENGAN MENGGUNAKAN PENERAPAN METODE *RANDOM FOREST* DAN *FRAMEWORK FLASK***

**Nama : Rismita Wahyu Widyastuti**

**NIM : 16611088**

**TUGAS AKHIR INI TELAH DIAJUKAN**

**PADA TANGGAL 29 April 2020**

| **Nama Penguji** | **Tanda Tangan** |
|---|---|
| 1. Dina Tri Utari, S.Si., M.Sc. | ..... |
| 2. Tuti Purwaningsih, S.Stat., M.Si | ..... |
| 3. Dr. Raden Bagus Fajriya Hakim, S.Si., M.Si. | ..... |

Mengetahui,
Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Islam Indonesia

**Prof. Riyanto, S.Pd., M.Si., Ph.D.**

## KATA PENGANTAR

***Assalamu'alaikum Wr.Wb***

*Alhamdulillahirabbil'alamin*, puji syukur penulis panjatkan kehadirat Allah SWT yang Maha Esa, yang telah melimpahkan rahmat dan karunia-Nya, sehingga penulis dapat menyelesaikan laporan Tugas Akhir ini. Shalawat serta salam penulis haturkan kepada junjungan Nabi Muhammad SAW beserta keluarga, sahabat, dan umatnya.

Laporan Tugas Akhir yang berjudul "Prediksi Harga Televisi dengan Menggunakan Penerapan Metode *Random Forest* dan *Framework Flask*" disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana Statistika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Islam Indonesia.

Penulis dalam menyelesaikan laporan Tugas Akhir ini banyak mendapatkan bantuan dari berbagai pihak, baik berupa saran, bimbingan maupun bantuan lainnya sehingga dapat memperlancar penyusunan laporan Tugas Akhir ini. Oleh karena itu, dalam kesempatan ini penulis ingin menyampaikan ucapan terima kasih kepada :

1. Bapak Fathul Wahid, S.T., M.Sc., Ph.D. Selaku Rektor Universitas Islam Indonesia
2. Bapak Prof. Riyanto, S.Pd., M.Si., Ph.D. Selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Islam Indonesia.
3. Bapak Dr. Edy Widodo, S.Si., M.Si. Selaku Kepala Program Studi Statistika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Islam Indonesia.
4. Bapak Dr. R. B. Fajriya Hakim, S.Si., M.Si. Selaku Dosen Pembimbing Tugas Akhir atas segala waktu yang diberikan dalam memberikan bimbingan kepada penulis selama proses pengerjaan Tugas Akhir.
5. Seluruh Dosen Pengajar di Program Studi Statistika Universitas Islam Indonesia yang telah memberikan ilmu kepada penulis.
6. Orang tua yang senantiasa memberikan kasih sayang, dukungan dan

doanya sehingga penulis dapat menyelesaikan studi Starta Satu (S1) di Jurusan Statistika, Universitas Islam Indonesia.

7. Rahadyan Nandiwardhana, terima kasih telah menjadi salah satu *support system* di kehidupan penulis.
8. Teman-teman Statistika UII Angkatan 2016 (ARTCOS) yang bersama-sama menjadi pejuang gelar S.Stat.
9. Serta semua pihak lainnya yang tidak bisa dituliskan penulis satu per satu yang telah membantu selama pembuatan Tugas Akhir ini.

Demikian laporan Tugas Akhir ini, penulis mengucapkan terima kasih kepada semua pihak yang telah membantu penulis sehingga laporan Tugas Akhir ini dapat diselesaikan. Penulis menyadari bahwa laporan Tugas Akhir ini masih jauh dari kata sempurna dan masih banyak kekurangan. Oleh karena itu, penulis mengharapkan kritik dan saran yang membangun dari pembaca untuk dapat menyempurnakan penulisan laporan Tugas Akhir ini. Semoga laporan Tugas Akhir ini dapat memberikan manfaat khususnya bagi penulis dan umumnya bagi semua pihak yang membaca. Akhir kata, semoga Allah SWT senantiasa melimpahkan rahmat serta hidayah-Nya.

***Wassalamualaikum Wr.Wb***

**Yogyakarta, 29 April 2020**

Penulis

# DAFTAR ISI

# DAFTAR TABEL

# DAFTAR GAMBAR

## PERNYATAAN

Dengan ini saya menyatakan bahwa dalam Tugas Akhir ini tidak terdapat karya yang sebelumnya pernah diajukan untuk memperoleh gelar kesarjanaan di suatu Perguruan Tinggi dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan orang lain, kecuali yang diacu dalam naskah ini dan disebutkan dalam daftar pustraka.

Yogyakarta, 29 April 2020


(Rismita Wahyu Widyastuti)

# PREDIKSI HARGA TELEVISI DENGAN MENGGUNAKAN PENERAPAN METODE *RANDOM FOREST* DAN *FRAMEWORK FLASK*

**Rismita Wahyu Widyastuti**

**Program Studi Statistika Fakultas Matematika dan Ilmu Pengetahuan Alam**

**Universitas Islam Indonesia**

## INTISARI

Dengan adanya perkembangan teknologi informasi dan komunikasi memiliki peranan yang penting bagi masyarakat dalam memenuhi kebutuhan informasi, karena berbagai aspek kehidupan tidak dapat luput dari teknologi informasi dan komunikasi. Media hasil teknologi yang dapat menyampaikan informasi secara massa yaitu media massa. Media massa terbagi menjadi dua jenis yaitu media cetak dan media elektronik. Televisi adalah media massa elektronik yang paling efektif dalam penyebaran informasi menggunakan perpaduan antara media *audio* dan *visual*. Televisi banyak diminati oleh masyarakat Indonesia, karena televisi sebagai media komunikasi yang cepat, relatif murah, dan dapat juga sebagai media hiburan. Tujuan dalam penelitian ini adalah untuk memprediksi harga penjualan televisi dengan mengimplementasi *machine learning* yaitu *Random Forest.* Variabel independen yang digunakan antara lain: nama produk, merek, ukuran layar dan tipe layar, sedangkan variabel dependen adalah harga. *Random Forest* merupakan pengembangan dari *Decision Tree* model *Classification and Regression Tree* (CART). Model *Random Forest* dianalisis dengan bahasa pemrograman *python* dan diaplikasikan menjadi sebuah *website* dengan menggunakan *framework flask.* Berdasarkan hasil analisis *random forest* menggunakan data *training* dan data *testing* dengan perbandingan 75%:25%. Dilakukan pelatihan pertama dengan membuat model *random forest regressor* menggunakan parameter *default* menghasilkan akurasi sebesar 75,75%. Dalam meningkatkan akurasi dilakukan *tuning hyperparameter* untuk mendapatkan parameter optimal antara lain: *n_estimators* ($n_{tree}$) optimal sebanyak 1.000 pohon dan parameter *max_features* ($m_{try}$) optimal sebanyak 4 variabel. Dilakukan pelatihan kedua dengan membuat model *random forest regressor* menggunakan parameter optimal hasil *tuning hyperparameter* sehingga didapatkan akurasi sebesar 75,81%.

**Kata Kunci:** Televisi, *Random Forest, Flask, Machine Learning, Website*

# *PRICE PREDICTION OF TELEVISION USING RANDOM FOREST ALGORITHM AND FLASK FRAMEWORK*

**Rismita Wahyu Widyastuti**

**Statistics Department, Faculty of Mathematics and Natural Science**
**University Islam Indonesia**

## ABSTRACT

*Development of technology, the role of information and communication technology is very important for the community in meeting information needs because various aspects of life cannot be separated from information technology. Mass media is divided into two types, namely print media and electronic media. Television is the most effective electronic mass media in the dissemination of information using a combination of sound (audio) and image (visual) media. Television is much in demand by the people of Indonesia, because television is a medium of communication that is fast, relatively inexpensive, and can also be a medium of entertainment. The purpose of this research is to predict the price of television sales by implementing machine learning, namely Random Forest. The independent variables used are product name, brand, layer size and layer type, while the dependent variable is price. Random Forest is a development of the Decision Tree model of Classification and Regression Tree (CART). The Random Forest model was analyzed using the python programming language and made into a website using the framework flask and HTML until the deployment process to Heroku. Based on the results of random forest analysis using training data and testing data with a ratio of 75%: 25%. The first random forest model using default parameters resulting in an accuracy of 75.75%. In improving the accuracy use tuning hyperparameter to get optimal parameters, including: n_estimators (n_tree) of 1.000 trees and max_features (m_try) of 4 variables. The second random forest model using the optimal parameters of result tuning hyperparameter so that an accuracy of 75.81%.*

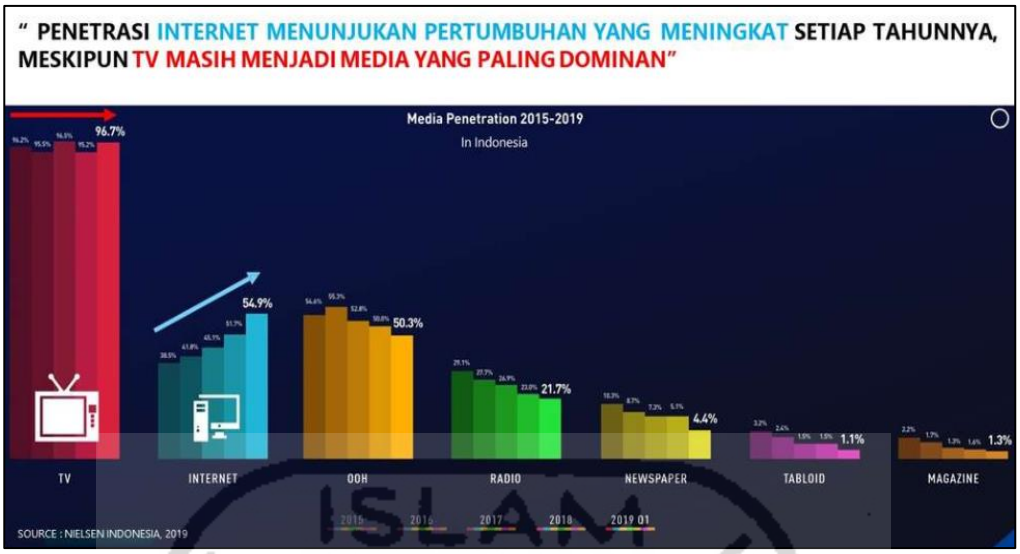
***Key Words*:** *Television, Random Forest, Flask, Machine Learning, Website*

# BAB I
# PENDAHULUAN

## 1.1 Latar Belakang Masalah

Dengan adanya perkembangan teknologi di bidang informasi dan komunikasi telah membawa perubahan yang berdampak besar dalam berbagai aspek kehidupan manusia. Dalam era globalisasi, peranan teknologi informasi dan komunikasi sangat penting bagi masyarakat dalam memenuhi kebutuhan akan informasi. Hal ini dikarenakan berbagai aspek kehidupan masyarakat tidak dapat luput dari teknologi informasi dan komunikasi. Masyarakat sangat membutuhkan informasi dengan kualitas informasi yang aktual dan akurat dalam pemenuhan kebutuhan informasi dan memperluas pengetahuan. Salah satu sumber informasi yang dapat digunakan untuk memenuhi kebutuhan dalam menyediakan informasi yaitu media massa. Media yang dimaksud adalah hasil teknologi sebagai media komunikasi yang dapat menyampaikan informasi secara massa.

Media massa merupakan sarana penyampaian komunikasi dan informasi dengan penyebarannya kepada khalayak yang bersifat massa dan dapat diakses secara luas. (Wiryanto, 2004). Pengertian media massa dijelaskan bahwa media massa menghasilkan suatu produk informasi berupa berita, hiburan, ruang publik, ekonomi, budaya, dan politik yang dapat diakses oleh semua lapisan masyarakat. Media massa terbagi menjadi dua jenis yaitu media cetak dan media elektronik. Media cetak adalah media komunikasi massa melalui tulisan yang tercetak, seperti surat kabar, majalah, dan buku. Sedangkan media massa elektronik adalah media komunikasi melalui perangkat elektronik seperti televisi dan radio. Berdasarkan kedua jenis media massa tersebut, diketahui bahwa media elektronik lebih diminati oleh masyarakat Indonesia, terutama televisi. Teknologi yang berkembang pesat semakin membuat media massa elektronik memiliki kelebihan yang terletak pada kecanggihan tampilan *visual*, warna dan suara. Media televisi merupakan media elektronik yang banyak diminati oleh masyarakat Indonesia, karena televisi sebagai media komunikasi yang cepat, relatif murah, dan dapat juga sebagai media hiburan yang menarik.




**Gambar 1.1** Media Penetrasi yang Diakses Pada Tahun 2015-2019

Sebuah data hasil riset dari Nielsen *Company Consumer Media View* tahun 2019, media penetrasi televisi masih memimpin dengan 96,7% disusul dengan internet (54,9%), media luar ruang (50,3%), radio (21,7%), koran (4,4%), tabloid (1,1%) dan majalah (1,3%). Meski banyak orang yang mengklaim tak lagi menonton televisi, hasil survei yang dilakukan oleh Nielsen menunjukkan hal sebaliknya. Data Nielsen *Consumer Media View & Radio Audience Measurenment* pada kuartal kedua 2019 menunjukkan bahwa mayoritas dari sekitar 17 ribu responden berusia di atas 10 tahun sekitar 96 persen menjawab masih menonton televisi. (Azizah, 2019)

Televisi merupakan salah satu media massa yang paling efektif dalam penyebaran informasi yang penyampaian informasinya menggunakan perpaduan antara media suara (*audio*) dan gambar (*visual*). Televisi merupakan salah satu media massa yang memberikan informasi (fungsi informatif) sehingga masyarakat dapat mengetahui informasi terkini melalui berita yang disiarkan. Sebagai alat yang mendidik (fungsi edukatif) yang dapat memperluas wawasan dan keterampilan. Selain itu, televisi juga sebagai media penghibur (fungsi *entertaintment*) yang dapat memberikan hiburan.

Sejarah televisi lahir karena perkembangan teknologi dalam penemuan hukum gelombang elektromagnetik yang ditemukan oleh Michael Faraday di tahun 1831. Pada tahun 1884, Ilmuan Paul Nipkov dalam gagasannya berhasil menciptakan teleskop elektrik yang menyalurkan sinyal gambar untuk

mengirim gambar elektronik melalui udara. Pada saat itulah Paul Nipkow mendapat julukan sebagai Bapak Televisi. (J.B, Wahyudi, 1983)

Perkembangan teknologi yang mendorong dalam menciptakan inovasi untuk mempermudah pekerjaan manusia di bidang perdagangan. Dengan kemajuan teknologi internet dalam dunia bisnis perdagangan, menjadikan alat untuk mengaplikasikan strategi bisnis melalui internet yang dikenal sebagai *e-commerce* yaitu aktivitas penjualan, pembelian, pemasaran produk dengan jaringan internet. Di Indonesia, perdagangan melalui *e-commerce* semakin meningkat, karena dari sebelumnya semua proses yang dilakukan secara manual berubah menjadi lebih mudah dan cepat dengan mengakses *e-commerce* tanpa perlu mengunjungi toko secara langsung. Salah satu jenis produk yang saat ini juga sudah mulai merambah menggunakan *e-commerce* adalah produk-produk elektronik salah satunya televisi. (Heriyanti, 2018)

Perkembangan teknologi yang berada di era digital, setiap aspek kehidupan manusia sangat berhubungan dengan teknologi komputasi yang diikuti dengan berkembangnya kebutuhan data. Untuk memproses sebuah data, dibutuhkan model untuk menjelaskan data tersebut dengan pengaplikasianya menggunakan *Machine Learning*. *Machine Learning* merupakan salah satu cabang dari ilmu Kecerdasan Buatan (*Artificial Intelegence*) yang kemampuan komputer untuk melakukan pembelajaran tanpa harus menjelaskan secara eksplisit kepada komputer. *Machine learning* merujuk pada sebuah metode yang membuat komputer memiliki kemampuan dalam mempelajari dan melakukan sebuah pekerjaan secara otomatis. (Samuel, 1959).

Salah satu model yang sederhana dan sering digunakan di data science dalam pemodelan prediksi data adalah model *Random Forest.* Definisi *random forest* adalah salah satu metode pengklasifikasian yang terdiri dari kumpulan pohon terstruktur dan setiap pohon memberikan suara (*vote*) untuk menentukan kelas dengan jumlah *vote* terbanyak. Dengan kata lain, *Random Forest* terdiri dari kumpulan *decision tree* yang tidak saling berkorelasi dan digunakan untuk mengklasifikasi data ke dalam suatu kelas. Konsep *random forest* dengan terdapatnya banyak pohon *(tree)* yang dihasilkan yang akhirnya membentuk sebuah hutan (*forest*). Metode ini menghubungkan banyak pohon untuk

membuat klasifikasi dan kelas prediksi. *Random Forest* menjadi salah satu metode klasifikasi populer untuk diimplementasikan di berbagai bidang. (Adnyana, 2016).

Pengembangan *website* adalah menciptakan konsep, membuat, dan mengoperasikan *website*. Dengan perkembangan teknologi saat ini, terdapat banyak alat untuk *web development*, salah satunya adalah *web framework* yang dibuat menggunakan bahasa pemrograman *Python*. *Framework web development* pada *Python* yang paling populer digunakan adalah *Django* dan *Flask*. *Flask* adalah *microframework web python* yang membantu membuat kerangka *website* dan *Flask* dapat digunakan untuk pengembang *website* untuk pemula karena mudah untuk dipelajari dan mudah digunakan. Dalam mengaplikasikan model *machine learning* akan dibuat menjadi sebuah *website application* dengan menggunakan *framework Flask* dan Heroku.

Berdasarkan latar belakang yang telah diuraikan, maka dalam penelitian ini akan dilakukan implementasi *machine learning* dengan menggunakan metode *Random Forest* dan diaplikasikan menjadi sebuah *website application* dengan menggunakan *framework Flask* dan Heroku yang digunakan untuk memprediksi harga televisi di Indonesia.

## 1.2 Rumusan Masalah

Berdasarkan penjelasan latar belakang, maka rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana membuat model *machine learning* dengan mengimplementasikan metode *Random Forest* yang digunakan untuk memprediksi harga televisi?
2. Bagaimana tingkat akurasi yang dihasilkan dengan mengimplementasikan metode *Random Forest* dalam memprediksi harga televisi?
3. Variabel apa yang memberikan pengaruh terbesar dalam mempengaruhi harga televisi?
4. Bagaimana membuat sebuah *website* dengan mengimplementasikan metode *Random Forest* dan *framework Flask* dalam bahasa *Python* yang digunakan untuk memprediksi harga televisi?

## 1.3 Batasan Masalah

Dalam penelitian ini terdapat beberapa batasan sehingga penelitian dapat lebih spesifik, beberapa batasan masalah yang dimiliki sebagai berikut:

1. *Dataset* yang digunakan adalah data harga penjualan televisi yang didapatkan dengan cara *scrapping web* diambil dari *website* Pricebook.
2. Metode yang digunakan dalam penelitian yaitu metode *Random Forest* yang digunakan untuk memprediksi harga televisi.
3. Dalam penelitian ini menggunakan empat variabel prediktor untuk memprediksi harga televisi antara lain: nama produk, merek, tipe layar, dan ukuran layar (satuan *inch*).
4. Dalam pembuatan *website* menggunakan *framework flask* dengan bahasa pemrograman *Python* dan proses *deploy website* menggunakan Heroku.
5. Alat pendukung dalam penelitian ini adalah *Ms Excel* dan *Python.*

## 1.4 Tujuan Penelitian

Tujuan dalam penelitian ini sebagai berikut:

1. Mengetahui pembuatan model dengan mengimplementasikan metode *Random Forest* yang digunakan untuk memprediksi harga televisi.
2. Mengetahui tingkat akurasi yang dihasilkan berdasarkan hasil implementasi metode *Random Forest* dalam memprediksi harga televisi.
3. Mengetahui variabel yang memberikan pengaruh terbesar dalam mempengaruhi harga televisi.
4. Mengetahui proses pembuatan *website* dengan mengimplementasikan metode *Random Forest* dan *framework Flask* dalam bahasa pemrograman *Python* yang digunakan untuk memprediksi harga televisi.

## 1.5 Manfaat Penelitian

Manfaat yang diharapkan dari penelitian ini adalah memberikan tambahan wawasan tentang implementasi *machine learning* dalam memprediksi harga televisi dengan menggunakan metode *Random Forest* dan pembuatan *website* dengan menggunakan *framework flask.* Selain itu, dalam penelitian ini juga diharapkan dengan *output* yang dihasilkan berupa *website* dapat membantu masyarakat Indonesia (konsumen) yang ingin membeli televisi sesuai dengan kebutuhan dan harga yang sesuai kemampuan daya beli.

# BAB II
# TINJAUAN PUSTAKA

Berhubungan dengan penelitian yang sedang dilakukan, diperlukan untuk merujuk beberapa referensi dari penelitian terdahulu dengan tujuan untuk mengetahui hubungan antara penelitian yang penulis lakukan dengan penelitian terdahulu sehingga dapat menghindari suatu duplikasi dalam penelitian yang akan dilakukan.

Penelitian yang dilakukan oleh Agenda Yudha Samudra, mengenai pendekatan *random forest* untuk model peramalan harga tembakau rajangan di Kabupaten Temanggung. Dalam penelitiannya mengenai tidak adanya standarisasi harga dan mutu tembakau menyebabkan harga kerugian di pihak petani dikarenakan untuk menentukan harga tembakau rajangan masih bergantung kepada *grader* pabrik. Oleh karena itu, dalam penelitian ini dibangun perangkat lunak untuk memprediksi harga tembakau rajangan di kabupaten Temanggung dengan menggunakan metode *Random Forest* dan diimplementasikan menggunakan *Python*. Metode *Random Forest* merupakan pengembangan dari CART dengan menerapkan metode *bagging* dan *random feature selection*. Tidak hanya menghasilkan satu pohon tetapi puluhan hingga ratusan pohon dari data *replacement* yang selanjutnya dilakukan pengumpulan informasi sehingga *Random Forest* dapat dihasilkan pohon dengan ukuran yang berbeda-beda. Ini dikarenakan pohon yang dihasilkan oleh dari *bootstrap* dan *replacement* tidak dilakukan pemangkasan. Pada proses klasifikasi, didasarkan pada *vote* dari suara terbanyak pada kumpulan populasi pohon, sedangkan untuk regresi menggunakan hasil rata-rata populasi pohon. Atribut yang digunakan dalam penelitian ini adalah suhu rata-rata, kelembaban rata-rata, curah hujan, lama penyinaran, seri, g*rade* dan harga. Berdasarkan hasil penelitian ini dapat diketahui bahwa dengan menggunakan data iklim dan penjualan tembakau sebanyak 1244 *record*, menunjukkan bahwa hasil prediksi mencapai akurasi 82.39%. (Samudra A. , 2019).

Penelitian yang dilakukan oleh Vanissa Wanika Siburian dan Ika Elvina Mulyana, mengenai prediksi harga ponsel menggunakan metode *random forest*. Dalam penelitiannya mengenai harga ponsel dapat dipengaruhi oleh spesifikasi yang dimiliki ponsel tersebut. Dengan spesifikasi harga dapat ditentukan. Pada

penelitian ini dilakukan klasifikasi untuk memprediksi harga ponsel dengan spesifikasi yang diberikan dengan metode Random Forest. Berdasarkan hasil penelitian ini dapat diketahui bahwa pengklasifikasian ini menggunakan tujuh variabel prediksi dan satu variabel respon menghasilkan akurasi sebesar 81%. Kemudian tingkat akurasi tertinggi pada variabel respon terdapat pada kategori harga murah. (Siburian & Mulyana, 2018).

Penelitian yang dilakukan oleh Reinardus Aji Haristu**,** mengenai penerapan metode *random forest* untuk prediksi *win ratio* pemain *player unknown battleground*. Dalam penelitiannya mengenai strategi bermain yang efektif pada *game Player Unknown Battleground* dengan melakukan penambangan data terhadap data statistic pemain yang diambil dari *website Kaggle*. Dari data tersebut akan dibuat model untuk memprediksi *win ratio* dari setiap pemain menggunakan metode *Random Forest*. Percobaan prediksi *win ratio* dengan metode *Random Forest* menghasilkan akurasi tertinggi sebesar 88.19%. Hasil tersebut didapatkan dengan menggunakan *dataset* statistic Pemain *Player Unknown Battleground* tanpa membuang data *outlier* dan data yang tidak ternomalisasi dengan jumlah *tree* sebanyak 70. Dari model yang berhasil dibuat, atribut yang paling berpengaruh dalam melakukan klasifikasi adalah atribut "solo_KillDeathRatio". (Haristu, 2019)

Penelitian yang dilakukan oleh Syauqi Amri Yahya**,** mengenai klasifikasi ketepatan lama studi mahasiswa menggunakan metode *support vector machine* dan *random forest* (studi kasus: data lama studi alumni Universitas Islam Indonesia tahun kelulusan 2000-2017). Dalam penelitiannya mengenai setiap perguruan tinggi berusaha untuk terus memperbaiki manajemennya, supaya meningkatkan mutu pendidikan dan meningkatkan akreditasi. Salah satu elemen penilaian akreditasi perguruan tinggi adalah lulus tepat waktu. Selain itu, ketepatan lama studi mahasiswa merupakan isu yang penting karena ketepatan tersebut menjadi dasar efektifnya suatu perguruan tinggi. Universitas Islam Indonesia (UII) adalah salah satu perguruan tinggi swasta terkemuka di Indonesia. Sebagai universitas yang sudah cukup tua di Indonesia, UII sudah berakreditas-A dan sudah menghasilkan banyak alumni dari berbagai daerah dan latar belakang. Untuk terus meningkatkan universitas, UII tentunya harus mempertimbangkan juga aspek ketepatan waktu mahasiswanya untuk menempuh lama studi, karena itu merupakan salah satu aspek

penilaian akreditasi dari BAN-PT. Klasifikasi adalah metode untuk memprediksi suatu kejadian atau keputusan yang akan datang berada di suatu titik tertentu. Analisis klasifikasi bisa digunakan untuk memprediksi bahwa seorang mahasiswa dikatakan lulus tepat waktu atau tidak. Metode *Support Vector Machine* (SVM) dan *Random Forest* adalah bagian dari metode klasifikasi. Analisis klasifikasi SVM dan Random Forest dilakukan dengan menggunakan data historis dari alumni UII tahun kelulusan 2000-2017. Berdasarkan hasil penelitian ini dapat diketahui bahwa tingkat akurasi SVM kernel RBF dengan nilai optimum C=1 dan gamma = 1 adalah 77%, akurasi SVM kernel sigmoid dengan nilai optimum C=10, dan gamma = 1 adalah 68%, dan akurasi Random Forest dengan nilai optimum m = 2 dan k = 500 adalah 80%. Oleh karena itu, metode terbaik untuk menentukan ketepatan lama studi mahasiswa UII adalah *Random Forest*. (Yahya, 2018)

Penelitian yang dilakukan oleh Maulana Dhawangkhara dan Edwin Riksakomara, mengenai prediksi intensitas hujan kota Surabaya dengan matlab menggunakan teknik *random forest* dan CART (studi kasus kota Surabaya). Keakuratan prediksi potensi curah hujan di Kota Surabaya dibutuhkan untuk antisipasi bencana akibat hujan seperti banjir bandang, membantu memprediksi kondisi penerbangan dan membantu majaemen saluran sanitasi di Surabaya. Prediksi dilakukan dengan data hari sebelumnya menggunakan perbandingan teknik *Classification and Regression Trees* (CART) dan *Random Forest* (RF) pada data cuaca selama 17 tahun (2000-2016) berasal dari stasiun cuaca Juanda, Surabaya melalui website NCDC yang terdiri dari data suhu udara, titik embun, keepatan angin, tekanan udara, visibilitas dan curah hujan. Evaluasi pembuatan model dengan pengukuran akurasi, precision dan recall menunjukkan bahwa baik metode CART maupun *Random Forest* mampu mengklasifikasi dengan akurasi baik sebesar 78% untuk 4 dari 5 kelas intensitas hujan, dengan kelas terakhir belum mampu diklasifikasi oleh kedua metode. Metode Random forest memiliki nilai performa sedikit lebih baik dibandingkan dengan CART sebesar 6%. Eksperimen *tuning hyperparameter* untuk kedua metode membuktikan performa lebih baik dibandingkan parameter *default* metode dan mampu memberikan kestabilan hasil performa dari segi uji coba proporsi data training dan testing. Variabel yang berpengaruh besar dalam model CART dan random forest dengan nilai uji performa

yang baik antara lain adalah suhu udara, titik embun, suhu udara maksimum dan suhu udara minimum beserta variabel turunannya (selisih suhu udara maksimum dan minimum, selisih suhu udara dan titik embun dan kelembapan relatif). Penelitian ini menghasilkan aplikasi pengklasifikasi intensitas hujan yang memiliki akurasi baik atas kelas intesitas hujan (tidak hujan, ringan, sedang, deras, sangat deras). (Dhawangkhara & Riksakomara , 2017)

Penelitian yang dilakukan oleh Nariswari Karina Dewi dan Utami Dyah Syafitri, mengenai penerapan metode *random forest* dalam *driver analysis*. Penelitian ini menunjukkan bahwa *Random Forest* dapat diaplikasikan dalam bidang biostatistika. Dari hasil penelitian ini menunjukkan penyusunan *driver analysis* berdasarkan MDG menghasilkan *driver analysis* yang stabil jika ukuran *Random Forest* lebih dari 500. Untuk penyusunan *driver analysis* berdasarkan rataan MDG dari 1000 *driver analysis* cukup stabil meskipun menggunakan *Random Forest* dengan ukuran yang kecil. Hasil analisis juga stabil pada ukuran pada contoh peubah penjelas. (Dewi & Syafitri , 2011).

Penelitian yang dilakukan oleh Alfina Nur Firmani, mengenai penyelesaian regresi semiparametrik menggunakan regresi *random forest*. Studi kasus yang digunakan pada penelitian adalah mengestimasi persentase kemiskinan berdasarkan faktor-faktor yang mempengaruhinya. Analisis regresi merupakan analisis yang sering digunakan untuk mengetahui hubungan yang terjadi antara variabel dependen dan independennya. Namun metode ini masih cenderung terbatas pada asumsi dan keadaan data tertentu, terutama dalam penyelesaian regresi semiparametrik. Oleh karena itu muncul metode regresi yang sangat fleksibel dan mudah, yang merupakan pengembangan dari pohon keputusan. Metode ini dinamakan regresi *Random Forest*. Regresi *Random Forest* merupakan gabungan dari banyak CART yang ditumbuhkan sehingga akurasi yang dihasilkan akan lebih akurat dari pohon tunggal. Dari hasil regresi ini didapatkan hasil bahwa faktor angka melek huruf merupakan faktor utama penyebab kemiskinan. Dari hasil tersebut diusulkan saran penanggulangan kemiskinan dengan memfokuskan pada faktor utama, sehingga penekanan jumlah penduduk miskin menjadi lebih efisien. Penelitian ini menunjukkan bahwa Random Forest dapat digunakan untuk menyelesaikan regresi. Hasil penelitian menunjukkan bahwa estimasi presentasse

kemiskinan dan variabel yang paling berpengaruh terhadap presentase kemiskinan adalah angka melek huruf, MSE yang dihasilkan adalah 0.000308. (Firmani, 2016)

Penelitian yang dilakukan oleh Muhammad Irhamna Putra, mengenai sistem rekomendasi kelayakan kredit menggunakan metode *random forest* pada BRI kantor cabang Pelaihari. Penelitian ini melakukan pembahasan tentang penggunaan salah satu algoritma *machine learning* untuk klasifikasi yang bernama *random forest* untuk melakukan prediksi terhadap potensi kelayakan kredit seorang calon nasabah untuk dijadikan acuan dasar oleh bank sebelum diproses lebih lanjut. Permasalahan yang ada pada bank adalah tidak adanya sistem rekomendasi untuk sistem pendukung keputusan yang dapat melakukan perhitungan secara otomatis untuk mengurangi resiko kredit dalam bank, dalam hal ini mencegah adanya potensi kredit macet. untuk itu dibangunlah sebuah sistem rekomendasi atau sistem pendukung keputusan untuk melakukan prediksi potensi kelayakan suatu nasabah sebagai acuan dasar oleh pihak bank agar nantinya dilakukan proses analisa kredit tindak lanjut berdasarkan analisis 5C tentang perbankan yang berbasis *webservice*. Berdasarkan hasil penelitian ini, model perhitungan prediksi yang telah dilakukan *testing* dengan beberapa skenario data *training* menggunakan *record* data tagihan kredit nasabah, memiliki rata – rata ditemui rata – rata hasil *accuracy* sebesar 96,57%%, *precision* sebesar 96,46% dan *recall* sebesar 100%. (Putra M. , 2019)

Penelitian yang dilakukan oleh Anugerah Ganda Putra, mengenai klasifikasi tulisan tangan berupa angka menggunakan *random forest* dan *histogram of oriented gradient*. Penelitian ini melakukan pembahasan tentang membuat mesin atau komputer mngenali tulisan tangan masih merupakan masalah yang terus diteliti di bidang Computer Vision. Angka sendiri tidak bisa dipisahkan dari kehidupan manusia sehingga kemampuan mengenali tulisan tangan berupa angka akan sangat membantu pekerjaan manusia. Berbagai jenis algoritma klasifikasi yang tersedia seperti ANN dan SVM mampu mengenali tulisan tangan, namun kelemahan keduanya terletak pada komplesitas waktu training yang cukup lama Random Forest menjadi salah satu algoritma alternatif dan diuji dalam kasus klasifikasi tulisan tangan berupa angka pada Tugas Akhir ini. Random Forest tidak membutuhkan preprocessing khusus dalam pengimpelementasiannya untuk mencapai akurasi yang bagus. Diuji juga performansi Random Forest menggunakan

fitur ciri Histogram of Gradient yang sering dipakai untuk mengenali objek dalam citra. Hasil pengujian menunjukkan Random Forest dengan jumlah 10 tree dan vektor ciri HOG yang berukuran mampu mengklasifikasi tulisan tangan berupa angka dalam dataset MNIST dengan akurasi bagus yang mencapai 97% dan waktu training sekitar 4 menit. (Putra A. , 2014)

Penelitian yang dilakukan oleh I Made Budi Adnyana, mengenai prediksi lama studi mahasiswa dengan metode *random forest* (studi kasus: STIKOM Bali). Kelulusan tepat waktu merupakan salah satu elemen penilaian akreditasi dari perguruan tinggi. Selain itu wisuda tepat waktu merupakan isu yang penting karena tingkat kelulusan sebagai dasar efektifnya suatu perguruan tinggi. Kurangnya informasi dan analisa yang diperoleh Bidang Akademik STIKOM Bali mengakibatkan sulitnya melakukan prediksi lama studi mahasiswa. Prediksi lama studi mahasiswa dapat membantu Bidang Akademik dalam menyusun strategi yang tepat untuk menekan atau memperpendek lama studi mahasiswa. Pada permasalahan ini dapat diterapkan teknik data mining untuk melakukan prediksi yaitu dengan menggunakan metode klasifikasi Random Forest. Random Forest merupakan suatu kumpulan dari beberapa tree, dimana masing-masing tree bergantung pada nilai piksel pada tiap vektor yang daimbil secara acak dan independen. Data sampel diperoleh langsung dari bagian Akademik STIKOM Bali. Data yang digunakan adalah data lulusan 2 tahun terakhir, meliputi IPK, SKS, jumlah cuti dan non-aktif, nilai mahasiswa, dan lama studi mahasiswa. Output dari sistem ini berupa klasifikasi yang terdiri dari 2 kelas, yaitu lulus tepat waktu dan lulus lewat batas waktu. Dari hasil eksperimen diperoleh nilai akurasi adalah 83.54%. (Adnyana, 2016)

**Tabel 2.1** Hasil Penelitian Sebelumnya

| No | Nama Peneliti | Judul Penelitian | Tahun | Hasil Penelitian | Perbandingan | |
|---|---|---|---|---|---|---|
| | | | | | Persamaan | Perbedaan |
| 1 | Agenda Yudha Samudra | Pendekatan *Random Forest* Untuk Model Peramalan Harga Tembakau Rajangan Di Kabupaten Temaggung | 2019 | - Dalam penelitiannya dibangun perangkat lunak untuk memprediksi harga tembakau rajangan di kabupaten Temanggung dengan menggunakan metode *Random Forest* dan diimplementasikan menggunakan *Python*.<br>- Berdasarkan hasil penelitiannya dengan menggunakan data iklim dan penjualan tembakau menghasil prediksi akurasi sebesar 82.39%. | Menggunakan metode analisis yang sama yaitu *Random Forest.* | - Objek penelitiannya berupa peramalan harga tembakau rajangan di Kabupaten Temaggung.<br>- Perbandingan pembagian data *training* dan data *testing* sebesar 70%:30%.<br>- Pembangunan *random forest* menggunakan parameter *bootstrap = true* dan *n_estimators*.<br>- Jumlah pohon optimal adalah 64 pohon.<br>- Hasil akurasi prediksi sebesar 82,39%. |

| No | Nama Peneliti | Judul Penelitian | Tahun | Hasil Penelitian | Perbandingan | |
|---|---|---|---|---|---|---|
| | | | | | Persamaan | Perbedaan |
| 2 | Vanissa Wanika Siburian dan Ika Elvina Mulyana | Prediksi Harga Ponsel Menggunakan Metode *Random Forest* | 2018 | - Dalam penelitiannya prediksi harga ponsel yang dipengaruhi oleh spesifikasi yang dimiliki dan klasifikasi untuk memprediksi harga ponsel dengan metode *Random Forest*.<br>- Hasil klasifikasi mengunakan 7 variabel prediksi dan 1 variabel respon dan hasil akurasi sebesar 81%. | Menggunakan metode analisis yang sama yaitu *Random Forest.* | - Objek penelitiannya: prediksi harga ponsel.<br>- Perbandingan pembagian data *training* dan data *testing* sebesar 70%:30%.<br>- Variabel respon dengan *range price* murah, standar, mahal dan sangat mahal.<br>- Hasil akurasi prediksi sebesar 81%. |
| 3 | Reinardus Aji Haristu | Penerapan Metode *Random Forest* Untuk Prediksi *Win Ratio* Pemain *Player Unknown Batleground* | 2019 | - Penelitian model prediksi *win ratio* dari setiap pemain dengan *Random Forest.*<br>- Akurasi tertinggi dari hasil prediksi *Random Forest* adalah 88,19%. | Menggunakan metode analisis yang sama yaitu *Random Forest.* | - Objek penelitian : prediksi *win ratio* pemain PUBG.<br>- Jumlah pohon optimal adalah 70 pohon.<br>- Hasil akurasi sebesar 88,19%. |

| No | Nama Peneliti | Judul Penelitian | Tahun | Hasil Penelitian | Perbandingan | |
|---|---|---|---|---|---|---|
| | | | | | Persamaan | Perbedaan |
| 4 | Syauqi Amri Yahya | Klasifikasi Ketepatan Lama Studi Mahasiswa Menggunakan Metode *Support Vector Machine* dan *Random Forest* (Studi Kasus: Data Lama Studi Alumni Universitas Islam Indonesia Tahun Kelulusan 2000-2017) | 2018 | - Dalam penelitiannya memprediksi mahasiswa UII lulus tepat waktu atau tidak dengan SVM dan *Random Forest*.<br>- Hasil akurasi SVM kernel RBF dengan C=1 dan gamma = 1 adalah 77%, akurasi SVM kernel sigmoid dengan C=10, dan gamma = 1 adalah 68%, dan akurasi *Random Forest* dengan m = 2 dan k = 500 adalah 80%. | Menggunakan metode analisis yang sama yaitu *Random Forest.* | - Objek penelitian : prediksi ketepatan lama studi mahasiswa UII.<br>- Perbandingan SVM dengan parameter C dan gamma, *random forest* dengan *ntree* (k) dan jumlah variabel (m).<br>- Hasil akurasi optimal SVM kernel RBF dengan C=1 dan gamma=1 adalah 77%, akurasi SVM kernel sigmoid dengan C=10, dan gamma = 1 adalah 68%, dan akurasi *Random Forest* dengan m = 2 dan k = 500 adalah 80%. |

| No | Nama Peneliti | Judul Penelitian | Tahun | Hasil Penelitian | Perbandingan | |
|---|---|---|---|---|---|---|
| | | | | | Persamaan | Perbedaan |
| 5 | Maulana Dhawangkhara dan Edwin Riksakomara | Prediksi Intensitas Hujan Kota Surabaya dengan Matlab Mengunakan Teknik Random Forest dan CART (Studi kasus Kota Surabaya) | 2017 | - Dalam penelitiannya bertujuan untuk memprediksi kondisi penerbangan dan membantu manajemen saluran sanitasi di Surabaya. Prediksi dengan data cuaca selama 17 tahun (2000-2016) dari stasiun cuaca Juanda dengan menggunakan CART dan *Random Forest*.<br>- Hasil akurasi tertinggi dengan *random forest* sebesar 78%. | Menggunakan metode analisis yang sama yaitu *Random Forest.* | - Objek penelitian : prediksi kelas intesitas hujan (tidak, ringan, sedang, lebat, sangat lebat)<br>- Kombinasi parameter model terbaik dengan akurasi tertinggi klasifikasi kelas 0,1,2,3, untuk CART dengan *leaf* (1-5) dan *parent* (30). Untuk *random forest* dengan jumlah variabel acak (6) dan jumlah pohon (10).<br>- Hasil akurasi prediksi dengan CART sebesar 0,72% dan *Random Forest* sebesar 0,78 %. |

| No | Nama Peneliti | Judul Penelitian | Tahun | Hasil Penelitian | Perbandingan | |
|---|---|---|---|---|---|---|
| | | | | | Persamaan | Perbedaan |
| 6 | Nariswari Karina Dewi dan Utami Dyah Syafitri | Penerapan Metode *Random Forest* Dalam *Driver Analysis* | 2011 | - *Driver analysis* dilakukan untuk memahami pengaruh peubah penjelas terhadap respons sehingga diketahui prioritas setiap peubah penjelas dalam menggerakkan peubah respons.<br>- Penyusunan *driver analysis* dengan MDG menghasilkan *driver analysis* yang stabil jika *random forest* lebih dari 500. *Random forest* dengan peubah penjelas sebesar 4 dan memberikan akurasi tingkat misklasifikasi berkisar antara 33% dan 35.5% dengan nilai rataannya sebesar 34.5%. | Menggunakan metode analisis yang sama yaitu *Random Forest.* | - Objek penelitian tentang *driver analysis* dengan peubah respon status kesediaan untuk membeli produk dan peubah penjelasnya status setuju tehadap produk<br>- *Driver analysis* dengan stabil MDG terbaik saat k lebih dari 500 pohon dan jumlah peubah penjelas = 4.<br>- Tingkat misklsifikasi terendah akurasi *random forest* antara 33% - 35.5%, dan pemusatan rataan 34.5%. |

| No | Nama Peneliti | Judul Penelitian | Tahun | Hasil Penelitian | Perbandingan | |
|---|---|---|---|---|---|---|
| | | | | | Persamaan | Perbedaan |
| 7 | Alfina Nur Firmani | Penyelesaian Regresi Semi Parametrik dengan Mengunakan Regresi *Random Forest* | 2019 | - Dalam penelitiannya mengetahui persentase kemiskinan dan faktor-faktor yang mempengarhinya. Nilai prediksi terbaik dengan *m* = 2 dan jumlah pohon 200 dan MSE sebesar 0,000308 | Menggunakan metode analisis yang sama yaitu *Random Forest* | - Objek penelitian : prediksi faktor-faktor yang mempengaruhi kemiskinan.<br>- Model *random forest* optimal dengan jumlah peubah penjelas = 2 dan jumlah pohon = 200. |
| 8 | Muhammad Irhamna Putra | sistem rekomendasi kelayakan kredit mengunakan metode *random forest* pada BRI kantor cabang Pelaihari | 2019 | - Penelitian ini penggunaan *random forest* melaukan prediksi terhadap potensi kelayakan kredit calon nasabah berdasarkan usia, alamat, pekerjaan, jumlah pinjaman, dan jangka waktu.<br>- Hasil rata-rata *accuracy* sebesar 96,57%, *precision* sebesar 96,46% dan *recall* sebesar 100%. | Menggunakan metode analisis yang sama yaitu *Random Forest* | - Objek penelitian : prediksi potensi kelayakan kredit calon nasabah lancar atau kurang.<br>- *Random forest* dengan jumlah pohon = 10.<br>- Hasil *acuracy*, *precision*, dan *recall* = 96,57%, 96,46% dan 100%. |

| No | Nama Peneliti | Judul Penelitian | Tahun | Hasil Penelitian | Perbandingan | |
|---|---|---|---|---|---|---|
| | | | | | Persamaan | Perbedaan |
| 9 | Anugerah Ganda Putra | Klasifikasi Tulisan Tangan Berupa Angka Mengunakan *Random Forest* dan *Histogram Of Oriented Gradient* | 2014 | - Random Forest diuji kasus klasifikasi dan *Histogram of Gradient* untuk mengenali objek dalam citra tulisan tangan berupa angka dari dataset MNIST.<br>- Pengujian jenis vector ciri menggunakan HOG menghasilkan akurasi lebih baik.<br>- Jumlah tree yang paling akurat =15 dengan akurasi 97,29%.<br>- Parameter jumlah atribut yang diseleksi secara acak dan jumlah data minimum splitting, tidak terlihat hubungan dengan akurasi klasifikasi. | Menggunakan metode analisis yang sama yaitu *Random Forest* | - Objek penelitian : klasifikasi tulisan tangan angka MNIST.<br>- Tahap ektraksi citra angka dengan HOG menghasilkan akurasi lebih baik.<br>- Jumlah tree paling akurat dan efesien waktu = 15 dengan akurasi97,29%<br>- Parameter jumlah atribut secara acak dan jumlah data minimum *splitting* tidak ada hubungan dengan akurasi karena tidak mempengaruhi secara signifikan. |

| No | Nama Peneliti | Judul Penelitian | Tahun | Hasil Penelitian | Perbandingan | |
|---|---|---|---|---|---|---|
| | | | | | Persamaan | Perbedaan |
| 10 | I Made Budi Adnyana | Prediksi Lama Studi Mahasiswa Dengan Metode *Random Forest* (Studi Kasus: STIKOM Bali) | 2016 | - Prediksi lama studi mahasiswa STIKOM Bali dengan metode *Random Forest*.<br>- Data diperoleh dari Akademik STIKOM Bali. Data yang digunakan adalah data lulusan 2 tahun terakhir, meliputi IPK, SKS, jumlah cuti dan non-aktif, nilai mahasiswa, dan lama studi mahasiswa.<br>- *Output* klasifikasi terdiri dari 2 kelas, yaitu lulus tepat waktu dan lulus lewat batas waktu.<br>- Hasil eksperimen diperoleh nilai akurasi adalah 83.54%. | Menggunakan metode analisis yang sama yaitu *Random Forest* | - Objek penelitian : prediksi lama studi mahasiswa kelulusan STIKOM Bali.<br>- Perbandingan data *training* dan data *testing* sebesar 60%:40%.<br>- Pengujian random forest dengan 10 *tree* dan jumlah peubah 3 fitur.<br>- Hasil akurasi kelas Lulus Tepat Waktu sebesar81,81% dan akurasi kelas Lulus Lewat Waktu Studi sebesar 84,98%. Hasil akurasi keseluruhan sebesar 83.54%. |

# BAB III
# LANDASAN TEORI

## 3.1 Televisi

Definisi televisi secara etimologi, televisi terbentuk dari kata *tele* yang berarti "jarak" dalam bahasa Yunani dan kata *visi* yang berarti "citra atau gambar" dalam bahasa Latin. Jika digabungkan, maka televisi memiliki makna yaitu suatu sistem penyajian gambar dan suara dari suatu tempat yang berjarak jauh. Sedangkan definisi televisi secara terminologi yaitu sistem yang menggunakan peralatan yang mengubah cahaya dan suara ke dalam gelombang elektrik dan mengubahnya kembali menjadi berkas cahaya yang dapat dilihat dan suara yang dapat didengar. (Sutisno, 1993)

## 3.2 Sejarah Televisi

Cikal bakal penemuan televisi terlahir adanya perkembangan teknologi informasi dan komunikasi. Dengan seiringnya kemajuan teknologi digital membuat televisi yang mengalami perkembangan. Awal mula terlahirnya televisi tidak dapat dipisahkan dari penemuan hukum gelombang elektromagnetik yang ditemukan oleh Joseph Henry dan Michael Faraday di tahun 1831. Sejak ditemukannya hukum gelombang elektromagnetik, penemuan dasar televisi berkembang pesat yang dikembangkan oleh banyak penemu di dunia.

Pada tahun 1884, Ilmuan Paul Nipkov dalam gagasannya berhasil menciptakan teleskop elektrik yang menyalurkan sinyal gambar untuk mengirim gambar elektronik melalui udara dengan menggunakan kepingan logam. Pada tahun 1897, terlahirlah televisi tabung *Cathoda Ray Tube* (CRT) hitam putih pertama yang diciptakan oleh ilmuwan Karl Ferdinand Braun yang dibuat dengan layar berpendar apabila terkena sinar. Televisi tabung dilakukan penyempurnaan hingga pada tahun 1940 terciptalah televisi tabung berwarna oleh Peter Goldmark. Setelah peristiwa Perang Dunia II berkahir, kemajuan teknologi yang berkembang pesat membuat permintaan televisi meningkat. Memasuki era digital ditandai dengan munculnya televisi digital (berbasis komputer). Pada tahun 1964,

*prototype* sel tunggal *display* televisi plasma pertama kali diciptakan oleh Donald Bitzer dan Gene Slottow. Pada tahun 1975, Larry Weber berhasil menciptakan televisi layar plasma yang lebih stabil dan berwarna. Selanjutnya ilmuan Walter Spear dan Peter Le Comber membuat display warna *Liquid Crystal Display* (LCD) dari bahan *thin film transfer* yang ringan. Para ilmuwan dari perusahaan Kodak di tahun 1979, berhasil menciptakan tampilan televisi dengan teknologi *Organic Light Emitting Diode* (OLED) dan mematenkan OLED sebagai peralatan *display* pertama kali pada tahun 1987. Memasuki abad 20, penemuan dan inovasi televisi layar semakin disempurnakan, baik layar dengan teknologi LCD, Plasma maupun LED. (J.B, Wahyudi, 1983)

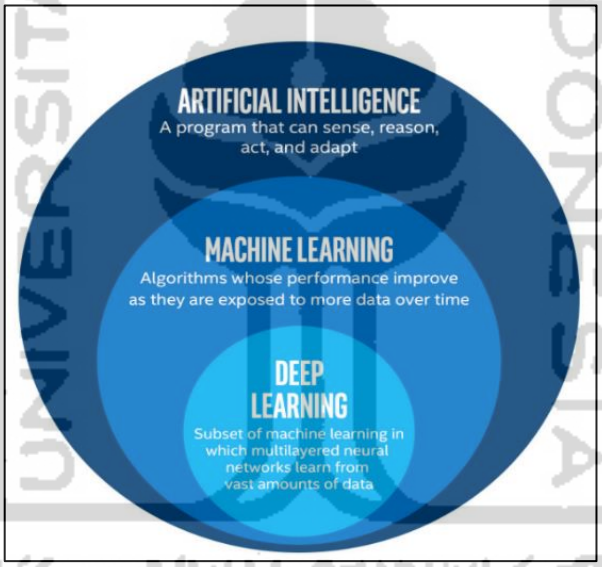
Televisi merupakan gabungan dari suara dan gambar yang bersifat informatif, hiburan, dan pendidikan, atau bahkan gabungan dari ketiga unsur tersebut. Informasi yang disampaikan oleh televisi, akan lebih mudah dimengerti karena informasi diterima dengan terdengar secara audio dan terlihat secara visual. Televisi pada umumnya menyiarkan informasinya secara massa. (Ardianto, 2007)

### 3.3 *Artificial Intelligence*

Kecerdasan Buatan (*Artificial Intelligence*) adalah bidang ilmu komputer yang mempelajari bagaimana menjadikan komputer dapat berfikir dan bekerja seperti dan sebaik yang dilakukan oleh manusia. *Artificial Intelligence* merupakan teknologi yang meliputi pemrograman komputer dengan mesimulasikan kecerdasan manusia untuk penyelesaian suatu masalah dengan meniru bagaimana cara pemikiran manusia.

Menurut McCarthy (1956), *artificial intelligence* adalah suatu sistem komputer yang bertujuan untuk mengetahui atau memodelkan proses berpikir manusia dan mendesain mesin sehingga bisa menirukan perilaku manusia. *Artificial Intelligence* (AI) merupakan salah satu disiplin ilmu yang luas yang terbagi menjadi beberapa pengaplikasian antara lain: pengolahan bahasa alami (*Natural Language Processing*), sistem pakar (*Expert System)*, pengenalan ucapan (*Speech Recognition*), robotika dan sistem sensor, dan *computer vision.*

*Artificial Intelligence* adalah suatu cara untuk menjadikan komputer berpikir secerdas manusia dengan tujuan untuk komputer dapat memiliki kemampuan berperilaku, berpikir, dan mengambil keputusan layaknya manusia. Dengan kemajuan perkembangan *Artificial Intelligence* secara signifikan melahirkan dua konsep kecerdasan buatan yaitu *machine learning* dan *deep learning*. *Artificial Intelligence* diibaratkan payung yang lebih luas, *Machine Learning* merupakan cabang ilmu dari *Artificial Intelligence* dan *Deep Learning* berada di dalam lingkup *Machine Learning*. Secara visualisasi digambarkan pada Gambar 3.1 yang menunjukkan *Artificial Intelligence* merupakan bidang ilmu terbesar, kemudian *Machine Learning* adalah bagian dari *Artificial Intelligence*, dan *Deep Learning* adalah bagian dari *Machine Learning*.




(Sumber: Towards Data Science)

**Gambar 3.2** Perbedaan *Artificial Intelligence, Machine Learning,* dan *Deep Learning*

### 3.4 *Machine Learning*

*Machine Learning* merupakan salah satu cabang dari ilmu Kecerdasan Buatan (*Artificial Intelegence*) yang kemampuan komputer untuk melakukan pembelajaran tanpa harus menjelaskan secara eksplisit kepada komputer. *Machine learning* merujuk pada sebuah metode yang membuat komputer memiliki kemampuan dalam mempelajari dan melakukan sebuah pekerjaan secara otomatis. Proses *machine learning* dilakukan melalui algoritma tertentu, sehingga pekerjaan yang diperintahkan kepada komputer dapat

dilakukan secara otomatis. Algoritma semacam ini bekerja dengan cara membangun sebuah model dari *input* untuk dapat menghasilkan suatu prediksi atau pengambilan keputusan berdasarkan data yang ada.

*Machine Learning* merupakan sebuah pemrograman komputer yang digunakan untuk mengoptimalkan performa sebuah pekerjaan menggunakan beberapa contoh data atau *experience* masa lalu. Peneliti memiliki sebuah model yang didefinisikan dengan satu atau beberapa buah parameter, sedangkan proses pembelajaran atau *learning* dilakukan dengan pemrograman komputer untuk mengoptimalkan parameter dari model menggunakan data *training* atau *experience* masa lalu. Model memungkinkan untuk digunakan dalam memprediksi permasalahan kedepan atau analisis deskriptif untuk mendapatkan pengetahuan dari data. (Alpaydin, 2010).

*Machine learning* menggunakan teori statistika untuk membangun model matematis. Hal ini didasarkan pada tujuan yang ingin dicapai yaitu mengambil kesimpulan dari sampel data. Peran *computer science* terbagi menjadi dua bagian yaitu *Training* yang peneliti membutuhkan algoritma yang efisien untuk menyelesaikan permasalahan dengan optimal. Dan *Testing* setelah model dapat dilatih, representasi dan solusi algoritmiknya untuk inferensi juga harus efisien, sehingga perlu dilakukan pengujian akurasi prediksi. Dalam pembelajaran *machine learning*, terdapat beberapa tipe algoritma *machine learning* antara lain: (Russel & Norvig, 1995).

1. *Supervised Learning*

   Pada *supervised learning*, data yang dimiliki dilengkapi dengan label yang menunjukkan kelompok data tersebut. Algoritma yang belajar berdasarkan sekumpulan contoh pasangan masukan dan keluaran yang diinginkan. Model yang dihasilkan adalah model prediksi dari data yang telah diberi label dengan memetakan masukan yang baru menjadi keluaran yang tepat.

2. *Unsupervised Learning*

   Pada *Unsupervised Learning*, data pembelajaran tidak memiliki label sehingga harus mencari struktur atau pola dari data yang ada, kemudian melakukan pengelompokan berdasarkan informasi yang dimiliki.

Setelah itu mencoba untuk mengelompokan data berdasarkan karakteristik yang ditemui. Salah satu algoritma *unsupervised learning* yang paling umum digunakan adalah *clustering.*

3. *Reinforcement Learning*

   Pada skenario *reinforcement learning* fase pembelajaran dan tes saling dicampur. Untuk mengumpulkan informasi pembelajar secara aktif dengan berinteraksi ke lingkungan sehingga untuk mendapatkan balasan untuk setiap aksi dari pembelajar.

### 3.5 Klasifikasi

Klasifikasi adalah proses untuk menemukan model atau fungsi yang dapat menjelaskan atau membedakan konsep atau kelas data. Tujuan klasifikasi adalah untuk memperkirakan kelas dari suatu objek yang labelnya tidak diketahui. Proses klasifikasi biasanya dibagi menjadi dua fase yaitu *fase learning* dan *fase test*. Pada *fase learning*, sebagian data yang telah diketahui kelas datanya diumpankan untuk membentuk model perkiraan. Kemudian pada *fase test* model yang sudah terbentuk diuji dengan sebagian data lainnya untuk mengetahui akurasi dari model. Teknik ini dapat memberikan klasifikasi pada data baru dengan memanipulasi data yang ada yang telah diklasifikasi. (Ulwan, 2016).

### 3.6 Analisis Regresi

Analisis regresi adalah persamaan matematik yang dapat meramalkan nilai suatu variabel tak bebas dari nilai variabel bebas (Walpole R. , 1992). Analisis regresi adalah salah satu metode statistik yang digunakan secara luas pada ilmu terapan untuk menyelesaikan masalah sebab-akibat yang digunakan untuk mempelajari hubungan antara satu variabel dengan variabel lainnya. Analisis regresi bertujuan untuk menentukan hubungan antara dua variabel atau lebih, sehingga dapat mengetahui pola hubungannya. Analisis regresi dapat digunakan untuk mengetahui bagaimana variabel bebas mempengaruhi variabel tak bebas, sehingga dapat memprediksi perubahan variabel tak bebas apabila diberikan nilai variabel bebas. Pada dasarnya analisis regresi sebagai kajian terhadap hubungan satu variabel yang diterangkan dengan satu atau lebih variabel yang menerangkan.

Analisis regresi terdapat tiga pendekatan antara lain: regresi parametrik, regresi non-parametrik dan regresi semiparametrik. Regresi parametrik

mengasumsikan hubungan antara variabel prediktor dan variabel dependen diketahui seperti linear, kubik, polynomial, eksponensial dan lain-lain. Regresi non-parametrik digunakan sebagai alternatif untuk mengatasi masalah pola hubungan antar variabel yang tidak diketahui hubungannya serta tidak memperhatikan asumsi. Regresi semiparametrik merupakan kombinasi dari regresi parametrik dan non-paramterik. Namun pendekatan semiparametrik belum dapat memberikan solusi secara menyeluruh karena tergolong tidak sederhana dan memiliki asumsi yang harus dipenuhi.

Perkembangan kemajuan teknologi komputasi, menemukan solusi algoritma yang dapat mengatasi permasalahan regresi dengan pola hubungan yang tidak diketahui dan masih cenderung terbatas pada asumsi dan keadaan data tertentu, terutama dalam penyelesaian regresi semiparametrik. Oleh karena itu muncul metode regresi yang sangat fleksibel dan mudah, yang merupakan pengembangan dari pohon keputusan yaitu *Random Forest*. *Random forest* adalah algoritma gabungan klasifikasi dan regresi yang menjadi bagian dari kelompok *ensemble learning* yang dapat memecahkan masalah regresi dengan fleksibel dan menghasilkan akurasi prediksi yang lebih baik.

### 3.7 *Decision Tree*

Pohon keputusan (*Decision tree*) merupakan metode klasifikasi yang menggunakan ilustrasi struktur pohon. Metode *Decision tree* merupakan metode klasifikasi yang paling popular digunakan karena hasil model yang dibangun lebih mudah untuk diinterpretasi oleh manusia. Hal ini dikarenakan pemetaan yang terbentuk mirip dengan bentuk pohon dalam alternatif-alternatif pemecahan masalah. Dengan visualisasi percabangan pohon dapat memperlihatkan faktor-faktor probabilitas yang mempengaruhi keputusan dan hasil akhir dalam mengambil alternatif keputusan tersebut.

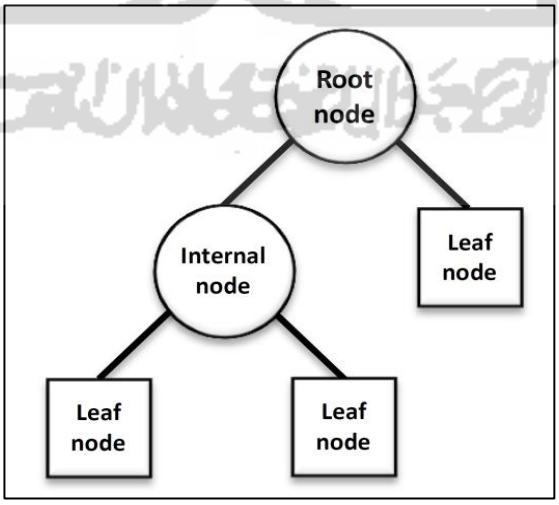
*Decision Tree* merupakan suatu metode klasifikasi yang menggunakan struktur pohon dan setiap *node*nya merepresentasikan atribut dan cabangnya merepresentasikan nilai dari atribut, sedangkan daunnya digunakan untuk merepresentasikan kelas. Metode ini merupakan metode yang mudah untuk

dipahami, dinamakan pohon keputusan karena aturan yang terbentuk mirip dengan bentuk pohon. (Bernard, Heutte, & Adam, 2009).

Pohon *(tree)* adalah sebuah struktur data yang terdiri dari simpul *(node)* dan rusuk *(edge). Decision tree* adalah sebuah diagram alir yang berbentuk struktur pohon, setiap *internal node* menunjukan pengujian terhadap atribut, setiap cabang menunjukan hasil pegujian dan *leaf node* menunjukan kelas. *Node* yang teratas adalah *root node* yang tidak memiliki *edge* masuk tapi memiliki beberapa *edge* keluar. Pada *internal node* akan memiliki satu *edge* masuk dan beberapa *edge* keluar, sedangkan *leaf node* hanya akan memiliki satu *edge* masuk tanpa memiliki *edge* keluar. (Hermawati, 2013).

Konsep dari *decision tree* adalah mengklasifikasikan data yang belum diketahui kelasnya lalu dimasukan ke dalam prediksi kelas dengan pemetaan proses pengambilan keputusan, sehingga pengambil keputusan akan lebih menginterpretasikan solusi dari permasalahan. Pada *decision tree* memiliki 3 jenis *node* antara lain:

1. *Root Node* merupakan node teratas dengan tidak memiliki *input* dan bisa memiliki *output* lebih dari satu.
2. *Internal Node* merupakan *node* percabangan, pada *node* ini hanya memiliki satu *input* dan memiliki *output* minimal dua.
3. *Leaf node* merupakan *node* akhir yang hanya memiliki satu *input* dan tidak mempunyai *output*, dan pada *leaf node* terdapat keputusan akhir.




(Sumber: researchgate.net)

**Gambar 3.3** Konsep Dasar *Decision Tree*

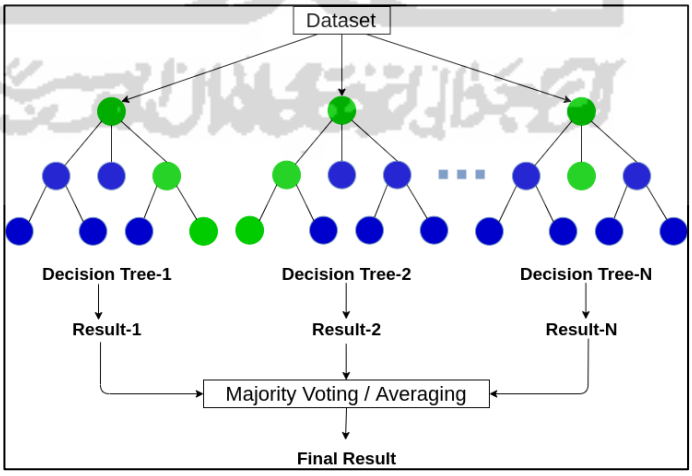
### 3.8 *Random Forest*

*Random Forest* pertama kali tahun 2001 diperkenalkan oleh Breiman. *Random Forest* adalah algoritma yang digunakan untuk menyelesaikan klasifikasi dan regresi berdasarkan agregasi sejumlah pohon keputusan. *Random Forest* adalah pengembangan lebih lanjut dari *Classification and Regression Tree* (CART) yang menerapkan *Bootstrap Aggregating* (*Bagging)* dan *Random Feature Selection*. *Random Forest* menjadi metode klasifikasi populer untuk diimplementasikan di berbagai bidang. Model *random forest* sering dipakai untuk masalah klasifikasi seperti prediksi karena *random forest* bekerja lebih efisien khususnya menggali informasi pada data yang berukuran besar, model relatif cukup efektif untuk data diskrit dan data kontinu, algoritma klasifikasi yang dikenal mampu mencapai akurasi yang bagus tanpa melakukan pencarian yang banyak pada parameter, efektif ketika berhadapan dengan data yang memiliki *missing value*, mampu menghasilkan keputusan kompleks menjadi lebih sederhana dan membangun lebih dari satu pohon keputusan sehingga memungkinkan untuk meningkatkan hasil prediksi. (Breiman, 2001).

Teknik dasar *Random Forest* adalah *decision tree*. Pada *decision tree*, data dimasukan pada bagian *root node* kemudian dilanjutkan ke *leaf node* untuk menentukan suatu kelas prediksi. *Random forest* adalah salah satu metode pengklasifikasian yang terdiri dari kumpulan pohon terstruktur dan setiap pohon memberikan suara (*vote*) untuk menentukan kelas dengan jumlah *vote* terbanyak. Dengan kata lain, *Random Forest* terdiri dari kumpulan *decision tree* yang tidak saling berkorelasi dan digunakan untuk mengklasifikasi data ke dalam suatu kelas. Konsep *random forest* dengan terdapatnya banyak pohon *(tree)* yang dihasilkan yang akhirnya membentuk sebuah hutan (*forest*). Algoritma *random forest* merupakan algoritma yang tidak memiliki pemangkasan variabel seperti pada algoritma *decision tree*. Metode ini menghubungkan banyak pohon untuk membuat klasifikasi dan kelas prediksi. Pada *random forest* pembuatan pohon dilakukan dengan cara melakukan pelatihan sampel data. Variabel digunakan untuk menentukan pemisahan (*split*) terbaik ditentukan secara acak dengan cara *voting* dari masing-masing pohon (*tree*), sehingga kelas dengan jumlah *vote* terbanyak

akan menjadi kelas pemenang keputusan. *Random Forest* merupakan sebuah metode *ensemble* yang merupakan cara untuk meningkatkan akurasi metode klasifikasi dengan cara mengkombinasikan metode klasifikasi. Klasifikasi berbasis *ensemble* akan mempunyai performa yang maksimal jika antar *basic learner* mempunyai korelasi yang rendah. Sebuah *ensemble* harus membangun *basic learner* yang lemah, karena *learner* yang kuat kemungkinan besar akan mempunyai korelasi yang tinggi dan biasanya juga menyebabkan *overfitting*, sedangkan *random forest* meminimalkan korelasi serta mempertahankan kekuatan klasifikasi dengan cara melakukan pengacakan pada proses *training*, yaitu dengan memilih sejumlah fitur secara acak dari semua fitur yang ada pada setiap melakukan *training tree*, kemudian menggunakannya menggunakan fitur-fitur yang terpilih untuk mendapatkan percabangan *tree* yang optimal. (Han, 2001).

*Bagging* atau *Bootstrap Aggregating* merupakan metode yang dapat memperbaiki hasil dari *algorithm* klasifikasi. *Bagging* merupakan salah satu metode yang berdasarkan pada *ensemble method*, yaitu metode yang menggunakan kombinasi dari beberapa model. *Bagging* prediktor adalah metode yang digunakan untuk membangkitkan *multiple version* dari prediktor dan menggunakannya untuk mendapatkan kumpulan prediktor, *multiple version* dibentuk dari replikasi *(replacement) bootstrap* dari sebuah data percobaan atau data *train*. (Breiman, 1996).




(Sumber: medium)

**Gambar 3.4** Ilustrasi Konstruksi Regresi *Random Forest*

### 3.8.1 *Tuning Hyperparameter*

Dilakukannya *tuning* (penyetelan) parameter pada pemodelan dengan *Random Forest* bertujuan untuk meningkatkan kemampuan prediksi atau membuat lebih mudah untuk melakukan pelatihan pada model. Regresi *Random Forest* terdapat dua parameter yang yang dapat disesuaikan dan dilakukan *tuning* yaitu jumlah pohon ($n_{tree}$) dan jumlah fitur acak untuk setiap *split* ($m_{try}$) pada hutan yang akan dibangun. (Liaw & Wiener, 2002). *Tuning hyperparameter* adalah nilai parameter *Random Forest* yang harus dioptimalkan sehingga dapat meningkatkan hasil prediksi dibandingkan dengan *default* parameter.

Banyaknya jumlah pohon ($n_{tree}$) adalah salah satu parameter regresi *Random Forest* yang dapat dioptimalkan untuk mendapatkan hasil prediksi yang optimal. Salah satu sentral parameter dari *Random Forest* adalah jumlah fitur acak untuk setiap *split* ($m_{try}$), yang didefinisikan sebagai jumlah kandidat fitur pada setiap pemisahan merupakan parameter yang nilai optimalnya tergantung pada *dataset* yang terpilih ketika membangun *tree*. Pada metode pemilihan jumlah fitur acak yang disarankan Breiman yaitu $\sqrt{p}$ untuk klasifikasi dan *p/3* untuk regresi, dan *p* adalah jumlah fitur.

### 3.8.2 *K-Fold Cross Validation*

*Cross-validation* (CV) adalah metode yang digunakan untuk mengevaluasi kinerja model. Teknik *Cross Validation* digunakan untuk melakukan prediksi model dan memperkirakan seberapa akurat sebuah model prediksi. Dalam masalah prediksi, model biasanya diberikan *dataset* yang diketahui untuk digunakan dalam menjalankan *dataset* pelatihan terhadap *dataset* pengujian. Cara kerja *K-fold cross validation* yaitu dengan mengelompokkan data latih dan data uji yang saling terpisah, kemudian melakukan pengujian yang diulang sebanyak *K* kali.

*K-fold* adalah metode *Cross Validation* yang popular. Metode standar untuk evaluasi yaitu *10-fold cross-validation* dengan nilai *k*=10 sangat umum diterapkan di bidang *machine learning* dan banyak hasil eksperimen menunjukan bahwa menggunakan *k*=10 adalah pilihan terbaik untuk mendapatkan estimasi hasil prediksi yang akurat. (Schneider, 1997)

### 3.8.3 *Variable Importance*

Variabel *importance* adalah ukuran yang mencerminkan kepentingan variabel prediktor dalam memprediksi. Ukuran variabel *importance* ini juga dapat menangkap hubungan *non-linear* dan efek interaksi dan juga berlaku ketika lebih banyak variabel daripada jumlah observasi. (Breiman, 2001)

Nilai *variable importance* diukur berdasarkan pemilihan variabel prediktor menjadi variabel pemisah terhadap *error*. Pada data *X*, *x* dikatakan kuat mempengaruhi variabel y jika menghasilkan penurunan *error* yang signifikan dalam memilih *x* dari hasil *random subset* untuk dijadikan pemisahan. *Variable importance* hanya menghitung seberapa penting variabel prediktor dalam mempengaruhi variabel responnya.

## 3.9 Evaluasi Model

### 3.9.1 *Root Mean Square Error* (RMSE)

*Root Mean Square Error* (RMSE) sering digunakan untuk mengukur perbedaan antara nilai yang diprediksi oleh model dan nilai aktual.

$$\boldsymbol{RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}} \tag{3.1}$$

### 3.9.2 Akurasi

Evaluasi model prediksi dengan mengukur tingkat akurasi dalam memprediksi menggunakan pendekatan ukuran *error* dari hasil prediksi. Nilai *error* dapat dihitung dengan menggunakan *Mean Absolute Percentage Error* (MAPE). MAPE adalah rata-rata dari keseluruhan persentase selisih antara data aktual dan data hasil prediksi. MAPE dirumuskan sebagai berikut:

$$MAPE = \frac{1}{N}\sum_{t=1}^{N}\left|\frac{A_t - F_t}{A_t}\right| \tag{3.2}$$

Keterangan:

$A_t$ = nilai data sesungguhnya (data aktual)

$F_t$ = nilai data hasil prediksi

$N$ = jumlah banyaknya observasi

$t$ = waktu

Nilai akurasi didapatkan dengan persamaan sebagai berikut:

$$Akurasi = (1 - MAPE)x100 \tag{3.3}$$

### 3.10 *Scraping Web*

*Scraping web* adalah salah satu teknik pengambilan data semi terstruktur dari internet pada umumnya berupa halaman *website* dalam bahasa *HTML* atau *XTML,* untuk mempermudah mengambil data dari sebuah *website* secara otomatis tanpa perlu menyalin data secara manual dan menganalisa dokumen tersebut untuk diambil data tertentu dari halaman *website* yang digunakan untuk kepentingan lain. (Turland, 2010).

Proses pengumpulan data menggunakan teknik *scraping web*. Adapun metode *scraping web* yang dilakukan dengan cara mengambil data tertentu dari *website* dengan menggunakan *Python* dan *BeautifulSoup*. *BeautifulSoup* adalah salah satu *Python library* yang dapat digunakan untuk melakukan *web scraping* yang mengekstraksi *file website*. *BeautifulSoup* menawarkan metode sederhana dan efektif untuk navigasi, pencarian dan modifikasi data tertentu. Proses *scraping web* dengan bantuan *library BeautifulSoup* memiliki fungsi yang diperlukan untuk mengambil data dari sebuah halaman *website* karena mudah untuk diaplikasikan.

### 3.11 *Website*

*Website* merupakan halaman yang akan digunakan pada tampilan informasi, gambar, video maupun gabungan dari keseluruhan tersebut untuk sifat yang tetap (statis) dan juga yang berubah-ubah (dinamis), akan membentuk suatu rangkaian yang saling terkait, dan dihubungkan menggunakan *link*. Pada desain *website* menggunakan beberapa *software* seperti bahasa pemrograman *HTML* dan *CSS*. Dalam membangun sebuah *website* menggunakan *HTML* dan *CSS* dengan tugas yang berbeda namun berkesinambungan. *HTML* digunakan untuk membangun suatu *website* sedangkan *CSS* digunakan untuk mendesain tampilan *website* yang dibangun dengan bahasa *HTML*. (Hadi, 2016).

*HTML (Hypertext Markup Language)* adalah sebuah bahasa *formatting* yang digunakan dalam membuat *page website.* Dalam bidang pemrograman berbasis *website*, *HTML* menjadi dasar pada *page website.* Sebuah *file HTML* disimpan dengan ekstensi `.html` dan dapat diakses menggunakan *web browser*. *HTML* memiliki beberapa elemen yang tersusun dari *tag* yang

memiliki fungsinya masing-masing, seperti tag *heading*, paragraf, *form*, *button*, *list*, membuat *hyperlink* yang menghubungkan antar halaman *website* dan lain-lain.

*CSS (Cascading Style Sheets)* adalah bahasa pemrograman yang digunakan untuk *web design. CSS* digunakan untuk mendesain sebuah tampilan halaman *web.* Dalam melakukan *design* halaman *web, CSS* menggunakan penanda yang dikenal dengan *id* dan *class. CSS* dapat mengubah *font, size,* dan *colour* serta format *font,* mengatur ukuran *layout*, warna elemen, mengubah tampilan *form,* membuat halaman *website* yang responsif, menarik dan lain-lain.

### 3.12 *Feature Encoding*

Dalam banyak ilmu *Machine Learning*, terdapat satu hal yang perlu diperhatikan bahwa sebagian besar algoritma bekerja lebih baik dengan data bertipe numerik untuk mencapai hasil yang lebih baik. Perkembangan data secara pesat membuat tipe data tidak hanya berupa numerik, namun juga berupa teks atau label pada suatu *dataset*. Oleh karena itu, dalam penelitian perlu dilakukan mengubah data menjadi numerik untuk membuat sebuah model. Salah satu teknik yang dapat dilakukan adalah *feature encoding* yaitu teknik mengkonversi variabel teks atau kategorik menjadi nilai numerik.

Terdapat dua cara yang paling umum dilakukan dengan menggunakan *Label Encoder* atau *One Hot Encoder* sebagai berikut:

1. *Label Encoding*: metode ini mengacu pada konversi label ke dalam bentuk angka sehingga dapat mengubahnya menjadi bentuk yang dapat dibaca oleh mesin, langkah ini merupakan *pre-processing* data yang penting untuk *dataset* terstruktur dalam *supervised learning.*
2. *One Hot Encoding*: metode ini akan mengkonversi setiap labelnya menjadi 0 dan 1, dan setiap label akan dipecah menjadi beberapa kolom sesuai dengan banyak label.

### 3.13 *Framework Flask*

*Flask* adalah sebuah *web framework* dengan bahasa *python* yang menyediakan *library* yang dapat digunakan untuk membangun *website*. Untuk membuat *website* dengan menggunakan *Python* diperlukan *framework* pengembangan *web*. Pada *python framework* pengembangan *web* yang populer dan mudah dipelajari terdiri dari *Django* dan *Flask*. Sebagai pemula untuk membuat sebuah *web* dengan bahasa *python* sebaiknya menggunakan *framework flask*, karena dapat membuat *website* dengan lebih mudah dan dapat diterapkan untuk membuat *website* berskala kecil. *Flask* adalah salah satu *framework* yang tersedia di *repository* publik *python* yang semua orang dapat mengunduh dan menggunakannya.

### 3.14 *Heroku*

Heroku adalah sebuah *cloud platform* yang menjalankan bahasa pemrograman tertentu seperti *Phyton, Ruby, Java, PHP, Node.js* dan lain-lain. Heroku adalah salah satu *web hosting* berbasis *cloud* dan *cloud computing* merupakan salah satu contoh dari komputasi modern. *Cloud computing* ini berfungsi dalam hal *deployment* sebuah aplikasi *web*. Heroku termasuk ke dalam salah satu model layananan *cloud computing* yaitu *Platform As A Service* (PaaS), sehingga dalam melakukan *deploy* aplikasi *web* ke Heroku cukup dengan melakukan konfigurasi aplikasi yang ingin di-*deploy* dan menyediakan *platform* yang memungkinkan pelanggan untuk mengembangkan, menjalankan, dan mengelola aplikasi tanpa kompleksitas membangun dan memelihara infrastruktur yang biasanya terkait dengan pengembangan dan peluncuran aplikasi. Hal yang bersangkutan dengan *server*, *building*, *deploying*, *running*, dan *scaling* aplikasi dapat dilakukan oleh Heroku. Heroku dapat melakukan *deploy* secara otomatis terhadap aplikasi *web* yang di-*upload* menggunakan *git*. Untuk menggunakan Heroku ini pastikan bahwa telah terpasang perangkat *git*, karena aplikasi yang akan dibangun terhubung dengan Heroku melalui *git.* (Hanjura , 2014)

# BAB IV
# METODOLOGI PENELITIAN

## 4.1 Populasi dan Sampel Penelitian

Populasi dalam penelitian ini adalah data penjualan televisi dengan spesifikasi yang dimiliki masing-masing televisi yang ditampilkan pada *website* "Pricebook". Sedangkan dalam pemilihan sampel dalam penelitian ini adalah data penjualan televisi dengan hanya mengambil beberapa variabel antara lain: Nama Produk, Merek, Tipe Layar, Ukuran Layar dan Harga dengan total keseluruhan data sebanyak 925 data.

## 4.2 Teknik Pengumpulan Data

Pada penelitian ini menggunakan data penjualan televisi di Indonesia dengan spesifikasi yang dimiliki. Data diperoleh dari sebuah situs *website* yang bersumber di *internet* dan untuk mendapatkan data tersebut dapat dilakukan dengan teknik *web scraping.* Dengan dilakukannya *web scraping* dalam penelitian ini mempermudah untuk mengambil data dari sebuah *website* secara otomatis tanpa perlu menyalin data secara manual.

Proses pengumpulan data pada penelitian ini menggunakan teknik *scraping web*. Adapun metode *scraping web* yang dilakukan adalah dengan cara mengambil data tertentu dari *website* dengan menggunakan *Python* dan *BeautifulSoup*. *BeautifulSoup* adalah salah satu *Python library* yang dapat digunakan untuk melakukan *web scraping* yang mengekstraksi *file website* dengan format *HTML* atau *XTML*. *BeautifulSoup* adalah alat yang mudah untuk melakukan *scraping web* berbagai data.

## 4.3 Variabel dan Definisi Operasional

Variabel yang digunakan dalam penelitian ini terdapat lima variabel yang terdiri dari satu variabel terikat (*dependent variable*) yaitu variabel harga dan empat variabel bebas (*independent variable*) antara lain: nama produk, merek, ukuran layar dan tipe layar. Penjelasan setiap variabel yang digunakan dalam penelitian ini ditampilkan pada tabel 4.1 sebagai berikut:

**Tabel 4.2** Definisi Operasional Variabel

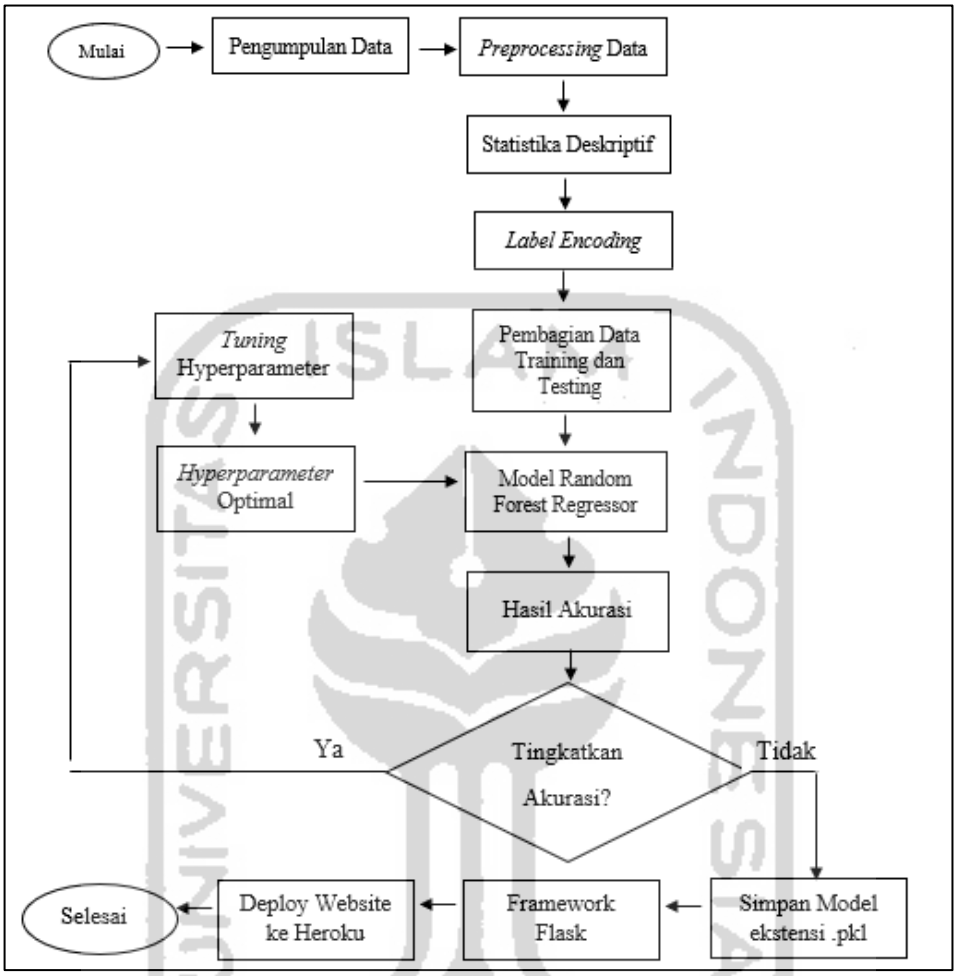
| No | Variabel | Definisi Variabel |
|---|---|---|
| 1. | Produk | Variabel yang mencantumkan nama televisi beserta nomor serinya yang bertujuan untuk membedakan produk televisi satu dengan lainnya. |
| 2. | Merek | Variabel yang mencatumkan nama merek dagang televisi yang diproduksi oleh suatu perusahaan elektronik tertentu. |
| 3. | Tipe | Variabel yang menunjukan televisi hadir dalam berbagai tipe layar seperti: Plasma, LCD, LED, OLED, dan QLED. |
| 4. | Ukuran | Variabel yang menunjukan ragam ukuran layar televisi dalam satuan *inch*. |
| 5. | Harga | Variabel yang mencantumkan nominal harga penjualan televisi berdasarkan spesifikasi televisi yang dimiliki. |

## 4.4 Metode Analisis

Metode analisis yang digunakan dalam penelitian ini adalah *Random Forest* dengan bahasa pemrograman *Python* yang digunakan untuk memprediksi harga penjualan televisi. Perangkat yang digunakan dalam penelitian ini menggunakan *Microsoft Excel* dan *Python* 3.7.0. Adapun tahapan metode analisis yang dilakukan dalam penelitian ini antara lain: *scraping web*, *pre-processing* data, analisis deskriptif, melatih model *Random Forest* dalam memprediksi harga penjualan televisi, pembuatan *website* prediksi harga penjualan televisi dengan menggunakan *framework Flask* dan *deploy* aplikasi *web* menggunakan Heroku.

### 4.5 Tahapan Analisis Data

Tahapan atau langkah dalam penelitian ini digambarkan dalam *flowchart* sebagai berikut:



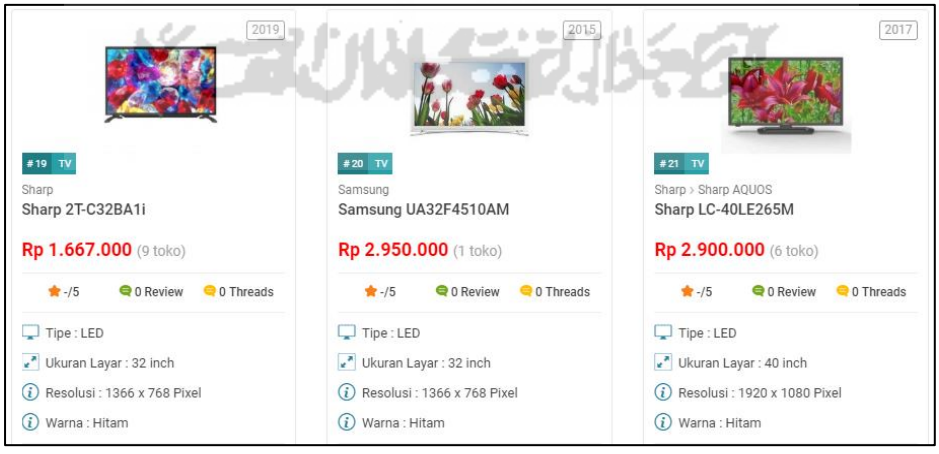

**Gambar 4.5** *Flowchart* Tahapan Penelitian

# BAB V
# PEMBAHASAN

## 5.1 Pengumpulan Data

Data yang digunakan dalam penelitian ini adalah data penjualan televisi yang terdapat pada suatu situs *website* Pricebook yang dapat diakses dengan menggunakan internet. Proses pengumpulan data pada penelitian ini menggunakan teknik *scraping website*. Adapun metode *scraping web* yang dilakukan adalah dengan cara mengambil data tertentu dari *website* tersebut dengan menggunakan *Python* dan *BeautifulSoup*. *BeautifulSoup* adalah salah satu *Python library* yang dapat digunakan untuk melakukan *scraping web* yang mengekstraksi *file website* dengan format *HTML* atau *XTML*. *BeautifulSoup* adalah alat yang mudah untuk melakukan *scraping* berbagai data, seperti daftar, paragraf, tabel dan lainnya. *BeautifulSoup* menawarkan metode sederhana dan efektif untuk navigasi, pencarian dan modifikasi data tertentu. Proses *scraping web* dengan bantuan *library BeautifulSoup* memiliki fungsi yang diperlukan untuk mengambil data dari sebuah halaman *website* karena mudah untuk diaplikasikan.

Data penjualan televisi yang digunakan dalam penelitian ini hanya menggunakan beberapa variabel antara lain: nama produk, merek, tipe layar, ukuran layar dan harga. Tampilan halaman *website* Pricebook yang akan dilakukan proses *scraping web* sebagai berikut:




**Gambar 5.6** Tampilan *Website*

Dalam proses *scraping web* yang dilakukan dengan menggunakan *python* dan *BeautifulSoup* untuk *scraping* data dari *website* yang menjadi sumber data dan hasil *scraping web* tersebut akan disimpan ke dalam *file excel* yang berekstensi *csv*.

Dalam penelitian ini, hal pertama yang dilakukan dalam proses *scraping web* dengan menggunakan *BeautifulSoup* adalah meng-*install* modul yang digunakan yaitu *BeautifulSoup* dengan cara: "`pip install beautifulsoup4`" pada *command prompt.* Penulis melakukan proses *scraping web* pada *Jupyter Notebooks,* sehingga perlu meng*import* beberapa *packages* yang dibutuhkan antara lain: *pandas, numpy, BeautifulSoup, urlopen* dan *requests*.

```
print("Masukkan keyword : ")
keyword = input(" > ")
url = requests.get("https://www.pricebook.co.id/"+keyword)
url = url.text
url_base_page = "https://www.pricebook.co.id/"+keyword+"?price=100000---9999999999&page="
url_page = [url]
page_dicari = input (" Total Pencarian Page :  ")
page_dicari = int(page_dicari)

for i in range(1,page_dicari) :
    i += 1
    prhalaman = requests.get(url_base_page + str(i))
    prhalaman = prhalaman.text
    url_page.append(prhalaman)
```

**Gambar 5.7** Proses *Input url Web* yang akan di *Scraping Web*

Dalam penelitian ini ingin men*scraping* data penjualan televisi dari *website* Pricebook, pada Gambar 5.2 terlihat *syntax* untuk mendeklarasikan alamat URL terlebih dahulu dengan memasukan alamat *website* (*link*) dan selanjutnya melakukan *request* dengan menggunakan *urlopen* untuk membuka *link* tersebut dan untuk mengambil data *website* yang akan di*scraping.* Dalam proses *scraping web* ini, menggunakan *keyword* yang ditambahkan setelah *link* utama dari Pricebook. *Keyword* dapat digunakan untuk meng*input* kata kunci nama kategori yang ingin dicari dalam *website* tersebut. Sebagai contoh dalam penelitian ini, akan dilakukan *scraping web* untuk mendapatkan data penjualan televisi sehingga *keyword* yang di*input*kan adalah kategori "tv". Dalam *scraping web* dapat dilakukan proses *scraping* untuk mendapatkan data dari keseluruhan halaman dalam sekali *running*. Dalam satu halaman di *website* tersebut terdapat 21 daftar televisi. Dalam proses *scraping* ini, penulis melakukan pembatasan harga dengan minimal harga 100.000 hingga 9.999.999.999 (tidak terdapat batasan harga maksimal) dengan tujuan untuk

mendapatkan data penjualan televisi yang telah tercantumkan harganya. Selanjutnya diperlukan meng*input* "Total Pencarian *Page*: " yang berarti mengetikan jumlah berapa banyak halaman yang akan di-*scraping* yaitu sebanyak 45 halaman. Selanjutnya dilakukan perulangan dari halaman pertama hingga sebanyak halaman yang di-*input* sebelumnya, untuk mendapatkan data *scraping* semua halaman dalam sekali *running*.

```
nama_produk = []
merek_produk = []
ukuran_produk = []
tipe_layar = []
harga_produk = []

for halaman in url_page :
    soup = BeautifulSoup(halaman, "lxml")
    produk = soup.find_all("div", "feat-prod")
    for p in produk:
            nama = p.find("a", "link-inherit").get_text()
            merek = p.find('div','category-breadcrumb').contents[1].contents[1].get_text()
            UnJ = p.find_all("span","pb-primary-specification")
            ukuran = UnJ[1].get_text()
            tipe = UnJ[0].get_text()
            harga = p.find("h4", "title-price").get_text()
```

**Gambar 5.3** Fungsi Mengambil Data Pada Halaman *Website*

Selanjutnya seperti pada Gambar 5.3 ditunjukan koding untuk membuat suatu nama variabel yang fungsinya untuk menyimpan hasil *scraping* masing-masing variabel ke dalam bentuk *array*. Misalkan variabel nama_produk akan menyimpan data hasil *scraping* untuk variabel nama-nama produk TV dan seterusnya untuk variabel lainnya. Selanjutnya melakukan proses *parsing* (mengurai) *html* untuk setiap halaman pada keseluruhan halaman yang dicari yang sebelumnya telah didefinisikan oleh variabel *url_page* dan menggunakan *parser* "lxml" yang tersedia di *Library BeautifulSoup* yang akan disimpan ke dalam variabel *soup*.

Pada tahap selanjutnya adalah mens*craping* data spesifik yang diinginkan untuk masing-masing variabel. Dalam proses *scraping* setiap variabel atau elemen perlu diketahui *tag html* elemen yang didapatkan dengan cara *inspect element.* Selanjutnya dalam pendefinisian variabel produk yang menunjukan untuk menyeleksi semua *tag* setiap elemen yang tersimpan di dalam *class* yang sama. Oleh karena itu, pada variabel produk dengan menggunakan fungsi *find_all( )* untuk mencari seluruh *tag div* yang berada di dalam *class name container* "*feat-prod*". Selanjutnya untuk setiap "*p*" di dalam variabel produk, dilakukan pendefinisian untuk masing-masing variabel menggunakan *tag* elemen sesuai dengan *class*nya. Selanjutnya untuk variabel

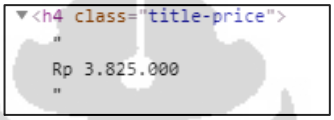
nama menggunakan fungsi *find( )* dengan *tag* berupa *a* yang berada di dalam *class "link-inherit"* dengan menggunakan *get_text( )* untuk mengekstrak teks nama TV. Pada variabel merek menggunakan fungsi *find( )* dengan tag berupa *div* yang berada di dalam *class "category-breadcrumb"* dan mengambil *child class* dengan menggunakan perintah *contents[ ]* untuk mengambil nama merek. Pada Gambar 5.4 merupakan tampilan *inspect element* untuk nama produk televisi dan merek televisi sebagai berikut:


**Gambar 5.4** *Inspect Element* Variabel Nama dan Merek

Selanjutnya pada variabel UnJ (ukuran dan jenis) yang terdiri dari ukuran layar dan tipe layar yang disebabkan karena keduanya terdapat di dalam satu *class* yang sama yaitu *class "pb-primary-specification"*. Oleh karena itu, diproses dengan menggunakan fungsi *find_all( )* dengan tag *span* yang berada di dalam *class "pb-primary-specification"*, selanjutnya dipisahkan antara variabel ukuran dan tipe dengan mengidentifikasi informasi berdasarkan posisinya di dalam daftar atau urutan, sehingga *item* data pertama dalam baris diidentifikasikan oleh indeks [0] dan yang kedua atau berada dibawahnya diidentifikasikan oleh indeks [1] serta mengaplikasikan fungsi *get_text* untuk mengekstrak teks kedua elemen variabel (ukuran dan tipe) yang berada di dalam *class* yang sama. Oleh karena itu, dalam pemisahan variabel ukuran dan tipe dapat diketahui dengan melihat posisi urutannya. Variabel tipe berada di urutan atas maka diindentifikasikan dengan indeks [0] dan untuk variabel ukuran karena urutannya berada di bawahnya maka diindentifikasikan dengan indeks [1]. Pada Gambar 5.5 merupakan tampilan *inspect element* untuk ukuran televisi dan tipe televisi sebagai berikut:


**Gambar 5.5** Tampilam *Inspect Element* untuk Variabel Ukuran dan Tipe

Selanjutnya untuk variabel harga dengan menggunakan fungsi *find( )* dengan *tag* berupa *h4* yang berada di dalam *class "title-price"* dengan menggunakan *get_text( )* untuk mengekstrak harga TV. Pada Gambar 5.6 merupakan tampilan *inspect element* untuk harga televisi sebagai berikut:




**Gambar 5.6** Tampilam *Inspect Element* untuk Variabel Harga

Selanjutnya pada Gambar 5.7 terlihat koding untuk hasil akhir *scraping website* yang disimpan ke dalam *file excel* berekstensi *csv* terdiri dari beberapa kolom antara lain: produk, merek, ukuran, tipe, dan harga sebagai berikut:

```
'''result disimpan pada file csv'''
produk_dict = {"produk":nama_produk,"merek":merek_produk,
               "ukuran":ukuran_produk,"tipe":tipe_layar,
               "harga":harga_produk}
df = pd.DataFrame(produk_dict,columns = ["produk","merek","ukuran","tipe", "harga"])
df.sort_values('harga')
df.to_csv("dataTVfix.csv",sep=",")
```

**Gambar 5.7** Proses Menyimpan Hasil *Scraping* Menjadi *File csv*

Selanjutnya data penjualan televisi dari hasil *scraping website* disajikan dalam *file excel* yang berekstensi *csv* terlihat pada Gambar 5.8 sebagai berikut:

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | produk | merek | ukuran | tipe | harga |
| 2 | CooCaa 40S5G | CooCaa | 40 | OLED | 2225000 |
| 3 | CooCaa 32S5C | CooCaa | 32 | OLED | 1870000 |
| 4 | LG 43UM7100 | LG | 43 | LED | 4490000 |
| 5 | Sharp AQUOS LC-32LE180i | Sharp | 32 | LED | 1648000 |
| 6 | Polytron PLD 20D901 | Polytron | 20 | LED | 1090000 |
| 7 | Toshiba 40PU200EJ | Toshiba | 40 | LED | 4350000 |
| 8 | Sharp AQUOS LC-24LE175i | Sharp | 24 | LED | 1169000 |
| 9 | Polytron PLD 43TV865 | Polytron | 43 | LED | 4250000 |
| 10 | Polytron Cinemax Soundbar PLD-32B1550 | Polytron | 32 | LED | 1998000 |

**Gambar 5.8** Data Hasil *Scraping Website*

## 5.2 *Pre-processing* Data

Tahapan *pre-processing* perlu untuk dilakukan karena terdapat proses data transformasi dan data *cleaning* yang bertujuan untuk menghasilkan data dengan variabel-variabel yang siap digunakan untuk analisis selanjutnya. Dalam tahapan *pre-processing* ini menggunakan *regular expressions*, sehingga perlu untuk *import* modul "*re*". Dilakukan *pre-processing* untuk variabel harga, tipe dan ukuran dengan menggunakan *regular expressions*.

Pada variabel harga merupakan variabel yang menyimpan nominal harga setiap televisi. Pada variabel harga diharapkan hasil akhirnya berupa *integer*, sehingga dilakukan proses transformasi bentuk dengan menghilangkan kata "Rp" yang berada di depan nominal harga dan menghilangkan tanda titik pada nominal harga. Pada variabel harga dengan menggunakan metode `re.sub()` untuk mengganti semua karakter di dalam *tag html* yang ditemukan dan menggantinya ke dalam bentuk *empty string*. Selanjutnya dilakukan eliminasi hasil *string* yang didapatkan yang berbentuk selain dalam *range* 0 sampai dengan 9 (*integer*), maka menggantinya menjadi *null*. Proses *pre-processing* untuk variabel harga untuk mendapatkan hasil dalam *integer* ditunjukan pada Gambar 5.9 sebagai berikut:

```
harga = re.sub("([\(\[]).*?([\)\]])", "\g<1>\g<2>", harga)
harga = re.sub('[^0-9]','',harga)
```

**Gambar 5.9** Langkah *Pre-Processing* Variabel Harga

Pada variabel ukuran merupakan variabel yang menyimpan nilai ukuran layar setiap televisi. Pada variabel ukuran ingin didapatkan hasil berupa *integer*, maka menggunakan metode `re.sub()` untuk mengganti semua karakter di dalam *tag html* yang ditemukan dan menggantinya ke dalam bentuk *empty string*. Selanjutnya dilakukan eliminasi hasil *string* yang didapatkan yang berbentuk selain dalam *range* 0 sampai dengan 9 (*integer*), maka menggantinya menjadi *null*. Selanjutnya pada variabel ukuran juga dilakukan penghapusan "\n" dan menggantinya menjadi *null*. Proses *pre-processing* untuk variabel ukuran untuk mendapatkan hasil *integer* ditunjukan pada Gambar 5.10 sebagai berikut:

```
ukuran = re.sub("([\(\[]).*?([\)\]])", "\g<1>\g<2>", ukuran)
ukuran = re.sub('[^0-9]','',ukuran)
ukuran = ukuran.replace("\n","")
```

**Gambar 5.10** Langkah *Pre-Processing* Variabel Ukuran

Selanjutnya variabel tipe merupakan variabel yang menyimpan tipe layar setiap televisi. Pada variabel tipe juga dilakukan pengahapusan kata "Tipe : " dan menggantinya menjadi *null*, sehingga hasil variabel tipe hanya berupa nama tipe layar. Selain itu juga dilakukan pembersihan spasi yang terletak di depan nama tipe layarnya dengan cara menghapus spasi (" ") dan menggantinya menjadi *null*. Proses *pre-processing* untuk variabel tipe ditunjukan pada Gambar 5.11 sebagai berikut:

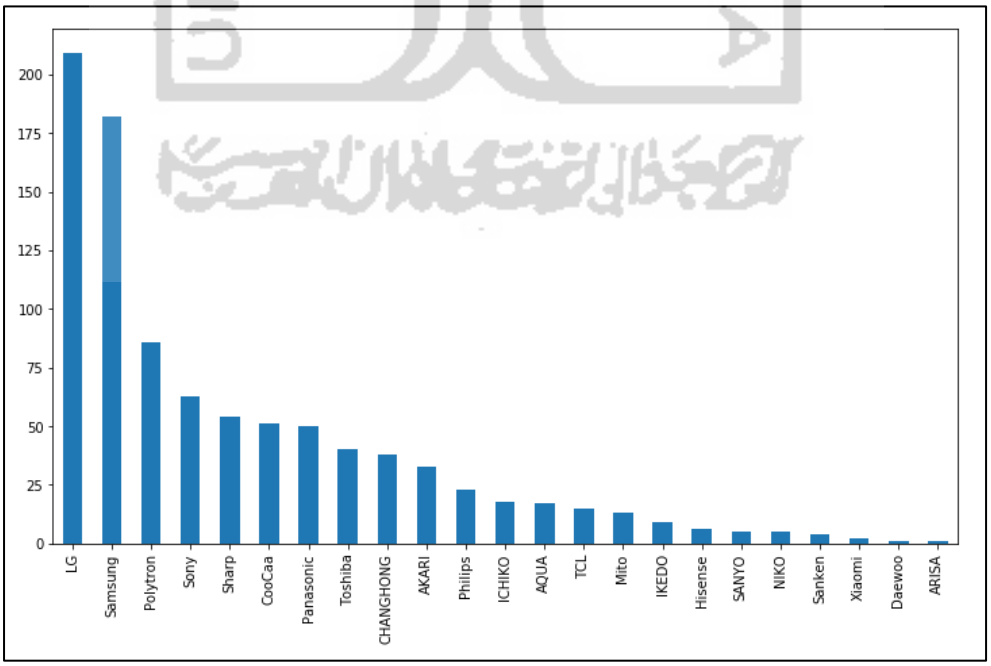
```
tipe = tipe.replace("Tipe :","")
tipe = tipe.replace(" ","")
```

**Gambar 5.11** Langkah *Pre-Processing* Variabel Tipe

## 5.3 Statistika Deskriptif

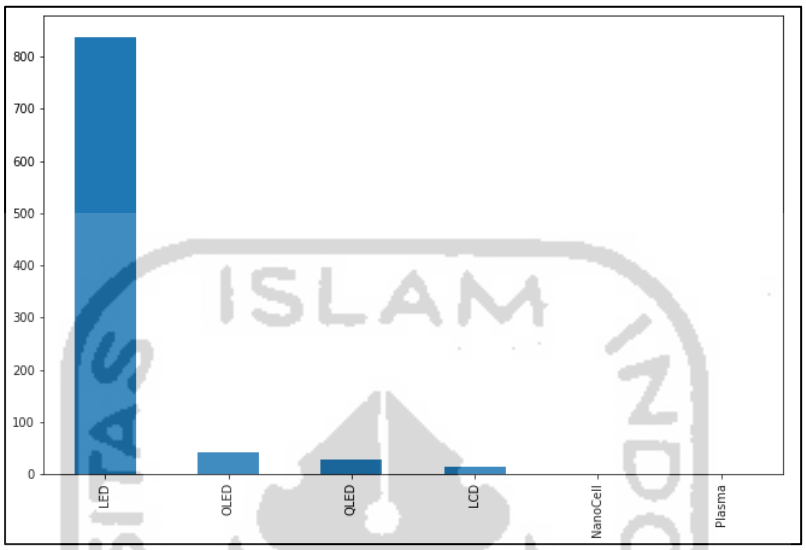
Analisis statistika deskriptif adalah suatu metode yang berfungsi untuk mendeskripsikan atau memberikan gambaran umum terhadap objek yang diteliti melalui data atau sampel yang telah terkumpul, tanpa melakukan analisis ataupun menarik kesimpulan. (Sugiyono, 2009)

Selanjutnya statistika deskriptif digunakan untuk mengetahui gambaran umum dari data dalam penelitian ini yaitu penjualan televisi sebagai berikut:



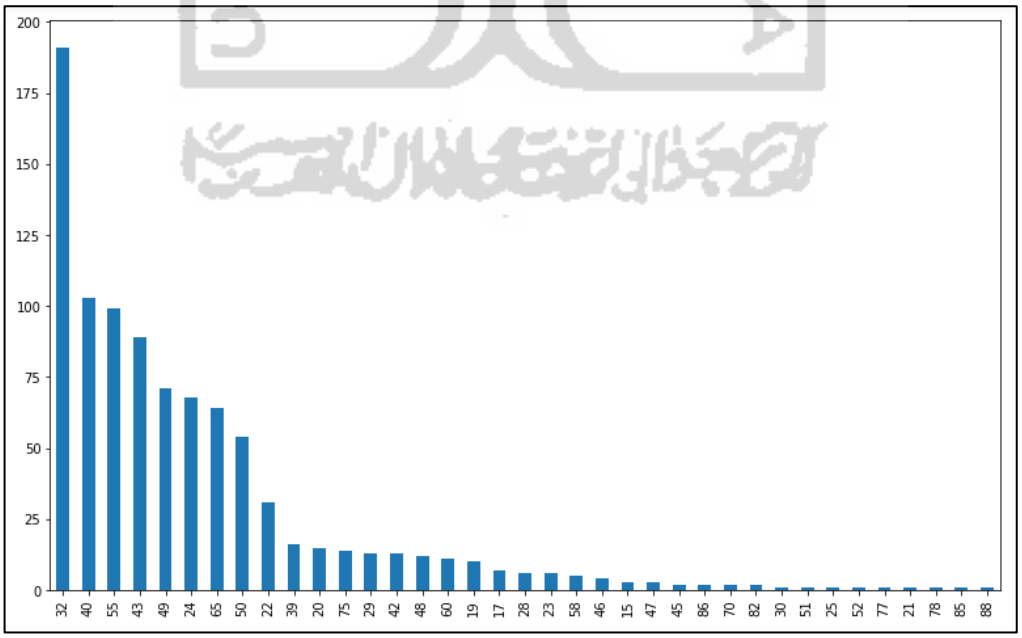

**Gambar 5.12** *Plot* Jumlah Merek Terbanyak

Berdasarkan *bar chart* pada Gambar 5.12, diketahui bahwa LG merupakan merek televisi dengan jumlah data terbanyak yang tersedia di dalam *dataset* yaitu sebanyak 209 unit, disusul oleh merek Samsung sebanyak 182 unit, merek Polytron sebanyak 86 unit dan merek lainnya.



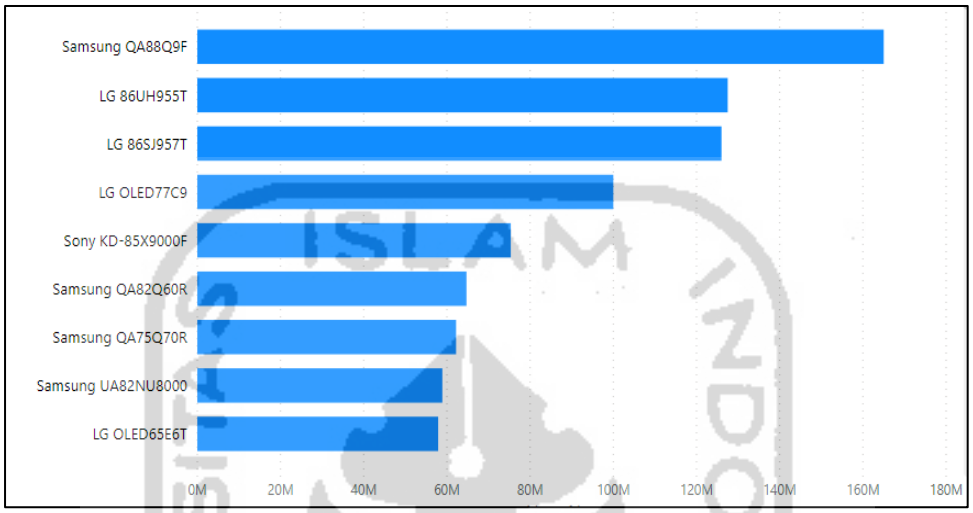

**Gambar 5.13** *Plot* Jumlah Tipe Layar Terbanyak

Berdasarkan *bar chart* pada Gambar 5.13, diketahui bahwa televisi dengan tipe layar LED merupakan tipe layar televisi dengan jumlah data terbanyak yang tersedia di dalam *dataset* yang mencapai 838 unit, disusul oleh tipe layar OLED sebanyak 42 unit, QLED sebanyak 29 unit, LCD sebanyak 14 unit, tipe layar *nanocell* dan plasma masing-masing sebanyak 1 unit.



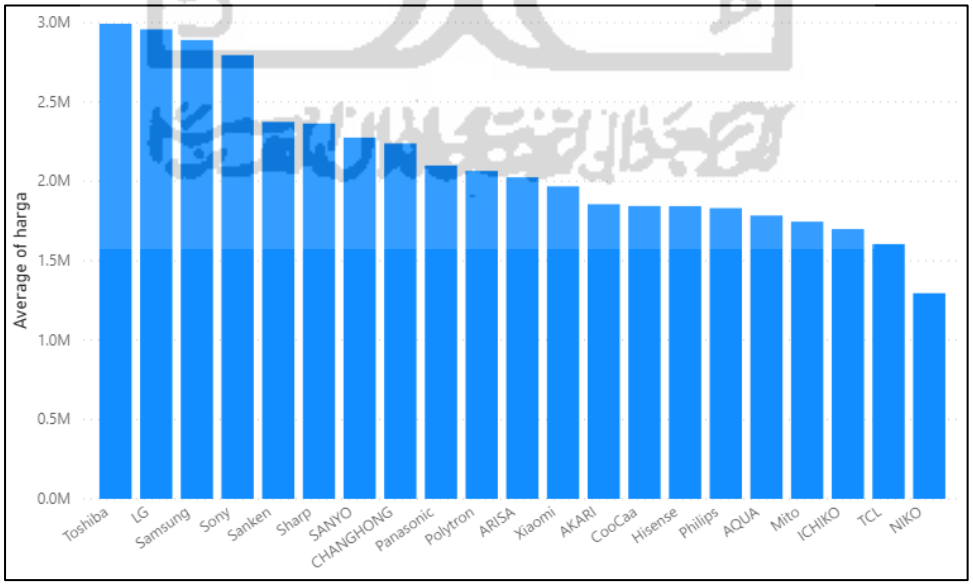

**Gambar 5.14** *Plot* Jumlah Ukuran Layar Terbanyak

Berdasarkan *bar chart* pada Gambar 5.14, diketahui bahwa televisi dengan ukuran layar 32 *inch* merupakan ukuran layar televisi dengan jumlah data terbanyak yang tersedia di dalam *dataset* yang mencapai 191 unit. Ukuran layar terbanyak urutan kedua disusul oleh ukuran 40 *inch* sebanyak 103 unit, urutan ketiga disusul oleh ukuran 55 *inch* sebanyak 99 unit dan seterusnya.




**Gambar 5.15** *Plot* Produk Televisi dengan Harga Jual Termahal

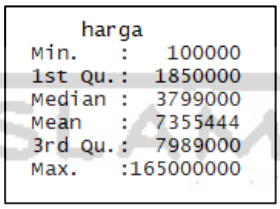
Berdasarkan *bar chart* pada Gambar 5.15, diketahui bahwa produk televisi urutan pertama dengan harga jual termahal yaitu Samsung QA88Q9F sebesar Rp 165.000.000. Selanjutnya pada urutan kedua produk dengan harga jual termahal yaitu LG 86UH955T sebesar Rp 127.500.000 dan disusul oleh produk LG 86SJ957T sebesar Rp 126.000.000 dan seterusnya.




**Gambar 5.16** *Plot* Rata-Rata Harga Jual Berdasarkan Merek Televisi dengan Ukuran Layar 32 *inch*

Berdasarkan *bar chart* pada Gambar 5.16, diketahui bahwa merek televisi urutan pertama dengan rata-rata harga penjualan tertinggi dengan *cluster* ukuran layar 32*inch* yaitu merek Sony sebesar Rp 2.992.000. Selanjutnya pada urutan kedua merek LG dengan rata-rata harga penjualan sebesar Rp 2.956.000 dan pada urutan ketiga merek Samsung dengan rata-rata harga penjualan sebesar Rp 2.889.000 dan seterusnya.

Selanjutnya gambaran umum harga penjualan televisi sebagai berikut:




**Gambar 5.17** Deskriptif Variabel Harga

Diketahui pada Gambar 5.17 diketahui harga penjualan televisi secara umum bahwa harga penjualan televisi terendah yaitu dengan harga Rp 100.000 pada produk Sharp AQUOS LC-24LE155M, sedangkan harga penjualan televisi tertinggi dengan harga Rp 165.000.000 pada produk Samsung QA88Q9F. Rata-rata harga jual televisi yaitu Rp 7.355.444.

## 5.4 *Label Encoder*

Dalam banyak ilmu *Machine Learning*, terdapat satu hal yang perlu diperhatikan bahwa sebagian besar algoritma bekerja lebih baik dengan data bertipe numerik untuk mencapai hasil lebih baik. Oleh karena itu, dalam penelitian perlu dilakukan mengubah data kategorikal menjadi data numerik untuk membuat sebuah model. Ada banyak cara untuk mengubah nilai teks atau kategorik menjadi nilai numerik, penulis menggunakan metode *Label-Encoder* yang tersedia dalam bagian dari *SciKit-learn* yang digunakan untuk mengubah data kategorik menjadi data numerik.

Pendekatan *Categorical's Codes* mengharuskan kolom variabel kategorik memiliki tipe data '*category*'. Secara *default*, kolom *non*-numerik akan bertipe '*object*'. Oleh karena itu, perlu mengubahnya menjadi tipe data '*category*' pada variabel yang akan dilakukan konversi dari kategorik menjadi numerik seperti variabel produk, merek, dan tipe menggunakan metode "`astype`" yang ditampilkan pada Gambar 5.18 sebagai berikut:

```
dataTV["produk"] = dataTV["produk"].astype('category')
dataTV["merek"] = dataTV["merek"].astype('category')
dataTV["tipe"] = dataTV["tipe"].astype('category')
dataTV.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 925 entries, 0 to 924
Data columns (total 5 columns):
produk    925 non-null category
merek     925 non-null category
ukuran    925 non-null int64
tipe      925 non-null category
harga     925 non-null int64
dtypes: category(3), int64(2)
memory usage: 66.4 KB
```

**Gambar 5.18** Mengubah Tipe Data Menjadi *Category*

Selanjutnya memulai untuk mengkonversi variabel dengan tipe data '*category*' menjadi numerik untuk beberapa variabel antara lain: produk, merek dan tipe. Hasil konversi akan disimpan ke dalam variabel yang diberi nama "produk_map", "merek_map" dan "tipe_map" sehingga didapatkan hasil konversinya yang ditampilkan pada Gambar 5.19 sebagai berikut:

```
def get_cat(col):
    return dict([(cat, code) for code, cat in enumerate(col.cat.categories)])

produk_map = get_cat(dataTV.produk)
merek_map = get_cat(dataTV.merek)
tipe_map = get_cat(dataTV.tipe)
```

**Gambar 5.19** Konversi Variabel Produk, Merek dan Tipe

Secara *default* kategori diurutkan berdasarkan abjad, sehingga nama pertama akan diwakili oleh indeks nol. Hasil konversi menjadi data numerik untuk variabel produk, merek dan tipe yang ditampilkan pada Gambar 5.20 sebagai berikut:

```
produk_map
```

```
{'AKARI LE-19D88': 0,
 'AKARI LE-20D88': 1,
 'AKARI LE-20K88': 2,
 'AKARI LE-20V89': 3,
 'AKARI LE-23B88': 4,
 'AKARI LE-23K88': 5,
 'AKARI LE-24K88': 6,
 'AKARI LE-24V89': 7,
 'AKARI LE-25B88': 8,
 'AKARI LE-25V89': 9,
 'AKARI LE-29P57': 10,
 'AKARI LE-29V89': 11,
 'AKARI LE-3288T2': 12,
 'AKARI LE-3289T2': 13,
 'AKARI LE-32D88': 14,
 'AKARI LE-32K88': 15,
 'AKARI LE-32M88': 16,
 'AKARI LE-32P88': 17,
 'AKARI LE-32V90': 18,
 'AKARI LE-32V99': 19,
```

```
merek_map
```

```
{'AKARI': 0,
 'AQUA': 1,
 'ARISA': 2,
 'CHANGHONG': 3,
 'CooCaa': 4,
 'Daewoo': 5,
 'Hisense': 6,
 'ICHIKO': 7,
 'IKEDO': 8,
 'LG': 9,
 'Mito': 10,
 'NIKO': 11,
 'Panasonic': 12,
 'Philips': 13,
 'Polytron': 14,
 'SANYO': 15,
 'Samsung': 16,
 'Sanken': 17,
 'Sharp': 18,
 'Sony': 19,
 'TCL': 20,
 'Toshiba': 21,
 'Xiaomi': 22}
```

```
tipe_map
```

```
{'LCD': 0, 'LED': 1, 'NanoCell': 2, 'OLED': 3, 'Plasma': 4, 'QLED': 5}
```

**Gambar 5.20** Hasil Konversi Variabel Produk, Merek, dan Tipe

Setelah dilakukan konversi dari data bertipe '*category*' menjadi data numerik, sehingga didapatkan hasil akhir data keseluruhan variabel telah berdata numerik yang ditampilkan pada Gambar 5.21 sebagai berikut:

```
dataTV.head()
```

| | produk | merek | ukuran | tipe | harga |
|---|---|---|---|---|---|
| 0 | 127 | 4 | 40 | 3 | 2225000 |
| 1 | 113 | 4 | 32 | 3 | 1870000 |
| 2 | 267 | 9 | 43 | 1 | 4490000 |
| 3 | 770 | 18 | 32 | 1 | 1648000 |
| 4 | 482 | 14 | 20 | 1 | 1090000 |

**Gambar 5.21** Hasil Konversi dengan Metode *Label Encoder*

## 5.5 *Random Forest*

Pada tahapan analisis klasifikasi *Random Forest* dengan menentukan terlebih dahulu variabel independen dan dependennya. Dalam penelitian ini terdapat lima variabel yang terdiri dari empat variabel *independent* (x) antara lain: produk, merek, ukuran dan tipe, sedangkan terdapat satu variabel *dependent* (y) yaitu variabel harga. Pada langkah awal diperlukan untuk membagi data menjadi dua yaitu atribut sebagai variabel "x" dan label sebagai variabel "y". Diketahui bahwa sebagai label (y) hanya terdiri dari variabel harga dan sebagai atribut (x) adalah semua variabel independen kecuali variabel harga yang ditampilkan pada Gambar 5.22 sebagai berikut:

```
y = dataTV['harga']
x = dataTV.drop(['harga'], axis=1)
```

**Gambar 5.22** Langkah Penentuan Atribut dan Label

Untuk melakukan analisis klasifikasi untuk membangun model *machine learning*, data perlu dibagi terlebih dahulu menjadi dua bagian yaitu data *training* dan data *testing*. Data *training* digunakan untuk melatih model dalam memprediksi variabel dependennya sehingga mampu mengenali pola yang ada, sedangkan data *testing* digunakan untuk menguji apakah hasil analisis dari data *training* untuk dapat mengetahui kebaikan model dan mengetahui seberapa besar tingkat ketepatan metode dalam melakukan klasifikasi data. Pada penelitian ini, perbandingan antara data *training* dan data *testing* dengan perbandingan 75% : 25% yang ditampilkan pada Gambar 5.23 sebagai berikut:

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.25)
```

**Gambar 5.23** Pembagian Data *Training* dan Data *Testing*

Tahapan selanjutnya memulai untuk membangun model *Random Forest* yang ditampilkan pada Gambar 5.24. Untuk langkah pengujian pembuatan model pertama, penulis ingin membuat model *Random Forest* dengan menggunakan parameter *default* dari *Random Forest Regressor*. Seperti diketahui bahwa jumlah pohon (*n_estimators*) yang akan dibangun berdasarkan hasil *default* yaitu sebanyak 10 pohon, sedangkan nilai banyaknya *feature random* yang akan dipilih untuk *split* (*max_features*) juga secara *default* adalah "*auto*" sebagai berikut:

```
from sklearn.ensemble import RandomForestRegressor

model = RandomForestRegressor()
model.fit(X_train, y_train)
```

**Gambar 5.24** Pembuatan Model *Random Forest* dengan Parameter *Default*

Selanjutnya melihat hasil akurasi dengan menggunakan model prediksi *Random Forest* pertama dengan parameter *default*. Hasil prediksi dengan model *Random Forest* secara *default*, maka menghasilkan nilai RMSE sebesar 3.773.812 dan hasil nilai akurasi dari model pertama *Random Forest* dengan parameter *default* adalah sebesar 75,75% yang ditampilkan pada Gambar 5.25 sebagai berikut:

```
from sklearn import metrics
y_pred = model.predict(X_test)
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
print('MAPE :', np.mean(np.abs((y_test - y_pred) / y_test))*100)
print ('Nilai akurasi :', (1-(np.mean(np.abs((y_test - y_pred) / y_test))))*100)
```

```
Mean Absolute Error: 1739144.0238793103
Mean Squared Error: 14241657890072.396
Root Mean Squared Error: 3773812.116424504
MAPE : 24.245680884010053
Nilai akurasi : 75.75431911598996
```

**Gambar 5.25** Hasil Model *Random Forest* dengan Parameter *Default*

Hasil evaluasi model prediksi *Random Forest* pertama dengan parameter *default* mendapatkan nilai akurasi sebesar 75,75%, sehingga masih dapat dioptimalkan dengan menggunakan metode *tuning hyperparameter* untuk mendapatkan parameter yang optimal.

### 5.5.1 *Tuning Hyperparameter*

*Tuning hyperparameter* adalah salah satu proses yang dilakukan untuk menemukan parameter yang optimal. Mengoptimalkan parameter untuk model *machine learning* adalah langkah solusi dalam membuat prediksi yang lebih baik dan akurat. *Hyperparameters* mendefinisikan karakteristik model yang dapat mempengaruhi akurasi model dan efisiensi komputasi.

Pada *Random Forest* terdapat dua parameter yang dapat disesuaikan dan dilakukan *tuning* yaitu jumlah pohon ($n_{tree}$) dan jumlah fitur acak untuk setiap *split* ($m_{try}$). Parameter yang akan dilakukan *tuning hyperparameter* pada penelitian ini adalah *n_estimators* ($n_{tree}$) dan *max_features* ($m_{try}$). Pendekatan metode untuk melakukan *tuning hyperparameter* terdapat dua metode antara lain: *grid search* dan *random search*. Dalam penelitian ini lebih mudah untuk menggunakan *random search* yang mengevaluasi hanya sejumlah kombinasi acak untuk setiap *hyperparameter* di setiap iterasi. Oleh karena itu, metode *random search* lebih efisien dalam mencari spesifikasi *hyperparameter* terbaik daripada menggunakan metode *grid search*.

Dengan menggunakan pendekatan metode *random search*, maka akan dicari parameter optimal antara lain: *n_estimators* ($n_{tree}$) dan *max_features* ($m_{try}$). Dalam menggunakan metode *RandomizedSearchCV*, maka diperlukan tahapan membuat *parameter grid* untuk mengambil sampel di masing-masing parameter yang dicari. Keuntungan dari *random search* adalah tidak perlu mencoba setiap kombinasi, tetapi dapat memilih dari hasil sampel yang didapatkan. Proses mencari sampel untuk parameter *n_estimators* dan *max_features* yang ditampilkan pada Gambar 5.26 sebagai berikut:

```
from sklearn.model_selection import RandomizedSearchCV
from pprint import pprint

# Number of trees in random forest (n_tree)
n_estimators = [int(x) for x in np.linspace(start = 100, stop = 1000, num = 10)]
#max_feature (m_try)
max_features = [1, 2, 3, 4]

# Create the random grid
random_grid = {'n_estimators': n_estimators,
               'max_features': max_features}

pprint(random_grid)
```

```
{'max_features': [1, 2, 3, 4],
 'n_estimators': [100, 200, 300, 400, 500, 600, 700, 800, 900, 1000]}
```

**Gambar 5.26** Hasil Pemilihan Sampel Parameter

Untuk mengevaluasi jumlah total kombinasi yang mungkin dari masing-masing kandidat sampel *hyperparameter*. Argumen yang penting dalam *RandomizedSearchCV* adalah *cv* yang merupakan jumlah lipatan yang digunakan untuk validasi silang. Dalam penelitian ini, menggunakan *cv k-fold* adalah 10. Dalam menggunakan fungsi *scikit-learn RandomizedSearchCV*, perlu menyimpan hasil nilai-nilai kandidat sampel *hyperparameter* untuk *n_estimators* dan *max_features* ke dalam variabel yang diberi nama "param_dist". Selanjutnya, *RandomizedSearchCV* menggunakan objek model dalam hal ini adalah model *random forest regressor*, kandidat *hyperparameter*, dan jumlah validasi silang yang ditampilkan pada Gambar 5.27 sebagai berikut:

```
# specify parameters and distributions to sample from
param_dist = {'max_features': [1, 2, 3, 4],
              'n_estimators': [100, 200, 300, 400, 500, 600, 700, 800, 900, 1000]}

random_search = RandomizedSearchCV(model,param_dist,cv=10)
random_search.fit(X_train, y_train)
```

**Gambar 5.27** *Tuning Hyperparameter*

Selanjutnya menggunakan perintah *random_grid.best_params_* untuk melihat nilai parameter terbaik dari hasil *tuning hyperparameter*. Menggunakan metode *RandomizedSearchCV* didapatkan hasil dari *tuning hyperparameter* bahwa nilai *n_estimators* optimal adalah 1.000 pohon dan nilai *max_features* optimal sebesar 4 variabel yang ditampilkan pada Gambar 5.28 sebagai berikut:

```
print(random_search.best_params_)
```

```
{'n_estimators': 1000, 'max_features': 4}
```

**Gambar 5.28** Parameter Optimal Hasil *Tuning hyperparameter*

Berdasarkan hasil *tuning hyperparameter* yang telah dilakukan, diketahui bahwa didapatkan *n_estimators* ($n_{tree}$) optimal sebesar 1.000 pohon dan *max_features* ($m_{try}$) optimal sebesar 4 variabel. Dengan menggunakan parameter $n_{tree}$ dan $m_{try}$ optimal yang akan diaplikasikan ke dalam pelatihan model *random forest regressor* baru dengan tujuan untuk memprediksi harga penjualan televisi. Hasil prediksi model *random forest regressor* percobaan kedua dengan menggunakan nilai parameter optimal hasil *tuning hyperparameter* yang ditampilkan pada Gambar 5.29 sebagai berikut:

```
from sklearn.ensemble import RandomForestRegressor

model2 = RandomForestRegressor(n_estimators= 1000, max_features= 4)
model2.fit(X_train, y_train)
```

```
RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion='mse',
                      max_depth=None, max_features=4, max_leaf_nodes=None,
                      max_samples=None, min_impurity_decrease=0.0,
                      min_impurity_split=None, min_samples_leaf=1,
                      min_samples_split=2, min_weight_fraction_leaf=0.0,
                      n_estimators=1000, n_jobs=None, oob_score=False,
                      random_state=None, verbose=0, warm_start=False)
```

**Gambar 5.29** Model *Random Forest* dengan Parameter Optimal

Hasil prediksi model dengan menggunakan parameter optimal hasil dari proses *tuning hyperparameter*, pada Gambar 5.30 diketahui bahwa didapatkan nilai RMSE sebesar 3.872.324. Sedangkan untuk hasil akurasi model kedua *random forest regressor* dengan menggunakan parameter optimal hasil dari proses *tuning hyperparameter* untuk memprediksi harga penjualan televisi sebesar 75,81% sebagai berikut:

```
from sklearn import metrics
y_pred = model2.predict(X_test)
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
print('MAPE :', np.mean(np.abs((y_test - y_pred) / y_test))*100)
print ('Nilai akurasi :', (1-(np.mean(np.abs((y_test - y_pred) / y_test))))*100)
```

```
Mean Absolute Error: 1733753.1585043103
Mean Squared Error: 14994900505453.293
Root Mean Squared Error: 3872324.9483292713
MAPE : 24.190034565932418
Nilai akurasi : 75.80996543406758
```

**Gambar 5.30** Hasil Evaluasi Model *Random Forest* dengan Parameter Optimal Hasil *Tuning hyperparameter*

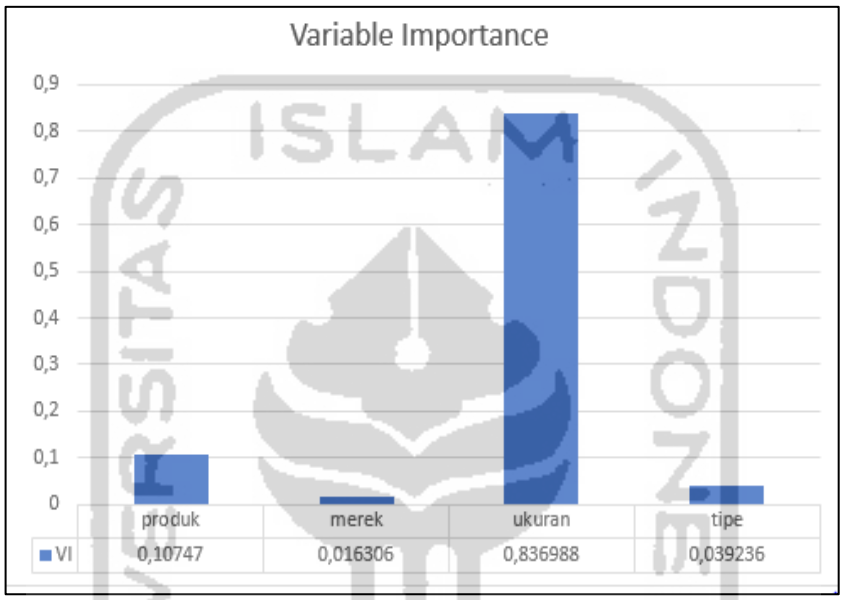
Di dalam *Python* terdapat sebuah modul yang berguna dapat menyimpan model yang disimpan di dalam memori yang diserialisasi ke dalam *file* berupa *pickle* dan dapat membaca obyek *Python* dari *file*. Model *random forest* yang telah dibuat dengan bahasa *python* akan melalui proses *serializing* untuk mengkonversi model *Random Forest* yang telah dilatih dan akan disimpan model hasil latih ke dalam *pickle* dengan nama "modelRF" dengan ekstensi `.pkl` dan disimpan di dalam *folder* yang digunakan untuk membangun *website*. Proses menyimpan model dengan *pickle* yang ditampilkan pada Gambar 5.31 sebagai berikut:

```
import pickle

# Saving model to disk
pickle.dump(model2, open('modelRF.pkl','wb'))
```

**Gambar 5.31** Penyimpanan Model Menggunakan *Pickle*

Selanjutnya untuk mengetahui variabel apa saja yang memberikan pengaruh terhadap variabel harga dengan menggunakan metode *Variable Importance* sebagai berikut:




**Gambar 5.32** *Variable Importance*

Berdasarkan hasil variabel *importance* pada Gambar 5.32 diketahui bahwa variabel ukuran memberikan pengaruh terbesar terhadap variabel harga penjualan televisi sebesar 0,836988. Berikutnya variabel produk di urutan kedua memberikan pengaruh sebesar 0,107470. Berikutnya variabel tipe di urutan ketiga memberikan pengaruh sebesar 0,039236. Selanjutnya di posisi terakhir terdapat variabel merek yang memberikan pengaruh sebesar 0.016306.

## 5.6 Tahapan Pembuatan *Website*

Selanjutnya memasuki tahapan pembuatan *website* dengan menggunakan *framework flask* dan bahasa pemrogramman *Python.* Flask adalah sebuah *web framework* dengan bahasa *python* yang menyediakan *library* dan kumpulan *code* yang dapat digunakan untuk membangun *website*. Untuk membuat *website* dengan menggunakan *Python* diperlukan *framework* pengembangan *web*. Pada *python framework* pengembangan *web* yang paling populer dan mudah dipelajari terdiri dari *Django* dan *Flask*. Sebagai pemula untuk membuat sebuah *website* dengan *python*, sebaiknya menggunakan *framework flask* karena dapat membuat *website* dengan lebih mudah. *Flask* adalah *framework* yang diterapkan untuk membuat *website* berskala kecil. Oleh karena itu, dalam penelitian ini diputuskan untuk menggunakan *framework flask* dengan bahasa pemrograman *Python* dalam mengaplikasikan model *machine learning* yang telah dilakukan pada tahapan sebelumnya menjadi sebuah *website* dan proses *deploy website* dengan menggunakan *Heroku*.

Dalam bahasa pemrograman *Python*, *package* seperti *Flask* tersedia di repositori publik sehingga semua orang dapat mengunduhnya. Dalam meng*install* sebuah *package* sangat mudah karena *Python* memiliki *tool* bernama *pip*. Oleh karena itu, dalam meng-*install flask* dapat menggunakan perintah `pip install flask`.

Dalam penelitian ini, tahapan untuk membangun sebuah model *machine learning* menjadi sebuah *website*, perlu dipersiapkan *file python* yang memuat model *Random Forest* yang telah dibuat sebelumnya dan disimpan dengan ekstensi `.pkl` yang diberi nama "`modelRF.pkl`".

Selanjutnya *file* yang diperlukan adalah *file HTML* yang berisi konten *website* yang akan dibangun. HTML menjadi bagian penting sebuah pondasi untuk membuat *website*. Elemen pada *HTML* membentuk sebuah struktur yang terdiri dari *tag* pembuka, konten, dan *tag* penutup. Dalam pembuatan *file html* dimulai dengan *tag* pembuka <html> dan diakhiri dengan *tag* penutup </html>. Dalam pembuatan *file HTML* menggunakan aplikasi editor teks yaitu *Notepad++*.

Dalam membuat *website* dibutuhkan *HTML* dan *CSS*. *HMTL* (*Hypertext Markup Language*) merupakan bahasa pemrograman yang menjadi standar untuk pembuatan struktur *website*. Sedangkan *CSS* (*Cascading Style Sheet*) digunakan untuk mendesain *layout* dari *website*. *File HTML* yang dibuat dalam penelitian ini diberi nama "`index.html`".

*Layout* dari *website* yang dibuat terdiri dari *header* dan *main Body*. Pada bagian *body* berisi konten yang ditampilkan pada *browser* seperti *form* untuk mengisi variabel produk, merek, tipe dan ukuran. *Form* untuk variabel produk, merek dan tipe menggunakan tampilan *drop down list* sedangkan untuk variabel ukuran menggunakan tampilan *input form* yang ditampilkan pada Gambar 5.33 sebagai berikut:

```
<!-- jenis produk -->
<div class="form-group">
  <label>Produk</label>
  <select name="produk" required class="form-control" id="produk_form_ex">
    <option value="" disabled selected>Produk</option>
  </select>
</div>

<!-- merek -->
<div class="form-group">
  <label for="exampleSelectGender">Merek</label>
  <select name="merek" required class="form-control" id="merek_form">
    <option value="" disabled selected>Merek</option>
  </select>
</div>

<!-- tipe -->
<div class="form-group">
  <label for="exampleSelectGender">Tipe</label>
  <select name="tipe" required class="form-control" id="tipe_form">
    <option value="" disabled selected>Tipe</option>
  </select>
</div>

<!-- ukuran -->
<div class="form-group">
  <label>Ukuran</label>
  <div class="input-group col-xs-12">
    <input name="ukuran" required type="text" id="ukuran_form" class="form-control"
      placeholder="Ukuran dalam Inch">
    <div class="input-group-append">
      <span class="input-group-text">Inch</span>
    </div>
  </div>
</div>
```

**Gambar 5.33** Pembuatan *Form HTML* Variabel Spesifikasi

Tampilan setelah *form* untuk mengisi empat variabel tersebut, terdapat *tag button* berupa *submit* dan *clear*. Tombol *submit* digunakan untuk memproses data yang di*input* untuk *HTML* mengirimkan nilainya agar diproses ke model prediksi untuk memperoleh harga televisi hasil prediksi. Sedangkan tombol *clear* digunakan untuk mengosongkan nilai pada *form*.

Selanjutnya hasil prediksi harga televisi akan muncul pada *form* harga berdasarkan variabel spesifikasi yang di*input*kan pada masing-masing *form*. Pembuatan kolom *form* untuk menampilkan harga yang ditampilkan pada Gambar 5.34 sebagai berikut:

```
  <button onclick="kirim_form()" class="btn btn-gradient-primary mr-2">Submit</button>
  <button onclick="clear_form()" class="btn btn-light">Clear</button>


</div>
<div class="card-body">

  <h4 class="card-title">Hasil Predict Harga Tv : <div id="loader" style="visibility: hidden"
      class="spinner-border text-primary"></div>
  </h4>


  <div class="form-group">
    <label>Harga</label>
    <div class="input-group col-xs-12">
      <div class="input-group-prepend">
        <span class="input-group-text bg-gradient-primary text-white">Rp. </span>
      </div>
      <input readonly type="text" id="hasil_predict" class="form-control"
        aria-label="Amount (to the nearest dollar)">
      <div class="input-group-append">
        <span class="input-group-text">.00</span>
      </div>
    </div>
  </div>
```

**Gambar 5.34** Pembuatan *Form HTML* Variabel Harga

Selanjutnya tahapan pembuatan *framework flask* dengan langkah pertama adalah melakukan inisialisasi *Flask API* dengan menggunakan "`app = Flask(__name__)`". Memuat ulang model dan variabel yang telah disimpan menggunakan *pickle* untuk memuat model dan *joblib* untuk memuat variabel. Mendefinisikan "`predict_harga`", dimana variabel data merupakan sebuah *data frame* yang menggabungkan empat variabel *independent* antara lain: produk, merek, tipe dan ukuran. Pada variabel harga merupakan hasil model prediksi dari data yang ditampilkan pada Gambar 5.31 sebagai berikut:

```
from flask import Flask, render_template, Response, jsonify, request
import os
import pandas as pd
import pickle
import joblib
import json

APP_ROOT = os.path.dirname(os.path.abspath(__file__))
app = Flask(__name__)

harga = 0
model = pickle.load(open('static/model/modelRF.pkl','rb'))
produk_map = joblib.load('static/var_cat/produk_map.pkl')
merek_map = joblib.load('static/var_cat/merek_map.pkl')
tipe_map = joblib.load('static/var_cat/tipe_map.pkl')

def predict_harga(produk, merek, tipe, ukuran):
    data = pd.DataFrame({"produk":[produk], "merek":[merek], "ukuran":[ukuran], "tipe":[tipe]})
    harga = model.predict(data)
    return harga
```

**Gambar 5.35** Pembuatan *Framework Flask*

Selanjutnya terdapat perintah dengan `@app.route("/")`, simbol @ merupakan sebuah *decorator*. Suatu pola umum dengan *decorator* adalah untuk menggunakannya sebagai *callback* di perintah tertentu. Pada `@app.route` pertama terdapat URL yang akan memanggil "/" sedangkan URL yang kedua akan memanggil "`index`" yang masing-masing terhubung dengan fungsi di bawahnya. Sehingga setiap mengakses dari URL tersebut, *Flask* akan memanggil fungsi di bawahnya dan mengirimkan *response* (perintah *return*) ke fungsi `render_template()`. Operasi yang mengubah sebuah *template* menjadi halaman *HTML* disebut *rendering*. Fungsi ini meminta nama *file template* dan daftar variabel untuk diisi ke dalam *placeholder* yang ada di dalam *template* tersebut yaitu ke dalam *index.html*. Oleh karena itu, setiap mengakses alamat *URL* web *application* prediksi harga televisi ini akan diarahkan ke halaman tampilan utama *website* dan akan dapat menginput data spesifikasi televisi pada *form* masing-masing variabel.

Selanjutnya untuk mendefinisikan fungsi prediksi untuk melakukan proses prediksi harga, sehingga akan mengirimkan *URL* ke halaman *predict* `@app.route('/predict')`. Dalam proses menginput spesifikasi televisi terdapat proses permintaan (*request*) dari *form* dari data yang telah di*input*kan. Metode *request* dengan menggunakan `metode= ['POST']`, sehingga *input* data spesifikasi berdasarkan nama produk, merek, tipe dan ukuran akan merespon permintaan dan memberikan hasil berupa prediksi harga televisi. Selanjutnya *flask* disimpan dengan nama "app.py" yang ditampilkan pada Gambar 5.36 sebagai berikut:

```
@app.route('/')
def index():
    return render_template('index.html')

@app.route('/predict', methods = ['GET','POST'])
def predict():
    def pred():
        if request.method == 'POST':
            produk = request.form['produk']
            merek = request.form['merek']
            tipe = request.form['tipe']
            ukuran = request.form['ukuran']
            global harga
            harga = predict_harga(produk, merek, tipe, ukuran)[0]
        print(harga)
        return "data:"+str(harga)+"\n\n"

    return Response(pred(), mimetype='text/event-stream')
```

**Gambar 5.36** Pembuatan *Flask* Prediksi

## 5.7 *Deploy Website* Menggunakan Heroku

Heroku adalah sebuah *cloud platform* yang menjalankan bahasa pemrograman tertentu seperti Phyton, Ruby, Java, PHP, Node.js dan lain-lain. Heroku adalah salah satu *web hosting* berbasis *cloud* yang merupakan contoh dari komputasi modern. *Cloud computing* ini berfungsi dalam hal *deployment* sebuah aplikasi *web*. Heroku termasuk ke dalam salah satu model layanan *cloud computing* yaitu *Platform As A Service* (PaaS), sehingga dalam melakukan *deploy* aplikasi *web* ke Heroku cukup dengan melakukan konfigurasi aplikasi yang ingin di*deploy*. Heroku dapat melakukan *deploy* secara otomatis terhadap aplikasi *web* yang di-*upload* menggunakan *git*.

Dalam tahapan *deploying web* menggunakan heroku, terdapat beberapa yang perlu dipersiapkan antara lain: meng*install git* karena *app* akan terhubung dengan Heroku, meng*install* Heroku CLI dan membuat akun terlebih dahulu pada *website* Heroku. Setelah pendukung untuk melakukan *deploy* ke Heroku selesai, maka tahapan meng*deploy* web *application* ke Heroku sebagai berikut:

1. Sambungkan dengan terminal *cmd, login* ke Heroku dengan menggunakan perintah "`heroku login`" seperti pada Gambar 5.37 sebagai berikut:

```
C:\SkripsiTV>heroku login
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/cli/browser/6a83e7e3-114f-46e8-9ddb-23678fdb833b
Logging in... done
Logged in as rismitawahyu@gmail.com
```

**Gambar 5.37** Login Ke Heroku

2. Secara otomatis akan masuk ke URL Heroku, kemudian akan muncul tampilan "*Log In to the Heroku CLI*" dan klik *button log in.* Selanjutnya pada Gambar 5.38 akan terlihat bahwa telah melakukan *log in* ke akun heroku yang terdaftar di Heroku sebagai berikut:

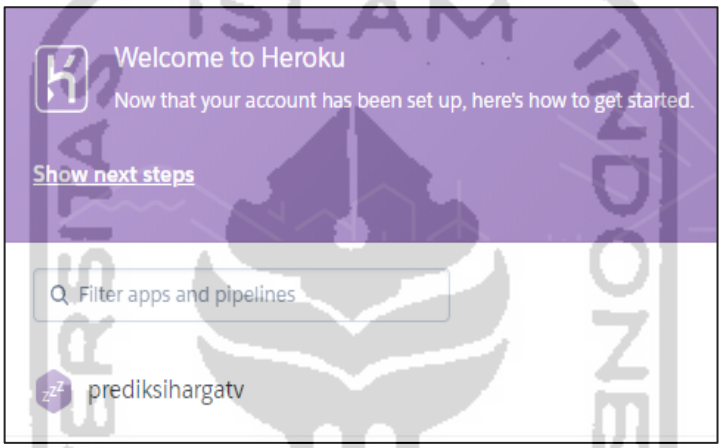

**Gambar 5.38** Tampilan *Log In to the Heroku CLI*

3. Selanjutnya membuat nama aplikasi dengan perintah "`heroku create nama-app`". Nama web *application* yang dibuat adalah "prediksihargatv", nama aplikasi tersebut akan menjadi *subdomain* dari URL *website* yang dibuat di Heroku seperti pada Gambar 5.39 sebagai berikut:

```
C:\SkripsiTV>heroku create prediksihargatv --buildpack heroku/python
Creating ⬚ prediksihargatv... done
Setting buildpack to heroku/python... done
https://prediksihargatv.herokuapp.com/ | https://git.heroku.com/prediksihargatv.git
```

**Gambar 5.39** Pembuatan Nama Aplikasi

4. Jika berhasil nama aplikasi telah dibentuk, maka akan muncul di dalam akun Heroku seperti pada Gambar 5.40 sebagai berikut:




**Gambar 5.40** Tampilan *Dashboard* Heroku

5. Selanjutnya menghubungkan *git* dengan Heroku untuk penghubung *web app* yang dibuat dengan *heroku cloud* dengan menjalankan perintah "`git init`" untuk menginisiasi repositori *git* seperti pada Gambar 5.41 sebagai berikut:

```
C:\SkripsiTV>heroku git init
manage local git repository for app

USAGE
  $ heroku git:COMMAND

COMMANDS
  git:clone   clones a heroku app to your local machine at DIRECTORY (defaults to app name)
  git:remote  adds a git remote to an app repo
```

**Gambar 5.41** Proses *Git Init*

6. Dalam melakukan *push app* ke Heroku, dipastikan terlebih dahulu nama *remote* adalah Heroku dengan menggunakan "`git remote`" seperti pada Gambar 5.42 sebagai berikut:

```
C:\SkripsiTV>heroku git:remote -a prediksihargatv
set git remote heroku to https://git.heroku.com/prediksihargatv.git
```

**Gambar 5.42** Proses *Git Remote*

7. Dalam men*deploy* aplikasi *web*nya, diperlukan untuk menambahkan *file* lokal aplikasinya menggunakan perintah "`git add .`" seperti pada Gambar 5.43 sebagai berikut:

```
C:\SkripsiTV>git add .
warning: LF will be replaced by CRLF in static/assets/css/style.css.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in static/assets/images/dashboard/circle.svg.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in static/assets/images/logo-mini.svg.
```

**Gambar 5.43** Proses *Git Add*

8. Selanjutnya untuk merekam terjadinya perubahan yang terjadi dapat menggunakan perintah `git commit -m "komentar bebas"`. Sehingga setiap terdapat perubahan, selalu melakukan add dan commit dengan komentar bebas yang sama, dalam penelitian ini menggunakan komentar bebasnya adalah "rismita" seperti pada Gambar 5.44 sebagai berikut:

```
C:\SkripsiTV>git commit -am "rismita"
[master (root-commit) b9b13d3] rismita
 145 files changed, 41412 insertions(+)
 create mode 100644 .vscode/settings.json
 create mode 100644 Procfile
 create mode 100644 app.py
 create mode 100644 requirements.txt
 create mode 100644 static/assets/css/style.css
 create mode 100644 static/assets/css/style.css.map
 create mode 100644 static/assets/fonts/Ubuntu/Ubuntu-Bold.eot
 create mode 100644 static/assets/fonts/Ubuntu/Ubuntu-Bold.ttf
 create mode 100644 static/assets/fonts/Ubuntu/Ubuntu-Bold.woff
 create mode 100644 static/assets/fonts/Ubuntu/Ubuntu-Bold.woff2
 create mode 100644 static/assets/fonts/Ubuntu/Ubuntu-Light.eot
 create mode 100644 static/assets/fonts/Ubuntu/Ubuntu-Light.ttf
 create mode 100644 static/assets/fonts/Ubuntu/Ubuntu-Light.woff
 create mode 100644 static/assets/fonts/Ubuntu/Ubuntu-Light.woff2
```

**Gambar 5.44** Proses *Git Commit*

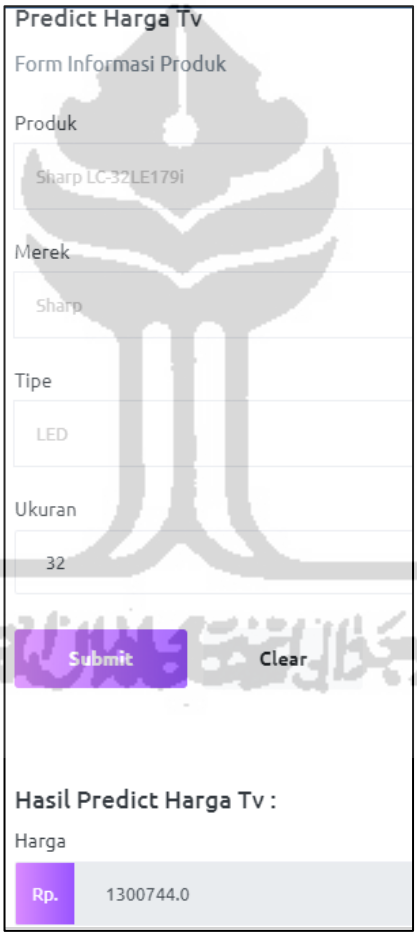
9. Selanjutnya tahapan terakhir adalah proses *deploy* ke Heroku dengan mem-*push* aplikasi *web* ke master Heroku dengan perintah "`git push heroku master`" seperti pada Gambar 5.45 sebagai berikut:

```
C:\SkripsiTV>git push heroku master
Enumerating objects: 151, done.
Counting objects: 100% (151/151), done.
Delta compression using up to 8 threads
Compressing objects: 100% (142/142), done.
Writing objects: 100% (151/151), 10.34 MiB | 487.00 KiB/s, done.
Total 151 (delta 7), reused 0 (delta 0), pack-reused 0
remote: Compressing source files... done.
```

**Gambar 5.45** Proses *Git Push Heroku Master*

Setelah melakukan langkah-langkah dalam proses *deploy* aplikasi *web* prediksi harga televisi yang telah dirancang dengan menggunakan Heroku,

maka dapat membuka *website* tersebut dengan mengakses alamat URL https://prediksihargatv.herokuapp.com/. Sehingga akan muncul tampilan halaman utama dari web *application* prediksi harga televisi yang telah dirancang. Selanjutnya dapat mlakukan uji coba pada *website* yang telah dibuat dengan mengisi *form* spesifikasi televisi antara lain: pada kolom produk memilih nama televisi Sharp LC-32LE179i, pada kolom merek memilih *Sharp*, pada kolom tipe memilih tipe layar LED dan pada kolom ukuran mengetikan ukuran layar sebesar 32 *inch* dan menekan *button submit*, maka akan muncul harga hasil prediksi televisi dengan spesifikasi yang telah di*input* sebelumnya yaitu sebesar Rp 1.300.744 yang ditampilkan pada Gambar 5.46 sebagai berikut:




**Gambar 5.46** Hasil Prediksi Harga Penjualan Televisi

# BAB VI
# KESIMPULAN

## 6.1 Kesimpulan

Berdasarkan hasil analisis dengan menggunakan *random forest* untuk memprediksi harga penjualan televisi yang telah dilakukan, maka dapat ditarik kesimpulan sebagai berikut:

1. Berdasarkan data yang diperoleh dengan melakukan proses *scraping web* pada *website* Pricebook, maka didapatkan *dataset* yang terdiri dari variabel nama produk, merek, ukuran layar, tipe layar dan harga dengan total keseluruhan data sebanyak 925 data.
2. Dalam analisis *random forest* menggunakan data *training* dan data *testing* dengan perbandingan 75%:25%. Dilakukan percobaan pertama dengan membuat model *random forest regressor* menggunakan parameter *default* menghasilkan akurasi sebesar 75,75%. Dalam meningkatkan akurasi dilakukan *tuning hyperparameter* untuk mendapatkan parameter optimal antara lain: *n_estimators* ($n_{tree}$) optimal sebanyak 1.000 pohon dan parameter *max_features* ($m_{try}$) optimal sebanyak 4 variabel. Dilakukan pelatihan kedua dengan membuat model *random forest regressor* menggunakan parameter optimal hasil *tuning hyperparameter* sehingga didapatkan akurasi sebesar 75,81%.
3. Pada *variable importance* yang mempengaruhi dalam menentukan harga penjualan televisi berdasarkan urutan variabel yang memberikan pengaruh terbesar hingga terkecil sebagai berikut: ukuran, produk, tipe dan merek.
4. *Website application* yang telah berhasil dibuat dengan *framework flask* dan di*deploy* dengan menggunakan Heroku yang dapat diakses dengan menggunkan alamat URL sebagai berikut: https://prediksihargatv.herokuapp.com/.

## 6.2 Saran

Adapun saran yang dapat diberikan berdasarkan hasil penelitian saat ini untuk penelitian selanjutnya adalah:

1. Penelitian selanjutnya diharapkan menggunakan perbandingan dengan metode klasifikasi lainnya.
2. Penelitian selanjutnya diharapkan perlunya penggunaan *treatment* untuk mengatasi data *imbalanced*.
3. Penelitian selanjutnya diharapkan dapat memperluas dalam menggunakan parameter saat melakukan proses *tuning hyperparameter*.
4. Penelitian selanjutnya diharapkan dapat juga diaplikasikan untuk memprediksi harga televisi bekas.
5. Penelitian selanjutnya diharapkan terdapat tambahan fitur filter pada tampilan *website*.

# LAMPIRAN

**Lampiran 1** *Dataset* Penelitian

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| CooCaa 40S5G | CooCaa | 40 | OLED | 2225000 |
| CooCaa 32S5C | CooCaa | 32 | OLED | 1870000 |
| LG 43UM7100 | LG | 43 | LED | 4490000 |
| Sharp AQUOS LC-32LE180i | Sharp | 32 | LED | 1648000 |
| Polytron PLD 20D901 | Polytron | 20 | LED | 1090000 |
| Toshiba 40PU200EJ | Toshiba | 40 | LED | 4350000 |
| Sharp AQUOS LC-24LE175i | Sharp | 24 | LED | 1169000 |
| Polytron PLD 43TV865 | Polytron | 43 | LED | 4250000 |
| Polytron Cinemax Soundbar PLD-32B1550 | Polytron | 32 | LED | 1998000 |
| Sharp LC-32SA4100 | Sharp | 32 | LED | 1670000 |
| Sharp AQUOS LC-24LE170i | Sharp | 24 | LED | 821300 |
| Sharp 2TC45AD1X | Sharp | 45 | LED | 3699000 |
| Sharp AQUOS LC-32LE107 | Sharp | 32 | LED | 2600000 |
| Sharp AQUOS LC-32LE265I | Sharp | 32 | LED | 2099333 |
| Samsung UA75NU800 | Samsung | 75 | OLED | 9575000 |
| Polytron PLD 32D905 | Polytron | 32 | LED | 2750000 |
| Samsung UA43M5100 | Samsung | 43 | LED | 3117000 |
| Polytron Cinemax Soundbar PLD-40B150 | Polytron | 39 | LED | 2685000 |
| Xiaomi Mi TV 4A 32 inch | Xiaomi | 32 | LED | 1837000 |
| Polytron Cinemax Soundbar PLD-43B150 | Polytron | 43 | LED | 3159000 |
| Polytron PLD 24T8511 | Polytron | 24 | LED | 1299000 |
| Samsung 32FH4003R | Samsung | 32 | LED | 1782000 |
| Polytron PLD 40T100 | Polytron | 40 | LED | 4099000 |
| TCL 32B3 | TCL | 32 | LED | 1450000 |
| Samsung UA32J4003 | Samsung | 32 | LED | 1830000 |
| LG 43LM5500 | LG | 43 | LED | 3495000 |
| Sharp AQUOS LC-24SA4000i | Sharp | 24 | LED | 1099000 |
| LG 32LJ500D | LG | 32 | LED | 1849000 |
| Polytron PLD 24D123 | Polytron | 24 | LED | 1349000 |
| TCL 32A3 | TCL | 32 | LED | 1819000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| Sharp 2T-C32BA1i | Sharp | 32 | LED | 1640000 |
| Samsung 43N5500 | Samsung | 43 | LED | 4235000 |
| Panasonic TH-32G302G | Panasonic | 32 | LED | 1675000 |
| CooCaa 40D3A | CooCaa | 40 | LED | 1995000 |
| AQUA LE32AQT6100 | AQUA | 32 | LED | 1449000 |
| LG 43UK6300 | LG | 43 | LED | 4799000 |
| LG 32LM550 | LG | 32 | LED | 1790000 |
| Panasonic TH-32F302G | Panasonic | 32 | LED | 1650000 |
| Polytron PLD 32T1550 | Polytron | 32 | LED | 1849900 |
| LG 32LH510D | LG | 32 | LED | 2199000 |
| Samsung UA32N4300 | Samsung | 32 | LED | 1900000 |
| Samsung UA43N5003 | Samsung | 43 | LED | 3124999 |
| Sharp LC-32SA4200i | Sharp | 32 | LED | 1850000 |
| Samsung UA32N4001 | Samsung | 32 | LED | 1798000 |
| Sharp AQUOS LC-32LE185i | Sharp | 32 | LED | 1969000 |
| LG 22LN4000 | LG | 22 | LED | 1950000 |
| AQUA LE43AQT6000 | AQUA | 43 | LED | 2700000 |
| TCL 40A3 | TCL | 40 | LED | 2613000 |
| Samsung UA32J4303 | Samsung | 32 | LED | 2345000 |
| CooCaa 50S5G | CooCaa | 50 | OLED | 4499000 |
| Polytron PLD 32T1500 | Polytron | 32 | LED | 1831500 |
| Samsung UA40N5000 | Samsung | 40 | LED | 2347000 |
| LG 32LM630BPTB | LG | 32 | LED | 2325000 |
| Polytron 43AS1558 | Polytron | 43 | LED | 3650000 |
| Samsung UA24H4150 | Samsung | 24 | LED | 1169000 |
| Toshiba 32L5650 | Toshiba | 32 | LED | 2318000 |
| Samsung 43K5002 | Samsung | 43 | LED | 3899000 |
| CooCaa 24W1900 | CooCaa | 24 | LED | 1100000 |
| Samsung UA43J5100 | Samsung | 43 | LED | 3159000 |
| LG 43LK5000 | LG | 43 | LED | 2950000 |
| Philips 32PHT4002S | Philips | 32 | LED | 1695000 |
| CooCaa 32E20W | CooCaa | 32 | LED | 1900000 |
| Polytron PLD 32V7510 | Polytron | 32 | LED | 2331900 |
| Polytron PLD 22D900 | Polytron | 22 | LED | 1125000 |
| Samsung QLED Q7F QA65Q7FNAKPXD | Samsung | 65 | QLED | 20400000 |
| LG 22MT47AC | LG | 22 | LED | 1715000 |
| Samsung UA32J4005 | Samsung | 32 | LED | 1885000 |
| TCL 32S62 | TCL | 32 | LED | 1750000 |
| CHANGHONG L32H4 | CHANGHONG | 32 | LED | 1850000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| Polytron 32T7511 | Polytron | 32 | LED | 1799000 |
| Sharp LC-40SA5100 | Sharp | 40 | LED | 2584000 |
| Samsung UA58RU7100KPXD | Samsung | 58 | LED | 7999000 |
| LG 24TL520A | LG | 24 | LED | 1214000 |
| Sony KDL-32R300B | Sony | 32 | LCD | 2860000 |
| LG 42LB550A | LG | 42 | LED | 5250000 |
| Polytron PLD 32D1500 | Polytron | 32 | LED | 1734000 |
| Toshiba 32L1600 | Toshiba | 32 | LED | 1659000 |
| CHANGHONG L32G3 | CHANGHONG | 32 | LED | 1374900 |
| Sharp AQUOS LC-40LE185i | Sharp | 40 | LED | 2224900 |
| LG 42LF550A | LG | 42 | LED | 4990000 |
| Samsung UA50MU6100 | Samsung | 50 | QLED | 6498000 |
| LG 43LF630T | LG | 43 | LED | 3255000 |
| Sharp AQUOS LC-23LE100M | Sharp | 23 | LED | 1100000 |
| Panasonic TH-32E305G | Panasonic | 32 | LED | 1799000 |
| Samsung 43j5202 | Samsung | 43 | LED | 3800000 |
| Panasonic TH-32E302G | Panasonic | 32 | LED | 1699990 |
| Polytron PLD 32D710 | Polytron | 32 | LED | 1849000 |
| LG 32LH500D | LG | 32 | LED | 2599000 |
| CooCaa 32A4 | CooCaa | 32 | LED | 1424900 |
| Samsung UA20J4003 | Samsung | 20 | LED | 1650000 |
| Sharp LC-32LE260I | Sharp | 32 | LED | 1709000 |
| Sharp AQUOS LC-19LE150M | Sharp | 19 | LED | 1450000 |
| LG 43LM5500PTA | LG | 43 | LED | 2995000 |
| LG 55UK6300 | LG | 55 | OLED | 6499000 |
| TCL 32D2900 | TCL | 32 | LED | 1420000 |
| LG OLED65C9 | LG | 65 | OLED | 46000000 |
| Panasonic TH-32A402G | Panasonic | 32 | LED | 421000 |
| Sony KDL-48W650D | Sony | 48 | LED | 5350000 |
| Sharp AQUOS LC-32DX888I | Sharp | 32 | LED | 3250000 |
| Polytron PLD 32T710 | Polytron | 32 | LED | 1925000 |
| Polytron PLD 24D9501 | Polytron | 24 | LED | 1190000 |
| Sharp LC-32LE179i | Sharp | 32 | LED | 1690000 |
| CooCaa 50UB5100 | CooCaa | 50 | LED | 3854800 |
| Samsung UA32F5105AR | Samsung | 32 | LED | 3050000 |
| Toshiba 24L1600 | Toshiba | 24 | LED | 1160000 |
| Samsung UA24H4053 | Samsung | 24 | LED | 2100000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| Polytron PLD 24D901 | Polytron | 24 | LED | 1325000 |
| Toshiba 32L3750 | Toshiba | 32 | LED | 1725000 |
| Polytron PLD 20D900 | Polytron | 20 | LED | 1289900 |
| Panasonic TH-32F305G | Panasonic | 32 | LED | 1800000 |
| Xiaomi Mi TV 4A | Xiaomi | 32 | LED | 2099000 |
| Polytron PLD 29D700 | Polytron | 29 | LED | 2450000 |
| LG 24MT48AF-PT | LG | 24 | LED | 1248000 |
| LG 32LM570BPTC | LG | 32 | LED | 2150000 |
| LG 43LH500T | LG | 43 | LED | 5799000 |
| LG 42LN5100 | LG | 42 | LED | 5450000 |
| TCL 40S4900 | TCL | 40 | LED | 2150000 |
| LG 43UM7300 | LG | 43 | LED | 4788000 |
| CooCaa 24D3A | CooCaa | 24 | LED | 989000 |
| Panasonic TH-24F305G | Panasonic | 24 | LED | 1199000 |
| LG 49UJ652T | LG | 49 | QLED | 6499000 |
| Panasonic TH-43E302G | Panasonic | 43 | LED | 2049000 |
| LG 32LK500BPTA | LG | 32 | LED | 1744000 |
| ICHIKO S1998 | ICHIKO | 19 | LED | 849000 |
| Philips 43PFT5250 | Philips | 43 | LED | 3099000 |
| Polytron PLD 40TS153 | Polytron | 40 | LED | 2799000 |
| CHANGHONG 32E6000 | CHANGHONG | 32 | LED | 1450000 |
| AKARI LE-32V90 | AKARI | 32 | LED | 1536500 |
| AKARI LE-55D88 | AKARI | 55 | LED | 5300000 |
| LG 50UK6300PTE | LG | 50 | LED | 5799000 |
| AQUA LE32AQT6000 | AQUA | 32 | LED | 1549000 |
| Sharp AQUOS LC-32LE347I | Sharp | 32 | LED | 650000 |
| LG 20MK400A-B | LG | 20 | LED | 899000 |
| LG 24TK425A | LG | 24 | LED | 1327000 |
| Samsung UA32N4003AKPXD | Samsung | 32 | LED | 1798000 |
| AKARI LE-32K88 | AKARI | 32 | LED | 1524000 |
| LG 20MT45A | LG | 20 | LED | 1475000 |
| Sony KDL-50W800C | Sony | 50 | LED | 5345000 |
| LG 22TK420A | LG | 22 | LED | 1200000 |
| LG 20MT48AF-PT | LG | 20 | LED | 1099000 |
| Sony KDL-55W800C | Sony | 55 | LED | 10500000 |
| LG 29MT45A | LG | 29 | LED | 2489000 |
| Panasonic TH-22D305G | Panasonic | 22 | LED | 1170000 |
| LG 43LJ500T | LG | 43 | QLED | 3485000 |
| Sharp LC-40LE265M | Sharp | 40 | LED | 2900000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| AKARI LE-50D88 | AKARI | 50 | LED | 4493000 |
| LG 65UM7600 | LG | 65 | LED | 13860000 |
| TCL 32D3000 | TCL | 32 | LED | 990000 |
| Polytron PLD 32T100 | Polytron | 32 | LED | 2100000 |
| Toshiba 24L2600 | Toshiba | 24 | LED | 1249000 |
| Polytron PC52UV222 | Polytron | 21 | LCD | 1459000 |
| Sharp AQUOS LC-32LE295i | Sharp | 32 | LED | 2350000 |
| Mito 4231 | Mito | 43 | LED | 2999000 |
| Sharp LC-40SA5500i | Sharp | 40 | LED | 3699000 |
| Philips 32PHA4100S | Philips | 32 | LED | 1639000 |
| CooCaa 40E6 | CooCaa | 40 | LED | 3099000 |
| CooCaa 39E20W | CooCaa | 39 | LED | 3049000 |
| CooCaa 55G7200 | CooCaa | 55 | LED | 5999000 |
| Polytron PLD 24T810 | Polytron | 23 | LED | 1348000 |
| Samsung UA60J6200 | Samsung | 60 | LED | 18000000 |
| LG 32LF520A | LG | 32 | LED | 2385000 |
| Samsung UA50RU7400KPXD | Samsung | 50 | LED | 7300000 |
| Toshiba 40L1600 | Toshiba | 40 | LED | 2899000 |
| Polytron PLD 24D8511 | Polytron | 24 | LED | 1245000 |
| AKARI LE-29P57 | AKARI | 29 | LED | 1494000 |
| LG 32LF550A | LG | 32 | LED | 1792000 |
| LG 39LN5100 | LG | 39 | LED | 4750000 |
| Sharp LC-50SA5200X | Sharp | 50 | LED | 5100000 |
| Toshiba 40L5650 | Toshiba | 40 | LED | 3385000 |
| LG 43LF540T | LG | 43 | LED | 3400000 |
| Sharp 2T-C32BA2i | Sharp | 32 | LED | 1668000 |
| Polytron PLD 22D901 | Polytron | 22 | LED | 1299000 |
| CooCaa 32W4 | CooCaa | 32 | LED | 1485000 |
| LG 32LJ550D | LG | 32 | LED | 2345000 |
| Sony KD-65X7000G | Sony | 65 | LED | 12290000 |
| IKEDO LT-22T1U | IKEDO | 22 | LED | 1200000 |
| Toshiba 43L3750 | Toshiba | 43 | LED | 3370000 |
| CooCaa 32E3000T | CooCaa | 32 | LED | 1930000 |
| LG 24TL520 | LG | 24 | LED | 1080000 |
| LG 49UK6300 | LG | 49 | LED | 5798000 |
| Sharp LC-32SA4102I | Sharp | 32 | LED | 1735000 |
| SANYO LE40S7000 | SANYO | 49 | LED | 3198000 |
| Polytron PLD 40D100 | Polytron | 40 | LED | 2649000 |
| Samsung UA46F5000AM | Samsung | 46 | LED | 10850000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| LG OLED55C9 | LG | 55 | OLED | 26699000 |
| Polytron PLD 32TS1503 | Polytron | 32 | LED | 1799000 |
| Panasonic TH-43C305G | Panasonic | 43 | LED | 3490000 |
| Polytron PLD 50T555 | Polytron | 50 | LED | 4899000 |
| Samsung UA23H4003 | Samsung | 23 | LED | 2050000 |
| Samsung UA40EH5000 | Samsung | 40 | LED | 2848000 |
| AQUA LE40AQT6900 | AQUA | 40 | LED | 2250000 |
| Panasonic TH-22E302G | Panasonic | 22 | LED | 990000 |
| AKARI LE-43D99SBS | AKARI | 43 | LED | 3498000 |
| Panasonic TH-43F302G | Panasonic | 43 | LED | 3019900 |
| Toshiba 40L3750 | Toshiba | 40 | LED | 2920000 |
| LG OLED77C9 | LG | 77 | OLED | 99999000 |
| Mito A101 | Mito | 24 | LED | 1320000 |
| Panasonic TH-40F305G | Panasonic | 40 | LED | 2699000 |
| Panasonic TH-49D305G | Panasonic | 49 | LED | 5600000 |
| Samsung UA32D5000 | Samsung | 32 | LED | 3350000 |
| LG 19LN4050 | LG | 19 | LED | 1450000 |
| ICHIKO S3258 | ICHIKO | 32 | LED | 1499000 |
| AKARI LE-25B88 | AKARI | 24 | LED | 1389800 |
| TCL 32S4900 | TCL | 32 | LED | 2199000 |
| LG 55UM7100 | LG | 55 | LED | 7550000 |
| LG 42LW5700 | LG | 42 | LED | 3440000 |
| Sony KD-65X7000E | Sony | 65 | LED | 12520000 |
| CooCaa 40E39 | CooCaa | 40 | LED | 3399000 |
| Hisense L32D50 | Hisense | 32 | LED | 1660000 |
| Polytron PLD 32T711 | Polytron | 32 | LED | 300000 |
| Toshiba 32P1400 | Toshiba | 32 | LED | 2500000 |
| Polytron PLD-22D1150 | Polytron | 22 | LED | 1198000 |
| Polytron PLD 24D810 | Polytron | 24 | LED | 1375000 |
| LG 49LF540T | LG | 49 | LED | 9850000 |
| LG 65SK8500 | LG | 65 | LED | 14599000 |
| Polytron PLD 24D900 | Polytron | 24 | LED | 1370000 |
| Polytron PLD 20D9501 | Polytron | 20 | LED | 1010000 |
| Toshiba 24P2300 | Toshiba | 24 | LED | 1750000 |
| CooCaa 32E20 | CooCaa | 32 | LED | 1615000 |
| Toshiba 23PU200 | Toshiba | 23 | LED | 2250000 |
| Sharp AQUOS LC-32LE260M | Sharp | 32 | LED | 2400000 |
| LG 28MT49VF-PT | LG | 28 | LED | 1799000 |
| TCL 29D2900 | TCL | 29 | LED | 1137000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| Samsung QLED Q7F QA75Q7FNAKPXD | Samsung | 75 | QLED | 40000000 |
| Panasonic TH-55E306G | Panasonic | 55 | LED | 5000000 |
| Polytron PLD 43V863 | Polytron | 40 | LED | 4075000 |
| Polytron PLD 32D900 | Polytron | 32 | LED | 3250000 |
| CooCaa 40E2000T | CooCaa | 40 | LED | 2199000 |
| Mito 3218 | Mito | 32 | LED | 1699000 |
| Panasonic TH-32F306G | Panasonic | 32 | LED | 1749000 |
| Sharp AQUOS LC-50UE630X | Sharp | 50 | LED | 8450000 |
| LG 42LB561T | LG | 42 | LED | 6250000 |
| Sharp LC-24SA4100 | Sharp | 24 | LED | 1198000 |
| Polytron PLD 50B880 | Polytron | 50 | LED | 4275000 |
| CooCaa 32E2000T | CooCaa | 32 | LED | 2300000 |
| LG 43LM550 | LG | 43 | LED | 2900000 |
| LG 50LN5400 | LG | 50 | LED | 11750000 |
| LG 43LH511T | LG | 43 | LED | 4150000 |
| Sanken SLE-323HDB | Sanken | 32 | LED | 2349000 |
| CHANGHONG 32G4A | CHANGHONG | 32 | LED | 1449000 |
| Polytron PLD 32D700 | Polytron | 32 | LED | 2149000 |
| Samsung UA40MU6103 | Samsung | 40 | OLED | 5750000 |
| CHANGHONG L40G5i | CHANGHONG | 40 | LED | 3897700 |
| Samsung UA48JU6600 | Samsung | 48 | LED | 12000000 |
| Polytron PLD 24T810W | Polytron | 24 | LED | 1390000 |
| CHANGHONG 32D2200 | CHANGHONG | 32 | LED | 2450000 |
| Samsung UA40M5050AK | Samsung | 40 | LED | 3250000 |
| Toshiba 43L5650 | Toshiba | 43 | LED | 3983000 |
| Sharp LC-50UA330X | Sharp | 50 | LED | 9950000 |
| Toshiba 40L2400VJ | Toshiba | 40 | LED | 3599000 |
| CHANGHONG 19D1000 | CHANGHONG | 19 | LED | 1045900 |
| Panasonic TH-24E305G | Panasonic | 24 | LED | 1250000 |
| Sharp AQUOS LC-46LE840X | Sharp | 46 | LED | 7699000 |
| LG 49LJ510T | LG | 49 | QLED | 4299000 |
| LG 65UK6540PTD | LG | 65 | OLED | 12499000 |
| AQUA LE32AQT9100T | AQUA | 32 | LED | 2429100 |
| Sharp AQUOS LC-40LE355M | Sharp | 40 | LED | 3749000 |
| AKARI LE-40D88 | AKARI | 40 | LED | 2549000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| LG 43UJ632T | LG | 43 | LED | 4500000 |
| Panasonic TH-43E306G | Panasonic | 43 | LED | 3388000 |
| LG 42UB820T | LG | 42 | LED | 8950000 |
| Samsung 55NU7300 | Samsung | 55 | LED | 8300000 |
| LG 32LK540BPTA | LG | 32 | LED | 2229000 |
| Panasonic TH-32D302G | Panasonic | 32 | LED | 2099000 |
| Panasonic TH-43D306G | Panasonic | 43 | LED | 4499000 |
| Samsung UA48J5100 | Samsung | 48 | LED | 6300000 |
| CHANGHONG 24D1000 | CHANGHONG | 24 | LED | 1598800 |
| LG 24MT47A | LG | 24 | LED | 1590000 |
| Samsung S24D300HL | Samsung | 24 | LED | 1400000 |
| Sharp AQUOS LC-32LE150M | Sharp | 32 | LED | 2600000 |
| AKARI LE-23B88 | AKARI | 23 | LED | 1489000 |
| LG 55SM8100 | LG | 55 | LED | 10000000 |
| CHANGHONG 50D2200 | CHANGHONG | 50 | LED | 6899000 |
| LG 28MT47A | LG | 28 | LED | 2000000 |
| Samsung UA40H5003 | Samsung | 40 | LED | 4499000 |
| CooCaa 32S3A12G | CooCaa | 32 | LED | 2299000 |
| AKARI LE-32P88 | AKARI | 32 | LED | 1450000 |
| Philips 24PHA4100S | Philips | 24 | LED | 1479333 |
| Sharp AQUOS LC-32DX288I | Sharp | 32 | LED | 2850000 |
| LG 43LK5400 | LG | 43 | LED | 3750000 |
| LG 32LB561T | LG | 32 | LED | 3100000 |
| Samsung UA40J5100 | Samsung | 40 | LED | 3799000 |
| NIKO NK-32ALPHA | NIKO | 32 | LED | 1295000 |
| Samsung UA65RU7300KPXD | Samsung | 65 | LED | 16350000 |
| AKARI LE-32M88 | AKARI | 32 | LED | 1650000 |
| Sharp LC-60LE275X | Sharp | 60 | LED | 10500000 |
| Samsung UA65RU7400KPXD | Samsung | 65 | LED | 14500000 |
| ICHIKO S5568 | ICHIKO | 55 | LED | 6200000 |
| Mito A120 | Mito | 17 | LED | 675000 |
| CooCaa 50E2000T | CooCaa | 50 | LED | 4599999 |
| Philips 32PHT5853S | Philips | 32 | LED | 2299000 |
| Sharp 2T-C50AD1i | Sharp | 50 | LED | 4675000 |
| Sharp LC-40SA5200 | Sharp | 40 | LED | 2695000 |
| LG 49LF550A | LG | 49 | LED | 6299000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| AKARI LE-23K88 | AKARI | 32 | LED | 1399000 |
| LG 22LN4100 | LG | 22 | LED | 1650000 |
| LG 32LK5000 | LG | 32 | LED | 1744000 |
| Panasonic TH-43D305G | Panasonic | 43 | LED | 1869000 |
| Philips 43PUT6002 | Philips | 43 | LED | 4690000 |
| CooCaa 32A2A12 | CooCaa | 32 | LED | 1400000 |
| LG 24TK42 | LG | 24 | LED | 1199000 |
| AKARI SC-52V32 | AKARI | 32 | LED | 1800000 |
| AQUA LE32AQT6900 | AQUA | 32 | LED | 1545000 |
| Samsung UA32J6300 | Samsung | 32 | LED | 5850000 |
| Philips 32PHA3052 | Philips | 32 | LED | 1690000 |
| Panasonic TH-43E305G | Panasonic | 43 | LED | 3175000 |
| Samsung UA55M5500 | Samsung | 55 | QLED | 8999000 |
| Sharp AQUOS LC-50LE275X | Sharp | 50 | LED | 6499900 |
| Samsung UA55K5100 | Samsung | 55 | LED | 8700000 |
| Panasonic TH-24E302G | Panasonic | 24 | LED | 1229000 |
| Samsung UA43N5500 | Samsung | 43 | LED | 4198000 |
| LG 55LB561T | LG | 55 | LED | 16750000 |
| Philips 40PFA4150S | Philips | 40 | LED | 3499000 |
| LG 60UF770T | LG | 50 | LED | 18200000 |
| Polytron PLD 22D9500 | Polytron | 22 | LCD | 1142500 |
| Samsung UA32J4100 | Samsung | 32 | LED | 2820000 |
| CooCaa 19E66B | CooCaa | 19 | LED | 1350000 |
| ICHIKO ST3976 | ICHIKO | 39 | LED | 3400000 |
| LG M2241A | LG | 22 | LCD | 200000 |
| Sharp AQUOS LC-24LE155M | Sharp | 24 | LED | 100000 |
| Sharp LC-32SA4101I | Sharp | 32 | LED | 1701000 |
| Sony KD-55X8000G | Sony | 55 | LED | 8729000 |
| Samsung UA58H5200 | Samsung | 58 | LED | 13000000 |
| Samsung UA50F6400AM | Samsung | 50 | LED | 11550000 |
| CHANGHONG 40C1600A | CHANGHONG | 40 | LED | 3400000 |
| Samsung UA50F5000AM | Samsung | 50 | LED | 12850000 |
| CooCaa 32E89 | CooCaa | 32 | LED | 1425000 |
| Polytron PLD 40TS853 | Polytron | 40 | LED | 3149000 |
| Toshiba 39L4300 | Toshiba | 39 | LED | 6499000 |
| Samsung UA40J5000 | Samsung | 40 | LED | 3162000 |
| LG 22MT45A | LG | 22 | LED | 1719000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| LG 65SK8000 | LG | 65 | LED | 15999000 |
| Samsung UA43K5500 | Samsung | 43 | LED | 3950000 |
| Sharp 2T-C40AE1 | Sharp | 40 | LED | 3199000 |
| LG 55UH600T | LG | 55 | LED | 10700000 |
| Polytron PLD 40T851 | Polytron | 39 | LED | 2750000 |
| LG 43UH6500 | LG | 43 | LED | 7360000 |
| Toshiba 40L5400 | Toshiba | 32 | LED | 6850000 |
| Samsung QA55Q7FAMKXXA | Samsung | 55 | QLED | 18900000 |
| Philips 50PUT6002 | Philips | 50 | LED | 5599000 |
| Samsung 43NU7100 | Samsung | 43 | OLED | 4799000 |
| LG 49LK5400 | LG | 49 | LED | 4750000 |
| CooCaa 55E2000T | CooCaa | 55 | LED | 7399000 |
| AKARI LE-20D88 | AKARI | 20 | LED | 1099000 |
| AKARI LE-20K88 | AKARI | 20 | LED | 1150000 |
| AKARI LE-4399T2SB | AKARI | 43 | LED | 2998000 |
| AKARI SC-52V40 | AKARI | 40 | LED | 2699000 |
| AQUA LE40AQT8300 | AQUA | 40 | LED | 3099000 |
| LG 50UM7300 | LG | 50 | LED | 5940000 |
| Toshiba 32L2900 | Toshiba | 32 | LED | 1685000 |
| LG 49UH610T | LG | 49 | LED | 7509000 |
| Toshiba 47L2400 | Toshiba | 32 | LED | 6988000 |
| LG 24MN33A | LG | 24 | LED | 1750000 |
| Polytron PLD-43TS153 | Polytron | 43 | LED | 3070000 |
| IKEDO LT-15P1W | IKEDO | 15 | LED | 631900 |
| LG 43UH650T | LG | 43 | LED | 6250000 |
| IKEDO LT-20H1U | IKEDO | 20 | LED | 850000 |
| LG 43UK6500PLA | LG | 43 | OLED | 4950000 |
| LG 70UK6540PTA | LG | 70 | LED | 15999000 |
| LG 42LA6130 | LG | 42 | LED | 7650000 |
| Toshiba 32P2400 | Toshiba | 32 | LED | 2595000 |
| Polytron PLD 43TS156 | Polytron | 43 | LED | 4000000 |
| Samsung UA24H4003 | Samsung | 24 | LED | 2150000 |
| Toshiba 24P2301 | Toshiba | 24 | LED | 1800000 |
| LG 22MT48AF-PT | LG | 22 | LED | 1215000 |
| Panasonic TH-22G302G | Panasonic | 22 | LED | 1000000 |
| LG 55UM7300 | LG | 55 | LED | 7590000 |
| Panasonic TH-32E306G | Panasonic | 32 | LED | 1999000 |
| CHANGHONG L29G3 | CHANGHONG | 29 | LED | 1299000 |
| CooCaa 32A2A11A | CooCaa | 32 | LED | 1450000 |
| LG 32LF630T | LG | 32 | LED | 4550000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| Polytron PLD 22D851 | Polytron | 22 | LED | 1750000 |
| CooCaa 43E2A22G | CooCaa | 43 | LED | 3300000 |
| Sony KD-49X8000G | Sony | 49 | LED | 7080000 |
| LG 32LB582D | LG | 32 | LED | 4350000 |
| LG 32LN5400 | LG | 32 | LED | 3550000 |
| Samsung UA32EH4003 | Samsung | 32 | LED | 1750000 |
| Sony KD-55X7000E | Sony | 55 | LED | 7570000 |
| LG 24MT44A | LG | 24 | LED | 1585000 |
| LG 49SK8500 | LG | 49 | OLED | 6790000 |
| Sony KDL-49W750D | Sony | 49 | LED | 7999000 |
| SANYO LE32S6500 | SANYO | 32 | LED | 1560000 |
| CHANGHONG 50E2000 | CHANGHONG | 50 | LED | 4981100 |
| ICHIKO S4088 | ICHIKO | 40 | LED | 3100000 |
| LG 55SK9500 | LG | 55 | LED | 29837500 |
| LG 65UM7300 | LG | 65 | LED | 12999000 |
| TCL 48P1CFS | TCL | 48 | LED | 5000000 |
| AKARI LE-5099T2SB | AKARI | 50 | LED | 4012000 |
| AKARI LE-29V89 | AKARI | 29 | LED | 1275000 |
| Philips 43PFT6100S | Philips | 43 | LED | 3990000 |
| Panasonic TH-40A403G | Panasonic | 40 | LED | 750000 |
| LG 42LM6200 | LG | 42 | LED | 10250000 |
| Samsung QA88Q9F | Samsung | 88 | QLED | 1,65E+08 |
| CooCaa 40S5C | CooCaa | 40 | LED | 2600000 |
| LG 20LB450A | LG | 20 | LED | 1500000 |
| CHANGHONG 3D51C2000 | CHANGHONG | 51 | LED | 4900000 |
| Sony KD-49X7000G | Sony | 49 | LED | 6299000 |
| Samsung UA40J5200 | Samsung | 40 | LED | 3900000 |
| CHANGHONG L43H4 | CHANGHONG | 43 | LED | 3049000 |
| Samsung UA50NU7100 | Samsung | 50 | LED | 6099000 |
| LG 43UJ652T | LG | 43 | QLED | 5199000 |
| Polytron PLD 32D100 | Polytron | 32 | LED | 2725000 |
| Polytron PLD 32V710 | Polytron | 32 | LED | 2699000 |
| AKARI LE-24K88 | AKARI | 24 | LED | 1144000 |
| Toshiba 65U9750VJ | Toshiba | 65 | LED | 14995000 |
| Samsung QA75Q8C | Samsung | 75 | OLED | 42095000 |
| Samsung UA32F6400AM | Samsung | 32 | LED | 4950000 |
| LG OLED55C8PTA | LG | 55 | OLED | 19999000 |
| Samsung UA65MU8000 | Samsung | 65 | LED | 14999000 |
| Polytron PLD 50S883 | Polytron | 50 | LED | 4750000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| NIKO NK-24 Gamma | NIKO | 24 | LED | 999000 |
| LG 49UJ632T | LG | 49 | QLED | 7088000 |
| Sharp AQUOS LC-58UE630X | Sharp | 50 | LED | 11499000 |
| LG 32LM570 | LG | 32 | LED | 2140000 |
| Samsung UA50KU6000 | Samsung | 50 | LED | 9000000 |
| ICHIKO S1518 | ICHIKO | 15 | LED | 799000 |
| Samsung UA40F6400AM | Samsung | 40 | LED | 7350000 |
| LG 28TK430V | LG | 28 | LED | 1400000 |
| CHANGHONG 32B2700 | CHANGHONG | 32 | LED | 1969500 |
| LG OLED55B6T | LG | 55 | OLED | 28450000 |
| Sony KD-55X8500D | Sony | 55 | LED | 11375000 |
| LG 49UH650T | LG | 49 | LED | 8000000 |
| Sony KD-43X8000G | Sony | 43 | LED | 5999000 |
| Samsung UA55JU6600 | Samsung | 55 | LED | 25000000 |
| Sony KD-43X7500F | Sony | 43 | OLED | 6945000 |
| LG 47LB561T | LG | 47 | LED | 8250000 |
| CooCaa 50E6000 | CooCaa | 50 | LED | 9299000 |
| Sony KDL-48W700C | Sony | 48 | LED | 7350000 |
| Philips 22PFA5403S / 70 | Philips | 22 | LED | 1075700 |
| CooCaa 24E100 | CooCaa | 24 | LED | 1050000 |
| Polytron PLD 22D110 | Polytron | 22 | LED | 1130000 |
| Sony KDL-40W650D | Sony | 40 | LED | 4199000 |
| AQUA LE24AQT8300 | AQUA | 25 | LED | 1375000 |
| Samsung UA43RU7100KPXD | Samsung | 43 | LED | 4565000 |
| Sony KDL-40W660E | Sony | 40 | LED | 5800000 |
| Sony KLV-32R302C | Sony | 32 | LED | 2499000 |
| IKEDO LT-22P1U | IKEDO | 22 | LED | 1125000 |
| Polytron PLD 24D552 | Polytron | 24 | LED | 1850000 |
| Sharp AQUOS LC-29LE440M | Sharp | 29 | LED | 2450000 |
| Sony KDL-49W660E | Sony | 49 | LED | 7300000 |
| Samsung UA55EH6000 | Samsung | 55 | LED | 4850000 |
| LG 55UK6100 | LG | 55 | OLED | 7889000 |
| Sony KD-75X9500G | Sony | 75 | LED | 38700000 |
| Polytron PLD 24D800 | Polytron | 24 | LED | 1549000 |
| CHANGHONG L50G3 | CHANGHONG | 50 | LED | 3699000 |
| LG 55UF680T | LG | 55 | LED | 12995000 |
| Panasonic TH-49E305G | Panasonic | 49 | LED | 3899000 |
| Samsung S32F351 | Samsung | 32 | LED | 3560001 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| LG 40UB800T | LG | 40 | LED | 7350000 |
| Philips 24PHA4110 | Philips | 24 | LED | 1479333 |
| CooCaa 32E6000 | CooCaa | 40 | LED | 5499000 |
| LG 65UK6300PTE | LG | 65 | LED | 11000000 |
| AKARI LE-32V99 | AKARI | 32 | LED | 1345000 |
| Samsung UA43KU6000 | Samsung | 43 | LED | 5958000 |
| TCL 29E4200 | TCL | 29 | LED | 1349000 |
| Samsung QA55Q8C | Samsung | 55 | QLED | 17500000 |
| AKARI LE-25V89 | AKARI | 24 | LED | 1102000 |
| Sony KD-55X8500C | Sony | 55 | LED | 11230000 |
| Samsung UA55MU6300 | Samsung | 55 | LED | 8900000 |
| Toshiba 32L2550 | Toshiba | 32 | LED | 2850000 |
| Hisense L40D50P | Hisense | 40 | LED | 3190000 |
| AKARI LE-32D88 | AKARI | 32 | LED | 1475000 |
| Sony KD-65X7500F | Sony | 65 | OLED | 11500000 |
| LG 28MT48AF-PT | LG | 28 | LED | 1890000 |
| LG 43LM5700PTC | LG | 43 | LED | 3996000 |
| Samsung UA40F5105AR | Samsung | 40 | LED | 5150000 |
| Philips 39PHA4251 | Philips | 39 | LED | 2599000 |
| Sony KDL-40R352C | Sony | 40 | LED | 3750000 |
| CooCaa 40S3A12G | CooCaa | 40 | LED | 2865000 |
| SANYO LE32S500 | SANYO | 32 | LED | 2990000 |
| Samsung UE43NU7090 | Samsung | 43 | LED | 4399000 |
| LG 32LM553 | LG | 32 | LED | 1799000 |
| Panasonic TH-43F305G | Panasonic | 43 | LED | 3199000 |
| IKEDO LT-22T1AU | IKEDO | 22 | LED | 1200000 |
| CHANGHONG U58H7A | CHANGHONG | 58 | LED | 6299000 |
| Sony KD-49X8300C | Sony | 49 | LED | 15888000 |
| Polytron PLD 32V9500 | Polytron | 32 | LED | 2475000 |
| Samsung UA40F5500AM | Samsung | 40 | LED | 5500000 |
| Samsung UA49M5100 | Samsung | 49 | LED | 4350000 |
| Polytron PLD 32D715 | Polytron | 32 | LED | 1950000 |
| CHANGHONG U65H7A | CHANGHONG | 65 | LED | 9499000 |
| CooCaa 55G2 | CooCaa | 55 | OLED | 5999000 |
| Sony KD-49X8000E | Sony | 49 | LED | 4999000 |
| Sony KD-55X8000E | Sony | 55 | LED | 8435000 |
| Polytron PLD 28T655 | Polytron | 28 | LED | 2650000 |
| LG 55UK6540PTD | LG | 55 | OLED | 7789000 |
| AQUA LE32AQT7000T | AQUA | 32 | LED | 2150000 |
| Polytron PLD 40D851 | Polytron | 39 | LED | 999000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| Panasonic TH-40C304G | Panasonic | 40 | LED | 2800000 |
| Polytron MX 1503R | Polytron | 15 | LCD | 1964000 |
| LG 55UH615T | LG | 55 | LED | 10225000 |
| Sony KD-49X7000E | Sony | 49 | LED | 6745000 |
| Sony KD-43X7000E | Sony | 43 | LED | 6499000 |
| Toshiba 32L2800 | Toshiba | 32 | LED | 1725000 |
| Samsung UA32F4510AM | Samsung | 32 | LED | 2950000 |
| Samsung UA43k5005 | Samsung | 43 | LED | 3123000 |
| Samsung UA40MU6100 | Samsung | 40 | LED | 4200000 |
| LG 50UK6540PTD | LG | 50 | OLED | 6149000 |
| CHANGHONG 29C2000 | CHANGHONG | 29 | LED | 1999000 |
| LG OLED55C7T | LG | 55 | OLED | 22999000 |
| Polytron PLD 32D9505 | Polytron | 32 | LED | 1775500 |
| Sony KDL-40R350C | Sony | 40 | LED | 4085000 |
| Daewoo DTV-49S1 | Daewoo | 49 | LED | 5249000 |
| Panasonic TH-32C305G | Panasonic | 32 | LED | 2450000 |
| CHANGHONG 32C2000 | CHANGHONG | 32 | LED | 1800000 |
| Sony KD-65A8F | Sony | 65 | OLED | 37800000 |
| Samsung UA49J5250 | Samsung | 49 | LED | 4999000 |
| Samsung UA32F4105AR | Samsung | 32 | LED | 2800000 |
| Sony KD-65X8000G | Sony | 65 | LED | 13215000 |
| AQUA LE24AQT6500T | AQUA | 24 | LED | 1820900 |
| LG 49UM7290PTA | LG | 49 | LED | 6175000 |
| Polytron PLD 32D758 | Polytron | 32 | LED | 1728500 |
| Samsung UA48JU6000 | Samsung | 48 | LED | 8900000 |
| LG 55UM7600 | LG | 55 | LED | 7989000 |
| Samsung UA43N5001 | Samsung | 43 | LED | 3097000 |
| LG 49UB850T | LG | 49 | Plasma | 7450000 |
| Samsung QA65Q7FAMKXXA | Samsung | 65 | QLED | 22540000 |
| Toshiba 43U7750VJ | Toshiba | 43 | LED | 6495000 |
| CooCaa 32E21W | CooCaa | 32 | LED | 1895000 |
| Polytron PLD 43S883 | Polytron | 43 | LED | 3450000 |
| Panasonic TH-32D400G | Panasonic | 32 | LED | 2575000 |
| Sony KD-55X7000D | Sony | 55 | LED | 7575000 |
| CHANGHONG 32D2000 | CHANGHONG | 32 | LED | 2002000 |
| Samsung 55NU7090 | Samsung | 55 | LED | 6670000 |
| Samsung UA49M5000 | Samsung | 49 | LED | 4408012 |
| Samsung UA48JU7500 | Samsung | 48 | LED | 9900000 |
| LG 55LH575T | LG | 55 | LED | 10005000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| Mito 5011 | Mito | 49 | LED | 3599000 |
| Polytron PLD 32T451 | Polytron | 32 | LED | 2750000 |
| ICHIKO S4998 | ICHIKO | 49 | LED | 4300000 |
| LG 49LK5100 | LG | 49 | LED | 4105000 |
| Samsung UA50JS7200 | Samsung | 50 | LED | 10500000 |
| CHANGHONG 20E2000 | CHANGHONG | 20 | LED | 1050000 |
| LG 60UH770T | LG | 60 | LED | 22599000 |
| CooCaa 50E700A | CooCaa | 52 | LED | 6669100 |
| ICHIKO ST3256 | ICHIKO | 32 | LED | 1899000 |
| Mito 3255 | Mito | 32 | LED | 1395000 |
| LG 43UF670T | LG | 43 | LED | 5899000 |
| LG 65UH850T | LG | 65 | LED | 28600000 |
| Panasonic VIERA TH-L24C28 | Panasonic | 24 | LCD | 1570000 |
| Samsung UA55F6400AM | Samsung | 55 | LED | 18500000 |
| Sony KD-49X7000D | Sony | 49 | LED | 10100000 |
| Sharp AQUOS LC-40LE180i | Sharp | 40 | LED | 3550000 |
| Samsung UA55RU7100 | Samsung | 55 | LED | 7990000 |
| Samsung UA55ES6220 | Samsung | 55 | LED | 10224000 |
| Sony KDL-40R550C | Sony | 40 | LED | 4990000 |
| ICHIKO S4878 | ICHIKO | 48 | LED | 3799000 |
| ICHIKO S4879 | ICHIKO | 48 | LED | 4699000 |
| ICHIKO S6596 | ICHIKO | 65 | LED | 8250000 |
| IKEDO LT-17L2U | IKEDO | 17 | LED | 750000 |
| CooCaa 50E3000T | CooCaa | 50 | LED | 6599000 |
| LG 55SK8500 | LG | 55 | LED | 9998000 |
| LG 65UJ632T | LG | 65 | QLED | 15495000 |
| CooCaa 40E3000T | CooCaa | 40 | LED | 3699000 |
| Panasonic TH-32G306G | Panasonic | 32 | LED | 1929000 |
| Samsung S20D300HL | Samsung | 20 | LED | 965000 |
| Panasonic TH-49C305G | Panasonic | 49 | LED | 3999000 |
| CooCaa 19E88 | CooCaa | 19 | LED | 1395000 |
| Sony KDL-32R300E | Sony | 32 | LED | 2669500 |
| TCL 29D2950 | TCL | 29 | LED | 1137000 |
| Sony KD-65X8500C | Sony | 65 | LED | 26500000 |
| Toshiba 40L5550 | Toshiba | 40 | LED | 3799000 |
| Samsung UA65KU6500 | Samsung | 65 | LED | 23000000 |
| Samsung UA55K6300 | Samsung | 55 | LED | 10300000 |
| LG 43LH570T | LG | 43 | LED | 5000000 |
| LG 55LF550T | LG | 50 | LED | 12688000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| CHANGHONG 40E6000 | CHANGHONG | 40 | LED | 2699000 |
| Panasonic TH-22G305G | Panasonic | 22 | LED | 1175000 |
| ICHIKO S2219 | ICHIKO | 22 | LED | 760000 |
| AKARI LE-19D88 | AKARI | 19 | LED | 1277000 |
| Samsung UA55KU6000 | Samsung | 55 | LED | 11000000 |
| Sharp AQUOS LC-40LE830M | Sharp | 40 | LCD | 4350000 |
| Hisense L24D33 | Hisense | 24 | LED | 1479000 |
| Polytron PLD 24T811 | Polytron | 24 | LED | 1375000 |
| Sharp AQUOS LC-24LE107I | Sharp | 24 | LED | 1079000 |
| Sharp AQUOS LC-32LE155M | Sharp | 32 | LED | 2198000 |
| CHANGHONG 55D3000i | CHANGHONG | 55 | LED | 6899000 |
| Samsung LS19F350HNEX | Samsung | 19 | LED | 908000 |
| Sharp AQUOS LC-40LE295i | Sharp | 40 | LED | 3298000 |
| LG 49LF630T | LG | 49 | LED | 9880000 |
| Sharp LC-40UA330X | Sharp | 40 | LED | 5390000 |
| LG 32LB552A | LG | 32 | LED | 3000000 |
| Sharp AQUOS LC-50LE580X | Sharp | 50 | LED | 8487900 |
| LG 60LN5400 | LG | 60 | LED | 19750000 |
| Sharp AQUOS LC-45LE280X | Sharp | 45 | QLED | 5249000 |
| Panasonic TH-49DS630G | Panasonic | 49 | LED | 8298000 |
| Samsung UA43MU6100 | Samsung | 43 | LED | 5650000 |
| LG 49UH600T | LG | 49 | LED | 7128000 |
| CHANGHONG L39G3A | CHANGHONG | 39 | LED | 2449000 |
| AQUA LE32AQT6500 | AQUA | 32 | LED | 2009900 |
| AQUA LE49AQT1000U | AQUA | 49 | LED | 4499000 |
| Philips 43PFT5853S / 70 | Philips | 43 | LED | 3880000 |
| Samsung UA55MU6100 | Samsung | 55 | LED | 8448000 |
| Mito 298 | Mito | 22 | LED | 1615000 |
| AKARI LE-3288T2 | AKARI | 32 | LED | 2240000 |
| Samsung QLED Q7F QA55Q7FNAKPXD | Samsung | 55 | QLED | 14675000 |
| Panasonic TH-55EX400G | Panasonic | 55 | LED | 6999000 |
| Samsung UA55J6300 | Samsung | 55 | LED | 14578000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| Sony KD-65A9F | Sony | 65 | OLED | 53145000 |
| Toshiba 22L2800 | Toshiba | 22 | LED | 1265000 |
| LG 65UH615T | LG | 65 | LED | 17999000 |
| Samsung UA55KU6500 | Samsung | 55 | LED | 13100000 |
| Philips 49PFA4300S | Philips | 49 | LED | 6126550 |
| Sharp AQUOS LC-50LE570X | Sharp | 32 | LED | 8997000 |
| Polytron PLD-40TS156 | Polytron | 40 | LED | 3067000 |
| Samsung UA40H5500 | Samsung | 40 | LED | 4499000 |
| LG 65UH600T | LG | 65 | LED | 20345000 |
| AQUA LE32AQT1000U | AQUA | 32 | LED | 1562500 |
| Sony KD-75X8000G | Sony | 75 | LED | 26460000 |
| CooCaa 32E28W | CooCaa | 32 | LED | 1449000 |
| Panasonic TH-43F306G | Panasonic | 43 | LED | 3498000 |
| LG 55SJ800T | LG | 55 | QLED | 13500000 |
| Samsung 49NU7100 | Samsung | 49 | LED | 5799000 |
| Sharp AQUOS LC-40LE835X | Sharp | 40 | LED | 4650000 |
| Samsung UA49K6300 | Samsung | 49 | LED | 7269000 |
| AKARI LE-3289T2 | AKARI | 32 | LED | 2089000 |
| LG 43LH540T | LG | 43 | LED | 3666000 |
| Samsung UA65NU800 | Samsung | 65 | OLED | 12400000 |
| Hisense L20D50 | Hisense | 20 | LED | 1169000 |
| CHANGHONG L43G3 | CHANGHONG | 43 | LED | 2599000 |
| Sharp AQUOS LC-60LE380X | Sharp | 60 | LED | 10500000 |
| Samsung UA22H5003 | Samsung | 22 | LED | 1650000 |
| LG 49LH570T | LG | 49 | LED | 7088000 |
| AQUA LE32AQT6500T | AQUA | 32 | LED | 1565000 |
| AKARI LE-20V89 | AKARI | 19 | LED | 1015000 |
| AKARI LE-24V89 | AKARI | 24 | LED | 1129000 |
| AKARI LE-43P88 | AKARI | 43 | LED | 2800000 |
| AKARI SC-52V43 | AKARI | 43 | LED | 3799000 |
| CHANGHONG 55D2200 | CHANGHONG | 55 | LED | 6399000 |
| Toshiba 43L3650VJ | Toshiba | 43 | LED | 3524000 |
| Panasonic TH-40D305G | Panasonic | 40 | LED | 3000000 |
| Toshiba 50L4300 | Toshiba | 50 | LED | 11200000 |
| LG 24LB452A | LG | 24 | LED | 2100000 |
| Panasonic TH-22F302G | Panasonic | 22 | LED | 1080000 |
| Samsung UA43J5500 | Samsung | 43 | LED | 3975000 |
| Samsung UA55KS9000 | Samsung | 55 | LED | 22000000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| NIKO NK-1702 | NIKO | 17 | LED | 718900 |
| Sony KDL-50W660G | Sony | 50 | LCD | 5250000 |
| Panasonic TH-49F306G | Panasonic | 49 | LED | 4699000 |
| LG 50UM7600 | LG | 50 | LED | 7350000 |
| LG 29MT48AF-PT | LG | 29 | LED | 2050000 |
| Samsung UA49MU6300 | Samsung | 49 | LED | 6988000 |
| Samsung UA55NU7100 | Samsung | 55 | LED | 6485000 |
| CHANGHONG L24G3A | CHANGHONG | 24 | LED | 1050000 |
| AKARI LE-50D99SBS | AKARI | 50 | LED | 4498000 |
| ICHIKO S1798 | ICHIKO | 17 | LED | 675000 |
| Philips 40PFT5063S / 70 | Philips | 40 | LED | 2489000 |
| Philips 22PHA5403S | Philips | 22 | LED | 999000 |
| CHANGHONG 40D3000i | CHANGHONG | 40 | LED | 4815200 |
| ICHIKO S3998 | ICHIKO | 39 | LED | 2849000 |
| Panasonic TH-32ES500G | Panasonic | 32 | LED | 4888000 |
| Samsung UA65RU7100 | Samsung | 65 | LED | 11700000 |
| Sony KD-55X7500F | Sony | 55 | OLED | 8475000 |
| LG 65SM8100 | LG | 65 | LED | 18900000 |
| Samsung UA32J5100 | Samsung | 32 | LED | 3750000 |
| Sony KD-70X8300F | Sony | 70 | LED | 25695000 |
| Sharp 4T-C50AH1X | Sharp | 50 | LED | 5887100 |
| Samsung UA55NU800 | Samsung | 55 | OLED | 8250000 |
| Polytron PLD 32D750 | Polytron | 32 | LED | 1734000 |
| Samsung UA65NU7100 | Samsung | 65 | LED | 13590000 |
| Panasonic TH-24E303G | Panasonic | 24 | LED | 1315000 |
| LG 24LF450A | LG | 24 | LED | 2250000 |
| Polytron PLD 40V853 | Polytron | 39 | LED | 4075000 |
| Samsung QA55Q80R | Samsung | 55 | LED | 22495000 |
| Polytron PLD 32T106 | Polytron | 32 | LED | 1975000 |
| ICHIKO S5588 | ICHIKO | 55 | LED | 5400000 |
| Mito 3212 | Mito | 32 | LED | 1299400 |
| CooCaa 39LF20 | CooCaa | 39 | LED | 3400000 |
| LG OLED65C8PTA | LG | 65 | OLED | 31998000 |
| Sony KD-55X7000G | Sony | 55 | LED | 7670000 |
| AKARI LE-4289T2 | AKARI | 32 | LED | 3900000 |
| Mito 1930 | Mito | 19 | LED | 1420000 |
| LG 55SK8000 | LG | 55 | LCD | 8190000 |
| CHANGHONG 50E6000 | CHANGHONG | 50 | LED | 4375000 |
| Sony KD-65A9G | Sony | 65 | OLED | 52900000 |
| LG 39LN5400 | LG | 39 | LED | 4950000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| LG 55UJ632T | LG | 55 | LED | 9000000 |
| LG 65SM9000PTA | LG | 65 | NanoCell | 20000000 |
| Polytron PLD 40TV853 | Polytron | 40 | LED | 3027900 |
| LG 43UF690T | LG | 43 | LED | 7280000 |
| LG 55UH770T | LG | 55 | LED | 11999000 |
| Sony KDL-42W800B | Sony | 42 | LCD | 7649100 |
| Samsung QA55Q70R | Samsung | 55 | QLED | 18875000 |
| LG 20MT47A | LG | 20 | LED | 1398000 |
| ARISA AL32-S200 | ARISA | 32 | LED | 2025000 |
| IKEDO LT-17L1H | IKEDO | 17 | LED | 825000 |
| Sony KD-55X9500G | Sony | 55 | LED | 19990000 |
| Samsung 65NU7300 | Samsung | 65 | LED | 13575000 |
| LG 43UH610T | LG | 43 | LED | 5600000 |
| Polytron PLD 50T951 | Polytron | 50 | LED | 4950000 |
| LG 49UF770T | LG | 49 | LED | 11200000 |
| CHANGHONG 40D2100T | CHANGHONG | 40 | LED | 3175000 |
| LG 86SJ957T | LG | 86 | LED | 1,26E+08 |
| Panasonic TH-55F306G | Panasonic | 30 | LED | 6190000 |
| Toshiba 29P2300 | Toshiba | 29 | LED | 2250000 |
| Sony KLV-32R300B | Sony | 32 | LCD | 3150000 |
| Sony KD-49X7500F | Sony | 49 | OLED | 5850000 |
| LG 42LM6700 | LG | 42 | LED | 12250000 |
| Samsung UA40F6100AM | Samsung | 40 | LED | 7150000 |
| Samsung UA46F6100AM | Samsung | 46 | LED | 12500000 |
| Samsung UA60F6300AM | Samsung | 60 | LED | 21750000 |
| Samsung UA65F6400AM | Samsung | 65 | LED | 37500000 |
| Toshiba 40PX200 | Toshiba | 40 | LED | 4550000 |
| LG 32LN541B | LG | 32 | LED | 2950000 |
| Panasonic VIERA TH-L39B6G | Panasonic | 39 | LED | 4250000 |
| Samsung UA32H5100AW | Samsung | 32 | LED | 3850000 |
| Samsung UA48H5100AW | Samsung | 48 | LED | 5925000 |
| Samsung UA22H5000 | Samsung | 22 | LED | 1650000 |
| Samsung UA40H6400AW | Samsung | 40 | LED | 9950000 |
| LG 55LN5400 | LG | 55 | LED | 13750000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| LG 55LB582T | LG | 55 | LED | 18750000 |
| LG 60LB561T | LG | 60 | LED | 17500000 |
| LG 60LB582T | LG | 60 | LED | 19500000 |
| Toshiba 32P1300 | Toshiba | 32 | LED | 2750000 |
| Polytron PLD 40T555 | Polytron | 40 | LED | 650000 |
| LG 42LB620T | LG | 42 | LED | 8250000 |
| LG 50LB561T | LG | 50 | LED | 11750000 |
| Samsung UA40H4200 | Samsung | 40 | LED | 4750000 |
| Toshiba 23S2400 | Toshiba | 23 | LED | 1800000 |
| LG 49LB620T | LG | 49 | LED | 7888000 |
| LG 22LB452A | LG | 22 | LED | 1750000 |
| LG 50LB582T | LG | 50 | LED | 13750000 |
| LG 55UB950T | LG | 55 | LED | 20999000 |
| LG 22MT44A | LG | 22 | LED | 1830000 |
| LG 47LN5710 | LG | 47 | LED | 11750000 |
| LG 28MT45A | LG | 28 | LED | 1800000 |
| Polytron PLD 42T551 | Polytron | 42 | LED | 4850000 |
| Toshiba 46WL700 | Toshiba | 46 | LED | 8750000 |
| LG 65UF670T | LG | 65 | LED | 18995000 |
| LG 47LN5400 | LG | 47 | LED | 9150000 |
| LG 29LN4510 | LG | 29 | LED | 2500000 |
| Samsung UA65JS9000 | Samsung | 65 | LED | 35300000 |
| Samsung UA75JU6400 | Samsung | 75 | LED | 38495000 |
| Samsung UA55JS9000 | Samsung | 55 | LED | 29999000 |
| Samsung UA65JU6600 | Samsung | 65 | LED | 27000000 |
| Samsung UA55JU6400 | Samsung | 55 | LED | 18499000 |
| Samsung UA48J6300 | Samsung | 48 | LED | 10200000 |
| Samsung UA40J6300 | Samsung | 40 | LED | 7550000 |
| Samsung UA40JU6600 | Samsung | 40 | LED | 9435000 |
| Samsung UA40JU6400 | Samsung | 40 | LED | 8999000 |
| CooCaa 24E88 | CooCaa | 24 | LED | 1430000 |
| LG 55UF850T | LG | 55 | LED | 17200000 |
| LG 32LF561T | LG | 32 | LED | 3950000 |
| LG 42LF561T | LG | 42 | LED | 6500000 |
| Mito A868 | Mito | 32 | LED | 2299900 |
| LG 24MT45A | LG | 24 | LED | 1220000 |
| Samsung S24D300HLR | Samsung | 24 | LED | 2225000 |
| CHANGHONG 40D1200 | CHANGHONG | 40 | LED | 2299000 |
| Toshiba 50L5550 | Toshiba | 50 | LED | 10250000 |
| LG 43LF510A | LG | 43 | LED | 4725000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| Samsung UA55JS7200 | Samsung | 55 | LED | 17994000 |
| LG 58UF830T | LG | 58 | LED | 15000000 |
| Samsung UA40H5003AKXXD | Samsung | 40 | LED | 5199000 |
| SANYO LE24S8000 | SANYO | 24 | LED | 1529000 |
| CooCaa 32E390 | CooCaa | 32 | LED | 2348000 |
| Sanken SLE-50 | Sanken | 50 | LED | 5750000 |
| Samsung UA48J5000 | Samsung | 48 | LED | 5899000 |
| Panasonic TH-L32C304 | Panasonic | 32 | LED | 2050000 |
| CHANGHONG 43D3000i | CHANGHONG | 43 | LED | 2975000 |
| Panasonic TH-40CS500G | Panasonic | 40 | LED | 4999000 |
| LG 49LF590 | LG | 49 | LED | 8475000 |
| LG 49LH540T | LG | 49 | LED | 6000000 |
| LG 49LH511T | LG | 49 | LED | 6579000 |
| LG 55EG910T | LG | 55 | LED | 14750000 |
| LG 49LF510 | LG | 49 | LED | 7370000 |
| LG 55UH650T | LG | 55 | LED | 11380000 |
| LG 60UH650T | LG | 60 | LED | 17199000 |
| LG 65UH650T | LG | 65 | LED | 18900000 |
| LG 49UH770T | LG | 49 | LED | 11880000 |
| LG 55UH850T | LG | 55 | LED | 14000000 |
| Sanken SLE-321HDJ | Sanken | 32 | LED | 2399000 |
| SANYO 24S6500T | SANYO | 24 | LED | 1950000 |
| Samsung UA65KU6000 | Samsung | 65 | LED | 19550000 |
| Samsung UA40K6300 | Samsung | 40 | LED | 5800000 |
| LG 43UF680T | LG | 43 | LED | 7280000 |
| Mito 3221 | Mito | 32 | LED | 2036000 |
| Samsung UA78KS9000 | Samsung | 78 | LED | 50800000 |
| Samsung UA43KU6300 | Samsung | 43 | LED | 9700000 |
| Samsung UA60KS7000 | Samsung | 60 | LED | 21499000 |
| Philips 39PHA4250S | Philips | 39 | LED | 3200000 |
| Samsung UA55K5500 | Samsung | 55 | LED | 11750000 |
| Polytron PLD 32D106 | Polytron | 32 | LED | 2899000 |
| LG 86UH955T | LG | 86 | LED | 1,28E+08 |
| CHANGHONG 50D3000i | CHANGHONG | 50 | LED | 6174100 |
| Panasonic TH-40DS500 | Panasonic | 40 | LED | 4385000 |
| AQUA LE32AQT9000T | AQUA | 32 | LED | 1800000 |
| Samsung UA40KU6000 | Samsung | 40 | LED | 5850000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| CHANGHONG 24D2000 | CHANGHONG | 24 | LED | 1370000 |
| Samsung UA49K5100 | Samsung | 49 | LED | 5999000 |
| Samsung UA49KU6300 | Samsung | 49 | LED | 8998000 |
| Polytron PLD 55UV5900 | Polytron | 55 | LED | 8300000 |
| Polytron PLD 65UV5900 | Polytron | 65 | LED | 19309900 |
| LG 65UF860T | LG | 65 | LED | 17890000 |
| Sony KD-65X9000E | Sony | 65 | LED | 24495000 |
| Panasonic TH-43ES630G | Panasonic | 43 | LED | 5488000 |
| ICHIKO S1788 | ICHIKO | 17 | LED | 650000 |
| IKEDO LT-17L2H | IKEDO | 17 | LED | 850000 |
| IKEDO LT-24P1U1 | IKEDO | 24 | LED | 1389000 |
| Samsung UA49MU6100 | Samsung | 49 | LED | 5599900 |
| Samsung UA65MU6100 | Samsung | 65 | LED | 16899000 |
| LG 49LJ550T | LG | 49 | QLED | 5700000 |
| LG 55UJ652T | LG | 55 | QLED | 7750000 |
| LG 65UJ652T | LG | 65 | QLED | 13000000 |
| LG 75UJ657T | LG | 75 | LED | 35700000 |
| LG 55SJ850T | LG | 55 | LCD | 10050000 |
| LG 65SJ850T | LG | 65 | LCD | 19750000 |
| LG OLED65B6T | LG | 65 | OLED | 33369000 |
| LG OLED65E6T | LG | 65 | OLED | 57899000 |
| Samsung UA40M5000 | Samsung | 40 | LED | 2900000 |
| Samsung UA43M5500 | Samsung | 43 | LED | 2999000 |
| Samsung UA49M6300 | Samsung | 49 | LED | 6750000 |
| Samsung UA55M6300 | Samsung | 55 | LED | 8990000 |
| Samsung UA49MU7000 | Samsung | 49 | LED | 5699000 |
| Samsung UA49MU8000 | Samsung | 49 | LED | 10895000 |
| Samsung UA55MU7000 | Samsung | 55 | LED | 14795000 |
| Samsung UA55MU8000 | Samsung | 55 | LED | 11990000 |
| Samsung UA65MU6500 | Samsung | 65 | QLED | 20000000 |
| Samsung UA75MU6100 | Samsung | 75 | QLED | 32300000 |
| Samsung QA65Q8C | Samsung | 65 | QLED | 26895000 |
| AQUA LE24AQT6550T | AQUA | 24 | LED | 1550000 |
| LG 49SJ800T | LG | 49 | LED | 9500000 |
| CooCaa 40E2100T | CooCaa | 40 | LED | 3499000 |
| TCL 49C2US | TCL | 49 | LED | 7450000 |
| CHANGHONG 50E2100 | CHANGHONG | 50 | LED | 3950000 |
| CooCaa 50E2A12G | CooCaa | 50 | LED | 4890000 |
| CHANGHONG 55E6000 | CHANGHONG | 32 | LED | 5799900 |
| ICHIKO S6558 | ICHIKO | 65 | LED | 10700000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| CooCaa 32E2A22G | CooCaa | 32 | LED | 2249000 |
| Polytron PLD 32T511 | Polytron | 32 | LED | 1800000 |
| Samsung UA49J5200 | Samsung | 49 | LED | 5400000 |
| Samsung UA40K5100 | Samsung | 40 | LED | 5360000 |
| ICHIKO S5058 | ICHIKO | 50 | LED | 6000000 |
| Toshiba 32L2605 | Toshiba | 32 | LED | 2259000 |
| Philips 43PFA3002S | Philips | 43 | LED | 4006700 |
| AQUA LE40AQT1000U | AQUA | 40 | LED | 2117600 |
| CHANGHONG 24E2000 | CHANGHONG | 24 | LED | 999000 |
| LG 65UK6100 | LG | 65 | OLED | 15500000 |
| Philips 24PHA4003 / 70 | Philips | 24 | LED | 1325000 |
| Philips 40PFA4160S | Philips | 40 | LED | 3399000 |
| Philips 55PUT6002 | Philips | 55 | LED | 7999000 |
| Polytron PLD 40TS856 | Polytron | 40 | LED | 3823000 |
| Polytron PLD 40TS865 | Polytron | 40 | LED | 4600000 |
| LG 49UJ654V | LG | 49 | LED | 7500000 |
| Polytron PLD 43TS866 | Polytron | 43 | LED | 3300000 |
| Polytron PLD 32D711 | Polytron | 32 | LED | 1776000 |
| LG 49UH652V | LG | 49 | LED | 8999000 |
| Samsung UA55NU7300J | Samsung | 55 | LED | 8750000 |
| Samsung UA49NU7300J | Samsung | 49 | LED | 4500000 |
| Samsung UA75NU7100 | Samsung | 75 | LED | 24000000 |
| Samsung UE50NU7090 | Samsung | 50 | LED | 5770000 |
| Samsung 49NU7300 | Samsung | 49 | LED | 7050000 |
| Samsung 40M5050 | Samsung | 40 | LED | 3999000 |
| LG OLED65C7T | LG | 65 | OLED | 30000000 |
| LG OLED65C8 | LG | 65 | OLED | 49999000 |
| Toshiba 24L2800 | Toshiba | 24 | LED | 1215000 |
| CooCaa 24W3 | CooCaa | 24 | LED | 1049000 |
| CooCaa 39W3 | CooCaa | 40 | LED | 1899000 |
| CooCaa 32E2A12AG | CooCaa | 32 | LED | 1515000 |
| CooCaa 32E6 | CooCaa | 32 | LED | 2799000 |
| TCL 49A3 | TCL | 49 | LED | 3999000 |
| TCL 49P32 | TCL | 49 | LED | 5999000 |
| LG OLED65E8PTA | LG | 65 | OLED | 42180900 |
| LG OLED55B8PTA | LG | 55 | OLED | 14800000 |
| Philips 43PFT5200 | Philips | 43 | LED | 3850000 |
| Samsung 40M6100 | Samsung | 40 | LED | 4575000 |
| Samsung UA40M5100 | Samsung | 40 | LED | 4900000 |
| Samsung UA55MU6103 | Samsung | 55 | LED | 9749000 |
| CooCaa 40E2A12G | CooCaa | 40 | LED | 2499000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| CooCaa 50G2 | CooCaa | 50 | OLED | 5500000 |
| LG 55EG9A7T | LG | 55 | LED | 14420000 |
| LG 32GK850G-B | LG | 32 | LED | 10500000 |
| CooCaa 24D1A | CooCaa | 24 | LED | 989900 |
| AKARI LE-40P88 | AKARI | 40 | LED | 2400000 |
| Samsung 58NU7103 | Samsung | 58 | LED | 8425000 |
| Samsung QA65Q6FNA | Samsung | 65 | LED | 21300000 |
| Samsung UA55LS003 | Samsung | 55 | LED | 22500000 |
| Sony KD-55A8F | Sony | 55 | OLED | 15999000 |
| Sony KD-55A9F | Sony | 55 | OLED | 38745000 |
| Sony KD-55X8500G | Sony | 55 | LED | 11235000 |
| Sony KD-85X9000F | Sony | 85 | LED | 75350000 |
| Sony KDL-43W660F | Sony | 43 | LED | 4750000 |
| Polytron PLD 65UV5901 | Polytron | 65 | LED | 17803900 |
| Polytron PLD 75UV5901 | Polytron | 75 | LED | 34652900 |
| Polytron PLD 55UV6900 | Polytron | 55 | LED | 10189900 |
| Panasonic LC-32SA4500I | Panasonic | 32 | LED | 2704000 |
| Samsung UA75RU7100KPXD | Samsung | 75 | LED | 19800000 |
| Sony KD-65X8500G | Sony | 65 | LED | 17000000 |
| Sony KD-65X9500G | Sony | 65 | LED | 28350000 |
| Polytron PLD 32D7511 | Polytron | 32 | LED | 1799000 |
| Hisense 32N2170 | Hisense | 32 | LED | 2027400 |
| Hisense 43N2170 | Hisense | 43 | LED | 3455000 |
| Sony KD-55X8500F | Sony | 55 | LED | 10165000 |
| Sony KDL-40R350E | Sony | 40 | LED | 4100000 |
| Samsung QA82Q60R | Samsung | 82 | LED | 64700000 |
| LG 60UM7100 | LG | 60 | LED | 9499000 |
| LG OLED65E7T | LG | 65 | OLED | 32990000 |
| Sony KD-49X8500G | Sony | 49 | LED | 9750000 |
| Sony KDL-43W660G | Sony | 43 | LED | 4765000 |
| Sony KD-65X9000F | Sony | 65 | LED | 21999000 |
| Samsung QA65Q70R | Samsung | 65 | QLED | 25500000 |
| Sony KD-55X9000F | Sony | 55 | LED | 15995000 |
| Sony KD-75X8500F | Sony | 75 | LED | 29695000 |
| LG 55SM8600 | LG | 55 | LED | 12495000 |
| LG 49SM8600 | LG | 49 | LED | 9575000 |
| LG 49SM8100 | LG | 49 | LED | 9499000 |
| NIKO K-50 Omega | NIKO | 50 | LED | 3129000 |
| Sanken SL-40SN1950 | Sanken | 40 | LED | 1999000 |

| PRODUK | MEREK | UKURAN | TIPE | HARGA |
|---|---|---|---|---|
| Mito 2922 | Mito | 22 | LED | 2300000 |
| Samsung UA65NU8000 | Samsung | 65 | LED | 19300000 |
| Samsung UA49N5000 | Samsung | 49 | LED | 4490000 |
| Sony KDL-50W660F | Sony | 50 | LCD | 5799000 |
| Toshiba 49L3750 | Toshiba | 49 | LED | 4500000 |
| Toshiba 55L3750 | Toshiba | 55 | LED | 6090000 |
| NIKO NK-39BETA | NIKO | 39 | LED | 1799000 |
| Samsung UA82NU8000 | Samsung | 82 | QLED | 58900000 |
| Polytron PLD 43D1500 | Polytron | 43 | LED | 3099000 |
| Polytron PLD 43S153 | Polytron | 43 | LED | 3299000 |
| Polytron PLD 40Y853 | Polytron | 40 | LED | 3699000 |
| Toshiba 49L3650VJ | Toshiba | 49 | LED | 5099000 |
| Mito 3962E | Mito | 40 | LED | 1999000 |
| Samsung QA55Q60R | Samsung | 55 | QLED | 13250000 |
| Samsung QA75Q60R | Samsung | 75 | QLED | 41500000 |
| Samsung QA75Q70R | Samsung | 75 | QLED | 62200000 |
| Samsung UA55RU7300KPXD | Samsung | 55 | LED | 8350000 |
| Samsung UA55RU7400KPXD | Samsung | 55 | LED | 8500000 |

**Lampiran 2** *Syntax Scraping Web* dengan *BeautifulSoup*

```
from bs4 import BeautifulSoup
import requests
import pandas as pd
from string import *
import numpy
import re
from urllib.request import urlopen

print("Masukkan keyword : ")
keyword = input(" > ")
url = requests.get("https://www.pricebook.co.id/"+keyword)
url = url.text
url_base_page=
"https://www.pricebook.co.id/"+keyword+"?price=100000---
9999999999&page="
url_page = [url]
page_dicari = input (" Total Pencarian Page :  ")
page_dicari = int(page_dicari)

nama_produk = []
merek_produk = []
ukuran_produk = []
tipe _layar = []
harga_produk = []


for i in range(1,page_dicari) :
      i += 1
      prhalaman = requests.get(url_base_page + str(i))
      prhalaman = prhalaman.text
      url_page.append(prhalaman)

for halaman in url_page :
      soup = BeautifulSoup(halaman, "lxml")
      produk = soup.find_all("div", "feat-prod")
      for p in produk:
            nama = p.find("a", "link-inherit").get_text()
            merek            =            p.find('div','category-
breadcrumb').contents[1].contents[1].get_text()   #nanti  di  child
pas dimasukkin
            UnJ = p.find_all("span","pb-primary-specification")
            ukuran = UnJ[1].get_text()
            tipe = UnJ[0].get_text()
                  #ukuran     =     p.find("ul",     "prod-bycat-
spec").contents[2].find("div","inner_tittle_category_second").cont
ents[2].get_text() #cari childnya yang pertama
                  #     tipe     =     p.find("ul",     "prod-bycat-
spec").contents[1].find("div","inner_tittle_category_second").cont
ents[2].get_text()#cari childnya yg kedua
                  harga = p.find("h4", "title-price").get_text()
                  harga      =      re.sub("([\(\[]).*?([\)\]])",
"\g<1>\g<2>", harga)
                  harga = re.sub('[^0-9]','',harga)

                  ukuran      =      re.sub("([\(\[]).*?([\)\]])",
"\g<1>\g<2>", ukuran)
                  ukuran = re.sub('[^0-9]','',ukuran)
```

```
                ukuran = ukuran.replace("\n","")
                if ukuran == "" :
                      continue
                tipe = tipe.replace("Tipe :","")
                tipe = tipe.replace(" ","")
                if len(tipe) == 3 :
                      continue
                tipe = tipe.replace("\xa0","")
          nama_produk.append(nama.replace("\n","").encode('utf-
          8').decode('utf-8'))
          merek_produk.append(merek.replace("\n","").encode('utf
          -8').decode('utf-8'))
          ukuran_produk.append(int(ukuran.encode('utf-
          8').decode('utf-8')))
          tipe  _layar.append(tipe.replace("\n","").encode('utf-
          8').decode('utf-8'))
          harga_produk.append(harga.replace("\n","").encode('utf
          -8').decode('utf-8'))
##result disimpan pada file csv
produk_dict                                                    =
{"produk":nama_produk,"merek":merek_produk,"ukuran":ukuran_produk,
"tipe": tipe _layar, "harga":harga_produk,}
df = pd.DataFrame(produk_dict,columns = ["produk","merek","ukuran",
"tipe", "harga"])
df.sort_values('harga')
df.to_csv("dataTVnewfix.csv",sep=",")
```

**Lampiran 3** *Syntax Random Forest* dengan Bahasa *Python*

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

#load dataset
dataTV = pd.read_csv("C://SkripsiTV//dataTVnewfix.csv", sep=";")
dataTV.head()
dataTV.info()

#cek variabel
dataTV["merek"].value_counts()
dataTV["ukuran"].value_counts()
dataTV["tipe"].value_counts()
dataTV["harga"].describe()
dataTV.nunique()

#Konversi menjadi numerik
dataTV["produk"] = dataTV["produk"].astype('category')
dataTV["merek"] = dataTV["merek"].astype('category')
dataTV["tipe"] = dataTV["tipe"].astype('category')
dataTV.info()

def get_cat(col):
    return    dict([(cat,    code)    for    code,    cat    in
enumerate(col.cat.categories)])
produk_map = get_cat(dataTV.produk)
merek_map = get_cat(dataTV.merek)
tipe_map = get_cat(dataTV.tipe)
dataTV.produk = dataTV.produk.cat.codes
dataTV.merek = dataTV.merek.cat.codes
dataTV.tipe = dataTV.tipe.cat.codes
dataTV.head()

#Pembagian x dan y
y = dataTV['harga']
x = dataTV.drop(['harga'], axis=1)
x.head()

#pembagian data training dan data testing
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y,
test_size=0.25)

#model random forest (parameter default)
from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor()
model.fit(X_train, y_train)

#Akurasi model random forest (parameter default)
from sklearn import metrics
y_pred = model.predict(X_test)
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test,
y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test,
y_pred))
```

```
print('Root Mean Squared Error:',
np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
print('MAPE :', np.mean(np.abs((y_test - y_pred) / y_test))*100)
print ('Nilai akurasi :', (1-(np.mean(np.abs((y_test - y_pred) /
y_test))))*100)

#Tuning Hyperparameter
from sklearn.model_selection import RandomizedSearchCV
from pprint import pprint
### n_estimators (n_tree)
n_estimators = [int(x) for x in np.linspace(start = 100, stop =
1000, num = 10)]
### max_features (m_try)
max_features = [1, 2, 3, 4]
### Create the random grid
random_grid = {'n_estimators': n_estimators,
               'max_features': max_features}
pprint(random_grid)
###spesifikasi dari sampel parameter Tuning hyperparameter
param_dist = {'max_features': [1, 2, 3, 4],
              'n_estimators': [100, 200, 300, 400, 500, 600, 700,
800, 900, 1000]}
###Random Search
random_search = RandomizedSearchCV(model,param_dist,cv=10)
random_search.fit(X_train, y_train)
### Menentukan parameter optimal hasil tuning hyperparameter
print(random_search.best_params_)

#model   random   forest   (parameter   optimal   hasil   tuning
hyperparameter)
from sklearn.ensemble import RandomForestRegressor
model2 = RandomForestRegressor(n_estimators= 1.000, max_features=
4)
model2.fit(X_train, y_train)

#Akurasi model random forest (parameter optimal hasil tuning
hyperparameter)
from sklearn import metrics
y_pred = model2.predict(X_test)
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test,
y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test,
y_pred))
print('Root Mean Squared Error:',
np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
print('MAPE :', np.mean(np.abs((y_test - y_pred) / y_test))*100)
print ('Nilai akurasi :', (1-(np.mean(np.abs((y_test - y_pred) /
y_test))))*100)

import pickle
# Saving model pickle
pickle.dump(model2, open('modelRF.pkl','wb'))
import joblib
joblib.dump(produk_map, 'produk_map.pkl')
joblib.dump(merek_map, 'merek_map.pkl')
joblib.dump(tipe_map, 'tipe_map.pkl')
kode = []
produk = []
for i in produk_map:
```

```
        kode.append(produk_map[i])
        produk.append(i)
dict_produk  = {"kode":kode, "produk":produk}
kode = []
merek = []
for i in merek_map:
        kode.append(merek_map[i])
        merek.append(i)
dict_merek  = {"kode":kode, "merek":merek}
kode = []
tipe = []
for i in tipe_map:
        kode.append(tipe_map[i])
        tipe.append(i)
dict_tipe  = {"kode":kode, "tipe":tipe}
import joblib
joblib.dump(dict_produk, 'produk_map.pkl')
joblib.dump(dict_merek, 'merek_map.pkl')
joblib.dump(dict_tipe, 'tipe_map.pkl')

#variable importances
importances = model2.feature_importances_
std = np.std([tree.feature_importances_ for tree in
model2.estimators_], axis=0)
indices = np.argsort(importances)[::-1]
print("Feature ranking:")
for f in range(x.shape[1]):
    print("%d. feature %d (%f)" % (f + 1, indices[f],
importances[indices[f]]))
```

**Lampiran 4** *Syntax Framework Flask*

```
from flask import Flask, render_template, Response, jsonify,
request
import os
import pandas as pd
import pickle
import joblib
import json

APP_ROOT = os.path.dirname(os.path.abspath(__file__))
app = Flask(__name__)

harga = 0
model = pickle.load(open('static/model/modelRF.pkl','rb'))
produk_map = joblib.load('static/var_cat/produk_map.pkl')
merek_map = joblib.load('static/var_cat/merek_map.pkl')
tipe_map = joblib.load('static/var_cat/tipe_map.pkl')

def predict_harga(produk, merek, ukuran, tipe):
    data   =   pd.DataFrame({"produk":[produk],   "merek":[merek],
"ukuran":[ukuran], "tipe":[tipe]})
    harga = model.predict(data)
    return harga

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/predict', methods = ['GET','POST'])
def predict():
    def pred():
        if request.method == 'POST':
            produk = request.form['produk']
            merek = request.form['merek']
            tipe = request.form['tipe']
            ukuran = request.form['ukuran']
            global harga
            harga = predict_harga(produk, merek,ukuran, tipe)[0]
        print(harga)
        return "data:"+str(harga)+"\n\n"
    return Response(pred(), mimetype='text/event-stream')

@app.route('/harga')
def harga_():
    def get_harga():
        yield"data:"+str(harga)+"\n\n"
    return Response(get_harga(), mimetype='text/event-stream')

@app.route('/produk')
def produk():
    def load_produk():
        yield"data:"+str(produk_map)+"\n\n"
    return Response(load_produk(), mimetype='text/event-stream')
```

```
@app.route('/merek')
def merek():
    def load_merek():
        yield"data:"+str(merek_map)+"\n\n"
    return Response(load_merek(), mimetype='text/event-stream')

@app.route('/tipe')
def tipe():
    def load_tipe():
        yield"data:"+str(tipe_map)+"\n\n"
    return Response(load_tipe(), mimetype='text/event-stream')

if __name__ == '__main__':
    app.run(host='localhost', debug=True, threaded=True)
```

**Lampiran 5** *Syntax File HTML*

```
<!DOCTYPE html>
<html lang="en">

<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta   name="viewport"   content="width=device-width,   initial-
scale=1, shrink-to-fit=no">
  <title>Predict Harga Tv</title>
  <!-- plugins:css -->
  <link rel="stylesheet"
href="../../static/assets/vendors/mdi/css/materialdesignicons.min.
css">
  <link rel="stylesheet"
href="../../static/assets/vendors/css/vendor.bundle.base.css">
  <!-- endinject -->
  <!-- Plugin css for this page -->
  <!-- End plugin css for this page -->
  <!-- inject:css -->
  <!-- endinject -->
  <!-- Layout styles -->
  <link rel="stylesheet" href="../../static/assets/css/style.css">
  <!-- End layout styles -->
  <link                    rel="shortcut                        icon"
href="../../static/assets/images/favicon.png" />
</head>
<script>
  var source_produk = new EventSource('/produk');
  source_produk.onmessage = function (event) {
    let produk = event.data;
    produk_str = produk.replace(/'/g, '"');
    produk_json = JSON.parse(produk_str);
    var kode_produk = produk_json.kode;
    var nama_produk = produk_json.produk;
    source_produk.close();

    for (let i = 0; i < kode_produk.length; i++) {
      const kode = kode_produk[i];
      const produk = nama_produk[i];
      produk_input(kode, produk);
    }

    function produk_input(kode, produk) {
      var markup = "<option value = " + kode + ">" + produk +
"</option>";
      $('#produk_form_ex').append(markup);
    }
  }
</script>

<script>
  var source_merek = new EventSource('/merek');
  source_merek.onmessage = function (event) {
    let merek = event.data;
    merek_str = merek.replace(/'/g, '"');
    merek_json = JSON.parse(merek_str);
    var kode_merek = merek_json.kode;
```

```
    var nama_merek = merek_json.merek;
    source_merek.close();

    for (let i = 0; i < kode_merek.length; i++) {
      const kode = kode_merek[i];
      const merek = nama_merek[i];
      merek_input(kode, merek);
    }

    function merek_input(kode, merek) {
      var markup = "<option value = " + kode + ">" + merek +
"</option>";
      $('#merek_form').append(markup);
    }
  }
</script>

<script>
  var source_tipe = new EventSource('/tipe');
  var count = 0;
  source_tipe.onmessage = function (event) {
    let tipe = event.data;
    tipe_str = tipe.replace(/'/g, '"');
    tipe_json = JSON.parse(tipe_str);
    var kode_tipe = tipe_json.kode;
    var nama_tipe = tipe_json.tipe;
    source_tipe.close();

    for (let i = 0; i < kode_tipe.length; i++) {
      const kode = kode_tipe[i];
      const tipe = nama_tipe[i];
      merek_input(kode, tipe);
    }

    function merek_input(kode, tipe) {
      var markup = "<option value = " + kode + ">" + tipe +
"</option>";
      $('#tipe_form').append(markup);
    }
  }
</script>

<script>
  function kirim_form() {
      count = 1;
    document.getElementById('loader').style.visibility          =
'visible';
    var produk = document.getElementById('produk_form_ex').value;
    var merek = document.getElementById('merek_form').value;
    var tipe = document.getElementById('tipe_form').value;
    var ukuran = document.getElementById('ukuran_form').value;
    console.log(produk, merek, tipe, ukuran);
    const data = { produk: produk, merek: merek, tipe: tipe, ukuran:
ukuran };
    url = '/predict';
    $.post(url, data);
  }
  function clear_form() {
    $('#kode_produk').val('');
```

```
    $('#merek_form').val('');
    $('#tipe_form').val('');
    $('#ukuran_form').val('');
    $('#nama_produk_form').val('');
      $('#hasil_predict').val('');
  }
</script>
<script>
  var source_harga = new EventSource('/harga');
  source_harga.onmessage = function (event) {
    let harga = event.data;
    console.log(harga);
      if (count == 1) {
    $('#hasil_predict').val(harga);
    document.getElementById('loader').style.visibility = 'hidden';
      count--;
      }
  }
</script>

<script>
   function stop_check(){
            source_harga.close();
   }
</script>

<body>
        <div class="content-wrapper">
          <div class="row">
            <div class="col-12 grid-margin stretch-card">
              <div class="card">
                <div class="card-body">
                  <h4 class="card-title">Predict Harga Tv</h4>
                    <p   class="card-description">  Form   Informasi
Produk </p>
                  <form class="forms-sample">

                    <!-- jenis produk -->
                    <div class="form-group">
                      <label>Produk</label>
                      <select name="produk" required class="form-
control" id="produk_form_ex">
                        <option          value=""          disabled
selected>Produk</option>
                      </select>
                    </div>

                    <!-- merek -->
                    <div class="form-group">
                      <label
for="exampleSelectGender">Merek</label>
                      <select  name="merek"  required  class="form-
control" id="merek_form">
                        <option          value=""          disabled
selected>Merek</option>
                      </select>
                    </div>

                    <!-- tipe -->
```

```
                    <div class="form-group">
                      <label
for="exampleSelectGender">Tipe</label>
                      <select  name="tipe"  required  class="form-
control" id="tipe_form">
                        <option          value=""          disabled
selected>Tipe</option>
                      </select>
                    </div>

                    <!-- ukuran -->
                    <div class="form-group">
                      <label>Ukuran</label>
                      <div class="input-group col-xs-12">
                        <input  name="ukuran"  required  type="text"
id="ukuran_form" class="form-control"
                          placeholder="Ukuran dalam Inch">
                        <div class="input-group-append">
                          <span                class="input-group-
text">Inch</span>
                        </div>
                      </div>
                    </div>

                    <!-- btn -->

                  </form>
                  <button  onclick="kirim_form()"  class="btn  btn-
gradient-primary mr-2">Submit</button>
                  <button  onclick="clear_form()"  class="btn  btn-
light">Clear</button>


                </div>
                <div class="card-body">

                  <h4 class="card-title">Hasil Predict Harga Tv :
<div id="loader" style="visibility: hidden"
                      class="spinner-border text-primary"></div>
                  </h4>

                  <div class="form-group">
                    <label>Harga</label>
                    <div class="input-group col-xs-12">
                      <div class="input-group-prepend">
                        <span class="input-group-text bg-gradient-
primary text-white">Rp. </span>
                      </div>
                      <input          readonly          type="text"
id="hasil_predict" class="form-control"
                        aria-label="Amount    (to    the    nearest
dollar)">
                      <div class="input-group-append">
                        <span class="input-group-text">.00</span>
                      </div>
                    </div>
                  </div>
                </div>
              </div>
```

```
            </div>
          </div>
        </div>
        <!-- content-wrapper ends -->
        <!-- partial:../../partials/_footer.html -->
        <!-- partial -->
        <!-- container-scroller -->
        <!-- plugins:js -->
  <script
src="../../static/assets/vendors/js/vendor.bundle.base.js"></scrip
t>
  <!-- endinject -->
  <!-- Plugin js for this page -->
  <!-- End plugin js for this page -->
  <!-- inject:js -->
  <script src="../../static/assets/js/off-canvas.js"></script>
  <script                    src="../../static/assets/js/hoverable-
collapse.js"></script>
  <script src="../../static/assets/js/misc.js"></script>
  <!-- endinject -->
  <!-- Custom js for this page -->
  <script src="../../static/assets/js/file-upload.js"></script>
  <!-- End custom js for this page -->
</body>
</html>
```

## DAFTAR PUSTAKA


Adnyana, I. (2016). *PREDIKSI LAMA STUDI MAHASISWA DENGAN METODE RANDOM FOREST (STUDI KASUS : STIKOM BALI)*. *Csrid Journal*, Vol. 8 No. 3.

Alpaydin, E. (2010). *Introduction to Machine Learning.* The MIT Press.

Ardianto, E. (2007). *Komunikasi Massa Suatu Pengantar*. Bandung: Simbiosa Rekatama.

Azizah. (2019). *Masyarakat tak Lagi Menonton TV, Benarkah?*. Retrieved from Republika.co.id : https://www.republika.co.id/berita/gaya-hidup/trend/py4k9l463/trendtek/internet/19/09/20/py3y1f463-masyarakat-tak-lagi-menonton-tv-benarkah.

Bernard, S., Heutte, B., & Adam, S. (2009). *Influence of hyperparameters on random forest accuracy. Lecture Notes in Computer Science* (pp. 171-180). Springer.

Breiman, L. (1996). Bagging Predictors. *Machine Learning*, 123-140.

Breiman, L. (2001). Random Forest. *Machine Learning*, 5-32.

Dewi , N., & Syafitri , U. (2011). *PENERAPAN METODE RANDOM FOREST DALAM DRIVER ANALYSIS*. Vol. 16 No. 1.

Dhawangkhara , M., & Riksakomara , E. (2017). *Prediksi Intensitas Hujan Kota Surabaya dengan Matlab menggunakan Teknik Random Forest dan CART (Studi Kasus Kota Surabaya)*. Vol. 6, No. 1.

Firmani, A. N. (2016). *PENYELESAIAN REGRESI SEMIPARAMETRIK DENGAN MENGGUNAKAN REGRESI RANDOM FOREST*. Yogyakarta: Universitas Gadjah Mada.

Hadi, D. A. (2016). *Ebook Belajar HTML dan CSS Dasar*. Retrieved January 18, 2020, from http://malasngoding.com

Han, J. (2001). *Data Mining : Concept and Techniques.* California: Morgan Kaufmann.

Hanjura , A. (2014). *Heroku Cloud Application Development.* Birmingham: Packt.

Haristu, R. A. (2019). *Penerapan Metode Random Forest Untu k Prediksi Win Ratio Pemain Player Unknown Battleground*. Yogyakarta: Universitas Sanata Dharma.

Heriyanti, A. (2018, Maret 9). *Penjualan Elektronik via Online Melonjak*. Retrieved from Kontan.co.id: https://industri.kontan.co.id/news/penjualan-elektronik-via-online-melonjak

Hermawati, F. (2013). *Data Mining.* Yogyakarta: Andi Offset.

J.B, Wahyudi. (1983). *Jurnalistik Radio dan Televisi.* Jakarta: PT Pustaka Utama.

Lewis, M. D., & J, R. (2000). *An Introduction to Classification and Regression Trees (CART) Analysis.* California: UCLA Medical Center.

Liaw, A., & Wiener, M. (2002). Classification and regression by Random Forest. *R News*, 18-22.

Putra, A. (2014. *KLASIFIKASI TULISAN TANGAN BERUPA ANGKA MENGGUNAKAN RANDOM FOREST DAN HISTOGRAM OF ORIENTED GRADIENT*. *e-Proceeding of Engineering*, Vol.1, No.1, Page 738.

Putra, M. (2019). *SISTEM REKOMENDASI KELAYAKAN KREDIT MENGGUNAKAN METODE RANDOM FOREST PADA BRI KANTOR CABANG PELAIHARI*. Surabaya: Universitas Islam Negeri Sunan Ampel.

Russel, S., & Norvig, P. (1995). *Artificial Intelligence: A Modern Approach.* Prentice Hall International.

Samudra, A. (2019). *Pendekatan Random Forest Untuk Model Peramalan Harga Tembakau Rajangan Di Kabupaten Temanggung.* Yogyakarta: Universitas Sanata Dharma.

Samuel, A. (1959). Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal 3*, 535-554.

Schneider, J. (1997, Februari 7). *Cross Validation.* Retrieved from https://www.cs.cmu.edu/~schneide/tut5/node42.html

Siburian, V., & Mulyana, I. (2018). *Prediksi Harga Ponsel Menggunakan Metode Random Forest.* Palembang: Universitas Sriwijaya.

Sugiyono. (2009). *Metode Penelitian Kuantitatif, Kualitatif dan R&D.* Bandung: Alfabeta.

Sutisno. (1993). *Pedoman Praktis Penulisan Skenario Televisi Dan Video.* Jakarta: PT. Gramedia Widia Sarana Indonesia.

Turland, M. (2010). *Architect"s Guide to Web Scraping with PHP.* Toronto: Marco Tabini & Associates, Inc.

Ulwan, N. (2016). *Pattern Recognition Pada Unstructured Data Teks Menggunakan Support Vector Machine Dan Association.* Yogyakarta: Universitas Islam Indonesia.

Walpole, R. (1992). *Pengantar Statistika Edisi ke-3.* Gramedia Pustaka.

Wiryanto. (2004). *Pengantar Ilmu Komunikasi.* Jakarta: PT. Gramedia Widasarana.

Yahya, S. (2018). *Metode Support Vector Machine dan Random Forest (Studi Kasus: Data Lama Studi Alumni Universitas Islam Indonesia Tahun Kelulusan 2000-2017).* Yogyakarta: Universitas Islam Indonesia.