PREDIKSI TREN PERGERAKAN HARGA SAHAM MENGGUNAKAN ALGORITMA *TEMPORAL CONVOLUTIONAL NETWORK* (TCN)

Disusun Oleh:

N a m a : Harry Akbar Al Hakim

NIM : 17523229

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2021

HALAMAN PENGESAHAN DOSEN PEMBIMBING

PREDIKSI TREN PERGERAKAN HARGA SAHAM MENGGUNAKAN ALGORITMA *TEMPORAL CONVOLUTIONAL NETWORK* (TCN)

TUGAS AKHIR

Disusun Oleh:

N a m a : Harry Akbar Al Hakim

NIM : 17523229

Yogyakarta, 14 Juli 2021

Pembimbing,

(Dhomas Hatta Fudholi, S.T, M.Eng., Ph.D.)

HALAMAN PENGESAHAN DOSEN PENGUJI

PREDIKSI TREN PERGERAKAN HARGA SAHAM MENGGUNAKAN ALGORITMA *TEMPORAL CONVOLUTIONAL NETWORK* (TCN)

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 14 Juli 2021

Tim Penguji

Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D. _____

Anggota 1

Irving Vitra Paputungan, S.T., M.Sc., Ph.D. _____

Anggota 2

Aridhanyati Arifin, S.T., M.Cs. _____

Mengetahui,

Ketua Program Studi Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia

(Dr. Raden Teduh Dirgahayu, S.T., M.Sc.)

iv

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Harry Akbar Al Hakim

NIM : 17523229

Tugas akhir dengan judul:

PREDIKSI TREN PERGERAKAN HARGA SAHAM MENGGUNAKAN ALGORITMA *TEMPORAL CONVOLUTIONAL NETWORK* (TCN)

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila dikemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung resiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 14 Juli 2021

(Harry Akbar Al Hakim)

HALAMAN PERSEMBAHAN

Puji syukur bagi Allah SWT, dengan kesempatan dan rezeki yang diberikan dapat berkuliah di Jurusan Informatika Universitas Islam Indonesia dan dapat mencapai tahap akhir menyusun penelitian tugas akhir ini. Dengan segala proses dan perjuangan yang dihadapi, saya mengucapkan terima kasih kepada beberapa pihak yang yaitu Ibu saya tercinta, saudara-saudara saya yang selalu memberikan nasihat, sahabat-sahabat saya yang setia mendukung dikala senang dan duka, teman-teman seperjuangan, bapak Dhomas Hatta Fudholi, S.T, M.Eng., Ph.D. selaku dosen pembimbing, dan Jurusan Informatika Universitas Islam Indoesia dimana tempat saya menimba ilmu.

HALAMAN MOTO

“Kapan seseorang akan mati? Saat dia terkena tembakan? Tidak! Saat dia terkena racun mematikan? Tidak! Saat dia memakan makanan beracun? Tidak! Seseorang akan mati apabila dia telah dilupakan”

- Dr. Hiluluk (One Piece)

“History repeat it self”

- Wicky Zeroski (Trader Professional)

“Mimpi bukanlah sesuatu yang anda liaht saat tidur, melainkan sesuatu yang membuat anda tidak bisa tidur”

- Cristiano Ronaldo

“Barangsiapa keluar mencari ilmu, maka ia berada di jalan Allah hingga ia kembali”

- HR Tirmidzi

KATA PENGANTAR

Assalamualaikum Warahmatullai Wabarakatuh

Alhamdulillah, segala puji dan syukur saya panjatkan atas kehadirat Allah SWT, yang telah memberikan ridho dan rahmatnya. Sehingga penulis dapat menyelesaikan laporan tugas akhir ini yang berjudul "Prediksi Tren Pergerakan Harga Saham Menggunakan Algoritma *Temporal Convolutional Network* (TCN)" yang merupakan bagian dari ibadah saya kepada Allah SWT. Laporan ini disusun untuk memenuhi tugas akhir sebagai syarat untuk menyelesaikan pendidikan pada jenjang Strata 1 (S1) pada Jurusan Informatika Universitas Islam Indonesia. Penulis sadar laporan ini tidak akan dapat selesai dengan waktu yang cepat tanpa dukungan serta motivasi dari berbagai pihak. Oleh karena ini penulis tidak lupa menyampakan rasa terima kasih yang sebesar-besarnya kepada:

1. Allah SWT, atas limpahan rahmat dan hidayah-Nya yang selalu ada di setiap langkah dalam memberikan kekuatan, kemampuan, dan menjaga semangat untuk menuntut ilmu.
2. Ibu saya tercinta, Sri Rejeki, S.E. yang memberikan segala dukungan dan cinta kasihnya kepada saya.
3. Bapak Fathul Wahid, S.T., M.Sc., Ph.D. selaku Rektor Universitas Islam Indonesia
4. Bapak Prof. Dr. Ir. Hari Purnomo, M.T. selaku Dekan Fakultas Teknologi Industri Universitas Islam Indonesia.
5. Bapak Hendrik, S.T., M.Eng. selaku Ketua Jurusan Informatika Universitas Islam Indonesia.
6. Bapak Raden Teduh Dirgahayu, S.T., M.Sc., selaku Ketua Program Studi Informatika Program Sarjana Fakultas Teknologi Industri Universitas Islam Indonesia.
7. Bapak Dhomas Hatta Fudholi, S.T, M.Eng., Ph.D. selaku dosen pembimbing tugas akhir yang memberikan arahan dalam penyusunan laporan tugas akhir.
8. Bapak dan Ibu dosen Program Studi Informatika, yang telah memberikan ilmu yang telah mendidik serta memberika ilmunya selama saya menempuh studi.
9. Saudara, Sahabat, dan teman-teman saya yang selalu memberika dukungan disaat keadaan senang dan duka.

Semoga segala pihak yang memberikan bantuan, dukungan, bimbingan, serta doa yang diberikan kepada penulis mendapatkan kebaikan dari Allah SWT. Penulis memohon maaf apabila selama proses pengerjaan laporan tugas akhir terjadi kekhilafan dan kesalahan. Penulis

harapkan, laporan tugas akhir ini dapat menjadi bermanfaat kepada dunia pendidikan dan pengembangan bidang informatika dan teknologi.

Yogyakarta, 14 Juli 2021

(Harry Akbar Al Hakim)

SARI

Masyarakat semakin memahami akan pentingnya pengelolaan aset yang mereka miliki untuk yang digunakan untuk rencana masa depan ataupun melipatgandakan nilainya. Salah satu instrumen finansial yang populer di tengah masyarakat Indonesia adalah saham. Sedangkan pengertian saham sendiri menurut Bursa Efek Indonesia (BEI), saham adalah tanda penyertaan modal suatu badan usaha dalam sebuah perusahaan ataupun perseroan terbatas. Harga saham sanggatlah fluktuatif, dapat dipengaruhi oleh faktor *internal* dan *eksternal.* Untuk memprediksi tren pergerakan harga saham selain memperhatikan fundamental perusahaan dan membaca grafik harga saham, dapat memanfaatkan teknologi *artificial intelligence* terkhusus lagi *deep learning*. Pada penelitian ini digunakan algoritama *deep learning Temporal Convolutional Network* (TCN) yang menggunakan konsep konvolusi untuk memprediksi tren pergerakan harga saham memanfaatkan *library Darts* yang dikembangkan untuk mengolah data *time series*, data saham yang digunakan adalah saham dari Netflix, Inc (NFLX). Hasil yang didapatkan, TCN mampu memberikan hasil prediksi yang mengungguli beberapa algoritama lainnya seperti *Long Short Term Memory* (LSTM), *Linear Regression*, dan *Decision Tree Regression*. Dengan nilai perhitugnan *Root Mean Squared Error* (RMSE), TCN mendapatkan nilai 9.865, LSTM 10.710, *Lineaer Regression* 11.894, dan *Decision Tree Regression* 30.868. Dapat diketahui bahwa algoritma TCN cukup efektif untuk memprediksi tren pergerakan harga saham dan algoritmanya lebih ringkas dalam penggunaannya.

Kata kunci: saham, M*achine Learning, Deep Learning, Temporal Convolutional Network, Long Short Term Memory, Linear Regression, Decision Tree Regression.*

GLOSARIUM

Batch Size	banyaknya data yang dilatih dalam satu waktu
	langkah untuk menelusuri kesalahan kode program.
Dataset	sekelompok data yang digunakan pada pembentukan model
Epoch	banyaknya pengulangan pada pelatihan data
Pre-processing	tahapan menyiapkan data agar dapat diolah pada model algoritama

.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Akses informasi yang semakin mudah membuka jalan dunia finansial kian dikenal oleh masyarakat secara luas. Masyarkat semakin memahami akan pentingnya pengelolaan aset yang mereka miliki untuk yang digunakan untuk rencana masa depan ataupun melipatgandakan nilainya. Salah satu instrumen finansial yang populer di tengah masyarakat Indonesia adalah saham. Berdasarkan data yang dikeluarkan oleh Bursa Efek Indonesia (BEI) tercatat dalam kurun waktu satu tahun terakhir hingga Maret 2021, pertumbuhan investor baru di Indonesia mencapai 27% (investasi.kontan.co.id, 2021). Jika dilihat dari sisi demografi, investor baru tersebut didominasi oleh generasi muda usia di bawah 30 tahun dengan presentase 57,40% (CNBC Indonesia, 2021). Pertumbuhan yang cukup signifikan tersebut merupakan salah satu dampak positif akibat terjadinya pandemi Covid-19, masyarakat memindahkan asetnya ke instrumen investasi untuk menjaga nilainya. Walapun pada awal pandemi Indeks Harga Saham Gabungan (IHSG) yang sempat mengalami koreksi mecapai 32%. Selain itu, banyaknya perusahaan *fintech* yang menyediakan *platform* investasi semakin memudahkan masyarakat untuk memulai investasinya.

Tidak hanya di Indonesia saja, bursa saham Amerika Serikat juga mengalami dampak yang serupa akibat pandemi Covid-19. Kasus positif Covid-19 di Amerika Serikat sendiri merupakan yang paling tinggi di dunia mencapai 18 juta kasus pada akhir tahun 2020 berdasatkan data dari Universitas Johns Hopkins yang mana sangat berpengaruh pada keadaan banyak perusahaan besar yang sahamnya ada di bursa Amerika. Namun setelah mengalami koreksi pada awal pandemi, pada paruh kedua 2020 saham-saham yang ada di indeks S&P 500 dan Nasdaq mulai menguat melebihi harga sebelum mengalami koreksi yang dibarengi dengan perkembangan produksi vaksin Covid-19. Indeks S&P 500 ditutup naik 0,5% persen, sedangkan indeks Nasdaq yang berisi saham perusahaan teknologi ditutup naik 0,95%. Namun indeks Dow Jones yang berisi saham industri ditutup di bawah harga sebelum penurunan sebesar 0,3%. Perusahaan-perusahaan teknologi yang mendapatkan manfaat dari masyarakat yang harus tetap di rumah seperti Amazon, Netflix, dan FedEx memimpin pemulihan yang terjadi dan bahkan saham FedEx adalah saham yang mempunyai kinerja terbaik di indeks S&P 500 (Ika Ramadhani, 2020).

.Sedangkan pengertian saham sendiri menurut Bursa Efek Indonesia (BEI), saham adalah tanda penyertaan modal suatu badan usaha dalam sebuah perusahaan ataupun perseroan terbatas. Dengan penyertaan modal, maka pihak tersebut memiliki klaim atas pendapatan perusahaan, klaim atas aset perusahaan, dan berhak hadir dalam Rapat Umum Pemegang Saham (RUPS) (BEI, n.d.). Tanda penyteraan tersebut nantinya dapat diperjualbelikan di bursa saham seiring dengan terjadinya fluktuasi harganya untuk mendapatkan keuntungan dari *capital gain*. Ataupun didapatkan dari pembagian keuntungan yang dihasilkan oleh perusahaan, atau yang biasa disebut dengan *dividen*.

Namun seiring saham sebuah perusahaan diluncurkan ke publik dan diperjualbelikan di bursa, harga saham tersebut akan mengalami fluktuasi. Faktor *internal* dan *eksternal* dapat mempengaruhi harga saham tersebut (Zainuddin & Hartono, 1999). Faktor *internal* dapat dilihat dari kinerja perusahaan yang mengeluarkan saham tersebut, kinerja perusahaan dapat dilihat dari laporan keuangan perusahaan. Sedangkan faktor *eksternal* merupakan dampak dari lingkungan di luar perusahaan yang dapat mempengaruhi secara langsung maupun tidak langsung. Yang meliputi kondisi ekonomi dimana perusahaan tersebut berada, kondisi sosial serta politik negara tersebut. Selain itu investor harus memahami hukum permintaan serta penawaran yang terjadi di bursa saham, memahami hal tersebut dapat membantu investor untuk menentukan prediksi tren pergerakan harga saham (Kusumodestoni & Sarwido, 2017). Dengan mengetahui tren pergerakan harga saham, investor dapat memanfaatkannya untuk memaksimalkan keuntungan dan meminimalisir kerugian.

Untuk memprediksi tren pergerakan harga saham selain memperhatikan fundamental perusahaan dan membaca grafik harga saham, dapat memanfaatkan teknologi *artificial intelligence* terkhusus lagi *deep learning*. Deep learning sendiri merupakan turunan dari *machine learning*, yang mana merupakan bagian dari cabang ilmu *artificial intelligence*. Konsep dari *machine learning* adalah mencoba meniru bagaimana cara otak manusia bekerja, mesin dilatih menggunakan algoritma tertentu untuk belajar dari data-data yang kompleks dan banyak secara terus menerus dan berulang-ulang. Sehingga mesin mampu mendapatkan informasi ataupun pola dari data-data tersebut (Karno, 2020). Informasi yang telah didapatkan nantinya akan dimanfaatkan untuk prediksi pola dari data yang akan terjadi di masa depan pada data saham.

Pada pemanfaatannya, data yang dapat diolah pada model *deep learning* sangat beragam. Algoritma-algoritma yang diterapkan pada model *deep learning* juga harus disesuaikan dengan data yang digunakan. Pada data saham sendiri memiliki tipe data *time series*, dimana setiap

data dicatatkan berdasarkan waktu dan data tersebut memilikin keterikatan dengan data yang telah tercatat sebelumnya. Maka hasil dari prediksi akan digunakan untuk memperbaharui data yang digunakan untuk melakukan pembelajaran (Karno, 2020). Data dengan karakteristik tersebut dapat diolah menggunakan algoritma *Recurrent Neural Network* (RNN). Algoritma RNN dirancang untuk mengolah sekuensial dengan memanfaatkan memori untuk menyimpan data yang telah diolah sebelumnya yang memungkinkan untuk mengenali pola dari data dengan baik (Olah, 2015). Pada penelitian ini, akan mencoba mencari model algoritama *machine learning* ataupun *deep learning* untuk memprediksi tren pergerakan harga saham, terutama akan berfokus pada algoritma *Temporal Convolutional Network* (TCN). Berdasarkan penelitain yang dialukan oleh (Bai et al., 2018)algoritma TCN yang memanfaatkan proses konvolusi menunjukkan hasil yang lebih akurat daripada algoritama seperti LSTM dan GRU dalam mengolah data *sequential*, serta arsitektur yang dibentuk lebih sederhana dan jelas. Untuk data saham yang akan digunakan adalah data saham dari Netflix, Inc (NFLX), karena selama masa pandemi Covid-19 ini Netflix merupakan salah satu layanan yang terdampak positif. Dengan pendapatan perushaan mengalami peningkatan mencapai 24% pada akhir kuartal tahun 2020 menurut data dari (money.kompas.com, 2021), sehingga sahamnya menarik untuk dinalisis dan diprediksi.

1.2 Rumusan Masalah

Dari latar belakang yang telah dijabarkan sebelumnya, maka rumusan masalah adalah sebagai berikut:

a. Bagaimana membangun dan mengevaluasi algoritma *Temporal Convolutional Network* (TCN) dalam memprediksi tren pergerakan harga saham?
b. Membandingan performa algoritma Temporal Convolutional Network (TCN) dengan algoritma lain pada kasus prediksi pergerakan harga saham.

1.3 Tujuan Penelitian

Tujuan pencapaian pada penelitian ini adalah:

a. Mengetahui performa algoritma *Temporal Convolutional Network* (TCN) dan membandingkan dengan beberapa algoritma *machine learning* untuk mengetahui algoritma mana yang memberikan hasil maksimal pada prediksi tren pergerakan harga saham.

b. Membangun visualisasi tren pergerakan harga saham yang diprediksi oleh algoritma *Temporal Convolutional Network* (TCN) dan membandingakannya dengan harga saham aktual yang terjadi.
c. Mengetahui apakah prediksi yang dilakukan dengan algoritma *Temporal Convolutional Network* (TCN) dapat digunakan pada pengambilan keputusan investasi saham.

1.4 Manfaat Penelitian

Adapun manfaat yang diharapkan dari penelitian ini adalah:

a. Dengan diketahuinya performa algoritma *Temporal Convolutional Network* (TCN) dan algoritma lainnya, diharapkan algoritma yang mendapatkan performa baik dapat dimanfaatkan untuk prediksi tren pergerakan harga saham pada pasar saham .
b. Memberikan *insight* secara teknikal kepada investor maupun pelaku pasar yang diharapkan dapat digunakan sebagai referensi sebelum mengambil keputusan investasi saham.

1.5 Batasan Masalah

Batasan masalah yang digunakan pada penelitian ini adalah:

a. Model yang dikembangkan pada penelitian ini hanya untuk satu emiten saham yaitu saham dari Netflix, Inc (NFLX).
b. Prediksi yang dilakukan adalah selama 60 hari.
c. Perbandingan hasil prediksi berdasarkan nilai yang didapat dari *Root Mean Squared Error* (RMSE).

1.6 Metodologi Penelitian

Penelitian ini menggunakan metode penelitian sebagai berikut:

a. Studi literatur, tahapan pertama yang dilakukan adalah pencarian masalah yang ingin diangkat pada penelitian kali ini dengan cara pencarian melalui literatur-literatur penelitian yang telah dilakukan sebelumnya maupun dari masalah yang terjadi di sekitar. Studi literatur dilakukan dengan tujuan mengetahui *state of the art* penelitian-penelitian dengan topik bahasan sejenis.

b. Pengumpulan data, data yang akan digunakan merupakan data historis harga saham terpilih. Data didapatkan dari website *Yahoo Finance*, rentang data yang diambil dapat ditentukan sesuai dengan kebutuhan.
c. Pengolahan data, setelah data saham didapatkan selanjutnya data harus diolah terlebih dahulu dengan tujuan untuk menyesuaikannya dengan algoritma yang akan digunakan. Serta, melakukan pengecekan apakah data yang digunakan terdapat kejanggalan dan data dibagi menjadi data *training* dan *testing*.
d. Pemodelan algoritma, membangun arsitektur algoritma machine learning yang akan digunakan dan melakukan *hyperparameter tuning*.
e. Proses training dan testing, setelah model algoritma *machine learning* selesai dibangun kemudian data dimasukkan pada model dan dilakukan proses training. Model yang sudah selesai dilakukan proses *training* kemudian akan ditest performanya menggunakan data *testing*.
f. Evaluasi model, performa testing model akan dievaluasi menggunakan metrik pengukuran error yang terjadi. Selanjutnya akan dibandingkan serta dianalisis dengan performa dari model algoritma lainnya.

1.7 Sistematika Penulisan

Pada sistematika penulisan, tulisan dibagi ke dalam lima bab, yaitu sebagai berikut:

BAB IPendahuluan

Pada bab ini berisikan penjelasan mengapa penelitian ini dilakukan.

BAB II Landasan Teori

Pada bab ini berisikan literatur dari penelitian-penelitian yang sudah dilakukan sebelumnya pada tema yang diangkat pada penelitian ini, yang akan digunakan sebagai referensi.

BAB III Landasan Metodologi Penelitian

Pada bab ini berisikan tahap-tahap dalam melakukan penelitian, tahapan diuraikan secara mendetail.

BAB IV Hasil dan Pembahasan

Pada bab ini berisikan hasil dari penelitian yang telah dilakukan, permasalahan yang terjadi pada penelitian serta analisis dari hasil yang telah didapatkan.

BAB V Penutup

Pada bab ini berisikan kesimpulan dari penelitian yang telah dilakukan serta saran yang ditujukan untuk penelitian yang akan dilakukan ke depannya.

BAB II
LANDASAN TEORI

2.1 Analisa Saham

Sebelum melakukan investasi saham investor haruslah memahami instrumen investasi yang akan ingin diinvestasikannya. Banyak pertimbangan yang harus diambil secara matang oleh investor, agar mendapatkan perusahaan yang memiliki prospek baik dan harga saham terbaik (Filbert, 2020). Seperti yang diketahui pergerakan harga saham sangatlah *volatile*, oleh karena itu saham digolongkan ke dalam instrumen investasi dengan risiko yang sangat tinggi. Walaupun begitu sejalan dengan risiko yang ditanggung, potensi keuntungan yang didapatkan juga semakin besar. Dengan analisa yang tepat, potensi tersebut dapat dimanfaatkan oleh investor untuk mendulang keuntungan. Dalam menganalisis suatu saham, terdapat tiga cara yang biasanya dilakukan:

2.1.1 Analisa Fundamental

Analisa Fundamental merupakan analisa yang dilakukan dengan menitikberatkan pada laporan keuangan yang dikeluarkan oleh perusahaan yang melantai di bursa dengan melihat sisi: *Profitability* (laba), *Solvability* (kewajiban dan aset perusahaan), *Liquidity* (kemampuan melunasi hutang), dan *Activity* (kelancaran usaha) (Filbert, 2020a). Selain dari sisi internal perusahaan, dilihat juga dari sisi eksternal yang berhubungan dengan sektor perusahaan tersebut seperti kebijakan pemerintah, perubahan tingkat suku bunga, inflasi dan lain sebagainya. Hasil dari analisis tersebut menghasilkan penilaian apakah saham perusahaan tersebut layak untuk dibeli atau tidak. Jika kesimpulan yang didapatkan harga saham dianggap terlalu mahal dibandingkan harga wajarnya berdasarkan nilai perusahaannya, maka saham tersebut direkomendasikan untuk dijual daripada membelinya. Sementara jika yang terjadi sebaliknya, maka saham tersebut direkomendasikan untuk dibeli (Agusta Nanda et al., 2016). Penilaian tersebut berjangka waktu panjang, dikarenakan laporan keuangan perusahaan diteribitkan setiap kuartal ataupun tahunan. Maka penilaian akan diperbaharui setiap ada terbitan laporan keuangan baru. Namun permasalahannya seringkali investor kekurangan informasi karena hanya mengandalkan analisa fundamental yang mengakibatkan tidak tepatnya timing saat mengambil keputusan. Untuk itu investor dapat melihat pergerakan harga saham melalui analisa teknikal.

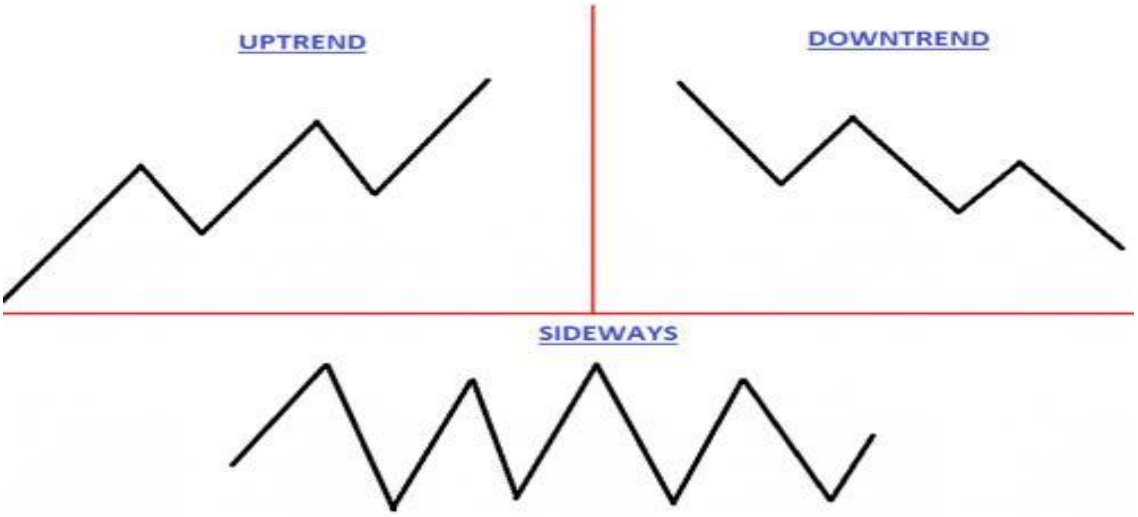
2.1.2 Analisa Teknikal

Analisa teknikal merupakan analisa pergerakan historis harga saham yang berupa harga pembukaan, harga penutupan, harga tertinggi, harga terendah, dan volume perdagangan pada masa tertentu yang disajikan dalam bentuk grafik (Filbert, 2020). Untuk mengamati grafik pergerakan harga saham tersebut dibutuhkan apliaksi *chartting* yang biasanya sudah disediakan oleh broker saham yang dipilih untuk melakukan investasi. Analisa teknikal lebih digunakan untuk mengemati tren harga yang sedang terjadi dan di dalam tren tersebut terdapat pola-pola harga yang terbentuk, pola-pola tersebutlah yang dimanfaatkan untuk memprediksi tren yang akan terjadi selanjutnya. Dengan demikian, analisa teknikal memberikan gambaran yang disederhanakan dan komprehensif tentang harga saham yang sedang terjadi (Suresh, 2013).

Analisa teknikal sendiri pertama kali dikemukakan oleh Charles Henry Dow pada tahun awal abad ke-19 yang kini namanya digunakan pada nama indeks saham industrial di Amerika yaitu *Dow Jones Industrial Average* (DJIA). Teorinya menjelaskan bahwa pergerakan harga saham 90% dipengaruhi oleh faktor psikologis dan 10% faktor logika. Harga ditentukan oleh para pelaku pasar itu sendiri, apakah sudah dianggap terlalu mahal atau murah. Teori dow hanya menjelaskan arah pergerakan tren saja dan bukan memprediksi harga masa depan. Teori ini menggunakan harga saham sebagai barometer kondisi bisnis suatu perusahaan, diasumsikan bahwa harga saham mengikuti bisnis yang mendasarinya (Suresh, 2013). Berikut beberapa informasi penting yang rerdapat pada analisa teknikal (Hafizah et al., 2019).

Tren Pasar

Tren merupakan kecenderungan pergerakan harga saham menuju ke arah tertentu pada suatu rentang waktu (Hafizah et al., 2019). Pada perdagangan saham terdapat tiga tren yaitu, *uptrend* (tren kenaikan) ketika permintaan saham tersebut lebih banyak dari penawaran maka harga akan mengalami kenaikan. *Downtrend* (tren penurunan) merupakan kebalikan dari tren kenaikan, ketika permintaan saham tersebut lebih sedikit daripada penawaran. Dan tren *sideways* (konsolidasi) yaitu ketika harga sedang mencari arah tren berikutnya yang biasa terjadi setelah tren kenaikan dan penurunan. Gambar 2.1 menunjukkan tren yang pada harga saham dalam bentuk grafik, sebuah tren dibentuk oleh fase kenaikan dan penurunan. Tren kenaikan terjadi ketika harga membentuk lembah dan puncak yang lebih tinggi. Tren penurunan ketika harga membentuk lembah dan puncak yang lebih rendah. Sedangkan tren konsolidasi membentuk harga yang cenderung datar.



Gambar 2.1 Tren pergerakan harga saham
Sumber: (Tri Retno, 2021)

Dalam pembentukan sebuah tren, dapat dibagi dalam empat fase dasar sehingga terbentuk sebuah pergerakan harga (Filbert, 2020a). Pada fase pertama adalah fase pendakian atau pembalikan, pada fase ini biasanya dibarengi dengan berita-berita yang tidak men-*support* situasi pasar. Fase kedua adalah fase kenaikan, investor mulai masuk dan mengakibatkan harga mulai mengalami kenaikan. Selanjutnya fase ketiga adalah fase jenuh, dimana harga dianggap sudah terlalu tinggi sehingga dianggap kurang menarik untuk dibeli. Yang terakhir adalah fase pembalikan, harga mengalami pembalikan arah dari tren semula. Fase-fase tersebut terjadi secara berulang pada setiap pergerakan harga dengan jangka waktu yang berbeda-beda.

Sejatinya tren pergerakan harga saham sendiri merupakan representasi dari pengambilan keputusan oleh investor saat melakukan perdagangan saham, dalam pengambilan keputusannya psikologi investor dipengaruhi oleh emosi "*fear and greed*" (R Tristia, 2019). Yaitu dimana ketika investor mengalami emosi *fear* atau ketakutan maka akan cenderung menjual asetnya karena takut harganya akan semakin jatuh sehingga menyebabkan harga yang terdapat di pasar akan semakin jatuh. Sebaliknya ketika investor terlalu serakah atau *greed* untuk membeli aset yang sedang mengalami kenaikan harga, sehingga harga aset akan semakin mengalami kenaikan dan tidak menutup kemungkinan akan membeli di harga yang sudah terlampau tinggi. Pengambilan keputusan oleh investor lebih bersifat subjektif, emosi dan faktor psikologis yang dapat mendominasi dalam pengambilan keputusan, jatuhnya harga saham seringkali terjadi karena reaksi pasar yang berlebihan (Lestari & Pranyoto, 2015).

Chart

Chart digunakan untuk merepresentasikan pergerakan harga saham yang membantu investor untuk mengamati secara visual. *Chart* disusun dalam urutan waktu pada sumbu X (sumbu horizontal) dan harga disusun pada sumbu Y (sumbu vertikal) (Hafizah et al., 2019). Ada beberapa *chart* yang digunakan dalam merepresentasikan harga saham yaitu (Sekar, 2020):

Yang pertama adalah *line chart*, disusun oleh garis tunggal yang menghubungkan antar harga yang terbentuk. *Line chart* sangatlah sederhana, hanya menampilkan informasi harga penutupan saja. Karena hal tersebut, *line chart* cocok digunakan untuk menyimpulkan kondisi tren harga saham sedang mengalami tren naik ataupun turun dan memudahkan menentukan titik-titik harga yang menjadi kunci. Gambar 2.2 menunjukkan harga saham NFLX menggunakan line chart.

Yang kedua adalah *bar chart*, disusun oleh batang yang disusun setiap interval waktu yang berisikan informasi mengenai harga pembukaan (*open*), harga penutupan (*close*), harga tertinggi (*high*), harga terendah (*low*). Jika harga penutupan lebih rendah daripada harga pembukaan maka batang akan digambarkan dengan warna merah. Sebaliknya jika harga penutupan lebih tinggi daripada harga pembukaan maka batang akan digambarkan dengan warna hijau. Pada *bar chart* menunjukkan informasi yang lebih banyak ketimbang *line chart*, *bar chart* dapat dimanfaatkan untuk mengetahui seberapa signifikan perubahan harga yang terjadi, semakin panjang batang yang dibentuk maka semakin signifikan perubahan yang terjadi. Gambar 2.3 menunjukkan harga saham NFLX menggunakan *bar chart*.

Yang terakhir adalah *candlestick chart*, memiliki bentuk seperti *bar chart* namun memiliki badan yang berbentuk seperti lilin dan memiliki tampilan yang lebih nyaman untuk digunakan. *Candlestick chart* merupakan *chart* yang paling banyak digunakan di bursa. Selain menampilkan harga, *candlestick chart* juga membentuk pola-pola *candle* yang dapat diidentifikasi sehingga memudahkan untuk membaca tren ataupun momentum yang terjadi. Gambar 2.4 menunjukkan harga saham NFLX menggunakan *candlestick chart*.

Gambar 2.2 *Line chart* pada saham NFLX

Sumber: (tradingview.com, 2021)

Gambar 2.3 *Bar chart* pada saham NFLX

Sumber: (tradingview.com, 2021)

Gambar 2.4 *Candlestick chart* pada saham NFLX

Sumber: (tradingview.com, 2021)

Support* dan *Resistance

Selanjutnya untuk mendapatkan informasi lebih dari *chart* dapat menggambarkan garis degan konsep *support* dan *resistance*. *Support* merupakan harga yang diyakini sebagai titik atau area terendah pada rentang waktu tertentu. Sedangkan *resistance* merupakan harga yang diyakini sebagai titik atau area tertinggi pada rentang waktu tertentu (Filbert, 2020b). Untuk menentukan titik atau area pada *support* dan *resistance* dapat menggunakan cara menghubungkan titik-titik ekstrim yang dibentuk oleh harga dalam bentuk garis.horizontal. Sebuah area *support* merupakan area *resistance* yang terbentuk di masa lalu, jika harga telah membentuk puncak baru melebihi area *resistance* tersebut maka akan berubah menjadi area *support* baru. Gambar 2.5 menunjukkan area-area yang menjadi *support* dan *resistance* penting pada harga saham NFLX.

Gambar 2.5 Area support dan resistance pada saham NFLX

Sumber: (tradingview.com, 2021)

Indikator

Selain menggunakan cara yang tradisional menganalisa pergerakan tren harga saham, dapat juga menerapkan analisis teknikal modern yang menafaatkan indikator harga saham. Indikator pada pergerakan harga saham merupakan alat bantu yang digunakan untuk membantu dalam analisa teknikal, yang memanfaatkan formula statistik pada harga saham untuk menentukan tren pergerakan (Hafizah et al., 2019). Terdapat dua jenis indikator, yang pertama adalah *trend following indicator* yang bertujuan mengikuti tren pergerakan harga saham tersebut. *Trend following indicator* terbentuk lebih lambat ketimbang pergerakan harga dan terletak menempel pada *chart*. Dan yang kedua adalah *oscillator indicator* yang menunjukkan level sebuah harga saham sudah mengalami keadaan jenuh beli (*over buy*) atau jenuh jual (*over bought*) (Pramudya & Pramudya, 2020). *Ocsillator indicator* sendiri merupakan indikator yang bergerak mendahului pergerakan harga dan terletak terpisah dengan *chart* (Ahmar, 2018).

2.1.3 *News*

Selain menganalisis secara fundamental dan tenikal, dapat juga memanfaatkan berita-berita atau sentimen-sentimen yang beredar terkait sektor industri ataupun perusahaan yang memiliki saham tersebut. Menganalisis berdasarkan berita-berita biasa dikatakan sebagai analisis sentimen. Saham sangatlah sensitif terhadap berita-berita yang beredar dikarenakan para investor dapat mengalami kepanikan atas sentimen yang terbentuk adalah negatif, sehingga harga saham cenderung mengalami banyak penjualan, hal tersebut biasa dikenal

dengan *Fear, Uncertainly, and Doubt* (FUD). Sebaliknya jika sentimen yang dihasilkan adalah positif, maka investor akan cenderung berminat untuk membeli.

2.2 Netflix

Netflix merupakan platform streaming berbayar asal Amerika Serikat yang menyediakan berbagai pilihan film dan *TV series*. Layanan tersebut dapat dinikmati melalui berbagai perangkat seperti *smart TV, smartphone, tablet, PC, game console*, dan *laptop*. Perusahaan Netflix, Inc. didirikan oleh Marc Randolph dan Reed Hastings pada 1997 di Los Galtos, California, Amerika Serikat. Pada mulanya Netflix menyediakan layanan penyewaan DVD secara fisik yang dapat dipesan secara online. Pada tahun berikutnya, Netflix mulai mengalihkan strategi bisnisnya menjadi sistem berlangganan untuk menikmati layanannya. Pelanggan membayar biaya langganan dan bebas menyewa berbagai DVD yang tersedia, dan juga Netflix mulai memperkenalkan sistem rekomendasi bagi para pelanggannya.

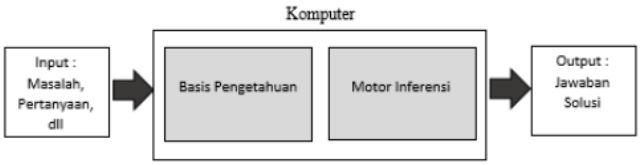
Setelah bisnisnya mulai cukup berkembang, pada tahun 2002 mengambil keputusan untuk melakukan *Initial Public Offering* (IPO) di bursa saham Amerika, *National Association of Securities Dealers Automated Quotation* (NASDAQ) dengan nama saham "NFLX". Netflix berhasil mendapatkan 4,2 juta pelanggan pada tahun 2005. Pada tahun 2007 Netflix mulai merilis platform layanan *streaming*-nya yang memungkinkan pelanggan dapat langsung menikmati layanan melalui komputer pribadinya secara langsung. Selain itu untuk melebarkan cakupan bisnisnya, Netflix mulai membuka berbagai kerja sama dengan perusahaan-perusahaan teknologi seperti Xbox, Apple, Playstation, Blue-ray disc player dengan tujuan agar layanannya dapat dinikmati di berbagai perangkat yang terhubung ke internet. Pada tahun 2013 Netflix mencatatkan memiliki 31 juga orang pelanggan. Tidak hanya di Amerika Serikat saja, layanan Netflix mulai bisa dinikmati di berbagai negara di dunia seiring berkembangnya internet, Di Indonesia sendiri, Netflix mulai masuk pada tahun 2016 (axa.co.id, 2019).

Menurut pendirinya Hastings, kesuksesan Netflix didasarkan bahwa para pelanggan sudah mulai tidak puas pada tayangan TV tradisional yang diakibatkan karena adanya iklan. Netflix tidak memiliki iklan yang mengganggu, karena layanannya berbayar. Pelanggan dapat menikmati tontonan apapun yang mereka inginkan kapanpun dan dimanapun tanpa harus mengikuti jadwal dan terganggu iklan. Kini, layanan *streaming* Netflix merupakan salah satu yang terdampak positif atas adanya pandemi Covid-19 yang sedang terjadi. Pandemi Covid-19 mengakibatkan masyarakat harus mengurangi aktivitasnya di luar ruangan dan banyak melalukan aktivitas di dalam ruangan. Hal tersebut mengakibatkan lonjakan pengguna baru

Netflix dikarenakan dibutuhkannya layanan hiburan untuk menemani masyarakat selama pandemi. Berdasarkan data (money.kompas.com, 2021) pelanggan baru Netflix tercatatat mengalami peningkatan sebesar 30% yaitu sebanyak 37 juga pengguna baru yang mayoritas berada di luar Amerika Serikat. Pendapatan perusahaan juga mengalami peningkatan sebesar 24% pada kuartal akhir tahun 2020, dicatatkan pendapatan sebesar 6,6 miliar *dollar* AS dengan laba sebesar 524 juta *dollar* AS.

2.3 *Artificial Intelligence*

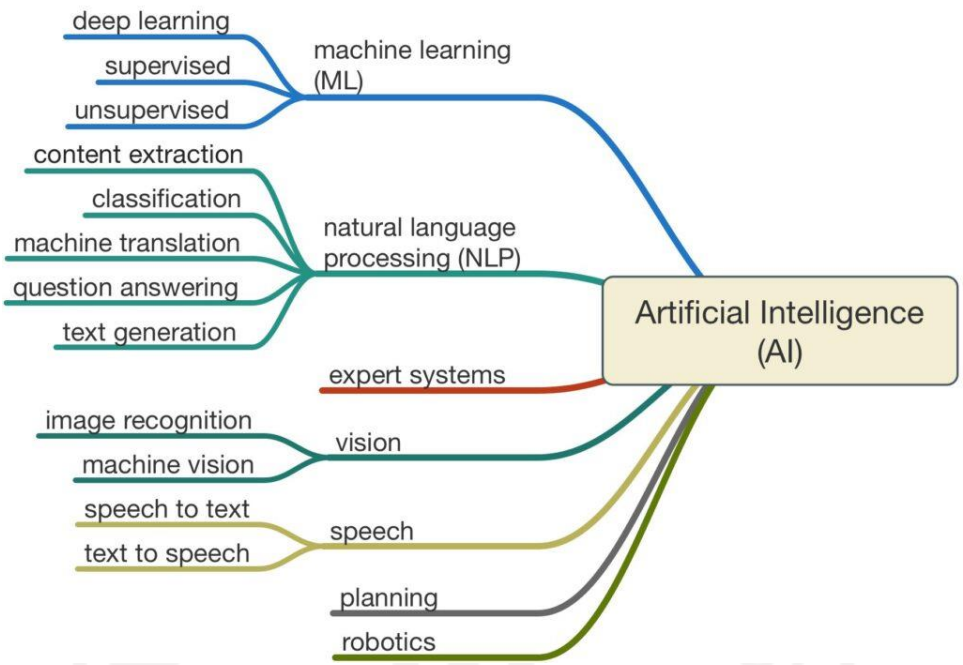
Artificial Intelligence (Kecerdasan Buatan) merupakan salah satu cabang dari ilmu komputer yang yang dikembangkan agar *software* dan *hardware* dapat berpikir berprilaku seperti halnya manusia (Roihan et al., 2019) bahkan dapat lebih baik dan mampu mengerjakan tugas-tugas yang tidak bisa dikerjakan oleh manusia. Sedangkan menurut (Rich et al., 2019) adalah bagaimana merancang komputer agar mampu melakukan hal-hal yang saat ini dikerjakan oleh manusia. Pada awalnya komputer digunakan untuk mengerjakan tugas komputasi saja, namun seiring berkembangnya teknologi diharapkan komputer dapat diperdayakan untuk mempermudah berbagai pekerjaan manusia. Dalam merancang sistem *artificial intelligence* terdapat dua hal yang menjadi bagian utama. Gambar 2.6 menunjukkan bagian dalam sistem *artificial intelligence*.



Gambar 2.6 Bagian utama pada sistem *artificial intelligence*
Sumber: (Nurhikmat, 2018)

a. *Knowledge Base* (Basis Pengetahuan), bagian ini berisikan data-data yang digunakan oleh sistem untuk belajar dan menjadikannya cerdas.
b. *Inference Engine* (Mesin Pengambil Keputusan), bagian yang memiliki kemampuan untuk mengambil keputusan ataupun kesimpulan berdasarkan basis pengetahuan yang dimiliki oleh sistem.

Artificial intelligence membutuhkan data untuk dijadikan basis pengetahuan seperti manusia. Data tersebut akan digunakan untuk pembelajaran, seiring semakin banyaknya pembelajaran sistem akan semakin cerdas. Poin penting dari *artificial ntelligence* adalah proses *learning*, *reasoning*, dan *self-correction* (Dicoding, 2020). Bidang ilmu *artificial intelligence* memiliki cakupan yang sangat luas. Gambar 2.7 Menunjukkan berbagai bagian dari *artificial intelligence*.



Gambar 2.7 Bagian-bagian pada *artificial intelligence*
Sumber: (Roziq, 2021)

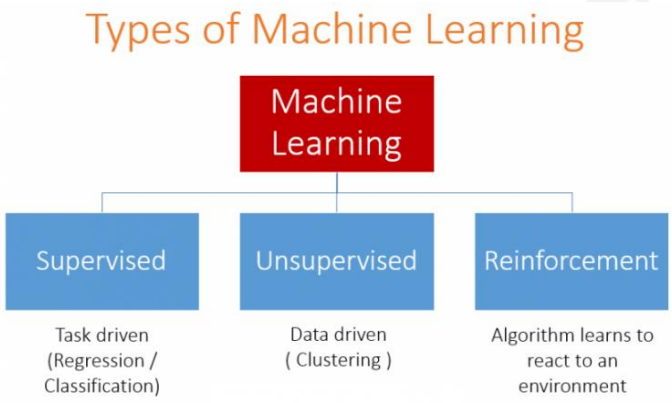
2.3.1 *Machine Learning*

Machine Learning (Pembelajaran Mesin) merupakan salah satu bagian dari bidang ilmu komputer *Artificial intelligence* yang dikembangkan agar *software* dan *hardware* dapat berpikir berprilaku seperti halnya manusia (Roihan et al., 2019). Untuk bisa mengembangkan sistem *Artificial intelligence* agar dapat menjalankan tugas seperti halnya manusia, aspek terpenting adalah bagaimana sistem tersebut dapat belajar sehingga mendapatkan informasi-informasi yang dibutuhkan untuk melakukan tugasnya.

Untuk itu dikembangkanlah *machine learning*, kosepnya adalah melatih mesin menggunakan model algoritma matematika melalui data-data masukan dengan tujuan memperoleh informasi (*knowledge discovery/discovery unknown structure*) dan memprediksi pola (*pattern discovery*) secara mandiri (Gotama, 2018). Proses pembelajaran mesin untuk

memperoleh kecerdasan melalui tahapan latihan (*training*) dan pengujian (*testing*) (Huang et al., 2006). Semakin sering mesin dilatih dan diuji, maka mesin akan semakin cerdas dan kemampuan untuk mengenali informasi jaga semakin akurat. Keilmuan yang digunakan pada pembelajaran mesin lebih mengarah ke matematika dan statistika. Seperti rumus matematika, pembelajaran mesin digunakan sesuai dengan permasalahan yang dihadapi dan data yang digunakan.

Machine learning sendiri dapat dibagi menjadi tiga kategori yaitu: S*upervised lLearning, Unsupervised Learning* dan *Reinforcement Learning* (Roihan et al., 2019). Gambar 2.6 menunjukkan pembagian kategori pada *machine learning*.



Gambar 2.8 Pembagian kategori pada *machine learning*

Sumber: (Mustofa, 2017)

a. *Supervised Learning*, merupakan pembelajaran yang dilakukan dengan memberikan label pada data sehingga dapat diklasifikasikan, seperti data pada data *input* di mana *output* yang diinginkan sudah diketahui. Tahapan pembelajaran yang dilakukan, mesin akan menerima serangkaikan data *input* dengan *output* yang sudah diketahui dan mempelajarinya kemudian akan membandingkannya dengan *output* yang aktual dengan prediksi yang dilakukan oleh mesin. *Supervised learning* kemudian akan menggunakan pola yang sudah ditemukan pada proses pembelajaran untuk memprediksi nilai ataupun label pada data baru (Samsudiney, 2019). S*upervised learning* dapat diklasifikasikan dalam bentuk klasifikasi dan regresi, klasifikasi terjadi ketika *output* yang dihasilkan berbentuk kategori sedangkan regresi ketika *output* yang dihasilkan adalah nilai riil (Brownlee, 2016).

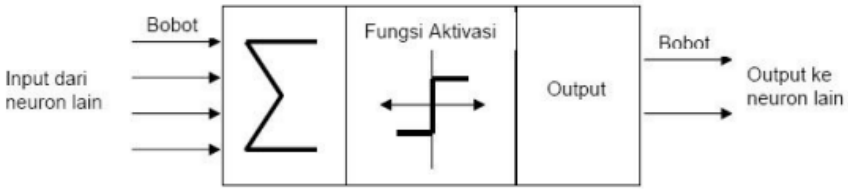
b. *Unsupervised Learning*, merupakan pembelajaran yang dilakukan tanpa adanya label pada data. Mesin tidak mengetahui mana output yang benar, tujuannya adalah untuk melakukan eksplorasi pada data (Samsudiney, 2019). *Unsupervised learning* dapat diklasifiaksikan dalam bentuk *clustering* dan asosiasi, *clustering* dilakukan untuk mengelompokkan data berdasarkan kesamaannya sedangkan asosiasi dilakukan untuk menggambarkan keterikatan antar data (Brownlee, 2016).
c. *Reinforcement Learning*, merupakan pembelajaran yang tidak memerlukan pengetahuan atau informasi sebelumnya. Mesin dapat belajar secara mandiri untuk mendapatkan informasi melalui proses "*trial and error*" secara terus menerus dalam lingkungan yang dinamis (Qiang & Zhongli, 2011).

2.3.2 *Deep Learning*

Deep learning (Pembelajaran Mendalam) atau *Deep Structured Learning* (Pembelajaran Struktural Mendalam) merupakan bidang turunan dari *machine learning* yang memanfaatkan *Artificial Neural Network* (Jaringan Syaraf Tiruan). Yaitu dengan konsep menambahkan banyak lapisan pada model sehingga proses belajar menjadi lebih kompleks dan mendalam. Kemudian yang membedakan dengan *machine learning* adalah data yang digunakan adalah data yang sangat banyak. Dengan begitu diperlukan pula proses komputasi yang sangat berat yang membutuhkan spesifikasi *hardware* yang mumpuni, biasanya digunakan *Graphics Processing Unit* (GPU). Dan juga *deep learning* memiliki fitur utama yaitu *feature engineering*, yaitu menurunkan tingkat kompleksitas data dengan mengekstak pola yang berguna untuk memudahkan model dalam membedakan kelas.

2.3.3 *Artificial Neural Network*

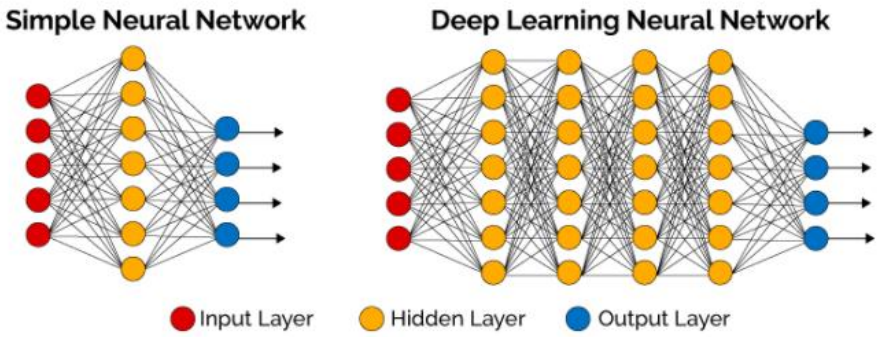
Seperti halnya jaringan syaraf pada otak dan sel saraf manusia, *artificial neural network* memiliki konsep neuron-neuron yang saling terhubung satu sama lain. *Neuron* akan menerima informasi dari data input kemudian disalurkan ke *neuron* lain hingga nantinya berakhir dengan hasil *output*. Pada neuron terdapat *Weight* (Bobot) dan fungsi aktivasi yang digunakan untuk menentukan sifat pada sebuah *neuron* (Nurhikmat, 2018). Menurut (Pham & Pham, 1999) *artificial neural network* merupakan sistem adaptif yang dapat memecahkan suatu masalah dengan mengubah strukturnya yang berdasarkan informasi internal maupun eksternal. Gambar 2.9 menunjukkan struktur dari sebuah *neuron*.



Gambar 2.9 Struktur dari *neuron*

Sumber: (Nurhikmat, 2018)

Cara kerja sebuah unit *neuron* seperti halnya jaringan syaraf manusia, *neuron* akan menerima data input berupa *independent variable* (X_1, X_2, X_3, …, X_n). Kemudian data *input p*ada *neuron* terdapat *weight* yang berupa (W_1, W_2, W_3, …, W_n) yang saling terhubung dengan setiap *node*. Kemudian data *input* tadi dan *weight* akan dilakukan proses matematis menggunakan sebuah fungsi aktivasi. Hasil dari proses tersebut selanjutnya akan dibandingan dengan suatu nilai *treshold (*ambang), jika hasil melebih nilai *treshold* maka *neuron* akan diaktifkan dan informasi dilanjutkan menuju *neuron* selanjutnya melalui *weight output*-nya dan begitu seterunya. Jika tidak, *neuron* tersebut tidak akan aktif dan informasi tidak dapat dilanjutkan. *Neuron* akan disusun pada sebuah *layer* (lapisan) yang berisikan banyak *neuron*. Setiap *layer* tersebut akan saling dihubungkan dengan *layer* lain sebelum dan sesudahnya, kecuali pada *input layer* dan *output layer*. Koneksi antar *layer* dihubungkan pada setiap *neuron*, jadi sebuah neuron akan terhubung pada seluruh *neuron* di *layer* sebelum dan setelahnya. Pada sebuah *layer ,weight* setiap *neuron* akan memiliki nilai yang sama. Struktur pada *artificial neural nerwork* dapat dibedakan menurut banyaknya *layer* yang terdapat pada jaringan. Gambar 2.10 Menunjukkan struktur pada *artificial neural network*.



Gambar 2.10 Struktur pada *artificial neural network*

Sumber : (Wayan, 2018)

Yang paling sederhana, terdapat struktur *single layer neural network* yang hanya memiliki lapisan tunggal yaitu *input layer* dan *output layer* saja. Setiap *neuron* pada *input layer* langsung terhubung dengan *output layer*, struktur jaringan ini hanya menerima *input* kemudian diproses langsung menjadi hasil *output*. Kemudian terdapat struktur *multiple layers neural network*, pada struktur ini terdapat *input layer*, *output layer*, dan *hidden layer*. Pada *hidden layer* inilah *layer* bisa disetel seberapa banyak tergantung kompleksitas masalah yang dihadapai. Algoritma yang menggunakan struktur seperti inilah yang dinamakan algoritma *deep learning*, karena layer digunakan sangat banyak sehingga proses pembelajaran akan semakin dalam dan kompleks.

2.4 *Temporal Convolutional Network* (TCN)

Penggunaan pendekatan dengan cara konvolusi sudah sangat banyak diterapkan pada proses prediksi dan pembuatan model *deep learning* menggunakan algoritma *Convolutional Neural Network* (CNN). Berbagai penelitian terkini juga sudah banyak menerapkan algoritma CNN dengan berbagai modifikasi untuk berbagai kebutuhan. Namun hal ini menimbulkan pertanyaan kepada para peneliti, apakah keberhasilan pemodelan menggunakan pendekatan konvolusi hanya terbatas pada domain tertentu saja atau juga dapat diterapkan pada domain *sequence processing* dan *recurrent network* juga (Bai et al., 2018). Pemodelan *sequence processing* dan *recurrent network* umumnya menggunakan algoritma yang memanfaatkan *Recurrent Neural Network* (RNN) seperti *Long Short Term Memory* (LSTM) dan *Gated Recurrent Unit* (GRU). Algoritma RNN merupakan salah satu pengembangan dari arsitektur algoritma ANN, dikembangkan khusus untuk mengatasi permasalah data yang memiliki karakteristik *sequential* (berurutan). Selain itu algoritma RNN juga memanfaatkan sebuah memori untuk menyimpan hasil dari masa lalu dalam proses pembelajarannya, yang memungkinkan model mengenali pola pada data lebih akurat (Yanuar, 2018).

Berbagai penelitian mencoba menggunakan pendekatan konvolusi untuk memproses permasalahan pada *sequential data*, kinerja yang didapatkan mampu mengunguli algoritma RNN dalam berbagai permasalahan dan dapat mengurangi masalah yang sering terjadi yaitu *exploding/vanishing gradient* (Lässig, 2020). *Vanishing gradient* (Hilangnya Gradien) merupakan permasalah yang timbul saat model melakukan proses perhitungan nilai turunan dari *weight* yang dimiliki oleh setiap *neuron* dengan cara *backpropagation* (Perhitungan mundur ke belakang). Semakin panjang layer yang dilalui kembali ke belakang maka nilai *weight* akan semakin mengecil dan mendekati 0, apalagi jika proses yang dilakukan berulang-

ulang (Herlambang, 2019). Untuk itulah diperkenalkan algoritma *Temporal Convolutional Network* (TCN) merupakan algoritma yang memanfaatkan proses konvolusi satu dimensi dan digunakan untuk memproses permasalahan pada data *sequential*. TCN memiliki karakteristik khusus (Bai et al., 2018).

a. Proses konvolusi pada arsitektur TCN bersifat kausal, artinya tidak ada "kebocoran" informasi dari masa depan ke masa lalu.
b. Arsitektur dapat mengolah berapa pun panjang data dan akan menghasilkan *output* yang sama panjangnya.
c. Kemampuan untuk melihat jauh ke masa lalu untuk melakukan prediksi, menggunakan arsitektur yang sangat dalam dan ditambah dengan *residual layers* dan *dilated convolution*.

2.4.1 *Sequence Modelling*

Sebelum membahas arsitektur dari *Temporal Convolutional Network* (TCN), harus dipahami tentang masalah pada pemodelan data yang berurutan atau biasa disebut *time series*. Dengan data *input* sebagai berikut X = ($\mathcal{X}_0$, $\mathcal{X}_1$, $\mathcal{X}_2$, …, $\mathcal{X}_T$) dan mencoba memprediksi output sebagai berikut Y = (y_0, y_1, y_2, …, y_T). Batasannya adalah bahwa untuk memprediksi *output* y_T, hanya dapat menggunakan data input yang telah diamati sebelumnya (Wan et al., 2019).

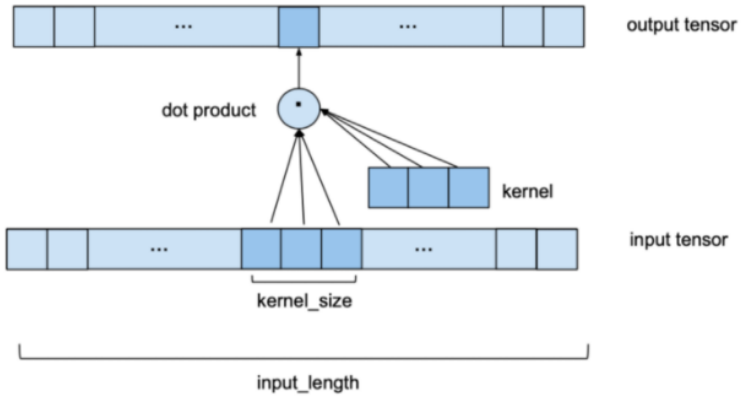
$$y_0 , y_1 , y_2 , \dots , y_T = f(x_0 , x_1 , x_2 , \dots , x_T) \quad (2.1)$$

Dapat diketahui bahwa y_T hanya bergantung pada X = ($\mathcal{X}_0$, $\mathcal{X}_1$, $\mathcal{X}_2$, …, $\mathcal{X}_T$) Bukan pada data *input* masa depan seperti $\mathcal{X}_{T+1}$. Tujuan dari *sequence modelling* adalah untuk mencari jaringan yang meminimalisir beberapa perbedaan antara data *output* aktual dengan hasil prediksi. Dimana target output kemudian digunakan menjadi data *input* untuk prediksi selanjutnya, dan terus bergeser menuju data selanjutnya oleh langkah waktu (Bai et al., 2018)

2.4.2 1 *Dimentional Convolutional Networks*

Proses 1 *dimensional convolutional network* merupakan proses konvolusi yang dilakukan pada data yang memiliki satu dimensi saja seperti *sequential data*. Bentuk data *input tensor* memiliki konfigurasi (*bartch_size, input_length, input_size*) dan nantinya akan menghasilkan data *output tensor* yang sama juga yaitu (*batch_size, input_length, output_size*). Karena setiap layer pada arsitektur TCN memiiki panjang *input* dan *output* yang sama, maka hanya *input* dan

output tensor saja yang berbeda. Karena hanya satu dimensi, maka nilai dari *input_size* dan *output s*ize adalah sama dengan satu. Pada gambar 2.11 menunjukkan bagaimana proses 1 *dimentional convolution* yang mengubah input menjadi *output*.



Gambar 2.11 Proses 1 *dimention convolution*
Sumber: (Lässig, 2020)

Setiap proses konvolusi akan berlansung di setiap data pada seluruh *input tensor* tersebut. Pada gambar, digunakan *kernel size* yaitu tiga, jadi proses akan dilakukan setiap tiga data. *Kernel* memiliki bobot yang sama pada setiap proses yang berjalan. Dapat dilihat bahwa setiap satu data pada output tensor merupakan hasil perhitungan *dot* dari *kernel* pada *input tensor*. Proses berlangsung sama dengan *kernel* akan mulai bergerak ke arah data selanjutnya setiap satu data. Namun untuk dapat memperoleh panjang data yang sama pada *input* dan *ouput tensor* perlu ditambahkan *padding* bernilai nol di awal dan akhir data.

2.4.3 *Causal Convolutions*

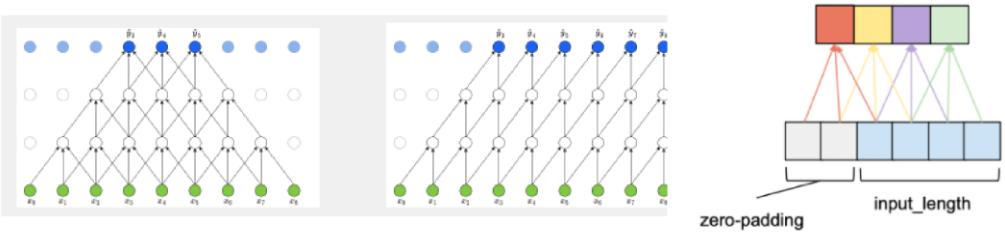
Disebutkan di awal bahwa TCN tidak bisa menggunakan informasi dari masa depan untuk melakukan proses konvolusi. Jadi untuk menghindari masalah tersebut, proses prediksinya menggunakan *causal convolution* yaitu hanya menggunakan data saat ini dan data yang sebelumnya (Liu et al., 2019). Hal tersebut dapat dicapai dengan cara menambahkan *padding* dengan nilai nol di awal dan akhir data *input tensor*, untuk menghitung jumlah *padding* yang digunakan rumus (Wang et al., 2020):

$$P = k - 1 \tag{2.2}$$

Dimana:

$\mathcal{P}$ = Jumlah *padding*

k = Ukuran *kernel*



a. Non Causal Convolutions b. Causal Convolutions c. Penambahan Padding

Gambar 2.12 Proses *causal convolutions*

Sumber: (Lässig, 2020; Wang et al., 2020)

Dapat dilihat pada gambar 2.12 a dan b, perbedaan jika menggunakan proses *causal convolutions* dan tidak. Pada gambar c dicontohkan terdapat empat data dan digunakan ukuran *kernel* tiga, jadi jumlah *padding* yang harus ditambahkan adalah ukuran *kernel* – 1 yaitu 2. Proses konvolusi pada data pertama yaitu menggunakan dua data sebelumnya yaitu *padding*, tersebut untuk mendapatkan hasil *output* dari data pertama, dan berlaku hal yang sama pada data selanjutnya sehingga data output tensor memiliki panjang yang sama dengan data *input*.

2.4.4 *Dilated Convolutions*

Proses konvolusi pada data input yang nantinya menghasilkan data *output*, hasilnya tergantung dari *kernel* yang digunakan dan data sebelumnya. Misalkan panjang data *input* adalah 5 dengan ukuran *kernel* adalah 3, maka data ke 5 akan sangat bergantung pada data ke 3 dan 4. Untuk itu untuk menghindari ketergantungan pada data model mampu menangkap keterikatan dengan data jangka panjang maka digunakanlah *konsep dilated convolutions,* yaitu menambahkan *layer* lagi di dalam jaringan dan melebarkan jangkauan dari *kernel* pada *layer* berikutnya dengan melewati beberapa data. Artinya, data *output* harus terhubung dengan data-data *input* dari masa lalu yang sangat jauh. *dilated convolution* dapat di definisikan sebagai berikut:

$$F(s) = (x \cdot_d f) = \sum_{i=0}^{i} f(i) \cdot x_{s-dxi} \quad (2.3)$$

Dimana d adalah nilai dari dilatasi, k adalah nilai dari *kernel* dan $x_{s\text{-}dxi}$ menunjukkan langkah yang diambil ke belakang.Untuk menghitung berapa banyak layer yang dibutunhkan, dan mengitung *receptive field* dapat menggunakan rumus:

$$n = \frac{(l-1)}{(k-1)} \quad (2.4)$$

$$r = 1 + n(k-1) \quad (2.5)$$

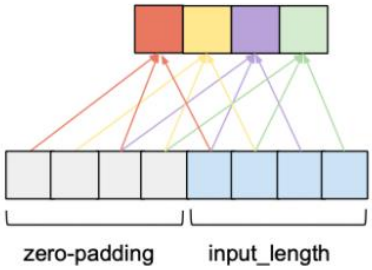
Dimana:

n = Jumlah *layer*

l = Panjang data

k = Ukuran *kernel*

r = Rentang data yang dilalui proses dilatasi



Gambar 2.13 Dilated Convolution

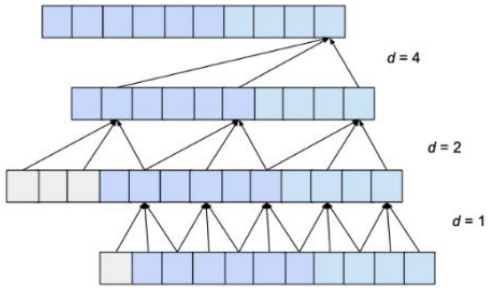
Sumber: (Lässig, 2020)

Pada gambar 2.13 dapat dilihat contoh penggunaan *dilated convolution* dengan panjang data adalah 4, ukuran *kernel* adalah 3, nilai dilatasi adalah 2, dan rentang data yang dilalui adalah 5. Jika tidak mengalami proses dilatasi, rentang data yang dilalui sama dengan ukuran *kernel* yaitu 3, sedangkan jika mengalami proses dilatasi rentang data melebar menjadi 5. Namun jika nilai dilatasi tetap, akan dibutuhkan data yang sangat panjang pada *layer* berikutnya. Untuk itu pada *layer* berikutnya, nilai dilatasi akan mengalami peningkatan secara eksponensial ditunjukkan pada gambar 2.14, *dilation base* konstan yang direpresentasikan

dengan b dan perhitungannya ditunjukkan pada rumus. Pada gambar 2.14 digunakan panjang data adalah 10, ukuran *kernel* adalah 3, *dilation base* adalah 2, dan banyaknya *layer* adalah 3. Pada penerapannya nilai dari *kernel* harus lebih besar dari *dilatation base*. Dan juga berimbas jumlah *padding* akan mengalami peningkatan untuk setiap *layer* sehingga dibutuhkan rumus baru yaitu:

$$d = b^i \tag{2.6}$$

$$p = b^i \cdot (k - 1) \tag{2.7}$$

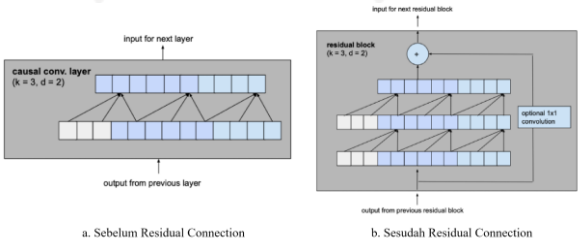


Gambar 2.14 *Dilation base*

Sumber: (Lässig, 2020)

2.4.5 *Residual Connections*

Pada arsitektur TCN sederhana, setiap *layer* akan saling terhubung dengan *layer* lainnya, sedangkan jika menggunakan *residual connection* data dapat langsung disalurkan tanpa harus melewati berbagai *layer* (Wang et al., 2020). Gambar 2.15 menunjukkan perbedaan struktur TCN sebelum ditambahkan *residual connection* dan sesudahnya.



Gambar 2.15 Struktur *residual connections*

Sumber: (Lässig, 2020)

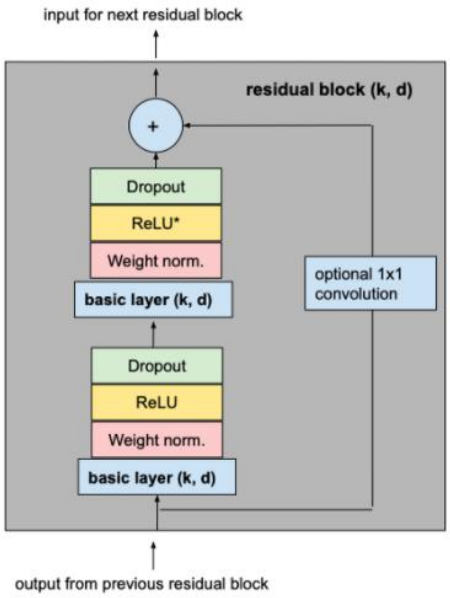
Pada gambar 2.15 digunakan ukuran *kernel* adalah 3 dan nilai dilatasi adalah 2. Dapat dilihat perubahan sesudah ditambahkan *residual connection*, hasil *output* dari kedua *layer* tersebut nantinya akan ditambahkan ke *input* dari *residual block* untuk menghasilkan *input* pada *block* selanjutnya. Karena proses konvolusi pada *layer* pertama dari *residual block* pertama dan proses konvolusi pada *layer* kedua dari *residual block* terakhir mungkin memiliki panjang data *input* yang berbeda, maka panjang dari *residual block* harus disesuaikan dengan proses konvolusi 1x1. Menambahkan *residual block* ke arsitektur TCN menjadikan lebih banyak rentang data yang dilalui dari pada model biasa. Perubahan tersebut mengharuskan dilakukan perhitungan kembali berapa *layer* yang dibutuhkan untuk dapat mencakup nilai seluruh data. Jadi rentang data dengan *dilatation base* b, *kernel size k* dengan $k \geq b$, dan jumlah *residual block* n dapat dirumuskan sebagai berikut:

$$r = 1 + \sum_{i=1}^{i} 2 \cdot (k-1) \cdot b^i = 1 + 2 \cdot (k-1) \cdot \frac{(b_n - 1)}{(b-1)} \tag{2.8}$$

$$n = \left[\log_b \left(\frac{(l-1) \cdot (b-1)}{(k-1)} + 1 \right) \right] \tag{2.9}$$

2.4.6 *Activation Function, Normalization,* dan *Regularization*

Untuk menjadikan TCN tidak sekedar model *linear regression* yang kompleks, perlu ditambahkan *activation function* di atas *convolution layer*. *Activation function* yang digunakan adalah *Rectified Linear Unit* (ReLU) yang menyebabab model tidak menjadi linier (Bai et al., 2018). Proses menormalisasi *input* pada *hidden layer* agar tidak terjadi *exploding gradien*t, ditambahkan *weight normalizatio*n pada setiap *convolution laye*r. Selanjutnya untuk mencegak terjadinya *overfitting*, digunakanlah *regularization* menggunakan *dropout function* pada akhir setiap *convolution layer*. Struktur TCN setelah ditambahkan ketiga fungsi tersebut ditunjukkan pada gambar 2.16.

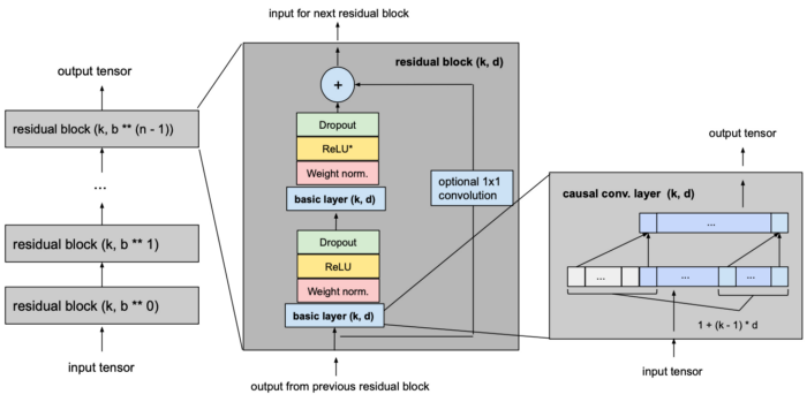


Gambar 2.16 Penambahan fungsi *activation, normalization*, dan *regularization*

Sumber: (Lässig, 2020)

2.4.7 Model Akhir

Pada gambar 2.17 menunjukkan model akhir dari algoritma *Temporal Convolutional Network* (TCN). Dengan l merupakan panjang data, k adalah ukuran *kernel*, b merupakan *dilatation base* dengan k ≥ b, jumlah minimal *residual block* adalah n.



Gambar 2.17 Model akhir TCN

Sumber: (Lässig, 2020)

2.5 Matrik Evaluasi

Untuk mengetahui tingkat keakuratan hasil prediksi yang didapatkan oleh model machine learning dapat menggunakan matrik evaluasi. Perlu diketahui bahwa pada pemodelan machine learning sendiri dapat dibagi menjadi dua tipe yaitu pemodelan klasifikasi dan regresi. Klasifikasi sendiri merupakan pemodelaan yang digunakan untuk mengelompokkan data-data ke dalam sejumlah kelas ataupun kategori tertentu, sedangkan regresi merupakan pemodelan machine learning dengan melakukan proses identifikasi pada data untuk menemukan suatu relasi (Suyanto, 2018). Proses evaluasi pada pemodelan klasifikasi umumnya dapat menggunakan matrik evaluasi dengan confusion matrix yaitu menghitung tingkat akurasi dari model, sedangkan untuk pemodelan regresi dapat menggunakan matrik evaluasi Root Mean Squared Error (RMSE) yaitu dengan menghitung selisih nilai prediksi dengan nilai aktualnya. Semakin kecil nilai RMSE yang dihasilkan dari suatu algoritma, amka semaik kecil nilai errornya atau selisih nilai prediksi dan aktualnya sangat kecil.

2.5.1 Mean Absolute Error (MAE)

Mean Absolute Error (MAE) merupakan matrik evaluasi yang menghitung nilai absolut dari rata-rata selisih nilai prediksi dan nilai aktual. Rumus dari MAE sendiri didefinisikan sebagai berikut:

$$\sum \frac{|Y' - Y|}{n} \tag{2.10}$$

Dimana:

Y’ = Nilai prediksi

Y = Nilai aktual

N = Jumlah data

2.5.2 Mean Square Error (MSE)

Mean Squared Error (MSE) merupakan matrik evaluasi yang mirip dengan MAE, namun menggunakan proses kuadrat untuk menghitung nilai errornya. Rumus dari MSE sendiri didefinisikan sebagai berikut:

$$\sum \frac{(Y' - Y)^2}{n} \tag{2.11}$$

2.5.3 Root Mean Squared Error (MAE)

Root Mean Squared Error (RMSE) merupakan matrik evaluasi yang menambahkan proses perhitungan akar pada matrik MSE. Rumus dari RMSE sendiri dapat didefinisikan sebagai berikut:

$$\sqrt{\sum \frac{(Y' - Y)^2}{n}} \tag{2.12}$$

2.6 Penelitian Terkait

Terdapat 7 jurnal penelitian yang digunakan sebagai referensi dalam proses perencanaan pada penelitian ini. Penelitian-penelitian terkait disajikan dalam bentuk tabel agar memudahkan dalam melihat hasil dan solusi yang ditawarkan pada prediksi tren pergerakan harga saham.

Tabel 2.1 Penelitian Terkait

No	Jurnal	Algoritma	Dataset	Hasil
1.	(Zayini Anwar & Habibi, n.d.)	*Long Short Term Memory* (LSTM)	Saham Apple (AAPL)	Menggunakan algoritma LSTM untuk melakukan prediksi. Hasil prediksi yang didapatkan cukup baik, dengan nilai RMSE 0.2286. Sehingga peneliti menyimpulkan bahwa algoritma LSTM cukup efektif untuk prediksi harga saham.
2.	(Mehtab et al., n.d.)	*Long Short Term Memory* (LSTM) *Multivariat* dan *Grid Search*	Indeks Saham NIFTY 50	Penelitian ini menggunakan 4 model LSTM yang berbeda, (i) LSTM *univariate input data of 1 wee*k (ii) LSTM *univariate input data of 2 week* (iii) *Encoder-Decoder LSTM univariate input data of 1 week* (iv) *Encoder-Decoder LSTM multivariate input data of 2 week*. Prediksi harga dilakukan untuk harga satu mingguan. Didapatkan hasil bahwa model LSTM cukup efektif untuk prediksi harga saham, namun dimensi *multivariate* memberikan hasil yang kurang memuaskan dan komputasi yang lama.

3.	(Kim & Kim, 2019)	Kombinasi *Long Short Term Memory* (LSTM) dengan *Convolutional Neural Network* (CNN)	Indeks Saham S&P 500	Mengkombinasikan algoritma LSTM dengan CNN, kemudian membandingkan hasil dengan dengan algoritma LSTM dan CNN normal. Mengenkstak fitur serta gambar grafik pada data saham. Hasil yang didapatkan model kombinasi dapat mengunguli performa model normal. Selain itu, didapatkan informasi bahwa *candlestick chart* merupakan *chart* yang paling efektif untuk prediksi harga saham, dan dapat meminimalisir kesalahan prediksi yang terjadi. Dengan nilai RMSE 0.098.
4.	(Bai et al., 2018)	*Temporal Convolutional Network* (TCN)	-	Memperkenalkan kosen dari algoritma *Temporal Convolutional Network* (TCN).
5.	(Liu et al., 2019)	*Temporal Convolutional Network* (TCN)	Makey-Glass data, OM2.5 data	Memperkenalkan algoritma TCN dengan modifikasi *Multi Channel* (M-GTCN). Algoritma tersebut kemudian dibandingkan dengan LSTM, GRU, dan TCN. Disimpulkan bahwa memang TCN memiliki performa yang baik daripada LSTM dan GRU pada kasus *single factor*. Namun pada kasus *multi factor* hasil dari TCN sangat buruk. M-GTCN dapat mengungguli algoritma lainnya, namun arsitekturnya lebih kompleks.
6.	(Wan et al., 2019)	*Temporal Convolutional Network* (TCN) dengan kasus *multivariate*	Beijing PM2.5	Menggunakan algoritma TCN untuk mengolah kasus dengan tipe data *multivariate* (M-TCN). Hasil yang didapatkan, algoritma M-TCN memiliki kemampuan generalisasi yang rendah, hasil prediksi sangat bervariasi pada *dataset* yang berbeda.
7.	(Wang et al., 2020)	Kombinasi *Temporal Convolutional Network* (TCN) dan *Representative Learning*	Indeks saham S&P 500	Mengkombinasikan algoritma TCN dengan r*epresentative learning* yang disebut Stock2Vec. Hal tersebut dilakukan untuk mengetahui hubungan di antara saham satu dengan yang lain. Hasil yang didapatkan cukup baik dengan nilai RMSE 2.22.

Dari penelitian-penelitian terkait yang telah disebutkan, memprediksi tren pergerakan harga saham banyak dikembangkan menggunakan algoritma *Long Short Term Memory* (LSTM). Namun, pada algoritma LSTM sendiri atapun algoritma perkembangan dari LSTM masih banyak terjadi masalah pada *vanising gradient*. Untuk itu pada penelitian yang dilakukan

oleh (Bai et al., 2018) mulai memperkenalkan algoritma baru yang dengan menggunakan konsep konvolusi pada pengolahan data *time series* yaitu *Temporal Convolutional Network* (TCN). TCN sendiri mempunyai beberapa kelebihan dibandingakn LSTM, yaitu menggunakan proses *backpropagation* untuk meminimalisir terjadinya *vanisihing gradient* dan mampu memproses data secara pararel sehingga data yang panjang dapat dieksekusi dengan cepat. Namun pemrosesan data menggunakan TCN memerlukan penyimpanan yang lebih besar saat melakukan proses evaluasi. Algoritma TCN Mampu memberikan hasil yang baik dan mampu mengungguli algoritma yang umum dipakai, ditunjukkan pada penelitian (Bai et al., 2018) yaitu LSTM, RNN, dan *Gated Recurrent Unit* (GRU).

Untuk itu pada penelitian ini dipilih menggunakan algoritma TCN karena berdasarkan penelitian terkini yang menunjukkan bahwa TCN memberikan hasil yang lebih baik walaupun menggunakan pendekatan yang berbeda. Hasil yang didapatkan TCN sangat dipengaruhi oleh panjang data yang diolah, sedangkan LSTM dan GRU dengan ukuran panjang data yang sama hasilnya berubah-ubah. Keakuratan LSTM dapat turun hingga 20% untuk data di bawah 50, sedangkan GRU turun hingga 20% untuk data di bawah 200. Dengan hasil tersebut dapat menunjukkan bahwa TCN mampu mengolah data yang panjang dan mempertahankan riwayat data yang lebih jauh (Bai et al., 2018). Algoritma TCN juga dapat dilakukan modifikasi untuk berbagai kebutuhan sesuai masalah yang dihadapi, seperti mengkombinasikan dengan basis pengetahuan untuk mengetahui korelasi dari antar data seperti pada penelitian (Wang et al., 2020). Penelitan ini akan mencoba membandingkan performa dari algoritma TCN dengan algoritma LSTM sebagai algoritma yang cukup banyak digunakan untuk memprediksi harga saham, algoritma Linear Regression dan Decision Tree Regression yang jarang digunakan untuk memprediksi harga saham. Serta pemodelan algoritma TCN menggunakan library yang cukup baru dikembangkan yaitu Darts.

BAB III
METODOLOGI PENELITIAN

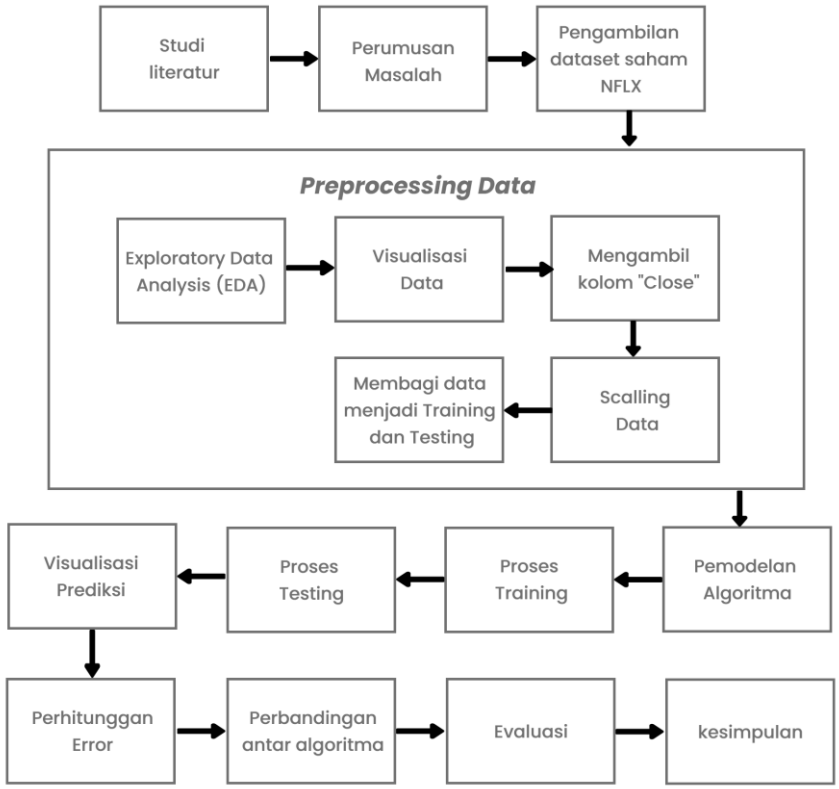
3.1 Pengumpulan Data

Data saham NFLX dipilih karena merupakan salah satu saham yang termasuk ke dalam indeks saham NASDAQ-100, yang mana mempunyai kapitalisasi pasar sangat besar dan likuiditas yang tinggi sehingga harga sahamnya tidak mudah untuk dimanipulasi dan terjadi lonjakan harga yang ekstrim secara tiba-tiba. Pergerakan harga saham NFLX sudah mewakili berbagai tren yang terjadi, seperti tren kenaikan, tren penurunan, dan tren konsolidasi. Tidak menutup kemungkinan untuk menggunakan data saham yang lain, setiap harga saham pasti terbentuk dari kombinasi ketiga tren tersebut, namun juga terdapat data harga saham yang harganya dalam jangka waktu yang cukup lama tidak mengalami pergerakan yang signifikan dan bahkan harganya terkapar. Untuk itu dipilih saham yang memiliki fluktuasi harga yang mewakili pergerakan tren sehingga model yang dibangun dapat mengetahui pola-pola pergerakan tren dari saham yang dipilih.

Pada penelitian ini data saham yang akan digunakan dapat dikelompokkan sebagai data sekunder, yaitu data yang tidak didapatkan langsung oleh peneliti namun data yang sudah disediakan oleh lembaga ataupun sesorang yang dijadikan subjek penelitian (Jati Lantang M, 2020). Data saham Netflix, Inc (NFLX) disediakan oleh *website* penyedia informasi tentang dunia keuangan yaitu *Yahoo Finance*. Data tersusun dalam *timeframe daily*, yaitu data dicatatkan setiap satu hari pada hari bursa (Senin-Jumat). Data saham dalam satu tahun berjumlah rata-rata 252 hingga 253, tidak utuh 365 hari karena pada bursa terdapat hari libur. Rentang data saham yang digunakan yaitu sejak 1 Januari 2015 hingga 31 Mei 2021, dengan jumlah data 1613 dan terdapat 6 kolom yaitu harga pembukaan (*open*), harga penutupan (*close*), harga terendah (*low*), harga tertinggi (*high*), volume transaksi, dan *adj close*. Pada penelitian ini hanya akan digunakan harga penutupan (*close*), karena pada hari bursa harga akan mengalami fluktuasi maka digunakan harga penutupan yang merupakan harga terakhir pada hari bursa tersebut. Kemudian data dengan *timeframe daily* digunakan karena prediksi tren yang akan dilakukan untuk mengetahui gambaran besar dari tren dan prediksi akan digunakan untuk investasi jangka menengah. Data diambil memanfaatkan *library pandas data reader*, hanya dengan memasukkan informasi mengenai *ticker* saham, sumber data, dan rentang waktu yang diinginkan.

3.2 Perancangan Penelitian

Perancangan penelitian merupakan gambaran secara terstruktur tentang penelitian yang akan dilakukan untuk memudahkan dalam prosesnya. Perancangan penelitian disajikan dalam bentuk *flowchart* dari awal hingga akhir. Gambar 3.1 menunjukkan *flowchart* penelitian ini.



Gambar 3.1 *Flowchart* penelitian

Penjelasan *flowchart* penelitian:

a. Studi literatur dilakukan untuk mengetahui penelitian terkini dan *state of the art* dari permasalahan yang ingin diangkat.
b. Perumusan masalah dilakukan untuk menentukan masalah yang dihadapi dalam penelitian, dan selanjutnya dilakukan perancangan penelitian.
c. Langkah pertama yang dilakukan adalah pengambilan data saham Netflix (NFLX) yang telah disediakan oleh *website Yahoo Finance*. Data dapat diambil dengan memanfaatkan *library dari pandas yaitu pandas data reader*. Dengan memanfaatkan *library* tersebut, proses pengambilan data menjadi lebih mudah, hanya dengan

memasukkan ticker dari saham yang diinginkan kemudian sumber data tersebut dan rentang data yang diinginkan dengan menyetel tanggal awal dan akhir.

d. Langkah selanjutnya adalah *preprocessing* data saham yang telah didapatkan. Proses pertama yang dilakukan dalam *preprocessing* adalah *Exploratory Data Analysis* (EDA). Yaitu melakukan proses investigasi untuk memahami kondisi data yang digunakan dan menyiapkan data agar mampu diolah oleh model algoritma. Beberapa proses yang dilakukan adalah: mengecek dimensi dan jumlah data, mengecek tipe data, dan mengecek apakah terdapat data yang kosong.
e. Setelah dilakukan proses EDA, selanjutnya data akan divisualisasikan menggunakan line chart untuk melihat bagaimana pergerakan harga yang terjadi.
f. Mengambil kolom harga penutupan (*close*) saja untuk digunakan pada penelitian.
g. *Scaling* data dilakukan pada data dengan rentang nilai 0 hingga 1, agar saat melakukan prediksi model algoritma tidak mengalami bias karena rentang harga aktual bisa sangat jauh.
h. Proses terakhir pada tahap *preprocessing* adalah melakukan peemisahan data menjadi data *traning* dan data *testing*. Data training digunakan sebagai data untuk melakukan pembelajaran pada model algoritma. Dan data testing digunakan untuk menguji model yang telah dilatih.
i. Langkah selanjutnya adalah melakukan pemodelan pada algoritma yang digunakan.
j. Proses *traning* dilakukan untuk melakukan pembelajaran pada pada model algoritma.
k. Selanjutnya model akan dilakukan proses *testing* untuk mengetahui hasil prediksi yang dilakukan.
l. Hasil prediksi akan divisualisasikan untuk mengetahui bagaimana perbandingannya dengan harga aktual.
m. Hasil prediksi dengan harga aktual akan dihitung nilai errornya untuk mendapatkan kesimpulan apakah model menghasilkan prediksi yang baik atau buruk.
n. Hasil perhitungan nilai *error* yang didapatkan kemudian akan dikomparasikan dengan dengan nilai *error* dari algoritma lain. Algoritma manakah yang mendapatkan hasil yang paling baik dan paling buruk. Dan akan dievaluasi hal apakah yang menyebabkan model algoritma mendapatkan hasil prediksi seperti itu.
o. Setelah didapatkan hasil prediksi dan perhitungan *error* antar algoritma, maka akan menghasilkan kesimpulan mengenai penelitian yang dihadapi.

3.2.1 *Temporal Convolutional Network* (TCN)

Gambar 3.2 Flowchart model *Temporal Convolutional Network* (TCN)

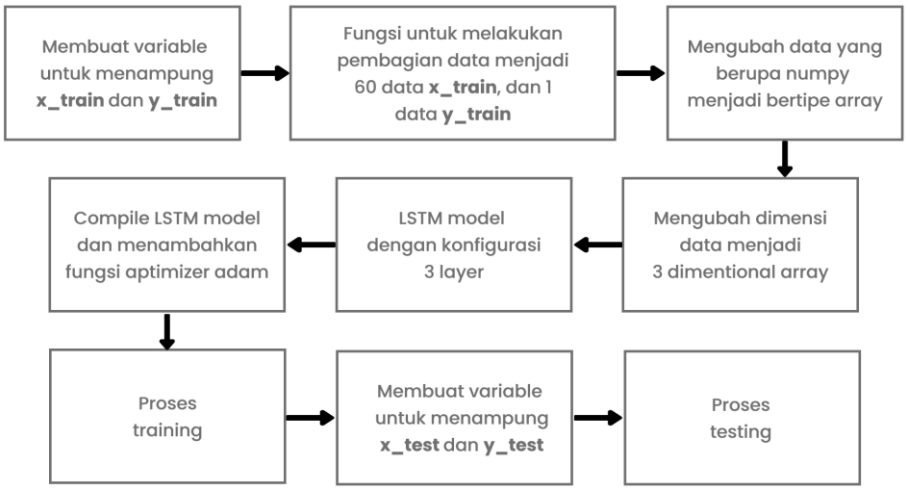
Penjelasan *flowchart* model *Temporal Convolutional Network* (LSTM):

a. Membangun model algoritma TCN memanfaatkan library darts, data yang sudah disiapkan sebelumnya tinggal dimasukkan pada fungsi darts untuk mengolahnya pada algoritma TCN kemudian melakukan *hyperparameter* tuning disesuaikan dengan data.
b. Proses *training* pada model TCN
c. Proses *backtesting* dilakukan untuk mengevaluasi performa model menggunakan fungsi yang disediakan oleh darts dan melakukan prediksi pada data.

3.2.2 Algoritma Pembanding

Untuk membandingakan performa dari model *Temporal Convolutional Network* (TCN), digunakan algoritma yang paling populer dan umum digunakan pada penyelesaian masalah prediksi harga saham, yaitu algoritma *Long Short Term Memory (LSTM)*. Dan dua lagoritma paling umum dan sederhana digunakan pada model *machine learning*, yaitu adalah algoritma *Linear Regression* dan *Decision Tree Regression*. Berikut adalah *flowchart* penelitian pada ketiga algoritma tersebut.

***Long Short Term Memory* (LSTM)**

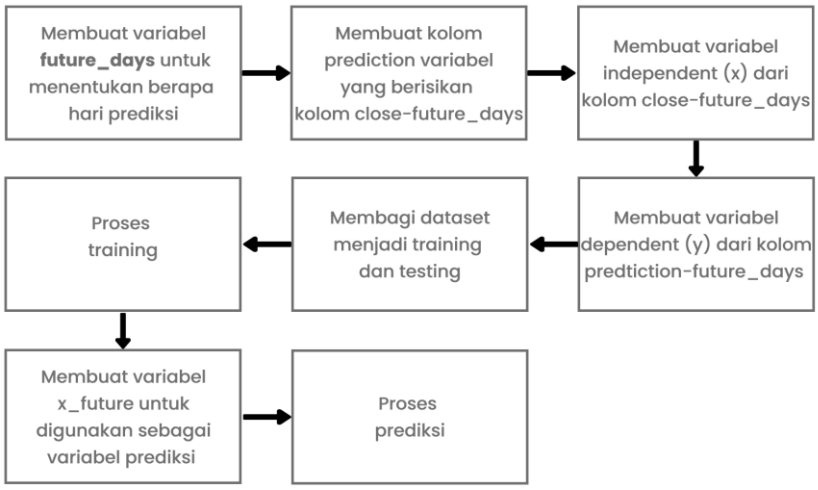


Gambar 3.3 *Flowchart* model *Long Short Term Memory* (LSTM)

Penjelasan flowchart model *Long Short Term Memory* (LSTM):

a. Pertama-tama membuat variabel untuk menampung data yang akan digunakan pada prediksi model LSTM.
b. Membuat fungsi untuk melakukan pembagian pada data menjadi 60 data yang akan digunakan sebagai data prediksi dan 1 data menajdi data hasil prediksi. 60 data dimulai dari data indeks ke 0 hingga 59, dan 1 data dimulai dari indeks ke 60. Jadi prediksi akan dilakukan setiap 60 data sekali dengan hasil prediksi data ke 61, kemudian prediksi akan berlanjut lagi dari data indeks ke 1 hingga 60 dan hasil prediksi dengan indeks ke 61 dan akan berlangsung begitu seterusnya hingga akhir dari data.
c. Mengubah data yang masih berupa *numpy* menjadi data *array*.
d. Kemudian membangun model LSTM dengan konfigurasi tiga *layer*, dengan *layer* pertama sebanyak 50 *neuron*, layer kedua dengan 50 *neuron*, dan layer terakhir dengan 25 *neuron*.
e. Melakukan compile pada model yang telah dibangun, dan ditambahkan fungsi *optimezer adam*.
f. Proses *training* pada model LSTM dan menyetel *hyperparameter* yang tersedia.
g. Proses *testing* pada model LSTM dan menyetel *hyperparameter* yang tersedia.

Linear Regression* dan *Decission Tree Regression



Gambar 3.4 *Flowchart* model *Linear Regression* dan Decission *Tree Regression*

Penjelasan *flowchart* model *Linear Regression* dan *Decission Tree Regression*:

a. Pertama-tama harus ditentukan berapa hari prediksi yang diinginkan, lama hari yang akan diprediksi akan disimpan dalam variabel *future_days*.
b. Algoritma *Linear Regression* dan *Decission Tree Regression* membutuhkan variabel *independent* yang merupakan variabel bebas yang akan dihitung untuk mendapatkan hasil prediksi, dan variabel *dependent* atau variabel terikat yang nilainya dipengaruhi variabel *independent*. Kolom harga *close* merupakan variabel *independent*, untuk variabel *dependent* akan dibuat dari kolom harga *close* dikurangi berapa lama hari yang akan diprediksi (*future_days*) yang dinamai sebagai kolom *prediction*. Kemudian pada kolom prediction akan dikenakan fungsi *shift* untuk menaikkan data sehingga terdapat data yang kosong pada bagian akhir data yang akan diisi oleh harga prediksi.
c. Variabel *independent* akan dipisahkan dari variabel *dependent* dan dikurangi berapa lama hari yang akan diprediksi (*future_days*), kemudian ditampung pada variabel baru yaitu x.
d. Variabel *dependent* akan dipisahkan dari variabel *independent* dan dikurangi berapa lama hari yang akan diprediksi (*future_days*), kemudian ditampung pada variabel baru yaitu y.
e. Kemudian dari variabel x dan y akan dibagi menjadi data *training* dan data *testing*.

f. Proses *training* pada model *Linear Regression* dan *Decission Tree Regression*.
g. Membuat variabel baru untuk digunakan prediksi dengan nama *x_future*, dengan mengambil data sepanjang *future_days* pada akhir data *close*.
h. Prose prediksi pada model *Linear Regression* dan *Decission Tree Regression*.

3.3 Evaluasi

Proses evaluasi akan dilakukan setelah proses prediksi menggunakan model algoritma sudah didapatkan hasilnya, serta sudah diketahui bagaimana visualisasi dari hasil prediksi tersebut. Evaluasi yang dilakukan yaitu mengamati bagaimana hasil prediksi yang dihasilkan oleh model algoritma menggunakan matrik perhitungan *error* dengan menggunakan matrik *Mean Absolute Error* (MAE), *Mean Squared Error* (MAE), *Root Mean Squared Error* (RMSE), apakah prediksi harga dan tren yang terbentuk dapat mengikuti harga aktualnya atau tidak. Akan dianalisa kenapa hal tersebut dapat terjadi.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Implementasi

Semua tahapan perancangan penelitian yang telah dijabarkan dalam bentuk *flowchart* pada gambar 3.1, diimplementasikan pada kode program menggunakan bahasa pemrograman Python 3.7. Proses pada kode program akan dijabarkan pada bab ini.

4.1.1 *Temporal Convolutional Network* (TCN)

Berikut adalah implementasi pada kode program menggunakan bahasa pemrograman Python dari *flowchart* gambar 3.2 algoritma *Temporal Convolutional Network* (TCN).

***Import Library* yang Dibutuhkan**

Gambar 4.1 mendefinisikan *library* yang digunakan. Terlebih dahulu melakukan *library* dan *package* yang dibutuhkan yang nantinya akan digunakan pada pemrosesan algoritma TCN yang telah disediakan oleh *library darts*.

```
1.  # Import libraries
2.  import math
3.  import numpy as np
4.  import pandas as pd
5.  from darts import TimeSeries
6.  from darts.models import TCNModel, RNNModel
7.  from darts.dataprocessin.transormers import Scaler
8.  from farts.utils.timeseries_generation
9.      import datetime_attribute_
10. from darts.metrics import mape, r2_score
11. from darts.utils.missing_values import fill_missing_values
```

Gambar 4.1 *Import library*

Mendapatkan Data Saham

Gambar 4.2 mendefinisikan fungsi untuk mendapatkan data saham Netflix (NFLX) memanfaatkan *library pandas data reader* pada baris kode ke 2, memanggil fungsi *DataReader* dengan memasukkan parameter kode *ticker* dari saham, sumber data, rentang data dari awal hingga akhir. Kemudian ditampilkan data saham yang sudah didapatkan, data memiliki 1613 baris dan 6 kolom bertipe *pandas data frame* seperti ditunjukkan pada gambar 4.3.

```
# Get the stock data
nflx_tcn = web.DataReader('NFLX', data_source = 'yahoo', start =
'2015-01-01', end = '2021-05-31'

# Show the stock data
print(nflx_tcn.head())
print(nflx_tcn.tail())
```

Gambar 4.2 Mendapatkan data saham

Date	High	Low	Open	Close	Volume	Adj Close
2015-01-02	50.331429	48.731430	49.151428	49.848572	13475000.0	49.848572
2015-01-05	49.258572	47.147144	49.258572	47.311428	18165000.0	47.311428
2015-01-06	47.639999	45.661430	47.347141	46.501431	16037700.0	46.501431
2015-01-07	47.421429	46.271427	47.347141	46.742859	9849700.0	46.742859
2015-01-08	47.835712	46.478573	47.119999	47.779999	9601900.0	47.779999
...	...	...	...	...	...	...
2021-05-24	504.250000	499.510010	501.049988	502.899994	2412600.0	502.899994
2021-05-25	506.369995	499.220001	506.000000	501.339996	2699500.0	501.339996
2021-05-26	504.140015	500.500000	502.339996	502.359985	2465300.0	502.359985
2021-05-27	505.100006	498.540009	501.799988	503.859985	3253800.0	503.859985
2021-05-28	511.760010	502.529999	504.399994	502.809998	2910300.0	502.809998

1613 rows × 6 columns

Gambar 4.3 Data saham NFLX

***Exploratory Data Analysis* (EDA) dan Visualisasi Data**

Gambar 4.4 mendefinisikan fungsi untuk melakukan proses *Exploratory Data Analysis* (EDA). Karena data yang digunakan sudah tersusun dengan rapi dan baik maka proses EDA tidak terlalu banyak dilakukan. Dengan proses EDA didapatkan beberapa *insight* bahwa jumlah baris data 1613 dengan 6 kolom dan *date* menjadi indeks data tersebut. Kemudian tipe data adalah *float64* dan tidak terdapat data yang kosong ditunjukkan pada gambar 4.5. Kemudian data divisualisasikan dilakukan untuk mengetahui bagaimana data ditampilkan pada *line chart* dengan menggunakan kolom harga *close* seperti pada gambar 4.6.

```
1.  # Exploratory data analysis
2.  print(nflx_tcn.shape)
3.  print(nflx_tcn.info())
4.  print(nflx_tcn.isnull())
5.
6.  # Visualize the close price
7.  plt.figure(figsize = (20, 10))
8.  plt.plot(nflx_tcn['Close'])
9.  plt.title('NFLX Stock Closing Price', fontsize - 24)
10. plt.xlabel('Days')
11. plt.ylabel('Close Price $USD')
12. plt.show()
```

Gambar 4.4 *Exploratory Data Analysis* (EDA) pada data saham

```
Jumlah kolom dan baris :  (1613, 6)
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1613 entries, 2015-01-02 to 2021-05-28
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   High       1613 non-null   float64
 1   Low        1613 non-null   float64
 2   Open       1613 non-null   float64
 3   Close      1613 non-null   float64
 4   Volume     1613 non-null   float64
 5   Adj Close  1613 non-null   float64
dtypes: float64(6)
memory usage: 88.2 KB
None
Cek missing value : High         0
Low          0
Open         0
Close        0
Volume       0
Adj Close    0
dtype: int64
```

Gambar 4.5 Hasil *Exploratory Data Analysis*

Gambar 4.6 Visualisasi harga *close*

Mengambil Kolom *Close* dan *Scaling* Data

Gambar 4.7 mendefinisikan fungsi untuk mengambil kolom *close* pada *dataframe nflx_tcn*, hanya kolom *close* saja yang akan digunakan pada penelitian ini. Kemudian didefinisikan fungsi *scaling* yang disimpan di variabel *scaler*, yang memanfaatkan *library darts.dataprocessing*. Data kemudian akan di-*scaling* menjadi dalam rentang antara 0 hingga 1. Pada baris kode ke 7 fungsi *scaler* dikenakan pada data nflx_tcn menggunakan fungsi *fit_transform*.

```
1. # Get only the close price
2. nflx_tcn = nflx_daily[['Close']]
3. nflx_tcn.head()
4.
5. # Scaling Data
6. scaler = Scaler()
7. nflx_tcn = scaler.fit_transform(nflx_tcn)
```

Gambar 4.7 Mengambil kolom *close* dan *scaling* data

Fungsi *Split Dataset*

Gambar 4.8 mendefinisikan fungsi untuk melakukan *split* pada *dataset*. Data dibagi menjadi data *training* dan *testing* dan di-*split* setelah tanggal '2021-03-05' karena akan dilakukan prediksi selama 60 hari, jadi dari data terakhir dikurangi 60 hari.

```
1. # Creatte training and validation dataset
2. train, val = nflx_tcn(pd.Timestamp('20210503'))
```

Gambar 4.8 Split dataset

Membangun Model

Gambar 4.9 mendefinisikan fungsi untuk membangun model TCN, beberapa *hyperparameter* yang tersedia dapat diubah-ubah sesuai dengan kebutuhan. Model dari algoritma TCN didefinisikan pada baris kode ke 2 dengan nama variabel *tcn_model* dengan memanfaatkan fungsi TCNModel. Panjang data yang akan dilakukan *training* adalah 1553 dan dengan panjang *output* = 60. Pada penelitian ini *hyperparameter* disetel menjadi: *n_epoch* = 20, *droupout* = 0.1, *dilatation_base* = 2, *weight_norm* = true, *kernel_size* = 5, *num_filter* = 3. Jumlah epoch yang digunakan sebanyak 20 mengacu pada penelitian , kemudian nilai *dropout* digunakan 0.1 karena merupakan nilai paling umum digunakan, dilatation_base menggunakan nilai standar yaitu 2 agar pola yang diketahui tidak terlalu lebar. Dan kemudian dengan besar layer adalah 5 karena hari bursa buka selama 5 hari dalam seminggu sehingga ingin dilakukan proses konvolusi pada setiap minggunya, dan digunakan konfigurasi 3 layer.

```
1.  # Build TCN model
2.  tcn_model = TCNModel(
3.   input_chunk_length = 1553,
4.   output chunk_length = 60,
5.   n_epoch = 20,
6.   dropout = 0.1,
7.   dilatation_base = 2,
8.   weight_norm = True,
9.   kernel_size = 5,
10.  num_filters = 3
11.  random_state = 0
12. )
```

Gambar 4.9 Membangun model *Temporal Convolutional Network* (TCN)

Proses *Training*

Gambar 4.10 mendefinisikan fungsi untuk melakukan proses *training* pada data.

```
1.  # Training the model
2.  tcn_model.fit(series = train, val_series = val, verbose = True)
```

Gambar 4.10 Proses *training*

Proses *Backtesting*

Gambar 4.11 mendefinisikan fungsi untuk mengevaluasi kinerja model TCN yang telah dibentuk sebelumnya pada baris kode ke 2 dengan nama variabel *backtest* dan dengan memanfaatkan fungsi *historical_forecast*. Beberapa *hyperparameter* digunakan pada fungsi ini, *forecast_horizon* adalah berapa lama hari yang akan diprediksi. Dengan rentang 60 hari dan tidak dilakukan training ulang.

```
1.  # Backtesting the TCN model
2.  backtest = tcn_model.historical_forecast(
3.          Series_transformed,
4.          Start = 0.7,
5.          Forecast_horizon = 60,
6.          Retrain = False,
7.          Verbose = True
8.      )
```

Gambar 4.11 Proses *backtesting*

4.1.2 *Long Short Term Memory* (LSTM)

Berikut adalah implementasi pada kode program menggunakan bahasa pemrograman Python dari flowchart gambar 3.3 algoritma *Long Short Term Memory* (LSTM).

***Import Library* yang Dibutuhkan**

Gambar 4.12 mendefinisikan *library* yang digunakan. Terlebih dahulu melakukan *import library* dan *package* yang dibutuhkan yang nantinya akan digunakan pada pemrosesan algoritma LSTM yang telah disediakan oleh *library keras*.

```
1.  # Import libraries
2.  import math
3.  import numpy as np
4.  import pandas as pd
5.  import pandas_datareader as web
6.  from sklearn import mertics
7.  from sklearn.preprocessing import MinMaxScaler
8.  from keras.models import Sequential
9.  from keraas.layers import Dense, LSTM
10. import matplotlib.pyplot as plt
11. plt.style.use('fivethirtyeight')
```

Gambar 4.12 *Import library*

Mendapatkan Data Saham

Gambar 4.13 mendefinisikan fungsi untuk mendapatkan data saham Netflix (NFLX) memanfaatkan *library pandas data reader* pada baris kode ke 2 dengan memasukkan kode *ticker* dari saham, sumber data, rentang data dari awal hingga akhir. Kemudian ditampilkan data saham yang sudah didapatkan, data memiliki 1613 baris dan 6 kolom bertipe *pandas data frame* seperti ditunjukkan pada gambar 4.14.

```
1.  # Get the stock data
2.  nflx_df = web.DataReader('NFLX', data_source = 'yahoo', start =
3.  '2015-01-01', end = '2021-05-31'
4.
5.  # Show the stock data
6.  print(nflx_df.head())
7.  print(nflx_df.tail())
```

Gambar 4.13 Mendapatkan data saham

Date	High	Low	Open	Close	Volume	Adj Close
2015-01-02	50.331429	48.731430	49.151428	49.848572	13475000.0	49.848572
2015-01-05	49.258572	47.147144	49.258572	47.311428	18165000.0	47.311428
2015-01-06	47.639999	45.661430	47.347141	46.501431	16037700.0	46.501431
2015-01-07	47.421429	46.271427	47.347141	46.742859	9849700.0	46.742859
2015-01-08	47.835712	46.478573	47.119999	47.779999	9601900.0	47.779999
...	...	...	...	...	...	...
2021-05-24	504.250000	499.510010	501.049988	502.899994	2412600.0	502.899994
2021-05-25	506.369995	499.220001	506.000000	501.339996	2699500.0	501.339996
2021-05-26	504.140015	500.500000	502.339996	502.359985	2465300.0	502.359985
2021-05-27	505.100006	498.540009	501.799988	503.859985	3253800.0	503.859985
2021-05-28	511.760010	502.529999	504.399994	502.809998	2910300.0	502.809998

1613 rows × 6 columns

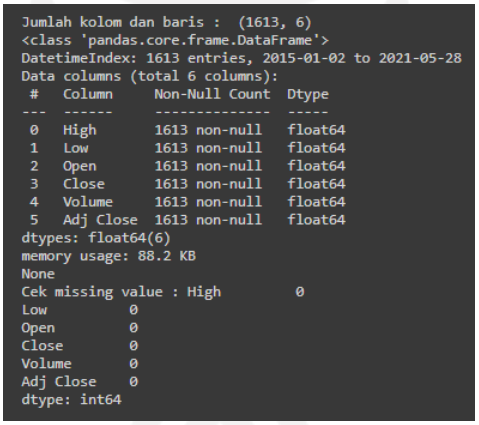
Gambar 4.14 Data saham NFLX

***Exploratory Data Analysis* (EDA) dan Visualisasi Data**

Gambar 4.15 mendefinisikan fungsi untuk melakukan proses *Exploratory Data Analysis* (EDA). Karena data yang digunakan sudah tersusun dengan rapi dan baik maka proses EDA tidak terlalu banyak dilakukan. Dengan proses EDA didapatkan beberapa *insight* bahwa jumlah baris data 1613 dengan 6 kolom dan *date* menjadi indeks data tersebut. Kemudian tipe data adalah *float64* dan tidak terdapat data yang kosong ditunjukkan pada gambar 4.16. Kemudian data divisualisasikan dilakukan untuk mengetahui bagaimana data ditampilkan pada *line chart* dengan menggunakan kolom harga *close* seperti pada gambar 4.17.

```
1.  # Exploratory data analysis
2.  Print(nflx_df.shape)
3.  Print(nflx_df.info())
4.  Print(nflx_df.isnull())
5.
6.  # Visualize the close price
7.  Plt.figure(figsize = (20, 10))
8.  Plt.plot(nflx_df['Close'])
9.  Plt.title('NFLX Stock Closing Price', fontsize - 24)
10. Plt.xlabel('Days')
11. Plt.ylabel('Close Price $USD')
12. Plt.show()
```

Gambar 4.15 *Exploratory Data Analysis* (EDA) pada data saham



Gambar 4.16 Hasil *Exploratory Data Analysis*

Gambar 4.17 Visualisasi harga *close*

Mengambil Kolom *Close* dan Fungsi *Split Dataset*

Gambar 4.18 mendefinisikan fungsi untuk mengambil kolom *close* pada *dataframe nflx_df,* hanya kolom *close* saja yang akan digunakan pada penelitian ini. Kemudian data diubah menjadi bertipe *array* dan dibagi menjadi data *training* dan *testing* dengan perbandingan 80% dan 20%, pada baris kode ke 8 dengan menyetel parameter data training menjadi 0.80.

```
1.  # Create a new dataframe with only the "Close" column
2.  df = nflx_df.filter(['Close'])
3.
4.  # Convert the dataframe to a numpy array
5.  dataset = df.values
6.
7.  # Get the number of rows to train the model, 80% of the dataset
8.  training_data_len = math.ceil(len(dataset) * .80)
9.  training_data_len
```

Gambar 4.18 Mengambil kolom *close*

Fungsi *Scaling*

Gambar 4.19 mendefinisikan fungsi scaling yang disimpan di variabel *scaler*, yang memanfaatkan *library sklearn.preprocessing*. Data kemudian akan di-*scaling* menjadi dalam rentang antara 0 hingga 1 pada baris kode ke 3 dan dikenakakan fungsi *fit_transform* pada

dataset yang akan di-*scaling* pada baris kode ke 4. gambar 4.20 menunjukkan data setelah dikenakan fungsi *scaling*.

```
1.  # Scaling the dataset
2.  # Scale will be in range between 0 and 1
3.  scaler = MinMaxScaler(feature_range = (0, 1))
4.  scaled_data = scaler.fit_transform(dataset)
5.  scaled_data
```

Gambar 4.19 Fungsi *scaling*

```
array([[0.00795393],
       [0.00326241],
       [0.00176461],
       ...,
       [0.84470942],
       [0.84748312],
       [0.84554155]])
```

Gambar 4.20 Hasil *scaling*

Membuat *Variabel X Train, Y Train*, dan Fungsi Pembagi Data

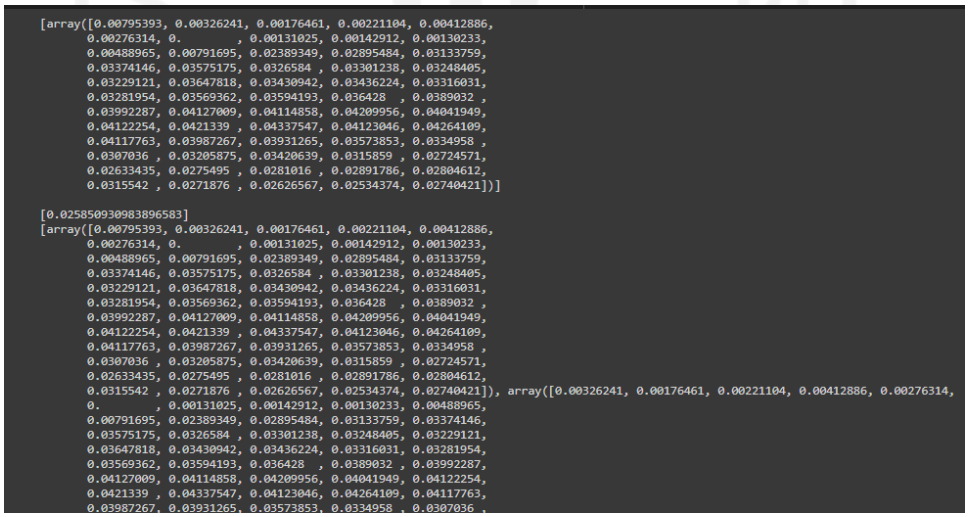
Gambar 4.21 mendefinisikan variabel kosong bernama *x_train* dan *y_train* pada baris kode ke 7 dan 8, yang akan diisi oleh pembagian data yang akan dilakukan menggunakan data yang sudah di-*scaling* sebelumnya. Kemudian data akan dibagi menjadi 60 data *x_training* yang dimulai dari data dengan index 0 hingga 59 dan 1 data *y_train* yang diambil dari indeks ke 60. Data dengan indeks 60 tersebut merupakan target dari prediksi setiap 60 data awal yang didefinisikan pada fungsi *for* di baris kode ke 13. Pembagian selanjutnya adalah data dengan indeks 1 hingga 60 dan data dengan indeks 61, dan begitu seterusnya. Hasil dari pembagian data tersebut ditunjukkan pada gambar 4.22.

```
# Create the training dataset
# Create the scaled training dataset
train_data = scaled_data[0:training_data_len, :]

# Split the data into x_train and y_train
# x_train is independent variable, y_train is dependentt variable
x_train = []
y_train = []

# First pass through x_train will contain 60 data indexed 0 to 59
# Then y_train for the pass through will contain 1data indexed 60
for i in range(60, len(train_data)):
  x_train.append(train_data[i-60:i, 0])
  y_train.append(train_data[i, 0])
  if i <= 61:
    print(x_train)
    print(y_train)
```

Gambar 4.21 Membuat variabel untuk menyimpan pembagian data



Gambar 4.22 Hasil pembagian data

Fungsi Mengubah Menjadi *Array* dan Fungsi *Reshape*

Gambar 4.23 mendefinisikan fungsi untuk melakukan *reshape* pada data. *Input* dari model LSTM mengaruskan untuk array 3 dimensi berupa (*number of samples, number of time steps, number of features*). Data yang dimiliki masih berbentuk 2 dimensi, jadi harus dilakukan *reshaping,* didefinisikan pada baris kode ke 8. Hasil dari proses *reshape* adalah (1489, 60, 1).

```
1.  # Convert the x_train and y_train to numpy array
2.  x_train, y_train = np.array(x_train), np.array(y_train)
3.
4.  # Reshape the dataset
5.  # LSTM network expect to be 3 dimentional input
6.  # Shape:(number of samples, number of time steps, number of
7.    feature)
8.  x_train=np.reshape(x_train, (x_train.shape[0], x_train.shape[1],
9.          1))
```

Gambar 4.23 Fungsi *reshape*

Membangun Model dan Fungsi *Compile*

Gambar 4.24 mendefinisikan fungsi untuk meembangun model dari algoritma LSTM. Pertama-tama memanfaatkan *library keras.models* untuk memanggil fungsi *Sequential()* yang terdapat pada baris kode ke 2, untuk menyiapkan basis model LSTM yang disimpan pada variabel model. Kemudian memanfaatkan *library keras.layer* untuk mengkonfigurasikan layer pada model LSTM, pada penelitian ini digunakan konfigurasi 3 layer dengan jumlah *neuron* 50, 50, dan 25 didefinisikan pada baris kode ke 3 hingga 7. Setelah model selesai dibangun, model akan di-*compile* dan ditambahkan fungsi o*ptimizer adam*. Model akan dilakukan proses *training* dengan *hyperparameter* disetel *batch_size* = 1 dan *epoch* = 3 yang didefinisikan pada baris kode ke 10 dan 13.

```
1.  # Build the LSTM model
2.  model = Sequential()
3.  model.add(LSTM(50, return_sequences = True, input_shape = (
4.          X_train.shape[1], 1)))
5.  model.add(LSTM(50, return_sequences = False))
6.  model.add(Dense(25))
7.  model.add(Dense(1))
8.
9.  # Compile the LSTM model
10. model.compile(optimizer = 'adam', loss = 'mean_squared_error')
11.
12. # Training the LSTM model
13. model.fit(x_train, y_train, batch_size = 1, epochs = 3)
```

Gambar 4.24 Membangun model LSTM dan proses *training*

Membuat Variabe*l X Test, Y Test*, dan Fungsi Pembagi Data

Gambar 4.25 medefinisikan variabel kosong bernama *x_test* dan *y_test*, yang akan diisi oleh pembagian data yang akan dilakukan menggunakan data yang sudah dibagi dan di-scaling sebelumnya pada baris kode ke 6 dan 7. Kemudian data akan dibagi menjadi 60 data *x_test* yang dimulai dari data dengan index 0 hingga 59 (*data_testing*) dan 1 data *y_test* (d*ata_testing*) yang diambil dari indeks ke 60. Data dengan indeks 60 tersebut merupakan target dari prediksi setiap 60 data awal didefinisikan pada fungsi for di baris kode ke 9. Pembagian selanjutnya adalah data dengan indeks 1 hingga 60 dan data dengan indeks 61, dan begitu seterusnya.

```
1.  # Create testing dataset
2.  # Create new array containing scaled value for testing dataset
3.  test_data = scaled[training_data_len - 60:, :]
4.  # Create the dataset x_test and y_test
5.  x_test = []
6.  y_test = dataset[training_data_len:, :]
7.  for i in range(60, lne(test_data)):
10.   x_test_append(test_data[i-60:i, 0])
```

Gambar 4.25 Membuat variabel untuk menyimpan data *testing*

Proses Prediksi

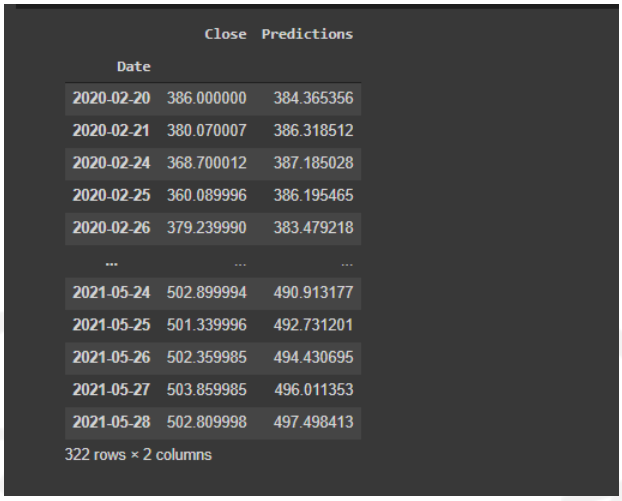
Gambar 4.26 mendefinisikan fungsi untuk melakukan *reshape* pada data. *Input* dari model LSTM mengaruskan untuk *array* 3 dimensi berupa (*number of samples, number of time steps, number of features*). Data yang dimiliki masih berbentuk 2 dimensi, jadi harus dilakukan *reshaping* kemudian data akan dilakukan prediksi. Setelah hasil prediksi didapatkan akan diterapkan fungsi *inverse* untuk membalikkan harga seperti semula. Hasil prediksi ditunjukkan pada gambar 4.27.

```
# Convert the data to numpy array
x_test = np.array(x_test)

# Reshape the dataset
# LSTM network expect to be 3 dimentional input
# Shape:(number of samples, number of time steps, number of
feature)
x_test = np.reshapee(x_test, (x_test.shape[0], x_test.shape[1],
         1))
x_test.shape

# Predict the LSTM model
lstm_prediction = model.predict(x_test)
# inverse the scaling data
lstm_prediction = scaler.inverse_transform(lstm_prediction)
```

Gambar 4.26 Proses prediksi



Date	Close	Predictions
2020-02-20	386.000000	384.365356
2020-02-21	380.070007	386.318512
2020-02-24	368.700012	387.185028
2020-02-25	360.089996	386.195465
2020-02-26	379.239990	383.479218
...	...	...
2021-05-24	502.899994	490.913177
2021-05-25	501.339996	492.731201
2021-05-26	502.359985	494.430695
2021-05-27	503.859985	496.011353
2021-05-28	502.809998	497.498413

322 rows × 2 columns

Gambar 4.27 Hasil prediksi dari model LSTM

4.1.3 *Linear Regression* dan *Decision Tree Regression*

Berikut adalah implementasi pada kode program menggunakan bahasa pemrograman Python dari *flowchart* gambar 3.4 algoritma *Linear Regression* dan *Decision Tree Regression*.

Import Library yang Dibutuhkan

Gambar 4.28 mendefinisikan *library* yang digunakan. Terlebih dahulu melakukan *import library* dan *package* yang dibutuhkan yang nantinya akan digunakan pada pemrosesan algoritma *Linear Regression* dan *Decision Tree Regression* yang telah disediakan oleh *library Scikit-Learn (sklearn)*.

```
1.  # Import libraries
2.  import math
3.  import numpy as np
4.  import pandas as pd
5.  import pandas_datareader as web
6.  from sklearn import metrics
7.  from sklearn.tree import DecisionTreeRegressor
8.  from sklearn.linear_model import LinearRegression
9.  from sklearn.model_selection import train_test_split
10. import matplotlib.pyplot as plt
11. plt.style.use('fivethirtyeight')
```

Gambar 4.28 *Import library*

Mendapatkan Data Saham

Gambar 4.29 mendefinisikan fungsi untuk mendapatkan data saham Netflix (NFLX) memanfaatkan *library pandas data reader* pada baris kode ke 2. Dengan memasukkan kode *ticker* dari saham, sumber data, rentang data dari awal hingga akhir. Kemudian ditampilkan data saham yang sudah didapatkan, data memiliki 1613 baris dan 6 kolom bertipe *pandas data frame* seperti ditunjukkan pada gambar 4.30.

```
1.  # Get the stock data
2.  Nflx_daily = web.DataReader('NFLX', data_source = 'yahoo', start
3.              = '2015-01-01', end = '2021-05-31'
4.
5.  # Show the stock data
6.  Print(nflx_daily.head())
7.  Print(nflx_daily.tail())
```

Gambar 4.29 Mendapatkan data saham

Date	High	Low	Open	Close	Volume	Adj Close
2015-01-02	50.331429	48.731430	49.151428	49.848572	13475000.0	49.848572
2015-01-05	49.258572	47.147144	49.258572	47.311428	18165000.0	47.311428
2015-01-06	47.639999	45.661430	47.347141	46.501431	16037700.0	46.501431
2015-01-07	47.421429	46.271427	47.347141	46.742859	9849700.0	46.742859
2015-01-08	47.835712	46.478573	47.119999	47.779999	9601900.0	47.779999
...	...	...	...	...	...	...
2021-05-24	504.250000	499.510010	501.049988	502.899994	2412600.0	502.899994
2021-05-25	506.369995	499.220001	506.000000	501.339996	2699500.0	501.339996
2021-05-26	504.140015	500.500000	502.339996	502.359985	2465300.0	502.359985
2021-05-27	505.100006	498.540009	501.799988	503.859985	3253800.0	503.859985
2021-05-28	511.760010	502.529999	504.399994	502.809998	2910300.0	502.809998

1613 rows × 6 columns

Gambar 4.30 Data saham NFLX

***Exploratory Data Analysis* (EDA) dan Visualisasi Data**

Gambar 4.31 mendefinisikan fungsi untuk melakukan proses *Exploratory Data Analysis* (EDA). Karena data yang digunakan sudah tersusun dengan rapi dan baik maka proses EDA tidak terlalu banyak dilakukan. Dengan proses EDA didapatkan beberapa *insight* bahwa jumlah baris data 1613 dengan 6 kolom dan *date* menjadi indeks data tersebut. Kemudian tipe data adalah *float64* dan tidak terdapat data yang kosong ditunjukkan pada gambar 4.32. Kemudian

data divisualisasikan dilakukan untuk mengetahui bagaimana data ditampilkan pada *line chart* dengan menggunakan kolom harga *close* seperti pada gambar 4.33.

```
1.  # Exploratory data analysis
2.  print(nflx_daily.shape)
3.  print(nflx_daily.info())
4.  print(nflx_daily.isnull())
5.
6.  # Visualize the close price
7.  plt.figure(figsize = (20, 10))
8.  plt.plot(nflx_daily['Close'])
9.  plt.title('NFLX Stock Closing Price', fontsize - 24)
10. plt.xlabel('Days')
11. plt.ylabel('Close Price $USD')
12. plt.show()
```

Gambar 4.31 *Exploratory Data Analysis* (EDA) pada data saham

```
Jumlah kolom dan baris :  (1613, 6)
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1613 entries, 2015-01-02 to 2021-05-28
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   High       1613 non-null   float64
 1   Low        1613 non-null   float64
 2   Open       1613 non-null   float64
 3   Close      1613 non-null   float64
 4   Volume     1613 non-null   float64
 5   Adj Close  1613 non-null   float64
dtypes: float64(6)
memory usage: 88.2 KB
None
Cek missing value : High        0
Low          0
Open         0
Close        0
Volume       0
Adj Close    0
dtype: int64
```

Gambar 4.32 Hasil *Exploratory Data Analysis*

Gambar 4.33 Visualisasi harga *close*

Mengambil Kolom *Close*

Gambar 4.34 mendefinisikan fungsi untuk mengambil kolom *close* pada *dataframe nflx_daily*. Hanya kolom *close* saja yang akan digunakan pada penelitian ini.

```
1.  # Get only the close price
2.  nflx_daily = nflx_daily[['Close']]
3.  nflx_daily.head()
```

Gambar 4.34 Mengambil kolom *close*

Membuat Variabel *Future Days* dan Kolom *Prediction*

Gambar 4.35 mendefinisikan variabel baru bernama *future_days* dengan nilai 60 yang merupakan berapa lama hari yang akan diprediksi nantinya pada baris kode ke 2. Kemudian terdapat fungsi untuk membuat kolom baru bernama *Prediction* yang nilainya diambil dari kolom *close* pada *dataframe nflx_daily*. Kolom *Prediction* nilainya dikenakan fungsi *shift* sejumlah nilai dari variabel *future_days*, hasilnya ditunjukkan pada gambar 4.36.

```
1.  # Create a variable to predict "x" days out into the future
2.  future_days = 60
3.
4.  # Create a new column (as a target) and shifted "x" days up
5.  nflx_daily['Prediction'] = nflx_daily[['Close']]
6.                            .shift(-future-days)
7.  print(nflx_daily.head())
8.  print(nflx_daily.tail())
```

Gambar 4.35 Membuat variabel *future days* dan kolom *prediction*

Date	Close	Prediction
2015-01-02	49.848572	59.527142
2015-01-05	47.311428	59.017143
2015-01-06	46.501431	59.154285
2015-01-07	46.742859	60.330002
2015-01-08	47.779999	60.494286
Date	**Close**	**Prediction**
2021-05-24	502.899994	NaN
2021-05-25	501.339996	NaN
2021-05-26	502.359985	NaN
2021-05-27	503.859985	NaN
2021-05-28	502.809998	NaN

Gambar 4.36 Kolom baru *Prediction*

Membuat Variabel *Independent* dan Variabel *Dependent*

Gambar 4.37 mendefinisikan fungsi untuk membuat variabel baru yang bertipe *array* bernama X (*independent*) dan y (*dependent*) diambil dari *dataframe nflx_daily yang* didefinisikan pada baris kode ke 3 dan 4.

```
1.  # Create the independent variable (X)
2.  # Convert to a numpy array and remove the last "x" days
3.  x = np.array(nflx_daily.drop(['Prediction'], 1))[:-future_days]
4.  y = np.array(nflx_daily['Prediction'])[:-future_days]
5.  print(X)
6.  print(y)
```

Gambar 4.37 Membuat variabel *independent* dan *dependent*

Fungsi *Split Dataset*

Gambar 4.38 mendefinisikan *fungsi* untuk *split dataset* menjadi data *training* dan data *testing* dengan perbandingan 75% berbanding 25%, data dikenakan fungsi *random_state* agar data tidak mengalami pengacakan dan tetap runtut yang didefinisikan pada baris kode ke 2. Setelah dilakukan *split*, akan dicek dimensi dari data dengan hasil (1164, 1) data *training*, (389, 1) data *testing*.

```
1.  # Split the dataset into 75% training and 25Z% testing
2.  x_train, x_test, y_train, y_test = train_test_split(X, y,
3.  test_size = 0.25, random_state = 0)
4.
5.  # Check the shapes
6.  print("shape of x_train :", x_train.shape)
7.  print("shape of y_train :", y_train.shape)
8.  print("shape of x_test :", x_test.shape)
9.  print("shape of y_test :", y_test.shape)
```

Gambar 4.38 Membagi *dataset* menjadi data *training* dan *testing*

Proses *Training*

Gambar 4.39 mendefinisikan fungsi untuk memanggil kedua model yang memanfaatkan *library* dari *sklearn.linar_model* pada baris kode ke 3 dan *sklearn_tree* pada baris kode ke 6. Pada parameter yang tersedia, tinggal memasukkan data *x_train* dan *y_train*, dan proses *training* model akan dilakukan.

```
1.  # Create the model
2.  # Create the Linear Regression model
3.  lr = LinearRegression().fit(x_train, y_train)
4.
5.  # Create the Decision Tree Regression
6.  tree = DecisionTreeRegressor().fit(x_train, y_train)
```

Gambar 4.39 Pembuatan model dan proses *training*

Membuat Variabel *X Future*

Gambar 4.40 mendefinisikan fungsi untuk membuat variabel baru yaitu *x_future* yang berisikan nilai dari kolom *close* pada *data frame nflx_daily* yang hanya diambil sebanyak jumlah *future_days* dimulai dari data paling bawah.

```
1.  # Get the last "x" rows of the independent variable
2.  X_future = nflx_daily.drop(['Prediction'], 1)[:-future_days]
3.  X_future = x_future.tail(future_days)
4.  X_future = np.array(x_future)
5.  X_future
```

Gambar 4.40 Membuat variabel *x_future*

Proses Prediksi

Gambar 4.41 mendefinisikan fungsi untuk melakukan proses prediksi pada kedua model memanfaatkan *library* dari *sklearn.linar_model* pada baris kode ke 2 dan *sklearn_tree* pada baris kode ke 6. Pada parameter yang tersedia, tinggal memasukkan data *x_test* dan *y_test*, dan proses prediksi model akan dilakukan. Hasil dari prediksi kedua model ditunjukkan pada gambar 4.42, *array* yang bagian atas merupakan hasil prediksi model *Decision Tree Regression* dan yang bawah merupakan hasil prediksi model *Linear Regressio*n.

```
1.  # Show the Linear Regression model prediction
2.  lr_prediction = lr.predict(x_future)
3.  print(lr_prediction)
4.
5.  # Show the Decision Tree Regression prediction
6.  tree_prediction = tree.predict(x_future)
7.  print(tree_prediction)
```

Gambar 4.41 Proses prediksi

```
[504.57998657 493.32998657 539.80999756 504.54000854 523.05999756
 518.02001953 520.25       524.0300293  524.44000244 504.79000854
 482.88000488 523.10998535 535.09002686 520.80999756 502.85998535
 508.04998779 513.95001221 513.39001465 521.65997314 539.41998291
 540.66998291 544.5300293  546.98999023 488.92999268 539.80999756
 503.05999756 553.72998047 482.83999634 504.20999146 546.53997803
 554.44000244 549.57000732 508.8999939  508.77999878 505.54998779
 510.29998779 505.54998779 506.51998901 509.         513.4699707
 491.35998535 503.17999268 496.07998657 499.54998779 549.57000732
 486.69000244 508.8999939  484.98001099 486.66000366 493.36999512
 508.89001465 486.27999878 487.70001221 501.67001343 497.89001465
 502.8999939  501.33999634 502.35998535 503.85998535 502.80999756]

[528.06714201 525.02171222 506.41762672 513.72846931 515.80752815
 534.54827939 531.97146891 536.90066971 544.77765638 546.29057114
 540.88304025 539.34087402 526.79818811 526.3003769  531.32722161
 542.79618188 536.66641963 552.42033339 534.97775773 532.96703176
 513.14281431 521.34192463 522.81578778 511.78607621 507.05207388
 520.26822878 513.50395979 510.69286944 514.39219773 596.93946614
 590.59493493 576.27577313 568.08649273 573.11327787 535.38775495
 550.34127455 544.27984517 550.77075289 559.67261346 551.17097976
 563.57694036 562.23971316 559.43836337 570.32169844 574.7336069
 568.87711655 567.83270195 568.5745336  562.77660577 559.73117598
 551.92252218 545.63661306 557.71073923 564.79704251 558.24757227
 550.58529498 562.09333665 559.34077903 532.86944742 523.68451481]
```

Gambar 4.42 Hasil prediksi dari kedua model

4.2 Visualisasi

Berikut adalah hasil visualisasi dari hasil prediksi yang telah dilakukan sebelumnya pada masing-masing algoritma.

4.2.1 *Temporal Convolutional Network* (TCN)

Setelah melakukan prediksi dengan model yang telah dibangun, kemudian memvisualisasikan hasilnya, untuk mengetahu tren harga yang diprediksi dengan tren aktual. Gambar 4.43 mendefinisikan fungsi untuk memvisualisasikan hasil prediksi.

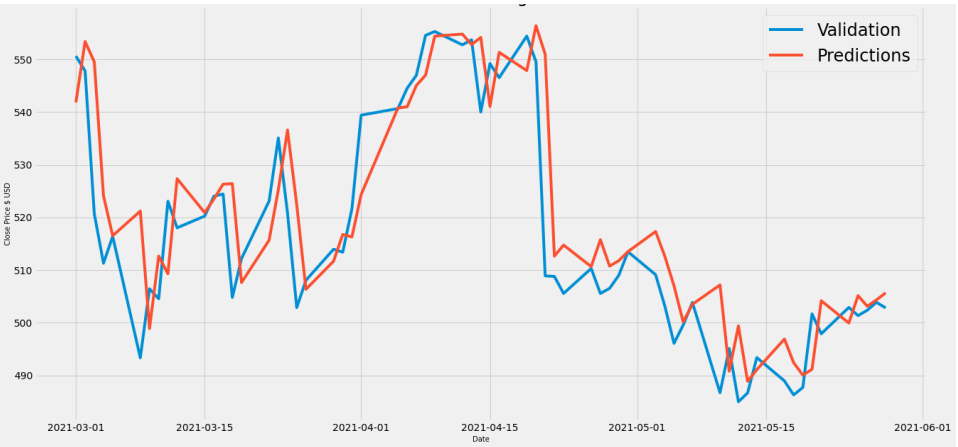
```
# Plot the data
valid['Prediction'] = backtest

# Visualize the prediction
plt.figure(figsize = (20, 20))
plt.plot(train['Close'])
plt.plot(valid['Close', 'Predictions'])
plt.title('TCN Model - NFLX Closing Price Prediction')
plt.xlabel('Date', fontsize = 10)
plt.ylabel('Close Price $USD', fontsize = 10)
plt.legend(['Train', 'Validation', 'Predictions'], loc = 'best',
           fotnsize = 24)
plt.show()
```

Gambar 4.43 Fungsi untuk menampilkan visualisasi model LSTM

Berikut adalah visualisasi dari prediksi yang dilakukan menggunakan algoritma TCN selama 60 hari. Gambar 4.44 dan 4.45 menujukkan hasil visulalisasi. Dapat dilihat pada visualisasi yang dihasilkan oleh hasil prediksi model TCN menunjukkan tren yang terjadi mampu mengikuti tren aktualnya. Selisih harga antara harga aktual dengan harga prediksi cukup kecil.

Gambar 4.44 Visualisasi prediksi model TCN secara keseluruhan



Gambar 4.45 Visualisasi prediksi model TCN secara mendetail

4.2.2 *Long Short Term Memory* (LSTM)

Setelah melakukan prediksi dengan model yang telah dibangun, kemudian memvisualisasikan hasilnya, untuk mengetahu tren harga yang diprediksi dengan tren aktual. Gambar 4.46 mendefinisikan fungsi untuk memvisualisasikan hasil prediksi.

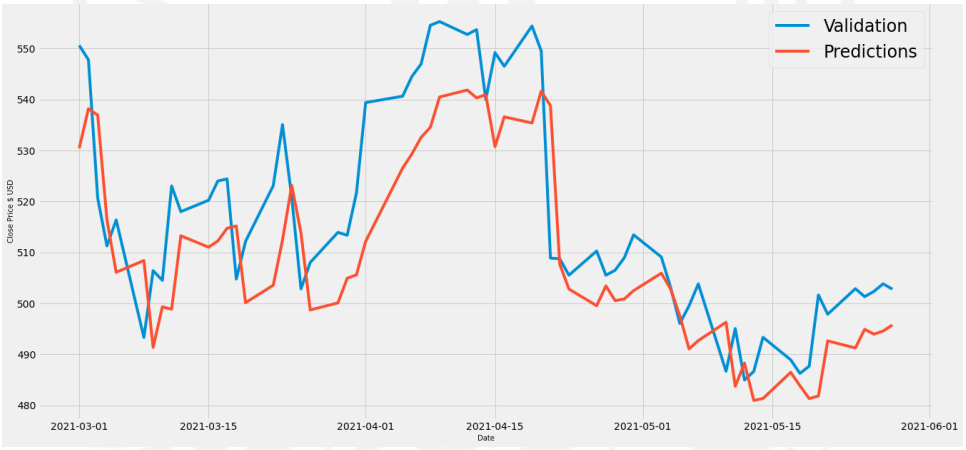
```
# Plot the data
train = df[:training_data_len]
valid = df[training_data_len:]
valid['Predictions'] = lstm_prediction

# Visualize the prediction
plt.figure(figsize = (20, 20))
plt.plot(train['Close'])
plt.plot(valid['Close', 'Predictions'])
plt.title('LSTM Model - NFLX Closing Price Prediction')
plt.xlabel('Date', fontsize = 10)
plt.ylabel('Close Price $USD', fontsize = 10)
plt.legend(['Train', 'Validation', 'Predictions'], loc = 'best',
           fotnsize = 24)
plt.show()
```

Gambar 4.46 Fungsi untuk menampilkan visualisasi model LSTM

Berikut adalah visualisasi dari prediksi yang dilakukan menggunakan algoritma LSTM selama 60 hari. Gambar 4.47 dan 4.48 menujukkan hasil visulalisasi. Dapat dilihat pada visualisasi yang dihasilkan oleh hasil prediksi model LSTM menunjukkan tren yang terjadi mampu mengikuti tren aktualnya. Selisih harga antara harga aktual dengan harga prediksi tidak terlalu jauh.

Gambar 4.47 Visualisasi prediksi model LSTM secara keseluruhan



Gambar 4.48 Visualisasi prediksi model LSTM secara mendetail

4.2.3 *Linear Regression* dan *Decision Tree Regression*

Setelah melakukan prediksi dengan model yang telah dibangun, kemudian memvisualisasikan hasilnya, untuk mengetahu tren harga yang diprediksi dengan tren aktual. Gambar 4.49 dan Gambar 4.52 mendefinisikan fungsi untuk memvisualisasikan hasil prediksi.

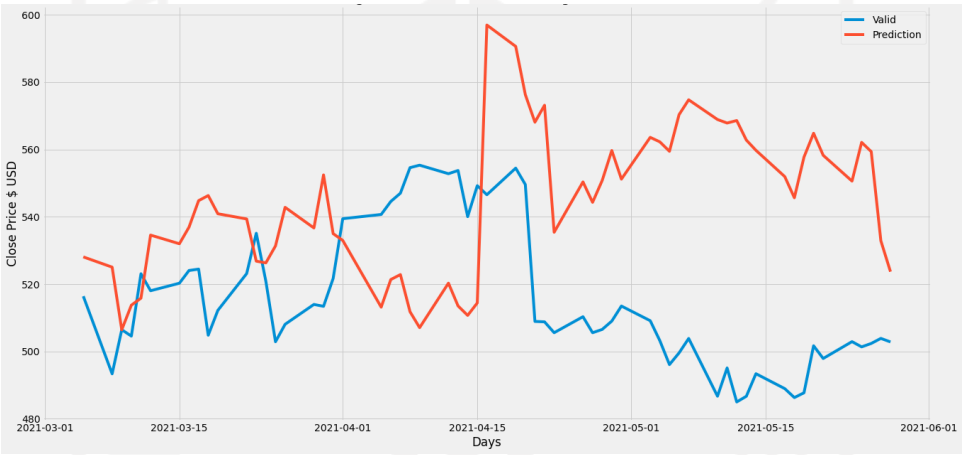
```
# Plot the data
valid = nflx_daily[X.shape[0]:]
valid['Predictions'] = lr_prediction

# Visualize the prediction
Plt.figure(figsize = (20, 20))
plt.plot(train['Close'])
plt.plot(valid['Close', 'Predictions'])
plt.title('LSTM Model – NFLX Closing Price Prediction')
plt.xlabel('Date', fontsize = 10)
plt.ylabel('Close Price $USD', fontsize = 10)
plt.legend(['Train', 'Validation', 'Predictions'], loc = 'best',
           fotnsize = 24)
plt.show()
```

Gambar 4.49 Fungsi untuk menampilkan visualisasi model *Linear Regression*

Berikut adalah visualisasi dari prediksi yang dilakukan menggunakan algoritma *Linear Linear Regression* selama 60 hari. Gambar 4.50 dan 4.51 menujukkan hasil visulalisasi. Dapat dilihat pada visualisasi yang dihasilkan oleh hasil prediksi model *Linear Regression* menunjukkan tren yang terjadi tidak mampu mengikuti tren aktualnya. Selisih harga antara harga aktual dengan harga prediksi terlalu jauh. Hal ini dapat dikategorikan sebagai *underfitting*, karena model tidak dapat menangkap pola yang terjadi dengan baik sehingga hasil prediksi mendapatkan akurasi yang rendah. Tren yang diprediksi juga berkebalikan dengan tren aktualnya, tren yang diprediksi mengalami kenaikan harga sedangkan tren aktualnya cenderung mengalami penurunan harga.

Gambar 4.50 Visualisasi prediksi model *Linear Regression* secara keseluruhan

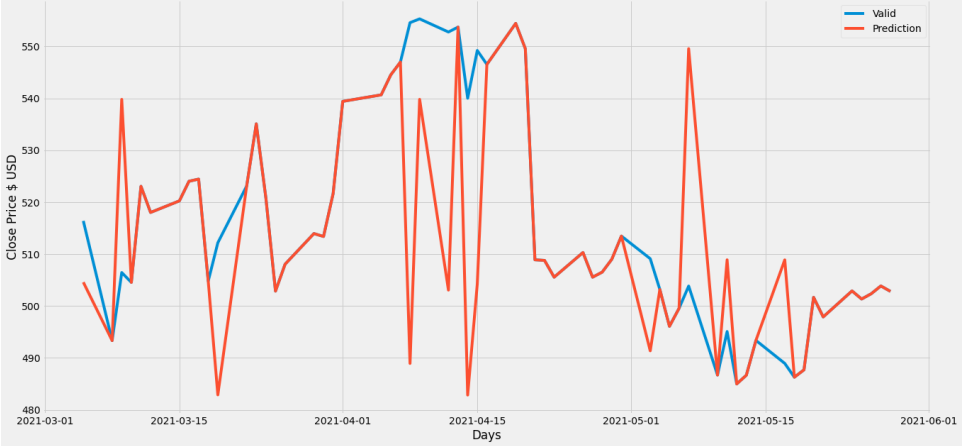


Gambar 4.51 Visualisasi prediksi model *Linear Regression* secara mendetail

```
1.  # Plot the data
2.  valid = nflx_daily[X.shape[0]:]
3.  Valid['Predictions'] = tree_prediction
4.
5.  # Visualize the prediction
6.  Plt.figure(figsize = (20, 20))
7.  Plt.plot(train['Close'])
8.  Plt.plot(valid['Close', 'Predictions'])
9.  Plt.title('LSTM Model – NFLX Closing Price Prediction')
10. Plt.xlabel('Date', fontsize = 10)
11. Plt.ylabel('Close Price $USD', fontsize = 10)
12. Plt.legend(['Train', 'Validation', 'Predictions'], loc = 'best',
13. Fotnsize = 24)
14. Plt.show()
```

Gambar 4.52 Fungsi untuk menampilkan visualisasi model *Decision Tree Regression*

Gambar 4.53 Visualisasi prediksi model *Decision Tree Regression* secara keseluruhan



Gambar 4.54 Visualisasi prediksi model *Decision Tree Regression* secara mendetail

Berikut adalah visualisasi dari prediksi yang dilakukan menggunakan algoritma *Decision Tree Regression* selama 60 hari. Gambar 4.53 dan 4.54 menujukkan hasil visulalisasi. Dapat dilihat pada visualisasi yang dihasilkan oleh hasil prediksi model *Decision Tree Regression* menunjukkan tren yang terjadi mampu mengikuti tren aktualnya. Selisih harga antara harga aktual dengan harga prediksi sangat dekat, namun pada beberapa titik harga prediksi mengalami lonjakan yang tajam. Hal ini dapat dikategorikan sebagai *overfitting*, karena model terlalu fokus pada data *training* sehingga jika memprediksi data lain akan mendapatkan akurasi yang rendah. Tren yang diprediksi terlalu mirip dengan tren aktualnya.

4.3 Evaluasi

Setelah melakukan prediksi dengan model yang telah dibangun dan memvisualisasikan hasilnya, selanjutnya adalah melakukan perhitungan menggunakan matrik *error* yang terjadi antara hasil prediksi dengna harga aktual. Gambar 4.55 mendefinisikan fungsi untuk menghitung matrik.

```
1.    # TCN Model evaluation
2.   Print("TCN Metric")
3.   Print("Mean Absolute Error: ", metrics.mean_absolute_error(
4.         val, backtest))
5.   Print("Mean Squared Error:", metrics.mean_squared_error(
6.         val, backtest))
7.   Print("Root Mean Squared Error:", np.sqrt(metrics.
8.  mean_squared_error(val, backtest)))
9.
10.  # LSTM Model evaluation
11.  Print("TCN Metric")
12.  Print("Mean Absolute Error: ", metrics.mean_absolute_error(
13.        Y_test, lstm_prediction))
14.  Print("Mean Squared Error:", metrics.mean_squared_error(
15.        Y_test, lstm_prediction))
16.  Print("Root Mean Squared Error:", np.sqrt(metrics.
17.        mean_squared_error(Y_test, lstm_prediction)))
18.
19.  # Linear Regression Model evaluation
20.  Print("TCN Metric")
21.  Print("Mean Absolute Error: ", metrics.mean_absolute_error(
22.        X_future, lr_prediction))
23.  Print("Mean Squared Error:", metrics.mean_squared_error(
24.        X_future, lr_prediction))
25.  Print("Root Mean Squared Error:", np.sqrt(metrics.
26.        mean_squared_error(x_future, lr_prediction)))
27.
28.  # Decision Tree Regression Model evaluation
29.  Print("TCN Metric")
30.  Print("Mean Absolute Error: ", metrics.mean_absolute_error(
31.        X_future, tree_prediction))
32.  Print("Mean Squared Error:", metrics.mean_squared_error(
33.        X_future, tree_prediction))
34.  Print("Root Mean Squared Error:", np.sqrt(metrics.
35.        mean_squared_error(x_future, tree_prediction)))
```

Gambar 4.55 Fungsi untuk menghitung matrik *error*

Tabel 4.1 Perbandingan Matrik

Matrik	TCN	LSTM	Linear Regression	Decision Tree Regression
MAE	9.865	10.710	11.894	30.868
MSE	98.318	117.391	141.765	1345.383
RMSE	9.890	10.843	11.906	36.679

Berdasarkan hasil prediksi yang sudah didapatkan, dan nilai matrik error yang terdapat pada tabel 4.1, implementasi dari model Temporal Convolutional Network (TCN) memiliki performa yang dapat mengungguli algoritma-algoritma pembanding lainnya. Dengan nilai RMSE 9.890 yang merupakan nilai paling rendah. Kemudian performa dari TCN dapat diikuti oleh algoritma yang cukup populer digunakan pada yaitu Long Short Term Memory (LSTM) yang memiliki selisih nilai RMSE sangat kecil yaitu 10.843 dengan nilai RMSE TCN. Sedangkan untuk algoritama yang tidak dikhususkan untuk mengolah data time series hasil yang didapatkan sangat buruk. Namun, untuk algoritma Linear Regression sendiri mendapatkan nilai RMSE yang cukup rendah yaitu 11.906, sedangkan algoritma Decision Tree Regression mendapatkan nilai RMSE yang paling buruk yaitu 36.679. Proses dilatasi yang dilakukan oleh algoritma TCN dapat mengenali data yang jauh di masa lalu, sehingga model dapat mengenali pola dari data lebih baik dan tidak tergantung dengan data jangka pendek saja. Walaupun hasil yang didapatkan TCN pada penelitian ini tidak terlalu signifikan dibandingkan dengan LSTM.

Sedangkan jika performa model algoritma dilihat dari sisi visualisai, prediksi yang dihasilkan algoritma TCN dan LSTM mampu memberikan prediksi tren yang dapat mengikuti tren aktualnya dan memiliki selisih harga yang tidak terlalu besat. Untuk algoritma Linear Regression, hasil prediksi tren yang didapatkan berkebalikan dengan tren aktualnya sehingga dapat dikatakan model tersebut mengalami underfitting karena tidak dapat mengenali pola-pola dengan baik. Sedangkan pada algoritma Decision Tree Regression, hasil prediksi tren yang didapatkan terlalu mirip dengan tren aktualnya atau biasa disebut prediksinya mengalami overfitting. Model tersebut terlalu terpaku pada data yang digunakan untuk proses training sehingga model tidak dapat memprediksi dengan baik pada data baru yang diberikan.

BAB V
PENUTUP

5.1 Kesimpulan

Pada penelitian ini dapat diperoleh kesimpulan sebagai berikut:

a. Algoritma *Temporal Convolutional Network* (TCN) dapat diterapkan untuk proses prediksi tren pergerakan harga saham dan algoritma *Long Short Term Memory* (LSTM) juga dapat mengikuti performa dari algoritma TCN.
b. Algoritma TCN dapat dibangun menggunakan *library Darts* yang khusus dikembangkan untuk mengolah data *time series* dan terdapat modul algoritma TCN, kemudian dilakukan proses *hyperparameter tuning* disesuaikan dengan data yang diolah.
c. Baik TCN maupun LSTM memiliki nilai error yang cukup kecil, yaitu dengan nilai RMSE TCN adalah 9.890 dan RMSE LSTM adalah 10.843. Sedangkan untuk algoritma *Linear Regression* dan *Decision Tree Regression* memiliki nilai *error* yang cukup besar yaitu 11.906 dan 36.679, walaupun nilai RMSE *Linear Regression* dapat dikatakan mendekati nilai dari TCN dan LSTM. Niali RMSE. Sehingga algoritma yang tidak dikembangkan untuk mengolah data *time series* tidak cocok diterapkan untuk prediksi tren pergerakan harga saham.
d. Visualisasi yang dihasilkan oleh algoritma TCN dan LSTM mampu mengikuti tren harga saham yang terjadi, dengan selisih harga yang cukup kecil.
e. Algoritma TCN dan LSTM dapat dimanfaatkan untuk membantu investor dalam memprediksi tren pergerakan harga saham pada bursa saham, namun tidak serta merta digunakan begitu saja. Harus digunakan oleh profesional dan memiliki penetahuan tentang dunia investasi saham dengan baik serta siap menerima resiko yang akan ditanggung.

5.2 Saran

Setalah didapatkan kesimpulan dari penelitian yang dilakukan, penelitian ini amsih terdapat banyak kekurangan yang harus diperbaiki kedepannya untuk dapat memberikan hasil yang lebih optimal dan bermanfaat. Berikut adalah saran yang dapat dipertimbangkan untuk pengembangan penelitian selanjutnya:

a. Data saham yang digunakan tidak hanya satu emiten saja, diharapkan model algoritma dapat diterapkan diberbagai emiten saham atau saham-saham dalam suatu indeks dan sektor saham tertentu.
b. Melakukan eksplorasi yang lebih mendalam pada algoritma yang digunakan, mencoba menggunakan berbagai kemungkinan konfigurasi pada algoritma tersebut untuk mendapatkan hasil yang lebih maksimal.
c. Diharapkan model algoritama yang sudah dibangun dapat diintegrasikan pada platform jual beli saham yang digunakan oleh investor.

DAFTAR PUSTAKA

Agusta Nanda, R., Rizal, S., & Yuswaliani, S. (2016). Analisis Fundamental dan Analisis Teknikal. https://www.academia.edu/32854302/Analisis_fundamental

Ahmar, A. S. (2018). *Sutte indicator: An Approach to Predict the Direction of Stock Market Movements*. *Songklanakarin Journal of Science and Technology*, 40(5), 1228–1231. https://doi.org/10.14456/sjst-psu.2018.150

axa.co.id. (2019). Awal Mula Sejarah Platform Streaming Netflix yang Perlu Kamu Ketahui! https://portal.axa.co.id/direct/Tips/Detail/awal-mula-sejarah-netflix-yang-perlu-kamu-ketahui

Bai, S., Kolter, J. Z., & Koltun, V. (2018). *An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling*. http://arxiv.org/abs/1803.01271

BEI. (n.d.). Bursa Efek Indonesia. https://www.idx.co.id/produk/saham/

CNBC Indonesia. (2021, June 29). Jumlah Investor Pasar Modal RI. https://www.cnbcindonesia.com/market/20210629153854-17-256818/naik-56-jumlah-investor-pasar-modal-ri-mencapai-388-juta

Dicoding. (2020). Apa Itu Kecerdasan Buatan? Berikut Pengertian dan Contohnya. https://www.dicoding.com/blog/kecerdasan-buatan-adalah/

Filbert, R. (2020a). Investasi Saham Ala *Swing Trader* Dunia (H. Yulianto, Ed.; Cetakan Keempat belas). Elex Media Komputindo.

Filbert, R. (2020b). Investasi Saham Ala *Swing Trader* Dunia (H. Yulianto, Ed.; 14th ed.). Elex Media Komputindo.

Hafizah, N., Noviani, E., & Perdana INTISARI, H. (2019). Analisis Teknikal Saham Lq-45 Menggunakan Indikator *Bollinger Bands*. Buletin Ilmiah Math. Stat. dan Terapannya (Bimaster) (Vol. 08, Issue 4).

Ika Ramadhani, P. (2020, July 2). *S*&P 500 dan *Nasdaq* Menguat Didorong Perkembangan Positif Vaksin Corona. https://www.liputan6.com/saham/read/4294060/sampp-500-dan-nasdaq-menguat-didorong-perkembangan-positif-vaksin-corona

investasi.kontan.co.id. (2021, March). Sampai Maret 2021, BEI Catat Kenaikan Jumlah Investor Sebesar 27%. https://investasi.kontan.co.id/news/sampai-maret-2021-bei-catat-kenaikan-jumlah-investor-sebesar-27

Jati Lantang M, N. (2020). Perbandingan Antara SVM dan CNN Untuk Mendeteksi Objek Kapal Pada Citra Satelit.

Karno, A. S. B. (2020). Prediksi Data *Time Series* Saham Bank BRI Dengan Mesin Belajar LSTM (*Long ShortTerm Memory*). *Journal of Informatic and Information Security*, *1*(1). https://doi.org/10.31599/jiforty.v1i1.133

Kim, T., & Kim, H. Y. (2019). *Forecasting stock prices with a feature fusion LSTM-CNN model using different representations of the same data. PLoS ONE*, *14*(2). https://doi.org/10.1371/journal.pone.0212320

Kusumodestoni, R. H., & Sarwido, S. (2017). Komparasi Model *Support Vector Machines* (SVM) dan *Neural Network* Untuk Mengetahui Tingkat Akurasi Prediksi Tertinggi Harga Saham. Jurnal Informatika Upgris, *3*(1). https://doi.org/10.26877/jiu.v3i1.1536

Lässig, F. (2020, October 28). *Temporal Convolutional Networks and Forecasting*. https://medium.com/unit8-machine-learning-publication/temporal-convolutional-networks-and-forecasting-5ce1b6e97ce4

Lestari, W. R., & Pranyoto, D. E. (2015). Faktor Psikologi yang Membentuk Perilaku Keuangan (*Behavioral Finance*) Investor dalam Transaksi Saham pada Pasar Modal di Lampung *Psychological Factors Shaping the Behavior of Finance (Behavioral Finance) Investor Shares in the Capital Market Transactions in* Lampung. Winda Rika Lestari dan Edi Pranyotol (Vol. 5, Issue 1).

Liu, Y., Dong, H., Xingmei, W., & Han, S. (2019). *Time Series Prediction Based on Temporal Convolutional Network. 2019 IEEE/ACIS 18th International Conference on Computer and Information Science (ICIS)*, 300–305.

Mehtab, S., Sen, J., & Dutta, A. (n.d.). *Stock Price Prediction Using Machine Learning and LSTM-Based Deep Learning Models End-to-End Connectivity Management in Heterogeneous Wireless Networks View project Forecasting of Tourist Inflow for Infrastructure and Logistics Planning View project Sidra Mehtab NSHM Knowledge Campus Stock Price Prediction Using Machine Learning and LSTM-Based Deep Learning Models*. https://www.researchgate.net/publication/344324240

money.kompas.com. (2021). Jumlah Pelanggan Melonjak di Tengah Pandemi, Netflix Raup Pendapatan Mencapai Rp 350 Triliun pada akhir tahun 2020 ini. https://money.kompas.com/read/2021/01/20/165327826/jumlah-pelanggan-melonjak-di-tengah-pandemi-netflix-raup-pendapatan-rp-350

Nurhikmat, T. (2018). Implementasi Deep Learning Untuk Image Classification Menggunakan Algoritma Convolutional *Neural Network* (CNN) pada Citra Wayang Golek.

https://dspace.uii.ac.id/bitstream/handle/123456789/7843/TUGAS%20AKHIR_TRIANO%20NURHIKMAT_14611209_STATISTIKA_UII.pdf?sequence=1&isAllowed=y

Olah, C. (2015, August 27). Github *Repository Understanding LSTM Networks*. https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Pham, D. T., & Pham, P. T. N. (1999). *Artificial Intelligence in Engineering*. *International Journal of Machine Tools and Manufacture*, *39*(6). https://doi.org/10.1016/S0890-6955(98)00076-5

Pramudya, R., & Pramudya, R. (2020). *Technical Analysis to Determine Buying and Selling Signal in Stock Trade. International Journal of Finance & Banking Studies (2147-4486)*, *9*(1), 58–67. https://doi.org/10.20525/ijfbs.v9i1.666

R Tristia, R. (2019, October 30). Psikologi Trading dan Pengendalian Emosi untuk Trader. https://id.catinstitute.org/market-story/read/psikologi-trading

Rich, E., Knight, K., & B Nair, S. (2019). *Artificial Intelligence* (3rd ed.). The McGrow Hill.

Roziq, M. (2021). Mengenal Apa itu AI (*Artificial Intelligence*)? Berikut Tipe, Bentuk dan Penerapannya dalam Kehidupan. https://caraguna.com/apa-itu-ai-artificial-intelligence-berikut-tipe-bentuk-dan-penerapannya-dalam-kehidupan/

Sekar. (2020, July 31). Mengenal Jenis-Jenis *Chart* Saham Demi Keuntungan. https://ajaib.co.id/mengenal-jenis-jenis-chart-saham/

tradingview.com. (2021). *Netflix,Inc. (NFLX) Stock*. https://id.tradingview.com/chart/?symbol=FX_IDC:USDIDR&source=unauth_header&feature=launch_chart

Tri Retno, I. (2021, March 4). Mengenal tentang Tren *Sideways* atau Konsolidasi. https://www.finansialku.com/sideways/

Wan, R., Mei, S., Wang, J., Liu, M., & Yang, F. (2019). *Multivariate Temporal Convolutional Network: A Deep Neural Networks Approach for Multivariate Time Series Forecasting. Electronics (Switzerland)*, 8(8). https://doi.org/10.3390/electronics8080876

Wang, X., Wang, Y., Weng, B., & Vinel, A. (2020). *Stock2Vec: A Hybrid Deep Learning Framework for Stock Market Prediction with Representation Learning and Temporal Convolutional Network*. http://arxiv.org/abs/2010.01197

Wayan, D. (2018, February 6). Memahami Kecerdasan Buatan berupa *Deep Learning* dan *Machine Learning*. https://warstek.com/deepmachinelearning/

Zainuddin, Z., & Hartono, J. (1999). Manfaat Rasio Keuangan dalam Memprediksi Pertumbuhan Laba: Suatu Studi Empiris pada Perusahaan Perbankan yang Terdaftar di Bursa Efek Jakarta. 2(1), 66–90.

Zayini Anwar, M., & Habibi, S. (n.d.). Analisis Prediksi Performasi Arah Pergerakan Saham Apple (APPL) Menggukan Metode *Recurrent Neural Networks/Long Short Term Memory Networks* (RNN/LSTM).

LAMPIRAN