BAB III
METODE PENELITIAN

3.1 Analisis Kebutuhan

Dalam penelitian ini pastinya membutuhkan beberapa hal yang harus dipersiapkkan sebelum memulai untuk meneliti. Kebutuhan dalam pembuatan penelitian ini meliputi dua hal, yaitu kebutuhan penelitian dan kebutuhan sistem.

3.1.1 Kebutuhan Penelitian

Pada penelitian ini memiliki dua hal yang dibutuhkan yaitu perangkat keras dan lunak. Adapun kebutuhan perangkat keras seperti komputer untuk melakukan pengkodean serta pembutan laporan penelitian, buku sebagai bahan acuan dalam pengerjaan penelitian, dan media penyimpanan eksternal (*flashdisk*) untuk memudahkan mobilitas membawa data.

Kebutuhan perangkat lunak yang dibutuhkan dalam penelitian ini berupa *text editor* untuk membuka dan merubah isi *file* text yang dalam penelitian ini menggunakan Sublime, Python IDE sebagai dasar untuk menjalankan perintah menggunakan bahasa pemrograman Python, Microsoft Word digunakan dalam pembuatan laporan, aplikasi Excell digunakan dalam pembuatan *dataset*.

3.1.2 Kebutuhan Sistem

Pada kebutuhan sistem, untuk pembuatan penlitian membutuhkan data *tweet* yang berupa teks menggunakan bahasa yang akan diteliti pada penelitian ini. Serta dibutuhkan pula sebuah *library* yang digunakan untuk mempermudah berlangsungnya penelitian.

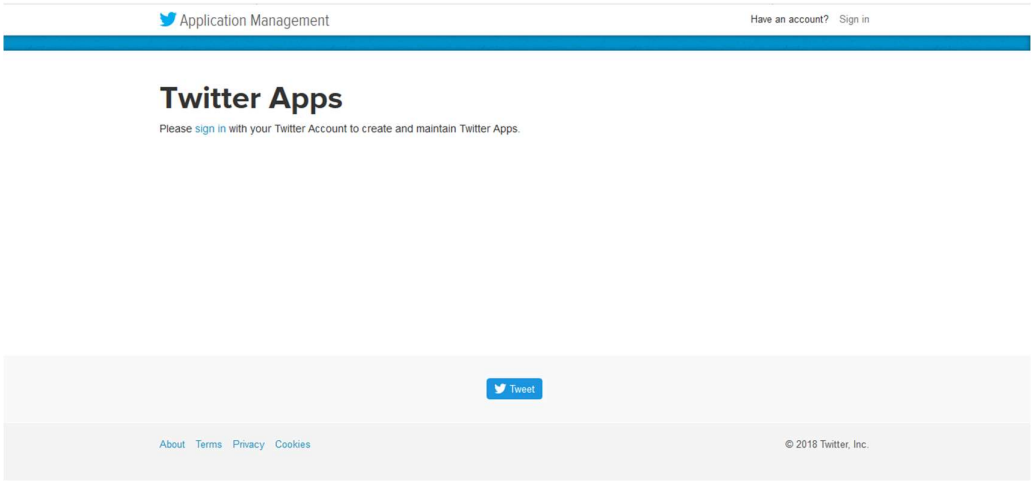
Library yang digunakan dalam penelitian ini adalah Scikit-learn dan Pandas. Scikit-learn merupakan sebuah *library* yang menyediakan fitur-fitur yang dibutuhkan untuk melakukan perhitungan-perhitungan metode *machine learning*. Sedangkan Pandas adalah *library* yang digunakan untuk *deliveriy data* yang akan digunakan pada Scikit-learn. Beberapa fitur yang digunakan dalam penelitian ini adalah sebagai berikut:

- Pandas read_excel(), digunakan untuk membaca file yang berekstensi .xlsx. Dengan menggunakan fungsi ini, peneliti bisa memilih kolom mana yang akan diambil dan memasukkan kedalam variabel.

- Scikit-learn tfidfVectorizer(), digunakan untuk membentuk *word vector* yang mana *word vector* langsung terbentuk dengan nilai TF-IDF dan N-Gram
- Scikit-learn train_test_split (), digunakan untuk membagi variabel yang sudah dibentuk dengan *library* Pandas menjadi *training* dan *test set*.
- Scikit-learn confusion_matrix(), digunakan untuk menampilkan tabel *confusion matrix*.

3.2 Data dan Sumber Data

Data yang digunakan dalam penelitian ini adalah data *tweet* yang didapatkan dari aplikasi Twitter. Data yang dikumpulkan berupa data teks berbahasa Indonesia, Jawa, dan Sunda.



Gambar 3.1 Halaman awal Apps Twitter

Adapun jumlah akun untuk masing-masing bahasa yaitu:

Bahasa Indonesia : 4 akun

Bahasa Jawa : 6 akun

Bahasa Sunda : 4 akun

Gambar 3.2 Contoh akun Twitter Bahasa Jawa

Dari beberapa akun yang sudah disebutkan menghasilkan beberapa data yang digunakan sebagai data latih yang akan digunakan, yang mana jumlah data dati setiap bahasa adalah sebagai berikut:

Bahasa Indonesia: 981 baris *tweet*

Bahasa Jawa: 799 baris *tweet*

Bahasa Sunda: 688 baris *tweet*

Sehingga keseluruhan data yang digunakan sebagai data latih adalah 2468 baris *tweet*. Namun setelah dilakukan *pre-processing,* jumlah data yang siap digunakan sebagai data latih adalah sejumlah 1305 baris *tweet.*

Gambar 3.3 Contoh akun Twitter Bahasa Sunda

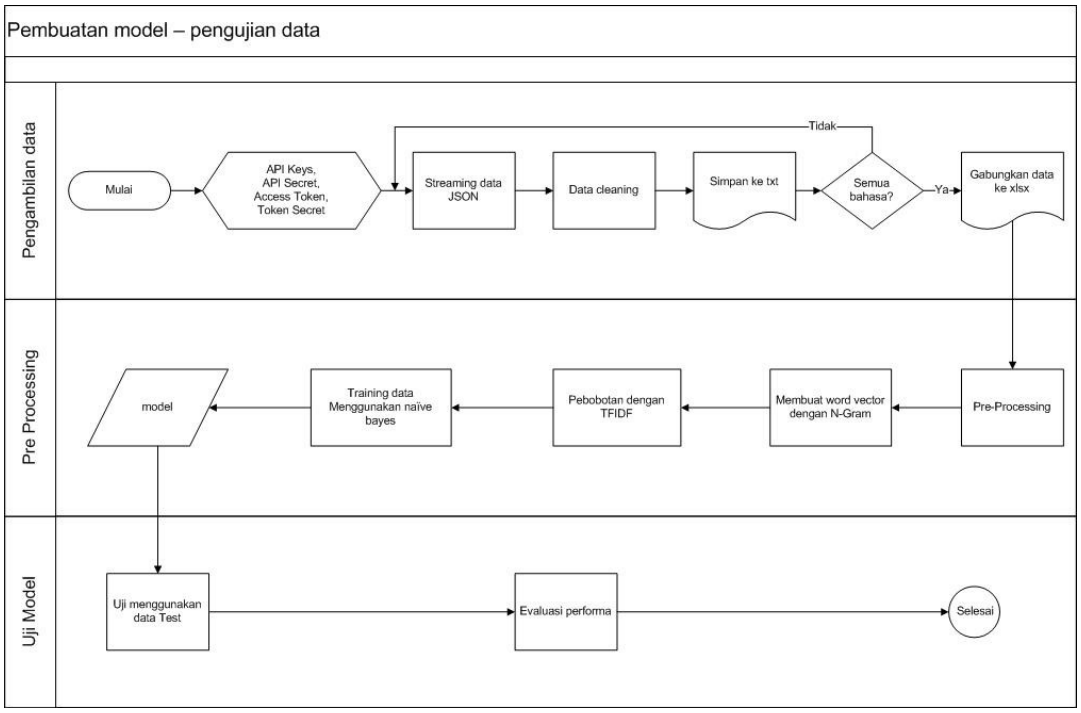
Akun dipilih jika terindikasi menggunakan bahasa-bahasa yang dimaksud, indikasinya dilakukan secara manual.Terhadap akun-akun yang telah terindikasi menggunakan bahasa-bahasa yang dimaksud, dilakukan suatu proses pengambilan data (*retrieve*), prosesnya sebagai berikut :

1. Masuk apps.twitter.com
2. Mendaftarkan akun twitter sebagai *developer*
3. Mendaftarkan nama aplikasi atau projek yang akan dibuat.
4. Mendapat *public* dan *secret key*
5. Menggunakan *public* dan *secret key* untuk *retrieve*data dari akun-akun terpilih.

Data didapat dalam bentuk JSON, yaitu suatu format data ringan yang digunakan dalam pertukaran data, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemah dan dibuat oleh komputer (JSON, n.d.).

3.3 Gambaran Sistem

Gambaran sistem adalah suatu perancangan yang menggambarkan proses sejak data diambil sampai menghasilkan keluaran/*output*. Gambar 3.4 adalah gambaran umum sistem.



Gambar 3.4 Gambaran Sistem

Berdasarkan Gambar 3.4, terdapat 3 subsistem yang masing-masing dijabarkan dalam paparan berikut ini:

3.3.1 Pengambilan Data.

Seperti yang telah dipaparkan pada subbab 3.2 tentang data dan sumber data, data diambil melalui Twitter API menggunakan *key* yang telah didapatkan setelah mendaftarkan akun dan sistem yang akan dibuat. Setelah data didapatkan dalam format JSON, dilakukan suatu fungsi *decode*atau pengubahan data yang berformat JSON menjadi sebuah objek, sehingga isi data bisa diambil. Dari banyak atribut yang ada pada objek, maka diambil data '*text'* yang merupakan isi dari *tweet* pengguna akun yang diambil datanya.

Karena penelitan ini akan membahas pendeteksian bahasa daerah yang digunakan pada teks, maka ada beberapa hal yang harus diperlakukan terlebih dahulu kepada teks yaitu*data cleaning. Data cleaning* adalah proses untuk menghilangkan nama akun, tanda baca, dan url pada teks. Dan proses ini dilakukan menggunakan program, sehingga dilakukan secara otomatis sebelum menyimpan hasil *decode* ke dalam bentuk txt.

Setelah data tersimpan kedalam format txt, maka dilakukan *datacleaning* lagi untuk menghapus dokumen-dokumen yang terindikasi kosong dikarenakan sebelumnya teks hanya

berisi url saja, namun para proses penghapusan dilakukan secara manual.Sampai pada tahapan ini, ada tiga buah *file* txt yang berisi sejumlah dokumen teksyang merepresentasikan masing-masing bahasa.

Sebelum masuk ke tahapan *pre-processing*, data dari 3 *file* digabung ke dalam format .xlsx dengan isi setiap barisnya meruppakan perwakilan dari setiap dokumen, dan pada kolom pertama berisi teks yang sudah dibersihkan pada proses sebelumnya, dan pada kolom kedua berisikan label atau bahasa yang digunakan dalam teks tersebut. Label pada dokumen tersebut diisi oleh peneliti sebagai basis untuk nantinya dilakukan *pre-processing* dan -*training.* Label juga bisa disebut sebagai kelas dari teks tersebut apabila dipandang dari segi klasifikasi.

3.3.2 *Pre-processing*

Pada subsistem ini, ada beberapa proses yang dilakukan secara sekuensial untuk mendapatkan *data training*, yaitu tokenisasi atau membuat data yang berupa teks menjadi kumpulan *array* kata, misal:

Kalimat	“saya sedang belajar”
Dirubah menjadi	[‘saya’],[‘sedang],[‘belajar’]

Setelah semua kata sudah ter-tokenisasi, maka langkah selanjutnya adalah membuat *word vector*. *Word vector* atau dalam Bahasa Indonesia bisa disebut vektor kata, yaitu membuat kalimat yang sudah menjadi kumpulan *array* menjadi suatu matriks, yang mana setiap baris matriks tersebut mewakili baris dokumen, sedangkan kolom pada matriks mewakili seluruh kata yang ada di seluruh teks yang ada, bisa digambarkan seperti berikut:

Kumpulan kalimat seperti berikut ([‘saya’],[‘sedang],[‘belajar’]),
([‘saya’],[‘ingin],[‘bermain]),
([‘kamu],[‘teman],[‘saya])

Akan dirubah menjadi matriks seperti ini:

Tabel 3.1 Tabel contoh *word vector*

saya	sedang	belajar	Ingin	bermain	teman	kamu
1	1	1	0	0	0	0
1	0	0	1	1	0	0
1	0	0	0	0	1	1

Setelah sudah berubah menjadi vektor kata, maka selanjutnya adalah memberikan pembobotan terhadap setiap kata pada setiap kalimat atau dokumen menggunakan *Unigram* dan TF-IDF menggunakan rumus yang bisa dilihat pada bab sebelumnya, maka *dataset* sudah siap untuk digunakan dalam training menggunakan perhitungan pada *naive bayes*.

3.3.3 Uji Model

Uji model dilakukan untuk mengetahui kinerja model. Nilainya didapat dengan menghitung *accuration*, *precision*, dan *recall* (tiga pendekatan perhitungan dalam PEM). Untuk menghitung nilai itu, maka dibutuhkan kalimat yang sudah diketahui terlebih dahulu kelasnya. Oleh karena itu dilakukan pengecekan silang antara setiap data dalam dataset, untuk menyilangkannya data akan dibagi menjadi dua bagian, bagian pertama (*data training*) sebagi data training, yaitu data yang dijadikan basis seperti yang dijelaskan sebelumnya, sedangkan bagian kedua (*data test*) dianggap sebagai teks baru yang belum diketahui kelasnya (kelasnya disembunyikan terlebih dahulu).

Setelah *data test* diujikan terhadap data training, maka akan menghasilkan daftar kelas-kelas dari *data test,* sebut sata prediksi kelas. Kemudian prediksi kelas dibandingkan dengan kelas yang sebenarnya dari data *test* yang disembunyikan sebelumnya. Sehingga dapat dilihat dan dihitung nilai *accuration*, *precision*, dan *recall* menggunakan cara-cara yang sudah dijelaskan pada bab sebelumnya.