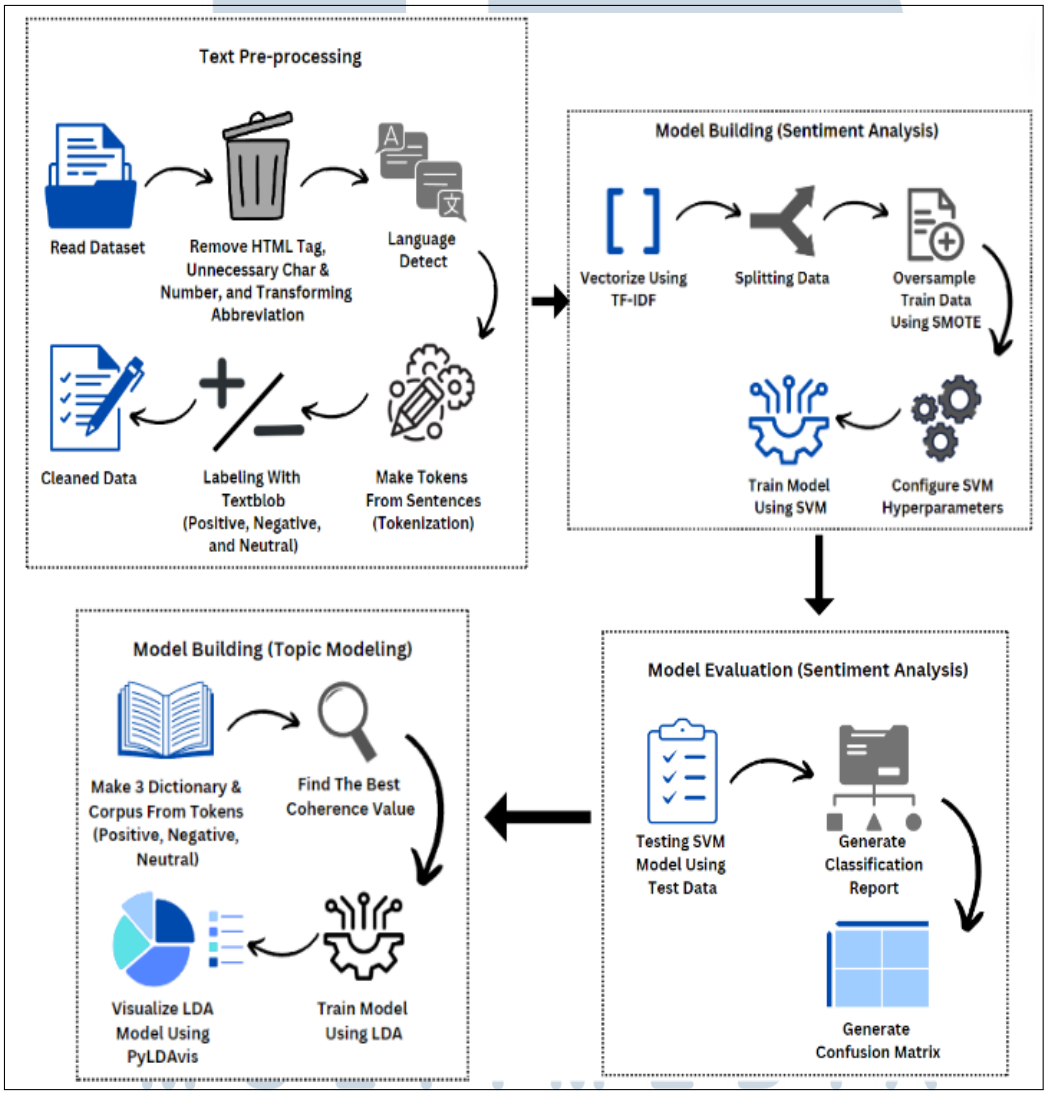
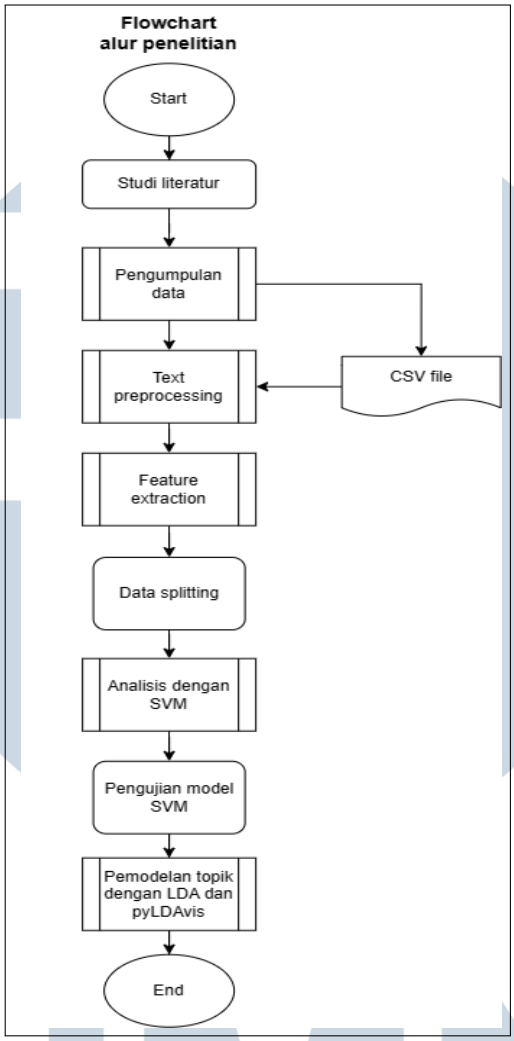
# BAB 3
# METODOLOGI PENELITIAN

## 3.1 Alur Penelitian

Gambaran teknis dan *flowchart* alur penelitian dari tahapan awal hingga akhir dapat dilihat pada Gambar 3.1 dan Gambar 3.2.




Gambar 3.1. Teknis alur penelitian

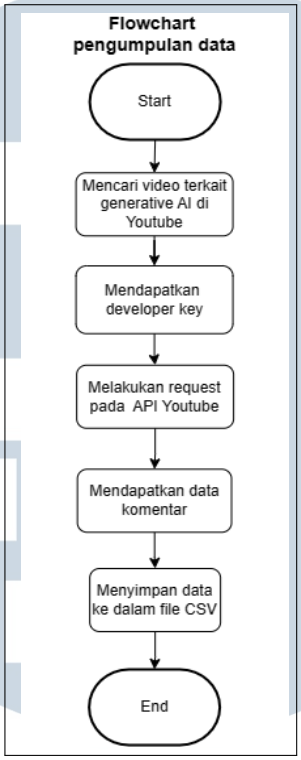



Gambar 3.2. Flowchart alur penelitian

## 3.2 Studi Literatur

Studi literatur dilakukan untuk mendapatkan berbagai pemahaman dan teori yang dapat membantu selama penelitian dilakukan. Literatur ini berguna sebagai pedoman utama dan juga mendukung berbagai pernyataan yang ada pada penelitian ini. Literatur yang dipelajari berasal dari jurnal-jurnal internasional yang berkaitan dengan *generative AI*, analisis sentimen, SVM, pemodelan topik, LDA, Youtube, dan berbagai jurnal penelitian yang membahas tentang cara meningkatkan performa dari algoritma yang digunakan. Studi literatur dilakukan dengan bantuan *website* Google Scholar.

## 3.3 Pengumpulan Data




Gambar 3.3. Flowchart pengumpulan data

*Flowchart* pengumpulan data dapat dilihat pada Gambar 3.3. Pengumpulan data dilakukan dengan menggunakan *library* google-api-python-client yang dapat di akses dengan menggunakan Jupyter Notebook dan bahasa pemrograman Python. Untuk mengakses API Youtube, diperlukan sebuah API *key* sebagai sebuah *credentials* untuk mengakses API. *Key* ini bisa didapatkan melalui *website* Google Cloud Platform dengan mencari Youtube Data API v3 pada menu APIs & Services. Setelah mendapatkan API *key*, *developer* dapat mengakses API Youtube dan melakukan *request* untuk *read* data sebanyak 10.000 *request* per hari. *Request* ini dapat digunakan untuk mendapatkan komentar pengguna pada suatu video dengan memanfaatkan id video yang terdapat pada *URL* Youtube. Pengumpulan data dilakukan selama 3 hari dengan menggunakan *library* google-api-python-client pada 3 video (bukan *shorts*) terkait dengan *generative AI* yang mempunyai *views* dan komentar terbanyak. Data yang diambil berupa nama *user*, waktu komentar, dan isi komentar. Setelahnya komentar-komentar tersebut di-*export* ke dalam *file* CSV-nya masing-masing dan digabungkan menggunakan fungsi *concat* pada

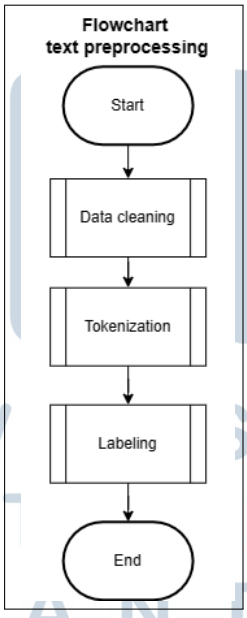
*library* pandas. Tampilan untuk hasil pengumpulan data dapat dilihat pada gambar 3.4.

| | author | published_at | updated_at | like_count | text |
|---|---|---|---|---|---|
| 0 | @PiXimperfect | 2022-08-02T17:42:20Z | 2022-08-02T17:44:25Z | 3393 | ← Correction: Yes, you can remove your creatio... |
| 1 | @MTechShorts49 | 2024-02-29T12:19:59Z | 2024-02-29T12:19:59Z | 0 | and it been a year what are your thouts on ai |
| 2 | @yjg30737 | 2024-02-28T06:45:30Z | 2024-02-28T06:45:30Z | 0 | There are people who keep saying "As many... |
| 3 | @wheretao6960 | 2024-02-05T15:44:00Z | 2024-02-05T15:44:00Z | 0 | don't get me wrong, midjourney is mind-blo... |
| 4 | @myrealitymedia | 2024-02-02T18:32:03Z | 2024-02-02T18:32:03Z | 0 | I can't believe it's been a year since you did... |
| ... | ... | ... | ... | ... | ... |
| 21872 | @will4282 | 2022-10-17T22:22:06Z | 2022-10-17T22:44:10Z | 50 | I can't wait to watch and, more importantl... |
| 21873 | @samankucher5117 | 2022-10-17T22:14:32Z | 2022-10-17T22:14:32Z | 18 | <a href="https://www.youtube.com/watch?v=tjSxF... |
| 21874 | @uhoh7545 | 2022-10-17T22:12:29Z | 2022-10-17T22:12:29Z | 16 | I'm more of a rice guy in terms of grains,... |
| 21875 | @samankucher5117 | 2022-10-17T22:10:39Z | 2022-10-17T22:10:39Z | 15 | ah he finally posted it . |
| 21876 | @dinoblaster736 | 2022-10-17T22:07:53Z | 2022-10-17T22:07:53Z | 26 | The long awaited video |

21877 rows × 5 columns

Gambar 3.4. Data frame dari CSV yang digunakan

## 3.4 Text Pre-processing

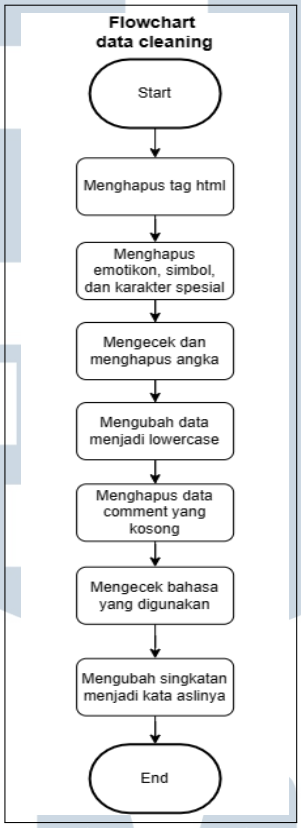



Gambar 3.5. Flowchart text pre-processing

*Flowchart text pre-processing* dapat dilihat pada Gambar 3.5. Tahap *pre-processing* dalam penelitian ini terdiri dari beberapa proses, yaitu *data cleaning*,

*tokenization*, *removing stopwords*, *lemmatization*, serta *labeling*.

### 3.4.1 Data Cleaning

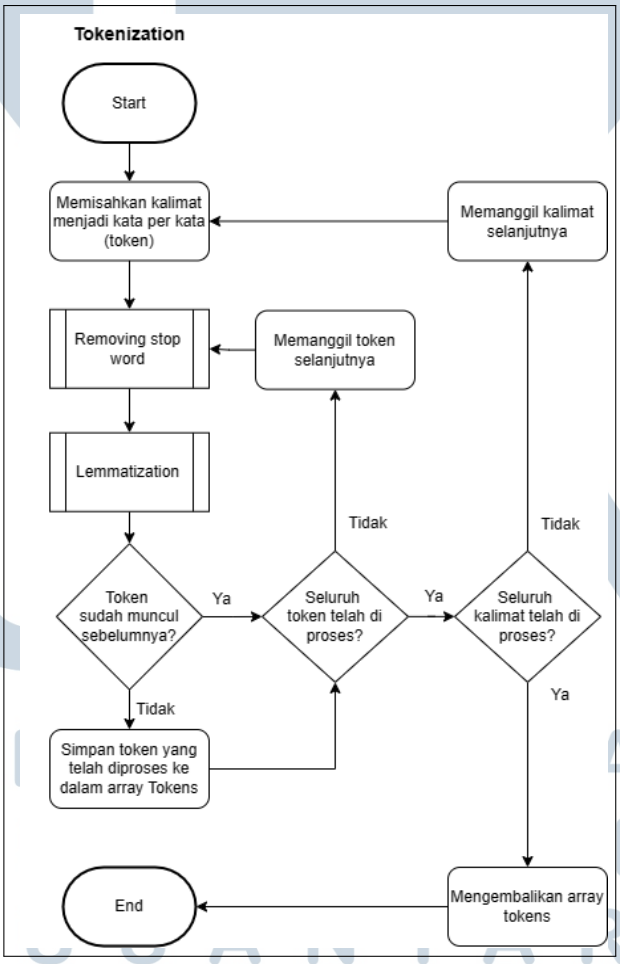



Gambar 3.6. Flowchart data cleaning

*Flowchart data cleaning* dapat dilihat pada Gambar 3.6. Pada tahapan *data cleaning* dilakukan proses untuk menghapus karakter spesial, simbol, tanda baca, *tag HTML* dan *URLs*. Setelah itu, dilakukan pengecekan jika terdapat data *null* atau tidak. Jika terdapat data *null* pada kolom komentar maka baris data tersebut akan dibuang. Selanjutnya, dilakukan *case folding* pada komentar untuk mengubah semua karakter yang ada menjadi *lowercase* kemudian dilakukan pengecekan bahasa dari komentar yang diketikkan oleh pengguna Youtube yang ada di *field comment*. Pengecekan ini dilakukan dengan menggunakan *library* langdetect yang ada pada Python dan dipilih komentar dengan bahasa mayoritas yaitu Bahasa Inggris (en). Terakhir dilakukan *abbreviation mapping* untuk mengubah kata-kata singkatan pada komentar sehingga memiliki makna dan lebih mudah untuk diklasifikasikan.

### 3.4.2 Tokenization

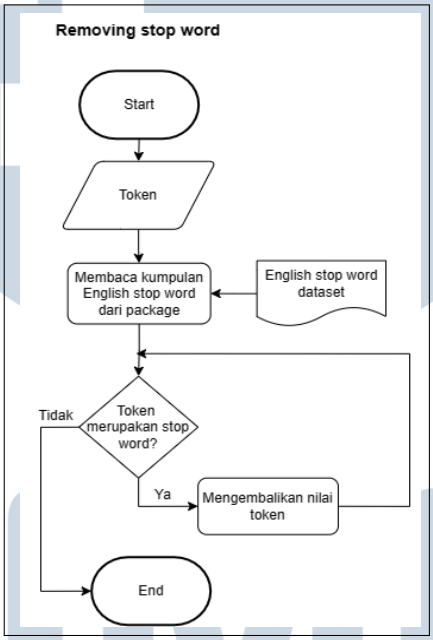
Dalam penelitian ini, *tokenization* dilakukan pada level kata. Proses *tokenization* dilakukan dengan melakukan *import* pada *function* word_tokenize yang terdapat pada *library* nltk.tokenize. *Function* ini diterapkan untuk setiap *field* yang berisikan komentar yang telah di-*clean* sebelumnya. Pada tahap ini, dilakukan juga pengecekan agar memastikan *stop word* tidak termasuk ke dalam token yang dibuat. Kemudian dilakukan juga pengecekan apakah token merupakan *root word* atau bukan menggunakan *lemmatization*. Hasil dari pengecekan tersebut kemudian disimpan ke dalam *array* 'tokens'. Proses ini dilakukan berulang kali untuk seluruh token pada semua kalimat komentar. Berikut adalah *flowchart* dari proses *tokenization* yang dapat dilihat pada Gambar 3.7.




Gambar 3.7. Flowchart tokenization

## A Removing Stop Word

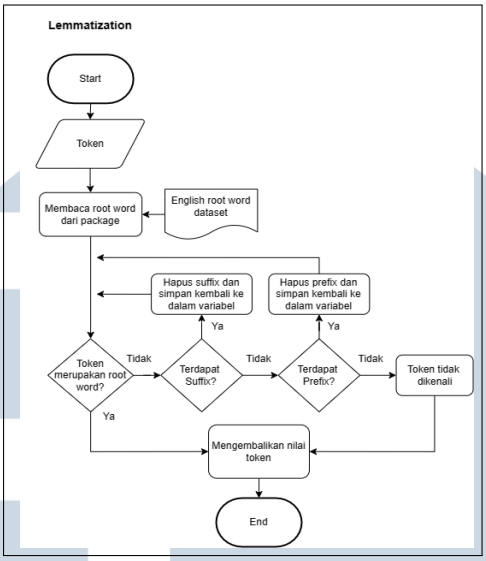
Pada tahapan ini dilakukan penghapusan *stop word* menggunakan *function* stopwords yang terdapat di *library* nltk.corpus. Saat ingin menggunakan *function* stopwords, diperlukan *input* berupa bahasa yang dipakai, dalam kasus ini adalah '*english*'. Variabel 'stop_words' digunakan untuk menampung seluruh *stop word* dalam Bahasa Inggris yang terdaftar dalam *library* ini. variabel ini nantinya akan digunakan dalam proses *tokenization* untuk mencegah token berisikan kata-kata dari kumpulan *stop word* yang ada di variabel 'stop_words'. *Flowchart* dari proses *removing stop word* dapat dilihat pada Gambar 3.8.




Gambar 3.8. Flowchart removing stop word

## B Lemmatization

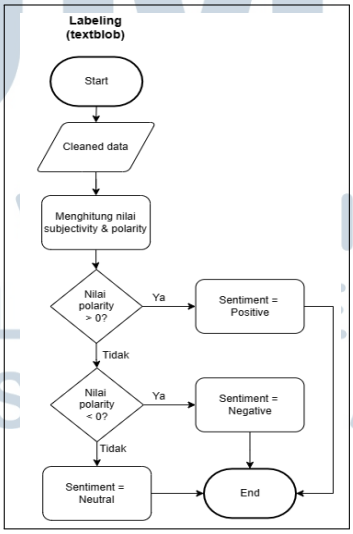
*Lemmatization* dilakukan menggunakan *function* WordNetLemmatizer() yang terdapat pada *library* nltk.stem. *Lemmatization* dilakukan terhadap setiap token yang ada pada variabel 'tokens'. Tahapan ini akan mengubah kata yang terkandung dalam variabel token menjadi kata dasarnya. *Flowchart* dari proses *lemmatization* dapat dilihat pada Gambar 3.9.




Gambar 3.9. Flowchart lemmatization

### 3.4.3 Labeling (Textblob)

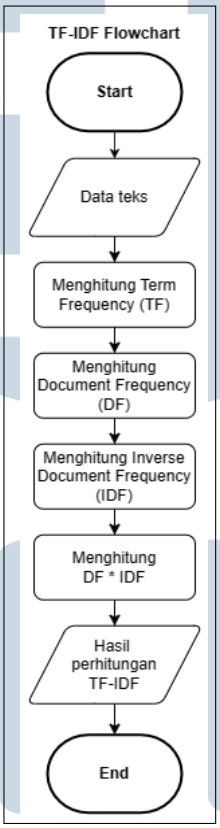
*Labeling* dilakukan dengan memberikan label positif, negatif, maupun netral pada data teks. Proses *labeling* dilakukan dengan memanfaatkan *library* textblob. Pada proses ini textblob akan menghitung nilai *polarity* dan *subjectivity* dari masing-masing *comment*. Jika nilai *polarity* pada suatu *comment* $< 0$ maka akan diberi label '*Negative*' sedangkan jika nilai nya $> 0$ maka akan diberi label '*Positive*'. Label '*Neutral*' akan diberikan pada *comment* dengan nilai *polarity* = 0. *Flowchart* dari proses *labeling* dapat dilihat pada Gambar 3.10.




Gambar 3.10. Flowchart labeling (textblob)

## 3.5 TF-IDF

Vektorisasi menggunakan TF-IDF dapat dilakukan dengan melakukan *import* terhadap TfidfVectorizer yang ada pada *library* sklearn.feature_extraction.text. *Max features* pada *function* TfidfVectorizer diatur menjadi 5000. Vektorisasi ini dilakukan pada *comment* yang terdapat di data *training* dan *testing*. Vektor ini yang nantinya akan digunakan sebagai data *training* dan *testing* baru dalam pembuatan model SVM. *Flowchart* dari proses *TF-IDF* dapat dilihat pada Gambar 3.11.

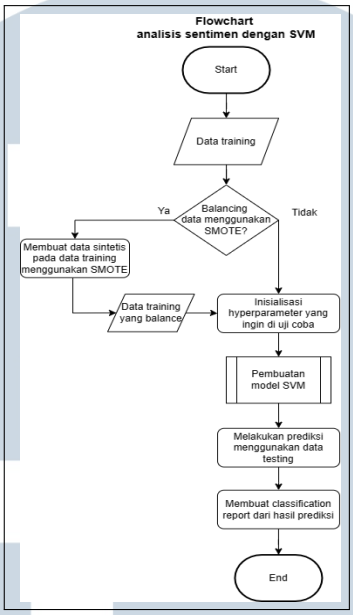



Gambar 3.11. Flowchart proses TF-IDF

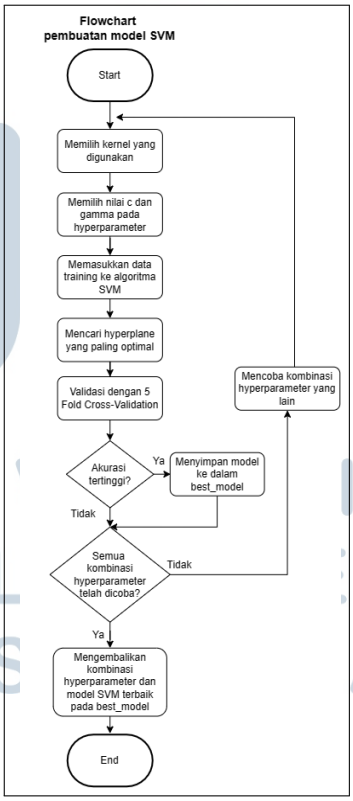
## 3.6 Data Splitting

*Data splitting* dilakukan menggunakan *function* train_test_split pada *library* sklearn. Rasio untuk data *training* dan *testing* yang digunakan di awal adalah 80:20 [18, 19, 50, 54, 59]. Hal ini berarti dari total 20.538 data akan dibagi menjadi 17.768 data yang digunakan dalam *training* dan 4108 data yang digunakan untuk *testing*.

## 3.7 Analisis Sentimen dengan SVM




Gambar 3.12. Flowchart analisis sentimen dengan SVM



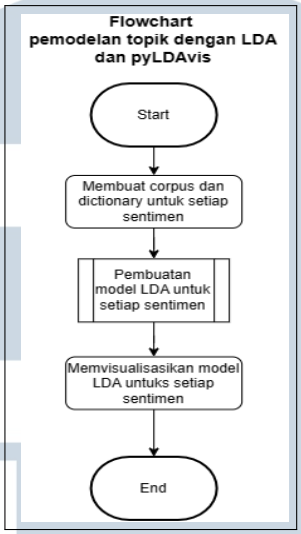

Gambar 3.13. Flowchart pembuatan model SVM

*Flowchart* analisis sentimen dengan SVM dapat dilihat pada Gambar 3.12. Pada tahapan ini, data *training* dan *testing* untuk *comment* (X_train) telah divektorisasi menggunakan TF-IDF. Setelah itu, variabel 'X_train' dan 'Y_train' yang berisikan label sentimen di *balancing* menggunakan algoritma SMOTE yang dapat di-*import* melalui *library* imblearn.over_sampling. Kemudian dilakukan pemilihan *hyperparameter* yang akan dipakai pada percobaan *hyperparameter tuning* pada model SVM yang dibuat. Model SVM dan *hyperparameter* yang digunakan sebagai *hyperparameter* yang digunakan adalah 'C': [0.1, 1, 10, 100], 'gamma': ['*scale*', '*auto*'], 'kernel': ['linear', 'poly, 'sigmoid', 'rbf']. *Function* SVC (Support Vector Classifier) digunakan untuk membuat model SVM. Model SVM dan *hyperparameter* ini kemudian dimuat ke dalam variabel 'grid_search' untuk dilakukan proses pencarian parameter terbaik dari model menggunakan *grid search* pada *function* GridSearchCV di *library* sklearn.model_selection. Proses ini kemudian ditampilkan dalam bentuk *line chart* agar dapat terlihat lebih jelas.

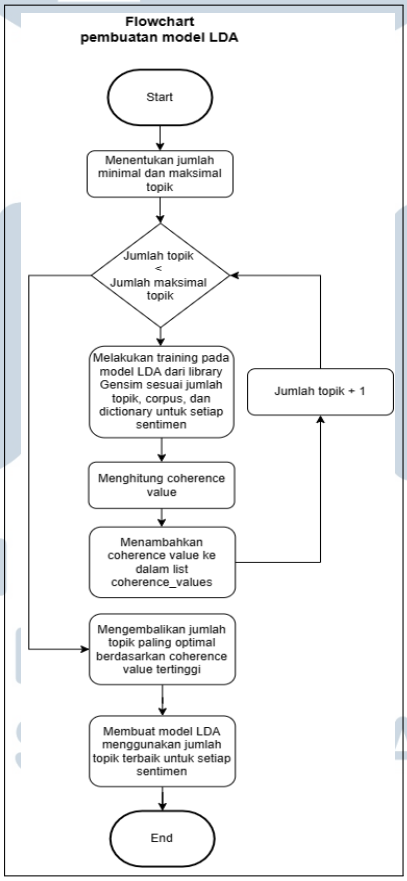
## 3.8 Pengujian Model SVM

Pengujian model dilakukan dengan memanfaatkan *library* sklearn_metrics. Hasil dari pengujian model dimuat ke dalam *classification report* yang memunculkan angka *precision*, *recall*, *f1-score*, *support* dari masing-masing label (*positive*, *negative*, *neutral*) dan nilai *accuracy* keseluruhan. Terdapat juga data mengenai *macro average* dan *weighted average* dari *precision*, *recall*, dan *f1-score* secara keseluruhan. Selanjutnya untuk memudahkan visualisasi hasil pengujian, dibuatlah *confusion matrix* yang di-*import* melalui sklearn_metrics. *Confusion matrix* yang dibuat berukuran 3x3 dengan label *negative*, *neutral*, dan *positive*. Kolom berisikan jumlah label yang diprediksi oleh model dan baris berisikan *actual label* dari proses klasifikasi sentimen menggunakan *library* textblob.

## 3.9 Pemodelan Topik dengan LDA dan PyLDAvis




Gambar 3.14. Flowchart pemodelan topik dengan LDA dan PyLDAvis




Gambar 3.15. Flowchart pembuatan model LDA

*Flowchart* pemodelan topik dengan LDA dan PyLDAvis dapat dilihat pada Gambar 3.14. Algoritma LDA digunakan untuk melakukan pemodelan topik dengan mengekstrak topik-topik pada setiap *class* sentimen ('Positive', 'Negative', dan 'Neutral') yang diprediksi menggunakan model SVM terbaik yang telah dibuat sebelumnya beserta dengan 30 keyword yang membentuk topik tersebut. Pemodelan topik diawali dengan pencarian nilai *coherence value* terlebih dahulu. *Coherence value* akan mencari jumlah topik terbaik yang digunakan pada model dengan mengukur kualitas dari topik tersebut. Kualitas topik dilihat dari seberapa baik kata-kata yang tergabung dalam suatu topik saling berhubungan dan memiliki keterkaitan yang kuat. Pada penelitian ini, jumlah minimal topik yang digunakan adalah 2 dan maksimal topik adalah 15. Model LDA dibuat dengan menggunakan *function* models.LdaModel yang ada di *library* gensim.

Pembuatan model LDA memerlukan *corpus*, *dictionary* dan juga nilai jumlah topik yang diinginkan. Pembuatan *dictionary* dilakukan menggunakan *function* corpora yang terdapat di *library* gensim. *Dictionary* dibuat untuk setiap label sentimen sehingga terdapat 3 *dictionary* yang disimpan ke dalam variabel 'positive_dictionary', 'negative_dictionary', dan 'neutral_dictionary'. Variabel 'num_topics' diambil berdasarkan angka di antara jumlah minimal dan maksimal topik. Perulangan dilakukan dalam pembuatan model dengan mengubah nilai 'num_topics' dari 2 sampai 15. Setelah perulangan selesai dipilihlah model LDA dengan jumlah topik dengan nilai *coherence value* tertinggi. Terakhir dilakukan proses visualisasi topik yang didapatkan dari model LDA menggunakan *function* PyLDAvis. *Function* ini akan menampilkan kluster topik berdasarkan jumlah 'num_topics' dan tiap kluster akan berisikan 30 topik yang di-*ranking* berdasarkan relevansi yang dimiliki oleh topik tersebut.

## 3.10 Dokumentasi

Proses dokumentasi dilakukan dengan melakukan penulisan laporan skripsi. Penulisan laporan dibimbing secara langsung oleh dosen pembimbing melalui konsultasi dan bimbingan secara tatap muka. Laporan skripsi yang dibuat mencakup segala hal terkait dengan penelitian yang dilakukan dimulai dari latar belakang, landasan teori, metodologi, hasil & pembahasan, kesimpulan & saran, serta lampiran dan daftar pustaka terkait penelitian yang telah dilakukan.