Mengenal Data Science

By: Zanuar Ekaputra Rus'an

Introduction

Pada zaman sekarang, ilmu tentang *Data Science* memang sudah banyak diperbincangkan dan juga banyak dicari oleh banyak perusahaan. *Data science* merupakan sebuah ilmu yang menggabungkan beberapa keilmuan dalam satu bidang. Beberapa ilmu yang ada pada *data science* adalah matematika, statistika, programming, dan juga analisis.

Data scientist

Pada bidang *Data Science*, *Data scientist* adalah seseorang yang mengerti tentang *software engineering* dan juga lebih mengerti statistika, probabilitas, dan ilmu-ilmu lainnya yang ada pada *Data Science*. Selain prinsip-prinsip pemrograman, seorang data scientist juga harus mengerti tentang data.

Practical

Dalam bidang yang akan kita pelajari, ada dua tahapan yang harus kita pahami dan juga kita pelajari. Dua tahapan tersebut adalah:

1. Analisis Deskriptif (*Descriptive Analytic*)

   Pada tahapan ini, kita akan berfokus pada pemahaman data. Seperti, apa yang terjadi pada data tersebut, dan kenapa suatu hal bisa terjadi pada data tersebut. Contohnya, Apakah voucher yang ada pada tanggal tertentu mempengaruhi penjualan? dan kenapa voucher tersebut bisa mempengaruhi penjualan?

2. Analisis Prediktif (*Predictive Analytic*)

   Pada tahapan ini, kita akan berfokus pada pembuatan prediksi dari hasil deskripsi analisis yang sudah dilakukan sebelumnya. *Machine learning* dan juga *deep learning* menjadi *tools* yang dipakai pada tahapan ini.

Dalam praktiknya, biasanya *Data Science* menggunakan bahasa R dan juga Python. Pada kali ini, kita akan menggunakan bahasa pemrograman Python dan juga menggunakan tools yang ada seperti *Google Collaboratory* atau bisa memakai *Jupyter Notebook*. Untuk lebih memahami proses tentang *Data Science*, mari kita coba *hands on lab* berikut ini!

Tahapan Analisis Deskriptif

1. Load & Read Dataset

Dataset yang akan kita pakai pada praktikal sekarang adalah dataset *Women's E-Commerce Clothing Reviews.* Kita bisa men*download* datasetnya disini.

```
import pandas as pd
df = pd.read_csv('/content/Womens Clothing E-Commerce Reviews.csv')
```

Kita menggunakan *library pandas* untuk membaca dataset yang ada dalam bentuk csv. Setelah kita load datasetnya, kita baca dan tampilkan 5 data paling atas terlebih dahulu dari datasetnya.

```
df.head(5)
```

	Unnamed: 0	Clothing ID	Age	Title	Review Text	Rating	Recommended IND	Positive Feedback Count	Division Name	Department Name	Class Name
0	0	767	33	NaN	Absolutely wonderful - silky and sexy and comf...	4	1	0	Initmates	Intimate	Intimates
1	1	1080	34	NaN	Love this dress! it's sooo pretty. i happene...	5	1	4	General	Dresses	Dresses
2	2	1077	60	Some major design flaws	I had such high hopes for this dress and reall...	3	0	0	General	Dresses	Dresses
3	3	1049	50	My favorite buy!	I love, love, love this jumpsuit. it's fun, fl...	5	1	0	General Petite	Bottoms	Pants
4	4	847	47	Flattering shirt	This shirt is very flattering to all due to th...	5	1	6	General	Tops	Blouses

Setelah melihat data tersebut, kita juga harus mengetahui jumlah seluruh data yang ada dengan menuliskan kode berikut.

```
df.shape
```

```
(23486, 11)
```

Dataset tersebut artinya memiliki 23.486 banyak data dengan 11 kolom. Lalu selanjutnya kita lihat apakah dataset tersebut memiliki data yang kosong di setiap barisnya? Jika ada mari kita urutkan dari yang terbesar.

```
null_data = df.isnull().sum()
null_data.sort_values(ascending=False)
```

```
title                      3810
review text                 845
class name                   14
department name              14
division name                14
positive feedback count       0
recommended ind               0
rating                        0
age                           0
clothing id                   0
unnamed: 0                    0
dtype: int64
```

Setelah dilihat, data tersebut memiliki data yang kosong pada kolom *Title, Review Text, Division Name, Department Name, dan Class Name.* Data yang kosong ini dapat kita hapus, atau dapat kita rubah jika nilai yang ada pada data tersebut bersifat numerik.

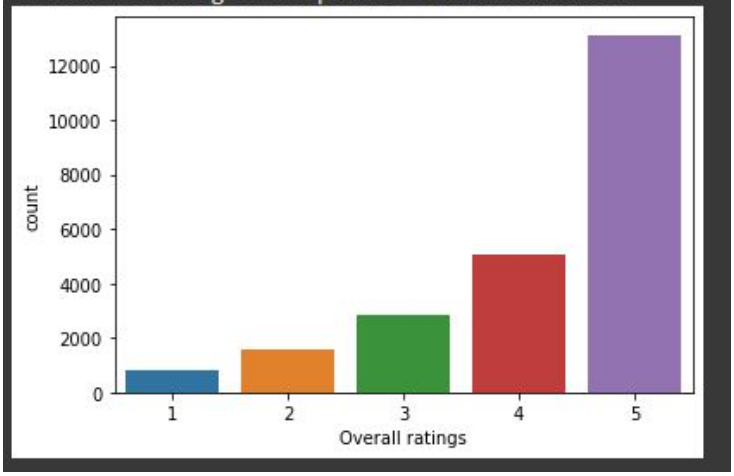
2. *Data Understanding* & Visualisasi

Tahapan ini digunakan untuk memudahkan kita dalam melakukan EDA (*Exploratory Data Analisis*). Visualisasi yang kita gunakan bisa menggunakan matplotlib dan juga seaborn. Agar lebih mudah, kita rubah juga nama kolom yang ada menjadi lowercase. Untuk mencari rata-rata pada setiap bagiannya, kita akan menggunakan pandas.

a. Visualisasi Kolom Rating dan Tampilkan rata-rata rating

```
import matplotlib.pyplot as plt
import seaborn as sns

sns.countplot(x=df.rating)
plt.xlabel('Overall ratings')

rate_mean = df['rating'].mean()
print("Rata-rata rating semua produk", rate_mean)
```

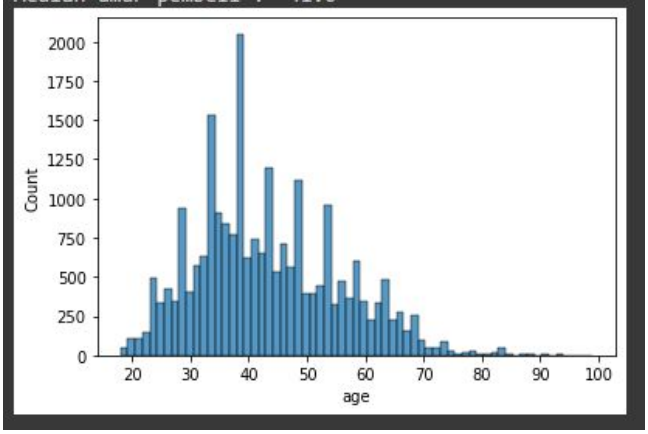
```
Rata-rata rating semua produk 4.196031678446734
```



b. Visualisasi jumlah data berdasarkan pembeli dan rata-rata, dan nilai tengah umur pembeli

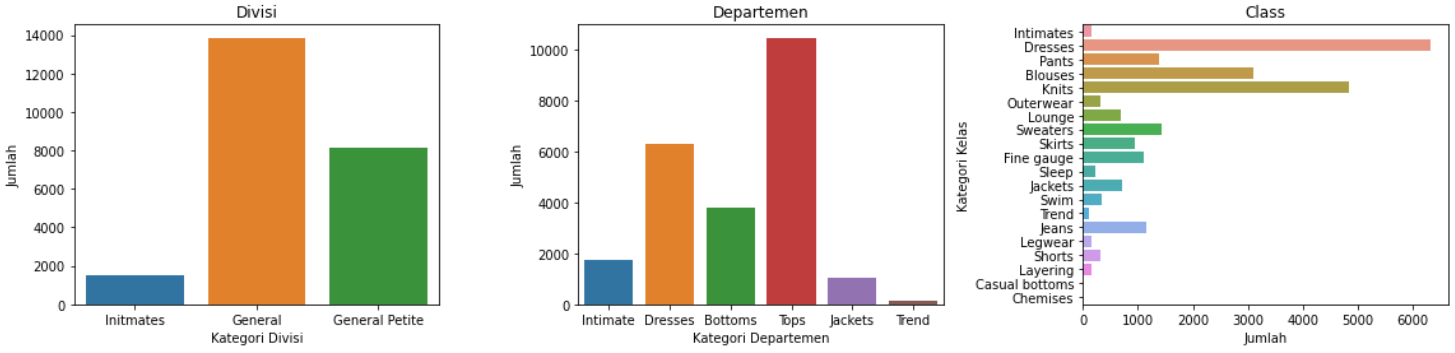
```
sns.histplot(x=df.age)
age_mean = df['age'].mean()
age_med = df['age'].median()
print("Rata-rata umur pembeli : ", age_mean)
print("Median umur pembeli : ", age_med)
```

```
Rata-rata umur pembeli :  43.198543813335604
Median umur pembeli :  41.0
```



c. Visualisasi jumlah data dari kolom division name, departement name, dan juga class name.

```
f, ax = plt.subplots(1, 3, figsize=(16,4), sharey=False)
sns.countplot(x=df.division_name, ax=ax[0])
ax[0].set_title("Divisi")
ax[0].set_xlabel("Kategori Divisi")
ax[0].set_ylabel("Jumlah")
sns.countplot(x=df.department_name, ax=ax[1])
ax[1].set_title("Departemen")
ax[1].set_xlabel("Kategori Departemen")
ax[1].set_ylabel("Jumlah")
sns.countplot(y=df.class_name, ax=ax[2])
ax[2].set_title("Class")
ax[2].set_xlabel("Jumlah")
ax[2].set_ylabel("Kategori Kelas")
plt.tight_layout()
plt.show()
```



Setelah kita mendapatkan hasil dari visualisasi, kita bisa jauh lebih memahami apa yang sedang terjadi pada data tersebut. Kita jadi mengetahui bahwa data tersebut masih mempunyai nilai kosong pada beberapa kolom. Lalu data tersebut merupakan data yang memiliki jumlah rating yang rata-rata bagus, yaitu bernilai 4. Rata-rata umur pembeli yaitu berusia 40 tahunan. Selain itu, penjualan yang paling laris terdapat pada kelas Dresses.

Tahapan *Predictive Analytic*

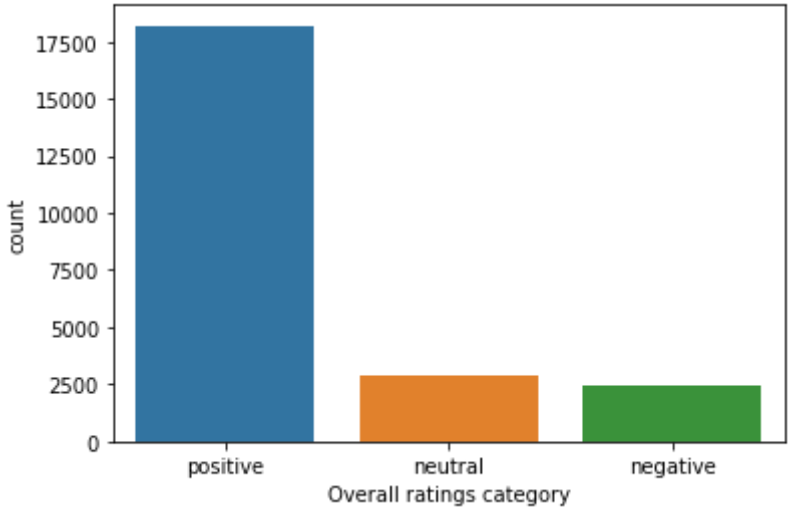
Di tahapan ini, kita akan membuat sebuah klasifikasi text yaitu *sentiment analysis* dengan menggunakan *Tensorflow*.

1. *Data Preprocessing*

Pada bagian ini, kita akan memproses data tersebut agar bisa dilakukan proses prediktif berdasarkan review yang muncul. Hasil akhir dari proses ini yaitu data akan menjadi lebih bersih, dan mudah untuk diproses saat *modeling.*

Pertama, kita akan mengganti rating yang mempunyai value 1 - 5 menjadi n*egative, neutral, dan juga positive.*

```
df['ratings_category'] = df.rating.replace({
    1:'negative',
    2:'negative',
    3:'neutral',
    4:'positive',
    5:'positive'
})
sns.countplot(df.ratings_category)
plt.xlabel('Overall ratings category')
```



Dapat dilihat bahwa data yang kita lihat memiliki data value yang tidak seimbang. Nilai *positive* lebih banyak daripada nilai *negative*. Untuk mengatasi ini, kita bisa mengurangi value dari dari *positive*. Tetapi, hal tersebut dapat berpengaruh terhadap modeling yang akan mengakibatkan kurangnya akurasi dan juga overfitting yang disebabkan karena kurangnya data untuk dilatih.

Solusi lainnya yaitu dengan menambah value dari neutral dan negative. Kita bisa menduplikat data *neutral* dan juga data *negative* agar jumlahnya sama dengan *positive.* Hal ini bisa disebut dengan *oversampling.*

Tetapi, sebelum melakukan *oversampling* kita akan menghapus data yang kosong terlebih dahulu.

```
df.isnull().sum()
```

```
review_text         845
ratings_category      0
dtype: int64
```

```
df = df.dropna(how='any', axis=0)
df.isnull().sum()
```

```
review_text         0
ratings_category    0
dtype: int64
```

```
df.shape
```

```
(22641, 2)
```

Pertama, kita akan split terlebih dahulu datasetnya untuk menjadi train dan test. Kenapa kita tidak melakukan *oversampling* terlebih dahulu? karena, *oversampling* dapat menipu metric model train. Akibatnya akurasi pada train dan test data akan terlihat bagus bagus, tetapi sebenarnya tidak.

```
from sklearn.model_selection import train_test_split
train, test = train_test_split(df, test_size=0.2, random_state=42,
shuffle=True)
```

Setelah kita lakukan split, kita akan memisahkan setiap ratings_category untuk menambah datanya. Ingat kita hanya akan melakukan *oversampling* pada data train saja.

```
df_positive = train[(train['ratings_category'] == 'positive')]
df_positive.shape
```

```
(13964, 2)
```

```
df_neutral = train[(train['ratings_category'] == 'neutral')]
df_neutral.shape
```

```
(2235, 2)
```

```
df_negative = train[(train['ratings_category'] == 'negative')]
df_negative.shape
```

```
(1913, 2)
```

Selanjutnya kita gabungkan data tersebut pada dataframe baru bernama train_data

```
train_data = df_positive.append(df_neutral, ignore_index=True)
train_data = train_data.append(df_negative, ignore_index=True)
train_data
```

	review_text	ratings_category
0	This top looks better on than on the hanger. i...	positive
1	There wasn't much question as to whether or no...	positive
2	These leggings are so warm and comfortable. th...	positive
3	This arrived in white at my store and it was s...	positive
4	I have to admit, seeing these tops online, i w...	positive
...	...	...
18107	It is a shame when a product such as this is l...	negative
18108	But it runs super small. i wear a medium for e...	negative
18109	The fabric and colors of this dress are beauti...	negative
18110	This one didn't work for me. ordered this and ...	negative
18111	I was excited to see these jeans since they ca...	negative

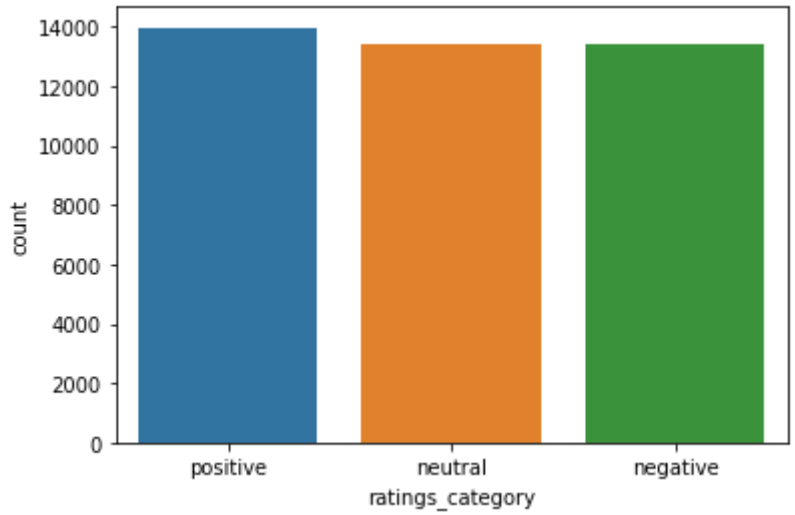
18112 rows × 2 columns

Lalu kita lakukan looping untuk pada dataframe tersebut sejumlah dengan banyaknya data positive. Dan beginilah hasilnya

```
for i in range(0, 5):
    train_data = train_data.append(df_neutral, ignore_index=True)

for i in range(0, 6):
    train_data = train_data.append(df_negative, ignore_index=True)

train_data
```

	review_text	ratings_category
0	This top looks better on than on the hanger. i...	positive
1	There wasn't much question as to whether or no...	positive
2	These leggings are so warm and comfortable. th...	positive
3	This arrived in white at my store and it was s...	positive
4	I have to admit, seeing these tops online, i w...	positive
...	...	...
40760	It is a shame when a product such as this is l...	negative
40761	But it runs super small. i wear a medium for e...	negative
40762	The fabric and colors of this dress are beauti...	negative
40763	This one didn't work for me. ordered this and ...	negative
40764	I was excited to see these jeans since they ca...	negative

40765 rows × 2 columns



Tahap selanjutnya yaitu merubah ratings category pada train_data dan test menjadi one hot encoding.

```
category = pd.get_dummies(train_data['ratings_category'])
train_data = pd.concat([train_data, category], axis = 1)
train_data = train_data.drop(['ratings_category'],  axis = 1)
train_data
```

	review_text	negative	neutral	positive
0	This top looks better on than on the hanger. i...	0	0	1
1	There wasn't much question as to whether or no...	0	0	1
2	These leggings are so warm and comfortable. th...	0	0	1
3	This arrived in white at my store and it was s...	0	0	1
4	I have to admit, seeing these tops online, i w...	0	0	1
...	...	...	...	...
40760	It is a shame when a product such as this is l...	1	0	0
40761	But it runs super small. i wear a medium for e...	1	0	0
40762	The fabric and colors of this dress are beauti...	1	0	0
40763	This one didn't work for me. ordered this and ...	1	0	0
40764	I was excited to see these jeans since they ca...	1	0	0

40765 rows × 4 columns

```
category = pd.get_dummies(test['ratings_category'])
test = pd.concat([test, category], axis = 1)
test = test.drop(['ratings_category'],  axis = 1)
test
```

	review_text	negative	neutral	positive
13365	This sweater is so beautiful on. it is thick m...	0	0	1
19834	This piece is almost what i want... i tried on...	0	0	1
18722	Really like this blouse but am returning for a...	0	0	1
10635	These are the perfect light weight relaxing su...	0	0	1
7348	These look nothing like the picture! they are ...	1	0	0
...	...	...	...	...
1092	Great that it's hand washable because i hate t...	0	0	1
8448	I am usually a regular xs with retailer tops. ...	0	0	1
5875	Boxy, short & wide! luv this brand but super d...	1	0	0
23285	This top is very gorgeous and chic. it is very...	0	0	1
13402	I tried this on in the store in the xs size. t...	0	1	0

4529 rows x 4 columns

Selanjutnya yaitu melakukan inisialisasi pemrosesan data untuk dilatih.

```
x_train = train_data['review_text'].values.astype(str)
y_train = train_data[['negative', 'neutral', 'positive']].values

x_test = test['review_text'].values.astype(str)
y_test = test[['negative', 'neutral', 'positive']].values
```

Setelah melakukan inisialisasi data, kita lakukan tokenisasi dan sequence pada data yang ada. Tokenisasi digunakan agar data dapat dikenali dengan mudah.

```
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

tokenizer = Tokenizer(num_words = 10000, oov_token = 'x')
tokenizer.fit_on_texts(x_train)
tokenizer.fit_on_texts(x_test)

seq_train = tokenizer.texts_to_sequences(x_train)
seq_test = tokenizer.texts_to_sequences(x_test)

padded_train = pad_sequences(seq_train)
padded_test = pad_sequences(seq_test)
```

2. Modeling

Pada tahapan ini, kita akan membuat model *Deep Learning* dengan menggunakan LSTM (Long Short Term Memories).

```
import tensorflow as tf
model = tf.keras.Sequential([
    tf.keras.layers.Embedding(input_dim = 10000, output_dim = 64),
    tf.keras.layers.LSTM(64),
    tf.keras.layers.Dense(64, activation = 'relu'),
    tf.keras.layers.Dropout(0.8),
    tf.keras.layers.Dense(32, activation = 'relu'),
    tf.keras.layers.Dropout(0.8),
    tf.keras.layers.Dense(16, activation = 'relu'),
    tf.keras.layers.Dropout(0.8),
    tf.keras.layers.Dense(3, activation = 'softmax')
])

model.compile(
    loss = 'categorical_crossentropy',
    optimizer = 'adam',
    metrics = ['accuracy']
)
```

Kita menggunakan dropout agar tidak terjadi *overfitting* pada model. Selanjutnya kita compile model tersebut dengan menggunakan *categorical_crossentropy* untuk klasifikasi lebih dari 2 kelas. karena kita memiliki 3 nilai yaitu negative, neutral, dan positive.

Lalu kita akan mulai melakukan proses pelatihan data dengan model yang sudah dibuat. Kita akan melatih data tersebut sebanyak 10 kali.

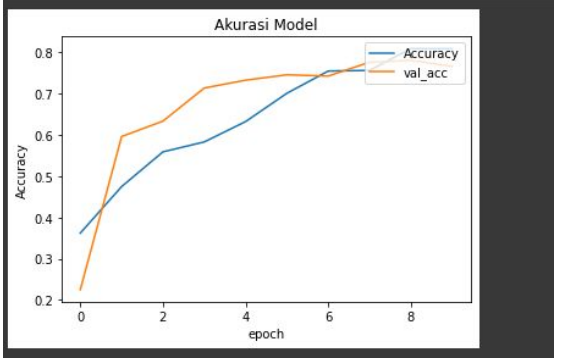
```
num_epochs = 10
history = model.fit(
    padded_train,
    y_train,
    epochs = num_epochs,
    validation_data = (padded_test, y_test),
    verbose= 2
)
```

```
Epoch 1/10
1344/1344 - 41s - loss: 1.0973 - accuracy: 0.3621 - val_loss: 1.0933 - val_accuracy: 0.2250
Epoch 2/10
1344/1344 - 38s - loss: 0.9973 - accuracy: 0.4750 - val_loss: 0.7912 - val_accuracy: 0.5964
Epoch 3/10
1344/1344 - 38s - loss: 0.8379 - accuracy: 0.5592 - val_loss: 0.9531 - val_accuracy: 0.6337
Epoch 4/10
1344/1344 - 39s - loss: 0.7667 - accuracy: 0.5833 - val_loss: 1.5600 - val_accuracy: 0.7141
Epoch 5/10
1344/1344 - 39s - loss: 0.6891 - accuracy: 0.6326 - val_loss: 2.4462 - val_accuracy: 0.7331
Epoch 6/10
1344/1344 - 39s - loss: 0.6265 - accuracy: 0.7013 - val_loss: 5.2019 - val_accuracy: 0.7461
Epoch 7/10
1344/1344 - 39s - loss: 0.5551 - accuracy: 0.7552 - val_loss: 7.7262 - val_accuracy: 0.7430
Epoch 8/10
1344/1344 - 39s - loss: 0.5687 - accuracy: 0.7567 - val_loss: 17.2060 - val_accuracy: 0.7766
Epoch 9/10
1344/1344 - 39s - loss: 0.4785 - accuracy: 0.8096 - val_loss: 23.4832 - val_accuracy: 0.7810
Epoch 10/10
1344/1344 - 38s - loss: 0.4833 - accuracy: 0.8092 - val_loss: 21.7764 - val_accuracy: 0.7666
```

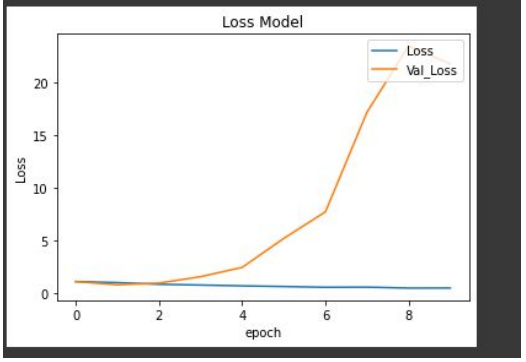
3. Hasil

Hasil dari pelatihan tersebut dapat kita tampilkan dengan menggunakan diagram metric.

```
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Akurasi Model')
plt.ylabel('Accuracy')
plt.xlabel('epoch')
plt.legend(['Accuracy', 'val_acc'], loc = 'upper right')
plt.show()
```




```
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Loss Model')
plt.ylabel('Loss')
plt.xlabel('epoch')
plt.legend(['Loss', 'Val_Loss'], loc = 'upper right')
plt.show()
```



Dari hasil yang dilihat, ternyata melakukan *oversampling* pada data tersebut masih menjadikan model yang sudah dilatih menjadi overfitting. Maka dari itu kita akan mencoba melatih data yang tidak melakukan *oversampling*.

4. Melatih data tanpa *Oversampling*

Baiklah kita akan kembali mempersiapkan datanya. Jadi, setelah kita load datanya seperti pada saat awal, kita akan membuat data frame baru lagi yang yang berisi data “review_text” dan “ratings_category”. Lalu, pada dataframe baru ini, kita akan menghapus kembali data yang kosong seperti yang sudah dilakukan sebelumnya.

```
df = df[['review_text', 'ratings_category']]
df.isnull().sum()
```

```
review_text         845
ratings_category      0
dtype: int64
```

```
df = df.dropna(how='any', axis=0)
df.isnull().sum()
```

```
review_text         0
ratings_category    0
dtype: int64
```

```
len(df)
```

```
22641
```

Selanjutnya kita lakukan juga One Hot Encoding pada “ratings_category”.

```
category = pd.get_dummies(df['ratings_category'])
df_baru = pd.concat([df, category], axis = 1)
df_baru = df_baru.drop(['ratings_category'],  axis = 1)
df_baru
```

	review text	negative	neutral	postive
0	Absolutely wonderful - silky and sexy and comf...	0	0	1
1	Love this dress! it's sooo pretty. i happene...	0	0	1
2	I had such high hopes for this dress and reall...	0	1	0
3	I love, love, love this jumpsuit. it's fun, fl...	0	0	1
4	This shirt is very flattering to all due to th...	0	0	1
...	...	...	...	...
23481	I was very happy to snag this dress at such a ...	0	0	1
23482	It reminds me of maternity clothes. soft, stre...	0	1	0
23483	This fit well, but the top was very see throug...	0	1	0
23484	I bought this dress for a wedding i have this ...	0	1	0
23485	This dress in a lovely platinum is feminine an...	0	0	1

22641 rows × 4 columns

Langkah selanjutnya adalah kita akan melakukan inisialisasi data parameter dan juga targetnya.

```
review = df_baru['review'].values
category = df_baru[['negative','neutral','postive']].values
```

Lalu kita langsung membagikan datanya menjadi data latih dan juga data testing dengan perbandingan 80:20. Setelah itu kita langsung melakukan tokenisasi seperti pada sebelumnya.

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split (review, category, test_size = 0.2)
```

```
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

tokenizer = Tokenizer(num_words = 5000, oov_token = 'x')
tokenizer.fit_on_texts(x_train)
tokenizer.fit_on_texts(x_test)

seq_train = tokenizer.texts_to_sequences(x_train)
seq_test = tokenizer.texts_to_sequences(x_test)

padded_train = pad_sequences(seq_train)
padded_test = pad_sequences(seq_test)
```

Lalu kita akan latih datanya dengan model seperti ini. Kita hanya merubah beberapa aspek seperti output dim dan juga LSTM nya.

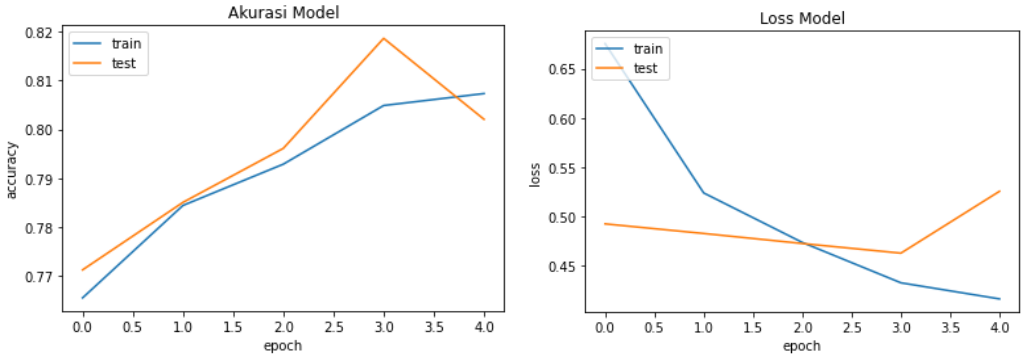
```
import tensorflow as tf
model = tf.keras.Sequential([
    tf.keras.layers.Embedding(input_dim = 5000, output_dim = 16),
    tf.keras.layers.LSTM(64),
    tf.keras.layers.Dense(64, activation = 'relu'),
    tf.keras.layers.Dropout(0.8),
    tf.keras.layers.Dense(32, activation = 'relu'),
    tf.keras.layers.Dropout(0.8),
    tf.keras.layers.Dense(3, activation = 'softmax')
])

model.compile(
    loss = 'categorical_crossentropy',
    optimizer = 'adam',
    metrics = ['accuracy']
)
```

Setelah itu kita akan mulai latih datanya sebanyak 5 epochs.

```
num_epochs = 5
history = model.fit(
    padded_train,
    y_train,
    epochs = num_epochs,
    validation_data = (padded_test, y_test),
    verbose= 2
)
```

```
Epoch 1/5
588/588 - 36s - loss: 0.6753 - accuracy: 0.7654 - val_loss: 0.4923 - val_accuracy: 0.7712
Epoch 2/5
588/588 - 39s - loss: 0.5237 - accuracy: 0.7844 - val_loss: 0.4826 - val_accuracy: 0.7850
Epoch 3/5
588/588 - 40s - loss: 0.4735 - accuracy: 0.7928 - val_loss: 0.4724 - val_accuracy: 0.7961
Epoch 4/5
588/588 - 40s - loss: 0.4325 - accuracy: 0.8049 - val_loss: 0.4626 - val_accuracy: 0.8186
Epoch 5/5
588/588 - 41s - loss: 0.4161 - accuracy: 0.8073 - val_loss: 0.5254 - val_accuracy: 0.8020
```

5. Hasil Tanpa *Oversampling*



Dapat dilihat ternyata, tanpa melakukan *oversampling* pun model akan tetap bagus hasilnya.

Kesimpulan

Dari semua proses yang dilakukan, kita sudah melakukan *descriptive analytic* dan juga *predictive analytic*. Model yang dibuat juga sudah mempunyai akurasi yang cukup bagus.

Terimakasih