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.

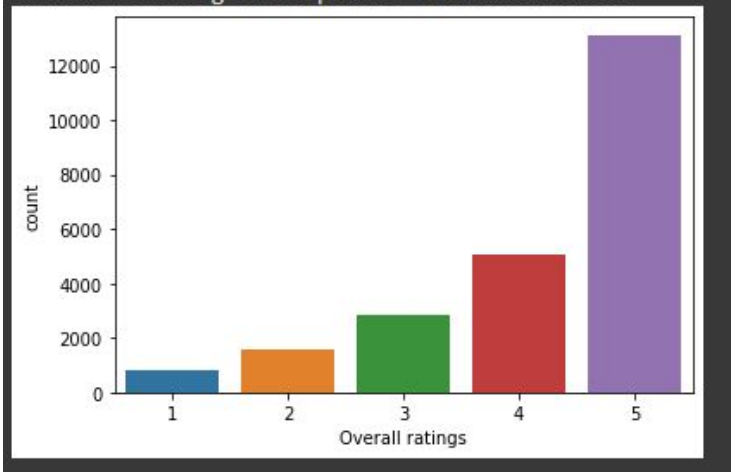
a. Visualisasi Kolom Rating dan Tampilkan rata-rata rating

```
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)
```

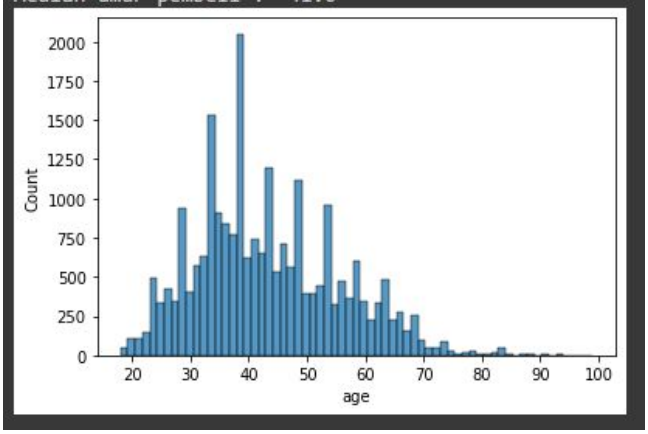
```
Rata-rata rating semua produk 4.196031678446734
```



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

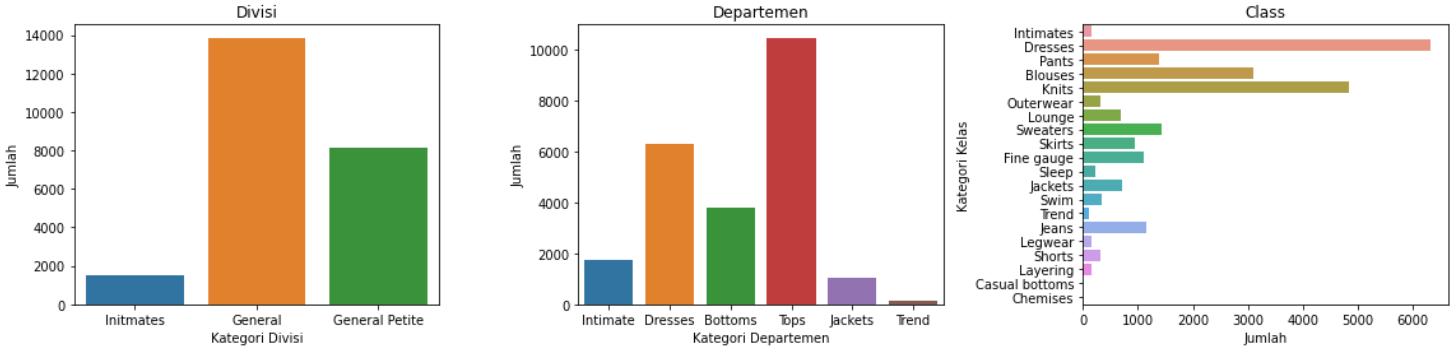
```
sns.histplot(x=df.age)
age_mean = df['age'].mean()
age_med = df['age'].median()
print("Rata-rata umur pembeli : ", age_mean)
print("Median umur pembeli : ", age_med)
```

```
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*

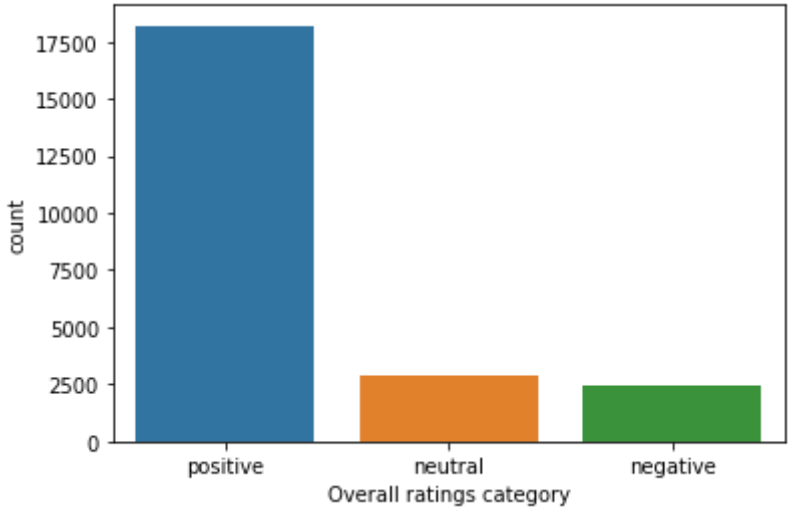
Di tahapan ini, kita akan membuat sebuah klasifikasi text yaitu *sentiment analysis* dengan menggunakan *Tensorflow*.

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.*

Pertama kita harus memisahkan data yang akan kita latih.

```
df = df[['review_text', 'ratings_category']]
df
```

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

23486 rows × 2 columns

Jika kita lihat, jumlah data yang ada masih sama dengan jumlah data awal. Jika kita ingat, data awal memiliki data yang kosong pada kolom *review_text.* Maka dari itu tahap selanjutnya adalah menghilangkan data kosong tersebut.

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

```
review_text         0
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)
```

Selanjutnya kita akan memisahkan setiap *review_text* dengan ratingnya.

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

```
(17448, 2)
```

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

```
(2823, 2)
```

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

```
(2370, 2)
```

Kita akan mengambil 10000 data positive, lalu akan kita gabungkan semuanya dalam satu dataframe.

```
df_positive = df_positive[:10000]
df_positive.shape
```

```
(10000, 2)
```

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

	review_text	ratings_category
0	Absolutely wonderful - silky and sexy and comf...	positive
1	Love this dress! it's sooo pretty. i happene...	positive
2	I love, love, love this jumpsuit. it's fun, fl...	positive
3	This shirt is very flattering to all due to th...	positive
4	I aded this in my basket at hte last mintue to...	positive
...	...	...
15188	I was very excited to find a fun and lightweig...	negative
15189	Before i ordered this i noted the other review...	negative
15190	What drew me to this shirt was the beautiful s...	negative
15191	This dress is so cute in the photo and fit tru...	negative
15192	I was surprised at the positive reviews for th...	negative

15193 rows × 2 columns

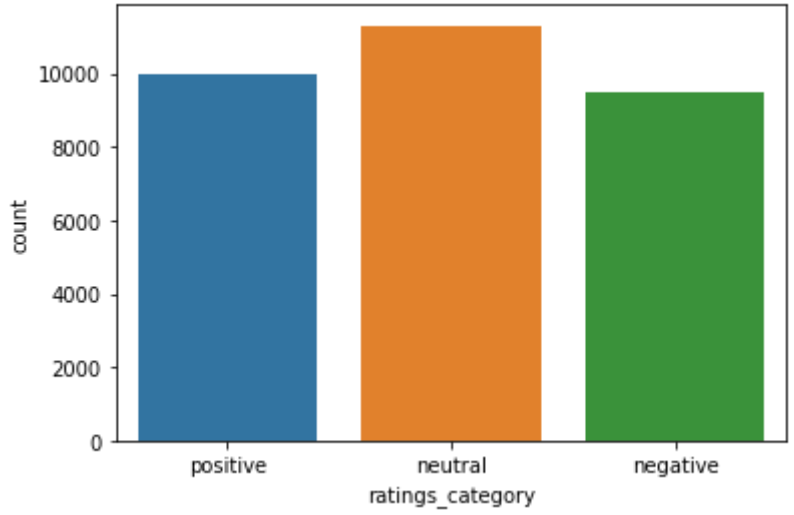
Untuk data *neutral dan negative* akan kita tambahkan dengan menduplikat datanya dengan menggunakan pengulangan.

```
for i in range(0, 3):
    df_baru = df_baru.append(df_neutral, ignore_index=True)
    df_baru = df_baru.append(df_negative, ignore_index=True)

df_baru
```

	review_text	ratings_category
0	Absolutely wonderful - silky and sexy and comf...	positive
1	Love this dress! it's sooo pretty. i happene...	positive
2	I love, love, love this jumpsuit. it's fun, fl...	positive
3	This shirt is very flattering to all due to th...	positive
4	I aded this in my basket at hte last mintue to...	positive
...	...	...
30767	I was very excited to find a fun and lightweig...	negative
30768	Before i ordered this i noted the other review...	negative
30769	What drew me to this shirt was the beautiful s...	negative
30770	This dress is so cute in the photo and fit tru...	negative
30771	I was surprised at the positive reviews for th...	negative

30772 rows × 2 columns



Berikut adalah data yang sudah kita duplikat. Data tersebut sekarang sudah seimbang dan siap untuk masuk ke proses training.

Tahap selanjutnya adalah menjadikan *ratings_category* menjadi *One Hot Encoding*. kita bisa melakukannya dengan memakai pandas dan beginilah hasilnya.

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

	review_text	negative	neutral	positive
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 love, love, love this jumpsuit. it's fun, fl...	0	0	1
3	This shirt is very flattering to all due to th...	0	0	1
4	I aded this in my basket at hte last mintue to...	0	0	1
...	...	...	...	...
30767	I was very excited to find a fun and lightweig...	1	0	0
30768	Before i ordered this i noted the other review...	1	0	0
30769	What drew me to this shirt was the beautiful s...	1	0	0
30770	This dress is so cute in the photo and fit tru...	1	0	0
30771	I was surprised at the positive reviews for th...	1	0	0

30772 rows x 4 columns

Selanjutnya, kita akan menentukan variabel yang akan menjadi parameter dan juga target dari proses train dan test pada data yang dipakai. untuk data parameternya kita akan menggunakan nilai dari kolom *'review_text'* dan targetnya kita akan menjadikan *ratings_category* barusan sebagai targetnya.

Lalu kita akan membagi datanya sebanyak 80% untuk menjadi train data, dan 20% menjadi test data dengan menggunakan sklearn.

```
review = df_baru['review_text'].values.astype(str)
category = df_baru[['negative','neutral','positive']].values

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)
```

Setelah melakukan pembagian 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.

```
import tensorflow as tf
model = tf.keras.Sequential([
    tf.keras.layers.Embedding(input_dim = 10000, output_dim = 16),
    tf.keras.layers.LSTM(128),
    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']
)

model.summary()
```

```
Model: "sequential_44"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_44 (Embedding)     (None, None, 16)          160000
_________________________________________________________________
lstm_43 (LSTM)               (None, 128)               74240
_________________________________________________________________
dense_84 (Dense)             (None, 64)                8256
_________________________________________________________________
dropout_31 (Dropout)         (None, 64)                0
_________________________________________________________________
dense_85 (Dense)             (None, 32)                2080
_________________________________________________________________
dropout_32 (Dropout)         (None, 32)                0
_________________________________________________________________
dense_86 (Dense)             (None, 3)                 99
=================================================================
Total params: 244,675
Trainable params: 244,675
Non-trainable params: 0
_________________________________________________________________
```

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.

Selanjutnya, kita train data tersebut dengan model yang sudah dibuat.

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

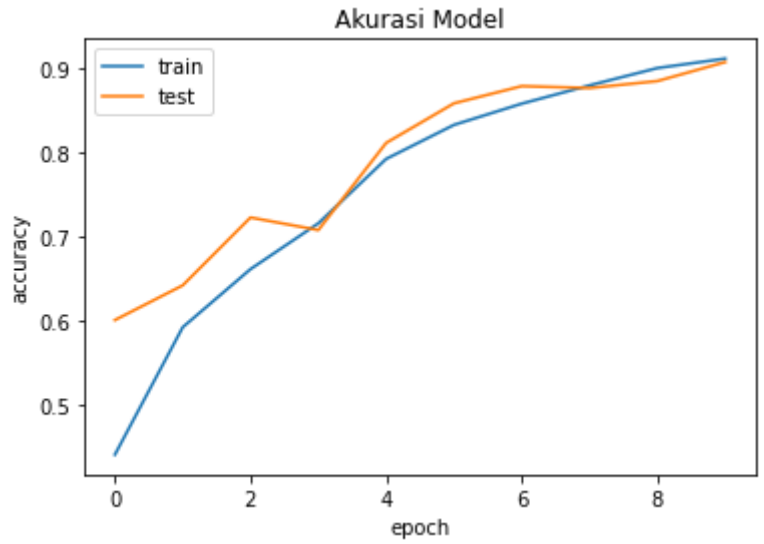
```
Epoch 1/10
770/770 - 27s - loss: 1.0242 - accuracy: 0.4402 - val_loss: 0.7368 - val_accuracy: 0.6003
Epoch 2/10
770/770 - 24s - loss: 0.7938 - accuracy: 0.5917 - val_loss: 0.6850 - val_accuracy: 0.6416
Epoch 3/10
770/770 - 25s - loss: 0.6870 - accuracy: 0.6609 - val_loss: 0.6325 - val_accuracy: 0.7223
Epoch 4/10
770/770 - 25s - loss: 0.6179 - accuracy: 0.7154 - val_loss: 0.6425 - val_accuracy: 0.7074
Epoch 5/10
770/770 - 25s - loss: 0.5154 - accuracy: 0.7920 - val_loss: 0.5274 - val_accuracy: 0.8110
Epoch 6/10
770/770 - 25s - loss: 0.4451 - accuracy: 0.8325 - val_loss: 0.5416 - val_accuracy: 0.8582
Epoch 7/10
770/770 - 24s - loss: 0.4015 - accuracy: 0.8578 - val_loss: 0.4364 - val_accuracy: 0.8786
Epoch 8/10
770/770 - 24s - loss: 0.3558 - accuracy: 0.8795 - val_loss: 0.5054 - val_accuracy: 0.8764
Epoch 9/10
770/770 - 24s - loss: 0.3117 - accuracy: 0.9001 - val_loss: 0.5515 - val_accuracy: 0.8846
Epoch 10/10
770/770 - 25s - loss: 0.2797 - accuracy: 0.9114 - val_loss: 0.4887 - val_accuracy: 0.9072
```

3. Evaluasi

Model yang sudah dibuat akan kita tampilkan hasil dari train dan test yang sudah kita lakukan.

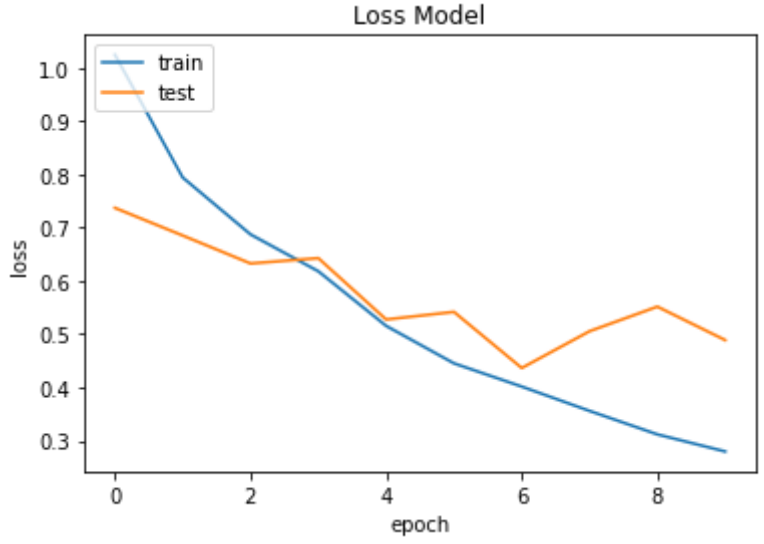
untuk menampilkan plot akurasi

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



Untuk menampilkan plot Loss

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



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, dan data yang kita proses juga sudah cukup bersih dan juga seimbang.

Terimakasih