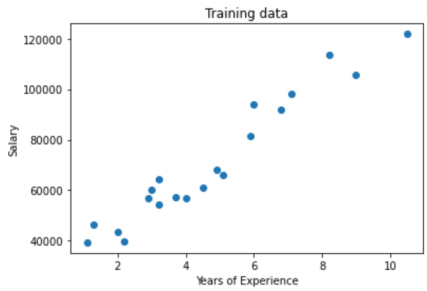
Question 1:

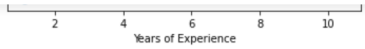
Read_csv() is used to read the dataset and train_test_split() is used to split the data into training and testing sets

```
In [77]: import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt
         from sklearn.model_selection import train_test_split
         from sklearn.linear_model import LinearRegression
         from sklearn import metrics
         from sklearn import preprocessing
         from sklearn.metrics import mean_squared_error
         from sklearn.cluster import KMeans
         from sklearn.impute import SimpleImputer
         from sklearn.decomposition import PCA
         from sklearn.preprocessing import LabelEncoder, StandardScaler
         import seaborn as sns

In [78]: sal_df=pd.read_csv("./datasets/Salary_Data.csv")

In [79]: X = sal_df.iloc[:, :-1].values
         Y = sal_df.iloc[:, 1].values
         X_Train, X_Test, Y_Train, Y_Test = train_test_split(X,Y, test_size=1/3,random_state = 0)

In [80]: regressor = LinearRegression()
         regressor.fit(X_Train, Y_Train)
         Y_Pred = regressor.predict(X_Test)

In [81]: print("Mean square error is",mean_squared_error(Y_Test,Y_Pred))
```

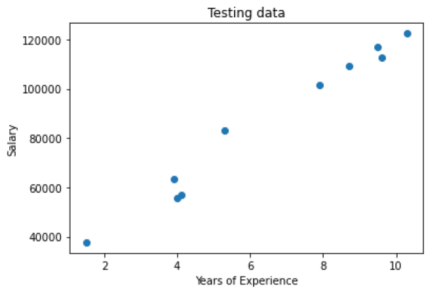
Mean_squared_error is used to find the mean square error

```
In [81]: print("Mean square error is",mean_squared_error(Y_Test,Y_Pred))

         Mean square error is 21026037.329511296

In [82]: plt.title('Training data')
         plt.xlabel('Years of Experience')
         plt.ylabel('Salary')
         plt.scatter(X_Train, Y_Train)
         plt.show()
```






```
In [83]: plt.title('Testing data')
         plt.xlabel('Years of Experience')
         plt.ylabel('Salary')
         plt.scatter(X_Test, Y_Test)
         plt.show()
```




```
In [84]: kmean_df2=pd.read_csv("datasets/K-Mean_Dataset.csv")
```

Question 2:

All the missing values are replaced with mean by using simple imputer and the model is fitted with the resulting data

In [84]:
```
kmean_df2=pd.read_csv("datasets/K-Mean_Dataset.csv")
kmean_df2.head()
```

Out[84]:

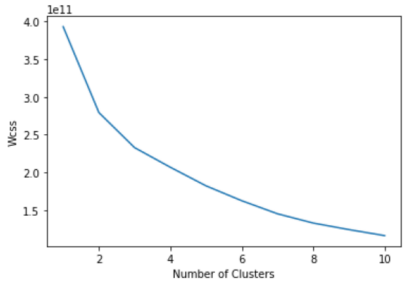
	CUST_ID	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUEN
0	C10001	40.900749	0.818182	95.40	0.00	95.4	0.000000	0.166
1	C10002	3202.467416	0.909091	0.00	0.00	0.0	6442.945483	0.000
2	C10003	2495.148862	1.000000	773.17	773.17	0.0	0.000000	1.000
3	C10004	1666.670542	0.636364	1499.00	1499.00	0.0	205.788017	0.083
4	C10005	817.714335	1.000000	16.00	16.00	0.0	0.000000	0.083

In [85]:
```
X = kmean_df2.iloc[:,1:].values
imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
imputer = imputer.fit(X)
X = imputer.transform(X)
```

In [86]:
```
wcss = []
for i in range(1,11):
    kmeans = KMeans(n_clusters=i,init='k-means++',max_iter=300,n_init=10,random_state=0)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)
plt.plot(range(1,11),wcss)
plt.xlabel('Number of Clusters')
```

The elbow graph bends at 2 so we consider this as the number of clusters

```
plt.ylabel('Wcss')
plt.show()
```



In [87]:
```
# The above graph bends at 2 so number of clusters is 2
from sklearn.cluster import KMeans
nclusters = 2
km = KMeans(n_clusters=nclusters)
km.fit(X)
```

Out[87]: KMeans(n_clusters=2)

The Silhouette score is reduced after scaling

```
from sklearn import metrics
score = metrics.silhouette_score(X, y_cluster_kmeans)
print('Silhouette score is,',score)
```

```
Silhouette score is, 0.511639269641848
```

In [89]:
```
scaler = preprocessing.StandardScaler()
scaler.fit(X)
X_scaled_array = scaler.transform(X)
X_scaled = pd.DataFrame(X_scaled_array)
```

In [90]:
```
from sklearn.cluster import KMeans
nclusters = 2
km = KMeans(n_clusters=nclusters)
km.fit(X_scaled)
```

Out[90]:
```
KMeans(n_clusters=2)
```

In [91]:
```
y_scaled_cluster_kmeans = km.predict(X_scaled)
from sklearn import metrics
score = metrics.silhouette_score(X_scaled, y_scaled_cluster_kmeans)
print('Silhouette score after applying scaling is,',score)
```

```
Silhouette score after applying scaling is, 0.20961923516469821
```

In [92]:
```
#Silhouette score is reduced after scaling
```

Video Link: https://drive.google.com/file/d/15qIP32aTPrZWDL5U-lrqqlePfCM12zXc/view?usp=sharing

Github link: https://github.com/sainikhil3280/ML-Assignment-4