Assignment 5 - Data Analytics II

Atharva Taras (TE A - 73)

1. Implement logistic regression using Python/R to perform classification on Social_Network_Ads.csv dataset.

Dataset - Social_Network_Ads.csv (https://www.kaggle.com/datasets/akram24/social-network-ads?resource=download)

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
```

In [2]:
```
data = pd.read_csv('Social_Network_Ads.csv')
data.head()
```

Out[2]:

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0

In [3]:
```
data = data.drop('User ID', axis=1)
print(data.isnull().sum(), '\n')
data.info()
```

```
Gender             0
Age                0
EstimatedSalary    0
Purchased          0
dtype: int64

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 4 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Gender           400 non-null    object
 1   Age              400 non-null    int64
 2   EstimatedSalary  400 non-null    int64
 3   Purchased        400 non-null    int64
dtypes: int64(3), object(1)
memory usage: 12.6+ KB
```

In [4]:
```
1 data.describe()
```

Out[4]:

	Age	EstimatedSalary	Purchased
count	400.000000	400.000000	400.000000
mean	37.655000	69742.500000	0.357500
std	10.482877	34096.960282	0.479864
min	18.000000	15000.000000	0.000000
25%	29.750000	43000.000000	0.000000
50%	37.000000	70000.000000	0.000000
75%	46.000000	88000.000000	1.000000
max	60.000000	150000.000000	1.000000

Normalizing salary for better accuracy

In [5]:
```
1 tmp = []
2 min_sal = min(data['EstimatedSalary'])
3 diff = max(data['EstimatedSalary']) - min_sal
4
5 for salary in data['EstimatedSalary']:
6     tmp.append((salary-min_sal)/diff)
7
8 data['Normalized_Salary'] = tmp
```

In [6]:
```
1 tmp = []
2
3 for value in data['Gender']:
4     if value == 'Male':
5         tmp.append(1)
6     else:
7         tmp.append(0)
8
9 data['Gender_Catg'] = tmp
```

In [7]:
```
1 data.head(2)
```

Out[7]:

	Gender	Age	EstimatedSalary	Purchased	Normalized_Salary	Gender_Catg
0	Male	19	19000	0	0.029630	1
1	Male	35	20000	0	0.037037	1

In [8]:
```
X = data[['Age','Normalized_Salary','Gender_Catg']].values
y = data['Purchased'].values

X_train,X_test,y_train,y_test = train_test_split(X, y, random_state=0, t

classifier = LogisticRegression(random_state=0)
classifier.fit(X_train, y_train)
```

Out[8]:
```
LogisticRegression
LogisticRegression(random_state=0)
```

2. Compute Confusion matrix to find TP, FP, TN, FN, Accuracy, Error rate, Precision, Recall on the given dataset.

In [9]:
```
y_pred = classifier.predict(X_test)
cm = confusion_matrix(y_test, y_pred)

TP, FN, FP, TN = cm[0][0], cm[0][1], cm[1][0], cm[1][1]
print(f'TP {TP} FN {FN} \nFP {FP} TN {TN}')
```

```
TP 56 FN 2
FP 5 TN 17
```

In [10]:
```
print(f'Accuracy: {(TP+TN)/(TP+TN+FP+FN)}')
print(f'Error: {(FP+FN)/(TP+TN+FP+FN)}')
print(f'Precison: {TP/(TP+FP)}')
print(f'Recall: {TP/(TP+FN)}')
```

```
Accuracy: 0.9125
Error: 0.0875
Precison: 0.9180327868852459
Recall: 0.9655172413793104
```