Experiment 10.

To implement KNN Technique using Python programming Language.

Name:	**Rishabh Anand**	**UID:**	19BCS4525
Branch:	CSE-IOT	**Sec/Grp:**	1-A
Sem:	6th	**Date:**	05/05/2022
Subject:	ML Lab	**Code:**	CSD-386

1. Aim/Overview of the practical:

To implement KNN Technique using Python programming Language.

2. Task to be done:

- Install the prerequisite Python
- Load the dataset.
- Apply KNN **Technique**
- Analyse results/output.

3. Steps for experiment/practical:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

```
datasets = pd.read_csv('Social_Network_Ads.csv')
X=datasets.iloc[:,[2,3]].values
Y=datasets.iloc[:, 4].values
```

```
datasets.head()
```

	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

```
datasets.head
```

```
<bound method NDFrame.head of       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
..        ...     ...  ...              ...        ...
395  15691863  Female   46            41000          1
396  15706071    Male   51            23000          1
397  15654296  Female   50            20000          1
398  15755018    Male   36            33000          0
399  15594041  Female   49            36000          1

[400 rows x 5 columns]>
```

```
# To Split the data set
from sklearn.model_selection import train_test_split
X_Train, X_Test, Y_Train, Y_Test=train_test_split(X,Y, test_size=0.25, random_state=0)
```

```
# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc_X= StandardScaler()
X_Train = sc_X.fit_transform(X_Train)
X_Test = sc_X.transform(X_Test)
```

```
# Fitting the classifier into the training set
from sklearn.neighbors import KNeighborsClassifier
classifier= KNeighborsClassifier(n_neighbors=5, metric='minkowski', p=2 )
classifier.fit(X_Train, Y_Train)
```

```
KNeighborsClassifier()
```

```
# Predicting the test set results
Y_Pred = classifier.predict(X_Test)
```

```
# To check accuracy
from sklearn.metrics import classification_report
print(classification_report(Y_Test, Y_Pred))
```

```
              precision    recall  f1-score   support

           0       0.96      0.94      0.95        68
           1       0.88      0.91      0.89        32

    accuracy                           0.93       100
   macro avg       0.92      0.92      0.92       100
weighted avg       0.93      0.93      0.93       100
```

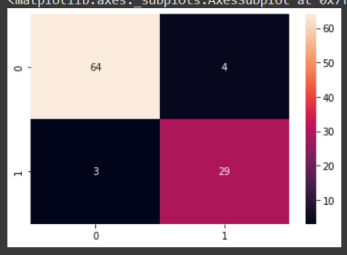
```
from sklearn.metrics import accuracy_score
print(accuracy_score(Y_Test, Y_Pred))
```

```
0.93
```

5. Result/Output/Writing Summary:

```
[ ]  # Visulize data by Confusion Matrix
     from sklearn.metrics import confusion_matrix
     cm = confusion_matrix(Y_Test, Y_Pred )
     print(cm)

     [[64  4]
      [ 3 29]]

[ ]  sns.heatmap(confusion_matrix(Y_Test, Y_Pred ))

     <matplotlib.axes._subplots.AxesSubplot at 0x7fb5fa387e10>
```


```
[ ]  sns.heatmap(confusion_matrix(Y_Test, Y_Pred ),annot=True)

     <matplotlib.axes._subplots.AxesSubplot at 0x7fb5f7a98bd0>
```



Learning outcomes (What I have learnt):

1. Implementing KNN using python.