

knn-classifier

April 5, 2024

0.1 To predict whether a person will purchase from Social Network Ads

```
[1]: ## Importing Necessary Libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report, \
    confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt
```

```
[2]: ## Loading dataset
data = pd.read_csv(r'C:\Users\ntpc\Desktop\Social_Network_Ads.csv')
data.head()
```

```
[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

```
[3]: data.dtypes
```

```
[3]: User ID          int64
Gender          object
Age            int64
EstimatedSalary int64
Purchased       int64
dtype: object
```

```
[4]: #checking each value counts
data['Purchased'].value_counts()
```

```
[4]: Purchased
0      257
1      143
```

Name: count, dtype: int64

```
[5]: #dropping User ID column
data = data.drop('User ID',axis = 1)
data.head()
```

```
[5]:
```

	Gender	Age	EstimatedSalary	Purchased
0	Male	19	19000	0
1	Male	35	20000	0
2	Female	26	43000	0
3	Female	27	57000	0
4	Male	19	76000	0

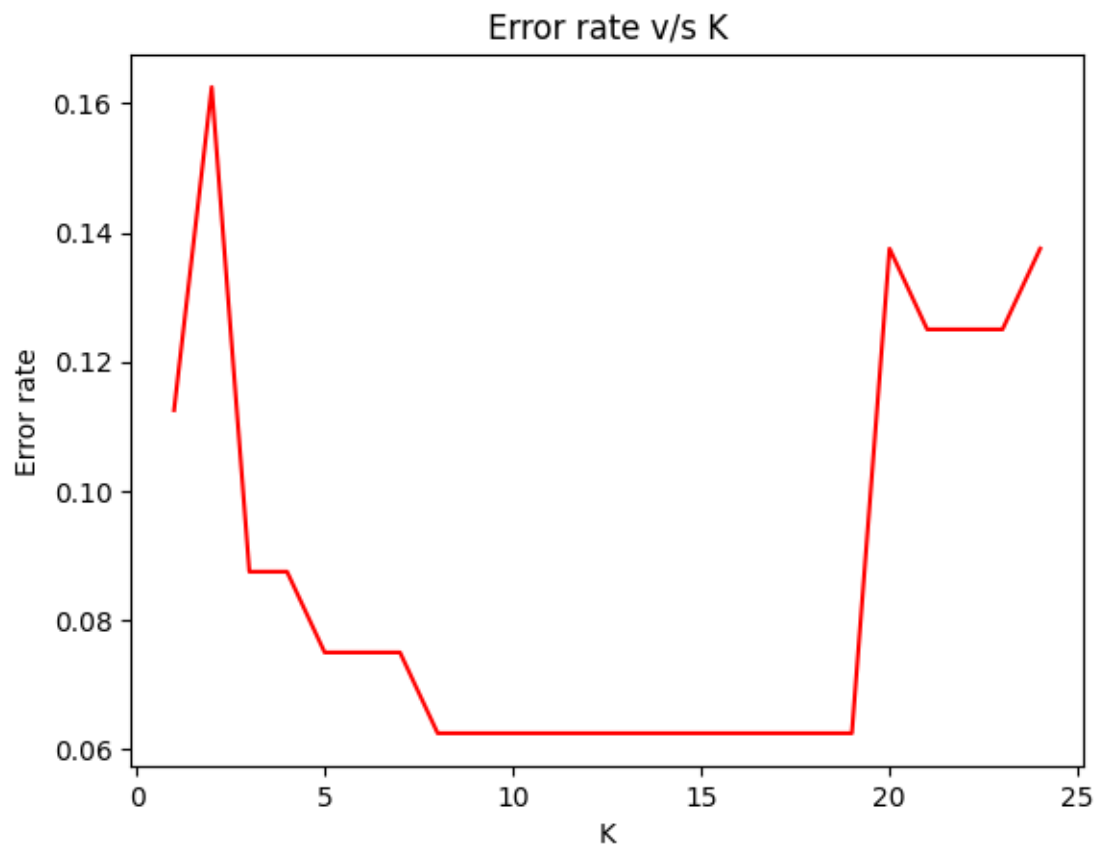
```
[6]: # Encoding categorical variables
le = LabelEncoder()
data['Gender'] = le.fit_transform(data['Gender'])
```

```
[7]: # Splitting the data into training and testing sets
X = data.drop('Purchased', axis=1)
y = data['Purchased']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
↪random_state=42)
```

```
[8]: ## Doing Feature Scaling for better accuracy
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

```
[9]: ## Selecting the k value
acc_list = []
err_list = []
for i in range(1,25):
    model = KNeighborsClassifier(n_neighbors=i)
    model.fit(X_train,y_train)
    y_pred = model.predict(X_test)
    acc = accuracy_score(y_test,y_pred)
    acc_list.append(acc)
    err_list.append(1-acc)
```

```
[10]: plt.plot(list(range(1,25)),err_list,c= 'r')
plt.title('Error rate v/s K')
plt.xlabel('K')
plt.ylabel('Error rate')
plt.show()
```



```
[11]: model = KNeighborsClassifier(n_neighbors=5)
      model.fit(X_train,y_train)
      y_pred = model.predict(X_test)
```

```
[ ]:
```