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

In [2]:
```
import sklearn.svm as svm
```

In [3]:
```
dataset = pd.read_csv(r'C:\Users\HP\Desktop\car_data.csv')
dataset.head()
```

Out[3]:

	User ID	Gender	Age	AnnualSalary	Purchased
0	385	Male	35	20000	0
1	681	Male	40	43500	0
2	353	Male	49	74000	0
3	895	Male	40	107500	1
4	661	Male	25	79000	0

In [4]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   User ID       1000 non-null   int64
 1   Gender        1000 non-null   object
 2   Age           1000 non-null   int64
 3   AnnualSalary  1000 non-null   int64
 4   Purchased     1000 non-null   int64
dtypes: int64(4), object(1)
memory usage: 39.2+ KB
```

In [5]:
```
dataset['Gender'].value_counts()
```

Out[5]:
```
Female    516
Male      484
Name: Gender, dtype: int64
```

In [6]:
```
#Converting gender values from object values to numerical values
#A sign Female to (0) and Male to (1)

convert = {"Gender": {"Female":0, "Male":1}}
```

In [7]:
```
dataset = dataset.replace(convert)
```

In [8]:
```
#dataset after convert the gender to numerical values
dataset
```

Out[8]:

	User ID	Gender	Age	AnnualSalary	Purchased
0	385	1	35	20000	0
1	681	1	40	43500	0
2	353	1	49	74000	0
3	895	1	40	107500	1
4	661	1	25	79000	0
...	...	...	...	...	...
995	863	1	38	59000	0
996	800	0	47	23500	0
997	407	0	28	138500	1
998	299	0	48	134000	1
999	687	0	44	73500	0

1000 rows × 5 columns

In [9]:
```
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, 4].values
```

In [10]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
```

In [11]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

In [12]:
```
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors=5)
classifier.fit(X_train, y_train)
```

Out[12]:
```
KNeighborsClassifier()
```

In [13]:
```
y_pred = classifier.predict(X_test)
```

```
C:\Users\HP\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis`), the default behavior of `mode` typically preserves the axis it acts along. In SciPy 1.11.0, this behavior will change: the default value of `keepdims` will become False, the `axis` over which the statistic is taken will be eliminated, and the value None will no longer be accepted. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

In [14]:
```
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
[[94 15]
 [10 81]]
              precision    recall  f1-score   support

           0       0.90      0.86      0.88       109
           1       0.84      0.89      0.87        91

    accuracy                           0.88       200
   macro avg       0.87      0.88      0.87       200
weighted avg       0.88      0.88      0.88       200
```

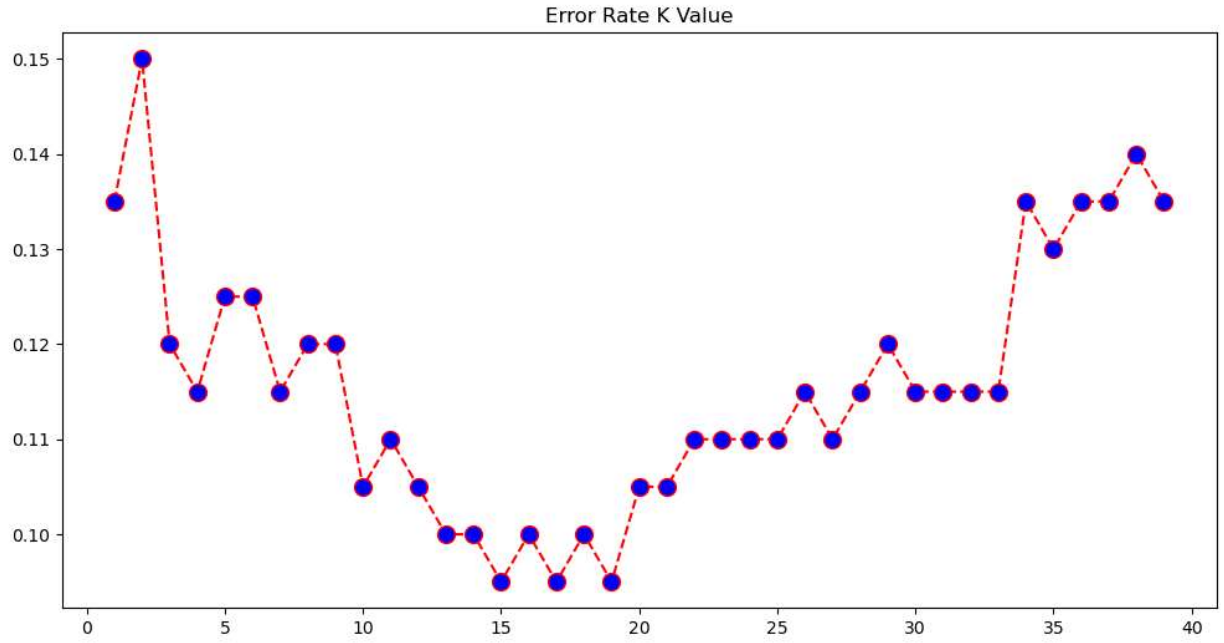
In [15]:
```
# Comparing the error values with the change in k values - the number of neighbor
#- by calculating the error value for k values between 1 and 40
#drawing the graph

error = []
# Calculating error for K values between 1 and 40
for i in range(1, 40):
    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(X_train, y_train)
    pred_i = knn.predict(X_test)
    error.append(np.mean(pred_i != y_test))
```

```
C:\Users\HP\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:
228: FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis
`), the default behavior of `mode` typically preserves the axis it acts alon
g. In SciPy 1.11.0, this behavior will change: the default value of `keepdims
` will become False, the `axis` over which the statistic is taken will be eli
minated, and the value None will no longer be accepted. Set `keepdims` to Tru
e or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Users\HP\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:
228: FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis
`), the default behavior of `mode` typically preserves the axis it acts alon
g. In SciPy 1.11.0, this behavior will change: the default value of `keepdims
` will become False, the `axis` over which the statistic is taken will be eli
minated, and the value None will no longer be accepted. Set `keepdims` to Tru
e or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Users\HP\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:
228: FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis
`), the default behavior of `mode` typically preserves the axis it acts alon
```

In [17]:
```
plt.figure(figsize=(12, 6))
plt.plot(range(1, 40), error,
   color='red', linestyle='dashed',
   marker='o', markerfacecolor='blue',
   markersize=10)
plt.title('Error Rate K Value')
#plt.xlabel('K Value') plt.ylabel('Mean Error')
```

Out[17]:
```
Text(0.5, 1.0, 'Error Rate K Value')
```



In [ ]: