Question 2: KNN Classification

[CM6]

Basic Model

[23]:
```
# Basic Model

# importing libraries
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn import metrics
from sklearn.metrics import roc_auc_score
from sklearn.metrics import f1_score
```

[24]:
```
# replacing "species" values with [0,1,2]
df_iris["species"]=df_iris["species"].replace("Iris-setosa",0).
 ↪replace("Iris-versicolor",1).replace("Iris-virginica",2)
#df_iris=pd.get_dummies(df_iris, columns = ['species'])
```

[25]:
```
# dividing data and target
y = df_iris['species']
X = df_iris.drop(['species'], axis = 1)
```

[26]:
```
# dividing the data into train, validation, and test sets (60%, 20%, 20%) with␣
 ↪random_state=275
X_train, X_test, y_train, y_test = train_test_split(X,  y, test_size=0.4,␣
 ↪random_state=275)

X_val, X_test, y_val, y_test= train_test_split(X_test, y_test, test_size=0.5,␣
 ↪random_state=275)
```

[27]:
```
# train the model with the classifier's default parameters
knn = KNeighborsClassifier()
knn.fit(X_train, y_train.values.ravel())
y_pred = knn.predict(X_test)
accuracy = metrics.accuracy_score(y_test, y_pred)
print('The accuracy of the basic KNN model with default parameters on the test set␣
 ↪is', accuracy * 100, '%')
```

```
The accuracy of the basic KNN model with default parameters on the test set is
90.47619047619048 %
```

[28]:
```
# finding best parameter for the classifier
k_range = [1,5,10,15,20,25,30,35]
Scores = {}
Scores_list = []
best_k = 0
accuracy_max = 0
for k in k_range:
    knn =  KNeighborsClassifier(n_neighbors = k)
    knn.fit(X_train, y_train.values.ravel())
```

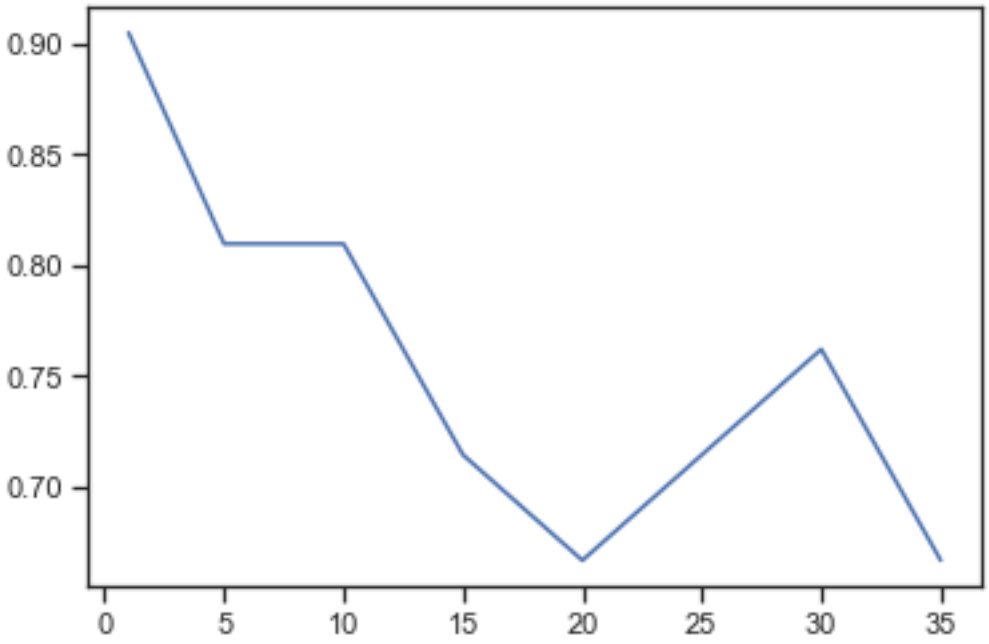
```
    y_pred = knn.predict(X_val)
    Scores[k] = metrics.accuracy_score(y_val, y_pred)
    Scores_list.append(metrics.accuracy_score(y_val, y_pred))

# plotting the graph showing relationship between the accuracy and parameters
plt.plot(k_range, Scores_list)

# finding best k value
accuracy = max(Scores_list)
key_list = list(Scores.keys())
val_list = list(Scores.values())
position = val_list.index(accuracy)
best_k = key_list[position]

print('The best value of k is', best_k)
print('The accuracy of the basic KNN model on the validation set is', accuracy * 100,␣
 ↪'%')
```

```
The best value of k is 1
The accuracy of the basic KNN model on the validation set is 90.47619047619048 %
```



We find that the model has the highest accuracy score for k=1. When k=1, probability estimation is based on a single sample i.e the nearest neighbor. This is very sensitive to distortions like noise, outliers etc. By using a higher value for k, the model becomes more robust against such distortions. Hence, selecting the k value with next best accuracy i.e k=5 & 10.