- Name: Prerna Sunil Jadhav
- Sap ID: 60004220127
- Batch: C22
- Branch: Computer Engineering
- Course: Machine Learning
- Experiment 6: K-Nearest Neighbours

```
import numpy as np
from sklearn.preprocessing import LabelEncoder

def euclidean_distance(p1, p2):
  return np.sqrt(np.sum(np.square(p1 - p2)))

def predict_knn(dataset, new_data, k):
  distances = np.array([euclidean_distance(point[:-1], new_data) for point in dataset])
  sorted_data = np.array([dataset[i] for i in np.argsort(distances)])
  k_nearest_targets = sorted_data[:k, -1]
  prediction = np.mean(k_nearest_targets)
  return prediction

def knn_categorical(dataset, unknown, k):
    num_data = dataset[:, :-1].astype(float)
    cat_data = dataset[:, -1]
    label_encoder = LabelEncoder()
    cat_data_encoded = label_encoder.fit_transform(cat_data)

    unknown_num = unknown[:-1].astype(float)
    distances = np.sqrt(np.sum((num_data - unknown_num)**2, axis=1))
    nearest_indices = np.argsort(distances)[:k]
    nearest_labels = cat_data_encoded[nearest_indices]
    prediction = np.argmax(np.bincount(nearest_labels))
    return label_encoder.inverse_transform([prediction])[0]

dataset2 = np.array([
    [167, 51, 'Underweight'],
    [182, 62, 'Normal'],
    [176, 69, 'Normal'],
    [173, 64, 'Normal'],
    [172, 65, 'Normal'],
    [174, 56, 'Underweight'],
    [169, 58, 'Normal'],
    [173, 57, 'Normal'],
    [170, 55, 'Normal']
])

dataset = np.array([
    [5,  45, 77],
    [5.11, 26, 47],
    [5.6,  30, 55],
    [5.9,  34, 59],
    [4.8,  40, 72],
    [5.8,  36, 60],
    [5.3,  19, 40],
    [5.8,  28, 60],
    [5.5,  23, 45],
    [5.6,  32, 58]
])

# new_data = np.array([170, 57, None])  # None as a placeholder for numerical data
new_data = np.array([5.5, 38])
# print(type(dataset[:, -1]))
if (dataset[:, -1]).dtype != 'float64':
  for i in range(1,6,2):
    prediction_cat = knn_categorical(dataset, new_data, i,typeofknn='categorical')
    print("Predicted value for categorical data k:",i,":", prediction_cat)
else:
  for i in range(1,6,2):
    prediction = predict_knn(dataset, new_data, i)
    print("Predicted target value for numeric data k:",i,":", prediction)
```

```
new_data = np.array([170, 57, None])  # None as a placeholder for numerical data
print("\n")
if (dataset2[:, -1]).dtype != 'float64':
  for i in range(1,6,2):
    prediction_cat = knn_categorical(dataset2, new_data, i)
    print("Predicted value for categorical data k:",i,":", prediction_cat)
else:
  for i in range(1,6,2):
    prediction = predict_knn(dataset2, new_data, i)
    print("Predicted target value for numeric data k:",i,":", prediction)
```

```
Predicted target value for numeric data k: 1 : 60.0
Predicted target value for numeric data k: 3 : 63.666666666666664
Predicted target value for numeric data k: 5 : 65.2


Predicted value for categorical data k: 1 : Normal
Predicted value for categorical data k: 3 : Normal
Predicted value for categorical data k: 5 : Normal
```

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

import sklearn.metrics as metrics

data = pd.read_csv('../content/iphone_purchase_records.csv')
print(data.head())

data = data.drop('Gender',axis=1)

X = data.drop('Purchase Iphone', axis=1)
y = data['Purchase Iphone']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)

for i in range(1,6,2):

  knn = KNeighborsClassifier(n_neighbors=i)

  knn.fit(X_train,y_train)

  y_pred_knn = knn.predict(X_test)

  score_knn = metrics.accuracy_score(y_test,knn.predict(X_test))
  print('Accuracy for k =',i,':{0:f}'.format(score_knn))
```

```
   Gender  Age  Salary  Purchase Iphone
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
Accuracy for k = 1 :0.840000
Accuracy for k = 3 :0.780000
Accuracy for k = 5 :0.830000
```