```
import pandas as pd
import numpy as np
XXX = pd.read_csv("tmp_hms_sav_ass.csv", sep=';')
XXX
```

	id	idResource	idProject	refType	refId	rate	assignedWork	realWork	left
0	1	5	1	Activity	1	100	NaN	21.00	
1	4	15	4	Activity	4	100	NaN	131.00	
2	6	15	4	Activity	6	100	NaN	21.00	
3	8	15	4	Activity	9	100	NaN	4.00	
4	9	15	4	Activity	10	100	NaN	2.00	
...	...	...	...	...	...	...	...	...	
7196	7601	9	34	Activity	594	100	NaN	0.00	
7197	7602	32	34	Activity	594	100	NaN	NaN	
7198	7603	9	34	Activity	597	100	NaN	0.25	
7199	7604	7	12	Activity	1524	100	NaN	0.50	
7200	7605	7	50	Activity	1525	100	NaN	2.50	

7201 rows × 25 columns

```
XX =  XXX[['idResource','idProject','refId','realCost','realWork']]
dataset = XX.replace(np.nan,0)
dataset = XX.replace(np.nan,0)
dataset['realWork'] =  dataset['realWork'].astype(int)
dataset['realCost'] =  dataset['realCost'].astype(int)
dataset
```

	idResource	idProject	refId	realCost	realWork
0	5	1	1	0	21
1	15	4	4	0	131

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

7196	9	34	594	0	0

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

7199	7	12	1524	0	0

```
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)
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(X_train)

X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

```
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors=1)
classifier.fit(X_train, y_train)
```

```
KNeighborsClassifier(n_neighbors=1)
```

```
y_pred = classifier.predict(X_test)
```

```
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

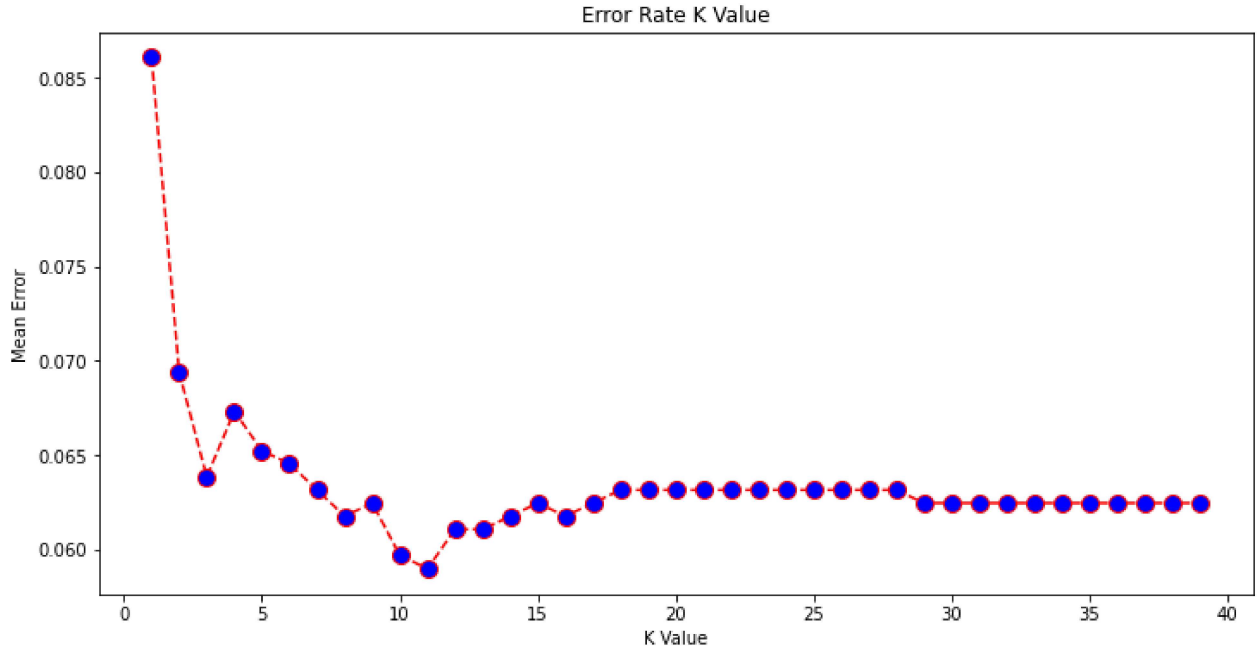
```
[[1308   15    6 ...    0    0    0]
 [   9    8    1 ...    0    0    0]
 [   6    4    1 ...    0    0    0]
 ...
 [   1    0    0 ...    0    0    0]
 [   1    0    0 ...    0    0    0]
 [   0    0    0 ...    0    0    0]]
              precision    recall  f1-score   support

           0       0.97      0.97      0.97      1351
           1       0.24      0.38      0.29        21
           2       0.08      0.08      0.08        13
           3       0.00      0.00      0.00         7
           4       0.00      0.00      0.00         5
```

```
           5       0.00      0.00      0.00         5
           6       0.00      0.00      0.00         4
           7       0.00      0.00      0.00         3
           8       0.00      0.00      0.00         6
           9       0.00      0.00      0.00         2
          10       0.00      0.00      0.00         1
          11       0.00      0.00      0.00         0
          12       0.00      0.00      0.00         1
          13       0.00      0.00      0.00         1
          14       0.00      0.00      0.00         2
          15       0.00      0.00      0.00         2
          18       0.00      0.00      0.00         3
          19       0.00      0.00      0.00         1
          20       0.00      0.00      0.00         0
          21       0.00      0.00      0.00         2
          22       0.00      0.00      0.00         1
          23       0.00      0.00      0.00         1
          27       0.00      0.00      0.00         0
          32       0.00      0.00      0.00         0
          35       0.00      0.00      0.00         0
          37       0.00      0.00      0.00         1
          38       0.00      0.00      0.00         1
          39       0.00      0.00      0.00         1
          54       0.00      0.00      0.00         2
          62       0.00      0.00      0.00         1
          81       0.00      0.00      0.00         1
         163       0.00      0.00      0.00         1
         214       0.00      0.00      0.00         1
         235       0.00      0.00      0.00         0

    accuracy                           0.91      1441
   macro avg       0.04      0.04      0.04      1441
weighted avg       0.92      0.91      0.92      1441

C:\Users\saifn\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1221: U
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\saifn\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1221: U
  _warn_prf(average, modifier, msg_start, len(result))
```

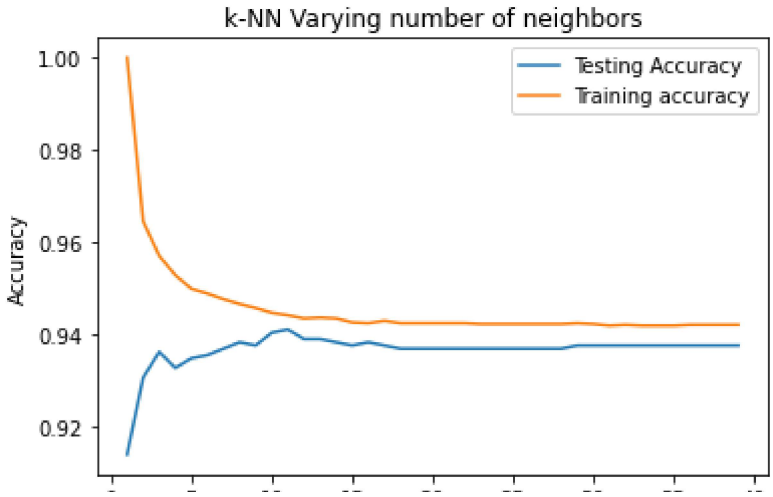
```
error = []
# Calculating error for K values between 1 and 40
for i in range(1, 40):
    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(X_train, y_train)
    pred_i = knn.predict(X_test)
    error.append(np.mean(pred_i != y_test))
```

```
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')
```

```
Text(0, 0.5, 'Mean Error')
```




```
from sklearn.neighbors import KNeighborsClassifier

#Setup arrays to store training and test accuracies
neighbors = np.arange(1,40)
train_accuracy =np.empty(len(neighbors))
test_accuracy = np.empty(len(neighbors))
#Compute accuracy on the training set
for i,k in enumerate(neighbors):
    #Setup a knn classifier with k neighbors
    knn = KNeighborsClassifier(n_neighbors=k)

    #Fit the model
    knn.fit(X_train, y_train)

    train_accuracy[i] = knn.score(X_train, y_train)

    #Compute accuracy on the test set

    test_accuracy[i] = knn.score(X_test, y_test)


#Generate plot
plt.title('k-NN Varying number of neighbors')
plt.plot(neighbors, test_accuracy, label='Testing Accuracy')
plt.plot(neighbors, train_accuracy, label='Training accuracy')
plt.legend()
plt.xlabel('Number of neighbors')
plt.ylabel('Accuracy')
plt.show()
```




```
knn.score(X_test, y_test)
```

```
0.9375433726578765
```