1.

```python
from random import random
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.neighbors import KNeighborsClassifier
import matplotlib.pyplot as plt
import seaborn as sns


df = pd.read_csv('hsbdemo.csv')

print(df.head())

# coverting features to numeric values
df.loc[df['gender'] == 'male' , 'gender'] = 0
df.loc[df['gender'] == 'female' , 'gender'] = 1

df.loc[df['ses'] == 'low' , 'ses'] = 0
df.loc[df['ses'] == 'middle' , 'ses'] = 1
df.loc[df['ses'] == 'high' , 'ses'] = 2

df.loc[df['schtyp'] == 'public' , 'schtyp'] = 0
df.loc[df['schtyp'] == 'private' , 'schtyp'] = 1

df.loc[df['honors'] == 'not enrolled' , 'honors'] = 0
df.loc[df['honors'] == 'enrolled' , 'honors'] = 1

df.loc[df['prog'] == 'vocation' , 'prog'] = 0
df.loc[df['prog'] == 'general' , 'prog'] = 1
df.loc[df['prog'] == 'academic' , 'prog'] = 2

print(df.head())


X = np.array(df.loc[:, ['gender','ses','schtyp', 'read', 'write','math','science', 'socst','honors', 'awards']])
y = np.array(df['prog'])
y = y.astype('int')

#use random_state=value to select the same data points in every run
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.10, random_state=3) #set seed to 42 all the time
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)

pred = knn.predict(X_test)
```

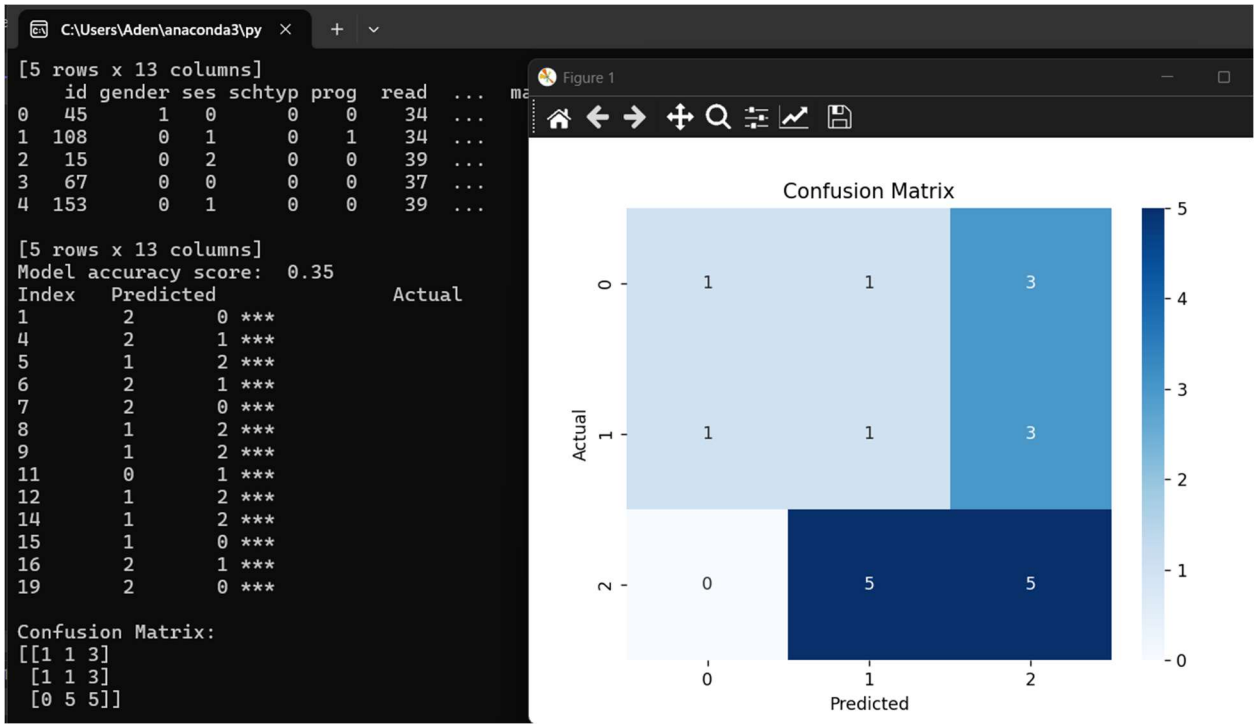
```python
pred = knn.predict(X_test)

print('Model accuracy score: ', accuracy_score(y_test, pred))
print('Index\tPredicted\t\tActual')
for i in range(len(pred)):
    if pred[i] != y_test[i]:
        print(i, '\t', pred[i], '\t', y_test[i], '***')

conf_matrix = confusion_matrix(y_test, pred)

print(f'\nConfusion Matrix: \n{conf_matrix}')
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
xticklabels=knn.classes_, yticklabels=knn.classes_)

plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```


C:\Users\Aden\anaconda3\py
[5 rows x 13 columns]
id gender ses schtyp prog read ... ma
0 45 1 0 0 0 34 ...
1 108 0 1 0 1 34 ...
2 15 0 2 0 0 39 ...
3 67 0 0 0 0 37 ...
4 153 0 1 0 0 39 ...
[5 rows x 13 columns]
Model accuracy score: 0.35
Index Predicted Actual
1 2 0 ***
4 2 1 ***
5 1 2 ***
6 2 1 ***
7 2 0 ***
8 1 2 ***
9 1 2 ***
11 0 1 ***
12 1 2 ***
14 1 2 ***
15 1 0 ***
16 2 1 ***
19 2 0 ***
Confusion Matrix:
[[1 1 3]
[1 1 3]
[0 5 5]]
Figure 1
Confusion Matrix
Actual
Predicted
0
1
2
1
1
3
1
1
3
0
5
5
5
4
3
2
1
0

2.

```python
from random import random
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.neighbors import KNeighborsClassifier
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA


df = pd.read_csv('hsbdemo.csv')

print(df.head())

# coverting features to numeric values
df.loc[df['gender'] == 'male' , 'gender'] = 0
df.loc[df['gender'] == 'female' , 'gender'] = 1

df.loc[df['ses'] == 'low' , 'ses'] = 0
df.loc[df['ses'] == 'middle' , 'ses'] = 1
df.loc[df['ses'] == 'high' , 'ses'] = 2

df.loc[df['schtyp'] == 'public' , 'schtyp'] = 0
df.loc[df['schtyp'] == 'private' , 'schtyp'] = 1

df.loc[df['honors'] == 'not enrolled' , 'honors'] = 0
df.loc[df['honors'] == 'enrolled' , 'honors'] = 1

df.loc[df['prog'] == 'vocation' , 'prog'] = 0
df.loc[df['prog'] == 'general' , 'prog'] = 1
df.loc[df['prog'] == 'academic' , 'prog'] = 2

print(df.head())

columns = ['gender','ses','schtyp', 'read', 'write','math','science', 'socst','honors', 'awards']
X = np.array(df.loc[:, columns])
y = np.array(df['prog'])
y = y.astype('int')

print(f'Before Standrdization: {X}')
X = StandardScaler().fit_transform(X)
print(f'After Standrdization: {X}')
X = StandardScaler().fit_transform(X)
print('Standard')
print(X)
```

```python
x= np.arange(1,11)

pca = PCA(n_components=10)
principalComponents = pca.fit_transform(X)
explained_variance = pca.explained_variance_ratio_
print(f'Variance: {explained_variance}')
print(np.cumsum(explained_variance))

plt.plot(x, np.cumsum(explained_variance))
plt.xlabel('Principal Components')
plt.ylabel('Cumulative Ratio')
plt.title('PC= 1-10')
plt.xticks(range(1,11))
plt.show()
```

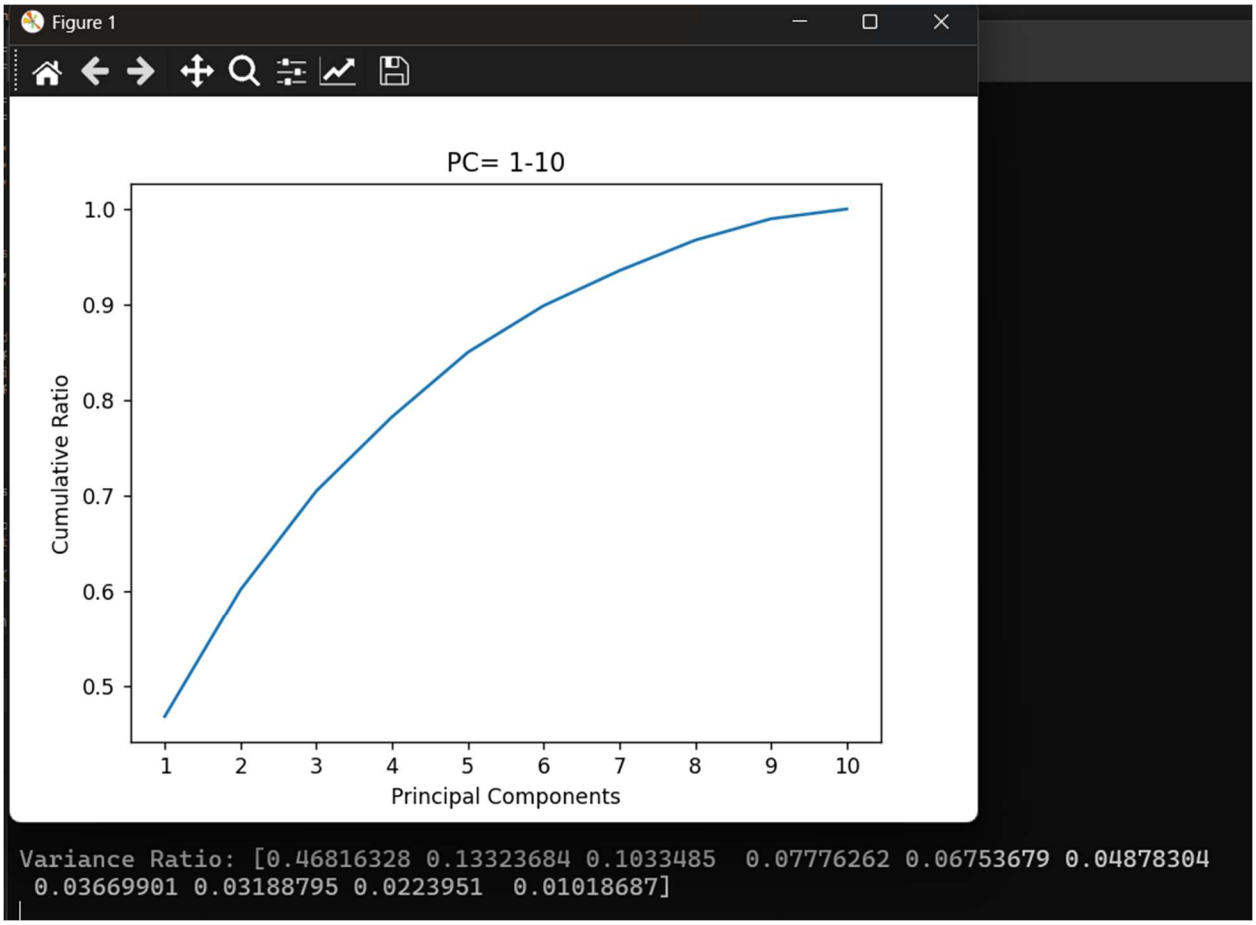


Figure 1
PC= 1-10
Cumulative Ratio
1.0
0.9
0.8
0.7
0.6
0.5
1
2
3
4
5
6
7
8
9
10
Principal Components
Variance Ratio: [0.46816328 0.13323684 0.1033485 0.07776262 0.06753679 0.04878304
 0.03669901 0.03188795 0.0223951 0.01018687]