```
and average temperatures for all dates
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from scipy import stats

df=pd.read_csv('avgHigh_jan_1895-2018.csv')
#print(df)

df=df.drop(['Anomaly'],axis=1)
#print(df)

x=np.array(df['Date'])
y=np.array(df['Value'])

slope,intercept,r,p,std_err=stats.linregress(x,y)
print('Slope: ',slope,'y-intercept: ',intercept,'Correlation(r): ',r,'p-value: ',p,'Standard Error: ',std_err)
mymodel=(slope*x)+intercept
p=np.array([201901,202301,202401])
prediction=(slope*p)+intercept
print(f'predicted temps {prediction}')

plt.scatter(x,y,c='b',label='Data Points')
plt.scatter(p,prediction,c='g',label='Predicted')
plt.plot(x,mymodel,c='r',label='Model')
plt.xlabel('Years')
plt.ylabel('Temperatures')
plt.title(f'Janurary avg high temp. slope: {round(slope,3)} intercept: {round(intercept,2)}')
plt.legend(loc='lower right')
plt.show()
```

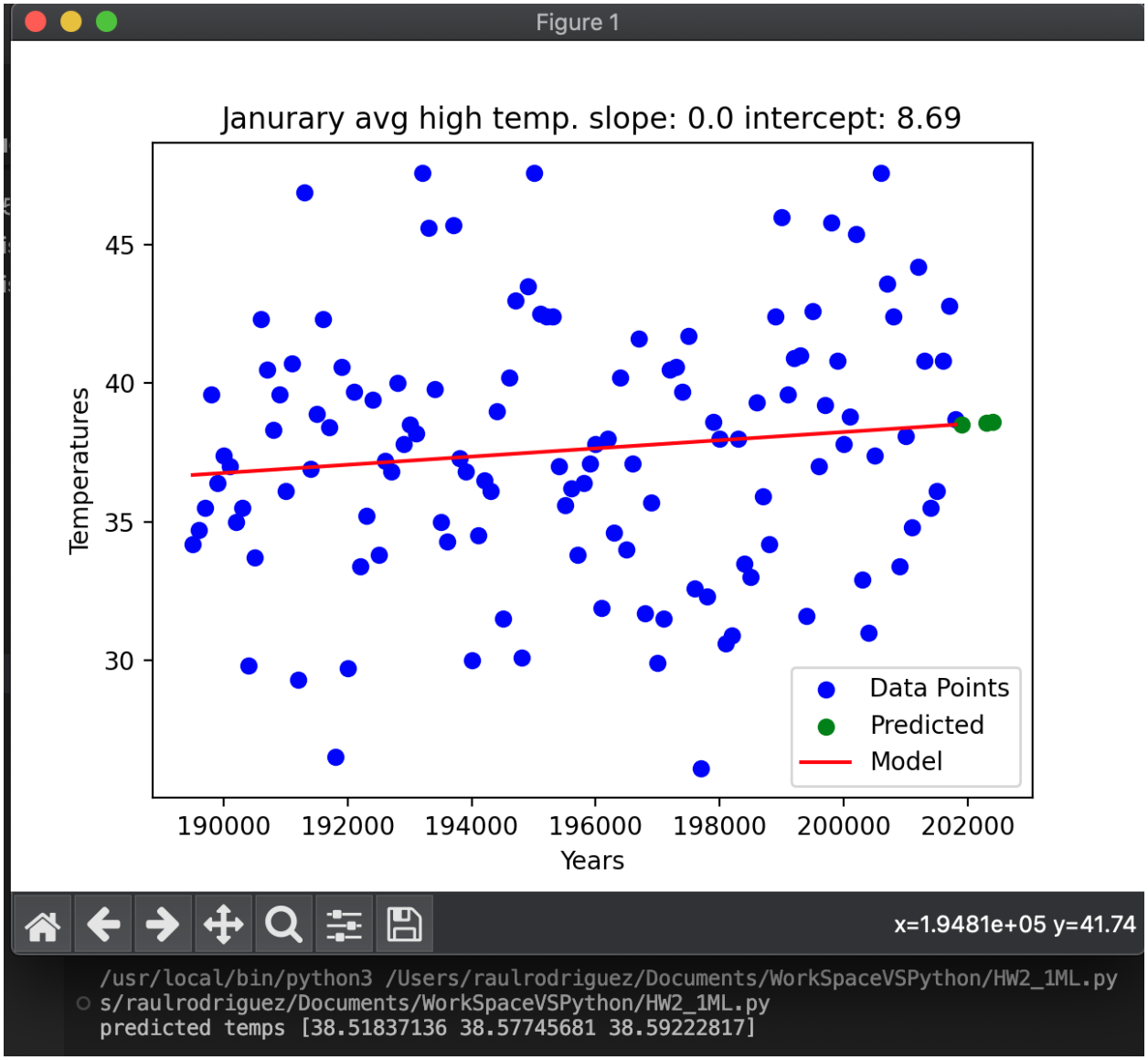
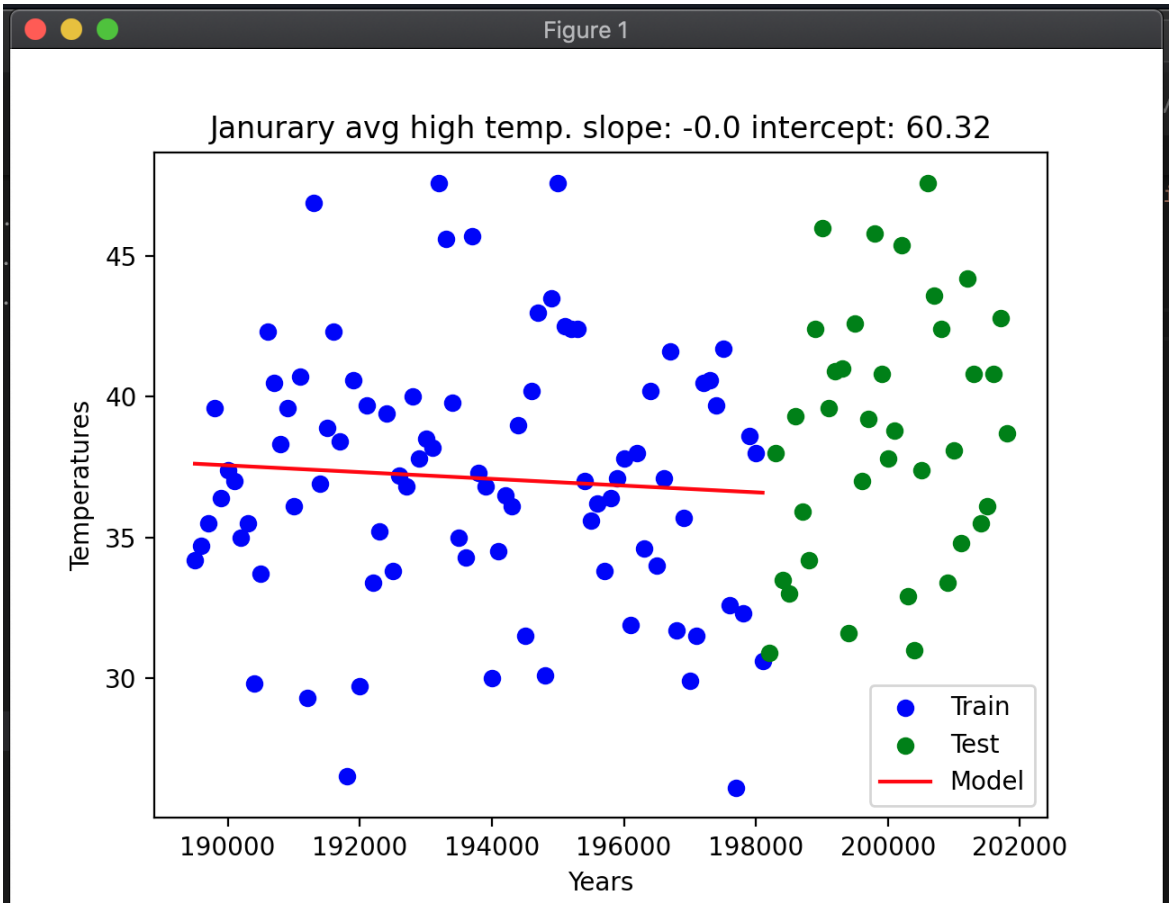


Figure 1
Janurary avg high temp. slope: 0.0 intercept: 8.69
Temperatures
45
40
35
30
190000
192000
194000
196000
198000
200000
202000
Years
Data Points
Predicted
Model
x=1.9481e+05 y=41.74
/usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceVSPython/HW2_1ML.py
s/raulrodriguez/Documents/WorkSpaceVSPython/HW2_1ML.py
predicted temps [38.51837136 38.57745681 38.59222817]

```
Note 2: You can use any built in functions you wish apart from functions to split the data set
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from scipy import stats
df=pd.read_csv('avgHigh_jan_1895-2018.csv')
df=df.drop(['Anomaly'],axis=1)
p=float(input("Enter test size percentage "))
print(len(df))
n=round(p*len(df))
print(n)
trainDF=df.iloc[:n]
testDF=df.iloc[n:]
xTrain=np.array(trainDF.iloc[:,0])
yTrain=np.array(trainDF.iloc[:,1])
xTest=np.array(testDF.iloc[:,0])
yTest=np.array(testDF.iloc[:,1])
slope,intercept,r,p,std_err=stats.linregress(xTrain,yTrain)
#print('Slope: ',slope,'y-intercept: ',intercept,'Correlation(r): ',r,'p-value: ',p,'Standard Error: ',std_err)
mymodel=(slope*xTrain)+intercept
prediction=(slope*xTest)+intercept
print(f'predicted temps {prediction}')
print(f'actual temp {yTest}')
plt.scatter(xTrain,yTrain,c='b',label='Train')
plt.scatter(xTest,yTest,c='g',label='Test')
plt.plot(xTrain,mymodel,c='r',label='Model')
plt.xlabel('Years')
plt.ylabel('Temperatures')
plt.title(f'Janurary avg high temp. slope: {round(slope,3)} intercept: {round(intercept,2)}')
plt.legend(loc='lower right')
plt.show()
```




```
/usr/local/bin/python3 /Users/raulrodriguez/Documents/WorkSpaceVSPython/HW2_2ML.py
raulrodriguez@Rauls-MacBook-Air WorkSpaceVSPython % /usr/local/bin/python3 /Users/raulrodriguez
Enter test size percentage .7
124
87
predicted temps [36.57626303 36.56428155 36.55230007 36.54031858 36.5283371  36.51635562
 36.50437413 36.49239265 36.48041117 36.46842969 36.4564482  36.44446672
 36.43248524 36.42050375 36.40852227 36.39654079 36.38455931 36.37257782
 36.36059634 36.34861486 36.33663337 36.32465189 36.31267041 36.30068893
 36.28870744 36.27672596 36.26474448 36.25276299 36.24078151 36.22880003
 36.21681855 36.20483706 36.19285558 36.1808741  36.16889262 36.15691113
 36.14492965]
actual temp [30.9 38.  33.5 33.  39.3 35.9 34.2 42.4 46.  39.6 40.9 41.  31.6 42.6
 37.  39.2 45.8 40.8 37.8 38.8 45.4 32.9 31.  37.4 47.6 43.6 42.4 33.4
 38.1 34.8 44.2 40.8 35.5 36.1 40.8 42.8 38.7]
```

```
components and print the variance ratio as shown in the figure below
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
import seaborn as sns
from pandas import DataFrame

names=['ID number','radius','texture','perimeter','area','smoothness','compactness','concavity','concave points',
'symmetry','fractal dimension']
df=pd.read_csv('wdbc.data.csv',names=names)
df=df.drop(df.columns[0],axis=1)
x = np.array(df.loc[:, 'texture' : 'fractal dimension'])
y = np.array(df['radius'])
x=StandardScaler().fit_transform(x)
pca=PCA(n_components=2)
principalComponents=pca.fit_transform(x)
explained_variance=pca.explained_variance_ratio_
df_comp=pd.DataFrame(pca.components_)
pc1=principalComponents[:,0]
pc2=principalComponents[:,1]
plt.scatter(principalComponents[:,0],principalComponents[:,1],c='g')
plt.xlabel('pc1')
plt.ylabel('pc2')
plt.title(f'PCA=2 Variance: {explained_variance}')
plt.show()
```

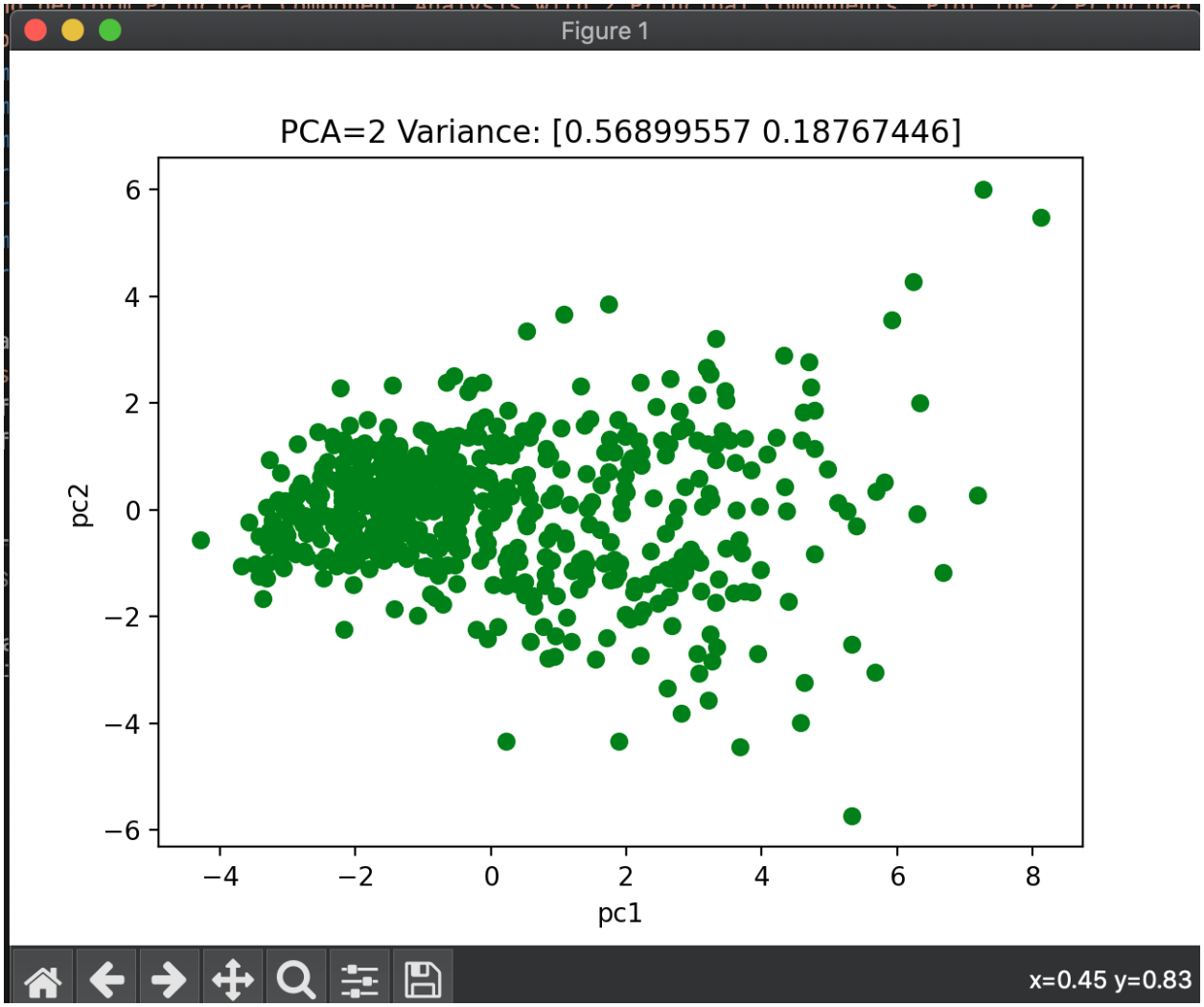


Figure 1
PCA=2 Variance: [0.56899557 0.18767446]
pc2
pc1
6
4
2
0
−2
−4
−6
−4
−2
0
2
4
6
8
x=0.45 y=0.83