```
import pandas as pd
import numpy as np


# read the data in a pandas dataframe
data = pd.read_csv("austin_weather.csv")

# drop or delete the unnecessary columns in the data.
data = data.drop(['Events', 'Date', 'SeaLevelPressureHighInches',
                  'SeaLevelPressureLowInches'], axis = 1)

data.head()
```

```
# some values have 'T' which denotes trace rainfall
# we need to replace all occurrences of T with 0
# so that we can use the data in our model
data = data.replace('T', 0.0)

# the data also contains '-' which indicates no
# or NIL. This means that data is not available
# we need to replace these values as well.
data = data.replace('-', 0.0)

# save the data in a csv file
data.to_csv('austin_final.csv')
```

```
import pandas as pd
import numpy as np
import sklearn as sk
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split

data = pd.read_csv("austin_final.csv")
data.drop(data.columns[data.columns.str.contains('unnamed',case = False)],axis = 1,
data.head()
```

	TempHighF	TempAvgF	TempLowF	DewPointHighF	DewPointAvgF	DewPointLowF	Hu
0	74	60	45	67.0	49.0	43.0	
1	56	48	39	43.0	36.0	28.0	
2	58	45	32	31.0	27.0	23.0	
3	61	46	31	36.0	28.0	21.0	
4	58	50	41	44.0	40.0	36.0	

```
# the features or the 'x' values of the data
# these columns are used to train the model
# the last column, i.e, precipitation column
# will serve as the label
X = data.drop(['PrecipitationSumInches'], axis = 1)

# the output or the label.
Y = data['PrecipitationSumInches']
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)

# consider a random day in the dataset
# we shall plot a graph and observe this
# day
# day_index = 798
# days = [i for i in range(Y.size)]


clf = LinearRegression()
clf.fit(X_train,y_train)
```

```
LinearRegression()
```

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

```
array([-1.51463070e-01,  1.95833119e-01,  6.39370575e-01, -5.25195862e-02,
        2.51222829e-01, -7.27247049e-02,  1.31133769e-01,  5.64228950e-01,
       -6.34277199e-02,  2.92726788e-02, -2.32002722e-01,  1.43549980e-01,
       -1.68311313e-01,  2.12218270e-01, -2.54061878e-01, -2.99463186e-02,
        1.42634265e-01,  4.68015158e-01,  1.62426794e-01,  5.09349240e-02,
       -2.65738057e-02, -8.79616565e-02, -2.66894575e-02,  4.08993237e-01,
       -1.28369054e-01, -1.17954453e-01,  8.05473874e-03,  5.79142689e-02,
        4.33166925e-01,  3.83098965e-01, -2.60887264e-01, -1.70375024e-02,
        5.56254748e-01,  5.60003314e-02,  5.96975657e-01,  4.79552599e-02,
        2.20372743e-02,  4.85907564e-02,  2.96619185e-01, -7.29662011e-02,
        5.97912367e-01, -1.64285314e-01,  2.68871401e-02,  8.07603786e-01,
        5.78234679e-02,  2.73419846e-01,  9.87205919e-02,  4.03629592e-01,
        3.88518101e-01, -5.07300884e-02,  1.90148607e-01, -6.95146225e-02,
        3.01731791e-02,  2.27090874e-02,  8.95042164e-05, -7.98114831e-02,
        1.69960248e-01, -1.50110321e-02,  3.65479398e-01, -5.47190258e-03,
        5.02556551e-01,  4.55488068e-01, -1.91462826e-02,  1.32881332e-01,
        3.21956936e-02,  1.26060402e-02, -4.88241384e-02,  2.77508905e-01,
       -4.47076262e-02,  2.30895296e-01,  5.43591668e-01,  1.50185667e-01,
```

```
       -1.20974985e-01, -4.41097755e-02,  3.84626371e-01,  4.39234675e-01,
       -2.28911040e-02,  1.13306748e-01, -7.64265922e-02, -1.84184866e-01,
        2.70691856e-02,  2.54356294e-01,  5.42691947e-02,  2.14878960e-02,
       -4.98830464e-02,  5.88934326e-02,  2.69240402e-01,  6.36517164e-02,
       -3.19330261e-02,  6.83983020e-02,  6.70476554e-01,  3.87617587e-02,
        4.70869904e-01,  1.21358084e-01, -7.55258281e-03, -3.60940721e-02,
       -4.61367436e-02, -6.47174828e-03, -8.91654455e-02, -1.51767393e-01,
       -1.26010241e-01, -1.51976692e-02, -2.86195129e-02, -4.90718263e-03,
       -3.28533914e-02,  3.10903660e-01,  5.50408252e-01,  4.76153443e-01,
       -4.54026384e-02, -5.98458599e-02,  1.66753655e-01,  2.99462723e-02,
        7.86888782e-02, -7.00090746e-02, -6.97857531e-02, -1.37373814e-02,
        3.26124889e-02,  2.47751413e-02,  5.53967284e-02,  1.61187501e-02,
        9.01543528e-01,  1.26399252e-01,  1.49197121e-01,  7.93379970e-03,
        8.02929616e-01, -2.18780137e-02,  6.55878990e-01, -1.75994299e-01,
        6.21639634e-01,  4.67898912e-01, -2.51156531e-02, -8.29588170e-02,
       -1.64804263e-02, -2.89005047e-02,  3.45186974e-01,  4.86890170e-02,
        1.63434372e-02, -9.27867562e-03,  1.70368466e-01,  4.30789356e-01,
        5.94459026e-02, -1.14197203e-01,  1.99950752e-01,  3.75022021e-02,
        4.26987748e-01,  5.40393600e-01, -5.94222999e-02,  1.54863716e-02,
       -7.29394653e-02,  6.13426788e-01, -9.56347355e-02, -2.12924218e-01,
        4.59383609e-01,  9.80019784e-02,  7.03104509e-01, -5.75867970e-02,
        9.52869878e-02, -2.04856990e-02,  3.15743886e-02,  9.65342512e-02,
        4.33114735e-01,  1.76360023e-01,  6.63696812e-01, -2.90029771e-03,
       -1.72533163e-01,  2.43870814e-02, -1.25053221e-03,  5.41747104e-02,
        1.21765194e-01,  1.07613453e-01, -6.00115205e-02,  3.42504988e-01,
       -1.00552740e-01,  4.82125883e-02,  9.13373493e-02, -1.57256441e-01,
        8.87629811e-02,  1.99784537e-01,  1.11730358e-01,  5.70894814e-02,
        7.44141363e-02,  3.35509839e-01,  3.67329818e-01, -1.65740746e-02,
       -8.64222422e-02, -1.48697105e-01,  3.18175494e-02,  1.75384488e-01,
       -7.67035531e-02, -5.28878969e-02, -1.62275474e-02,  4.46991620e-02,
       -2.54424983e-01, -6.94968788e-02,  2.71771197e-02, -1.14957828e-02,
        3.23546013e-01,  9.77147911e-02,  4.60237765e-01,  1.78122005e-01,
       -3.60182433e-02, -8.40115827e-04, -1.69742974e-01, -1.40525562e-02,
       -8.76517942e-02,  9.54932323e-01, -1.04503018e-02,  4.05220759e-03,
       -1.06055003e-01,  9.58828830e-02, -1.36886214e-01, -2.17814039e-01,
       -4.39271015e-02,  3.01336643e-01,  1.76092960e-01,  2.01429513e-01,
       -1.68067730e-01, -1.01312480e-01, -1.77472905e-02,  1.28311625e-01,
       -1.70242227e-01, -1.46689876e-04, -9.14913221e-02,  2.16442374e-02,
        4.93666378e-03,  2.09086259e-02, -1.60391138e-01, -2.39361794e-02,
        3.10953767e-01, -9.48388609e-02, -1.98836686e-02,  4.39408931e-02,
```

```
plt.scatter(y_test, y_pred)
plt.xlabel("True values")
plt.ylabel("Predictions")
plt.show()
```

```
print("the precipitation trend graph: ")
plt.scatter(days, Y, color = 'g')
plt.scatter(days[day_index], Y[day_index], color ='r')
plt.title("Precipitation level")
plt.xlabel("Days")
plt.ylabel("Precipitation in inches")

plt.show()
```

```
x_vis = X.filter(['TempAvgF', 'DewPointAvgF', 'HumidityAvgPercent',
                  'SeaLevelPressureAvgInches', 'VisibilityAvgMiles',
                  'WindAvgMPH'], axis = 1)

# plot a graph with a few features (x values)
# against the precipitation or rainfall to observe
# the trends

print("Precipitation vs selected attributes graph: ")

for i in range(x_vis.columns.size):
    plt.subplot(3, 2, i + 1)
    plt.scatter(days, x_vis[x_vis.columns.values[i][:100]],
                                            color = 'g')

    plt.scatter(days[day_index],
                x_vis[x_vis.columns.values[i]][day_index],
                color ='r')

    plt.title(x_vis.columns.values[i])

plt.show()
```

```
X.shape, data.shape
```

```
((1319, 16), (1319, 17))
```

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

```
0.3239676456837982
```

✓ 0s completed at 07:38