Assign2

In [61]:
```
import pandas as pd
import numpy as py
import matplotlib.pyplot as plt
```

In [62]:
```
data = pd.read_csv('temperatures.csv')
```

In [63]:
```
data
```

Out[63]:

	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	ANNU
0	1901	22.40	24.14	29.07	31.91	33.41	33.18	31.21	30.39	30.47	29.97	27.31	24.49	2
1	1902	24.93	26.58	29.77	31.78	33.73	32.91	30.92	30.73	29.80	29.12	26.31	24.04	2
2	1903	23.44	25.03	27.83	31.39	32.91	33.00	31.34	29.98	29.85	29.04	26.08	23.65	2
3	1904	22.50	24.73	28.21	32.02	32.64	32.07	30.36	30.09	30.04	29.20	26.36	23.63	2
4	1905	22.00	22.83	26.68	30.01	33.32	33.25	31.44	30.68	30.12	30.67	27.52	23.82	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
112	2013	24.56	26.59	30.62	32.66	34.46	32.44	31.07	30.76	31.04	30.27	27.83	25.37	2
113	2014	23.83	25.97	28.95	32.74	33.77	34.15	31.85	31.32	30.68	30.29	28.05	25.08	2
114	2015	24.58	26.89	29.07	31.87	34.09	32.48	31.88	31.52	31.55	31.04	28.10	25.67	2
115	2016	26.94	29.72	32.62	35.38	35.72	34.03	31.64	31.79	31.66	31.98	30.11	28.01	3
116	2017	26.45	29.46	31.60	34.95	35.84	33.82	31.88	31.72	32.22	32.29	29.60	27.18	3

117 rows × 18 columns

In [64]:
```
data.shape
```

Out[64]:
```
(117, 18)
```

In [65]:
```
data.dtypes
```

Out[65]:
```
YEAR         int64
JAN        float64
FEB        float64
MAR        float64
APR        float64
MAY        float64
JUN        float64
JUL        float64
AUG        float64
SEP        float64
OCT        float64
NOV        float64
DEC        float64
ANNUAL     float64
JAN-FEB    float64
MAR-MAY    float64
JUN-SEP    float64
OCT-DEC    float64
dtype: object
```

In [66]:
```
data.columns
```

Out[66]:
```
Index(['YEAR', 'JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL', 'AUG', 'SEP',
       'OCT', 'NOV', 'DEC', 'ANNUAL', 'JAN-FEB', 'MAR-MAY', 'JUN-SEP',
       'OCT-DEC'],
      dtype='object')
```

In [67]:
```
data.isnull().sum()
```

Out[67]:
```
YEAR       0
JAN        0
FEB        0
MAR        0
APR        0
MAY        0
JUN        0
JUL        0
AUG        0
SEP        0
OCT        0
NOV        0
DEC        0
ANNUAL     0
JAN-FEB    0
MAR-MAY    0
JUN-SEP    0
OCT-DEC    0
dtype: int64
```

In [68]:
```
data.describe()
```

Out[68]:

	YEAR	JAN	FEB	MAR	APR	MAY	JUN	
count	117.000000	117.000000	117.000000	117.000000	117.000000	117.000000	117.000000	117.0
mean	1959.000000	23.687436	25.597863	29.085983	31.975812	33.565299	32.774274	31.0
std	33.919021	0.834588	1.150757	1.068451	0.889478	0.724905	0.633132	0.4
min	1901.000000	22.000000	22.830000	26.680000	30.010000	31.930000	31.100000	29.7
25%	1930.000000	23.100000	24.780000	28.370000	31.460000	33.110000	32.340000	30.7
50%	1959.000000	23.680000	25.480000	29.040000	31.950000	33.510000	32.730000	31.0
75%	1988.000000	24.180000	26.310000	29.610000	32.420000	34.030000	33.180000	31.3
max	2017.000000	26.940000	29.720000	32.620000	35.380000	35.840000	34.480000	32.7

In [69]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 117 entries, 0 to 116
Data columns (total 18 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   YEAR     117 non-null    int64  
 1   JAN      117 non-null    float64
 2   FEB      117 non-null    float64
 3   MAR      117 non-null    float64
 4   APR      117 non-null    float64
 5   MAY      117 non-null    float64
 6   JUN      117 non-null    float64
 7   JUL      117 non-null    float64
 8   AUG      117 non-null    float64
 9   SEP      117 non-null    float64
 10  OCT      117 non-null    float64
 11  NOV      117 non-null    float64
 12  DEC      117 non-null    float64
 13  ANNUAL   117 non-null    float64
 14  JAN-FEB  117 non-null    float64
 15  MAR-MAY  117 non-null    float64
 16  JUN-SEP  117 non-null    float64
 17  OCT-DEC  117 non-null    float64
dtypes: float64(17), int64(1)
memory usage: 16.6 KB
```

January Month

Training and testing the data

In [94]:
```
X = data[['YEAR']]
Y = data[['JAN']]
```

In [95]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(X,Y,test_size = 0.3,random_stat
```

Linear Regression

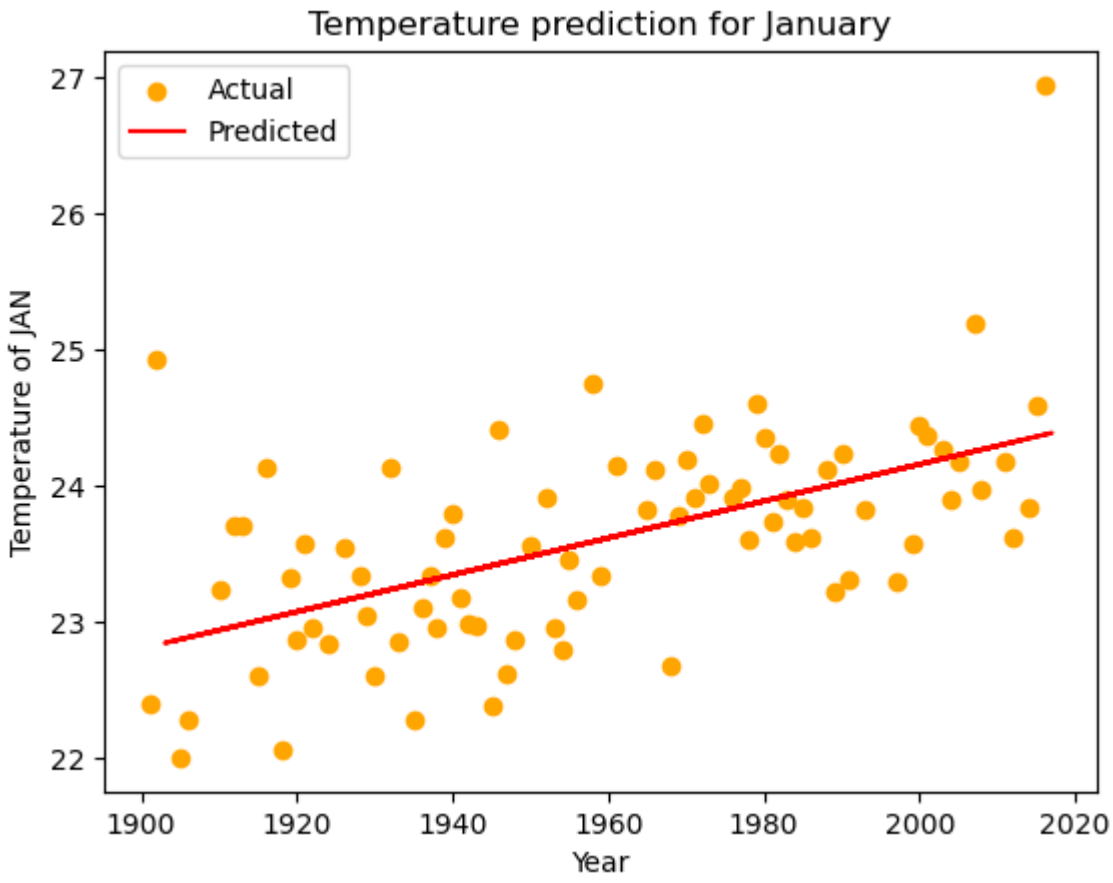
In [96]:
```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(x_train, y_train)
```

Out[96]:
```
▾ LinearRegression

LinearRegression()
```

In [97]:
```
#Prediction of data
y_test_predict = model.predict(x_test)
```

In [98]:
```
plt.scatter(x_train, y_train, color='orange', label='Actual')
plt.plot(x_test, y_test_predict, color='red', label='Predicted')
plt.xlabel('Year')
plt.ylabel('Temperature of JAN')
plt.title('Temperature prediction for January')
plt.legend()
plt.show()
```



Accuracy

In [105…
```
from sklearn import metrics
r2_square = metrics.r2_score(y_test, y_test_predict)
mse = metrics.mean_squared_error(y_test, y_test_predict)
mae = metrics.mean_absolute_error(y_test, y_test_predict)
print("R2 score: {0}\nMSE: {1}\nMAE: {2}".format(r2_square,mse,mae))
```

```
R2 score: 0.2792172351531238
MSE: 0.6080338203121168
MAE: 0.6231302838065338
```

March Month

Training and testing data

In [106…
```
march=data['MAR']
```

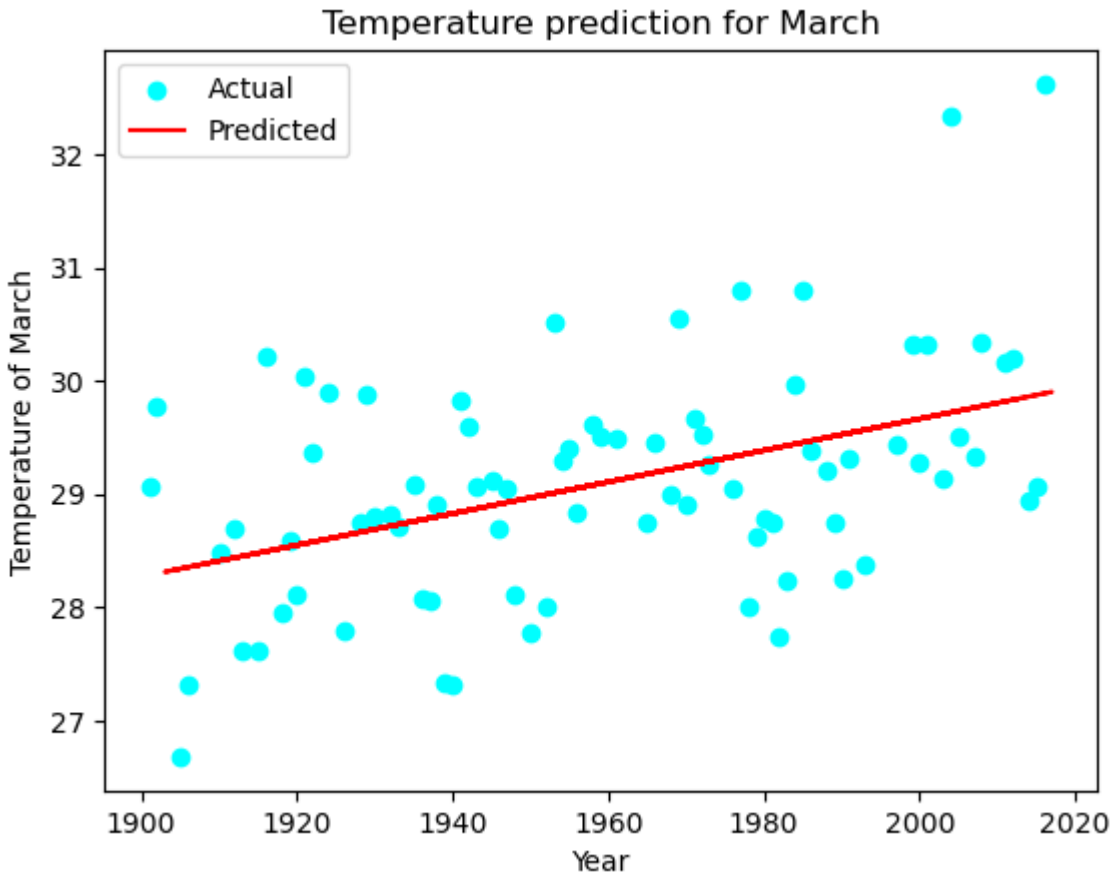
In [107…
```
from sklearn.model_selection import train_test_split
x_train,x_test,march_train, march_test = train_test_split(X,march,test_size=0.3,ran
```

In [108…
```
model.fit(x_train, march_train)
```

Out[108]:
```
LinearRegression()
```

In [109…
```
#Prediction of data
march_test_predict = model.predict(x_test)
```

In [110…
```
plt.scatter(x_train, march_train,color='cyan',label='Actual')
plt.plot(x_test, march_test_predict, color='red',label='Predicted')
plt.xlabel('Year')
plt.ylabel('Temperature of March')
plt.title('Temperature prediction for March')
plt.legend()
plt.show()
```



Seasonal Data

In [111…
```
seasonal = data[['JUN-SEP']]
```

In [112…
```
seasonal.shape
```

Out[112]:
```
(117, 1)
```

In [113…
```
from sklearn.model_selection import train_test_split
x_train,x_test,seasonal_train, seasonal_test = train_test_split(X,seasonal,test_siz
```

In [114…
```
model.fit(x_train, seasonal_train)
```

Out[114]:
```
▾ LinearRegression
LinearRegression()
```

In [115…
```
#Prediction of data
seasonal_test_predict = model.predict(x_test)
```

In [116…
```
plt.scatter(x_train, seasonal_train,color='cyan',label='Actual')
plt.plot(x_test, seasonal_test_predict, color='red',label='Predicted')
plt.xlabel('Year')
plt.ylabel('Temperature of JUN-SEP')
plt.title('Temperature prediction for JUN-SEP')
plt.legend()
plt.show()
```