In [88]:

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from matplotlib import pyplot as plt
```

In [89]:

```python
df=pd.read_csv("temperatures.csv")  #Reading the dataset
print(df)
```

```
     YEAR    JAN    FEB    MAR    APR    MAY    JUN    JUL    AUG    SEP  \
0    1901  22.40  24.14  29.07  31.91  33.41  33.18  31.21  30.39  30.47   
1    1902  24.93  26.58  29.77  31.78  33.73  32.91  30.92  30.73  29.80   
2    1903  23.44  25.03  27.83  31.39  32.91  33.00  31.34  29.98  29.85   
3    1904  22.50  24.73  28.21  32.02  32.64  32.07  30.36  30.09  30.04   
4    1905  22.00  22.83  26.68  30.01  33.32  33.25  31.44  30.68  30.12   
..    ...    ...    ...    ...    ...    ...    ...    ...    ...    ...   
112  2013  24.56  26.59  30.62  32.66  34.46  32.44  31.07  30.76  31.04   
113  2014  23.83  25.97  28.95  32.74  33.77  34.15  31.85  31.32  30.68   
114  2015  24.58  26.89  29.07  31.87  34.09  32.48  31.88  31.52  31.55   
115  2016  26.94  29.72  32.62  35.38  35.72  34.03  31.64  31.79  31.66   
116  2017  26.45  29.46  31.60  34.95  35.84  33.82  31.88  31.72  32.22   

       OCT    NOV    DEC  ANNUAL  JAN-FEB  MAR-MAY  JUN-SEP  OCT-DEC  
0    29.97  27.31  24.49   28.96    23.27    31.46    31.27    27.25  
1    29.12  26.31  24.04   29.22    25.75    31.76    31.09    26.49  
2    29.04  26.08  23.65   28.47    24.24    30.71    30.92    26.26  
3    29.20  26.36  23.63   28.49    23.62    30.95    30.66    26.40  
4    30.67  27.52  23.82   28.30    22.25    30.00    31.33    26.57  
..     ...    ...    ...     ...      ...      ...      ...      ...  
112  30.27  27.83  25.37   29.81    25.58    32.58    31.33    27.83  
113  30.29  28.05  25.08   29.72    24.90    31.82    32.00    27.81  
114  31.04  28.10  25.67   29.90    25.74    31.68    31.87    28.27  
115  31.98  30.11  28.01   31.63    28.33    34.57    32.28    30.03  
116  32.29  29.60  27.18   31.42    27.95    34.13    32.41    29.69  

[117 rows x 18 columns]
```

In [90]:

```python
df.head()
```

Out[90]:

	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	ANNUAL	JAN-FEB
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	28.96	23.27
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	29.22	25.75
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	28.47	24.24
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	28.49	23.62
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	28.30	22.25

In [91]:
```
df.tail()
```

Out[91]:

	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	ANNUAL	JAN-FEB
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	29.81	25.58
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	29.72	24.90
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	29.90	25.74
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	31.63	28.33
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	31.42	27.95

In [92]:
```
df.describe
```

Out[92]:
```
<bound method NDFrame.describe of       YEAR    JAN    FEB    MAR    APR    MAY    JUN  
JUL    AUG    SEP  \
0    1901  22.40  24.14  29.07  31.91  33.41  33.18  31.21  30.39  30.47   
1    1902  24.93  26.58  29.77  31.78  33.73  32.91  30.92  30.73  29.80   
2    1903  23.44  25.03  27.83  31.39  32.91  33.00  31.34  29.98  29.85   
3    1904  22.50  24.73  28.21  32.02  32.64  32.07  30.36  30.09  30.04   
4    1905  22.00  22.83  26.68  30.01  33.32  33.25  31.44  30.68  30.12   
..    ...    ...    ...    ...    ...    ...    ...    ...    ...    ...   
112  2013  24.56  26.59  30.62  32.66  34.46  32.44  31.07  30.76  31.04   
113  2014  23.83  25.97  28.95  32.74  33.77  34.15  31.85  31.32  30.68   
114  2015  24.58  26.89  29.07  31.87  34.09  32.48  31.88  31.52  31.55   
115  2016  26.94  29.72  32.62  35.38  35.72  34.03  31.64  31.79  31.66   
116  2017  26.45  29.46  31.60  34.95  35.84  33.82  31.88  31.72  32.22   

       OCT    NOV    DEC  ANNUAL  JAN-FEB  MAR-MAY  JUN-SEP  OCT-DEC  
0    29.97  27.31  24.49   28.96    23.27    31.46    31.27    27.25  
1    29.12  26.31  24.04   29.22    25.75    31.76    31.09    26.49  
2    29.04  26.08  23.65   28.47    24.24    30.71    30.92    26.26  
3    29.20  26.36  23.63   28.49    23.62    30.95    30.66    26.40  
4    30.67  27.52  23.82   28.30    22.25    30.00    31.33    26.57  
..     ...    ...    ...     ...      ...      ...      ...      ...  
112  30.27  27.83  25.37   29.81    25.58    32.58    31.33    27.83  
113  30.29  28.05  25.08   29.72    24.90    31.82    32.00    27.81  
114  31.04  28.10  25.67   29.90    25.74    31.68    31.87    28.27  
115  31.98  30.11  28.01   31.63    28.33    34.57    32.28    30.03  
116  32.29  29.60  27.18   31.42    27.95    34.13    32.41    29.69  

[117 rows x 18 columns]>
```

In [93]:
```
df.columns
```

Out[93]:
```
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 [94]:
```
df.dtypes
```

Out[94]:
```
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 [95]:
```
df.shape  # Display shape of data
```

Out[95]:
```
(117, 18)
```

In [96]:
```
df.isnull
```

Out[96]:
```
<bound method DataFrame.isnull of      YEAR    JAN    FEB    MAR    APR    MAY    JUN    
JUL    AUG    SEP  \
0    1901  22.40  24.14  29.07  31.91  33.41  33.18  31.21  30.39  30.47   
1    1902  24.93  26.58  29.77  31.78  33.73  32.91  30.92  30.73  29.80   
2    1903  23.44  25.03  27.83  31.39  32.91  33.00  31.34  29.98  29.85   
3    1904  22.50  24.73  28.21  32.02  32.64  32.07  30.36  30.09  30.04   
4    1905  22.00  22.83  26.68  30.01  33.32  33.25  31.44  30.68  30.12   
..    ...    ...    ...    ...    ...    ...    ...    ...    ...    ...   
112  2013  24.56  26.59  30.62  32.66  34.46  32.44  31.07  30.76  31.04   
113  2014  23.83  25.97  28.95  32.74  33.77  34.15  31.85  31.32  30.68   
114  2015  24.58  26.89  29.07  31.87  34.09  32.48  31.88  31.52  31.55   
115  2016  26.94  29.72  32.62  35.38  35.72  34.03  31.64  31.79  31.66   
116  2017  26.45  29.46  31.60  34.95  35.84  33.82  31.88  31.72  32.22   

       OCT    NOV    DEC  ANNUAL  JAN-FEB  MAR-MAY  JUN-SEP  OCT-DEC  
0    29.97  27.31  24.49   28.96    23.27    31.46    31.27    27.25  
1    29.12  26.31  24.04   29.22    25.75    31.76    31.09    26.49  
2    29.04  26.08  23.65   28.47    24.24    30.71    30.92    26.26  
3    29.20  26.36  23.63   28.49    23.62    30.95    30.66    26.40  
4    30.67  27.52  23.82   28.30    22.25    30.00    31.33    26.57  
..     ...    ...    ...     ...      ...      ...      ...      ...  
112  30.27  27.83  25.37   29.81    25.58    32.58    31.33    27.83  
113  30.29  28.05  25.08   29.72    24.90    31.82    32.00    27.81  
114  31.04  28.10  25.67   29.90    25.74    31.68    31.87    28.27  
115  31.98  30.11  28.01   31.63    28.33    34.57    32.28    30.03  
116  32.29  29.60  27.18   31.42    27.95    34.13    32.41    29.69  

[117 rows x 18 columns]>
```

Temperature Prediction Of JAN Month

In [97]:

```
X=df[['YEAR']]
Y=df[['JAN']]
print(X)
print(Y)
```

```
     YEAR
0    1901
1    1902
2    1903
3    1904
4    1905
..    ...
112  2013
113  2014
114  2015
115  2016
116  2017

[117 rows x 1 columns]
       JAN
0    22.40
1    24.93
2    23.44
3    22.50
4    22.00
..     ...
112  24.56
113  23.83
114  24.58
115  26.94
116  26.45

[117 rows x 1 columns]
```

In [98]:
```
df.dropna  #Filter out missing data
```

Out[98]:
```
<bound method DataFrame.dropna of      YEAR    JAN    FEB    MAR    APR    MAY    JUN
JUL    AUG    SEP  \
0    1901  22.40  24.14  29.07  31.91  33.41  33.18  31.21  30.39  30.47
1    1902  24.93  26.58  29.77  31.78  33.73  32.91  30.92  30.73  29.80
2    1903  23.44  25.03  27.83  31.39  32.91  33.00  31.34  29.98  29.85
3    1904  22.50  24.73  28.21  32.02  32.64  32.07  30.36  30.09  30.04
4    1905  22.00  22.83  26.68  30.01  33.32  33.25  31.44  30.68  30.12
..    ...    ...    ...    ...    ...    ...    ...    ...    ...    ...
112  2013  24.56  26.59  30.62  32.66  34.46  32.44  31.07  30.76  31.04
113  2014  23.83  25.97  28.95  32.74  33.77  34.15  31.85  31.32  30.68
114  2015  24.58  26.89  29.07  31.87  34.09  32.48  31.88  31.52  31.55
115  2016  26.94  29.72  32.62  35.38  35.72  34.03  31.64  31.79  31.66
116  2017  26.45  29.46  31.60  34.95  35.84  33.82  31.88  31.72  32.22

       OCT    NOV    DEC  ANNUAL  JAN-FEB  MAR-MAY  JUN-SEP  OCT-DEC
0    29.97  27.31  24.49   28.96    23.27    31.46    31.27    27.25
1    29.12  26.31  24.04   29.22    25.75    31.76    31.09    26.49
2    29.04  26.08  23.65   28.47    24.24    30.71    30.92    26.26
3    29.20  26.36  23.63   28.49    23.62    30.95    30.66    26.40
4    30.67  27.52  23.82   28.30    22.25    30.00    31.33    26.57
..     ...    ...    ...     ...      ...      ...      ...      ...
112  30.27  27.83  25.37   29.81    25.58    32.58    31.33    27.83
113  30.29  28.05  25.08   29.72    24.90    31.82    32.00    27.81
114  31.04  28.10  25.67   29.90    25.74    31.68    31.87    28.27
115  31.98  30.11  28.01   31.63    28.33    34.57    32.28    30.03
116  32.29  29.60  27.18   31.42    27.95    34.13    32.41    29.69

[117 rows x 18 columns]>
```

In [99]:
```
from sklearn.model_selection import train_test_split
Xtrain,Xtest,Ytrain,Ytest=train_test_split(X,Y,test_size=0.2)  # Splitting the data into tr
```

In [100]:
```
print(len(Xtrain))
print(len(Xtest))
```

```
93
24
```

In [101]:
```
#Training our model
from sklearn import linear_model,metrics
regr1=linear_model.LinearRegression()
regr2=linear_model.LinearRegression()
regr3=linear_model.LinearRegression()
regr4=linear_model.LinearRegression()
regr5=linear_model.LinearRegression()
```

In [102]:
```
#Fit the model over the training dataset
model=regr1.fit(Xtrain,Ytrain)
print(model.intercept_)
print(model.coef_)
```

```
[-4.15090705]
[[0.01422303]]
```

```
In [103]: y_predict=model.predict(Xtest)
          print("predictions",y_predict)
```

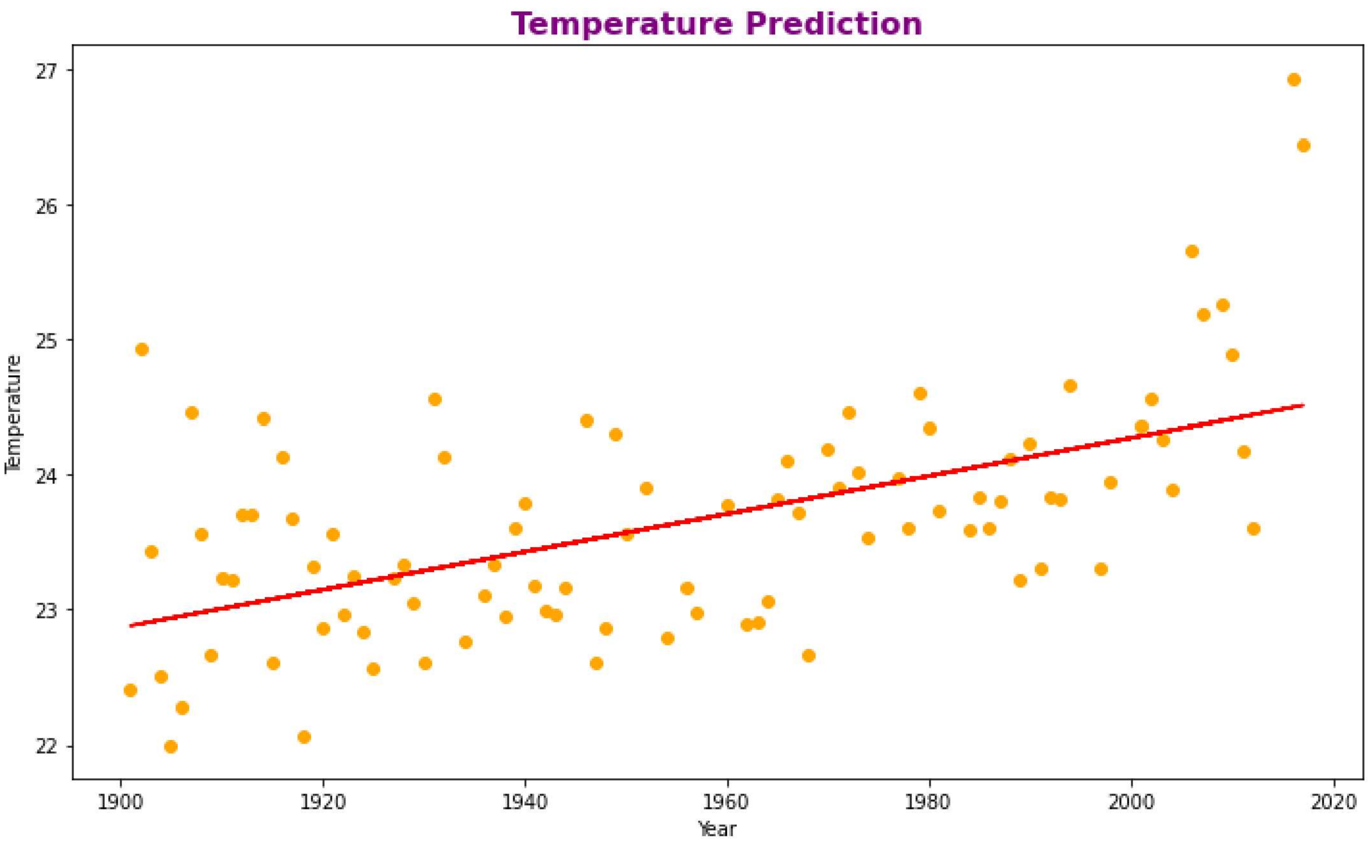
```
predictions [[24.50850029]
 [24.2382627 ]
 [23.69778753]
 [24.22403967]
 [23.34221176]
 [24.48005423]
 [24.40893908]
 [24.49427726]
 [24.03914027]
 [23.71201056]
 [23.37065782]
 [23.59822631]
 [23.65511844]
 [24.0533633 ]
 [23.74045662]
 [24.2809318 ]
 [24.36626998]
 [24.29515483]
 [23.24265054]
 [23.85424087]
 [23.95380209]
 [23.51288813]
 [23.93957905]
 [23.62667237]]
```

In [104]:

```
plt.figure(figsize=(12,7))
plt.scatter(Xtrain,Ytrain,color='orange')
plt.plot(Xtrain,regr.predict(Xtrain),color='red')
plt.xlabel("Year")
plt.ylabel("Temperature")
plt.title("Temperature Prediction",fontweight="bold",fontsize=16,color='purple')
```

Out[104]:

```
Text(0.5, 1.0, 'Temperature Prediction')
```



In [105]:

```
# Evaluation Metrics For Regression
from sklearn.metrics import mean_squared_error
print("MSE : ",mean_squared_error(Ytest,y_predict))
```

```
MSE :  0.3380509983787099
```

In [106]:

```
import math
mse=mean_squared_error(Ytest,y_predict)
rmse=math.sqrt(mse)
rmse
print("RMSE : ",np.sqrt(mean_squared_error(Ytest,y_predict)))
```

```
RMSE :  0.5814215324346956
```

In [107]:

```
print("RMSLE :",np.log(np.sqrt(mean_squared_error(Ytest,y_predict))))
```

```
RMSLE : -0.5422792560517659
```

```
In [108]: from sklearn.metrics import r2_score
          r2 = r2_score(Ytest,y_predict)
          print("R Squared : ",r2)
```

```
R Squared :  0.34688405604778705
```

Temperature Prediction Of FEB Month

```
In [109]: X=df[['YEAR']]
          Y2=df[['FEB']]
          print(X)
          print(Y2)
```

```
     YEAR
0    1901
1    1902
2    1903
3    1904
4    1905
..    ...
112  2013
113  2014
114  2015
115  2016
116  2017

[117 rows x 1 columns]
       FEB
0    24.14
1    26.58
2    25.03
3    24.73
4    22.83
..     ...
112  26.59
113  25.97
114  26.89
115  29.72
116  29.46

[117 rows x 1 columns]
```

```
In [110]: model=regr2.fit(X,Y2)
          print(model.intercept_)
          print(model.coef_)
```

```
[-17.40757239]
[[0.02195275]]
```

In [111]:

```
y_predict_Feb=model.predict(Xtest)
print("predictions",y_predict_Feb)
```

```
predictions [[26.8272172 ]
 [26.41011497]
 [25.5759105 ]
 [26.38816222]
 [25.02709177]
 [26.7833117 ]
 [26.67354796]
 [26.80526445]
 [26.10277648]
 [25.59786325]
 [25.07099727]
 [25.42224125]
 [25.51005225]
 [26.12472923]
 [25.64176875]
 [26.47597322]
 [26.60768971]
 [26.49792596]
 [24.87342252]
 [25.81739074]
 [25.97105998]
 [25.29052476]
 [25.94910723]
 [25.46614675]]
```

Temperature Prediction Of MARCH Month

In [112]:

```
X=df[['YEAR']]
Y3=df[['MAR']]
print(X)
print(Y3)
```

```
     YEAR
0    1901
1    1902
2    1903
3    1904
4    1905
..    ...
112  2013
113  2014
114  2015
115  2016
116  2017

[117 rows x 1 columns]
       MAR
0    29.07
1    29.77
2    27.83
3    28.21
4    26.68
..     ...
112  30.62
113  28.95
114  29.07
115  32.62
116  31.60

[117 rows x 1 columns]
```

In [113]:

```
model=regr3.fit(X,Y3)
print(model.intercept_)
print(model.coef_)
```

```
[-5.09354548]
[[0.01744744]]
```

In [114]:

```
y_predict_March=model.predict(Xtest)
print("predictions",y_predict_March)
```

```
predictions [[30.06303936]
 [29.73153806]
 [29.06853547]
 [29.71409063]
 [28.63234955]
 [30.02814448]
 [29.9409073 ]
 [30.04559192]
 [29.48727395]
 [29.08598291]
 [28.66724443]
 [28.94640341]
 [29.01619316]
 [29.50472139]
 [29.12087778]
 [29.78388037]
 [29.88856499]
 [29.80132781]
 [28.5102175 ]
 [29.26045727]
 [29.38258933]
 [28.84171879]
 [29.36514189]
 [28.98129829]]
```

Temperature Prediction Of APRIL Month

```
In [115]: X=df[['YEAR']]
          Y4=df[['APR']]
          print(X)
          print(Y4)
```

```
     YEAR
0    1901
1    1902
2    1903
3    1904
4    1905
..    ...
112  2013
113  2014
114  2015
115  2016
116  2017

[117 rows x 1 columns]
       APR
0    31.91
1    31.78
2    31.39
3    32.02
4    30.01
..     ...
112  32.66
113  32.74
114  31.87
115  35.38
116  34.95

[117 rows x 1 columns]
```

```
In [116]: model=regr4.fit(X,Y4)
          print(model.intercept_)
          print(model.coef_)
```

```
[4.20092683]
[[0.01417809]]
```

In [117]:

```
y_predict_April=model.predict(Xtest)
print("predictions",y_predict_April)
```

```
predictions [[32.7697852 ]
 [32.50040142]
 [31.96163387]
 [32.48622333]
 [31.60718154]
 [32.74142901]
 [32.67053855]
 [32.75560711]
 [32.30190812]
 [31.97581197]
 [31.63553772]
 [31.86238722]
 [31.91909959]
 [32.31608621]
 [32.00416815]
 [32.54293571]
 [32.62800427]
 [32.5571138 ]
 [31.50793488]
 [32.1175929 ]
 [32.21683956]
 [31.77731866]
 [32.20266146]
 [31.8907434 ]]
```

Temperature Prediction Of MAY Month

In [118]:

```
X=df[['YEAR']]
Y5=df[['MAY']]
print(X)
print(Y5)
```

```
     YEAR
0    1901
1    1902
2    1903
3    1904
4    1905
..    ...
112  2013
113  2014
114  2015
115  2016
116  2017

[117 rows x 1 columns]
       MAY
0    33.41
1    33.73
2    32.91
3    32.64
4    33.32
..     ...
112  34.46
113  33.77
114  34.09
115  35.72
116  35.84

[117 rows x 1 columns]
```

In [119]:

```
model=regr5.fit(X,Y5)
print(model.intercept_)
print(model.coef_)
```

```
[16.49828705]
[[0.0087121]]
```

In [120]:

```
y_predict_May=model.predict(Xtest)
print("predictions",y_predict_May)
```

```
predictions [[34.05317698]
 [33.887647  ]
 [33.55658704]
 [33.8789349 ]
 [33.33878444]
 [34.03575277]
 [33.99219225]
 [34.04446488]
 [33.76567754]
 [33.56529915]
 [33.35620864]
 [33.49560231]
 [33.53045073]
 [33.77438965]
 [33.58272335]
 [33.91378331]
 [33.96605594]
 [33.92249542]
 [33.27779971]
 [33.65242019]
 [33.71340492]
 [33.44332969]
 [33.70469281]
 [33.51302652]]
```