PROBLEM STATEMENT:- TO PREDICT THE RAIN FALL BASED ON VARIOUS FEATURES OF THE DATASET

IMPORTING THE ESSENTIAL LIBRARIES:-

In [1]:

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn import preprocessing,svm
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
df=pd.read_csv(r"C:\Users\smb06\OneDrive\Desktop\district wise rainfall normal.csv")
df
```

Out[2]:

	STATE_UT_NAME	DISTRICT	JAN	FEB	MAR	APR	MAY	JUN	JUL	A
0	ANDAMAN And NICOBAR ISLANDS	NICOBAR	107.3	57.9	65.2	117.0	358.5	295.5	285.0	27
1	ANDAMAN And NICOBAR ISLANDS	SOUTH ANDAMAN	43.7	26.0	18.6	90.5	374.4	457.2	421.3	42
2	ANDAMAN And NICOBAR ISLANDS	N & M ANDAMAN	32.7	15.9	8.6	53.4	343.6	503.3	465.4	46
3	ARUNACHAL PRADESH	LOHIT	42.2	80.8	176.4	358.5	306.4	447.0	660.1	42
4	ARUNACHAL PRADESH	EAST SIANG	33.3	79.5	105.9	216.5	323.0	738.3	990.9	71
...	...	...	...	...	...	...	...	...	...	
636	KERALA	IDUKKI	13.4	22.1	43.6	150.4	232.6	651.6	788.9	52
637	KERALA	KASARGOD	2.3	1.0	8.4	46.9	217.6	999.6	1108.5	63
638	KERALA	PATHANAMTHITTA	19.8	45.2	73.9	184.9	294.7	556.9	539.9	35
639	KERALA	WAYANAD	4.8	8.3	17.5	83.3	174.6	698.1	1110.4	59
640	LAKSHADWEEP	LAKSHADWEEP	20.8	14.7	11.8	48.9	171.7	330.2	287.7	21

641 rows × 19 columns

DATA PROCESSING

In [3]:

```
df.head()
```

Out[3]:

	STATE_UT_NAME	DISTRICT	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP
0	ANDAMAN And NICOBAR ISLANDS	NICOBAR	107.3	57.9	65.2	117.0	358.5	295.5	285.0	271.9	354.8
1	ANDAMAN And NICOBAR ISLANDS	SOUTH ANDAMAN	43.7	26.0	18.6	90.5	374.4	457.2	421.3	423.1	455.6
2	ANDAMAN And NICOBAR ISLANDS	N & M ANDAMAN	32.7	15.9	8.6	53.4	343.6	503.3	465.4	460.9	454.8
3	ARUNACHAL PRADESH	LOHIT	42.2	80.8	176.4	358.5	306.4	447.0	660.1	427.8	313.6
4	ARUNACHAL PRADESH	EAST SIANG	33.3	79.5	105.9	216.5	323.0	738.3	990.9	711.2	568.0

In [4]:

```
df.tail()
```

Out[4]:

	STATE_UT_NAME	DISTRICT	JAN	FEB	MAR	APR	MAY	JUN	JUL	AU
636	KERALA	IDUKKI	13.4	22.1	43.6	150.4	232.6	651.6	788.9	527
637	KERALA	KASARGOD	2.3	1.0	8.4	46.9	217.6	999.6	1108.5	636
638	KERALA	PATHANAMTHITTA	19.8	45.2	73.9	184.9	294.7	556.9	539.9	352
639	KERALA	WAYANAD	4.8	8.3	17.5	83.3	174.6	698.1	1110.4	592
640	LAKSHADWEEP	LAKSHADWEEP	20.8	14.7	11.8	48.9	171.7	330.2	287.7	217

In [5]:

```
df.isnull().any()
```

Out[5]:

```
STATE_UT_NAME    False
DISTRICT         False
JAN              False
FEB              False
MAR              False
APR              False
MAY              False
JUN              False
JUL              False
AUG              False
SEP              False
OCT              False
NOV              False
DEC              False
ANNUAL           False
Jan-Feb          False
Mar-May          False
Jun-Sep          False
Oct-Dec          False
dtype: bool
```

In [6]:

```
df.isnull().sum()
```

Out[6]:

```
STATE_UT_NAME    0
DISTRICT         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 [7]:

```
df.describe()
```

Out[7]:

	JAN	FEB	MAR	APR	MAY	JUN	JUL
count	641.000000	641.000000	641.000000	641.000000	641.000000	641.000000	641.000000
mean	18.355070	20.984399	30.034789	45.543214	81.535101	196.007332	326.033697
std	21.082806	27.729596	45.451082	71.556279	111.960390	196.556284	221.364643
min	0.000000	0.000000	0.000000	0.000000	0.900000	3.800000	11.600000
25%	6.900000	7.000000	7.000000	5.000000	12.100000	68.800000	206.400000
50%	13.300000	12.300000	12.700000	15.100000	33.900000	131.900000	293.700000
75%	19.200000	24.100000	33.200000	48.300000	91.900000	226.600000	374.800000
max	144.500000	229.600000	367.900000	554.400000	733.700000	1476.200000	1820.900000

In [8]:

```
df.info
```

Out[8]:

```
<bound method DataFrame.info of                        STATE_UT_NAME        DI
STRICT    JAN    FEB    MAR    APR
0    ANDAMAN And NICOBAR ISLANDS         NICOBAR  107.3  57.9   65.2  117.
0  \
1    ANDAMAN And NICOBAR ISLANDS   SOUTH ANDAMAN   43.7  26.0   18.6   90.
5
2    ANDAMAN And NICOBAR ISLANDS   N & M ANDAMAN   32.7  15.9    8.6   53.
4
3              ARUNACHAL PRADESH           LOHIT   42.2  80.8  176.4  358.
5
4              ARUNACHAL PRADESH      EAST SIANG   33.3  79.5  105.9  216.
5
..                           ...             ...    ...   ...    ...   ...
...
636                       KERALA          IDUKKI   13.4  22.1   43.6  150.
4
637                       KERALA        KASARGOD    2.3   1.0    8.4   46.
9
638                       KERALA  PATHANAMTHITTA   19.8  45.2   73.9  184.
9
639                       KERALA         WAYANAD    4.8   8.3   17.5   83.
3
640                  LAKSHADWEEP     LAKSHADWEEP   20.8  14.7   11.8   48.
9

       MAY    JUN     JUL    AUG    SEP    OCT    NOV    DEC  ANNUAL  Jan-
Feb
0    358.5  295.5   285.0  271.9  354.8  326.0  315.2  250.9  2805.2     16
5.2  \
1    374.4  457.2   421.3  423.1  455.6  301.2  275.8  128.3  3015.7      6
9.7
2    343.6  503.3   465.4  460.9  454.8  276.1  198.6  100.0  2913.3      4
8.6
3    306.4  447.0   660.1  427.8  313.6  167.1   34.1   29.8  3043.8     12
3.0
4    323.0  738.3   990.9  711.2  568.0  206.9   29.5   31.7  4034.7     11
2.8
..     ...    ...     ...    ...    ...    ...    ...    ...     ...    ...
...
636  232.6  651.6   788.9  527.3  308.4  343.2  172.9   48.1  3302.5      3
5.5
637  317.6  999.6  1108.5  636.3  263.1  234.9   84.6   18.4  3621.6
3.3
638  294.7  556.9   539.9  352.7  266.2  359.4  213.5   51.3  2958.4      6
5.0
639  174.6  698.1  1110.4  592.9  230.7  213.1   93.6   25.8  3253.1      1
3.1
640  171.7  330.2   287.7  217.5  163.1  157.1  117.7   58.8  1600.0      3
5.5

     Mar-May  Jun-Sep  Oct-Dec
0      540.7   1207.2    892.1
1      483.5   1757.2    705.3
2      405.6   1884.4    574.7
3      841.3   1848.5    231.0
4      645.4   3008.4    268.1
..       ...      ...      ...
636    426.6   2276.2    564.2
637    272.9   3007.5    337.9
638    553.5   1715.7    624.2
639    275.4   2632.1    332.5
```

In [9]:

```
df.columns
```

Out[9]:

```
Index(['STATE_UT_NAME', 'DISTRICT', 'JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JU
N',
       'JUL', 'AUG', 'SEP', 'OCT', 'NOV', 'DEC', 'ANNUAL', 'Jan-Feb',
       'Mar-May', 'Jun-Sep', 'Oct-Dec'],
      dtype='object')
```

In [10]:

```
df.shape
```

Out[10]:

```
(641, 19)
```

In [11]:

```
df['ANNUAL'].value_counts()
```

Out[11]:

```
ANNUAL
747.1     9
2080.0    4
1336.5    3
1824.8    3
2814.4    3
         ..
1037.6    1
907.2     1
944.5     1
1003.3    1
3233.1    1
Name: count, Length: 591, dtype: int64
```

```
640     232.4     998.5     333.6
```

In [12]:

```
[641 rows x 19 columns]>
df['Jan-Feb'].value_counts()
```

Out[12]:

```
Jan-Feb
32.7     9
18.2     5
21.4     5
0.8      5
17.5     5
        ..
107.7    1
87.0     1
101.0    1
135.2    1
65.0     1
Name: count, Length: 399, dtype: int64
```

In [13]:

```
df['Mar-May'].value_counts()
```

Out[13]:

```
Mar-May
43.5     9
27.9     5
36.6     4
468.6    4
40.4     3
        ..
16.3     1
23.3     1
49.6     1
20.5     1
232.4    1
Name: count, Length: 511, dtype: int64
```

In [14]:

```
df['Jun-Sep'].value_counts()
```

Out[14]:

```
Jun-Sep
636.2     9
1386.1    4
385.0     3
1122.3    3
1308.0    3
         ..
916.9     1
923.5     1
790.3     1
840.7     1
998.5     1
Name: count, Length: 592, dtype: int64
```

In [15]:

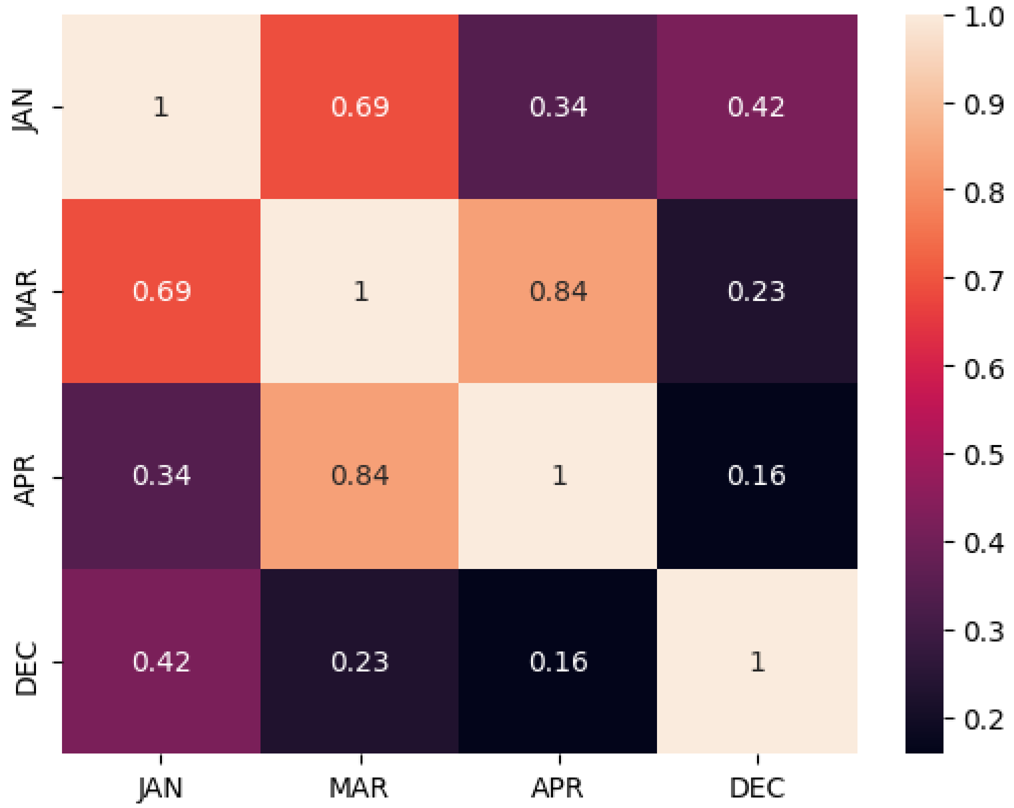
```
df['Oct-Dec'].value_counts()
```

Out[15]:

```
Oct-Dec
34.7     9
174.8    4
49.6     3
27.7     3
183.7    3
        ..
82.8     1
55.2     1
65.6     1
54.0     1
333.6    1
Name: count, Length: 524, dtype: int64
```

In [16]:

```
df=df[['JAN','MAR','APR','DEC']]
sns.heatmap(df.corr(),annot=True)
plt.show()
```



In [17]:

```
df.columns
```

Out[17]:

```
Index(['JAN', 'MAR', 'APR', 'DEC'], dtype='object')
```

FEATURE SCALING

To Split the data into train and test data

In [18]:

```
x=df[["DEC"]]
y=df["JAN"]
```

In [19]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.33,random_state=42)
```

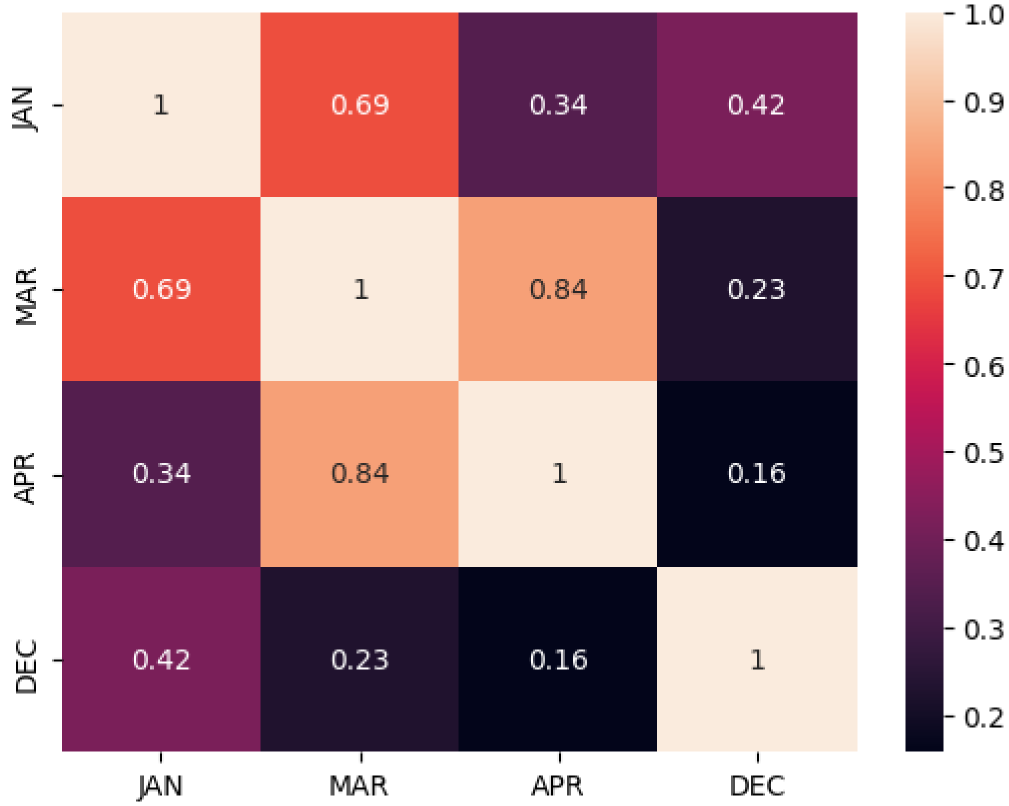
In [20]:

```
x=df[["DEC"]]
y=df["JAN"]
```

DATA VISUALIZATION

In [21]:

```
df=df[['JAN','MAR','APR','DEC']]
sns.heatmap(df.corr(),annot=True)
plt.show()
```

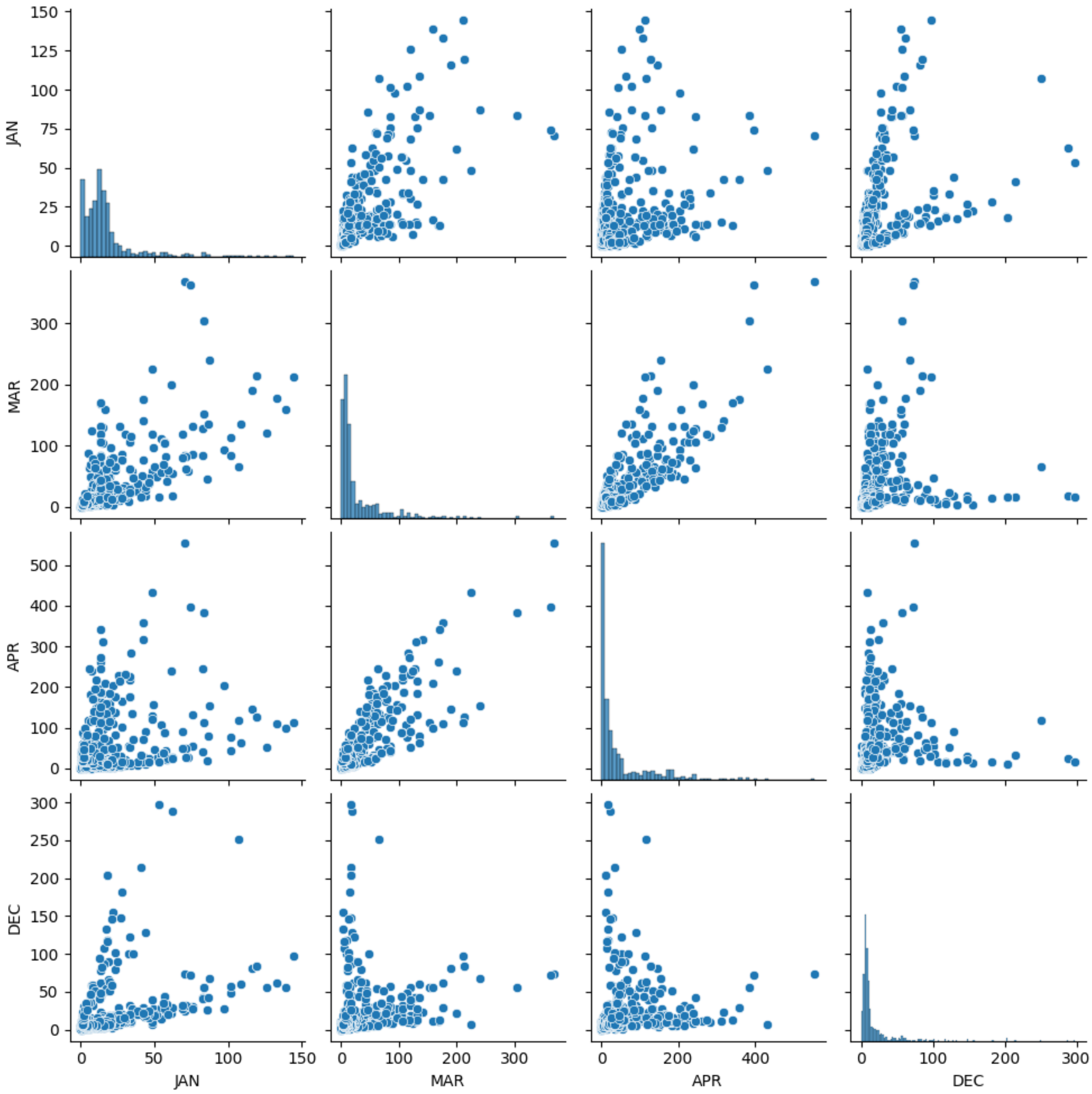


In [22]:

```
sns.pairplot(df)
```

Out[22]:

```
<seaborn.axisgrid.PairGrid at 0x2280b550b50>
```



DATA MODELLING

LINEAR REGRESSION

In [23]:

```
from sklearn.linear_model import LinearRegression
reg=LinearRegression()
reg.fit(X_train,y_train)
print(reg.intercept_)
coeff_=pd.DataFrame(reg.coef_,x.columns,columns=['coefficient'])
coeff_
```

```
12.782665729383353
```

Out[23]:

	coefficient
DEC	0.280279

In [24]:

```
score=reg.score(X_test,y_test)
print(score)
```

```
0.12294534437183668
```

In [25]:

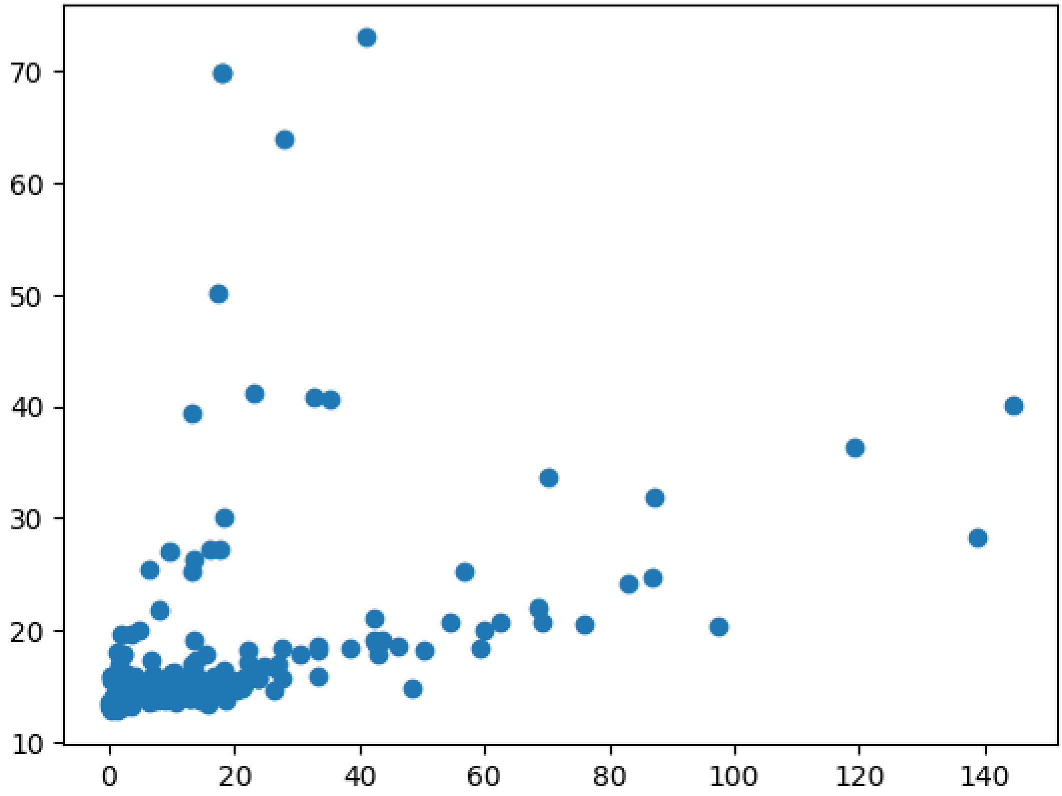
```
predictions=reg.predict(X_test)
```

In [26]:

```
plt.scatter(y_test,predictions)
```

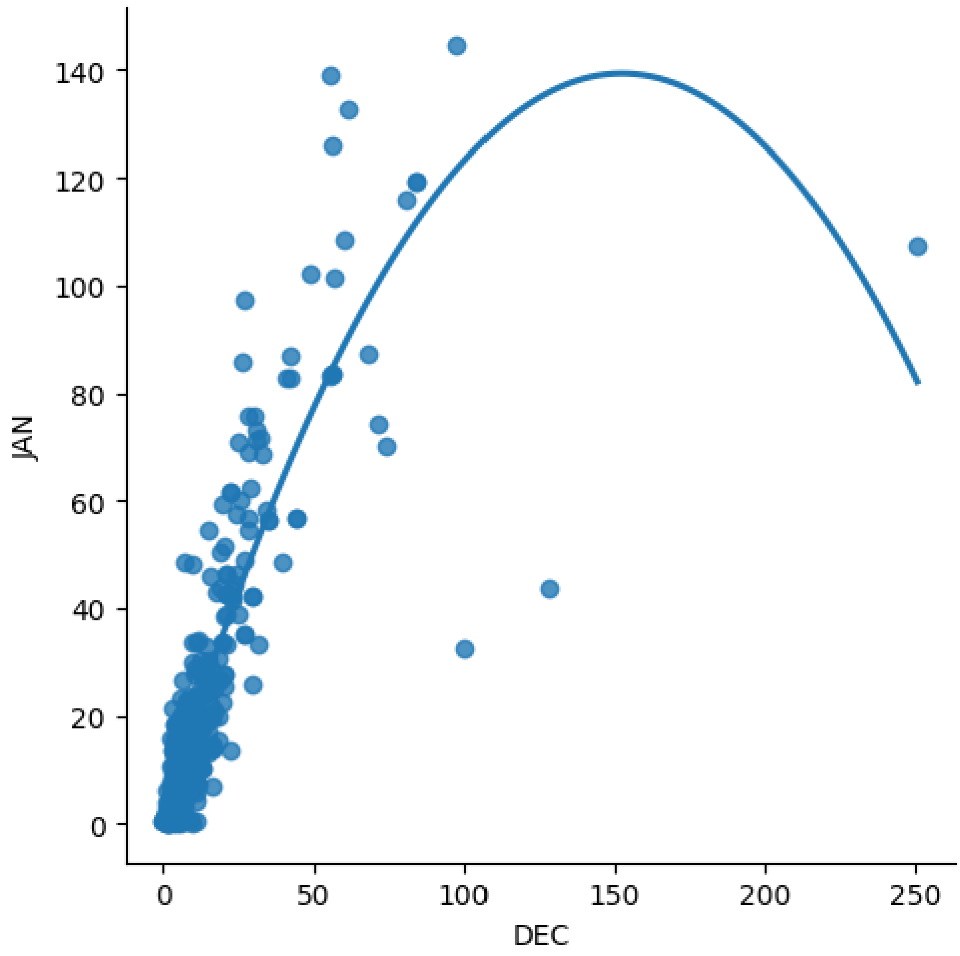
Out[26]:

```
<matplotlib.collections.PathCollection at 0x2280e2684d0>
```



In [27]:

```
df500=df[:][:500]
sns.lmplot(x="DEC",y="JAN",order=2,ci=None,data=df500)
plt.show()
```



In [28]:

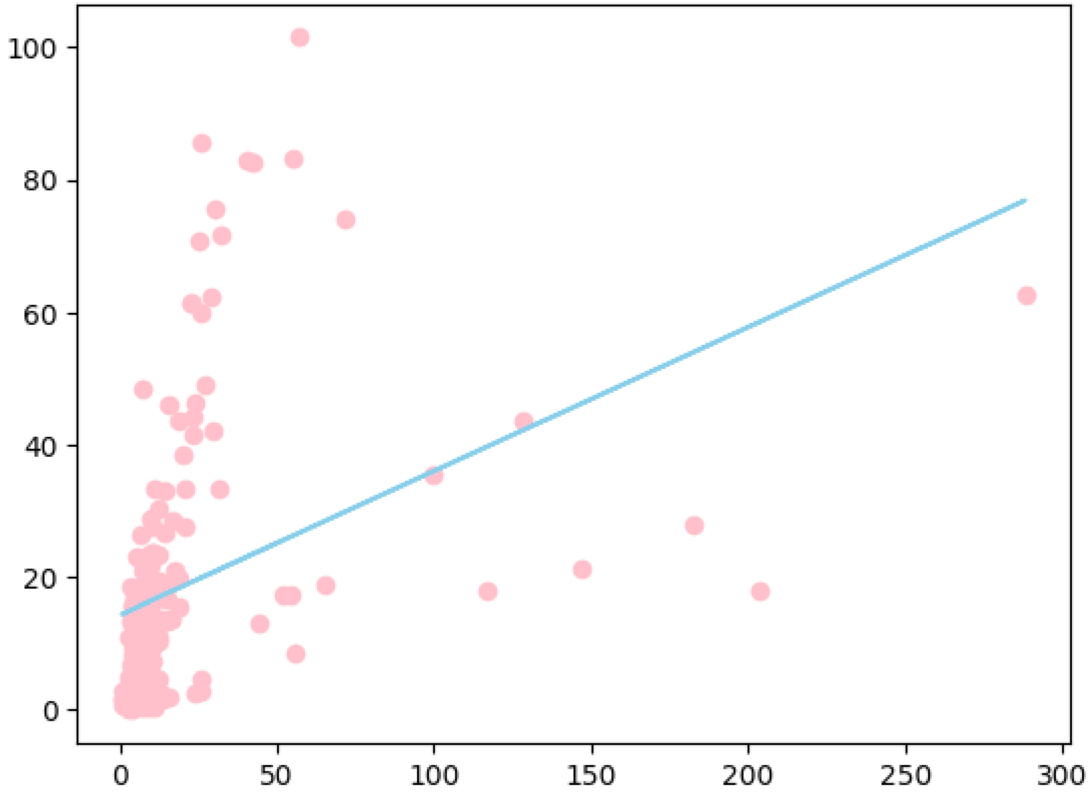
```
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.33)
reg.fit(X_train,y_train)
reg.fit(X_test,y_test)
```

Out[28]:

```
▾ LinearRegression
LinearRegression()
```

In [29]:

```
y_pred=reg.predict(X_test)
plt.scatter(X_test,y_test,color='pink')
plt.plot(X_test,y_pred,color='skyblue')
plt.show()
```



In [30]:

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
model=LinearRegression()
model.fit(X_train,y_train)
y_pred=model.predict(X_test)
r2=r2_score(y_test,y_pred)
print("R2 Score:",r2)
```

```
R2 Score: 0.12658466121292555
```

RIDGE MODEL

In [31]:

```
from sklearn.linear_model import Lasso,Ridge
from sklearn.preprocessing import StandardScaler
```

In [32]:

```
features= df.columns[0:5]
target= df.columns[-4]
```

In [33]:

```
x= df[features].values
y= df[target].values
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=17)
```

In [34]:

```
ridgeReg=Ridge(alpha=10)
ridgeReg.fit(x_train,y_train)
train_score_ridge=ridgeReg.score(x_train,y_train)
test_score_ridge=ridgeReg.score(x_test,y_test)
```

In [35]:

```
print("\n Ridge Model:\n")
print("the train score for ridge model is{}".format(train_score_ridge))
print("the test score for ridge model is{}".format(test_score_ridge))
```

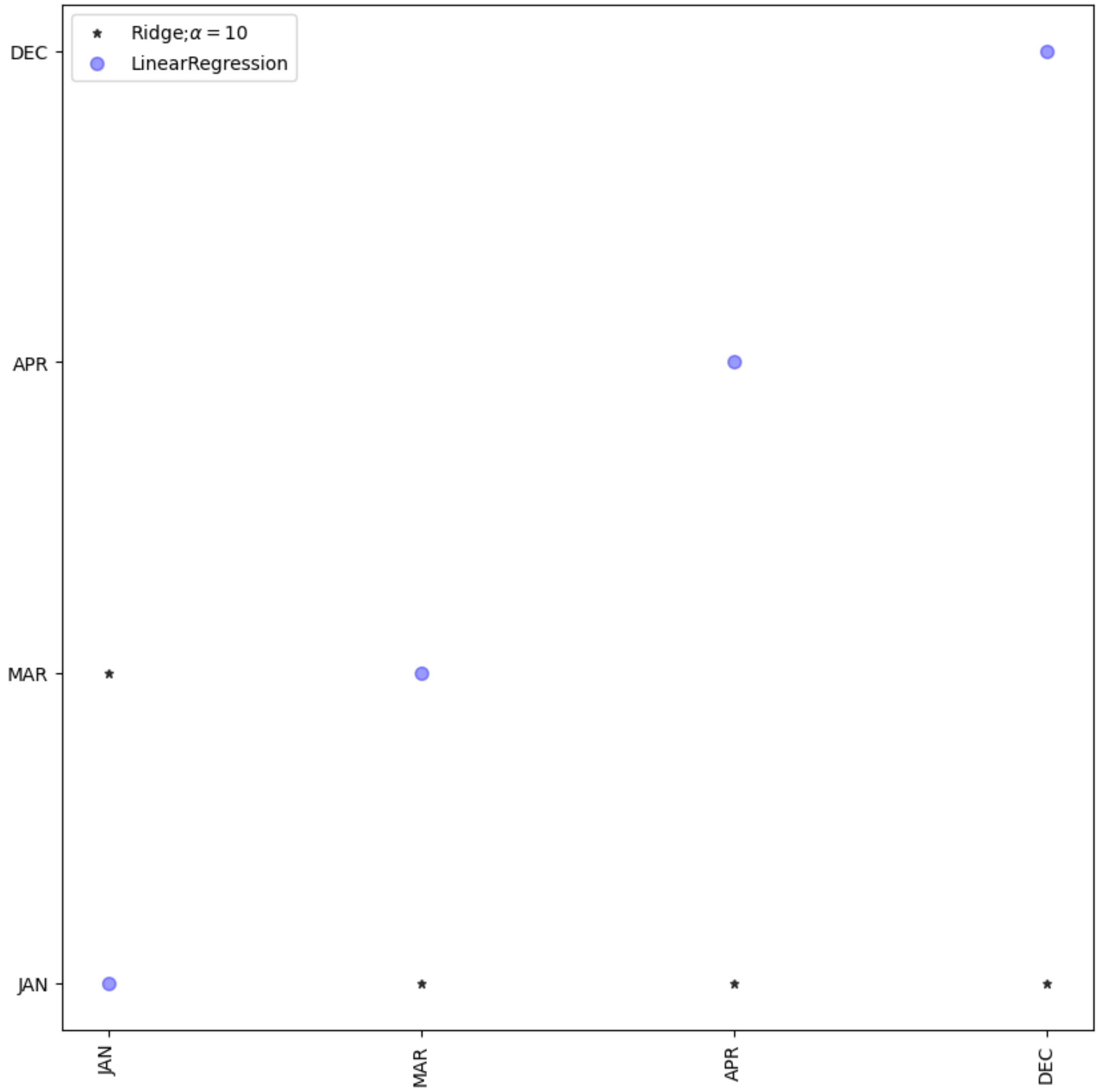
```
 Ridge Model:

the train score for ridge model is0.9999999872352109
the test score for ridge model is0.9999999899454203
```

In [36]:

```
lr=LinearRegression()
```

In [37]:

```
plt.figure(figsize=(10,10))
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker="*",markersize=5,colo
plt.plot(features,alpha=0.4,linestyle='none',marker="o",markersize=7,color='BLUE',label=
plt.xticks(rotation=90)
plt.legend()
plt.show()
```



LASSO MODEL

In [38]:

```
print("\n Lasso Model:\n")
lasso=Lasso(alpha=10)
lasso.fit(x_train,y_train)
train_score_ls=lasso.score(x_train,y_train)
test_score_ls=lasso.score(x_test,y_test)
print("The train score for ls model is {}".format(train_score_ls))
print("The test score for ls model is{}".format(test_score_ls))
```

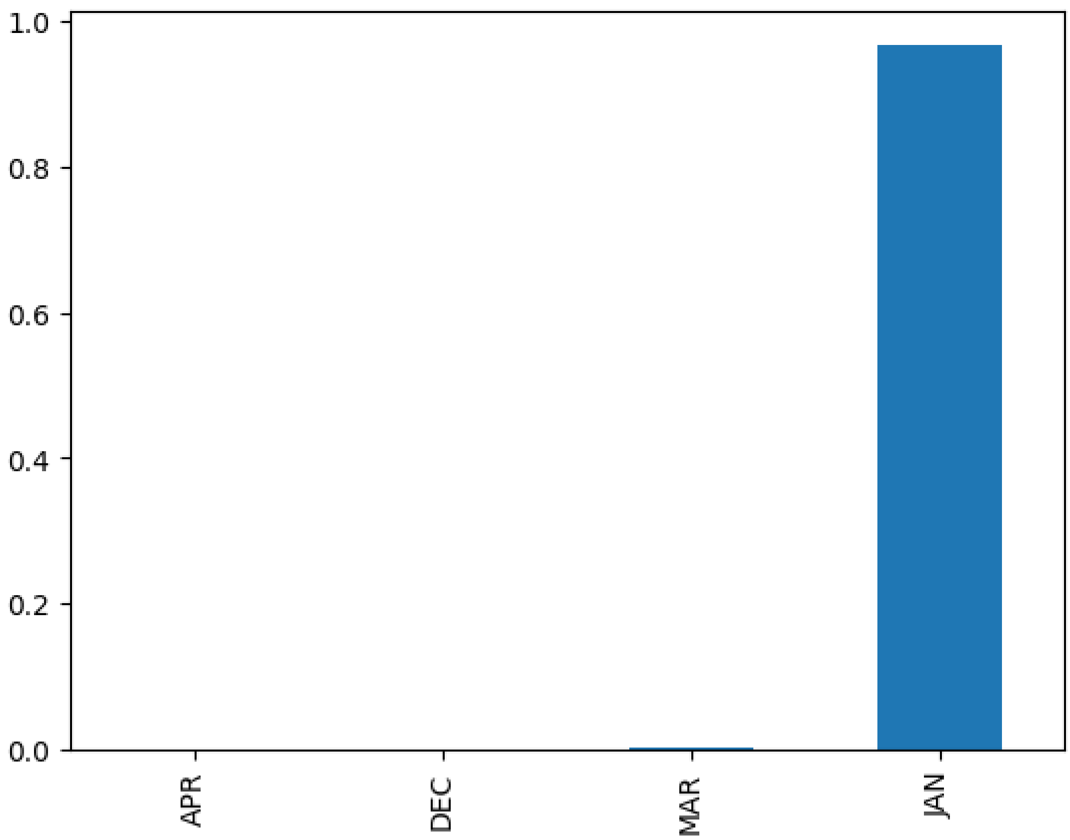
```
 Lasso Model:

The train score for ls model is 0.9991293224579318
The test score for ls model is0.9991975662336752
```

In [39]:

```
pd.Series(lasso.coef_,features).sort_values(ascending=True).plot(kind="bar")
```

Out[39]:

<Axes: >



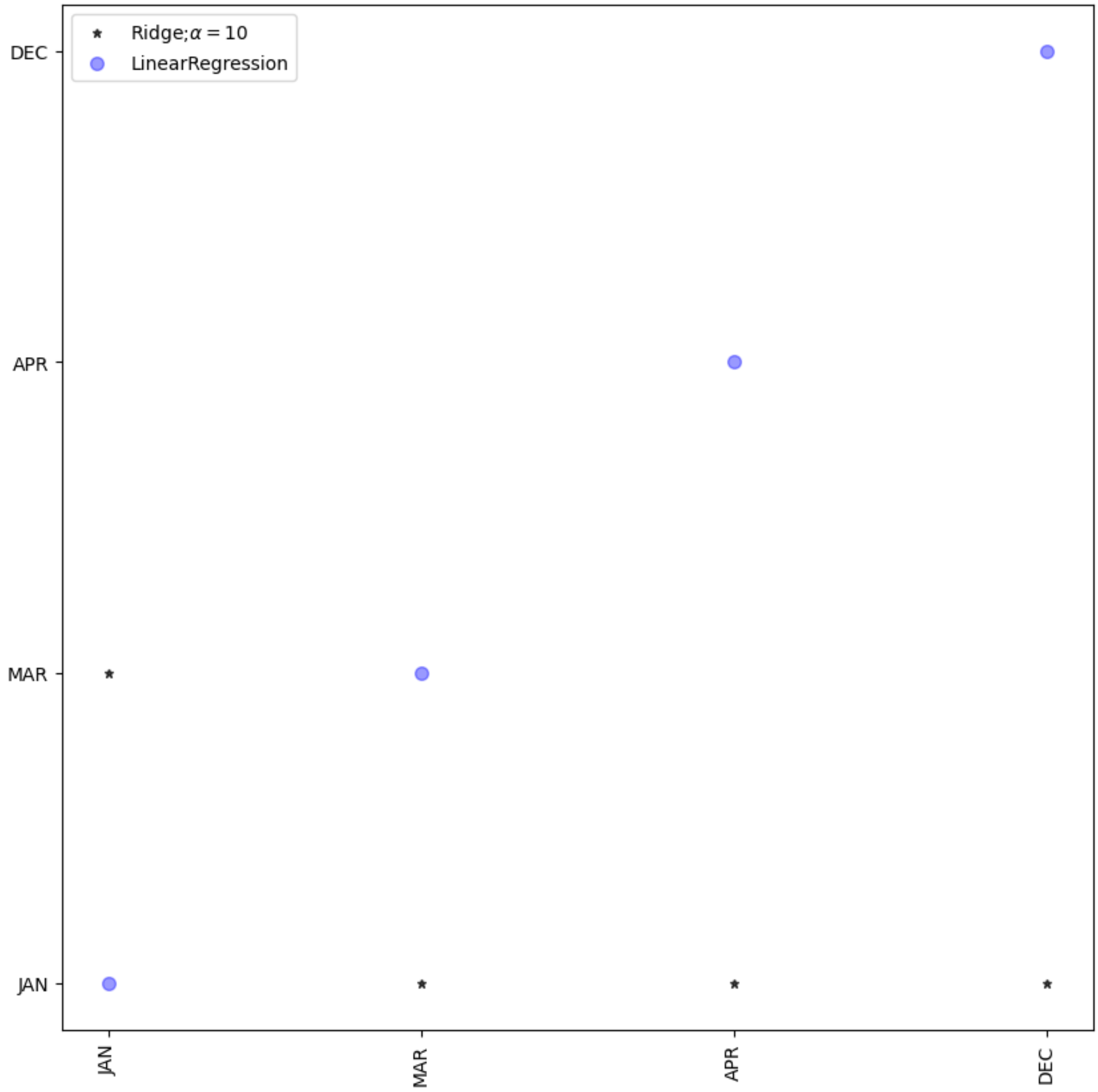
In [40]:

```
from sklearn.linear_model import LassoCV
lasso_cv=LassoCV(alphas=[0.0001,0.001,0.01,1,10],random_state=0).fit(x_train,y_train)
print(lasso_cv.score(x_train,y_train))
print(lasso_cv.score(x_test,y_test))
```

```
0.9999999999999198
0.9999999999999253
```

In [41]:

```
igure(figsize=(10,10))
lot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker="*",markersize=5,color='BLA
lot(features,alpha=0.4,linestyle='none',marker="o",markersize=7,color='BLUE',label='Linea
ticks(rotation=90)
egend()
how()
```



ELASTIC NET

In [42]:

```
from sklearn.linear_model import ElasticNet
eln=ElasticNet()
eln.fit(x,y)
print(eln.coef_)
print(eln.intercept_)
print(eln.score(x,y))
```

```
[ 9.96190198e-01  1.01899510e-03 -7.79895878e-05  2.63803659e-04]
0.03808751310977243
0.9999931631406689
```

In [45]:

```
y_pred_elastic = eln.predict(x_train)
mean_squared_error=np.mean((y_pred_elastic - y_train)**2)
print(mean_squared_error)
```

```
0.002635293637095672
```

CONCLUSION

```
From this we can conclude that the
score of LINEAR REGRESSION is 0.12658466121292555,
score of RIDGE MODEL is 0.9999999872352109,
score of LASSO MODEL is 0.9991293224579318,
score of ELASTIC NET is 0.002635293637095672,
 from above scores i observed that the RIDGE MODEL has an highest accuracy,so i prefer
RIDGE MODEL is set for this data set
```

In [ ]: