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

In [40]:

```
# importing necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn import preprocessing,svm
```

In [42]:

```
#Reading the dataset
df=pd.read_csv(r"C:\Users\mural\Downloads\rainfall.csv")
df
```

Out[42]:

	STATE_UT_NAME	DISTRICT	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NC
0	ANDAMAN And NICOBAR ISLANDS	NICOBAR	107.3	57.9	65.2	117.0	358.5	295.5	285.0	271.9	354.8	326.0	315
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	301.2	275
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	276.1	198
3	ARUNACHAL PRADESH	LOHIT	42.2	80.8	176.4	358.5	306.4	447.0	660.1	427.8	313.6	167.1	34
4	ARUNACHAL PRADESH	EAST SIANG	33.3	79.5	105.9	216.5	323.0	738.3	990.9	711.2	568.0	206.9	29
...	...	...	...	...	...	...	...	...	...	...	...	...	
636	KERALA	IDUKKI	13.4	22.1	43.6	150.4	232.6	651.6	788.9	527.3	308.4	343.2	172
637	KERALA	KASARGOD	2.3	1.0	8.4	46.9	217.6	999.6	1108.5	636.3	263.1	234.9	84
638	KERALA	PATHANAMTHITTA	19.8	45.2	73.9	184.9	294.7	556.9	539.9	352.7	266.2	359.4	213
639	KERALA	WAYANAD	4.8	8.3	17.5	83.3	174.6	698.1	1110.4	592.9	230.7	213.1	93
640	LAKSHADWEEP	LAKSHADWEEP	20.8	14.7	11.8	48.9	171.7	330.2	287.7	217.5	163.1	157.1	117

641 rows × 19 columns

DATA CLEANING AND PREPROCESSING

In [43]:

```
df.head()
```

Out[43]:

	STATE_UT_NAME	DISTRICT	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
0	ANDAMAN And NICOBAR ISLANDS	NICOBAR	107.3	57.9	65.2	117.0	358.5	295.5	285.0	271.9	354.8	326.0	315.2	250.9
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	301.2	275.8	128.3
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	276.1	198.6	100.0
3	ARUNACHAL PRADESH	LOHIT	42.2	80.8	176.4	358.5	306.4	447.0	660.1	427.8	313.6	167.1	34.1	29.8
4	ARUNACHAL PRADESH	EAST SIANG	33.3	79.5	105.9	216.5	323.0	738.3	990.9	711.2	568.0	206.9	29.5	31.7

In [44]:

```
df.tail()
```

Out[44]:

	STATE_UT_NAME	DISTRICT	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV
636	KERALA	IDUKKI	13.4	22.1	43.6	150.4	232.6	651.6	788.9	527.3	308.4	343.2	172.9
637	KERALA	KASARGOD	2.3	1.0	8.4	46.9	217.6	999.6	1108.5	636.3	263.1	234.9	84.6
638	KERALA	PATHANAMTHITTA	19.8	45.2	73.9	184.9	294.7	556.9	539.9	352.7	266.2	359.4	213.5
639	KERALA	WAYANAD	4.8	8.3	17.5	83.3	174.6	698.1	1110.4	592.9	230.7	213.1	93.6
640	LAKSHADWEEP	LAKSHADWEEP	20.8	14.7	11.8	48.9	171.7	330.2	287.7	217.5	163.1	157.1	117.7

In [45]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 641 entries, 0 to 640
Data columns (total 19 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   STATE_UT_NAME  641 non-null    object 
 1   DISTRICT       641 non-null    object 
 2   JAN            641 non-null    float64
 3   FEB            641 non-null    float64
 4   MAR            641 non-null    float64
 5   APR            641 non-null    float64
 6   MAY            641 non-null    float64
 7   JUN            641 non-null    float64
 8   JUL            641 non-null    float64
 9   AUG            641 non-null    float64
 10  SEP            641 non-null    float64
 11  OCT            641 non-null    float64
 12  NOV            641 non-null    float64
 13  DEC            641 non-null    float64
 14  ANNUAL         641 non-null    float64
 15  Jan-Feb        641 non-null    float64
 16  Mar-May        641 non-null    float64
 17  Jun-Sep        641 non-null    float64
 18  Oct-Dec        641 non-null    float64
dtypes: float64(17), object(2)
memory usage: 95.3+ KB
```

In [46]:

```
df.describe()
```

Out[46]:

	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	
count	641.000000	641.000000	641.000000	641.000000	641.000000	641.000000	641.000000	641.000000	641.0C
mean	18.355070	20.984399	30.034789	45.543214	81.535101	196.007332	326.033697	291.152262	194.6C
std	21.082806	27.729596	45.451082	71.556279	111.960390	196.556284	221.364643	152.647325	99.83
min	0.000000	0.000000	0.000000	0.000000	0.900000	3.800000	11.600000	14.100000	8.6C
25%	6.900000	7.000000	7.000000	5.000000	12.100000	68.800000	206.400000	194.600000	128.8C
50%	13.300000	12.300000	12.700000	15.100000	33.900000	131.900000	293.700000	284.800000	181.3C
75%	19.200000	24.100000	33.200000	48.300000	91.900000	226.600000	374.800000	358.100000	234.1C
max	144.500000	229.600000	367.900000	554.400000	733.700000	1476.200000	1820.900000	1522.100000	826.3C

In [47]:

```
#Checking for null values
df.isnull().sum()
```

Out[47]:

```
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 [48]:

```
#Checking for duplicate values
df.duplicated().sum()
```

Out[48]:

```
0
```

In [50]:

```
df.columns
```

Out[50]:

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

In [52]:

```
df.shape
```

Out[52]:

```
(641, 19)
```

Feature scaling:spliting the data into train data and test data

In [53]:

```
x=df[['MAR']]
y=df[['JAN']]
```

In [54]:

```
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_state=100)
```

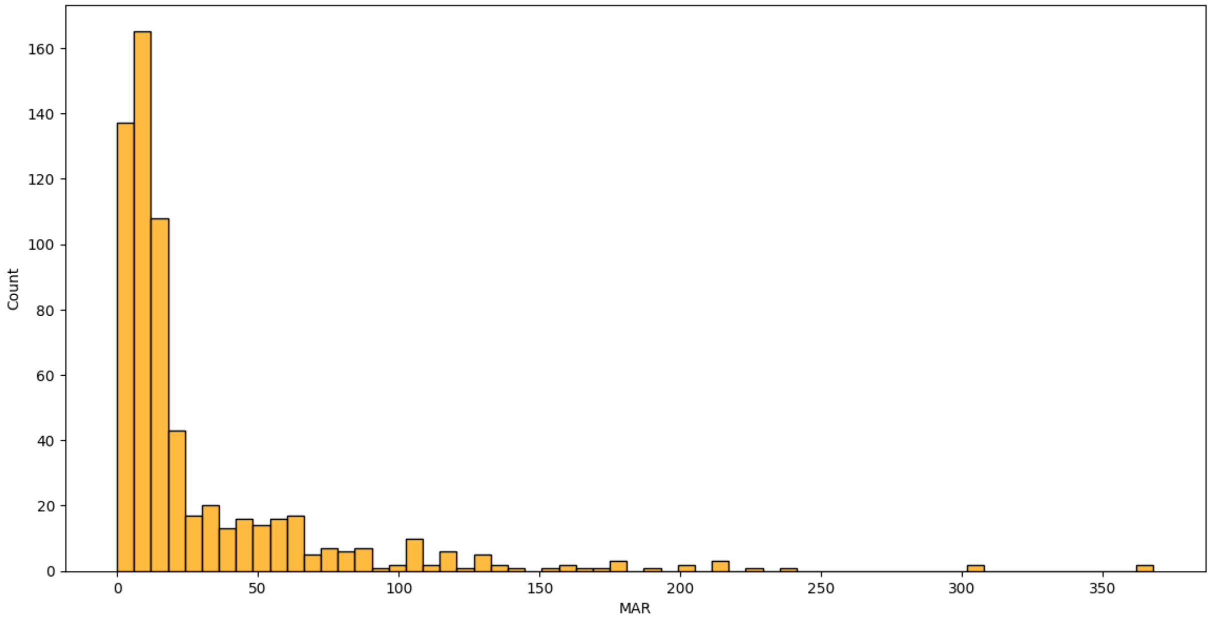
DATA VISUALIZATION

In [56]:

```
plt.figure(figsize=(14,7))
sns.histplot(data=df,x='MAR',color='orange')
```

Out[56]:

```
<Axes: xlabel='MAR', ylabel='Count'>
```

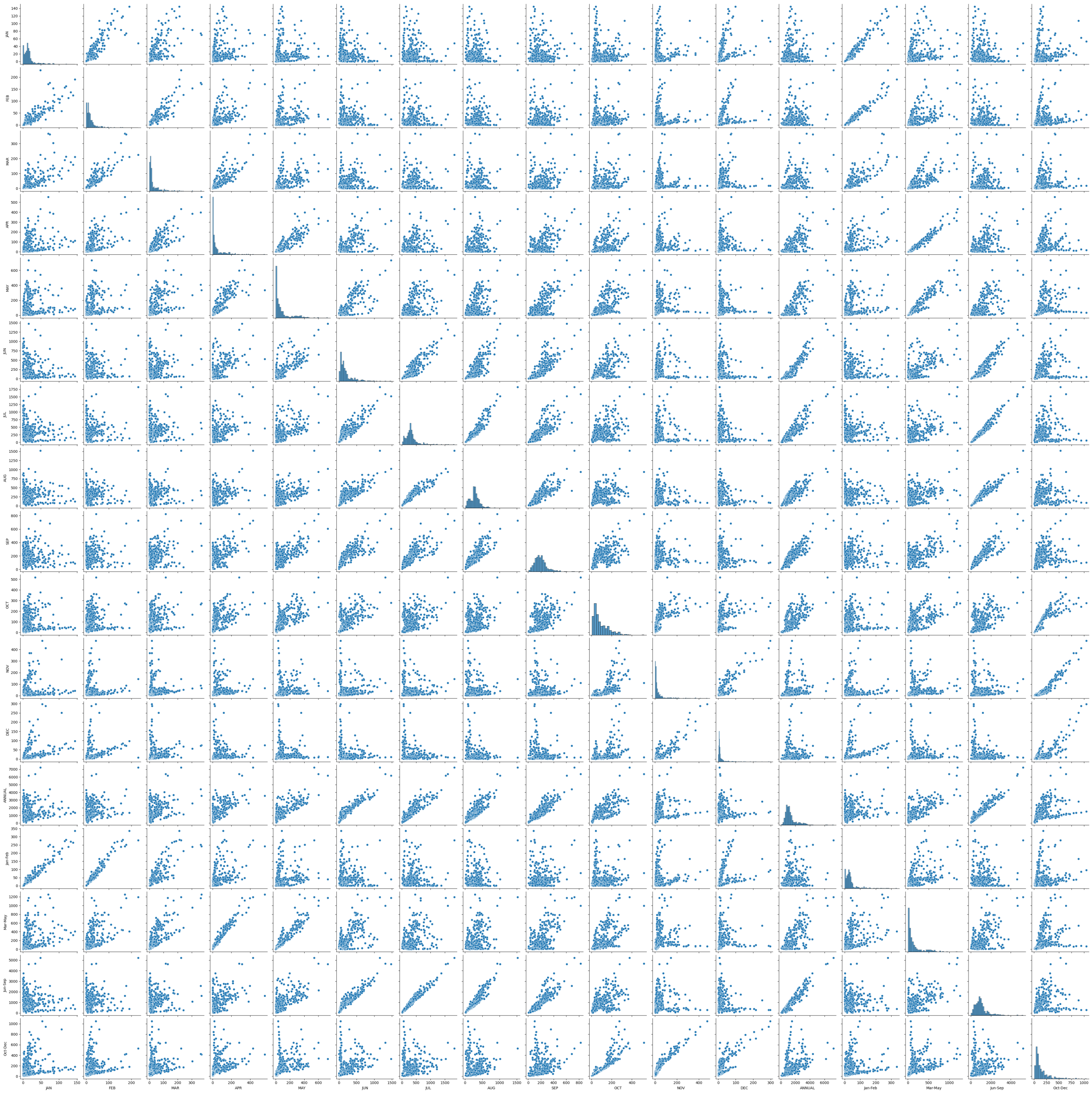


In [57]:

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

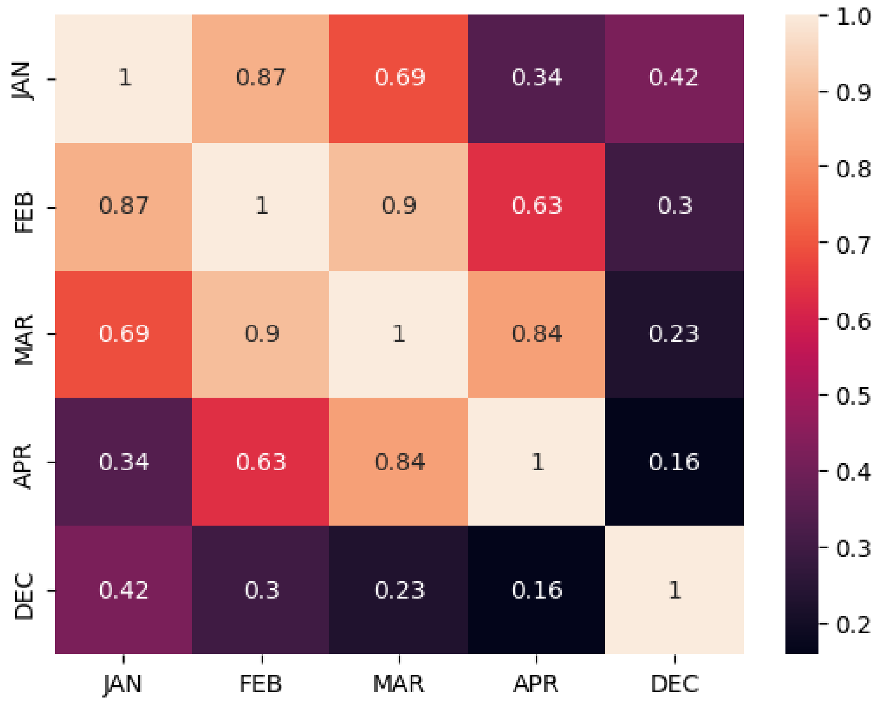
Out[57]:

<seaborn.axisgrid.PairGrid at 0x2038ff622d0>



In [58]:

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



LINEAR REGRESSION

In [59]:

```
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_
```

```
[8.05318034]
```

Out[59]:

	coefficient
MAR	0.342951

In [60]:

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

```
0.3746861023792344
```

In [61]:

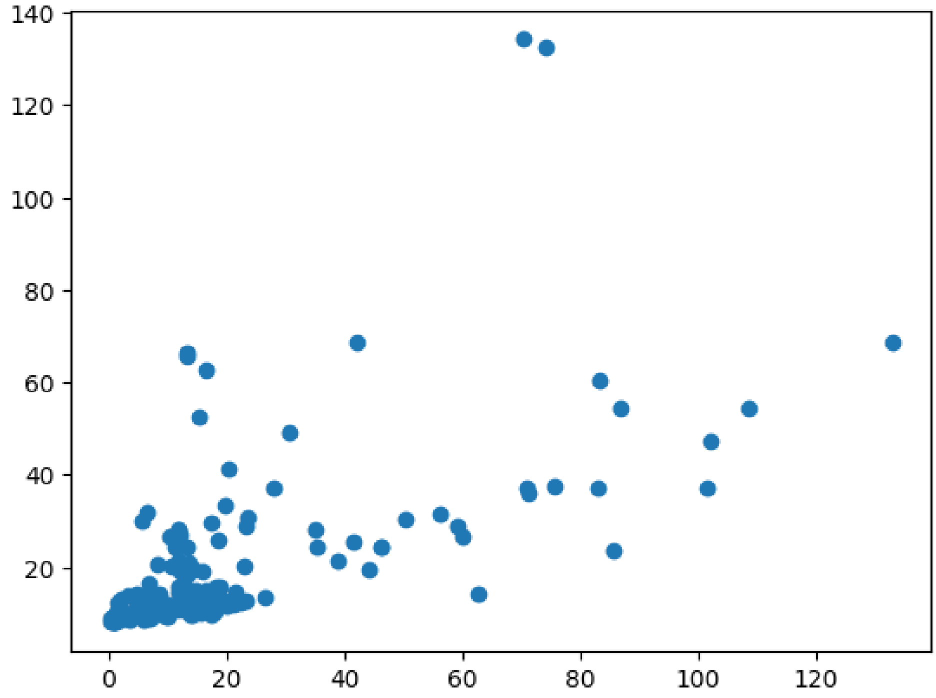
```
predictions=reg.predict(X_test)
```

In [62]:

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

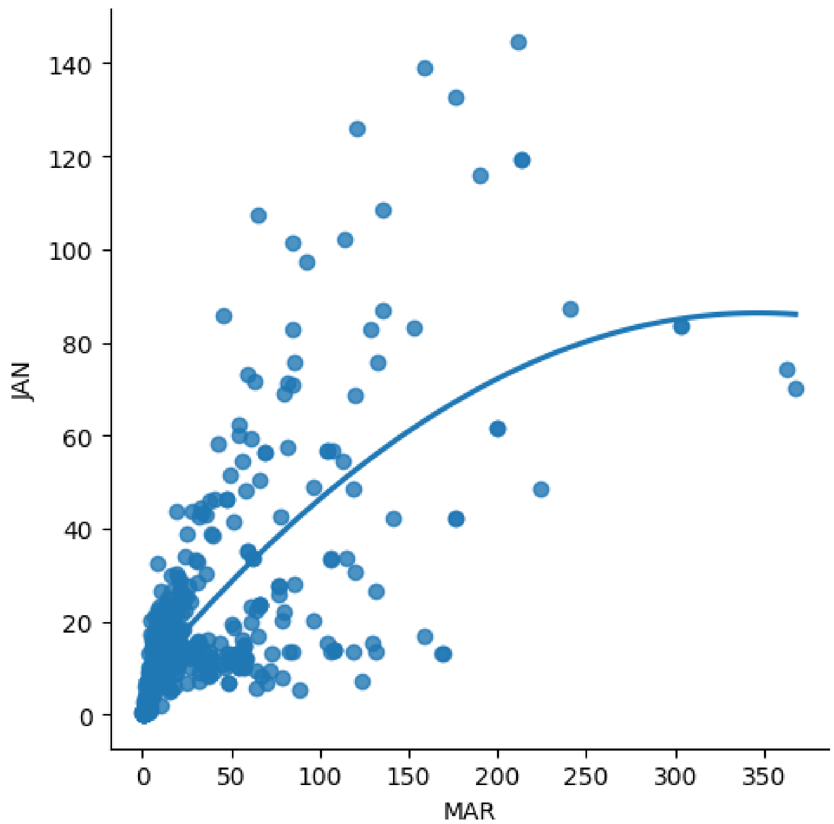
Out[62]:

```
<matplotlib.collections.PathCollection at 0x203a1dd3490>
```



In [63]:

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



In [64]:

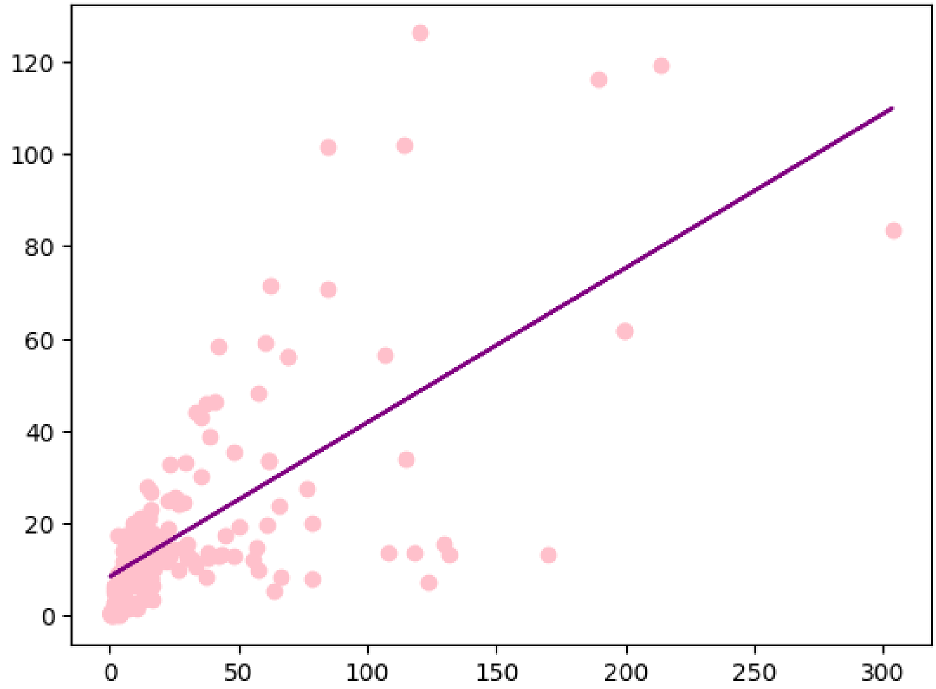
```
X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.33)
reg.fit(X_train,y_train)
reg.fit(X_test,y_test)
```

Out[64]:

```
▾ LinearRegression
LinearRegression()
```

In [66]:

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



In [67]:

```
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: 0.4723678424322967
```

RIDGE MODEL

In [68]:

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

In [69]:

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

In [70]:

```
x=np.array(df['MAR']).reshape(-1,1)
y=np.array(df['JAN']).reshape(-1,1)
```

In [71]:

```
x= df[features].values
y= df[target].values
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.4,random_state=100)
```

In [72]:

```
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)
```

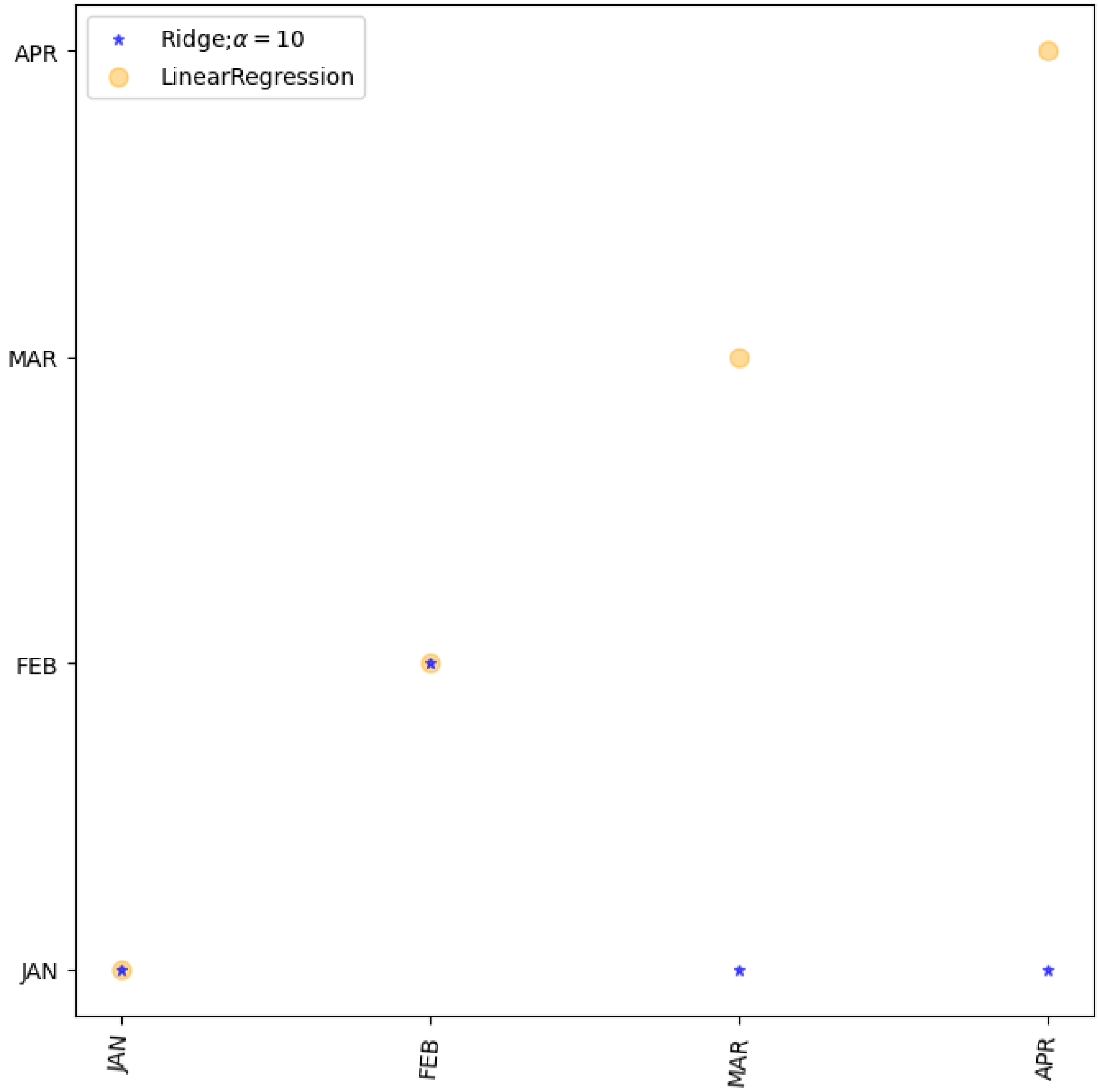
```
print("\n Ridge Model:\n")
print("the train score for ridge model is{}".format(train_score_ridge))
print("the test score for ridge model is{}".format(test_score_ridge))
```

In [74]:

```
lr=LinearRegression()
```

In [77]:

```
figure(figsize=(8,8))
plot(features,ridgeReg.coef_,alpha=0.6,linestyle='none',marker="*",markersize=5,color='blue',label=r'Ridge
plot(features,alpha=0.4,linestyle='none',marker="o",markersize=8,color='orange',label='LinearRegression')
xticks(rotation=85)
legend()
show()
```



LASSO MODEL

In [78]:

```
print("\n Lasso Model:\n")
lasso=Lasso(alpha=9)
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))
```

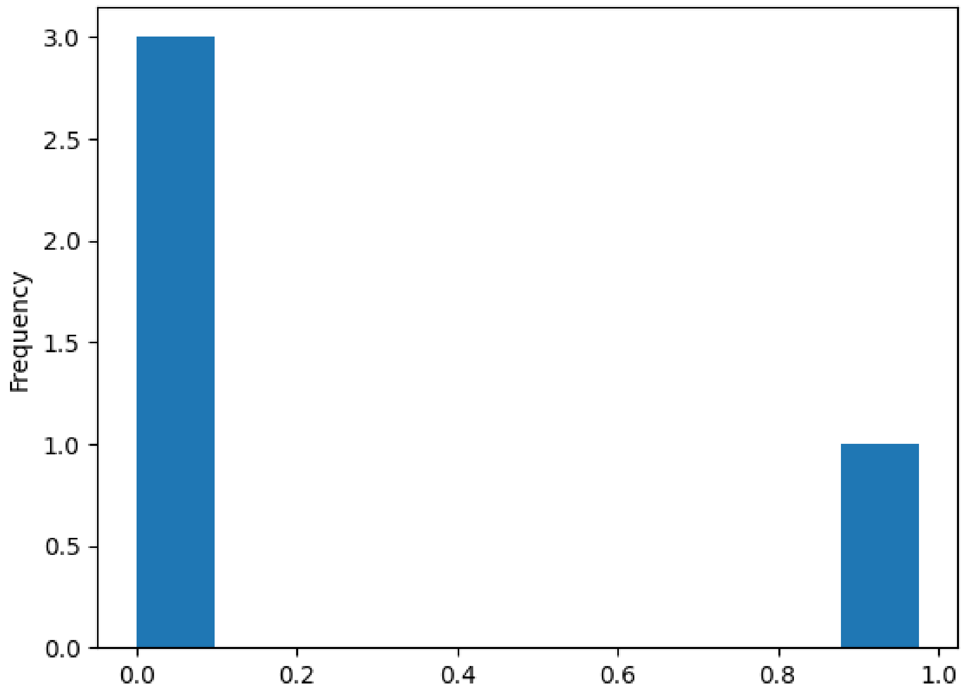
```
 Lasso Model:

The train score for ls model is 0.9998493280695062
The test score for ls model is0.9998642481931751
```

In [79]:

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

Out[79]:

```
<Axes: ylabel='Frequency'>
```



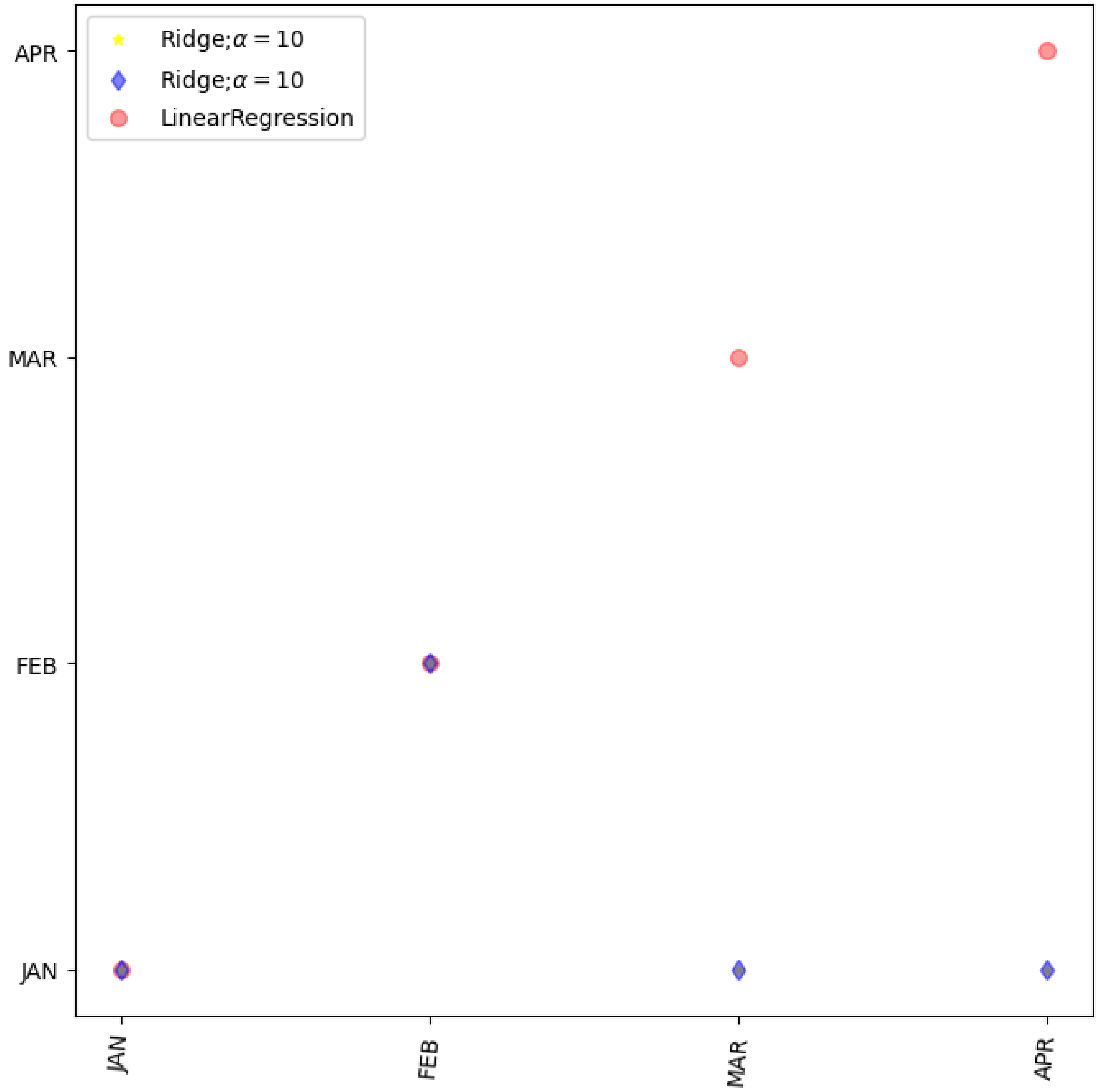
In [80]:

```
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))
```

```
0.9999999482560009
0.9999999653904249
```

In [83]:

```
gure(figsize= (8,8))
ot(features,ridgeReg.coef_,alpha=0.8,linestyle='none',marker="*",markersize=5,color='yellow',label=r'Ridge
ot(lasso_cv.coef_,alpha=0.5,linestyle='none',marker='d',markersize=6,color='blue',label=r'Ridge;$\alpha=10
ot(features,alpha=0.4,linestyle='none',marker='o',markersize=7,color="red",label='LinearRegression')
icks(rotation=85)
icks(rotation = 85)
gend()
ow()
```



ELASTICNET

In [84]:

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

```
[ 0.00345558  0.99207699  0.00300149 -0.        ]
0.012682999169118858
0.9999946106160088
```

In [85]:

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

```
0.005384495296322562
```

CONCLUSION: THE SCORE OF LINEAR REGRESSION IS :0.3746861023792344 THE SCORE OF RIDGE MODEL IS: 0.9999999792491524 THE SCORE OF LASSO MODEL IS : 0.9999999999999198 THE SCORE OFELASTIC NET IS :0.005384495296322562 AMONG ALL MODELS LASSO YEILD HIGHESTACCURACY.SO,WE PREFER LASSO MODEL AND RIDGE FOR THIS DATA SET