Rain Fall ¶

Linear Regression

1.PROBLEM STATEMENT:To predict and analysis the district wise Rainfall in a year

Test data

In [1]:
```
# importing all the libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing, svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [2]:
```
df=pd.read_csv(r"C:\Users\91756\Documents\district wise rainfall normal.csv")
df
```

Out[2]:

	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
...	...	...	...	...	...	...	...	...	...	...	...
636	KERALA	IDUKKI	13.4	22.1	43.6	150.4	232.6	651.6	788.9	527.3	308.4
637	KERALA	KASARGOD	2.3	1.0	8.4	46.9	217.6	999.6	1108.5	636.3	263.1
638	KERALA	PATHANAMTHITTA	19.8	45.2	73.9	184.9	294.7	556.9	539.9	352.7	266.2
639	KERALA	WAYANAD	4.8	8.3	17.5	83.3	174.6	698.1	1110.4	592.9	230.7
640	LAKSHADWEEP	LAKSHADWEEP	20.8	14.7	11.8	48.9	171.7	330.2	287.7	217.5	163.1

641 rows × 19 columns

In [3]:
```
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
```

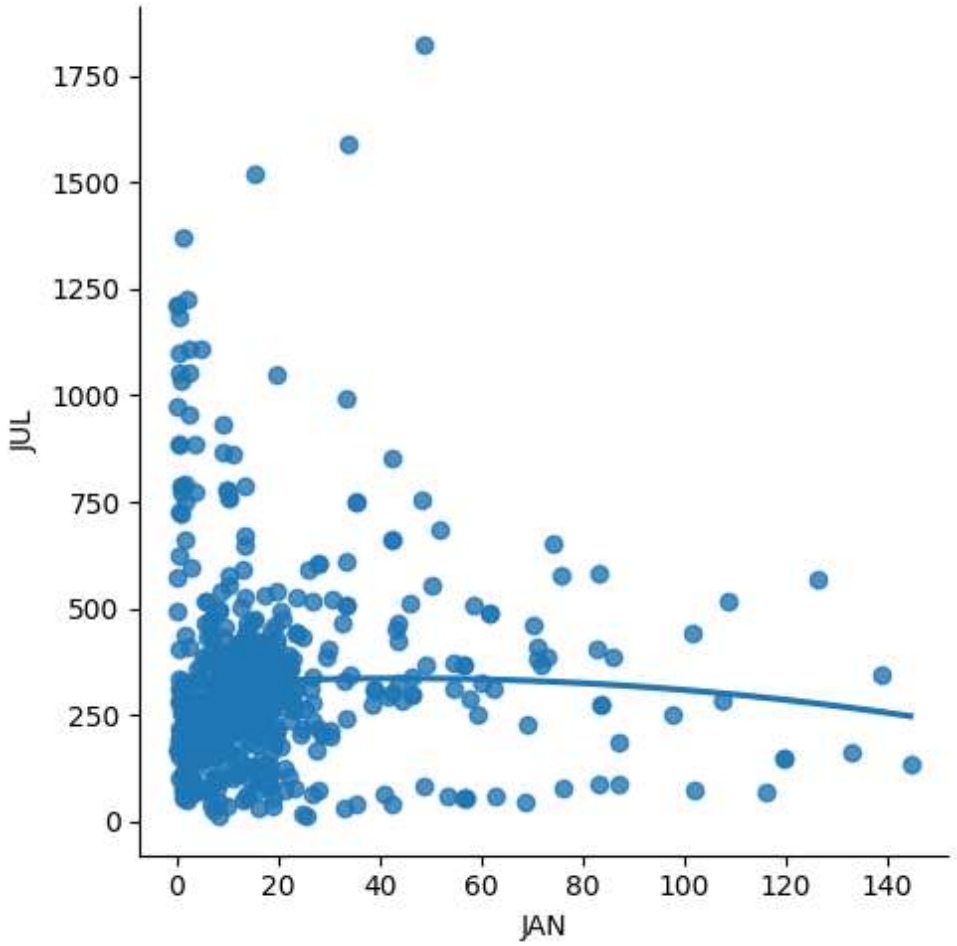
In [4]:
```
df.describe()
```

Out[4]:

	JAN	FEB	MAR	APR	MAY	JUN	JUL	
count	641.000000	641.000000	641.000000	641.000000	641.000000	641.000000	641.000000	641.0
mean	18.355070	20.984399	30.034789	45.543214	81.535101	196.007332	326.033697	291.1
std	21.082806	27.729596	45.451082	71.556279	111.960390	196.556284	221.364643	152.6
min	0.000000	0.000000	0.000000	0.000000	0.900000	3.800000	11.600000	14.1
25%	6.900000	7.000000	7.000000	5.000000	12.100000	68.800000	206.400000	194.6
50%	13.300000	12.300000	12.700000	15.100000	33.900000	131.900000	293.700000	284.8
75%	19.200000	24.100000	33.200000	48.300000	91.900000	226.600000	374.800000	358.1
max	144.500000	229.600000	367.900000	554.400000	733.700000	1476.200000	1820.900000	1522.1

In [5]:
```
sns.lmplot(x="JAN",y="JUL",data=df,order=2,ci=None)
```

Out[5]: <seaborn.axisgrid.FacetGrid at 0x18e4e4d7bb0>



In [6]:
```
df.fillna(method='ffill',inplace=True)
```

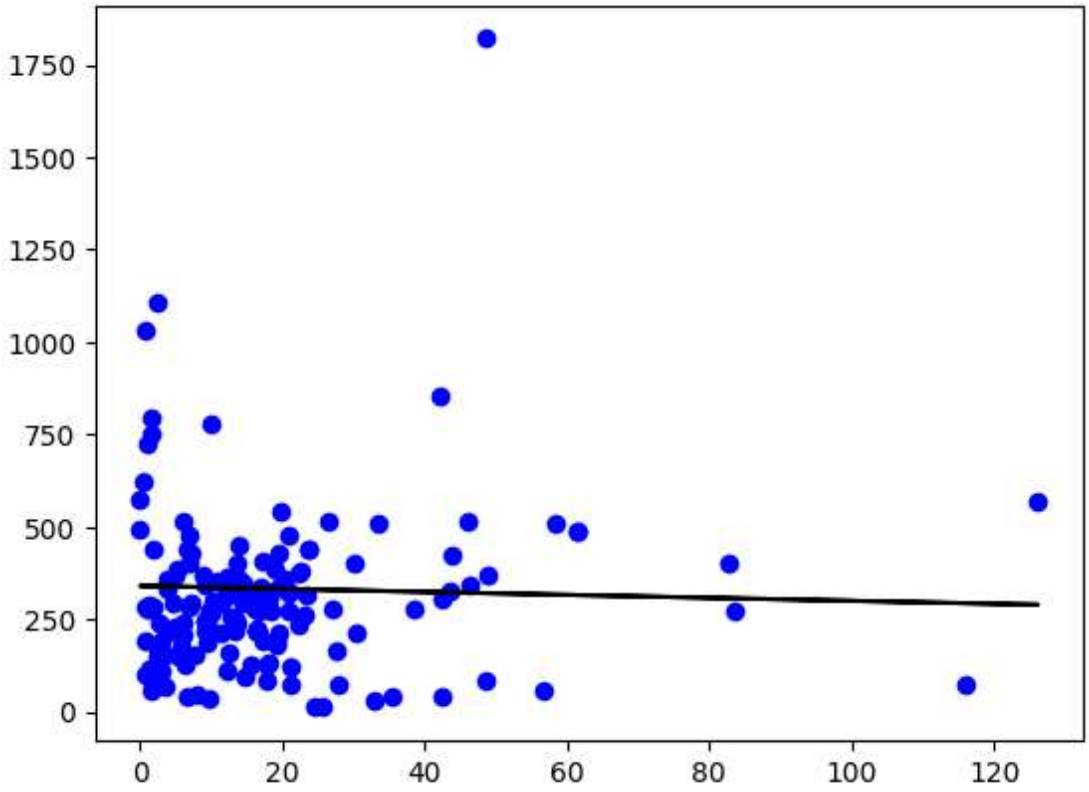
In [7]:
```
X = np.array(df['JAN']).reshape(-1,1)
y = np.array(df['JUL']).reshape(-1,1)
```

In [8]:
```
df.dropna(inplace = True)
```

In [9]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.25)
regr = LinearRegression()
regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

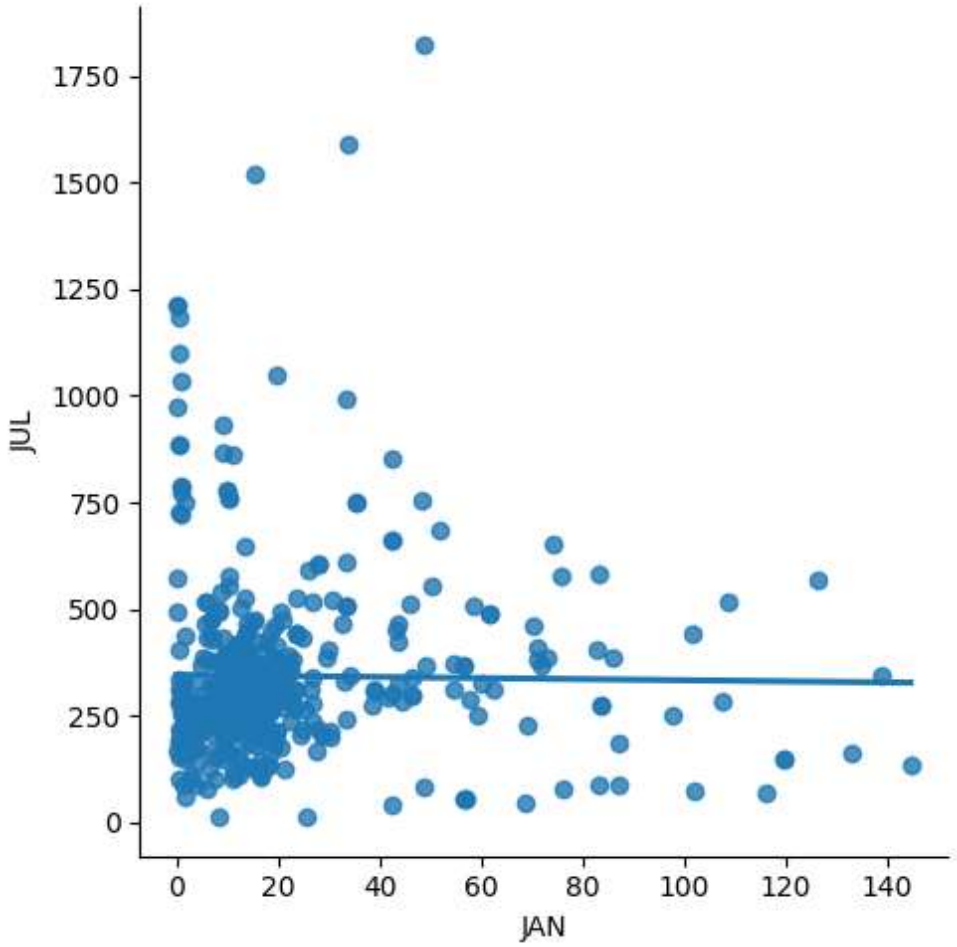
```
-0.024737972064376113
```

In [10]:
```
y_pred = regr.predict(X_test)
plt.scatter(X_test, y_test, color ='b')
plt.plot(X_test, y_pred, color ='k')
plt.show()
```



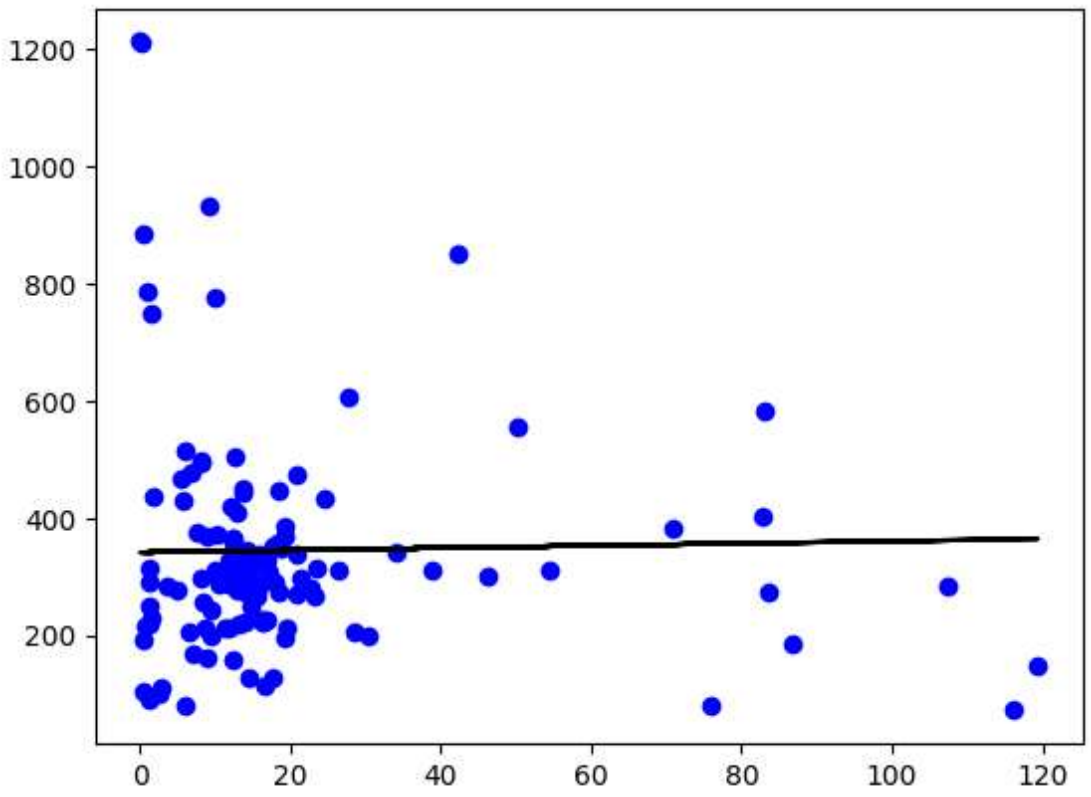
In [11]:
```
df500 = df[:][:500]
sns.lmplot(x = "JAN", y ="JUL", data = df500, order = 1, ci = None)
```

Out[11]: <seaborn.axisgrid.FacetGrid at 0x18e4e4d6410>



In [12]:
```
df500.fillna(method ='ffill', inplace = True)
X = np.array(df500['JAN']).reshape(-1,1)
y = np.array(df500['JUL']).reshape(-1,1)
df500.dropna(inplace = True)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25)
regr = LinearRegression()
regr.fit(X_train, y_train)
print("Regression: ",regr.score(X_test, y_test))
y_pred = regr.predict(X_test)
plt.scatter(X_test, y_test, color ='b')
plt.plot(X_test, y_pred, color = 'k')
plt.show()
```

```
Regression:  -0.007915302891319476
```



In [13]:
```
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:  -0.007915302891319476
```

Train data

In [14]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing, svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [15]:
```
df1=pd.read_csv(r"C:\Users\91756\Documents\rainfall in india 1901-2015.csv")
df1
```

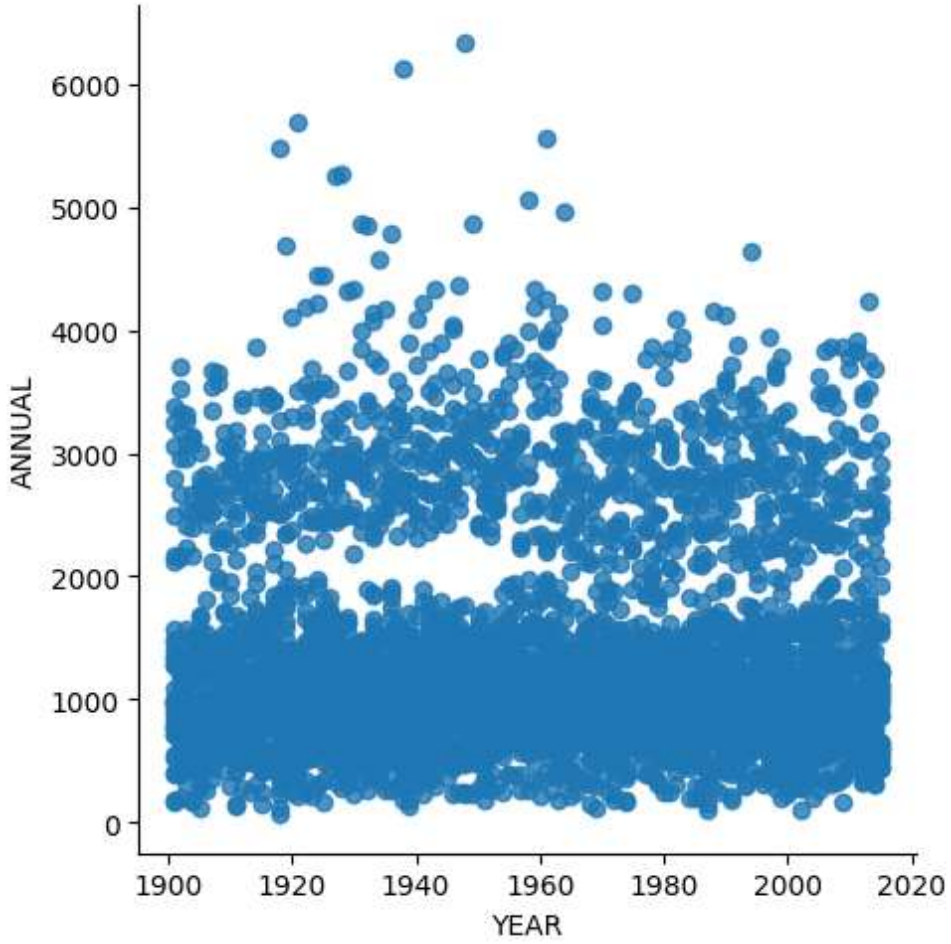
Out[15]:

	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NO
0	ANDAMAN & NICOBAR ISLANDS	1901	49.2	87.1	29.2	2.3	528.8	517.5	365.1	481.1	332.6	388.5	558.
1	ANDAMAN & NICOBAR ISLANDS	1902	0.0	159.8	12.2	0.0	446.1	537.1	228.9	753.7	666.2	197.2	359.
2	ANDAMAN & NICOBAR ISLANDS	1903	12.7	144.0	0.0	1.0	235.1	479.9	728.4	326.7	339.0	181.2	284.
3	ANDAMAN & NICOBAR ISLANDS	1904	9.4	14.7	0.0	202.4	304.5	495.1	502.0	160.1	820.4	222.2	308.
4	ANDAMAN & NICOBAR ISLANDS	1905	1.3	0.0	3.3	26.9	279.5	628.7	368.7	330.5	297.0	260.7	25
...	...	...	...	...	...	...	...	...	...	...	...	...	...
4111	LAKSHADWEEP	2011	5.1	2.8	3.1	85.9	107.2	153.6	350.2	254.0	255.2	117.4	184.
4112	LAKSHADWEEP	2012	19.2	0.1	1.6	76.8	21.2	327.0	231.5	381.2	179.8	145.9	12.
4113	LAKSHADWEEP	2013	26.2	34.4	37.5	5.3	88.3	426.2	296.4	154.4	180.0	72.8	78.
4114	LAKSHADWEEP	2014	53.2	16.1	4.4	14.9	57.4	244.1	116.1	466.1	132.2	169.2	59.
4115	LAKSHADWEEP	2015	2.2	0.5	3.7	87.1	133.1	296.6	257.5	146.4	160.4	165.4	231.

4116 rows × 19 columns

In [16]:
```
sns.lmplot(x="YEAR",y="ANNUAL",data=df1,order=2,ci=None)
```

Out[16]: <seaborn.axisgrid.FacetGrid at 0x18e53ad27d0>



In [17]:
```
df1.describe()
```

Out[17]:

	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL
	116.000000	4112.000000	4113.000000	4110.000000	4112.000000	4113.000000	4111.000000	4109.000000
	58.218659	18.957320	21.805325	27.359197	43.127432	85.745417	230.234444	347.214334
	33.140898	33.585371	35.909488	46.959424	67.831168	123.234904	234.710758	269.539667
	01.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.400000	0.000000
	30.000000	0.600000	0.600000	1.000000	3.000000	8.600000	70.350000	175.600000
	58.000000	6.000000	6.700000	7.800000	15.700000	36.600000	138.700000	284.800000
	87.000000	22.200000	26.800000	31.300000	49.950000	97.200000	305.150000	418.400000
	15.000000	583.700000	403.500000	605.600000	595.100000	1168.600000	1609.900000	2362.800000

In [18]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4116 entries, 0 to 4115
Data columns (total 19 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   SUBDIVISION  4116 non-null   object 
 1   YEAR         4116 non-null   int64  
 2   JAN          4112 non-null   float64
 3   FEB          4113 non-null   float64
 4   MAR          4110 non-null   float64
 5   APR          4112 non-null   float64
 6   MAY          4113 non-null   float64
 7   JUN          4111 non-null   float64
 8   JUL          4109 non-null   float64
 9   AUG          4112 non-null   float64
 10  SEP          4110 non-null   float64
 11  OCT          4109 non-null   float64
 12  NOV          4105 non-null   float64
 13  DEC          4106 non-null   float64
 14  ANNUAL       4090 non-null   float64
 15  Jan-Feb      4110 non-null   float64
 16  Mar-May      4107 non-null   float64
 17  Jun-Sep      4106 non-null   float64
 18  Oct-Dec      4103 non-null   float64
dtypes: float64(17), int64(1), object(1)
memory usage: 611.1+ KB
```

In [19]:
```
df1.fillna(method='ffill', inplace=True)
```

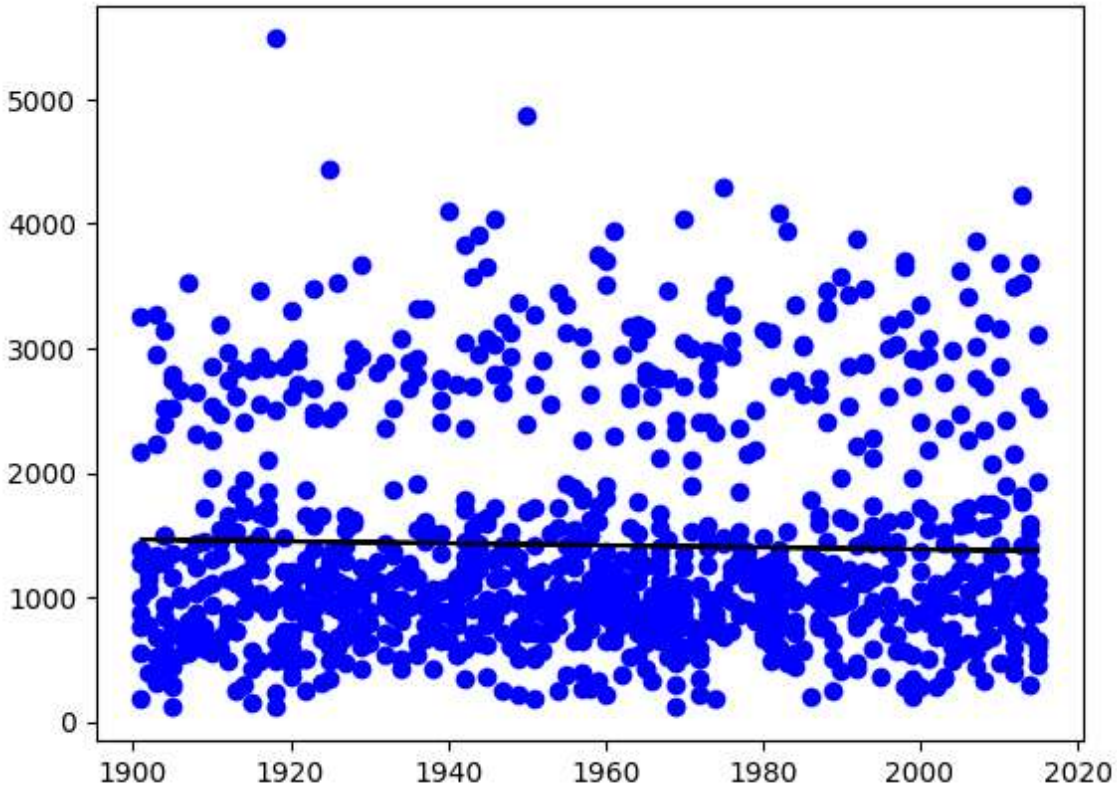
In [20]:
```
X = np.array(df1['YEAR']).reshape(-1,1)
y = np.array(df1['ANNUAL']).reshape(-1,1)
```

In [21]:
```
df1.dropna(inplace = True)
```

In [22]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.25)
regr = LinearRegression()
regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

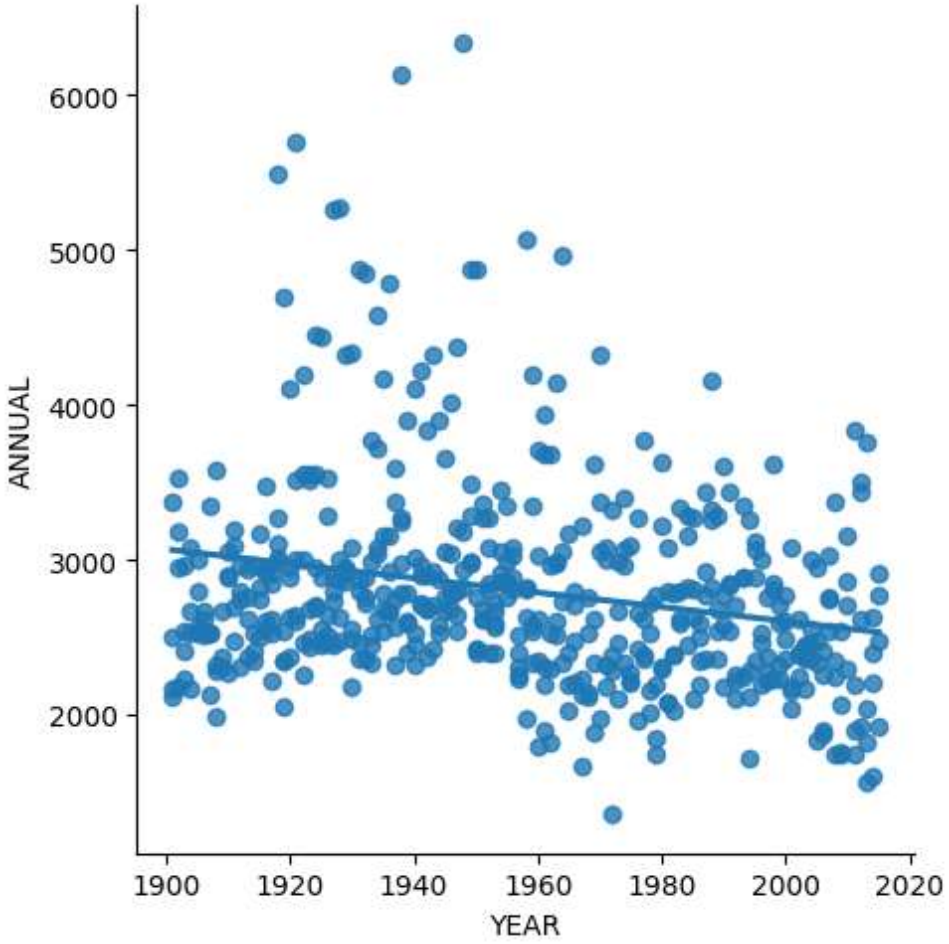
```
-0.0035970369409776826
```

In [23]:
```
y_pred = regr.predict(X_test)
plt.scatter(X_test, y_test, color ='b')
plt.plot(X_test, y_pred, color ='k')
plt.show()
```



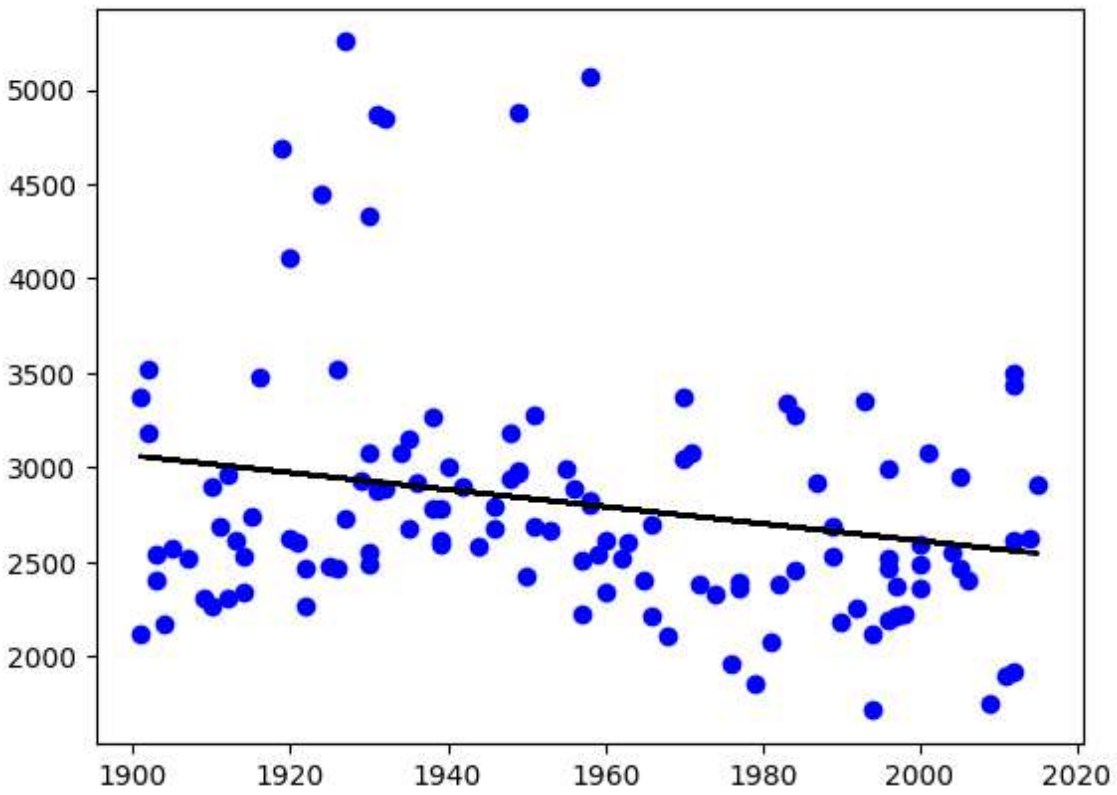
In [24]:
```
df500 = df1[:][:500]
sns.lmplot(x = "YEAR", y ="ANNUAL", data = df500, order = 1, ci = None)
```

Out[24]: <seaborn.axisgrid.FacetGrid at 0x18e53ad3b80>



In [25]:
```
df500.fillna(method ='ffill', inplace = True)
X = np.array(df500['YEAR']).reshape(-1,1)
y = np.array(df500['ANNUAL']).reshape(-1,1)
df500.dropna(inplace = True)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25)
regr = LinearRegression()
regr.fit(X_train, y_train)
print("Regression: ",regr.score(X_test, y_test))
y_pred = regr.predict(X_test)
plt.scatter(X_test, y_test, color ='b')
plt.plot(X_test, y_pred, color = 'k')
plt.show()
```

```
Regression:  0.0639531162288759
```



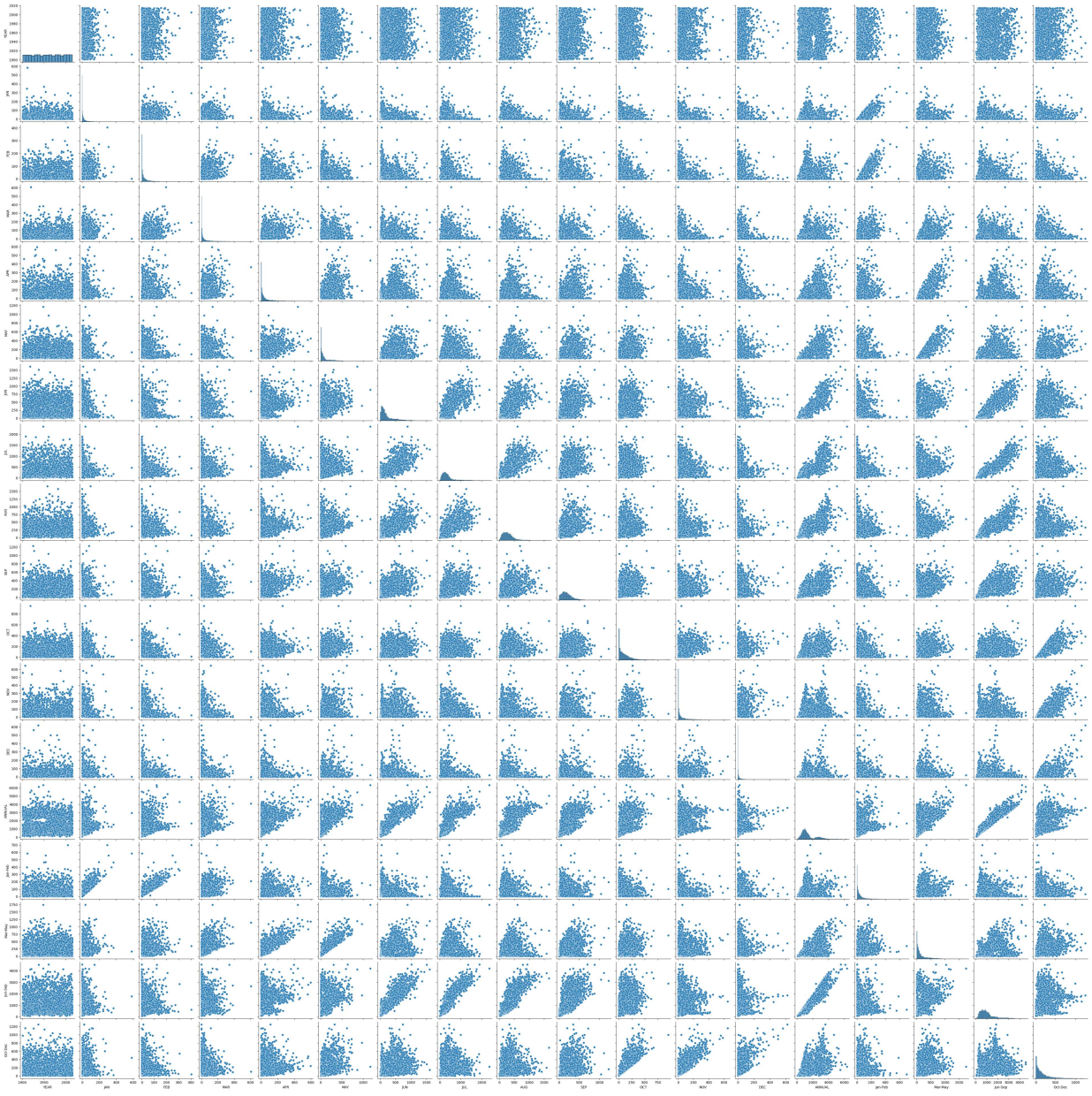
In [26]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
model = LinearRegression()
model.fit(X_train, y_train)
y_pred=model.predict(X_test)
r2=r2_score(y_test,y_pred)
print("R2 score: ",r2)
```

```
R2 score:  0.0639531162288759
```

Exploratory Data Analysis

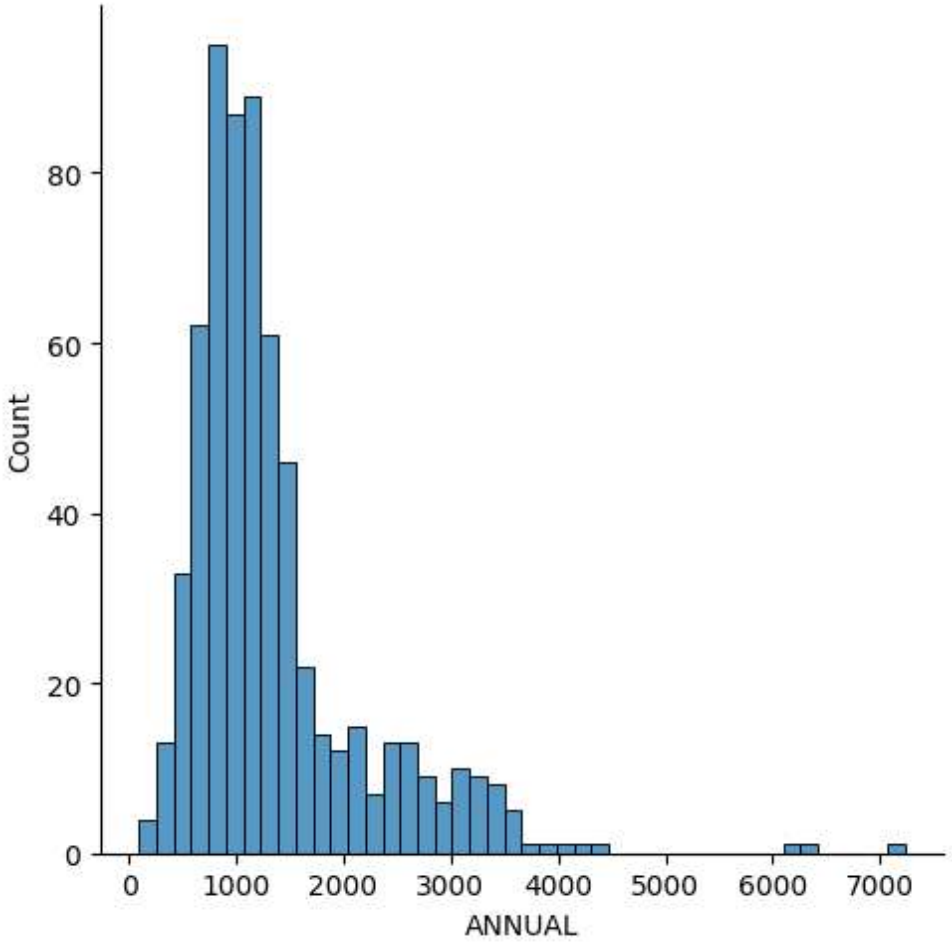
In [27]:
```
sns.pairplot(df1)
```

Out[27]: <seaborn.axisgrid.PairGrid at 0x18e53c1ee30>



In [28]:
```
sns.displot(df['ANNUAL'])
```

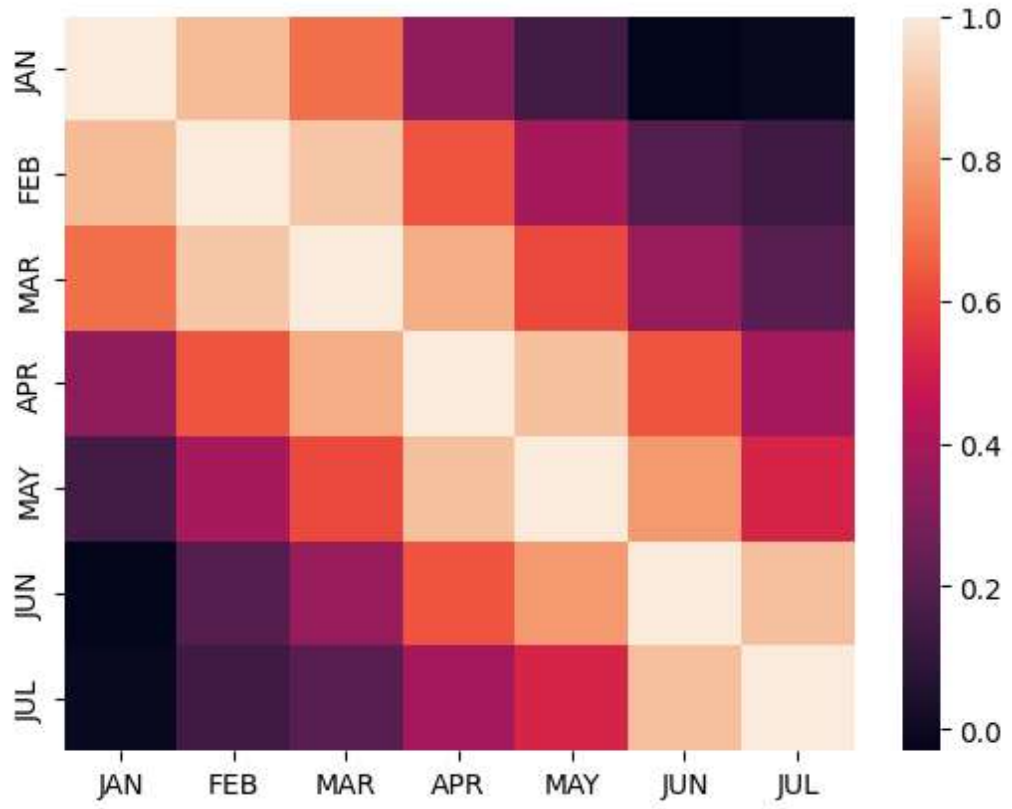
Out[28]: <seaborn.axisgrid.FacetGrid at 0x18e34180b80>



In [34]:
```
Housedf1=df[['JAN','FEB','MAR','APR','MAY','JUN','JUL']]
```

In [35]:
```
sns.heatmap(Housedf1.corr())
```

Out[35]: <Axes: >



In [41]:
```
x=Housedf1[['JAN','FEB','MAR','APR','MAY','JUN','JUL']]
y=df['ANNUAL']
```

In [42]:
```
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=101
```

In [43]:
```
from sklearn.linear_model import LinearRegression
lm=LinearRegression()
lm.fit(x_train,y_train)
```

Out[43]:
```
▾ LinearRegression

LinearRegression()
```

In [44]:
```
print(lm.intercept_)
```

```
286.04657460892054
```

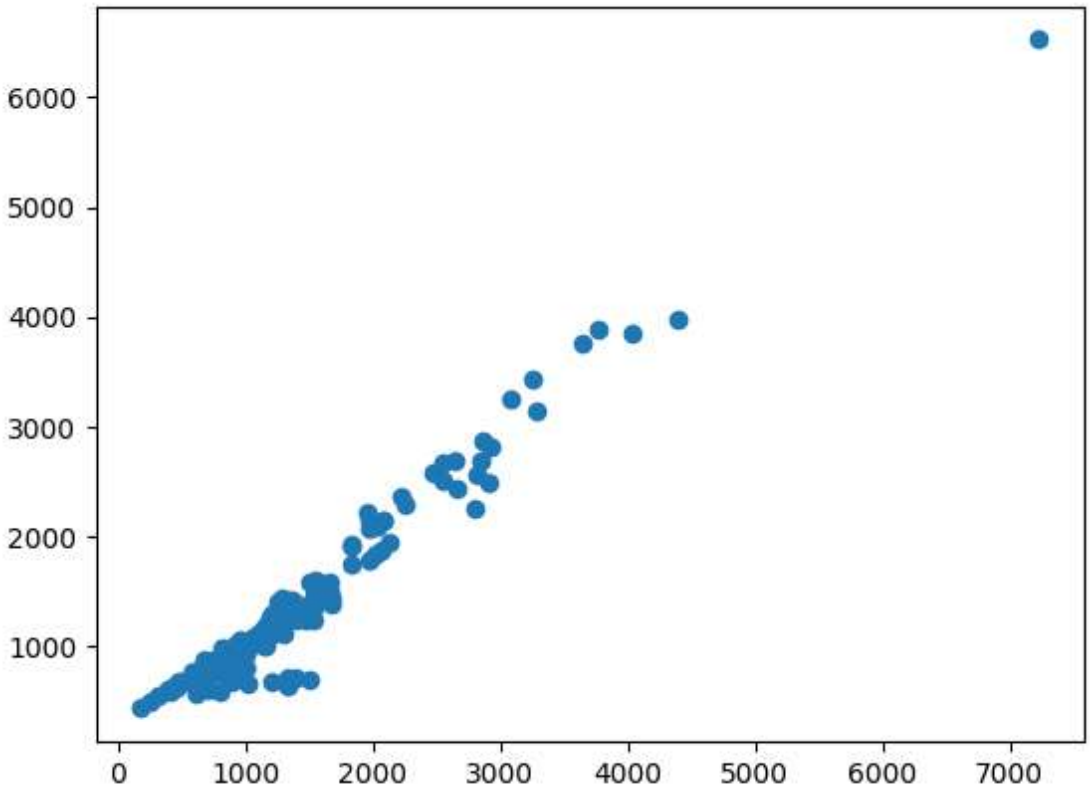
In [45]:
```
coeff_df=pd.DataFrame(lm.coef_,x.columns,columns=['coefficient'])
coeff_df
```

Out[45]:

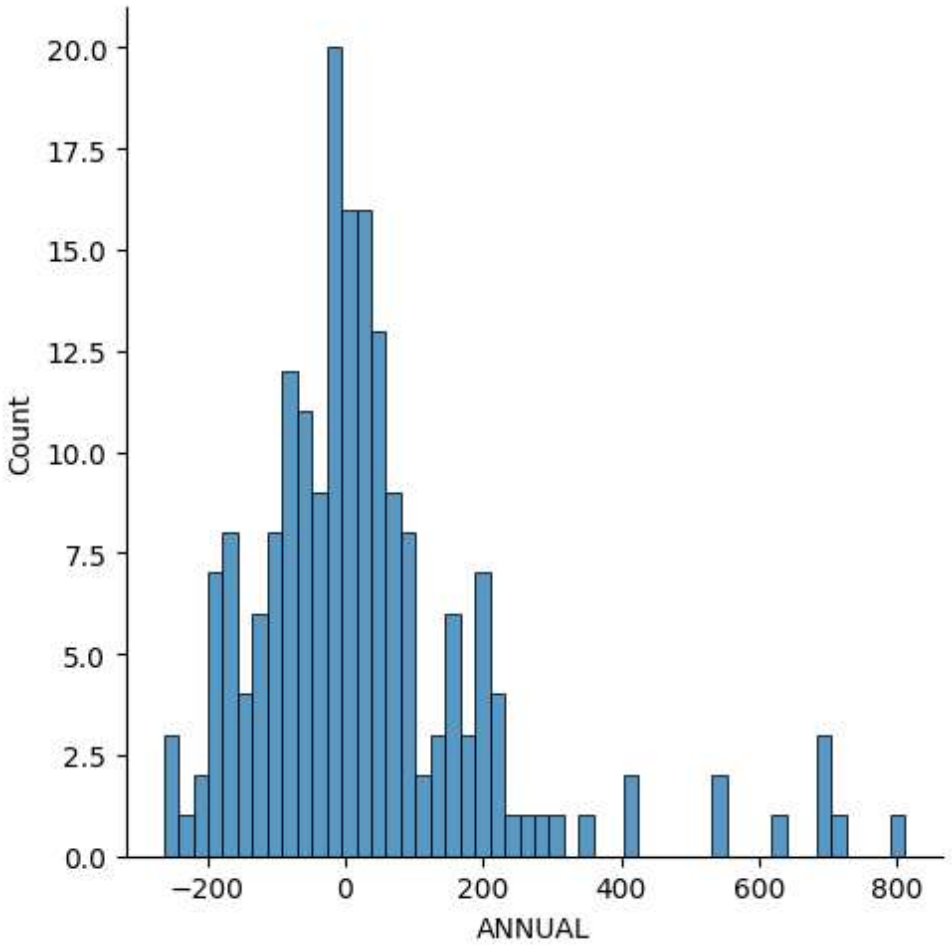
	coefficient
JAN	2.469164
FEB	0.681177
MAR	0.463948
APR	0.842979
MAY	1.923095
JUN	1.299172
JUL	1.620118

In [46]:
```
predictions=lm.predict(x_test)
plt.scatter(y_test,predictions)
```

Out[46]: <matplotlib.collections.PathCollection at 0x18e70db4af0>



In [47]:
```
sns.displot((y_test-predictions),bins=50);
```



In [48]:
```
from sklearn import metrics
print('MAE:',metrics.mean_absolute_error(y_test,predictions))
print('MSE:',metrics.mean_squared_error(y_test,predictions))
print('RMSE:',np.sqrt(metrics.mean_squared_error(y_test,predictions)))
```

```
MAE: 119.58133660913263
MSE: 33858.97884766022
RMSE: 184.00809451668212
```

In [ ]:
```

```