Sardar Patel Institute of Technology,Mumbai
Department of Electronics and Telecommunication Engineering
B.E. Sem-VII (2022-2023) Data Analytics

Experiment: Exploratory Data Analysis (EDA)

Name: Rahul Alshi **UID:** 2019110001 **BE ETRX** **DA LAB 4**

Aim: Perform Regression on a Dataset in Python

Dataset Overview

The dataset 'Weather Data in India from 1901 to 2017.csv' contains 13 columns with information on weather in India from 1901 to 2017 in 12 monthe of an year.:

Code :

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from sklearn import metrics
import warnings
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
warnings.filterwarnings('ignore')
%matplotlib inline
```

In [2]:
```
df = pd.read_csv('Weather Data in India from 1901 to 2017.csv')
```

In [3]:
```
df = df.drop(['JAN', 'FEB', 'MAR'], axis=1)
df.dropna()
df.reset_index(inplace=True, drop=True)
df = df[df['APR']>0]
df
```

Out[3]:

	Unnamed: 0	YEAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
0	0	1901	26.41	28.28	28.60	27.49	26.98	26.26	25.08	21.73	18.95
1	1	1902	26.54	28.68	28.44	27.29	27.05	25.95	24.37	21.33	18.78
2	2	1903	26.03	27.93	28.41	28.04	26.63	26.34	24.57	20.96	18.29
3	3	1904	26.73	27.83	27.85	26.84	26.73	25.84	24.36	21.07	18.84

In [2]:
```
df = pd.read_csv('Weather Data in India from 1901 to 2017.csv')
```

In [3]:
```
df = df.drop(['JAN', 'FEB', 'MAR'], axis=1)
df.dropna()
df.reset_index(inplace=True, drop=True)
df = df[df['APR']>0]
df
```

Out[3]:

	Unnamed: 0	YEAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
0	0	1901	26.41	28.28	28.60	27.49	26.98	26.26	25.08	21.73	18.95
1	1	1902	26.54	28.68	28.44	27.29	27.05	25.95	24.37	21.33	18.78
2	2	1903	26.03	27.93	28.41	28.04	26.63	26.34	24.57	20.96	18.29
3	3	1904	26.73	27.83	27.85	26.84	26.73	25.84	24.36	21.07	18.84
4	4	1905	24.84	28.32	28.69	27.67	27.47	26.29	26.16	22.07	18.71
...	...	...	...	...	...	...	...	...	...	...	...
112	112	2013	26.97	29.06	28.24	27.50	27.22	26.87	25.63	22.18	19.69
113	113	2014	26.91	28.45	29.42	28.07	27.42	26.61	25.38	22.53	19.50
114	114	2015	26.52	28.82	28.15	28.03	27.64	27.04	25.82	22.95	20.21
115	115	2016	29.56	30.41	29.70	28.18	28.17	27.72	26.81	23.90	21.89
116	116	2017	29.17	30.47	29.44	28.31	28.12	28.11	27.24	23.92	21.47

117 rows × 11 columns

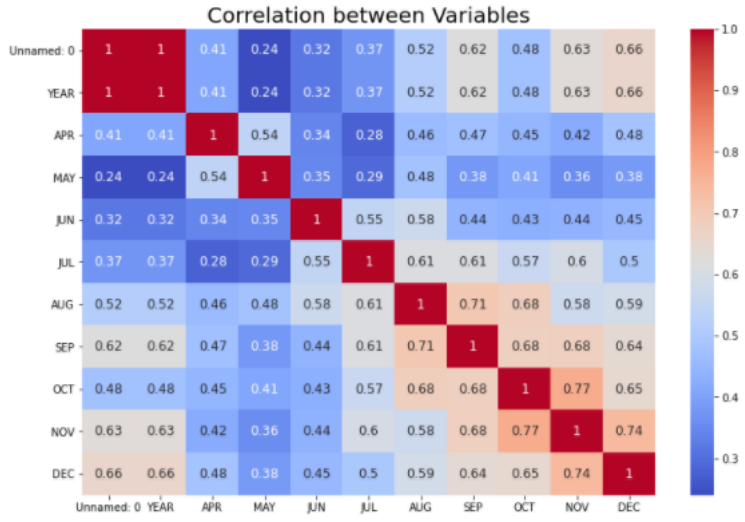
In [4]:
```
df.describe()
```

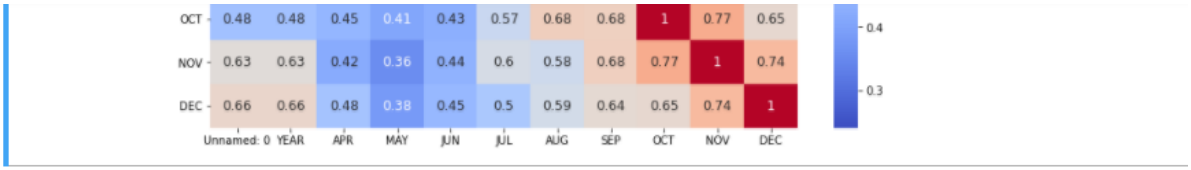
Out[4]:

	Unnamed: 0	YEAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
count	117.000000	117.000000	117.000000	117.000000	117.000000	117.000000	117.000000	117.000000	117.000000	117.000000	117.000000
mean	58.000000	1959.000000	26.514103	28.386410	28.300940	27.369231	26.940085	26.342650	24.742051	21.765726	19.173333
std	33.919021	33.919021	0.750740	0.644878	0.460603	0.345920	0.348876	0.387789	0.563152	0.634183	0.635912
min	0.000000	1901.000000	24.840000	26.970000	27.330000	26.480000	26.210000	25.470000	23.520000	20.590000	17.980000
25%	29.000000	1930.000000	26.000000	27.950000	28.020000	27.150000	26.730000	26.110000	24.390000	21.320000	18.780000

In [14]:
```
corr = df.corr()
fig, ax = plt.subplots(figsize = (12,8))
g= sns.heatmap(corr,ax=ax, annot= True, cmap='coolwarm', annot_kws={'size':12}) # or greys, coolwarm
ax.set_title('Correlation between Variables', size= 20)
```

Out[14]: Text(0.5, 1.0, 'Correlation between Variables')





In [6]:
```
df.dropna()
```

Out[6]:

	Unnamed: 0	YEAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
0	0	1901	26.41	28.28	28.60	27.49	26.98	26.26	25.08	21.73	18.95
1	1	1902	26.54	28.68	28.44	27.29	27.05	25.95	24.37	21.33	18.78
2	2	1903	26.03	27.93	28.41	28.04	26.63	26.34	24.57	20.96	18.29
3	3	1904	26.73	27.83	27.85	26.84	26.73	25.84	24.36	21.07	18.84
4	4	1905	24.84	28.32	28.69	27.67	27.47	26.29	26.16	22.07	18.71
...	...	...	...	...	...	...	...	...	...	...	...
112	112	2013	26.97	29.06	28.24	27.50	27.22	26.87	25.63	22.18	19.69
113	113	2014	26.91	28.45	29.42	28.07	27.42	26.61	25.38	22.53	19.50
114	114	2015	26.52	28.82	28.15	28.03	27.64	27.04	25.82	22.95	20.21
115	115	2016	29.56	30.41	29.70	28.18	28.17	27.72	26.81	23.90	21.89
116	116	2017	29.17	30.47	29.44	28.31	28.12	28.11	27.24	23.92	21.47

117 rows × 11 columns

In [8]:
```
x = df[df.columns[df.columns != 'MAY']]
y = df.MAY

# Statsmodels.OLS requires us to add a constant.
x = sm.add_constant(x)
model = sm.OLS(y,x)
results = model.fit()
print(results.summary())
```

In [9]:
```
df = df.dropna()
```

In [11]:
```
x.drop(columns = ['','JUN', 'JUL'],axis=1, inplace=True)
model = sm.OLS(y,x)
results = model.fit()
print(results.summary())
```

In [12]:
```
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))
```

```
0.4541275652917467
```

In [13]:
```
coefficients = pd.concat([pd.DataFrame(x.columns),pd.DataFrame(np.transpose(regr.coef_))], axis = 1)
coefficients
```

Output :

OLS Regression Results

Dep. Variable:	MAY	R-squared:	0.379
Model:	OLS	Adj. R-squared:	0.326
Method:	Least Squares	F-statistic:	7.246
Date:	Tue, 29 Nov 2022	Prob (F-statistic):	3.62e-08
Time:	22:10:48	Log-Likelihood:	-86.308
No. Observations:	117	AIC:	192.6
Df Residuals:	107	BIC:	220.2
Df Model:	9		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	2.919e-06	2.65e-06	1.103	0.272	-2.33e-06	8.16e-06
Unnamed: 0	-0.0043	0.003	-1.608	0.111	-0.010	0.001
YEAR	0.0012	0.003	0.435	0.664	-0.004	0.007
APR	0.3499	0.079	4.418	0.000	0.193	0.507
JUN	0.0732	0.139	0.525	0.601	-0.203	0.350
JUL	-0.0784	0.210	-0.374	0.709	-0.494	0.338
AUG	0.5404	0.243	2.223	0.028	0.059	1.022
SEP	-0.0366	0.221	-0.166	0.868	-0.474	0.401
OCT	0.0184	0.161	0.115	0.909	-0.300	0.337
NOV	0.0807	0.155	0.522	0.603	-0.226	0.387
DEC	0.0659	0.130	0.507	0.613	-0.192	0.324

Omnibus:	23.113	Durbin-Watson:	1.829
Prob(Omnibus):	0.000	Jarque-Bera (JB):	53.091
Skew:	0.746	Prob(JB):	2.96e-12
Kurtosis:	5.943	Cond. No.	8.17e+18

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 6.74e-30. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

OLS Regression Results

Dep. Variable:	MAY	R-squared:	0.256
Model:	OLS	Adj. R-squared:	0.216
Method:	Least Squares	F-statistic:	6.313
Date:	Tue, 29 Nov 2022	Prob (F-statistic):	9.93e-06
Time:	22:11:56	Log-Likelihood:	-96.839
No. Observations:	117	AIC:	207.7
Df Residuals:	110	BIC:	227.0
Df Model:	6		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	3.193e-06	2.54e-06	1.259	0.211	-1.83e-06	8.22e-06
Unnamed: 0	-0.0042	0.003	-1.678	0.096	-0.009	0.001
YEAR	0.0018	0.003	0.651	0.516	-0.004	0.007
AUG	0.6677	0.238	2.806	0.006	0.196	1.139
SEP	0.0434	0.230	0.189	0.851	-0.412	0.499
OCT	0.0727	0.172	0.422	0.674	-0.269	0.414
NOV	0.0428	0.161	0.266	0.791	-0.276	0.362
DEC	0.1668	0.138	1.211	0.228	-0.106	0.440

Omnibus:	26.482	Durbin-Watson:	1.946
Prob(Omnibus):	0.000	Jarque-Bera (JB):	66.606
Skew:	0.828	Prob(JB):	3.44e-15
Kurtosis:	6.305	Cond. No.	8.15e+18

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 6.77e-30. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

```
In [13]: coefficients = pd.concat([pd.DataFrame(x.columns),pd.DataFrame(np.transpose(regr.coef_))], axis = 1)
         coefficients
```

Out[13]:

	0	0
0	const	0.000000
1	Unnamed: 0	-0.001163
2	YEAR	-0.001163
3	AUG	0.626586
4	SEP	0.069522
5	OCT	0.003085
6	NOV	0.021472
7	DEC	0.193724

Conclusion:

1. Performed Regression analysis in Python for house classifier dataset.
2. Regression analysis is a statistical method to model the relationship between a dependent (target) and independent (predictor) variables with one or more independent variables.Regression analysis helps us to understand how the value of the dependent variable is changing corresponding to an independent variable when other independent variables are held fixed
3. The regression coefficient for the month of august month in the data set is calculated out to be aroung 0.62 whereas for other attributes it is approximately close to 0.
4. Few insights we found from the dataset:
    - The regression score calculated for the data set is found out to be 0.45
    - The highest r squared value obtained from the regression analysis is highest for the MAY attribute with 0.379