Koneru Lakshmaiah Education Foundation

(Deemed to be University estd. u/s. 3 of UGC Act,1956)

Off Campus: R.V.S NAGAR, Moinabad Road, Near TS Police Academy,
Aziz Nagar (PO), Hyderabad , Telangana - 500075.

DATA ANALYTICS AND VISUALIZATION

Lab Manual

Course Title	DATA ANALYTICS AND VISUALIZATION
Course Code	23SDAO1E
L-T-P-S Structure	0-0-6-4
Credits	4

Name: Dhruv Nair

Roll No. : 2310080001

Index

1. **Plotting different Python modules and reading data of different formats**

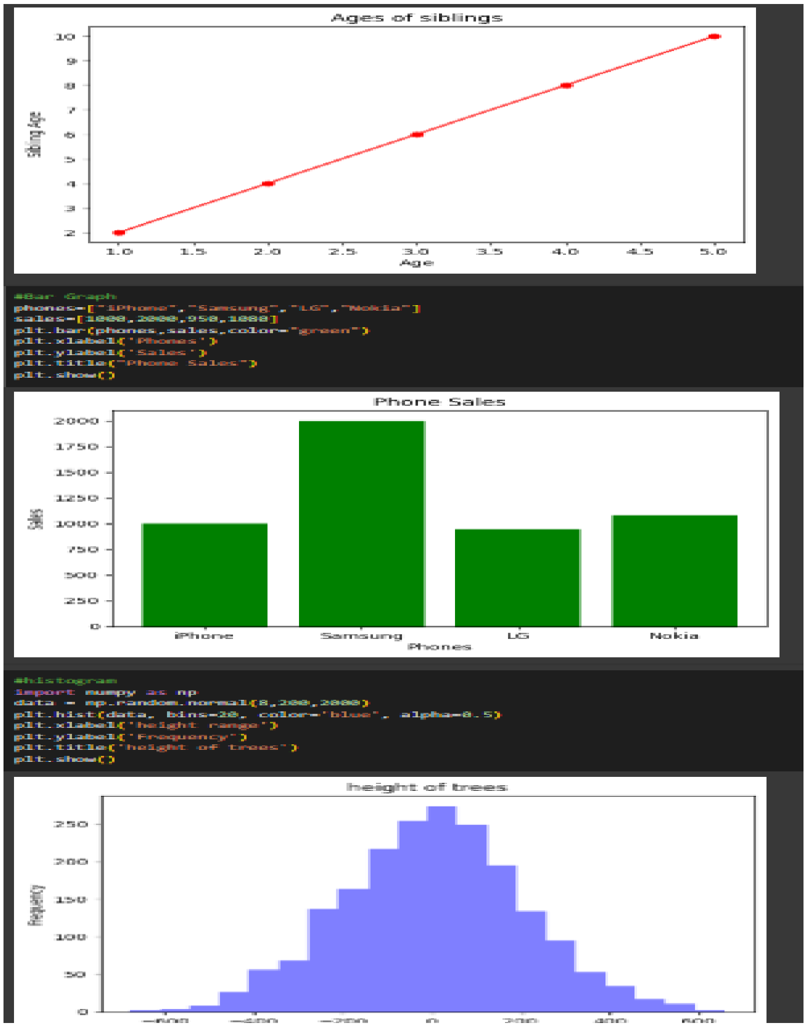
Aim: To explore and visualize data using different Python libraries and to read data from various formats.

Objective: To utilize Python modules like `pandas`, `matplotlib`, and `seaborn` for data visualization and handling different file formats.

Code:

Colab Notebook Link: Graphs.ipynb

Output:



```
#Bar Graph
phones=["iPhone","Samsung","LG","Nokia"]
sales=[1000,2000,950,1080]
plt.bar(phones,sales,color="green")
plt.xlabel('Phones')
plt.ylabel('Sales')
plt.title("Phone Sales")
plt.show()
```

```
#histogram
import numpy as np
data = np.random.normal(0,200,2000)
plt.hist(data, bins=20, color='blue', alpha=0.5)
plt.xlabel('height range')
plt.ylabel('Frequency')
plt.title('height of trees')
plt.show()
```

Result:

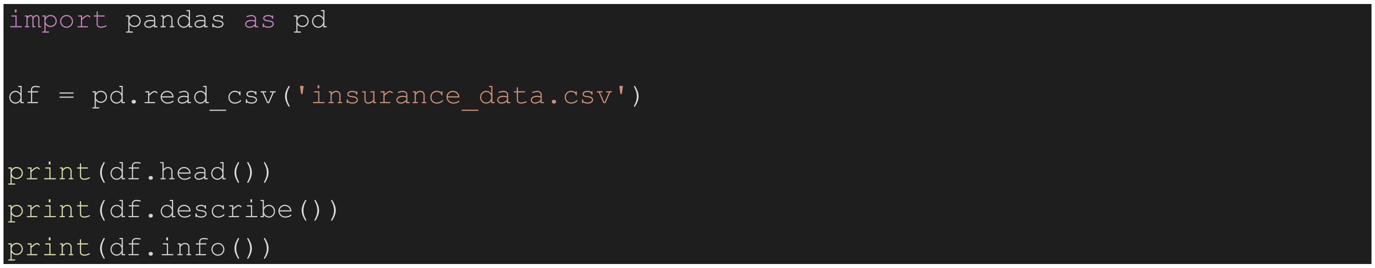
Different data formats were successfully read and visualized using Python libraries.

2. **Initial data exploration using Python**

Aim: To explore the structure of a dataset through initial data analysis.

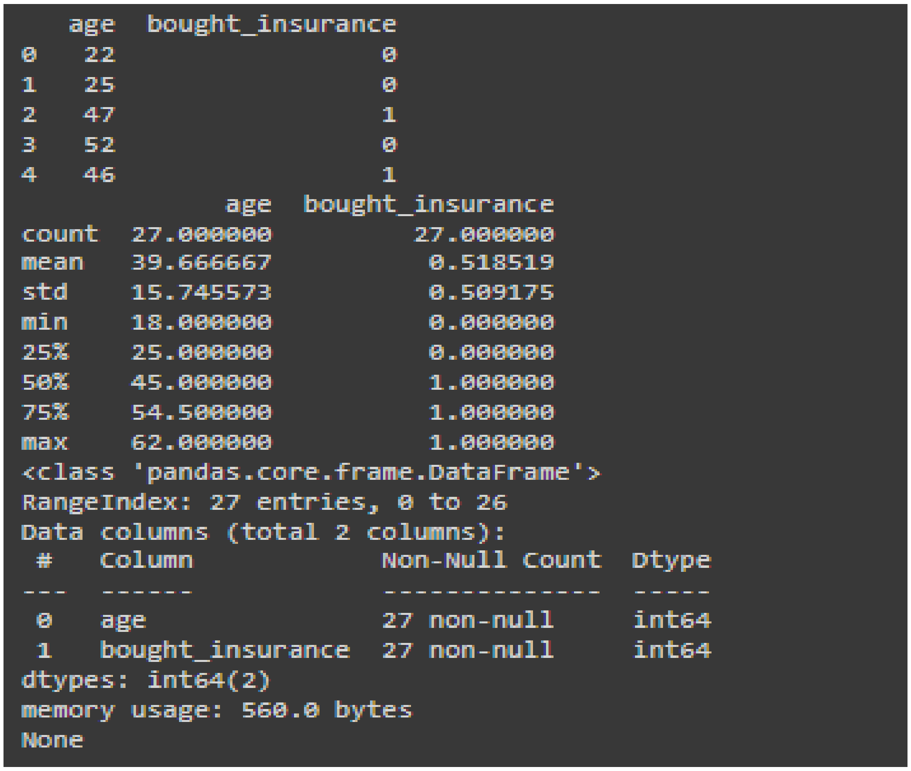
Objective: Understand basic statistics, distribution, and structure of a dataset using Python.

Code:



```python
import pandas as pd

df = pd.read_csv('insurance_data.csv')

print(df.head())
print(df.describe())
print(df.info())
```

Output:



```
   age  bought_insurance
0   22                 0
1   25                 0
2   47                 1
3   52                 0
4   46                 1
             age  bought_insurance
count  27.000000         27.000000
mean   39.666667          0.518519
std    15.745573          0.509175
min    18.000000          0.000000
25%    25.000000          0.000000
50%    45.000000          1.000000
75%    54.500000          1.000000
max    62.000000          1.000000
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27 entries, 0 to 26
Data columns (total 2 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   age               27 non-null     int64
 1   bought_insurance  27 non-null     int64
dtypes: int64(2)
memory usage: 560.0 bytes
None
```

Result: Different data formats were successfully read and visualized using Python libraries.

3. **Identifying and imputing missing values in the dataset**

Aim: To identify and impute missing values in a dataset.

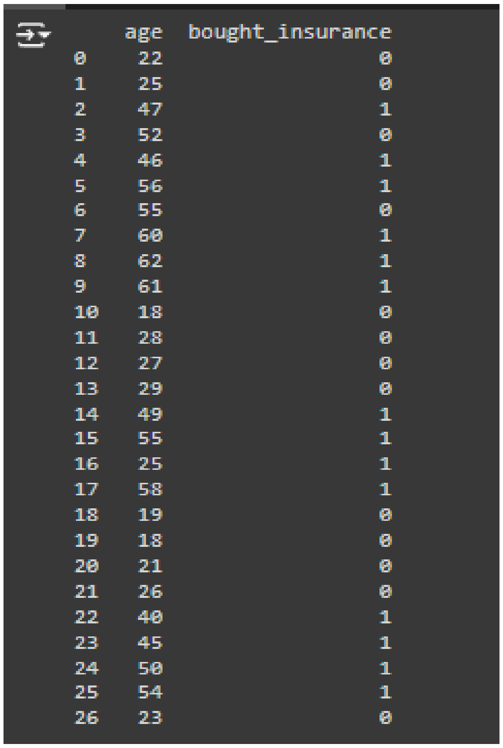
Objective: Learn how to detect missing values and fill them using appropriate imputation techniques.

Code:

Colab Notebook Link: DAV.ipynb

```
import pandas as pd

df = pd.read_csv('insurance_data.csv')
missing = df.isnull().sum()

df.fillna(df.mean(), inplace=True)
print(df)
```

Output:



```
    age  bought_insurance
0    22                 0
1    25                 0
2    47                 1
3    52                 0
4    46                 1
5    56                 1
6    55                 0
7    60                 1
8    62                 1
9    61                 1
10   18                 0
11   28                 0
12   27                 0
13   29                 0
14   49                 1
15   55                 1
16   25                 1
17   58                 1
18   19                 0
19   18                 0
20   21                 0
21   26                 0
22   40                 1
23   45                 1
24   50                 1
25   54                 1
26   23                 0
```

Result:

Missing values were identified and imputed successfully using various techniques.

4. **Detection and smoothening of outliers in the dataset**

Aim: To detect and handle outliers in a dataset.

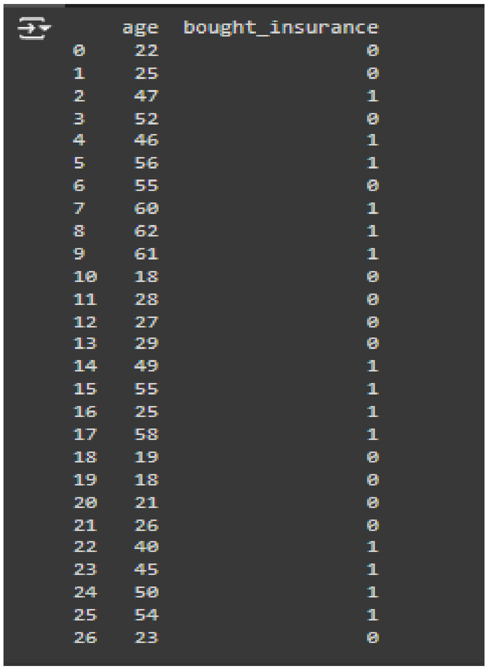
Objective: Identify outliers using Z-scores or the IQR method and smooth them.

Code:

Colab Notebook Link: DAV.ipynb

```
import pandas as pd
import numpy as np

df = pd.read_csv('insurance_data.csv')

Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1

outliers = (df < (Q1 - 1.5 * IQR)) | (df > (Q3 + 1.5 * IQR))
df[outliers] = np.nan
df.fillna(df.mean(), inplace=True)
print(df)
```

Output:



```
    age  bought_insurance
0    22                 0
1    25                 0
2    47                 1
3    52                 0
4    46                 1
5    56                 1
6    55                 0
7    60                 1
8    62                 1
9    61                 1
10   18                 0
11   28                 0
12   27                 0
13   29                 0
14   49                 1
15   55                 1
16   25                 1
17   58                 1
18   19                 0
19   18                 0
20   21                 0
21   26                 0
22   40                 1
23   45                 1
24   50                 1
25   54                 1
26   23                 0
```

Result:

Outliers were successfully identified and smoothed using the appropriate methods

.

5. **Implementing data transformations on temperature dataset**

Aim: To apply transformations on a temperature dataset to make it suitable for analysis.

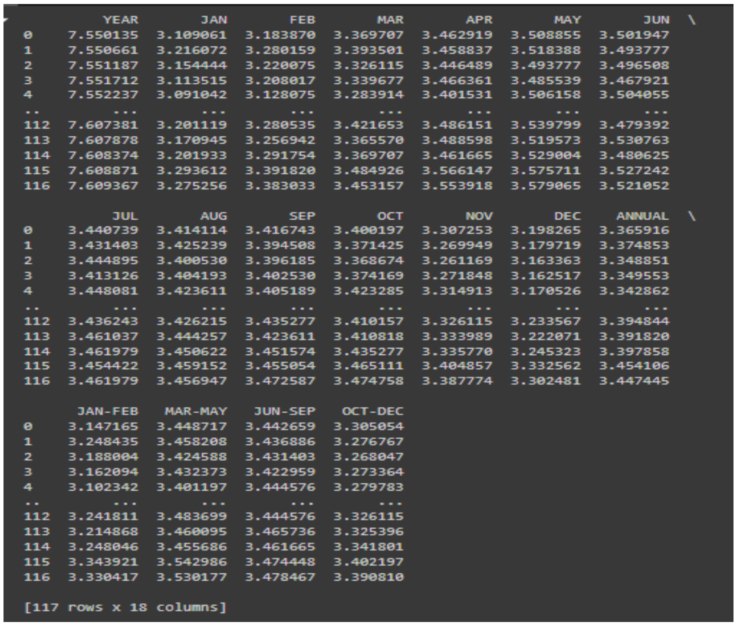
Objective: Learn different transformation techniques such as normalization, standardization, and log transformation.

Code:

Colab Notebook Link: DAV.ipynb

```
from sklearn.preprocessing import StandardScaler, MinMaxScaler
import pandas as pd
import numpy as np

df = pd.read_csv('temperatures.csv')

scaler = StandardScaler()
df_standardized = scaler.fit_transform(df)

minmax_scaler = MinMaxScaler()
df_normalized = minmax_scaler.fit_transform(df)

df_log_transformed = np.log(df)
print(df_log_transformed)
```

Output:



```
         YEAR       JAN       FEB       MAR       APR       MAY       JUN  \
0    7.550135  3.109061  3.183870  3.369707  3.462919  3.508855  3.501947
1    7.550661  3.216072  3.280159  3.393501  3.458837  3.518388  3.493777
2    7.551187  3.154444  3.220075  3.326115  3.446489  3.493777  3.496508
3    7.551712  3.113515  3.208017  3.339677  3.466361  3.485539  3.467921
4    7.552237  3.091042  3.128075  3.283914  3.401531  3.506158  3.504055
..        ...       ...       ...       ...       ...       ...       ...
112  7.607381  3.201119  3.280535  3.421653  3.486151  3.539799  3.479392
113  7.607878  3.170945  3.256942  3.365570  3.488598  3.519573  3.530763
114  7.608374  3.201933  3.291754  3.369707  3.461665  3.529004  3.480625
115  7.608871  3.293612  3.391820  3.484926  3.566147  3.575711  3.527242
116  7.609367  3.275256  3.383033  3.453157  3.553918  3.579065  3.521052

          JUL       AUG       SEP       OCT       NOV       DEC    ANNUAL  \
0    3.440739  3.414114  3.416743  3.400197  3.307253  3.198265  3.365916
1    3.431403  3.425239  3.394508  3.371425  3.269949  3.179719  3.374853
2    3.444895  3.400530  3.396185  3.368674  3.261169  3.163363  3.348851
3    3.413126  3.404193  3.402530  3.374169  3.271848  3.162517  3.349553
4    3.448081  3.423611  3.405189  3.423285  3.314913  3.170526  3.342862
..        ...       ...       ...       ...       ...       ...       ...
112  3.436243  3.426215  3.435277  3.410157  3.326115  3.233567  3.394844
113  3.461037  3.444257  3.423611  3.410818  3.333989  3.222071  3.391820
114  3.461979  3.450622  3.451574  3.435277  3.335770  3.245323  3.397858
115  3.454422  3.459152  3.455054  3.465111  3.404857  3.332562  3.454106
116  3.461979  3.456947  3.472587  3.474758  3.387774  3.302481  3.447445

      JAN-FEB   MAR-MAY   JUN-SEP   OCT-DEC
0    3.147165  3.448717  3.442659  3.305054
1    3.248435  3.458208  3.436886  3.276767
2    3.188004  3.424588  3.431403  3.268047
3    3.162094  3.432373  3.422959  3.273364
4    3.102342  3.401197  3.444576  3.279783
..        ...       ...       ...       ...
112  3.241811  3.483699  3.444576  3.326115
113  3.214868  3.460095  3.465736  3.325396
114  3.248046  3.455686  3.461665  3.341801
115  3.343921  3.542986  3.474448  3.402197
116  3.330417  3.530177  3.478467  3.390810

[117 rows x 18 columns]
```

Result:

Data transformations were successfully applied, improving the dataset's usability for further analysis

6. **Building Part to Whole Charts using Tableau**

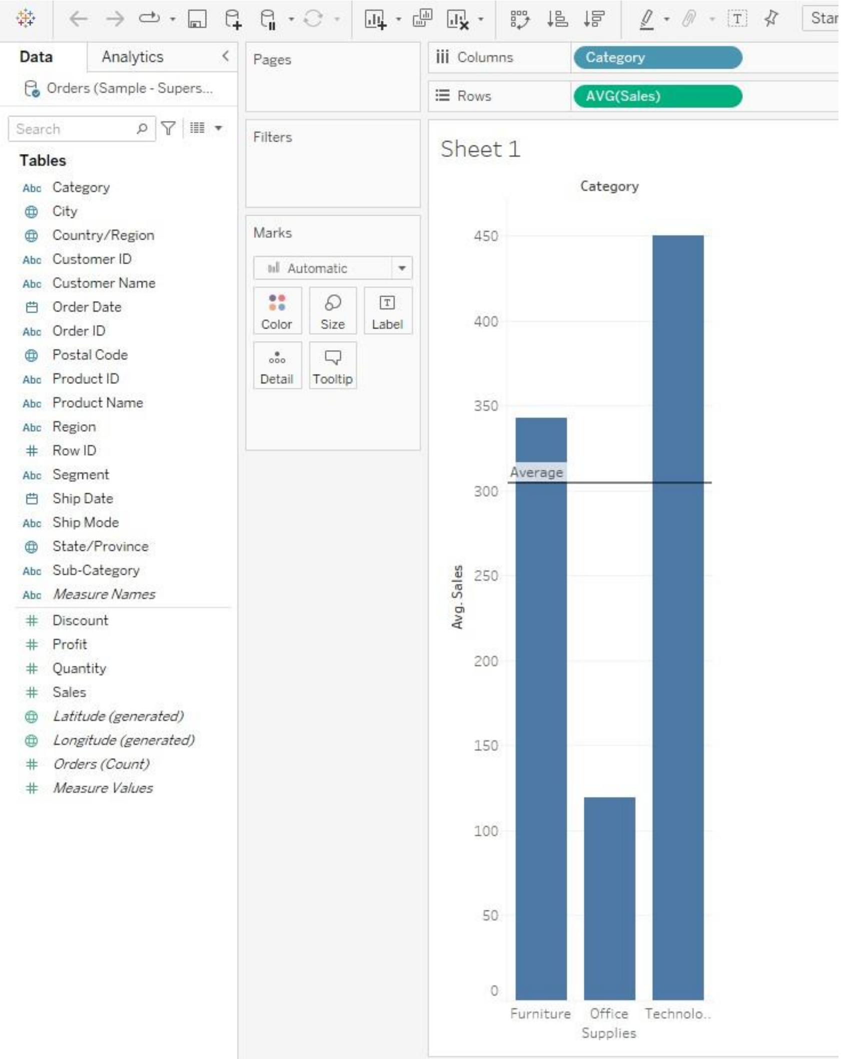
Aim: To create part-to-whole visualizations using Tableau.

Objective: Learn to visualize proportions and relationships in data using pie charts, stacked bar charts, etc

Code:

Colab Notebook Link: Graphs.ipynb

Output:



Columns: Category

Rows: AVG(Sales)

Result:

Data was successfully visualized in terms of proportions and parts-to-whole charts

7. **Building Correlation Charts using Python and Tableau**

Aim: To create correlation charts to visualize relationships between variables.

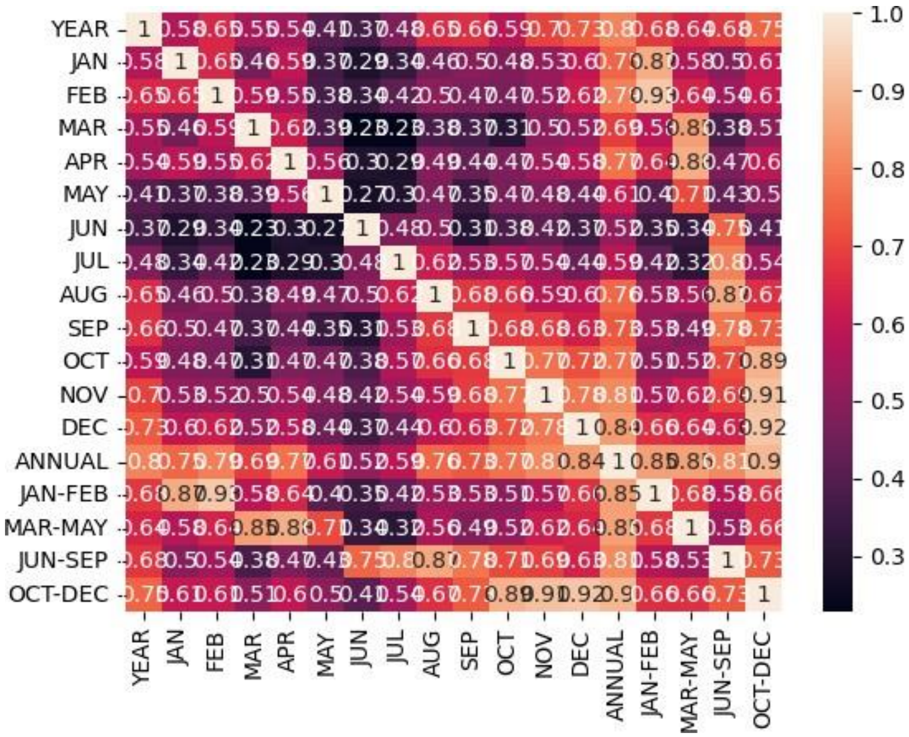
Objective: Understand the relationship between variables using correlation matrices.

Code:

Colab Notebook Link: DAV.ipynb

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv('temperatures.csv')
correlation_matrix = df.corr()

sns.heatmap(correlation_matrix, annot=True)
plt.show()
```

Output:



Result:

Correlations between variables were effectively visualized.

8. **Measuring Data Similarity and Dissimilarity using both tools**

Aim: To measure data similarity and dissimilarity using Python and Tableau.

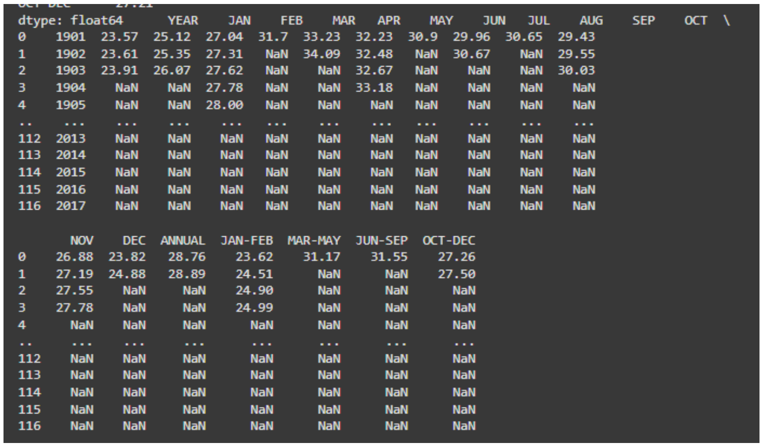
Objective:Calculate similarity and dissimilarity using metrics like Euclidean distance and Cosine similarity..

Code:

Colab Notebook Link: DAV.ipynb

```
from scipy.spatial.distance import euclidean, cosine
import pandas as pd

df = pd.read_csv('temperatures.csv')

similarity = cosine(df.iloc[0], df.iloc[1])
dissimilarity = euclidean(df.iloc[0], df.iloc[1])
print(similarity, dissimilarity)
```

Output:



```
OCT-DEC       27.21
dtype: float64      YEAR    JAN    FEB    MAR   APR    MAY    JUN   JUL    AUG    SEP    OCT  \
0    1901  23.57  25.12  27.04  31.7  33.23  32.23  30.9  29.96  30.65  29.43
1    1902  23.61  25.35  27.31   NaN  34.09  32.48   NaN  30.67    NaN  29.55
2    1903  23.91  26.07  27.62   NaN    NaN  32.67   NaN    NaN    NaN  30.03
3    1904    NaN    NaN  27.78   NaN    NaN  33.18   NaN    NaN    NaN    NaN
4    1905    NaN    NaN  28.00   NaN    NaN    NaN   NaN    NaN    NaN    NaN
..    ...    ...    ...    ...   ...    ...    ...   ...    ...    ...    ...
112  2013    NaN    NaN    NaN   NaN    NaN    NaN   NaN    NaN    NaN    NaN
113  2014    NaN    NaN    NaN   NaN    NaN    NaN   NaN    NaN    NaN    NaN
114  2015    NaN    NaN    NaN   NaN    NaN    NaN   NaN    NaN    NaN    NaN
115  2016    NaN    NaN    NaN   NaN    NaN    NaN   NaN    NaN    NaN    NaN
116  2017    NaN    NaN    NaN   NaN    NaN    NaN   NaN    NaN    NaN    NaN

       NOV    DEC  ANNUAL  JAN-FEB  MAR-MAY  JUN-SEP  OCT-DEC
0    26.88  23.82   28.76    23.62    31.17    31.55    27.26
1    27.19  24.88   28.89    24.51      NaN      NaN    27.50
2    27.55    NaN     NaN    24.90      NaN      NaN      NaN
3    27.78    NaN     NaN    24.99      NaN      NaN      NaN
4      NaN    NaN     NaN      NaN      NaN      NaN      NaN
..     ...    ...     ...      ...      ...      ...      ...
112    NaN    NaN     NaN      NaN      NaN      NaN      NaN
113    NaN    NaN     NaN      NaN      NaN      NaN      NaN
114    NaN    NaN     NaN      NaN      NaN      NaN      NaN
115    NaN    NaN     NaN      NaN      NaN      NaN      NaN
116    NaN    NaN     NaN      NaN      NaN      NaN      NaN
```

Result:

Different data formats were successfully read and visualized using Python libraries.

9. **Plotting different Python modules and reading data of different formats**

Aim: To compute central tendency (mean, median, mode), variance, and moments (skewness, kurtosis) for a dataset.

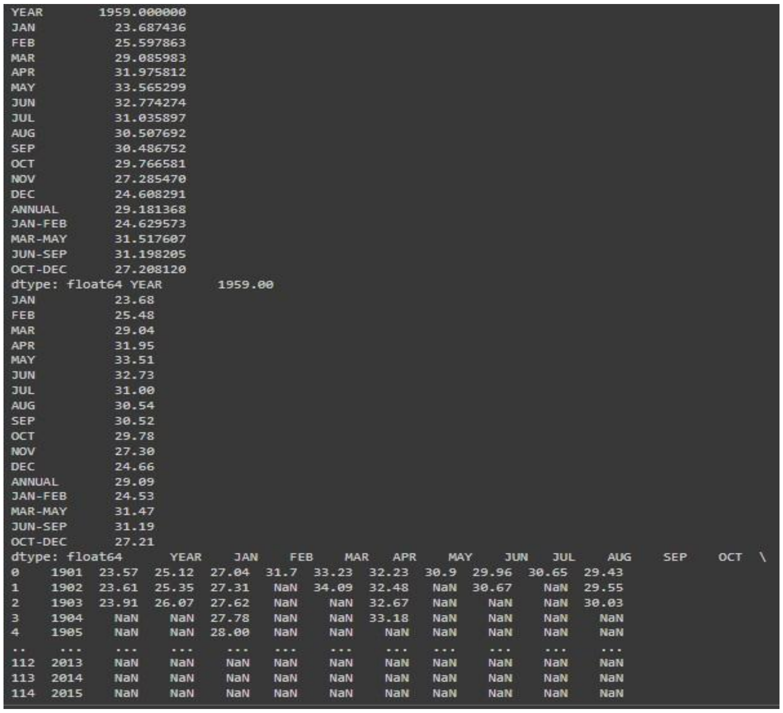
Objective: Summarize the dataset using statistical measures.

Code:

Colab Notebook Link: DAV.ipynb

```
import pandas as pd

df = pd.read_csv('temperatures.csv')

mean = df.mean()
median = df.median()
mode = df.mode()
variance = df.var()
skewness = df.skew()
kurtosis = df.kurt()
print(mean, median, mode, variance, skewness, kurtosis)
```

Output:



```
YEAR       1959.000000
JAN          23.687436
FEB          25.597863
MAR          29.085983
APR          31.975812
MAY          33.565299
JUN          32.774274
JUL          31.035897
AUG          30.507692
SEP          30.486752
OCT          29.766581
NOV          27.285470
DEC          24.608291
ANNUAL       29.181368
JAN-FEB      24.629573
MAR-MAY      31.517607
JUN-SEP      31.198205
OCT-DEC      27.208120
dtype: float64 YEAR       1959.00
JAN          23.68
FEB          25.48
MAR          29.04
APR          31.95
MAY          33.51
JUN          32.73
JUL          31.00
AUG          30.54
SEP          30.52
OCT          29.78
NOV          27.30
DEC          24.66
ANNUAL       29.09
JAN-FEB      24.53
MAR-MAY      31.47
JUN-SEP      31.19
OCT-DEC      27.21
dtype: float64      YEAR    JAN    FEB    MAR   APR    MAY    JUN   JUL    AUG    SEP    OCT  \
0    1901  23.57  25.12  27.04  31.7  33.23  32.23  30.9  29.96  30.65  29.43
1    1902  23.61  25.35  27.31   NaN  34.09  32.48   NaN  30.67    NaN  29.55
2    1903  23.91  26.07  27.62   NaN    NaN  32.67   NaN    NaN    NaN  30.03
3    1904    NaN    NaN  27.78   NaN    NaN  33.18   NaN    NaN    NaN    NaN
4    1905    NaN    NaN  28.00   NaN    NaN    NaN   NaN    NaN    NaN    NaN
..    ...    ...    ...    ...   ...    ...    ...   ...    ...    ...    ...
112  2013    NaN    NaN    NaN   NaN    NaN    NaN   NaN    NaN    NaN    NaN
113  2014    NaN    NaN    NaN   NaN    NaN    NaN   NaN    NaN    NaN    NaN
114  2015    NaN    NaN    NaN   NaN    NaN    NaN   NaN    NaN    NaN    NaN
```

Result:

Statistical measures provided a comprehensive summary of the dataset.

10. **Data classification (4 classifications) Logistic Regression using Python modules**

Aim: To classify data into four categories using Logistic Regression in Python.

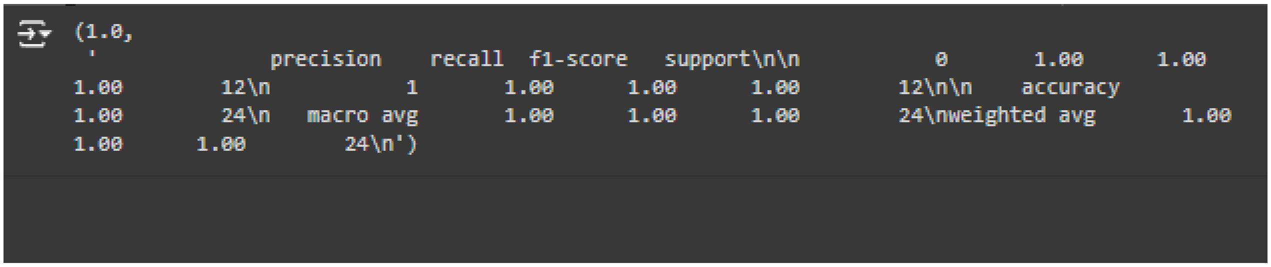
Objective: Understand multi-class classification and implement Logistic Regression for classifying data.

Code:

Colab Notebook Link: DAV.ipynb

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report

data = pd.read_csv("temperatures.csv")
median_temp = data['ANNUAL'].median()
data['HighTemp'] = (data['ANNUAL'] > median_temp).astype(int)

X = data[['JAN', 'FEB', 'MAR', 'APR', 'MAY', 'JUN', 'JUL', 'AUG', 'SEP', 'OCT',
'NOV', 'DEC']]
y = data['HighTemp']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
model = LogisticRegression(max_iter=200)
model.fit(X_train, y_train)

y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)

accuracy, report
```

Output:



```
(1.0,
 '              precision    recall  f1-score   support\n\n           0       1.00      1.00
 1.00        12\n           1       1.00      1.00      1.00        12\n\n    accuracy
 1.00        24\n   macro avg       1.00      1.00      1.00        24\nweighted avg       1.00
 1.00      1.00        24\n')
```

Result:

Different data formats were successfully read and visualized using Python libraries.