Experiment 1: Exploratory data analysis using Python

Aim: To understand the data through Exploratory data analysis

- Data cleaning- Missing Values, remove outliers
- Data transformation- Min-max normalization, Z-score normalization, Decimal Scaling
- Data Discretization- Binning
- Data analysis and Visualization

Theory:

Data Cleaning:

Data cleaning, also known as data cleansing or data scrubbing, is a crucial step in the data preparation process of any data analysis or machine learning project. It involves identifying and correcting errors, inconsistencies, and inaccuracies in datasets to ensure that the data is accurate, reliable, and ready for analysis.

Some common problems found in datasets are:

- Missing Values
- Outliers
- Inconsistent Formats, etc.

Some functions used for data cleaning:

- dropna()
- fillna()
- drop_duplicates()

Data transformation:

Data transformation refers to the process of converting or altering the raw data in a way that makes it more suitable for analysis, modeling, or visualization. It involves applying various mathematical, statistical, or logical operations to the data to achieve specific objectives, such as improving data quality, normalizing scales, handling outliers, or making the data conform to assumptions required by certain analytical methods.

Some of the techniques involved in data transformation are:

- Normalization
- Standardization
- Min-Max normalization
- Decimal Scaling

Data discretization:

Data discretization, also known as binning or discretization, is the process of converting continuous or numeric data into discrete bins or intervals. In other words, it involves dividing a continuous variable's range into smaller, non-overlapping intervals and assigning data points to the corresponding interval. This transformation is particularly useful when working

with data analysis, visualization, or certain types of machine learning algorithms that benefit from reduced data granularity or when data is naturally presented in grouped or categorized form.

Some functions that help us perform data discretization:

- pandas.qcut
- pandas.cut

Data visualisation:

Data visualization is the graphical representation of information and data. It involves using visual elements like charts, graphs, and maps to present complex data in a more accessible and understandable format. Data visualization is a powerful tool for exploring, analyzing, and communicating insights from data, making it an essential part of data analysis

The data can be presented in various forms via data visualisation for a better understanding of the data and some of those forms are:

- Pie-chart
- Histogram
- Scatterplot, etc.

Data analysis:

Data analysis is the process of inspecting, cleaning, transforming, and interpreting data to extract meaningful insights, discover patterns, and make informed decisions. It involves using various techniques, tools, and methodologies to understand the underlying structure of data, identify trends, relationships, and anomalies, and derive actionable information from the data.

Some functions used for data analysis are:

- .head()
- .info()
- .describe()

Steps:

1) **Load the libraries Download the data set from kaggle/ other sources**


```
In [2]: import pandas as pd
```

2) Read the file –select appropriate file read function according to data type of file

```
In [2]: import pandas as pd

        df = pd.read_csv("ufc_master_data.csv")
        df.head()
```

Out[2]:

	ranking	name	age	weight	gender	height	ufc_wins	ufc_loses	ufc_draws	ufc_no_contests	mma_wins	mma_loses	mma_draws	mma_no_contests
0	C	Brandon Moreno	27	125	Male	67	8	2	2.0	NaN	19.0	5.0	2.0	NaN
1	1	Deiveson Figueiredo	33	125	Male	65	9	2	1.0	NaN	20.0	2.0	1.0	NaN
2	2	Askar Askarov	29	125	Male	65	3	0	1.0	NaN	14.0	0.0	1.0	NaN
3	3	Alexandre Pantoja	31	125	Male	65	8	3	NaN	NaN	24.0	5.0	NaN	NaN
4	4	Alex Perez	29	125	Male	64	6	2	NaN	NaN	24.0	6.0	NaN	NaN

3) Describe the attributes name, count no of values, and find min, max, data type, range, quartile, percentile, box plot and outliers.

Attribute names and data-type:

```
In [64]: attributes_info = df.dtypes

In [65]: attributes_info

Out[65]: ranking           object
         name              object
         age                int64
         weight             int64
         gender            object
         height             int64
         ufc_wins           int64
         ufc_loses          int64
         ufc_draws        float64
         ufc_no_contests  float64
         mma_wins         float64
         mma_loses        float64
         mma_draws        float64
         mma_no_contests  float64
         dtype: object
```

No. of values:

```
In [67]: count_values

Out[67]: ranking           177
         name              617
         age               617
         weight            617
         gender            617
         height            617
         ufc_wins          617
         ufc_loses         617
         ufc_draws          44
         ufc_no_contests    43
         mma_wins          615
         mma_loses         615
         mma_draws          82
         mma_no_contests    68
         dtype: int64
```

Min and Max:

```
In [70]: max_val = numeric_col.max()

In [71]: max_val

Out[71]: age                44.0
         weight            265.0
         height             79.0
         ufc_wins           29.0
         ufc_loses          18.0
         ufc_draws           2.0
         ufc_no_contests     2.0
         mma_wins           59.0
         mma_loses          21.0
         mma_draws           2.0
         mma_no_contests     2.0
         dtype: float64

In [72]: min_val = numeric_col.min()

In [73]: min_val

Out[73]: age                21.0
         weight            115.0
         height             59.0
         ufc_wins            0.0
         ufc_loses           0.0
         ufc_draws           0.0
         ufc_no_contests     1.0
         mma_wins            1.0
         mma_loses           0.0
         mma_draws           0.0
         mma_no_contests     1.0
         dtype: float64
```

Range:

```
In [74]: range = max_val - min_val

In [75]: range

Out[75]: age                23.0
         weight            150.0
         height             20.0
         ufc_wins           29.0
         ufc_loses          18.0
         ufc_draws           2.0
         ufc_no_contests     1.0
         mma_wins           58.0
         mma_loses          21.0
         mma_draws           2.0
         mma_no_contests     1.0
         dtype: float64
```

Quartile:

```
In [76]: quartiles = numeric_col.quantile([0.25, 0.50, 0.75,1.0])

In [77]: quartiles
```

Out[77]:

	age	weight	height	ufc_wins	ufc_loses	ufc_draws	ufc_no_contests	mma_wins	mma_loses	mma_draws	mma_no_contests
0.25	28.0	135.0	67.0	1.0	1.0	1.0	1.0	10.0	2.0	1.0	1.0
0.50	31.0	145.0	69.0	3.0	2.0	1.0	1.0	14.0	4.0	1.0	1.0
0.75	34.0	170.0	72.0	6.0	4.0	1.0	1.0	18.0	6.0	1.0	1.0
1.00	44.0	265.0	79.0	29.0	18.0	2.0	2.0	59.0	21.0	2.0	2.0

Percentile:

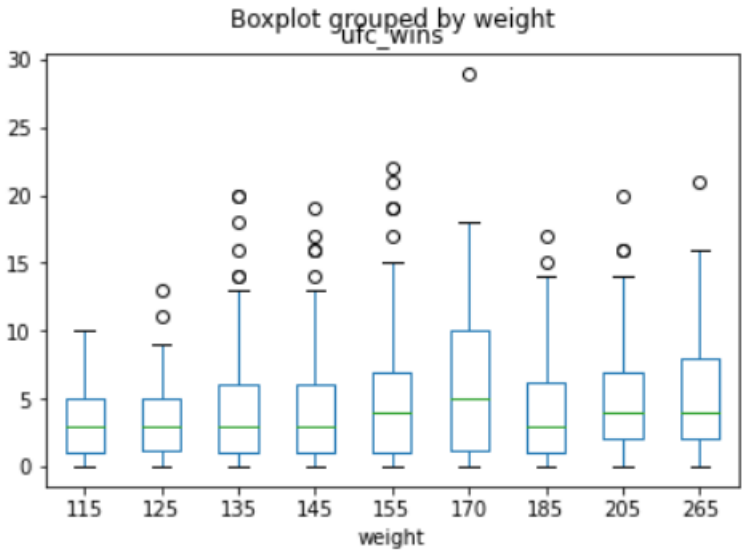
```
In [79]: percentiles = numeric_col.quantile([0.1,0.9])
         percentiles
```

Out[79]:

	age	weight	height	ufc_wins	ufc_loses	ufc_draws	ufc_no_contests	mma_wins	mma_loses	mma_draws	mma_no_contests
0.1	26.0	125.0	65.0	0.0	0.0	1.0	1.0	7.0	1.0	1.0	1.0
0.9	37.0	205.0	75.0	11.0	6.0	1.0	1.0	23.0	9.0	2.0	1.0

Boxplot and outliers:

```
In [80]: df.boxplot(by = 'weight', column = ['ufc_wins'],grid = False)
Out[80]: <AxesSubplot:title={'center':'ufc_wins'}, xlabel='weight'>
```



---

4) Perform cleaning,transformation , discretization and analysis

In [81]:
```
df.drop_duplicates(inplace = True)
df.fillna(df.mean(),inplace = True)
```

```
C:\Users\Komal\AppData\Local\Temp/ipykernel_7208/1661239573.py:2: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  df.fillna(df.mean(),inplace = True)
```

In [82]:
```
df.head()
```

Out[82]:

	ranking	name	age	weight	gender	height	ufc_wins	ufc_loses	ufc_draws	ufc_no_contests	mma_wins	mma_loses	mma_draws	mma_no_contests
0	C	Brandon Moreno	27	125	Male	67	8	2	2.0	1.046512	19.0	5.0	2.000000	1.088235
1	1	Deiveson Figueiredo	33	125	Male	65	9	2	1.0	1.046512	20.0	2.0	1.000000	1.088235
2	2	Askar Askarov	29	125	Male	65	3	0	1.0	1.046512	14.0	0.0	1.000000	1.088235
3	3	Alexandre Pantoja	31	125	Male	65	8	3	1.0	1.046512	24.0	5.0	1.097561	1.088235
4	4	Alex Perez	29	125	Male	64	6	2	1.0	1.046512	24.0	6.0	1.097561	1.088235

In [83]:
```
df['gender_encoded'] = df['gender'].map({'Male':0,'Female':1})
df.drop('gender', axis=1, inplace=True)
df.head()
```

Out[83]:

king	name	age	weight	height	ufc_wins	ufc_loses	ufc_draws	ufc_no_contests	mma_wins	mma_loses	mma_draws	mma_no_contests	gender_encoded
C	Brandon Moreno	27	125	67	8	2	2.0	1.046512	19.0	5.0	2.000000	1.088235	0
1	Deiveson Figueiredo	33	125	65	9	2	1.0	1.046512	20.0	2.0	1.000000	1.088235	0
2	Askar Askarov	29	125	65	3	0	1.0	1.046512	14.0	0.0	1.000000	1.088235	0
3	Alexandre Pantoja	31	125	65	8	3	1.0	1.046512	24.0	5.0	1.097561	1.088235	0
4	Alex Perez	29	125	64	6	2	1.0	1.046512	24.0	6.0	1.097561	1.088235	0

In [88]:
```
age_bins = [15,21,31,36,46,float('inf')]
age_labels = ['15-20','21-30','31-35','36-45','46+']
df['age_group'] = pd.cut(df['age'],bins=age_bins,labels=age_labels)
```

In [97]:
```
df.drop('age',axis=1,inplace=True)
df.head()
```

Out[97]:

ame	weight	height	ufc_wins	ufc_loses	ufc_draws	ufc_no_contests	mma_wins	mma_loses	mma_draws	mma_no_contests	gender_encoded	age_group
don eno	-0.92852	-0.693834	0.730117	-0.331949	10.140677	0.0	0.661012	0.106242	6.165279	2.358119e-15	-0.486285	21-30
son edo	-0.92852	-1.242323	0.950402	-0.331949	0.000000	0.0	0.812136	-0.757267	-0.666517	2.358119e-15	-0.486285	31-35
skar arov	-0.92852	-1.242323	-0.371306	-1.068684	0.000000	0.0	-0.094606	-1.332940	-0.666517	2.358119e-15	-0.486285	21-30
ndre ntoja	-0.92852	-1.242323	0.730117	0.036419	0.000000	0.0	1.416630	0.106242	0.000000	2.358119e-15	-0.486285	21-30
Alex erez	-0.92852	-1.516567	0.289548	-0.331949	0.000000	0.0	1.416630	0.394078	0.000000	2.358119e-15	-0.486285	21-30

```
In [100]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 617 entries, 0 to 616
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   ranking          177 non-null    object
 1   name             617 non-null    object
 2   weight           617 non-null    float64
 3   height           617 non-null    float64
 4   ufc_wins         617 non-null    float64
 5   ufc_loses        617 non-null    float64
 6   ufc_draws        617 non-null    float64
 7   ufc_no_contests  617 non-null    float64
 8   mma_wins         617 non-null    float64
 9   mma_loses        617 non-null    float64
 10  mma_draws        617 non-null    float64
 11  mma_no_contests  617 non-null    float64
 12  gender_encoded   617 non-null    float64
 13  age_group        617 non-null    category
dtypes: category(1), float64(11), object(2)
memory usage: 68.3+ KB
```

```
In [101]: df.describe()
```

Out[101]:

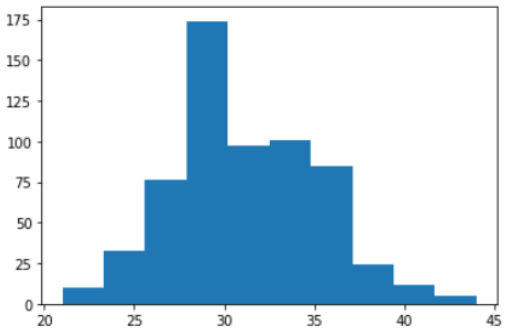
	weight	height	ufc_wins	ufc_loses	ufc_draws	ufc_no_contests	mma_wins	mma_loses	mma_draws	mma_no_contests	gen
count	6.170000e+02	6.170000e+02	617.000000	6.170000e+02	617.000000	6.170000e+02	6.170000e+02	617.000000	6.170000e+02	6.170000e+02	
mean	-4.606436e-17	4.606436e-17	0.000000	-1.151609e-17	0.000000	4.318534e-18	-2.303218e-17	0.000000	-1.151609e-17	4.030631e-17	
std	1.000811e+00	1.000811e+00	1.000811	1.000811e+00	1.000811	1.000811e+00	1.000811e+00	1.000811	1.000811e+00	1.000811e+00	
min	-1.195729e+00	-2.887789e+00	-1.032161	-1.068684e+00	-10.140677	-8.366261e-01	-2.059212e+00	-1.332940	-7.498312e+00	-9.370611e-01	
25%	-6.613105e-01	-6.938340e-01	-0.811876	-7.003166e-01	0.000000	1.036448e-16	-6.991001e-01	-0.757267	9.212872e-17	-7.753356e-17	
50%	-3.941012e-01	-1.453451e-01	-0.371306	-3.319489e-01	0.000000	1.036448e-16	-9.460581e-02	-0.181594	9.212872e-17	-7.753356e-17	
75%	2.739220e-01	6.773882e-01	0.289548	4.047866e-01	0.000000	1.036448e-16	5.098884e-01	0.394078	9.212872e-17	-7.753356e-17	
max	2.812410e+00	2.597099e+00	5.356096	5.561935e+00	10.140677	1.715084e+01	6.705955e+00	4.711624	6.165279e+00	9.682965e+00	

5) Give visualization of statistical description of data – in form of histogram, scatter plot, pie chart,Give correlation matrix

Histogram:

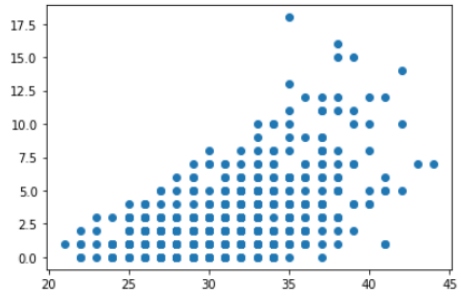
```
In [3]: plt.hist(df['age'])
```

```
Out[3]: (array([ 10.,  33.,  76., 174.,  97., 101.,  85.,  24.,  12.,   5.]),
         array([21. , 23.3, 25.6, 27.9, 30.2, 32.5, 34.8, 37.1, 39.4, 41.7, 44. ]),
         <BarContainer object of 10 artists>)
```



Scatter plot:

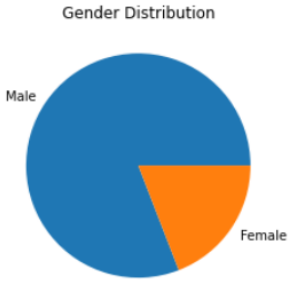
```
In [5]: plt.scatter(df['age'],df['ufc_loses'])
Out[5]: <matplotlib.collections.PathCollection at 0x1d0bab3b280>
```



Piechart:

```
In [7]: gender_counts = df['gender'].value_counts()

        plt.pie(gender_counts, labels=['Male','Female'])
        plt.title('Gender Distribution')
        plt.show()
```



Correlation matrix:

```
In [8]: two_df = df[['gender','age','weight']]
        correlation_matrix = two_df.corr()
        print(correlation_matrix)
```

```
                age    weight
age        1.000000  0.190549
weight     0.190549  1.000000
```

Frequency table:

```
In [8]: test= df.groupby(['gender','weight'])
        test.size()
```

```
Out[8]: gender  weight
        Female  115       43
                125       43
                135       26
                145        6
        Male    125       31
                135       82
                145       78
                155       82
                170       74
                185       68
                205       42
                265       42
        dtype: int64
```