pr-7

October 8, 2023

1 Exploratory Data Analysis (EDA) on IRIS Dataset

Exploratory Data Analysis (EDA) is a foundational step in the data analysis process. Before delving into complex modeling, it's imperative to understand the characteristics of the data we're working with.

1.1 Definition

> **Exploratory Data Analysis (EDA)** is an approach to analyze data sets to summarize their main characteristics, often using statistical graphics, plots, and information tables.

1.2 Purpose of EDA

1. **Detecting Outliers:** Identify unusual or unexpected data points.
2. **Testing Assumptions:** Validate if certain assumptions about data are true.
3. **Preliminary Selection of Models:** Inform the choice of suitable statistical models.
4. **Determining Relationships:** Identify patterns, relationships, or anomalies.

1.3 Techniques Used in EDA

1.3.1 1. Descriptive Statistics

- **Mean:** Average of all data points.
- **Median:** Middle value when data is sorted.
- **Mode:** Most frequently occurring value.
- **Standard Deviation:** Measures the amount of variation in the dataset.

1.3.2 2. Visualization

- **Histograms:** Show the distribution of a dataset.
- **Box Plots:** Visualize basic statistics like outliers, min/max.
- **Scatter Plots:** Show relationships between two variables.
- **Pair Plots:** Visualize pairwise relationships in a dataset.

1.3.3 3. Correlation

A statistical measure that expresses the extent to which two variables change together.

- **Positive Correlation:** As one variable increases, the other also does.
- **Negative Correlation:** As one variable increases, the other decreases.

1.3.4 4. Handling Missing Data

This is crucial, as many algorithms won't work if there are missing values in the data.

- **Drop:** Remove records with missing values.
- **Impute:** Replace missing values using methods like mean, median, or mode.

1.4 Benefits of EDA

- **Better Understanding of Data:** Deep insights into the nature and structure of data.
- **Informed Decision Making:** Equips analysts and stakeholders with a thorough understanding to make data-driven decisions.
- **Building Better Models:** Helps in feature engineering and model selection for better prediction and classification tasks.

1.5 Conclusion

EDA is a critical step in the data science pipeline. A well-executed EDA informs subsequent steps and can often be the difference between a successful model and a flawed one. Always prioritize EDA to ensure a deep, thorough understanding of any dataset you're working with.

[2]:
```
# Importing libraries in Python
import sklearn.datasets as datasets
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Loading the iris dataset
iris = pd.read_csv('iris.csv')
```

- `df.head()`: Returns the first few rows (default is 5) of the DataFrame `df`.

[4]:
```
iris.head()
iris.head(8)
```

[4]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
5	6	5.4	3.9	1.7	0.4	Iris-setosa
6	7	4.6	3.4	1.4	0.3	Iris-setosa
7	8	5.0	3.4	1.5	0.2	Iris-setosa

- `df.tail()`: Returns the last few rows (default is 5) of the DataFrame `df`.

[5]:
```
iris.tail()
iris.tail(8)
```

[5]:
```
       Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
142  143            5.8           2.7            5.1           1.9
143  144            6.8           3.2            5.9           2.3
144  145            6.7           3.3            5.7           2.5
145  146            6.7           3.0            5.2           2.3
146  147            6.3           2.5            5.0           1.9
147  148            6.5           3.0            5.2           2.0
148  149            6.2           3.4            5.4           2.3
149  150            5.9           3.0            5.1           1.8

             Species
142  Iris-virginica
143  Iris-virginica
144  Iris-virginica
145  Iris-virginica
146  Iris-virginica
147  Iris-virginica
148  Iris-virginica
149  Iris-virginica
```

- `df.shape`: Provides a tuple representing the dimensionality (rows, columns) of the DataFrame `df`.

[6]:
```
iris.shape
```

[6]:
```
(150, 6)
```

- `df.info()`: Gives a concise summary of the DataFrame, including data types and non-null values.

[7]:
```
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             150 non-null    int64
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

- `df.describe()`: Provides descriptive statistics of the columns in the DataFrame `df`.

[8]:
```
iris.describe()
```

[8]:
```
                Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
count   150.000000     150.000000    150.000000     150.000000    150.000000
mean     75.500000       5.843333      3.054000       3.758667      1.198667
std      43.445368       0.828066      0.433594       1.764420      0.763161
min       1.000000       4.300000      2.000000       1.000000      0.100000
25%      38.250000       5.100000      2.800000       1.600000      0.300000
50%      75.500000       5.800000      3.000000       4.350000      1.300000
75%     112.750000       6.400000      3.300000       5.100000      1.800000
max     150.000000       7.900000      4.400000       6.900000      2.500000
```

- `df.isnull()`: Returns a DataFrame of the same shape as `df` but with `True` for missing values and `False` for non-missing values.

[9]:
```
iris.isnull().sum()
```

[9]:
```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

- `df.drop_duplicates()`: Removes duplicate rows from the DataFrame `df`, keeping the first occurrence by default.

[10]:
```
df1 = iris.drop_duplicates()
print(df1.shape)
iris.shape
```

```
(150, 6)
```

[10]:
```
(150, 6)
```

- `df.value_counts()`: Returns a series containing counts of unique values, sorted in descending order.

[11]:
```
iris.value_counts('Species')
```

[11]:
```
Species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: count, dtype: int64
```

- `df.sample()`: Returns a random sample of items (rows by default) from the DataFrame `df`.

[12]:
```
iris.sample(10)
```

[12]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	\
2	3	4.7	3.2	1.3	0.2	
141	142	6.9	3.1	5.1	2.3	
79	80	5.7	2.6	3.5	1.0	
144	145	6.7	3.3	5.7	2.5	
46	47	5.1	3.8	1.6	0.2	
73	74	6.1	2.8	4.7	1.2	
10	11	5.4	3.7	1.5	0.2	
55	56	5.7	2.8	4.5	1.3	
124	125	6.7	3.3	5.7	2.1	
14	15	5.8	4.0	1.2	0.2	

	Species
2	Iris-setosa
141	Iris-virginica
79	Iris-versicolor
144	Iris-virginica
46	Iris-setosa
73	Iris-versicolor
10	Iris-setosa
55	Iris-versicolor
124	Iris-virginica
14	Iris-setosa

- `df.nlargest(n, 'column')`: Returns the first `n` occurrences ordered by a specified column in descending order.

[13]:
```
iris.nlargest(15,"SepalLengthCm")
```

[13]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	\
131	132	7.9	3.8	6.4	2.0	
117	118	7.7	3.8	6.7	2.2	
118	119	7.7	2.6	6.9	2.3	
122	123	7.7	2.8	6.7	2.0	
135	136	7.7	3.0	6.1	2.3	
105	106	7.6	3.0	6.6	2.1	
130	131	7.4	2.8	6.1	1.9	
107	108	7.3	2.9	6.3	1.8	
109	110	7.2	3.6	6.1	2.5	
125	126	7.2	3.2	6.0	1.8	
129	130	7.2	3.0	5.8	1.6	
102	103	7.1	3.0	5.9	2.1	
50	51	7.0	3.2	4.7	1.4	
52	53	6.9	3.1	4.9	1.5	
120	121	6.9	3.2	5.7	2.3	

	Species

```
131   Iris-virginica
117   Iris-virginica
118   Iris-virginica
122   Iris-virginica
135   Iris-virginica
105   Iris-virginica
130   Iris-virginica
107   Iris-virginica
109   Iris-virginica
125   Iris-virginica
129   Iris-virginica
102   Iris-virginica
50   Iris-versicolor
52   Iris-versicolor
120   Iris-virginica
```

- `df.nsmallest(n, 'column')`: Returns the first `n` occurrences ordered by a specified column in ascending order.

[14]:
```
iris.nsmallest(15,"SepalLengthCm")
```

[14]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
13	14	4.3	3.0	1.1	0.1	Iris-setosa
8	9	4.4	2.9	1.4	0.2	Iris-setosa
38	39	4.4	3.0	1.3	0.2	Iris-setosa
42	43	4.4	3.2	1.3	0.2	Iris-setosa
41	42	4.5	2.3	1.3	0.3	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
6	7	4.6	3.4	1.4	0.3	Iris-setosa
22	23	4.6	3.6	1.0	0.2	Iris-setosa
47	48	4.6	3.2	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
29	30	4.7	3.2	1.6	0.2	Iris-setosa
11	12	4.8	3.4	1.6	0.2	Iris-setosa
12	13	4.8	3.0	1.4	0.1	Iris-setosa
24	25	4.8	3.4	1.9	0.2	Iris-setosa
30	31	4.8	3.1	1.6	0.2	Iris-setosa

- `df.loc[]`: Accesses a group of rows and columns by labels or boolean array.

[15]:
```
iris.loc[iris["SepalLengthCm"]<7.9]
```

[15]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	\
0	1	5.1	3.5	1.4	0.2	
1	2	4.9	3.0	1.4	0.2	
2	3	4.7	3.2	1.3	0.2	
3	4	4.6	3.1	1.5	0.2	
4	5	5.0	3.6	1.4	0.2	

```
..   …             …           …             …           …
145  146           6.7         3.0           5.2         2.3
146  147           6.3         2.5           5.0         1.9
147  148           6.5         3.0           5.2         2.0
148  149           6.2         3.4           5.4         2.3
149  150           5.9         3.0           5.1         1.8

            Species
0       Iris-setosa
1       Iris-setosa
2       Iris-setosa
3       Iris-setosa
4       Iris-setosa
..              …
145  Iris-virginica
146  Iris-virginica
147  Iris-virginica
148  Iris-virginica
149  Iris-virginica

[149 rows x 6 columns]
```

- `df.iloc[]`: Accesses a group of rows and columns by integer-based index location.

[16]:
```
iris.iloc[1:4,1:3]
```

[16]:
```
   SepalLengthCm  SepalWidthCm
1            4.9           3.0
2            4.7           3.2
3            4.6           3.1
```

[19]:
```
# Loading the titanic dataset
titanic = pd.read_csv('titanic.csv')

print(titanic.head())
print(titanic.head(8))

print(titanic.tail())
print(titanic.tail(8))

print(titanic.shape)

print(titanic.info())

print(titanic.describe())

print(titanic.isnull().sum())
```

```
df1 = titanic.drop_duplicates()
print(df1.shape)
print(titanic.shape)

print(titanic.value_counts('age'))

print(titanic.sample(10))

print(titanic.nlargest(15,"fare"))

print(titanic.nsmallest(15,"fare"))

print(titanic.loc[titanic["fare"]<10.0])

print(titanic.iloc[1:4,1:3])
```

```
      sex   age  sibsp  parch     fare embarked  class    who  alone  survived
0    male  22.0      1      0   7.2500        S  Third    man  False         0
1  female  38.0      1      0  71.2833        C  First  woman  False         1
2  female  26.0      0      0   7.9250        S  Third  woman   True         1
3  female  35.0      1      0  53.1000        S  First  woman  False         1
4    male  35.0      0      0   8.0500        S  Third    man   True         0
      sex   age  sibsp  parch     fare embarked  class    who  alone  survived
0    male  22.0      1      0   7.2500        S  Third    man  False         0
1  female  38.0      1      0  71.2833        C  First  woman  False         1
2  female  26.0      0      0   7.9250        S  Third  woman   True         1
3  female  35.0      1      0  53.1000        S  First  woman  False         1
4    male  35.0      0      0   8.0500        S  Third    man   True         0
5    male   NaN      0      0   8.4583        Q  Third    man   True         0
6    male  54.0      0      0  51.8625        S  First    man   True         0
7    male   2.0      3      1  21.0750        S  Third  child  False         0
        sex   age  sibsp  parch   fare embarked   class    who  alone  \
886    male  27.0      0      0  13.00        S  Second    man   True
887  female  19.0      0      0  30.00        S   First  woman   True
888  female   NaN      1      2  23.45        S   Third  woman  False
889    male  26.0      0      0  30.00        C   First    man   True
890    male  32.0      0      0   7.75        Q   Third    man   True

     survived
886         0
887         1
888         0
889         1
890         0
        sex   age  sibsp  parch    fare embarked   class    who  alone  \
883    male  28.0      0      0  10.500        S  Second    man   True
```

```
884    male  25.0      0      0   7.050        S   Third    man   True
885  female  39.0      0      5  29.125        Q   Third  woman  False
886    male  27.0      0      0  13.000        S  Second    man   True
887  female  19.0      0      0  30.000        S   First  woman   True
888  female   NaN      1      2  23.450        S   Third  woman  False
889    male  26.0      0      0  30.000        C   First    man   True
890    male  32.0      0      0   7.750        Q   Third    man   True

     survived
883         0
884         0
885         0
886         0
887         1
888         0
889         1
890         0
(891, 10)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 10 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   sex       891 non-null    object
 1   age       714 non-null    float64
 2   sibsp     891 non-null    int64
 3   parch     891 non-null    int64
 4   fare      891 non-null    float64
 5   embarked  889 non-null    object
 6   class     891 non-null    object
 7   who       891 non-null    object
 8   alone     891 non-null    bool
 9   survived  891 non-null    int64
dtypes: bool(1), float64(2), int64(3), object(4)
memory usage: 63.6+ KB
None
              age       sibsp       parch        fare    survived
count  714.000000  891.000000  891.000000  891.000000  891.000000
mean    29.699118    0.523008    0.381594   32.204208    0.383838
std     14.526497    1.102743    0.806057   49.693429    0.486592
min      0.420000    0.000000    0.000000    0.000000    0.000000
25%     20.125000    0.000000    0.000000    7.910400    0.000000
50%     28.000000    0.000000    0.000000   14.454200    0.000000
75%     38.000000    1.000000    0.000000   31.000000    1.000000
max     80.000000    8.000000    6.000000  512.329200    1.000000
sex           0
age         177
sibsp         0
```

```
parch         0
fare          0
embarked      2
class         0
who           0
alone         0
survived      0
dtype: int64
(780, 10)
(891, 10)
age
24.00    30
22.00    27
18.00    26
30.00    25
28.00    25
         ..
20.50     1
14.50     1
12.00     1
0.92      1
80.00     1
Name: count, Length: 88, dtype: int64
        sex   age  sibsp  parch     fare embarked   class    who  alone  \
650    male   NaN      0      0   7.8958        S   Third    man   True
122    male  32.5      1      0  30.0708        C  Second    man  False
702  female  18.0      0      1  14.4542        C   Third  woman  False
600  female  24.0      2      1  27.0000        S  Second  woman  False
472  female  33.0      1      2  27.7500        S  Second  woman  False
187    male  45.0      0      0  26.5500        S   First    man   True
661    male  40.0      0      0   7.2250        C   Third    man   True
847    male  35.0      0      0   7.8958        C   Third    man   True
836    male  21.0      0      0   8.6625        S   Third    man   True
123  female  32.5      0      0  13.0000        S  Second  woman   True

     survived
650         0
122         0
702         0
600         1
472         1
187         1
661         0
847         0
836         0
123         1
        sex   age  sibsp  parch      fare embarked  class    who  alone  \
258  female  35.0      0      0  512.3292        C  First  woman   True
```

```
679    male  36.0      0      1  512.3292        C  First    man  False
737    male  35.0      0      0  512.3292        C  First    man   True
27     male  19.0      3      2  263.0000        S  First    man  False
88   female  23.0      3      2  263.0000        S  First  woman  False
341  female  24.0      3      2  263.0000        S  First  woman  False
438    male  64.0      1      4  263.0000        S  First    man  False
311  female  18.0      2      2  262.3750        C  First  woman  False
742  female  21.0      2      2  262.3750        C  First  woman  False
118    male  24.0      0      1  247.5208        C  First    man  False
299  female  50.0      0      1  247.5208        C  First  woman  False
380  female  42.0      0      0  227.5250        C  First  woman   True
557    male   NaN      0      0  227.5250        C  First    man   True
700  female  18.0      1      0  227.5250        C  First  woman  False
716  female  38.0      0      0  227.5250        C  First  woman   True

     survived
258         1
679         1
737         1
27          0
88          1
341         1
438         0
311         1
742         1
118         0
299         1
380         1
557         0
700         1
716         1
      sex   age  sibsp  parch  fare embarked   class  who  alone  survived
179  male  36.0      0      0   0.0        S   Third  man   True         0
263  male  40.0      0      0   0.0        S   First  man   True         0
271  male  25.0      0      0   0.0        S   Third  man   True         1
277  male   NaN      0      0   0.0        S  Second  man   True         0
302  male  19.0      0      0   0.0        S   Third  man   True         0
413  male   NaN      0      0   0.0        S  Second  man   True         0
466  male   NaN      0      0   0.0        S  Second  man   True         0
481  male   NaN      0      0   0.0        S  Second  man   True         0
597  male  49.0      0      0   0.0        S   Third  man   True         0
633  male   NaN      0      0   0.0        S   First  man   True         0
674  male   NaN      0      0   0.0        S  Second  man   True         0
732  male   NaN      0      0   0.0        S  Second  man   True         0
806  male  39.0      0      0   0.0        S   First  man   True         0
815  male   NaN      0      0   0.0        S   First  man   True         0
822  male  38.0      0      0   0.0        S   First  man   True         0
       sex   age  sibsp  parch    fare embarked  class    who  alone  \
```

```
0      male  22.0      1      0  7.2500        S  Third    man  False
2    female  26.0      0      0  7.9250        S  Third  woman   True
4      male  35.0      0      0  8.0500        S  Third    man   True
5      male   NaN      0      0  8.4583        Q  Third    man   True
12     male  20.0      0      0  8.0500        S  Third    man   True
..      …   …    …    …     …      …    …    …    …
877    male  19.0      0      0  7.8958        S  Third    man   True
878    male   NaN      0      0  7.8958        S  Third    man   True
881    male  33.0      0      0  7.8958        S  Third    man   True
884    male  25.0      0      0  7.0500        S  Third    man   True
890    male  32.0      0      0  7.7500        Q  Third    man   True

     survived
0           0
2           1
4           0
5           0
12          0
..        …
877         0
878         0
881         0
884         0
890         0

[336 rows x 10 columns]
    age  sibsp
1  38.0      1
2  26.0      0
3  35.0      1
```