Statistical Machine Learning Approaches to Liver Disease Prediction

TeamID:PNT2022TMID49098

TeamLeader: Manivanitha

TeamMember: Logapriya

TeamMember: Nisha

TeamMember: Sakthisubashini

Exploratory Data Analysis

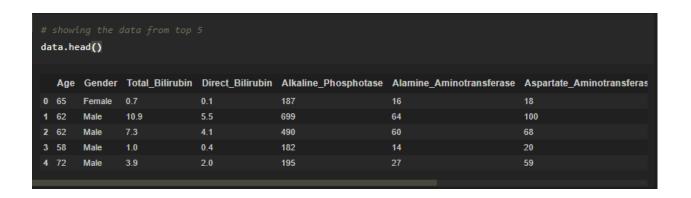
The exploratory data analysis (EDA) notebook is designed to assist you with discovering patterns in data, checking data sanity, and summarizing there levant data for predictive models.

The EDA notebook example was optimized with web-based data in mindand consists of two parts. Part one starts with using Query Service to view trends and data snapshots. Next, with a goal in mind for exploratory data analysis, the data is aggregated at the profile and visitor level.

Part two starts by performing descriptive analysis on aggregated data using Python libraries. This note book show cases visualizations such as histograms, scatterplots, boxplots and a correlation matrix to derive actionable insights used to determine which features are most likely to be helpful in predicting a goal.

| [4]; | | Year | Month | Count_days | First_date | Last_date | Count_hits |
|------|---|------|-------|------------|------------|-----------|------------|
| | 0 | 2020 | 1 | 1 | 31 | 31 | 117060 |
| | 1 | 2020 | 2 | 29 | 1 | 29 | 3503948 |
| | | | | | | | |

head(): To check the first five n rows of the dataset, we have a function called **head()**.



Tail(): To check the last five rows of the dataset, we have a function called tail().

| data.tail() | | | | | | | | | |
|-------------|-----|--------|-----------------|------------------|----------------------|--------------------------|-------------------------|--|--|
| | Age | Gender | Total_Bilirubin | Direct_Bilirubin | Alkaline_Phosphotase | Alamine_Aminotransferase | Aspartate_Aminotransfer | | |
| 578 | 60 | Male | 0.5 | 0.1 | 500 | 20 | 34 | | |
| 579 | 40 | Male | 0.6 | 0.1 | 98 | 35 | 31 | | |
| 580 | 52 | Male | 0.8 | 0.2 | 245 | 48 | 49 | | |
| 581 | 31 | Male | 1.3 | 0.5 | 184 | 29 | 32 | | |
| 582 | 38 | Male | 1.0 | 0.3 | 216 | 21 | 24 | | |
| | | | | | | | | | |

Will see how our dataset is, by using the info() method.

```
data.info()
 <class 'pandas.core.frame.DataFrame'>
 RangeIndex: 583 entries, 0 to 582
 Data columns (total 11 columns):
 # Column
                               Non-Null Count Dtype
 0 Age
                               583 non-null int64
                               583 non-null object
583 non-null float64
    Gender
                              583 non-null
     Total_Bilirubin
  3 Direct_Bilirubin
                              583 non-null float64
    Alkaline_Phosphotase
                               583 non-null int64
     Alamine_Aminotransferase
                               583 non-null
                                              int64
    Aspartate_Aminotransferase 583 non-null
                                             int64
     Total_Protiens
                               583 non-null
                                             float64
    Albumin
                                583 non-null
                                              float64
                                             float64
  9 Albumin_and_Globulin_Ratio 579 non-null
 10 Dataset
                               583 non-null
                                              int64
 dtypes: float64(5), int64(5), object(1)
 memory usage: 50.2+ KB
```

describe(): functions are used to compute values like count, mean, standard deviation and IQR (InterQuantile Ranges) and give a summary of numeric type data.

data.describe()

| data.describe() | | | | | | | | | |
|-----------------|------------|-----------------|------------------|----------------------|--------------------------|--------------------------|--|--|--|
| | Age | Total_Bilirubin | Direct_Bilirubin | Alkaline_Phosphotase | Alamine_Aminotransferase | Aspartate_Aminotransfera | | | |
| count | 583.000000 | 583.000000 | 583.000000 | 583.000000 | 583.000000 | 583.000000 | | | |
| mean | 44.746141 | 3.298799 | 1.486106 | 290.576329 | 80.713551 | 109.910806 | | | |
| std | 16.189833 | 6.209522 | 2.808498 | 242.937989 | 182.620356 | 288.918529 | | | |
| min | 4.000000 | 0.400000 | 0.100000 | 63.000000 | 10.000000 | 10.000000 | | | |
| 25% | 33.000000 | 0.800000 | 0.200000 | 175.500000 | 23.000000 | 25.000000 | | | |
| 50% | 45.000000 | 1.000000 | 0.300000 | 208.000000 | 35.000000 | 42.000000 | | | |
| 75% | 58.000000 | 2.600000 | 1.300000 | 298.000000 | 60.500000 | 87.000000 | | | |
| max | 90.000000 | 75.000000 | 19.700000 | 2110.000000 | 2000.000000 | 4929.000000 | | | |
| | | | | | | | | | |