21BDS0273
Anshika Singh

## Module 1: Introduction to Exploratory Data Analysis

**Objective:**

- Introduce the concept of EDA and understand the initial steps in data exploration.

**Key Operations:**

- Importing necessary libraries like `numpy`, `pandas`, `matplotlib`, and `seaborn`.
- Loading the dataset (`auto-mpg.csv`) into a pandas DataFrame.
- Displaying the first few rows using `df.head()`, checking for random samples using `df.sample()`, and getting summary information using `df.info()`.
- Analyzing the dataset structure, which consists of 398 records and 9 columns (mpg, cylinders, displacement, horsepower, weight, acceleration, model year, origin, car name).

**Summary:** This module helps in understanding the data's structure and determining the types of variables (numeric, categorical) before diving deeper into analysis.

## Module 2: Data Transformation

**Objective:**

- Clean and prepare the dataset for analysis.

**Key Operations:**

1. **Removing Duplicates:**
   - Checking for and removing duplicate records.
2. **Handling Missing Values:**
   - Replacing missing values represented as '?' with `NaN` and filling the missing 'horsepower' values using forward fill (`ffill`).
   - Using `df.isna().sum()` to get the total count of missing values and filling missing entries.
3. **Transforming Variables:**
   - Converting the 'horsepower' column to numeric, dealing with potential errors using `pd.to_numeric()`.
   - Creating a new categorical column `horsepower_bins` based on 'horsepower' ranges (Low, Medium, High).

**Summary:** This module focuses on cleaning the data by addressing missing values and ensuring that columns are in the correct format for analysis.

## Module 3: Correlation Analysis and Time Series Analysis

21BDS0273
Anshika Singh

**Objective:**

- Understand relationships between numerical features and analyze time-dependent patterns.

**Key Operations:**

1. **Correlation Analysis:**
    - Using the `corr()` method to compute correlation coefficients between numerical features.
    - Visualizing the correlation matrix using a heatmap with `seaborn` to identify which variables have strong correlations (e.g., mpg and weight).
2. **Time Series Analysis:**
    - Plotting the trend of average miles per gallon (`mpg`) over the model years to analyze how fuel efficiency has changed over time using `groupby()` and `plot()`.

**Summary:** This module helps in understanding how different numerical features relate to each other and how trends evolve over time.

## Module 4: Data Summarization and Visualization

**Objective:**

- Summarize the dataset with descriptive statistics and visualize distributions.

**Key Operations:**

1. **Descriptive Statistics:**
    - Using `describe()` to summarize numerical columns and get insights such as mean, standard deviation, min, and max values.
2. **Visualization:**
    - Plotting histograms for numerical variables like 'mpg', 'horsepower', 'weight', and 'displacement' to understand their distributions.
    - Using bar charts to visualize categorical variables (e.g., the distribution of cylinders and horsepower categories).
    - Additional analyses include plotting the MPG distribution and a scatter plot for horsepower vs. mpg.

**Summary:** This module highlights how to extract summary statistics and visualize the data using various plots, helping to identify patterns and distributions in the data.

## Module 5: Clustering Algorithms

**Objective:**

- Apply clustering techniques to group the data based on similarities.

**Key Operations:**

1. **Preprocessing:**
    - Preparing the dataset for clustering by selecting relevant features (e.g., mpg, cylinders, model year, origin) and scaling the features using `StandardScaler`.
2. **Clustering Models:**
    - **Agglomerative Clustering**: Hierarchical clustering method used to form clusters.
    - **Gaussian Mixture Models (GMM)**: A probabilistic model used for clustering with a Gaussian distribution assumption.
    - **Minimum Spanning Tree (MST) Clustering**: Based on distance measures between points to form clusters.
3. **Evaluation:**
    - Evaluating the clustering results using the silhouette score, which measures how similar objects are within the same cluster compared to other clusters.

**Summary:** This module demonstrates how to apply different clustering algorithms to group similar instances and how to evaluate the performance of these algorithms.

## Module 6: Data Visualization for Clusters

**Objective:**

- Visualize the results of clustering and understand cluster structures.

**Key Operations:**

1. **Visualizing Clusters:**
    - Scatter plots and pair plots for clustered data, helping to visually identify how different clusters are distributed across the feature space.
    - Using `sns.pairplot()` to see pairwise relationships in the dataset for the selected features.
2. **Cluster Comparison:**
    - Visualizing the mean MPG for each cluster to compare performance metrics between clusters.
    - Additional plots like boxplots for understanding the spread of 'mpg' by origin.

**Summary:** This module focuses on visualizing the clusters formed in the previous module to understand their characteristics and differences.

## Module 7: Advanced Analysis and Model Evaluation

**Objective:**

21BDS0273
Anshika Singh

- Apply advanced techniques for analysis and evaluate model performance.

**Key Operations:**

1. **Advanced Statistical Analysis:**
    - Exploring statistical measures like variance, mean, and skewness in features.
    - Using advanced visualizations to compare features within each cluster, such as box plots and strip plots.
2. **Model Evaluation:**
    - Evaluating the performance of different clustering models using the silhouette score, which indicates how well the data points fit their clusters.
    - Tuning models and adjusting hyperparameters to improve the quality of clustering.

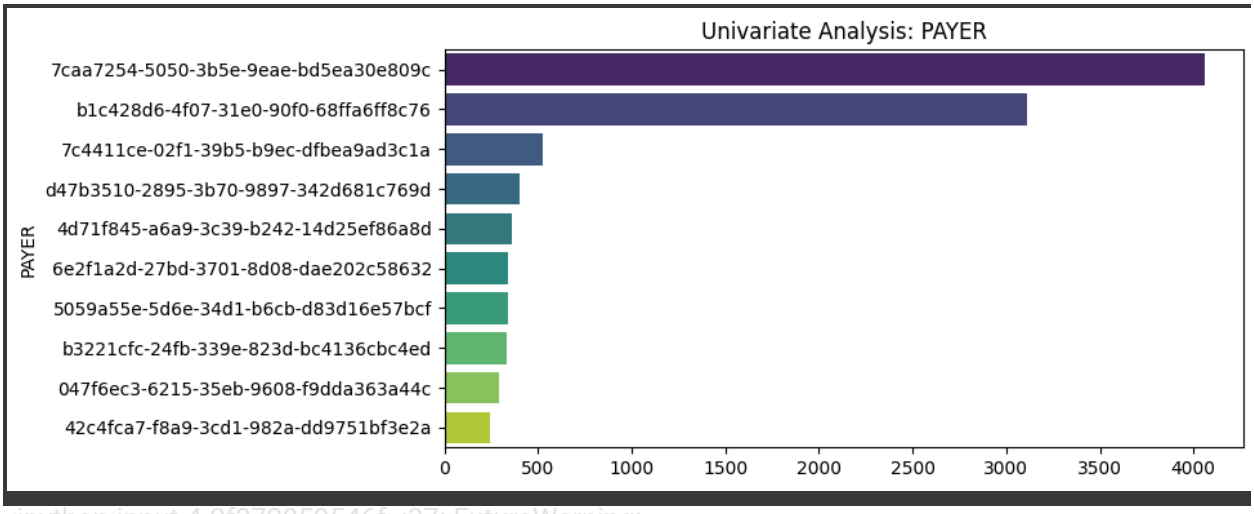
**Summary:** This final module applies advanced methods to analyze clusters in more detail and evaluates the quality of clustering models through performance metrics like the silhouette score.

## Final Thoughts:

This detailed breakdown covers the core steps of Exploratory Data Analysis (EDA) and the clustering analysis that follows. Each module progresses logically from loading and cleaning the data to complex analysis and model evaluation, with the goal of gaining actionable insights from the dataset.

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.

```
sns.barplot(y=value_counts.index, x=value_counts.values, palette='viridis')
```

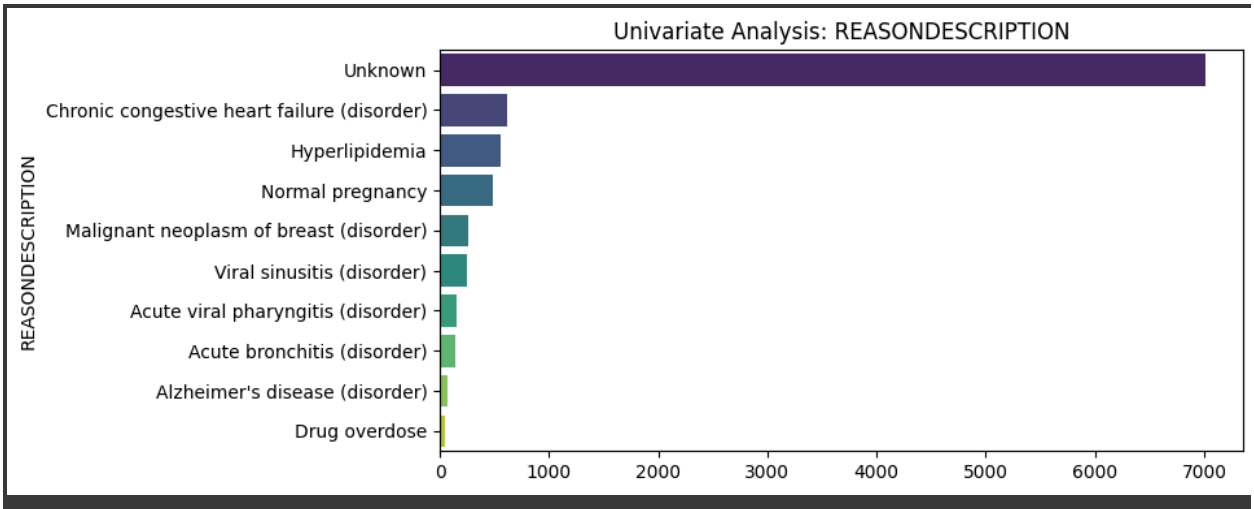


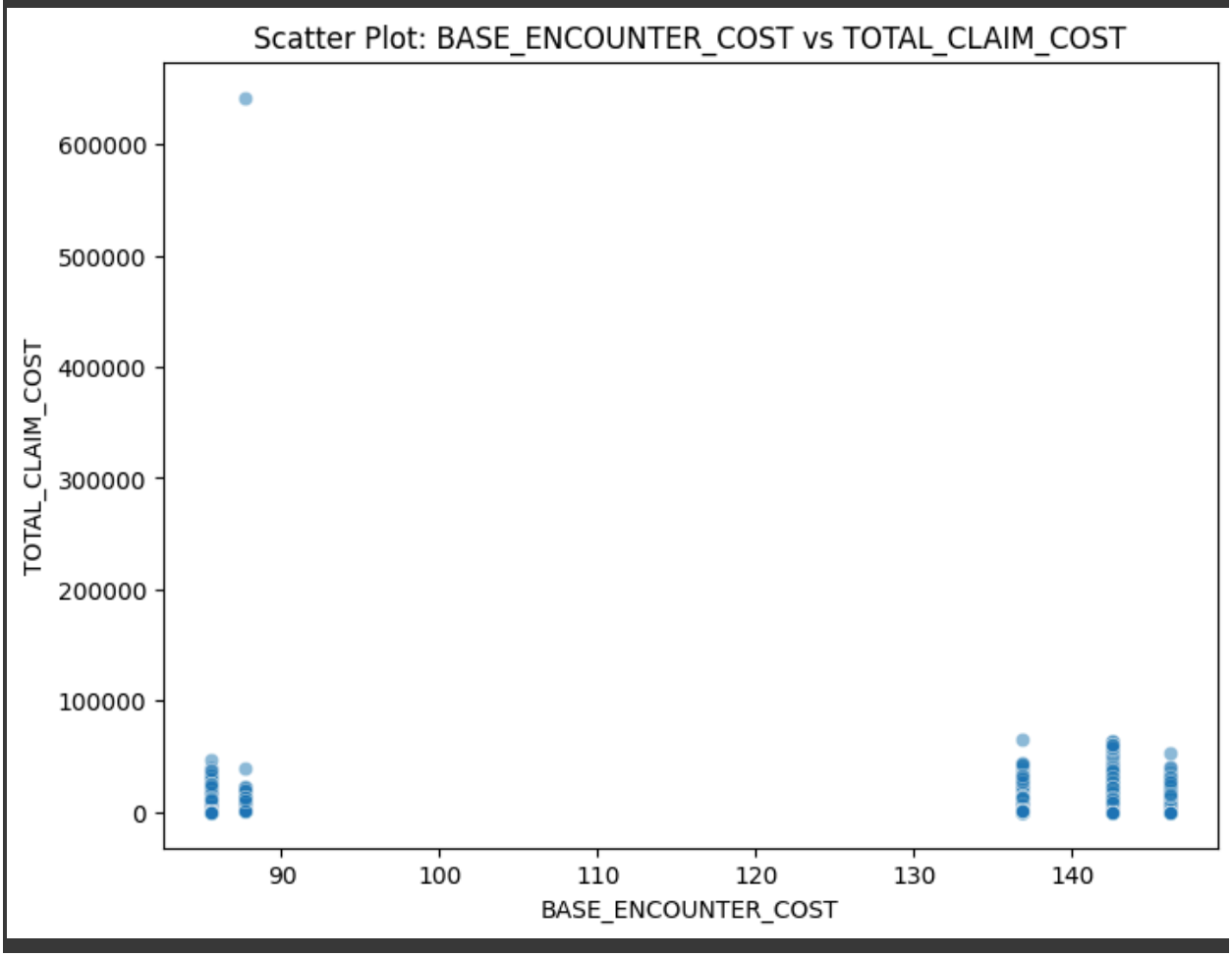

<ipython-input-4-2f372859546f>:37: FutureWarning:

21BDS0273
Anshika Singh

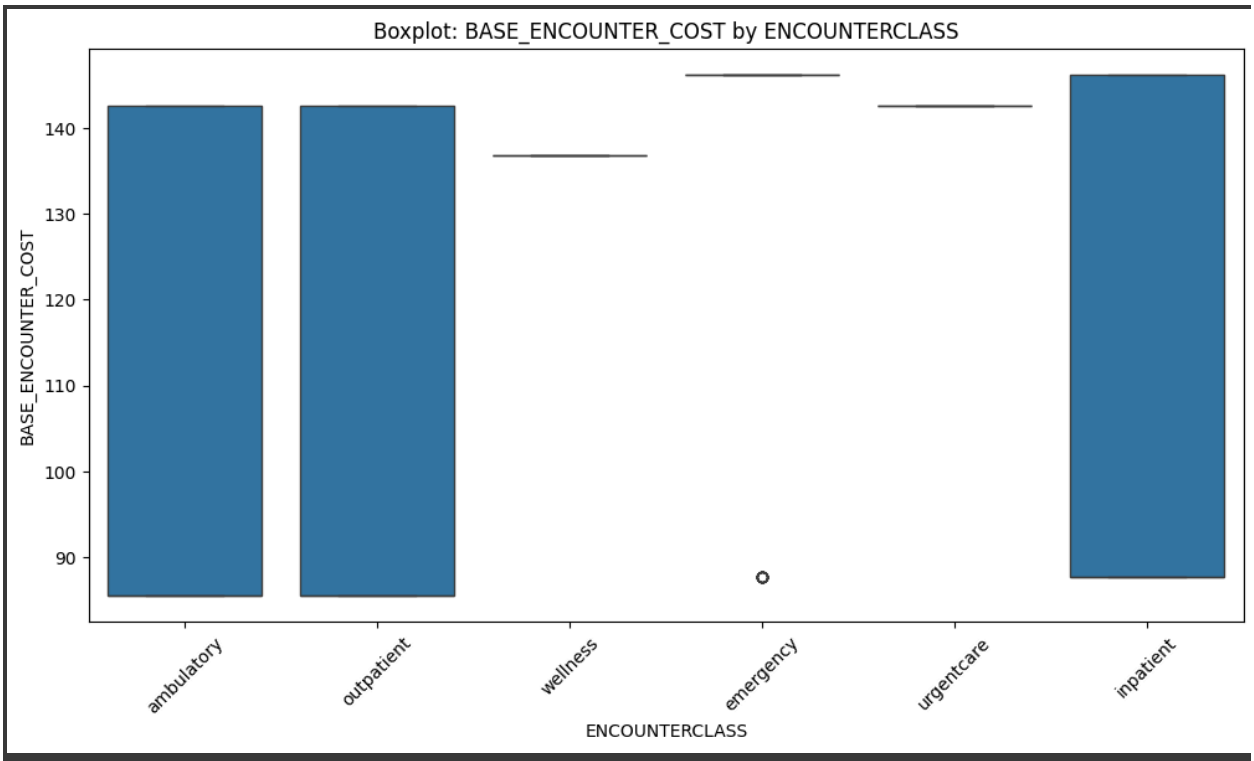
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.

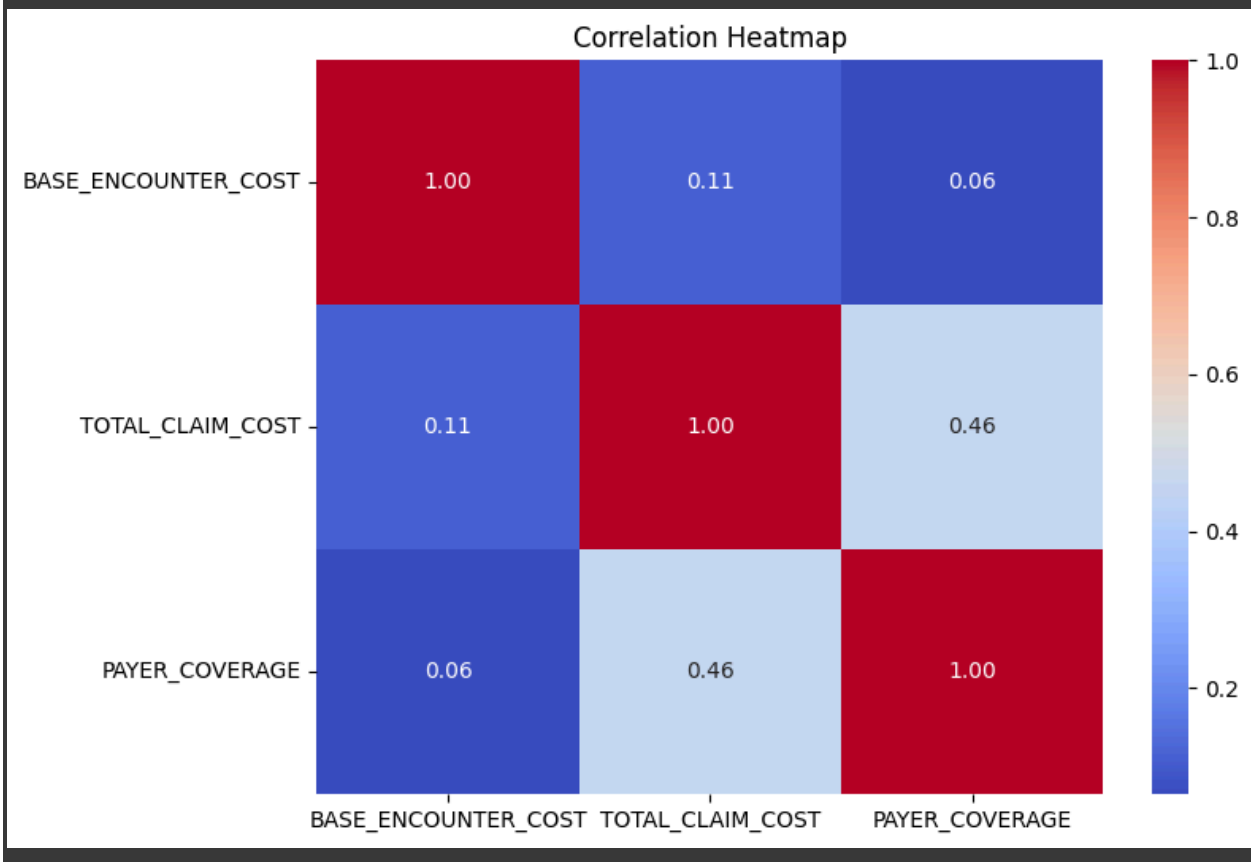
sns.barplot(y=value_counts.index, x=value_counts.values, palette='viridis')



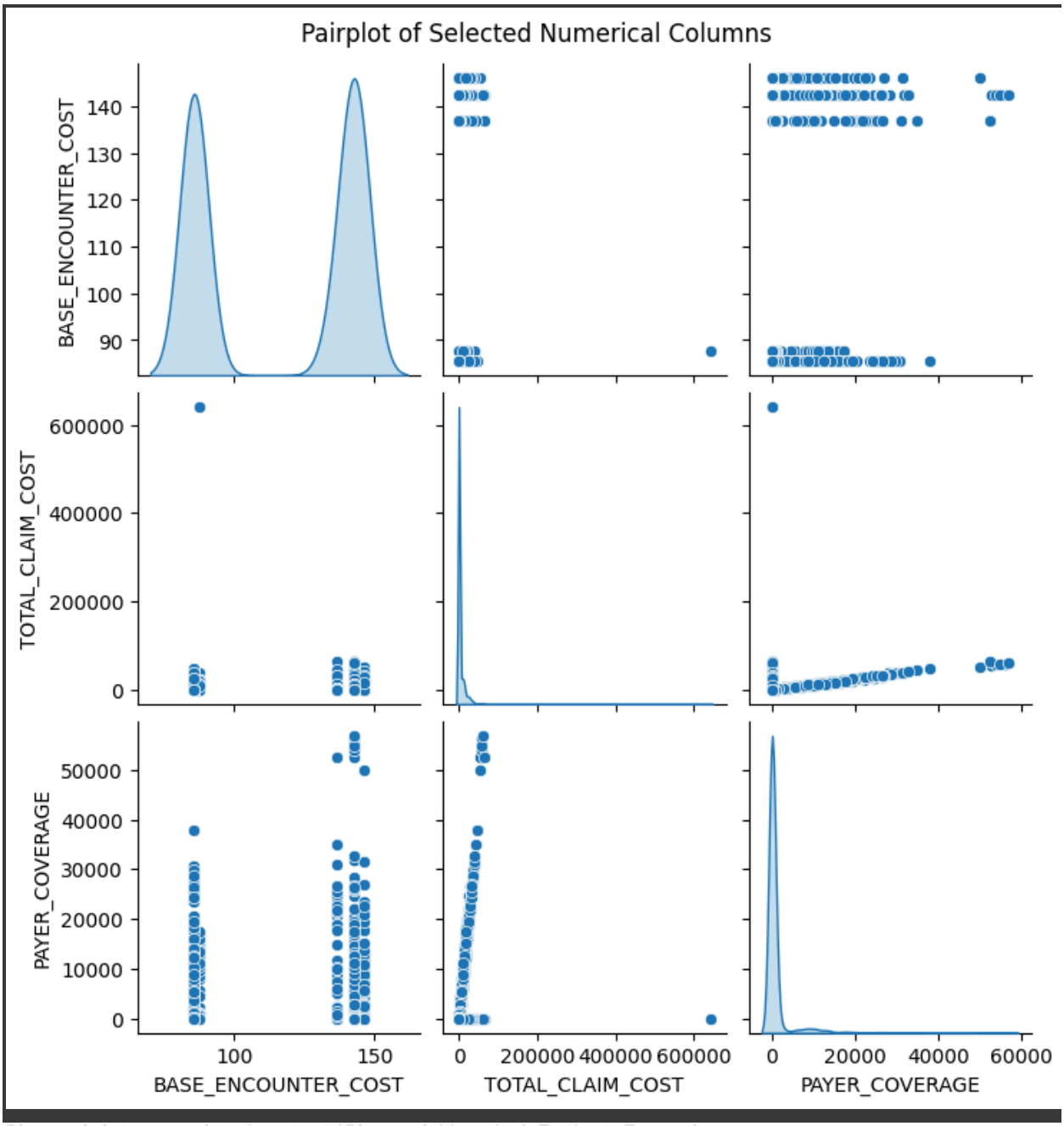



21BDS0273
Anshika Singh



21BDS0273
Anshika Singh




Cleaned data saved to /content/Cleaned_Hospital_Patient_Records.csv
Dataset Dimensions (Rows, Columns): (10000, 14)

Dataset Summary:

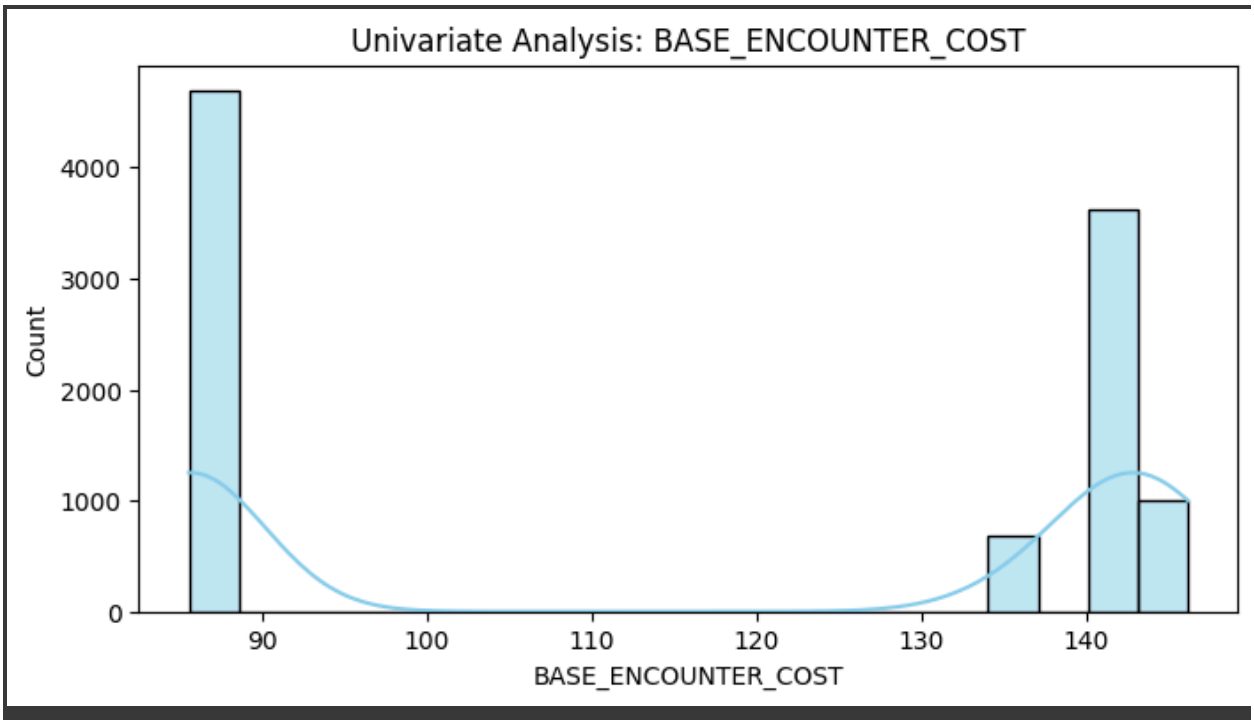
| | Id | START |
|---|---|---|
| count | 10000 | 10000 |
| unique | 10000 | 9943 |
| top | 45e12044-bee7-0cf6-1fa4-22cf463aa876 | 2012-07-23T17:55:09Z |
| freq | 1 | 2 |
| mean | NaN | NaN |

```
std                           NaN                NaN
min                           NaN                NaN
25%                           NaN                NaN
50%                           NaN                NaN
75%                           NaN                NaN
max                           NaN                NaN

                    STOP                       PATIENT  \
count              10000                         10000
unique              9979                           847
top     2016-11-09T09:18:33Z  1712d26d-822d-1e3a-2267-0a9dba31d7c8
freq                   2                           497
mean                 NaN                           NaN
std                  NaN                           NaN
min                  NaN                           NaN
25%                  NaN                           NaN
50%                  NaN                           NaN
75%                  NaN                           NaN
max                  NaN                           NaN

                         ORGANIZATION  \
count                           10000
unique                              1
top     d78e84ec-30aa-3bba-a33a-f29a3a454662
freq                            10000
mean                              NaN
std                               NaN
min                               NaN
25%                               NaN
50%                               NaN
75%                               NaN
max                               NaN

                                PAYER ENCOUNTERCLASS         CODE  \
count                           10000         10000  1.000000e+04
unique                             10             6          NaN
top     7caa7254-5050-3b5e-9eae-bd5ea30e809c     ambulatory          NaN
freq                             4061          4512          NaN
mean                              NaN           NaN  2.944941e+08
std                               NaN           NaN  1.998256e+08
min                               NaN           NaN  1.505002e+06
25%                               NaN           NaN  1.853450e+08
50%                               NaN           NaN  1.853490e+08
75%                               NaN           NaN  4.104100e+08
```

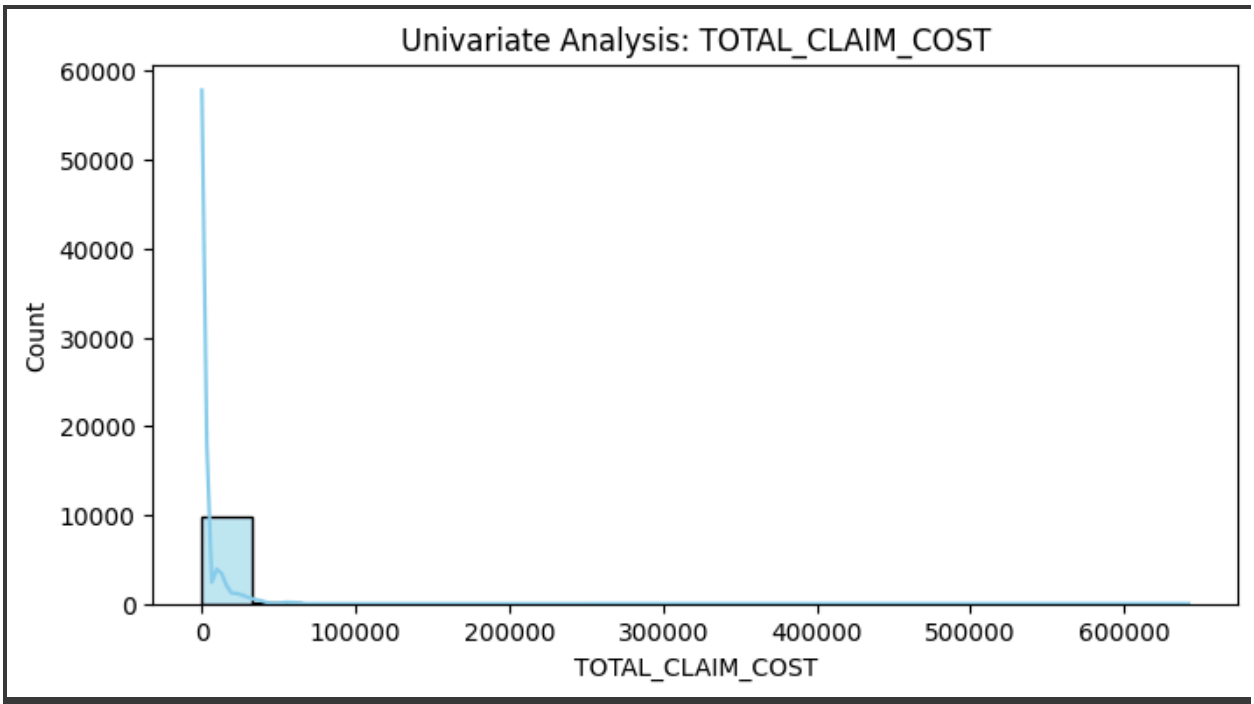
```
max                              NaN          NaN  7.029270e+08

                        DESCRIPTION  BASE_ENCOUNTER_COST  \
count                         10000        10000.000000
unique                           50                 NaN
top     Encounter for problem (procedure)               NaN
freq                           1574                 NaN
mean                            NaN          115.872884
std                             NaN           28.443555
min                             NaN           85.550000
25%                             NaN           85.550000
50%                             NaN          136.800000
75%                             NaN          142.580000
max                             NaN          146.180000

       TOTAL_CLAIM_COST  PAYER_COVERAGE    REASONCODE  \
count      10000.000000    10000.000000  2.992000e+03
unique              NaN             NaN           NaN
top                 NaN             NaN           NaN
freq                NaN             NaN           NaN
mean        3670.142553     1119.613660  2.456104e+11
std        10497.436123     4688.193825  4.350989e+12
min            0.000000        0.000000  5.602001e+06
25%          142.580000        0.000000  5.582200e+07
50%          278.580000       24.270000  7.549800e+07
75%         1391.400000      155.770000  1.956620e+08
max       641882.700000   188453.170000  1.241710e+14

                          REASONDESCRIPTION
count                                 2992
unique                                  64
top     Chronic congestive heart failure (disorder)
freq                                   611
mean                                   NaN
std                                    NaN
min                                    NaN
25%                                    NaN
50%                                    NaN
75%                                    NaN
max                                    NaN

Missing Values:
 Id                    0
START                  0
```

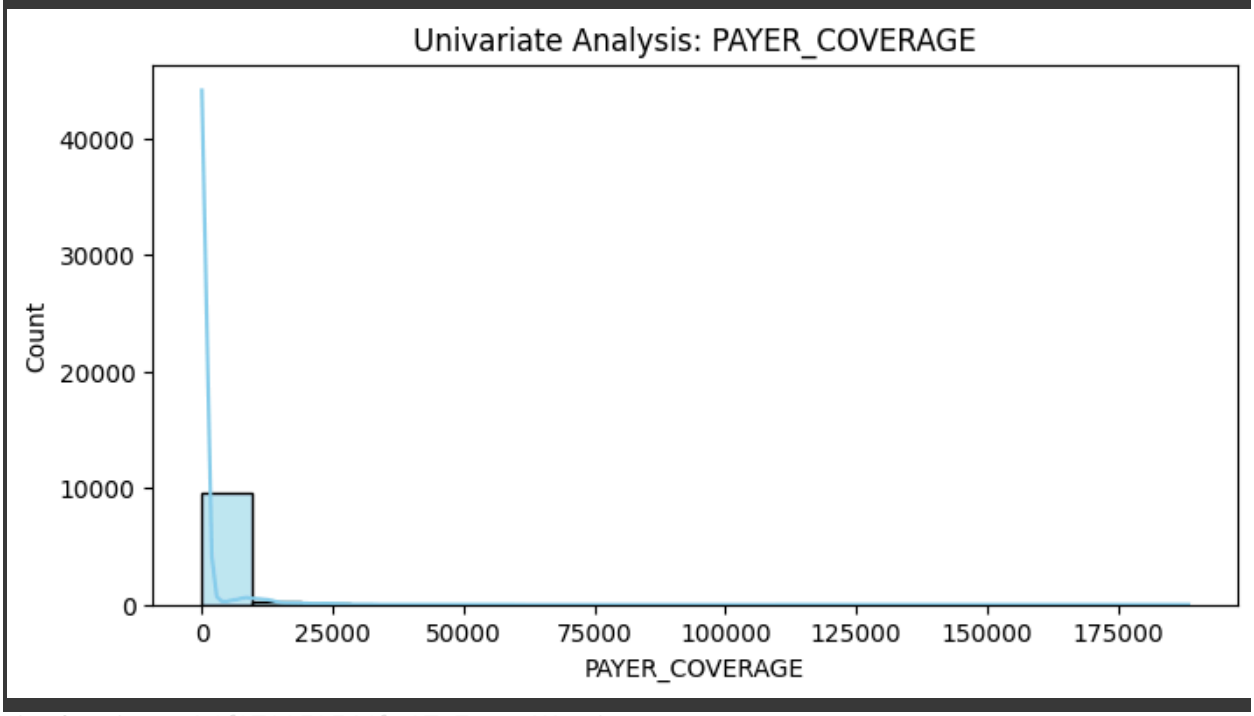
21BDS0273
Anshika Singh

```
STOP                    0
PATIENT                 0
ORGANIZATION            0
PAYER                   0
ENCOUNTERCLASS          0
CODE                    0
DESCRIPTION             0
BASE_ENCOUNTER_COST     0
TOTAL_CLAIM_COST        0
PAYER_COVERAGE          0
REASONCODE           7008
REASONDESCRIPTION    7008
dtype: int64
```



21BDS0273
Anshika Singh




<ipython-input-4-2f372859546f>:37: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.

sns.barplot(y=value_counts.index, x=value_counts.values, palette='viridis')

21BDS0273
Anshika Singh