Exploratory Data Analysis:Unveiling Insights from Raw Data

February 17, 2025

Siddhesh Nikam

NEU ID : 002378758

With Example 1

1 Exploratory Data Analysis: Unveiling Insights from Raw Data

Introduction

The Titanic dataset is one of the most well-known datasets in the field of data science and machine learning, frequently used for educational purposes and to illustrate various data analysis techniques. It provides a snapshot of the passengers aboard the RMS Titanic, which tragically sank on its maiden voyage in April 1912. The dataset includes a variety of features about the passengers, such as their age, sex, passenger class, ticket fare, and whether they survived or not.

The primary objective of analyzing the Titanic dataset is to uncover patterns and relationships that can help predict the likelihood of survival based on the available features. Understanding these patterns through Exploratory Data Analysis (EDA) is essential for preparing the data for more advanced modeling techniques. The dataset offers a rich set of variables, some of which are intuitive (e.g., sex, age, passenger class) while others are more complex (e.g., ticket fare, number of siblings or spouses aboard, and embarked location).

Research Question

How does Exploratory Data Analysis (EDA) help in understanding and preparing data for modeling?

Importance of EDA

Exploratory Data Analysis (EDA) is a crucial step in the data science pipeline, providing initial insights into the structure, quality, and patterns within a dataset. It helps in:

Identifying missing or incorrect values The Titanic dataset contains missing values in some columns like Age and Cabin. EDA allows you to spot these and decide how to handle them (e.g., by imputing or dropping).

Understanding data distributions For example, understanding the distribution of the Age column helps in deciding whether to treat it as continuous or categorize it into ranges (e.g., child, adult, senior). This can impact feature engineering for models.

Detecting outliers Outliers in numerical features like Fare could skew analysis or models. By visualizing distributions, you can detect extreme values and handle them appropriately—perhaps

by capping or removing them.

Revealing relationships between variables Correlations and interactions between features, like Sex, Age, and Pclass, can guide you in choosing relevant features for your predictive models. For example, a deeper relationship between Sex and Survival suggests it should be an important feature.

Guiding feature selection and engineering After performing EDA, you might create new features such as: Family Size: Adding together the SibSp (siblings/spouse) and Parch (parents/children) columns could reveal insights into how family size affects survival chances. Title: Extracting titles from the Name column (e.g., Mr., Mrs., Miss) can provide meaningful categorical features for predicting survival.

Theory and Background

Definition of EDA

EDA is the process of analyzing datasets to summarize their main characteristics, often using visual methods. Introduced by John Tukey in the 1970s, it emphasizes graphical techniques to uncover data patterns before formal modeling.

Key EDA Techniques

Summary Statistics: Mean, median, variance, skewness, kurtosis

Data Visualization: Histograms, box plots, scatter plots

Missing Data Handling: Imputation, deletion, transformation

Correlation Analysis: Pearson/Spearman correlation

Feature Engineering: Creating new variables based on domain knowledge

1.1 Problem Statement

Datasets Chosen Titanic Dataset: Used to understand survival factors.

Research Goals

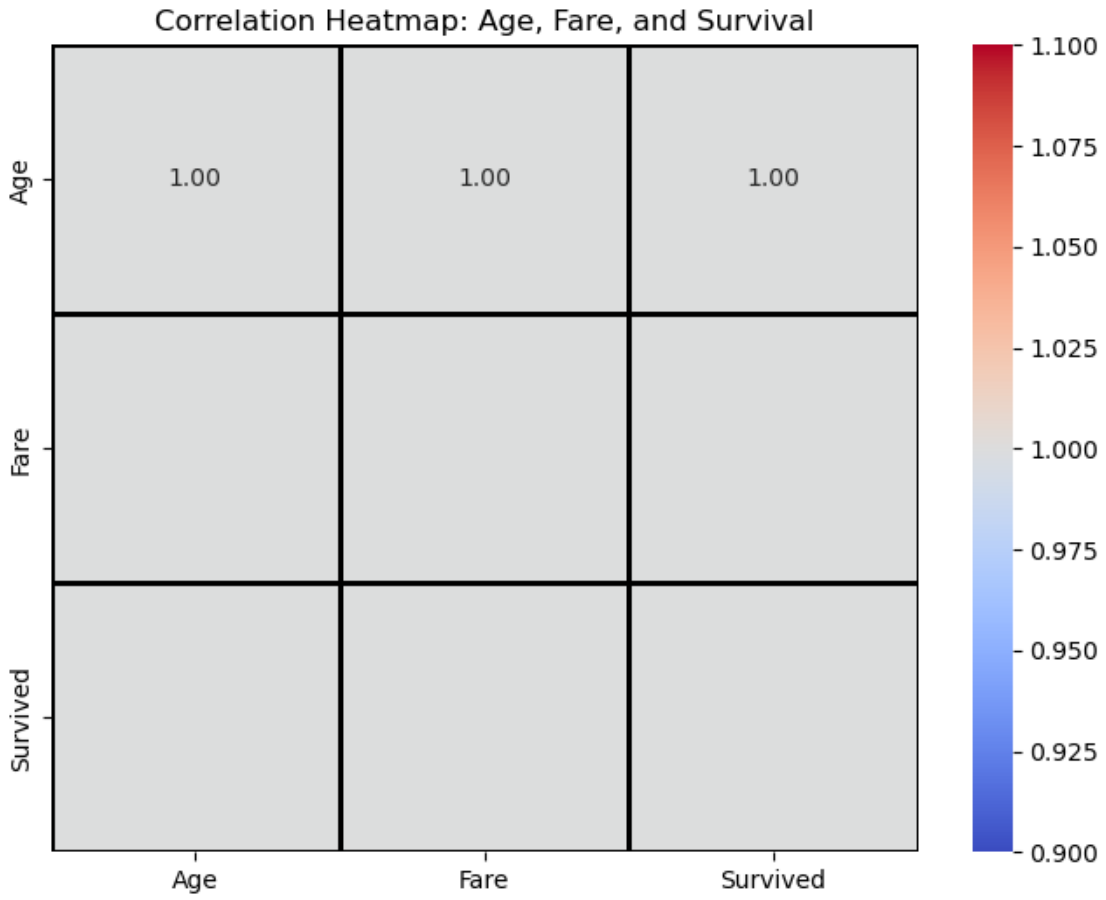
Titanic Dataset: Analyze passenger characteristics that affected survival rates.

Sample Inputs and Expected Outputs

[70]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Sample Titanic data
data = {
    'PassengerId': [1, 2],
    'Age': [22.0, 38.0],
    'Fare': [7.25, 71.28],
    'Survived': [0, 1],
    'Pclass': [3, 1],
    'Sex': ['male', 'female']
```

```
}

# Create DataFrame
df_titanic_sample = pd.DataFrame(data)

# Survival rate by Gender and Class
survival_by_gender_class = df_titanic_sample.groupby(['Sex',␣
 ↪'Pclass'])['Survived'].mean().reset_index()

# Display the survival rate by gender and class
print("\nSurvival Rate by Gender and Class:")
print(survival_by_gender_class)

# Heatmap: Correlation between Age, Fare, and Survival
correlation_data = df_titanic_sample[['Age', 'Fare', 'Survived']].corr()

# Plotting the heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_data, annot=True, cmap='coolwarm', linewidths=1,␣
 ↪linecolor='black', fmt='.2f')
plt.title("Correlation Heatmap: Age, Fare, and Survival")
plt.show()
```

```
Survival Rate by Gender and Class:
      Sex  Pclass  Survived
0  female       1       1.0
1    male       3       0.0
```



1.2 Problem Analysis

1.3 Data Constraints**

1.4 Missing values in the Age column (Titanic dataset)

The Titanic dataset, a historical dataset containing information about passengers aboard the ill-fated Titanic ship, is frequently used in data science for classification and predictive modeling tasks. However, before building any machine learning model, it is necessary to preprocess the data to ensure it is in a suitable format. The preprocessing stage addresses various issues, including missing values, categorical data encoding, and imbalanced class distributions. These challenges, if left unaddressed, can significantly affect the accuracy and efficiency of machine learning models. This paper discusses the constraints and preprocessing techniques that should be applied to the Titanic dataset.

Solutions for Missing Values Several strategies can be employed to handle missing data in the Age column:

Imputation

Mean/Median Imputation: The missing values can be replaced with the mean or median of the

existing values in the column. The median is often preferred in the case of numerical data with outliers, as it is less sensitive to extreme values. Predictive Imputation: Missing values can be predicted using machine learning algorithms, such as regression or k-Nearest Neighbors (KNN), which can use other features in the dataset (e.g., class, sex, or fare) to estimate the missing age values. Mode Imputation: Although typically used for categorical data, mode imputation can be applied to replace missing values with the most frequent value in certain situations. Removal:

As a last resort, rows with missing values may be removed if the proportion of missing data is small. However, imputation is generally recommended when there is a significant amount of data missing.

Categorical data (e.g., Sex, Embarked) requiring encoding

The Titanic dataset contains several categorical features, such as Sex and Embarked, which need to be converted into a numerical format, as most machine learning models require numerical inputs.

Encoding Methods There are various methods to encode categorical variables:

Label Encoding

Label encoding assigns each unique category in a feature an integer value. This method works well for ordinal variables (those with a meaningful order), such as Pclass (Passenger Class). However, it is not suitable for nominal variables, such as Sex and Embarked, because the algorithm may misinterpret the integer labels as having a hierarchical relationship. One-Hot Encoding:

One-hot encoding is a popular method for encoding nominal categorical variables. It creates a binary column for each unique category in a feature, where a value of 1 indicates the presence of that category and 0 indicates its absence. For example, the Embarked feature can be transformed into three binary columns: Embarked_C, Embarked_Q, and Embarked_S, each representing one of the embarkation ports. Target Encoding:

Target encoding involves replacing each category with the mean of the target variable for that category. This method can be useful when the categorical variable has a high cardinality or when there is a strong correlation between the categorical feature and the target.

Imbalanced class distributions (e.g., survival rates)

Another common issue in the Titanic dataset is imbalanced class distributions. The Survived feature, which indicates whether a passenger survived (1) or did not survive (0), is highly imbalanced, with a larger proportion of passengers not surviving.

Addressing Class Imbalance Imbalanced datasets can lead to biased model predictions, as models tend to predict the majority class more often. Several strategies can mitigate this issue:

Resampling:

Oversampling the Minority Class: One method is to artificially increase the number of samples in the minority class (survived passengers) to balance the class distribution. The SMOTE (Synthetic Minority Over-sampling Technique) algorithm can be used to generate synthetic data points for the minority class by interpolating between existing samples. Undersampling the Majority Class: Another approach is to reduce the number of samples in the majority class (non-survived passengers). However, this method can lead to a loss of valuable data. Class Weights Adjustment:

Some machine learning algorithms, such as logistic regression and random forests, allow the adjustment of class weights. By assigning higher weights to the minority class, these models can pay

more attention to the underrepresented class during training.

Data Types and Preprocessing Needs

Numerical Data: Requires scaling/normalization

Standardization (Z-score Normalization): Standardization transforms the data to have a mean of 0 and a standard deviation of 1. It is particularly useful when the data is normally distributed.

Standardization (Z-score Normalization)

The formula for Standardization is:

$$\text{Standardized Value} = \frac{\text{Feature Value} - \mu}{\sigma}$$

Where: - μ is the mean of the feature. - σ is the standard deviation of the feature.

Min-Max Scaling

The formula for Min-Max Scaling is:

$$\text{Scaled Value} = \frac{\text{Feature Value} - \text{Min Value}}{\text{Max Value} - \text{Min Value}}$$

Where: - Min Value is the minimum value of the feature. - Max Value is the maximum value of the feature.

Robust Scaling: This technique scales the data using the median and interquartile range (IQR), making it more robust to outliers.

Categorical Data: Needs one-hot encoding As previously mentioned, categorical variables such as Sex and Embarked need to be encoded. One-hot encoding is the most common technique for nominal variables. However, for high-cardinality categorical features, target encoding may be more efficient.

Missing Values: Requires imputation or removal Beyond the Age column, other features in the Titanic dataset, such as Cabin and Embarked, may also have missing values. These can be handled using similar imputation methods or, in the case of Cabin, by flagging missing values with a new category ("Unknown").

Solution Explanation

EDA Process Workflow

Load dataset Import the necessary libraries (e.g., pandas, numpy, matplotlib, seaborn). Load the dataset

Perform summary statistics Calculate basic statistics to understand the central tendency, spread, and shape of the data

Visualize distributions Use visualization tools like histograms, boxplots, and density plots to understand the distribution of numerical features

Handle missing values Check for missing values in the dataset and decide how to handle them (e.g., imputation or removal)

Examine correlations Identify relationships between numerical features using correlation matrices and heatmaps. This helps to understand the relationships between various wine attributes.

Engineer new features Create new features from existing ones (e.g., combine features like acidity levels, create bins for quality scores)

1.5 Worked Example : Exercises with the Titanic Dataset

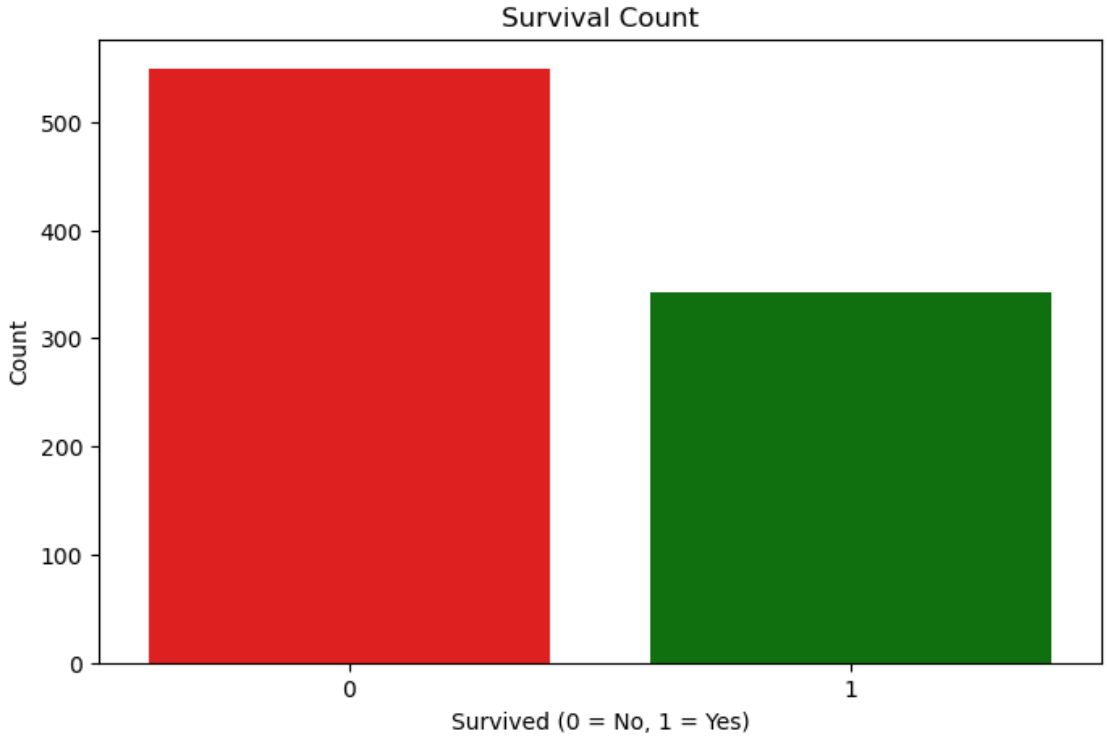
[74]:
```
# Load Titanic dataset directly from Seaborn
df_titanic = sns.load_dataset("titanic")

# Dataset info
print("\nDataset Overview:\n")
print(df_titanic.info())

# Summary Statistics
print("\nSummary Statistics:\n", df_titanic.describe())

# Check for missing values
print("\nMissing Values:\n", df_titanic.isnull().sum())

# Fill missing values
df_titanic["age"].fillna(df_titanic["age"].median(), inplace=True)
df_titanic["embarked"].fillna(df_titanic["embarked"].mode()[0], inplace=True)

# Checking the updated missing values
print("\nUpdated Missing Values:\n", df_titanic.isnull().sum())
```

```
Dataset Overview:

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   survived     891 non-null    int64
 1   pclass       891 non-null    int64
 2   sex          891 non-null    object
 3   age          714 non-null    float64
 4   sibsp        891 non-null    int64
 5   parch        891 non-null    int64
 6   fare         891 non-null    float64
 7   embarked     889 non-null    object
 8   class        891 non-null    category
 9   who          891 non-null    object
 10  adult_male   891 non-null    bool
 11  deck         203 non-null    category
 12  embark_town  889 non-null    object
 13  alive        891 non-null    object
 14  alone        891 non-null    bool
```

```
dtypes: bool(2), category(2), float64(2), int64(4), object(5)
memory usage: 80.7+ KB
None

Summary Statistics:
          survived      pclass         age       sibsp       parch        fare
count  891.000000  891.000000  714.000000  891.000000  891.000000  891.000000
mean     0.383838    2.308642   29.699118    0.523008    0.381594   32.204208
std      0.486592    0.836071   14.526497    1.102743    0.806057   49.693429
min      0.000000    1.000000    0.420000    0.000000    0.000000    0.000000
25%      0.000000    2.000000   20.125000    0.000000    0.000000    7.910400
50%      0.000000    3.000000   28.000000    0.000000    0.000000   14.454200
75%      1.000000    3.000000   38.000000    1.000000    0.000000   31.000000
max      1.000000    3.000000   80.000000    8.000000    6.000000  512.329200

Missing Values:
 survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64

Updated Missing Values:
 survived         0
pclass           0
sex              0
age              0
sibsp            0
parch            0
fare             0
embarked         0
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
```

```
alone            0
dtype: int64
```

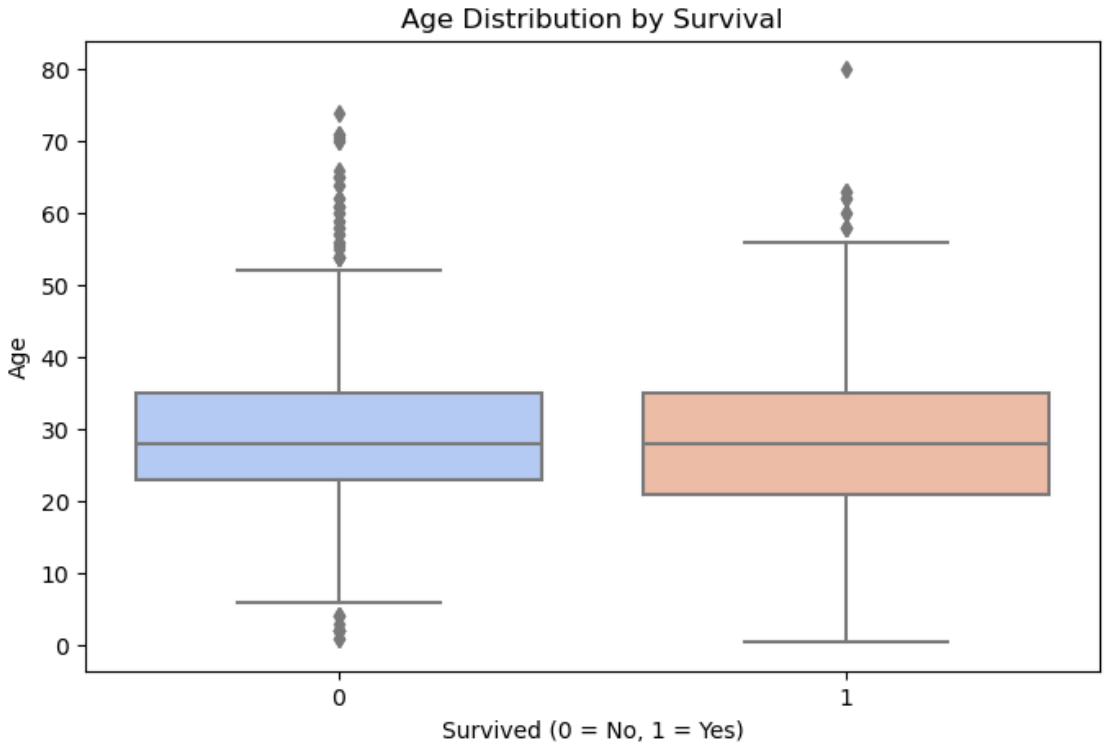
[75]:
```
# Survival Rate Count Plot
plt.figure(figsize=(8, 5))
sns.countplot(x="survived", data=df_titanic, palette=["red", "green"])
plt.title("Survival Count")
plt.xlabel("Survived (0 = No, 1 = Yes)")
plt.ylabel("Count")
plt.show()
```



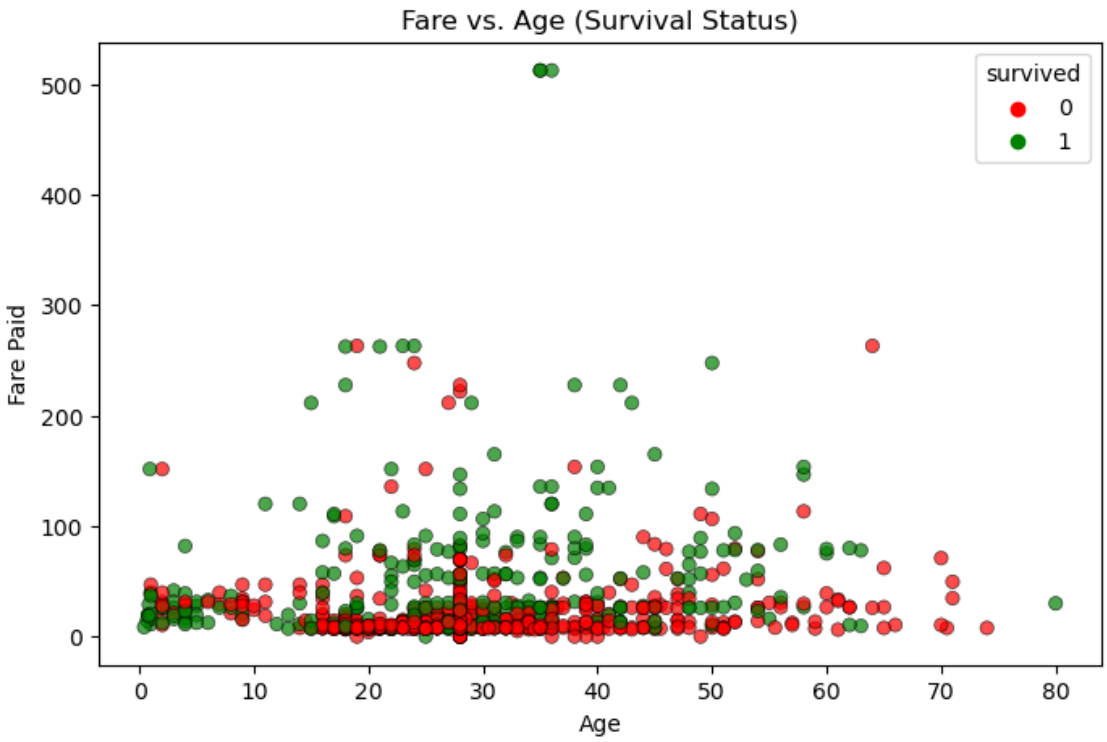
Findings: Petal length and petal width are strong indicators of species classification.

Insights: Feature relationships indicate clear separability between species.

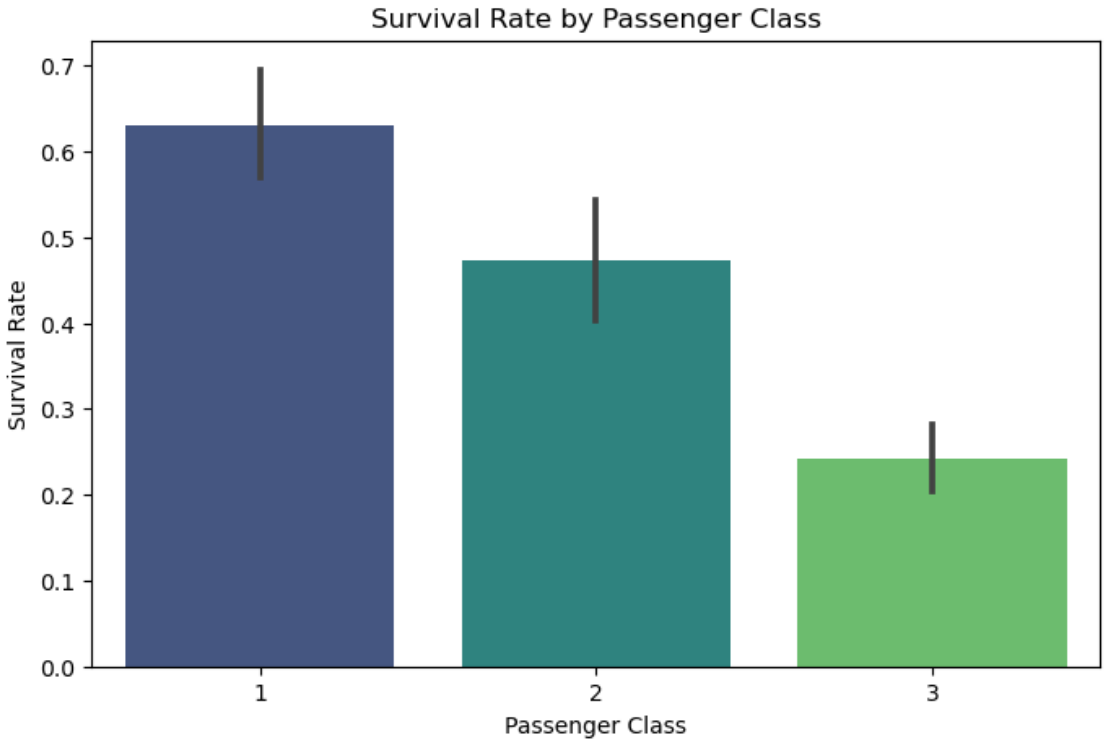
[77]:
```
# Box plot: Age vs. Survival
plt.figure(figsize=(8, 5))
sns.boxplot(x="survived", y="age", data=df_titanic, palette="coolwarm")
plt.title("Age Distribution by Survival")
plt.xlabel("Survived (0 = No, 1 = Yes)")
plt.ylabel("Age")
plt.show()
```



[78]:
```
# Scatter plot: Fare vs. Age (Colored by Survival)
plt.figure(figsize=(8, 5))
sns.scatterplot(x="age", y="fare", hue="survived", data=df_titanic, alpha=0.7,␣
↪edgecolor="black", palette=["red", "green"])
plt.title("Fare vs. Age (Survival Status)")
plt.xlabel("Age")
plt.ylabel("Fare Paid")
plt.show()
```

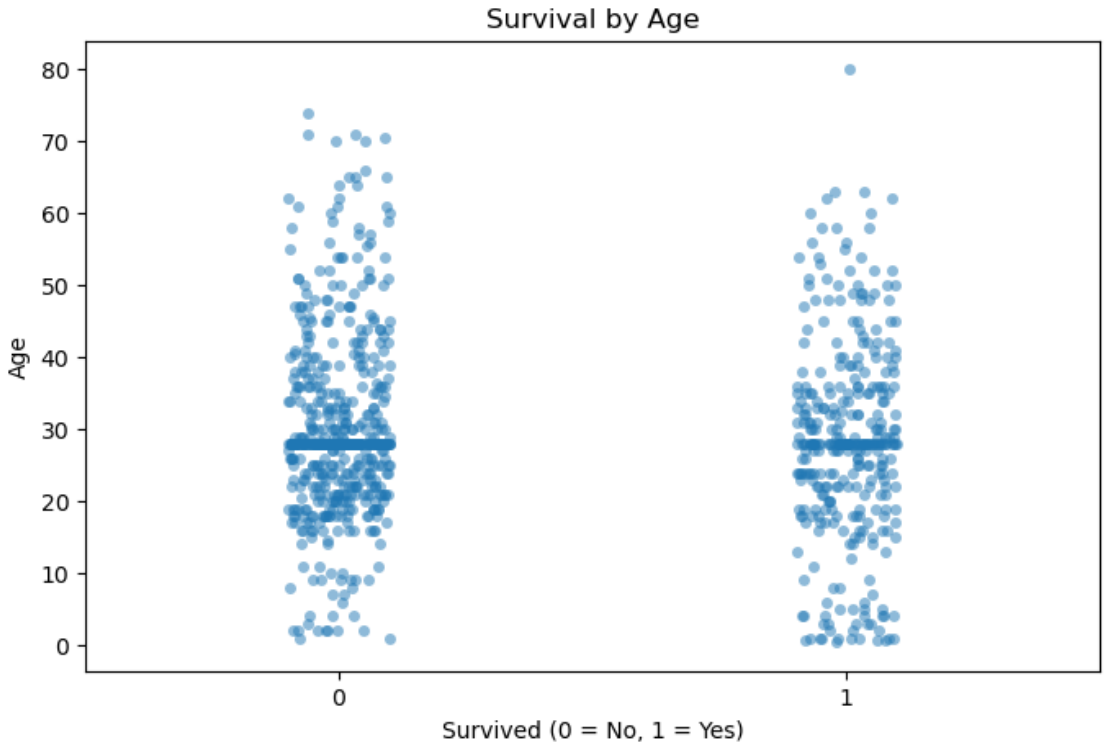


[79]:
```
# Bar plot: Survival Rate by Class
plt.figure(figsize=(8, 5))
sns.barplot(x="pclass", y="survived", data=df_titanic, palette="viridis")
plt.title("Survival Rate by Passenger Class")
plt.xlabel("Passenger Class")
plt.ylabel("Survival Rate")
plt.show()
```

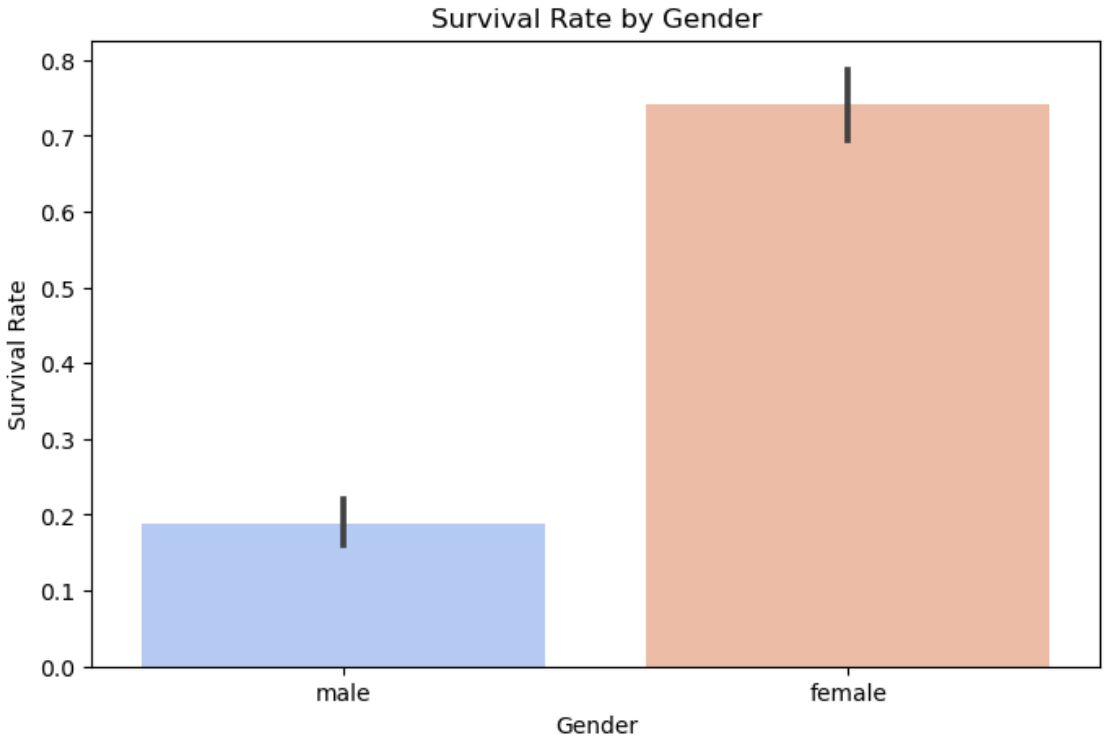


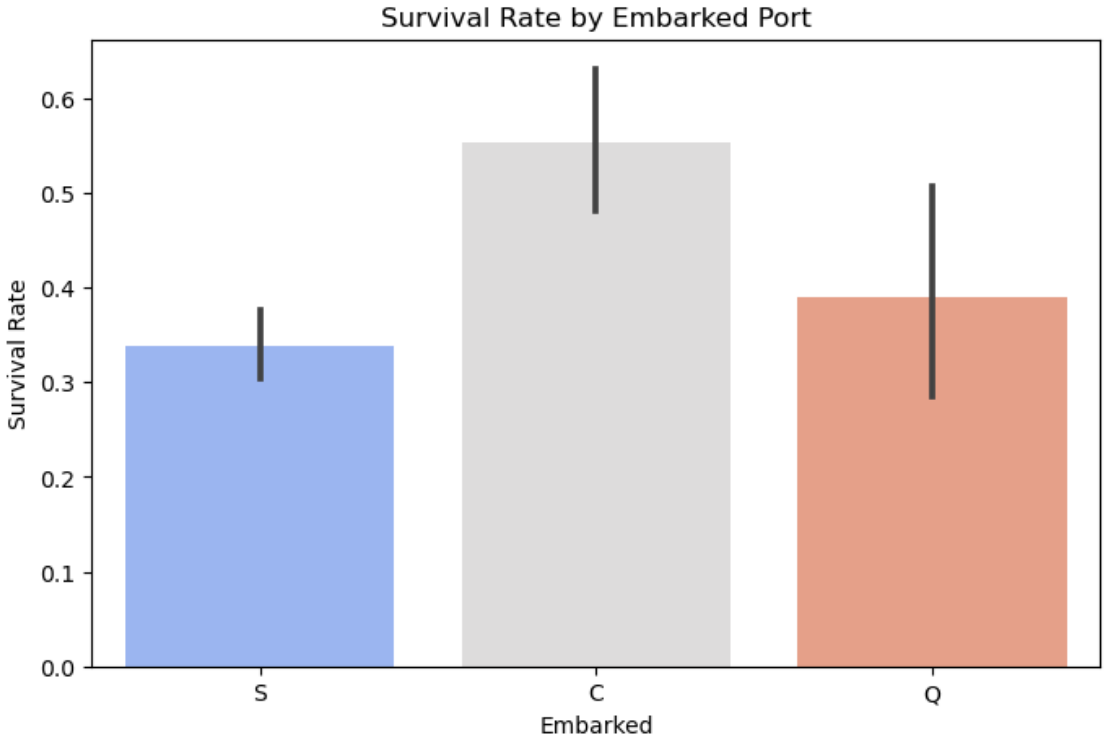
[80]:
```
# Dotted Graph (Survival vs. Age)
plt.figure(figsize=(8, 5))
sns.stripplot(x="survived", y="age", data=df_titanic, jitter=True, alpha=0.5)
plt.title("Survival by Age")
plt.xlabel("Survived (0 = No, 1 = Yes)")
plt.ylabel("Age")
plt.show()
```

```
/opt/conda/envs/anaconda-2024.02-py310/lib/python3.10/site-
packages/seaborn/_oldcore.py:1119: FutureWarning: use_inf_as_na option is
deprecated and will be removed in a future version. Convert inf values to NaN
before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/envs/anaconda-2024.02-py310/lib/python3.10/site-
packages/seaborn/_oldcore.py:1119: FutureWarning: use_inf_as_na option is
deprecated and will be removed in a future version. Convert inf values to NaN
before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



[81]:
```
# Bar plot: Survival Rate by Gender
plt.figure(figsize=(8, 5))
sns.barplot(x="sex", y="survived", data=df_titanic, palette="coolwarm")
plt.title("Survival Rate by Gender")
plt.xlabel("Gender")
plt.ylabel("Survival Rate")
plt.show()

# Survival rate by Embarked port
plt.figure(figsize=(8, 5))
sns.barplot(x="embarked", y="survived", data=df_titanic, palette="coolwarm")
plt.title("Survival Rate by Embarked Port")
plt.xlabel("Embarked")
plt.ylabel("Survival Rate")
plt.show()
```


Survival Rate by Gender
Survival Rate
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
male
female
Gender


Survival Rate by Embarked Port
Survival Rate
0.0
0.1
0.2
0.3
0.4
0.5
0.6
S
C
Q
Embarked

1.6 Results and Data Analysis

Titanic Dataset Findings

Survival Rate: Women had a much higher survival rate than men.

Feature Importance: Fare and passenger class significantly impacted survival chances.

1.7 Conclusion

EDA is an essential step in data analysis that allows for better decision-making and model performance. Through the Titanic datasets, we demonstrated key EDA techniques, including visualization, correlation analysis, and feature engineering. Additionally, the Iris Dataset exercise reinforced the importance of feature relationships in classification problems.

References

Tukey, John W. "Exploratory Data Analysis." Addison-Wesley, 1977. https://onlinelibrary.wiley.com/doi/10.1002/bimj.4710230408

McKinney, Wes. "Python for Data Analysis." O'Reilly Media, 2017. https://www.oreilly.com/library/view/python-for-data/9781491957653/

Fisher, R.A. "The Use of Multiple Measurements in Taxonomic Problems." Annals of Eugenics, 1936. https://onlinelibrary.wiley.com/doi/10.1111/j.1469-1809.1936.tb02137.x

[ ]: