ibimbing

TITANIC

EXPLORATORY DATA ANALYSIS

ibimbing

Project Introduction

Objective

This project aims to strengthen the fundamental skills in Exploratory Data Analysis (EDA) by applying them to a real-world dataset. The focus lies in understanding data structure, cleaning the dataset, and uncovering basic insights through visual exploration.

Dataset

The dataset used in this project is the well-known Titanic dataset, which provides demographic and survival information of passengers. Key variables include passenger name, gender, age, and survival status.

Scope of Work

- Inspect the dataset structure and types of variables
- Handle missing data to ensure quality and consistency
- Explore key patterns through descriptive statistics and visualizations
- Interpret survival patterns based on gender and age

Tools

The project is built using Python, leveraging powerful libraries such as pandas for data manipulation, matplotlib, and seaborn for visualization.

ibimbing

Head & Tail

```
[4] data.head()
```

	survived	name	sex	age
0	1	Allen, Miss. Elisabeth Walton	female	29.0000
1	1	Allison, Master. Hudson Trevor	male	0.9167
2	0	Allison, Miss. Helen Loraine	female	2.0000
3	0	Allison, Mr. Hudson Joshua Creighton	male	30.0000
4	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.0000

Showing top 5 rows of the data

```
[5] data.tail()
```

	survived	name	sex	age
495	1	Mallet, Mrs. Albert (Antoinette Magnin)	female	24.0
496	0	Mangiavacchi, Mr. Serafino Emilio	male	NaN
497	0	Matthews, Mr. William John	male	30.0
498	0	Maybery, Mr. Frank Hubert	male	40.0
499	0	McCrae, Mr. Arthur Gordon	male	32.0

Showing last 5 rows of the data

Understanding the Dataset

ibimbing

Random Sample

```
[6]  data.sample(5)
```

	survived	name	sex	age
6	1	Andrews, Miss. Kornelia Theodosia	female	63.0
217	0	Nicholson, Mr. Arthur Ernest	male	64.0
423	0	Gill, Mr. John William	male	24.0
136	1	Gracie, Col. Archibald IV	male	53.0
322	1	Young, Miss. Marie Grice	female	36.0

Observations:

1. The dataset contains **numerical columns** (survived & age) as well as **categorical columns** (name & sex)
2. The **survived** column appears to contain binary data (0,1)
3. The **sex** column contains only two values (male or female)
4. The **age** column contains a float data type that represents the year
5. No obvious defect on the data (column name vs its entries), all looks good

Understanding the Dataset

ibimbing

Info Data

```
[7]  data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   survived  500 non-null    int64
 1   name      500 non-null    object
 2   sex       500 non-null    object
 3   age       451 non-null    float64
dtypes: float64(1), int64(1), object(2)
memory usage: 15.8+ KB
```

Observation:

1. The dataset contains 4 columns with 500 rows
2. Only the **age** column has missing values (49 rows)
3. All data types appear to be corresponding to their columns

Understanding the Dataset

ibimbing

Statistical Summary

Splitting columns into **categorical** and n**umerical** types to apply statistical summaries

```
[8]  data.columns
```

```
Index(['survived', 'name', 'sex', 'age'], dtype='object')
```

```
[9]  # group column names based on type
     # it will make our life easier onwards
     categoricals = ['name', 'sex']

     numericals = ['survived', 'age']
```

Explore the Data

ibimbing

Describe Numerical Columns

```
[10] # Syntax numerical statistical summary
     data[numericals].describe()
```

	survived	age
count	500.000000	451.000000
mean	0.540000	35.917775
std	0.498897	14.766454
min	0.000000	0.666700
25%	0.000000	24.000000
50%	1.000000	35.000000
75%	1.000000	47.000000
max	1.000000	80.000000

Observation:

1. The minimum and maximum values make sense for each column
2. Mean ~ 50% (Median) in the **age** column, indicating a somewhat symmetrical distribution
3. The **survived** column is a boolean/binary column since the value is 0 or 1, no need to conclude its symmetry. Only need to check the balance level
4. The **mean** of the **survived** column is 0.54, slightly above 0.5, from which we can infer that there are somewhat more survivors (1) than non-survivors (0).

ibimbing

Describe Categorical Columns

```
[11] # Syntax describe method on categorical data
     data[categoricals].describe()
```

	name	sex
count	500	500
unique	499	2
top	Eustis, Miss. Elizabeth Mussey	male
freq	2	288

Observations:

1. The **sex** column has 2 unique values, male and female
2. Survivors are mostly male, with 288 rows, and the rest are female
3. The **name** column has 2 duplicated values, shown by the 499 unique values instead of 500
4. There are no missing values in the categorical column

ibimbing

Numerical Details

```
[12] for col in numericals:
       print(f"==== {col} ====")
       print(data[col].value_counts(), '\n')
```

```
==== survived ====
survived
1    270
0    230
Name: count, dtype: int64

==== age ====
age
24.0000    23
30.0000    20
36.0000    19
18.0000    14
42.0000    14
45.0000    14
35.0000    14
22.0000    12
28.0000    12
23.0000    11
21.0000    11
```

Observations:

1. The dataset shows that **270** passengers survived, which is slightly higher than the **230** passengers who did not survive
2. Ages 24, 30, and 36 were the most common among the other passengers.

ibimbing

Categorical Details

Observations:

1. There is duplicate data found in the name "Eustis, Miss. Elizabeth Mussey"
2. The number of male passengers, which is 288, compared to female passengers, which is only 212, indicates an imbalance in the gender dataset

```
[13] # showing the precise value counts
     # this code is especially useful if we have many categorical columns
     for col in categoricals:
       print(f"Value counts of {col} column")
       print(data[col].value_counts(), '\n')
```

```
Value counts of name column
name
Eustis, Miss. Elizabeth Mussey                2
Becker, Miss. Ruth Elizabeth                  1
Becker, Miss. Marion Louise                   1
Becker, Master. Richard F                     1
Beauchamp, Mr. Henry James                    1
Beane, Mrs. Edward (Ethel Clarke)             1
Beane, Mr. Edward                             1
Bateman, Rev. Robert James                    1
Banfield, Mr. Frederick James                 1
Ball, Mrs. (Ada E Hall)                       1


Value counts of sex column
sex
male      288
female    212
Name: count, dtype: int64
```

Explore the Data

ibimbing

Data Cleaning

Identifying Duplicate Values

```
[16] len(data.drop_duplicates()) / len(data)
     #if the output of the code in this cell is not 1 then there are duplicates
```

```
0.998
```

There are duplicate data in the dataset as proved by the output of the above equation which is below 1

```
[19] # Step 2: Calculate the frequency of occurrence of each duplicate row
     duplicate_counts = duplicates.groupby(list(data.columns)).size().reset_index(name='duplicate_count')

     # Step 3: Sort by number of duplicates
     sorted_duplicates = duplicate_counts.sort_values(by='duplicate_count', ascending=False)

     sorted_duplicates
```

	survived	name	sex	age	duplicate_count
0	1	Eustis, Miss. Elizabeth Mussey	female	54.0	2

The only duplicate data in the dataset was found in the name **Eustis, Miss. Elizabeth Mussey**, who appeared twice

ibimbing

Data Cleaning

Duplicate Handling

```
[20] #Handling Drop duplicate
     data = data.drop_duplicates()
```

Handling duplicate values using **.drop_duplicates()**

```
[21] len(data.drop_duplicates()) / len(data)

     1.0
```

After dropping the duplicates, we check the output of the previous equation, which is now 1, indicating that **there are no duplicate values anymore**

ibimbing

Data Cleaning

Identifying Missing Values

```
[22] # Identifying Missing value
     data.isna().sum()
```

	0
survived	0
name	0
sex	0
age	49

dtype: int64

The percentage of missing values is **below 20%**, so we handle numerically with the **median** and categorically with the mode. Since the categorical column (name and sex) and one of the numerical columns (survived) don't have any missing values, **we're focusing on the numerical age column.**

```
25] # percentage version
    total_rows = len(data)

    # Menghitung dan menampilkan persentase missing values di setiap kolom satu per satu
    for column in data.columns:
        missing_count = data[column].isna().sum()
        missing_percentage = (missing_count / total_rows) * 100
        print(f"Column '{column}' Has {missing_count} missing values ({missing_percentage:.2f}%)") # .2f means 2 decimal
```

```
Column 'survived' Has 0 missing values (0.00%)
Column 'name' Has 0 missing values (0.00%)
Column 'sex' Has 0 missing values (0.00%)
Column 'age' Has 49 missing values (9.82%)
```

Data Cleaning

Handling Missing Values

```
[28] #Handling the missing value specifically on 'age' column using .fillna() method
     data['age'] = data['age'].fillna(data['age'].median())
```

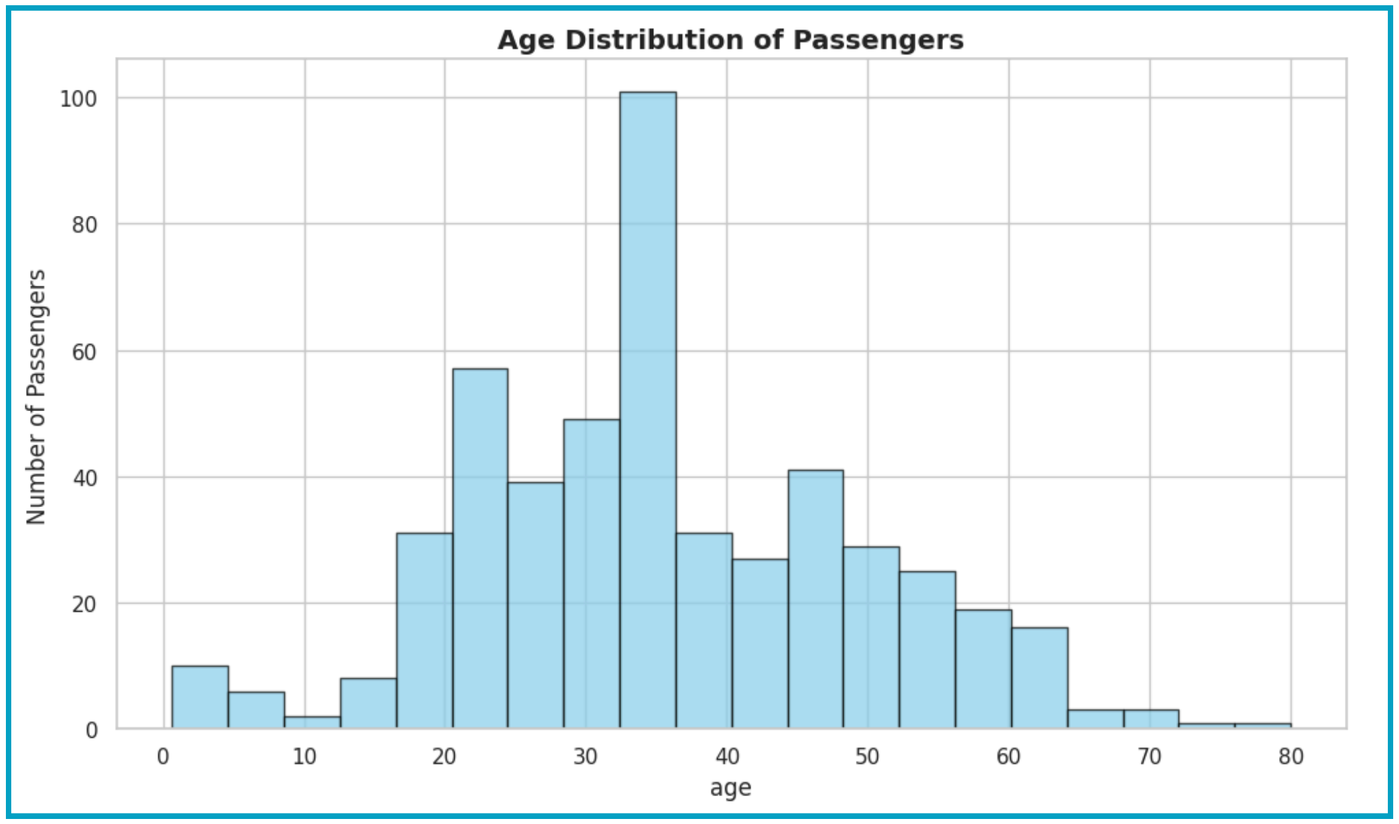
Handling missing values in the **age** column using **.fillna()**

```
[36] # percentage version
     total_rows = len(data)

     # Menghitung dan menampilkan persentase missing values di setiap kolom satu per satu
     for column in data.columns:
         missing_count = data[column].isna().sum()
         missing_percentage = (missing_count / total_rows) * 100
         print(f"Column '{column}' Has {missing_count} missing values ({missing_percentage:.2f}%)") # .2f means 2 decimal

     Column 'survived' Has 0 missing values (0.00%)
     Column 'name' Has 0 missing values (0.00%)
     Column 'sex' Has 0 missing values (0.00%)
     Column 'age' Has 0 missing values (0.00%)
```

```
[29] data.info()

     <class 'pandas.core.frame.DataFrame'>
     Index: 499 entries, 0 to 499
     Data columns (total 4 columns):
      #   Column    Non-Null Count  Dtype
     ---  ------    --------------  -----
      0   survived  499 non-null    int64
      1   name      499 non-null    object
      2   sex       499 non-null    object
      3   age       499 non-null    float64
     dtypes: float64(1), int64(1), object(2)
     memory usage: 19.5+ KB
```

After rechecking, it can be seen that the percentage of missing values for all columns is **0%**, which indicates that **there are no more missing values**, and the data is **ready to be used for the next stage of visualization or analysis**.

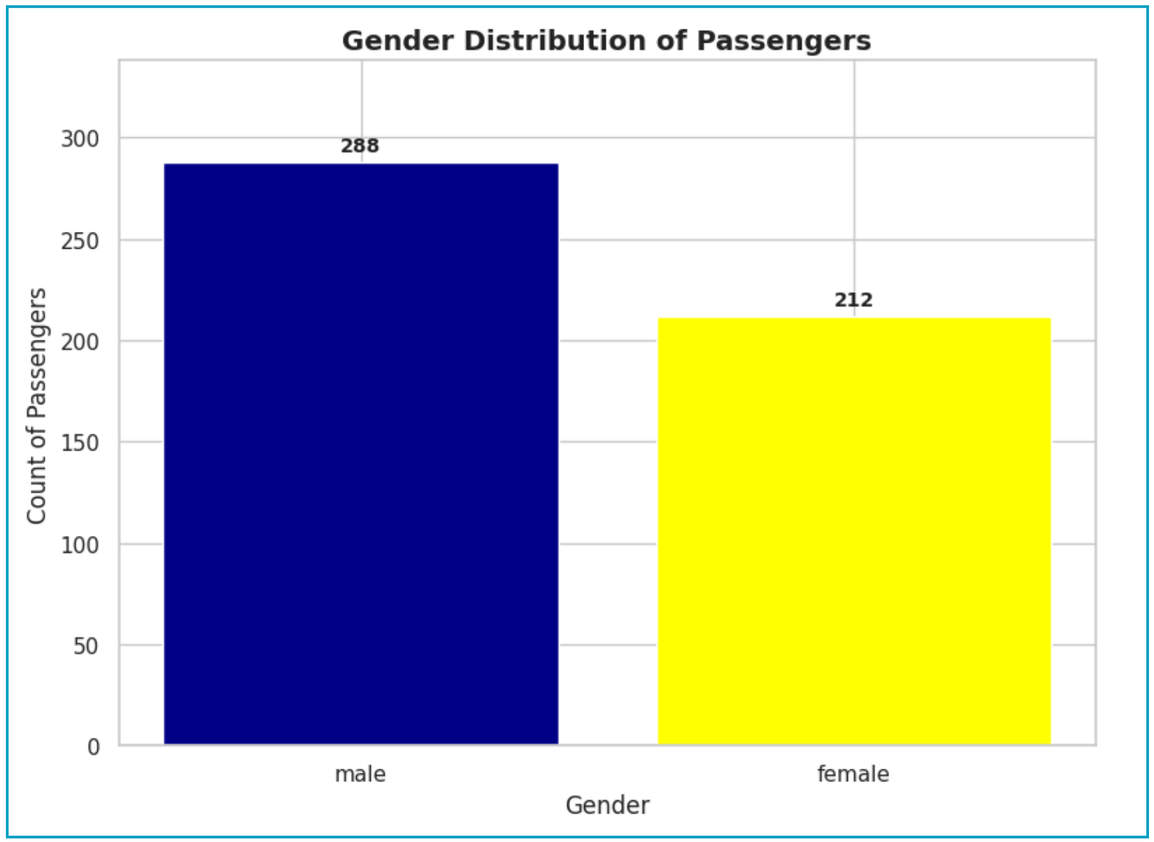
Get to Know the Data



Age Distribution

The age distribution of Titanic passengers reveals a wide range of ages on board, with a concentration of individuals between the ages of **20** and **40**. Notably, there is also a significant number of children under 10 and a small number of elderly passengers above 60.

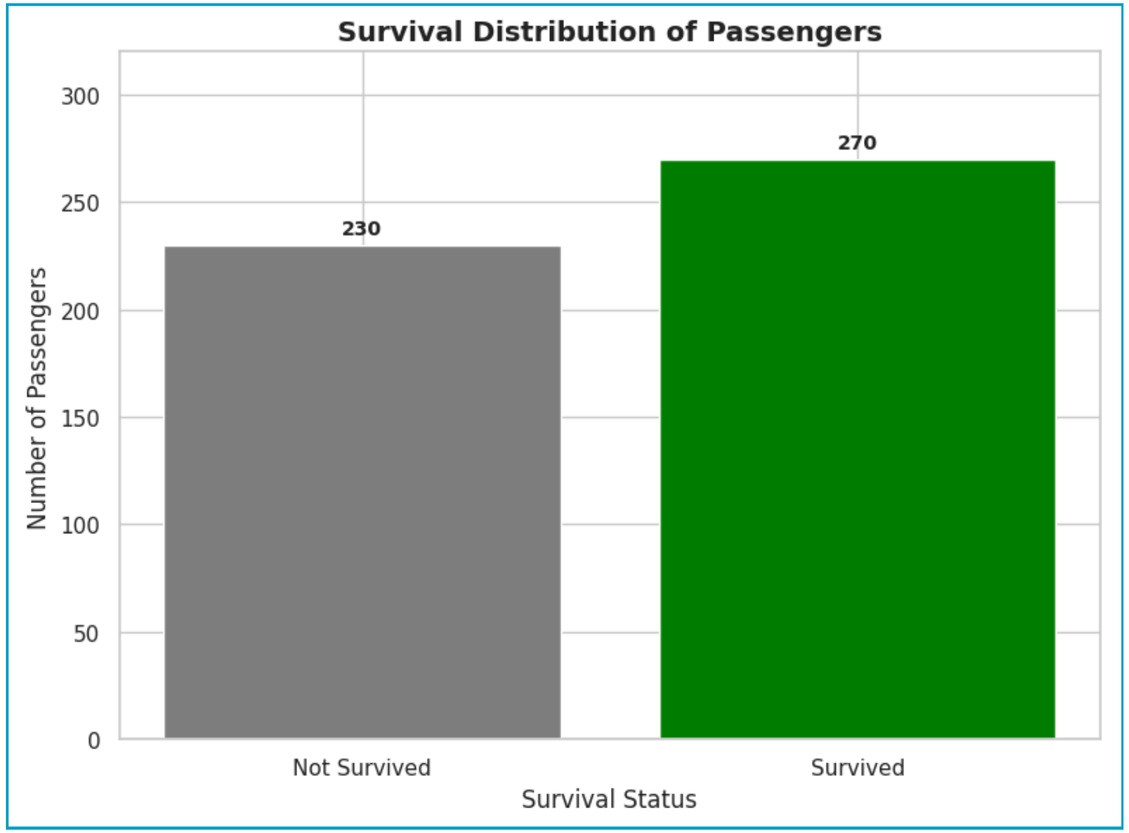
Get to Know the Data



Gender Distribution

The gender distribution reveals that **male passengers made up the majority on board**, with **288 males** compared to **211 females**. This imbalance could play a role in survival outcomes, especially considering **historical lifeboat priorities** during the Titanic tragedy, where the "women and children first" policy might have influenced rescue decisions.

Get to Know the Data

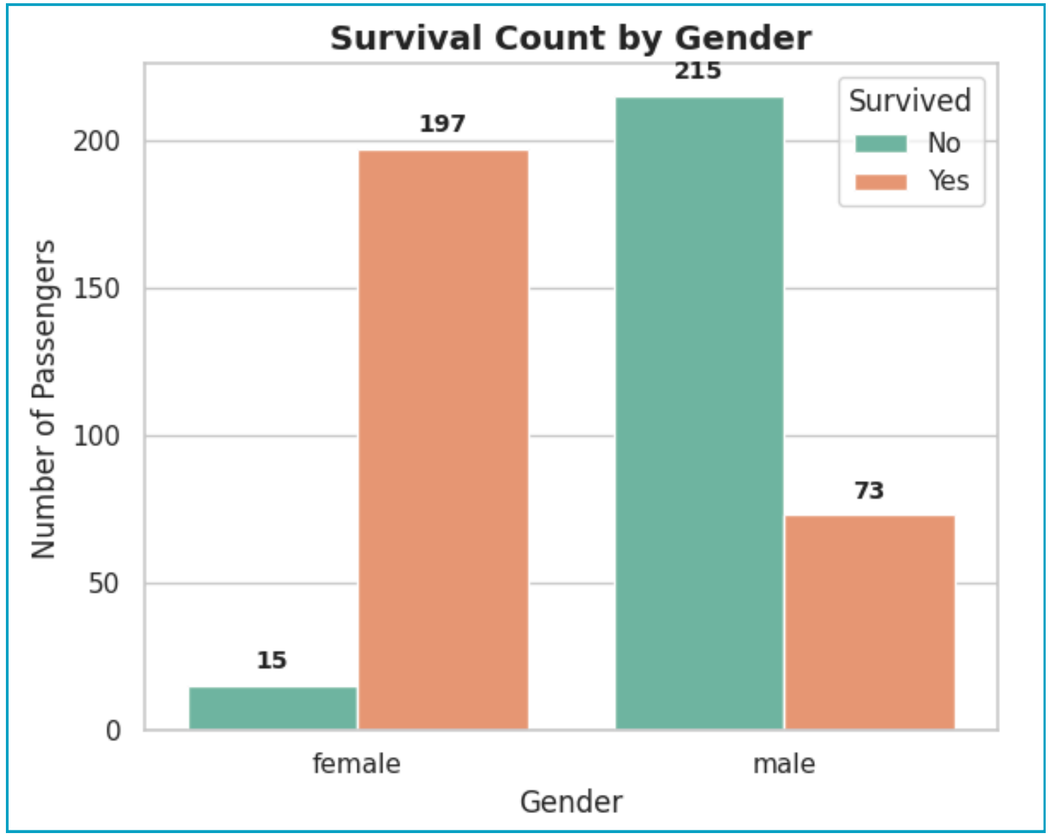


Survival Distribution

The survival distribution shows that **more passengers survived (270)** than perished (230) in the Titanic tragedy.

This indicates a slightly higher survival rate overall, suggesting that **evacuation efforts managed to save more than half of the passengers**. However, the small margin also highlights how **critical and limited** the lifeboat capacity and evacuation time were during the sinking.

Get to Know the Data



Survival Count by Gender

The survival analysis by gender reveals that **female passengers had a significantly higher survival rate** compared to males.

While the total number of male passengers was greater overall, more **females survived** than males. This aligns with the well-documented **“women and children first” policy** that was prioritized during the evacuation process, likely giving **females a better chance of accessing lifeboats** and surviving the disaster.

Thank You

Hadriana Nurul Pertiwi

tiwip0961@gmail.com

onenonlytw

tiwiihn