# Day 1 : Titanic

Sunday, 5 October 2025 6:43 PM

Platform : Google Colab
Py 3.11

(EDA) End-to-End Exploratory Data Analysis.
process to perform investigation on Data to discover patterns, and other things to know your Data well

imp
1. Understand the Data
2. Data cleaning
3. feature selection
4. Assumption checking

## Library Required :

1. Pandas :
   - i) python library for data manipul. and Analysis.
   - ii) Data structured (= Data frames) ↳ Tabular Data.
2. Seaborn :
   - i) statistical Data Visualization library.
   - ii) provides informative breaplots (countplots, histogram, boxplots ...)

```
import pandas as pd
import numpy as nm
import seaborn as sns
import matplotlib.pyplot as plt
```
for visualization

```
sns.set(style='whitegrid')
```

Step 2: Data Loading & Initial Inspection

```
!git clone '<---- url ---->'
df = pd.read_csv('---- csv file path ----')
df
```

**DATA SET**

- Pclass — Passengerclass (1=1st, 2=2nd, 3=3rd)
- Survival — (0=No, 1=Yes)
- name — Name
- sex — Sex
- age — Age
- Sibsp — Number of spouse
- Parch — Number of Parents
- ticket — Ticket Number
- fare — fare
- Cabin — Cabin
- embark — C=cherbourg, S=south, Q= Queenstown,
- boat — Life boat
- body — Body Identification num.
- home.dest — Home Destination

```
df.head()   // first 5 entries
df.Tail()   // last 5 entries
df.shape    // (Rows, Colm)   ← Attribute
df.info()   → Column, NOT NULL, Datatype
```

Problem: Some values are missing
How to fill those values ?

**Column Manipulation**

create new column like Has_cabin (Yes/No) ;

```
print("Descriptive Statistic:")
df.describe()
```
→ count, mean, std, min, 25%, 75%, max

# How To Filter out Data.

Column Name: `df['Age']`  // Display only column
`df['Age'] >= 18`  // Display True & False
`df[df['Age'] >= 18]`  // (df).

```
df[df['Embarked'] == 'C']
```

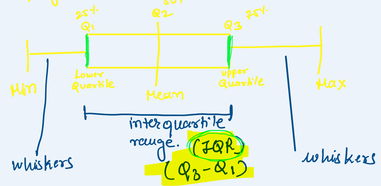
```
df[(df['Sex'] == 'female') &
   (df['Embarked'] == 'C')]
```

```
df[df['Fare'] == df['Fare'].max()]
```

## Step 3 : Data Cleaning

Outliers → An entity which does not belongs to that Range

↳ Perfect Solution




**Missing Value Imputation** (fill with)

1. Numerical Data (Age | Fare) → Median
2. Categorical Data → Mode (most frequent value)
3. High Cardinality / Too many Missing (Cabin) → Drop the column or Create Separate Column

```
df.isnull().sum()   // shows which column has How many NULL values
```

#1 to handle the missing value
Calculate, Print, & fill that Value

```
median_age = df['Age'].median
print(median)
df['Age'].fillna(median_age)
```
⟹ df(Age)

#2 to find mode, with index, Mapping

```
mode_embarked = df['embarked'].mode()[0]
df['Embarked'].fillna(mode_embarked)
```
index mapping: 0 = index, [0] = value

#3 To handle Cabin values.

```
df['Has_cabin'] = df['Cabin'].notna().astype(int)
df.drop('cabin', axis=1, inplace='true')
```
create new column; Table drop; convert object to int (1/0)

check with.

```
df
df['Has_Cabin'].value_counts()
df.isnull().sum()
```

## # STEP : UNIVariated Analysis

- Data is Analysed contains only one Variable.
- To find Patterns
  1. Categorical Variable → Frequency Table, Bar charts (countplot), Pie charts → to see • count or • Ratio.
  2. Numerical Variable → Histogram, Kernel density plot, box plot → to understand • central tendency • spread • outliers

```
print("Analying Categorial features:")
# set up
fig, axes = plt.subplots(2, 3, figsize=(18,12))   (R, C)
fig.suptitle('Univariated Analysis', fontsize=16)
# plotting
sns.countplot(ax=axes[0,0], x='survived', data=df).set_title('survived Distribution')
sns.countplot(ax=axes[0,1], x='Pclass', data=df)
sns.countplot(ax=axes[0,2], x='Gender')
sns.countplot(ax=axes[1,0], x='Sex')
sns.countplot(ax=axes[1,1], x='Embarked')
sns.countplot(ax=axes[1,2], x='---')
plt.tight_layout(react=[0,0,1,0.96])
plt.show()
```

# Numerical Distribution

1. Histogram 2. K.D. (a line over histogram)

```
fig, axes = plt.subplots(1, 2, figsize(16,6))
sns.histplot(ax=axes[0], data=df, x='Age', kde=true, bins=30).set_title('Age Distrib')
plt.show()
```

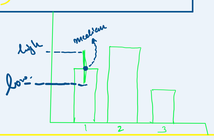
## STEP 5 : Bivariate Analysis

1. Target : Survived — primary

Types :
- (A) Categorical vs Numerical : Barplots, Box plots, Violin Plots
- (B) Categorical vs Categorical : Crosstabs.
- (C) Numerical vs Numerical : Scatter plots

```
fig, Axes = plt.subplots(2, 2, figsize=(16,12))
fig.suptitle('Bivariate Analysis', fontSize=16)
sns.barplot(ax=Axes[0,0], x='Pclass', y='Survived', data=df).set_title('Survival Rate by Pclass')
```
- Sex vs Survived
- Embarked vs Survival
- Has_Cabin vs Survived

```
plt.tight_layout(rect=[0,0,1,0.96])
plt.show()
```

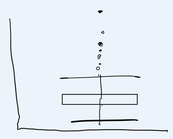



Categorical vs Numerical
[Survival] [Age]

```
g = sns.FacetGrid(df, col='survived', height=6)
g.map(sns.histplot, Age, bins=25, kde=True)
plt.suptitle('survival vs Age', y=1.02)
plt.show
```

## # Deeper Dive — Outliers Analysis

```
plt.figure(figsize=(10,8))
sns.boxplot(y='Fare', data=df)
plt.title('Box plot for Ticket price')
plt.ylabel("Fare")
plt.show().
```



## # Feature Engineering

→ create new features from existing one to uncover & get useful information.

Techniques :
1. Combining features (Sibling + Parents = Family)
2. Extracting from Text (Extracting title from Name)
3. Binning (Binning Age into 'child', 'Adult', 'Senior')

```
# Create a 'family Size' Columns
df['familySize'] = df[sibSp'] + df['Parch'] + 1
# Creating a 'IsAlone' Features
df['isAlone'] = 0
df.loc[df['FamilySize'] == 1, 'IsAlone'] = 1
print("Created 'FamilySize' & 'IsAlone' features")
df[['FamilySize', 'IsAlone']].head()
```
→ Now create barplot using these features

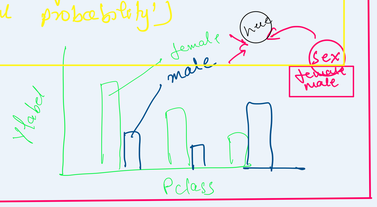
```
# 3. Extract 'Title' from 'Name'
df['Title'] = df['Name'].str.extract('([A-Za-z]+)\.', expand=False)   ← starting with Mr., Mrs., Miss., etc.
# Let's see the different titles
print("Extracted Titles:")
df['Title'].value_counts()
```

```
# To Replace Some Titles with Something
df['Title'] = df['Title'].replace('Lady', 'Countess', ...], 'Rare')
Mlle → Miss
Ms → Miss
Mme → Mrs
```

## # Multivariated Analysis

interaction between Diff Variables Simultaneously

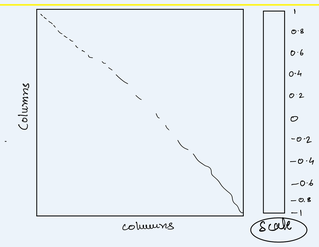
```
sns.catplot(x='Pclass', y='survived', hue='Sex', data=df, kind='bar', ht=6, aspect=1.5)
plt.title('survival Rate by Pclass & sex')
plt.ylabel('survival probability')
plt.show()
```




## # Correlation Analysis

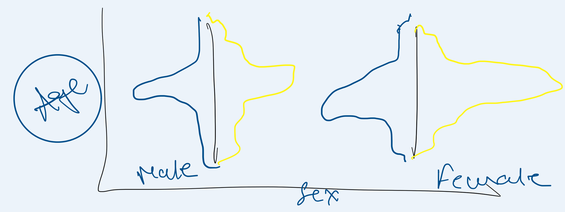
How one value acts on other value
Dependency.

```
plt.figure(figsize=(14,10))
numeric_cols = df.select_dtypes(include=[np.number])
Correlation Matrix = numeric_cols.corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('correlation Matrix of Numerical Feats')
plt.show()
```




## # Violin plot

```
plt.figure(figsize=(14,8))
sns.violinplot(x='Sex', y='Age', hue='survived', data=df, split=True, palette={0:'blue', 1:'orange'})
plt.title('-----')
plt.show()
```




## # Profiling Summery