

✓ The resource of data

<https://www.kaggle.com/competitions/titanic/data>

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.impute import KNNImputer
import warnings
warnings.filterwarnings("ignore")
```

```
train = pd.read_csv("/kaggle/input/competitions/titanic/train.csv")
```

```
train.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath	female	35.0	1	0	113803	53.1000	C123	S

```
train.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

✓ 1 - Handling Missing value

```
train.isnull().sum()
```

```
PassengerId    0
Survived        0
Pclass          0
Name            0
Sex             0
Age            177
SibSp           0
Parch           0
Ticket          0
Fare            0
Cabin          687
Embarked        2
dtype: int64
```

```
missing = train.isnull().mean() * 100
```

```
missing_30 = missing[missing > 30]
```

```
print("Columns with more than 30% missing values")
print(missing_30)
```

```
Columns with more than 30% missing values
Cabin    77.104377
dtype: float64
```

```
threshold = 30
```

```
columns_drop = missing[missing > threshold].index
```

```
# Drop the columns
train_clean = train.drop(columns = columns_drop)
```

```
print(f"Columns dropped: {list(columns_drop)}")
print(f"Remining columns: {train_clean.columns}")
```

```
Columns dropped: ['Cabin']
Remining columns: Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
                        'Parch', 'Ticket', 'Fare', 'Embarked'],
                        dtype='object')
```

```
train_clean['Age'].fillna(train_clean['Age'].mean(), inplace = True)
```

```
mode_value = train_clean['Embarked'].mode()[0]
train_clean["Embarked"].fillna(mode_value, inplace = True)
```

```
train_clean.isnull().sum()
```

```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

2 - Handling Outliers

```
train_clean.dtypes
```

```
PassengerId    int64
Survived       int64
Pclass         int64
Name           object
Sex            object
Age            float64
SibSp          int64
Parch          int64
Ticket         object
Fare           float64
Embarked       object
dtype: object
```

```
numeric = ['int64', 'float64']
new_train = train_clean.select_dtypes(include = numeric)
new_train
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
0	1	0	3	22.000000	1	0	7.2500
1	2	1	1	38.000000	1	0	71.2833
2	3	1	3	26.000000	0	0	7.9250
3	4	1	1	35.000000	1	0	53.1000
4	5	0	3	35.000000	0	0	8.0500
...	...	...	...	...	...	...	...
886	887	0	2	27.000000	0	0	13.0000
887	888	1	1	19.000000	0	0	30.0000
888	889	0	3	29.699118	1	2	23.4500
889	890	1	1	26.000000	0	0	30.0000
890	891	0	3	32.000000	0	0	7.7500

891 rows × 7 columns

```
def detect_outliers(column):
    Q1 = column.quantile(0.25)
    Q3 = column.quantile(0.75)
    IQR = Q3 - Q1
    lower = Q1 - 1.5 * IQR
    upper = Q3 + 1.5 * IQR
    # Replace outliers with NaN
    return column.where((column >= lower) & (column <=
upper), np.nan)
```

```
# Replace outliers with NaN for each feature
for col in new_train.columns:
    new_train[col] = detect_outliers(new_train[col])
```

```
null_value = new_train.isnull().sum()
print("Sum of null values in each column:")
print(null_value)
```

```
Sum of null values in each column:
PassengerId    0
Survived        0
Pclass          0
Age            66
SibSp          46
Parch         213
Fare          116
dtype: int64
```

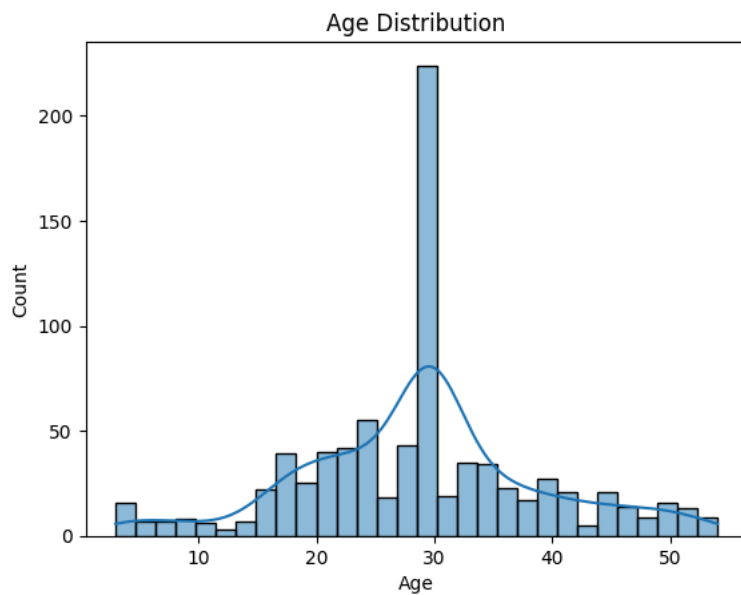
```
# Dealing with missing value after outliers as Lecture 2
imputer = KNNImputer(n_neighbors=2)
data_imputed = pd.DataFrame(imputer.fit_transform(new_train),
columns=new_train.columns)
data_imputed.isnull().sum()
```

```
PassengerId    0
Survived        0
Pclass          0
Age            0
SibSp          0
Parch          0
Fare           0
dtype: int64
```

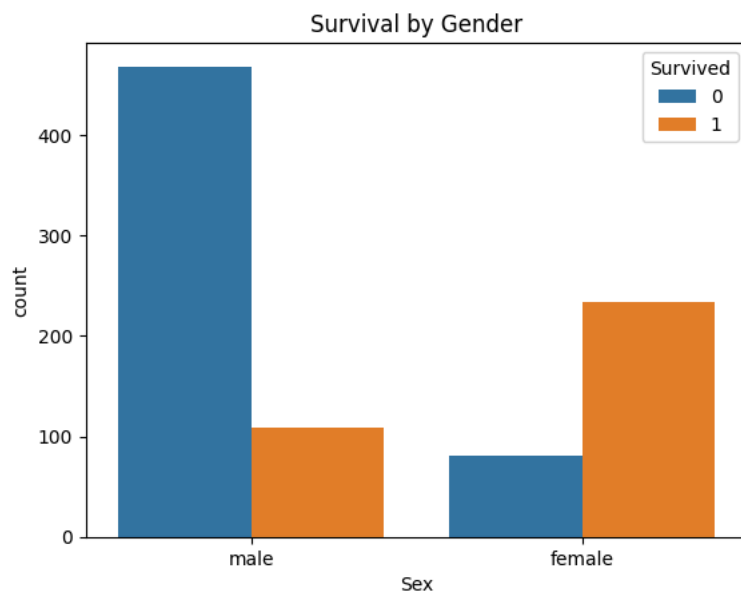
3 - Data Visualization.

```
# 1 - Histogram
plt.figure()
sns.histplot(new_train['Age'], bins=30, kde=True)
plt.title("Age Distribution")
```

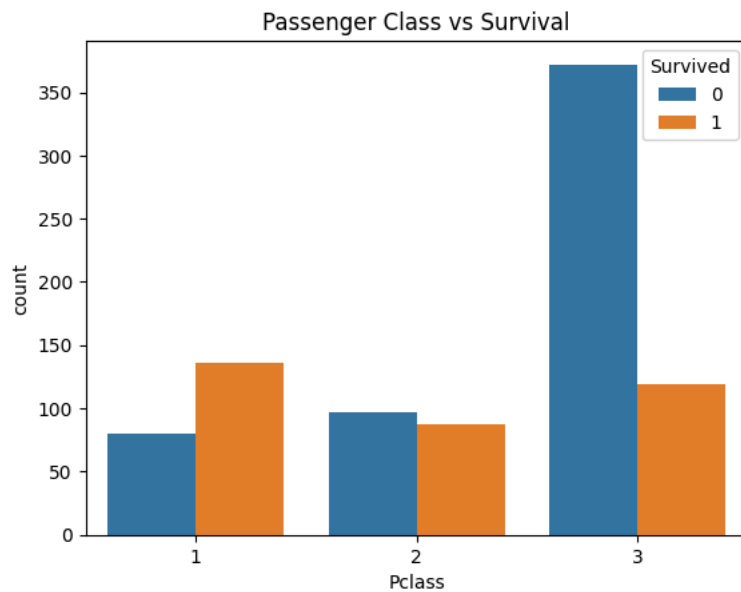
```
plt.xlabel("Age")  
plt.show()
```



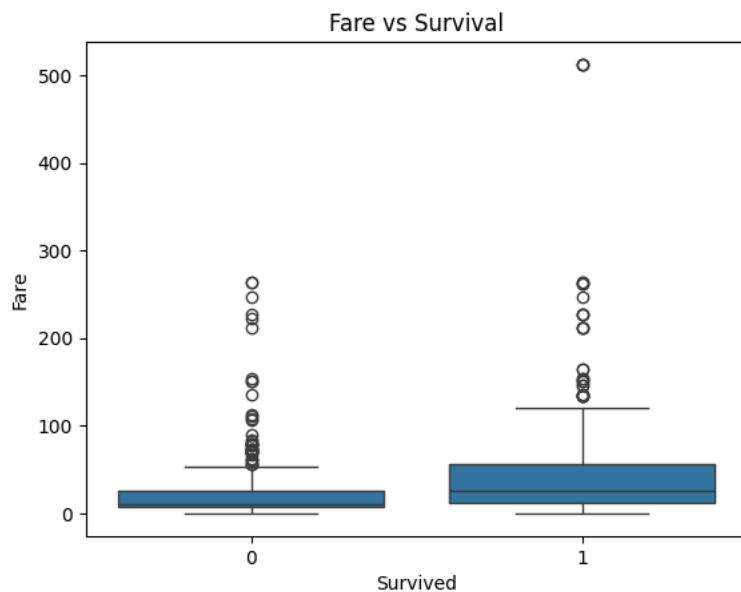
```
# Bar Chart  
plt.figure()  
sns.countplot(x='Sex', hue='Survived', data=train_clean)  
plt.title("Survival by Gender")  
plt.show()
```



```
# Bar Chart  
plt.figure()  
sns.countplot(x='Pclass', hue='Survived', data=train_clean)  
plt.title("Passenger Class vs Survival")  
plt.show()
```



```
# Boxplot
plt.figure()
sns.boxplot(x='Survived', y='Fare', data=train_clean)
plt.title("Fare vs Survival")
plt.show()
```



```
# Correlation Heatmap
plt.figure()
sns.heatmap(train_clean.corr(numeric_only=True), annot=True)
plt.title("Feature Correlation Heatmap")
plt.show()
```

