# SC310005 Artificial Intelligence

## Lecture 4: Seaborn

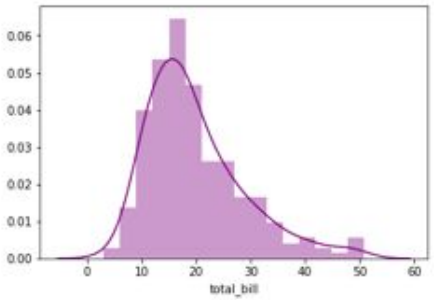
teerapong.pa@chula.ac.th

# Reference

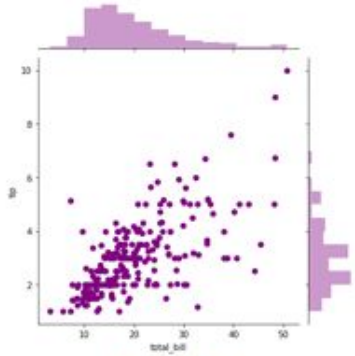

- https://seaborn.pydata.org/
- https://www.codexa.net/seaborn-python/
- https://patelsandeep88.medium.com/python-seaborn-library-for-data-visualization-in-line-plot-graph-35e86d378a45
- https://www.linkedin.com/pulse/data-visualisation-using-seaborn-mukul-kr-singh-chauhan
- https://note.com/kiyo_ai_note/n/nf44b6a9578db
- https://datamahadev.com/13-ultimate-seaborn-tricks-using-python/


Morris Charts
Line Chart
Area Chart
Bar Chart
Donut Chart
25%
Donut Chart
Sparkline Charts
Line Chart
Bar Chart
Pie Chart
Easy Pie Charts
25%
50%
75%
delete
shift
enter

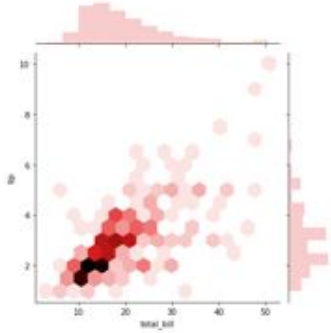
# Seaborn Plots



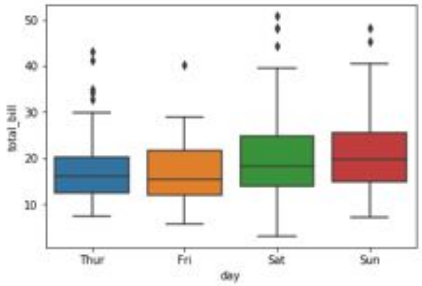

distplot



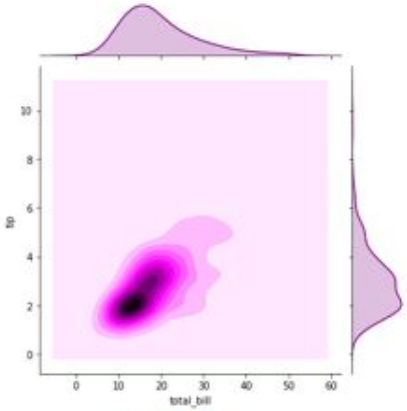

Jointplot



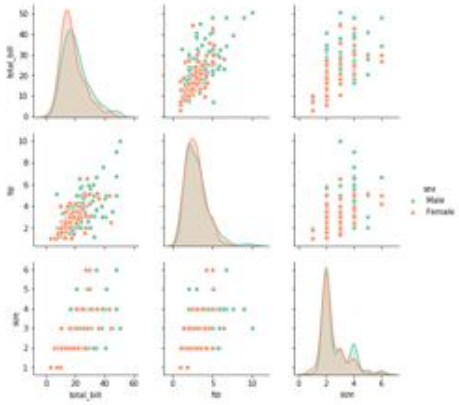

Hexplots



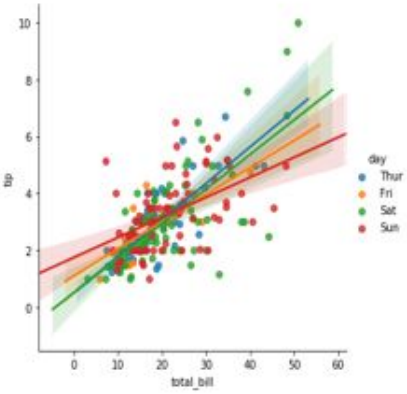

Boxplots



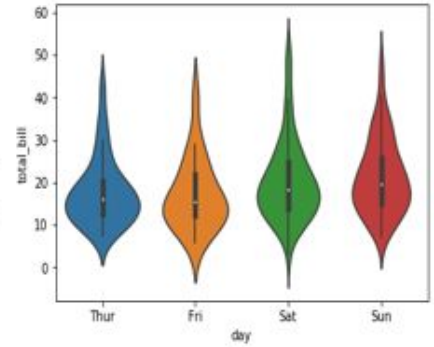

KDE Plot




Pair Plots



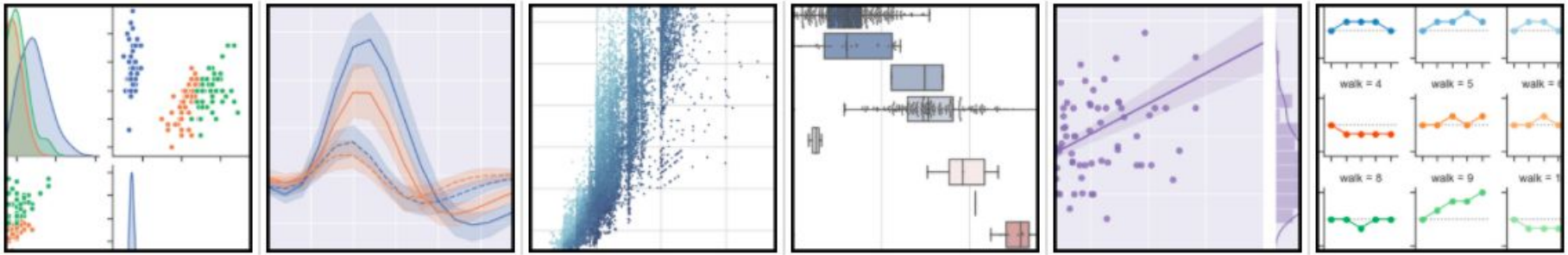

LM Plots




Violin Plots

# seaborn

## データ可視化入門



# What is Seaborn?

**Seaborn (SNS)** is a data visualisation library that helps in creating fancy data visualisations in Python. **Most of the Data Analysis** requires identifying trends and building models. This article will help you get started creating data visualisation using Seaborn library.

**Seaborn** is a library that helps in build us awesome plots and make our life easy. In order to begin with you should type the following command in your jupyter.

https://www.linkedin.com/pulse/data-visualisation-using-seaborn-mukul-kr-singh-chauhan

# Seaborn Python Library

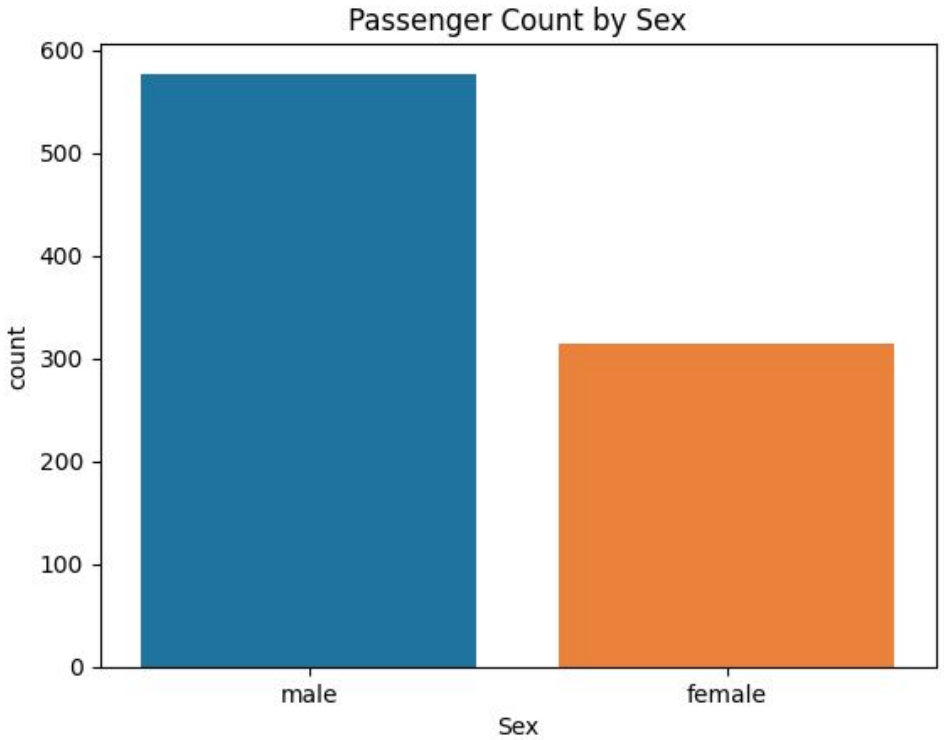
```
import pandas as pd

import seaborn as sns
import matplotlib.pyplot as plt

# Load the Titanic dataset
titanic_data = pd.read_csv('titanic_dataset.csv')
```

## Value Counts with Seaborn Countplot

```
[4]  # Count of passengers by 'Sex'
     sns.countplot(x='Sex', data=titanic_data)
     plt.title('Passenger Count by Sex')
     plt.show()
```

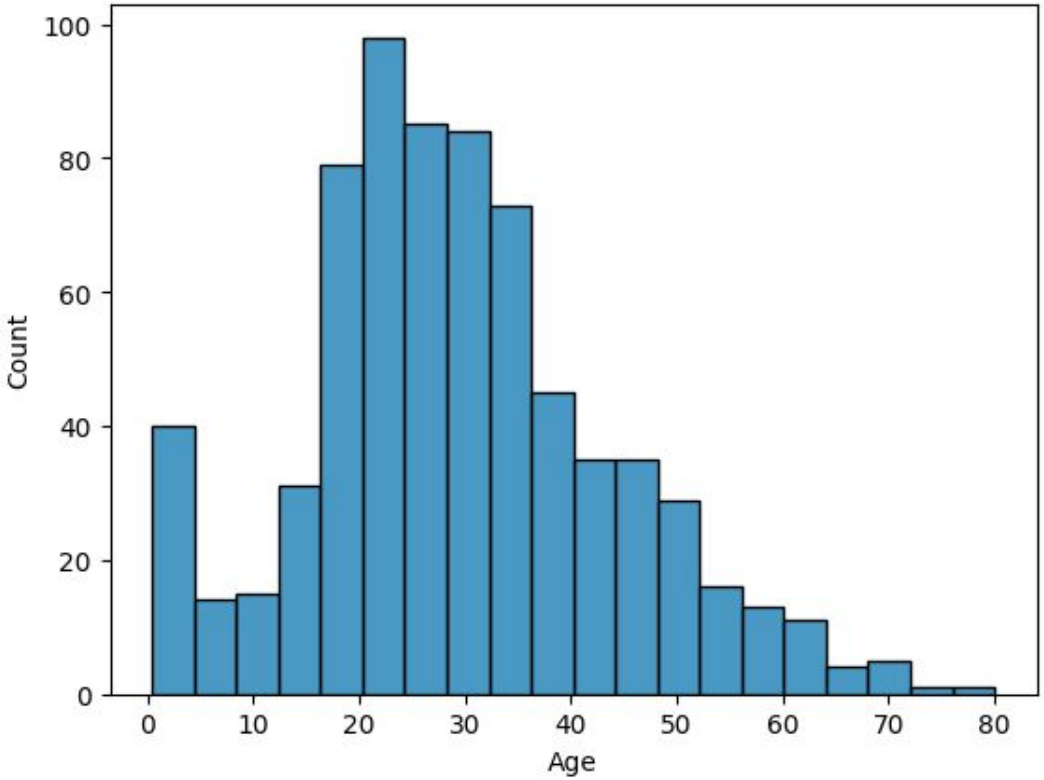


## Histogram of passenger ages using Seaborn

```
[6]  sns.histplot(titanic_data['Age'], bins=20)
```

0s

```
<Axes: xlabel='Age', ylabel='Count'>
```

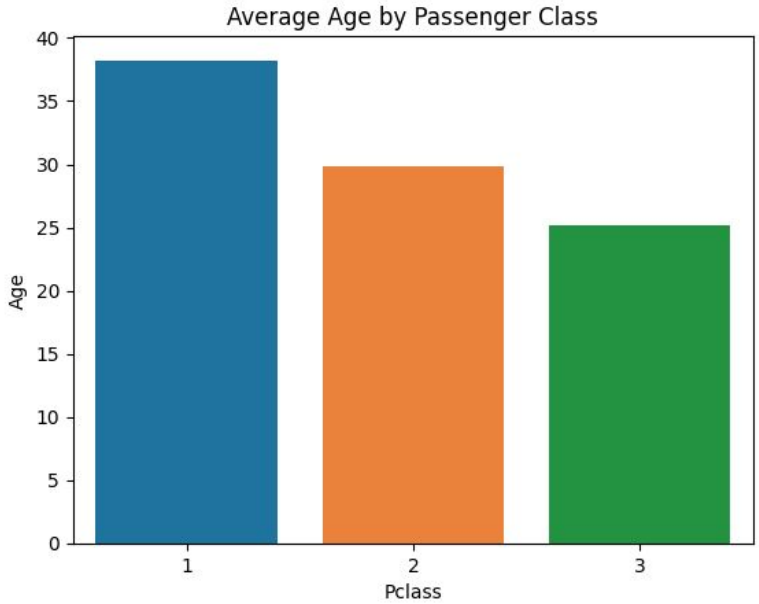


## Groupby and Aggregate with Seaborn Barplot

[7]
```
titanic_data.groupby('Pclass')['Age'].mean()
```

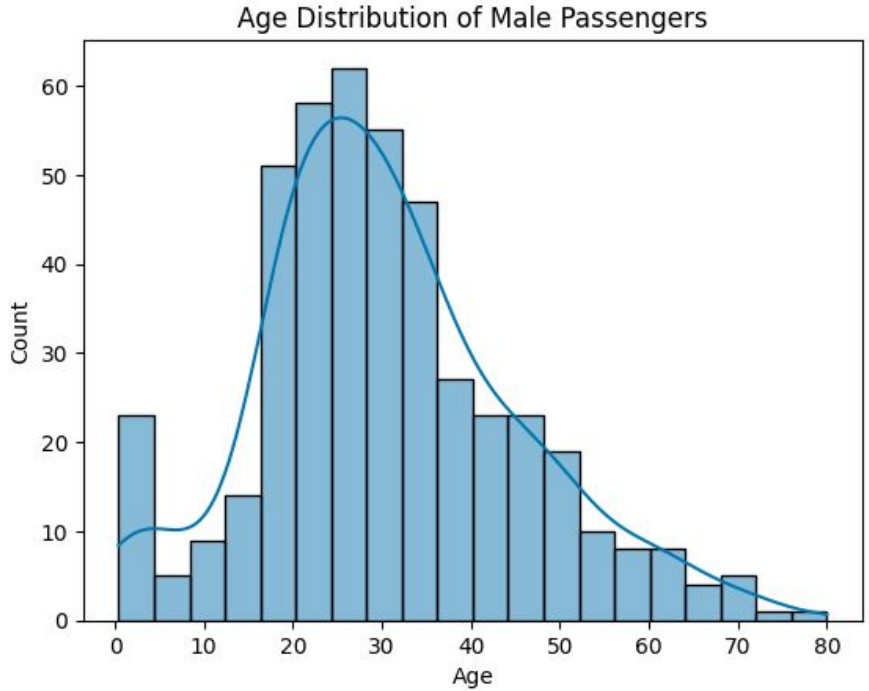
```
Pclass
1    38.233441
2    29.877630
3    25.140620
Name: Age, dtype: float64
```

[8]
```
# Average age of passengers by 'Pclass'
sns.barplot(x='Pclass', y='Age', data=titanic_data.groupby('Pclass')['Age'].mean().reset_index())
plt.title('Average Age by Passenger Class')
plt.show()
```



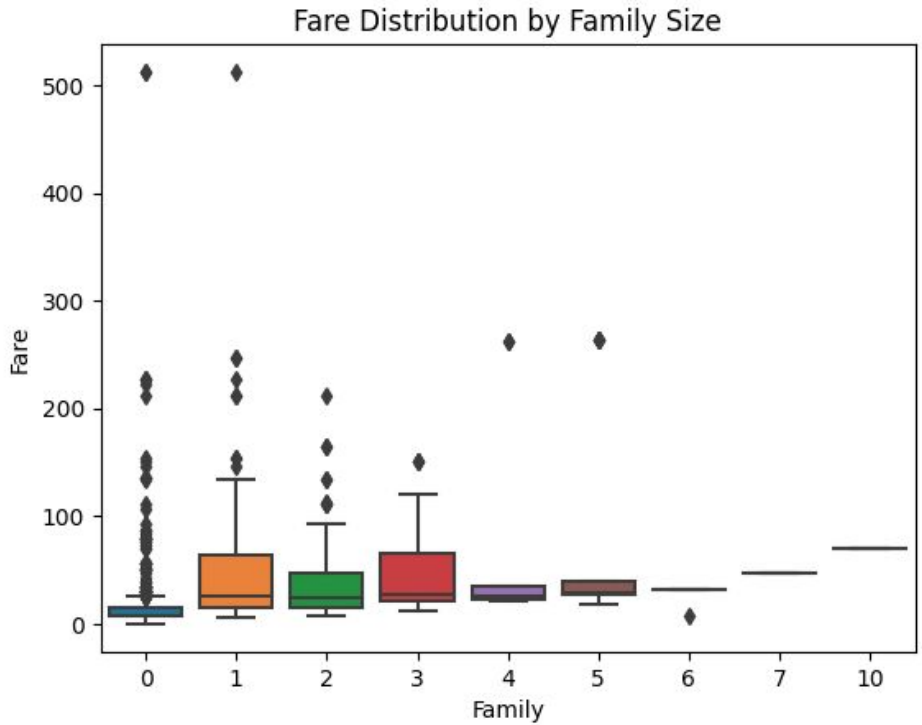
## Condition-based Filtering with Seaborn Histogram

```
[9]  # Age distribution of male passengers
     male_passengers = titanic_data[titanic_data['Sex'] == 'male']
     sns.histplot(male_passengers['Age'].dropna(), bins=20, kde=True)
     plt.title('Age Distribution of Male Passengers')
     plt.show()
```



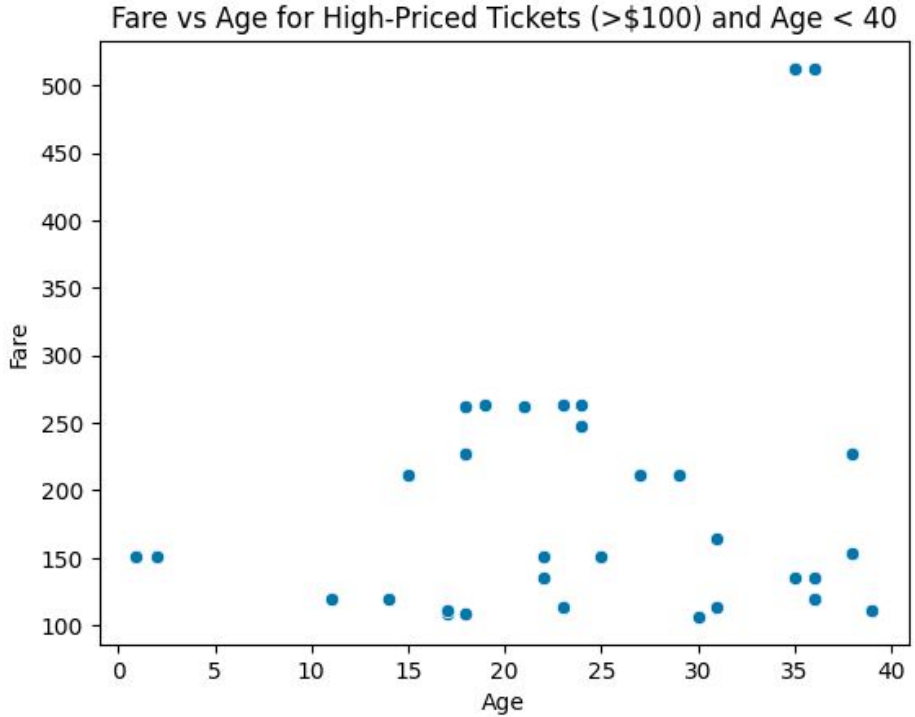
## Apply Function and Create Variable with Seaborn Boxplot

```
[10] # Creating a categorical variable 'Family' based on family size
     titanic_data['Family'] = titanic_data['SibSp'] + titanic_data['Parch']
     sns.boxplot(x='Family', y='Fare', data=titanic_data)
     plt.title('Fare Distribution by Family Size')
     plt.show()
```



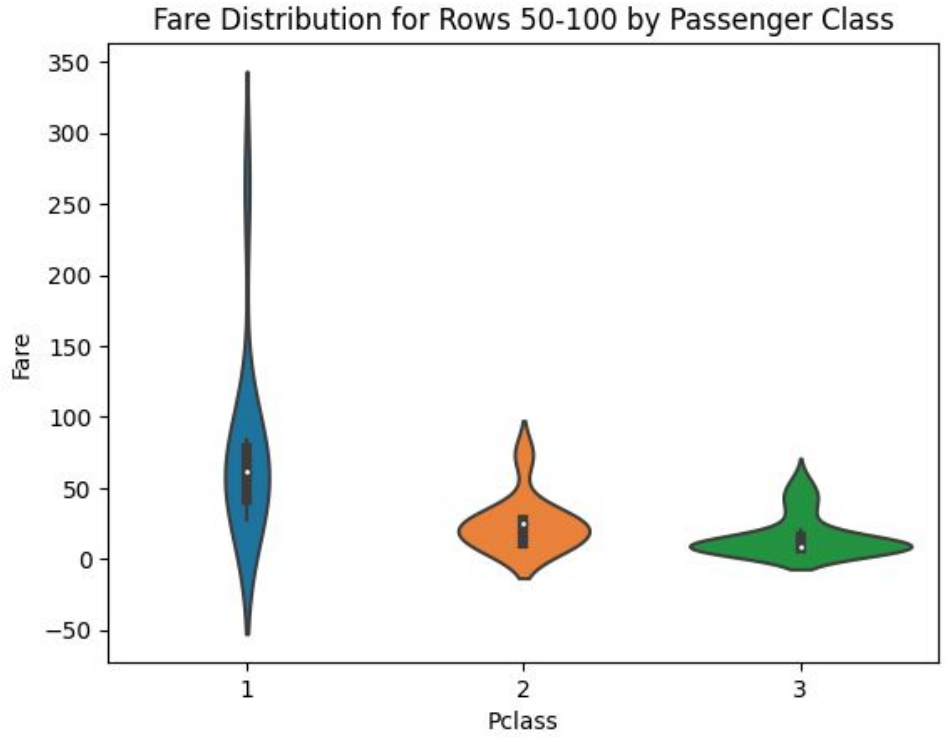
## Location-based Selection (loc) with Seaborn Scatterplot

```
[11] # Selecting passengers who paid more than $100 for their ticket and were younger than 40
     selected_passengers = titanic_data.loc[(titanic_data['Fare'] > 100) & (titanic_data['Age'] < 40)]
     sns.scatterplot(x='Age', y='Fare', data=selected_passengers)
     plt.title('Fare vs Age for High-Priced Tickets (>$100) and Age < 40')
     plt.show()
```



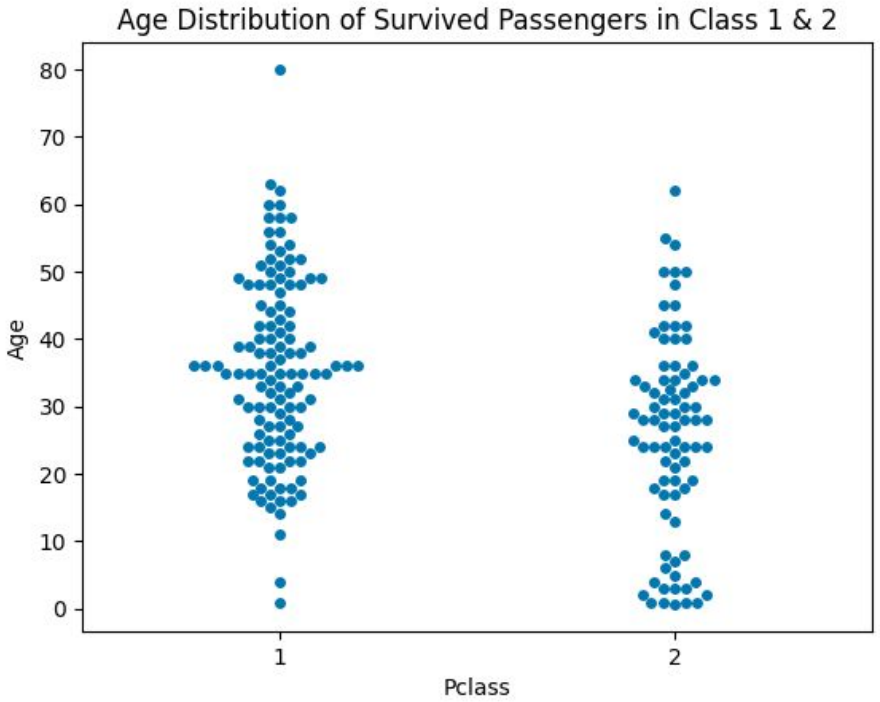
## Index Location-based Selection (iloc) with Seaborn Violinplot

```
[12] # Selecting rows 50 to 100 and plotting the distribution of fares using a violinplot
     sns.violinplot(x='Pclass', y='Fare', data=titanic_data.iloc[50:101])
     plt.title('Fare Distribution for Rows 50-100 by Passenger Class')
     plt.show()
```



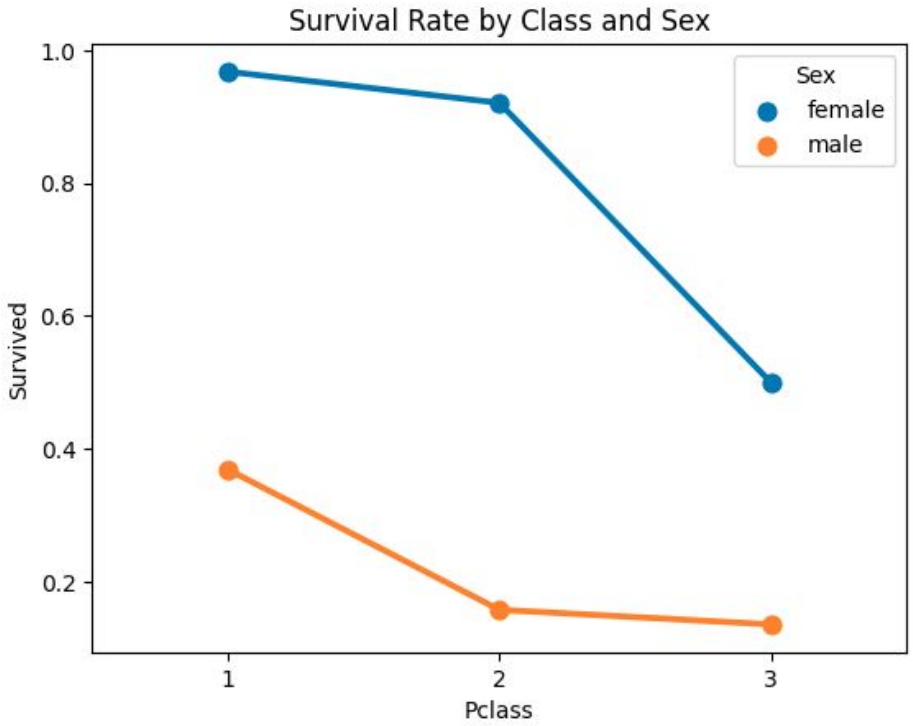
## Two Conditions Selection with Seaborn Swarmplot

```
[13] # Selecting passengers with 'Pclass' 1 or 2 who survived
     survived_class_1_2 = titanic_data[(titanic_data['Survived'] == 1) & (titanic_data['Pclass'].isin([1, 2]))]
     sns.swarmplot(x='Pclass', y='Age', data=survived_class_1_2)
     plt.title('Age Distribution of Survived Passengers in Class 1 & 2')
     plt.show()
```



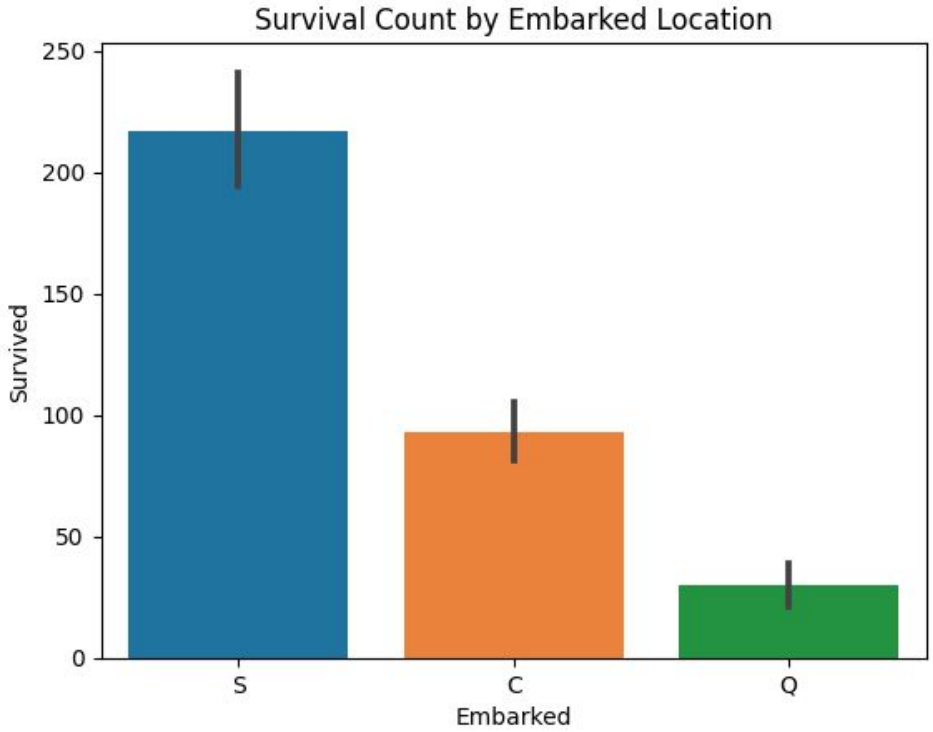
## Groupby and Aggregate with Seaborn Pointplot

```
# Pointplot showing survival rate by 'Pclass' and 'Sex'
sns.pointplot(x='Pclass', y='Survived', hue='Sex', data=titanic_data.groupby(['Pclass', 'Sex'])['Survived'].mean().reset_index())
plt.title('Survival Rate by Class and Sex')
plt.show()
```



## Conditional Count with Seaborn Barplot

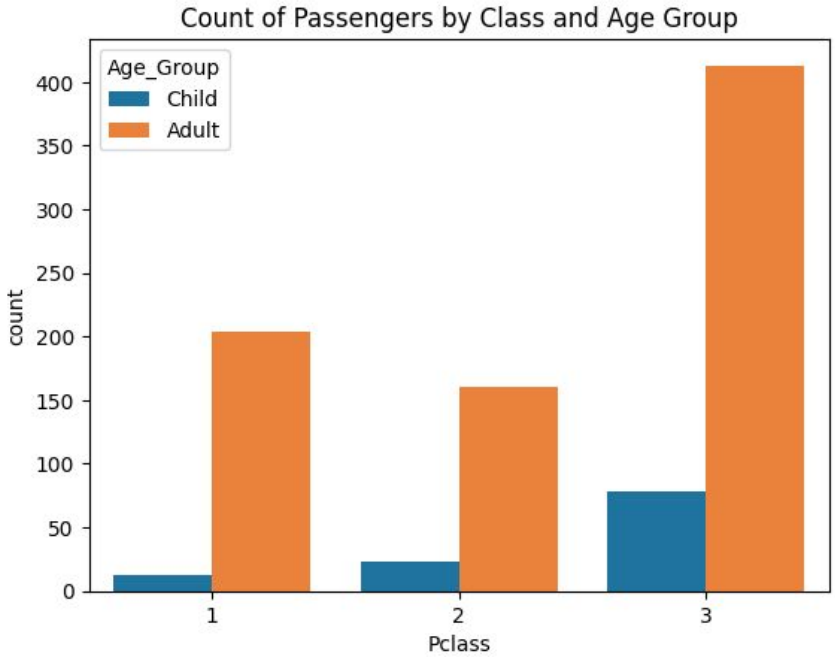
```
[15] # Barplot of survival count based on embarked location
     sns.barplot(x='Embarked', y='Survived', data=titanic_data, estimator=sum)
     plt.title('Survival Count by Embarked Location')
     plt.show()
```



## Countplot with Conditional Filtering

```
[16] # Countplot of survival status based on passenger class and age group (child or adult)
     def age_group(age):
         return 'Child' if age < 18 else 'Adult'

     titanic_data['Age_Group'] = titanic_data['Age'].apply(age_group)
     sns.countplot(x='Pclass', hue='Age_Group', data=titanic_data, hue_order=['Child', 'Adult'])
     plt.title('Count of Passengers by Class and Age Group')
     plt.show()
```



## Barplot with Groupby and Aggregate

```
[17] # Barplot showing the average fare paid by passengers in different 'Embarked' locations
     sns.barplot(x='Embarked', y='Fare', data=titanic_data.groupby('Embarked')['Fare'].mean().reset_index())
     plt.title('Average Fare by Embarked Location')
     plt.show()
```