Lecture 9

Learning outcomes:

- *Data Analysis and Manipulation:*
  - *Descriptive statistics and basic data analysis*

Descriptive Statistics

- Descriptive statistics refers to a branch of statistics that involves summarizing, organizing, and presenting data meaningfully and concisely. It focuses on describing and analyzing a dataset's main features and characteristics without making any generalizations or inferences to a larger population.
- The primary goal of descriptive statistics is to provide a clear and concise summary of the data, enabling researchers or analysts to gain insights and understand patterns, trends, and distributions within the dataset. This summary typically includes measures such as central tendency (e.g., mean, median, mode), dispersion (e.g., range, variance, standard deviation), and shape of the distribution (e.g., skewness, kurtosis).
- Descriptive statistics also involves a graphical representation of data through charts, graphs, and tables, which can further aid in visualizing and interpreting the information. Common graphical techniques include histograms, bar charts, pie charts, scatter plots, and box plots.

Types of Descriptive Statistics

- Measures of Central Tendency
- Measure of Variability
- Measures of Frequency Distribution

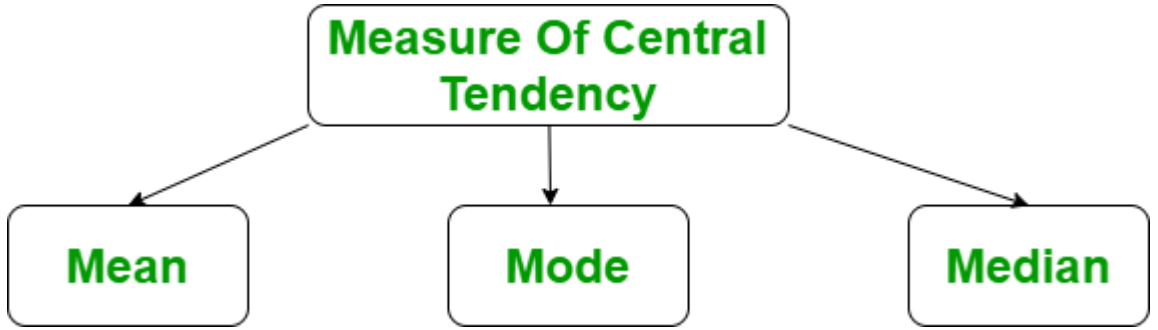
Measures of Central Tendency

- It represents the whole set of data by a single value. It gives us the location of the central points. There are three main measures of central tendency:

➢Mean

➢Mode

➢Median



Mean

- It is the sum of observations divided by the total number of observations. It is also defined as average which is the sum divided by count.

```
import numpy as np

# Sample Data

arr = [5, 6, 11]

# Mean

mean = np.mean(arr)

print("Mean = ", mean)
```

✓**Output**:

```
Mean =  7.333333333333333
```

Mode

- It is the value that has the highest frequency in the given data set. The data set may have no mode if the frequency of all data points is the same. Also, we can have more than one mode if we encounter two or more data points having the same frequency.

```
from scipy import stats
# sample Data
arr = [1, 2, 2, 3]


# Mode
mode = stats.mode(arr)
print("Mode = ", mode)
```

✓**Output**:

```
Mode =  ModeResult(mode=array([2]), count=array([2]))
```

Median

- It is the middle value of the data set. It splits the data into two halves. If the number of elements in the data set is odd then the center element is the median and if it is even then the median would be the average of two central elements.

```
import numpy as np
# sample Data
arr = [1, 2, 3, 4]

# Median
median = np.median(arr)
print("Median = ", median)
```

✓**Output**:

```
Median =  2.5
```

Measure of Variability

- Measures of variability are also termed measures of dispersion as it helps to gain insights about the dispersion or the spread of the observations at hand. Some of the measures which are used to calculate the measures of dispersion in the observations of the variables are as follows:

- Range
- Variance
- Standard deviation

Range

- The range describes the difference between the largest and smallest data point in our data set. The bigger the range, the more the spread of data and vice versa.

Range = Largest data value – smallest data value

```
import numpy as np
# Sample Data
arr = [1, 2, 3, 4, 5]

# Finding Max
Maximum = max(arr)
# Finding Min
Minimum = min(arr)

# Difference Of Max and Min
Range = Maximum-Minimum
print("Maximum = {}, Minimum = {} and Range = {}".format(Maximum, Minimum, Range))
```

✓ **Output**:

Maximum = 5, Minimum = 1 and Range = 4

Variance

- It is defined as an average squared deviation from the mean. It is calculated by finding the difference between every data point and the average which is also known as the mean, squaring them, adding all of them, and then dividing by the number of data points present in our data set.

```
import statistics
# sample data
arr = [1, 2, 3, 4, 5]
# variance
print("Var = ", (statistics.variance(arr)))
```

✓**Output**:

```
Var =  2.5
```

Standard Deviation

- It is defined as the square root of the variance. It is calculated by finding the Mean, then subtracting each number from the Mean which is also known as the average, and squaring the result. Adding all the values and then dividing by the no of terms followed by the square root.

```
import statistics
# sample data
arr = [1, 2, 3, 4, 5]

# Standard Deviation
print("Std = ", (statistics.stdev(arr)))
```

✓**Output**:

Std = 1.5811388300841898

Measures of Frequency Distribution

- Measures of frequency distribution help us gain valuable insights into the distribution and the characteristics of the dataset.

- A probability Distribution represents the predicted outcomes of various values for a given data. Probability distributions occur in a variety of forms and sizes, each with its own set of characteristics such as mean, median, mode, skewness, standard deviation, kurtosis, etc.

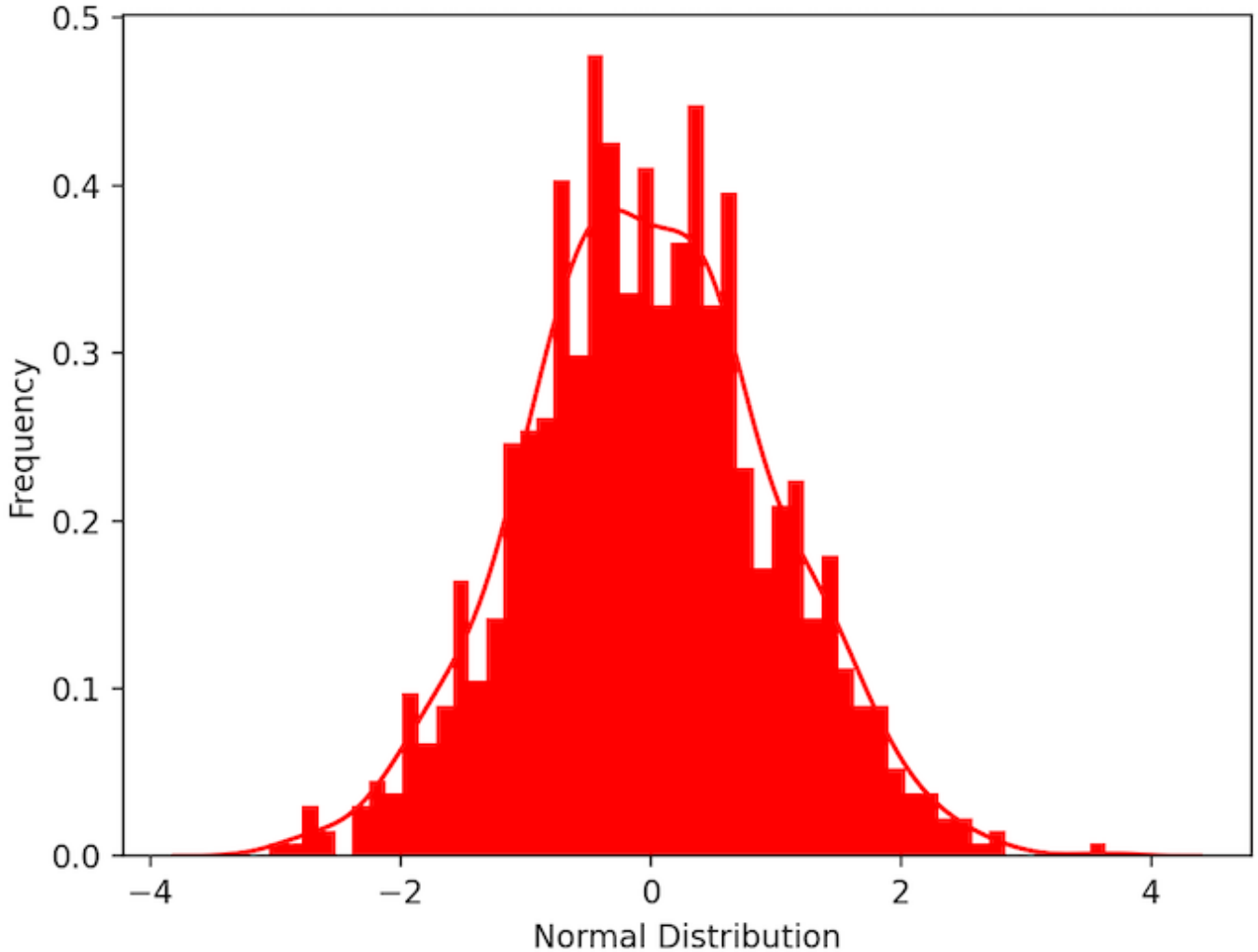
Normal Distribution

- The normal distribution is a symmetric probability distribution centered on the mean, indicating that data around the mean occur more frequently than data far from it. the normal distribution is also called Gaussian distribution.
- The normal distribution curve resembles a bell curve. In the below example we create normally distributed data using the function stats.norm() which generates continuous random data. the parameter scale refers to standard deviation and loc refers to mean. plt.distplot() is used to visualize the data. KDE refers to kernel density estimate, other parameters are for the customization of the plot. A bell-shaped curve can be seen as we visualize the plot.

```
# import packages
import scipy.stats as stats
import seaborn as sns
import matplotlib.pyplot as plt

# generate data
data =stats.norm(scale=1, loc=0).rvs(1000)

# plotting a histogram
ax = sns.distplot(data, bins=50, kde=True, color='red', hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Normal Distribution', ylabel='Frequency')
plt.show()
```


0.5
0.4
0.3
0.2
0.1
0.0
Frequency
−4
−2
0
2
4
Normal Distribution

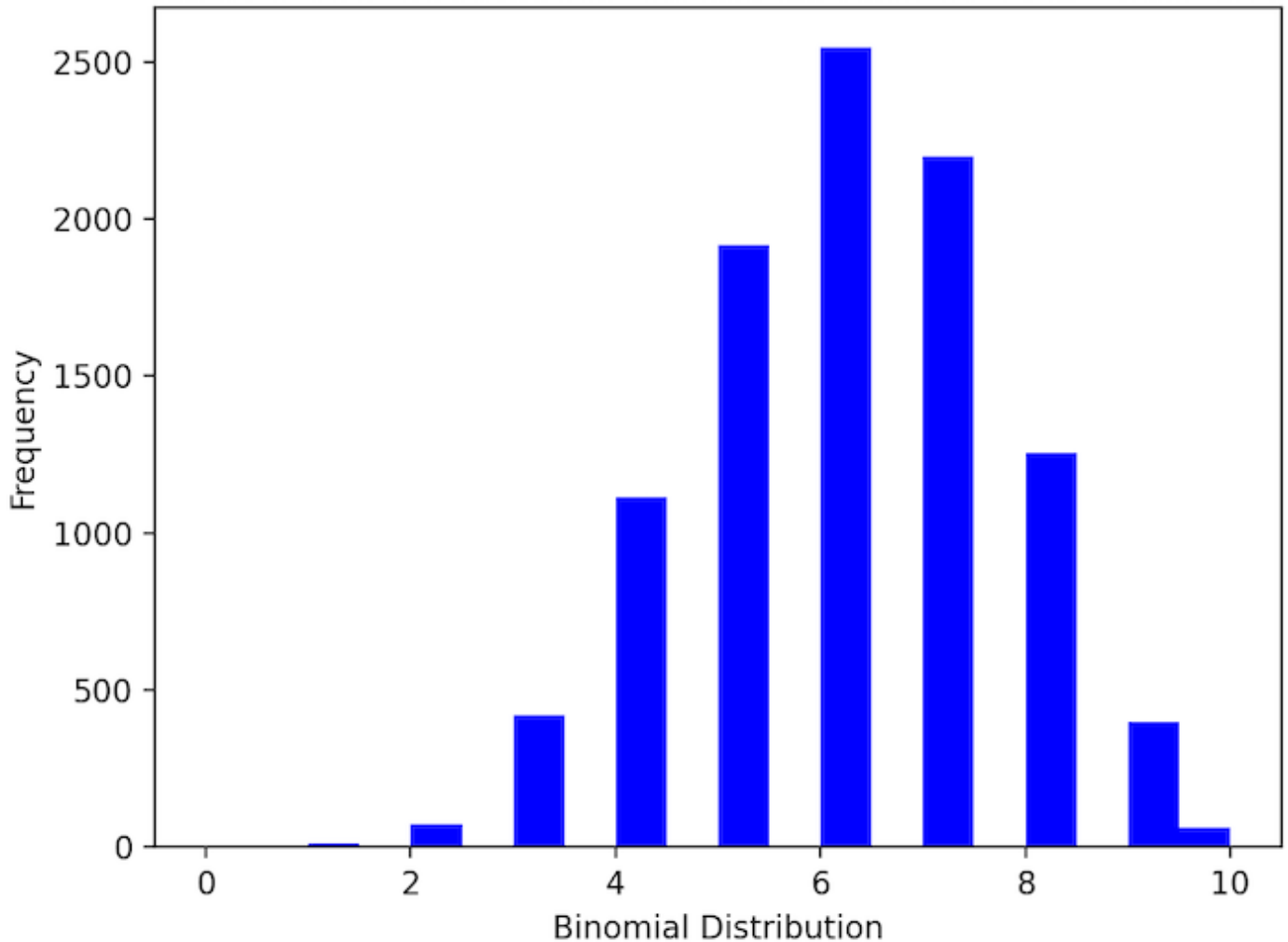
Binomial Distribution

- Under a given set of factors or assumptions, the binomial distribution expresses the likelihood that a variable will take one of two outcomes or independent values.

- Ex: if an experiment is successful or a failure. if the answer for a question is “yes” or “no” etc… . np.random.binomial() is used to generate binomial data. n refers to a number of trails and prefers the probability of each trail.

```
# import packages
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
# generate data
# n== number of trials,p== probability of each trial
n, p = 10, .6
data = np.random.binomial(n, p, 10000)

# plotting a histogram
ax = sns.distplot(data, bins=20, kde=False, color='blue', hist_kws={"linewidth": 15, 'alpha': 1})
ax.set(xlabel='Binomial Distribution', ylabel='Frequency')
plt.show()
```

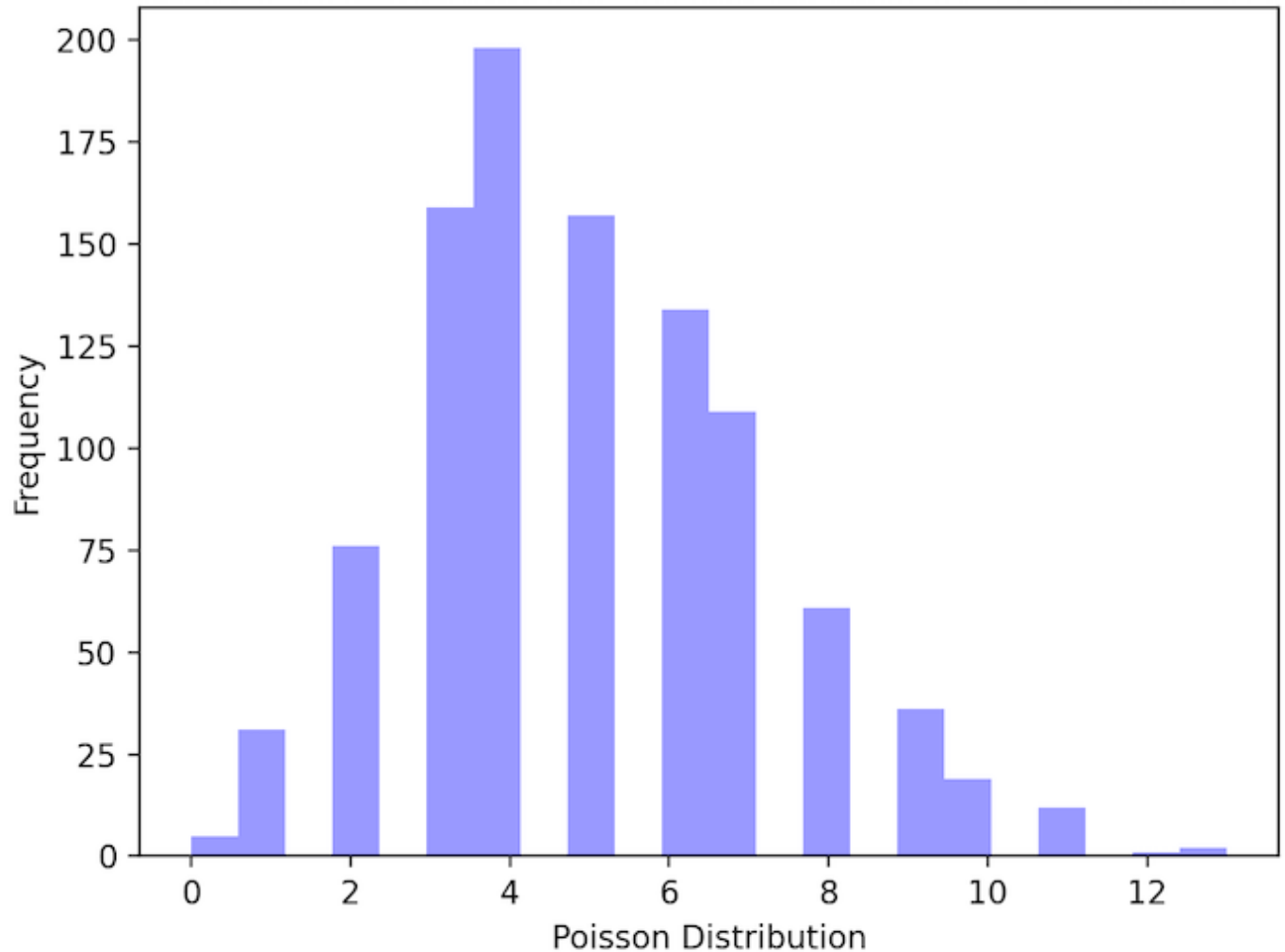

2500
2000
1500
1000
500
0
Frequency
0
2
4
6
8
10
Binomial Distribution

Poisson Distribution:

- A Poisson distribution is a kind of probability distribution used in statistics to illustrate how many times an event is expected to happen over a certain amount of time. It’s also called count distribution. np.random.poisson function() is used to create data for poisson distribution. lam refers to The number of occurrences that are expected to occur in a given time frame.
- In this example, we can take the condition as “if a student studies for 5 hours a day, the probability that he’ll study 6 hours a day is?.

```
# import packages
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
# generate poisson data
poisson_data = np.random.poisson(lam=5, size=1000)

# plotting a histogram
ax = sns.distplot(poisson_data, kde=False, color='blue')
ax.set(xlabel='Poisson Distribution', ylabel='Frequency')
plt.show()
```


200
175
150
125
100
75
50
25
0
Frequency
0
2
4
6
8
10
12
Poisson Distribution

Data Visualisation: Univariate, Bivariate and Multivariate Analysis

- Data Visualisation is a graphical representation of information and data.

- By using different visual elements such as charts, graphs, and maps data visualization tools provide us with an accessible way to find and understand hidden trends and patterns in data.

Univariate Analysis

- Univariate Analysis is a type of data visualization where we visualize only a single variable at a time. Univariate Analysis helps us to analyze the distribution of the variable present in the data so that we can perform further analysis.

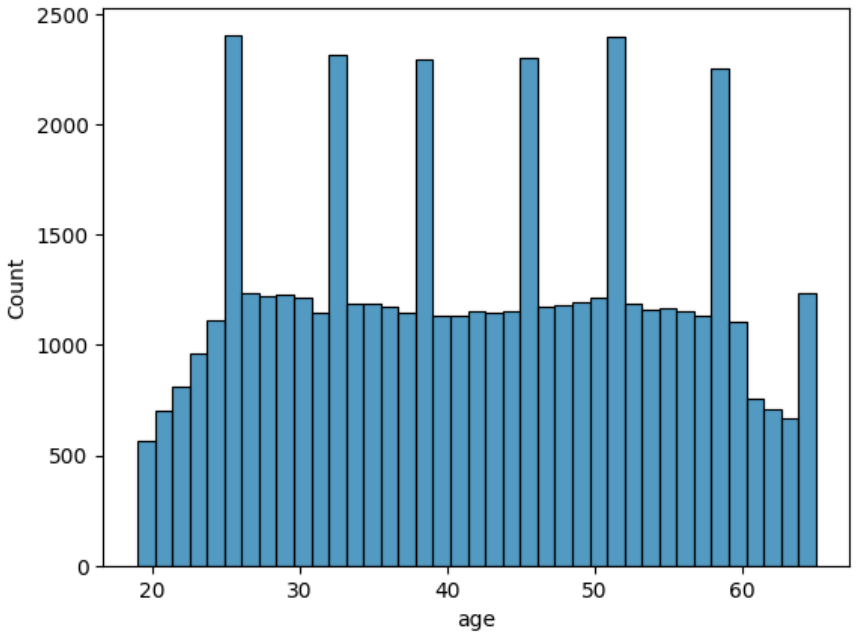
```
import pandas as pd
import seaborn as sns
data = pd.read_csv('Employee_dataset.csv')
print(data.head())
```

Histogram

- Here we'll be performing univariate analysis on Numerical variables using the histogram function.

sns.histplot(data['age'])

```
In [5]: sns.histplot(data['age'])
Out[5]: <AxesSubplot: xlabel='age', ylabel='Count'>
```



Bar Chart

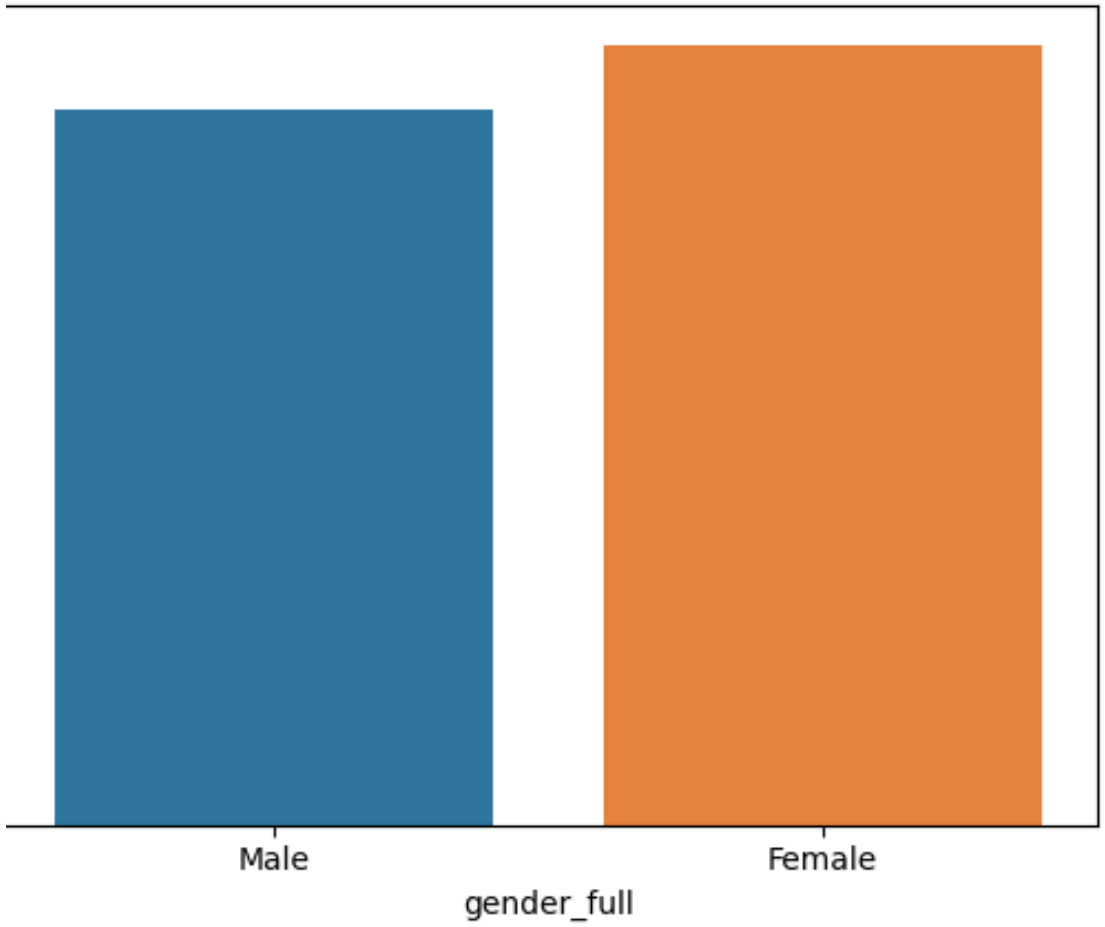
- Univariate analysis of categorical data. We'll be using the count plot function from the seaborn library

- The Bars in the chart are representing the count of each category present in the 'gender column.

sns.countplot(data['gender_full'])

```
lot(x=data['gender_full'])
```

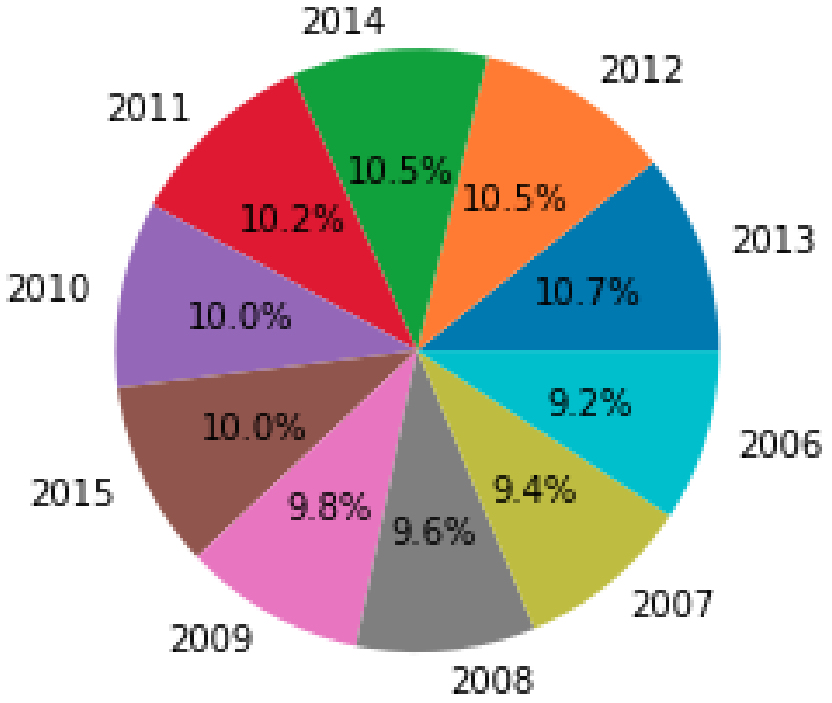
```
ot: xlabel='gender_full', ylabel='count'>
```



Pie Chart

- A piechart helps us to visualize the percentage of the data belonging to each category.

```
import matplotlib.pyplot as plt
x = data['STATUS_YEAR'].value_counts()
plt.pie(x.values,
        labels=x.index,
        autopct='%1.1f%%')
plt.show()
```



Bivariate analysis

- Bivariate analysis is the simultaneous analysis of two variables. It explores the concept of the relationship between two variable whether there exists an association and the strength of this association or whether there are differences between two variables and the significance of these differences.

- The main three types we will see here are:

➢Categorical v/s Numerical

➢Numerical V/s Numerical

➢Categorical V/s Categorical data

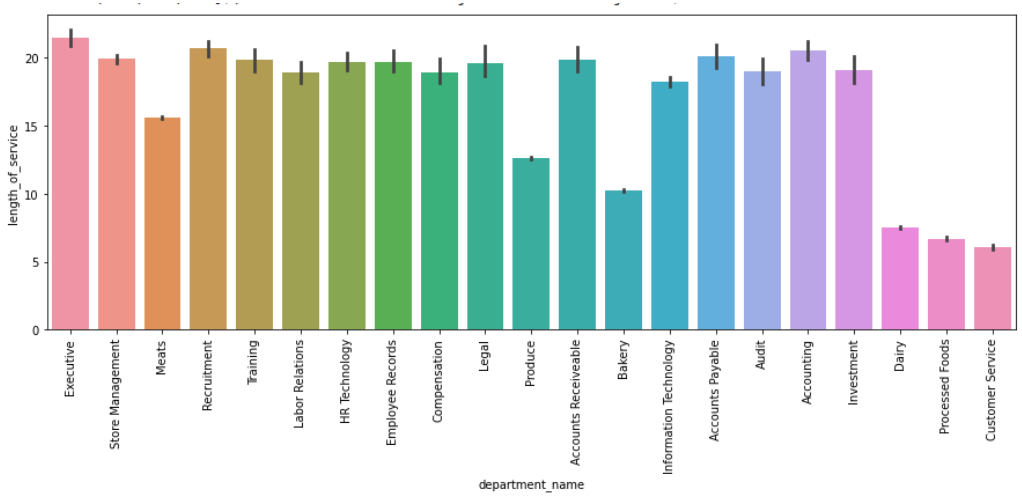
Categorical v/s Numerical

```
import matplotlib.pyplot as plt

plt.figure(figsize=(15, 5))

sns.barplot(x=data['department_name'],
y=data['length_of_service'])

plt.xticks(rotation='vertical')
```



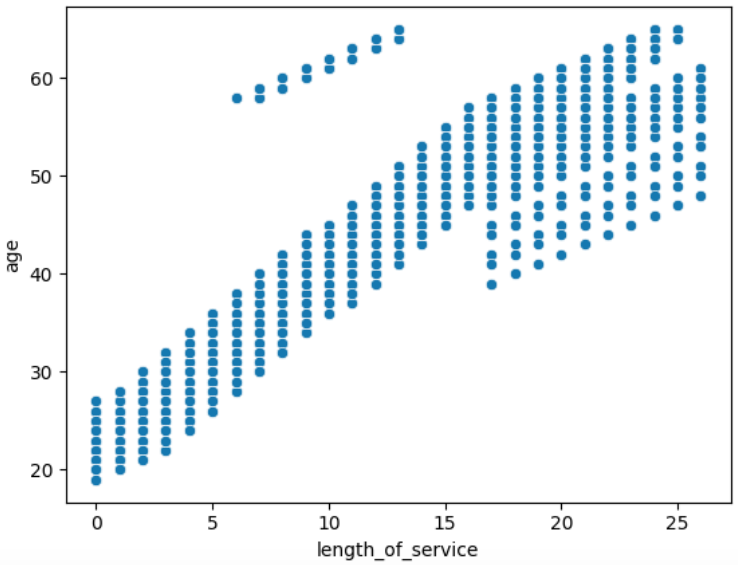
- ✓ Here the Black horizontal line is indicating huge differences in the length of service among different departments.

Numerical v/s Numerical

```
sns.scatterplot(x=data['length_of_service'],
                y=data['age'])
```

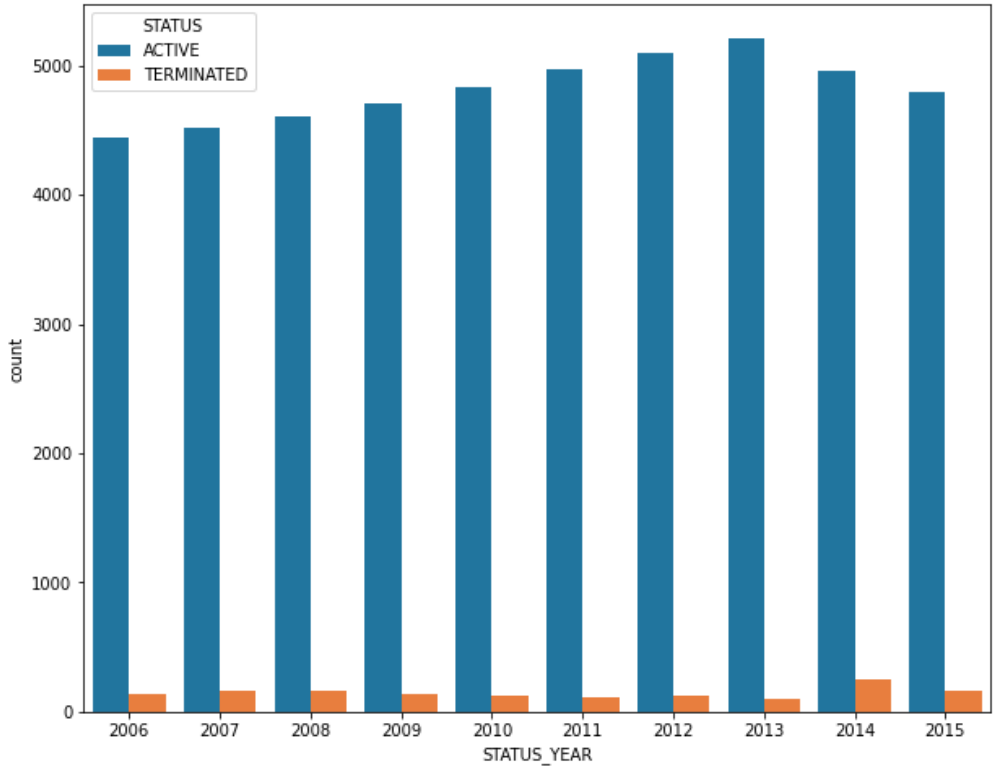
- ✓ It displays the age and length of service of employees in the organization as we can see that younger employees have less experience in terms of their length of service.

```
In [42]: sns.scatterplot(x=data['length_of_service'],y=data['age'])
Out[42]: <AxesSubplot: xlabel='length_of_service', ylabel='age'>
```



Categorical v/s Categorical

sns.countplot(x='STATUS_YEAR', hue='STATUS', data=data)



Multivariate Analysis

- It is an extension of bivariate analysis which means it involves multiple variables at the same time to find correlation between them.

- Multivariate Analysis is a set of statistical model that examine patterns in multidimensional data by considering at once, several data variable.