Data Visualization

February 20, 2025

1 Data Distribution and their Visualizations using Matplotlib and Seaborn

Data Visualization using Matplotlib and Seaborn

[ ]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import statistics
%matplotlib inline
```

[ ]:
```
df=pd.read_csv('/content/drive/MyDrive/Colab Notebooks/ML-DSE4/Project_College/
↪salary_data.csv')
```

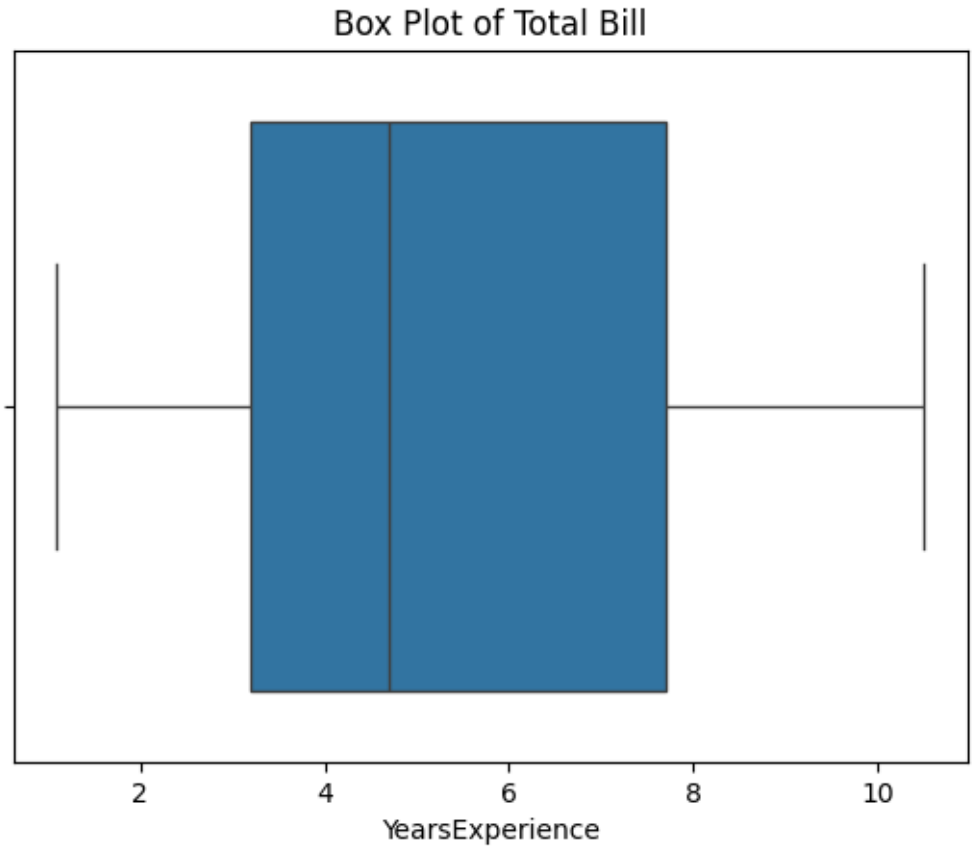
[ ]:
```
print(df)
```

```
    YearsExperience    Salary
0               1.1   39343.0
1               1.3   46205.0
2               1.5   37731.0
3               2.0   43525.0
4               2.2   39891.0
5               2.9   56642.0
6               3.0   60150.0
7               3.2   54445.0
8               3.2   64445.0
9               3.7   57189.0
10              3.9   63218.0
11              4.0   55794.0
12              4.0   56957.0
13              4.1   57081.0
14              4.5   61111.0
15              4.9   67938.0
16              5.1   66029.0
17              5.3   83088.0
18              5.9   81363.0
19              6.0   93940.0
```

20	6.8	91738.0
21	7.1	98273.0
22	7.9	101302.0
23	8.2	113812.0
24	8.7	109431.0
25	9.0	105582.0
26	9.5	116969.0
27	9.6	112635.0
28	10.3	122391.0
29	10.5	121872.0

Box-Plot of **Years of Experience** attribute of **Salary Dataset**

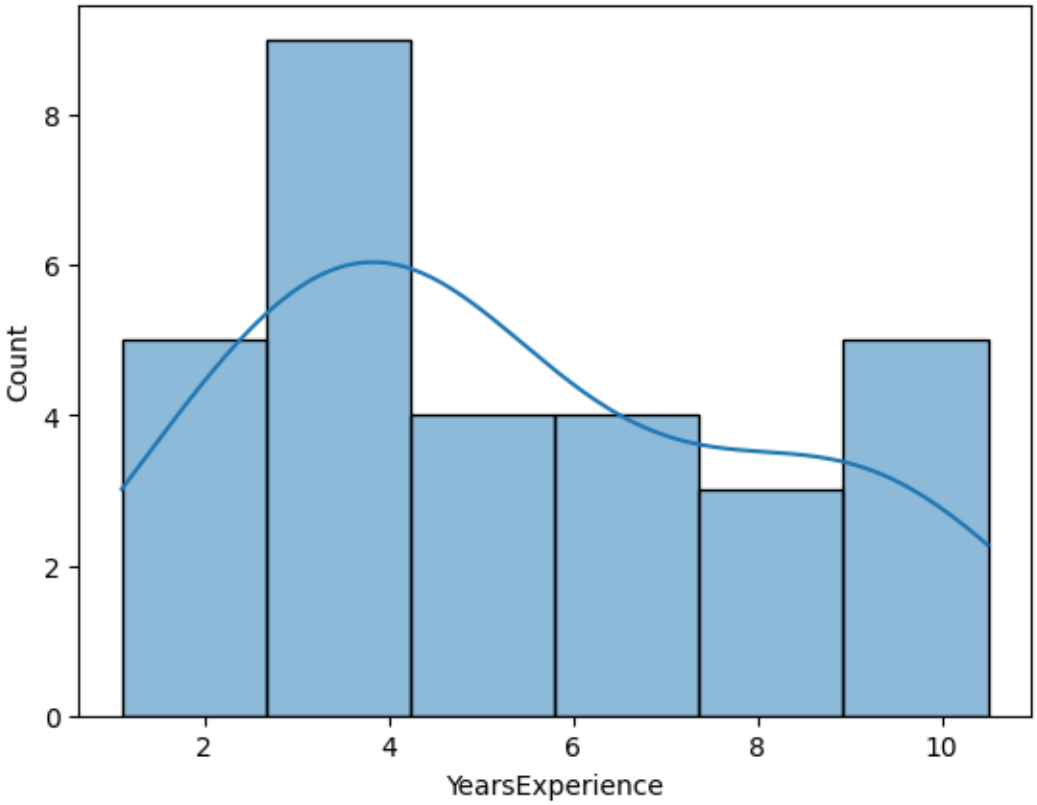
[ ]:
```
sns.boxplot(x=df['YearsExperience'])
plt.title('Box Plot of Total Bill')
plt.show()
```



Histogram Plot for **Years of Experience** and **Salary** attributes of Salary Dataset

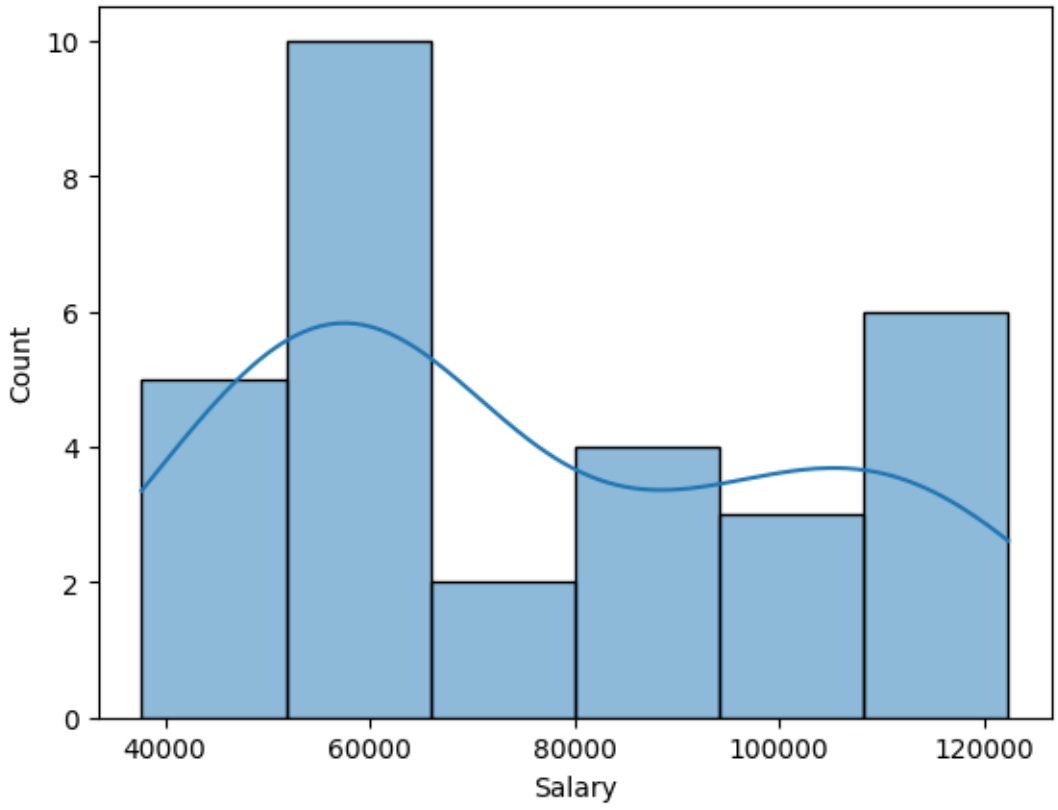
[ ]:
```
sns.histplot(df['YearsExperience'],kde=True)
```

[ ]: <Axes: xlabel='YearsExperience', ylabel='Count'>



[ ]:
```
sns.histplot(df['Salary'],kde=True)
```

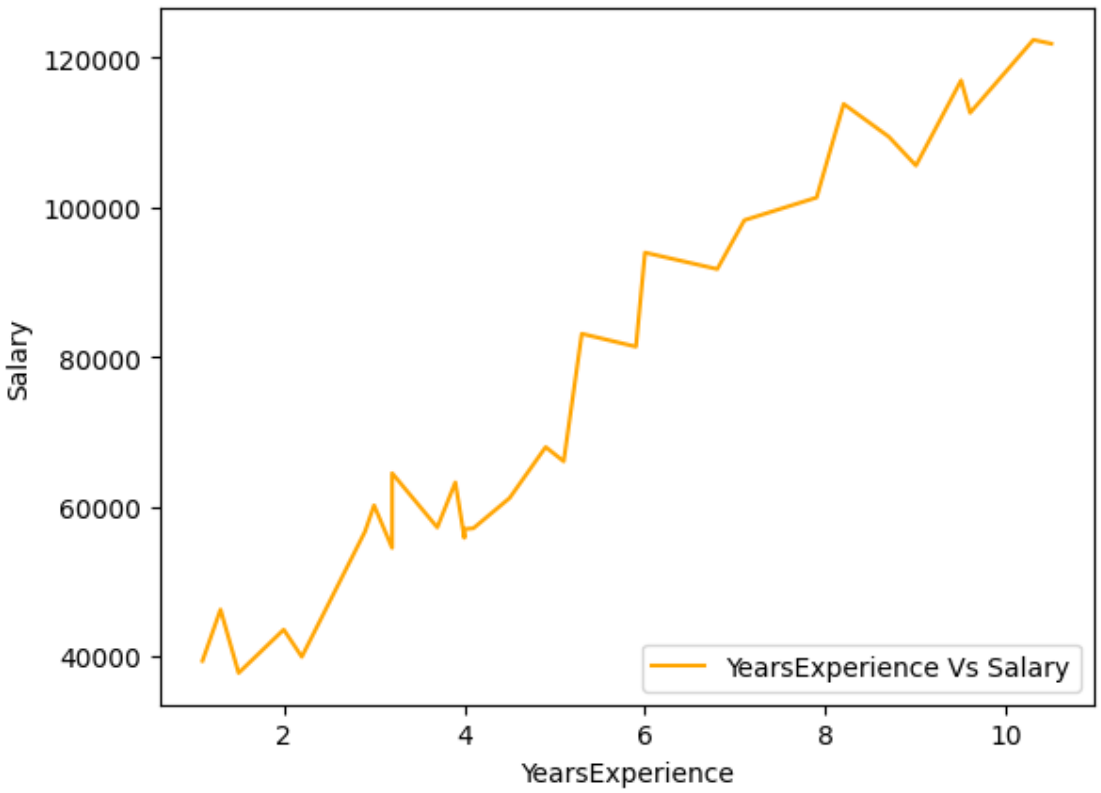
[ ]: <Axes: xlabel='Salary', ylabel='Count'>



Line Plot between YearsExperience and Salary

[ ]:
```
plt.xlabel('YearsExperience')
plt.ylabel('Salary')
plt.plot(df['YearsExperience'],df['Salary'],color='orange')
plt.legend(['YearsExperience Vs Salary'],loc="lower right")
```

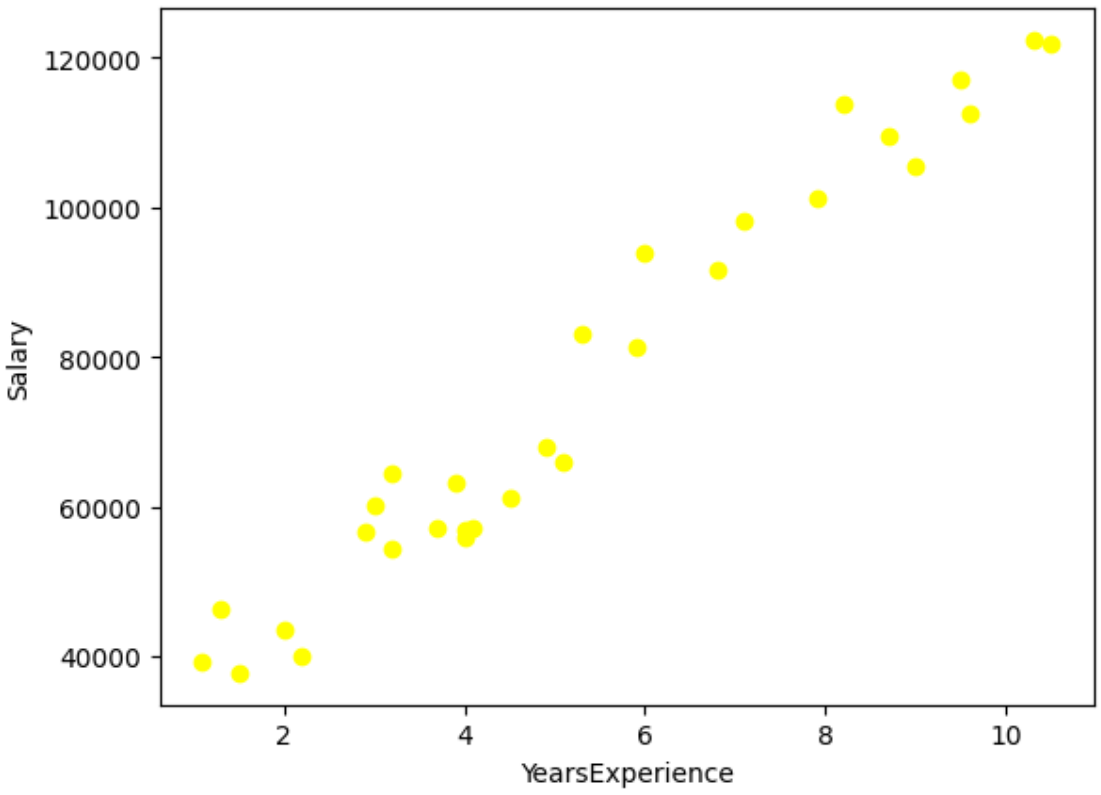
[ ]:
```
<matplotlib.legend.Legend at 0x7b345fec7010>
```



Scatter Plot between YearExperience and Salary

[ ]:
```
plt.scatter(df['YearsExperience'],df['Salary'],color='yellow')
plt.xlabel('YearsExperience')
plt.ylabel('Salary')
```

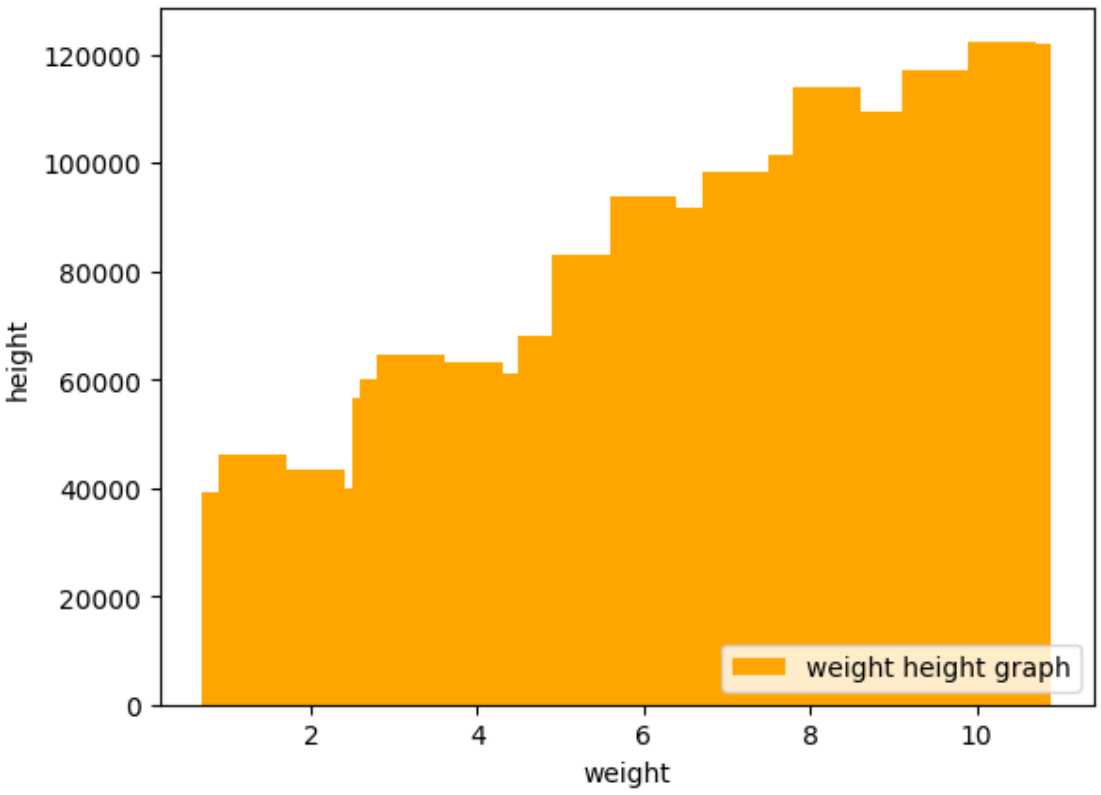
[ ]:
```
Text(0, 0.5, 'Salary')
```



Bar plot between YearExperience and Salary

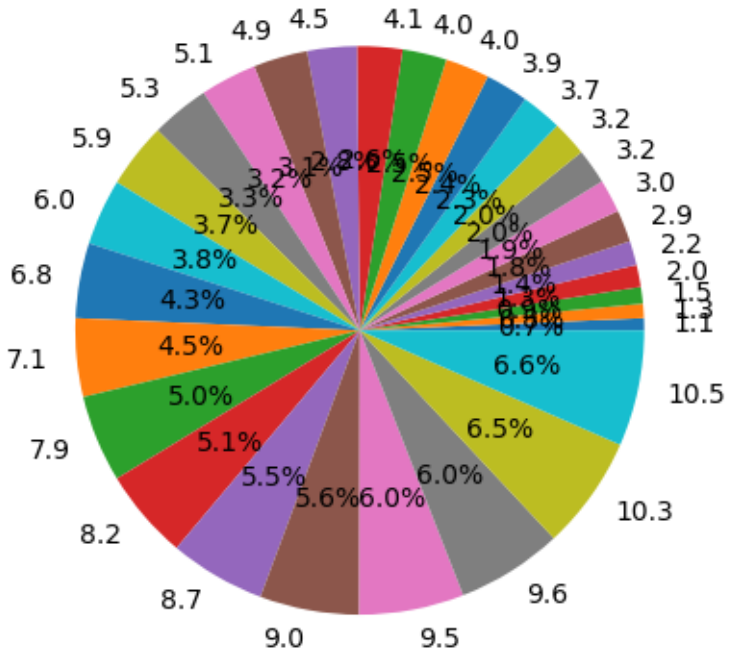
[ ]:
```
plt.bar(df['YearsExperience'],df['Salary'],color='orange')
plt.xlabel('weight')
plt.ylabel('height')
plt.legend(['weight height graph'],loc="lower right")
```

[ ]:
```
<matplotlib.legend.Legend at 0x7b34607bb650>
```



Pie chart figure for YearsExperience

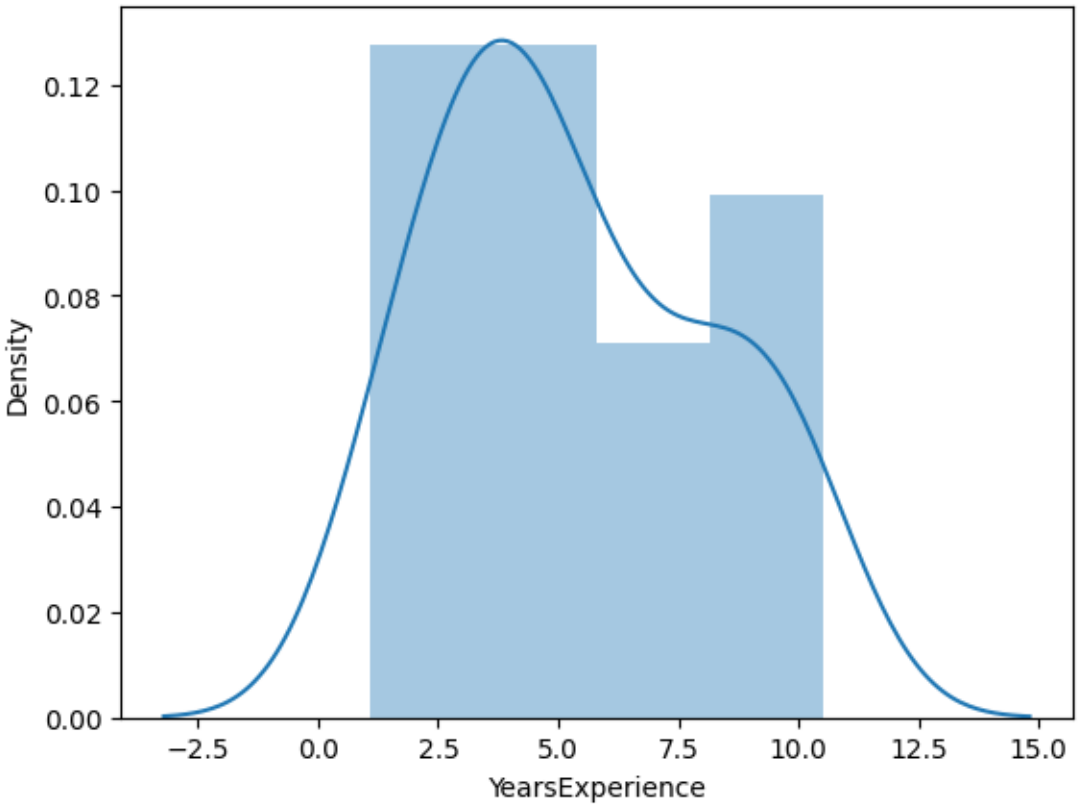
[ ]:
```
plt.pie(df['YearsExperience'],labels=df['YearsExperience'],autopct='%1.1f%%')
plt.show()
```



Plots to show nature of distribution for YearsExperience

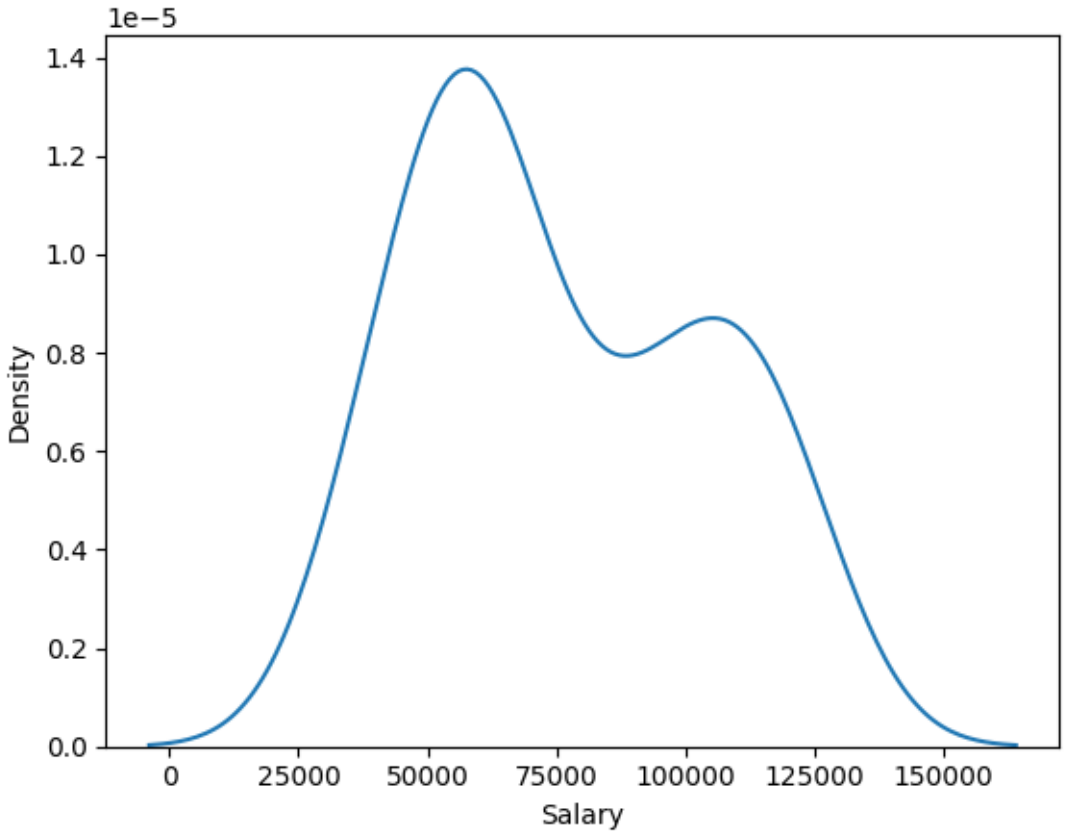
[ ]:
```
sns.distplot(df['YearsExperience'])
plt.show()
```

```
<ipython-input-17-661e514baad8>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['YearsExperience'])
```



[ ]:
```
sns.distplot(df['Salary'],hist=False)
plt.show()
```

```
<ipython-input-18-268d96b9db1c>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density
plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df['Salary'],hist=False)
```



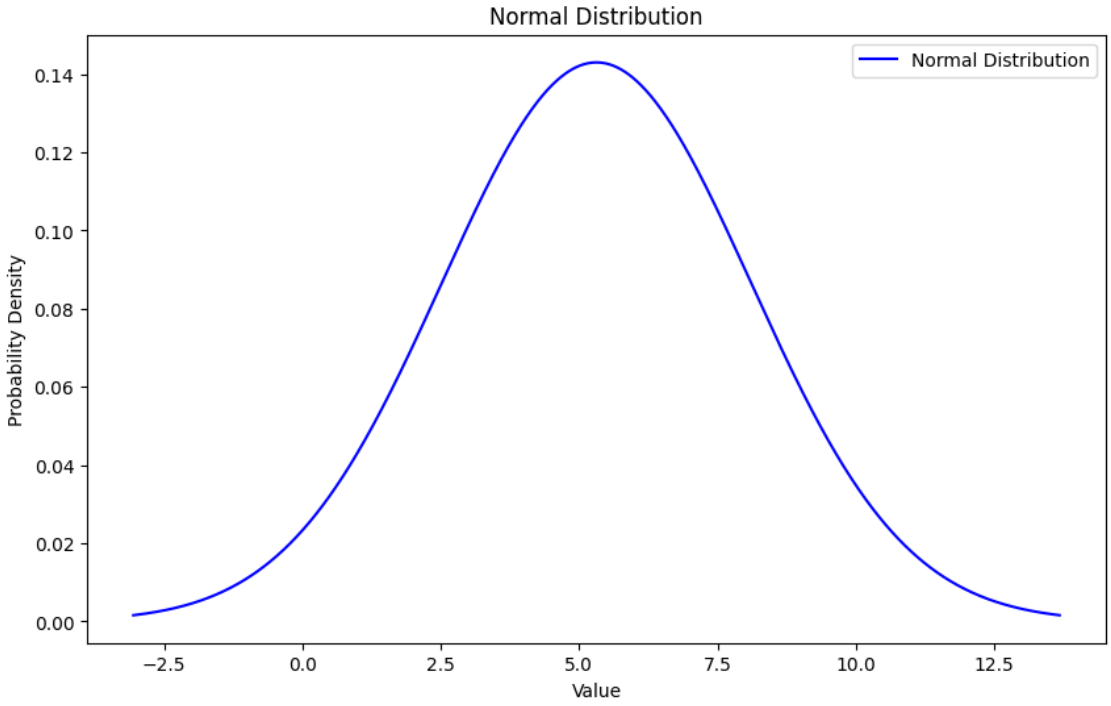
Plots for Normal and Binomial Distributions and their fittings

[ ]:
```
from scipy.stats import norm, binom
```

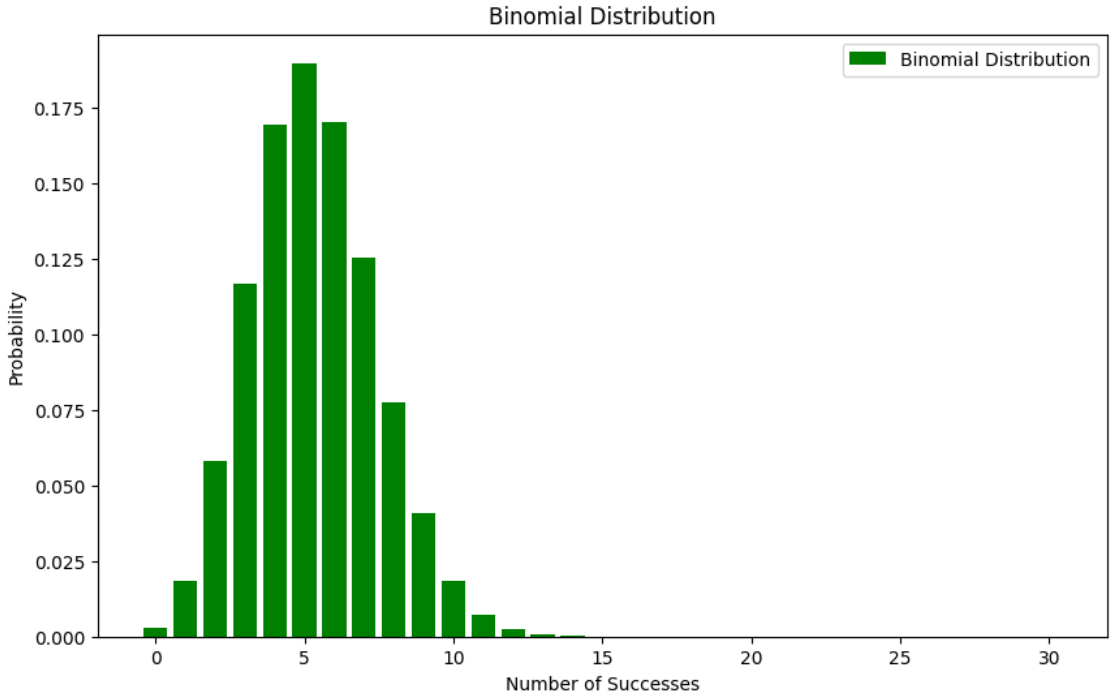
[ ]:
```
# Read the specific column
data = df['YearsExperience']
```

[ ]:
```
# Calculate mean and standard deviation
mean = np.mean(data)
std_dev = np.std(data)

# Create a range of values for the normal distribution
x = np.linspace(mean - 3*std_dev, mean + 3*std_dev, 1000)

# Calculate the normal distribution values
normal_dist = norm.pdf(x, mean, std_dev)

# Plot the normal distribution
plt.figure(figsize=(10, 6))
plt.plot(x, normal_dist, label='Normal Distribution', color='blue')
```

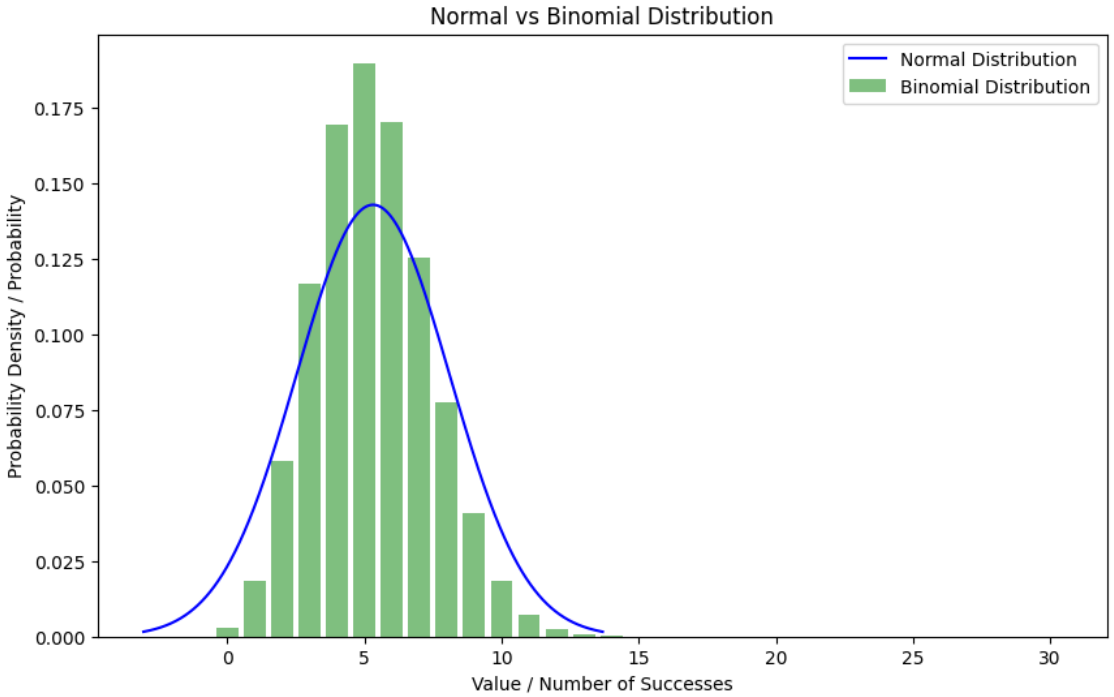
```
plt.title('Normal Distribution')
plt.xlabel('Value')
plt.ylabel('Probability Density')
plt.legend()
plt.show()
```



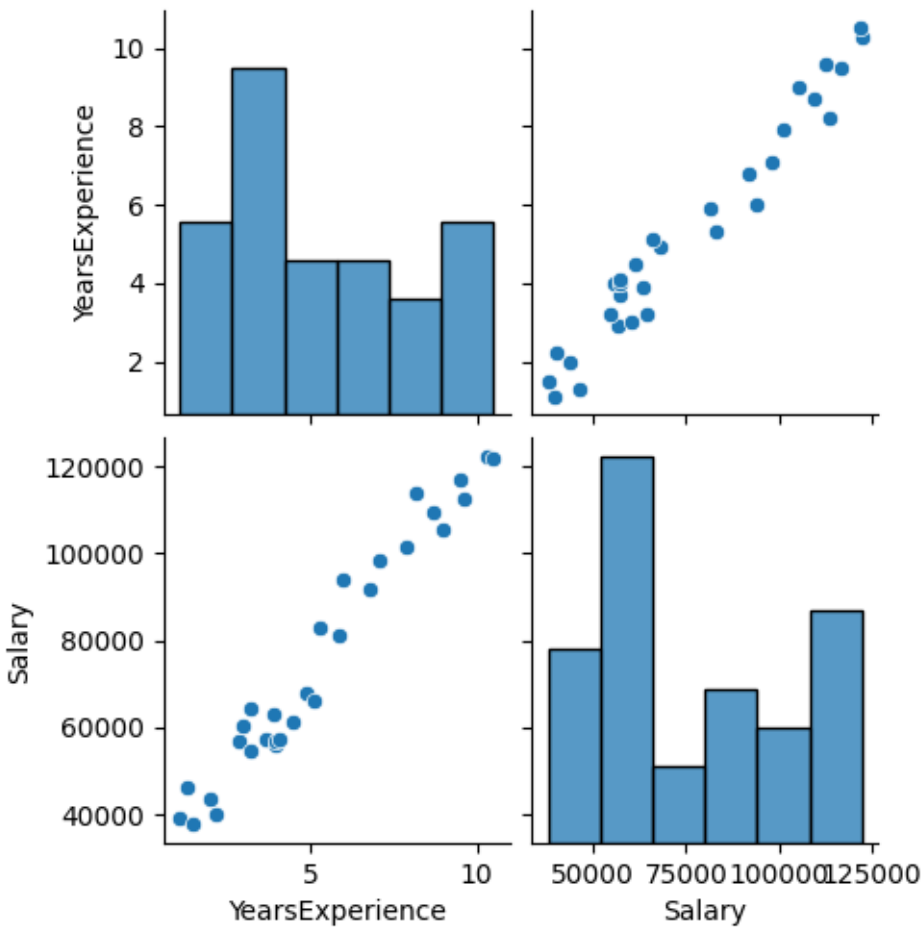
[ ]:
```
# Estimate parameters for the binomial distribution
n = len(data)  # Number of trials
p = np.mean(data) / n  # Probability of success

# Create a range of values for the binomial distribution
k = np.arange(0, n+1)

# Calculate the binomial distribution values
binomial_dist = binom.pmf(k, n, p)

# Plot the binomial distribution
plt.figure(figsize=(10, 6))
plt.bar(k, binomial_dist, label='Binomial Distribution', color='green')
plt.title('Binomial Distribution')
plt.xlabel('Number of Successes')
plt.ylabel('Probability')
plt.legend()
plt.show()
```



[ ]:
```
plt.figure(figsize=(10, 6))
plt.plot(x, normal_dist, label='Normal Distribution', color='blue')
plt.bar(k, binomial_dist, label='Binomial Distribution', color='green', alpha=0.
↪5)
plt.title('Normal vs Binomial Distribution')
plt.xlabel('Value / Number of Successes')
plt.ylabel('Probability Density / Probability')
plt.legend()
plt.show()
```



[ ]:
```
#Pair plot between YearsExperience Vs Salary
sns.pairplot(df)
plt.show()
```



[ ]:
```
data=df['YearsExperience']
mu, std = norm.fit(data)
```

[ ]:
```
n = 10  # Number of trials
p = 0.5  # Probability of success

# Generate binomial distribution
binomial_dist = binom.pmf(np.arange(0, n+1), n, p)
```

[ ]:
```
# Create a figure and axis
fig, ax = plt.subplots(figsize=(10, 6))

# Plot the histogram of the data
sns.histplot(data, kde=False, stat='density', bins=30, color='blue',␣
 ↪label='Data Histogram')

# Plot the fitted normal distribution
```

```
xmin, xmax = plt.xlim()
x = np.linspace(xmin, xmax, 100)
p = norm.pdf(x, mu, std)
ax.plot(x, p, 'k', linewidth=2, label='Normal Distribution')

# Plot the binomial distribution
ax.plot(np.arange(0, n+1), binomial_dist, 'r-', label='Binomial Distribution')

# Add labels and legend
ax.set_title('Normal and Binomial Distribution Fit')
ax.set_xlabel('Value')
ax.set_ylabel('Density')
ax.legend()

# Show the plot
plt.show()
```