

ML-Lab2 - Colab

colab.research.google.com/drive/1HuS73laNI4PWjT0sh6NLgdatkrEfupQg#scrollTo=wIMnT-QtFFPE

ML-Lab2

File Edit View Insert Runtime Tools Help All changes saved

Files

Marketing-File

sample\_data

README.md

anscombe.js...

california\_housi...

california\_housi...

mnist\_test.csv

mnist\_train\_s...

Employee\_Salarie...

+ Code + Text

[7]

1 # Basic summary of Employee Salaries data  
2 print("\nSummary of Employee Salaries Data:")  
3 print(employee\_data.describe())  
4

Summary of Employee Salaries Data:

	Base_Salary	Overtime_Pay	Longevity_Pay
count	10291.000000	10291.000000	10291.000000
mean	90312.165744	8081.288954	1533.882816
std	31240.842929	16491.833017	3209.041070
min	11147.240000	0.000000	0.000000
25%	70023.000000	0.000000	0.000000
50%	87328.000000	258.420000	0.000000
75%	108084.000000	9190.970000	1225.680000
max	292000.000000	227428.990000	20279.460000

[9]

1 # Check for missing values  
2 print("\nMissing values in Employee Salaries Data:")  
3 print(employee\_data.isnull().sum())  
4

Missing values in Employee Salaries Data:

Department	0
Department_Name	0
Division	0
Gender	0
Base_Salary	0
Overtime_Pay	0
Longevity_Pay	0
Grade	33

dtype: int64

[10]

1 # Basic summary of Employee Salaries data  
2 print("\nSummary of Employee Salaries Data:")  
3 print(employee\_data.describe())  
4

RAM Disk

Gemini

Disk

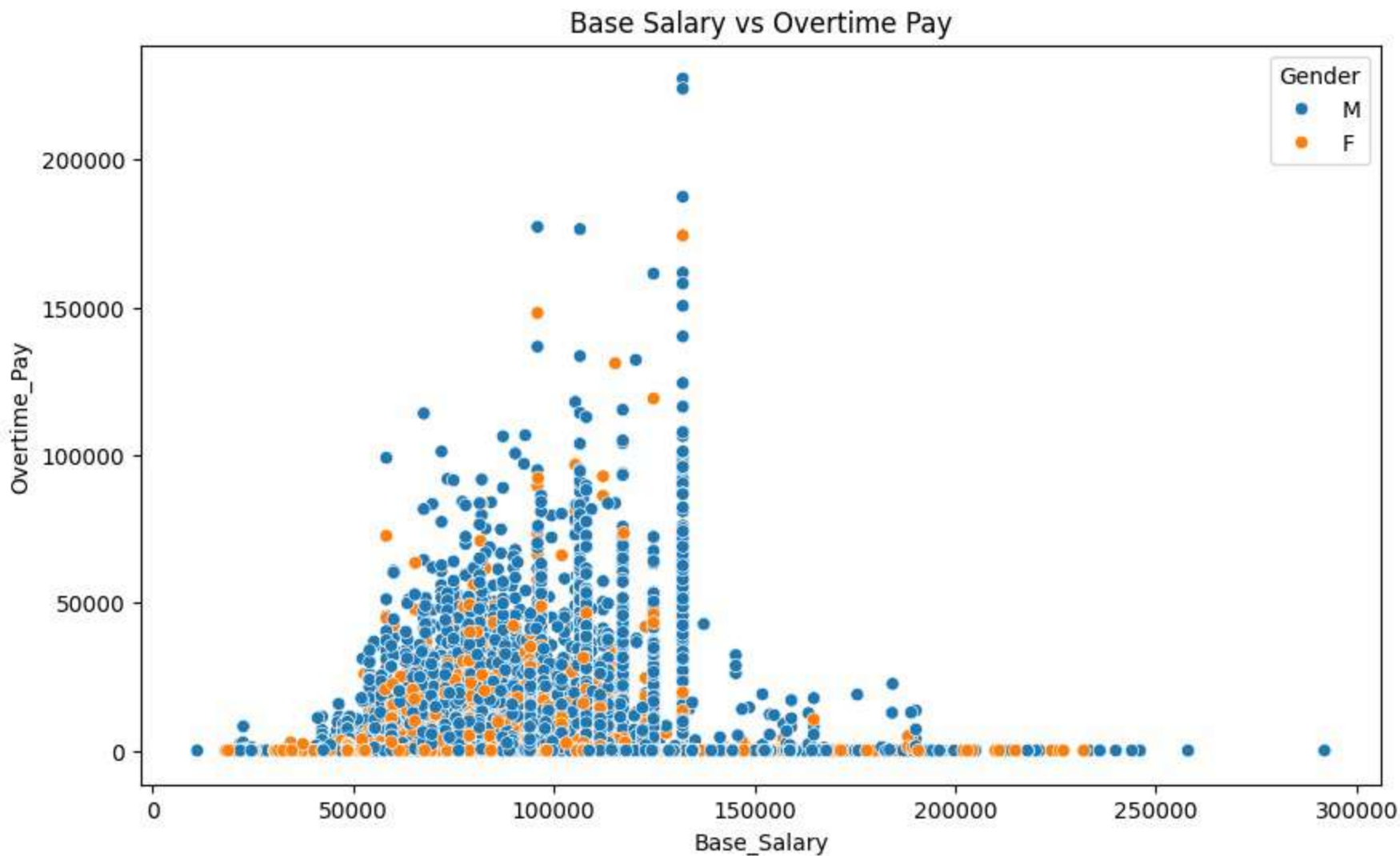
75.21 GB available

0s

completed at 10:05 AM

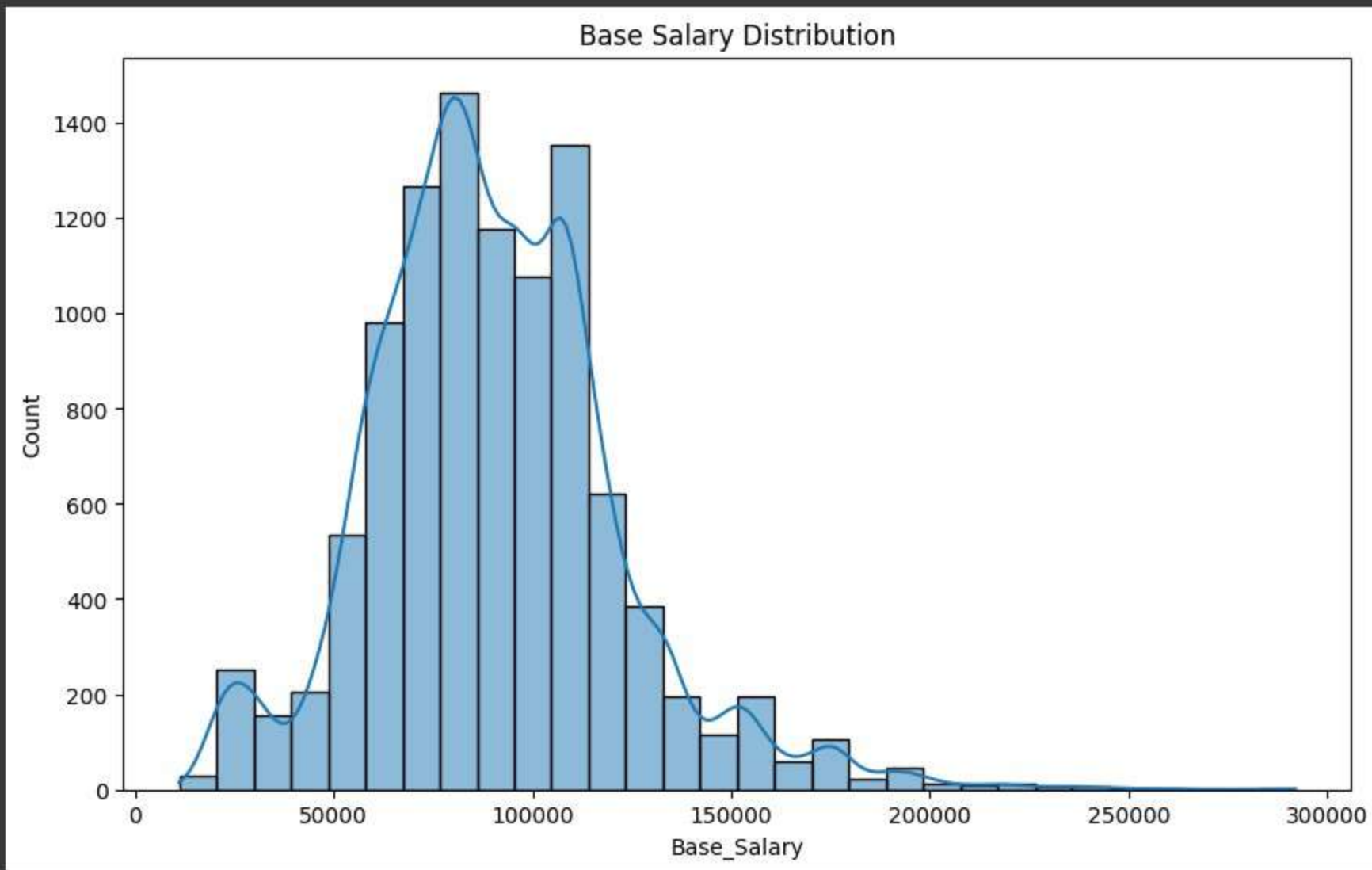


```
1 # Scatter plot to show Base Salary vs Overtime Pay
2 plt.figure(figsize=(10, 6))
3 sns.scatterplot(data=employee_data, x='Base_Salary', y='Overtime_Pay', hue='Gender')
4 plt.title("Base Salary vs Overtime Pay")
5 plt.show()
6
```





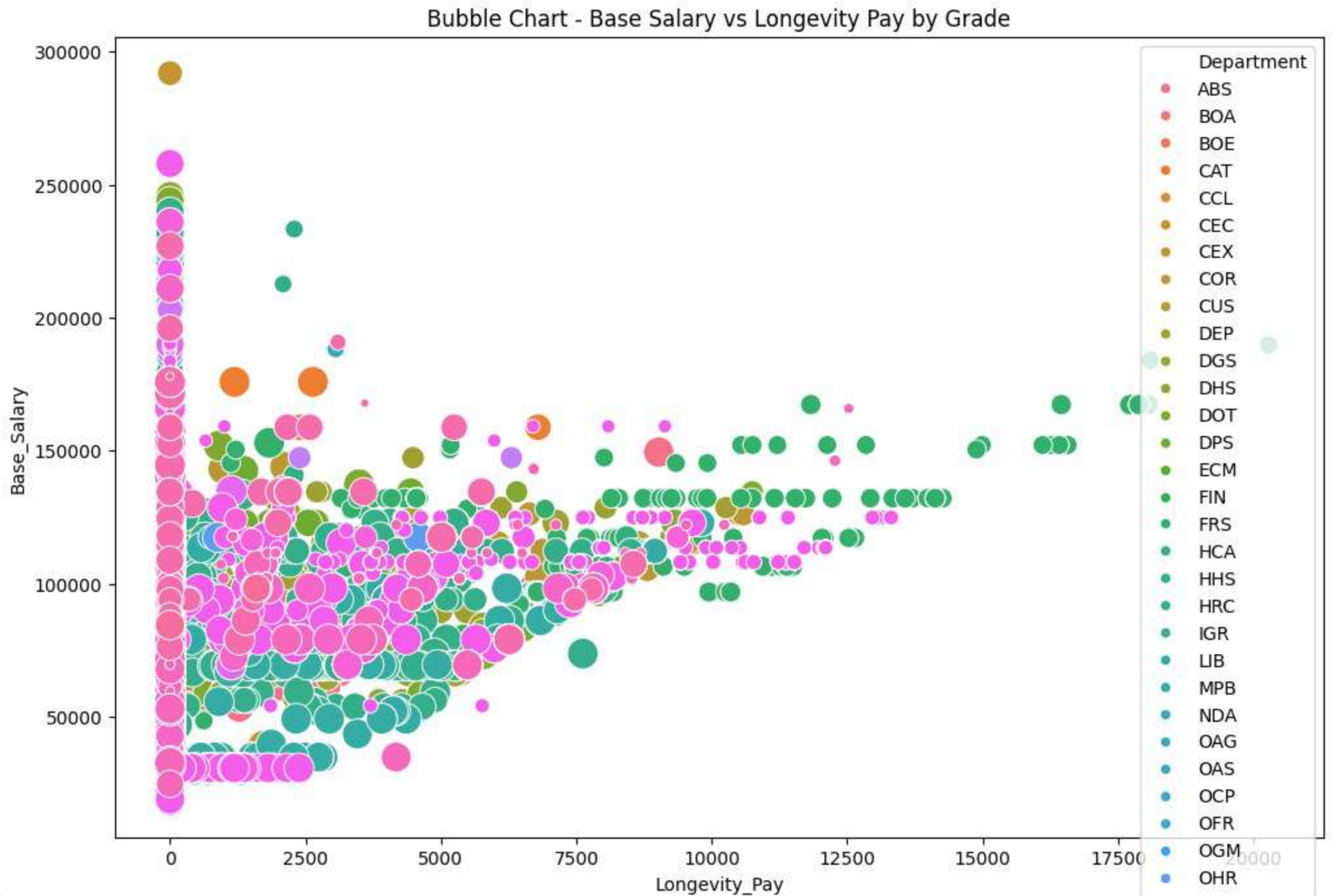
```
1 # Histogram to show Base Salary distribution
2 plt.figure(figsize=(10, 6))
3 sns.histplot(employee_data['Base_Salary'], kde=True, bins=30)
4 plt.title("Base Salary Distribution")
5 plt.show()
6
```



```

1 # Bubble chart for Base Salary vs Longevity Pay by Grade
2 plt.figure(figsize=(12, 8))
3 sns.scatterplot(data=employee_data, x='Longevity_Pay', y='Base_Salary', size='Grade', hue='Department', sizes=(20, 300))
4 plt.title("Bubble Chart - Base Salary vs Longevity Pay by Grade")
5 plt.show()
6

```





```

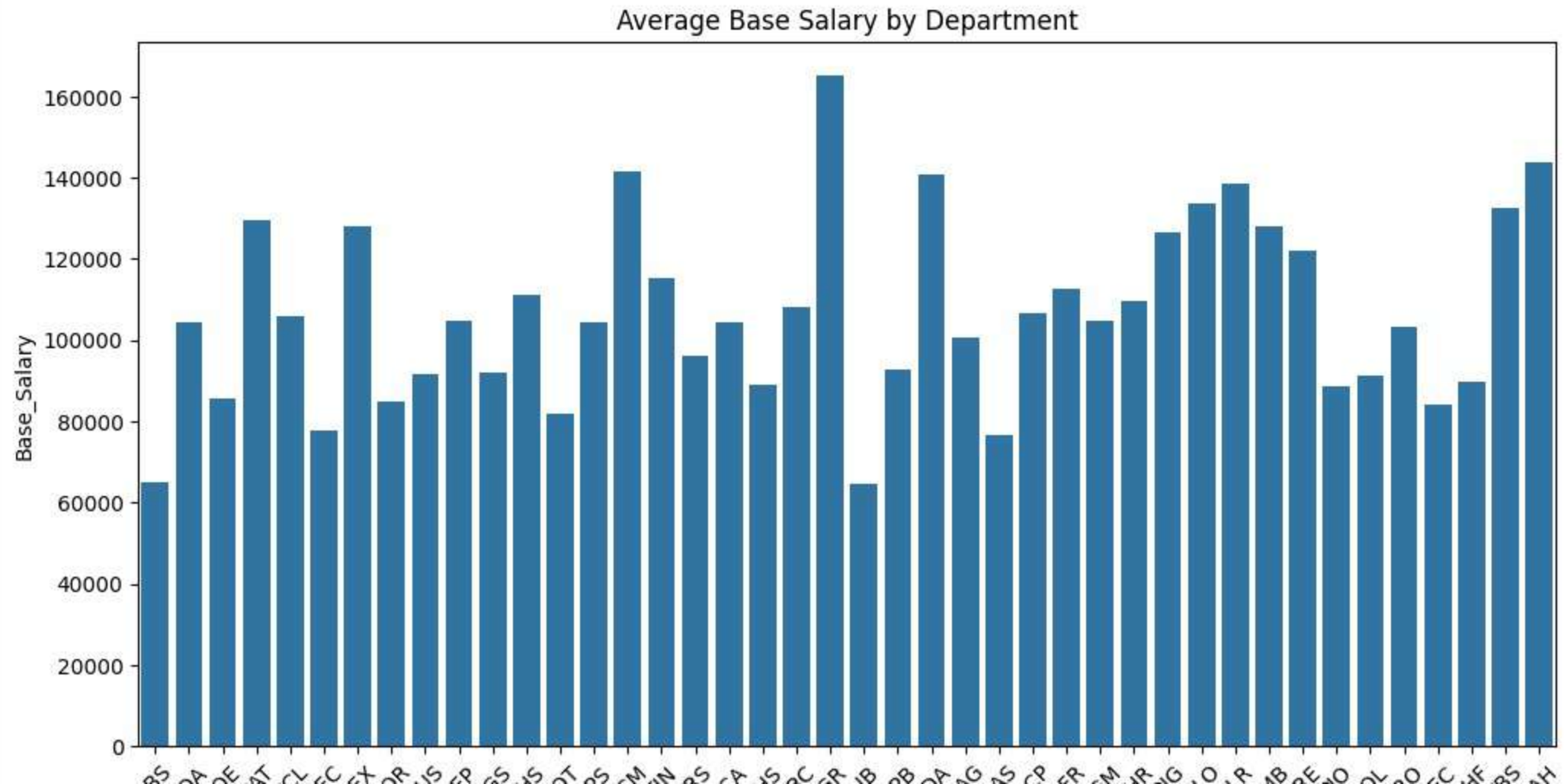
1 import matplotlib.pyplot as plt
2 import seaborn as sns
3
4 # Bar chart for average Base Salary by Department
5 plt.figure(figsize=(12, 6))
6 sns.barplot(data=employee_data, x='Department', y='Base_Salary', ci=None)
7 plt.title("Average Base Salary by Department")
8 plt.xticks(rotation=45)
9 plt.show()
10

```

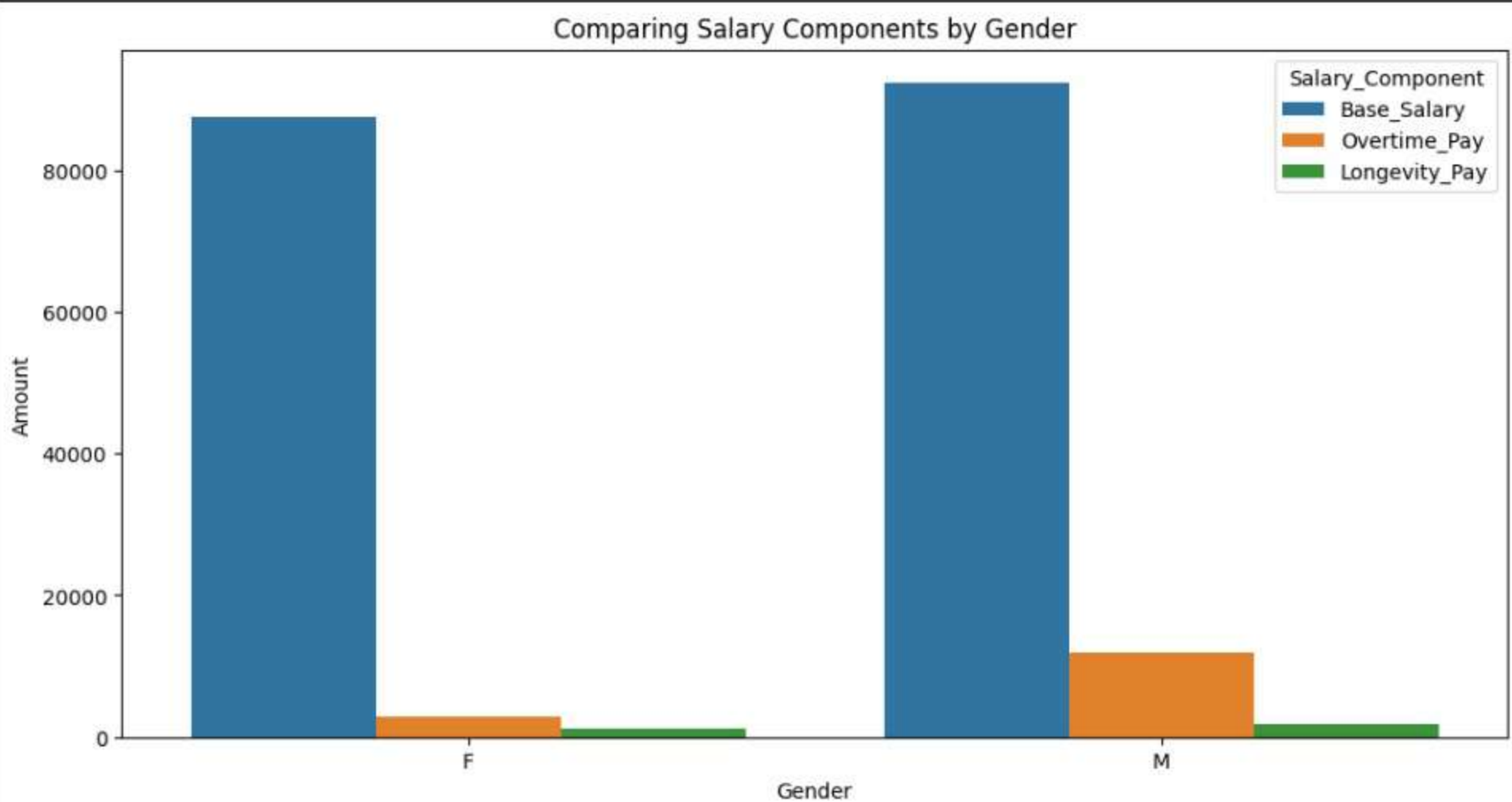
<ipython-input-11-d0727e033b0f>:6: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

```
sns.barplot(data=employee_data, x='Department', y='Base_Salary', ci=None)
```



```
1 # Grouping by Gender to calculate average Base Salary, Overtime Pay, and Longevity Pay
2 gender_salary = employee_data.groupby('Gender')[['Base_Salary', 'Overtime_Pay', 'Longevity_Pay']].mean().reset_index()
3
4 # Melt the DataFrame for easier plotting
5 salary_melted = gender_salary.melt(id_vars='Gender', var_name='Salary_Component', value_name='Amount')
6
7 # Plot
8 plt.figure(figsize=(12, 6))
9 sns.barplot(data=salary_melted, x='Gender', y='Amount', hue='Salary_Component')
10 plt.title("Comparing Salary Components by Gender")
11 plt.show()
12
```

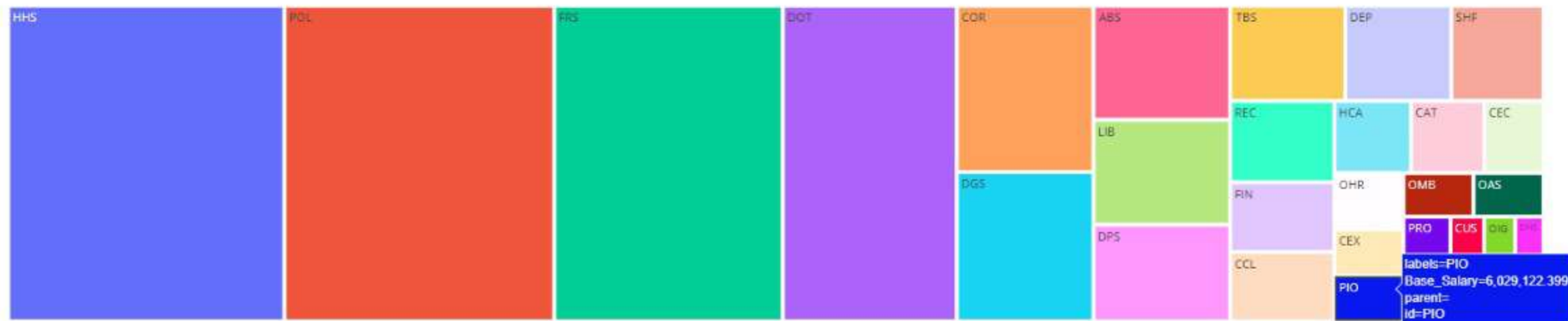


```

1 import plotly.express as px
2
3 # Group data by Department and sum Base Salary
4 dept_salary = employee_data.groupby('Department')['Base_Salary'].sum().reset_index()
5
6 # Create Treemap Chart
7 fig = px.treemap(dept_salary, path=['Department'], values='Base_Salary', title="Treemap of Salary Distribution by Department")
8 fig.show()
9

```

Treemap of Salary Distribution by Department



ML-Lab2 - Colab

colab.research.google.com/drive/1HuS73laNI4PWjT0sh6NLgdatkrEfupQg#scrollTo=wIMnT-QtFFPE

ML-Lab2

File Edit View Insert Runtime Tools Help All changes saved

Files

Marketing-File

sample\_data

README.md

anscombe.js...

california\_housi...

california\_housi...

mnist\_test.csv

mnist\_train\_s...

Employee\_Salarie...

+ Code + Text

```
1 # Basic summary of Employee Salaries data
2 print("\nSummary of Employee Salaries Data:")
3 print(employee_data.describe())
4
```

Summary of Employee Salaries Data:

	Base_Salary	Overtime_Pay	Longevity_Pay
count	10291.000000	10291.000000	10291.000000
mean	90312.165744	8081.288954	1533.882816
std	31240.842929	16491.833017	3209.041070
min	11147.240000	0.000000	0.000000
25%	70023.000000	0.000000	0.000000
50%	87328.000000	258.420000	0.000000
75%	108084.000000	9190.970000	1225.680000
max	292000.000000	227428.990000	20279.460000

[11]

```
1 import matplotlib.pyplot as plt
2 import seaborn as sns
3
4 # Bar chart for average Base Salary by Department
5 plt.figure(figsize=(12, 6))
6 sns.barplot(data=employee_data, x='Department', y='Base_Salary', ci=None)
7 plt.title("Average Base Salary by Department")
8 plt.xticks(rotation=45)
9 plt.show()
10
```

<ipython-input-11-d0727e033b0f>:6: FutureWarning:

The 'ci' parameter is deprecated. Use 'errorbar=None' for the same effect.

sns.barplot(data=employee\_data, x='Department', y='Base\_Salary', ci=None)

Average Base Salary by Department

0s completed at 10:05 AM

RAM

Disk

Gemini

Snipping Tool

Screenshot copied to clipboard

Automatically saved to screenshots folder.

Markup and share

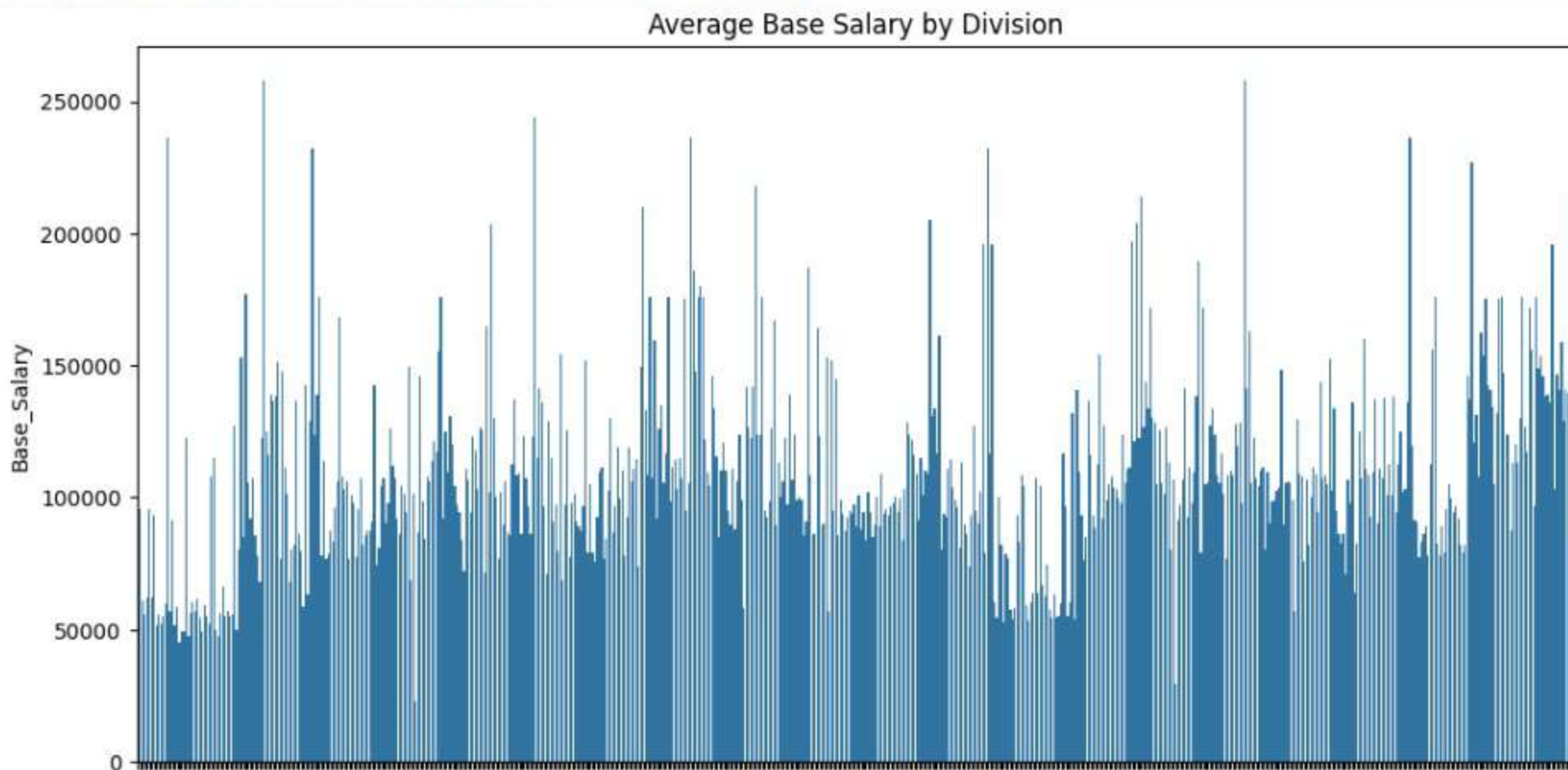


```
1 # Column chart to compare average salary by Division
2 plt.figure(figsize=(12, 6))
3 sns.barplot(data=employee_data, x='Division', y='Base_Salary', ci=None)
4 plt.title("Average Base Salary by Division")
5 plt.xticks(rotation=45)
6 plt.show()
7
```

<ipython-input-19-ed86b1a23cca>:3: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

```
sns.barplot(data=employee_data, x='Division', y='Base_Salary', ci=None)
```



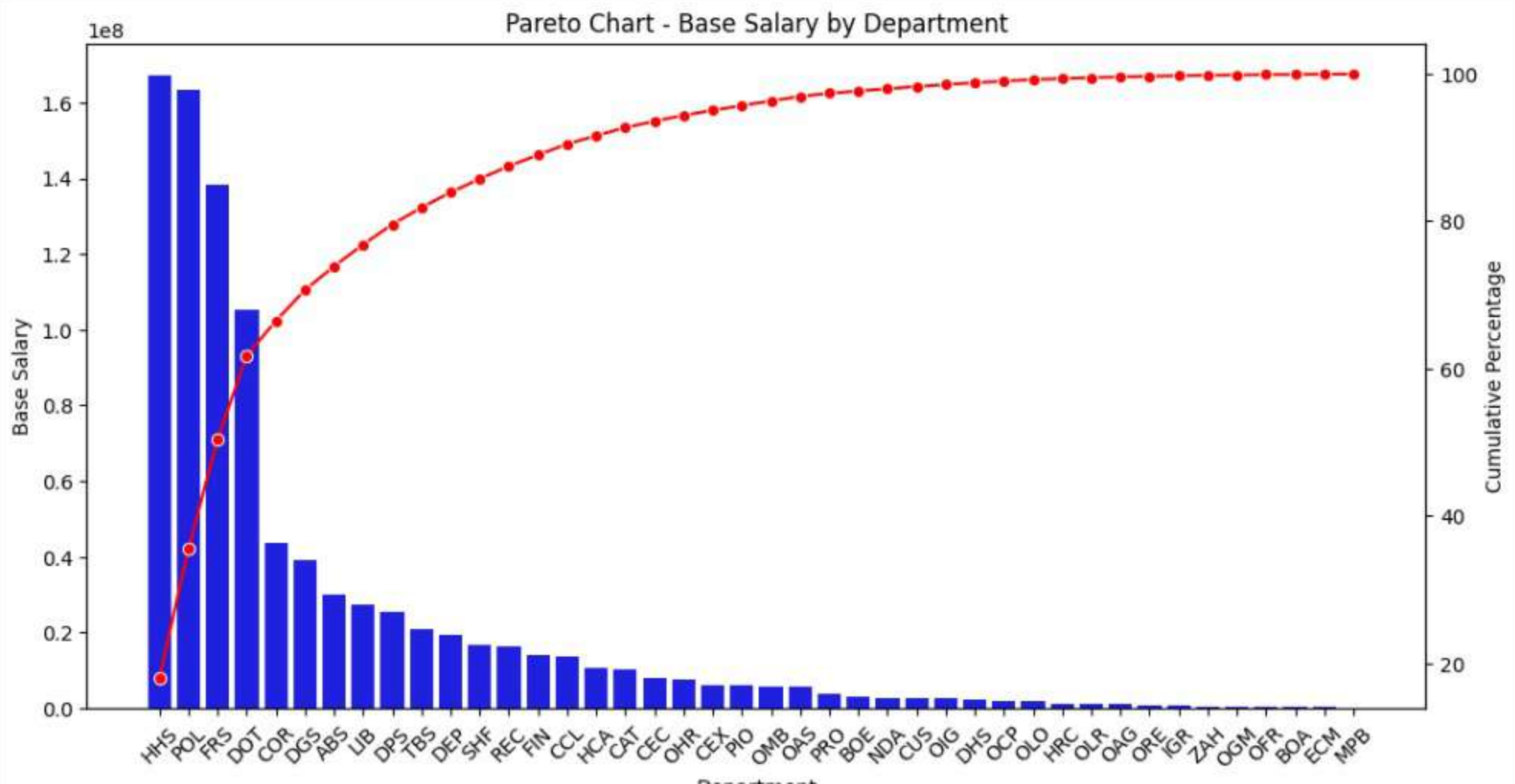
[illegible]

Division

```

1 # Sort by Base Salary
2 dept_salary_sorted = dept_salary.sort_values(by='Base_Salary', ascending=False)
3 dept_salary_sorted['Cumulative_Percentage'] = dept_salary_sorted['Base_Salary'].cumsum() / dept_salary_sorted['Base_Salary'].sum() * 100
4
5 # Plot Pareto Chart
6 plt.figure(figsize=(12, 6))
7 sns.barplot(data=dept_salary_sorted, x='Department', y='Base_Salary', color='b')
8 plt.xticks(rotation=45)
9 plt.ylabel("Base Salary")
10 plt.title("Pareto Chart - Base Salary by Department")
11
12 # Plot cumulative percentage
13 plt.twinx()
14 sns.lineplot(data=dept_salary_sorted, x='Department', y='Cumulative_Percentage', color='r', marker='o')
15 plt.ylabel("Cumulative Percentage")
16 plt.show()
17

```

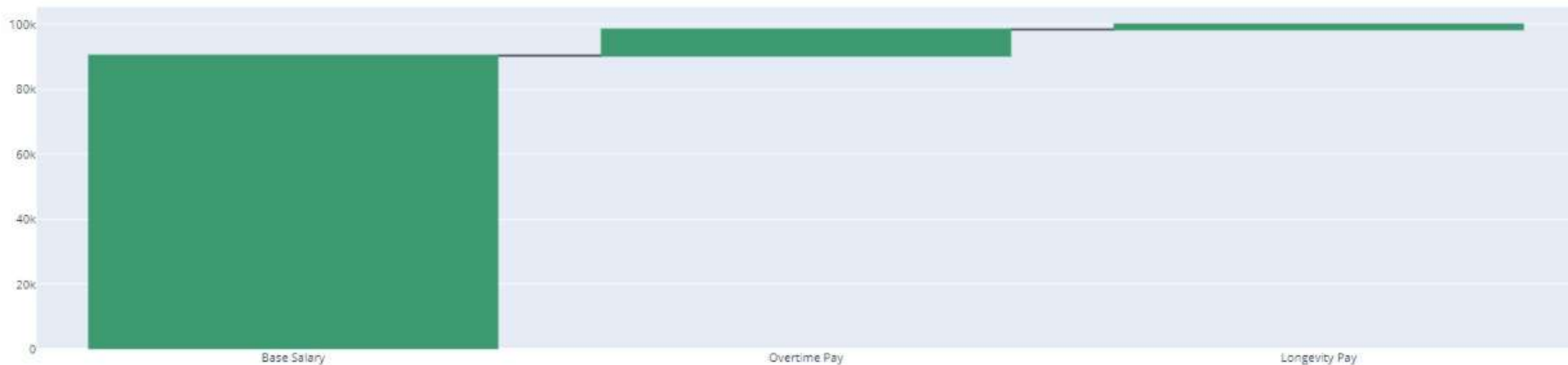


```

1 import plotly.graph_objects as go
2
3 # Prepare sample data
4 salary_components = {
5     'Component': ['Base Salary', 'Overtime Pay', 'Longevity Pay'],
6     'Amount': [
7         employee_data['Base_Salary'].mean(),
8         employee_data['Overtime_Pay'].mean(),
9         employee_data['Longevity_Pay'].mean()
10    ]
11 }
12
13 # Create Waterfall Chart
14 fig = go.Figure(go.Waterfall(
15     x=salary_components['Component'],
16     y=salary_components['Amount'],
17     connector=dict(line=dict(color="rgb(63, 63, 63)"))
18 ))
19
20 fig.update_layout(title="Waterfall Chart - Salary Components Breakdown")
21 fig.show()
22

```

Waterfall Chart - Salary Components Breakdown





```

1 # Create a sample Hire Date column if it doesn't exist
2 if 'Hire Date' not in employee_data.columns:
3     employee_data['Hire Date'] = pd.date_range(start='1/1/2015', periods=len(employee_data))
4
5 # Convert Hire Date to datetime
6 employee_data['Hire Date'] = pd.to_datetime(employee_data['Hire Date'])
7
8 # Plot Area Chart
9 plt.figure(figsize=(12, 6))
10 sns.lineplot(data=employee_data, x='Hire Date', y='Base_Salary', ci=None)
11 plt.fill_between(employee_data['Hire Date'], employee_data['Base_Salary'], alpha=0.3)
12 plt.title("Area Chart - Salary Growth Over Time")
13 plt.xlabel("Hire Date")
14 plt.ylabel("Base Salary")
15 plt.show()
16

```

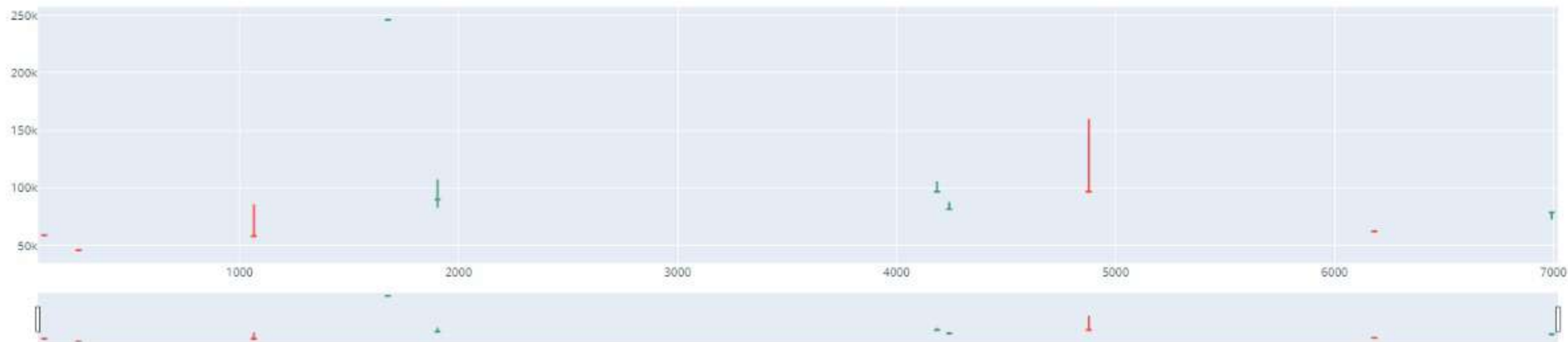
<ipython-input-28-be455d60bfd1>:10: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.





```
1 # Sample Candlestick Chart using Plotly
2 import plotly.graph_objects as go
3
4 # Example data
5 candlestick_data = employee_data[['Base_Salary', 'Overtime_Pay', 'Longevity_Pay']].sample(10)
6
7 # Plot Candlestick Chart
8 fig = go.Figure(data=[go.Candlestick(
9     x=candlestick_data.index,
10     open=candlestick_data['Base_Salary'],
11     high=candlestick_data['Base_Salary'] + candlestick_data['Overtime_Pay'],
12     low=candlestick_data['Base_Salary'] - candlestick_data['Longevity_Pay'],
13     close=candlestick_data['Base_Salary']
14 )])
15 fig.show()
16
```

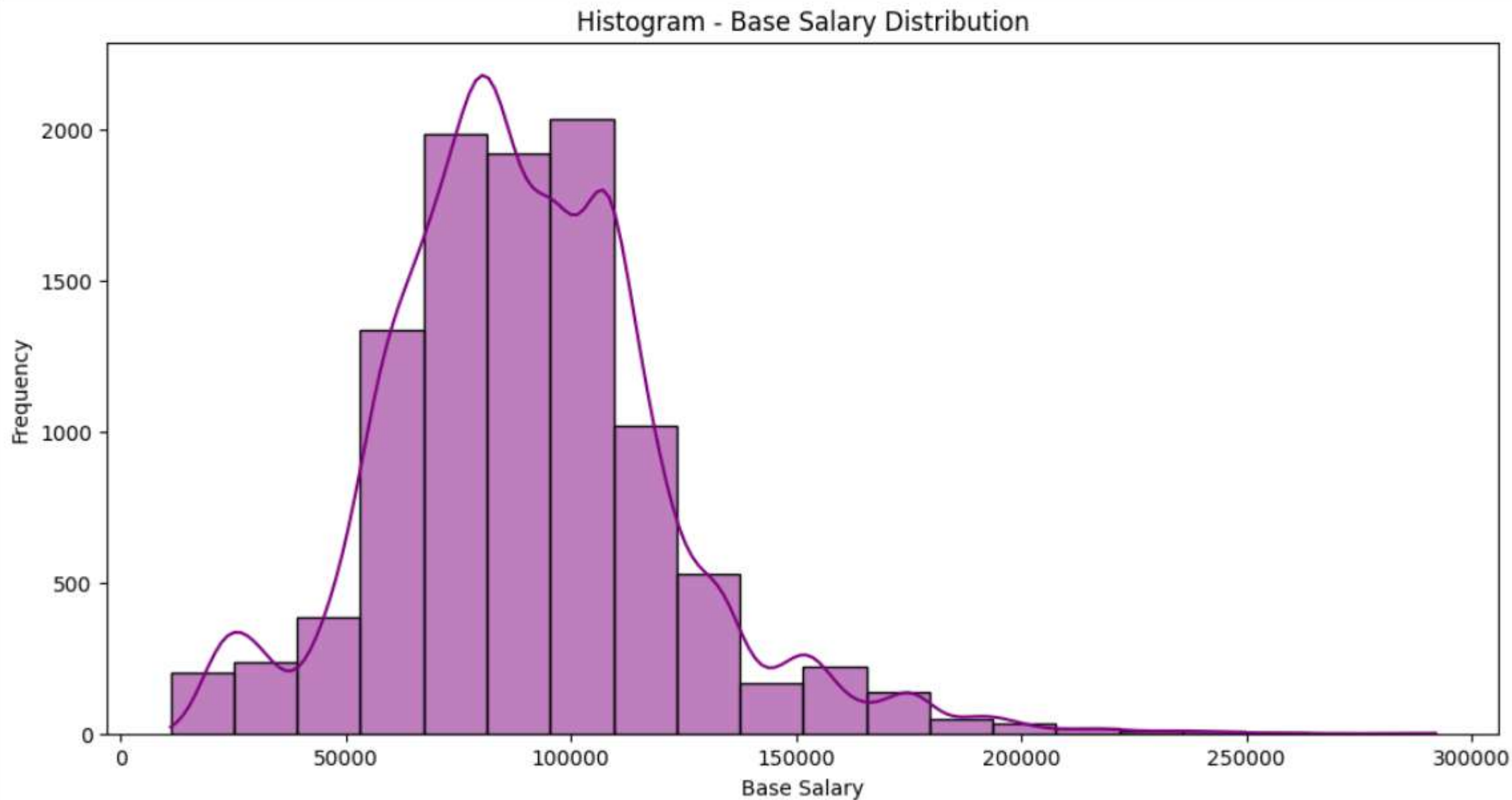




A donut chart titled "Donut Chart - Salary Distribution by Department" showing the percentage distribution of salaries across various departments. The chart is divided into 28 segments, each labeled with a department code and its corresponding percentage. The segments are arranged in a clockwise direction starting from the top.

Department	Percentage
POL	17.6%
PRO	1.4%
REC	0.1%
SHF	1.8%
TBS	2.3%
ZAH	0.1%
ABS	3.2%
BOA	0.1%
CAT	0.1%
CCL	1.5%
CEC	0.9%
CEX	0.7%
COR	4.7%
CUS	0.3%
DEP	2.1%
DGS	4.2%
DHS	0.3%
DOT	11.3%
DPS	2.7%
ECMIN	0.1%
FRS	14.9%
HCA	1.1%
HHS	18.0%
IRRC	3.0%
LIB	0.1%
PLG	0.1%
PLP	0.1%
PLS	0.1%
PLT	0.1%
PLU	0.1%
PLV	0.1%
PLW	0.1%
PLX	0.1%
PLY	0.1%
PLZ	0.1%

```
1 # Plot histogram of Base Salary
2 plt.figure(figsize=(12, 6))
3 sns.histplot(employee_data['Base_Salary'], kde=True, bins=20, color='purple')
4 plt.title("Histogram - Base Salary Distribution")
5 plt.xlabel("Base Salary")
6 plt.ylabel("Frequency")
7 plt.show()
8
```



```

1 # Create a sample Hire Date column if it doesn't exist
2 if 'Hire Date' not in employee_data.columns:
3     employee_data['Hire Date'] = pd.date_range(start='1/1/2015', periods=len(employee_data))
4
5 # Convert Hire Date to datetime
6 employee_data['Hire Date'] = pd.to_datetime(employee_data['Hire Date'])
7
8 # Plot Area Chart
9 plt.figure(figsize=(12, 6))
10 sns.lineplot(data=employee_data, x='Hire Date', y='Base_Salary', ci=None)
11 plt.fill_between(employee_data['Hire Date'], employee_data['Base_Salary'], alpha=0.3)
12 plt.title("Area Chart - Salary Growth Over Time")
13 plt.xlabel("Hire Date")
14 plt.ylabel("Base Salary")
15 plt.show()
16

```

<ipython-input-28-be455d60bfd1>:10: FutureWarning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

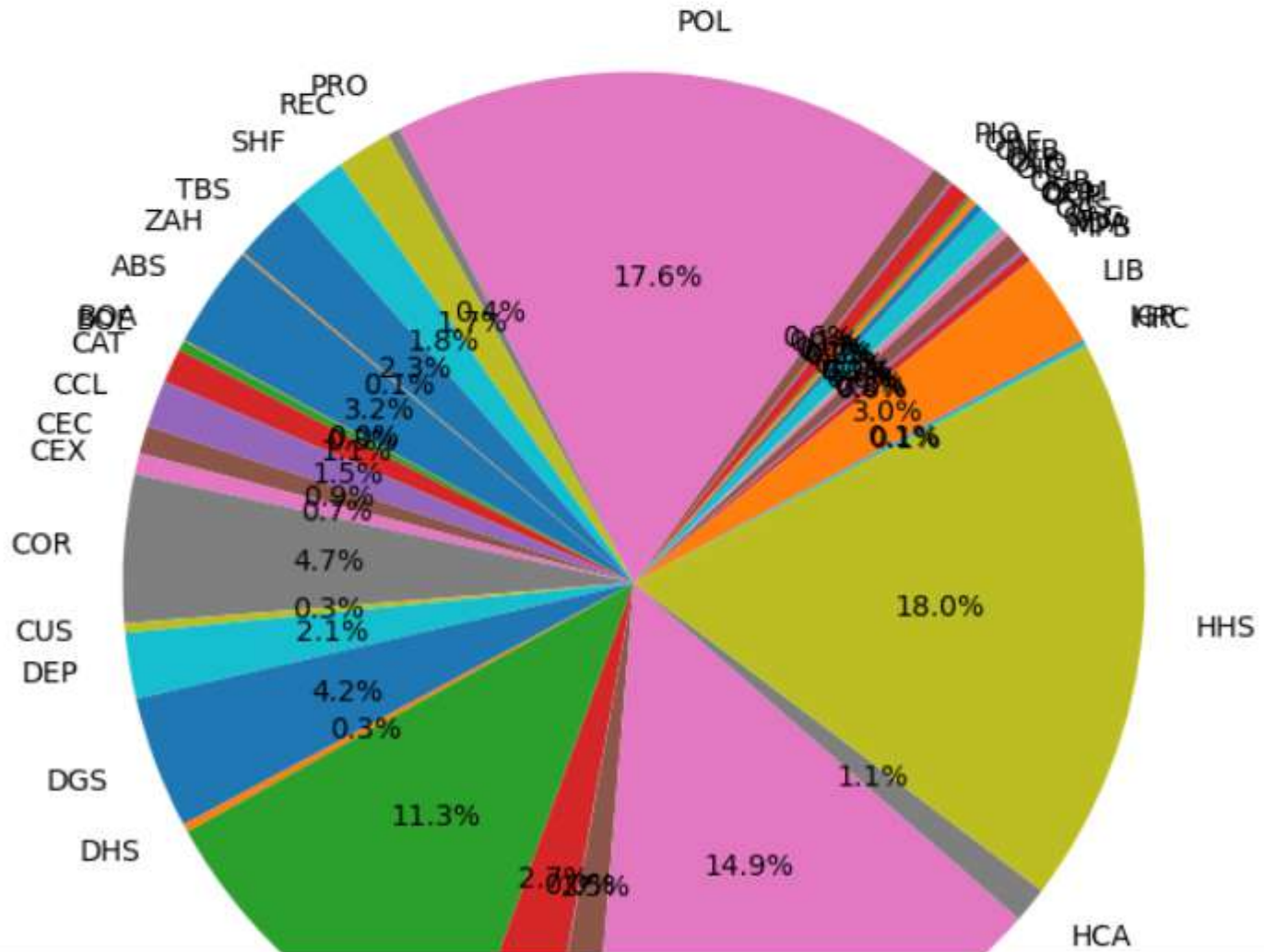


```

1 # Group data by Department and sum the Base Salary
2 dept_salary = employee_data.groupby('Department')['Base_Salary'].sum()
3
4 # Plot Pie Chart
5 plt.figure(figsize=(10, 8))
6 plt.pie(dept_salary, labels=dept_salary.index, autopct='%1.1f%%', startangle=140)
7 plt.title("Pie Chart - Salary Distribution by Department")
8 plt.show()
9

```

Pie Chart - Salary Distribution by Department



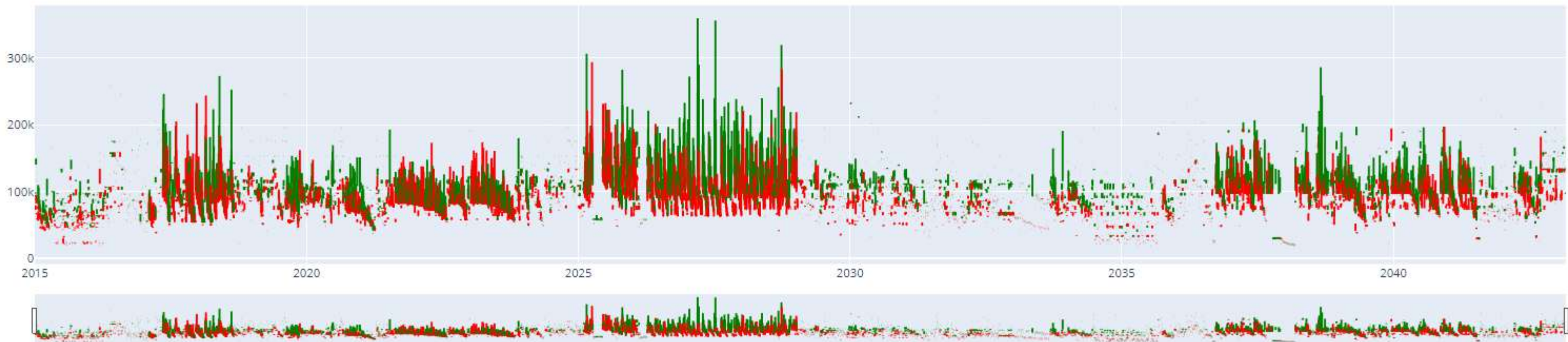


```

1 from plotly.graph_objects import Candlestick
2
3 # Create a sample 'Hire Date' column if it doesn't exist
4 if 'Hire Date' not in employee_data.columns:
5     employee_data['Hire Date'] = pd.date_range(start='1/1/2015', periods=len(employee_data))
6
7 # Sort data by date
8 employee_data = employee_data.sort_values(by='Hire Date')
9
10 # Create Candlestick Chart
11 fig = go.Figure(data=[go.Candlestick(
12     x=employee_data['Hire Date'],
13     open=employee_data['Base_Salary'],
14     high=employee_data['Base_Salary'] + employee_data['Overtime_Pay'],
15     low=employee_data['Base_Salary'] - employee_data['Longevity_Pay'],
16     close=employee_data['Base_Salary'],
17     increasing_line_color='green', decreasing_line_color='red'
18 )])
19 fig.update_layout(title="Candlestick Chart - Salary Components Over Time")
20 fig.show()
21

```

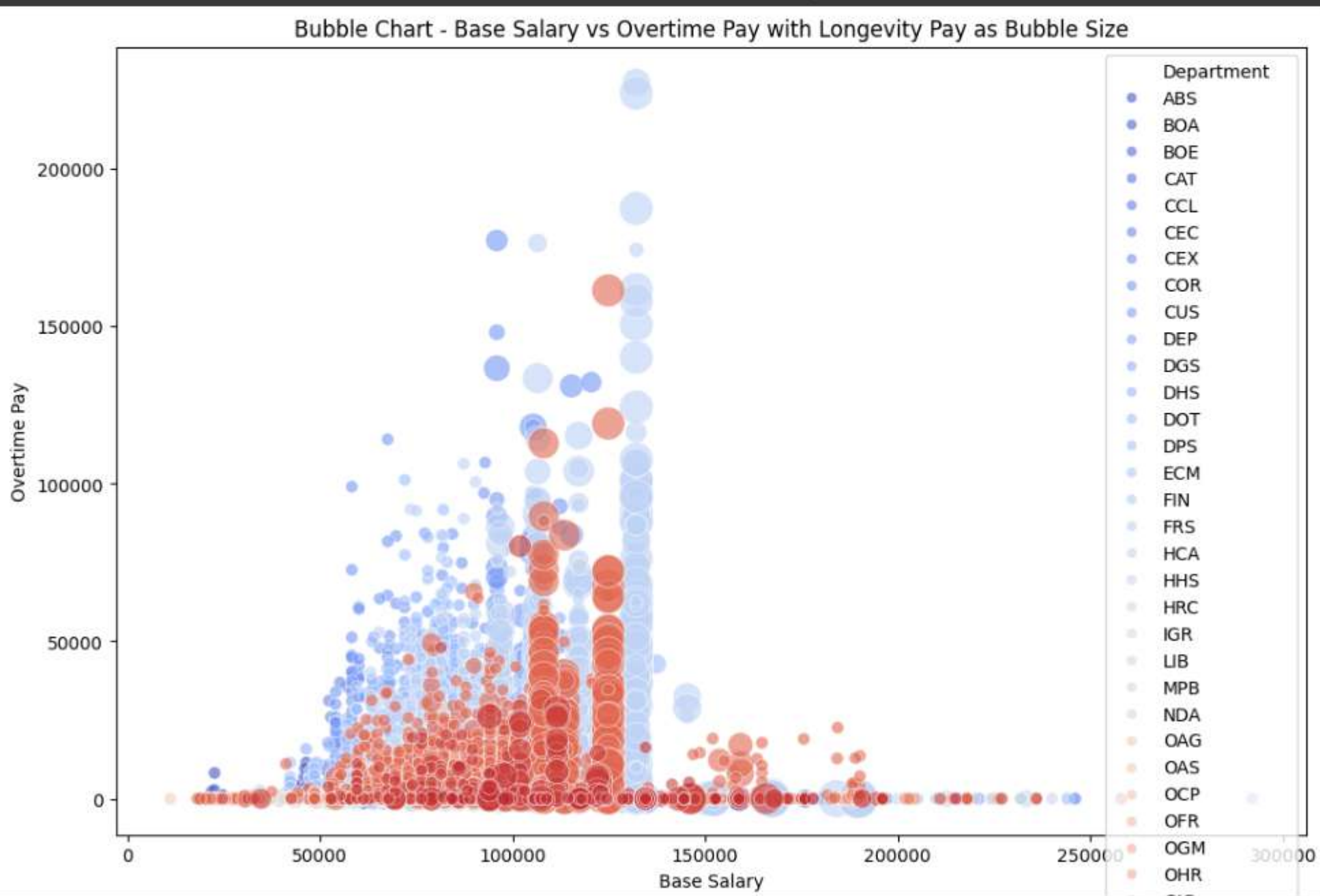
Candlestick Chart - Salary Components Over Time



```

1 # Bubble chart to show Base Salary vs Overtime Pay, with Longevity Pay as bubble size
2 plt.figure(figsize=(12, 8))
3 sns.scatterplot(data=employee_data, x='Base_Salary', y='Overtime_Pay', size='Longevity_Pay', hue='Department', alpha=0.6, palette="coolwarm", sizes=(50, 500))
4 plt.title("Bubble Chart - Base Salary vs Overtime Pay with Longevity Pay as Bubble Size")
5 plt.xlabel("Base Salary")
6 plt.ylabel("Overtime Pay")
7 plt.legend(loc="upper right")
8 plt.show()
9

```



```

1 import plotly.express as px
2
3 # Funnel chart to show average salary by department
4 dept_avg_salary = employee_data.groupby('Department')['Base_Salary'].mean().reset_index()
5
6 # Create Funnel Chart
7 fig = px.funnel(dept_avg_salary, x='Base_Salary', y='Department', title="Funnel Chart - Average Salary by Department")
8 fig.show()
9

```

Funnel Chart - Average Salary by Department

