

# Panda|Compare DataFrame

```
In [1]: import pandas as pd # For data manipulation and analysis
import numpy as np # For numerical computations
import matplotlib.pyplot as plt # For creating static, animated, and interactive visualizations
import seaborn as sns # For statistical data visualization based on Matplotlib
import scipy # For scientific and technical computing (including optimization, integration, and statistics)
```

```
In [3]: import pandas as pd
import numpy as np
# Creating realistic data for employees
data = {
    'Employee ID': np.arange(1001, 1011),
    'Employee Name': ['Satender Kumar', 'data 1', 'Jane Smith', 'Robert Brown', 'Emily Davis', 'Michael Wilson', 'Sarah Taylor', 'David Lee', 'Laura Johnson', 'James White'],
    'Department': ['Data Analyst', 'IT', 'Finance', 'Marketing', 'Sales', 'Operations', 'R&D', 'Support', 'Admin', 'Legal'],
    'Age': [24, np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60)],
    'Location': ['London, Canada', 'Toronto', 'London', 'Sydney', 'San Francisco', 'Paris', 'Berlin', 'Tokyo', 'Dubai', 'Singapore'],
    'Salary': np.random.randint(50000, 150000, size=10),
    'Years with Company': np.random.randint(1, 15, size=10),
    'Position': ['Data Analyst', 'Developer', 'Analyst', 'Designer', 'Consultant', 'Engineer', 'Scientist', 'Support Agent', 'Admin Assistant', 'Lawyer'],
    'Performance Score': np.random.randint(1, 5, size=10),
    'Bonus': np.random.randint(1000, 10000, size=10),
    'Gender': ['Male', 'Male', 'Female', 'Male', 'Female', 'Male', 'Female', 'Male', 'Female', 'Male'],
    'Marital Status': ['Single', 'Single', 'Married', 'Single', 'Single', 'Married', 'Married', 'Single', 'Married', 'Single'],
    'Education': ['Bachelor', 'Master', 'PhD', 'Bachelor', 'Master', 'PhD', 'Bachelor', 'Master', 'Bachelor', 'Master'],
    'Hire Date': pd.to_datetime(['2019-06-12', '2015-07-23', '2012-09-05', '2018-11-30', '2013-05-19', '2019-02-14', '2017-08-01', '2016-03-08', '2014-10-20', '2011-01-01']),
    'Overtime Hours': np.random.randint(0, 20, size=10),
    'Sick Days Taken': np.random.randint(0, 10, size=10),
    'Vacation Days Taken': np.random.randint(5, 20, size=10),
    'Training Hours': np.random.randint(10, 50, size=10),
    'Certifications': ['Yes', 'No', 'Yes', 'No', 'Yes', 'No', 'Yes', 'Yes', 'No', 'Yes'],
    'Supervisor': ['Anna Smith', 'Brian Adams', 'Clara Jones', 'Daniel Martin', 'Eva Rodriguez', 'Frank Bell', 'Grace Kim', 'Henry Lopez', 'Ivy Nguyen', 'Jack Parker'],
}

# Creating the DataFrame
df = pd.DataFrame(data)
```

In [5]: df

Out[5]:

	Employee ID	Employee Name	Department	Age	Location	Salary	Years with Company	Position	Performance Score	Bonus	Gender	Marital Status	Education
0	1001	Satender Kumar	Data Analyst	24	London, Canada	113479	3	Data Analyst	4	9480	Male	Single	Bachelor
1	1002	data 1	IT	34	Toronto	127453	9	Developer	4	6577	Male	Single	Bachelor
2	1003	Jane Smith	Finance	27	London	118316	7	Analyst	1	4972	Female	Married	Master
3	1004	Robert Brown	Marketing	59	Sydney	109725	10	Designer	3	4624	Male	Single	PhD
4	1005	Emily Davis	Sales	36	San Francisco	99006	14	Consultant	3	3218	Female	Single	Bachelor
5	1006	Michael Wilson	Operations	36	Paris	123386	8	Engineer	1	4450	Male	Married	Master
6	1007	Sarah Taylor	R&D	54	Berlin	77293	13	Scientist	2	2823	Female	Married	PhD
7	1008	David Lee	Support	32	Tokyo	65542	9	Support Agent	2	1486	Male	Single	Bachelor
8	1009	Laura Johnson	Admin	50	Dubai	87223	9	Admin Assistant	2	1705	Female	Married	Master
9	1010	James White	Legal	25	Singapore	110051	5	Lawyer	2	8724	Male	Single	PhD

```
In [6]: # Creating realistic data for a second set of employees
data1 = {
    'Employee ID': np.arange(1011, 1021),
    'Employee Name': ['Satender Kumar', 'data 1', 'Chris Evans', 'Natalie Portman', 'Tom Holland', 'Emma Watson', 'Daniel Craig', 'Scarlett Johansson', 'Robert Downey Jr.', 'Mark Ruffalo'],
    'Department': ['Data Analyst', 'HR', 'IT', 'Marketing', 'Finance', 'Sales', 'R&D', 'Operations', 'Legal', 'Support'],
    'Age': [24, np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60), np.random.randint(25, 60)],
    'Location': ['London, Canada', 'Los Angeles', 'New York', 'Chicago', 'Houston', 'Phoenix', 'Philadelphia', 'San Antonio', 'Dallas', 'Austin'],
    'Salary': np.random.randint(60000, 160000, size=10),
    'Years with Company': np.random.randint(1, 20, size=10),
    'Position': ['Data Analyst', 'HR Manager', 'IT Specialist', 'Marketing Coordinator', 'Financial Analyst', 'Sales Representative', 'Operations Manager', 'Legal Counsel', 'Support Specialist', 'Project Manager'],
    'Performance Score': np.random.randint(1, 5, size=10),
    'Bonus': np.random.randint(2000, 12000, size=10),
    'Gender': ['Male', 'Male', 'Female', 'Female', 'Male', 'Female', 'Male', 'Female', 'Male', 'Male'],
}
```

```
'Marital Status': ['Single', 'Married', 'Single', 'Single', 'Married', 'Single', 'Single', 'Married', 'Single',
'Education': ['Master', 'Bachelor', 'Master', 'PhD', 'Bachelor', 'Master', 'PhD', 'Bachelor', 'Master', 'PhD'],
'Hire Date': pd.to_datetime(['2018-07-15', '2014-03-22', '2011-10-12', '2017-04-17', '2015-09-23', '2016-11-01',
'Overtime Hours': np.random.randint(0, 25, size=10),
'Sick Days Taken': np.random.randint(0, 8, size=10),
'Vacation Days Taken': np.random.randint(7, 22, size=10),
'Training Hours': np.random.randint(15, 55, size=10),
'Certifications': ['Yes', 'Yes', 'No', 'Yes', 'No', 'Yes', 'No', 'Yes', 'Yes', 'No'],
'Supervisor': ['John Smith', 'Michael Johnson', 'Patricia Williams', 'Linda Brown', 'Barbara Jones', 'Elizabeth C
}

# Creating the second DataFrame
df1 = pd.DataFrame(data1)
```

In [8]: df1

Out[8]:

	Employee ID	Employee Name	Department	Age	Location	Salary	Years with Company	Position	Performance Score	Bonus	Gender	Marital Status
0	1011	Satender Kumar	Data Analyst	24	London, Canada	125763	9	Data Analyst	4	9043	Male	Single
1	1012	data 1	HR	29	Los Angeles	149163	18	HR Manager	4	5174	Male	Married
2	1013	Chris Evans	IT	58	New York	149334	15	IT Specialist	4	4389	Female	Single
3	1014	Natalie Portman	Marketing	41	Chicago	152579	13	Marketing Coordinator	4	3100	Female	Single
4	1015	Tom Holland	Finance	55	Houston	87396	11	Financial Analyst	1	5710	Male	Married
5	1016	Emma Watson	Sales	34	Phoenix	132586	8	Sales Manager	1	2777	Female	Single
6	1017	Daniel Radcliffe	R&D	56	Philadelphia	143832	10	Research Scientist	2	6264	Male	Single
7	1018	Scarlett Johansson	Operations	35	San Antonio	61204	1	Operations Manager	4	9008	Female	Married
8	1019	Robert Downey Jr.	Legal	26	San Diego	69724	17	Legal Advisor	4	7002	Male	Single
9	1020	Mark Ruffalo	Support	33	Dallas	134630	11	Support Specialist	1	2651	Male	Married

In [11]: df.head()

Out[11]:

	Employee ID	Employee Name	Department	Age	Location	Salary	Years with Company	Position	Performance Score	Bonus	Gender	Marital Status	Education
0	1001	Satender Kumar	Data Analyst	24	London, Canada	113479	3	Data Analyst	4	9480	Male	Single	Bachelor's
1	1002	data 1	IT	34	Toronto	127453	9	Developer	4	6577	Male	Single	Master's
2	1003	Jane Smith	Finance	27	London	118316	7	Analyst	1	4972	Female	Married	Master's
3	1004	Robert Brown	Marketing	59	Sydney	109725	10	Designer	3	4624	Male	Single	Bachelor's
4	1005	Emily Davis	Sales	36	San Francisco	99006	14	Consultant	3	3218	Female	Single	Master's

In [13]: df1.head()

Out[13]:

	Employee ID	Employee Name	Department	Age	Location	Salary	Years with Company	Position	Performance Score	Bonus	Gender	Marital Status	Ec
0	1011	Satender Kumar	Data Analyst	24	London, Canada	125763	9	Data Analyst	4	9043	Male	Single	
1	1012	data 1	HR	29	Los Angeles	149163	18	HR Manager	4	5174	Male	Married	
2	1013	Chris Evans	IT	58	New York	149334	15	IT Specialist	4	4389	Female	Single	
3	1014	Natalie Portman	Marketing	41	Chicago	152579	13	Marketing Coordinator	4	3100	Female	Single	
4	1015	Tom Holland	Finance	55	Houston	87396	11	Financial Analyst	1	5710	Male	Married	

```
In [15]: # Check if df and df1 are exactly the same
are_identical = df.equals(df1)
print(f"Are df and df1 identical? {are_identical}")
```

Are df and df1 identical? False

```
In [17]: # Find differences between df and df1
comparison_df = df.compare(df1, keep_shape=True, keep_equal=True)
print("Differences between df and df1:")
print(comparison_df)
```

Differences between df and df1:

	Employee ID		Employee Name		Department \
	self	other	self	other	
0	1001	1011	Satender Kumar	Satender Kumar	Data Analyst
1	1002	1012	data 1	data 1	IT
2	1003	1013	Jane Smith	Chris Evans	Finance
3	1004	1014	Robert Brown	Natalie Portman	Marketing
4	1005	1015	Emily Davis	Tom Holland	Sales
5	1006	1016	Michael Wilson	Emma Watson	Operations
6	1007	1017	Sarah Taylor	Daniel Radcliffe	R&D
7	1008	1018	David Lee	Scarlett Johansson	Support
8	1009	1019	Laura Johnson	Robert Downey Jr.	Admin
9	1010	1020	James White	Mark Ruffalo	Legal

		Age		Location		... \
		other	self	other	self	
0	Data Analyst	24	24	London, Canada	London, Canada	...
1	HR	34	29	Toronto	Los Angeles	...
2	IT	27	58	London	New York	...
3	Marketing	59	41	Sydney	Chicago	...
4	Finance	36	55	San Francisco	Houston	...
5	Sales	36	34	Paris	Phoenix	...
6	R&D	54	56	Berlin	Philadelphia	...
7	Operations	32	35	Tokyo	San Antonio	...
8	Legal	50	26	Dubai	San Diego	...
9	Support	25	33	Singapore	Dallas	...

	Sick Days Taken		Vacation Days Taken		Training Hours		\
	self	other	self	other	self	other	
0	9	7	12	15	19	53	
1	2	5	15	19	31	17	
2	2	6	10	13	30	45	
3	8	4	14	20	29	34	
4	9	1	18	18	14	17	
5	3	7	7	14	14	48	
6	5	4	6	8	11	46	
7	4	0	15	16	21	54	
8	8	4	15	7	44	18	
9	6	6	6	21	45	31	

	Certifications		Supervisor	
	self	other	self	other
0	Yes	Yes	Anna Smith	John Smith
1	No	Yes	Brian Adams	Michael Johnson
2	Yes	No	Clara Jones	Patricia Williams
3	No	Yes	Daniel Martin	Linda Brown
4	Yes	No	Eva Rodriguez	Barbara Jones
5	No	Yes	Frank Bell	Elizabeth Garcia
6	Yes	No	Grace Moore	Susan Martinez
7	Yes	Yes	Hannah Lewis	Jessica Hernandez
8	No	Yes	Ivan Scott	Sarah Lopez
9	Yes	No	Jake Miller	Karen Wilson

[10 rows x 40 columns]

```
In [20]: # Identify rows that differ between df and df1
differing_rows = df[df.ne(df1).any(axis=1)]
print("Rows that differ between df and df1:")
print(differing_rows)
```

Rows that differ between df and df1:

	Employee ID	Employee Name	Department	Age	Location	Salary	\
0	1001	Satender Kumar	Data Analyst	24	London, Canada	113479	
1	1002	data 1	IT	34	Toronto	127453	
2	1003	Jane Smith	Finance	27	London	118316	
3	1004	Robert Brown	Marketing	59	Sydney	109725	
4	1005	Emily Davis	Sales	36	San Francisco	99006	
5	1006	Michael Wilson	Operations	36	Paris	123386	
6	1007	Sarah Taylor	R&D	54	Berlin	77293	
7	1008	David Lee	Support	32	Tokyo	65542	
8	1009	Laura Johnson	Admin	50	Dubai	87223	
9	1010	James White	Legal	25	Singapore	110051	

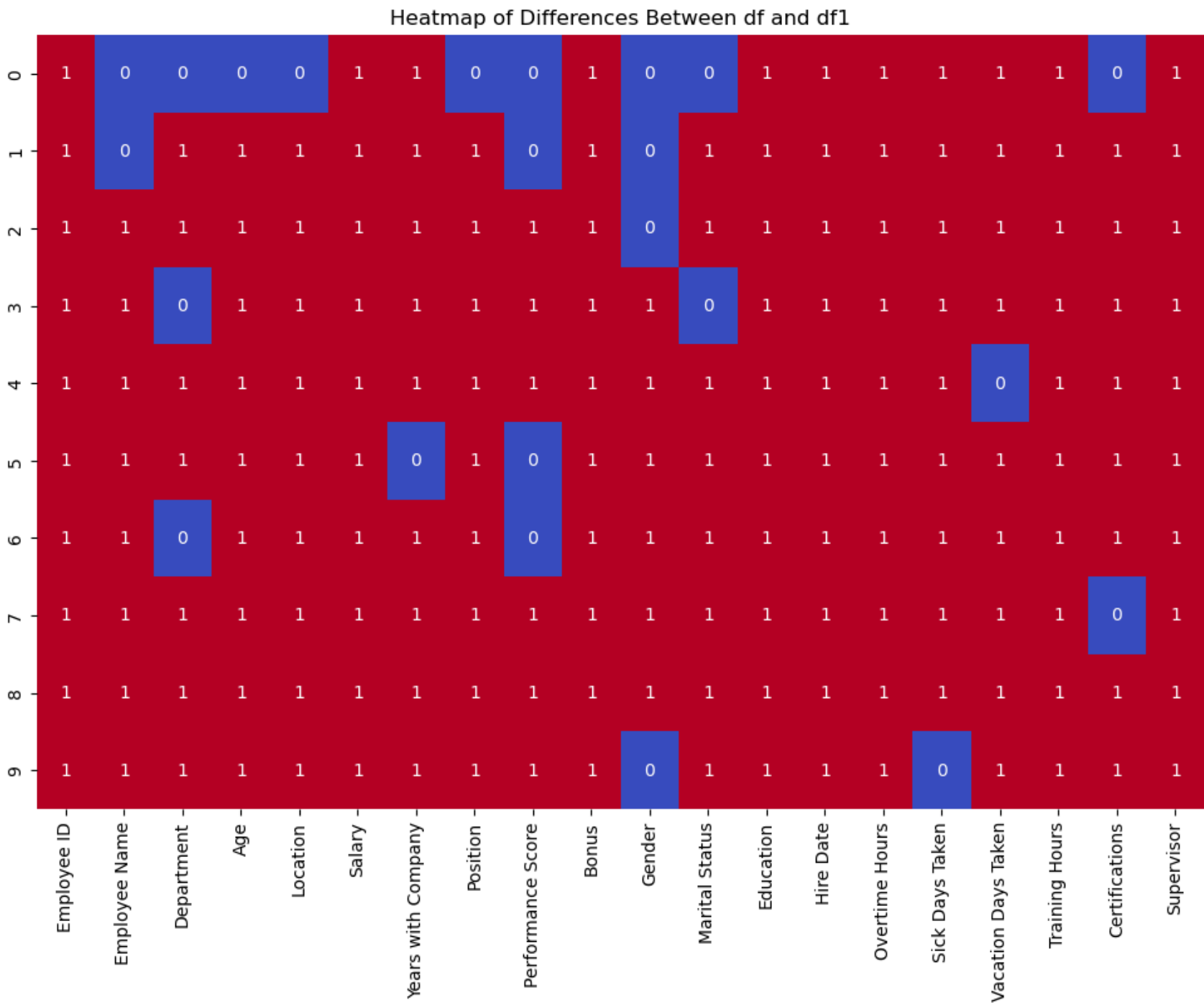
	Years with Company	Position	Performance Score	Bonus	Gender	\
0	3	Data Analyst		4 9480	Male	
1	9	Developer		4 6577	Male	
2	7	Analyst		1 4972	Female	
3	10	Designer		3 4624	Male	
4	14	Consultant		3 3218	Female	
5	8	Engineer		1 4450	Male	
6	13	Scientist		2 2823	Female	
7	9	Support Agent		2 1486	Male	
8	9	Admin Assistant		2 1705	Female	
9	5	Lawyer		2 8724	Male	

	Marital Status	Education	Hire Date	Overtime Hours	Sick Days Taken	\
0	Single	Bachelor	2019-06-12	18	9	
1	Single	Master	2015-07-23	11	2	
2	Married	PhD	2012-09-05	6	2	
3	Single	Bachelor	2018-11-30	17	8	
4	Single	Master	2013-05-19	16	9	
5	Married	PhD	2019-02-14	14	3	
6	Married	Bachelor	2020-08-21	4	5	
7	Single	Master	2016-06-03	3	4	
8	Married	Bachelor	2014-01-28	13	8	
9	Single	Master	2017-03-15	6	6	

	Vacation Days Taken	Training Hours	Certifications	Supervisor
0	12	19	Yes	Anna Smith
1	15	31	No	Brian Adams
2	10	30	Yes	Clara Jones
3	14	29	No	Daniel Martin
4	18	14	Yes	Eva Rodriguez
5	7	14	No	Frank Bell
6	6	11	Yes	Grace Moore
7	15	21	Yes	Hannah Lewis
8	15	44	No	Ivan Scott
9	6	45	Yes	Jake Miller

```
In [22]: import seaborn as sns
import matplotlib.pyplot as plt

# Create a heatmap to visualize differences between df and df1
diff = df.ne(df1).astype(int) # 1 where different, 0 where the same
plt.figure(figsize=(12, 8))
sns.heatmap(diff, cmap='coolwarm', cbar=False, annot=True)
plt.title('Heatmap of Differences Between df and df1')
plt.show()
```



```
In [24]: # Summarize the number of differences by column
summary = df.ne(df1).sum()
print("Summary of differences by column:")
print(summary)
```

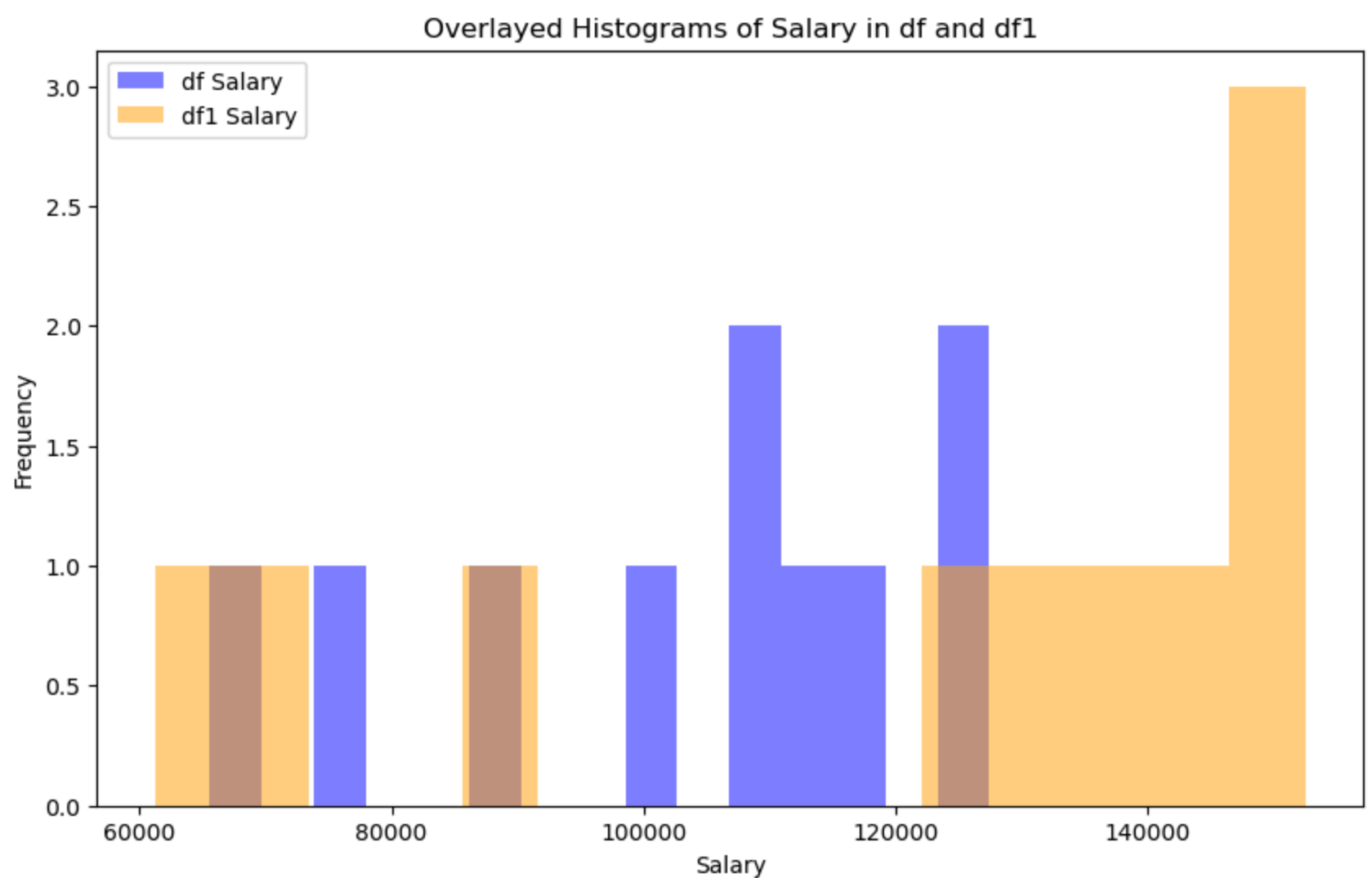
Summary of differences by column:

Employee ID	10
Employee Name	8
Department	7
Age	9
Location	9
Salary	10
Years with Company	9
Position	9
Performance Score	6
Bonus	10
Gender	6
Marital Status	8
Education	10
Hire Date	10
Overtime Hours	10
Sick Days Taken	9
Vacation Days Taken	9
Training Hours	10
Certifications	8
Supervisor	10

dtype: int64

```
In [26]: import matplotlib.pyplot as plt

# Overlaid histograms for 'Salary' in df and df1
plt.figure(figsize=(10, 6))
plt.hist(df['Salary'], bins=15, alpha=0.5, label='df Salary', color='blue')
plt.hist(df1['Salary'], bins=15, alpha=0.5, label='df1 Salary', color='orange')
plt.title('Overlaid Histograms of Salary in df and df1')
plt.xlabel('Salary')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```



```
In [29]: # Combine the data into a single DataFrame for side-by-side boxplots
combined = pd.DataFrame({
    'df_Salary': df['Salary'],
    'df1_Salary': df1['Salary']
})

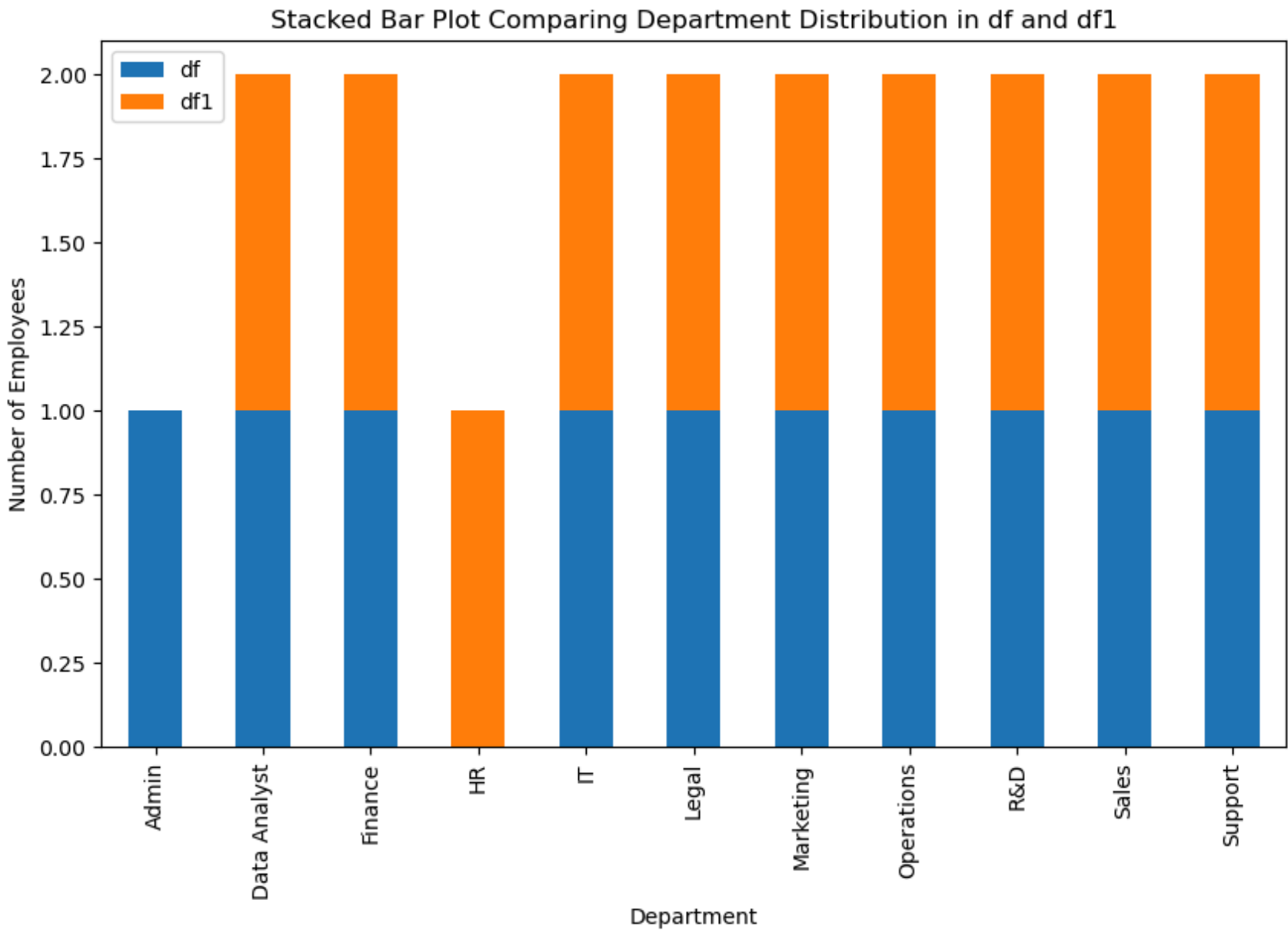
# Plot the boxplots
plt.figure(figsize=(10, 6))
combined.boxplot()
plt.title('Side-by-Side Boxplots of Salary in df and df1')
plt.ylabel('Salary')
plt.show()
```



```
In [31]: # Group by Department and count in both DataFrames
df_dept_counts = df['Department'].value_counts()
df1_dept_counts = df1['Department'].value_counts()

# Combine into a single DataFrame
dept_comparison = pd.DataFrame({'df': df_dept_counts, 'df1': df1_dept_counts})

# Plot stacked bar plot
dept_comparison.plot(kind='bar', stacked=True, figsize=(10, 6))
plt.title('Stacked Bar Plot Comparing Department Distribution in df and df1')
plt.ylabel('Number of Employees')
plt.show()
```



```
In [33]: # Scatter plot comparison for 'Age' vs 'Salary' in both DataFrames
plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)
plt.scatter(df['Age'], df['Salary'], color='blue', alpha=0.5)
plt.title('df: Age vs Salary')
plt.xlabel('Age')
plt.ylabel('Salary')

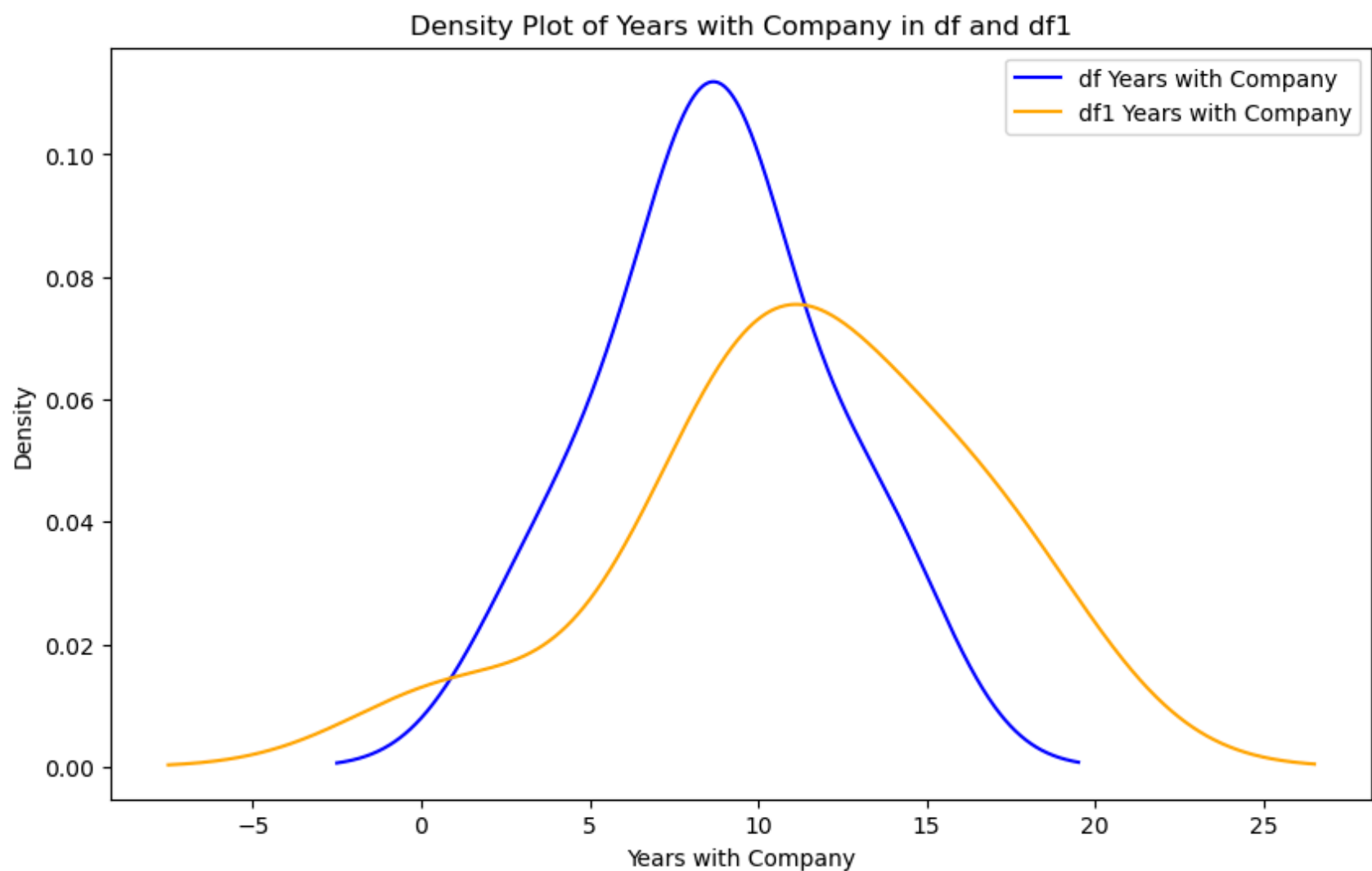
plt.subplot(1, 2, 2)
plt.scatter(df1['Age'], df1['Salary'], color='orange', alpha=0.5)
plt.title('df1: Age vs Salary')
plt.xlabel('Age')
plt.ylabel('Salary')

plt.show()
```



```
In [35]: # Density plot for 'Years with Company' in df and df1
plt.figure(figsize=(10, 6))
df['Years with Company'].plot(kind='density', label='df Years with Company', color='blue')
df1['Years with Company'].plot(kind='density', label='df1 Years with Company', color='orange')
plt.title('Density Plot of Years with Company in df and df1')
plt.xlabel('Years with Company')
plt.legend()
plt.show()
```

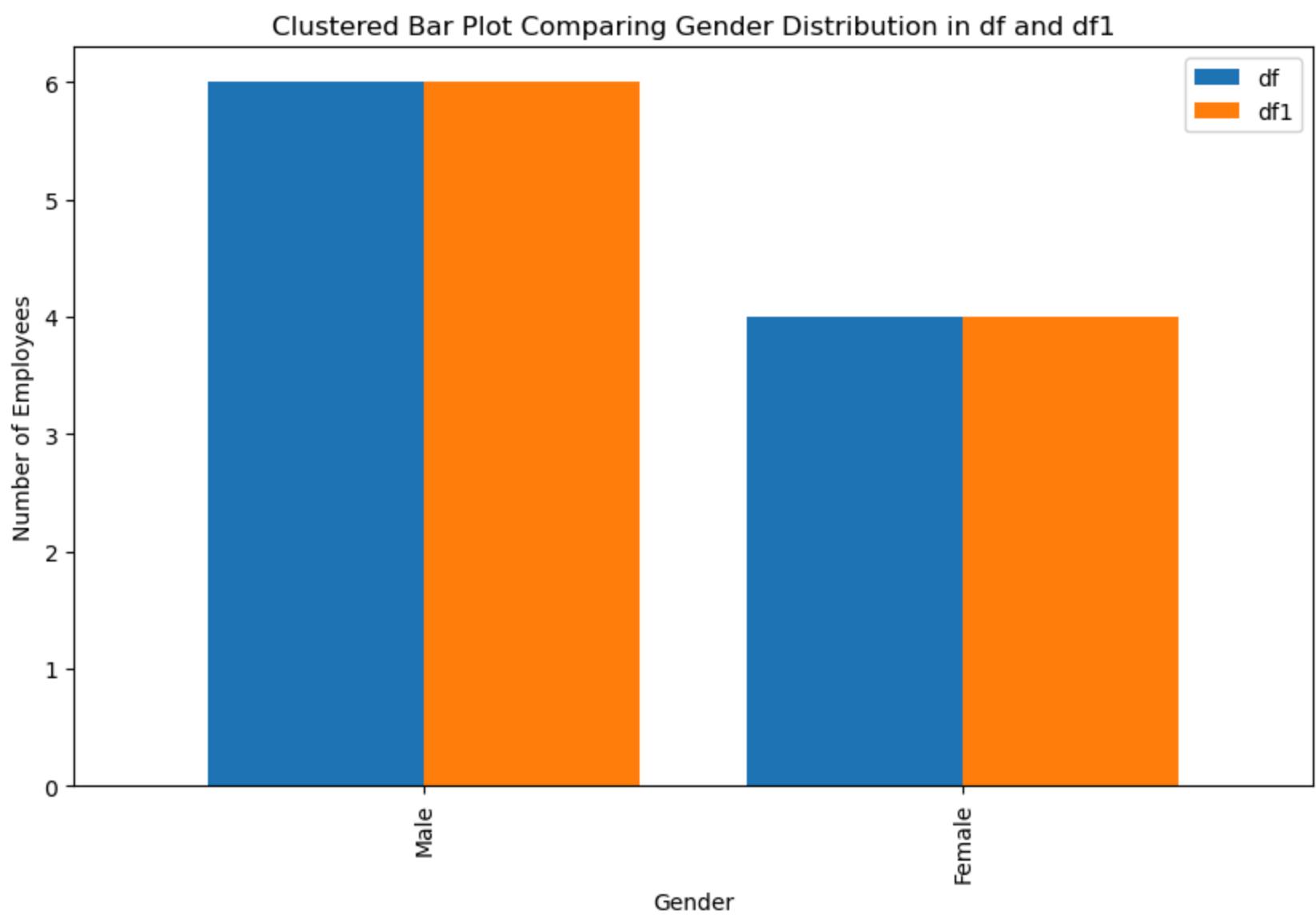




```
In [37]: # Group by Gender and count in both DataFrames
df_gender_counts = df['Gender'].value_counts()
df1_gender_counts = df1['Gender'].value_counts()

# Combine into a single DataFrame
gender_comparison = pd.DataFrame({'df': df_gender_counts, 'df1': df1_gender_counts})

# Plot clustered bar plot
gender_comparison.plot(kind='bar', width=0.8, figsize=(10, 6))
plt.title('Clustered Bar Plot Comparing Gender Distribution in df and df1')
plt.ylabel('Number of Employees')
plt.show()
```



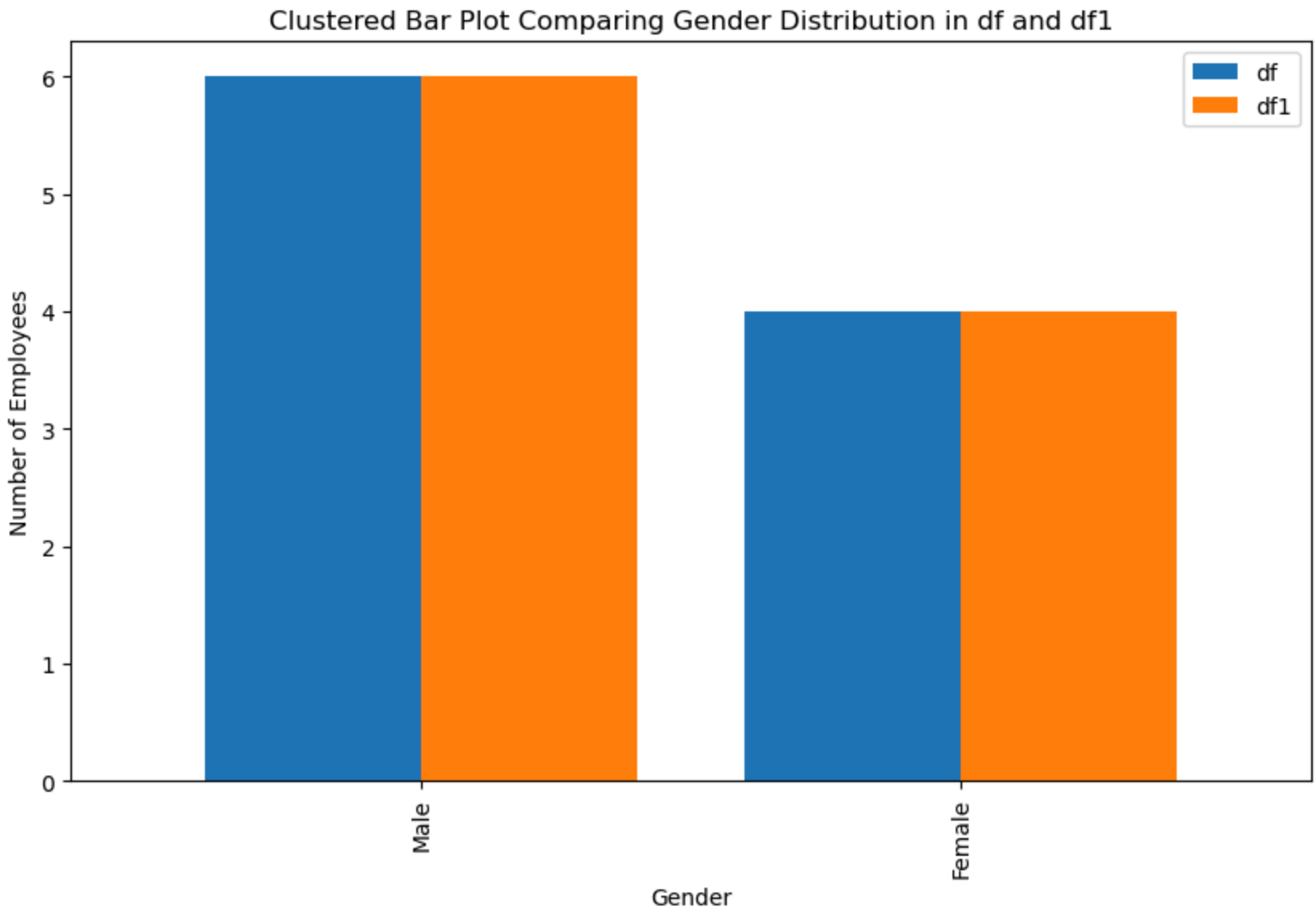
```
In [39]: # Group by Gender and count in both DataFrames
df_gender_counts = df['Gender'].value_counts()
df1_gender_counts = df1['Gender'].value_counts()

# Combine into a single DataFrame
gender_comparison = pd.DataFrame({'df': df_gender_counts, 'df1': df1_gender_counts})

# Plot clustered bar plot
gender_comparison.plot(kind='bar', width=0.8, figsize=(10, 6))
```



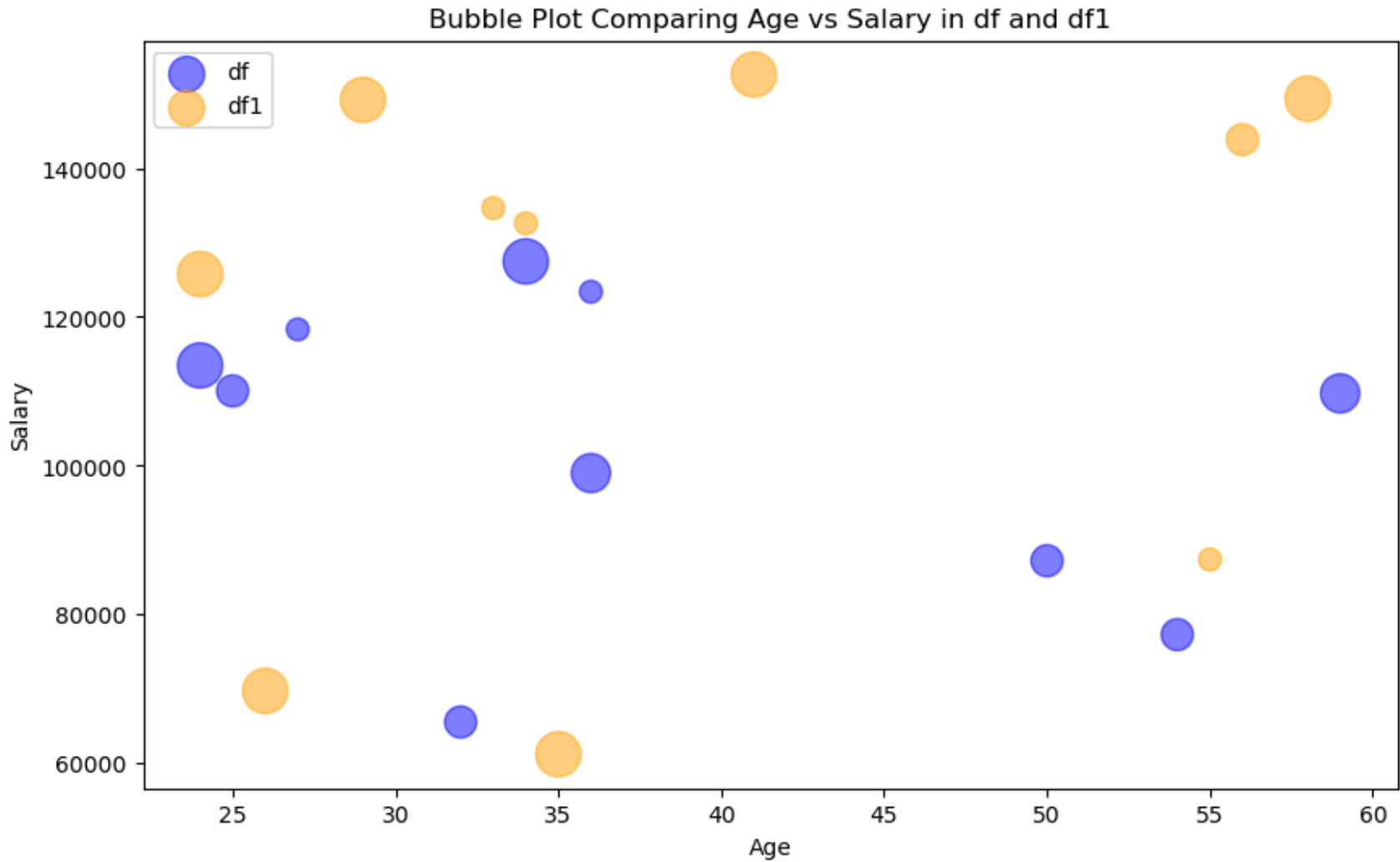
```
plt.title('Clustered Bar Plot Comparing Gender Distribution in df and df1')
plt.ylabel('Number of Employees')
plt.show()
```



```
In [41]: # Bubble plot for Age vs Salary with Performance Score as bubble size
plt.figure(figsize=(10, 6))

plt.scatter(df['Age'], df['Salary'], s=df['Performance Score']*100, alpha=0.5, label='df', color='blue')
plt.scatter(df1['Age'], df1['Salary'], s=df1['Performance Score']*100, alpha=0.5, label='df1', color='orange')

plt.title('Bubble Plot Comparing Age vs Salary in df and df1')
plt.xlabel('Age')
plt.ylabel('Salary')
plt.legend()
plt.show()
```



```
In [43]: # Comparing first employee in both DataFrames using a radar chart
categories = ['Age', 'Salary', 'Years with Company', 'Performance Score', 'Bonus']
df_values = df.loc[0, categories].values.flatten().tolist()
df1_values = df1.loc[0, categories].values.flatten().tolist()

# Closing the radar chart by adding the first value at the end
df_values += df_values[:1]
```

```
df1_values += df1_values[:1]
angles = np.linspace(0, 2 * np.pi, len(categories), endpoint=False).tolist()
angles += angles[:1]

fig, ax = plt.subplots(figsize=(6, 6), subplot_kw=dict(polar=True))
ax.fill(angles, df_values, color='blue', alpha=0.25, label='df')
ax.plot(angles, df_values, color='blue', linewidth=2)

ax.fill(angles, df1_values, color='orange', alpha=0.25, label='df1')
ax.plot(angles, df1_values, color='orange', linewidth=2)

ax.set_yticklabels([])
ax.set_xticks(angles[:-1])
ax.set_xticklabels(categories)
plt.title('Radar Chart Comparing First Employee in df and df1')
plt.legend()
plt.show()
```

