Assignment_Sorting_Filtering_and Handling Missing Data

October 22, 2024

[1]:
```
import pandas as pd
```

[2]:
```
# Initial DataFrame
data = {
    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve'],
    'Age': [25, 30, 22, 35, 28],
    'Salary': [50000, 60000, 45000, 70000, 55000],
    'Department': ['HR', 'Finance', 'IT', 'Finance', 'IT']
}
df = pd.DataFrame(data)
```

[4]:
```
df
```

[4]:

	Name	Age	Salary	Department
0	Alice	25	50000	HR
1	Bob	30	60000	Finance
2	Charlie	22	45000	IT
3	David	35	70000	Finance
4	Eve	28	55000	IT

[5]:
```
# Task 1: Sort the DataFrame by the 'Name' column in ascending order
df_sorted_name = df.sort_values(by='Name', ascending=True)
print("\nSorted by Name (Ascending):\n", df_sorted_name)
```

```
Sorted by Name (Ascending):
       Name  Age  Salary Department
0    Alice   25   50000         HR
1      Bob   30   60000    Finance
2  Charlie   22   45000         IT
3    David   35   70000    Finance
4      Eve   28   55000         IT
```

[6]:
```
# Task 2: Sort the DataFrame by the 'Salary' column in descending order
df_sorted_salary = df.sort_values(by='Salary', ascending=False)
print("\nSorted by Salary (Descending):\n", df_sorted_salary)
```

```
Sorted by Salary (Descending):
```

```
      Name  Age  Salary Department
3    David   35   70000    Finance
1      Bob   30   60000    Finance
4      Eve   28   55000         IT
0    Alice   25   50000         HR
2  Charlie   22   45000         IT
```

[7]:
```
# Task 3: Create a new DataFrame where 'Age' is greater than 25
df_age_gt_25 = df[df['Age'] > 25]
print("\nAge greater than 25:\n", df_age_gt_25)
```

```
Age greater than 25:
     Name  Age  Salary Department
1    Bob   30   60000    Finance
3  David   35   70000    Finance
4    Eve   28   55000         IT
```

[8]:
```
# Task 4: Create a new DataFrame where 'Department' is 'Finance'
df_finance = df[df['Department'] == 'Finance']
print("\nDepartment is Finance:\n", df_finance)
```

```
Department is Finance:
     Name  Age  Salary Department
1    Bob   30   60000    Finance
3  David   35   70000    Finance
```

[9]:
```
# Task 5: Use .where() method to create a new DataFrame where 'Salary' is␣
 ↪greater than 55000, and replace the rest with NaN
df_where_salary = df.where(df['Salary'] > 55000)
print("\nSalary greater than 55000 (with NaN for others):\n", df_where_salary)
```

```
Salary greater than 55000 (with NaN for others):
     Name   Age   Salary Department
0    NaN   NaN      NaN        NaN
1    Bob  30.0  60000.0    Finance
2    NaN   NaN      NaN        NaN
3  David  35.0  70000.0    Finance
4    NaN   NaN      NaN        NaN
```

[10]:
```
# Task 6: Use the .filter() method to filter the columns to include only 'Name'␣
 ↪and 'Department'
df_filtered_columns = df.filter(items=['Name', 'Department'])
print("\nFiltered Columns (Name and Department):\n", df_filtered_columns)
```

```
Filtered Columns (Name and Department):
       Name Department
```

```
0    Alice         HR
1      Bob    Finance
2  Charlie         IT
3    David    Finance
4      Eve         IT
```

[11]:
```
# Task 7: Calculate the mean age of employees in the DataFrame
mean_age = df['Age'].mean()
print("\nMean Age of Employees:", mean_age)
```

```
Mean Age of Employees: 28.0
```

[12]:
```
# Task 8: Calculate the maximum salary in the DataFrame
max_salary = df['Salary'].max()
print("\nMaximum Salary in the DataFrame:", max_salary)
```

```
Maximum Salary in the DataFrame: 70000
```

[13]:
```
# Task 9: Create a DataFrame where any rows with missing values (NaN) in any␣
 ↪column are removed
df_no_nan = df_where_salary.dropna()
print("\nDataFrame after removing rows with NaN values:\n", df_no_nan)
```

```
DataFrame after removing rows with NaN values:
     Name   Age   Salary Department
1    Bob  30.0  60000.0    Finance
3  David  35.0  70000.0    Finance
```

[14]:
```
# Task 10: Fill the missing values in the 'Salary' column with the mean salary␣
 ↪of the remaining employees
mean_salary = df['Salary'].mean()
df_filled_salary = df_where_salary.fillna({'Salary': mean_salary})
print("\nDataFrame after filling missing 'Salary' values with the mean salary:
 ↪\n", df_filled_salary)
```

```
DataFrame after filling missing 'Salary' values with the mean salary:
     Name   Age   Salary Department
0    NaN   NaN  56000.0        NaN
1    Bob  30.0  60000.0    Finance
2    NaN   NaN  56000.0        NaN
3  David  35.0  70000.0    Finance
4    NaN   NaN  56000.0        NaN
```