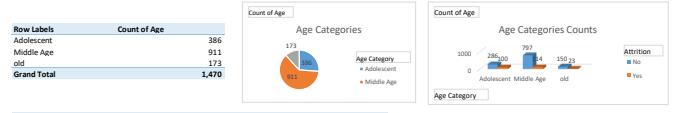
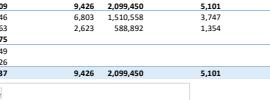
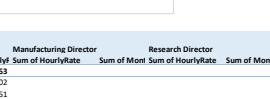
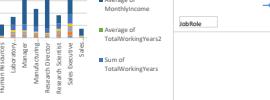
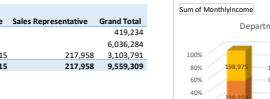
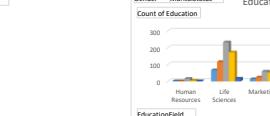
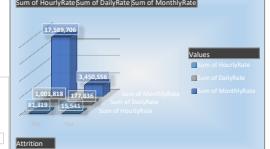
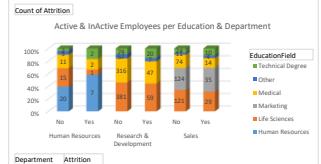


Summaries																												
Summary	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobInvolvement	JobLevel	JobSatisfaction	MonthlyIncome	MonthlyRate	NumCompaniesWorked	PercentSalaryHike	PerformanceRating	RelationshipSatisfaction	StandardHours	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	WorkLifeBalance	YearsAtCompany	YearsInCurrentRole	YearsSinceLastPromotion	YearsWithManager	SparkLine (Line)	SparkLine (Column)
Min	18	102	1	1	1	1	1	30	1	1	1	1,009	2,094	0	11	3	1	80	0	0	0	1	0	0	0	0	▲	----
Max	60	1,499	29	5	1	2,068	4	100	4	5	4	19,999	26,999	9	25	4	4	80	3	40	6	4	40	18	15	17	----	----
Average	37	802	9	3	1	1,025	3	66	3	2	3	6,503	14,313	3	15	3	3	80	1	11	3	3	7	4	2	4	----	----
Count	1,470	1,470	1,470	1,470	1,470	1,470	1,470	1,470	1,470	1,470	1,470	1,470,309	21,040,262	3,959	22,358	4,636	3,987	117,600	1,167	16,581	4,115	4,059	10,302	6,217	3,216	6,061	▲	----
Total	54,278	1,179,654	13,513	4,282	1,470	1,506,552	4,001	96,860	4,013	3,034	4,011	9,559,309	21,040,262	3,959	22,358	4,636	3,987	117,600	1,167	16,581	4,115	4,059	10,302	6,217	3,216	6,061	----	----



Count of Age

Row Labels	Column Labels	No	Yes	Grand Total
Adolescent	No	286	100	386
Middle Age	No	797	114	911
Old	No	150	23	173
Grand Total		1,233	237	1,470







HR Employee Dashboard



1,470

Employees Count

1233

Count of Active Emp

237

Count of InActive

36.92

Average of Age

6.83K

Average of MonthlyIncome

20K

Max of MonthlyIncome

1,051

Min of MonthlyIncome

1,499

Max of DailyRate

102

Min of DailyRate

802.49

Average of DailyRate

100

Max of HourlyRate

30

Min of HourlyRate

65.95

Average of HourlyRate

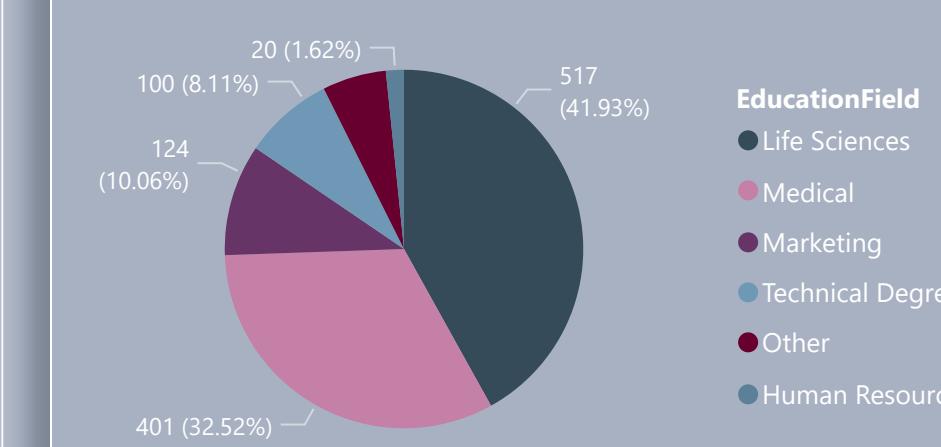
38

Max of TotalWorkingYears

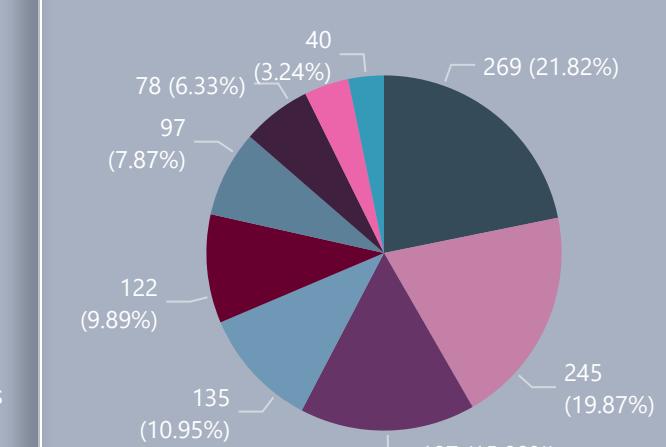
11.86

Average of TotalWorkingYears

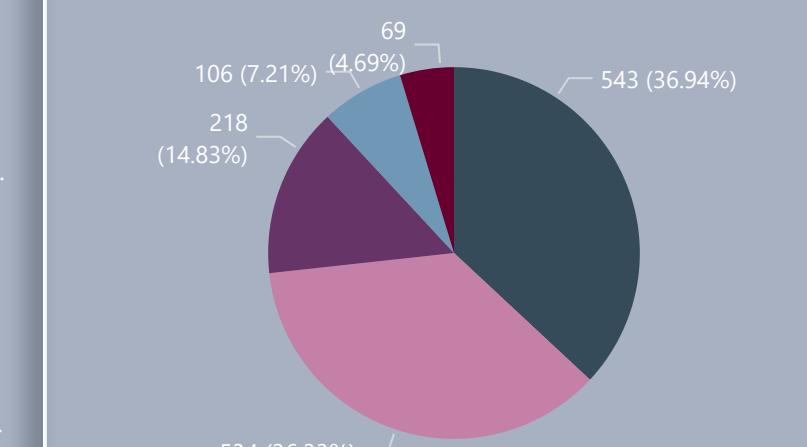
Employee Active per Education



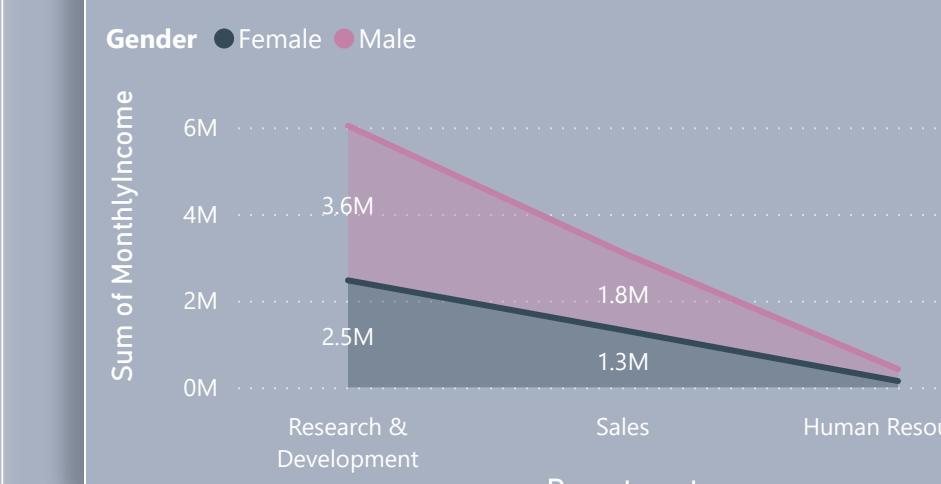
Employee Count per Job Role



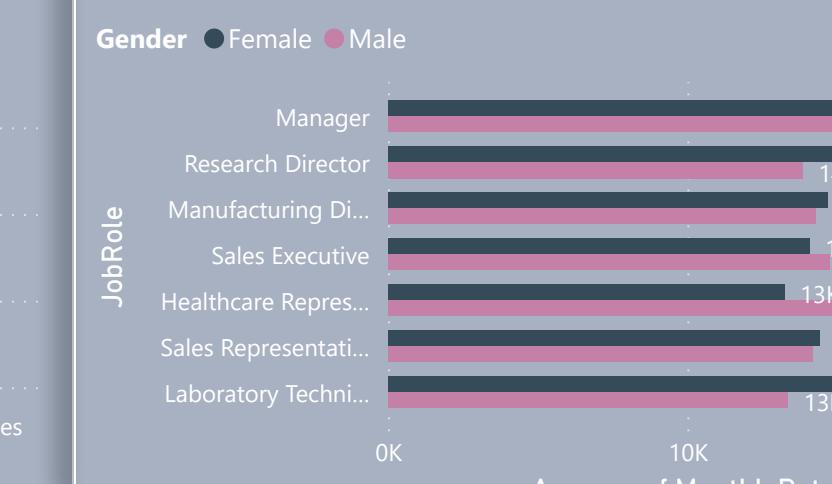
Employee Count per Job Level



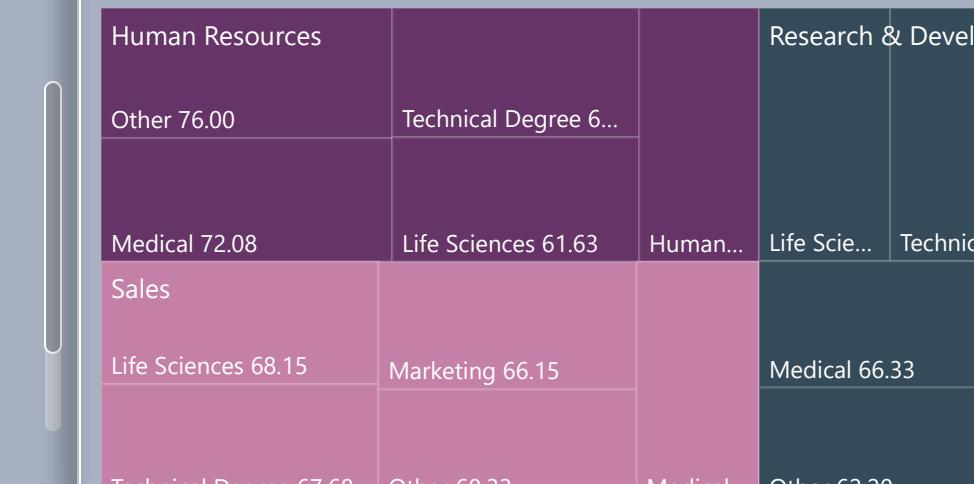
Sum of monthly income per department and gender



Avg Monthly Rate per job Role and Gender



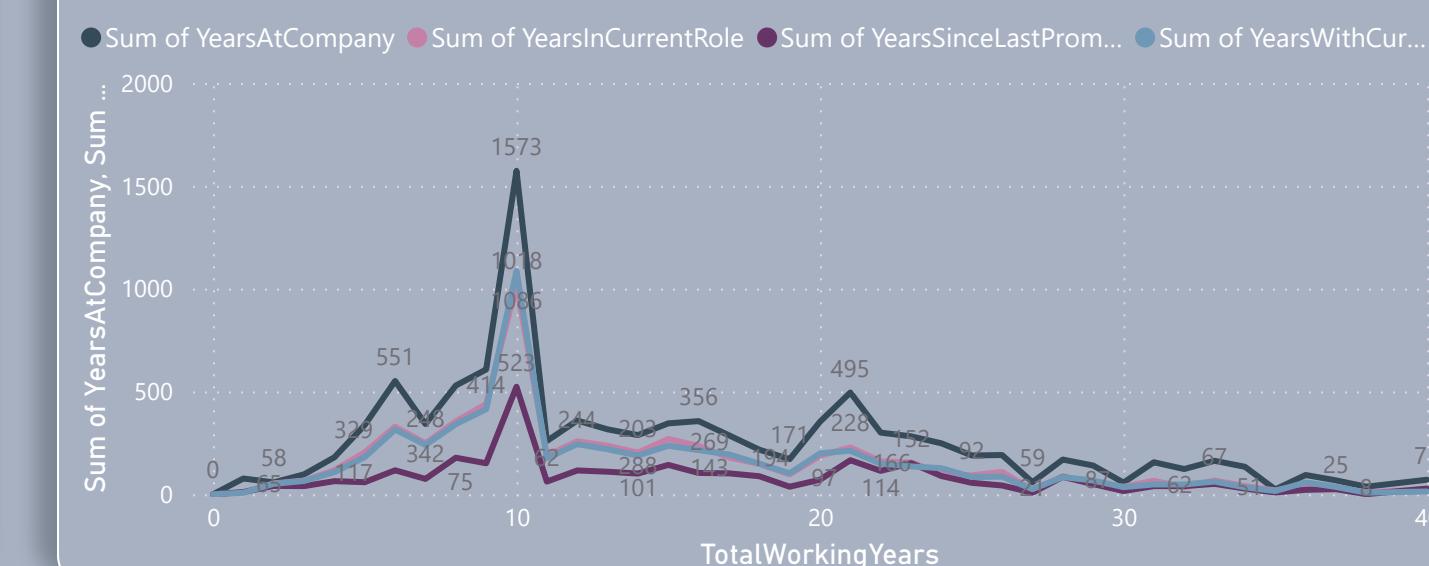
Avg Hourly Rate per department and Education



Department

Department	Average of JobSatisfaction	Average of PercentSalaryHike	Average of PerformanceRating	Average of RelationshipSatisfacti...
Human Resources	2.60	14.76	3.14	2.89
Human Resources	2.56	14.81	3.13	2.96
Manager	2.82	14.55	3.18	2.55
Research & Development	2.73	15.29	3.16	2.71
Healthcare Representative	2.79	15.45	3.15	2.69
Laboratory Technician	2.69	15.05	3.16	2.69
Manager	2.65	14.93	3.19	2.85
Manufacturing Director	2.68	15.59	3.19	2.72
Research Director	2.70	14.95	3.10	2.69
Total	2.73	15.21	3.15	2.71

Year In Service



At 1573, 10 had the highest Sum of YearsAtCompany and was Infinity higher than 0, which had the lowest Sum of YearsAtCompany at 0.

Sum of YearsAtCompany and total Sum of YearsInCurrentRole are positively correlated with each other.

10 accounted for 15.27% of Sum of YearsAtCompany.

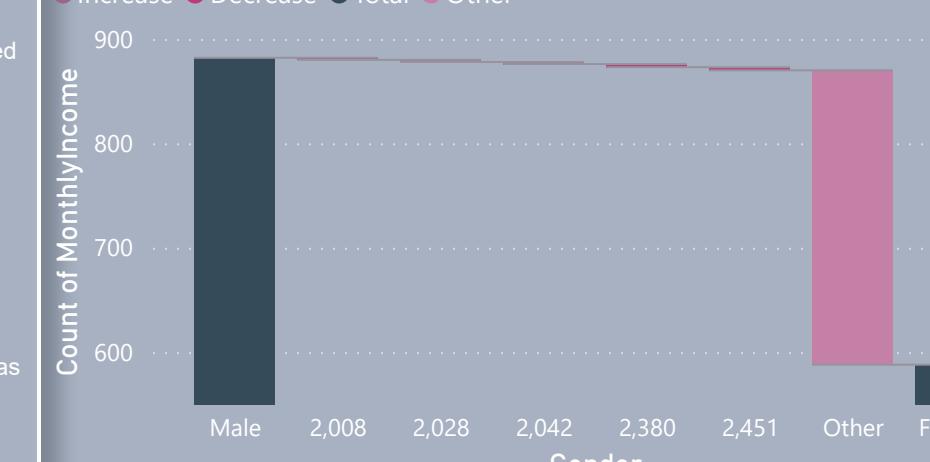
Total Sum of MonthlyIncome was higher for Male (5627608) than Female (3931701).

Research & Development in Gender made up 37.32% of Sum of MonthlyIncome.

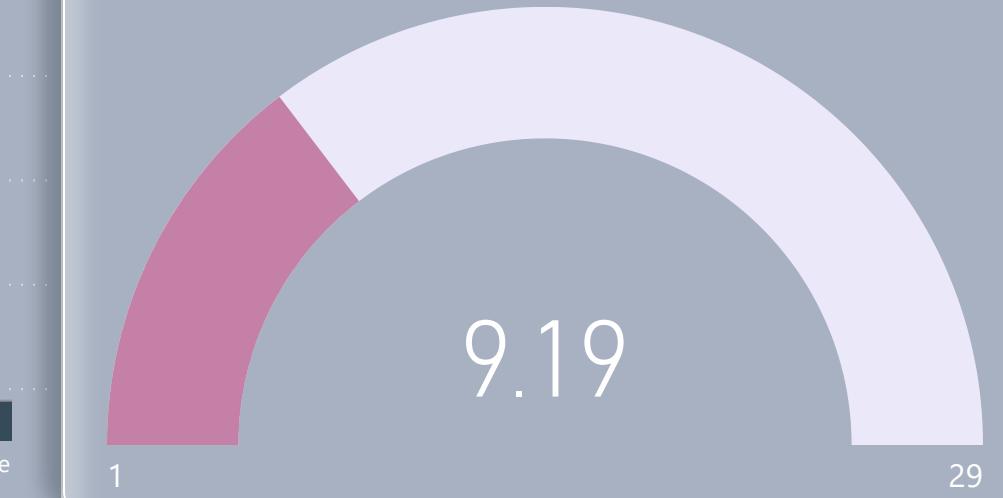
Average Sum of MonthlyIncome was higher for Male (1,875,869.33) than Female (1,310,567).

Average of MonthlyRate for Female and Male diverged the most when the JobRole was Research Director, when Female were 2,987.48 higher than Male.

Count of MonthlyIncome by Gender and MonthlyIncome



Distance from Home



Department

- (Blank)
- Human Resources
- Research & Development
- Sales

Attrition

- (Blank)
- No
- Yes

Gender

- (Blank)
- Female
- Male

MaritalStatus

- (Blank)
- Divorced
- Married

EducationField

- (Blank)
- Human Resources
- Life Sciences
- Marketing
- Medical
- Other
- Technical Degree

JobRole

- (Blank)
- Healthcare Representative
- Human Resources
- Laboratory Technician
- Manager
- Manufacturing Director
- Research Director
- Research Scientist
- Sales Executive
- Sales Representative



▼ HR Employees Analysis

In this file we'll be analyzing HR Employees information

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

%matplotlib inline
import plotly.figure_factory as ff
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
```

▼ Reading data:

```
pd.read_excel
```

```
<function pandas.io.xlsx._base.read_excel(io, sheet_name: 'str | int | list[IntStrT] | None' = 0, *, header: 'int | Sequence[int] | None' = 0, names: 'list[str] | None' = None, index_col: 'int | Sequence[int] | None' = None, usecols: 'int | str | Sequence[int] | Sequence[str] | Callable[[str], bool] | None' = None, dtype: 'DtypeArg | None' = None, engine: 'Literal['xlrd', 'openpyxl', 'odf', 'pyxlsb'] | None' = None, converters: 'dict[str, Callable] | dict[int, Callable] | None' = None, true_values: 'Iterable[Hashable] | None' = None, false_values: 'Iterable[Hashable] | None' = None, skiprows: 'Sequence[int] | int | Callable[[int], object] | None' = None, nrows: 'int | None' = None, na_values=None, keep_default_na: 'bool' = True, na_filter: 'bool' = True, verbose: 'bool' = False, parse_dates: 'list | dict | bool' = False, date_parser: 'Callable | lib.NoDefault' = <no_default>, date_format: 'dict[Hashable, str] | str | None' = None, thousands: 'str | None' = None, decimal: 'str' = '.', comment: 'str | None' = None, skipfooter: 'int' = 0, storage_options: 'StorageOptions' = None, dtype_backend: 'DtypeBackend | lib.NoDefault' = <no_default>) -> 'DataFrame | dict[IntStrT, DataFrame]'>
```

```
hr_df = pd.read_excel('E:\MeriSkillInternship\Project 3 - HR Analytics\HR-Employee-Attrition_python.xlsx')
```

```
hr_df.head()
```

	Age	Age Category	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	...	Re
0	41	Middle Age	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	...	
1	49	Middle Age	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	...	
2	37	Middle Age	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	...	
3	33	Middle Age	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	...	
4	27	Adolescent	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	...	

5 rows × 36 columns

```
hr_df.shape
```

```
(1470, 36)
```

```
hr_df.tail(5)
```

	Age	Age Category	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	...
1465	36	Middle Age	No	Travel_Frequently	884	Research & Development	23	2	Medical	1	...
1466	39	Middle Age	No	Travel_Rarely	613	Research & Development	6	1	Medical	1	...
1467	27	Adolescent	No	Travel_Rarely	155	Research & Development	4	3	Life Sciences	1	...
1468	49	Middle Age	No	Travel_Frequently	1023	Sales	2	3	Medical	1	...
1469	34	Middle Age	No	Travel_Rarely	628	Research & Development	8	3	Medical	1	...

5 rows × 36 columns

```
hr_df.head()
```

	Age	Age Category	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	...	Re
0	41	Middle Age	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	...	
1	49	Middle Age	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	...	
2	37	Middle Age	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	...	
3	33	Middle Age	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	...	
4	27	Adolescent	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	...	

5 rows × 36 columns

```
hr_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 36 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Age              1470 non-null    int64  
 1   Age Category     1470 non-null    object  
 2   Attrition        1470 non-null    object  
 3   BusinessTravel   1470 non-null    object  
 4   DailyRate        1470 non-null    int64  
 5   Department       1470 non-null    object  
 6   DistanceFromHome 1470 non-null    int64  
 7   Education        1470 non-null    int64  
 8   EducationField   1470 non-null    object  
 9   EmployeeCount    1470 non-null    int64  
 10  EmployeeNumber   1470 non-null    int64  
 11  EnvironmentSatisfaction 1470 non-null    int64  
 12  Gender            1470 non-null    object  
 13  HourlyRate        1470 non-null    int64  
 14  JobInvolvement   1470 non-null    int64  
 15  JobLevel          1470 non-null    int64  
 16  JobRole           1470 non-null    object  
 17  JobSatisfaction  1470 non-null    int64  
 18  MaritalStatus     1470 non-null    object  
 19  MonthlyIncome     1470 non-null    int64  
 20  MonthlyRate       1470 non-null    int64  
 21  NumCompaniesWorked 1470 non-null    int64  
 22  Over18            1470 non-null    object  
 23  Overtime          1470 non-null    object  
 24  PercentSalaryHike 1470 non-null    int64  
 25  PerformanceRating 1470 non-null    int64  
 26  RelationshipSatisfaction 1470 non-null    int64  
 27  StandardHours     1470 non-null    int64  
 28  StockOptionLevel  1470 non-null    int64  
 29  TotalWorkingYears 1470 non-null    int64  
 30  TrainingTimesLastYear 1470 non-null    int64
```

```

31 WorkLifeBalance      1470 non-null  int64
32 YearsAtCompany      1470 non-null  int64
33 YearsInCurrentRole  1470 non-null  int64
34 YearsSinceLastPromotion 1470 non-null  int64
35 YearsWithCurrManager 1470 non-null  int64
dtypes: int64(26), object(10)
memory usage: 413.6+ KB

```

```
hr_df.describe()
```

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobInv
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	1470.000000	1470.000000	1470.000000	14
mean	36.923810	802.485714	9.192517	2.912925	1.0	1024.865306	2.721769	65.891156	
std	9.135373	403.509100	8.106864	1.024165	0.0	602.024335	1.093082	20.329428	
min	18.000000	102.000000	1.000000	1.000000	1.0	1.000000	1.000000	30.000000	
25%	30.000000	465.000000	2.000000	2.000000	1.0	491.250000	2.000000	48.000000	
50%	36.000000	802.000000	7.000000	3.000000	1.0	1020.500000	3.000000	66.000000	
75%	43.000000	1157.000000	14.000000	4.000000	1.0	1555.750000	4.000000	83.750000	
max	60.000000	1499.000000	29.000000	5.000000	1.0	2068.000000	4.000000	100.000000	

8 rows × 26 columns

```
hr_df.columns
```

```

Index(['Age', 'Age Category', 'Attrition', 'BusinessTravel', 'DailyRate',
       'Department', 'DistanceFromHome', 'Education', 'EducationField',
       'EmployeeCount', 'EmployeeNumber', 'EnvironmentSatisfaction', 'Gender',
       'HourlyRate', 'JobInvolvement', 'JobLevel', 'JobRole',
       'JobSatisfaction', 'MaritalStatus', 'MonthlyIncome', 'MonthlyRate',
       'NumCompaniesWorked', 'Over18', 'OverTime', 'PercentSalaryHike',
       'PerformanceRating', 'RelationshipSatisfaction', 'StandardHours',
       'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear',
       'WorkLifeBalance', 'YearsAtCompany', 'YearsInCurrentRole',
       'YearsSinceLastPromotion', 'YearsWithCurrManager'],
      dtype='object')

```

▼ Check Null Values

```
hr_df.isna().sum()
```

Age	0
Age Category	0
Attrition	0
BusinessTravel	0
DailyRate	0
Department	0
DistanceFromHome	0
Education	0
EducationField	0
EmployeeCount	0
EmployeeNumber	0
EnvironmentSatisfaction	0
Gender	0
HourlyRate	0
JobInvolvement	0
JobLevel	0
JobRole	0
JobSatisfaction	0
MaritalStatus	0
MonthlyIncome	0
MonthlyRate	0
NumCompaniesWorked	0
Over18	0
OverTime	0
PercentSalaryHike	0
PerformanceRating	0
RelationshipSatisfaction	0
StandardHours	0
StockOptionLevel	0
TotalWorkingYears	0
TrainingTimesLastYear	0

```
WorkLifeBalance      0
YearsAtCompany      0
YearsInCurrentRole  0
YearsSinceLastPromotion 0
YearsWithCurrManager 0
dtype: int64
```

▼ Check Duplicate Values

```
hr_df.duplicated().sum()

0

#mapping non-numeric values to numeric values
hr_df_Nu = hr_df.copy()
YN_mapping = {"Yes": 1, "No": 0}
hr_df_Nu['Attrition'] = hr_df_Nu['Attrition'].map(YN_mapping)
hr_df_Nu['OverTime'] = hr_df_Nu['OverTime'].map(YN_mapping)
BusinessTravel_mapping = {"Travel_Rarely": 0, "Travel_Frequently": 1, "Non-Travel": 0}
hr_df_Nu['BusinessTravel'] = hr_df_Nu['BusinessTravel'].map(BusinessTravel_mapping)
Gender_mapping = {"Female": 1, "Male": 0}
hr_df_Nu['Gender'] = hr_df_Nu['Gender'].map(Gender_mapping)
Marital_mapping = {"Married": 1, "Single": 0, "Divorced": 0}
hr_df_Nu['MaritalStatus'] = hr_df_Nu['MaritalStatus'].map(Marital_mapping)
```

```
hr_df_Nu.head(3)
```

	Age	Age Category	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	...	Rela
0	41	Middle Age	1	0	1102	Sales	1	2	Life Sciences	1	...	
1	49	Middle Age	0	1	279	Research & Development	8	1	Life Sciences	1	...	
2	37	Middle Age	1	0	1373	Research & Development	2	2	Other	1	...	

3 rows × 36 columns

▼ Numerical analysis and visualization

We'll analyze the `MonthlyIncome` column:

```
hr_df['MonthlyIncome'].describe()
```

```
count    1470.000000
mean     6502.931293
std      4707.956783
min     1009.000000
25%    2911.000000
50%    4919.000000
75%    8379.000000
max    19999.000000
Name: MonthlyIncome, dtype: float64
```

```
hr_df['MonthlyIncome'].mean()
```

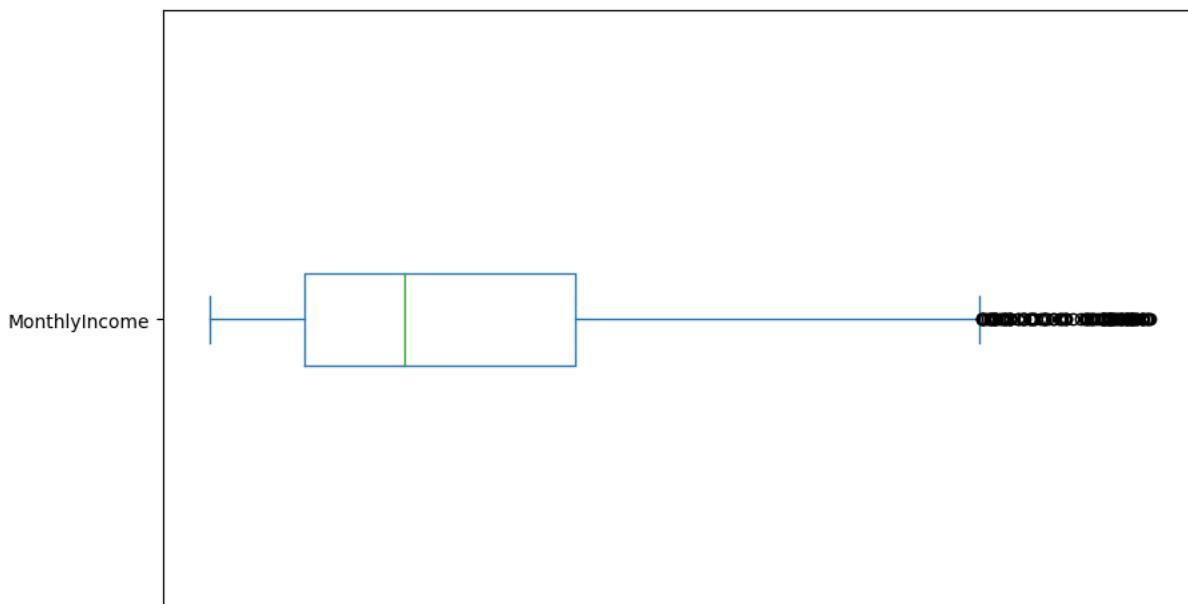
6502.931292517007

```
hr_df['MonthlyIncome'].median()
```

4919.0

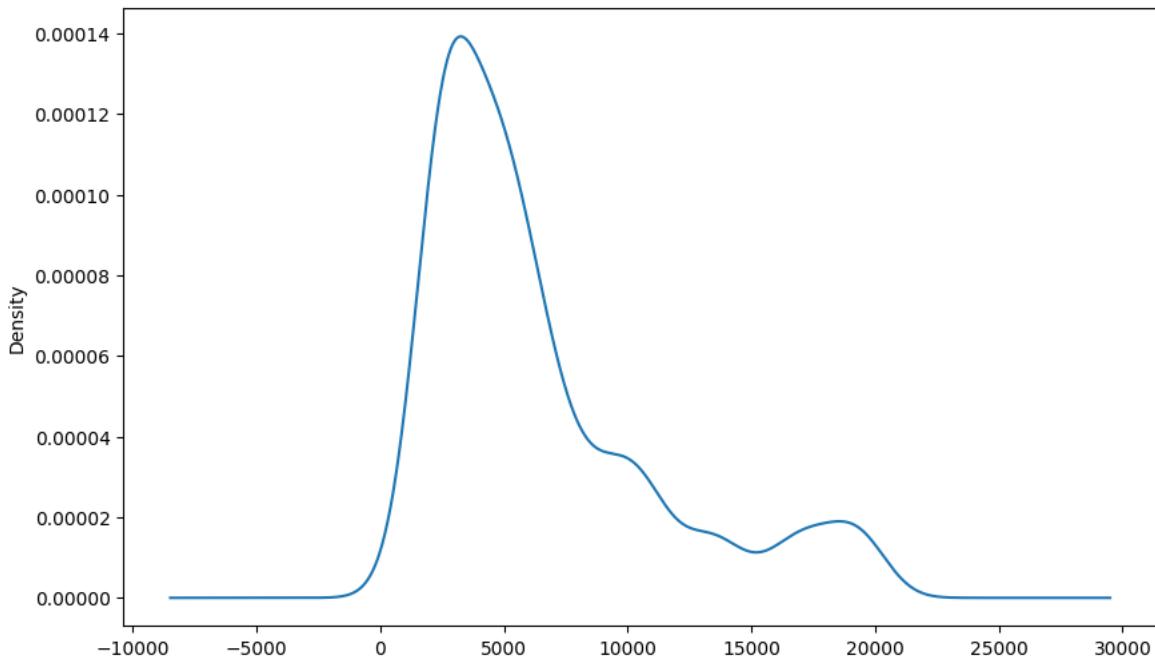
```
hr_df['MonthlyIncome'].plot(kind='box', vert=False, figsize=(10,6))
```

<Axes: >

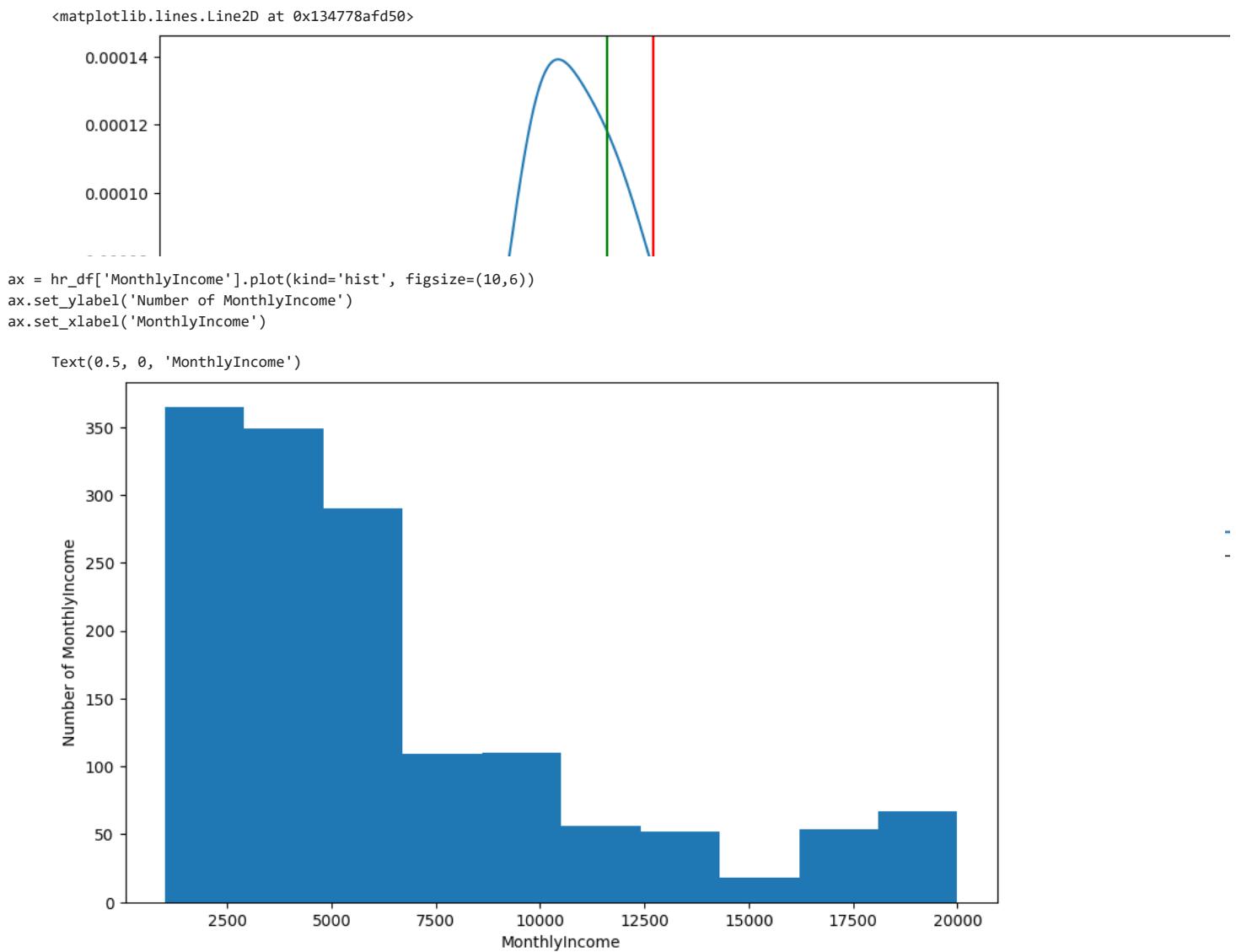


```
hr_df['MonthlyIncome'].plot(kind='density', figsize=(10,6))
```

<Axes: ylabel='Density'>



```
ax = hr_df['MonthlyIncome'].plot(kind='density', figsize=(14,6)) # kde  
ax.axvline(hr_df['MonthlyIncome'].mean(), color='red')  
ax.axvline(hr_df['MonthlyIncome'].median(), color='green')
```



▼ Categorical analysis and visualization

We'll analyze the `AgeCategory` column:

```
hr_df.head()
```

	Age	Age Category	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	...	Re
0	41	Middle Age	Yes	Travel_Rarely	1102	Sales		1	2	Life Sciences	1	...
1	49	Middle Age	No	Travel_Frequently	279	Research & Development		8	1	Life Sciences	1	...
2	37	Middle Age	Yes	Travel_Rarely	1373	Research & Development		2	2	Other	1	...
3	33	Middle Age	No	Travel_Frequently	1392	Research & Development		3	4	Life Sciences	1	...
4	27	Adolescent	No	Travel_Rarely	591	Research & Development		2	1	Medical	1	...

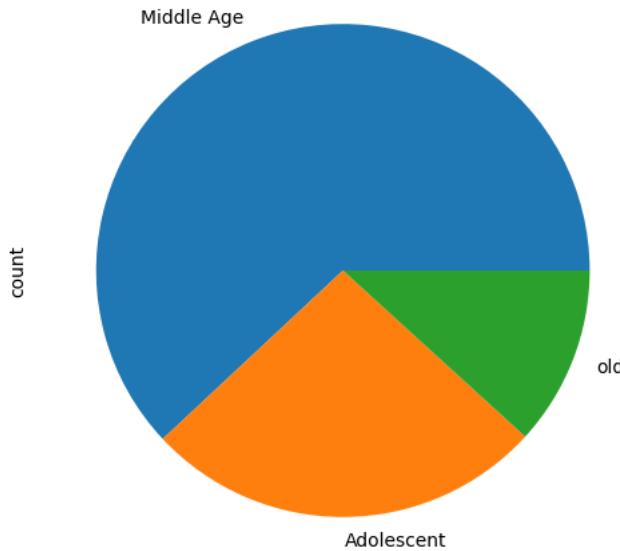
5 rows × 36 columns

```
hr_df['Age Category'].value_counts()
```

```
Age Category
Middle Age    911
Adolescent     386
old            173
Name: count, dtype: int64
```

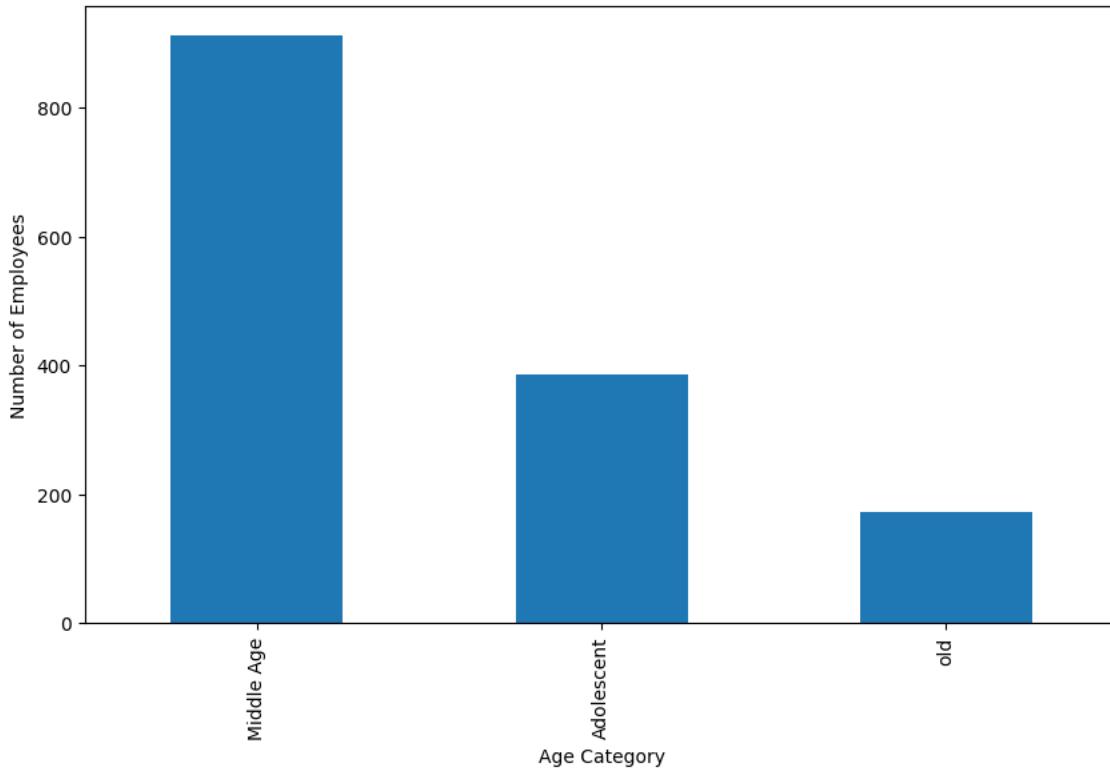
```
hr_df['Age Category'].value_counts().plot(kind='pie', figsize=(6,6))
```

```
<Axes: ylabel='count'>
```



```
ax = hr_df['Age Category'].value_counts().plot(kind='bar', figsize=(10,6))
ax.set_ylabel('Number of Employees')
```

```
Text(0, 0.5, 'Number of Employees')
```



Age Distribution

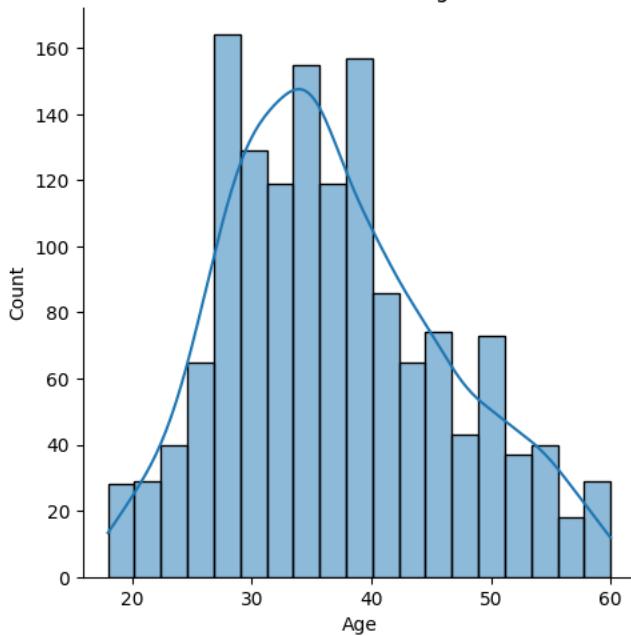
```
sns.displot(hr_df['Age'], kde=True)
n1+ i1/a('Distribution of Age')
```

<https://colab.research.google.com/drive/1PWpWLvR3H-XTJxnKapwAisA9-Zc75c7#scrollTo=rE0eXkR4kCuz&printMode=true>

```
plt.figure(figsize=(10, 6))  
plt.show()
```

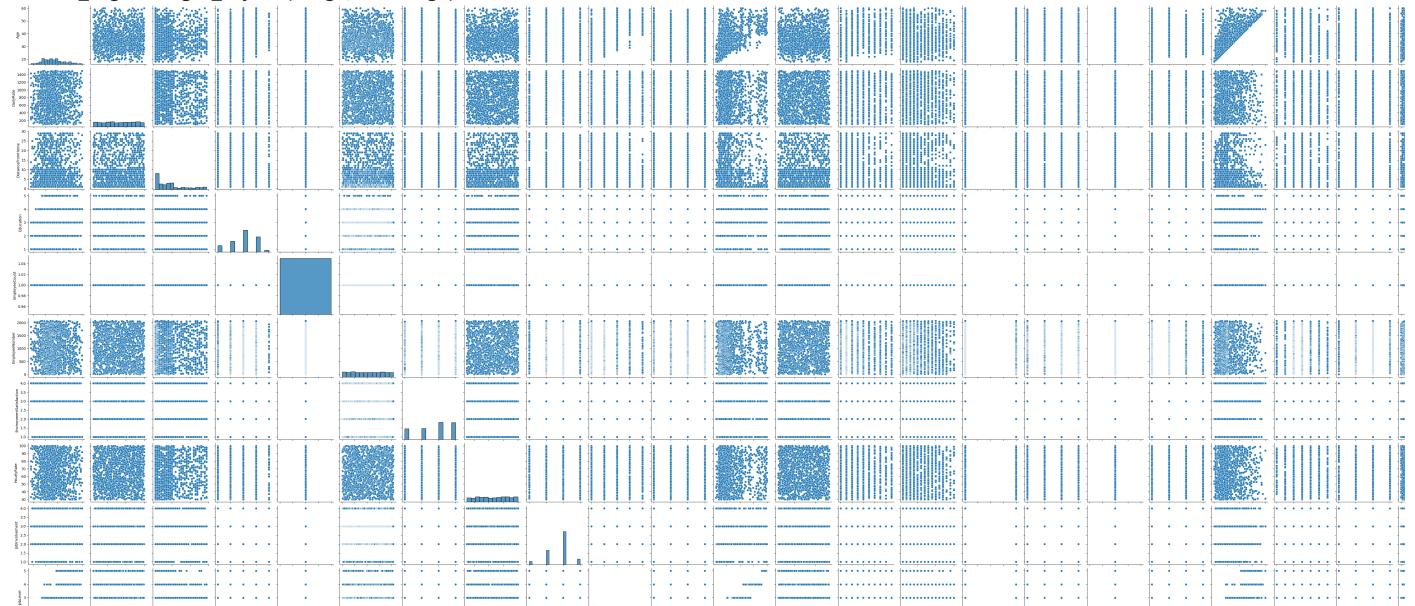
```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed.  
self._figure.tight_layout(*args, **kwargs)
```

Distribution of Age

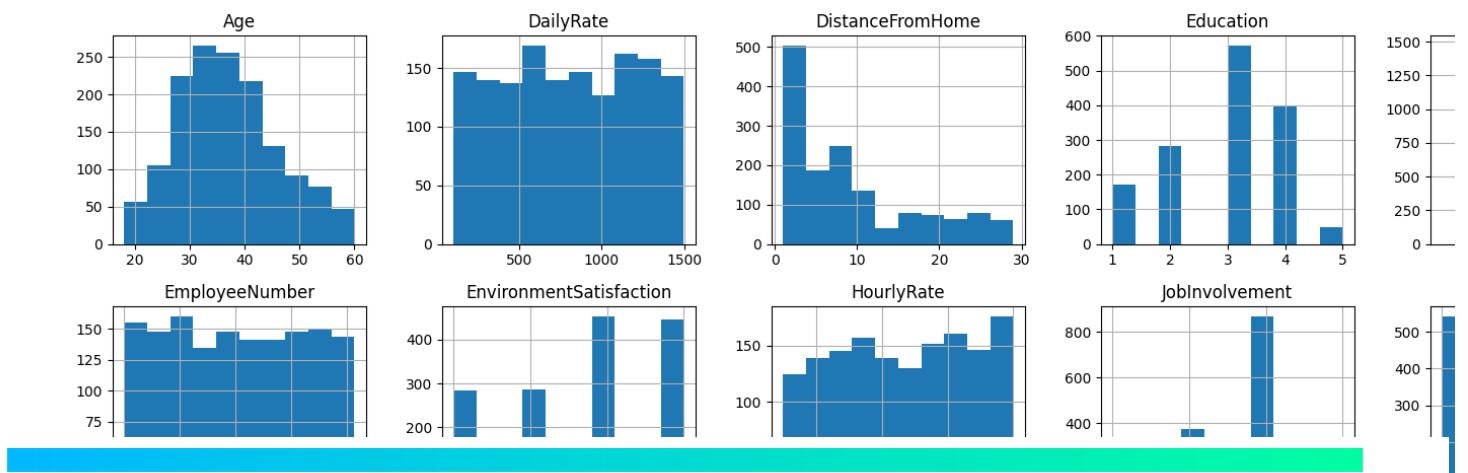


```
sns.pairplot(hr_df)  
plt.show()
```

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed!
```



```
hr_df.hist(figsize=(20,20))  
plt.show()
```



▼ Relationship between the columns?

We will find any significant relationship

```
front = hr_df['Attrition']
hr_df.drop(labels=['Attrition'], axis=1, inplace = True)
hr_df.insert(0, 'Attrition', front)
hr_df.head()
```

	Attrition	Age	Age Category	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	...	Re
0	Yes	41	Middle Age	Travel_Rarely	1102	Sales		1	2	Life Sciences	1	...
1	No	49	Middle Age	Travel_Frequently	279	Research & Development		8	1	Life Sciences	1	...
2	Yes	37	Middle Age	Travel_Rarely	1373	Research & Development		2	2	Other	1	...
3	No	33	Middle Age	Travel_Frequently	1392	Research & Development		3	4	Life Sciences	1	...
4	No	27	Adolescent	Travel_Rarely	591	Research & Development		2	1	Medical	1	...

5 rows × 36 columns

```
attrition_rate = hr_df.Attrition.value_counts() / 1470
attrition_rate
```

```
Attrition
No    0.838776
Yes   0.161224
Name: count, dtype: float64
```

```
AttritionSummary= hr_df.groupby('Attrition')
AttritionSummary
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x000001340B301090>
```

```
AttritionSummary.max()
```

```

cat_cols=hr_df.select_dtypes(include=object).columns.tolist()
cat_df=pd.DataFrame(hr_df[cat_cols].melt(var_name='column', value_name='value')
                     .value_counts().rename(columns={0: 'count'}).sort_values(by=['column', 'count']))
display(hr_df.select_dtypes(include=object).describe())
display(cat_df)

```

	Attrition	Age	Category	BusinessTravel	Department	EducationField	Gender	JobRole	MaritalStatus	Over18	Over
count	1470	1470		1470	1470	1470	1470	1470	1470	1470)
unique	2	3		3	3	6	2	9	3	1	
top	No	Middle Age	Travel_Rarely	Research & Development	Life Sciences	Male	Sales Executive	Married	Y		
freq	1233	911		1043	961	606	882	326	673	1470	
count											
	column		value								
Age Category		old	173								
		Adolescent	386								
		Middle Age	911								
Attrition		Yes	237								
		No	1233								
BusinessTravel		Non-Travel	150								
		Travel_Frequently	277								
		Travel_Rarely	1043								
Department		Human Resources	63								
		Sales	446								
		Research & Development	961								
EducationField		Human Resources	27								
		Other	82								
		Technical Degree	132								
		Marketing	159								
		Medical	464								
		Life Sciences	606								
Gender		Female	588								
		Male	882								
JobRole		Human Resources	52								
		Research Director	80								
		Sales Representative	83								
		Manager	102								
		Healthcare Representative	131								
		Manufacturing Director	145								

```

fig=make_subplots(rows=1, cols=2,
                  subplot_titles=("Employee Attrition by Department"),
                  specs=[[{"type": "bar"}, {"type": "pie"}]])

# Bar chart
plot_df=hr_df['Attrition'].value_counts(normalize=True)
plot_df=plot_df.mul(100).rename('Percent').reset_index().sort_values('Percent')
plot_df.rename(columns={'index':'Attrition'}, inplace=True)
plot_df['Attrition']=['Former Employees' if i == 'Yes' else 'Current Employees' for i in plot_df['Attrition']]
x=plot_df['Attrition']
y=plot_df['Percent']
fig.add_trace(
    go.Bar(x=x, y=y, text=y, opacity=.8,
           hovertemplate='Employee Attrition Rate<br>%{x}: %{y:.3}%<extra></extra>',
           showlegend=False), row=1, col=1)
fig.update_traces(texttemplate='%{text:.3s}', textposition='outside',
                   marker_line=dict(width=1, color='#F02020'), marker_color=['#C02B34', '#CDBBA7'])
fig.update_yaxes(zeroline=True, zerolinewidth=1, zerolinecolor='gray')
fig.update_layout(yaxis_ticksuffix = '%')

# Pie chart
plot_df2=hr_df[hr_df['Attrition']=='Yes']
plot_df2=plot_df2['Department'].value_counts(normalize=True)
plot_df2=plot_df2.mul(100).rename('Percent').reset_index().sort_values('Percent', ascending=False)
plot_df2.rename(columns={'index':'Department'}, inplace=True)
fig.add_trace(go.Pie(labels=plot_df2['Department'], values=plot_df2['Percent'], opacity=0.85, hole=0.4,
                      hovertemplate='%{label}<br>Attrition Rate: %{value:.3}%<extra></extra>',
                      marker_colors=['#587D65', '#ADC4B2', '#D1C9C2']), row=1, col=2)
fig.update_yaxes(tickmode = 'array', range=[0, 90], dtick=5)
fig.update_traces(textfont_size=14, textfont_color='black', marker=dict(line=dict(color='#28221D', width=1)))
fig.update_layout(title_text="Employee Attrition Statistics", font_color="#28221D",
                  paper_bgcolor="#F4F2F0", plot_bgcolor="#F4F2F0")
fig.show()

```

```

plot_df = hr_df.groupby(['WorkLifeBalance', 'Gender'])['Attrition'].value_counts(normalize=True)
plot_df = plot_df.mul(100).rename('Percent').reset_index()
fig = px.bar(plot_df, x='WorkLifeBalance', y='Percent', color='Attrition',
             facet_row='Gender', text='Percent', opacity=0.75, barmode='group',
             category_orders={'Attrition': ['Yes', 'No']},
             color_discrete_map={'Yes': '#C02B34', 'No': '#CDBBA7'})
fig.update_traces(texttemplate='%{text:.2s}', textposition='outside',
                  marker_line=dict(width=1, color='#28221D'))
fig.update_yaxes(title="", zeroline=True, zerolinewidth=1, zerolinecolor="#F02020", ticksuffix = '%')
fig.update_layout(title_text='Attrition Rates by Work Life Balance and Gender', height=750, font_color="#28221D",
                  xaxis_title='Work Life Balance', paper_bgcolor="#F4F2F0", plot_bgcolor="#F4F2F0",
                  xaxis = dict(tickmode = 'array', tickvals = [1, 2, 3, 4],
                               ticktext = ['Poor', 'Neutral', 'Good', 'Excellent']))
fig.show()

```

```

plot_df = hr_df.groupby(['Gender', 'Department'])['Attrition'].value_counts(normalize=True)
plot_df = plot_df.mul(100).rename('Percent').reset_index()
fig = px.bar(plot_df, x="Department", y="Percent", color="Attrition", barmode="group",
             text='Percent', opacity=.75, facet_col="Gender", category_orders={'Attrition': ['Yes', 'No']},
             color_discrete_map={'Yes': '#C02B34', 'No': '#CDBBA7'})
fig.update_traces(texttemplate='%{text:.3s}', textposition='outside',
                  marker_line=dict(width=1, color='#28221D'), width=.4)
fig.update_layout(title_text='Attrition Rates by Department and Gender', yaxis_ticksuffix = '%',
                  paper_bgcolor="#F4F2F0", plot_bgcolor="#F4F2F0", font_color="#28221D",
                  height=500, xaxis=dict(tickangle=30))
fig.update_xaxes(showticklabels=True, tickangle=30, col=2)
fig.update_yaxes(title = "", zeroline=True, zerolinewidth=1, zerolinecolor="#28221D")
fig.show()

```

```

plot_df = hr_df.groupby(['Attrition'])['JobSatisfaction'].value_counts(normalize=True)
plot_df = plot_df.mul(100).rename('Percent').reset_index().sort_values('JobSatisfaction')
plot_df.JobSatisfaction=pd.Categorical(plot_df.JobSatisfaction).rename_categories(
    {1:'Poor', 2:'Neutral', 3:'Good', 4:'Excellent'})
fig = px.bar(plot_df, x='JobSatisfaction', y='Percent', text='Percent', opacity=0.8,
             facet_col="Attrition", category_orders={"Attrition": ["Yes", "No"]})
fig.update_traces(texttemplate='%{text:.2s}', textposition='outside',

```

```

marker_color=['#B6735F', '#D7C2B0', '#497B7A', '#9EB5A3'],
marker_line=dict(width=1, color='#28221D'))
fig.update_yaxes(title="", zeroline=True, zerolinewidth=1, zerolinecolor="#28221D", ticksuffix='%')
fig.update_layout(title_text='Attrition Rates by Job Satisfaction', bargap=.09, font_color="#28221D",
                  xaxis_title='Job Satisfaction', paper_bgcolor='#F4F2F0', plot_bgcolor='#F4F2F0')
fig.show()

plot_df = hr_df.groupby(['Department', 'Gender'])['MonthlyIncome'].mean()
plot_df = plot_df.mul(12).rename('Salary').reset_index().sort_values('Salary', ascending=False)
fig = px.bar(plot_df, x='Department', y='Salary', color='Gender', text='Salary',
             barmode='group', opacity=0.75, color_discrete_map={'Female': '#214D5C', 'Male': '#ACBCC2'})
fig.update_traces(texttemplate='$\{text:,.0f}', textposition='outside',
                   marker_line=dict(width=1, color='#28221D'))
fig.update_yaxes(zeroline=True, zerolinewidth=1, zerolinecolor="#28221D")
fig.update_layout(title_text='Average Salaries by Department & Gender', font_color="#28221D",
                  yaxis=dict(title='Salary', tickprefix='$'), paper_bgcolor='#F4F2F0', plot_bgcolor='#F4F2F0')
fig.show()

plot_df = hr_df.groupby(['Department', 'Attrition', 'Gender'])['MonthlyIncome'].median()
plot_df = plot_df.mul(12).rename('Salary').reset_index().sort_values('Salary', ascending=False).sort_values('Gender')
fig = px.bar(plot_df, x='Department', y='Salary', color='Gender', text='Salary',
             barmode='group', opacity=0.75, color_discrete_map={'Female': '#214D5C', 'Male': '#ACBCC2'},
             facet_col='Attrition', category_orders={'Attrition': ['Yes', 'No']})
fig.update_traces(texttemplate='$\{text:,.0f}', textposition='outside',
                   marker_line=dict(width=1, color='#28221D'))
fig.update_yaxes(zeroline=True, zerolinewidth=1, zerolinecolor="#28221D")
fig.update_layout(title_text='Median Salaries by Department and Attrition Status', font_color="#28221D",
                  yaxis=dict(title='Salary', tickprefix='$', range=(0,79900)), width=950, height=500,
                  paper_bgcolor='#F4F2F0', plot_bgcolor='#F4F2F0')
fig.show()

plot_df = hr_df.groupby('JobRole')['MonthlyIncome'].mean()
plot_df = plot_df.mul(12).rename('Salary').reset_index().sort_values('Salary', ascending=False)
fig = px.bar(plot_df, x='JobRole', y='Salary', text='Salary', opacity=0.7)
fig.update_traces(texttemplate='$\{text:,.0f}', textposition='outside',
                   marker_line=dict(width=1, color='#28221D'), marker_color='#3A5F53')
fig.update_yaxes(zeroline=True, zerolinewidth=1, zerolinecolor="#28221D")
fig.update_layout(title_text='Average Salaries by Job Role', font_color="#28221D",
                  yaxis=dict(title='Salary', tickprefix='$'), height=500,
                  xaxis_title='', paper_bgcolor='#F4F2F0', plot_bgcolor='#F4F2F0')
fig.show()

cat_cols=[]
for i in hr_df.columns:
    if hr_df[i].nunique() <= 5 or hr_df[i].dtype == object:
        cat_cols.append(i)
df=hr_df.copy()
df.drop(df[cat_cols], axis=1, inplace=True)
df.drop('EmployeeNumber', axis=1, inplace=True)
corr=df.corr().round(2)
x=corr.index.tolist()
y=corr.columns.tolist()
z=corr.to_numpy()
fig = ff.create_annotated_heatmap(z=z, x=x, y=y, annotation_text=z, name='',
                                  hovertemplate="Correlation between %{x} and %{y}= %{z}",
                                  colorscale='GnBu')
fig.update_yaxes(autorange="reversed")
fig.update_layout(title="Correlation Matrix of Employee Attrition",
                  font_color="#28221D", margin=dict(t=180), height=600)
fig.show()

```

▼ Objective Factors

```
Age_YAC_NCW = hr_df[['Age', 'YearsAtCompany', 'NumCompaniesWorked', 'Attrition']]  
Age_YAC_NCW.head()  
AYN_melt = Age_YAC_NCW.melt(['Attrition'], var_name='cols', value_name='vals')  
AYN_melt.head()
```

	Attrition	cols	vals
0	Yes	Age	41
1	No	Age	49
2	Yes	Age	37
3	No	Age	33
4	No	Age	27

```
g = sns.FacetGrid(AYN_melt, col='cols', hue="Attrition", palette="Set1")  
g = (g.map(sns.distplot, "vals", hist=False, rug=False, kde_kws={"shade": True}).add_legend());  
#Focus on the red area--distribution of Attrition  
#Lots of attrition among the 'young' employees (for Age or for YearsAtCompany)-- not surprising -- trial to find a suit -- company culture..  
#Those who have worked in more than 5 companies tend to attrit. -- The average time spent in every company is shorter -- HR insight..
```

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:848: 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>

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\distributions.py:2511: FutureWarning:
```

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:848: 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>

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\distributions.py:2511: FutureWarning:
```

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:848: 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>

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\distributions.py:2511: FutureWarning:
```

```

# Data
r = [0,1]
Y_F = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['Gender'] == 'Female')])
N_F = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['Gender'] == 'Female')])
Y_M = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['Gender'] == 'Male')])
N_M = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['Gender'] == 'Male')])
raw_data = {'greenBars': [Y_F, Y_M], 'orangeBars': [N_F, N_M]}
df = pd.DataFrame(raw_data)

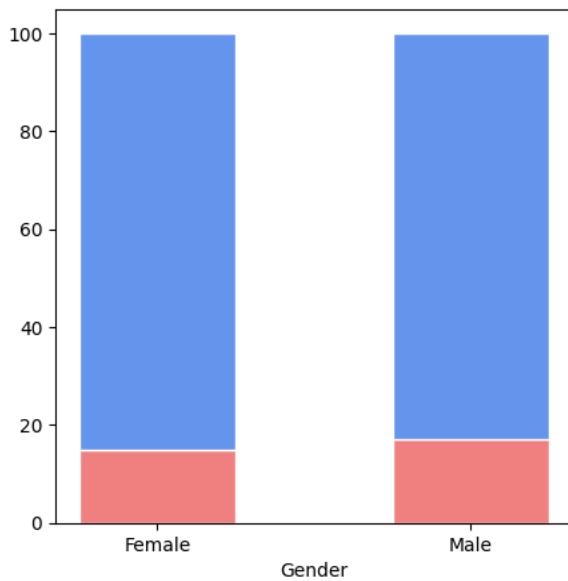
# From raw value to percentage
totals = [i+j for i,j in zip(df['greenBars'], df['orangeBars'])]
greenBars = [i / j * 100 for i,j in zip(df['greenBars'], totals)]
orangeBars = [i / j * 100 for i,j in zip(df['orangeBars'], totals)]

# plot
plt.figure(figsize=(5,5))
barWidth = 0.5
names = ('Female','Male')
# Create green Bars
plt.bar(r, greenBars, color='lightcoral', edgecolor='white', width=barWidth)
# Create orange Bars
plt.bar(r, orangeBars, bottom=greenBars, color='cornflowerblue', edgecolor='white', width=barWidth)
# Create blue Bars
# plt.bar(r, blueBars, bottom=[i+j for i,j in zip(greenBars, orangeBars)], color="#a3acff", edgecolor='white', width=barWidth)

# Custom x axis
plt.xticks(r, names)
plt.xlabel("Gender")

# Show graphic
plt.show()

```



▼ Return and Bonus

```

Income = hr_df[['MonthlyIncome', 'Attrition']]
Income_melt = Income.melt(['Attrition'], var_name='cols', value_name='vals')

g = sns.FacetGrid(Income_melt, col='cols', hue="Attrition", palette="Set1")
g = (g.map(sns.distplot, "vals", hist=False, rug=False, kde_kws={"shade": True}).add_legend())
#Employees with low income tend to attrit

```

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:848: 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>

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\distributions.py:2511: FutureWarning:
```

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:848: 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).

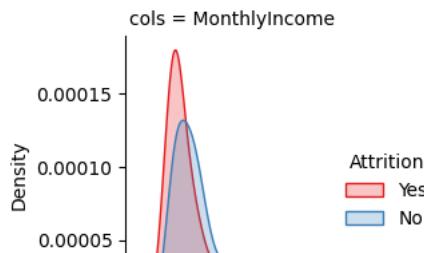
For a guide to updating your code to use the new functions, please see
<https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\distributions.py:2511: FutureWarning:
```

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

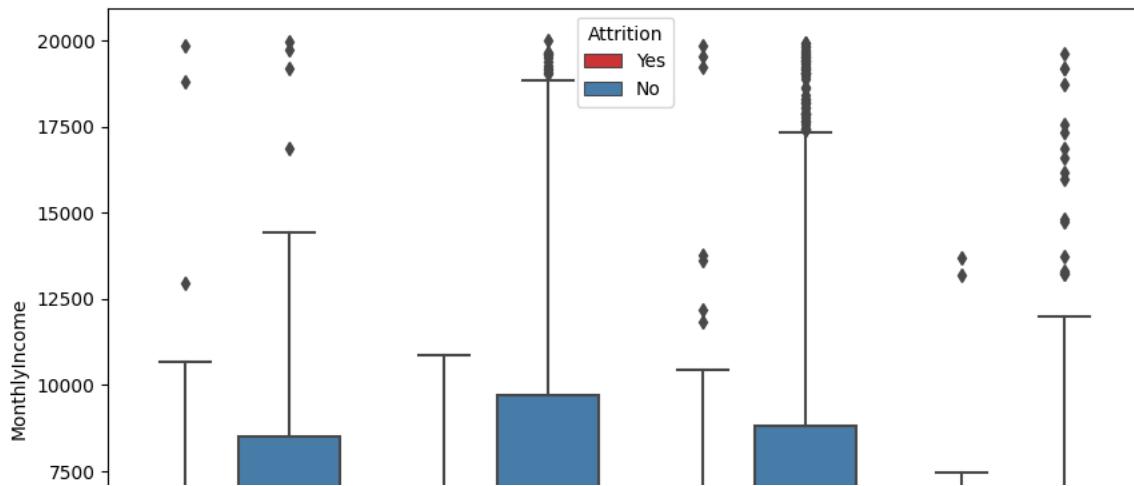
```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning:
```

The figure layout has changed to tight



```
plt.figure(figsize=(10,7))
sns.boxplot(x="JobInvolvement", y="MonthlyIncome", hue="Attrition",
            data=hr_df, palette="Set1")
##Interesting Finding: Low income together with similar involvement is the reason behind attrition
```

<Axes: xlabel='JobInvolvement', ylabel='MonthlyIncome'>



```

TS = hr_df[['TrainingTimesLastYear','StockOptionLevel','Attrition']]
#Age_YAC_NCW.head()
TS_melt = TS.melt(['Attrition'], var_name='cols', value_name='vals')
#AYN_melt.head()
g = sns.FacetGrid(TS_melt, col='cols', hue="Attrition", palette="Set1")
g = (g.map(sns.distplot, "vals", hist=False, rug=False, kde_kws={"shade": True}).add_legend())
#Employee pay attention to stock option -- when given option, work for the company may increase return.

```

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:848: 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>

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\distributions.py:2511: FutureWarning:
```

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:848: 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>

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\distributions.py:2511: FutureWarning:
```

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\axisgrid.py:848: 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>

Satisfaction

```
c:\Users\USER\AppData\Local\Programs\Python\Python311\Lib\site-packages\seaborn\distributions.py:2511: FutureWarning:
```

```
# Data
r = [0,1,2,3]
Y_J1 = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['JobSatisfaction'] == 1)])
N_J1 = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['JobSatisfaction'] == 1)])
Y_J2 = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['JobSatisfaction'] == 2)])
N_J2 = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['JobSatisfaction'] == 2)])
Y_J3 = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['JobSatisfaction'] == 3)])
N_J3 = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['JobSatisfaction'] == 3)])
Y_J4 = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['JobSatisfaction'] == 4)])
N_J4 = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['JobSatisfaction'] == 4)])
raw_data = {'greenBars': [Y_J1, Y_J2, Y_J3, Y_J4], 'orangeBars': [N_J1, N_J2, N_J3, N_J4]}
df = pd.DataFrame(raw_data)
```

```
# From raw value to percentage
totals = [i+j for i,j in zip(df['greenBars'], df['orangeBars'])]
greenBars = [i / j * 100 for i,j in zip(df['greenBars'], totals)]
orangeBars = [i / j * 100 for i,j in zip(df['orangeBars'], totals)]
```

```
# plot
plt.figure(figsize=(5,5))
barWidth = 0.5
names = ('1','2','3','4')
# Create green Bars
plt.bar(r, greenBars, color='lightcoral', edgecolor='white', width=barWidth)
# Create orange Bars
```

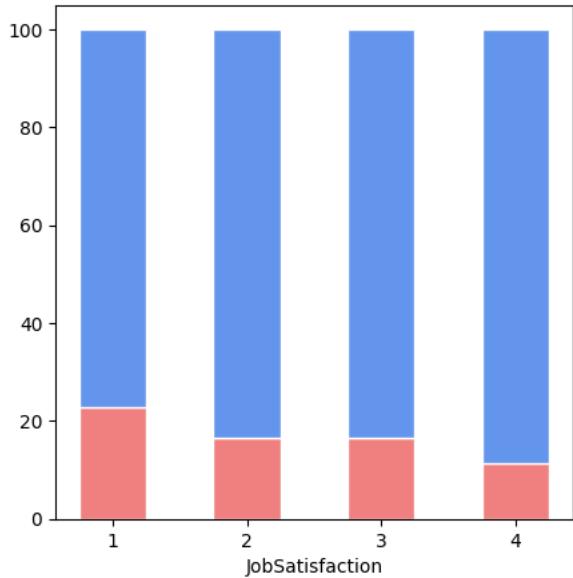
```

plt.bar(r, orangeBars, bottom=greenBars, color='cornflowerblue', edgecolor='white', width=barWidth)
# Create blue Bars
#plt.bar(r, blueBars, bottom=[i+j for i,j in zip(greenBars, orangeBars)], color='#a3acff', edgecolor='white', width=barWidth)

# Custom x axis
plt.xticks(r, names)
plt.xlabel("JobSatisfaction")

# Show graphic
plt.show()

```



```

# Data
r = [0,1,2,3]
Y_J1 = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['EnvironmentSatisfaction'] == 1)])
N_J1 = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['EnvironmentSatisfaction'] == 1)])
Y_J2 = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['EnvironmentSatisfaction'] == 2)])
N_J2 = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['EnvironmentSatisfaction'] == 2)])
Y_J3 = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['EnvironmentSatisfaction'] == 3)])
N_J3 = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['EnvironmentSatisfaction'] == 3)])
Y_J4 = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['EnvironmentSatisfaction'] == 4)])
N_J4 = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['EnvironmentSatisfaction'] == 4)])
raw_data = {'greenBars': [Y_J1, Y_J2, Y_J3, Y_J4], 'orangeBars': [N_J1, N_J2, N_J3, N_J4]}
df = pd.DataFrame(raw_data)

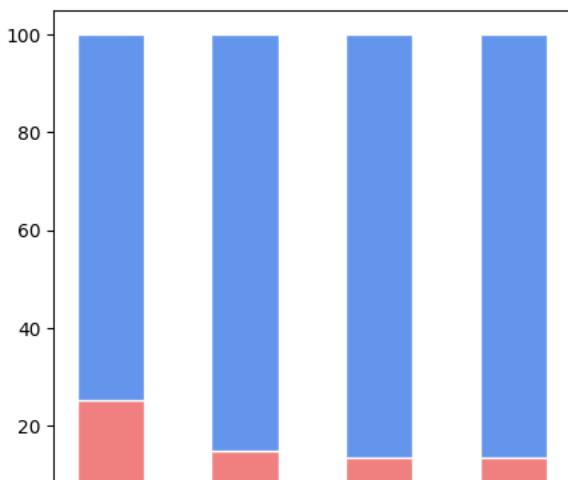
# From raw value to percentage
totals = [i+j for i,j in zip(df['greenBars'], df['orangeBars'])]
greenBars = [i / j * 100 for i,j in zip(df['greenBars'], totals)]
orangeBars = [i / j * 100 for i,j in zip(df['orangeBars'], totals)]

# plot
plt.figure(figsize=(5,5))
barWidth = 0.5
names = ('1','2','3','4')
# Create green Bars
plt.bar(r, greenBars, color='lightcoral', edgecolor='white', width=barWidth)
# Create orange Bars
plt.bar(r, orangeBars, bottom=greenBars, color='cornflowerblue', edgecolor='white', width=barWidth)
# Create blue Bars
#plt.bar(r, blueBars, bottom=[i+j for i,j in zip(greenBars, orangeBars)], color='#a3acff', edgecolor='white', width=barWidth)

# Custom x axis
plt.xticks(r, names)
plt.xlabel("EnvironmentSatisfaction")

# Show graphic
plt.show()

```



```

# Data
r = [0,1,2,3]
Y_J1 = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['RelationshipSatisfaction'] == 1)])
N_J1 = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['RelationshipSatisfaction'] == 1)])
Y_J2 = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['RelationshipSatisfaction'] == 2)])
N_J2 = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['RelationshipSatisfaction'] == 2)])
Y_J3 = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['RelationshipSatisfaction'] == 3)])
N_J3 = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['RelationshipSatisfaction'] == 3)])
Y_J4 = len(hr_df[(hr_df['Attrition'] == 'Yes') & (hr_df['RelationshipSatisfaction'] == 4)])
N_J4 = len(hr_df[(hr_df['Attrition'] == 'No') & (hr_df['RelationshipSatisfaction'] == 4)])
raw_data = {'greenBars': [Y_J1, Y_J2, Y_J3, Y_J4], 'orangeBars': [N_J1, N_J2, N_J3, N_J4]}
df = pd.DataFrame(raw_data)

# From raw value to percentage
totals = [i+j for i,j in zip(df['greenBars'], df['orangeBars'])]
greenBars = [i / j * 100 for i,j in zip(df['greenBars'], totals)]
orangeBars = [i / j * 100 for i,j in zip(df['orangeBars'], totals)]

# plot
plt.figure(figsize=(5,5))
barWidth = 0.5
names = ('1','2','3','4')
# Create green Bars
plt.bar(r, greenBars, color='lightcoral', edgecolor='white', width=barWidth)
# Create orange Bars
plt.bar(r, orangeBars, bottom=greenBars, color='cornflowerblue', edgecolor='white', width=barWidth)
# Create blue Bars
#plt.bar(r, blueBars, bottom=[i+j for i,j in zip(greenBars, orangeBars)], color='#a3acff', edgecolor='white', width=barWidth)

# Custom x axis
plt.xticks(r, names)
plt.xlabel("RelationshipSatisfaction")

# Show graphic
plt.show()

```

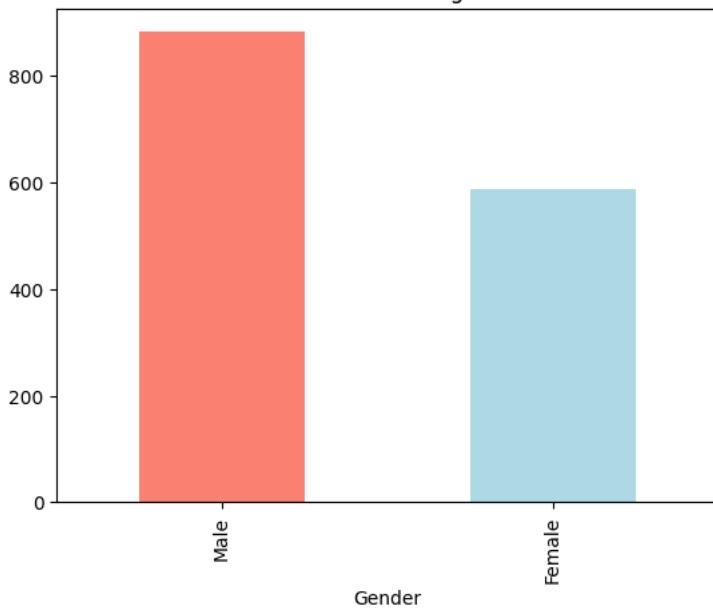


Understanding the balancing of the Gender column visually

```
hr_df['Gender'].value_counts().plot(kind='bar',color=['salmon','lightblue'],title="Count of different gender")
```

<Axes: title={'center': 'Count of different gender'}, xlabel='Gender'>

Count of different gender



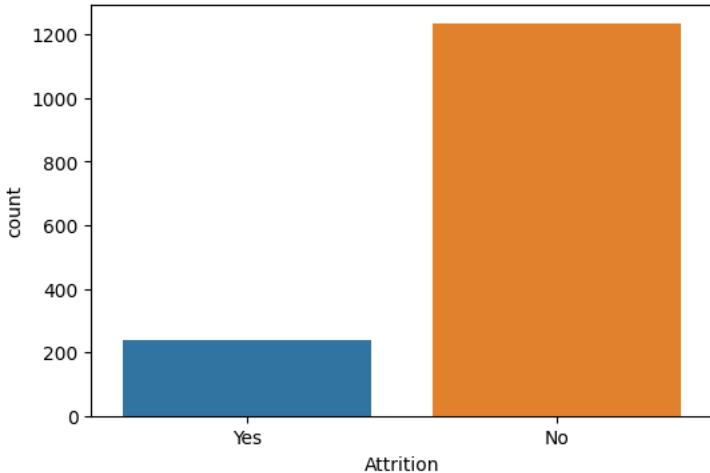
```
hr_df['Gender'].value_counts()
```

```
Gender
Male     882
Female   588
Name: count, dtype: int64
```

Attrition Distribution

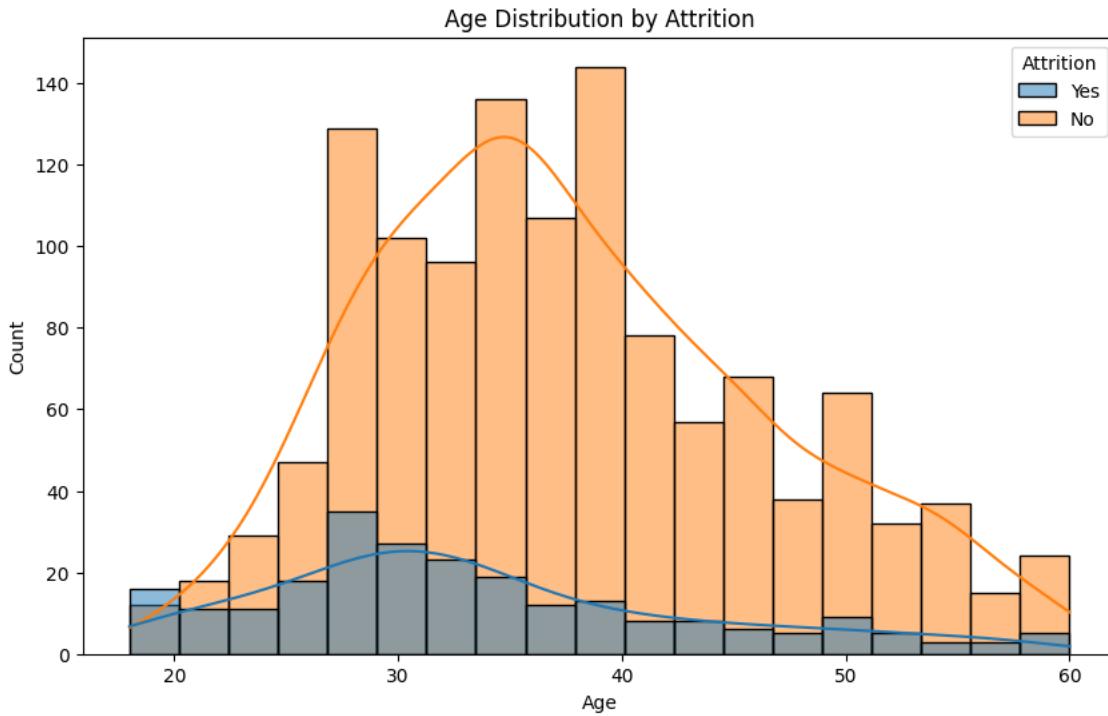
```
plt.figure(figsize=(6, 4))
sns.countplot(data=hr_df, x='Attrition')
plt.title('Attrition Distribution')
plt.show()
```

Attrition Distribution

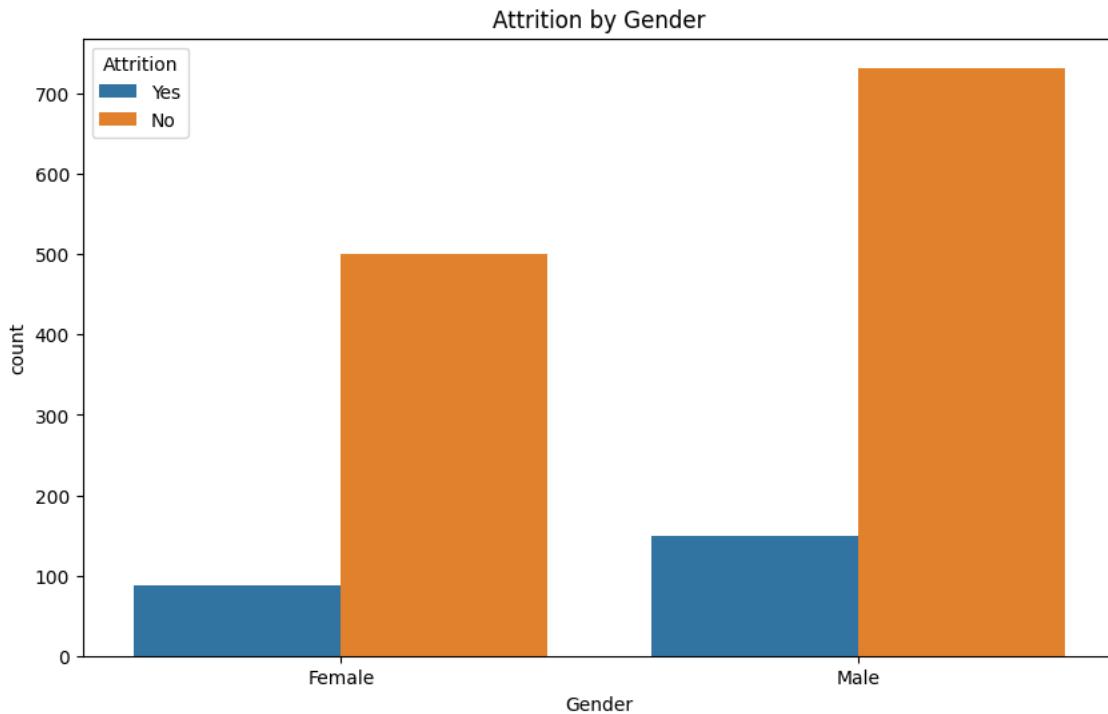


```
# Explore numeric variables (e.g., Age, MonthlyIncome) with histograms and box plots
plt.figure(figsize=(10, 6))
```

```
sns.histplot(data=hr_df, x='Age', hue='Attrition', kde=True)
plt.title('Age Distribution by Attrition')
plt.show()
```

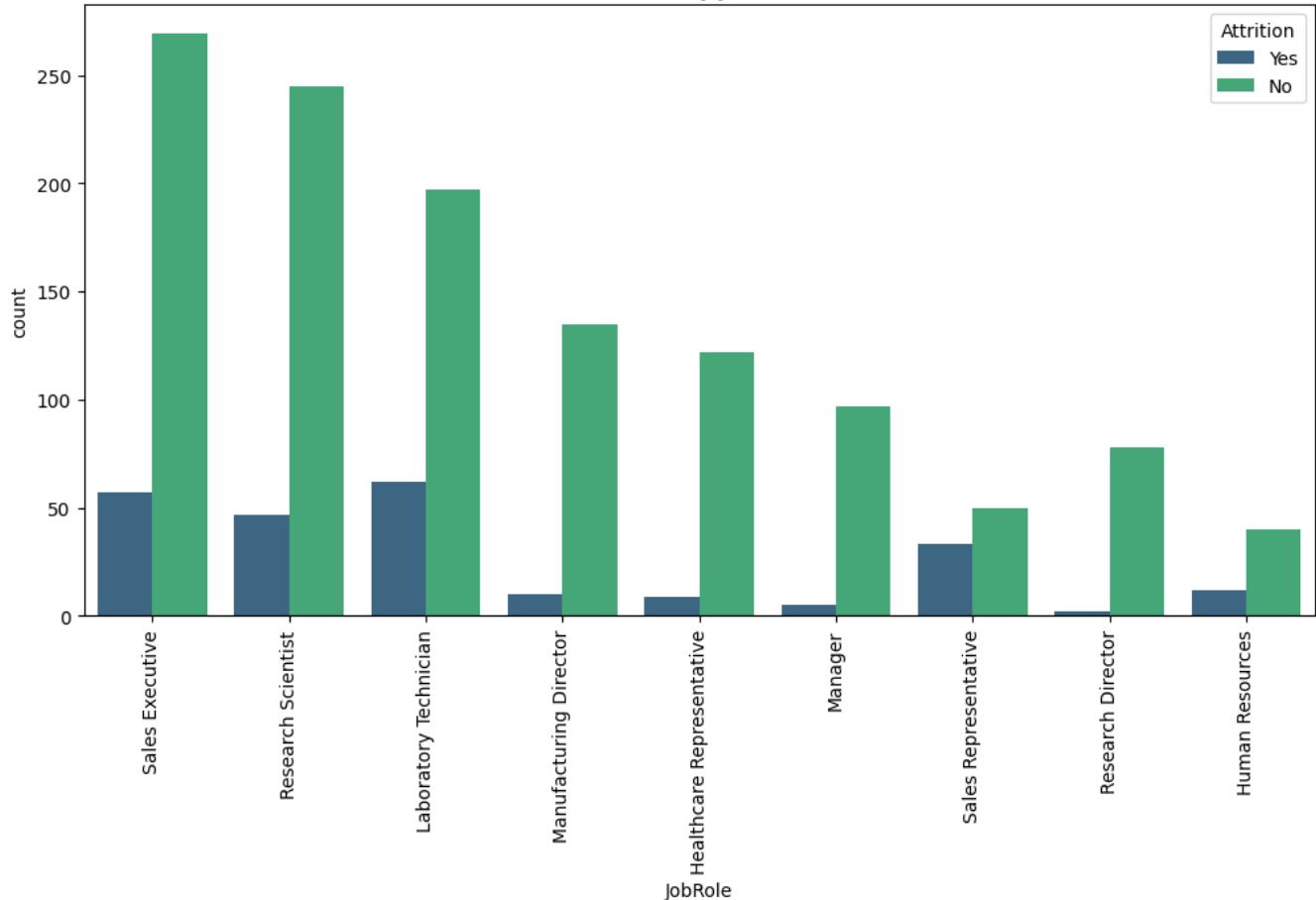


```
# Explore categorical variables (e.g., Gender, Department) with bar plots
plt.figure(figsize=(10, 6))
sns.countplot(data=hr_df, x='Gender', hue='Attrition')
plt.title('Attrition by Gender')
plt.show()
```



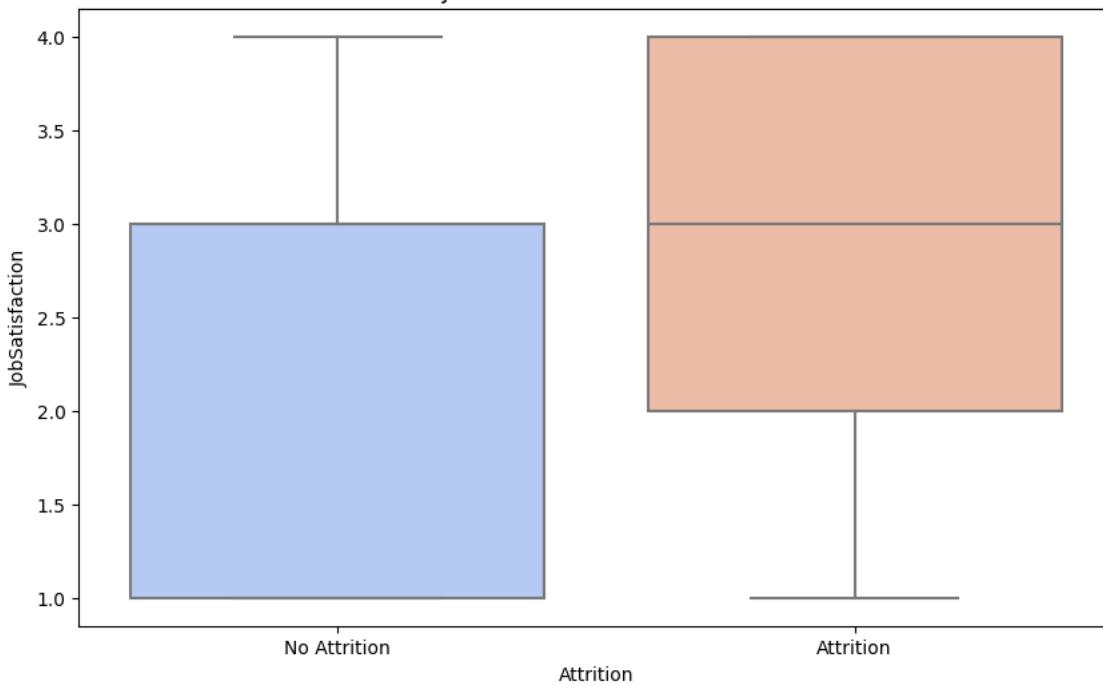
```
plt.figure(figsize=(12, 6))
sns.countplot(data=hr_df, x='JobRole', hue='Attrition', palette='viridis')
plt.title('Attrition by Job Role')
plt.xticks(rotation=90)
plt.show()
```

Attrition by Job Role

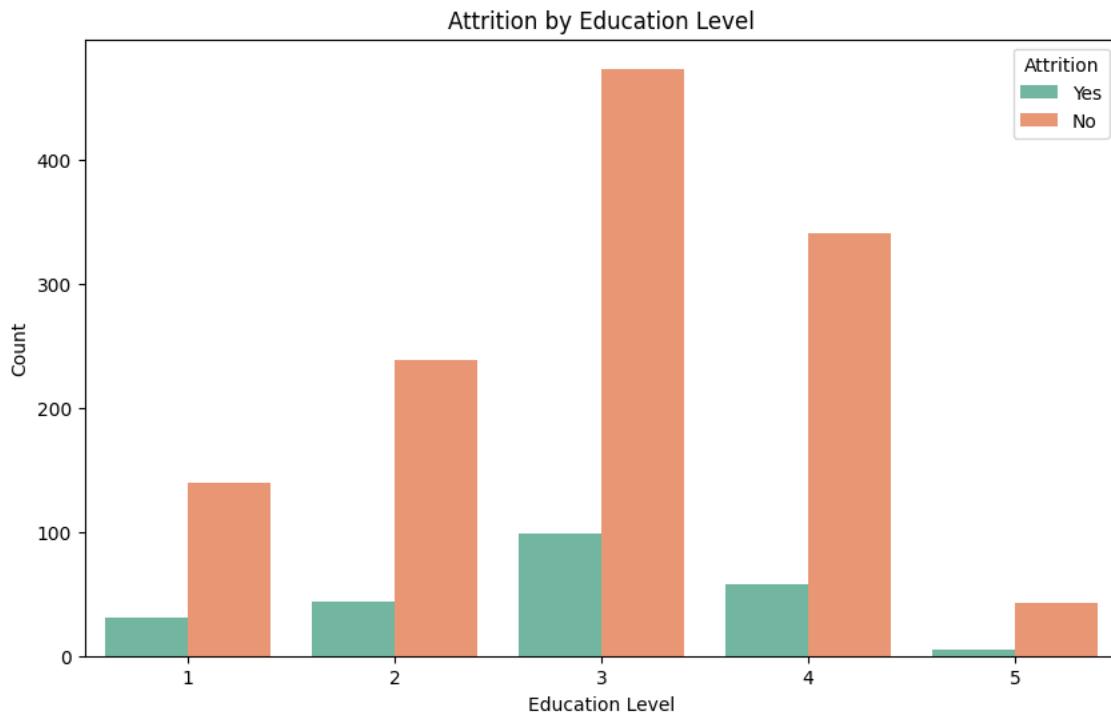


```
plt.figure(figsize=(10, 6))
sns.boxplot(data=hr_df, x='Attrition', y='JobSatisfaction', palette='coolwarm')
plt.title('Job Satisfaction vs. Attrition')
plt.xticks([0, 1], ['No Attrition', 'Attrition'])
plt.show()
```

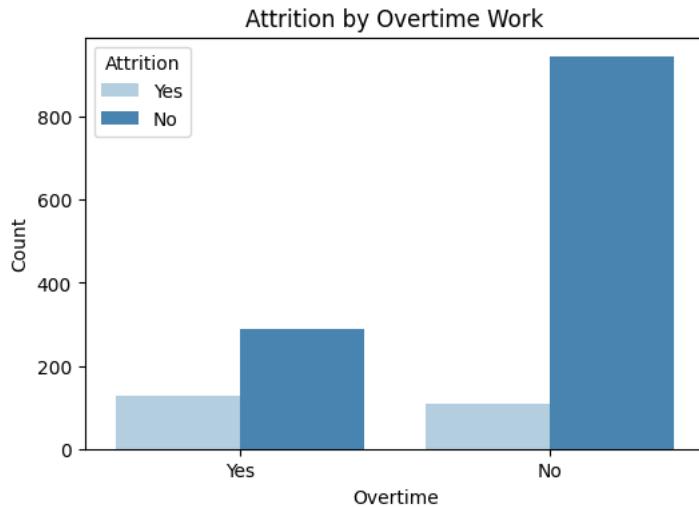
Job Satisfaction vs. Attrition



```
plt.figure(figsize=(10, 6))
sns.countplot(data=hr_df, x='Education', hue='Attrition', palette='Set2')
plt.title('Attrition by Education Level')
plt.xlabel('Education Level')
plt.ylabel('Count')
plt.show()
```

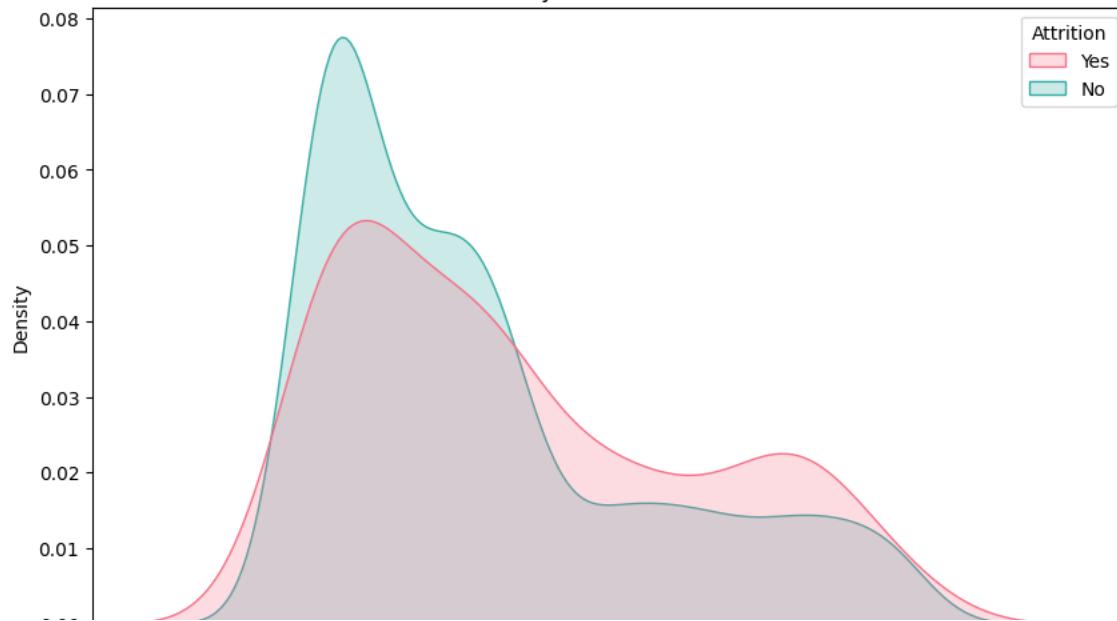


```
plt.figure(figsize=(6, 4))
sns.countplot(data=hr_df, x='OverTime', hue='Attrition', palette='Blues')
plt.title('Attrition by Overtime Work')
plt.xlabel('Overtime')
plt.ylabel('Count')
plt.show()
```



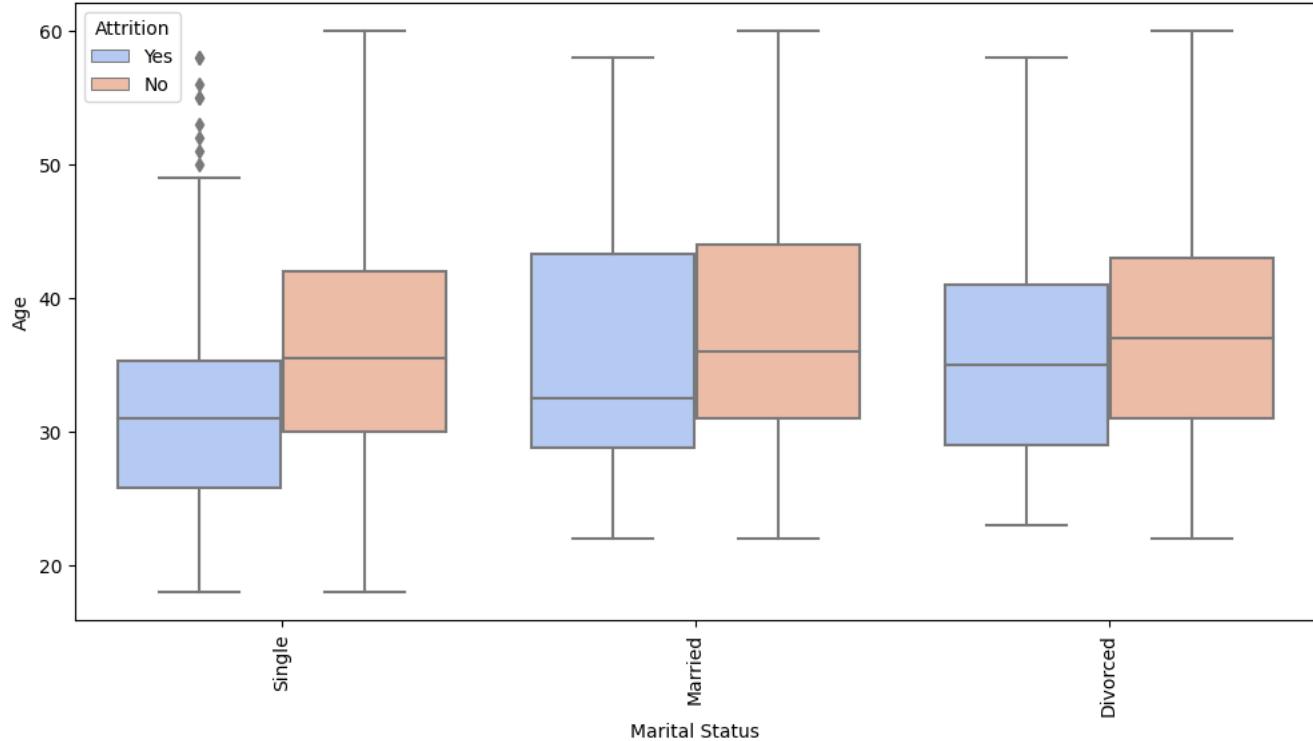
```
plt.figure(figsize=(10, 6))
sns.kdeplot(data=hr_df, x='DistanceFromHome', hue='Attrition', fill=True, common_norm=False, palette='husl')
plt.title('Attrition by Distance from Home')
plt.xlabel('Distance from Home (Miles)')
plt.ylabel('Density')
plt.show()
```

Attrition by Distance from Home



```
plt.figure(figsize=(12, 6))
sns.boxplot(data=hr_df, x='MaritalStatus', y='Age', hue='Attrition', palette='coolwarm')
plt.title('Attrition by Marital Status and Age')
plt.xticks(rotation=90)
plt.xlabel('Marital Status')
plt.ylabel('Age')
plt.show()
```

Attrition by Marital Status and Age



```
plt.figure(figsize=(10, 6))
sns.histplot(data=hr_df, x='MonthlyIncome', hue='Attrition', kde=True, palette='husl')
plt.title('Monthly Income Distribution by Attrition')
plt.xlabel('Monthly Income')
plt.ylabel('Count')
plt.show()
```