IBM HR Analytics Employee Attrition & Performance

Brief description of the data set and a summary of its attributes

The original dataset was obtained from data source [1] and contained 1470 observations, 31 features, and a target of 'Attrition'. Furthermore, the dataset was clean and contained no NULL values. Further details were provided through the data dictionary below in order to understand the context of the labels such as Education, Environment_Satisfaction, Job_Involvement, Job_Satisfaction, Performance_Rating, Relationship_Satisfaction, Work_Life_Balance, Job_Level, and Stock_Option_Level being on a scale from 1-5, while Distance_From_Home was measured in Kilometers and Percent_Salary_Hike is the percent increase in salary compared to the previous year.

Dataset statistics

Number of variables	32
Number of observations	1470
Missing cells	0

Variable types

Numeric	15
Categorical	16
Boolean	1

Data Dictionary

Education	1: 'Below College', 2: 'College', 3: 'Bachelor', 4: 'Master', 5: 'Doctor'
Environment_Satisfaction	1: 'Low', 2: 'Medium', 3: 'High', 4: 'Very High'
Job_Involvement	1: 'Low', 2: 'Medium', 3: 'High', 4: 'Very High'
Job_Satisfaction	1: 'Low', 2: 'Medium', 3: 'High', 4: 'Very High'
Performance_Rating	1: 'Low', 2: 'Good', 3: 'Excellent', 4: 'Outstanding'
Relationship_Satisfaction	1: 'Low', 2: 'Medium', 3: 'High', 4: 'Very High'
Work_Life_Balance	1: 'Bad', 2: 'Good', 3: 'Better', 4: 'Best'
Distance_From_Home	Measured in Kilometers
Stock_Option_Level	1 - 5 scale
Job_Level	1 - 5 scale
Percent_Salary_Hike	Percentage increase compared to the previous year

Quality of dataset

This dataset exported from [1], was immaculately clean. Moreover, additional data that would provide more insight would be the reason of Attrition. Did the individual seek further education, did they pursue a job within a completely different industry, or did they make a lateral move.

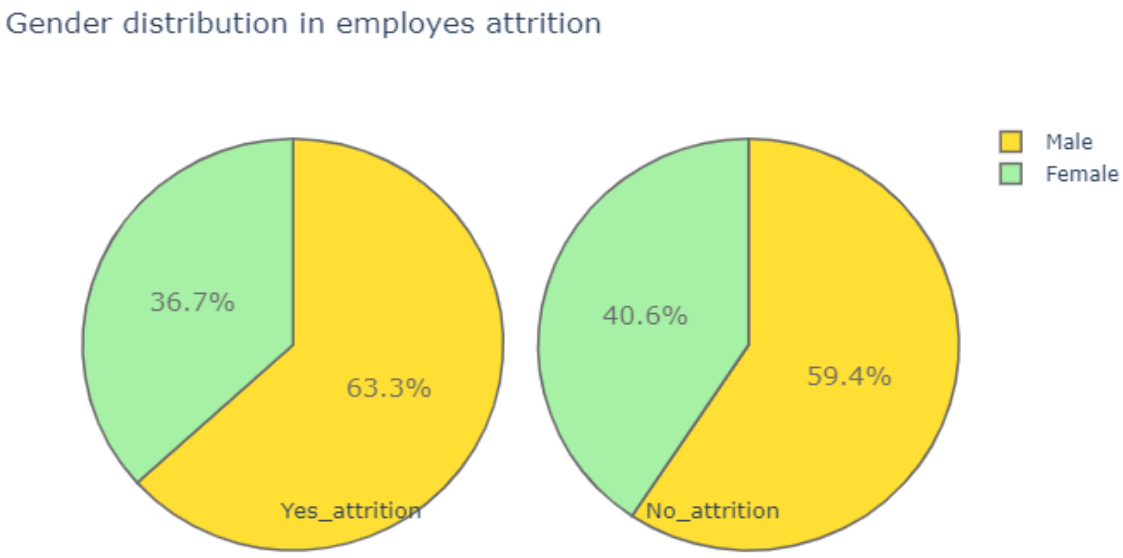
Initial plan for data exploration

- Check for missing data (NULLS).
- Review columns to identify what is needed or not.
- Review column names and understand what each label means.
- Make plots for initial insights.

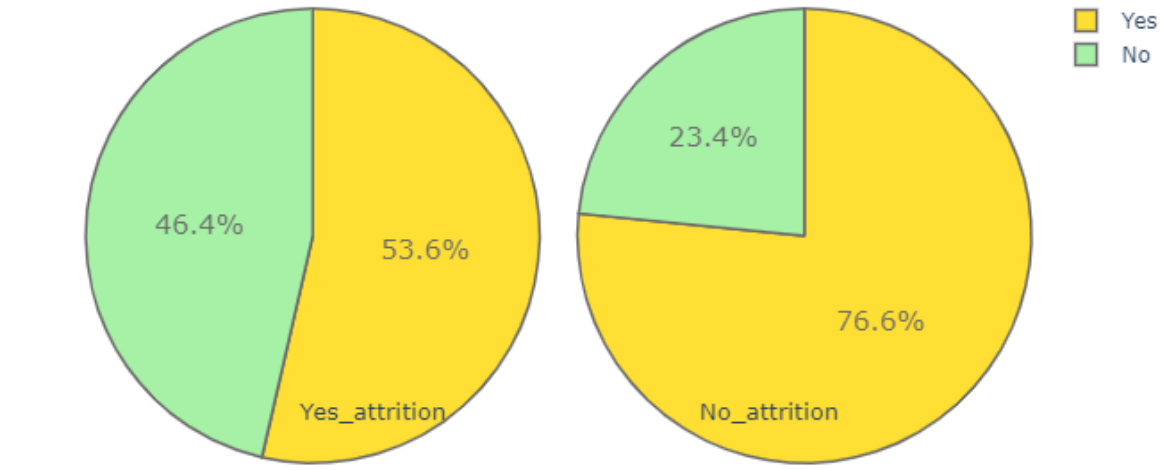
Actions taken for data cleaning and feature engineering

- Clean column names.
- Review target variable against other features ('Attrition' vs x).
- Create plots against 'Attrition'.
- Remove columns: 'EmployeeCount', 'Over18', 'StandardHours', 'EmployeeNumber'

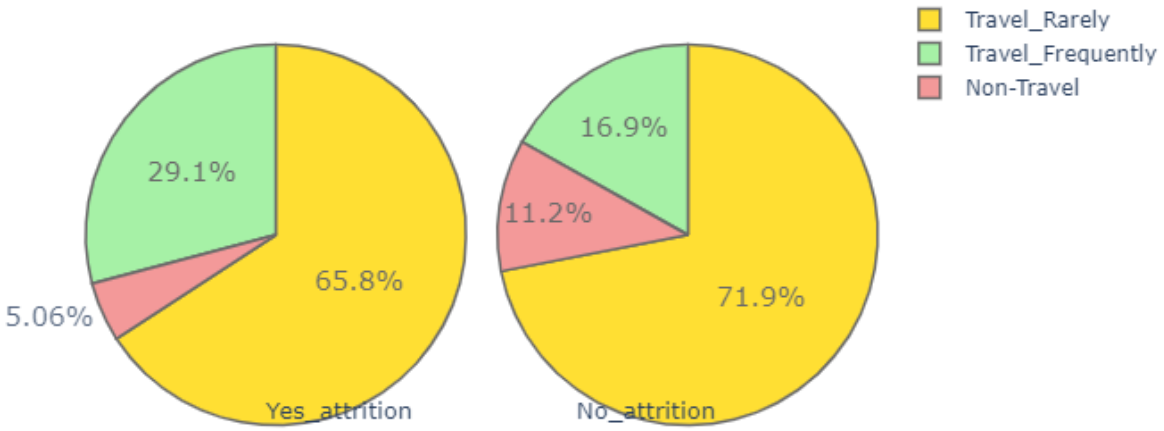
Plots



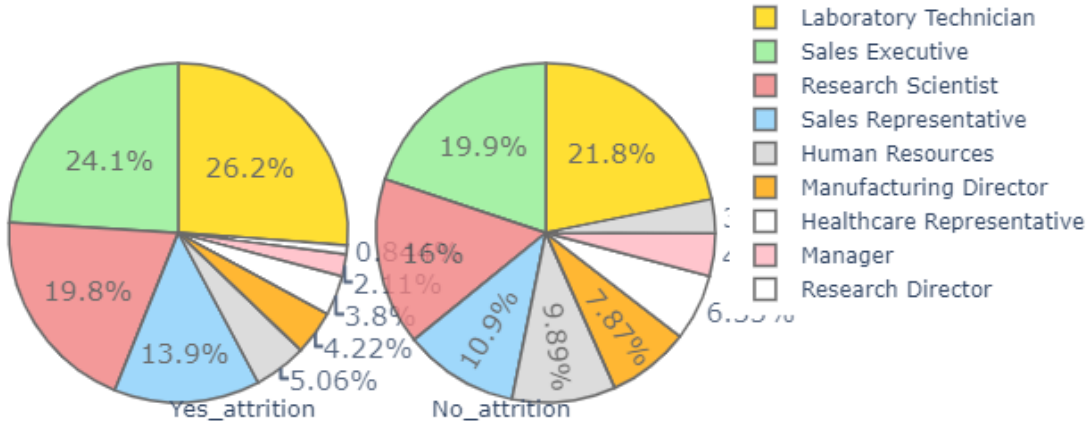
Overtime distribution in employes attrition



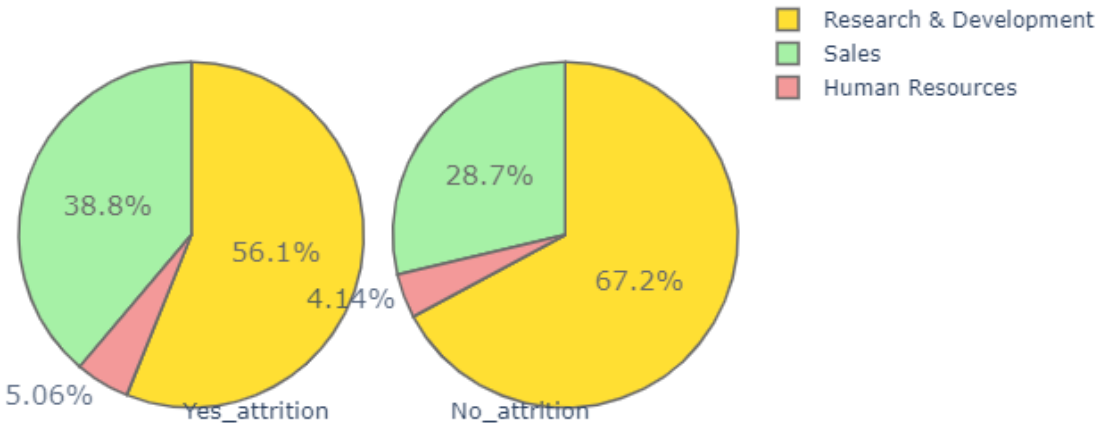
Business_Travel distribution in employes attrition



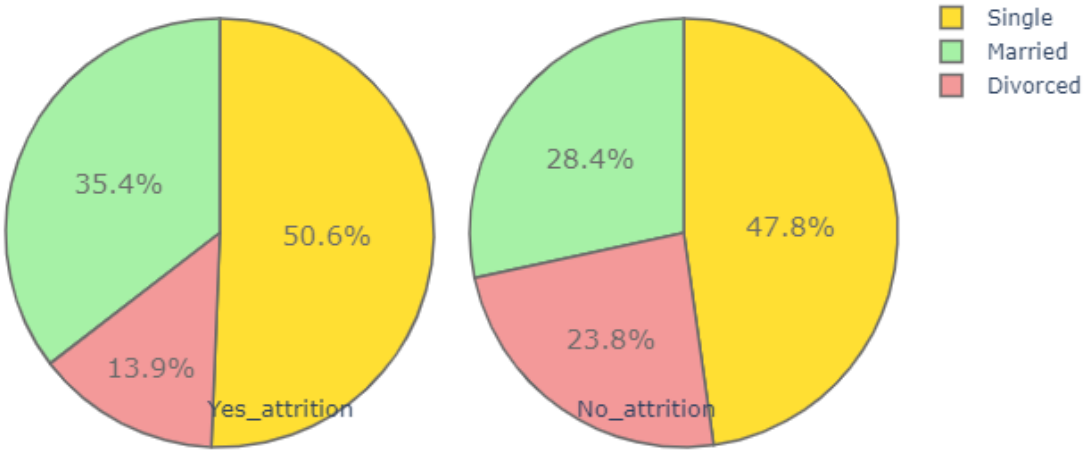
Job_Role distribution in employes attrition



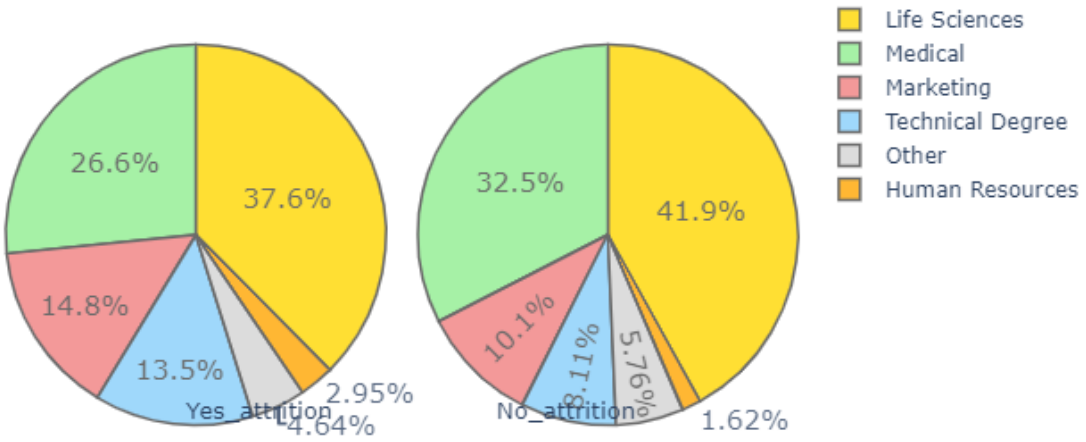
Department distribution in employes attrition



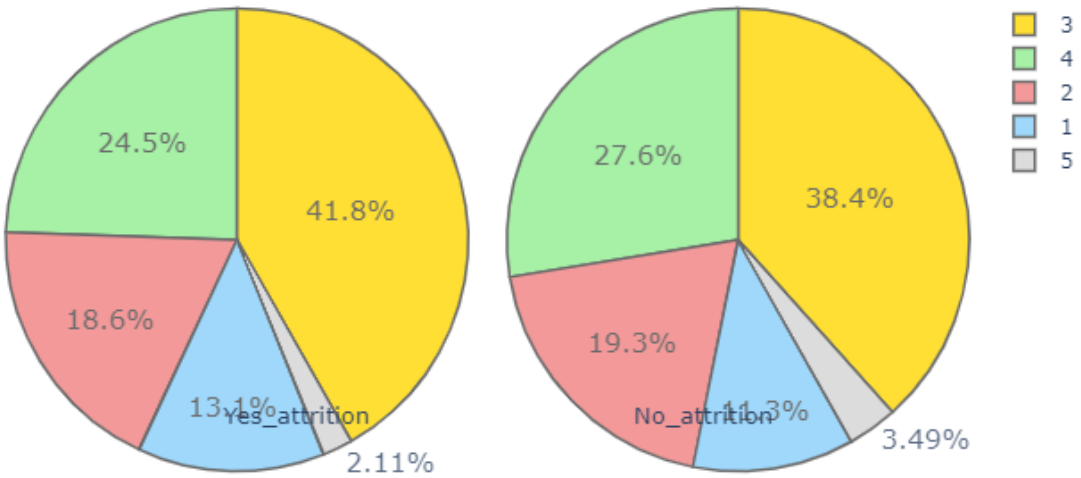
Marital_Status distribution in employes attrition



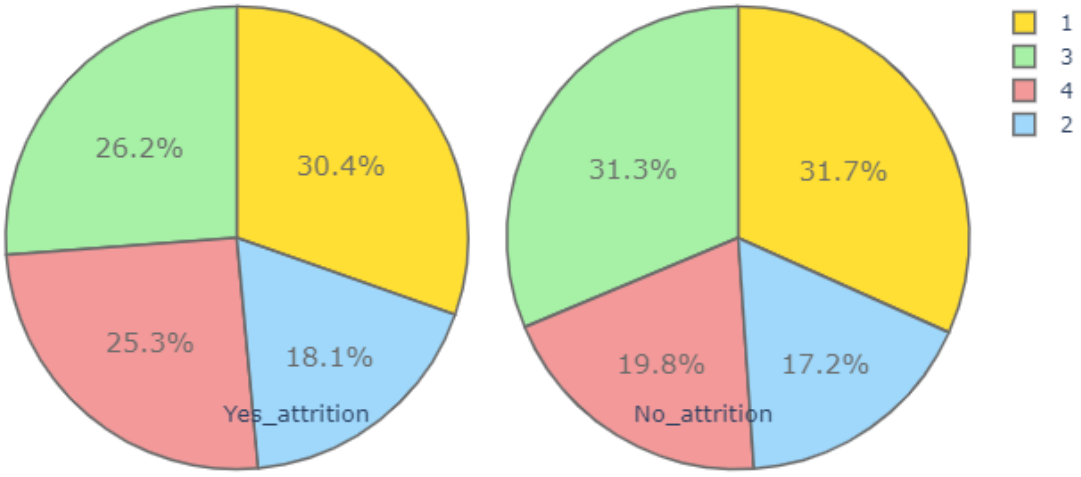
Education_Field distribution in employes attrition



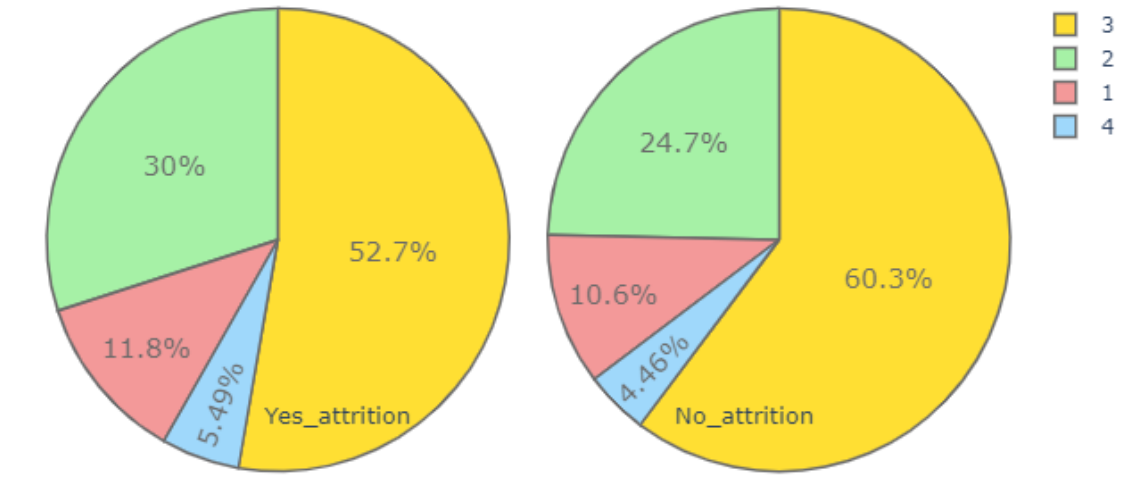
Education distribution in employes attrition



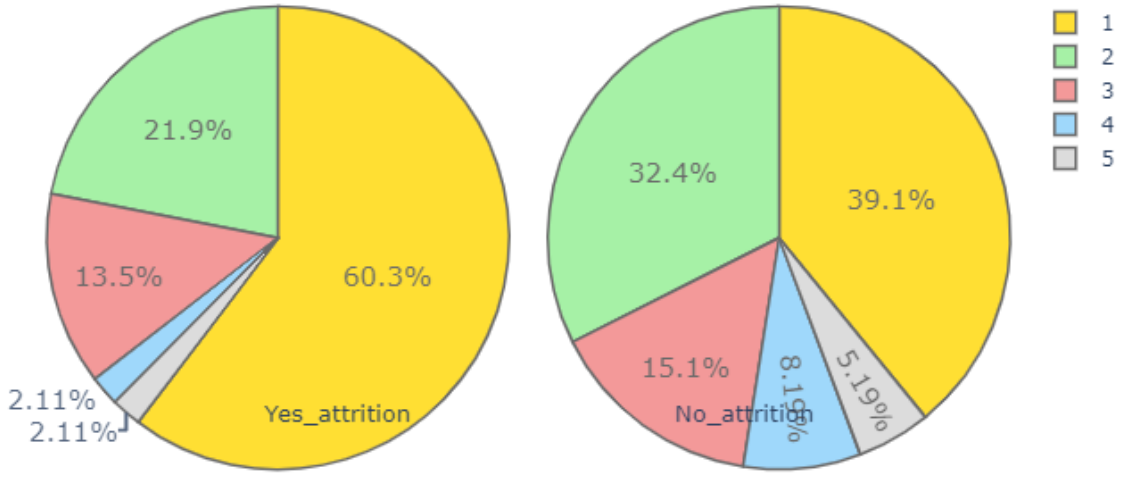
Environment_Satisfaction distribution in employes attrition



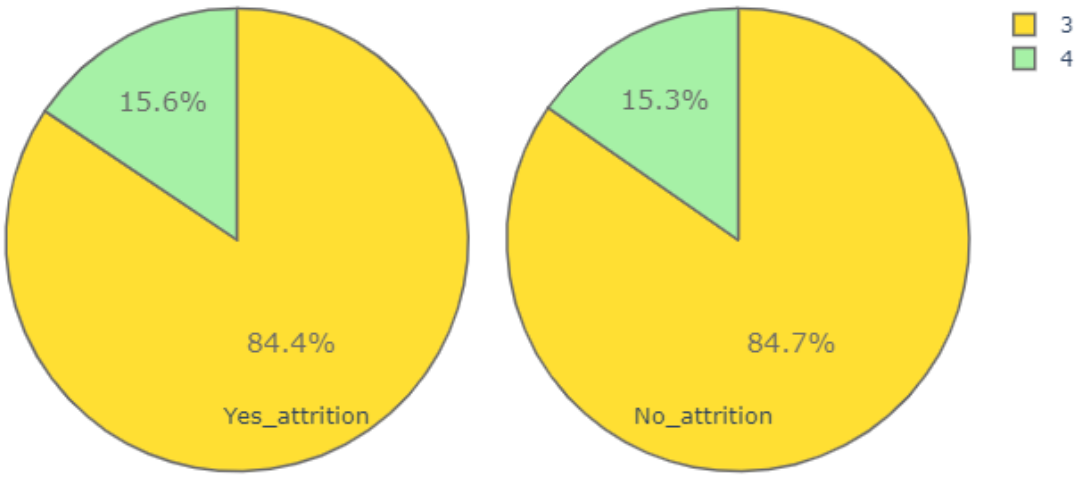
Job_Involvement distribution in employes attrition



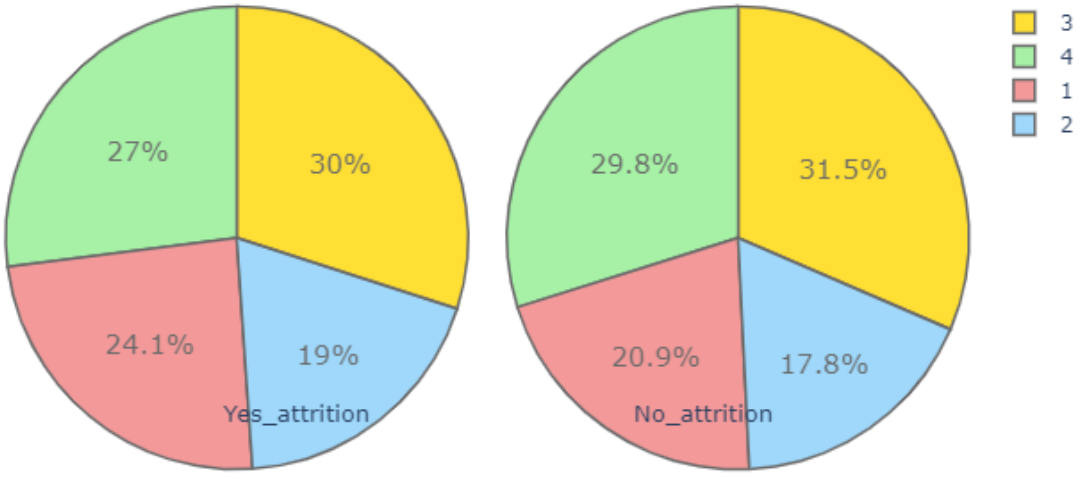
Job_Level distribution in employes attrition



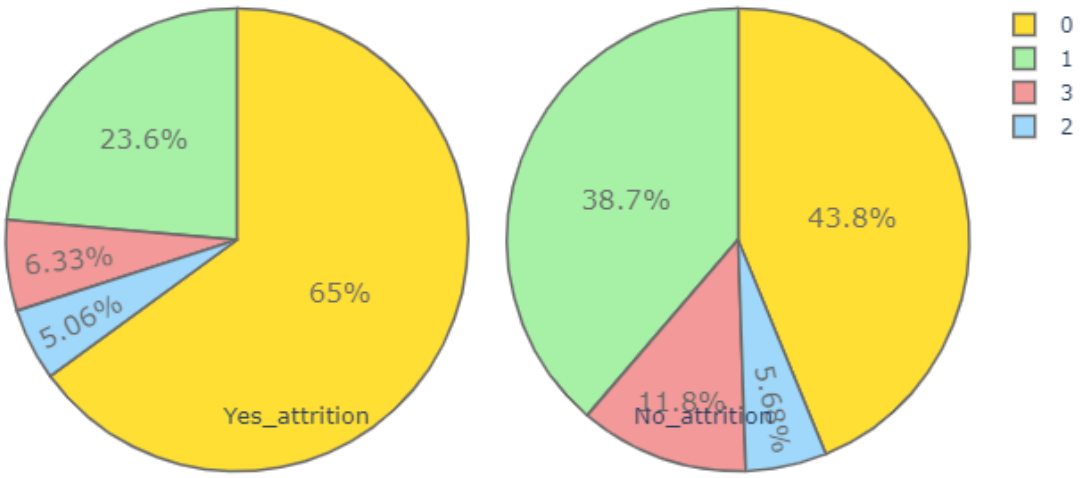
Performance_Rating distribution in employes attrition



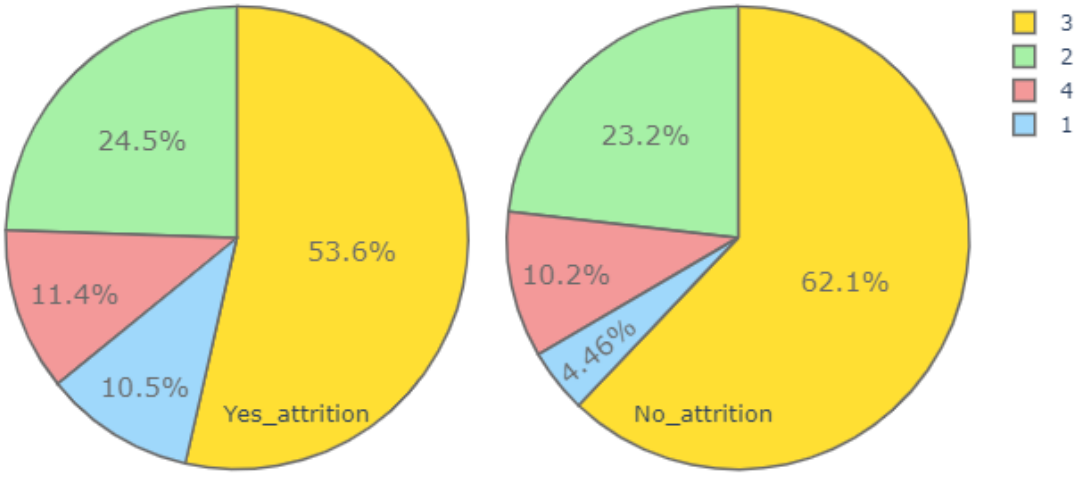
Relationship_Satisfaction distribution in employes attrition



Stock_Option_Level distribution in employes attrition



Work_Life_Balance distribution in employes attrition



Key Findings and Insights, which synthesizes the results of Exploratory Data Analysis in an insightful and actionable manner

'Attrition' was at 66.67% for 19-year-olds, however the 'Attrition' rate drastically drops until employees hit the age of 58, where 'Attrition' jumps up to 35.71%.

In cases 'Distance_From_Home' was 25 kilometers the "Attrition' rate 42.85%.

In cases 'Percent_Salary_Hike' was 24 (24% increase from previous year) 'Attrition' rate was 28.57%.

In cases 'Total_Working_Years' was 0 the 'Attrition' rate was 45.45%, when 1 the 'Attrition' rate was 49.38% and at 40 100%, likely due to retirement.

In cases 'Training_Time_Last_Year' was 0 the 'Attrition' rate was 27.78%.

In cases 'Year_At_Company' was 0 the 'Attrition' rate was 36.36%, when 1 34.50%.

Interestingly in cases an employee was 'Overtime' the 'Attrition' rate was 30.52% in contrast to in cases employee's 'Travel_Frequently' was 10.43%, the 'Attrition" rate was 24.90%.

Not suprisingly the 'Job_Role' of Sales Rep with the highest 'Attrition' rate was 39.75%. Employees with 'Marital_Status' single has an 'Attrition' rate of 25.53.

'Education_Field' of Human Resources has an 'Attrition' rate of 25.92%.

Formulating 3 hypothesis for the dataset analysis:

First Hypothesis:

- Ho: μ Education_Attrition == μ Education_Not_Attrition
- Ha: μ Education_Attrition != μ Education_Not_Attrition

Second Hypothesis:

- Ho: μ Age_Attrition == μ Age_Not_Attrition
- Ha: μ Age_Attrition != μ Age_Not_Attrition

Third Hypothesis:

- Ho: μ Job_Satisfaction_Attrition == μ Job_Satisfaction_Not_Attrition
- Ha: μ Job_Satisfaction_Attrition != μ Job_Satisfaction_Not_Attrition

Formal significance tests for the three hypothesis and Results

Hypothesis Testing

Test 1

Ho: μ Education_Attrition == μ Education_Not_Attrition

Ha: μ Education_Attrition != μ Education_Not_Attrition

```
attrition_df = clean_df[clean_df['Attrition'] == 1]
not_attrition_df = clean_df[clean_df['Attrition'] == 0]
```

[50] ✓ 0.3s

```
stats.ttest_ind(attrition_df['Education'], not_attrition_df['Education'], equal_var = False)
```

[51] ✓ 0.3s

```
... Ttest_indResult(statistic=-1.2177493963259696, pvalue=0.22417128841924902)
```

Kruskal Test because it is a non-parametric test.

```
ss.kruskal(attrition_df['Education'], not_attrition_df['Education'])
```

[52] ✓ 0.3s

```
... KruskalResult(statistic=1.3527640913093548, pvalue=0.2447954753326153)
```

pvalue > 0.05

There appears to be no statistically significant relationship between Attrition and Education, thus we fail reject the null hypothesis.

Hypothesis Testing

Test 2

Ho: μ Age_Attrition == μ Age_Not_Attrition

Ha: μ Age_Attrition != μ Age_Not_Attrition

```
stats.ttest_ind(attrition_df['Age'],not_attrition_df['Age'], equal_var = False)
```

[53] ✓ 0.3s

```
... Ttest_indResult(statistic=-5.828011853988949, pvalue=1.3797600649439775e-08)
```

Kruskal Test because it is a non-parametric test.

```
ss.kruskal(attrition_df['Age'], not_attrition_df['Age'])
```

[54] ✓ 0.3s

```
... KruskalResult(statistic=43.06268844023747, pvalue=5.3013684961038114e-11)
```

pvalue < 0.05

There appears to be a statistically significant relationship between Attrition and the Age, thus we can reject the null hypothesis.

Suggestions for next steps in analyzing this data

Continue testing different hypothesis and uncovering what the true indicators are for Attrition. Would like to model to uncover which feature is attributed most to Attrition.

References

[1] https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset

Appendix – Functions and Code for EDA

Libraries used for Analysis:

```
import pandas as pd
pd.set_option('display.max_columns', None)
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.offline as py
import plotly.graph_objs as go
import plotly.tools as tls
import plotly.figure_factory as ff
import scipy.stats as ss
import sklearn as skl
from scipy import stats
```

Data Reader

```
df = pd.read_csv('IBM-HR-STUDY.csv')
```

Data Inspect and Data Cleaning

```
df.head()
df.info()
df.isnull().any()
clean_df = df.drop(columns=['EmployeeCount', 'Over18', 'StandardHours', 'EmployeeNumber'])
clean_df.head()


clean_df.Attrition.replace(to_replace = dict(Yes = 1, No = 0), inplace = True)
```

```
header = ['Age', 'Attrition', 'Business_Travel', 'Daily_Rate',
'Department', 'Distance_From_Home', 'Education', 'Education_Field',
'Environment_Satisfaction', 'Gender',
'Hourly_Rate', 'Job_Involvement', 'Job_Level', 'Job_Role',
'Job_Satisfaction', 'Marital_Status', 'Monthly_Income',
'Monthly_Rate', 'Num_Companies_Worked', 'Overtime',
'Percent_Salary_Hike', 'Performance_Rating', 'Relationship_Satisfaction',
'Stock_Option_Level','Total_Working_Years',
'Training_Times_Last_Year', 'Work_Life_Balance', 'Years_At_Company',
'Years_In_Current_Role', 'Years_Since_Last_Promotion',
'Years_With_Current_Manager']

clean_df.columns = header
clean_df.info()
clean_df.describe()
```

Code for Plots

```
attrition = clean_df[(clean_df['Attrition'] != 0)]
no_attrition = clean_df[(clean_df['Attrition'] == 0)]
#Total Count of Yes/No 'Attrition'
trace = go.Bar(x = (len(attrition), len(no_attrition)), y = ['Yes_attrition', 'No_attrition'],
orientation = 'h', opacity = 0.8, marker=dict(color=['gold', 'lightskyblue'],
line=dict(color='#000000',width=1.5)))
layout = dict(title = 'Count of attrition variable')
fig = dict(data = [trace], layout=layout)
py.iplot(fig)
```

```
#Percentage of Yes/No 'Attrition'
trace = go.Pie(labels = ['No_attrition', 'Yes_attrition'], values = clean_df['Attrition'].value_counts(),
textfont=dict(size=15), opacity = 0.8,
marker=dict(colors=['lightskyblue','gold'],
line=dict(color='#000000', width=1.5)))
layout = dict(title = 'Distribution of attrition variable')
fig = dict(data = [trace], layout=layout)
py.iplot(fig)


def plot_distribution(var_select, bin_size):
# Calculate the correlation coefficient between the new variable and the target
    corr = clean_df['Attrition'].corr(clean_df[var_select])
    corr = np.round(corr,3)
    tmp1 = attrition[var_select]
    tmp2 = no_attrition[var_select]
    hist_data = [tmp1, tmp2]
    group_labels = ['Yes_attrition', 'No_attrition']
    colors = ['#FFD700', '#7EC0EE']
    fig = ff.create_distplot(hist_data, group_labels, colors = colors, show_hist = True,
    curve_type='kde', bin_size = bin_size)
    fig['layout'].update(title = var_select+' '+'(corr target ='+ str(corr)+')')
    py.iplot(fig, filename = 'Density plot')
```

```
def barplot(var_select, x_no_numeric) :
    tmp1 = clean_df[(clean_df['Attrition'] != 0)]
    tmp2 = clean_df[(clean_df['Attrition'] == 0)]
    tmp3 = pd.DataFrame(pd.crosstab(clean_df[var_select],clean_df['Attrition']), )
    tmp3['Attr%'] = tmp3[1] / (tmp3[1] + tmp3[0]) * 100
    if x_no_numeric == True :
        tmp3 = tmp3.sort_values(1, ascending = False)
    color=['lightskyblue','gold' ]
    trace1 = go.Bar(
    x=tmp1[var_select].value_counts().keys().tolist(),
    y=tmp1[var_select].value_counts().values.tolist(),
    name='Yes_Attrition',opacity = 0.8, marker=dict(
    color='gold',line=dict(color='#000000',width=1)))

    trace2 = go.Bar(
    x=tmp2[var_select].value_counts().keys().tolist(),
    y=tmp2[var_select].value_counts().values.tolist(),
    name='No_Attrition', opacity = 0.8, marker=dict(
    color='lightskyblue',
    line=dict(color='#000000',width=1)))

    trace3 = go.Scatter(
    x=tmp3.index, y=tmp3['Attr%'], yaxis = 'y2',
    name='% Attrition', opacity = 0.6,
    marker=dict(color='black', line=dict(color='#000000',width=0.5)))
    layout = dict(title = str(var_select), xaxis=dict(), yaxis=dict(title= 'Count'),
    yaxis2=dict(range= [0,75],overlaying= 'y', anchor= 'x', side= 'right',zeroline=False,
    showgrid= False,title= '% Attrition'))
    fig = go.Figure(data=[trace1, trace2, trace3], layout=layout)
    py.iplot(fig)
```

```
def plot_pie(var_select) :
    colors = ['gold', 'lightgreen', 'lightcoral', 'lightskyblue', 'lightgrey', 'orange', 'white', 'lightpink']
    trace1 = go.Pie(values = attrition[var_select].value_counts().values.tolist(),
    labels = attrition[var_select].value_counts().keys().tolist(),
    textfont=dict(size=15), opacity = 0.8, hoverinfo = "label+percent+name",
    domain = dict(x = [0,.48]), name = "attrition employes",
    marker = dict(colors = colors, line = dict(width = 1.5)))
    trace2 = go.Pie(values = no_attrition[var_select].value_counts().values.tolist(),
    labels = no_attrition[var_select].value_counts().keys().tolist(),
    textfont=dict(size=15), opacity = 0.8,
    hoverinfo = "label+percent+name", marker = dict(colors = colors, line = dict(width = 1.5)),
    domain = dict(x = [.52,1]), name = "Non attrition employes" )
    layout = go.Layout(dict(title = var_select + " distribution in employes attrition ",
    annotations = [dict(text = "Yes_attrition", font = dict(size = 13),
    showarrow = False, x = .22, y = 0.1),
    dict(text = "No_attrition", font = dict(size = 13), showarrow = False, x = .8, y = .1)]))
    fig = go.Figure(data = [trace1,trace2],layout = layout)
    py.iplot(fig)
```