Corporate Employee Attrition Analysis

A PROJECT COMPONENT REPORT

Submitted by

Sanjay Dass.M	**(Reg.No.110819104031)**
Boobalan.C	**(Reg.No.110819104003)**
Yuvaraj kumar.J.V	**(Reg.No.110819104040)**
Manikandan.S	**(Reg.No.110819104021)**
Raj kumar.A	**(Reg.No.110819104027)**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

JAYA ENGINEERING COLLEGE,THIRUNINRAVUR
(Affiliated to Anna University, Chennai)

TABLE OF CONTENTS

1. Introduction

For our IBM Project, we chose Data Analytics as our domain, for the Nalaiya Thiran initiative. Our topic is Corporate Employee Attrition

Attrition refers to the reduction of strength or effectiveness in an organisation, i.e., employees suddenly resigning from the post due to their own reasons, which leads to the organisation not being able to complete their due work timely. In a sense, it represents the lack of competency in a company to retain their employees when necessary.

We intend to analyse such organisation's employee data and provide them with a solution for preventing such happenings, and if possible, be able to even motivate said employees to work more efficiently.

1.1 Project Overview

- To identify and retain experienced, talented and interested employees
- Understanding employee's interest or lack thereof in order to provide them deserving raise and incentives for further progress
- Refers to the techniques implemented by the management to help the employees stay with the organisation for a longer period

1.3 Purpose

The purpose of our project is to help organisations to retain their employees within, and provide them with solutions which offer proper incentives for the employees to work committedly even further.

2. LITERATURE SURVEY

2.Existing Problem

More along the lines of prediction, based on past behaviour and choices, probably effecting the organisation as well

2.1 References

1.Machine Learning Approach for Employee Attrition Analysis
Dr. R. S. Kamath | Dr. S. S. Jamsandekar | Dr. P. G. Naik ,Published in International Journal of Trend in Scientific Research and Development (ijtsrd)

2.From Big Data to Deep Data to support people analytics for employee attrition prediction, NesrineBen Yahia, Hlel Jihen, Ricardo Colomo Palacio

3.Investigation of early career teacher attrition(ECT) and the impact of induction programsin Western Australia, Janine E.Wyatt, MichaelO'Neill

4. EMPLOYEE ATTRITION PREDICTION USING DEEP NEURAL NETWORK, Salah Al-Darraji, Dhafer G. Honi , Francesca Fallucchi, Ayad I. Abdulsada, Romeo Giuliano and Husam A. Abdulmalik

2.2 Problem state Definition

Customer Problem Statement Template:

Create a problem statement to understand your customer's point of view. The Customer Problem Statement template helps you focus on what matters to create experiences people will love.
A well-articulated customer problem statement allows you and your team to find the ideal solution for the challenges your customers face. Throughout the process, you'll also be able to empathize with your customers, which helps you better understand how they perceive your product or service.

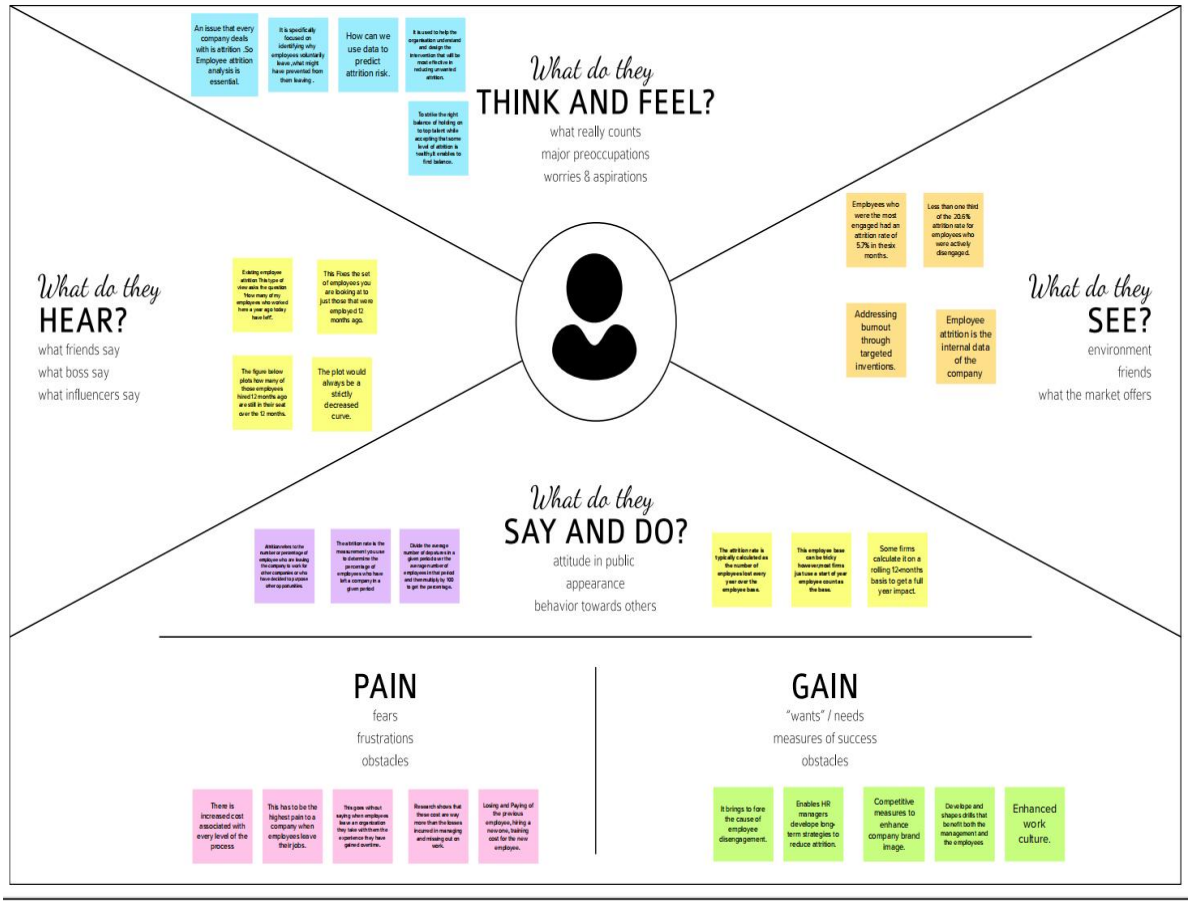
I am	Describe customer with 3-4 key characteristics - *who are they?*	Describe the customer and their attributes here
I'm trying to	*List their outcome or "job" the care about - what are they trying to achieve?*	List the thing they are trying to achieve here
but	Describe what problems or barriers stand in the way – *what bothers them most?*	Describe the problems or barriers that get in the way here
because	Enter the "root cause" of why the problem or barrier exists – *what needs to be solved?*	Describe the reason the problems or barriers exist
which makes me feel	Describe the emotions from the customer's point of view – *how does it impact them emotionally?*	Describe the emotions the result from experiencing the problems or barriers

Reference link : https://miro.com/

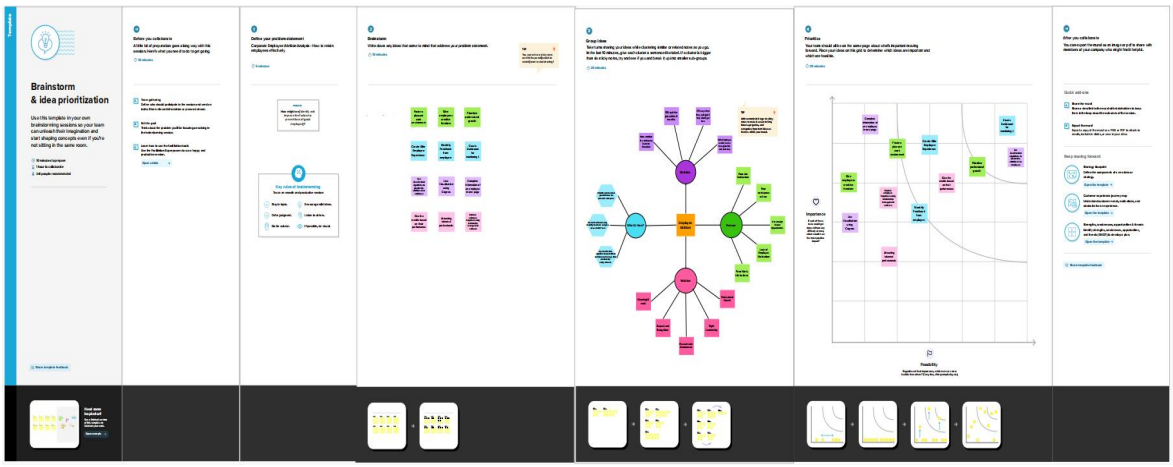
Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Employee	Work very hard	I am not able to achieve much for my dedication and hard work towards the organization.	Teams are not built according to personalization, right people are not hired and no flexibilities are offered.	Frustated.
PS-2	Employee	Innovator	This work is good for me but it is not convenient for me	It is because the innovator can't settle for media critic life.	This job makes me feel that the time is being wasted.

IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



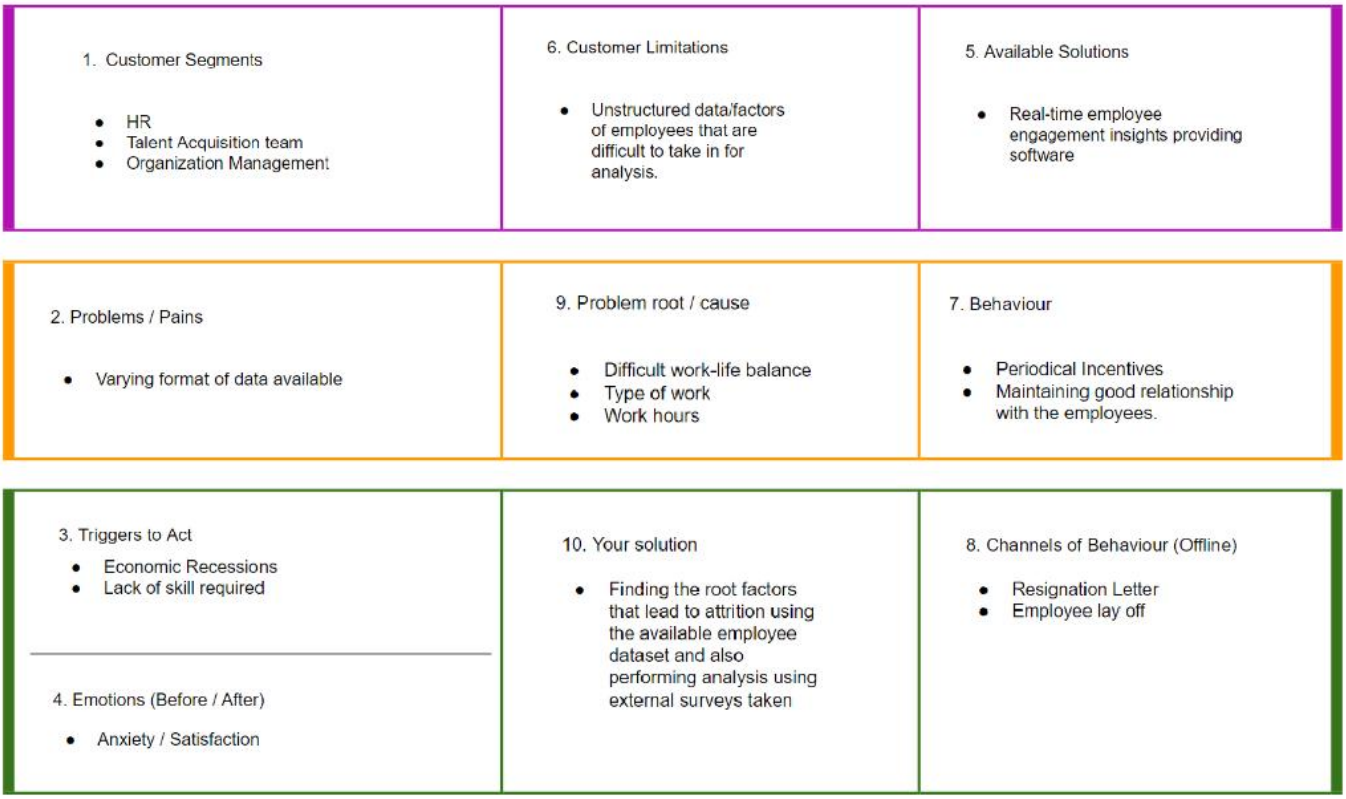
3.2 Ideation & Brainstorming



3.3 Proposed Solution

S No	Parameter	Description
1	Problem Statement (Problem to be solved)	Corporate Employee Attrition Analysis - How to retain employees effectively
2	Idea / Solution description	Prioritize the professional growth & give the pleasant workspace and use some classification algorithm to predict their retention and manage their relationship using this software.
3	Novelty / Uniqueness	Employee attrition prediction is specifically focused on identifying why employees voluntarily leave, what might have prevented them from leaving, and how we can use data to predict attrition risk.
4	Social Impact / Customer Satisfaction	Employee's attrition has huge impact on company, recruiting new employees and investing time to train them is increased. Losing a good employee creates a negative impact of profit on the company.
5	Business Model (Revenue Model)	The business is struggling with employee attrition. This software will be helpful to analyze the workforce trends and find the root cause of Attrition.
6	Scalability of the Solution	The dashboard is scalable for the companies when their employee's dataset is used for analysis. The model can successfully predict the futuristic approach and suggests preventive measures.

3.4 Problem Solution fit



4.REQUIREMENT ANALYSIS

4.1 Functional requirement

Functional Requirements:

Following are the functional requirements of the proposed solution.

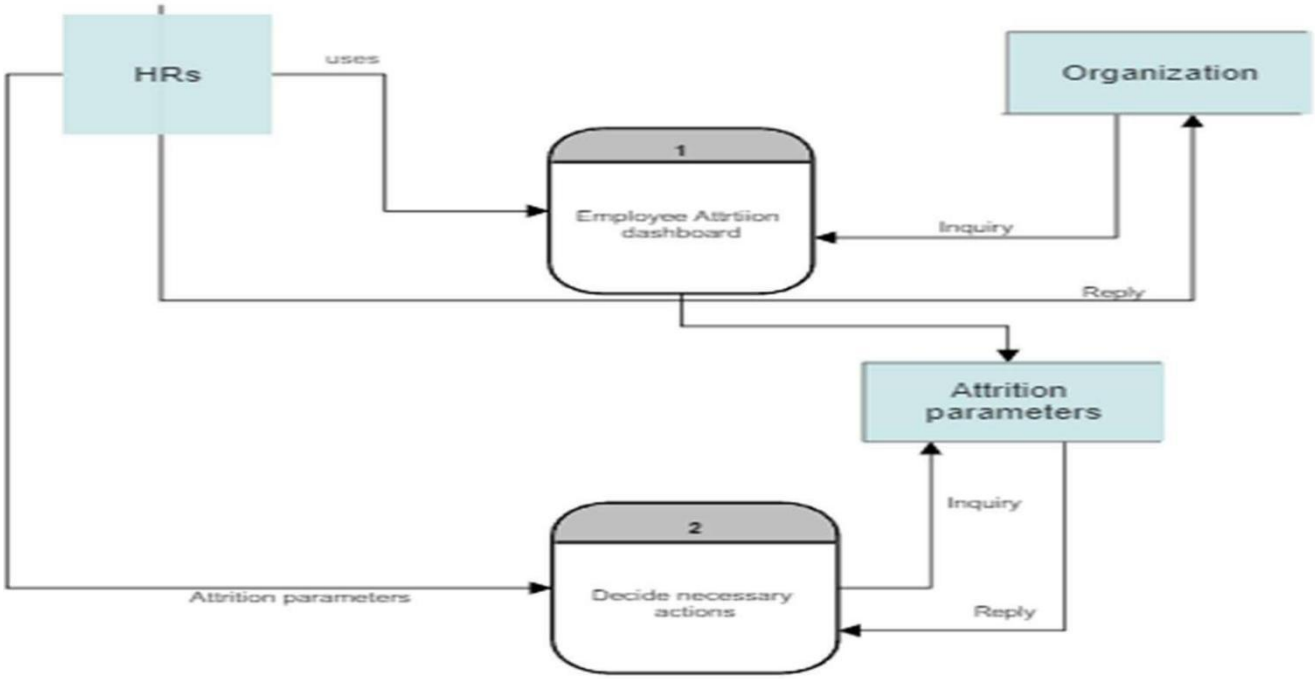
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form
FR-2	User Confirmation	Confirmation via Email
FR-3	User Authentication	Authenticate the user's attempt to login using the database
FR-4	Retention analysis	Employee attrition analysis by sentiment, work environment, daily contribution etc.
FR-5	Employee management	Validating and managing the registered employee details.
FR-6	Progress management	Add the progress of each employee to the company.
FR-7	Predict button	The predict route is used for prediction and it contains all the codes which are used for predicting our results. Firstly, inside launch function we are having the following things: • Getting our input and storing it . • Select the necessary attributes for the prediction. • Creating model. • Predicting our results . • Showcase the results with the help of dashboard. • Finally run the application.

4.2 Non-Functional requirements

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	This Data Visualization shall be easy to use for all users withminimal instructions. 100% of the languages on the graphical user interface (GUI) shallbe intuitive and understandable by non-technical users.
NFR-2	**Security**	The user of the system should be provided the surety that their account details are secure.
NFR-3	**Reliability**	The Link shall be operable in all conditions. The system must be less prone to errors.
NFR-4	**Performance**	The performance of the system must assist the system's quality.
NFR-5	**Portability**	The link shall be 100% portable to all operating platforms. Therefore, this link should not depend on the different operating systems.
NFR-6	**Scalability**	The system must be able to handle an increase in workload without performance degradation.

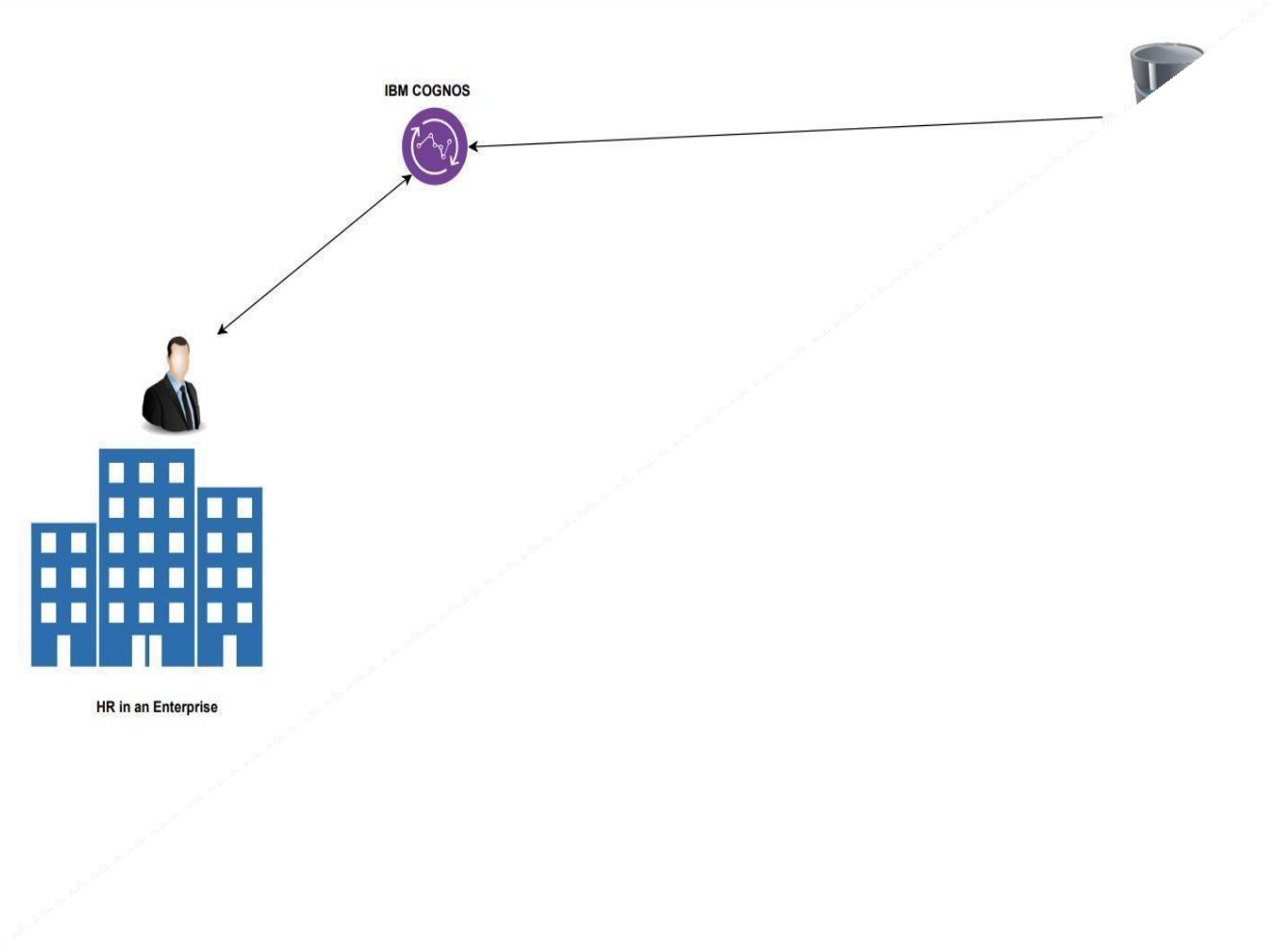
PROJECT DESIGN

5.1 Data Flow Diagrams

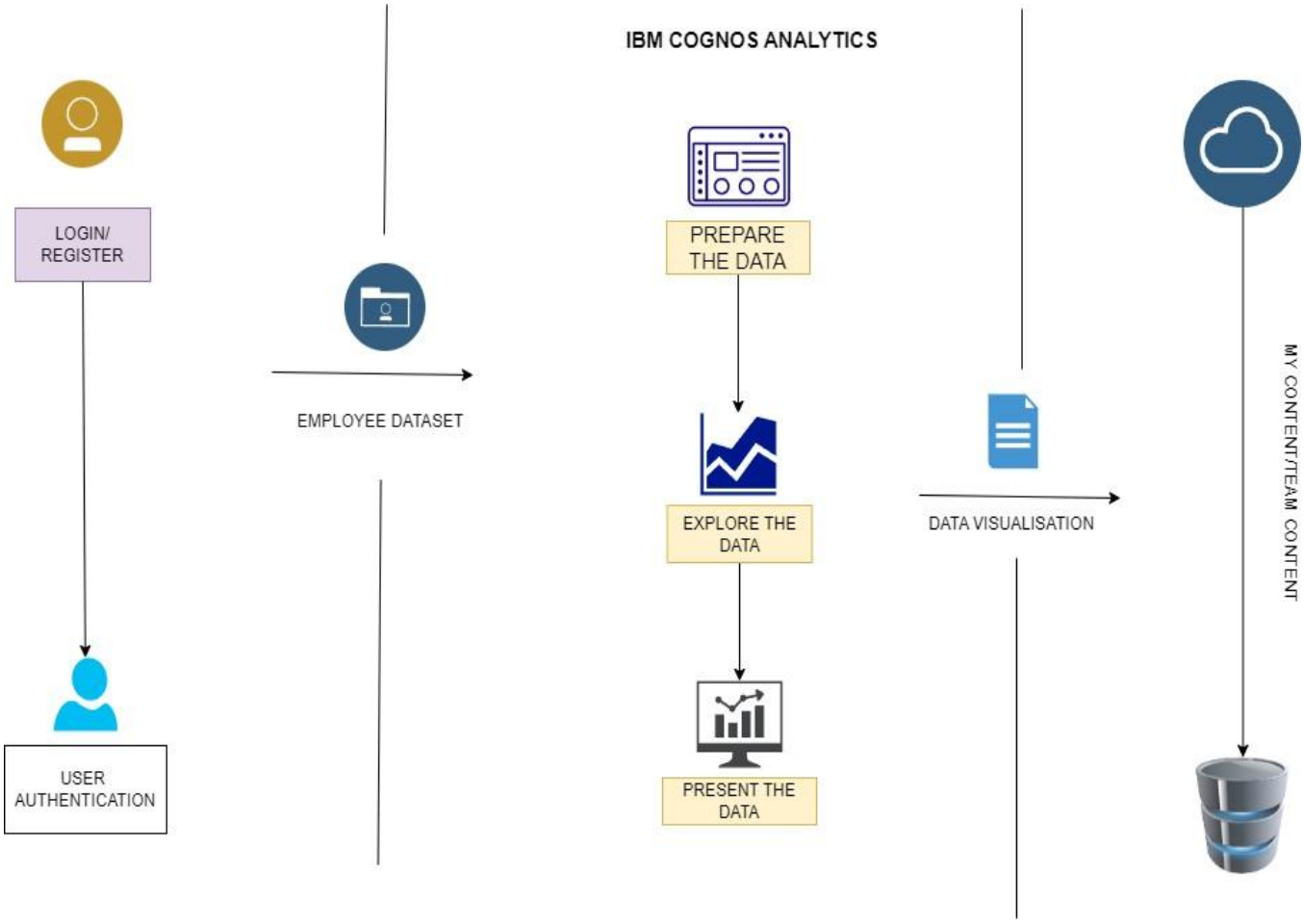


5.2 Solution & Technical Architecture

Solution Architecture



5.2.1Technical Architecture



5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Employees	Registration	USN-1	The employees can register to be a part of the organization	I can access my account / dashboard	High	Sprint-1
		USN-2	As an employee, I will receive confirmation email	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As an employee, I can register for the application through Gmail	I can get a verification link through email	Medium	Sprint-1
	Login	USN-4	As a employee, I can log into the application by entering email & password	I can enter the application	High	Sprint-2
	About	USN-5	I can view the Dashboard, Story and Report for attrition rates and determining the factors leading to them	I can get an idea about the project	Low	Sprint-2
	Launch	USN-7	As a HR, I can upload various analyzed parameters from the computer through link given in the pdf	I can choose any employee ('s all parameters) from my device	High	Sprint-2
	Link	USN-8	As a HR, I can review an employee's performance and offer appraisals biannually or Quarterly	I can view the employee's parameters on the dashboard along with the attrition rate.	High	Sprint-3
		USN-9	I can also upload csv format of employee retention parameters from cloud.	I can view the employee's parameters on the dashboard along with the attrition rate.	Medium	Sprint-3

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint Planning

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Dashboard	USN-1	As a user, I give the details of the employees working in our organization for the attrition detail.	5	High	SanjayDass, Boobalan,Yuvaraj manikandan,raj kumar
Sprint-1		USN-2	As an Analyst, I will check the dataset and perform exploratory data analysis in Cognos Analytics	3	High	SanjayDass, Boobalan,Yuvaraj manikandan,raj kumar
Sprint-2	Report	USN-3	As a user, I want Simpler limited number of visualizations that report a particular event	2	Low	SanjayDass, Boobalan,Yuvaraj manikandan,raj kumar
Sprint-2		USN-4	As an Analyst, I will use Cognos Analytics to generate a report	3	Medium	SanjayDass, Boobalan,Yuvaraj manikandan,raj kumar
Sprint-3	Story	USN-5	As a user, I can only understand the Analysis in animated presentation of dataset	3	Medium	SanjayDass, Boobalan,Yuvaraj manikandan,raj kumar
Sprint-3		USN-6	As an Analyst, I use Cognos Analytics to create an animated presentation (Story) of the dataset	3	Medium	SanjayDass, Boobalan,Yuvaraj manikandan,raj kumar
Sprint-4	Predictive Analysis	USN-7	As a user, I want to predict the attrition rate of the company from the dataset	5	Medium	SanjayDass, Boobalan,Yuvaraj manikandan,raj kumar
Sprint-4		USN-8	As an Analyst, I will perform Prediction Analysis by utilizing various libraries in python	3	High	SanjayDass, Boobalan,Yuvaraj manikandan,raj kumar

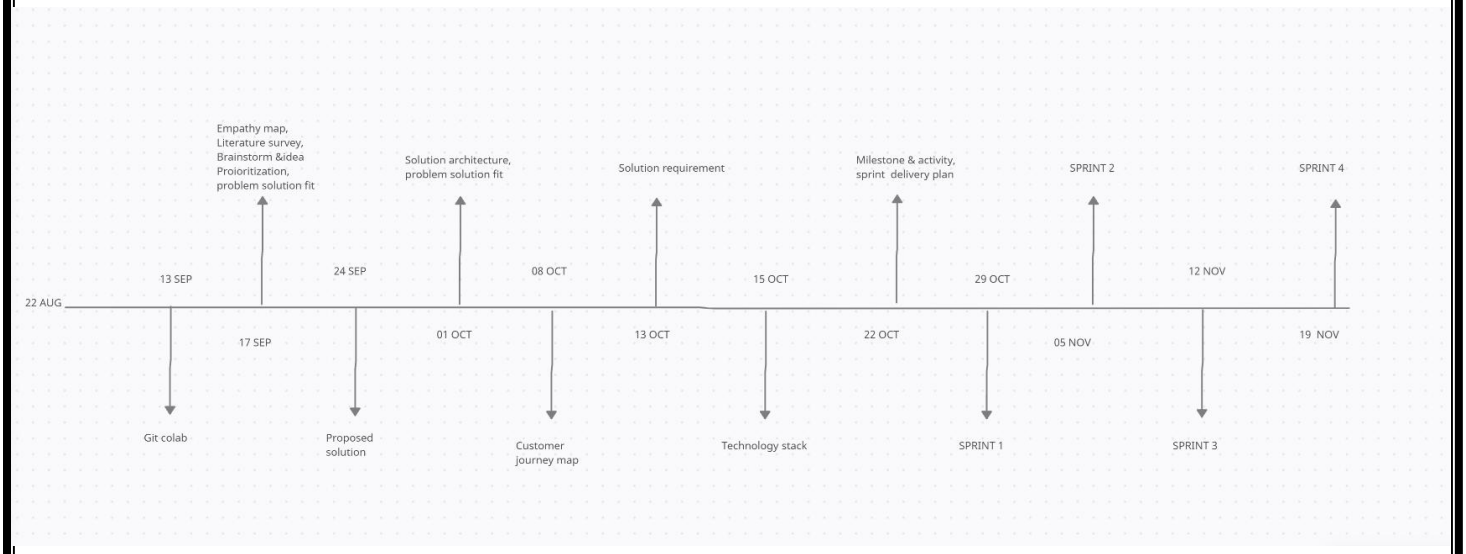
`Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	5	6 Days	24 Oct 2022	29 Oct 2022	5	29 Oct 2022
Sprint-2	5	6 Days	31 Oct 2022	05 Nov 2022	5	05 Nov 2022
Sprint-3	5	6 Days	07 Nov 2022	12 Nov 2022	5	12 Nov 2022
Sprint-4	5	6 Days	14 Nov 2022	19 Nov 2022	5	19 Nov 2022

6.2 Milestone and Activity List

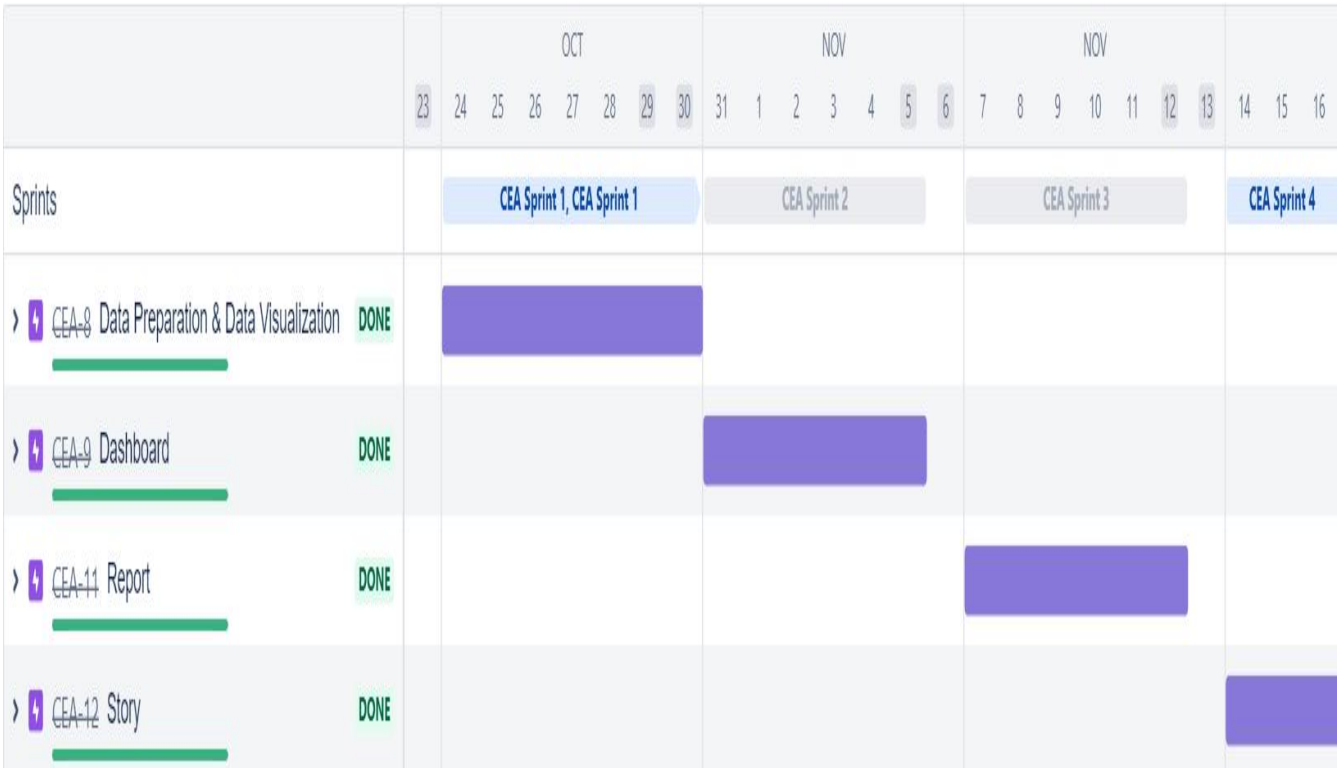
Activity number	Activity name	Detailed activity description	Assigned to
1	Preparation Phase	• Access the resources (courses) in project dashboard • Access the guided project workspace • Create GitHub account & collaborate with Project Repository in project workspace • Set-up the Laptop / Computers based on the prerequisites for each technology track	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
2	**Ideation Phase**		
2.1	Literature survey	Literature survey on the selected project & Information Gathering	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
2.2	Define a problem statement	Prepare the list of problem statements to understand the user needs	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
2.3	Empathy Map	Preparation of Empathy Map Canvas to capture the user Pains & Gains	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
2.4	Brainstorm & idea prioritization	List the ideas by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
3	**Project DesignPhase -I**		
3.1	Proposed Solution	Preparation of proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
3.2	Problem Solution Fit	Prepared problem is analyzed and make effective solutions for the problem	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
3.3	Solution Architecture	Prepare an architecture for solution	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
4	**Project Design Phase-II**		
4.1	Requirement Analysis	Prepare the Functional Requirement and Non- Functional Document	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
4.2	Customer Journey	Preparation of customer journey maps to understand the user interactions & experiences with the application (entry to exit)	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
4.3	Data Flow Diagrams	Prepare a Data Flow Diagram for Project use level0 (Industry Standard)	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
4.4	Technology Architecture	Prepare Technology Architecture of the solution	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar

Activity number	Activity name	Detailed activity description	Assigned to
5	**Project PlanningPhase**		
5.1	Milestones & Tasks	Prepare Milestone & Activity List	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
5.2	Sprint Schedules	Prepare Sprint Delivery Plan	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
Activity number	**Activity name**	**Detailed activity description**	**Assigned to**
6	**Project DevelopmentPhase**		
6.1	Coding & Solutioning	Sprint-1 Delivery: Develop the Code, Test and push it to GitHub.	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
6.2	Acceptance Testing	Sprint-2 Delivery: Develop the Code, Test and push it to GitHub. Sprint-3 Delivery: Develop the Code, Test and push it to GitHub.	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar
6.3	Performance Testing	Sprint-4 Delivery: Develop the Code, Test and push it to GitHub.	Sanjay Dass,Boobalan,Yuvaraj manikandan,raj kumar

6.2 Sprint Delivery Schedule



6.3 Reports from JIRA



7.User Acceptance Testing

Acceptance Testing
UAT Execution & Report Submission

Date	12-nov-2022
Team ID	PNT2022TMID36382
Project Name	Corporate Employee Attrition Analytics
Maximum Marks	4 Marks

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [Product Name] project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	1	2	1	0	3
Duplicate	1	0	0	0	1
External	2	0	0	1	3
Fixed	7	2	3	0	12
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	1	0	0	1
Totals	11	5	6	2	23

3. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
CSV File upload	2	0	0	2
IBM Cognos Dashboard embedment	5	2	0	3
Interaction charts	4	0	0	4
Correlations	1	0	0	1
EDA	1	0	0	1

Results

7.2 Performance Metrics

S.No.	Parameter	Screenshot / Values
1.	Dashboard design	No of Visualizations / Graphs - 11
2.	Data Responsiveness	Dynamic Responses
3.	Amount Data to Rendered (DB2 Metrics)	Connect anonymously Schema = RSW27482 (3/3)
4.	Utilization of Data Filters	Attrition= no, exclusively for most cases
5.	Effective User Story	No of Scene Added - 3
6.	Descriptive Reports	No of Visulizations / Graphs - 4

8.Advantages & Disadvantages

Advantages

1. Retaining of talented employees
2. Constant incentives lead to more productive work from employees
3. Much livelier work environments
4. Loyalty benefits
5. Satisfied employees with improved worklife balance
6. Provides accurate appraisal methods

Disadvantages

1. Dependency on third party analysts
2. Employee details privacy concern
3. Destructures the classic delegation of authority
4. Need for an cognos account

9.CONCLUSION

While employee attrition isn't necessarily a bad thing, you should do your best to monitor the pulse of your workplace to stop it in its tracks as early as you can. Similar to turnover, it's an important metric that tells a lot about your employer branding, hiring practices, and overall workplace culture.

10.Future Scope

The ever enhancing, more visual and better representation of unstructured data. It could also be integrated into custom applications within individual organisation. As the use of such techniques increases and more better solutions are identified, after a certain point, the underlying analysing pattern can even be automated.

11.Appendix

Source code

In [ ]:

In [1]:
```
import math, time, random, datetime

# data analysis and wrangling
import pandas as pd
import numpy as np
from pandas_profiling import ProfileReport
```

In [2]:
```
# visualization
import seaborn as sns
import matplotlib.pyplot as plt
plt.style.use('seaborn-whitegrid')

#import for interactive plotting
import plotly.offline as py
py.init_notebook_mode(connected=True)
import plotly.graph_objs as go
import plotly.tools as tls
import plotly.figure_factory as ff
from plotly.subplots import make_subplots
%matplotlib inline
```

In [3]:
```
# Preprocessing
from sklearn.preprocessing import OneHotEncoder, LabelEncoder, label_binarize, StandardScaler
```

In [4]:
```
pip install catboost
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting catboost
  Downloading catboost-1.1.1-cp37-none-manylinux1_x86_64.whl (76.6 MB)
     |████████████████████████████████| 76.6 MB 1.2 MB/s
Requirement already satisfied: pandas>=0.24.0 in /usr/local/lib/python3.7/dist-packages (from catboost) (1.3.5)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from catboost) (1.15.0)
Requirement already satisfied: plotly in /usr/local/lib/python3.7/dist-packages (from catboost) (5.5.0)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-packages (from catboost) (3.2.2)
Requirement already satisfied: graphviz in /usr/local/lib/python3.7/dist-packages (from catboost) (0.10.1)
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages (from catboost) (1.7.3)
Requirement already satisfied: numpy>=1.16.0 in /usr/local/lib/python3.7/dist-packages (from catboost) (1.21.6)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from pandas>=0.24.0->catboost) (2022.5)
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dist-packages (from pandas>=0.24.0->catboost) (2.8.2)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib->catboost) (3.0.9)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from matplotlib->catboost) (1.4.4)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages (from matplotlib->catboost) (0.11.0)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packages (from kiwisolver>=1.0.1->matplotlib->catboost) (4.1.1)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.7/dist-packages (from plotly->catboost) (8.1.0)
Installing collected packages: catboost
Successfully installed catboost-1.1.1
```

In [5]:
```
# machine Learning
from sklearn import model_selection, tree, preprocessing, metrics, linear_model
from sklearn.metrics import confusion_matrix,classification_report
from sklearn.svm import SVC, LinearSVC
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.linear_model import Perceptron,SGDClassifier,LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split,StratifiedKFold, GridSearchCV, learning_curve, cross_val_score
from catboost import CatBoostClassifier, Pool, cv
```

In [6]:
```
# ignore Warnings
import warnings
warnings.filterwarnings('ignore')
```

Import and Inspect Data

In [7]:
```
df = pd.read_csv("/content/Employee-Attrition.csv")
```

In [8]:
```
df.head()
```

Out[8]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	...	RelationshipSatisfaction	Standar
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	...	1	
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	...	4	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	...	2	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	...	3	
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	...	4	

5 rows × 35 columns

In [9]:
```
df.shape
```

Out[9]: (1470, 35)

Exploratory Data Analysis

In [ ]:
```
ProfileReport(df)
```

In [11]:
```
# drop the unnecessary coLumns
df.drop(['EmployeeNumber','Over18','StandardHours','EmployeeCount'],axis=1,inplace=True)
```

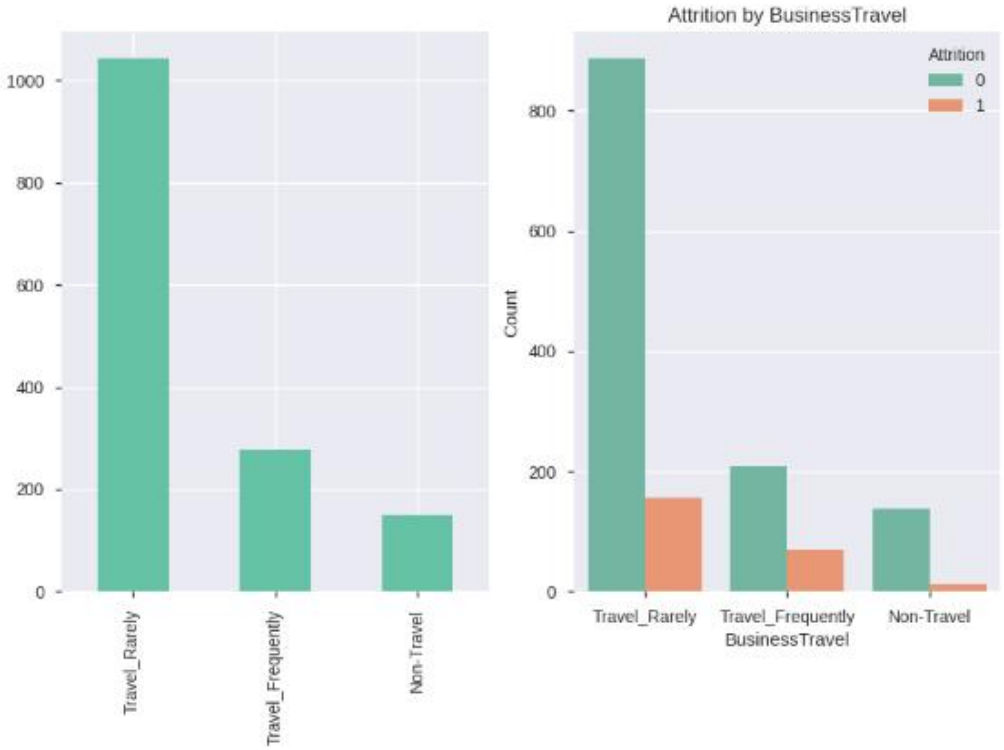
In [12]:
```
df['Attrition'] = df['Attrition'].apply(lambda x:1 if x == "Yes" else 0 )
df['OverTime'] = df['OverTime'].apply(lambda x:1 if x =="Yes" else 0 )
```

In [13]:
```
attrition = df[df['Attrition'] == 1]
no_attrition = df[df['Attrition']==0]
```

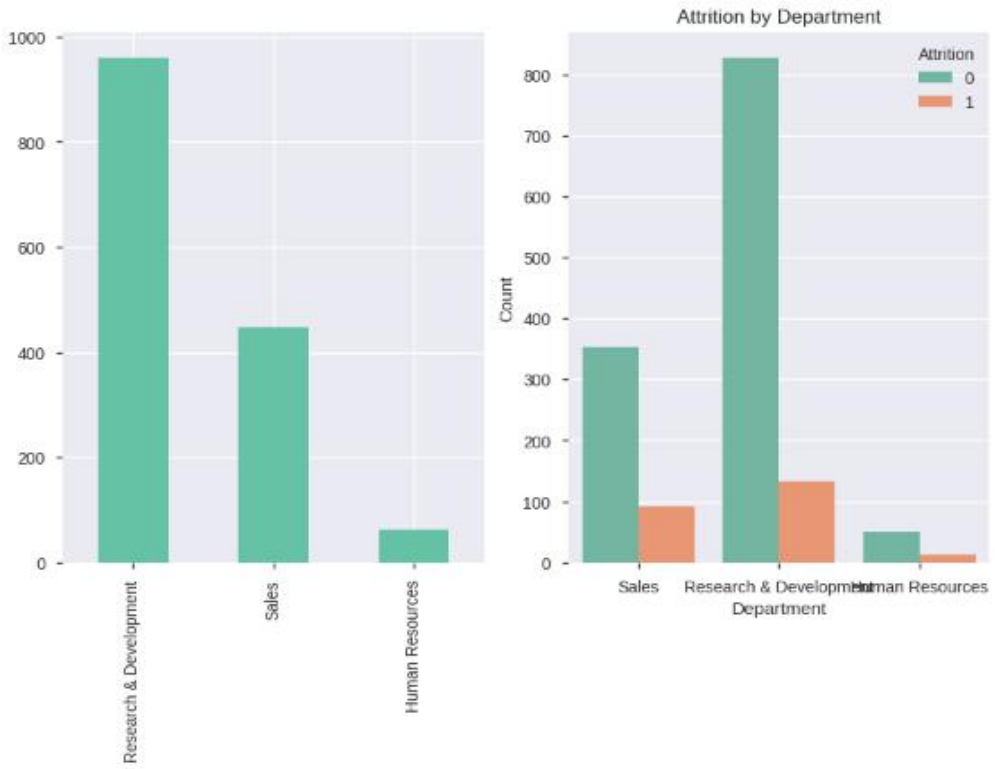
Visualization of Categorical Features

In [14]:
```
def categorical_column_viz(col_name):

    f,ax = plt.subplots(1,2, figsize=(10,6))

    # Count Plot
    df[col_name].value_counts().plot.bar(cmap='Set2',ax=ax[0])
    ax[1].set_title(f'Number of Employee by {col_name}')
    ax[1].set_ylabel('Count')
    ax[1].set_xlabel(f'{col_name}')

    # Attrition Count per factors
    sns.countplot(col_name, hue='Attrition',data=df, ax=ax[1], palette='Set2')
    ax[1].set_title(f'Attrition by {col_name}')
    ax[1].set_xlabel(f'{col_name}')
    ax[1].set_ylabel('Count')
```

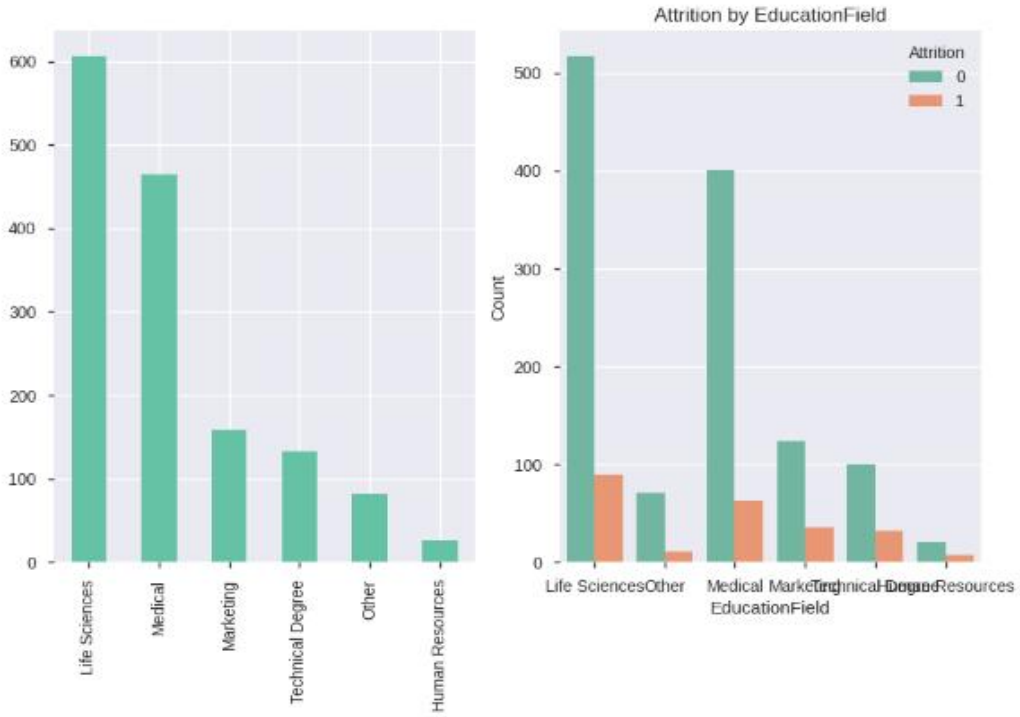
In [15]:
```
categorical_column_viz('BusinessTravel')
```



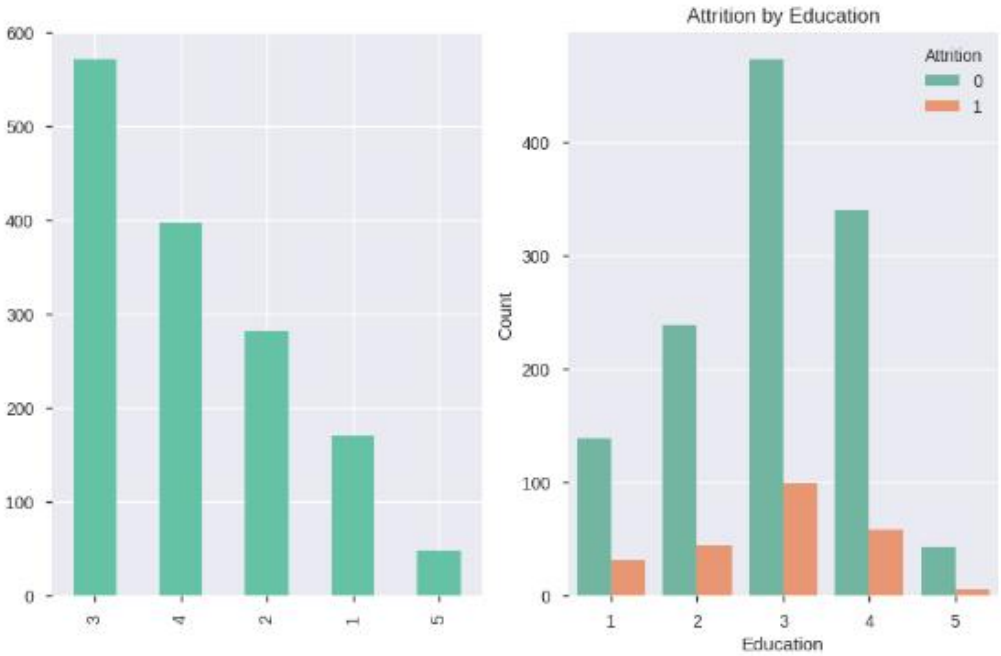
In [16]:
```
categorical_column_viz('Department')
```



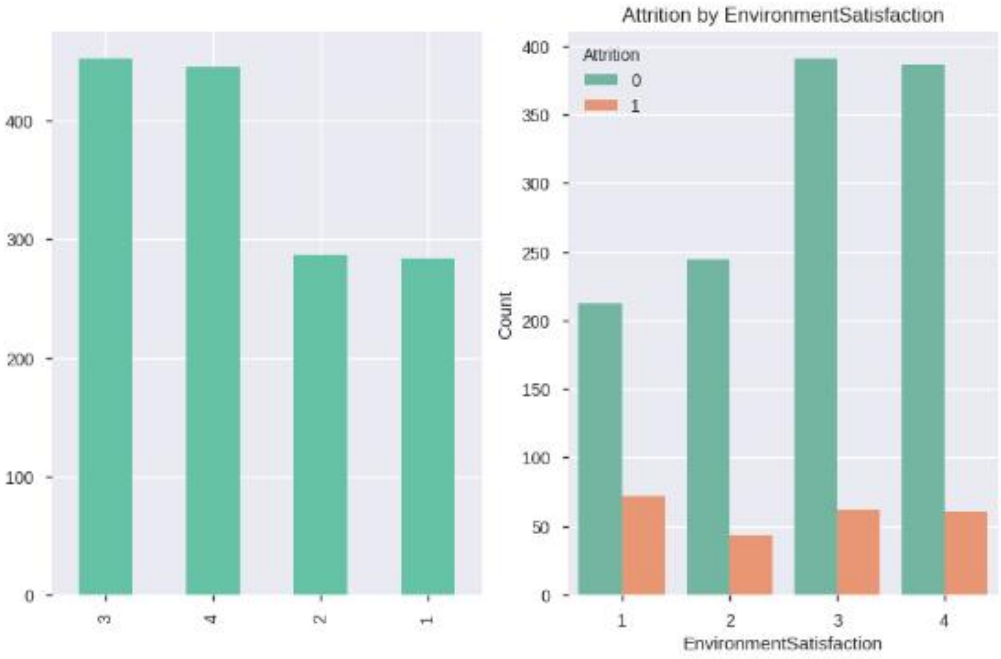
In [17]:
```
categorical_column_viz('EducationField')
```



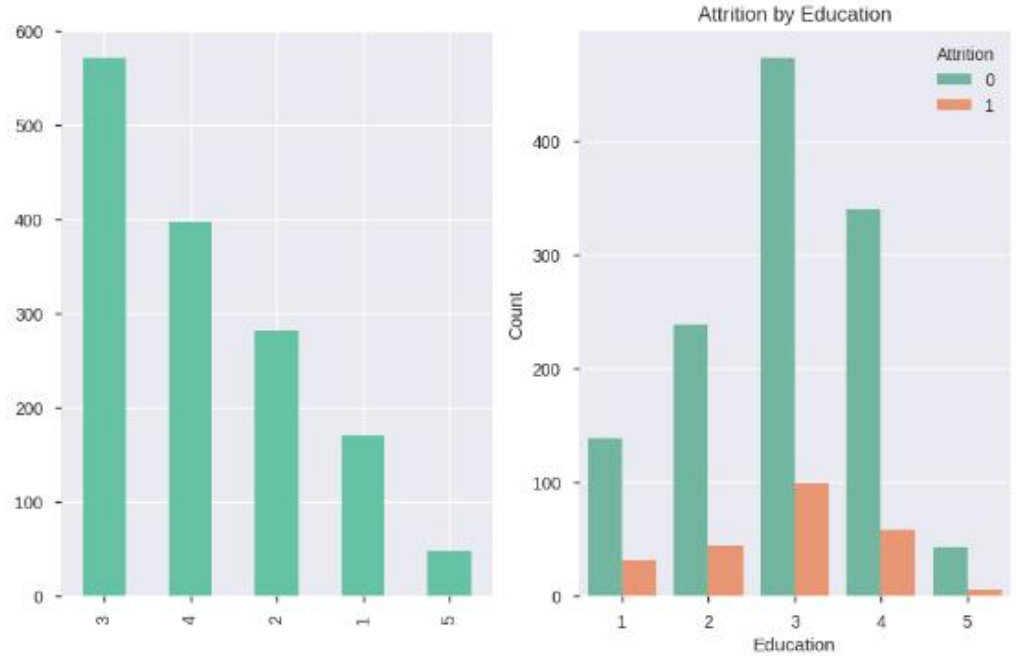
In [18]:
```
categorical_column_viz('Education')
```



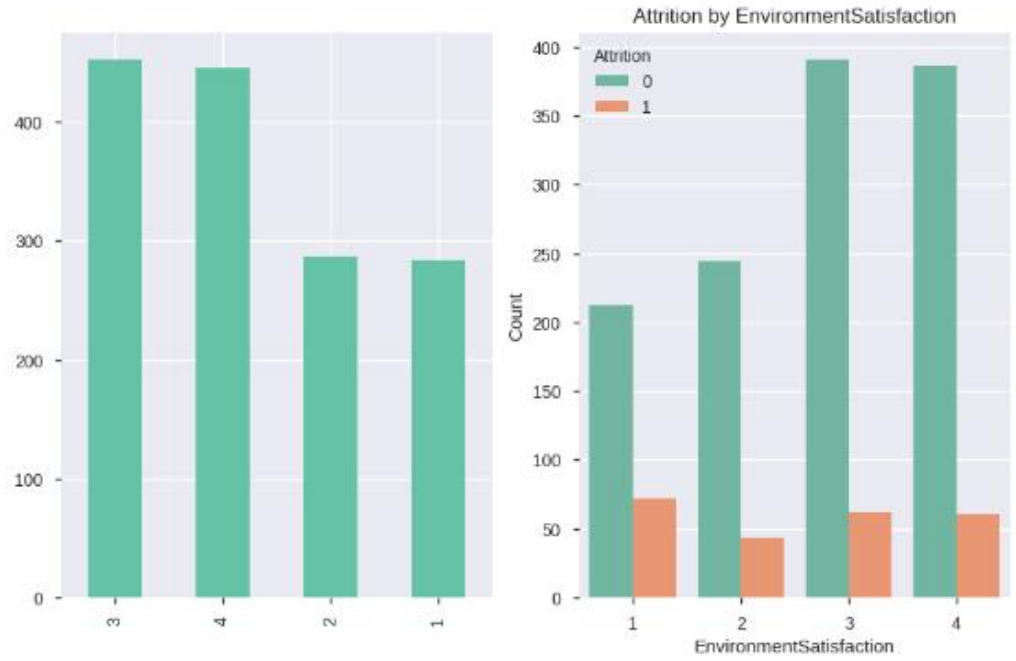
In [19]:
```
categorical_column_viz('EnvironmentSatisfaction')
```



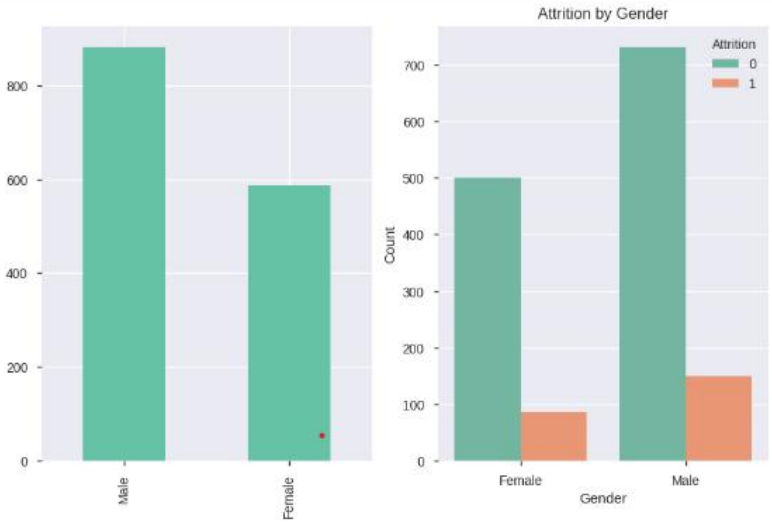
In [18]:
```
categorical_column_viz('Education')
```



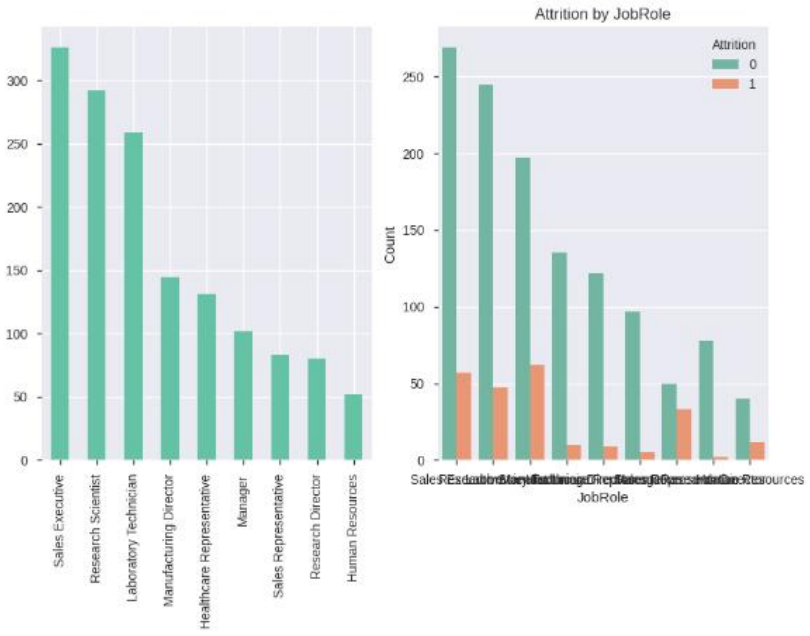
In [19]:
```
categorical_column_viz('EnvironmentSatisfaction')
```



In [20]:
```
categorical_column_viz('Gender')
```

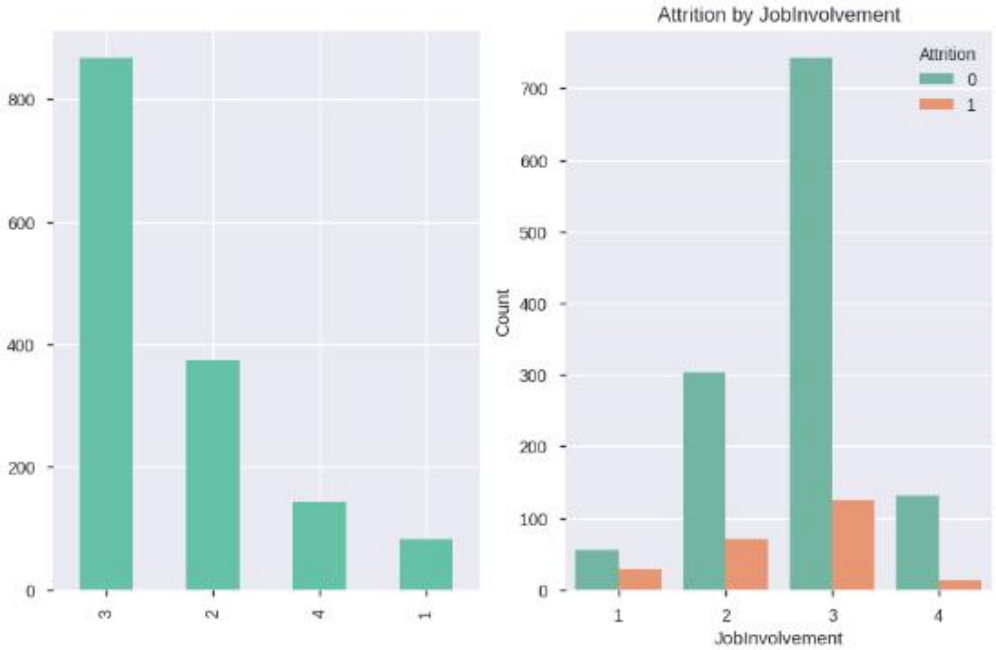


In [21]:
```
categorical_column_viz('JobRole')
```

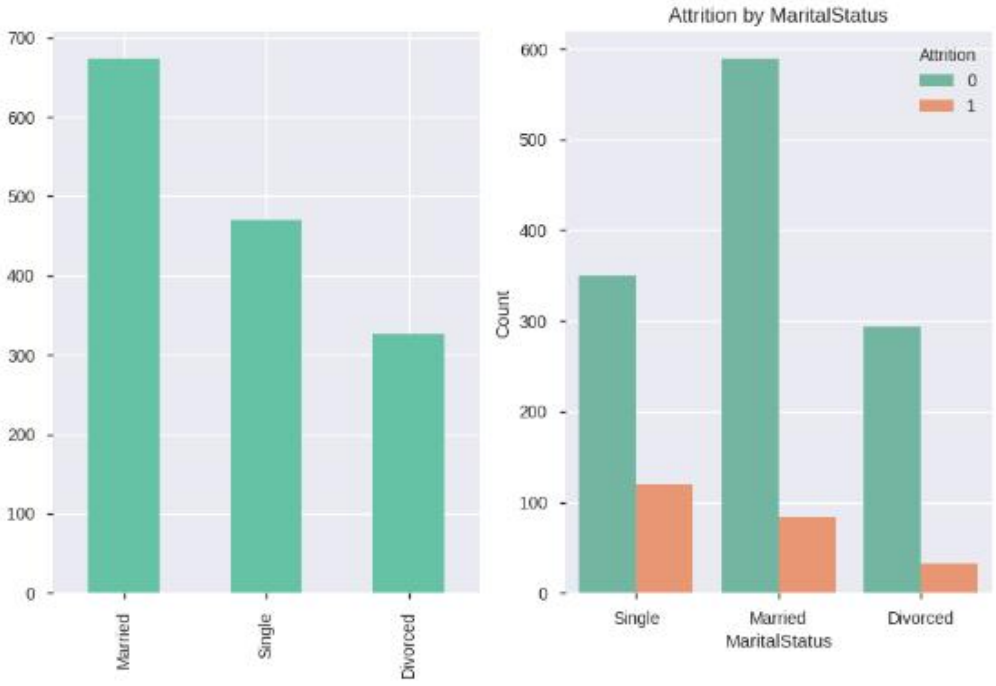


In [22]:

In [22]:
```
categorical_column_viz('JobInvolvement')
```

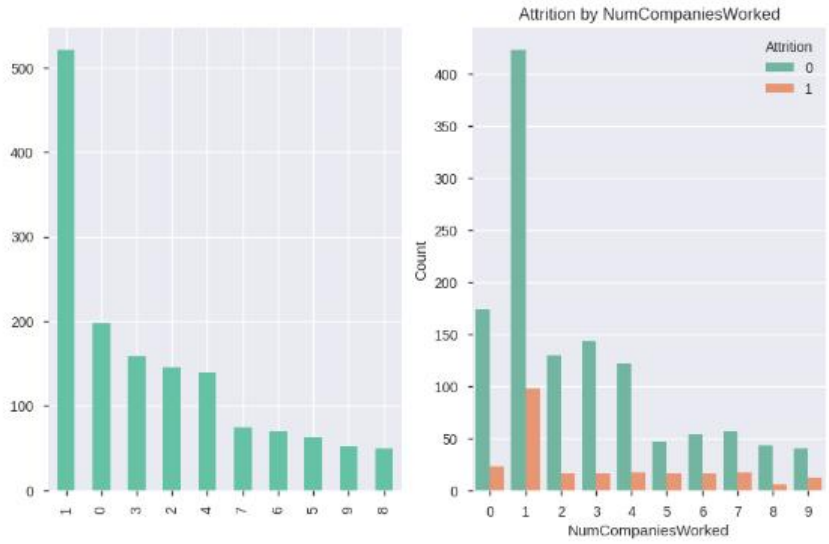


In [23]:
```
categorical_column_viz('MaritalStatus')
```

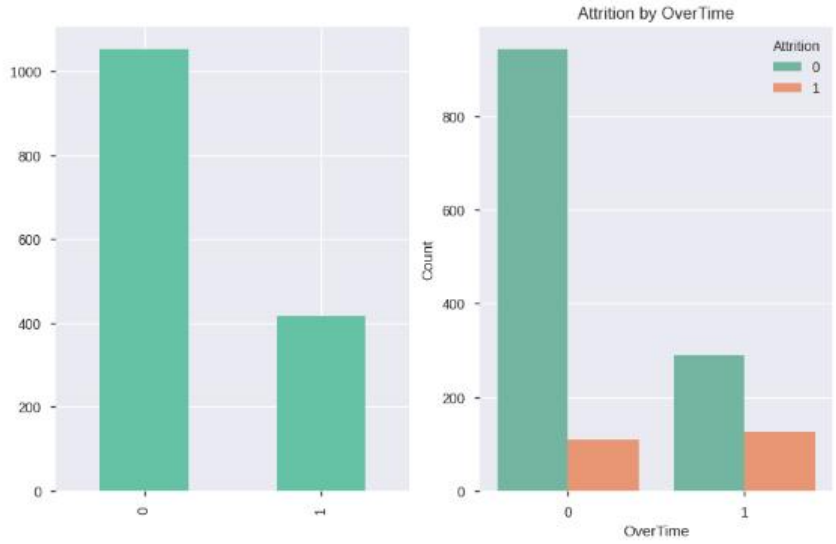


In [24]:
```
categorical_column_viz('NumCompaniesWorked')
```

In [24]:
```
categorical_column_viz('NumCompaniesWorked')
```

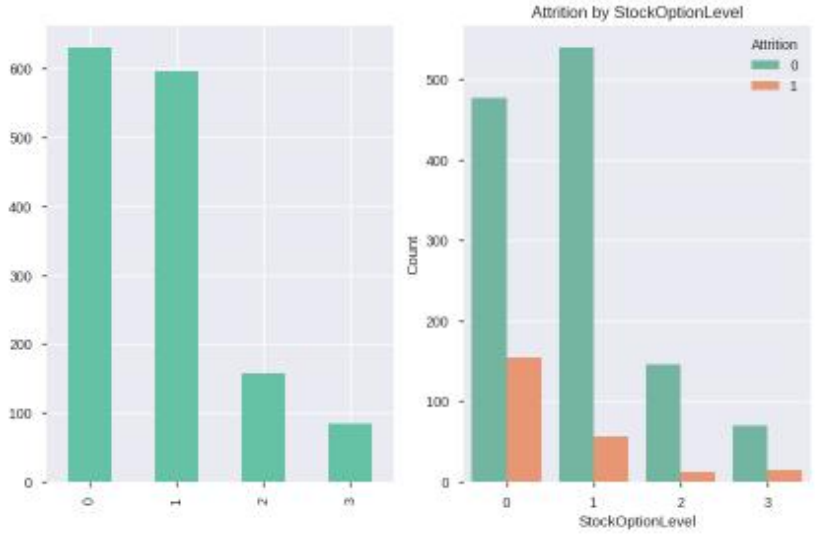


In [25]:
```
categorical_column_viz('OverTime')
```

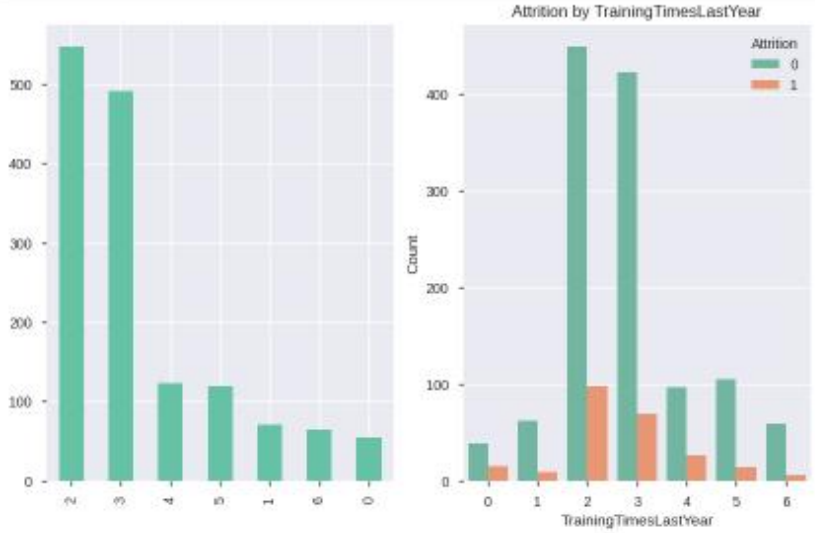


In [26]:
```
categorical_column_viz('StockOptionLevel')
```

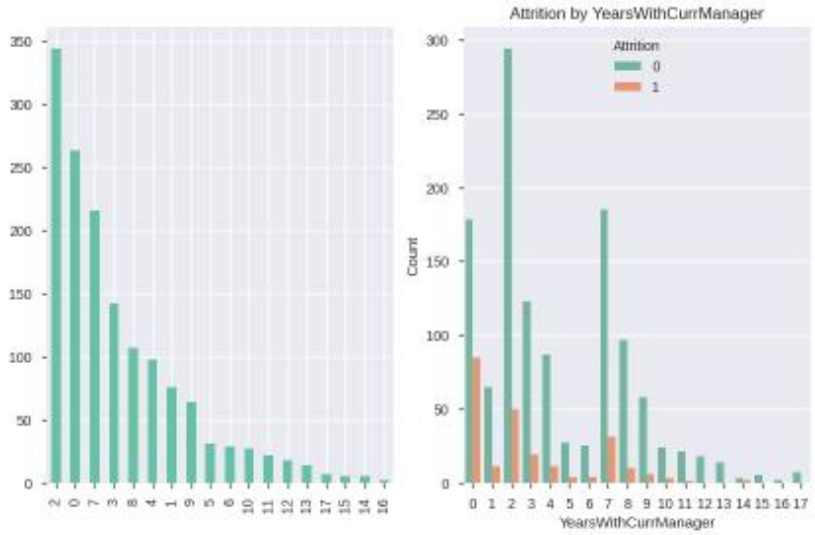
In [26]:
```
categorical_column_viz('StockOptionLevel')
```



In [27]:
```
categorical_column_viz('TrainingTimesLastYear')
```



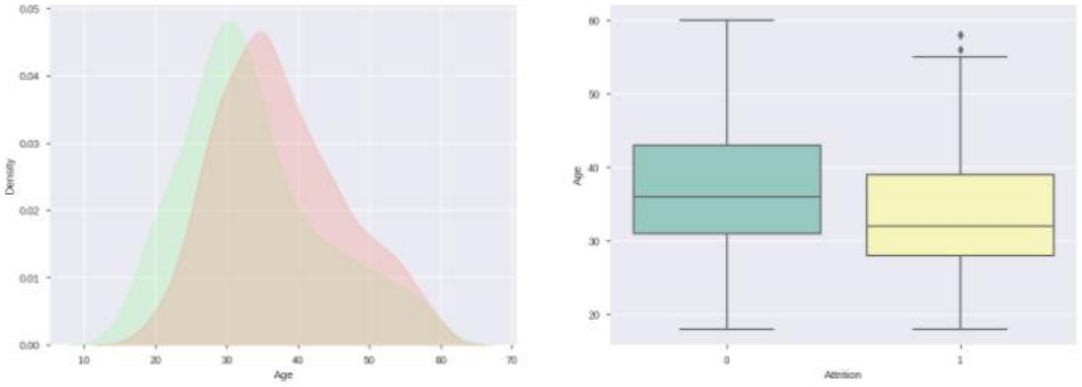
In [28]:
```
categorical_column_viz('YearsWithCurrManager')
```



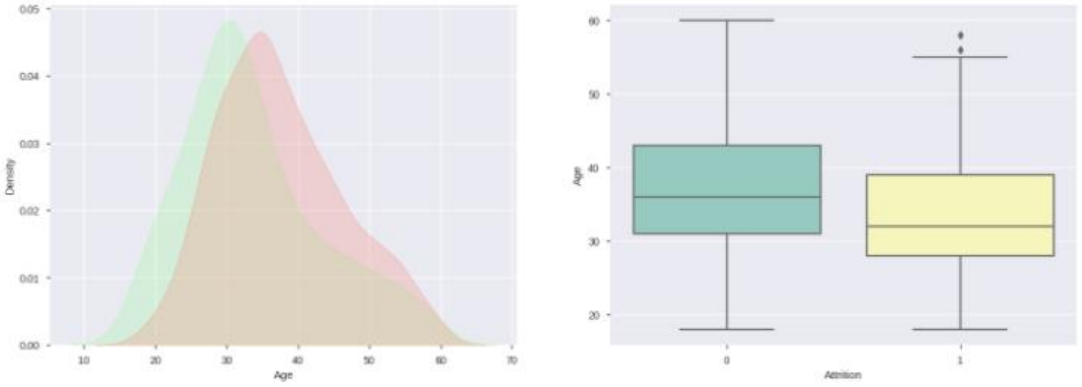
Visualization of Numerical Features

In [29]:
```
def numerical_column_viz(col_name):
    f,ax = plt.subplots(1,2, figsize=(18,6))
    sns.kdeplot(attrition[col_name], label='Employee who left',ax=ax[0], shade=True, color='palegreen')
    sns.kdeplot(no_attrition[col_name], label='Employee who stayed', ax=ax[0], shade=True, color='salmon')

    sns.boxplot(y=col_name, x='Attrition',data=df, palette='Set3', ax=ax[1])
```

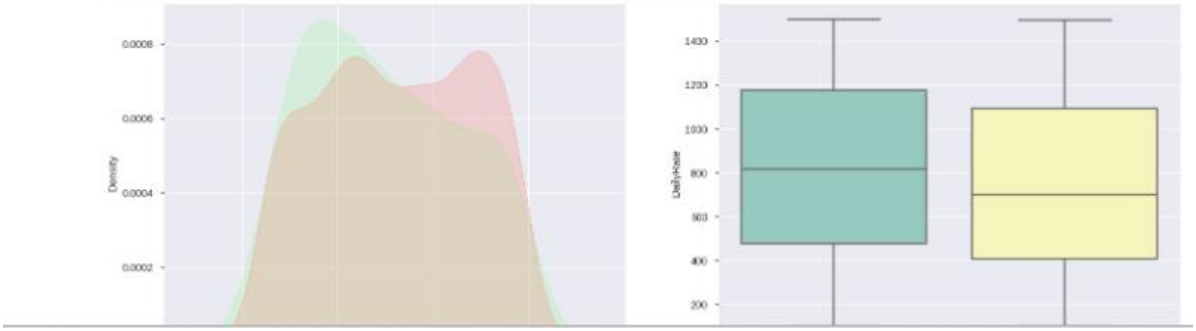
In [30]:
```
numerical_column_viz("Age")
```



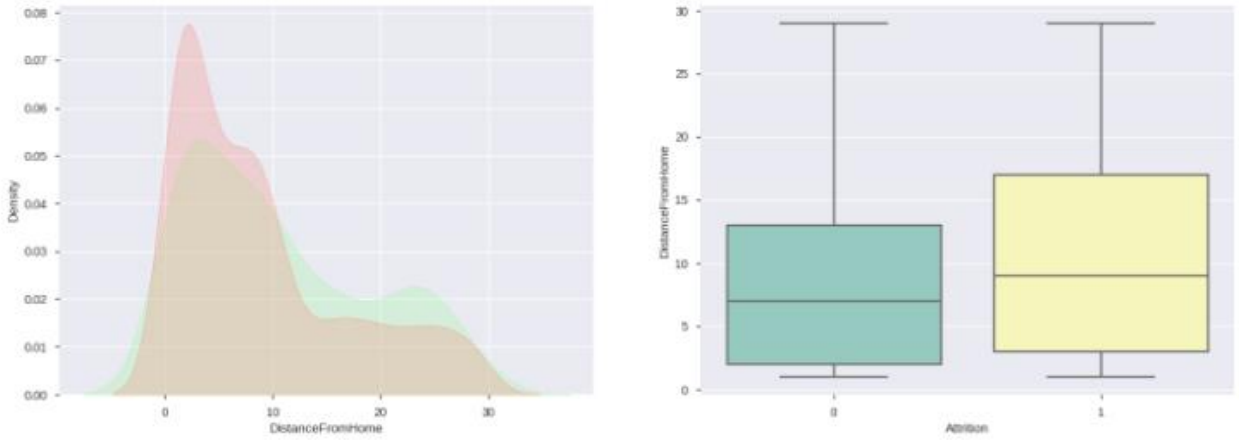
In [31]:
```
numerical_column_viz("Age")
```



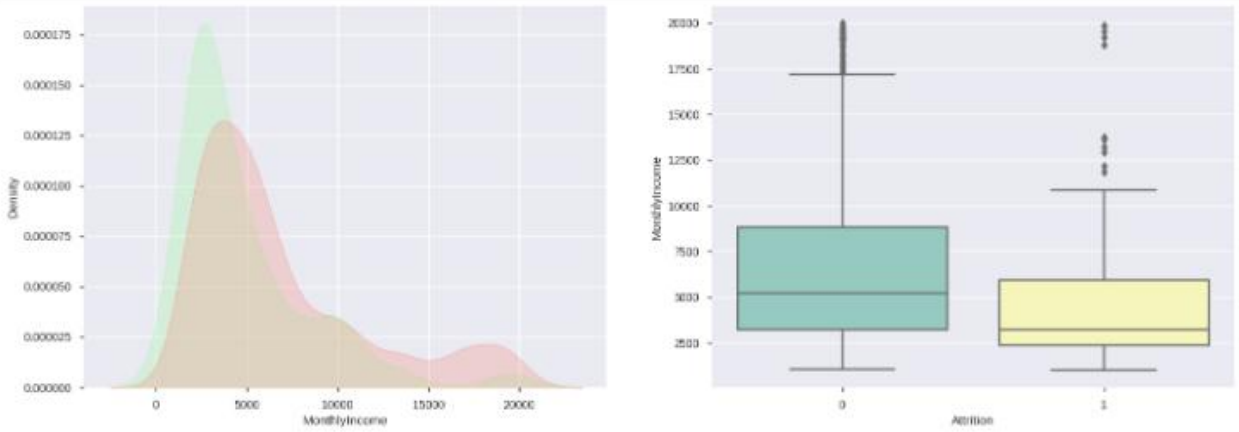
In [32]:
```
numerical_column_viz("DailyRate")
```



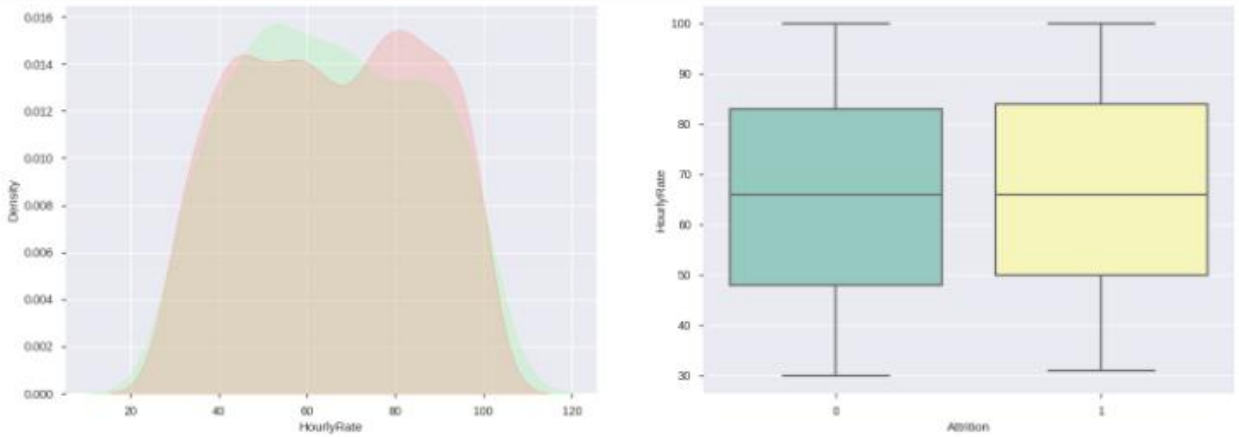
In [33]:
```
numerical_column_viz("DistanceFromHome")
```



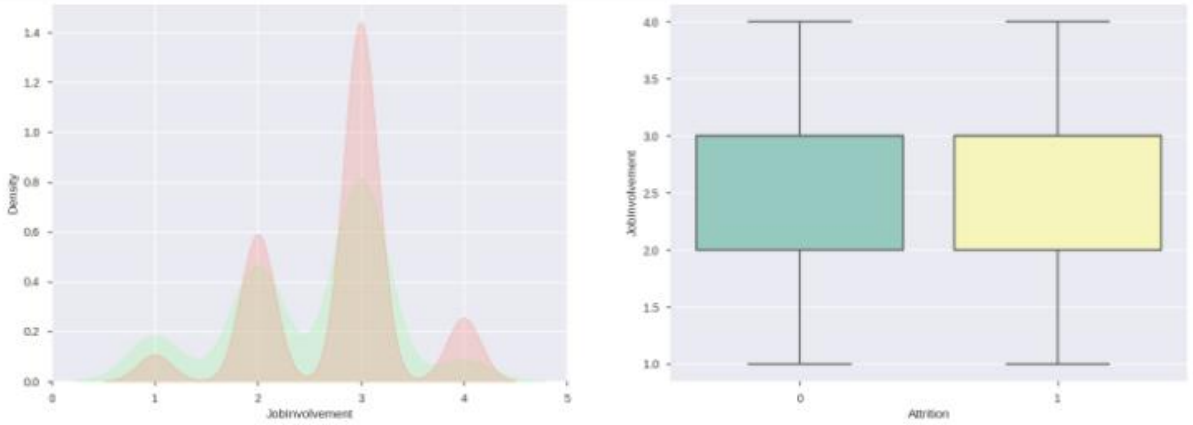
In [34]:
```
numerical_column_viz("MonthlyIncome")
```



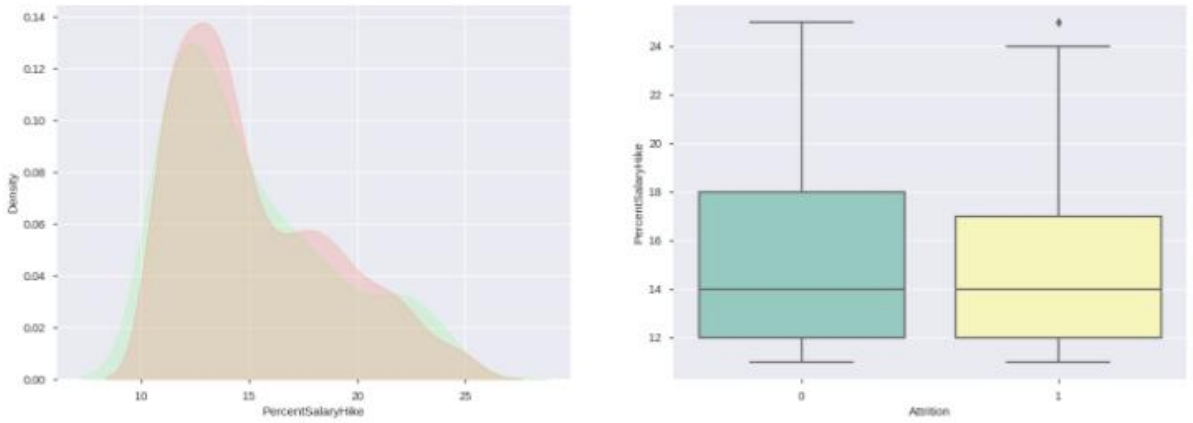
In [35]:
```
numerical_column_viz("HourlyRate")
```



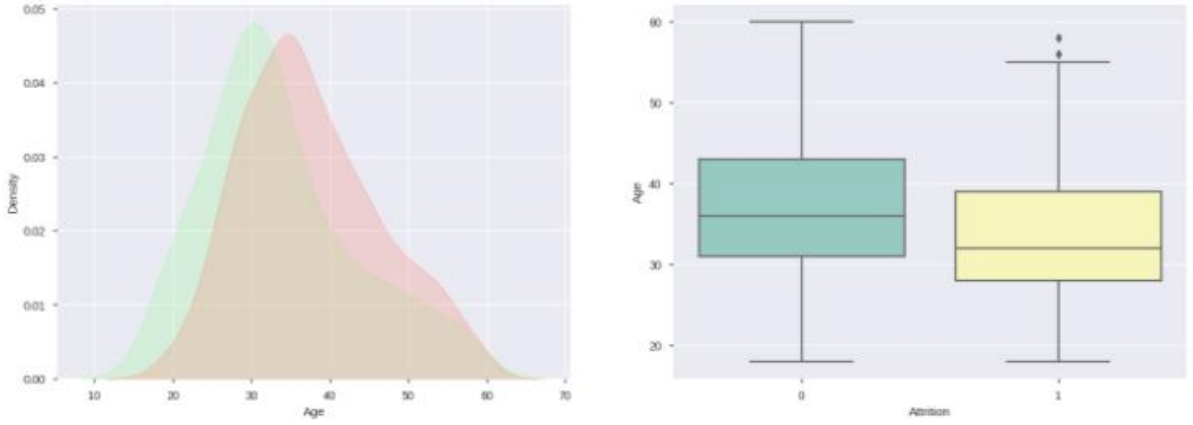
In [36]:
```
numerical_column_viz("JobInvolvement")
```



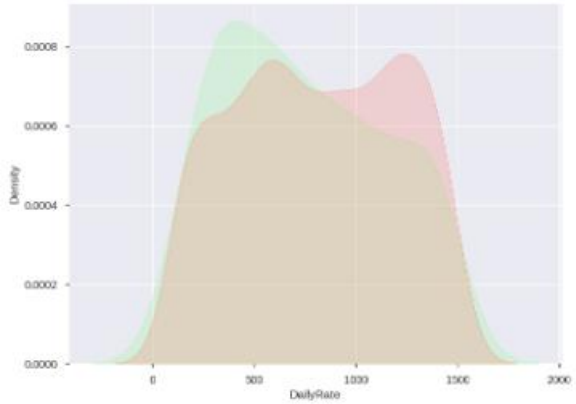
In [37]:
```
numerical_column_viz("PercentSalaryHike")
```

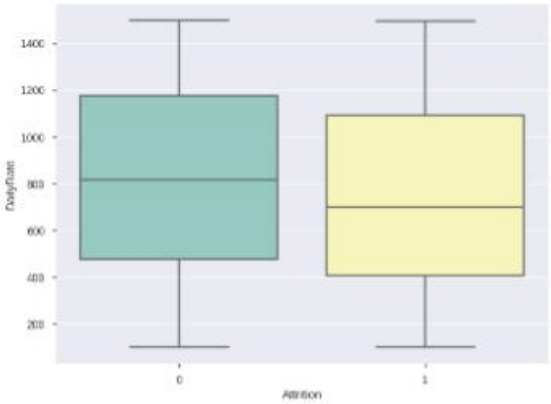


In [38]:
```
numerical_column_viz("Age")
```

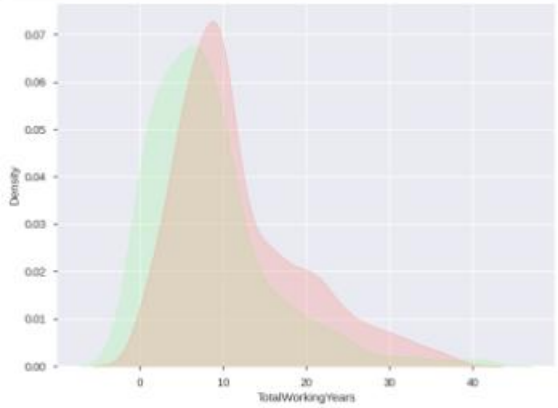


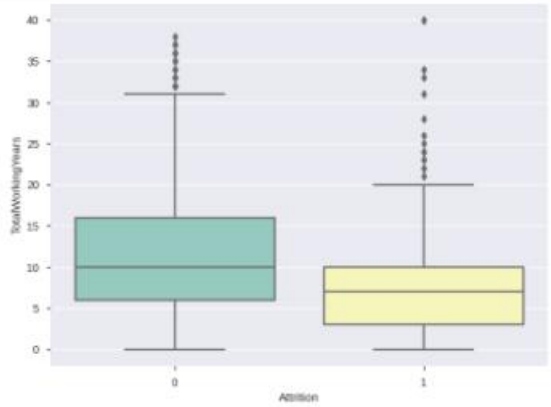
In [39]:
```
numerical_column_viz("DailyRate")
```



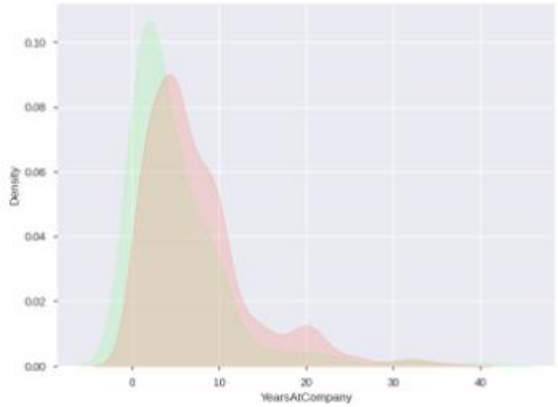


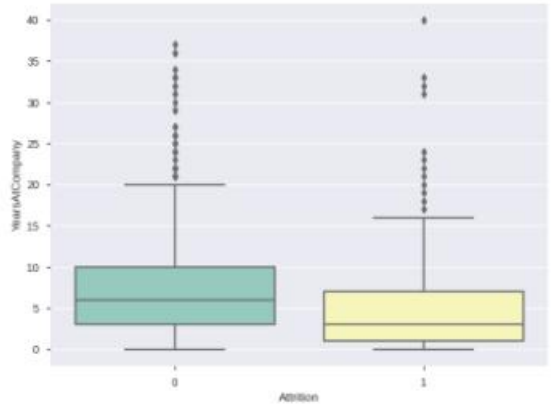
In [40]:
```
numerical_column_viz("TotalWorkingYears")
```



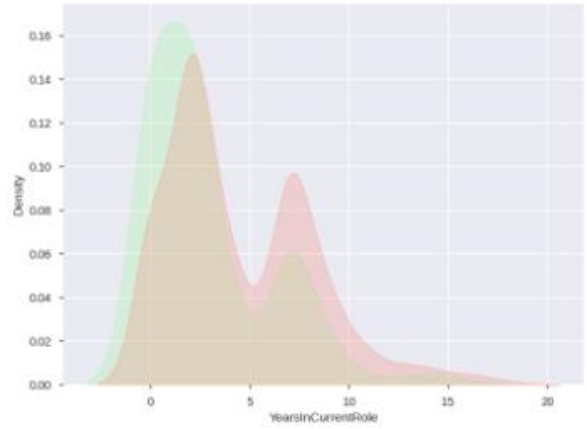


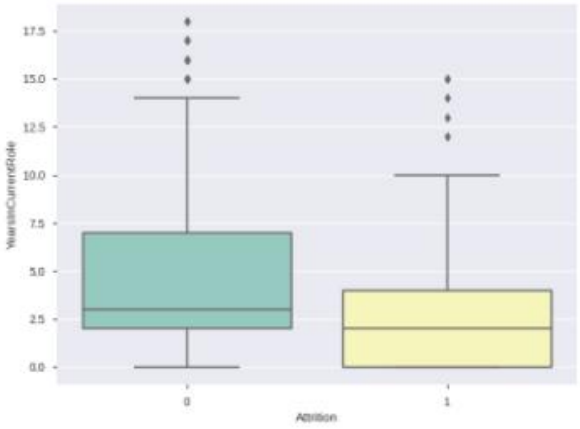
In [41]:
```
numerical_column_viz("YearsAtCompany")
```



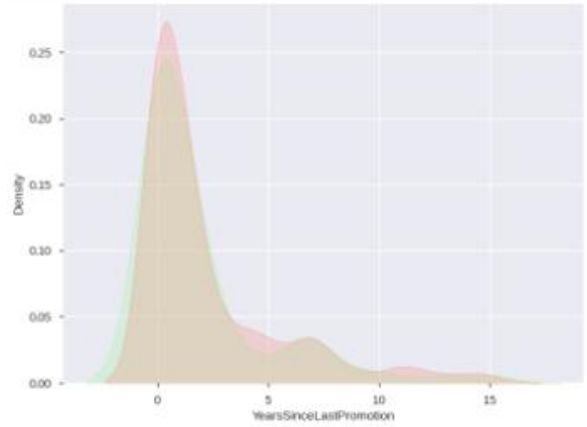


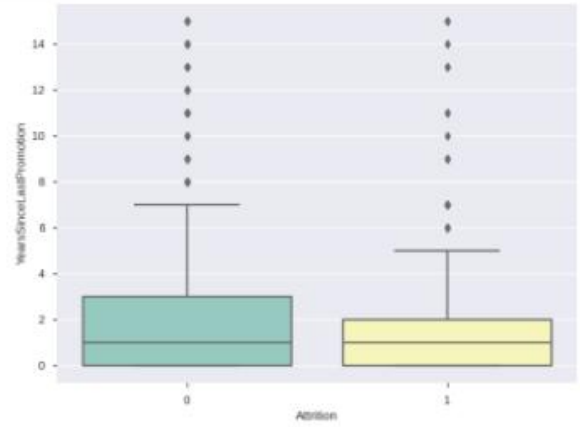
In [42]:
```
numerical_column_viz("YearsInCurrentRole")
```



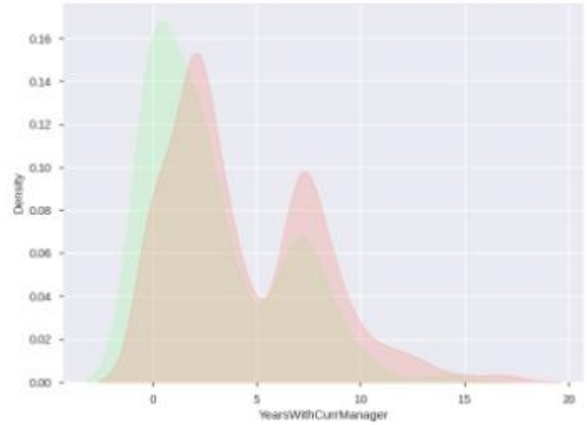


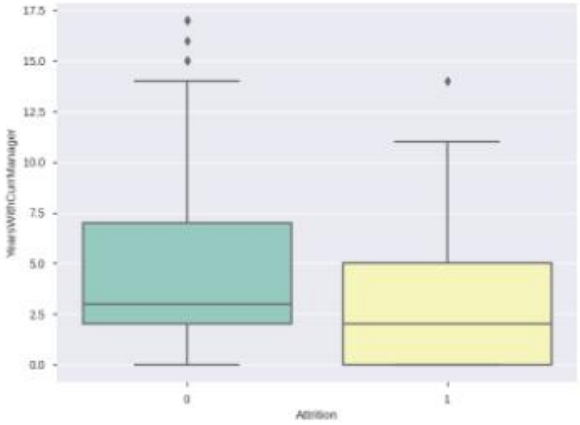
[46]:
```
numerical_column_viz("YearsSinceLastPromotion")
```





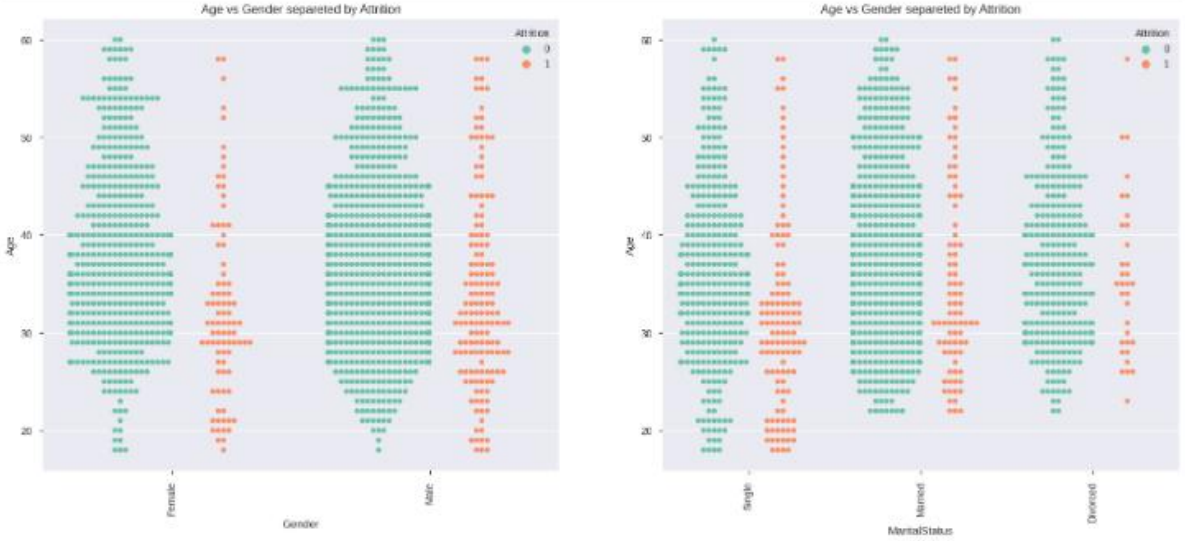
[44]:
```
numerical_column_viz("YearsWithCurrManager")
```



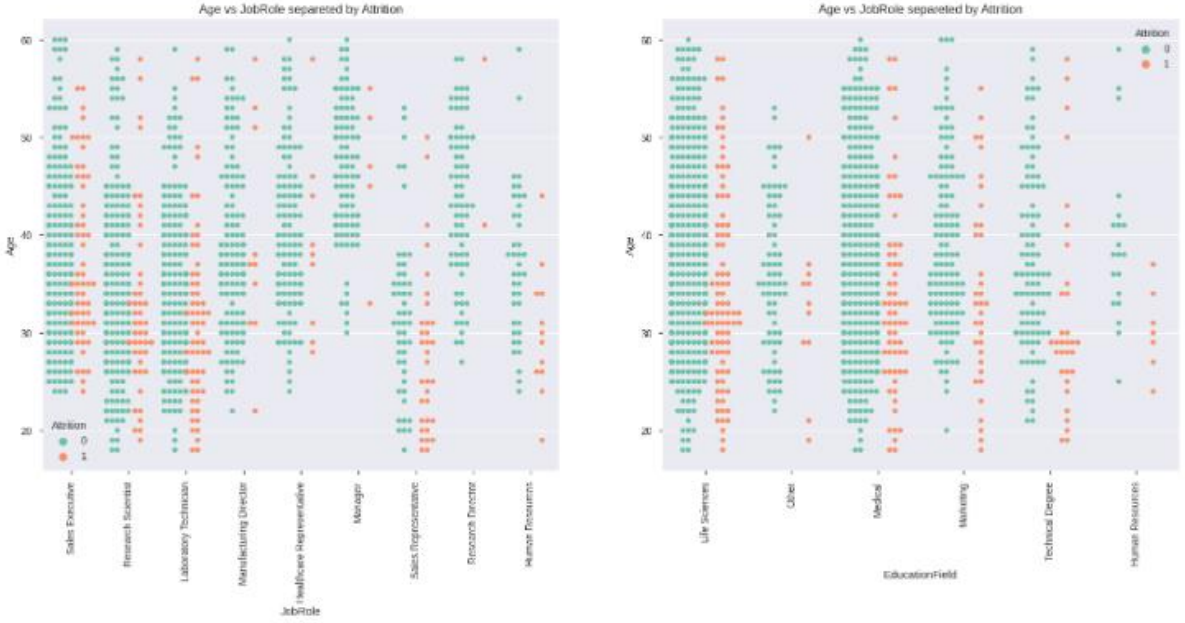


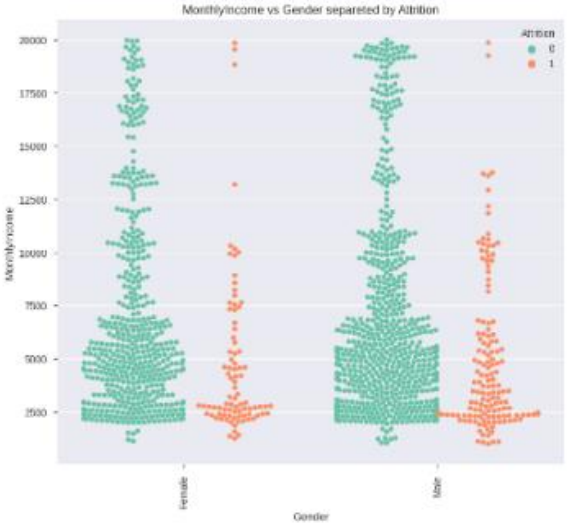
In [45]:
```
def categorical_numerical(numerical_col, categorical_col1, categorical_col2):


    f,ax = plt.subplots(1,2, figsize=(20,8))

    g1= sns.swarmplot( categorical_col1, numerical_col,hue='Attrition', data=df, dodge=True, ax=ax[0], palette='Set2')
    ax[0].set_title(f'{numerical_col} vs {categorical_col1} separeted by Attrition')
    g1.set_xticklabels(g1.get_xticklabels(), rotation=90)


    g2 = sns.swarmplot( categorical_col2, numerical_col,hue='Attrition', data=df, dodge=True, ax=ax[1], palette='Set2')
    ax[1].set_title(f'{numerical_col} vs {categorical_col1} separeted by Attrition')
    g2.set_xticklabels(g2.get_xticklabels(), rotation=90)
```

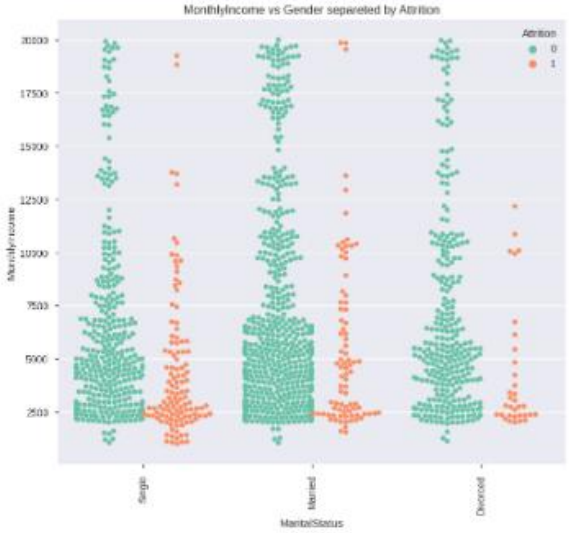
In [47]:
```
categorical_numerical('Age','Gender','MaritalStatus')
```



In [48]:
```
categorical_numerical('Age','JobRole','EducationField')
```





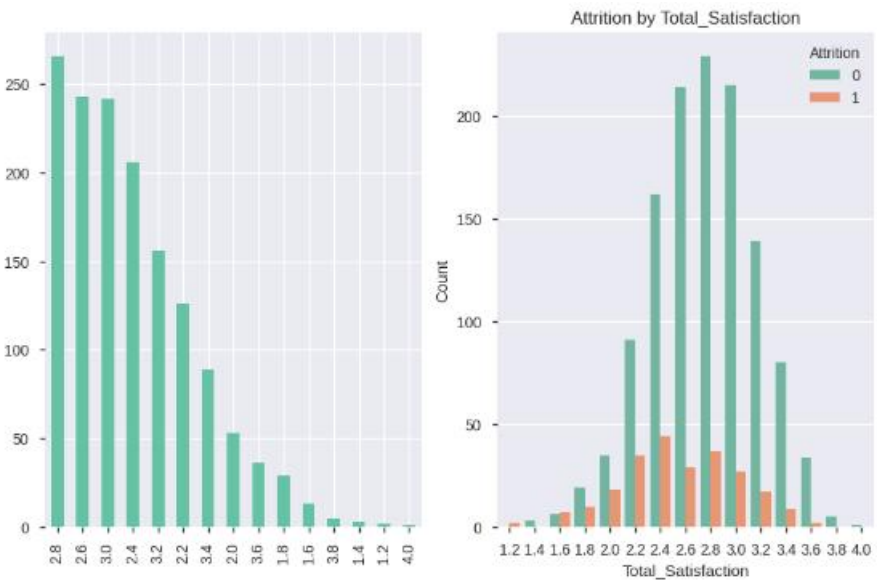


In [ ]:
```
Feature Engineering
```

n [57]:
```
# 'EnvironmentSatisfaction', 'JobInvolvement', 'JobSatisfaction', 'RelationshipSatisfaction', 'WorkLifeBalance' can be clubbed into a single feature

df['Total_Satisfaction'] = (df['EnvironmentSatisfaction'] +
                            df['JobInvolvement'] +
                            df['JobSatisfaction'] +
                            df['RelationshipSatisfaction'] +
                            df['WorkLifeBalance']) /5

# Drop Columns
df.drop(['EnvironmentSatisfaction','JobInvolvement','JobSatisfaction','RelationshipSatisfaction','WorkLifeBalance'], axis=1, inplace=True)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
/usr/local/lib/python3.7/dist-packages/pandas/core/indexes/base.py in get_loc(self, key, method, tolerance)
   3360             try:
-> 3361                 return self._engine.get_loc(casted_key)
   3362             except KeyError as err:

/usr/local/lib/python3.7/dist-packages/pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

/usr/local/lib/python3.7/dist-packages/pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'EnvironmentSatisfaction'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
 in
      4                           df['JobInvolvement'] +
      5                           df['JobSatisfaction'] +
----> 6                           df['RelationshipSatisfaction'] +
      7                           df['WorkLifeBalance']) /5
      8

/usr/local/lib/python3.7/dist-packages/pandas/core/frame.py in __getitem__(self, key)
   3456             if self.columns.nlevels > 1:
   3457                 return self._getitem_multilevel(key)
-> 3458             indexer = self.columns.get_loc(key)
   3459             if is_integer(indexer):
   3460                 indexer = [indexer]

/usr/local/lib/python3.7/dist-packages/pandas/core/indexes/base.py in get_loc(self, key, method, tolerance)
   3361                 return self._engine.get_loc(casted_key)
   3362             except KeyError as err:
-> 3363                 raise KeyError(key) from err
   3364
   3365         if is_scalar(key) and isna(key) and not self.hasnans:

KeyError: 'EnvironmentSatisfaction'
```

n [52]:
```
categorical_column_viz('Total_Satisfaction')
```



In [54]:
```
df.Total_Satisfaction.describe()
```

Out[54]:
```
count    1470.000000
mean        2.730748
std         0.428551
min         1.200000
25%         2.400000
50%         2.800000
75%         3.000000
max         4.000000
Name: Total_Satisfaction, dtype: float64
```

In [58]:
```
# Convert Total satisfaction into boolean
# median = 2.8
# x = 1 if x >= 2.8

df['Total_Satisfaction_bool'] = df['Total_Satisfaction'].apply(lambda x:1 if x>=2.8 else 0 )
df.drop('Total_Satisfaction', axis=1, inplace=True)
```

In [59]:
```
# It can be observed that the rate of attrition of employees below age of 35 is high

df['Age_bool'] = df['Age'].apply(lambda x:1 if x<35 else 0)
df.drop('Age', axis=1, inplace=True)
```

In [60]:
```
# It can be observed that the employees are more likey the drop the job if dailtRate less than 800

df['DailyRate_bool'] = df['DailyRate'].apply(lambda x:1 if x<800 else 0)
df.drop('DailyRate', axis=1, inplace=True)
```

In [61]:
```
# Employees working at R&D Department have higher attrition rate

df['Department_bool'] = df['Department'].apply(lambda x:1 if x=='Research & Development' else 0)
df.drop('Department', axis=1, inplace=True)
```

In [62]:
```
# Rate of attrition of employees is high if DistanceFromHome > 10

df['DistanceFromHome_bool'] = df['DistanceFromHome'].apply(lambda x:1 if x>10 else 0)
df.drop('DistanceFromHome', axis=1, inplace=True)
```

In [63]:
```
# Employees are more likey to drop the job if the employee is working as Laboratory Technician

df['JobRole_bool'] = df['JobRole'].apply(lambda x:1 if x=='Laboratory Technician' else 0)
df.drop('JobRole', axis=1, inplace=True)
```

In [64]:
```
# Employees are more likey to the drop the job if the employee's hourly rate < 65

df['HourlyRate_bool'] = df['HourlyRate'].apply(lambda x:1 if x<65 else 0)
```

In [65]:
```
# Employees are more likey to the drop the job if the employee's MonthlyIncome < 4000

df['MonthlyIncome_bool'] = df['MonthlyIncome'].apply(lambda x:1 if x<4000 else 0)
df.drop('MonthlyIncome', axis=1, inplace=True)
```

In [66]:
```
# Rate of attrition of employees is high if NumCompaniesWorked < 3

df['NumCompaniesWorked_bool'] = df['NumCompaniesWorked'].apply(lambda x:1 if x>3 else 0)
df.drop('NumCompaniesWorked', axis=1, inplace=True)
```

In [67]:
```
# Employees are more likey to the drop the job if the employee's TotalWorkingYears < 8

df['TotalWorkingYears_bool'] = df['TotalWorkingYears'].apply(lambda x:1 if x<8 else 0)
df.drop('TotalWorkingYears', axis=1, inplace=True)
```

In [68]:
```
# Employees are more likey to the drop the job if the employee's YearsAtCompany < 3

df['YearsAtCompany_bool'] = df['YearsAtCompany'].apply(lambda x:1 if x<3 else 0)
df.drop('YearsAtCompany', axis=1, inplace=True)
```

In [69]:
```
# Employees are more likey to the drop the job if the employee's YearsInCurrentRole < 3

df['YearsInCurrentRole_bool'] = df['YearsInCurrentRole'].apply(lambda x:1 if x<3 else 0)
df.drop('YearsInCurrentRole', axis=1, inplace=True)
```

In [70]:
```
# Employees are more likey to the drop the job if the employee's YearsSinceLastPromotion < 1

df['YearsSinceLastPromotion_bool'] = df['YearsSinceLastPromotion'].apply(lambda x:1 if x<1 else 0)
df.drop('YearsSinceLastPromotion', axis=1, inplace=True)
```

In [71]:
```
# Employees are more likey to the drop the job if the employee's YearsWithCurrManager < 1

df['YearsWithCurrManager_bool'] = df['YearsWithCurrManager'].apply(lambda x:1 if x<1 else 0)
df.drop('YearsWithCurrManager', axis=1, inplace=True)
```

In [72]:
```
df['Gender'] = df['Gender'].apply(lambda x:1 if x=='Female' else 0)
```

In [73]:
```
df.drop('MonthlyRate', axis=1, inplace=True)
df.drop('PercentSalaryHike', axis=1, inplace=True)
```

In [74]:
```
convert_category = ['BusinessTravel','Education','EducationField','MaritalStatus','StockOptionLevel','OverTime','Gender','TrainingTimesLastYear']
for col in convert_category:
        df[col] = df[col].astype('category')
```

In [75]:
```
df.info()
```

```
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 25 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Attrition                     1470 non-null   int64
 1   BusinessTravel                1470 non-null   category
 2   Education                     1470 non-null   category
 3   EducationField                1470 non-null   category
 4   Gender                        1470 non-null   category
 5   JobLevel                      1470 non-null   int64
 6   MaritalStatus                 1470 non-null   category
 7   OverTime                      1470 non-null   category
 8   PerformanceRating             1470 non-null   int64
 9   StockOptionLevel              1470 non-null   category
 10  TrainingTimesLastYear         1470 non-null   category
 11  Total_Satisfaction_bool       1470 non-null   int64
 12  Age_bool                      1470 non-null   int64
 13  DailyRate_bool                1470 non-null   int64
 14  Department_bool               1470 non-null   int64
 15  DistanceFromHome_bool         1470 non-null   int64
 16  JobRole_bool                  1470 non-null   int64
 17  HourlyRate_bool               1470 non-null   int64
 18  MonthlyIncome_bool            1470 non-null   int64
 19  NumCompaniesWorked_bool       1470 non-null   int64
 20  TotalWorkingYears_bool        1470 non-null   int64
 21  YearsAtCompany_bool           1470 non-null   int64
 22  YearsInCurrentRole_bool       1470 non-null   int64
 23  YearsSinceLastPromotion_bool  1470 non-null   int64
 24  YearsWithCurrManager_bool     1470 non-null   int64
```

```
 24  YearsWithCurrManager_bool     1470 non-null   int64
dtypes: category(8), int64(17)
memory usage: 208.3 KB
```

In [76]:
```
#separate the categorical and numerical data
X_categorical = df.select_dtypes(include=['category'])
X_numerical = df.select_dtypes(include=['int64'])
X_numerical.drop('Attrition', axis=1, inplace=True)
```

In [77]:
```
y = df['Attrition']
```

In [78]:
```
# One HOt Encoding Categorical Features

onehotencoder = OneHotEncoder()

X_categorical = onehotencoder.fit_transform(X_categorical).toarray()
X_categorical = pd.DataFrame(X_categorical)
X_categorical
```

Out[78]:

	0	1	2	3	4	5	6	7	8	9	...	22	23	24	25	26	27	28	29	30	31
0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	...	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	...	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
2	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
3	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	...	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
4	0.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1465	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	...	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
1466	0.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0
1467	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	...	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
1468	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
1469	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0

1470 rows × 32 columns

In [79]:
```
#concat the categorical and numerical values

X_all = pd.concat([X_categorical, X_numerical], axis=1)
X_all.head()
```

Out[79]:

	0	1	2	3	4	5	6	7	8	9	...	DistanceFromHome bool	JobRole bool	HourlyRate bool	MonthlyIncome bool	NumCompaniesWorked bool	TotalWorkingYears
0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	...	0	0	0	0	1	
1	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	...	0	0	1	0	0	
2	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	...	0	1	0	1	1	
3	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	...	0	0	1	1	0	
4	0.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0	1	1	1	1	

5 rows × 48 columns

In [80]:
```
X_all.info()
```

```
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 48 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   0                             1470 non-null   float64
 1   1                             1470 non-null   float64
 2   2                             1470 non-null   float64
 3   3                             1470 non-null   float64
 4   4                             1470 non-null   float64
 5   5                             1470 non-null   float64
 6   6                             1470 non-null   float64
 7   7                             1470 non-null   float64
 8   8                             1470 non-null   float64
 9   9                             1470 non-null   float64
 10  10                            1470 non-null   float64
 11  11                            1470 non-null   float64
 12  12                            1470 non-null   float64
 13  13                            1470 non-null   float64
 14  14                            1470 non-null   float64
```

```
 15  15                          1470 non-null   float64
 16  16                          1470 non-null   float64
 17  17                          1470 non-null   float64
 18  18                          1470 non-null   float64
 19  19                          1470 non-null   float64
 20  20                          1470 non-null   float64
 21  21                          1470 non-null   float64
 22  22                          1470 non-null   float64
 23  23                          1470 non-null   float64
 24  24                          1470 non-null   float64
 25  25                          1470 non-null   float64
 26  26                          1470 non-null   float64
 27  27                          1470 non-null   float64
 28  28                          1470 non-null   float64
 29  29                          1470 non-null   float64
 30  30                          1470 non-null   float64
 31  31                          1470 non-null   float64
 32  JobLevel                    1470 non-null   int64
 33  PerformanceRating           1470 non-null   int64
 34  Total_Satisfaction_bool     1470 non-null   int64
 35  Age_bool                    1470 non-null   int64
 36  DailyRate_bool              1470 non-null   int64
 37  Department_bool             1470 non-null   int64
 38  DistanceFromHome_bool       1470 non-null   int64
 39  JobRole_bool                1470 non-null   int64
 40  HourlyRate_bool             1470 non-null   int64
 41  MonthlyIncome_bool          1470 non-null   int64
 42  NumCompaniesWorked_bool     1470 non-null   int64
 43  TotalWorkingYears_bool      1470 non-null   int64
 44  YearsAtCompany_bool         1470 non-null   int64
 45  YearsInCurrentRole_bool     1470 non-null   int64
 46  YearsSinceLastPromotion_bool 1470 non-null  int64
 47  YearsWithCurrManager_bool   1470 non-null   int64
dtypes: float64(32), int64(16)
memory usage: 551.4 KB
```

Split Data

In [78]:
```
X_train,X_test, y_train, y_test = train_test_split(X_all,y, test_size=0.30)
```

In [ ]:
```
print(f"Train data shape: {X_train.shape}, Test Data Shape {X_test.shape}")
```

```
Train data shape: (1029, 48), Test Data Shape (441, 48)
```

In [ ]:
```
X_train.head()
```

Out[ ]:

	0	1	2	3	4	5	6	7	8	9	...	DistanceFromHome bool	JobRole bool	HourlyRate bool	MonthlyIncome bool	NumCompaniesWorked bool	TotalWorkingY
772	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	...	0	0	1	1	0	
1403	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	...	1	0	0	0	0	
9	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	...	1	0	0	0	1	
662	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	...	0	0	1	1	0	
1387	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	...	0	0	0	0	0	

5 rows × 48 columns

Train Data

In [ ]:
```
```

In [79]:
```
# Function that runs the requested algorithm and returns the accuracy metrics
def fit_ml_algo(algo, X_train,y_train, cv):

    # One Pass
    model = algo.fit(X_train, y_train)
    acc = round(model.score(X_train, y_train) * 100, 2)

    # Cross Validation
    train_pred = model_selection.cross_val_predict(algo,X_train,y_train,cv=cv,n_jobs = -1)

    # Cross-validation accuracy metric
    acc_cv = round(metrics.accuracy_score(y_train, train_pred) * 100, 2)

    return train_pred, acc, acc_cv
```

Logistic Regression

In [80]:
```
# Logistic Regression
start_time = time.time()
train_pred_log, acc_log, acc_cv_log = fit_ml_algo(LogisticRegression(), X_train,y_train, 10)
log_time = (time.time() - start_time)
print("Accuracy: %s" % acc_log)
print("Accuracy CV 10-Fold: %s" % acc_cv_log)
print("Running Time: %s" % datetime.timedelta(seconds=log_time))
```

```
Accuracy: 89.89
Accuracy CV 10-Fold: 87.76
Running Time: 0:00:02.063230
```

Support Vector Machine

In [81]:
```
# SVC
start_time = time.time()
train_pred_svc, acc_svc, acc_cv_svc = fit_ml_algo(SVC(),X_train,y_train,10)
svc_time = (time.time() - start_time)
print("Accuracy: %s" % acc_svc)
print("Accuracy CV 10-Fold: %s" % acc_cv_svc)
print("Running Time: %s" % datetime.timedelta(seconds=svc_time))
```

```
Accuracy: 87.37
Accuracy CV 10-Fold: 85.62
Running Time: 0:00:01.619185
```

Linear Support Vector Machines

In [82]:
```
# Linear SVC
start_time = time.time()
train_pred_svc, acc_linear_svc, acc_cv_linear_svc = fit_ml_algo(LinearSVC(),X_train, y_train,10)
linear_svc_time = (time.time() - start_time)
print("Accuracy: %s" % acc_linear_svc)
print("Accuracy CV 10-Fold: %s" % acc_cv_linear_svc)
print("Running Time: %s" % datetime.timedelta(seconds=linear_svc_time))
```

Gaussian Naive Bayes

In [84]:
```
# Gaussian Naive Bayes
start_time = time.time()
train_pred_gaussian, acc_gaussian, acc_cv_gaussian = fit_ml_algo(GaussianNB(),X_train,y_train,10)
gaussian_time = (time.time() - start_time)
print("Accuracy: %s" % acc_gaussian)
print("Accuracy CV 10-Fold: %s" % acc_cv_gaussian)
print("Running Time: %s" % datetime.timedelta(seconds=gaussian_time))
```

```
Accuracy: 76.77
Accuracy CV 10-Fold: 74.83
Running Time: 0:00:00.106342
```

Perceptron

In [86]:
```
# Perceptron
start_time = time.time()
train_pred_gaussian, acc_perceptron, acc_cv_perceptron = fit_ml_algo(Perceptron(),X_train,y_train,10)
perceptron_time = (time.time() - start_time)
print("Accuracy: %s" % acc_perceptron)
print("Accuracy CV 10-Fold: %s" % acc_cv_perceptron)
print("Running Time: %s" % datetime.timedelta(seconds=perceptron_time))
```

```
Accuracy: 88.24
Accuracy CV 10-Fold: 82.8
Running Time: 0:00:00.194112
```

Stochastic Gradient Descent

In [87]:
```
# Stochastic Gradient Descent
start_time = time.time()
train_pred_sgd, acc_sgd, acc_cv_sgd = fit_ml_algo(SGDClassifier(),X_train, y_train,10)
sgd_time = (time.time() - start_time)
print("Accuracy: %s" % acc_sgd)
print("Accuracy CV 10-Fold: %s" % acc_cv_sgd)
print("Running Time: %s" % datetime.timedelta(seconds=sgd_time))
```

```
Accuracy: 89.6
Accuracy CV 10-Fold: 85.52
Running Time: 0:00:00.211108
```

Decision Tree

In [88]:
```
# Decision Tree
start_time = time.time()
train_pred_dt, acc_dt, acc_cv_dt = fit_ml_algo(DecisionTreeClassifier(),X_train, y_train,10)
dt_time = (time.time() - start_time)
print("Accuracy: %s" % acc_dt)
print("Accuracy CV 10-Fold: %s" % acc_cv_dt)
print("Running Time: %s" % datetime.timedelta(seconds=dt_time))
```

```
Accuracy: 100.0
Accuracy CV 10-Fold: 79.11
Running Time: 0:00:00.135585
```

Gradient Boosting Trees

In [89]:
```
# Gradient Boosting Trees
start_time = time.time()
train_pred_gbt, acc_gbt, acc_cv_gbt = fit_ml_algo(GradientBoostingClassifier(),X_train, y_train,10)
gbt_time = (time.time() - start_time)
print("Accuracy: %s" % acc_gbt)
print("Accuracy CV 10-Fold: %s" % acc_cv_gbt)
print("Running Time: %s" % datetime.timedelta(seconds=gbt_time))
```

```
Accuracy: 92.61
Accuracy CV 10-Fold: 86.2
Running Time: 0:00:01.610005
```

Random Forest

In [90]:
```
# Random Forest
start_time = time.time()
train_pred_dt, acc_rf, acc_cv_rf = fit_ml_algo(RandomForestClassifier(n_estimators=100),X_train, y_train,10)
rf_time = (time.time() - start_time)
print("Accuracy: %s" % acc_rf)
print("Accuracy CV 10-Fold: %s" % acc_cv_rf)
print("Running Time: %s" % datetime.timedelta(seconds=rf_time))
```

```
Accuracy: 100.0
Accuracy CV 10-Fold: 86.01
Running Time: 0:00:01.846908
```

CatBoost Classifier

In [91]:
```
# Define the categorical features for the CatBoost model
cat_features = np.where(X_train.dtypes != np.float)[0]
cat_features
```

Out[91]:
```
array([32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47])
```

In [92]:
```
# pool training data and categorical feature labels together
train_pool = Pool(X_train, y_train,cat_features)
```

In [93]:
```
# CatBoost
catboost_model = CatBoostClassifier(iterations=1000,custom_loss=['Accuracy'],loss_function='Logloss')

# Fit CatBoost model
catboost_model.fit(train_pool,plot=True)

# CatBoost accuracy
acc_catboost = round(catboost_model.score(X_train, y_train) * 100, 2)
```

```
MetricVisualizer(layout=Layout(align_self='stretch', height='500px'))
Learning rate set to 0.010429
0:      learn: 0.6838909        total: 53.2ms   remaining: 53.1s
1:      learn: 0.6746596        total: 57.1ms   remaining: 28.5s
2:      learn: 0.6671749        total: 60.9ms   remaining: 20.2s
3:      learn: 0.6596794        total: 64ms     remaining: 15.9s
4:      learn: 0.6523877        total: 67.3ms   remaining: 13.4s
5:      learn: 0.6453233        total: 70.4ms   remaining: 11.7s
6:      learn: 0.6398173        total: 74.9ms   remaining: 10.6s
7:      learn: 0.6322516        total: 78.1ms   remaining: 9.68s
8:      learn: 0.6258855        total: 81.3ms   remaining: 8.96s
9:      learn: 0.6199996        total: 84.5ms   remaining: 8.36s
10:     learn: 0.6137649        total: 87.7ms   remaining: 7.89s
11:     learn: 0.6094101        total: 89ms     remaining: 7.33s
12:     learn: 0.6047487        total: 90.5ms   remaining: 6.87s
13:     learn: 0.5980198        total: 93.7ms   remaining: 6.59s
14:     learn: 0.5915926        total: 96.9ms   remaining: 6.36s
15:     learn: 0.5868952        total: 98.6ms   remaining: 6.06s
16:     learn: 0.5810079        total: 102ms    remaining: 5.88s
17:     learn: 0.5754514        total: 104ms    remaining: 5.68s
18:     learn: 0.5701051        total: 107ms    remaining: 5.54s
19:     learn: 0.5645224        total: 111ms    remaining: 5.43s
20:     learn: 0.5607403        total: 114ms    remaining: 5.33s
21:     learn: 0.5556906        total: 118ms    remaining: 5.24s
22:     learn: 0.5508891        total: 121ms    remaining: 5.16s
23:     learn: 0.5468321        total: 125ms    remaining: 5.07s
24:     learn: 0.5422969        total: 128ms    remaining: 5.01s
25:     learn: 0.5378868        total: 132ms    remaining: 4.94s
26:     learn: 0.5348540        total: 134ms    remaining: 4.83s
27:     learn: 0.5298800        total: 138ms    remaining: 4.78s
28:     learn: 0.5250294        total: 141ms    remaining: 4.72s
29:     learn: 0.5195701        total: 144ms    remaining: 4.67s
30:     learn: 0.5158574        total: 148ms    remaining: 4.62s
31:     learn: 0.5111602        total: 151ms    remaining: 4.57s
32:     learn: 0.5069574        total: 155ms    remaining: 4.53s
33:     learn: 0.5031138        total: 158ms    remaining: 4.49s
34:     learn: 0.5000085        total: 161ms    remaining: 4.45s
35:     learn: 0.4975229        total: 163ms    remaining: 4.37s
36:     learn: 0.4937597        total: 167ms    remaining: 4.34s
37:     learn: 0.4904901        total: 171ms    remaining: 4.32s
38:     learn: 0.4865948        total: 174ms    remaining: 4.29s
39:     learn: 0.4836626        total: 179ms    remaining: 4.29s
40:     learn: 0.4798425        total: 184ms    remaining: 4.31s
41:     learn: 0.4760123        total: 187ms    remaining: 4.28s
42:     learn: 0.4728319        total: 194ms    remaining: 4.33s
43:     learn: 0.4693484        total: 199ms    remaining: 4.32s
44:     learn: 0.4662450        total: 202ms    remaining: 4.29s
45:     learn: 0.4631085        total: 205ms    remaining: 4.26s
46:     learn: 0.4612613        total: 207ms    remaining: 4.19s
47:     learn: 0.4582029        total: 210ms    remaining: 4.17s
48:     learn: 0.4559119        total: 213ms    remaining: 4.14s
49:     learn: 0.4529584        total: 216ms    remaining: 4.11s
50:     learn: 0.4504972        total: 219ms    remaining: 4.07s
51:     learn: 0.4485357        total: 222ms    remaining: 4.04s
52:     learn: 0.4469112        total: 223ms    remaining: 3.99s
53:     learn: 0.4448109        total: 227ms    remaining: 3.97s
54:     learn: 0.4426249        total: 230ms    remaining: 3.95s
55:     learn: 0.4397636        total: 233ms    remaining: 3.93s
56:     learn: 0.4372681        total: 236ms    remaining: 3.91s
57:     learn: 0.4350696        total: 239ms    remaining: 3.88s
58:     learn: 0.4336583        total: 241ms    remaining: 3.85s
59:     learn: 0.4324908        total: 242ms    remaining: 3.8s
60:     learn: 0.4308368        total: 246ms    remaining: 3.78s
61:     learn: 0.4297242        total: 247ms    remaining: 3.74s
62:     learn: 0.4279196        total: 251ms    remaining: 3.73s
63:     learn: 0.4264802        total: 253ms    remaining: 3.7s
64:     learn: 0.4241847        total: 257ms    remaining: 3.69s
65:     learn: 0.4226741        total: 260ms    remaining: 3.68s
66:     learn: 0.4208603        total: 263ms    remaining: 3.67s
```

```
391:    learn: 0.2324504        total: 1.43s    remaining: 2.22s
392:    learn: 0.2322531        total: 1.43s    remaining: 2.21s
393:    learn: 0.2320403        total: 1.44s    remaining: 2.21s
394:    learn: 0.2318582        total: 1.44s    remaining: 2.21s
395:    learn: 0.2316972        total: 1.44s    remaining: 2.2s
396:    learn: 0.2313788        total: 1.45s    remaining: 2.2s
397:    learn: 0.2312077        total: 1.45s    remaining: 2.19s
398:    learn: 0.2309118        total: 1.45s    remaining: 2.19s
399:    learn: 0.2306646        total: 1.46s    remaining: 2.19s
400:    learn: 0.2304024        total: 1.46s    remaining: 2.18s
401:    learn: 0.2300931        total: 1.46s    remaining: 2.18s
402:    learn: 0.2298177        total: 1.47s    remaining: 2.17s
403:    learn: 0.2297181        total: 1.47s    remaining: 2.17s
404:    learn: 0.2295599        total: 1.48s    remaining: 2.17s
405:    learn: 0.2292545        total: 1.48s    remaining: 2.16s
406:    learn: 0.2290863        total: 1.48s    remaining: 2.16s
407:    learn: 0.2289229        total: 1.49s    remaining: 2.16s
408:    learn: 0.2287947        total: 1.49s    remaining: 2.15s
409:    learn: 0.2285542        total: 1.49s    remaining: 2.15s
410:    learn: 0.2285161        total: 1.5s     remaining: 2.14s
411:    learn: 0.2282309        total: 1.5s     remaining: 2.14s
412:    learn: 0.2279473        total: 1.5s     remaining: 2.14s
413:    learn: 0.2279402        total: 1.5s     remaining: 2.13s
414:    learn: 0.2275783        total: 1.51s    remaining: 2.13s
415:    learn: 0.2273892        total: 1.51s    remaining: 2.12s
416:    learn: 0.2271383        total: 1.51s    remaining: 2.12s
417:    learn: 0.2267395        total: 1.52s    remaining: 2.12s
418:    learn: 0.2266651        total: 1.52s    remaining: 2.11s
419:    learn: 0.2265208        total: 1.52s    remaining: 2.11s
420:    learn: 0.2260919        total: 1.53s    remaining: 2.1s
421:    learn: 0.2260011        total: 1.53s    remaining: 2.1s
422:    learn: 0.2259402        total: 1.54s    remaining: 2.1s
423:    learn: 0.2257747        total: 1.54s    remaining: 2.09s
424:    learn: 0.2256098        total: 1.54s    remaining: 2.09s
425:    learn: 0.2253291        total: 1.55s    remaining: 2.08s
426:    learn: 0.2250649        total: 1.55s    remaining: 2.08s
427:    learn: 0.2250511        total: 1.55s    remaining: 2.08s
428:    learn: 0.2248056        total: 1.56s    remaining: 2.08s
429:    learn: 0.2246231        total: 1.56s    remaining: 2.07s
430:    learn: 0.2242854        total: 1.57s    remaining: 2.07s
431:    learn: 0.2237819        total: 1.57s    remaining: 2.06s
432:    learn: 0.2237598        total: 1.57s    remaining: 2.06s
433:    learn: 0.2233915        total: 1.57s    remaining: 2.05s
434:    learn: 0.2231497        total: 1.58s    remaining: 2.05s
435:    learn: 0.2230083        total: 1.58s    remaining: 2.04s
436:    learn: 0.2229517        total: 1.58s    remaining: 2.04s
437:    learn: 0.2225797        total: 1.59s    remaining: 2.04s
438:    learn: 0.2222505        total: 1.59s    remaining: 2.03s
439:    learn: 0.2219331        total: 1.59s    remaining: 2.03s
440:    learn: 0.2216806        total: 1.6s     remaining: 2.02s
441:    learn: 0.2214688        total: 1.6s     remaining: 2.02s
442:    learn: 0.2212212        total: 1.6s     remaining: 2.02s
443:    learn: 0.2211801        total: 1.6s     remaining: 2.01s
444:    learn: 0.2210064        total: 1.61s    remaining: 2s
445:    learn: 0.2206778        total: 1.61s    remaining: 2s
446:    learn: 0.2203854        total: 1.61s    remaining: 2s
447:    learn: 0.2202921        total: 1.62s    remaining: 2s
448:    learn: 0.2200042        total: 1.62s    remaining: 1.99s
449:    learn: 0.2197034        total: 1.63s    remaining: 1.99s
450:    learn: 0.2193980        total: 1.63s    remaining: 1.98s
451:    learn: 0.2191153        total: 1.63s    remaining: 1.98s
452:    learn: 0.2187810        total: 1.64s    remaining: 1.98s
453:    learn: 0.2185205        total: 1.64s    remaining: 1.97s
454:    learn: 0.2182114        total: 1.64s    remaining: 1.97s
455:    learn: 0.2180615        total: 1.65s    remaining: 1.96s
456:    learn: 0.2179006        total: 1.65s    remaining: 1.96s
457:    learn: 0.2174739        total: 1.65s    remaining: 1.96s
458:    learn: 0.2172348        total: 1.66s    remaining: 1.96s
459:    learn: 0.2172339        total: 1.66s    remaining: 1.95s
460:    learn: 0.2170805        total: 1.67s    remaining: 1.95s
461:    learn: 0.2167332        total: 1.67s    remaining: 1.94s
462:    learn: 0.2167134        total: 1.67s    remaining: 1.94s
463:    learn: 0.2165896        total: 1.67s    remaining: 1.93s
464:    learn: 0.2161969        total: 1.68s    remaining: 1.93s
465:    learn: 0.2158316        total: 1.68s    remaining: 1.93s
466:    learn: 0.2154695        total: 1.68s    remaining: 1.92s
467:    learn: 0.2153695        total: 1.69s    remaining: 1.92s
468:    learn: 0.2152179        total: 1.69s    remaining: 1.92s
469:    learn: 0.2149081        total: 1.69s    remaining: 1.91s
470:    learn: 0.2146245        total: 1.7s     remaining: 1.91s
471:    learn: 0.2143091        total: 1.7s     remaining: 1.9s
472:    learn: 0.2142094        total: 1.7s     remaining: 1.9s
473:    learn: 0.2140705        total: 1.71s    remaining: 1.9s
474:    learn: 0.2138506        total: 1.71s    remaining: 1.89s
475:    learn: 0.2137389        total: 1.72s    remaining: 1.89s
476:    learn: 0.2134042        total: 1.72s    remaining: 1.88s
477:    learn: 0.2132010        total: 1.72s    remaining: 1.88s
478:    learn: 0.2130219        total: 1.73s    remaining: 1.88s
479:    learn: 0.2128212        total: 1.73s    remaining: 1.87s
480:    learn: 0.2126025        total: 1.73s    remaining: 1.87s
481:    learn: 0.2123610        total: 1.74s    remaining: 1.86s
```

```
995:    learn: 0.1258363        total: 3.67s    remaining: 14.7ms
996:    learn: 0.1257035        total: 3.67s    remaining: 11.1ms
997:    learn: 0.1254068        total: 3.68s    remaining: 7.37ms
998:    learn: 0.1253545        total: 3.68s    remaining: 3.68ms
999:    learn: 0.1251651        total: 3.68s    remaining: 0us
```

[34]:
```
start_time = time.time()

# Set params for cross-validation as same as initial model
cv_params = catboost_model.get_params()

# cross-validation
cv_data = cv(train_pool,cv_params,fold_count=10,plot=True)
catboost_time = (time.time() - start_time)

# Cross-validation accuracy metric
acc_cv_catboost = round(np.max(cv_data['test-Accuracy-mean']) * 100, 2)
```

```
MetricVisualizer(layout=Layout(align_self='stretch', height='500px'))
Streaming output truncated to the last 5000 lines.
24:     learn: 0.4085268        test: 0.4091761 best: 0.4091761 (24)    total: 92ms     remaining: 3.59s
25:     learn: 0.4062190        test: 0.4069096 best: 0.4069096 (25)    total: 93.9ms   remaining: 3.52s
26:     learn: 0.4014073        test: 0.4032473 best: 0.4032473 (26)    total: 97.9ms   remaining: 3.53s
27:     learn: 0.3962048        test: 0.4003775 best: 0.4003775 (27)    total: 101ms    remaining: 3.51s
28:     learn: 0.3913102        test: 0.3949219 best: 0.3949219 (28)    total: 105ms    remaining: 3.52s
29:     learn: 0.3873821        test: 0.3917105 best: 0.3917105 (29)    total: 109ms    remaining: 3.53s
30:     learn: 0.3838574        test: 0.3895761 best: 0.3895761 (30)    total: 113ms    remaining: 3.54s
31:     learn: 0.3803593        test: 0.3864618 best: 0.3864618 (31)    total: 117ms    remaining: 3.53s
32:     learn: 0.3788107        test: 0.3845112 best: 0.3845112 (32)    total: 119ms    remaining: 3.48s
33:     learn: 0.3770016        test: 0.3827767 best: 0.3827767 (33)    total: 122ms    remaining: 3.45s
34:     learn: 0.3725934        test: 0.3802804 best: 0.3802804 (34)    total: 126ms    remaining: 3.47s
35:     learn: 0.3704670        test: 0.3783828 best: 0.3783828 (35)    total: 130ms    remaining: 3.47s
36:     learn: 0.3669262        test: 0.3764228 best: 0.3764228 (36)    total: 134ms    remaining: 3.48s
37:     learn: 0.3638605        test: 0.3732641 best: 0.3732641 (37)    total: 138ms    remaining: 3.49s
38:     learn: 0.3595189        test: 0.3694903 best: 0.3694903 (38)    total: 144ms    remaining: 3.55s
39:     learn: 0.3560044        test: 0.3670181 best: 0.3670181 (39)    total: 150ms    remaining: 3.6s
40:     learn: 0.3536548        test: 0.3643924 best: 0.3643924 (40)    total: 156ms    remaining: 3.65s
41:     learn: 0.3515100        test: 0.3624046 best: 0.3624046 (41)    total: 161ms    remaining: 3.68s
42:     learn: 0.3478781        test: 0.3597069 best: 0.3597069 (42)    total: 165ms    remaining: 3.68s
43:     learn: 0.3454006        test: 0.3587800 best: 0.3587800 (43)    total: 169ms    remaining: 3.67s
44:     learn: 0.3425680        test: 0.3569759 best: 0.3569759 (44)    total: 173ms    remaining: 3.67s
45:     learn: 0.3396094        test: 0.3554194 best: 0.3554194 (45)    total: 176ms    remaining: 3.66s
46:     learn: 0.3375553        test: 0.3543705 best: 0.3543705 (46)    total: 180ms    remaining: 3.65s
47:     learn: 0.3345567        test: 0.3518620 best: 0.3518620 (47)    total: 186ms    remaining: 3.68s
48:     learn: 0.3327318        test: 0.3512771 best: 0.3512771 (48)    total: 189ms    remaining: 3.67s
49:     learn: 0.3324108        test: 0.3509940 best: 0.3509940 (49)    total: 191ms    remaining: 3.62s
50:     learn: 0.3297812        test: 0.3491933 best: 0.3491933 (50)    total: 194ms    remaining: 3.62s
51:     learn: 0.3270740        test: 0.3488999 best: 0.3488999 (51)    total: 198ms    remaining: 3.61s
52:     learn: 0.3249666        test: 0.3483568 best: 0.3483568 (52)    total: 201ms    remaining: 3.6s
53:     learn: 0.3246046        test: 0.3478865 best: 0.3478865 (53)    total: 203ms    remaining: 3.56s
54:     learn: 0.3234659        test: 0.3480310 best: 0.3478865 (53)    total: 207ms    remaining: 3.55s
55:     learn: 0.3204809        test: 0.3468302 best: 0.3468302 (55)    total: 210ms    remaining: 3.54s
56:     learn: 0.3175650        test: 0.3453218 best: 0.3453218 (56)    total: 214ms    remaining: 3.54s
57:     learn: 0.3157287        test: 0.3439091 best: 0.3439091 (57)    total: 218ms    remaining: 3.53s
58:     learn: 0.3155836        test: 0.3437996 best: 0.3437996 (58)    total: 219ms    remaining: 3.49s
59:     learn: 0.3126469        test: 0.3417016 best: 0.3417016 (59)    total: 222ms    remaining: 3.48s
60:     learn: 0.3100999        test: 0.3404825 best: 0.3404825 (60)    total: 226ms    remaining: 3.48s
61:     learn: 0.3081903        test: 0.3401036 best: 0.3401036 (61)    total: 230ms    remaining: 3.48s
62:     learn: 0.3068450        test: 0.3397717 best: 0.3397717 (62)    total: 234ms    remaining: 3.48s
63:     learn: 0.3046591        test: 0.3386870 best: 0.3386870 (63)    total: 238ms    remaining: 3.48s
64:     learn: 0.3030630        test: 0.3374181 best: 0.3374181 (64)    total: 242ms    remaining: 3.48s
65:     learn: 0.3006749        test: 0.3362926 best: 0.3362926 (65)    total: 246ms    remaining: 3.48s
66:     learn: 0.2989626        test: 0.3349831 best: 0.3349831 (66)    total: 250ms    remaining: 3.48s
67:     learn: 0.2961940        test: 0.3354710 best: 0.3349831 (66)    total: 292ms    remaining: 4s
68:     learn: 0.2942349        test: 0.3349415 best: 0.3349415 (68)    total: 322ms    remaining: 4.34s
69:     learn: 0.2938595        test: 0.3351414 best: 0.3349415 (68)    total: 336ms    remaining: 4.46s
70:     learn: 0.2938180        test: 0.3351310 best: 0.3349415 (68)    total: 338ms    remaining: 4.42s
71:     learn: 0.2935726        test: 0.3352231 best: 0.3349415 (68)    total: 412ms    remaining: 5.31s
72:     learn: 0.2918816        test: 0.3353099 best: 0.3349415 (68)    total: 473ms    remaining: 6s
73:     learn: 0.2909242        test: 0.3346928 best: 0.3346928 (73)    total: 516ms    remaining: 6.46s
74:     learn: 0.2894297        test: 0.3357594 best: 0.3346928 (73)    total: 521ms    remaining: 6.43s
75:     learn: 0.2885336        test: 0.3349855 best: 0.3346928 (73)    total: 527ms    remaining: 6.41s
76:     learn: 0.2885099        test: 0.3349873 best: 0.3346928 (73)    total: 529ms    remaining: 6.34s
77:     learn: 0.2874892        test: 0.3342052 best: 0.3342052 (77)    total: 535ms    remaining: 6.32s
78:     learn: 0.2853480        test: 0.3342170 best: 0.3342052 (77)    total: 541ms    remaining: 6.3s
79:     learn: 0.2851224        test: 0.3343153 best: 0.3342052 (77)    total: 544ms    remaining: 6.26s
80:     learn: 0.2834888        test: 0.3341552 best: 0.3341552 (80)    total: 550ms    remaining: 6.24s
81:     learn: 0.2822411        test: 0.3348395 best: 0.3341552 (80)    total: 556ms    remaining: 6.22s
82:     learn: 0.2800162        test: 0.3349010 best: 0.3341552 (80)    total: 562ms    remaining: 6.21s
83:     learn: 0.2783514        test: 0.3343483 best: 0.3341552 (80)    total: 568ms    remaining: 6.19s
84:     learn: 0.2765494        test: 0.3340288 best: 0.3340288 (84)    total: 574ms    remaining: 6.17s
85:     learn: 0.2754679        test: 0.3342019 best: 0.3340288 (84)    total: 579ms    remaining: 6.16s
86:     learn: 0.2741570        test: 0.3331985 best: 0.3331985 (86)    total: 586ms    remaining: 6.15s
87:     learn: 0.2726826        test: 0.3319855 best: 0.3319855 (87)    total: 593ms    remaining: 6.14s
88:     learn: 0.2715277        test: 0.3304319 best: 0.3304319 (88)    total: 598ms    remaining: 6.12s
89:     learn: 0.2698380        test: 0.3299548 best: 0.3299548 (89)    total: 604ms    remaining: 6.11s
```

```
bestTest = 0.3357946691
bestIteration = 270
```

Training Model Results

In [95]:
```
models = pd.DataFrame({
    'Model': ['Logistic Regression','SVM','Linear SVC','KNN','Naive Bayes','Perceptron',
              'Stochastic Gradient Decent','Decision Tree', 'Gradient Boosting Trees','Random Forest',
              'CatBoost'],
    'Score': [
        acc_log,
        acc_svc,
        acc_linear_svc,
        acc_knn,
        acc_gaussian,
        acc_perceptron,
        acc_sgd,
        acc_dt,
        acc_gbt,
        acc_rf,
        acc_catboost
    ]})
models.sort_values(by='Score', ascending=False)
```

Out[95]:

	Model	Score
7	Decision Tree	100.00
9	Random Forest	100.00
10	CatBoost	96.11
8	Gradient Boosting Trees	92.61
0	Logistic Regression	89.89
2	Linear SVC	89.80
6	Stochastic Gradient Decent	89.60
3	KNN	88.92
5	Perceptron	88.24
1	SVM	87.37
4	Naive Bayes	76.77

In [96]:
```
cv_models = pd.DataFrame({
    'Model': ['Logistic Regression','SVM','Linear SVC','KNN','Naive Bayes','Perceptron',
              'Stochastic Gradient Decent','Decision Tree', 'Gradient Boosting Trees','Random Forest',
              'CatBoost'],
    'Score': [
        acc_cv_log,
        acc_cv_svc,
        acc_cv_linear_svc,
        acc_cv_knn,
        acc_cv_gaussian,
        acc_cv_perceptron,
        acc_cv_sgd,
        acc_cv_dt,
        acc_cv_gbt,
        acc_cv_rf,
        acc_cv_catboost
    ]})
cv_models.sort_values(by='Score', ascending=False)
```

Out[96]:

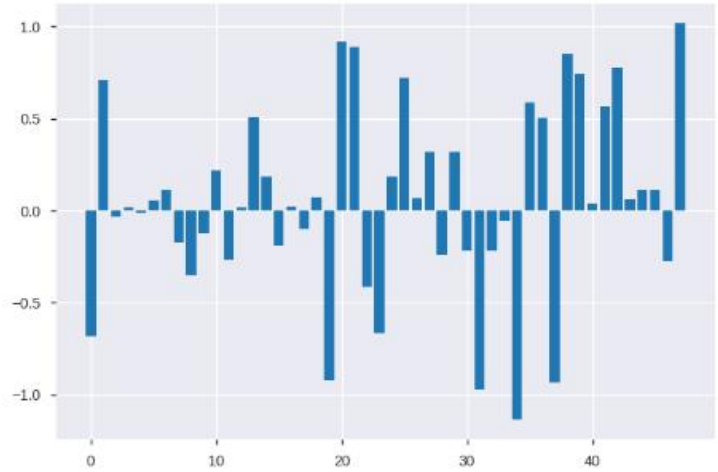
	Model	Score
0	Logistic Regression	87.76
2	Linear SVC	87.46
10	CatBoost	86.89
8	Gradient Boosting Trees	86.20
9	Random Forest	86.01
1	SVM	85.62
6	Stochastic Gradient Decent	85.52
3	KNN	83.58

```
   macro avg       0.81      0.69      0.73       441
weighted avg       0.87      0.88      0.86       441
```

In [104_

```
# get importance
importance = model.coef_[0]
# summarize feature importance
for i,v in enumerate(importance):
    print('Feature: %0d, Score: %.5f' % (i,v))
# plot feature importance
plt.bar([x for x in range(len(importance))], importance)
plt.show()
```

```
Feature: 0, Score: -0.68082
Feature: 1, Score: 0.70818
Feature: 2, Score: -0.03096
Feature: 3, Score: 0.01456
Feature: 4, Score: -0.01300
Feature: 5, Score: 0.05515
Feature: 6, Score: 0.11113
Feature: 7, Score: -0.17144
Feature: 8, Score: -0.35275
Feature: 9, Score: -0.12130
Feature: 10, Score: 0.21729
Feature: 11, Score: -0.27016
Feature: 12, Score: 0.01556
Feature: 13, Score: 0.50775
Feature: 14, Score: 0.18364
Feature: 15, Score: -0.18724
Feature: 16, Score: 0.02336
Feature: 17, Score: -0.09864
Feature: 18, Score: 0.07168
Feature: 19, Score: -0.92357
Feature: 20, Score: 0.91998
Feature: 21, Score: 0.89168
Feature: 22, Score: -0.41526
Feature: 23, Score: -0.66699
Feature: 24, Score: 0.18697
Feature: 25, Score: 0.72378
Feature: 26, Score: 0.06940
Feature: 27, Score: 0.31746
Feature: 28, Score: -0.24247
Feature: 29, Score: 0.32015
Feature: 30, Score: -0.21962
Feature: 31, Score: -0.97229
Feature: 32, Score: -0.21824
Feature: 33, Score: -0.05525
Feature: 34, Score: -1.13573
Feature: 35, Score: 0.59048
Feature: 36, Score: 0.50439
Feature: 37, Score: -0.93575
Feature: 38, Score: 0.85101
Feature: 39, Score: 0.74357
Feature: 40, Score: 0.03743
Feature: 41, Score: 0.56531
Feature: 42, Score: 0.77617
Feature: 43, Score: 0.06438
Feature: 44, Score: 0.11309
Feature: 45, Score: 0.11154
Feature: 46, Score: -0.27072
Feature: 47, Score: 1.01754
```



Video Link

GitHub Link

https://github.com/IBM-EPBL/IBM-Project-18044-1659678733