Literature Review on Employee Churn Prediction with three class Classifications.

Abstract:

This document examines the sources of attrition and, its effects & forwards some strategies on how to minimize attrition in an organization. Special attention is given to the emerging trend of employing three-class classification models, categorizing employees into distinct levels of likelihood to leave.

Our study created models to help companies figure out if employees might leave in the future. We have put employees into groups like "Highly likely to leave", "Moderately likely to leave" & "Slightly likely to leave". This way, companies can make plans to keep their employees happy. Each group has different needs, so companies can use special strategies to make sure their employees stay.

Objectives:

We have the data from Dunder Mifflin Paper Company. In the quaint town of Scranton, Pennsylvania, lies the regional branch of the Dunder Mifflin Paper Company, a well-established & somewhat quirky Paper Company.

Here we have to predict employee churn or attrition i.e. the employee will leave the company or not in which dataset contains three-class classifications 1543 rows and 18 attributes.

Data Sets:

The dataset used for model building contained 1543 observations of 18 variables. The information is about employee churn or attrition, including variables such as Tenure, Job satisfaction, Salary, Department, work-life balance, Marital Status, Performance Rating, Education, Training Hours, Overtime, Num of projects, Branch, Environment Satisfaction, Years Since Promotion & Classes (Target Variable) which states that:
Employees are classified into four classes based on their likelihood of leaving the company:

- Class A: Highly likely to leave.
- Class B: Moderately likely to leave.
- Class C: Slightly likely to leave.

Challenges:

There can be many challenges for employee attrition, as many of the employees tend to leave the job for various reasons like lack of job security, lack of career advancement, Job

dissatisfaction, problems with Supervisors & few other personal reasons. If this varies employees start looking for alternatives.

Data Understanding & Tools:

Data comes from a Kaggle competition so it can be downloaded directly. For this particular instance, we can use Pandas and Numpy libraries to process the data as we have data in CSV format.

Data Analysis:

We will perform exploratory data analysis (EDA), feature engineering, Data splitting, Hyperparameter Tuning, and build a machine learning model to predict how the employee will leave the company.

Exploratory Data Analysis (EDA):-

1. Libraries:

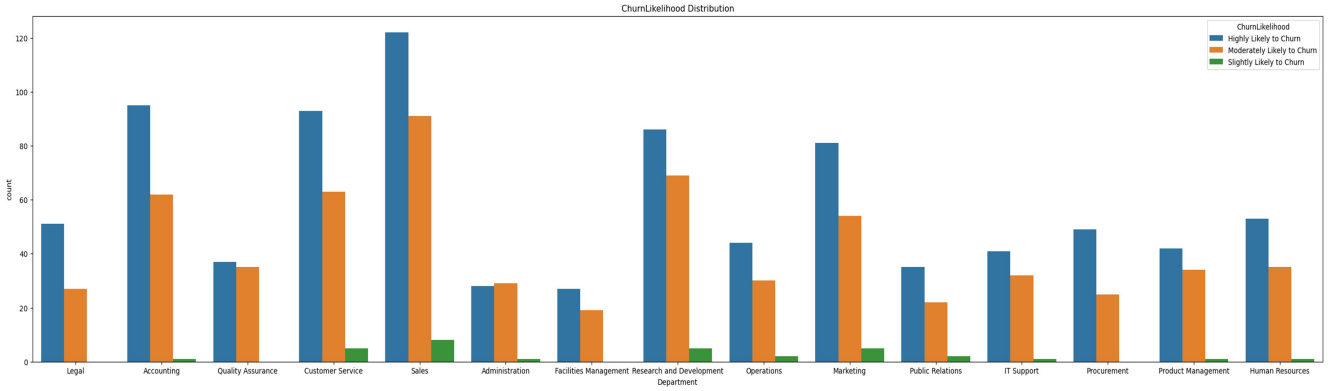
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.svm import SVC
from sklearn.preprocessing import LabelEncoder
from collections import Counter
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score
from xgboost import XGBClassifier
from sklearn.metrics import classification_report
from sklearn.ensemble import ExtraTreesRegressor
from sklearn.metrics import confusion_matrix
```

2. Cleaning data:

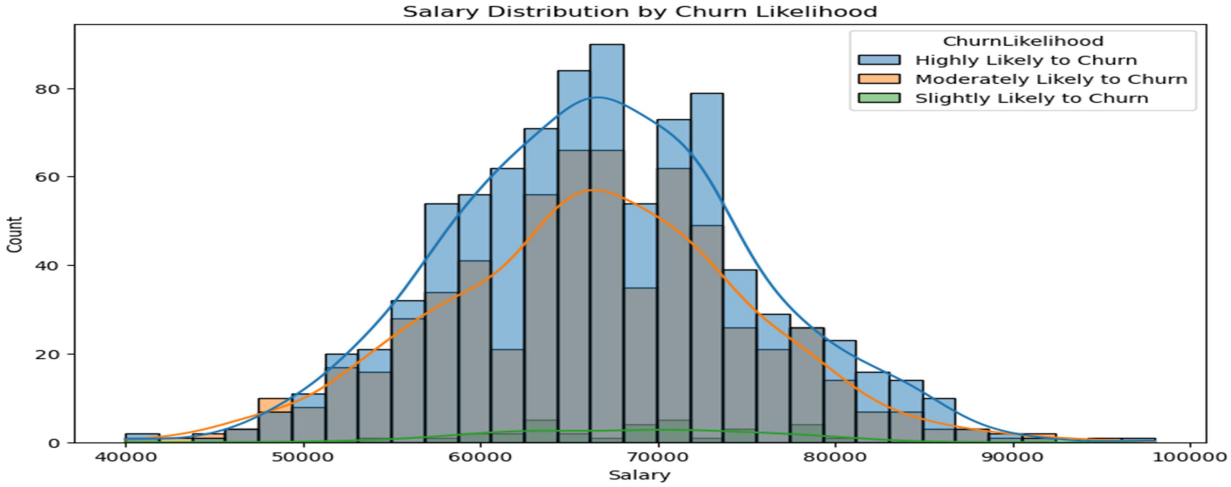
```
df["Branch"].fillna(df["Branch"].mode().iloc[0], inplace=True)
df["Tenure"].fillna(df["Tenure"].median(),inplace=True)
df["Salary"].fillna(df["Salary"].median(),inplace=True)
df["JobSatisfaction"].fillna(df["JobSatisfaction"].median(),inplace=True)
df["WorkLifeBalance"].fillna(df["WorkLifeBalance"].median(),inplace=True)
df["PerformanceRating"].fillna(df["PerformanceRating"].median(),inplace=True)
df["TrainingHours"].fillna(df["TrainingHours"].median(),inplace=True)
df["OverTime"].fillna(False, inplace=True)
df["NumProjects"].fillna(0 ,inplace=True)
df["YearsSincePromotion"].fillna(df["YearsSincePromotion"].median(),inplace=True)
df["EnvironmentSatisfaction"].fillna(df["EnvironmentSatisfaction"].median(),inplace=True)
```

3. Visualization:

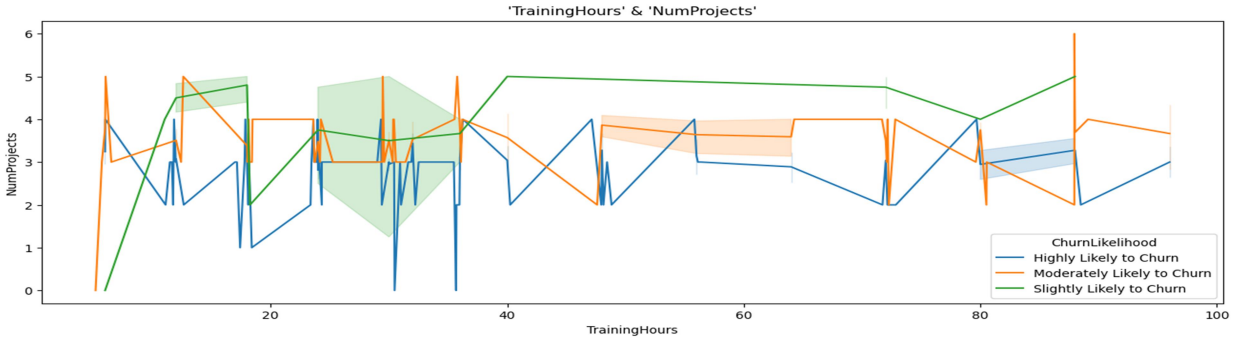
While visualization on Department it is clearly visible that Sales is highest, which shows highly likely to churn featuring with target variable i.e. Churnlikelylihood attribute. Where A, B & C represents "highly likely to churn", "Moderately likely to churn" & "Slightly likely to churn" respectively.



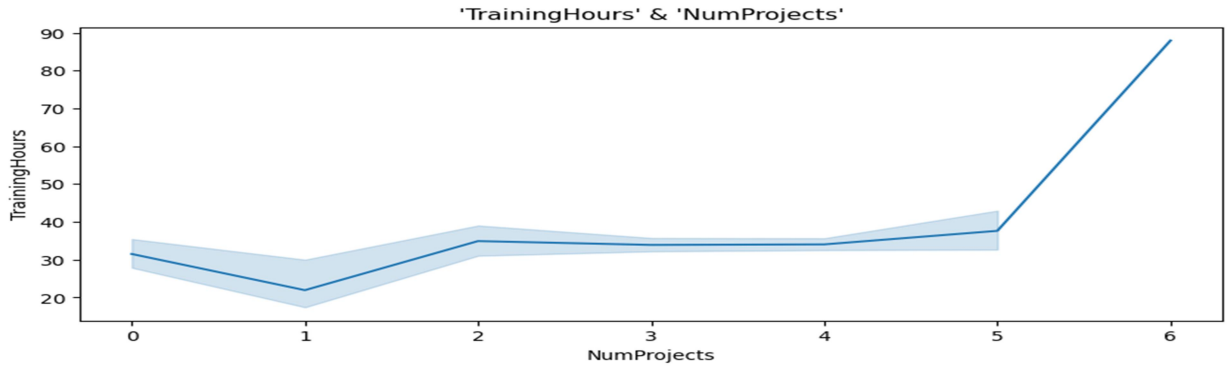
While visualization on SALARY we can see that salary between 60k to 75k tends to have high employee churn featuring with target variable i.e Churnlikelylihood attribute. Where A, B & C represents "highly likely to churn", "Moderately likely to churn" & "Slightly likely to churn" respectively.



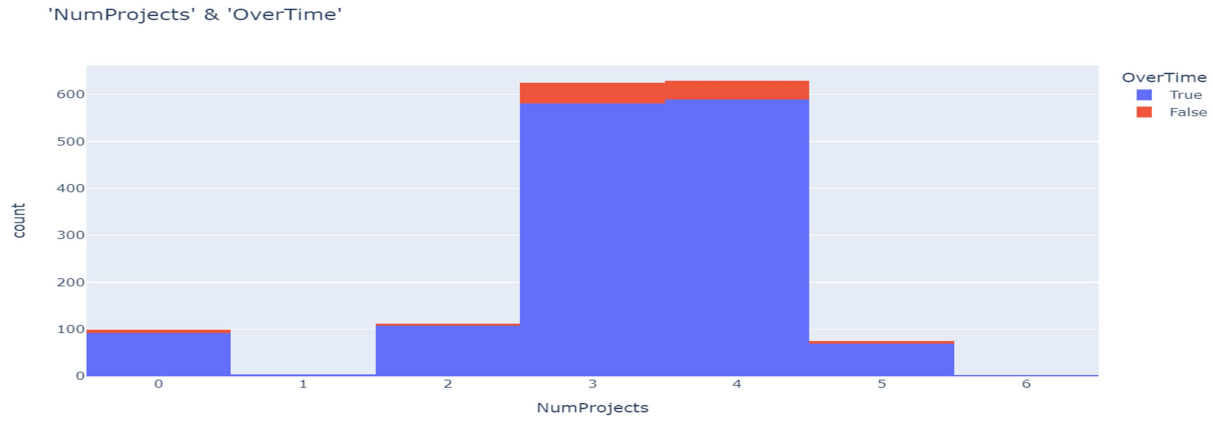
Line chart with x & y which represents Training hours and Num of projects done by the employees featuring with the target class.
Also after observing this chart class C i.e. "slightly likely to churn" tends to rise.



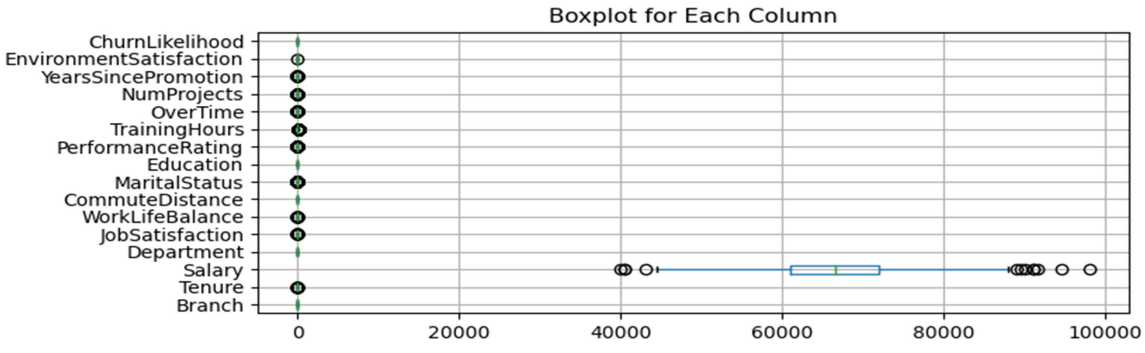
And when comparing both x & y with each other the observation says that more num of projects will have more training hours as we can see in the line chart it tends to rise.



When the employees are assigned 3 to 4 num of projects their is automatically rise in overtime. 3 to 4 are the commonly assigned projects in which they have worked more than their actual time.



Now I will try using label encoding for my data to convert into numeric.
After converting the data into numeric I worked on Outliers:



in which I found salary has some outliers by applying IQR to this to some extent the outliers were deducted.

Train Test Splitting:

```
from collections import Counter
counter = Counter(y)
print(counter)
```

```
Counter({0: 880, 1: 623, 2: 32})
```

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, stratify=y)
```

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Random Forest Classifier (Using possible Hyperparameters):

```
Best Hyperparameters: {'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 100}
Best Accuracy: 0.9055417288866765
Training Set Accuracy: 1.0
Testing Set Accuracy: 0.9153094462540716
              precision    recall  f1-score   support

           0       0.97      0.92      0.95       171
           1       0.85      0.96      0.90       126
           2       1.00      0.20      0.33        10

    accuracy                           0.92       307
   macro avg       0.94      0.69      0.73       307
weighted avg       0.92      0.92      0.91       307

Accuracy: 0.9153094462540716
```

XGB Classifier:

```
Best Hyperparameters: {'learning_rate': 0.01, 'max_depth': 3, 'min_child_weight': 1, 'n_estimators': 400}
Best Accuracy: 0.9609191969470716
Accuracy: 0.9771986970684039
Training Set Accuracy: 0.9690553745928339
Testing Set Accuracy: 0.9771986970684039
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       171
           1       0.98      0.96      0.97       126
           2       1.00      0.90      0.95        10

    accuracy                           0.98       307
   macro avg       0.99      0.95      0.97       307
weighted avg       0.98      0.98      0.98       307
```

Attrition Scenario in India: