HR ANALYTICS

HR analytics is also known as People Analytics. It is a data-driven approach to managing people at work.

Problems addressed by HR analytics-

* Hiring/Assessment
* Learning & Development
* Retention
* Collaboration/team composition
* Performance Evaluation
* Employee Attrition

In this blog I am discussing about Employee Attrition as it is a big issue for many companies are facing now-a-days. Acquiring a new employee can become costly in both time & money. Therefore, some companies are going great things trying to keep their employees happy & satisfied.

In HR analytics employee attrition is a major cost of an organizations and predicting turnover is the fore font of needs of human resources(HR) in many organizations. I know very well employee attrition is a major problem of an organization because it is only as good as is employees and these people are the true source of its competitive advantage.

Organizations face huge costs resulting from employee attrition. Some costs are tangible such as training expenses and the time it takes from when an employee starts to when they become a productive member. However, the most important costs are intangible such as consider what is lost when a productive employee quits, new product ideas, a great project management or customer relationships. With advances in machine learning and data science, it is not possible to predict employee attrition but to understand the key variables that influence turnover. Until now the main stream approach has been to use logistic regression to employee attrition. However, with advancements in Machine Learning, I can now get both better predictive performance and better explanations of what critical features are linked to employee attrition.

In this blog, I am introducing Random Forest Logistic Regression and Support Vector Machine. I will also measure the accuracy of models that are built by using Machine Learning. I also identify that how to keep employees from leaving is to analyze, why people who left the company decided to do so, predict who could be leaving next and try to take preemptive action.

**PROBLEM DEFINITION:-**

By analyzing the characteristics of past employees who have either stayed over the long term or left the company. I will use Machine Learning to build & apply classification predictive models to help the company develop policies & work conditions that will attract & retain valuable productive employees.

**DATA ANALYSIS:-**

In my blog I analyzed data from an HR dataset, containing basic information about the employees (age, attrition, Business Travel, Daily Rate etc.) & whether they left or stayed in the company. The dataset is relatively small, containing 1470 rows (values) & 35 columns (attributes).

***Note:-*** *The core of the project is prediction of attrition by Machine Learning methods & comparison of their results.*

**EDA CONCLUDING REMARKS:-**

Exploratory Data Analysis is an initial process of analysis, in which I can summarize characteristics of data such as pattern, trends, outliers, and hypothesis testing using descriptive statistics and visualization.

In this blog let’s a dataset as it is from Github. I have to first load the required HR dataset using pandas’ read CSV function. I can download the data from this link: https://github.com/AKANKSHA31-SPEC/HR-analytics-Project/blob/main/HR.csv.

There are 35 columns in the dataset i.e. Age, Attrition, BusinessTravel, DailyRate, Department, DistanceFromHome, Education, EducationField, EmployeeCount, EmployeeNumber, EnvironmentSatisfaction, Gender, HourlyRate, JobInvolvement, Joblevel, JobRole, JobSatisfaction, MaritalStatus, MonthlyIncome, MonthlyRate, NumCompaniesWorked, Over18, Overtime, PercentSalaryHike, PerformanceRating, RelationshipSatisfaction, StandardHours, StockOptionLevel, TotalWorkingYears, TrainingTimeLastYear, WorkLifeBalance, YearsAtCompany, YearsInCurrentRole, YearsSinceLastPromotion, YearsWithCurrManager.

After I have loaded the dataset, I want to know a little bit more about it. I can check attributes names and datatypes using info().Now I am checking the column of the data types use command dypes(). There is result come some categorical column and numerical column.

Categorical Column (object):- Attrition, BusinessTravel, Department, EducationField, Gender, MaritalStatus, Over18, Overtime and rather as numerical column which data type int64.

**PRE PROCESSING PIPELINE:-**

After that I have to check data info, data describe. When I have to describe data column age-

* Count:1470.00
* Mean:36.9238
* Std:9.135
* Min:18.00
* 25%:30.00
* 50%:36.00
* 75%:43.00
* Max:60.00

And rest column describe data as it is but one thing should be identify that code is run when only data type is numerical form. Our data is pretty clean no one null value in this dataset. So, I will start a process univariate and bivariate analysis. Attrition is our target value but it is categorical column so I use dummy row to convert into numerical column.

Use this code I can convert categorical column into numerical column:-

attrition\_dummies=pd.get\_dummies(df ['Attrition'])

attrition\_dummies.head()

After univariate and variate analysis I have visualize that out of total employees only 233 employees are turnover and rest 1233 employees stayed in our dataset.

Several observations:-

* In the Department column Sales and human resources department employees who stayed with the company is higher than that of the employees who left.
* In the Business Travel who travel rarely and travel frequently employees who stayed with the company is higher than that of the employees left and non travel is vice-versa.

With Statistics view I can also see that the variable EmployeeCount, Over18 and StandardHours have a single value in the whole dataset. So, I will remove them as they are useless with regard to predictive significance. I will also exclude the EmployeeNumber variable as it’s just an ID.

At this point I will analyze the correlation between independent variables and the target variables, attrition. I will find out some of the key attributes which can help to identify the attrition and I can build a prediction model using the same so that provided values for all these columns one can predict whether the employee is going to stay with the company or not. Our target attribute attrition showed poor correlation with other attributes, which can tell us that correct classification of leaving employees was not a piece of cake. The attrition correlates weakly only with attributes overtime and business travel. Attrition is more likely to depend on combination of attributes rather than on a single one.

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn import metrics

import matplotlib.pyplot as plt

import numpy as np

import seaborn as sn

#importing the warnings

import warnings

warnings.filterwarnings("ignore")

df=pd.read\_csv('HR.csv')

print(df)

df.columns

df.shape

df.dtypes

df.info()

df.describe()

df.isnull().values.any()

#Checking for null values

df.isnull().sum()

categorical=df.select\_dtypes(include='object')

print(len(categorical.columns))

numerical=df.select\_dtypes(include=['float64','int64'])

print(len(numerical.columns))

categorical.head()

numerical.head()

df=pd.DataFrame(df['Attrition'].value\_counts())

df

plt.pie(df['Attrition'],labels=['No','Yes'],explode=(0.2,0))

sn.set()

sn.distplot(df['Age'],color='red')

plt.show()

plt.figure(figsize=(10,5))

sn.set()

sn.distplot(df['DistanceFromHome'],color='green')

plt.show()

sn.set()

sn.distplot(df['TotalWorkingYears'],color='yellow')

plt.show()

sn.set()

sn.distplot(df['MonthlyIncome'],color='Orange')

plt.show()

sn.set()

sn.distplot(df['YearsAtCompany'],color='brown')

plt.show()

sn.set()

sn.distplot(df['YearsInCurrentRole'],color='Green')

plt.show()

sn.set()

sn.distplot(df['YearsSinceLastPromotion'],color='red')

plt.show()

sn.set()

sn.distplot(df['PercentSalaryHike'],color='yellow')

plt.show()

sn.factorplot(data=df,kind='count', size=6, aspect=3, x='BusinessTravel')

plt.show()

sn.factorplot(data=df,kind='count', size=6, aspect=3, x='Department')

plt.show()

sn.factorplot(data=df,kind='count', size=6, aspect=3, x='EducationField')

plt.show()

sn.factorplot(data=df,kind='count', size=6, aspect=3, x='Gender')

plt.show()

sn.factorplot(data=df,kind='count', size=6, aspect=3, x='JobRole')

plt.show()

sn.factorplot(data=df,kind='count', size=6, aspect=3, x='Over18')

plt.show()

sn.factorplot(data=df,kind='count', size=6, aspect=3, x='OverTime')

plt.show()

sn.factorplot(data=df,kind='count', size=6, aspect=3, x='MaritalStatus')

plt.show()

sn.factorplot(data=df,kind='count', size=6, aspect=3, x='Attrition')

plt.show()

df.drop(['EmployeeCount','EmployeeNumber'],axis=1)

attrition\_dummies=pd.get\_dummies(df['Attrition'])

attrition\_dummies.head()

df=pd.concat([df,attrition\_dummies],axis=1)

df.head()

df=df.drop(['Attrition','No'],axis=1)

df.head()

sn.barplot(x='Gender',y='Yes',data=df)

plt.show()

sn.barplot(x='Department',y='Yes',data=df)

plt.show()

sn.barplot(x='BusinessTravel',y='Yes',data=df)

plt.show()

plt.figure(figsize=(12,6))

sn.heatmap(df.corr())

plt.show()

df=df.drop(['Age','JobLevel'],axis=1)

from sklearn.preprocessing import LabelEncoder

for column in df.columns:

if df[column].dtype==np.number:

continue

else:

df[column]=LabelEncoder().fit\_transform(df[column])

df.dtypes

**BUILDING MACHINE LEARNING MODEL:-**

In a given dataset, target value in a categorical form. So I can use Classification algorithm. I will train 4 different models such as-

* Logistic Regression
* Decision Tree Classifier
* Random Forest Classifier

In this step, I should start modifying model parameters, perform feature engineering and balancing data strategies to improve the performance of the models.

**Model Validation:-**

Finally, after testing our models with the best set I conclude that best model was the Random Forest Classifier (RFC).Random Forest Classifier has the highest accuracy i.e. 85.26% of the prediction. The ROC curve (Receiver Operating Characteristic Curve) is also a good measure to choose the best model. AUC stands for area under the curve, and the larger this is the better the model. Applying the ROC curve node, I can visualize each ROC curve. The dotted line represents the ROC curve of a purely random classifier; a good classifier stays as far away from that line as possible (toward the top-left corner)

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

rf=RandomForestClassifier (n\_estimators=10,criterion='entropy',random\_state=0)

x=df.drop(['Yes'],axis=1)

y=df['Yes']

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.3,random\_state=0)

x\_train.head()

rf.fit(x\_train,y\_train)

rf.score(x\_train,y\_train)

pred=rf.predict(x\_test)

from sklearn.metrics import accuracy\_score

accuracy\_score(y\_test,pred)

**Accuracy for tested data= 85.26%**

from sklearn.linear\_model import LinearRegression

lr=LinearRegression()

lr.fit(x\_train,y\_train)

pred=lr.predict(x\_test)

from sklearn import metrics

print('Mean Absolute Error:', metrics.mean\_absolute\_error(y\_test, pred))

print('Mean Squared Error:', metrics.mean\_squared\_error(y\_test, pred))

print('Root Mean Squared Error:', np.sqrt(metrics.mean\_squared\_error(y\_test, pred)))

plt.scatter(x=y\_test,y=pred)

plt.show()

from sklearn.tree import DecisionTreeRegressor

dtr=DecisionTreeRegressor()

dtr.fit(x\_train,y\_train)

pred=dtr.predict(x\_test)

plt.scatter(x=y\_test,y=pred)

plt.xlabel('Y Test')

plt.ylabel('Predicted Y')

print('Mean Absolute Error:', metrics.mean\_absolute\_error(y\_test, pred))

print('Mean Squared Error:', metrics.mean\_squared\_error(y\_test, pred))

print('Root Mean Squared Error:', np.sqrt(metrics.mean\_squared\_error(y\_test, pred)))

from sklearn.ensemble import RandomForestRegressor

rdr = RandomForestRegressor()

rdr.fit(x\_train,y\_train)

pred=rdr.predict(x\_test)

print('MAE:', metrics.mean\_absolute\_error(y\_test, pred))

print('MSE:', metrics.mean\_squared\_error(y\_test, pred))

print('RMSE:', np.sqrt(metrics.mean\_squared\_error(y\_test, pred)))

sn.distplot((y\_test-pred),bins=50)

plt.show()

data = pd.DataFrame({'Y Test':y\_test , 'Pred':pred},columns=['Y Test','Pred'])

sn.lmplot(x='Y Test',y='Pred',data=data,palette='rainbow')

data.head()

#Plotting scatter plot between test data and predicted data for Ada Boost

plt.scatter(y\_test,pred)

plt.xlabel('y\_test')

plt.ylabel('predicted y')

plt.title("Scatter plot between testdata and predicted data")

plt.show()

**CONCLUDING REMARKS:-**

Now I have our dataset with our current employees and their probability of leaving the company. If I was the HR manager of our company, I would require a dashboard in which I could see what to expect regarding future attrition and hence adopt the correct strategy to retain the most talented employees. It contains analysis on percentage of predicted attrition, analysis by Gender, BusinessTravel, Department and DistanceFromHome. As a quick conclusion, male employees who travel frequently, work at HR department, have a low salary and live far from workplace have a high probability of leaving the company.