HR Analytics Project (Project 4)

Name = Chandrima Chatterjee

Introduction

Every year a lot of companies hire a number of employees. The companies invest time and money in training those employees, not just this but there are training programs within the companies for their existing employees as well. The aim of these programs is to increase the effectiveness of their employees.

HR Analytics
Human resource analytics (HR analytics) is an area in the field of analytics that refers to applying analytic processes to the human resource department of an organization in the hope of improving employee performance and therefore getting a better return on investment. HR analytics does not just deal with gathering data on employee efficiency. Instead, it aims to provide insight into each process by gathering data and then using it to make relevant decisions about how to improve these processes.

Attrition in HR
Attrition in human resources refers to the gradual loss of employees over time. In general, relatively high attrition is problematic for companies. HR professionals often assume a leadership role in designing company compensation programs; work culture and motivation systems that help the organization retain top employees.

Attrition affecting Companies
A major problem in high employee attrition is its cost to an organization. Job postings, hiring processes, paperwork and new hire training are some of the common expenses of losing employees and replacing them. Additionally, regular employee turnover prohibits your organization from increasing its collective knowledge base and experience over time. This is especially concerning if your business is customer facing, as customers often prefer to interact with familiar people. Errors and issues are more likely if you constantly have new workers.

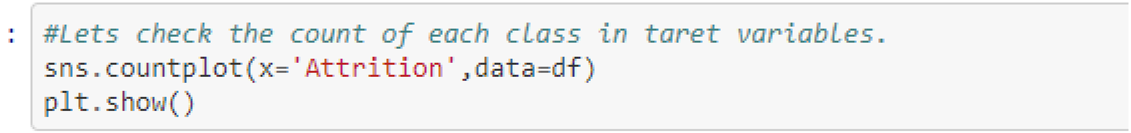
Problem Statement

Employee turnover (also known as "employee's month") is a costly problem for companies. The actual cost of replacing an employee from the organization with the new employee.
The actual cost of replacing employees remains outstanding for most of the employers. This is because of the amount of time spent on interviews and replacements, sign-on bonuses, and several months of reduced productivity, and new employees getting used to the new role.

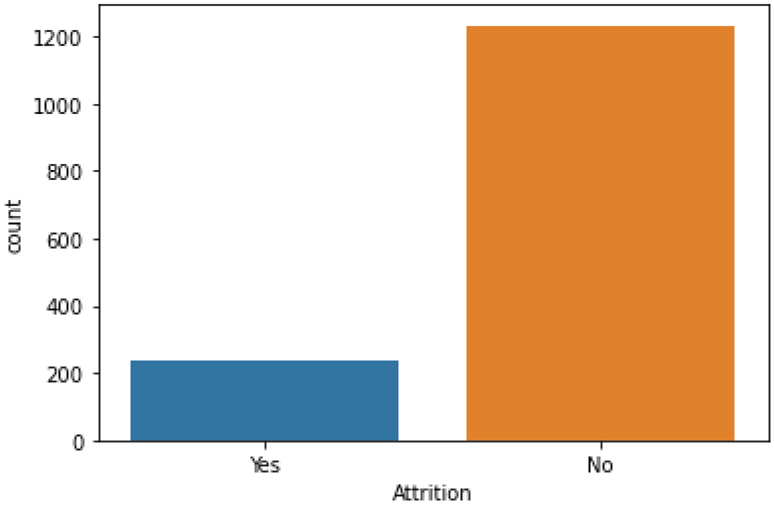
So the questions comes in our mind is following.

How does Attrition affect companies? and how does HR Analytics help in analyzing attrition? Here I will be using a step-by-step systematic approach using a method that could be used for a variety of ML problems. This project would fall under what is commonly known as HR Analytics or People Analytics.

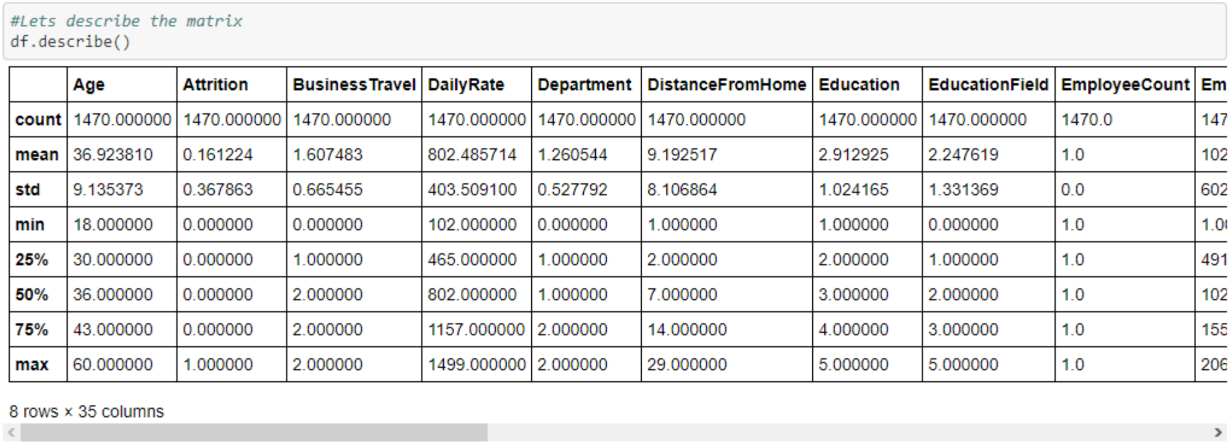
Data Analysis



```
: #Lets check the count of each class in taret variables.
  sns.countplot(x='Attrition',data=df)
  plt.show()
```



Observation: -we can see that attrition of employees(19%) is quiet low.

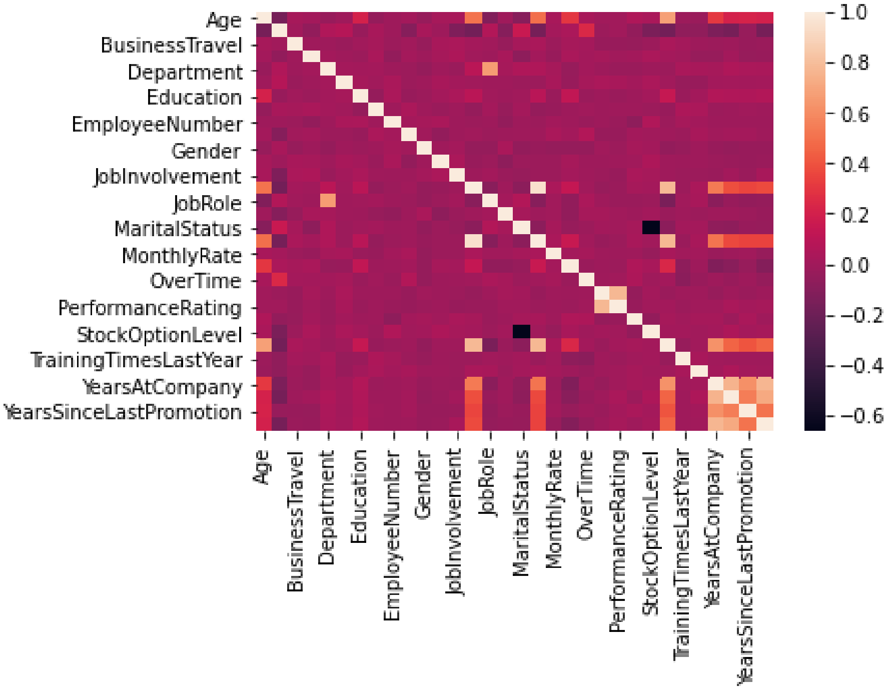


```
#Lets describe the matrix
df.describe()
```

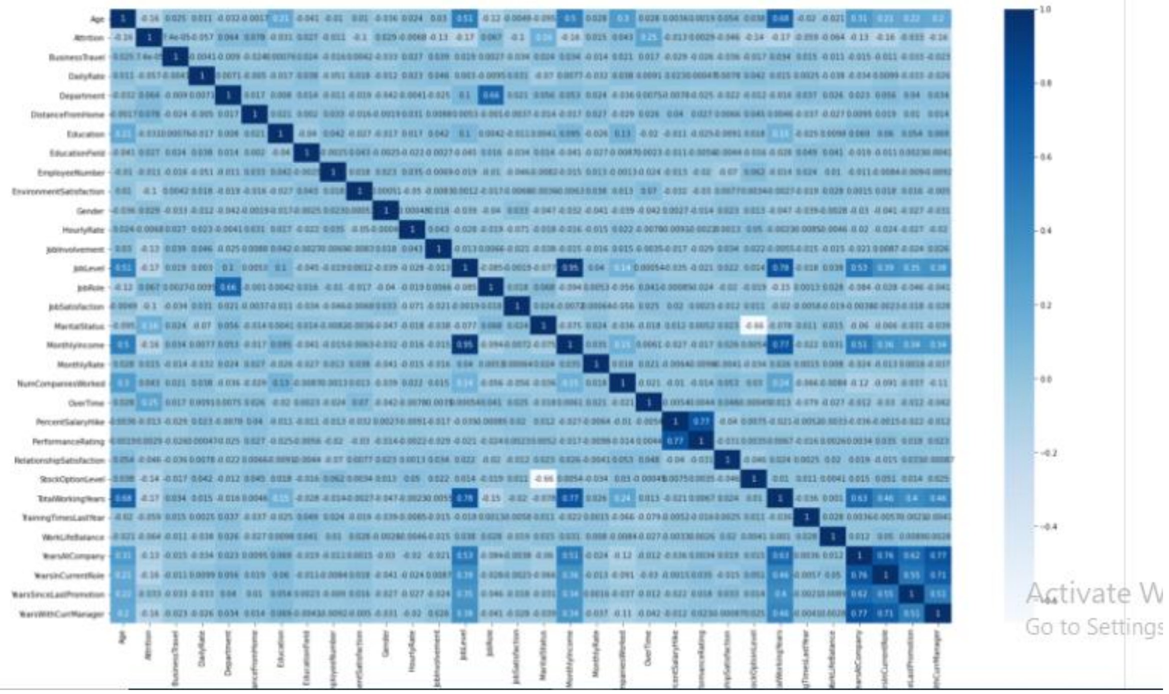
	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	Em
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	147
mean	36.923810	0.161224	1.607483	802.485714	1.260544	9.192517	2.912925	2.247619	1.0	102
std	9.135373	0.367863	0.665455	403.509100	0.527792	8.106864	1.024165	1.331369	0.0	602
min	18.000000	0.000000	0.000000	102.000000	0.000000	1.000000	1.000000	0.000000	1.0	1.0
25%	30.000000	0.000000	1.000000	465.000000	1.000000	2.000000	2.000000	1.000000	1.0	491
50%	36.000000	0.000000	2.000000	802.000000	1.000000	7.000000	3.000000	2.000000	1.0	102
75%	43.000000	0.000000	2.000000	1157.000000	2.000000	14.000000	4.000000	3.000000	1.0	155
max	60.000000	1.000000	2.000000	1499.000000	2.000000	29.000000	5.000000	5.000000	1.0	206

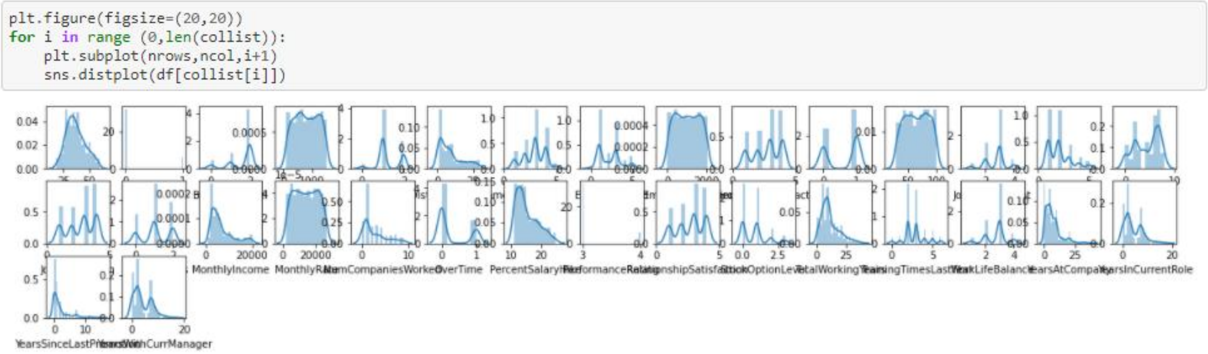
8 rows × 35 columns

```
: #Checking correlation with the help of heatmap.
  sns.heatmap(dfcor)

: <matplotlib.axes._subplots.AxesSubplot at 0x1511b4f0580>
```

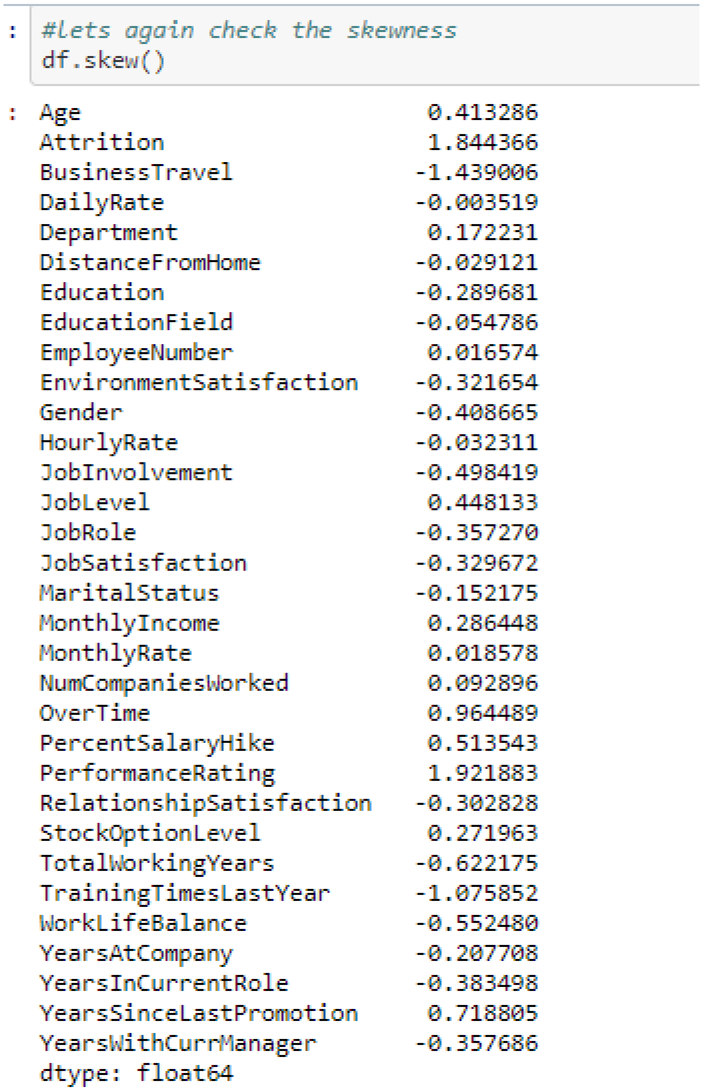


```
In [23]: plt.figure(figsize = (25,15))
         sns.heatmap(dfcor,cmap = 'Blues',annot = True)

Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x1511b4938b0>
```





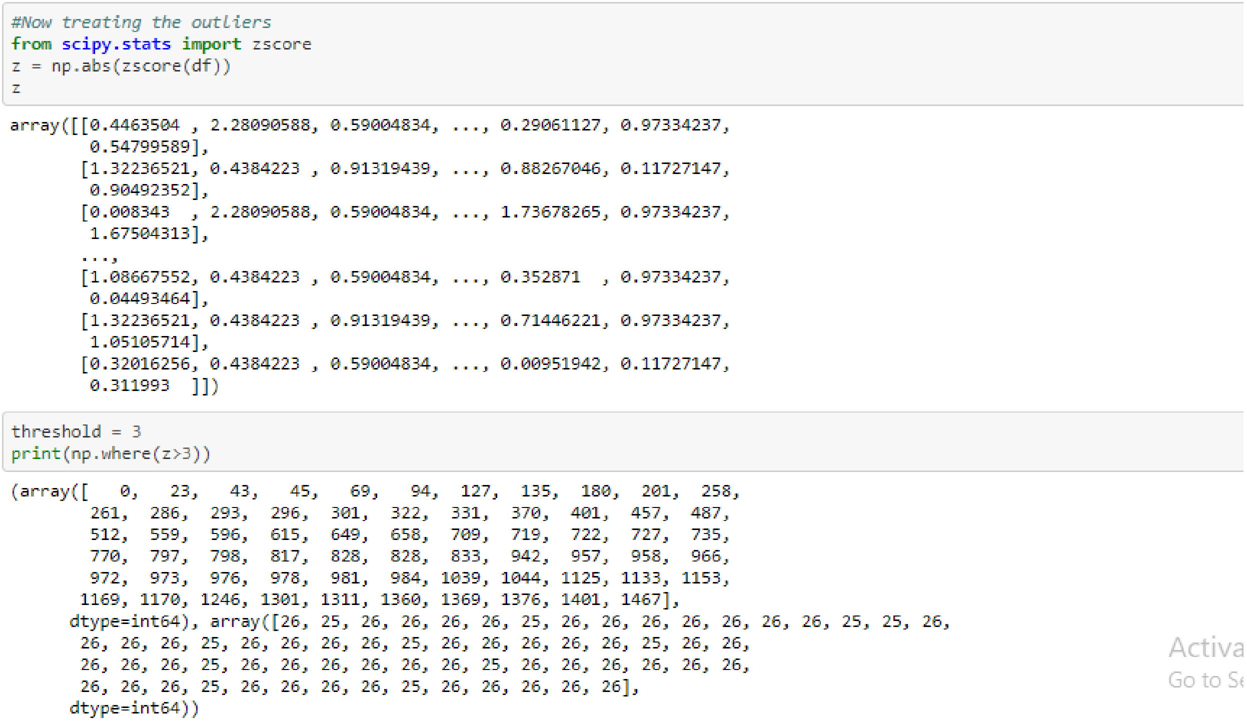
```
plt.figure(figsize=(20,20))
for i in range (0,len(collist)):
    plt.subplot(nrows,ncol,i+1)
    sns.distplot(df[collist[i]])
```



```
: #lets again check the skewness
  df.skew()
```

```
: Age                         0.413286
  Attrition                   1.844366
  BusinessTravel             -1.439006
  DailyRate                  -0.003519
  Department                  0.172231
  DistanceFromHome           -0.029121
  Education                  -0.289681
  EducationField             -0.054786
  EmployeeNumber              0.016574
  EnvironmentSatisfaction    -0.321654
  Gender                     -0.408665
  HourlyRate                 -0.032311
  JobInvolvement             -0.498419
  JobLevel                    0.448133
  JobRole                    -0.357270
  JobSatisfaction            -0.329672
  MaritalStatus              -0.152175
  MonthlyIncome               0.286448
  MonthlyRate                 0.018578
  NumCompaniesWorked          0.092896
  OverTime                    0.964489
  PercentSalaryHike           0.513543
  PerformanceRating           1.921883
  RelationshipSatisfaction   -0.302828
  StockOptionLevel            0.271963
  TotalWorkingYears          -0.622175
  TrainingTimesLastYear      -1.075852
  WorkLifeBalance            -0.552480
  YearsAtCompany             -0.207708
  YearsInCurrentRole         -0.383498
  YearsSinceLastPromotion     0.718805
  YearsWithCurrManager       -0.357686
  dtype: float64
```

Preprocessing Pipeline



```
#Now treating the outliers
from scipy.stats import zscore
z = np.abs(zscore(df))
z
```

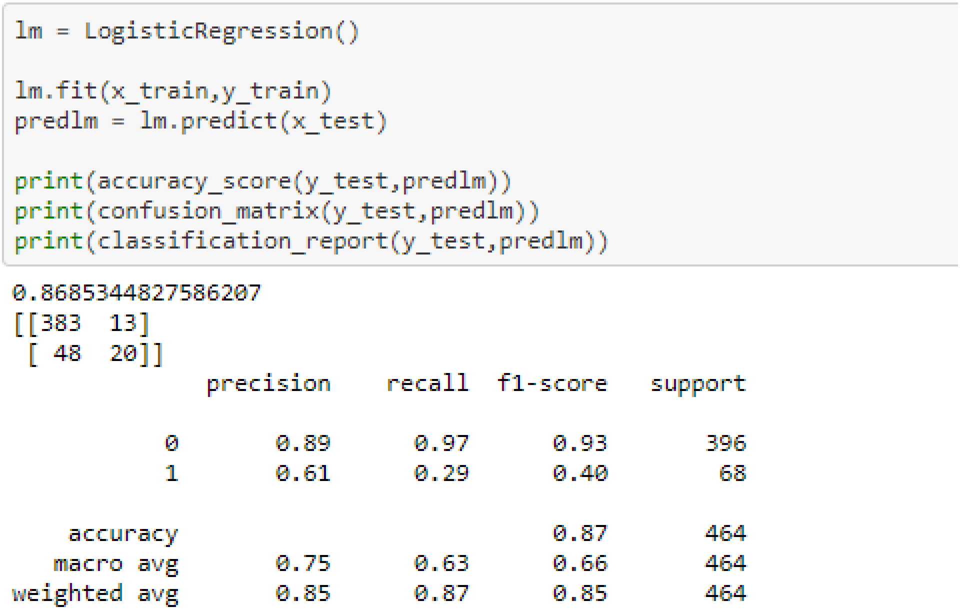
```
array([[0.4463504 , 2.28090588, 0.59004834, ..., 0.29061127, 0.97334237,
         0.54799589],
       [1.32236521, 0.4384223 , 0.91319439, ..., 0.88267046, 0.11727147,
         0.90492352],
       [0.008343  , 2.28090588, 0.59004834, ..., 1.73678265, 0.97334237,
         1.67504313],
       ...,
       [1.08667552, 0.4384223 , 0.59004834, ..., 0.352871  , 0.97334237,
         0.04493464],
       [1.32236521, 0.4384223 , 0.91319439, ..., 0.71446221, 0.97334237,
         1.05105714],
       [0.32016256, 0.4384223 , 0.59004834, ..., 0.00951942, 0.11727147,
         0.311993  ]])
```

```
threshold = 3
print(np.where(z>3))
```

```
(array([   0,   23,   43,   45,   69,   94,  127,  135,  180,  201,  258,
         261,  286,  293,  296,  301,  322,  331,  370,  401,  457,  487,
         512,  559,  596,  615,  649,  658,  709,  719,  722,  727,  735,
         770,  797,  798,  817,  828,  828,  833,  942,  957,  958,  966,
         972,  973,  976,  978,  981,  984, 1039, 1044, 1125, 1133, 1153,
        1169, 1170, 1246, 1301, 1311, 1360, 1369, 1376, 1401, 1467],
      dtype=int64), array([26, 25, 26, 26, 26, 26, 25, 26, 26, 26, 26, 26, 26, 26, 25, 25, 26,
        26, 26, 26, 25, 26, 26, 26, 26, 25, 26, 26, 26, 26, 26, 25, 26, 26,
        26, 26, 26, 25, 26, 26, 26, 26, 26, 26, 25, 26, 26, 26, 26, 26, 26,
        26, 26, 26, 25, 26, 26, 26, 26, 25, 26, 26, 26, 26, 26],
      dtype=int64))
```

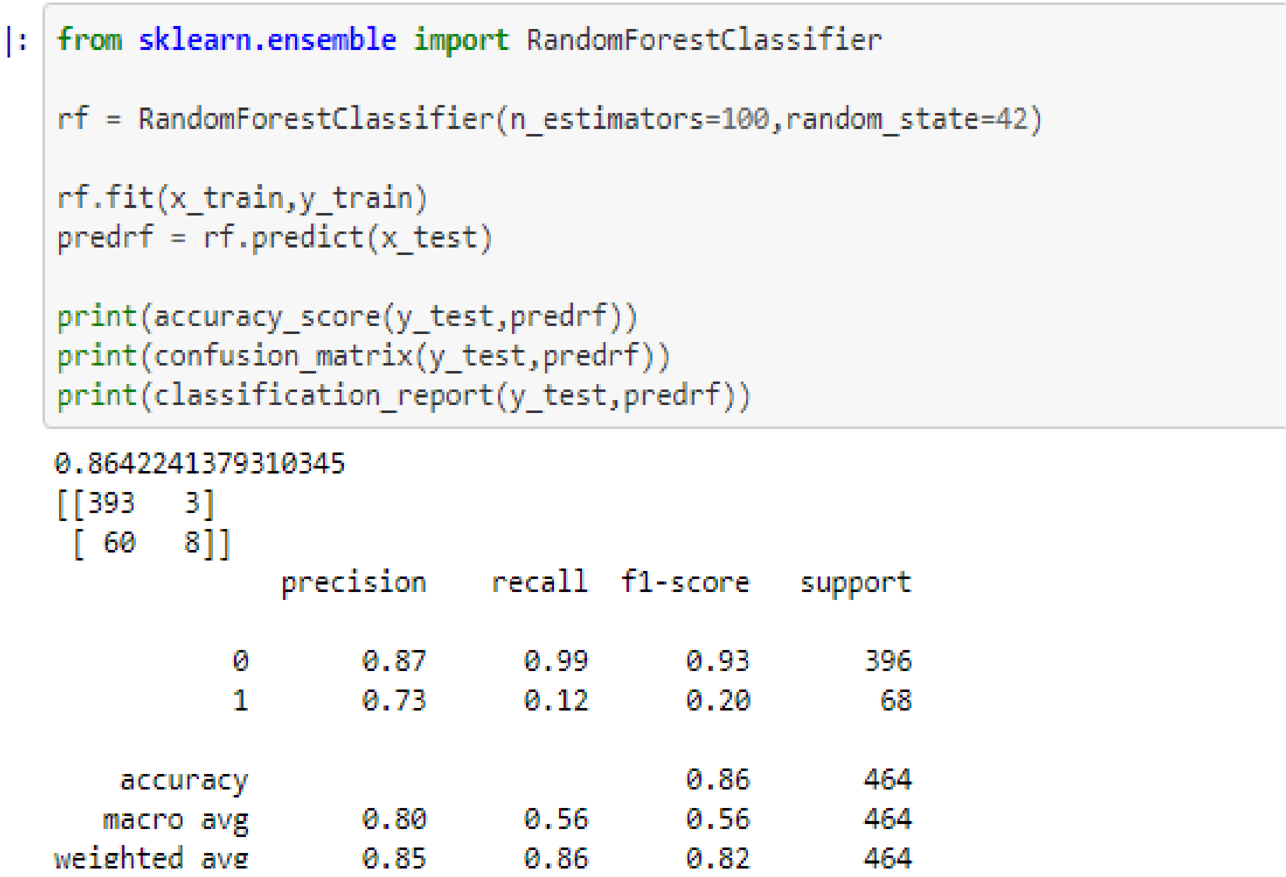
Activa
Go to Se

Building Machine Learning Models



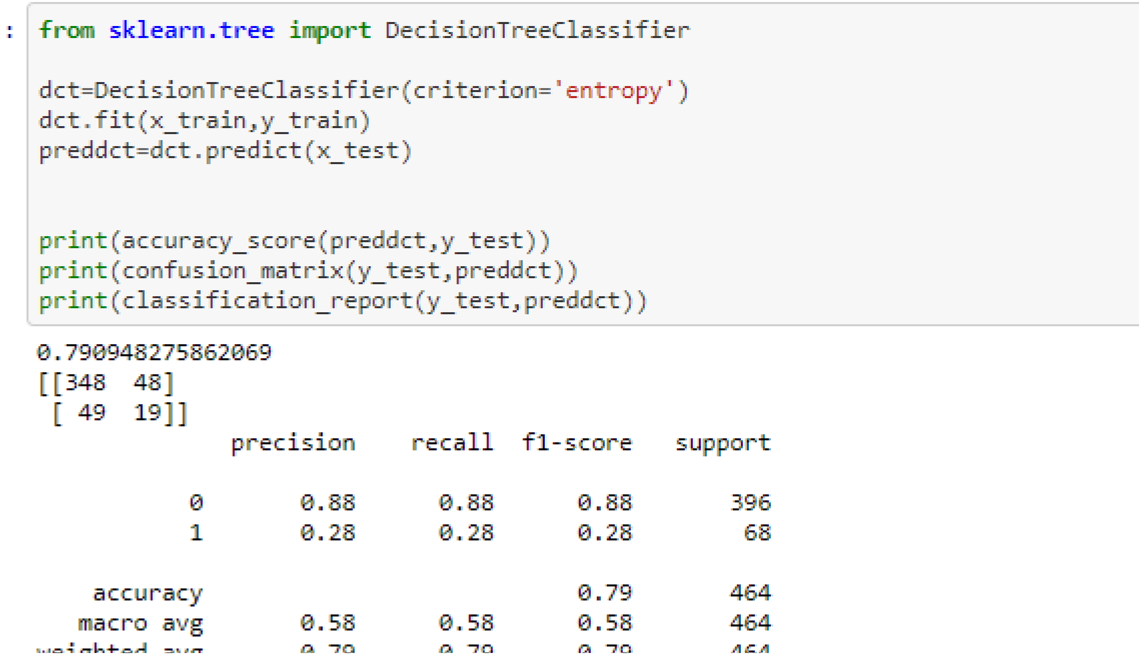
```
lm = LogisticRegression()

lm.fit(x_train,y_train)
predlm = lm.predict(x_test)

print(accuracy_score(y_test,predlm))
print(confusion_matrix(y_test,predlm))
print(classification_report(y_test,predlm))
```

```
0.8685344827586207
[[383  13]
 [ 48  20]]
              precision    recall  f1-score   support

           0       0.89      0.97      0.93       396
           1       0.61      0.29      0.40        68

    accuracy                           0.87       464
   macro avg       0.75      0.63      0.66       464
weighted avg       0.85      0.87      0.85       464
```



```python
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier(n_estimators=100,random_state=42)

rf.fit(x_train,y_train)
predrf = rf.predict(x_test)

print(accuracy_score(y_test,predrf))
print(confusion_matrix(y_test,predrf))
print(classification_report(y_test,predrf))
```

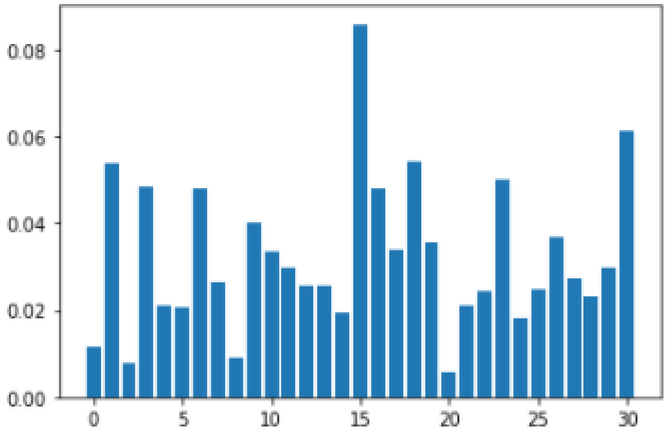
```
0.8642241379310345
[[393   3]
 [ 60   8]]
              precision    recall  f1-score   support

           0       0.87      0.99      0.93       396
           1       0.73      0.12      0.20        68

    accuracy                           0.86       464
   macro avg       0.80      0.56      0.56       464
weighted avg       0.85      0.86      0.82       464
```

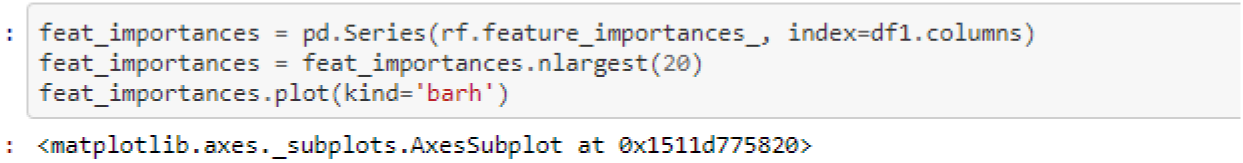


```
:  from sklearn.tree import DecisionTreeClassifier

   dct=DecisionTreeClassifier(criterion='entropy')
   dct.fit(x_train,y_train)
   preddct=dct.predict(x_test)


   print(accuracy_score(preddct,y_test))
   print(confusion_matrix(y_test,preddct))
   print(classification_report(y_test,preddct))
```

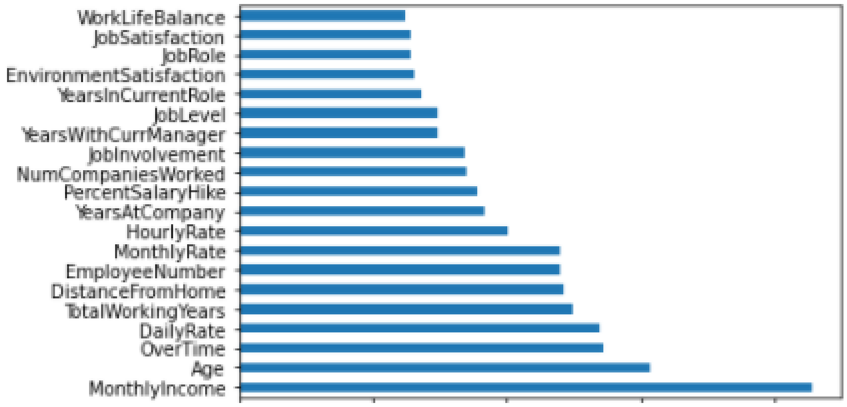
```
0.790948275862069
[[348  48]
 [ 49  19]]
              precision    recall  f1-score   support

           0       0.88      0.88      0.88       396
           1       0.28      0.28      0.28        68

    accuracy                           0.79       464
   macro avg       0.58      0.58      0.58       464
weighted avg       0.79      0.79      0.79       464
```

```
from matplotlib import pyplot
pyplot.bar([x for x in range(len(importance))], importance)
pyplot.show()
```

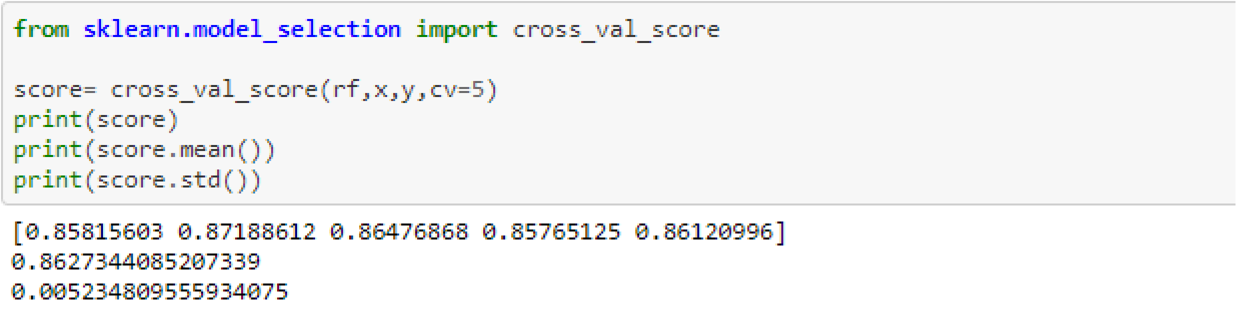




```
: feat_importances = pd.Series(rf.feature_importances_, index=df1.columns)
  feat_importances = feat_importances.nlargest(20)
  feat_importances.plot(kind='barh')

: <matplotlib.axes._subplots.AxesSubplot at 0x1511d775820>
```

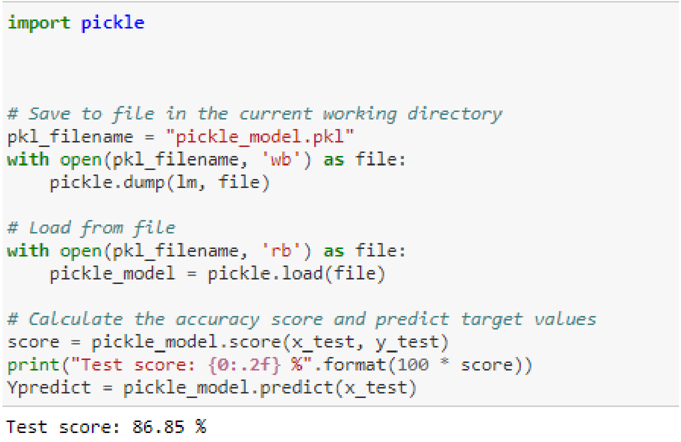


Cross Validating

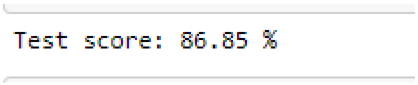


```
from sklearn.model_selection import cross_val_score

score= cross_val_score(rf,x,y,cv=5)
print(score)
print(score.mean())
print(score.std())

[0.85815603 0.87188612 0.86476868 0.85765125 0.86120996]
0.8627344085207339
0.005234809555934075
```

Conclusion

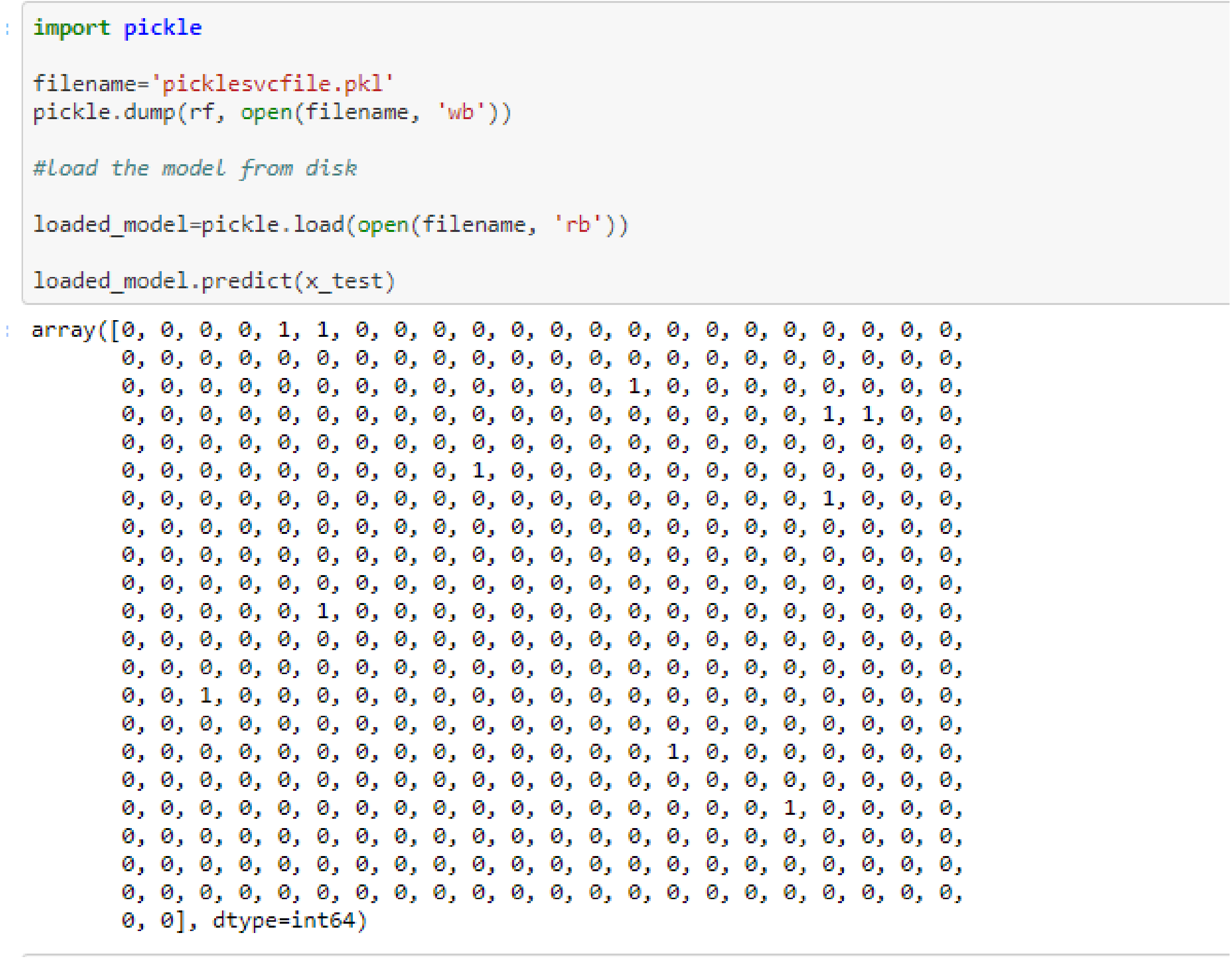


```
import pickle


# Save to file in the current working directory
pkl_filename = "pickle_model.pkl"
with open(pkl_filename, 'wb') as file:
    pickle.dump(lm, file)

# Load from file
with open(pkl_filename, 'rb') as file:
    pickle_model = pickle.load(file)

# Calculate the accuracy score and predict target values
score = pickle_model.score(x_test, y_test)
print("Test score: {0:.2f} %".format(100 * score))
Ypredict = pickle_model.predict(x_test)

Test score: 86.85 %
```



```
Test score: 86.85 %
```

Conclusion



```
import pickle

filename='picklesvcfile.pkl'
pickle.dump(rf, open(filename, 'wb'))

#load the model from disk

loaded_model=pickle.load(open(filename, 'rb'))

loaded_model.predict(x_test)
```

```
array([0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 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, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 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, 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, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 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, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 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, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 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,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 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, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0], dtype=int64)
```

In conclusion, we can say that HR attrition rate is major issue in every organization because it plays with people mindset. With the help of machine learning algorithms, we can minimize the spread. Also logistic regression is the best model for performance where I got **86.85%** of accuracy among of the entire performing machine learning model. This model can be used for detecting HR attrition rate.