Sayan Chatterjee

Prediction of Attrition of Employees Using Machine Learning

Here this given problem is actually a **Class imbalance** problem. No of samples with Attrition '1' in the training data is very much less compared to the no of samples with target variable 'Attrition' = 0.That's why it is having a good accuracy even if a model predicts all the outputs to be 0. The objective here is to fit a unbiased model that just not only predicts for 0 always, that is rather than random guessing, it does predict well in general by some suitable method ! That is why it is very important to select a metric first, as the metric **accuracy** may **not** do well on the entire dataset!

Observations:

1. It is a class imbalance problem. The class 0 is more likely than the class 1.
   (The frequency of class 1 in training set is 172, the frequency of class 0 in training set is 856)

   It is also classification problem of supervised learning as the labels are known for the training data.

2. The features given in the data set contains some features which seem to be not related in predicting the Attrition**.**

   The first one is the '**ID'.** This is just like a serial number, nothing else. It is just an identification number associated to each sample i.e. each employee. So, it is better to drop the column here.

   The second such column is **"Employee Number".** It is very similar to the previous one. It seems to be not related to the predicted "Attrition".

   The third feature is **"Employee Count".** This is another unrelated feature. It is 1 in the training set for all features hence the effect is nullified at the end.

   So, these three features are not useful ones.

3. All the features are not in appropriate format. Some of the features are categorical in nature. The supervised learning models can't be applied to these categorical features. Hence, it is required to convert those categorical ones to these numerical one.

4. There are no missing values here. This makes things a little bit

   easier. (Discussed in preprocessing part also).

5. It is always good to observe some basic statistics for all the features and the target variable. This is given below.

   This is a part of the visual EDA. It gives a good understanding of the data: whether the data is very widely spread or not, which values are more likely to take place, where does the mean lie etc. It also gives the info regarding any missing value. Here all the counts for all the features and the target variable gives a count = no of data points**, hence, it is a proof that there is no missing value here.**

	Age	Attrition	DailyRate	DistanceFromHome	Education
count	1028	1028	1028	1028	1028
mean	36.999027	0.167315	806.551556	9.010700	2.87
std	9.444297	0.373439	407.043735	8.078418	1.03
min	18.00000	0.000000	102.000000	1.000000	1.00
25%	30.000000	0.000000	465.750000	2.000000	2.00
50%	36.000000	0.000000	813.000000	7.000000	3.00
75%	43.000000	0.000000	1157.250000	13.00000000	4.00
max	60.000000	1.000000	1499.000000	29.000000	5.00

	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	HourlyRate
count	1028.0	1028.000000	1028.000000	1028.000000
mean	1.0	710.198444	2.719844	65.451362
std	0.0	418.513656	1.089614	20.274229
min	1.0	1.000000	1.000000	30.000000
25%	1.0	351.750000	2.000000	48.000000
50%	1.0	701.500000	3.000000	65.000000
75%	1.0	1069.250000	4.000000	83.000000
max	1.0	1447.000000	4.000000	100.000000

NOTE:

All the features and target value's basic statistics are not mentioned here. This is due to the reason that there are so many features and things become very noisy if we print all here. For any other feature, please refer to the coding file 'preprocessing and visualization .ipynb'.

Preprocessing:

1. The first step of preprocessing is to deal with variety of features. Some of the features are categorical here, they need to be converted to numerical ones by suitable techniques.

Luckily there is the option “get_dummies” available in pandas. This solves the issue. After doing that we are sure that all we have in our data are all numerical values.

Before getting dummies: The features and corresponding datatype looks like this:

Feature	Datatype
Age	Integer
BusinessTravel	Object
DailyRate	Integer
Department	Object
DistanceFromHome	Integer
Education	Integer
EducationField	Object
EmployeeCount	Integer
EmployeeNumber	Integer
EnvironmentSatisfaction	Integer
Gender	Object
HourlyRate	Integer
JobInvolvement	Integer
JobLevel	Integer
JobRole	Object

JobSatisfaction	Integer
MaritalStatus	Object
MonthlyIncome	Integer
MonthlyRate	Integer
NumCompaniesWorked	Integer
OverTime	Object
PercentSalaryHike	Integer
RelationshipSatisfaction	Integer
StockOptionLevel	Integer
TotalWorkingYears	Integer
TrainingTimesLastYear	Integer
WorkLifeBalance	Integer
YearsAtCompany	Integer
YearsInCurrentRole	Integer
YearsSinceLastPromotion	Integer
YearsWithCurrManager	Integer
ID	Integer

Here after getting dummies, all the datatype of type "object" (categorical ones) are getting converted to numerical form. Moreover, no of features also increased! This is due to the reason that the feature named "BusinessTravel" previously had 3 values namely frequently, rarely, nontravel. Now from one BusinessTravel feature, various features are extracted namely,

1. BusinessTravel_Non-Travel
2. BusinessTravel_Travel_Frequently
3. BusinessTravel_Travel_Rarely

All of them are binary in nature that means BusinessTravel_Non-Travel will have 0 or 1. 0 means BusinessTravel_Non-Travel is false, 1 means it is true. So for each data point, exactly one feature out of the 3 will have 1, rest all will be 0.

This sums up the entire conversion from Categorical features to Numerical Values.

(Note: This conversion is done in both the training set as well as test set.)

Step-2:

There is no missing values here, hence there is no imputation technique to be used.

Step-3

1. One point to note is that in step-1, there is some columns that may still be discarded. Consider the features derived from BusinessTravel; if for an employee, if he/she does not belong to BusinessTravel_Travel_Frequently ,BusinessTravel_Travel_Rarely , th en he/she must belong to BusinessTravel_Non-Travel . So, it may be okay to remove t hat column 3. Similar elimination can be done on the other features as well !

Step-4:

Balancing class imbalance:

One thing that can be done is to increase the no of data points of the rare class by either dupli cating data points or drawing sample from the underlying distribution. I have tried this by dup licating the data points and tried to fit the models on the data points. Unfortunately, this did n ot work out.

The other way is to decrease the no of data points of class 0 and match no of samples with **class** 0 and **class** 1. What it does is that the training points reduce very significantly, hence no t acceptable here and I did not try it out.

Exploratory Data Analysis: (EDA):

EDA is important in any kind of dataset. For this dataset also, to get the basic understanding of the dataset, I referred to various EDA techniques.

1. Finding out the missing values and fill it up with suitable technique.
   (Not required as already discussed)

2. Identification of variables and data types (already discussed)

3. Analysing various mean, median and other parameters to get some insight of the data. (Already discussed)

4. **Graphical Analysis of various features and the target variable.**

In this stage, some graphical analysis (univariate or bivariate) can be carried out**.**

Here histogram is plotted for all the possible features and the target variable. Kindly refer to that in the ‘preprocessing and visualization .ipynb’ file.

Choice of Metric: Accuracy is a not an appropriate metric here. **ROC AUC** is the main metric considered here.

List of various approaches used from the start of the competition to the final approach:

Various form of KNN to start with:

1. KNN is the first one that I started with. This is due to the reason that this is very easy to implement and only single hyper-parameter is required to be tuned here (k). With KNN, first I applied on the training data (except Employee Number , Employee Count, ID) using splitting the training dataset into two groups, validation and training. For validation, 20% of the sample was separated out. Here I tuned this to for the best value of k for which metric (area under ROC curve) is maximized. It turned out that the optimum k becomes 7. After that, I applied KNN with k=7 on the test set and got the result which showed somewhere around 88.383 percent accuracy.
2. Now I did scaling on the features and applied the same technique as of (1).

   Unfortunately, the metric on the validation set did not improve. .

Random Forest: (RF)

Metric selected – **ROC- AUC**

After KNN, I applied Random Forest in the hope of getting something better. Here it is an ensemble method and expected something good.

3. I did RF on the entire set of features (except EmployeeNumber , EmployeeCount, ID),separated 20% as validation set and tested it on the validation dataset! The hyperparameters are tuned now but still the metric on the validation set did not increase compared to method1 that I applied.

After applying Random Forest and KNN, I felt that **scaling overall** did not help that much here.

So, for the later models, I applied techniques on the maximum set of features (except ID, Employee Number, Employee Count) and used Cross Validation technique to find the best model and corresponding hyper-parameters for the model. After finding that I used it on the test data.

SVM:

4. The next model I used is Support Vector Machine. It is applied on the larger subset of features and tuned again. Kernel type, value of C are tuned here. Here also I used 5 fold cross validation and found that result is not improved. For the test set, SVM did not work out well.

Tuned hyper-parameters are:

C=0.001, gamma=100, kernel= ‘rbf’

Logistic Regression:

5. It is applied the same way where most of the features are taken into account (expect Employee Number , Employee Count, ID) and Logistic Regression is fitted on the training part (no on the validation) of it.
   Here the cross validated result on the training dataset improved. It is basically a 5 fold cross validation. Hence, I applied it on the test dataset and got an accuracy of 89.898 %.

6. Here I used Logistic Regression on the scaled set of features. Here, the performance metric dropped. Hence, I did not use it on the test set.

Boosting: (Gradient Boosting Classifier)

Here I started with the following parameters and later tuned most of the hyperparameters.

learning_rate=0.1, min_samples_leaf=50, max_depth= 5, min_samples_split=550, max_features='sqrt',subsample=0.8, random_state=42, **scoring='roc_auc'**, cross_validation_no of folds =5

The metric is **“ROC-AUC”.**

I fixed both scoring and random state and tuned all other parameters one after another. It would be great if I could tune all at the same time but I could not because that would be a huge no of iterations (10*5*98*20*8*6*8=37632000) that is very difficult to execute and very large amount of resource is required.

So, The classifier comes out to be,

Gradient Boosting Classifier

Performance metric**: ROC AUC** is the metric here.

This cross validated result on the trained set is way better than the previous ones.

EVALUATION AND SUMMARY

TEST ACCURACY

Now it's time to use this classifier on the test dataset, this accuracy is also the highest as expected. The accuracy comes out to be 97.36%

Summary:

Objective: To estimate the attrition of employees using various machine Learning Techniques

- **Executed data preprocessing and implemented KNN, SVM, Decision Tree, Random Forest on class imbalance dataset**
- **Carried out suitable Data Visualization, Data Preprocessing and Exploratory Data Analysis on class imbalance dataset**
- **Examined K-Nearest Neighbors, Support Vector Machines, Decision Tree and Random Forest on the given dataset**
- **Achieved 97.36% accuracy on test dataset during classification using ensemble technique(Gradient Boosting classifier)**