Predicting Employee Attrition -

APPROACH PDF

Train Data

Variable	Definition
MMMM-YY	Reporting Date (Monthly)
Emp_ID	Unique id for employees
Age	Age of the employee
Gender	Gender of the employee
City	City Code of the employee
Education_Level	Education level : Bachelor, Master or College
Salary	Salary of the employee
Dateofjoining	Joining date for the employee
LastWorkingDate	Last date of working for the employee
Joining Designation	Designation of the employee at the time of joining
Designation	Designation of the employee at the time of reporting
Total_Business_Value	The total business value acquired by the employee in a month (negative business indicates cancellation/refund of sold insurance policies)
Quarterly Rating	Quarterly rating of the employee: 1,2,3,4 (higher is better)

Test Data

Variable	Definition
Emp_ID	Unique Id for the employees

Business Problem:

To aid staffing, you are provided with the monthly information for a segment of employees for 2016 and 2017 and tasked to predict whether a current employee will be leaving the organization in the upcoming two quarters (01 Jan 2018 - 01 July 2018) or not, given:

1. Demographics of the employee (city, age, gender etc.)
2. Tenure information (joining date, Last Date)
3. Historical data regarding the performance of the employee (Quarterly rating, Monthly business acquired, designation, salary)

➔Let's Talk about the Approach we took to solve the Business Problem: -

1. Brief on the Approach:

- After looking at the train file the first thing that is noticeable is we don't have a clear Target feature, so we have to fetch that particular target feature using the features which have already been provided to us.
  So did some exploratory data analysis, and found that the last working date is the only particular column having some NaN entries. So there were multiple entries for a single employee ID according to the months so only the last entry of the particular employee ID has the corresponding last working date.
  So the idea is let's not take all the rows for any particular employee ID, let's just somehow try and compress all the rows of that particular employee ID into one row.
  After compressing we can clearly notice that we can make our target feature as last working date minus date of joining that would be nothing but the tenure of that particular person, and clearly we can figure out that if this tenure is less than the two quarters we can say that the person is more likely to leave the organization at an early stage otherwise not.

2. Data-preprocessing / Feature engineering ideas:

- Talking about the part of data pre-processing, if we talk about data cleaning part and particularly the NaN handling, so we only had Nan entries in last working day column, and as we were taking only the last occurrence of the row for the particular employee ID in order to gain the last working day column information, by this most of the NaN were already removed, but the problem was there were certain employee IDs which does not had the corresponding last working day information, so we tried

looking into the another column that was basically MMMM-YY column and tried to put last working day according to this column +15 more days.

- So, this was the way by which we handled all the Nan entries, was not a huge task as we did not have that many Nan entries.
- We also Tried visualising the data using different graphs as in we tried looking for the outliers by plotting histogram, probability plot and Box plot.
- So, we tried making two columns out of the salary. The first occurrence would be the starting salary and the last occurrence would be the ending salary and lately we tried converting the ending salary to the grace in salary where each value would be the ending salary minus the starting salary. Because at most of the places ending salary is equivalent to starting salary.
- The Second feature we created was using the date of joining in the last working date and we created tenure in days using the subtraction of both the features. The second feature was basically the target feature for our machine learning model.
- So, in this project basically I have used the regression machine learning algorithms because I would be using the continuous values as my target feature values to predict. And later on based on certain conditions I would be converting those predicted values to 0 and 1.
- Lastly, we also did categorical variable encoding over the three categorical features we had that were gender city and education level.
  So, for gender and city we use simply dummy variable encoding using pandas dummy function.
- But as education level had three values of being an employer in college or had done bachelors or had done Masters, all the three values clearly denoted three different levels, so we used ordinal encoding on this particular feature.
- After the categorical variable encoding was done, we copied that data frame into a new data frame and named it data frame with employee ID, because we would be using this particular Data frame to fetch the information of each employee ID that would be provided to us in the testing file and then lastly, we would be performing the prediction using our machine learning algorithm on the data that we would be fetching from here and saving it into the CSV file.
- The Idea is we would drop the employee ID feature from the encoded data frame and we would be changing this particular data frame to X and Y

3. Final model:

- Lastly after the features scaling part the only thing that is remaining is training the model.

- So, we made a data frame which will show us the model's name and its corresponding root mean square error value and also its train and test score.
- On the basis of which we would be charging which algorithm should we use.
- So, we observed that Lasso was performing pretty well, we tried to optimise the alpha value, it didn't get optimised as expected and the score didn't change that much.
- Support vector machine was the one which was performing the worst over there
- Decision trees completely over fitted the data and provided us the training score as 1.
- The 2 algorithms Random Forest and XG boost were performing very well but they were also quite near to overfitting.
- We tried optimising the random forest first and clearly got a good score along with reduced overfitting.
- Then we tried hyper parameter tuning over XG boost and we were able to get the best score out here along with their reduced overfitting.
- We also tried checking the cross-validation score of the particular XG boost algorithm model and it was also a good score.
- Lastly, we imported the test file and we fetched the information related to that particular employee ID provided in the test file
- We predicted the tenure for the same information that we just fetched.
- We would be converting these predicted values to zeros and ones using a particular condition if the predicted tenure is less than or equal to the number of days in quarters, we would predict one else we would be predicting zero.
- Then we would be saving the employee IDs and their corresponding target in zeros and one form into a CSV file that would be our prediction.CSV file
- As we have also checked F1 score using the truths and predicts both in the form of zero and one. And came out to be a good score.

Overall, the project was pretty good, fetching the target feature was quite challenging.
There were very less number of NaN entries in the whole data frame so handling NaN was not that tough. Also, the brainstorming for creating new features out of the features that we were provided was also pretty much interesting.

Thank You,
Nikunj Dhaka