Prediction of Submission Rate

Jen Huang

15 April, 2020

Project Description

There are two data sets called `train.csv` and `test.csv`.

In the `train.csv` data set is the information of the employees and whether they have left the job. The statistical information includes salary, business trip, acceptable working environment, degree of work input, whether to work overtime, whether to be promoted, salary increase ratio, etc. Our goal is to predict whether a employee would left the job through collected data in `test.csv`. For more information of this project, please see the URL behind:

`https://www.kaggle.com/c/rs6-attrition-predict`

Data Overview

0.1 A quick look

The file `train.csv` contains data with 1176 rows and 36 columns. The file `test.csv` contains data with 294 rows and 35 columns. Both files has complete and non-null entries, but on the other side, the data gets two types: int64 and object. Therefore, when we clean the data, we need label encode or One-Hot encode the object-type features. The one column between two files is the "Attrition", that is the target.

The features are: Age, Business Travel, Daily Rate, Department, Distance From Home, Education, Education Field, Employee Count, Employee Number, Environment Satisfaction, Gender, Hourly Rate, Job Involvement, Job Level, Job Role, Job Satisfaction, Marital Status, Monthly Income, Monthly Rate, Numbers of Companies Worked, Over18, Overtime, Percent Salary Hike, Performance Rating, Relationship Satisfaction, Standard Hours, Stock Option Level, Total Working Years, Training Times Last Year, Work Life Balance, Years At Company, Years In Current Role, Years Since Last Promotion, Years With Current Manager. We can see why they are chosen as features that effects the probability of leaving one's job. The features are explainable. (Otherwise, we should pay extra attention to features making no sense.)

0.2 Primary processing

Notice that when we clean the data, we need combine the two files together. Then, we can see that features "user_id" and "Employee Number" differs from individual to individual; meanwhile "Employee Count", "Over18", "Standard Hours" only have one value, meaning that they are all lack of decisive information. After dropping these features (we actually save the "user_id" column for individual-identification), we encode the object-type data using One-Hot encoding. (Label-Encode results in scalars and One-Hot-Encode results in vectors.)

Next, we split them into two data sets: train_clean and test_clean. We are also interested in correlation Matrix for `train_clean`.If a feature has direct relationship with the target, then we see it as a main feature. But, from the graphs there is no feature that is significant, meaning maybe we need to consider feature engineering for classic machine learning algorithms.

0.3 Code

Code is updated in `Prediction of submission rate_Data Overview.ipynb`

Basic Machine Learning Algorithms

In this section, we will directly exploit some basic machine learning algorithms. Our purpose is to compare the results and analyze the possible better way.

Firstly, we try TPOT just to show an automatic way to decide the best models. If there is one model truly suitable for this data, then TPOT will be a programming-free way to have result. But in this case, we do not have a satisfying result, so we start from the most basic machine learning algorithms.

Therefore, the algorithms we use are LR (logistics regression), CART (Classification and Regression Trees), SVM (support vector machine) and KNN (k-nearest neighbour). They are all basic enough and suitable to explain the data structure. For the details of each algorithm, we have written in the series of articles of *Ten ways for MNIST data*.

Thirdly, we focus on the ensemble learning. Typically, we expect a better result comparing to the basic algorithms, since ensemble learning combines multiple results from basic machine learning algorithms. Among them, XGBoost always works well in such a situation. We will also explore other ensemble learning methods such as Adaboost, Random Forest and so on.

0.4 TPOT

TPOT stands for Tree-based Pipeline Optimization Tool. For an automated machine learning (autoML) tool in Python, . In our case, we try TPOT twice and get two different results: for the first one, we have best pipeline to be Linear SVM; for the second test, the best pipeline is Logistic Regression. (Actually, XGBoost sometimes

became the best pipeline.) For each time, the AUC score is about 0.6095, which far from satisfying result. Here, we conclude that all kinds of basic algorithms might have sightly differences but none of them is good enough. What is more, the randomly split of the data set can seriously effect the result, so we can use k-fold to increase the AUC score.

0.5 LR, CART, SVM, KNN

Theses basic algorithms classify the data using different methods.

LR chooses each column as a vector and uses regression to fit the targets. If LR gets a great output meaning that there is a function between features and targets, and the function satisfies the form of a logistic function.

CART is a classification of the decision tree method. With each features splitting into branches, CART finally define a specific area in the data space where the positive targets most likely to be concentrated.

Then, SVM is a totally diverse approach. Using kernel to twist the data space, support vector machine is able to find a hyperplane to separate the targets as accurate as possible. But there are several difficulties such as the choice of the kernel.

KNN represents the k-nearest neighbour, classifying the test individuals by counting and comparing the number of classification of individuals' k neighbours. This algorithm assumes that if most people in the similar situation chooses to stay their jobs, then this individual will stay too.

By analyzing these four algorithms, we might find downsides such as there might be no specific function between the features and targets, there might have too many untypical candidates that not lies in the area CART gives, there could be very hard to find the right kernel for SVM, and there are some doubts for people who chooses to stay in the similar situations.

Hereby, we consider some realistic situations. The incentive of a person who leaves one's job can be a single motivation or a combination of reasons. A strategy is to use multiple decision trees to find concentrated areas in data space and then sum them together, which remind us of the GBDT.

The next section doesn't finish yet!

0.6 Adaboost, XGBoost, GBDT, LightGBM, CatBoost, NGBoost

For the details of ensemble learning, please see the article *Treasures of Ensemble Learning*. Here, we can see the GBDT obviously gets the best reult so far, which fits our strategy. However, considering the random split and the one-order of all features, we then impose some techniques.

Data-based Learning Model

Feature Engineering

0.7 Guesses

Firstly, we use fact-based guesses, that is to choose most possible reasons and combines them as new features. (Ideally we want all the given features involve, but considering the limited storage and cost, we need to choose some of them to create new combinations. The criteria in this situation are correlation value.) There are various methods that we use to combine two given feature, here we use element-wise multiplication to create new features. [Notice:

- When we scale the given features
  Correlation value is not always reliable! For example, if the model already take the most of the information of related features, then how much more can the multiplication provide. Therefore, we need to think what to take very carefully.

- When we combine them
  Usually, element-wise multiplication is a popular way to combine two features. However, what really happens during this process is losing information. For instance, when the chosen features are 'Age' and 'Overtime_Yes', we expect to have a new vector that indicates a crossing factor with both age and overtime, but the truth is we got a partially information of 'Age' since 'Overtime_Yes' is a binary vector. For this reason, the obtain vector seems to target more specific group. This sometimes is valuable, but not in this case because GDBT already produce a result for concentrated candidates.

- Therefore
  Our model to predict those who might to leave their jobs contains three steps.

  First, using decision tree model

] using Gaussian graph to piece features together

0.8 2-order crossing features: FM

Final Touch

References

[1] Catherine Powell *Code for Advanced Uncertainty Quantification.*

[2] Catherine Powell *Advanced Uncertainty Quantification.*

[3] Catherine Powell *Advanced Uncertainty Quantification.*

A MATLAB Code

A.1 Code for Example Sheet 1 Question 8