Empathy prediction

Final project for CS 412 Machine learning at University of Illinois at Chicago

Mirko Mantovani
Computer Science department
University of Illinois at Chicago
Chicago, Illinois
mmanto2@uic.edu

ABSTRACT

This document is a report for the final project of CS 412 Machine Learning at University of Illinois at Chicago. It consists in a binary classification prediction problem, where, given the Young People Survey dataset containing about 1000 examples (people) and 150 features about their personal tastes in music, movies, and personality, we have to train and build a model that is able to predict if a person is Very Empathetic (if Empathy is 4 or 5) or not very Empathetic (Empathy level is 1, 2 or 3), to help select suitable volunteers to help people with Alzheimer.

The repository containing the python notebook can be accessed through GitHub at:

mirkomantovani/Machine-Learning-empathy-prediction

1 DATA EXPLORATION - PREPROCESSING

1.1 Missing values

The first thing I noticed by looking at some statistics about the dataset, is that there were missing values in almost every feature. A very important observation was that there were also missing values in the Class (Empathy). I dropped the examples with missing Empathy because imputing them in any way would not have made any sense since it's what we are predicting.

Many other features had from about 0.1% to 2% of missing values, randomly distributed in the dataset. I decided to impute all these values using the mode.

1.2 One Hot Encoding

Since many algorithms cannot deal with categorical values, and the dataset presented 11 categorical features, I decided to encode them, based on the type of features and their values, I divided these categorical features in three groups: ordinal, binary, OHE.

Ordinal attributes are the ones whose values can be ordered by using some kind of logic.
Binary features are the ones whose value can be encoded to be either true or false.
I used OHE on the remaining categorical values because I thought that they could not really be ordinal, and some kind of information could have been lost in considering them as ordinal.

2 MODELLING THE PROBLEM

2.1 Reasoning about the problem

Since it is a classification problem, but the initial class is encoded with more than 2 values and it is ordinal, I started by thinking that this fact could be exploited, and that a regression model could be used to exploit this fact, predicting Empathy as a continuous value and then converting the predictions by splitting in binary based on a 3.5 pivot could be an interesting approach.

2.2 Baseline and simple models

The expected value of a majority class classifier is around 66% which highlights the unbalanced number of examples towards "Very empathetic".

I started by trying how some simple models and different approaches were performing.

I tried Logistic Regression multiclass, SVM OVO multiclass, SVM linear binary classifier.
They were performing pretty bad with respect to the baseline, and the accuracy was never higher than 64%. Based on my tests I decided to abandon the multiclass and regression approaches and concentrate on binary classification. Given the high number of features and small number of examples, I thought that algorithms like random forest, based on bagging and with an implicit feature selection, could be a very good solution. In fact, random forest without any tuning brought up the accuracy to 70%.

2.3 Error fixing

Inspired by the idea of boosting, I decided to see where random forests were performing bad. To do this I performed a Leave-one-out cross-validation with random forest on the entire dataset, took all the misclassified examples and compared their statistics with the ones of the entire dataset. I discovered that the major difference was in Empathy, the average was 2.98 while for the entire dataset it was 3.85, moreover, the 75% percentile is 3 for the misclassified examples. This means that at least 75% of these examples will fall in the category of Not very empathetic, therefore, the problem in the prediction is mainly the inability of understanding when to predict 0.

To fix this I thought about creating a custom ensemble of random forest and another model that uses a different approach and could spot Not very empathetic people in a different way. I tried to use a boosting algorithm as second model to integrate, in particular, I chose eXtreme Gradient Boosting.

2.4 Train/dev/set split and hyperparameters tuning

I split the dataset in 80% train and 20% test, as for the hyperparameters tuning, I used cross-validation on the training data.
I tuned n_estimators, max_depth and max_features for Random Forest using ValidationCurve with CV. For XGBoost instead, I selected a subsample of most important features with SelectKBest (since Boosting does not work well with a lot of features, especially if they are correlated), I did some and decided to retain only 20 features, I then tuned gamma, learning_rate (eta), max_depth, reg_alpha using RandomizedSearchCV.

2.5 Final model and evaluation

The idea for the final model ensemble is the following: since many times the difficult part for the models is to spot when an example is 0 (there are a lot of false positives), I will perform a logical AND between the prediction of the two different classifiers (Random Forest and XGBoost), in this way whenever one of the two predicts 0, it will be 0, whereas they need to agree on the 1 in order to predict 1.
Training the final model on the entire train (80%) and testing on the remaining 20% test set, the accuracy for the final model was around 79%, while it was 77% for XGBoost alone and 74% for Random forest alone.
The final ensemble is thus bringing improvement in accuracy.

3 SOFTWARE CREDIT AND CONCLUSIONS

The libraries used are: pandas, sklearn, numpy, xgboost. I did not share any results, nor I have talked about models and methods with anybody. I used small parts of code written by myself for another project, which is publicly available as a repository on my GitHub account.