Machine Learning for Automated Essay Grading

Author:
Anand Sakhare
MISM-BIDA Graduate Student | Carnegie Mellon University
Mobile: (412)708-7836
Email: asakhare@andrew.cmu.edu

Dataset:
The input data is a cleaned-up version of dataset from a kaggle competition.
Competition Name - The Hewlett Foundation: Automated Essay Scoring
Source: https://www.kaggle.com/c/asap-aes

Goal:
The goal of this project is to create a machine learning model to learn the graders behavior in grading essays written in English. The model can be used to automate the gradings of English essays written by students.

Flow:
The dataset has grades given by two different graders independently and a resolved score between the graders. The task is to learn the trends that graders follow and predict the resolved score based on the essays. To achieve this task, I have built a classification model which will take a processed version of the essays with features extracted from the essay and then predict the resolved score. Below is a stepwise flow of the code I wrote for building this model.

1. **Data Loading:**
   Read all the data from the csv input file. The input file has flowing columns:
   a. Essay_id – Unique id of the essay
   b. Essay_set – A set to which the essay belongs
   c. Essay – the actual essay written in English
   d. rater1_domain1 – Score given by rater number 1 in domain 1
   e. rater2_domain1 – Score given by rater number 2 in domain 1
   f. domain1_score – Resolved score between the two graders
2. **Data Preprocessing:**
   Below are the steps followed to do some data cleaning, preprocessing and feature extraction.
   a. Make all the text essays lowercase to better identify and similar words for the purpose of removing stop words and lemmatizing.
   b. Removing stop words can actually benefit a lot in nlp. I combined the stop words offered by two packages in Python (named- stop_words, nltk) and then removed these stop words from the essays.
   c. The next step was to lemmatize the words using WordNetLemmatizer in nltk. By definition, Lemmatization (or lemmatization) in linguistics, is the process of grouping together the different inflected forms of a word so they can be analyzed as a single item. (Source:http://textminingonline.com/dive-into-nltk-part-iv-stemming-and-lemmatization)
   The lemmatization is important to convert the words to their stem, base or root form. (e.g. convert 'saying' or 'says' or 'said' to 'say') This step is important since in the later part we'll create a sparse matrix using the tfidf (term frequency–inverse document frequency) statistic.

d. Next step is to remove the punctuation since punctuation can create an unnecessary bias.

Based on these steps we now have a cleaned-up version of the input essays.

3. **Feature selection:**

   Below is a list of features that I chose to select for the purpose of getting meaningful insights on the essays. Broadly the feature can be divided into three categories as give below:

   a. Geometrical features:
      i. Length of the essay
      ii. Number of digits
   b. Parts-of-speech features: I have calculated the number of various parts of speech words used in the essay. There are 32 parts of speech (e.g. verb, noun, adjective, etc.). Each essay will have 32 new features which represent the count of the words belonging to the specific part of speech for that essay. (package used: nltk.word_tokenize)
   c. Text features: using tfidf Vectorizer (term frequency–inverse document frequency) from scikit learn package, I have computed a sparse matrix which represents the weighted words frequencies. In this matrix each row represents the essay and each column represents different words used in those essays. Tfidf down weights the frequently used words across documents (essays).

4. **Reducing the dimensionality to decrease time complexity:**

   Due to the vast number of features given by tfidf sparse matrix, the time complexity in building the model is very high. Using TSNE from sklearn would be a better option than using PCA (Principal Component Analysis) since PCA assumes linearity. But due to the computational limitations I am reducing the dimensionality using PCA since the processing time for TSNE is a lot more than PCA. The original feature space for the tfidf sparse matrix includes 43000 + features, but to make it simple I'm reducing it down to 4 features that would represent the variability in the language.

5. **Model Building:**

   In this step I am holding out 20% of the data for testing. Meaning the model will train on the rest of the 80% of the data and will be tested for accuracy using the 20% of the data. I have tried a variety of classification models, such as decision tree classification, SVM with rbf kernel and Random Forest. Upon investigating these models against each other, as expected the performance of Random Forest classifier was better and hence I decided to further fine tune the model. I created a grid space and using grid search I've trained the random forest classifier. (The best set of features found after the cross validation was 'criterion': 'entropy', 'max_depth': 80, 'n_estimators': 150). For further expansion of the model tuning, I can extend the grid space to cover a variety of hyperparameters with greater depths, with a trade-off with time complexity of the model.

6. **Model evaluation:**

   I have tested the model on the held-out testing data, and the model gives an accuracy score of 54%. Also, the root mean squared error in the predicted and the actual scores for the testing data came out to be around 4, which means that on an average there a chance that the model may make a mistake of up to 4 (+2 or -2) scoring points in the prediction. Since, this is multiclass model a simple ROC (Receiver operating characteristic) curve may not give insights on the model performance. Also, the most important feature in the prediction was the length of the essay.

7. **Performance Improvement:**

   Although the model gives a good performance, there are multiple ways to improve it and make the model better. Adding more geometrical features (e.g. number of punctuations, number of capitalization), performing TSNE instead of PCA for dimensionality reduction would be the measures that can be taken towards performance improvement.