Machine Learning Engineer Nanodegree

Capstone Proposal

Stefan Dittforth
February 16th, 2018

Proposal

Domain Background

For this capstone project the field of natural language processing (NLP) and analysis of written text in particular has been chosen. With the rise of blogging and self-publishing written text became ubiquitous. NLP offers promising approaches to text manipulation and processing that makes information in natural language accessible to computers. NLP has developed out of the field of linguistics with the rise of computers about 50 years ago. Natural language processing is an interesting field as it defies mathematical solution approaches. Early attempts based on set of rules proved not scalable to the complexity (or more accurately "messiness") of human language (see also: History of Natural Language Processing).

Personally, I've been fascinated to see how natural language processing has made significant leaps since the 1990's. I remember well when IBM worked on their speech recognition system ViaVoice which was often difficult to handle and required training. This pales in comparison to the speech recognition systems available today. The breakthrough came with the increase of computational power, availability of larger datasets and the success of statistical models and machine learning algorithms over rule based systems.

Problem Statement

In this project we will develop a system that is able to learn the characteristics of articles written by various authors. After a learning phase the system predicts which author wrote a given article. The system will not have seen the given article before. The likelihood of an article being written by a particular author will be expressed as a probability value between 0 (unlikely) and 1 (very likely).

Such a system can be of help, for example, for historians that try to establish authorship of text fragments. Another use case might be the analysis of how similar or dissimilar a text from one author is to other authors.

Datasets and Inputs

In order to allow the system to learn the writing characteristics of different authors we require a dataset that provides a large number of articles for individual authors. There are rich, publicly available datasets for NLP research. A list, for example, can be found here. However, for this project we will attempt to build our own dataset. We will develop a web crawler that will collect articles from publishing platforms such as Medium or LinkedIn. The articles on these platforms seem to be reasonably long (at least several hundred words). There are enough authors that have published several hundreds articles. With this, it appears feasible to acquire a large enough data set to learn patterns in the writing characteristics to distinguish between individual authors.

This approach has been chosen as an opportunity to develop practical experience not only in machine learning but also around data acquisition. In data science and machine learning the acquisition and preparation of high quality data is often the bigger challenge than the actual development of the machine learning system itself. In "Datasets Over Algorithms" author Alexander Wissner-Gross points out that *"the average elapsed time between key [machine learning] algorithm proposals and corresponding advances was about eighteen years, whereas the average elapsed time between key dataset availabilities and corresponding advances was less than three years, or about six times faster, suggesting that datasets might have been limiting factors in the advances."*.

Solution Statement

The individual articles will be stored in files of one file per article. The NLP system will then be able to load individual files for further processing. We will extract article metrics such as total number of words, number of sentences, minimum, maximum, median and average number of words per sentence. In a first attempt we will use these article metrics to develop an authorship classifier using various classifier algorithms. We will then use the bag-of-word approach to establish article characteristics based on word frequencies. We will look into combining the article metrics with the bag-of-words approach using, for example, ensemble learning to see if prediction accuracy can be improved. We will further build a neural network and experiment with various parameters to improve prediction accuracy.

Benchmark Model

For our authorship classifier the benchmark model will be a random selection of an author for a given article. This will result in an accuracy of $1/n$, where n represents the number of authors in the dataset. Our authorship classifier needs to do better then this random selection of the author.

Evaluation Metrics

The benchmark model and versions of the solution model will be evaluated against classifier metrics such as precision, recall and the F1-score. Furthermore, we will look at multi-class confusion matrices for each model (an approach suggested in this Coursera video about Multi-Class Evaluation). These confusion matrices will give insight into the frequencies of different errors.

Project Design

The project will be split into two phases. In the first phase we will develop and run a script that will scrape articles from various authors from publishing platforms such as Medium or LinkedIn. Each article will be stored in a file. This requires research to identifying enough authors that have published a large enough number of articles. Ideally, we are looking to collect several hundred articles per author. That should give us enough training and test data to develop a solution model. The article scrape script needs to ensure that only the text data is extracted. For this project we are not interested in other information such as images, hyperlinks, formatting, etc.

In the second phase we will develop the solution model that is capable to predict the author for a given article. We will start with a review of the data captured. We will check what type of data cleansing actions might be required. Although the article scrape script will only extract text information we will check whether the data set still contains html code that needs to be removed. This should be fairly straight forward by searching for the occurrence of "<" or ">" characters in the article files. Hopefully, only a small number of files will be found. Some of these can be inspected visually to check for left over html code or tags.

Functions will be developed that remove these fragments. Beside html code fragments the inspection of sample files will show what further cleansing functions might be required.

The data set will then be enriched with the article metrics such as total number of words, number of sentences, minimum, maximum, median and average number of words per sentence. The dataset will be split into training and cross validation datasets. These will be used to measure the performance of our solution models on training data and unseen data.

First prediction models will be developed that utilizes the article metrics. For this we will use algorithms such as decision-tree, support vector machines (SVMs) and algorithms we might come across during further research. Next we will look into utilizing text analysis algorithms such as bag-of-words, bi and n-grams. Furthermore, we want to look into combining various prediction model by utilizing ensemble learning (e.g. combining prediction based on article metrics with the bag-of-words approach).

We will document our approaches and results in the form of a Jupyter Notebook. This allows to discuss different solution models in separate sections. For each model we will generate and discuss the evaluation metrics described above.