MACHINE LEARNING ENGINEER NANODEGREE

CAPSTONE PROPOSAL

Abhishek Verma
March 1st 2017
www.abhishekverma.me
github.com/hell-sing

Domain Background

The aim of this project is to explore the results of applying machine learning techniques to Message spam detection. Mobile phone spam also known as (unsolicited messages, especially advertising), directed at the text messaging or other communications services of mobile phones or smartphones. Fighting SMS spam is complicated by several factors (compared to Internet email), including the lower rate of SMS spam, which has allowed many users and service providers to ignore the issue, and the limited availability of mobile phone spam-filtering software. The dataset for this project originates from the UCI Machine Learning Repository.

Problem Statement

In the project, we would try to analysis different methods to identify spam messages. We will use the different approach, based on word count and term-frequency inverse document-frequency (tf-idf) transform to classify the messages using the SMS Spam Collection v.1 dataset originates from the UCI Machine Learning Repository.

Datasets and Inputs

The dataset used for this project is SMS Spam Collection v.1 dataset originates from the UCI Machine Learning Repository. This dataset has been collected from free or free for research sources at the Internet. The collection is composed of just one text file, where each line has the correct class followed by the raw message. This dataset is tab-separated values (TSV) file. There are total 5572 entries in the dataset and has two column “Class” and “Text” where each row represent different message and Class contain two unique categories ham and spam. Dataset does not require any kind of cleaning, wrangling and there is no null value in any column

Solution Statement

We are given labelled training data, so this makes it a supervised machine learning problem. For every message, it can be predicted whether it is ham or spam. The accuracy is quantifiable in terms of the f1_score. These performance scores can be compared against the public leader board scores available in the Kaggle website. A well-documented code with dataset will help anyone to replicate the work anywhere on any other machine. To begin with I would like to experiment with techniques which we are going to us are based on word count and term-frequency inverse document-frequency (tf-idf) transform. After which I would like to test the approach using many different algorithms like Naive Bayes, Decision Tree, AdaBoost, K-Nearest Neighbours and Random Forest and test the accuracy using f1_score.

Benchmark Model

Benchmark models are available in Kaggle discussion forums which uses different Machine Learning algorithms. The available public and private leader board score in the Kaggle competition can be used to benchmark the performance of my algorithm. Also, it is possible to explore how the proposed model perform compared to existing models. The result shows that Naïve Bayes work better on the dataset with an accuracy_score of 0.90.

Evaluation Matrix

Accuracy is the first metric to be checked when the algorithms are evaluated, is the sum of true positives and the true negative outputs divided by the data size. Accuracy means how closer you are to the true value, whereas precision means that your data points are not widely spread. The Scikit-learn library provides a convenience report when working on classification problems to give you a quick idea of the accuracy of a model using a number of matrices, one of them is F1_score which work with all the model.

$$F1 = 2\frac{P \times R}{P + R}$$

Project Design

The theoretical workflow of the project would look like:

1. Download and pre-process the SMS Spam Collection v.1 dataset.
2. Test and find best approach (word count or tf-idf vectorizer) to classify the messages.
3. Selection of approach and splitting the dataset into training and testing data.
4. Initialize various classifier and train it using training data.
5. Evaluate the classifiers and finding best the model for a dataset using testing data.