Bishop’s University
Division of Natural Sciences & Mathematics

SMS Spam Classification

A project report submitted for
CS504 Programming Languages for Data Analysis

by

Tianye Zhao (002254003)

Professor

Rami Yared

March 2019

Table of Contents

1 Introduction

The growth of mobile phone users has led to a dramatic increasing of SMS spam messages. This project would concentrate mainly on data analysis. It would implement a few methods regarding natural language processing to manipulate the SMS Spam Collection dataset. It implements some methods as following:

1. Read in and create data frame in messy text data;
2. Clean and tokenize data acquired in step 1;
3. Lemmatize, stem, and vectorize data acquired in step 2;
4. Build machine learning classifiers with data prepared in step 3;
5. Evaluate and test variations of machine learning models, such as random forest and gradient boosting.

2 Tools, Libraries & Pipeline

Main tools for this project are:

- Python 3.7.2
- Jupyter Notebook
- Anaconda (combine both above)

Below are some libraries needed for this project.

- nltk: natural language toolkit, guide book is here;
- pandas: manipulate text data set in data frame;
- re: regular expression operations;
- string: help remove punctuation;
- sklearn: extract feature, build classifiers, and train model;
- numpy: assist to analyze data;
- matplotlib: draw graphs to illustrate features.

The project pipeline lists as following:

1. Raw text – model can't distinguish words
2. Tokenize – tell the model what we are interested in

3. Clean text – remove stop words/punctuation, stemming, etc.
4. Vectorize – convert to numeric form
5. Machine learning algorithm – fit/train model
6. Spam filter – system to filter SMS

3 NLP Basics

Please refer to attached source code file – "*3 NLP Basics.ipynb*" for detailed text data analysis in preprocessing stage.

3.1 NLP & NLTK

NLP is a field concerned with the ability of a computer to understand, analyze, manipulate, and potentially generate human language. NLP in real life applications are:

- Spam Filter: email filter – inbox or spam;
- Auto-Complete: google searching;
- Auto-Correct: iPhone correct a misspelling.

NLP encompasses topics like:

- Sentiment analysis
- Topic modeling
- Text classification
- Sentence segmentation or part-of-speech tagging

Natural Language Toolkit (NLTK) is a suite of open-source tools created to make NLP processes in Python easier to build.

3.2 Read in and explore the text data

Most text data lacks the formal structure of numeric data. Unstructured data (such as emails, PDF files, and social media post) may have features like:

- Binary data
- No delimiters
- No indication of rows

There are 53 text datasets available in UCI machine learning repository website. This project may work on SMS Spam Collection Data Set which contains 5574 instances of SMS labeled messages. It's easy to manipulate small datasets first. Moreover, each row has a distinct text message and a distinct label as either spam or ham, so it's not terribly unstructured.

3.3 Regular expressions

Regular expressions are text string for describing a search pattern. They can be used for:

- Identify whitespace between words/tokens
- Identify/create delimiters or end-of-line escape characters
- Remove punctuation or numbers from text
- Clean HTML tags from text
- Identify textual patterns

Use cases are:

- Confirm passwords meet criteria
- Search URL for some substring
- Search for files on computer
- Document scraping

3.4 Clean data

Here we preprocess text data through all in one function for first three steps.

3.5 Stemming & lemmatizing

Stemming is the process of reducing inflected (or sometimes derived) words to their word stem or root. It crudely chops off the end of the word to leave only the base. Here are some examples:

- Stemming/stemmed –> Stem
- Electricity/electrical –> Electr
- Berries/berry –> Berri
- Connection/connected/connective –> Connect
- Meanness/meaning –> Mean

Reasons for stemming:

- Reduces the corpus of words the model is exposed to
- Explicitly correlates words with similar meanings

There are a few famous stemmers:

- Porter Stemmer
- Snowball Stemmer
- Lancaster Stemmer
- Regex-Based Stemmer

Lemmatizing is the process of grouping together the inflected forms of a word so they can be analyzed as a single term, identified by the word’s lemma. It uses vocabulary analysis of words aiming to remove inflectional endings to return the dictionary form of a word. The goal of both methods is to condense derived words into their base forms. Differences between stemming and lemmatizing are:

- Stemming is typically faster as it simply chops off the end of a word using heuristics, without any understanding of the context in which a word is used.
- Lemmatizing is typically more accurate as it uses more informed analysis to create groups of words with similar meaning based on the context around the word.

4 Vectorizing

Please refer to attached source code file – “*4 Vectorizing.ipynb*” for detailed text data analysis in preprocessing stage.

Vectorizing is the process of encoding text as integers to create feature vectors. Feature vector is an n-dimensional vector of numerical features that represent some object. When looking at a word, computers only see a string of characters. Raw text needs to be converted to numbers so that computers and the algorithms used for machine learning can understand. There are a few well-known types:

- Count vectorization
- N-grams
- Term frequency – inverse document frequency (TF-IDF)

5 Feature Engineering

Please refer to attached source code file – "*5 Feature Engineering.ipynb*" for detailed text data analysis in preprocessing stage.

From above we have completed:

- Read in messy raw text;
- Clean data by removing punctuation, removing numbers, removing stop words, stemming, and lemmatizing;
- Vectorize data to prepare it for a model build.

Feature Engineering is creating new features or transforming existing features to get the most out of data. Here are a few ways to create new features for our model:

- Length of text field
- Percentage of characters that are punctuation in the text
- Percentage of characters that are capitalized

Given new features, we maybe need to apply some sort of transformation to data to make it more well-behaved. Transformation is the process that alters each data point in a certain column in a systematic way. Here are a few broad popular types of transformations:

- Power transformations (square, square root, etc.): log-transformation for skewed data set with long right tail which is outliers
- Standardize data: transform all data to be on the same scale

6 Machine Learning Classifiers

Please refer to attached source code file – "*6 Machine Learning Classifiers.ipynb*" for detailed text data analysis for building machine learning classifiers.

6.1 Machine learning introduction

According to example from NVIDIA, machine learning is the practice of using algorithms to parse data, learn from it, and then make a determination or prediction about something in the world. There two broad types of machine learning:

- Supervised Learning: Infer a function from labeled training data to make predictions on unseen data, for example, predict whether any given email is spam based on known information about the email;
- Unsupervised Learning: Derive structure from data where the effect of any of the variables is unknown, for example, based on the content of an email, group similar emails together in distinct folders.

6.2 Cross-validation & evaluation metrics

Holdout test set is sample of data not used in fitting a model for purpose of evaluating the model's ability to generalize unseen data. *K*-fold cross-validation divides full data set into *k*-subsets and repeat holdout method *k* times. Each time, one of the *k*-subsets is used as the test set and the other *k*-1 subsets are put together to be used to train the model.

Evaluation metrics are:

$$Accuracy = \frac{\#\ predicted\ correctly}{total\ \#\ of\ observations}$$

$$Precision = \frac{\#\ predicted\ as\ spam\ that\ are\ actually\ spam}{total\ \#\ predicted\ as\ spam}$$

$$= \frac{True\ Positives}{True\ Positives + False\ Positives}$$

$$= \frac{True\ Positives}{Selected\ Elements}$$

$$Recall = \frac{\#\ predicted\ as\ spam\ that\ are\ actually\ spam}{total\ \#\ that\ are\ actually\ spam}$$

$$= \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

$$= \frac{True\ Positives}{Relevant\ Elements}$$

6.3 Random forest model

Ensemble method is the technique that creates multiple models and then combines them as metamodel for prediction to produce better results than any of the single models individually. Random forest is an ensemble learning method that constructs a collection of decision trees and then aggregates the predictions of each tree to determine the final prediction. The benefits of ensemble methods are:

- Can be used for classification or regression
- Easily handles outliers, missing values, etc.
- Accepts various types of inputs (continues, ordinal, etc.)
- Less likely to overfit
- Outputs feature importance

6.4 Gradient-boosting model

Gradient boosting is an ensemble learning method that takes an iterative approach to combining weak learners to create a strong learner by focusing on mistakes of prior iterations. Comparing with random forest method, both are ensemble methods and decision tree based. The differences between these two methods are following:

Random Forest	**Gradient Boosting**
Bagging	Boosting
Training done in parallel	Training done iteratively
Unweighted voting for final prediction	Weighted voting for final prediction
Easier to tune, harder to overfit	Harder to tune, easier to overfit

Here are trade-offs of gradient boosting:

Pros	**Cons**
Extremely powerful	Longer to train (can’t parallelize)
Accepts various types of inputs	More likely to overfit
Can be used for classification	More difficult to properly tune
Outputs feature importance	

6.5 Model selection

Machine learning pipeline is as following:

1. Read in raw text.
2. Clean text and tokenize.
3. Feature engineering
4. Fit simple model.
5. Tune hyperparameters and evaluate with GridSearchCV.
6. Final model selection.

Vectorizers should be fit on the training set and only be used to transform the test set. The process for this:

1. Split the data into training and test set.
2. Train vectorizers on training set and use that to transform test set.
3. Fit best random forest model and the best gradient boosting model on training set and predict on test set.
4. Thoroughly evaluate results of these two models to select best model.

7 Conclusion & Further Work

In this project, it consists of read-in text data without knowing how many columns there are, what separates the columns, or even what format it is in. Then, it shows clean up that data using regular expressions and tools from various packages to remove punctuation, numbers, tokenize, remove stop words, and stem and lemmatize data. Then it implements a few different methods to vectorize data to get it into a format that a machine learning model can understand and train on. Then it creates and transforms features. Lastly, it builds and evaluates a few different types of models using the text and created features.

Train other models such as neural network with tensorflow or keras, then make prediction on new incoming message. Jupyter notebook is perfectly working on data analysis to show interactive result, however, it would be slow comparing other IDE like PyCharm. Other IDE could be used for training stage or continuous development instead. Generate wordcloud for most frequent spam words using wordcloud library. Create GUI for interactive text data analysis using tkinter.