Snizhana Kurylyuk

Abstract

The goal of this project was to use natural language processing tools to build a classification model to predict whether a text message was spam or not. This would allow individuals to screen their text messages and avoid potential spam. I worked with data provided by kaggle. I preprocessed the data, conducted initial data visualization. Then I built a TFIDF vectorization model and performed topic modeling using NMF to create 10 topics. Then I built 10 initial classification models to find the best predictive model. I explored those further by hyper tuning the top three models. I used the best one to create a classification app on streamlit.

Design

Spam text messages are getting out of hand. It is a problem almost every individual has experienced. My project focused on using natural language processing to determine which text messages are spam and which ones are real.

The goal of this project was to build use unsupervised NLP resources to preprocess and analyze my dataset. Then I used it to create a classification algorithm to determine if text messages were spam or not.

Data

This data comes from the SMS Spam Collection v.1. It is a public set of SMS labeled messages that have been collected for mobile phone spam research. It has one collection composed by 5,574 English, real and non-encoded messages, tagged according being legitimate (ham) or spam. I

Algorithms

- *Text Pre-processing*: I used lemmatization and stopword removal to filter words. I then used regex expressions to remove punctuation, links, numbers and special characters

- *Vectorize Text:* used a tf-idf (Term Frequency * Inverse Document Frequency) to vectorize the text from each clue. In other words, I turned the raw text into a matrix whose entries are the numerical tf-idf features of each word in the text.
- *Dimensionality Reduction:* I then used Non-Negative Matrix Factorization (NMF) to create clusters of words, where each cluster can be thought of as a *meta-category* or latent topic, which is one of the goals of this analysis.
- *Scatter Text:* for finding distinguishing terms and presenting them in an interactive scatter plot with non-overlapping term labels.
- *Classification Algorithms*: Built a series of 10 initial classification algorithms and hypertuned the parameters of the best two.

Tools

- Numpy & Pandas for data processing
- Scattertext for finding distinguishing terms
- Streamlit for Classification app deployment
- Matplotlib, WordCloud for visualization
- Scikit-learn for machine learning
- NLTK for natural language processing

Communication

The project used powerpoint for the presentation and the python visualization libraries for the visuals.

WordCloud was used to generate word clouds

Streamlit was used to deploy a prototype application for the classifier.