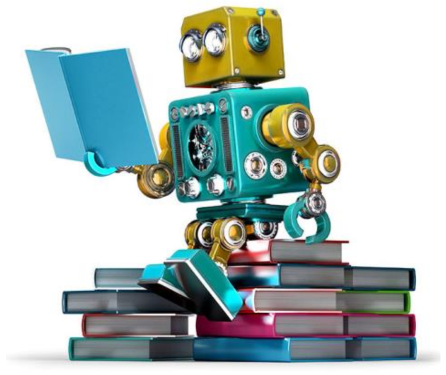
Machine learning
Final Project - Virtual Doctor



Ilakya Selvarajan

PhD student (DPMM)

University of Eastern Finland, Kuopio campus

Contents

Project 2. Virtual Doctor

Motivation and Goal

1. **T**he goal of the project is to build a tool to predict online patients' illness based on a list of their symptoms.
2. Machine learning provides computers the ability to iteratively learn from data and allows to find hidden insights without being explicitly programmed where to look [1]. Using this subfield of science, programs can be scripted to train computers to predict human disease from its symptoms.
3. There are lot of applications for the virtual doctor. One of the application is discovering medical decision support systems designed to support clinicians in their diagnosis.

Submitted files

1. Data_extraction.pl – PERL program used to scrap data. The script takes in a list of URLS as input from a file, and outputs text files with disease, summary and symptoms.
2. Virtualdoctor.py – Main python program for disease prediction. The program needs the following to run [2].
   a. numpy==1.9.2
   b. scipy
   c. scikit-learn
   d. nltk
   e. pandas==0.16.0
   f. beautifulsoup4
3. Word2VecUtility.py – Python subprogram for data cleaning and processing.

Hardware specification of test PC

Dual core CPU, 4GB RAM (Ubuntu 64bit virtual machine)

Approach

The problem in hand is straight forward. When user inputs the symptoms of interest, the program should predict the disease. The problem is text based since the input is set of strings (symptoms) and the output is also a string (disease name). No other parameters are considered here. Using the trained dataset and using the input as test-set, the program predicts the disease for the user (supervised learning).

The most appropriate technique for text based problems is Natural language processing (NLP). The **bag-of-words model** is a simplifying representation used in NLP. In this model, a text (i.e. symptoms) are represented as the bag of its words. The bag of words is then used for classification where the occurrence of each symptom is used as a feature for training a classifier.

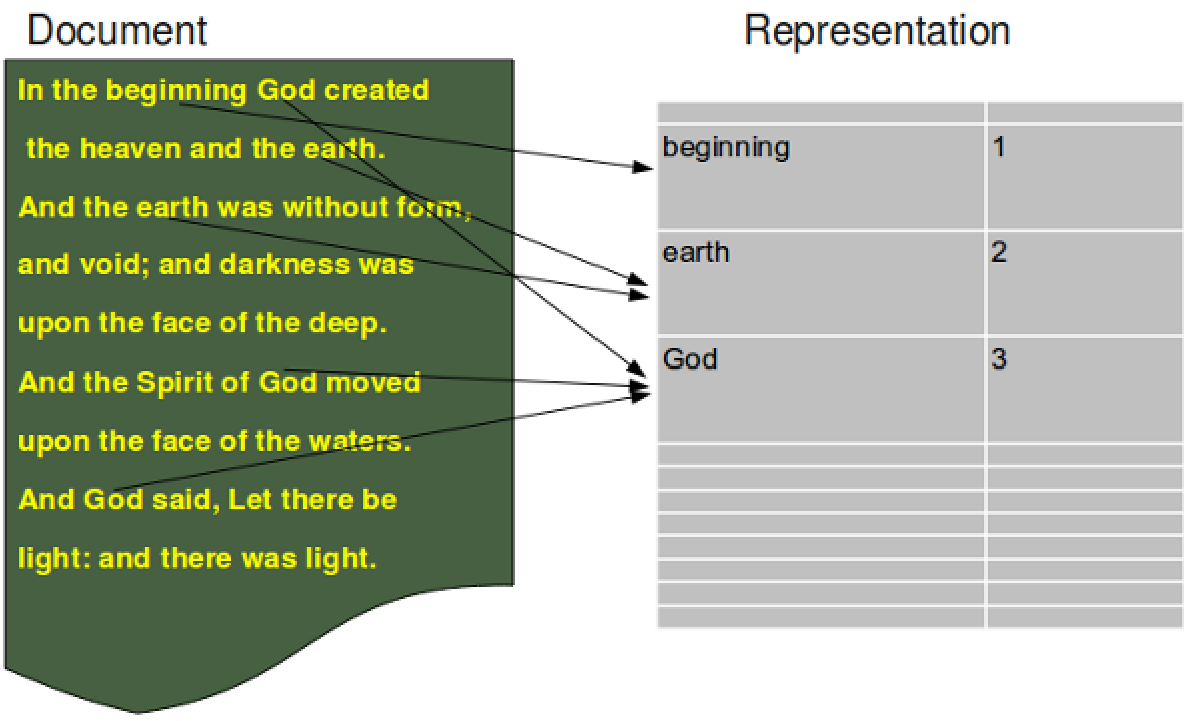

Document
In the beginning God created
the heaven and the earth.
And the earth was without form,
and void; and darkness was
upon the face of the deep.
And the Spirit of God moved
upon the face of the waters.
And God said, Let there be
light: and there was light.
Representation
beginning
1
earth
2
God
3

Figure 1. Document representation [3]

1. The dataset (disease, summary and its symptoms) was extracted from MedlinePlus website using PERL. The URLs were taken in as a list and each index is recursively read using LWP::Simple [4]. The source of the webpage was dumped into a buffer file when the webpage had symptom. Else, the disease was discarded. A total of 1359 diseases were included for the training. Once the downloading was complete, match expression was used to extract the lines containing the disease and its corresponding summary and symptoms. Finally, a tab limited text file was generated.
2. Train-set: Will have the disease name, summary and its symptoms (Data will be in HTML format with lots of tags).
3. Test-set: Once the program starts to run, it asks the user for the input. Note! Each symptom of the disease should be given with space.
4. Bag of words: The program will,
   a. Extract the summary and symptom columns for data cleaning and text pre-processing (using python beautifulsoup package). Finally, all the frequently occurring words such as "a", "and", "is", and "the" were removed using Python packages NLTK that come with stop word lists built in.
   b. After text processing, the words are converted into numeric representation for machine learning. The Bag of Words model learns a vocabulary from all the documents, then models each document by counting the number of times each word appears. scikit-learn module was used to create bag-of-words features.
   c. Random Forest uses many tree-based classifiers to make predictions. Here, I have set the number of trees to 300.

Discussion.

1. **Data extraction:** Extracting the symptoms seemed very tricky since not all webpages had symptom paragraph. The section-id in the webpage for symptoms seemed to be different for each disease.
2. **Selecting the classifier:** Since the training set doesn't deal with numbers, I spent a lot of time to figure out a prediction method that uses string.
3. **Text processing:** The symptom column when downloaded was a mess. Prediction was impossible without data cleaning. I tried to extract the symptoms terms by mapping to it's html page (Some of the symptoms were hyperlinks). But, the prediction accuracy was very bad.
4. **Windows numpy installation**: After 3 days of trying to figure out how to install numpy, I went to use virtual box Ubuntu to run the program.
5. **System power:** The virtual box had 4GB RAM. When I increased the tree size more than 300, the program was killed due to memory problem.
6. **Randomness:** One main problem with the randomforest (as the name suggests) was the random results. The result kept changing. Random_state was set to reduce randomness.
7. **Prediction power:** Since the number of disease included are quite high, the prediction model is not very accurate. For the query testing for 20, it gave me correct result for 14.

Acknowledgements

The dataset was extracted from MedlinePlus website [5].

The bag of words kaggle example tutorial was mainly used for this assignment (https://github.com/wendykan/DeepLearningMovies)

References

1. http://www.sas.com/en_id/insights/analytics/machine-learning.html
2. https://github.com/wendykan/DeepLearningMovies/blob/master/README.md
3. http://www.python-course.eu/
4. http://search.cpan.org/~ether/libwww-perl-6.15/lib/LWP/Simple.pm
5. https://medlineplus.gov/encyclopedia.html