CSCI E-89 Final Project:

Classification of Eligibility Criteria for Breast Camcer & Alzheimer’s Using RNNs

URL: <https://github.com/smathurdc/ClinicalTrials>

**Problem Statement:**

Selecting patients for clinical trials is expensive and hence eligibility criteria should be carefully drafted so that trials meet their recruitment targets. Using eligibility criteria for Triple Negative Breast Cancer and Alzhemer’s, I checked if they can be differentiated using NLP and deep learning, so that new eligibility criteria can be compared against these classes. This will help in discarding criteria that may be may be too generic/restrictive and retaining criteria that are specific to an indication.

**Overview of Technology:**

Dataset:

Consisted of 2 datasets:

1. Randomly selected ~11000 trials irrespective of the indication
2. TrainingSet: Eligibility criteria related to triple -ve Breast Cancer and Alzheimer’s
3. TestSet: Eligibility criteria related to triple -ve Breast Cancer and Alzheimer’s

Dataset Size = 24 MB

URL (in order):

1. <https://github.com/smathurdc/ClinicalTrials/blob/master/forWordEmbedding>
2. <https://github.com/smathurdc/ClinicalTrials/blob/master/TrainingData.txt>
3. <https://github.com/smathurdc/ClinicalTrials/blob/master/TestData.txt>

Data was obtained by using an API to AACT - <https://aact.ctti-clinicaltrials.org/>

**High Level Overview of Steps:**

1. Use R-API to extract data from AACT as python API is unavailable
2. Install NLTK, Keras, gensim packages
3. Performed NLP to clean up text and created a word embedding using skip gram model
4. Trained an LSTM, RNN, Bi-Directional LSTM using training data
5. Computed accuracy on test data

**Hardware:** Windows 2.4 GHz Intel with 16 MB RAM and Google Colab run on a GPU

**Software:** RStudio, GoogleColab: Keras, Gensim, String, NLTK

**Lessons Learned:**

It is important to clean up text before creating a word embedding, whereas stemming or lemmatization may make it worse. The LSTM and BiDirectional LSTM are able to classify inclusion criteria with a very high accuracy of ~93%. Using one-hot encoding the accuracy is ~88%, so word embedding was useful in increasing the accuracy

YouTube Presentations:

2 minute: <https://youtu.be/cJRaqV5CsYM>

15 minutes: <https://youtu.be/wHrhfr3NSz4>

# Project Report

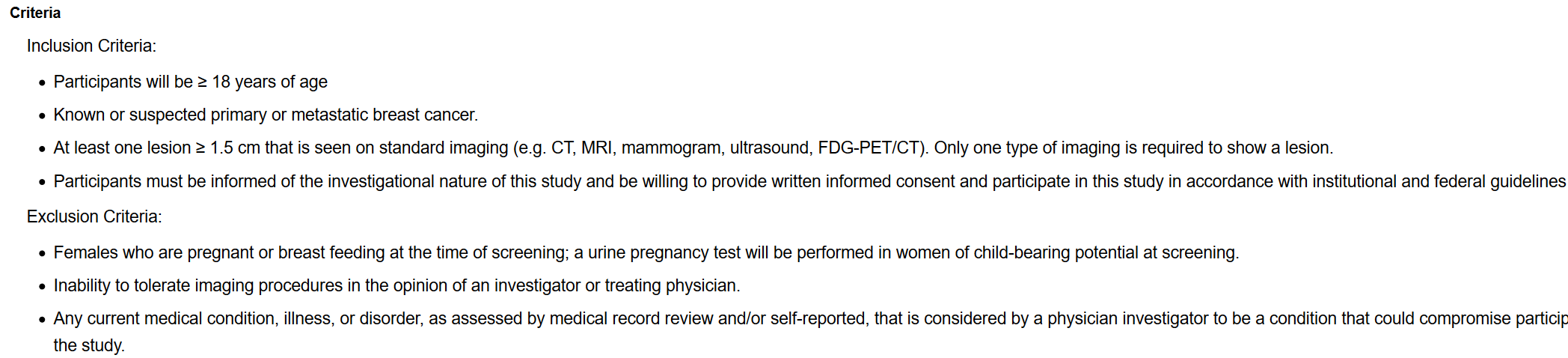
The project report consists of details of the dataset, instructions to download data, links to scripts and code snippets, and results

## Introduction

Before the drug is available in the market, it needs to undergo testing in humans, so drug company recruits patients for testing the drug. These are called clinical trials. The drug company selects patients based on certain criteria, for example a drug for Alzheimer’s needs to have patients who have been diagnosed with the disease. These are called Eligibility Criteria, which are divided into Inclusion Criteria and Exclusion Criteria.

If the criteria are too restrictive, then the trial may not recruit enough patients, and if the criteria are too generic then wrong type of patients may be included.

Below is an example of Inclusion and Exclusion Criteria for a Breast Cancer Study  
<https://clinicaltrials.gov/ct2/show/NCT03863457?cond=breast+cancer&rank=2>



## Usecase for This Project

When designing eligibility criteria of a clinical trial, it will be useful to use existing information of eligibility the disease, so that they are specific to the disease and do not overlap with others. The usecase for this project is trials related to Breast Cancer and Alzheimer’s Disease. Model was constructed to distinguish between eligibility criteria of Breast Cancer

## Data

Data Consisted of inclusion and exclusion criteria of clinical trials. There are more than 100,000 trials at <https://clinicaltrials.gov/> which is the repository of all clinical trials.

Trials that were for phase1, 2 and 3 were extracted for interventional clinical trials.

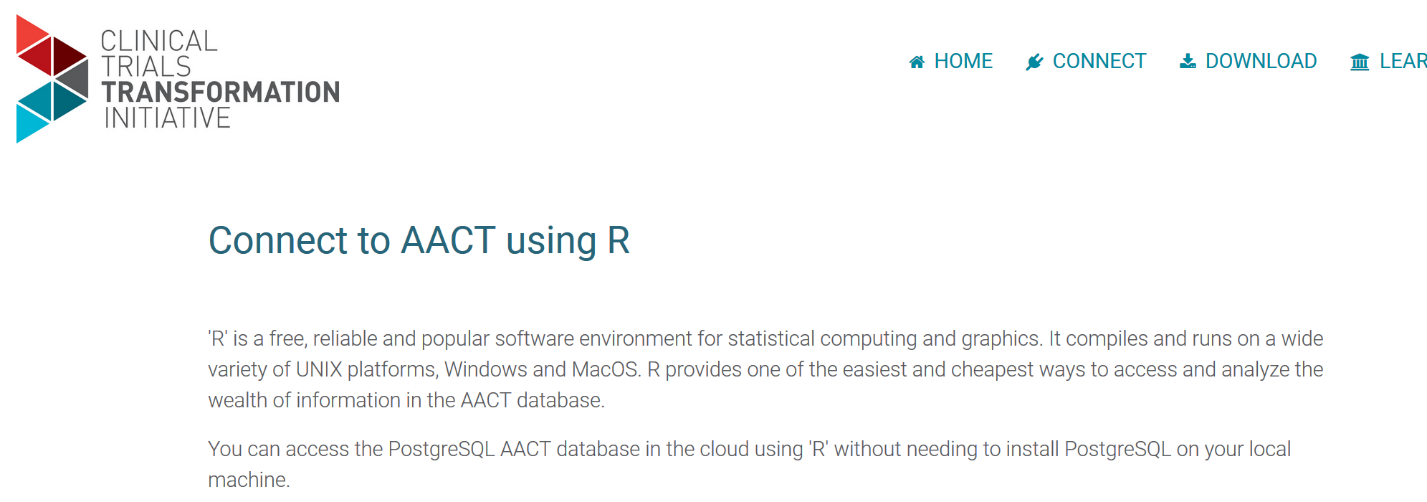
## Data Extraction

Data were extracted using an R API from <https://aact.ctti-clinicaltrials.org/>

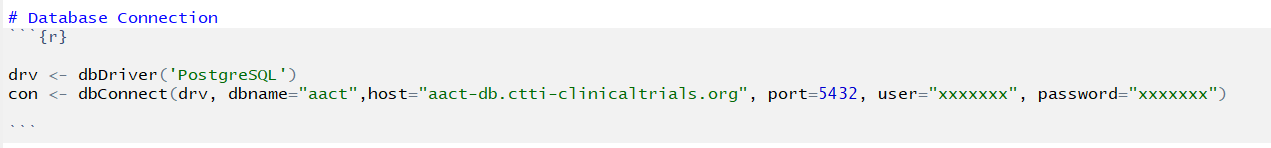
Follow these steps:

1. Create a free account <https://aact.ctti-clinicaltrials.org/users/sign_in>
2. Follow instructions on how to connect R with the API - <https://aact.ctti-clinicaltrials.org/r>
3. Run the R-script ExtractData\_AACT.Rmd <https://github.com/smathurdc/ClinicalTrials/blob/master/ExtractData.Rmd> - change the username and password
4. The scripts generate the input files for constructing the model

AACT – Connecting through R

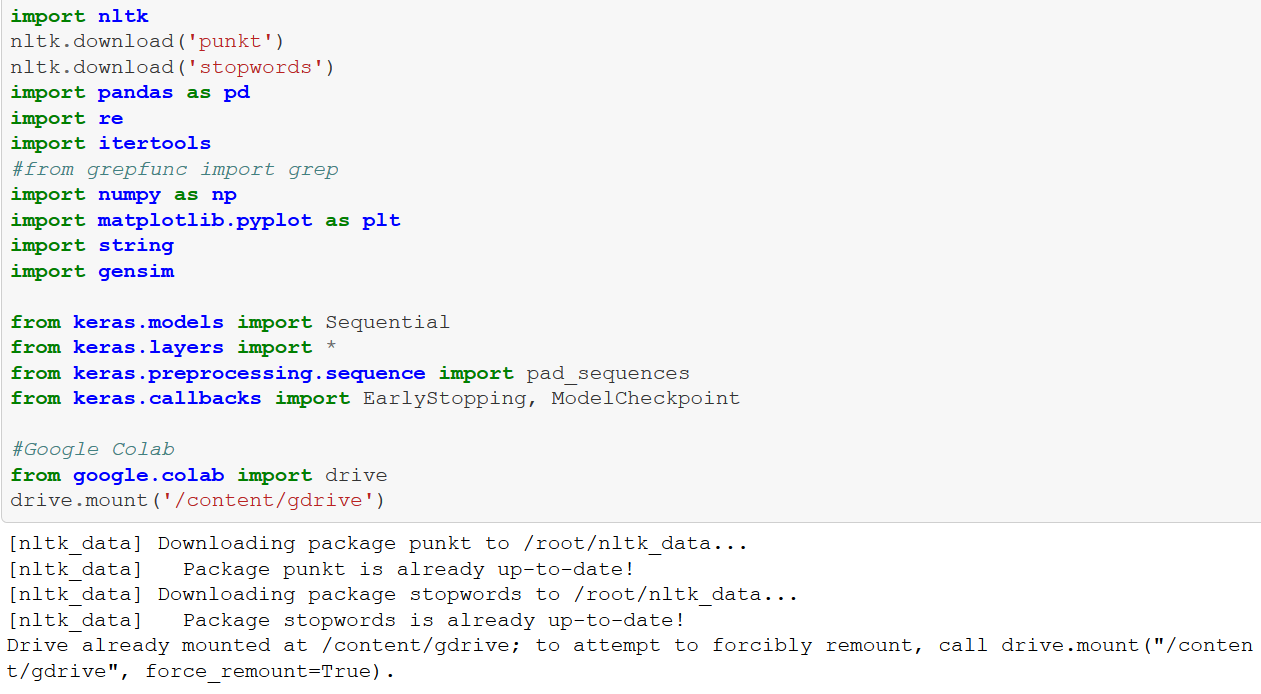
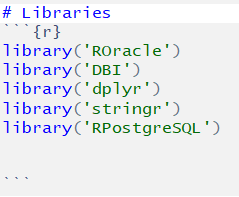


Change username and password



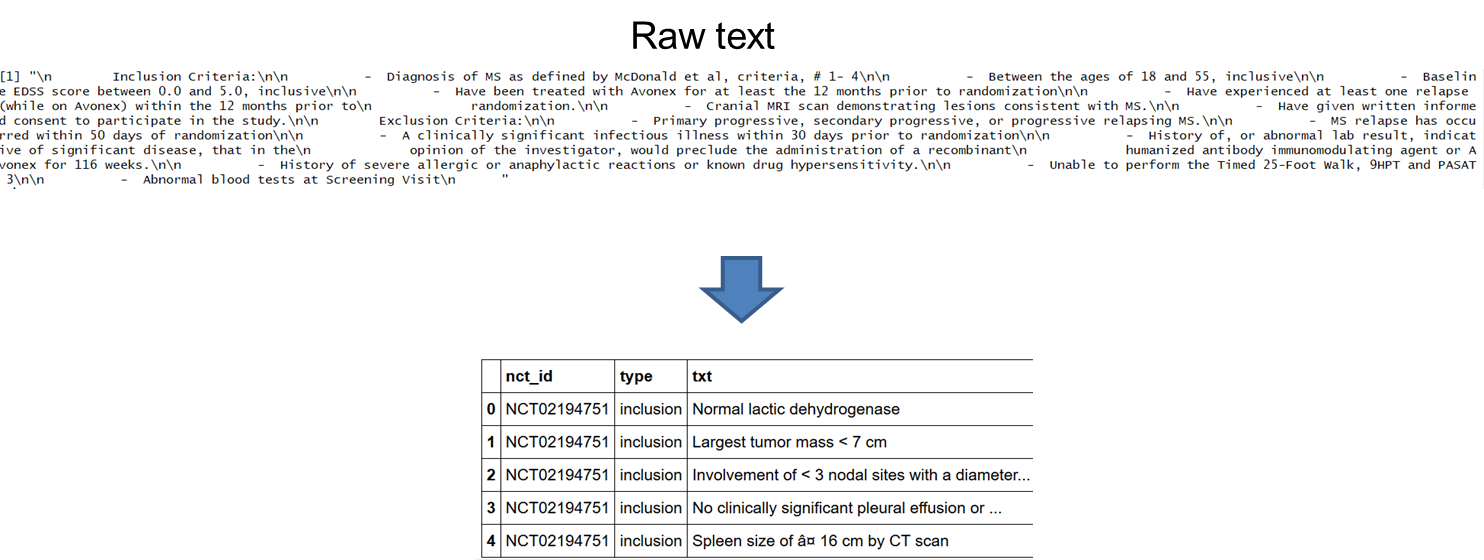
## Software Needs

Following R and python libraries must be installed

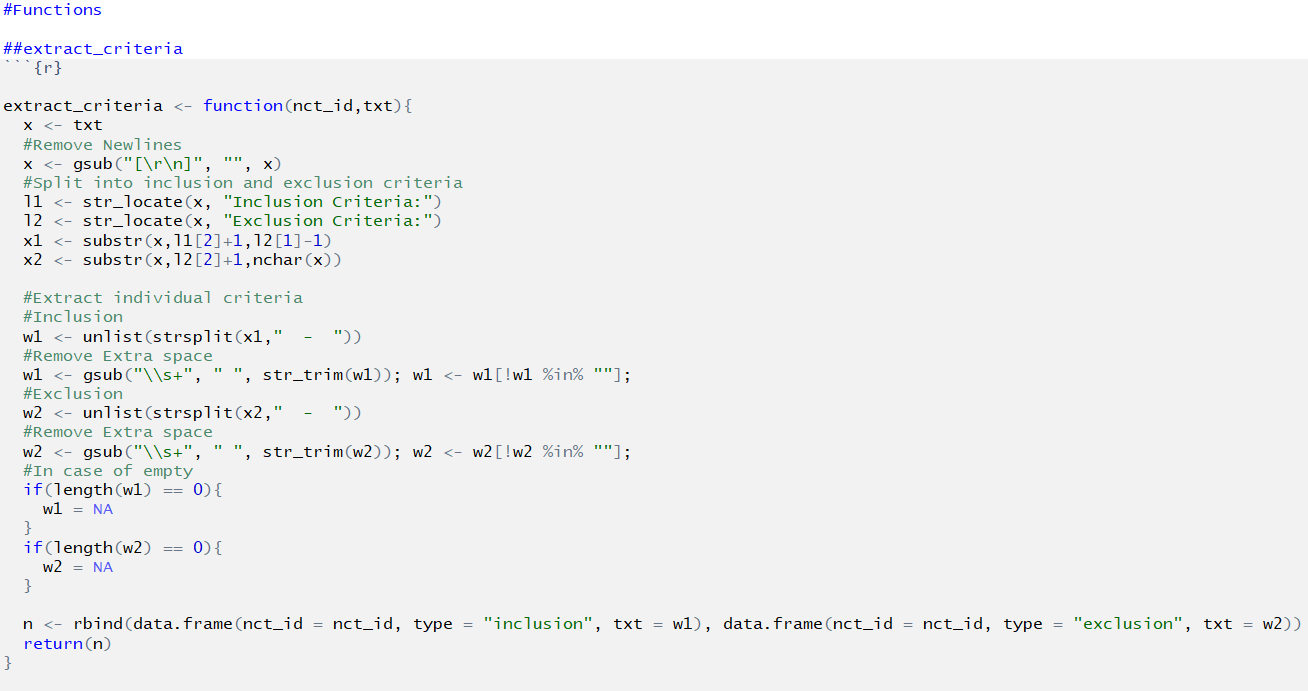


## Data Processing

The data is processed by R. It removes newlines and separates different criteria



There are 2 functions in dataExtract.Rmd that process the data



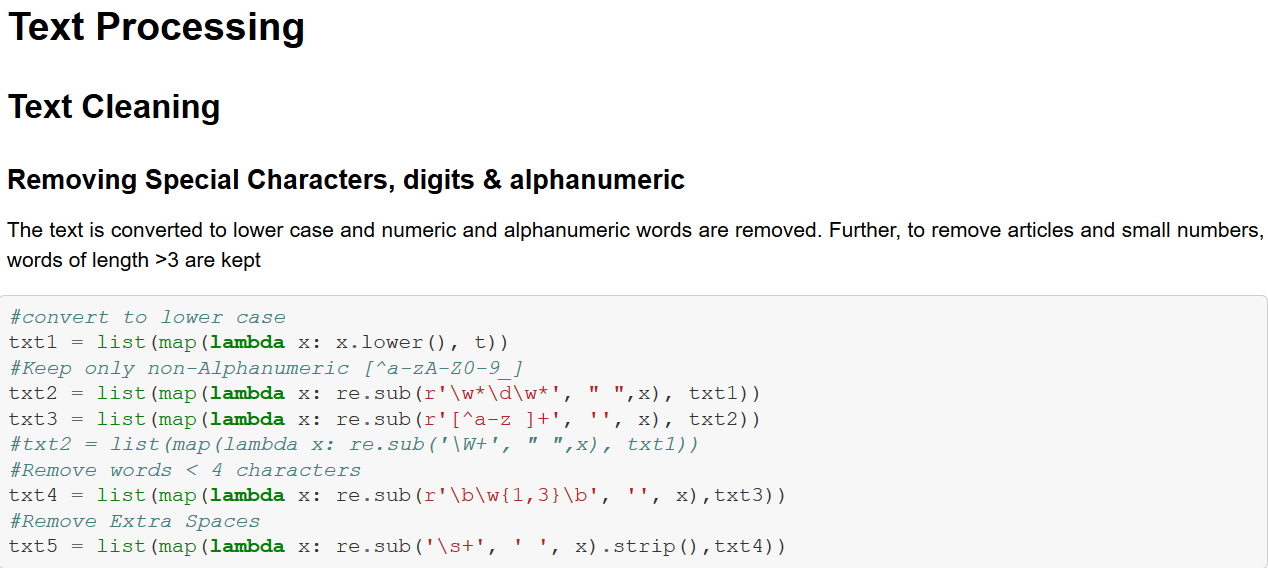
The output of the Rscript is 3 files

1. File for word embedding is > 20 MB - <https://github.com/smathurdc/ClinicalTrials/blob/master/forWordEmbedding> . A smaller file forWordEmbedding\_sample is uploaded. Please change the filename in the jupyter notebook as it is configured to use the 20 MB file
2. Training Data - <https://github.com/smathurdc/ClinicalTrials/blob/master/TrainingData.txt>
3. Test Data - <https://github.com/smathurdc/ClinicalTrials/blob/master/TestData.txt>

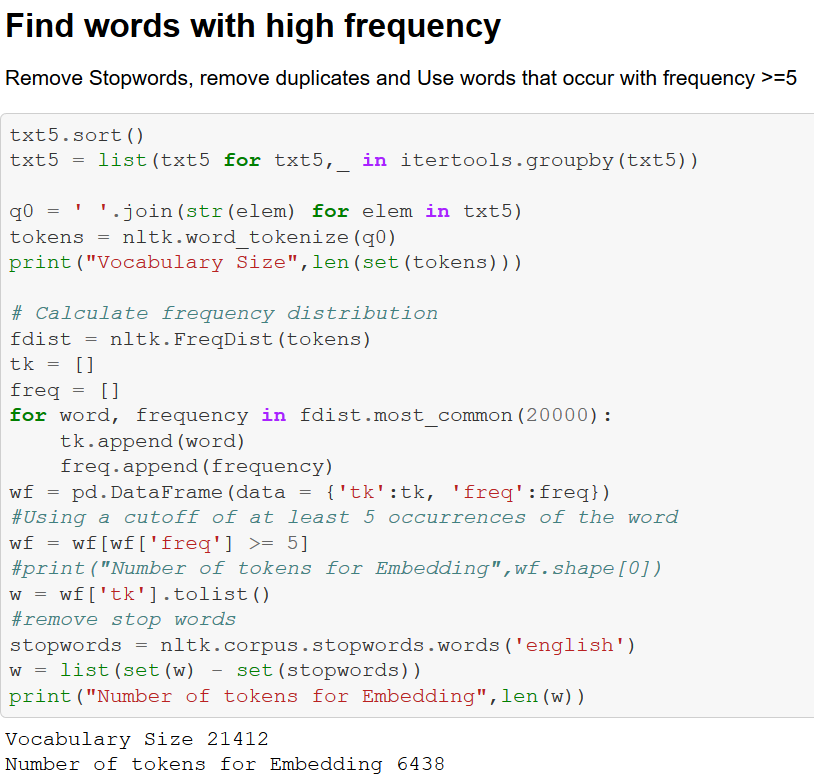
## Further Data Processing

Data processing is continued in the jupyter notebook - <https://github.com/smathurdc/ClinicalTrials/blob/master/ClinicalTrialEligibilityCriteria_SachinMathur.ipynb>

Text processing includes removing special characters, digits and alphanumeric tokens



Words of frequency >=5 in the corpus are used for word embedding

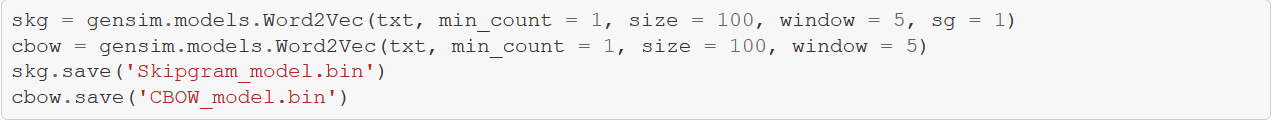


## Word Embedding

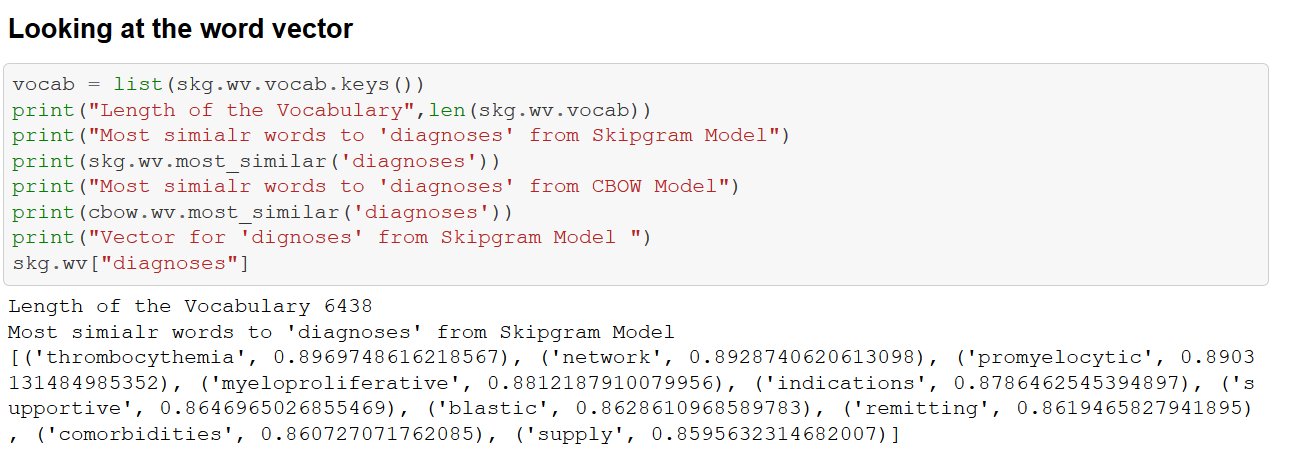
2 types of word embeddings are created

1. Skipgram
2. CBOW

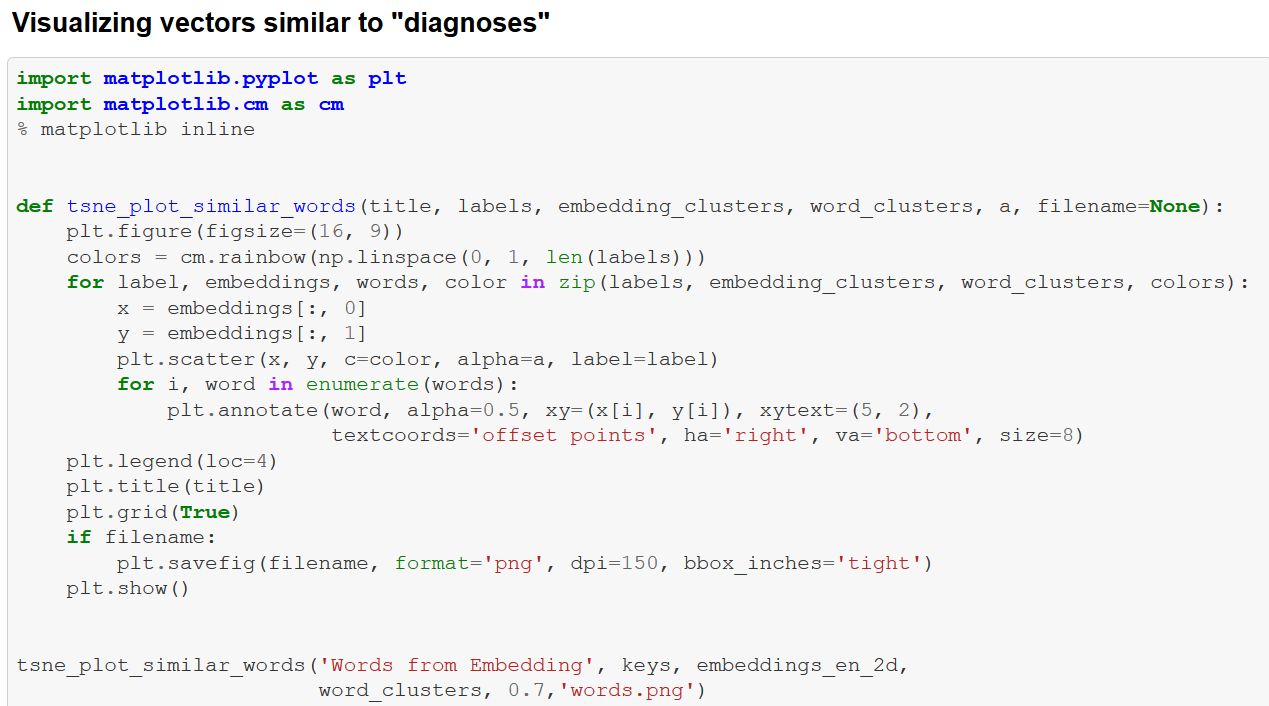
Only skipgram model is used for model creation with 100 dimensions

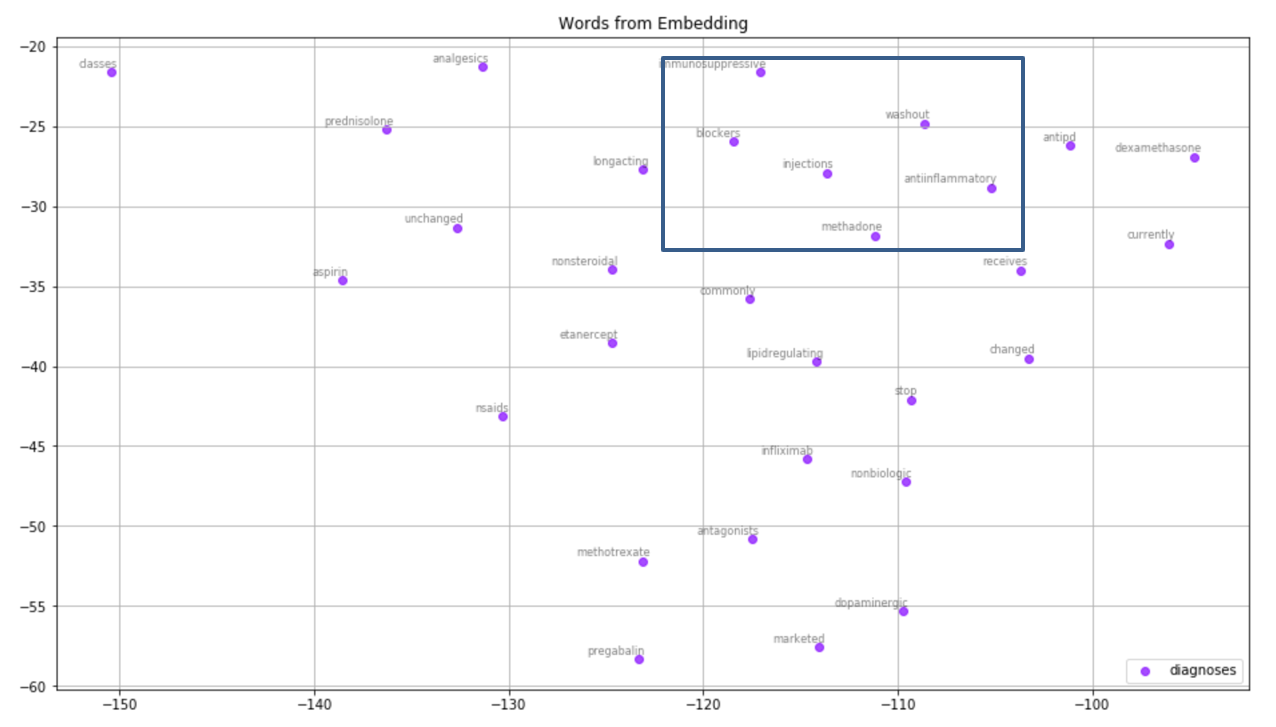


The word vectors are examined to see if similar words have vectors that are close to each other



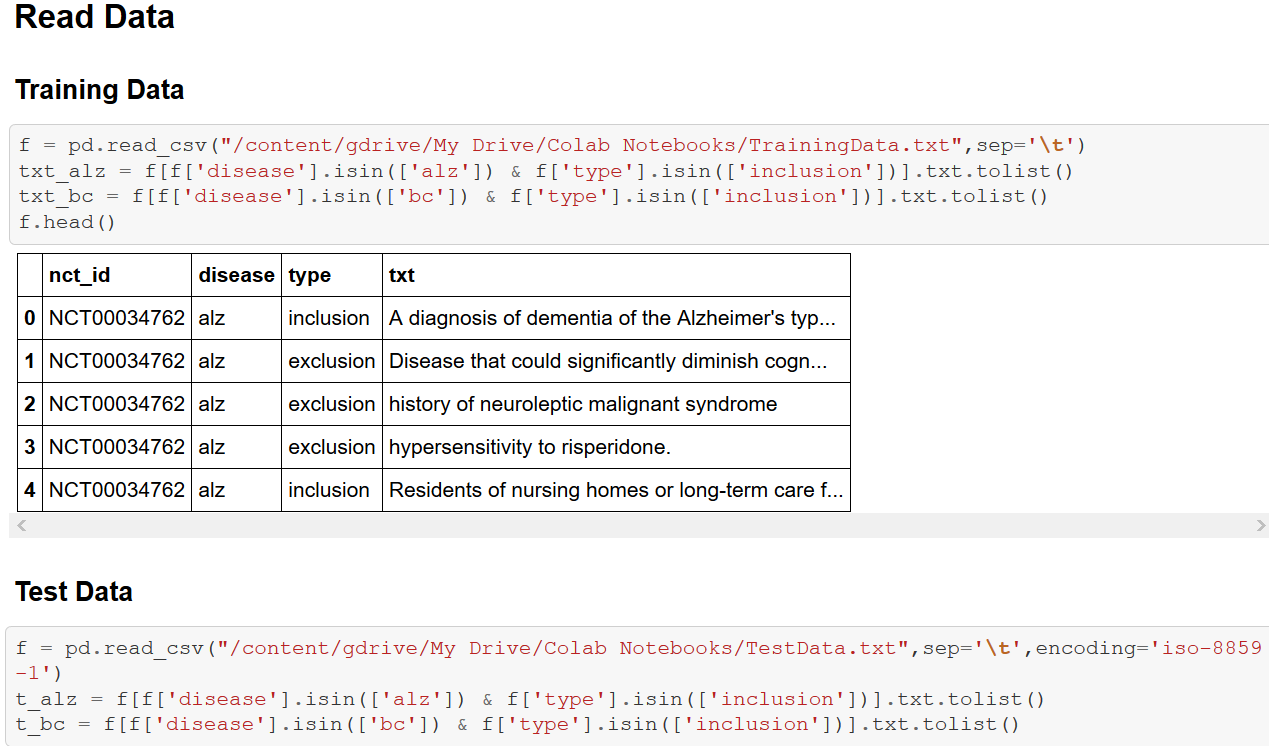
TSNE is used to visualize the similar words for ‘Diagnoses’

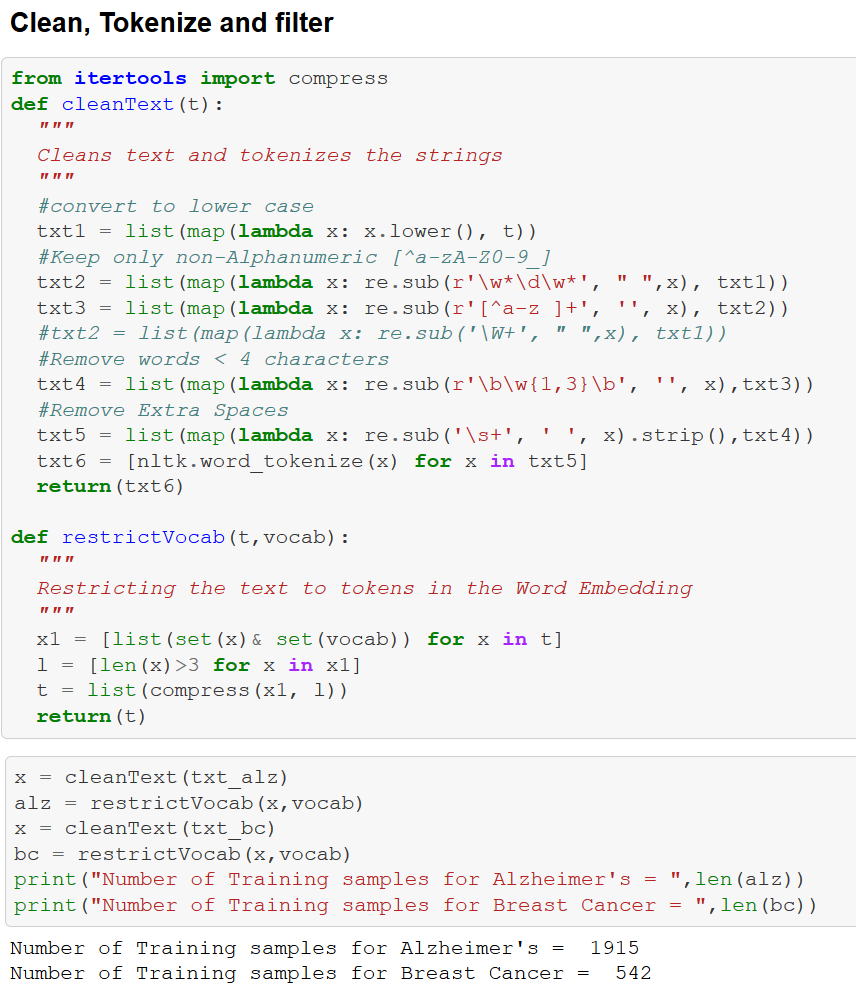


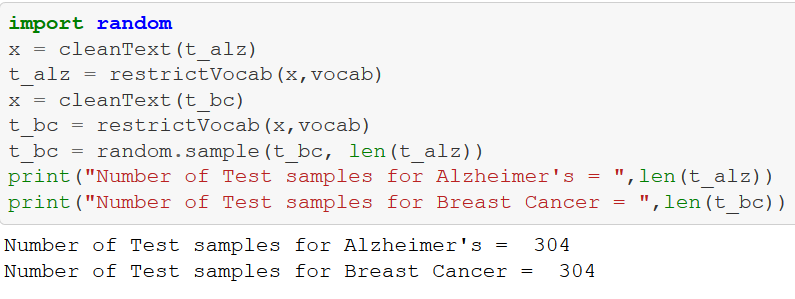


## Training and Test Data

Similar NLP processing is applied to the training and test sets







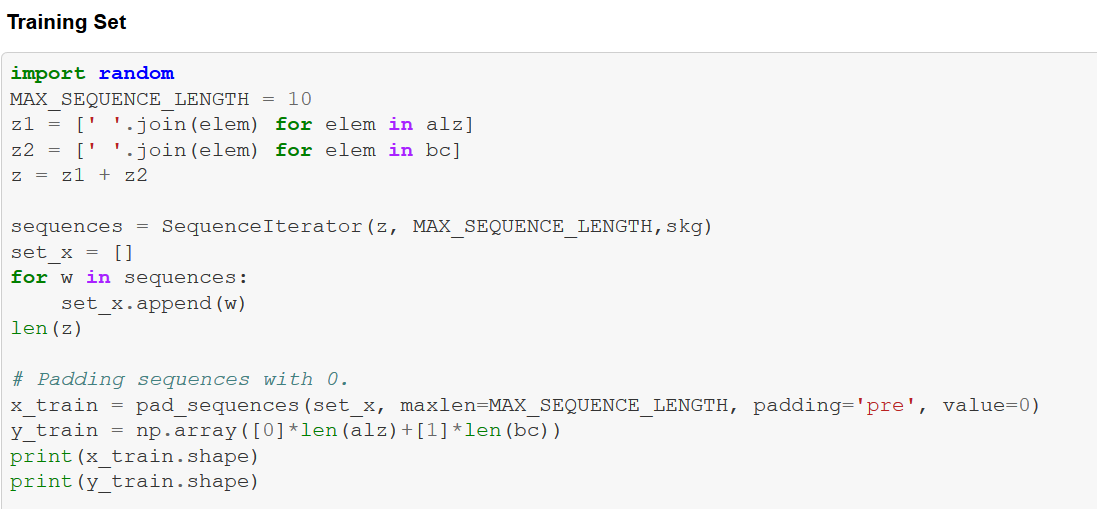
Function ‘CleanText()’ removes punctuation, digits and alphanumeric tokens. Same number of inclusion criteria are selected for breast cancer and Alzheimer’s disease

### Vectorize Data

To train the model, the training data is vectorized and the length of the vector is kept at 10 and the sequences are padded with 0 if the length is < 10

The vectorizing works by first converting the sentences to tokens and then getting the index of the token from the word embedding matrix

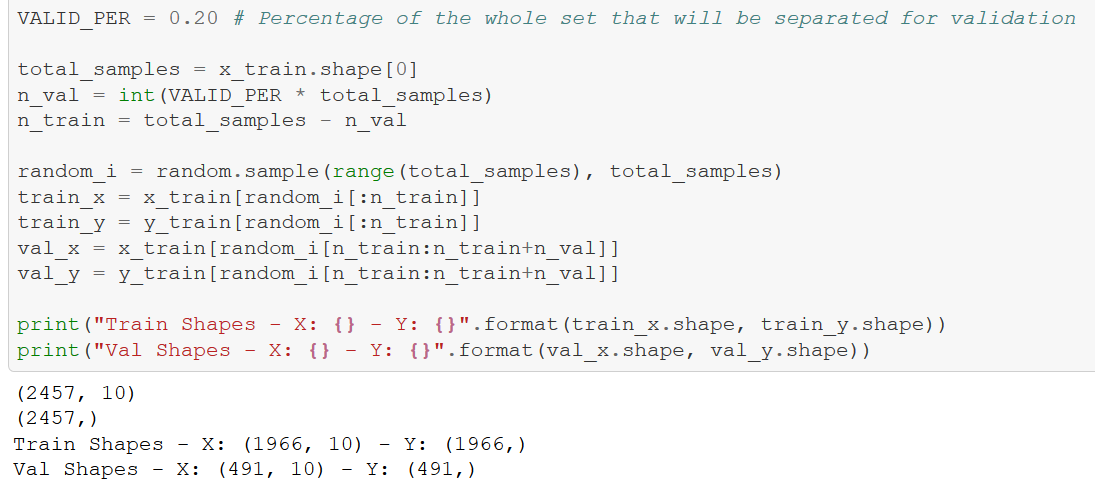




### Validation Set

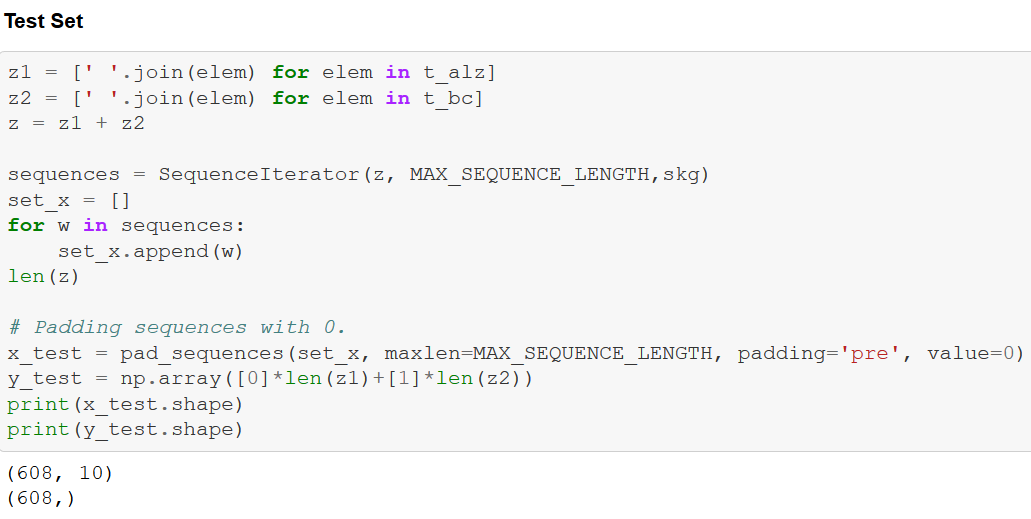
A validation set is constructed from the training set.

The validation set is 20% of the training set



### Test Set vectorization

Testset is also vectorized in the same way with sequence length=10 and padding if necessary



## Model Creation

4 models are created

1. No word embedding Neural network that used one-hot encoding
2. Bi-Directional LSTM using word embedding
3. LSTM using word embedding
4. Simple RNN using word embedding

Bi-Directional Model is shown.

Callbacks for EarlyStopping and best\_model are specified.

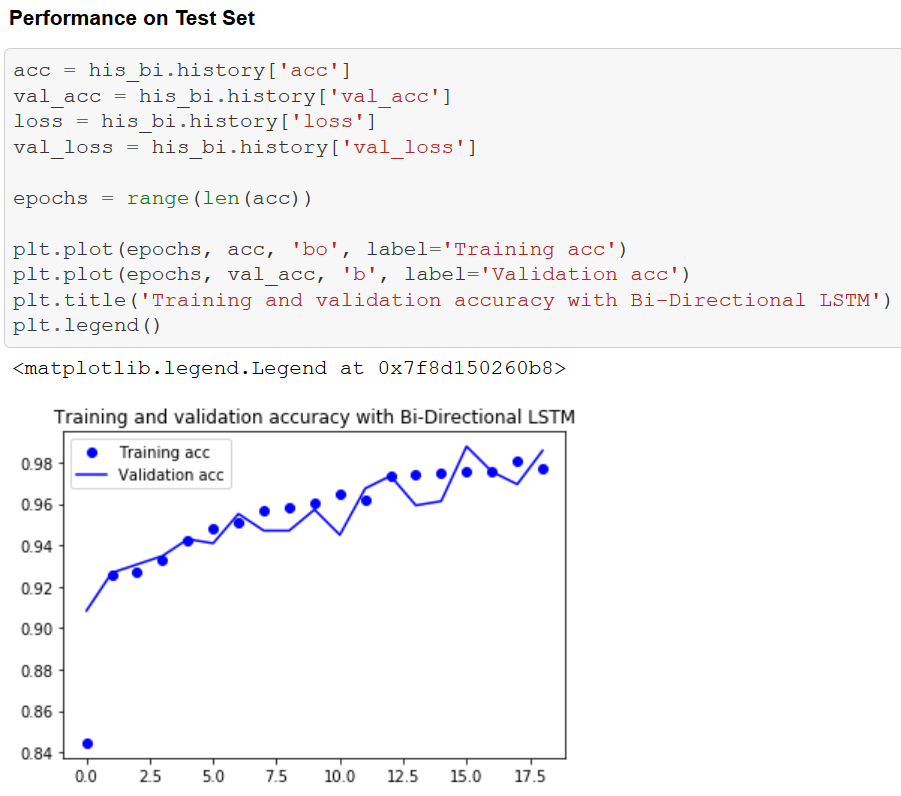
The model is trained with the training set and validation is done on the validation set.

Batch size=64

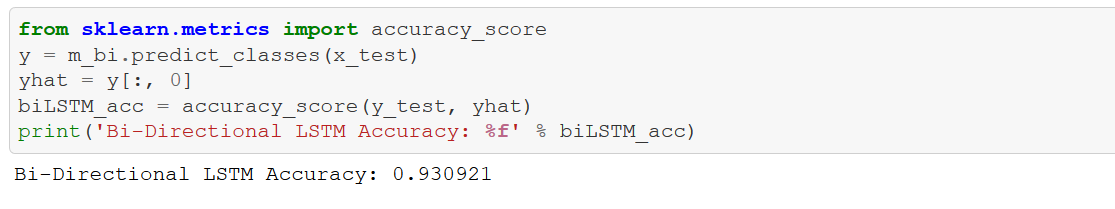
The word embedding is specified as the initial weights of the neural network and trainable is set to FALSE so these weights do not change



#### Performance on Validation set

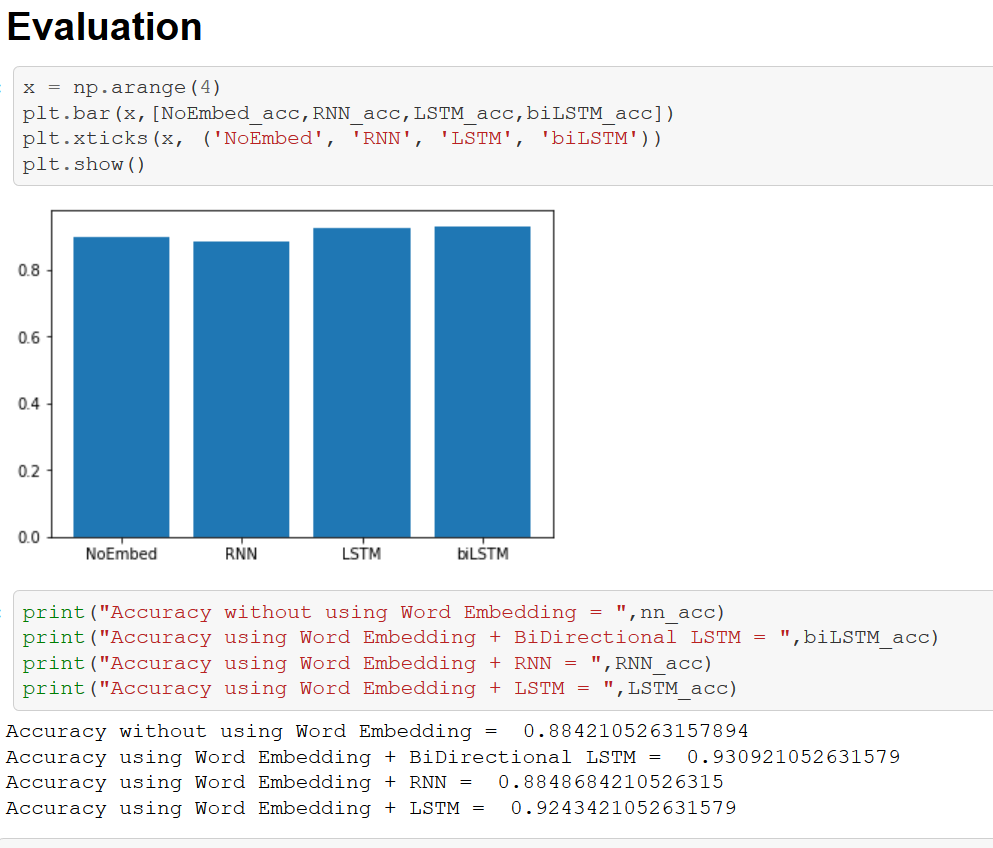


#### Performance on the test set



Similar metrics were calculated for LSTM, Simple RNN and Neural network model without word embedding

## Evaluation



Using an LSTM or Bi-Directional LSTM increases the accuracy to up to 93%.

## Conclusion, Lessons Learned & Future Work

Using word embedding in conjunction with LSTM gives very high accuracy for distinguishing eligibility criteria of Breast cancer and Alzheimer’s disease. Word embedding adds value by using the context of the text compared to merely using one-hot encoding.

Some lessons learned are

* It is important to clean up text before creating a word embedding
* Stemming or lemmatization worsen word embeddings
* One way to check if word embedding makes sense is to use a word and examine its similar words. If the similar words in context are chosen, then the word embedding is good
* Do not run word embedding on PC, always use a GPU
* The LSTM and BiDirectional LSTM are able to classify inclusion criteria with a very high accuracy of ~93%
* Using one-hot encoding the accuracy is ~88%, so word embedding was useful in increasing the accuracy

Future work will include using many more indications and creating separate models for inclusion and exclusion criteria. Different word embeddings such as FastText, Glove, ElMo can be tried to see if accuracy can be improved. Try NLP techniques like stemming in a limited fashion so that word inflexions are avoided. Lastly, find the reasons for the misclassifications in the current model and tweak the method to get higher accuracy.

## References:

<https://machinelearningmastery.com/develop-word-embeddings-python-gensim/>

<https://towardsdatascience.com/google-news-and-leo-tolstoy-visualizing-word2vec-word-embeddings-with-t-sne-11558d8bd4d>

<https://www.kaggle.com/guichristmann/lstm-classification-model-with-word2vec/notebook#Preparing-data-as-sequences-for-LSTM>

## YouTube Presentations

2 minute: <https://youtu.be/cJRaqV5CsYM>

15 minutes: <https://youtu.be/wHrhfr3NSz4>