**Music Cognition**

**Aahana** ​**Khajanchi, Shreya Chudasama, Sreerag Mandakathil**​

**COE, Northeastern University, Boston**

​**khajanchi.a@jhusky.neu.edu**​

# COE, Northeastern University, Boston chudasama .s@jhusky.neu.edu​

**COE, Northeastern University, Boston**

​**mandakathil.s@jhusky.neu.edu Research Paper Draft**

**CSYE – 7245**

**Big-Data Systems & Intelligence Analytics**

**Prof. Nicholas W. Brown**

**COE, Northeastern University, Boston ni.brown@northeastern.edu April 20, 2018**

# ABSTRACT

The idea is to build a machine learning model that will take in a sequence of words as lyrics to a song as input and create an output which is grammatically correct and has similar writing style of a genre. For example, if we give our model a lyric of The Script song as “You can throw your hands up. You can beat the clock. You can move a mountain” we can expect lyrics for a song having lyrics like “throw your hands up and say hello” as its output. To achieve this goal, we need to build a suitable model and train that model such that it can build its own lyrics from the existing data of that band or singer. We crawled down a lyrics site (lyrics.com) and pull in the lyrics and categorized them based on artists, dominant word and sentiments. We will then scrape the data and list down the possible columns and data set. Finally, we split the content of the lyrics into chorus and verse flat files. The lyrics from the site would be in different formats but the structure would be similar so we should be able to fetch that data in csv.

# 1 INTRODUCTION

Can we come up with lyrics of our favorite genre? Can we train our models to write lyrics that makes sense? These questions intrigued us to create a big data machine learning algorithm to scrap lyrics from the web and to generate synthetic computer-generated lyrics for various artist.

The plan is to scrape the data from a lyrics website and serve that as input to our model, train our model to generate the lyrics similar to the lyrics fed to it. For example, if we feed our model with “I'm a 45 spinning on an old Victrola”

We expect the model to return us a song consisting of “I'm a 45 spinning on an old Victrola. I'm a two-strike swinger, I'm a Pepsi Cola”

# 2 METHODS-​

We Create a big data machine learning algorithm to scrap lyrics from the web and to generate synthetic computer-generated lyrics for various artist, determine its billboard rating in the future, predict if the song will make on top 10 list and to display all the result in a web based application as UI.

Below is the high-level design of the process:



Fig. 1 – Process of pipeline

* Scrapping lyrics websites such as lyrics.com
  1. This will be done by giving either an artist or a genre as input to a web scraper

○ Library such as Scrapy will be used to run multiple web crawlers on a single website to optimize scrapping process

* Analyzing the data and segregating the content to lyrics, artist, band, rating
  1. The data will be stored in either a mongo DB or MySQL database for

further processing

* To try out different machine learning models and hybrid techniques to optimize the result
* To make a web based application to interact with the system

# 3 Model Deployment​ –

**3.1 Pickle** – It is the standard way of serializing objects in Python. We used pickle operation to serialize our machine learning algorithms and save the serialized format to a file. Later we can load this file to de-serialize our model and use it to make new predictions

**4** **Python Library** ​ –

**4.1 NLTK**

The Natural Language Toolkit, or more commonly NLTK, is a suite of ​libraries and programs for symbolic and statistical ​natural language processing (NLP) in the ​Python programming language​. NLTK includes graphical demonstrations and sample data. NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to ​over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active ​discussion forum​.

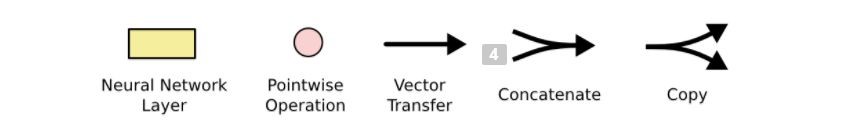
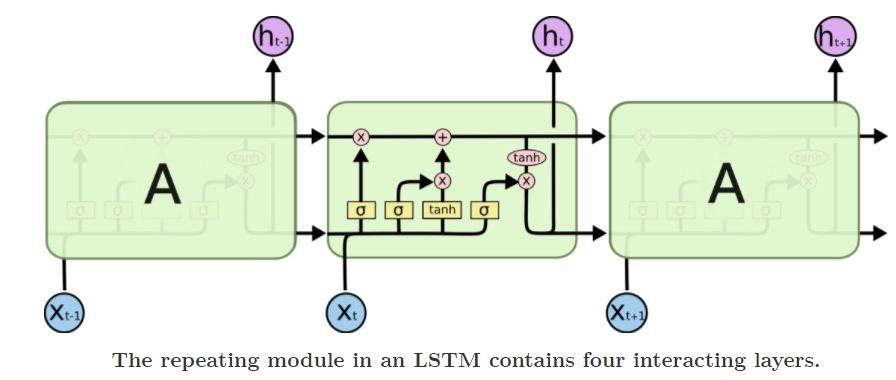


## 5.3 Django

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

## 5.4 LSTM

Long Short-Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies. LSTMs also have this chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way.



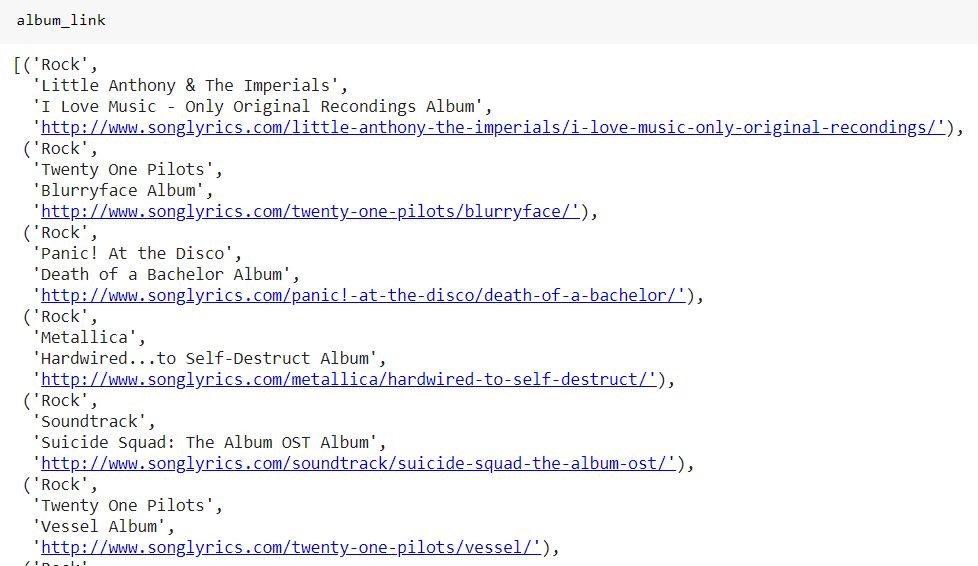
The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates.Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation.

The sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. A value of zero means “let nothing through,” while a value of one means “let everything through!”. LSTM follows a step by step procedure so remembering information for long periods of time is practically their default behavior, not something they struggle to learn.

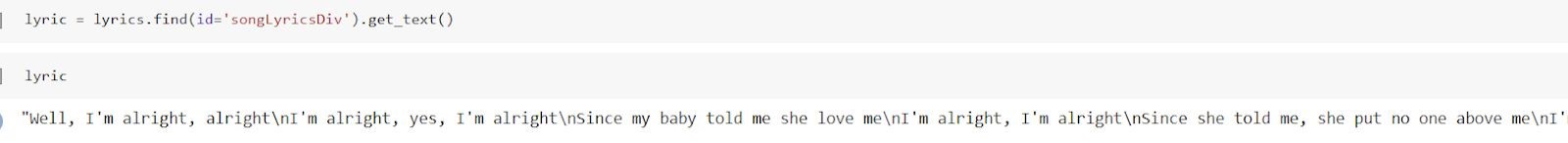
# 5 Code with Documentation ​-

5.1 Web Scraping – To scrap the links from all website.

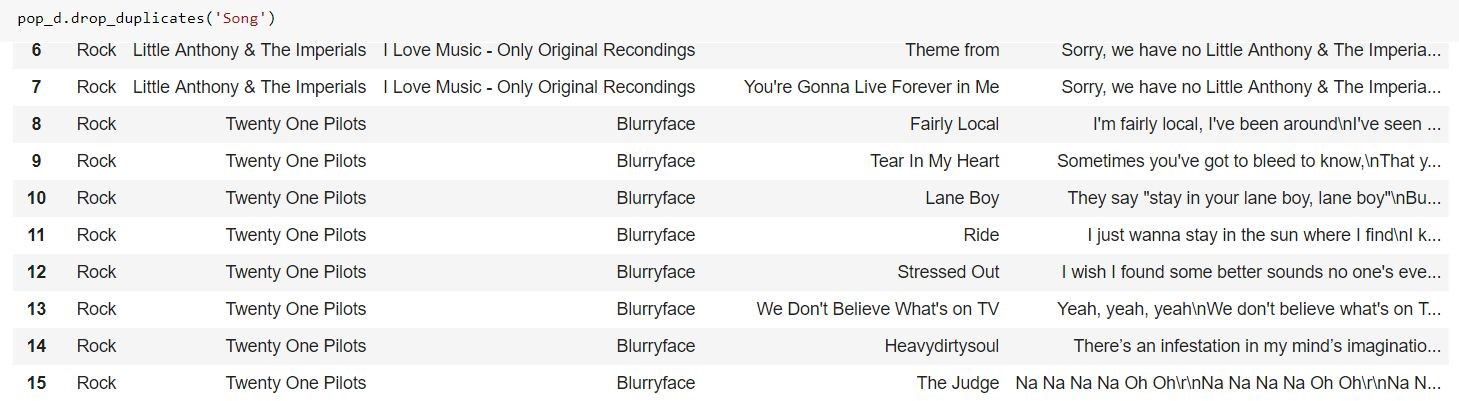




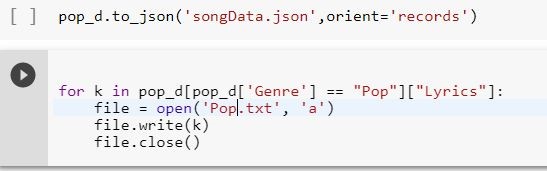
Converting it into a dataframe and scraping lyrics out of each link.



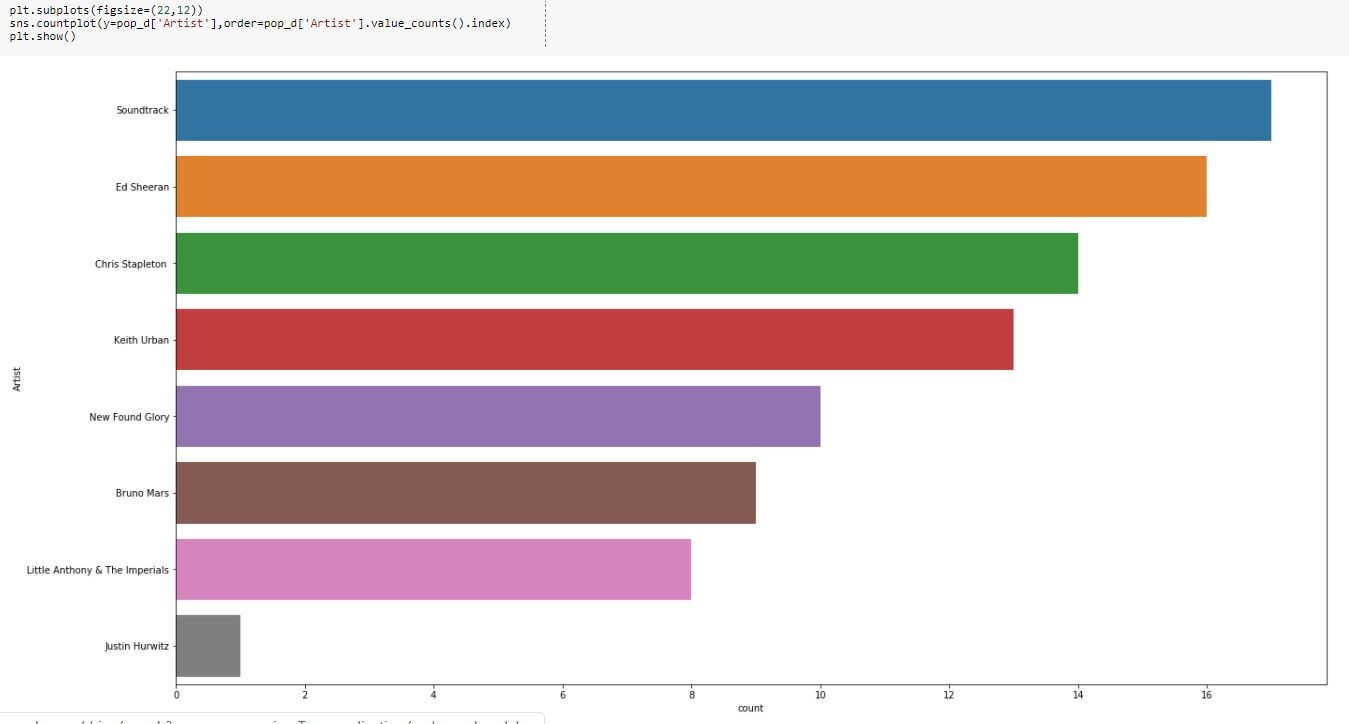
Cleaning the data frame and removing duplicate data



Storing the data in a Json file.

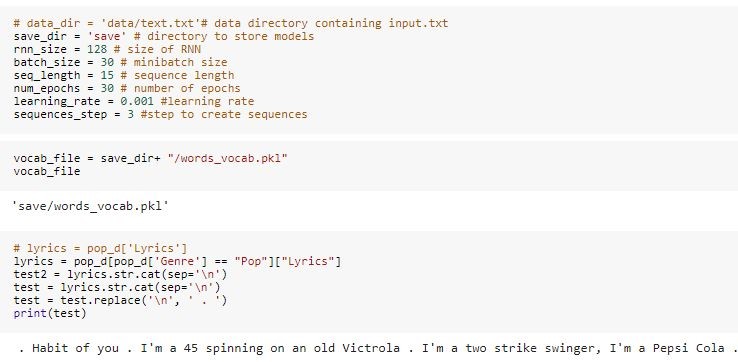


Plotting the graph to know number of songs of few artist.

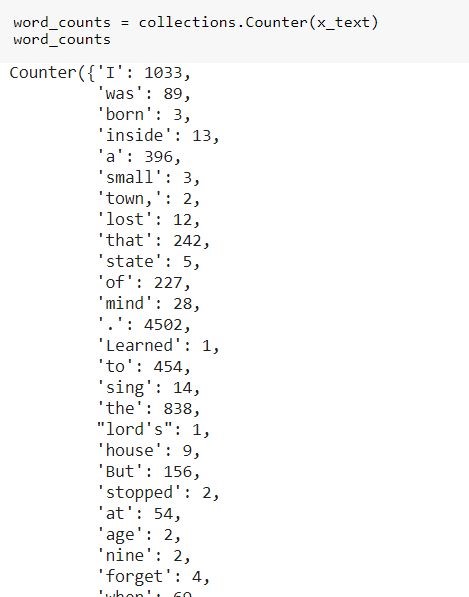


# Training the Model

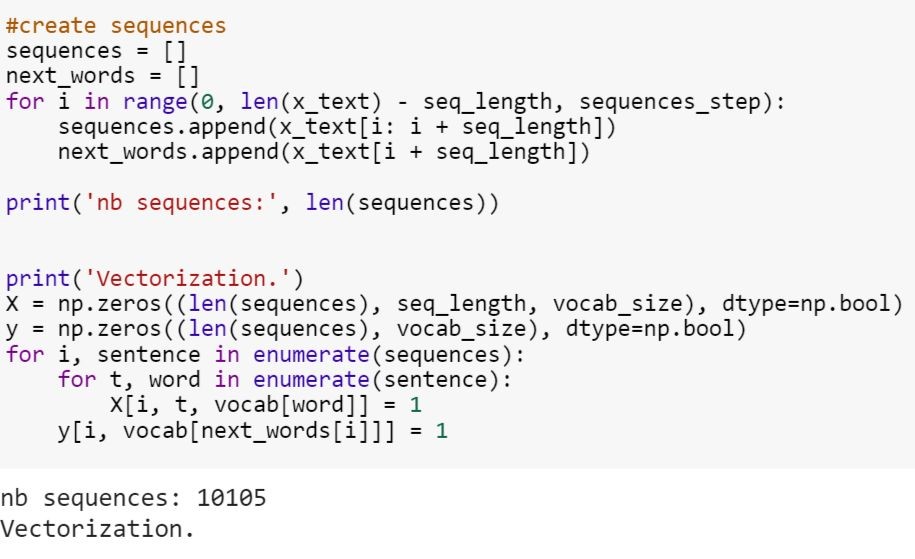
Creating directories and assigning space and building a vocabulary for the model



Counting the number of times, the word has been used



Creating Sequences



Building LSTM model and calculating Epoch and

loss on our training data

**Epoch**:​ A full pass over all of your training​ data

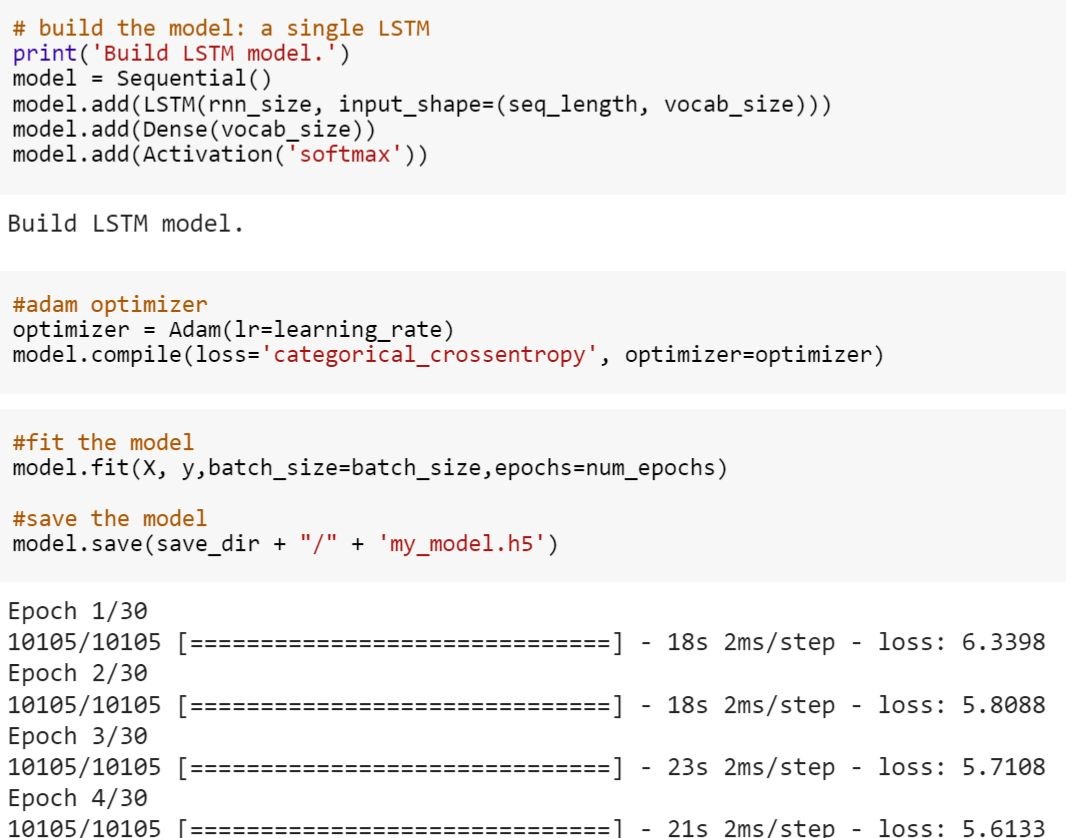
**Loss**:​ A scalar value that we attempt to minimize during our training of the model. The lower the loss, the closer our predictions are to the true labels.

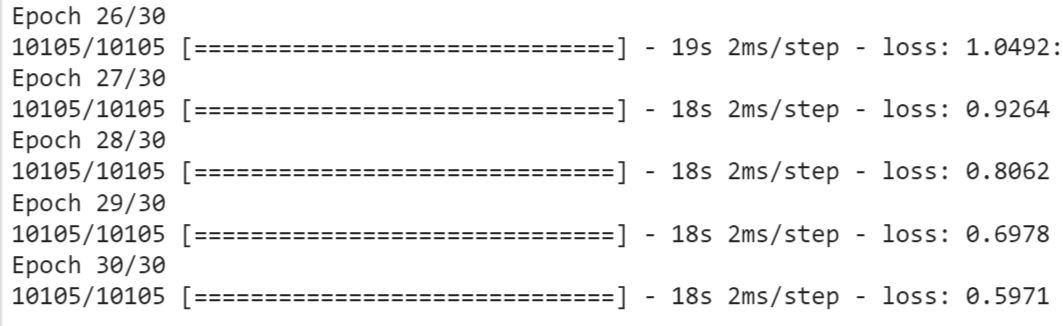
**Adam** is a combination of the advantages of two other extensions of stochastic gradient descent. Specifically:

**Adaptive Gradient Algorithm (AdaGrad)** that maintains a per-parameter learning rate that improves performance on problems with sparse gradients (e.g. natural language and computer vision problems).

**Root Mean Square Propagation (RMSProp)** that also maintains per-parameter learning rates that are adapted based on the average of recent magnitudes of the gradients for the weight (e.g. how quickly it is changing). This means the algorithm does well on online and non-stationary problems (e.g. noisy).

Adam realizes the benefits of both AdaGrad and RMSProp.

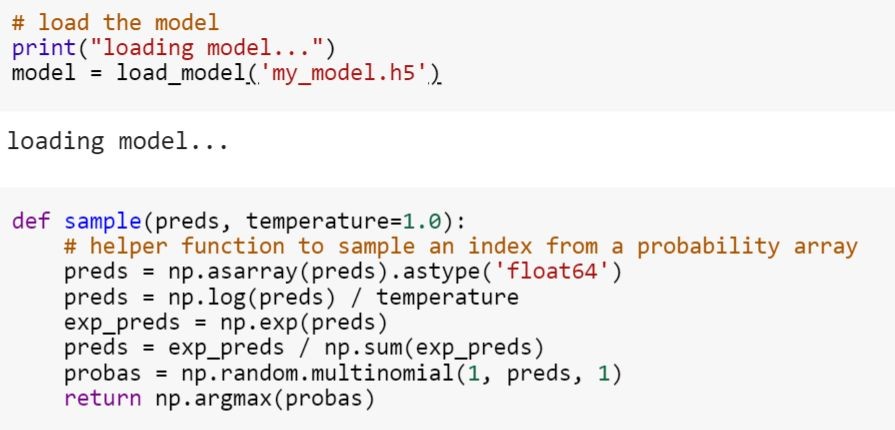




So an epoch concludes when it has finished a training pass over all 10105 of the observations.

As we can see in the above observation, there is a decrease in loss over 30 number of epochs. It has reduced to 0.59 from 6.33.

Building the model using LSTM Keras



**Methods for Python Text to Speech-**

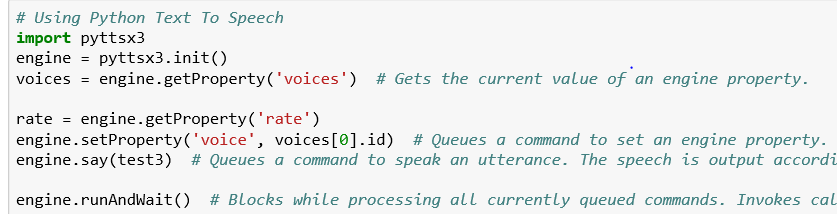
GetProperty(name : string) – gets the cu rrent value of an engine property.

SetProperty - Queues a command to set an engine property.

The new property value affects all utterances queued after this command.

Say() - Queues a command to speak an utterance. The speech is output according to the properties set before this command in the queue.

engine.runAndWait()- Blocks while processing all currently queued commands. Invokes callbacks for engine notifications appropriately. Returns when all commands queued before this call are emptied from the queue.



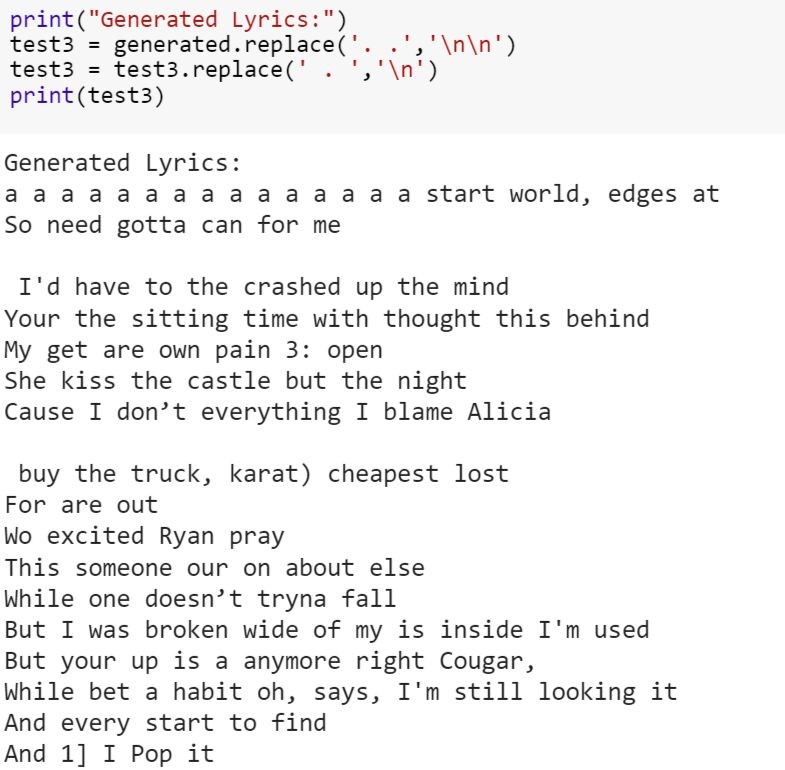
# Generating Lyrics

Below is the code that generates lyrics by

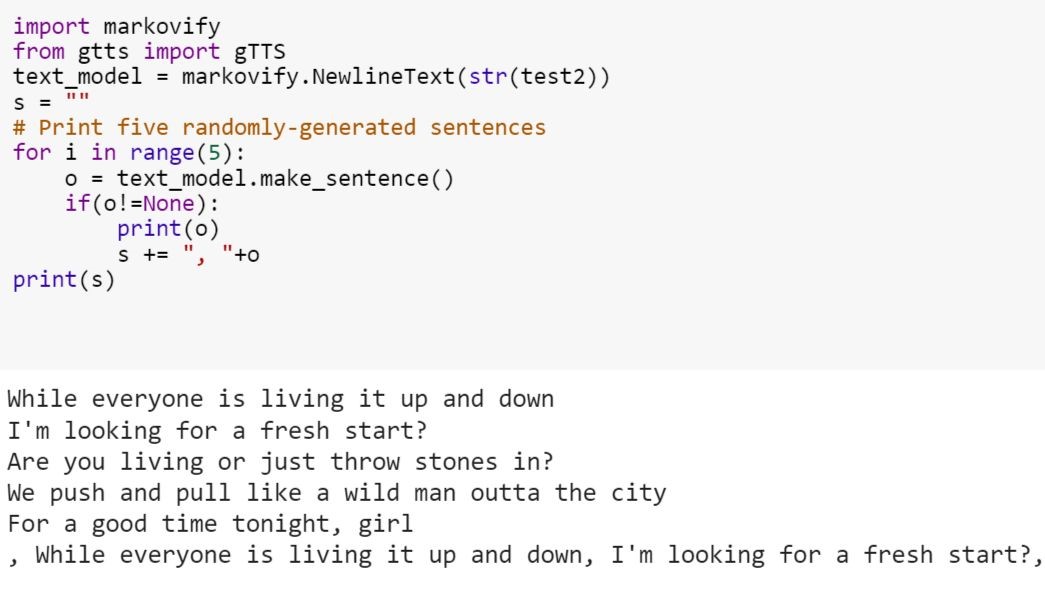
calculating and predicting the next word

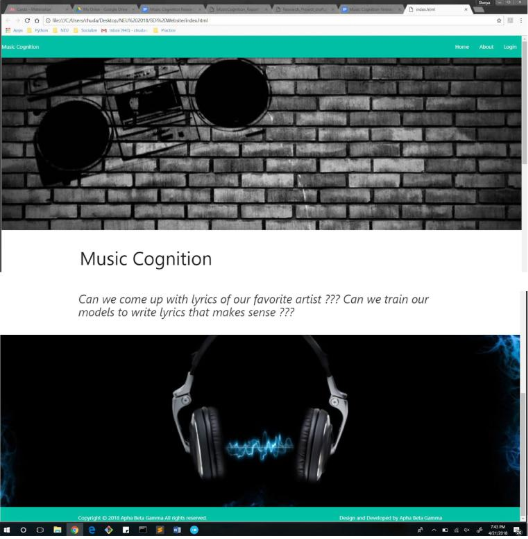


Printing out the lyrics generated using the model

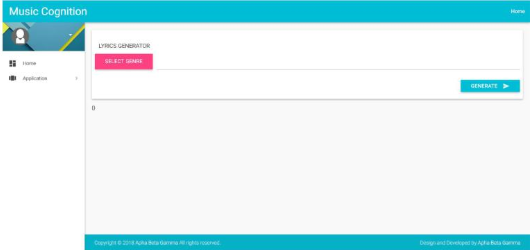


As you can see, the above sentences does not make much sense. We have used **markovify** to generate sentences that makes sense and are grammar correct.

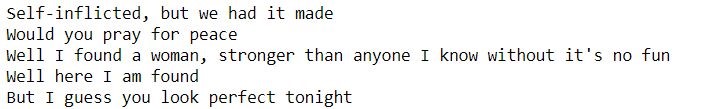




**Python Text to Speech**



**6 Results** – After implementing and testing the above models to generate lyrics below is the generated text.



Our best model is made using LSTM Neural Network and Markov chain rule.

**ACKNOWLEDGMENTS**

Authors would like to thank our professor, teaching assistants, family and friends who supported in the completion of this research project. Appreciating everyone who helped us knowingly or unknowingly for this project.

## Reference

1. ​http://colah.github.io/posts/2015-08-Understanding-LSTMs/
2. https://towardsdatascience.com/understanding-lstm-and-its-quick-implementation-in-keras-for-sentim ent-analysis-af410fd85b47
3. ​https://medium.com/@ivanliljeqvist/using-ai-to-generate-lyrics-5aba7950903
4. http://www.nltk.org/book/ch01.html
5. https://nlp.stanford.edu/courses/cs224n/2009/fp/5.pdf