**Apples and Apple’s**

Setup of MySQL database

Please run the file ‘Solution/mysql-python.py’ initially

This file will create a database named ‘intern’ in MySQL and will then create a table named ‘apple’

During its compilation, it will ask the user for three parameters from MySQL:

1. Host Name
2. User Name
3. Password

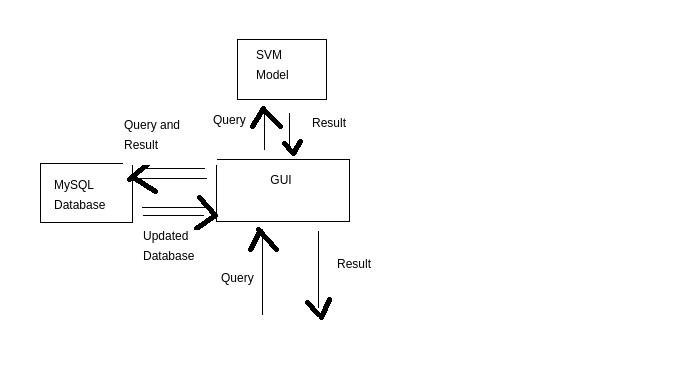
This MySQL database will be used to store the input and output.

Web Application (GUI):

After running ‘Solution/mysql-python.py’, run ‘Solution/app.py’.

This will run a flask based GUI. Hover over the ‘database’ icon to see your MySQL database intern’s table apple. This is a real-time database, ie, new records will be added to the database as soon as the output is fetched.

Deployment and Architecture of GUI and API



In the frontend, a search box will ask for query, upon submitting that request will be taken at the backend where our pre-trained model will produce a result based on query. The result will be shown in another window. Also at the backend of the GUI, a connection is set up with MySQL local database which stores each and every query and result and also shows the complete sql table in pandas dataframe style, upon requesting from the home page of the GUI. The database stores information in near real-time.

NLP Model for classification

The basic workflow of a NLP based classification task is:

1. Import the dataset
2. Cleaning the dataset (text files)
3. Feature Engineering
4. Model Building
5. Model Evaluation

Step 3,4,5 are to be repeated to identify the best model and fine tune the model

I approached on a similar line, will discuss about the section-wise approach now:

Section 1 - Import Required Modules

Imported the required modules

Section 2 - Load the dataset

Read the text file and split it sentence-by-sentence and kept only those sentences whose character length is greater than 5. This helped in removing blank lines as well as smaller sentences as they would not contribute much.

Also created a dataframe object to represent the data along with its label class

There are total 806 sentences (rows), where 533 instances are of apple computer-company and 273 instances are of apple fruit.

Section 3 - Data Preprocessing

The text data contains special characters, lower case-upper case words, numbers, etc. which produces noise within the dataset. The text data is processes using regular expressions, which eliminate a particular text format from a sentence.

Stop words: Common english words like - the, it, then, has, have.. Does not convey any information about the context of the sentence. Such words are also removed from the text data

Stemming: Words like read, reading … convey the same meaning. Such words are to be reduced to their parent word. Stemming is also applied on the text data

Section 4 - Word Cloud

Image of words where size of word is directly proportional to its frequency of appearance in the documents

Section 5 - Feature generation using Bag-of-Words

Bag of words is a method which gives a matrix where each sentence is broken down into a 1d array of 0 and 1, where joining the features corresponding to 1 will produce the sentence.

In out case, there are total 2676 features and 806 sentences.

Section 6 - Model Evaluation - Bag of Words

I’ve used 7 most commonly used classifiers for evaluation.

Using all of the features (2676), prediction accuracy is calculated corresponding to each model

Section 7 - Feature generation using TF-IDF

TF-IDF is used to weight words according to how important they are. The words that appear frequently in a document are given less weightage and vice versa.

Section 8 - Model Evaluation - TF-IDF

Similar to that of Bag of Words, models are again evaluated for TF-IDF based feature sets

Section 9 - Comparing models prepared by BoW and TF-IDF

Just on the basis of this bar graph, it can be concluded that TF-IDF increase model’s performance (in general). Naive Bayes classifier prepared using Bag of Words method has the highest accuracy.

Section 10 - To find out optimal number of features and best model (BOW)

In this section, I’ve attempted to assess the performance of the models prepared using Bag of Words method as the number of features is varied.

Observing the line chart, Naive bayes classifier has an accuracy of above 95% all the time. Random Forest and Decision Trees have accuracy below 90%. KNN have a relatively poor accuracy as compared to other models. SGD have an accuracy of over 90% most of the time but varies significantly with the number of features.

Logistic Regression, Naive Bayes and SVM have relatively constant accuracy (very little variation) throughout the time.

Section 11 - To find out optimal number of features and best model (TF-IDF)

Random Forest, Logistic regression and Decision Trees have accuracies below 88%. While SVM, SGD and Naive Bayes have accuracies above 90%

SGD, SVM and Naive Bayes are top three models as of now. TF-IDF method has increased the accuracy of SGD and SVM only.

TF-IDF method basically penalizes the most frequent word, as the problem in hand is of classification, it would not be a good idea to penalize the common terms as it would restrict the model to see the common words like vitamin, computer, which may appear frequently in the document.

So the best model is Naive Bayes classifier whose features are prepared from Bag of Words method. Its accuracy is above 95% for all the number of features.

From section 12, it can be seen that the difference between the highest and lowest accuracy of Naive Bayes model is less than 1%, which conveys that it has a stable performance over the entire range of number of features.

Section 13 - Final Model - NB with BoW with all features

The final model is NB whose feature set is prepared using BoW method

Section 14 - TEST

A function for testing. Just place the text file in the same directory as this .ipynb notebook and pass the name of the file as a parameter.