Mongo DB framework for Text Classification

## Why Mongo Db?

Mongo DB being a cross-platform oriented , document oriented database provides us with following benefits :

* High Performance
* High Availability
* Easy Scalability

Mongo Db works on concept of collection and Document.

## Mongo DB database

Like a file system, each database here has its own set of files. It’s a physical container for collections.

A single Mongo DB server typically has multiple databases.

## Collection

A group of Mongodb documents. Similar to RDBMS table. A collection exists within a single database.

Documents within a collection can have different fields.

## Document

Set of key-value pairs having dynamic schema.

|  |  |
| --- | --- |
| **RDBMS** | **MongoDB** |
| Database | Database |
| Table | Collection |
| Tuple/Row | Document |
| column | Field |
| Table Join | Embedded Documents |
| Primary Key | Primary Key (Default key \_id provided by mongodb itself) |
| **Database Server and Client** | |
| Mysqld/Oracle | mongod |
| mysql/sqlplus | mongo |

# Basics of Text Classification

A supervised machine learning task to label a text document and using labels to train a classifier.

The process of classification consists of Data preparation, Feature Engineering and Model Training.

Packages used for creating framework for text classification using mongo Db:

* Numpy
* Pandas
* Scikit Learn
* Pymongo
* Keras

# Procedure

## Data Preparation

1. Using pymongo for loading data from mongo db into python.
2. Connect to port on which mongo db server is active
3. Connect to collection you want to use as text data(Data corpus)
4. Using Cursor to list the data.
5. By Default, an id column is added to corpus on mongo Db side.
6. Convert the corpus to a list using cursor and load it into pandas data frame.
7. Remove the id column from the data frame
8. Create another data frame using texts and labels
9. Split the dataset into training and validation datasets
10. Using hot encoding to convert categorical variables into a form that could be provided for further processing

# Feature Engineering

1. Conversion of raw text data loaded into data frame into vectors.
2. Count vector as features ( Representation of matrix notation of dataset)
3. Create a count vectorizer object & transform the training & validation data using CV object.
4. Calculation of TF-IDF score which represents the relative importance of a term in document & entire corpus.
5. Calculation of word-level, n-gram level and character level TF-IDF.
6. Word Embedding to represent words & docs using Dense Vector Representation.
7. Loading the pre-trained word embeddings
8. Creating a tokenizer object
9. Transforming the text document to tokens & padding each sequence to same lengthj
10. Creating a mapping of token & their embeddings
11. Counting of basic text features like character count , word count , uppercase character counts ,title word count etc

# Model building

### Intro to RNN

A RNN forms a class of ANN where connections are made between the nodes form a directed graph along the sequence. Using an external embedding we can enhance the precision of RNN because it integrates new information about the words like glove, word2vec etc.

### Why RNN?

Traditional NN cannot do this which seems a major shortcoming. For example if you want to classify what kind of events is happening at every point in the movie, it’s unclear how a traditional NN could use it’s reasoning about previous events in the film to inform later ones.

Hence the answer Recurrent NNs, the networks with loops in them allowing info to persist. They may seem complex but are actually multiple copies of same network each passing a message to a successor forming a chain-like structure.

### Drawbacks of RNN

RNNs become unable to learn to connect the info hence the LSTM networks(Long Short Term Memory Networks) having long term dependencies.

LSTM also have a chain like structure but repeating module has a different structure, instead of having a single NN, there are four interacting in a very special way.

### Implementation

1. Add an input layer
2. Add the word embedding layer
3. Add the LSTM layer
4. Add the output layers
5. Compile the model.
6. Return the model
7. Taking this model , fit the training dataset on the classifier
8. Predict the labels on validation dataset
9. Generate the accuracy score.
10. Return the score.

## Important Notes

1. Load the Training and Testing dataset onto Mongo Db as an when required
2. Make changes into the collection for the database names
3. Make the necessary changes in the procedure.

## Conclusion

This framework provides a dynamic model for loading corpus into Mongo DB and using text classification.