LAB 3 -Discussion

Agenda

- What is Spark? Why Spark?
- Data collection for Lab 3
- Extracting Features from text
- Classification of text
- Trying out on new data and documentation

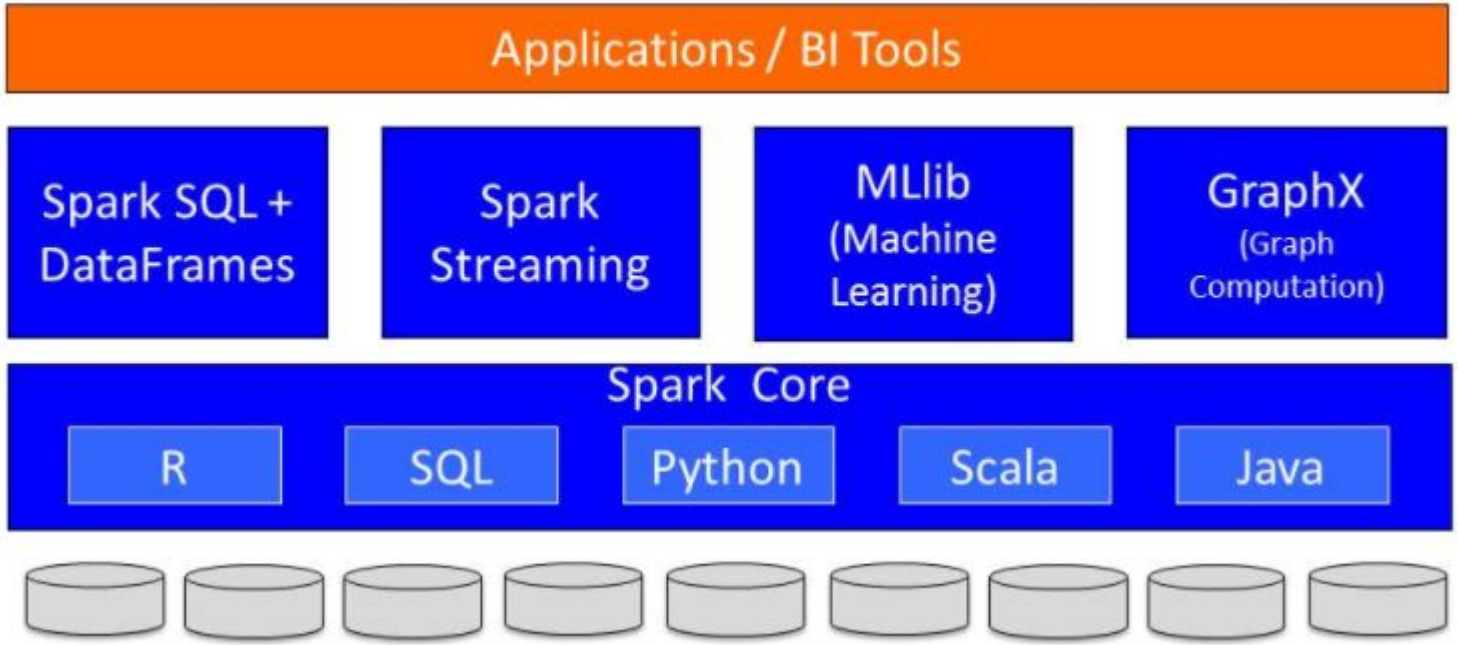
Spark

- Spark is a general-purpose data processing engine that is suitable for use in a wide range of circumstances.
- Spark is capable of handling several petabytes of data at a time, distributed across a cluster of thousands of cooperating physical or virtual servers. It has an extensive set of developer libraries and APIs and supports languages such as Java, Python, R, and Scala
- Advantage of Spark over Hadoop is that it is much faster. Spark’s ability to store data in memory and rapidly run repeated queries makes it a good choice for training machine learning algorithms.

Spark

- Spark Machine Learning Library (MLlib): A library of prebuilt analytics algorithms that can run in parallel across a Spark cluster on data loaded into memory.
- Spark SQL + DataFrames: Spark SQL enables querying structured data from inside Java-, Python-, R- and Scala-based Spark analytics applications using either SQL or the DataFrames distributed data collection
- Spark Streaming: Analysis of real-time streaming data
- GraphX: A graph analysis engine and set of graph analytics algorithms running on Spark

Apache Spark



Data Collection for LAB 3

- Collect news articles on Sports, Business, Politics and on another topic of your choice.
- Collect a suitable number of articles. Ideally above 50 articles per topic.
- Organize the documents into folders based on the topic.
- Don’t preprocess the articles while or after collecting. Preprocessing should be done only inside spark.
- This data will be split into two training and test data. Generally the ratio is about 80% of data for training and 20% testing

Importing Data into Spark

- Next task is to import the collected data into your spark program.
- sc.textFile() and sc.WholeTextFiles() are two function which you can consider when looking for ways to import the saved articles into your Spark program. A sample output after reading documents using sc.WholeTextFiles() method is

```
[(u'file:/home/pysparkbook/pysparkBookData/manyFiles/playData2.txt', u'Much
Ado About Nothing\nAs You Like It\n'),
 (u'file:/home/pysparkbook/pysparkBookData/manyFiles/playData1.txt',
u"Love's Labour's Lost\nA Midsummer Night's Dream\n")]
```

- You are free to explore ways to import the data into spark

Feature Engineering

- Feature Engineering is the task by which you find out features that differentiate between article belonging to different topics
- What do you think will be the differentiator between articles of different topics ?
- From the training set of documents you collected how can you find these ?
- What should be the characteristics of these features ?

Feature Engineering

- Yes, words are the features
- A sports article will have a different set of vocabulary compared to that of an article about technology.
- For example one can say if words like “Football”, “Soccer”, “Players” etc appear in a document, then it will mostly be a sports article.
- So the challenge is to find out these words that differentiate documents

Feature Engineering

- Document Term Matrix

We can create a database of words that appear in a set of documents.

The document-term matrix contains rows corresponding to the documents and columns corresponding to the terms. For instance if one has the following two (short) documents:

D1 = "I like databases"

D2 = "I hate databases",

then the document-term matrix would

	I	like	hate	databases
D1	1	1	0	1
D2	1	0	1	1

Feature Engineering

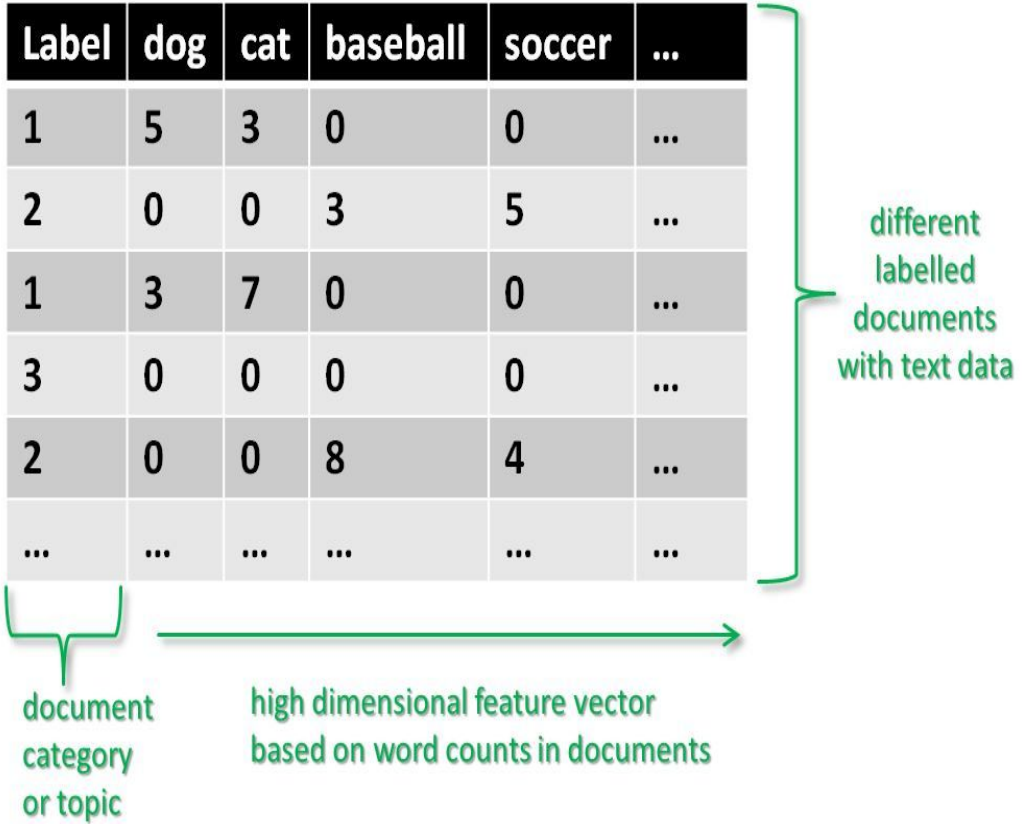
TF-IDF

- Taking just term frequency in documents might not be enough. A better approach would be to look at term frequency and inverse document frequency. This helps in finding words which are important in some documents, and doesn't appear much in documents of other topics
- **pyspark.mllib.feature import HashingTF, IDF** Will help you find out words which are important in the documents.

Feature Engineering

- Once the set of words are found out.

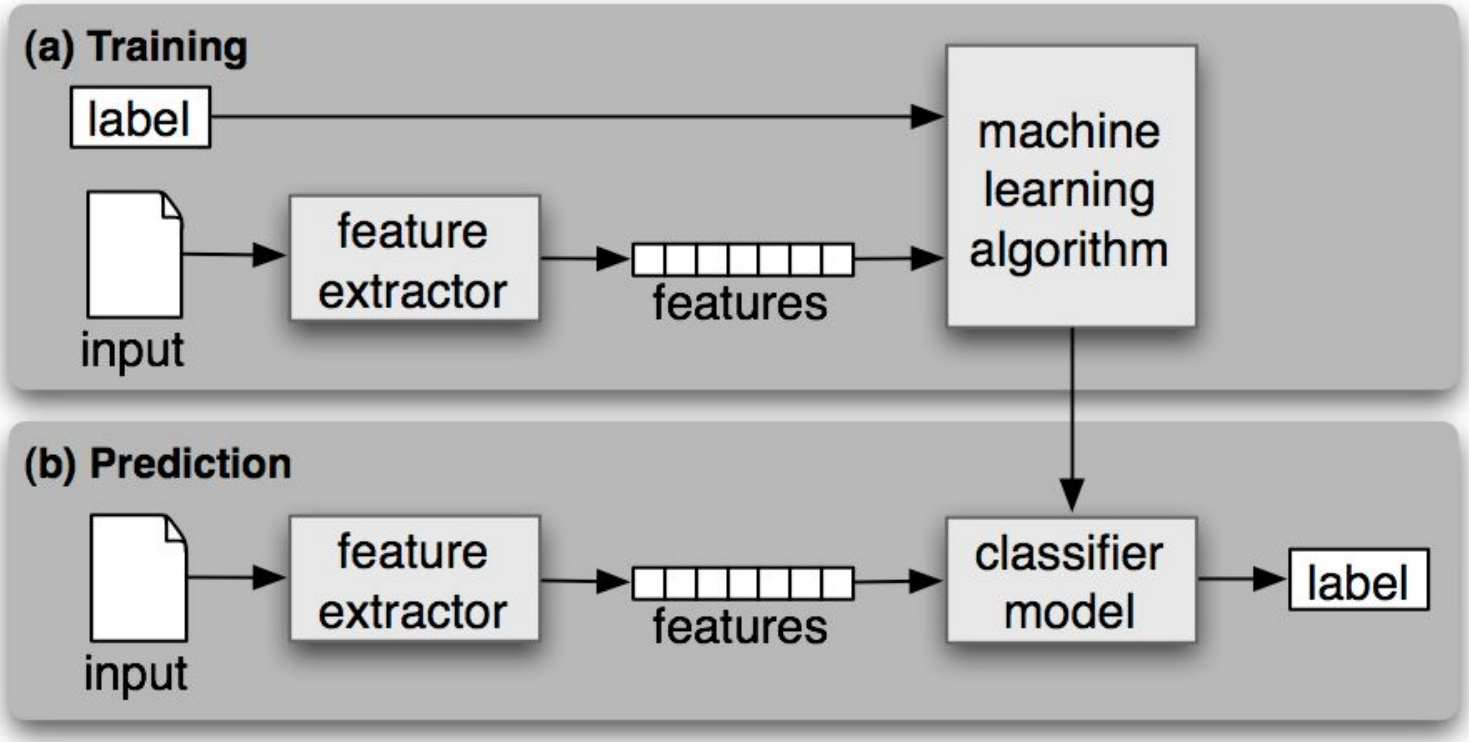
  We can make a term document matrix based on these terms for all the documents
- Each different topic will be assigned a class/label value
- And your training data will look something like this



Fitting a Model

- Once the training matrix is ready we can use any machine learning algorithm available in spark MLlib .
- Random Forest, Neural Networks, SVM etc can be used to train the system on the above data.
- Once the system is trained this model can be used to predict the classes of the 20% documents that we have saved (test data).
- The accuracy of the model should be evaluated on this 20%. If the model is not giving good accuracy. Go back and check the feature engineering methodology

Complete pipeline



Applications

- Sentiment Analysis
- Google News
- Email classification as SPAM or not
- Product matching in Retail Shops
- Chat bots

Key terms to search in Google

- Stop word removal, tokenization, stemming, lemmatization.
- Bag-of-words, TF, TF-IDF, Normalization of features
- Training and Test data split
- Spark MLlib Random Forest, SVM, Neural Network
- Text classification

Documentation

- A well documented report on what you did. This should include the following
- Assumption made, methodology used for featuring engineering
- Machine Learning models used and the accuracy obtained


ANY
QUESTIONS
?