

Dynamic Intelligent Q/A Systems

CS5560 Knowledge Discovery and Management

Project 2

Summer 2017

Team 7: TechGeeks



Team members:

- *Sai Jyothi Gudibandi*
- *Kalyan Kilaru*
- *Chaitanya Kumar Peravalli*

Project-2

1. Motivation

Nowadays, with the rapid growth in the use of internet, the information is growing more and more rapidly. To answer the user's questions, internet has become an important intermediary or a resource. The traditional search engines like google, yahoo search engine help the users to get the information they are looking for to some extent, but it sometimes returns irrelevant search results along with the relevant results. So, making a more Dynamic Intelligent Question answering system may help in information retrieval effectively.

Question-Answering system which supports natural language processing provides the users with a human-machine interface. This system be in similar type the way how the people ask questions to the search engine and it has more advantage compared to traditional search engines by understanding the purpose of the question and respond accordingly. Question-Answering system works more efficiently because it gives more appropriate information or answers to a question.

2. Objectives

The main goal of this Question and Answering System is to answer the queries posed by the humans in their normal language, using a pre-structured database or a assembly of Natural language documents. Here Question Answer system deals with the wide range of question like What, Why, When, How much, How many and Is/Are-means Yes or No type. In this system we will process the human language queries by using NLP(Natural Language Processing) and other techniques, based on the results it will automatically generate the answer for those questions.

Question Answering is an application for the knowledge base representations. Knowledge graph come under the knowledge representation.

3. Significance

Using Question Answer system searching become easy and it will give relevant answers for the questions, we can improve the document and knowledge organization. These kind of system will be very useful in case of helping desks, knowledge management systems and E-libraries.

4. Chosen Domain

For this project, we chosen Question and Answering which will be very helpful for searching and finding the correct answers to the user queries and to increase knowledge management.

5. Q/A Application

Q/A Application is a knowledge based application which will involve in so many steps to process the questions and to give the exact answer for the queries. In Q/A Application knowledge extraction is the first step to find the type of the question, POS tagging is used to determine the answer type. And TF-IDF is used for information retrieval. Name Entity Recognition is used to find corresponding “Place”, ”Date”, and “Person”. A WordNet which is a lexical dictionary is used to understand the context of the data.

Finally the Q/A Application give the answers for the questions posed by the humans in their language. This will improve the search results in that documents.

6. Data Sets

For this project we have chosen two datasets from the given data sets lists. Those two data sets are:

- BBCSport
 - <http://mlg.ucd.ie/datasets/bbc.html>
- WikiRef220
 - http://mklab.itι.gr/files/WikiRef_dataset.zip

In BBCSports, we have so many interesting topics on sports like athletics, Cricket, Football, Rugby and Tennis. Here, we mainly focused on the Cricket topic because that topic has many interesting things to know.

WikiRef220 data set have data about the Barack Obama, Financial Crisis, Elections, Airlines, Parris attacks. By this we can get to know some interesting this which are going around the world and somethings about the famous personalities.

Related Work:

In present real world, huge data leads to many data organization issues. There is a completeness and correctness problems with the existing algorithms. The solution algorithms for this problem should provide easy maintenance of large data and should give the accurate results. But there is no single approach or algorithm which satisfies those, For that we have to integrate different algorithms to design such an accurate search engines.

Searching for the required data from the large datasets is so hard. Knowledge graph gives somewhat better solution for this problem by representing the data into relations and entities. Now, we have so many knowledge graphs in the market, but Google knowledge graph became popular for searching data. By mixing different methods, knowledge graph gives the solution for the correctness and completeness problems.

To give the accurate search results for the user queries, we need to consider all the datasets which are existing in the internet. By using Knowledge vault, we can use the all data present in the internet. Knowledge vault consider the information in the form of RDD triplets, where triplets has subject, verb and predicate. After the collection of data, organizing the data is

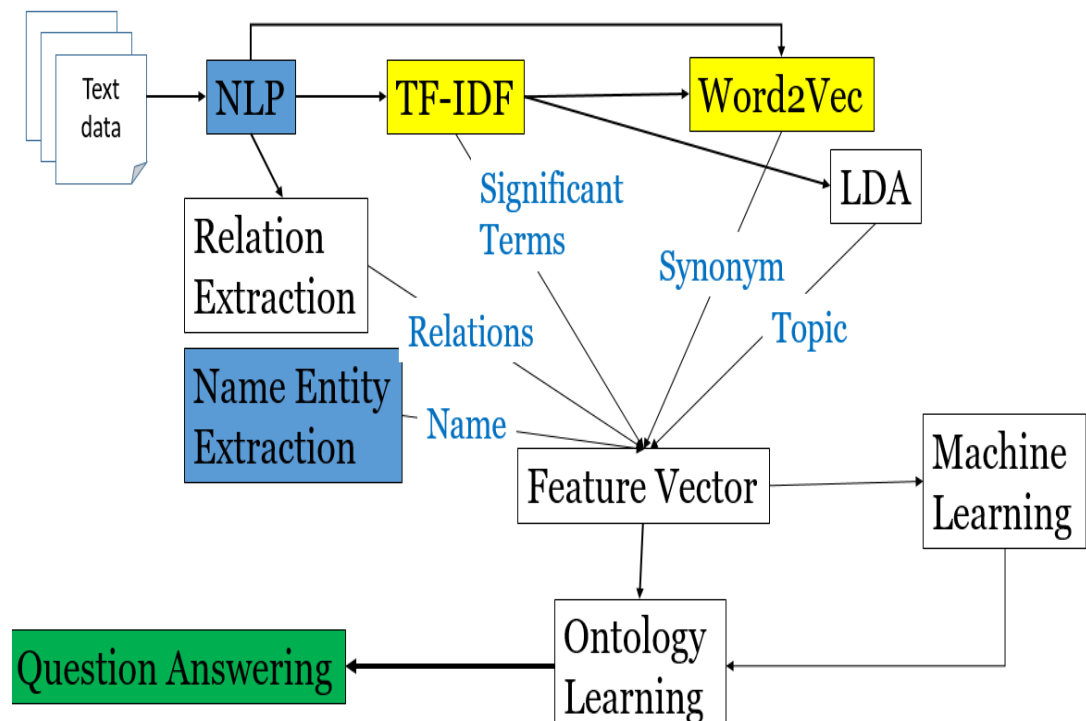
the next problem, where we have to find the relation between the entities. By using the Deep Dive concept we can get the solution for the above problem. Deep Dive helps for extracting and integrating the data in order to make the data training procedure easy.

After getting the entity relation, by using FchSen we can represent the data as RDF Triplets, where the semantic relationship between the words is organized and it combined with the related data in order to get the further simplified information.

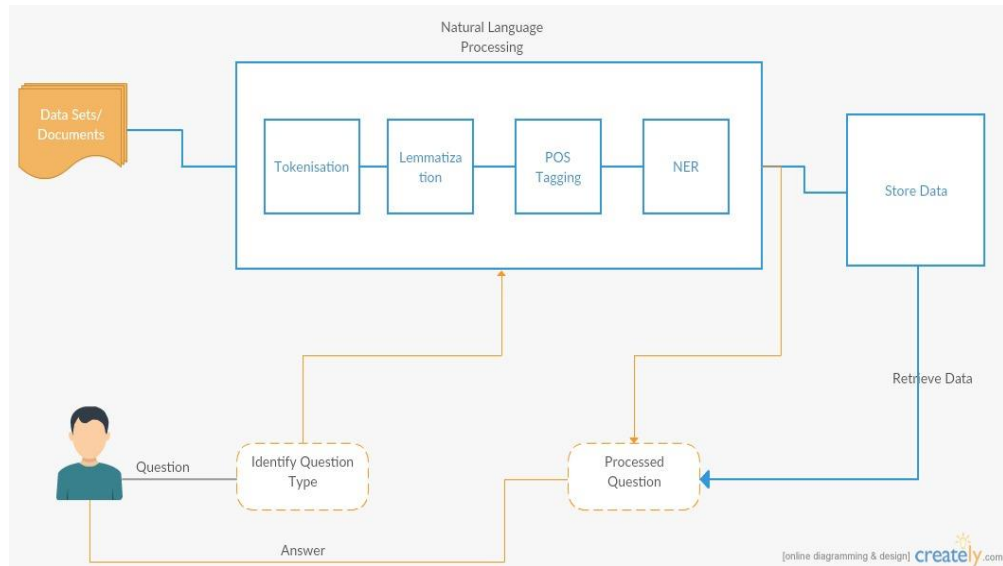
Founder is an framework for constructing the knowledge graph for richly formatted data. By using this framework we can handle the structured data and we can get the solutions for the data completeness and data correctness issues.

7. Design

a. Workflow

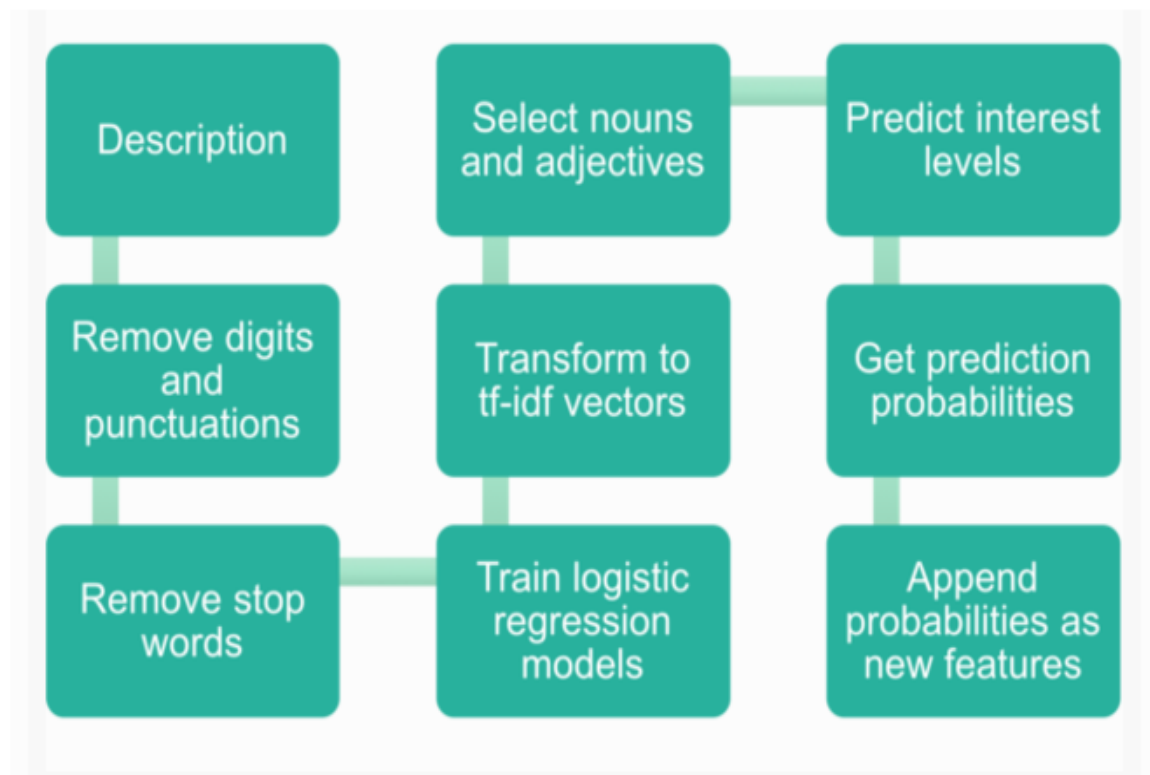


b. NLP



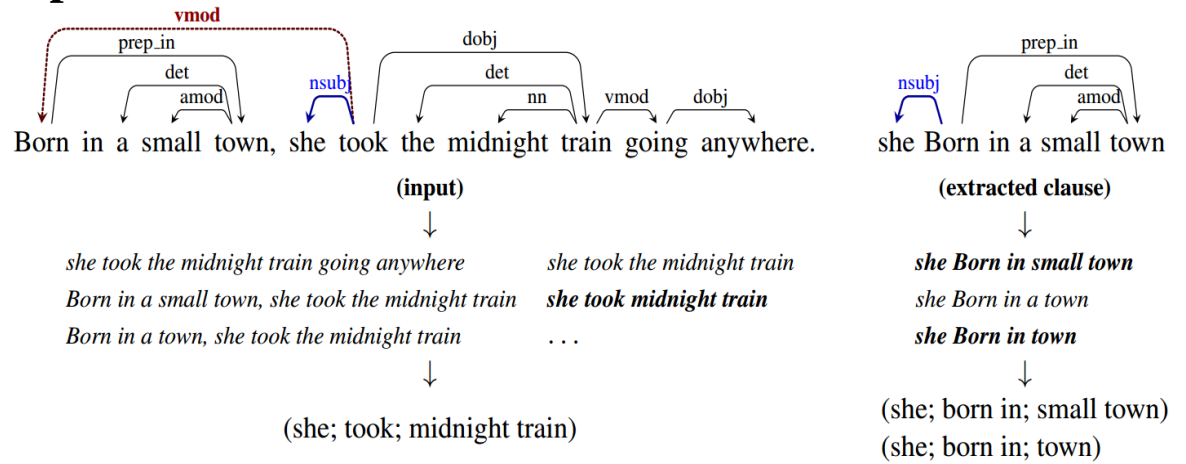
c. Information Retrieval (TFIDF, Word2Vec)

TFidf:



d. Information Extraction (OpenIE, WordNet)

OpenIE:

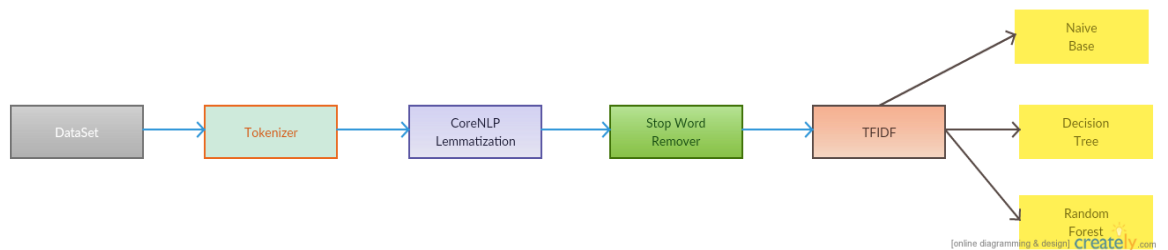


e. Machine Learning

KMeans Clustering:

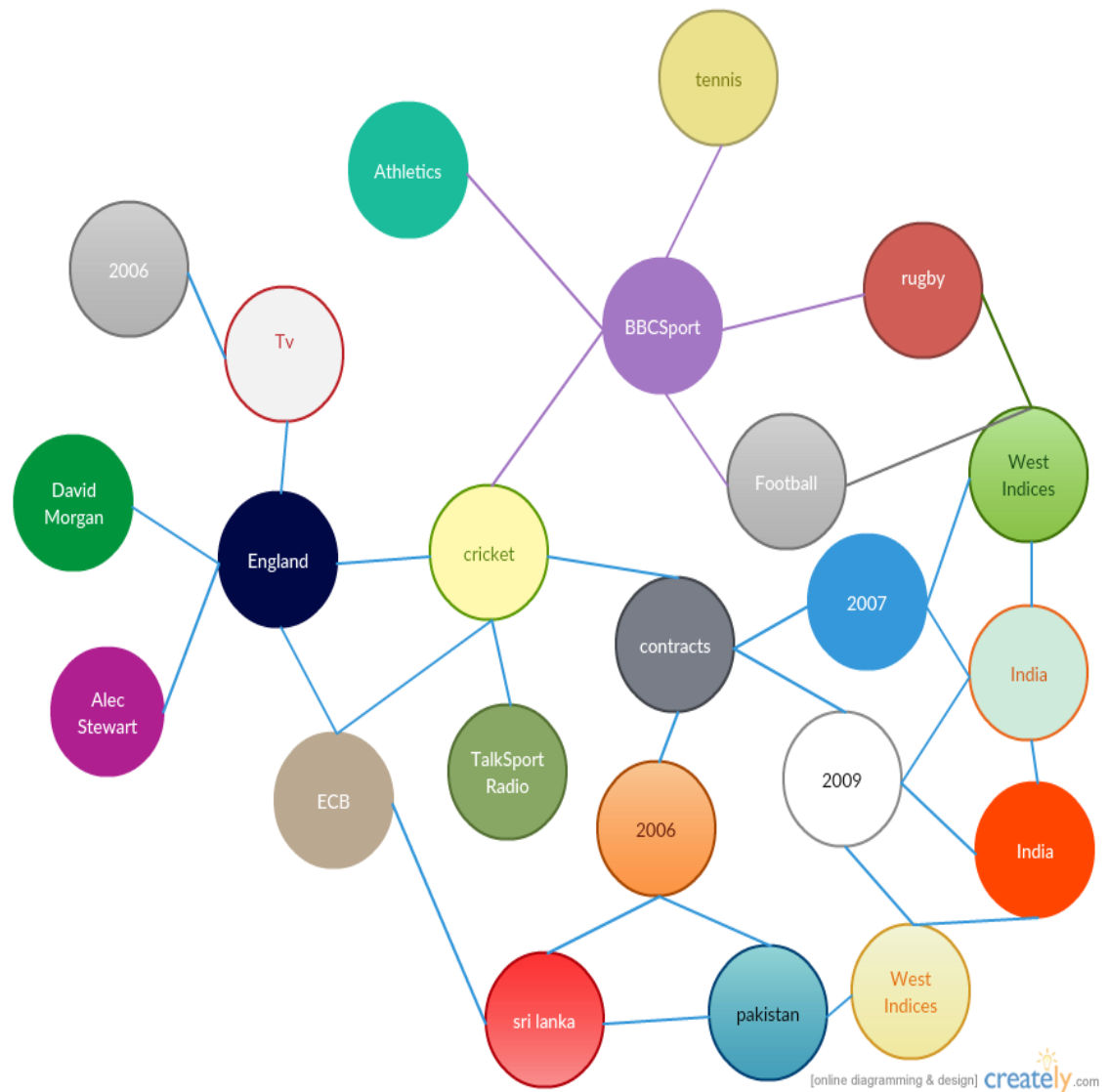


Classification(Naive Base, Decision Tree, Random Forest Algorithms) :

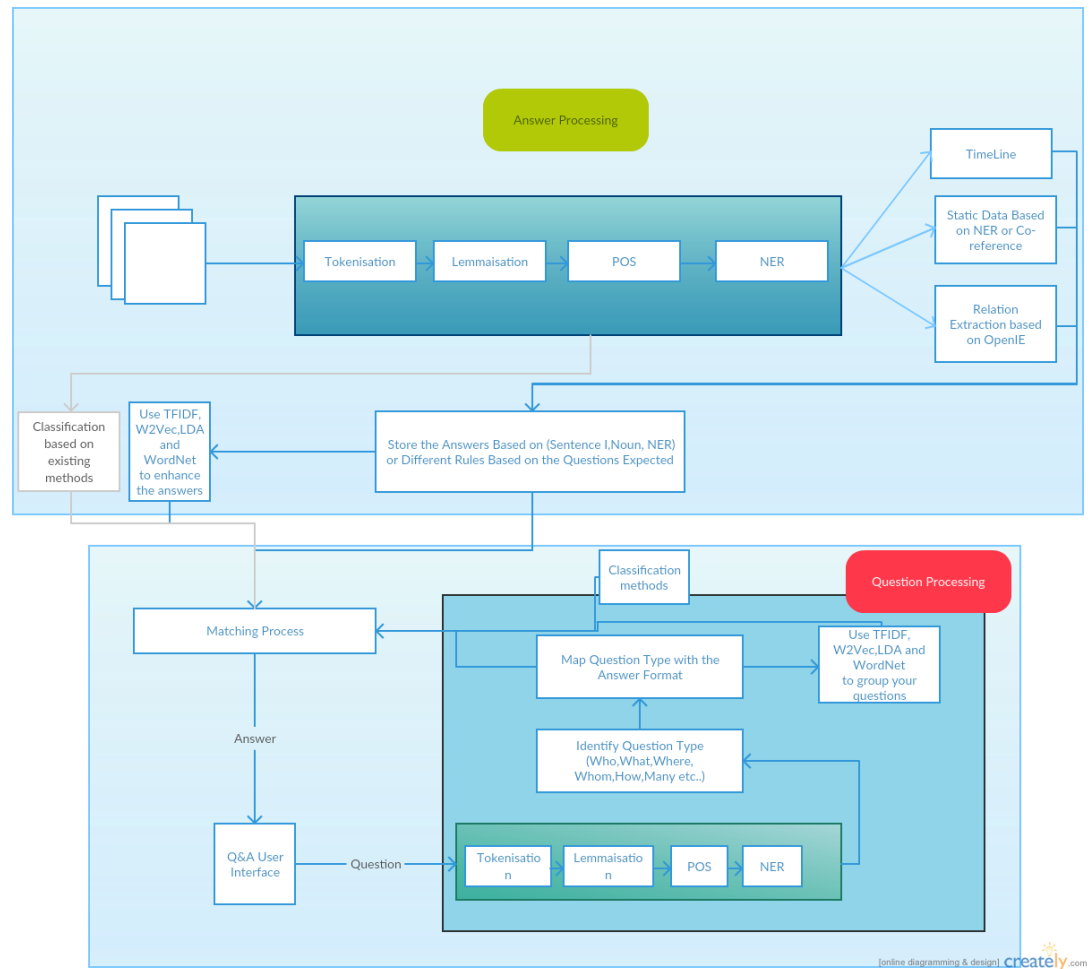


f. Knowledge Graph:

Knowledge graph doesn't have specific design, the difference between different knowledge graphs is the way they handle data like the type of data



g. Questions and Answers



8. Implementation

a. Using NLP:

NLP is a procedure which we have used for processing information given in natural language by the user. Here, NLP takes the users natural language data as input and converts that data into the machine readable format.

Here are the screen Shots of the program and execution of the NLP for the chosen dataset:

```

1  public void startSystem() {
2      if (!questionInput.equalsIgnoreCase("quit")) {
3          //System.out.println(questionInput);
4          String answer = extract(questionInput);
5          if (answer != null) {
6              System.out.println("Ans : " + answer);
7          }
8          else {
9              System.out.println("I Can't fetch answer for this question");
10         }
11     }
12     else {
13         break;
14     }
15 }

16 public void parse(String story) {
17     if (story != null && story.length() > 0) {
18         ...
19     }
20 }

```

Run: Test

```

2008-25 : CD : DATE
v-26 : LB : O
Zimbabwe-27 : NNP : LOCATION
and-28 : CC : O
South-29 : NNP : LOCATION
Africa-30 : NNP : LOCATION
--31 : : O
2009-32 : CD : DATE
v-33 : LB : O
New-34 : NNP : LOCATION
Zealand-35 : NNP : LOCATION
and-36 : CC : O
Australia-37 : NNP : LOCATION
(ORGANIZATION)[Wireless, Group, BBC, ECB, C&G, West, Wales, B&B, Radio, Talksport, Cricket, Sport, Board, Indies], LOCATION[New, Zealand, Lanka, Pakistan, Africa, England, South,
System started....

Ask questions,
Enter 'quit' to exit

```

```

92  //Map<Integer, CorefChain> graph =
93      annotation.get(CorefChainAnnotations.CorefChainAnnotation.class);
94
95      System.out.println(graph);
96
97  }
98  System.out.println(nexMap);
99
100 }
101
102 public String extract(String question) {
103     String answer = null;
104
105     parseQuestion(question);
106
107     return answer;
108 }
109

```

Run: Test

```

INFO: Read 25 rules
done [0.5 secl.
Reading input file at location:
C:\Users\chait\Desktop\069.txt
Token : POS : NER
B&B-1 : NNP : ORGANIZATION
lands-2 : NNS : O
England-3 : NNP : LOCATION
deal-4 : NN : O
Live-5 : VB : O
coverage-6 : NN : O
of-7 : IN : O
England-8 : NNP : LOCATION
's-9 : POS : O
home-10 : NN : O
Test-11 : NN : O
matches-12 : NNS : O
will-13 : MD : O
no-14 : RB : O

```

b. Information Retrieval:

Using TFIDF:

TFIDF Is nothing but the term frequency inverse document frequency, IFIDF is a mathematical measurement that is planned to reproduce how significant a word to that file in a group or corpus. The TF amount is compared to IDF count. Outcomes in matrix, by that the corpus reduced to fixed-length output.

Here is the formula for finding the Term frequency and weight of the term in that document:

(Normalized) term frequency of term k_i in Document d_j

$$tf(i, j) = \frac{freq(i, j)}{\max_{k \in T} freq(k, j)}$$

$$w_{t,d} = \begin{cases} 1 + \log_{10} tf_{t,d}, & \text{if } tf_{t,d} > 0 \\ 0, & \text{otherwise} \end{cases}$$

Here is the formula for finding the Inverse Document frequency and weight of the term in the document:

Inverse document frequency of term k_i

$$idf(i) = \log\left(\frac{n}{n_i}\right) \in [0, n]$$

n_i number of documents in which term k_i occurs

Term weight $w_{ij} = tf(i, j) idf(i)$

TFIF weight of term:

$$w_{t,d} = (1 + \log \text{tf}_{t,d}) \times \log N / \text{df}_t$$

Here are the screen Shots of the program and execution of the Information retrieval using TFIDF for the chosen dataset:

```

17/07/10 00:11:55 INFO TaskSetManager: Starting task 0.0 in stage 6.0 (PID 10, localhost, partition 0, NODE_LOCAL, 1813 bytes)
17/07/10 00:11:55 INFO Executor: Running task 0.0 in stage 6.0 (PID 10)
17/07/10 00:11:55 INFO ShuffleBlockFetcherIterator: Getting 2 non-empty blocks out of 2 blocks
17/07/10 00:11:55 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
17/07/10 00:11:55 INFO Executor: Finished task 0.0 in stage 6.0 (PID 10). 1555 bytes result sent to driver
17/07/10 00:11:55 INFO TaskSetManager: Finished task 0.0 in stage 6.0 (PID 10) in 15 ms on localhost (1/1)
17/07/10 00:11:55 INFO TaskSchedulerImpl: Removed TaskSet 6.0, whose tasks have all completed, from pool
17/07/10 00:11:55 INFO DAGScheduler: ResultStage 6 (take at TF_IDF.scala:76) finished in 0.015 s
  (to, 7.51657781092208)
  (of, 7.179452526446434)
  (with, 7.054125771490433)
  (in, 5.8377304471659395)
  (on, 5.740338101157291)
  (team, 4.702750514326955)
  (wo, 4.702750514326955)
  (time, 4.702750514326955)
  (development, 4.702750514326955)
  (package, 4.702750514326955)
17/07/10 00:12:00 INFO DAGScheduler: Job 3 finished: take at TF_IDF.scala:76, took 0.140472 s
17/07/10 00:12:00 INFO SparkContext: Invoking stop() from shutdown hook
17/07/10 00:12:00 INFO SparkUI: Stopped Spark web UI at http://192.168.1.144:4040
17/07/10 00:12:00 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
17/07/10 00:12:00 INFO MemoryStore: MemoryStore cleared
17/07/10 00:12:00 INFO BlockManager: BlockManager stopped
17/07/10 00:12:00 INFO BlockManagerMaster: BlockManagerMaster stopped
  
```

Using Word2Vec:

Word2Vec calculates scattered vector illustration of words. Main benefit of this distributed illustrations is, like words are near in vector space, which makes generalization to new patterns easier and model estimate further strong.

Based on the given input data, Word2Vec create a model. Word2Vector is comfortable with more input data. The input data is in sentence arrangement. Her, this program takes input is a text file and it will give output as the list of synonyms for a specified word. For example if we specified the word as “zero” depend on the document the we will get list of words like two, four, nine, eight including with corresponding cosine similarity.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

Here are the screen Shots of the program and execution of the Information retrieval using Word2Vec for the chosen dataset:

The screenshot displays an IDE window for a project named 'SparkOpenE_WordNET_LDA'. The main editor shows the 'WordNetSpark' class with the following Scala code:

```

1  package sparkopene
2
3  import org.apache.spark.{SparkConf, SparkContext}
4
5  object WordNetSpark {
6
7      def main(args: Array[String]): Unit = {
8          System.setProperty("hadoop.home.dir", "C://Users/saijy/Documents/HADOOP")
9          val conf = new SparkConf().setAppName("WordNetSpark").setMaster("local[*]").set("spark.driver.memory", "4g").set("spark.executor.memory", "4g")
10         val sc = new SparkContext(conf)
11         val fileWriteBuji = "Output/WordNet_loam.txt"
12         val FileWriter = new BufferedWriter(new OutputStreamWriter(new FileOutputStream(fileWriteBuji)))
13
14         val input = sc.textFile("C://Users/saijy/Downloads/kbosport/cricket/069.txt")
15
16         val dd = input.map(f => {
17             val wordnet = new RIWordNet("C://Users/saijy/Documents/WordNet-3.0")
18             val farr = f.split(" ")
19             getSynonyms(wordnet, "cricket")
20         })
21     }
22 }

```

The Run console at the bottom shows the execution logs for WordNetSpark, indicating successful compilation and execution. The logs include timestamps and messages from the TaskScheduler, TaskSetManager, Executor, HadoopRDD, DAGScheduler, and SparkUI.

```

17/07/10 07:35:37 INFO WordNetSpark: Submitting 1 waiting tasks from executor 0 = {mapreduce_shuffle} to map-as-wordnet-spark-ec2e1rj
17/07/10 07:35:37 INFO TaskSchedulerImpl: Adding task set 2.0 with 1 tasks
17/07/10 07:35:37 INFO TaskSetManager: Starting task 0.0 in stage 2.0 (TID 4, localhost, partition 0, PROCESS_LOCAL, 2076 bytes)
17/07/10 07:35:37 INFO Executor: Running task 0.0 in stage 2.0 (TID 4)
17/07/10 07:35:37 INFO HadoopRDD: Input split: file:/C:/Users/saijy/Downloads/kbosport/cricket/069.txt:0+1576
cricket
n
17/07/10 07:35:37 INFO Executor: Finished task 0.0 in stage 2.0 (TID 4). 2246 bytes result sent to driver
17/07/10 07:35:37 INFO TaskSetManager: Finished task 0.0 in stage 2.0 (TID 4) in 78 ms on localhost (1/1)
17/07/10 07:35:37 INFO TaskSchedulerImpl: Removed TaskSet 2.0, whose tasks have all completed, from pool
17/07/10 07:35:37 INFO DAGScheduler: ResultStage 2 (take at WordNetSpark.scala:20) finished in 0.078 s
17/07/10 07:35:37 INFO DAGScheduler: Job 2 finished: take at WordNetSpark.scala:20, took 0.085503 s
17/07/10 07:35:37 INFO SparkContext: Invoking stop() from shutdown hook
17/07/10 07:35:37 INFO SparkContext: Stopped Spark web UI at http://192.168.1.144:4040
17/07/10 07:35:37 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
17/07/10 07:35:37 INFO MemoryStore: MemoryStore cleared
17/07/10 07:35:37 INFO BlockManager: BlockManager stopped
17/07/10 07:35:37 INFO BlockManagerMaster: BlockManagerMaster stopped

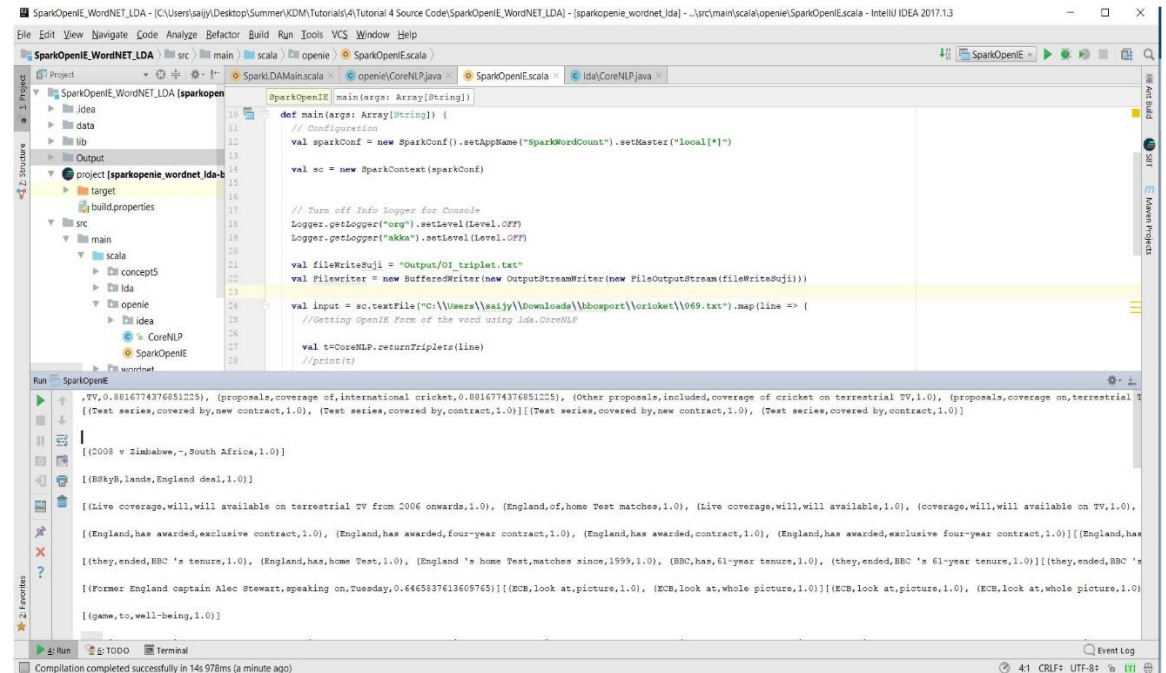
```

The status bar at the bottom indicates 'Compilation completed successfully in 2s 110ms (moments ago)' and the time is 22:37 CRLF+ UTF-8.

c. Information Extraction:

Using OpenIE:

Here are the screen Shots of the program and execution of the Information extraction using OpenIE for the chosen dataset:



The screenshot displays an IDE window for a project named 'SparkOpenIE_WordNET_LDA'. The main editor shows the 'main' method of the 'SparkOpenIE' class. The code configures a Spark application, sets the master to 'local[*]', and initializes a SparkContext. It then reads a text file '1069.txt' and processes it using the 'OpenIE' class. The output is printed to the console. The Run window at the bottom shows the execution output, which includes a list of extracted information triples.

```
def main(args: Array[String]) {  
  // Configuration  
  val sparkConf = new SparkConf().setAppName("SparkWordCount").setMaster("local[*"]  
  val sc = new SparkContext(sparkConf)  
  
  // Turn off Info Logger for Console  
  Logger.getLogger("org").setLevel(Level.OFF)  
  Logger.getLogger("akka").setLevel(Level.OFF)  
  
  val fileWriteSuij = "Output/OI_triplet.txt"  
  val FileWriter = new BufferedWriter(new OutputStreamWriter(new FileOutputStream(fileWriteSuij)))  
  
  val input = sc.textFile("C:\\Users\\saijy\\Downloads\\hadoopport\\1069.txt").map(line => {  
    //Getting OpenIE form of the word using lda.CoreNLP  
    val t=CoreNLP.returnTriplets(line)  
    //print(t)  
  })  
}
```

Run SparkOpenIE

```
TV,0.8616774376851225), (proposals,coverage of,international cricket,0.8616774376851225), (Other proposals,included,coverage of cricket on terrestrial TV,1.0), (proposals,coverage on,terrestrial TV,1.0), (Test series,covered by,new contract,1.0), (Test series,covered by,contract,1.0), (Test series,covered by,new contract,1.0), (Test series,covered by,contract,1.0)]  
  
[[[2008 v Zimbabwe,-,South Africa,1.0]]  
[[[BBKtyB,lands,England deal,1.0]]  
[[[Live coverage,will,will available on terrestrial TV from 2006 onwards,1.0), (England,of,home Test matches,1.0), (Live coverage,will,will available,1.0), (coverage,will,will available on TV,1.0),  
[[[England,has awarded,exclusive contract,1.0), (England,has awarded,four-year contract,1.0), (England,has awarded,contract,1.0), (England,has awarded,exclusive four-year contract,1.0), (England,has  
[[[they,ended,BBC 's tenure,1.0), (England,has,home Test,1.0), (England 's home Test,matches since,1999,1.0), (BBC,has,61-year tenure,1.0), (they,ended,BBC 's 61-year tenure,1.0), (they,ended,BBC 's  
[[[Former England captain Alec Stewart,speaking on,Tuesday,0.6465937613609765]], (ECB,look at,picture,1.0), (ECB,look at,whole picture,1.0)], (ECB,look at,picture,1.0), (ECB,look at,whole picture,1.0)  
[[[game,to,well-being,1.0]]
```

Using WordNet:

WordNet is a great English verbal catalogue. Nouns, verbs, adjectives and adverbs are assembled into groups of perceptive substitutes (synsets), separately stating a different idea. WordNet is implemented using RitaJS, which is a toolkit designed for experimentations in normal language. Languages used for the implementation are, a combined API for Java and JavaScript.

In this program, input is text file and we will specify a word in the program itself, then program use synsets, pos retrieval mechanisms and finally gives the output as the parts of

Here are the screen Shots of the program and execution of the Information extraction using WordNet for the chosen dataset:

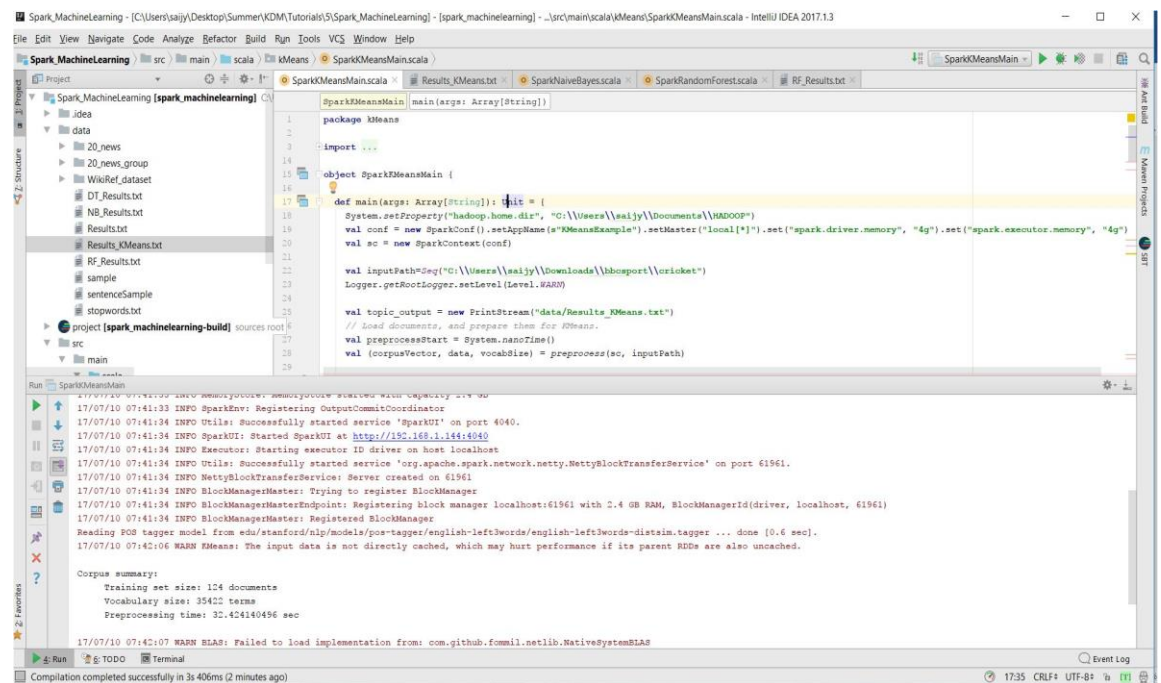


In this program we used KMeans for clustering. This K-means is most frequently used algorithm which cluster input information into a given no. of clusters. Here, k is the no. of wanted clusters. *maxIterations* is extreme no. of repetition. *initializationMode* requires any casual initialization or by k-means. *runs* is no. of times to execute that k-means process. *initializationSteps* regulates the no. of stages in k-means process. *epsilon* controls detachment edge. *initialModel* is the non-compulsory cluster centers which are used for start the process. If that constraint is provided, only single execution is done. For creating

model we used KMeans.train() method. Again we used vector concept inorder to check to which cluster that goes.

The questions from users will be categories and store based on topic/question along with processed answer.... kind etc....this helps in faster retrieval of answers in future. This can also be used to suggest recommended questions to the user based on the current query.

Here are the screen Shots of the program and execution of the Information extraction using WordNet for the chosen dataset:



```
package kMeans
import ...

object SparkKMeansMain {
  def main(args: Array[String]): Unit = {
    System.setProperty("hadoop.home.dir", "C:\\Users\\saiky\\Documents\\HADOOP")
    val conf = new SparkConf().setAppName(s"KMeansExample").setMaster("local[*]").set("spark.driver.memory", "4g").set("spark.executor.memory", "4g")
    val sc = new SparkContext(conf)

    val inputPath = Seq("C:\\Users\\saiky\\Downloads\\khsport\\cricket")
    Logger.getLogger().setLevel(Level.WARN)

    val topic_output = new PrintStream("data/Results_KMeans.txt")
    // load documents, and prepare them for KMeans.
    val preprocessStart = System.nanoTime()
    val (corpusVector, data, vocabSize) = preprocess(sc, inputPath)
  }
}
```

Run SparkKMeansMain

```
17/07/10 07:41:33 INFO SparkEnv: Registering OutputCommitCoordinator
17/07/10 07:41:34 INFO Utils: Successfully started service 'SparkUI' on port 4040.
17/07/10 07:41:34 INFO SparkUI: Started SparkUI at http://192.168.1.144:4040
17/07/10 07:41:34 INFO Executor: Starting executor ID driver on host localhost
17/07/10 07:41:34 INFO Utils: Successfully started service 'org.apache.spark.network.netty.NettyBlockTransferService' on port 61961.
17/07/10 07:41:34 INFO NettyBlockTransferService: Server created on 61961
17/07/10 07:41:34 INFO BlockManagerMaster: Trying to register BlockManager
17/07/10 07:41:34 INFO BlockManagerMasterEndpoint: Registering block manager localhost:61961 with 2.4 GB RAM, BlockManagerId(driver, localhost, 61961)
17/07/10 07:41:34 INFO BlockManagerMaster: Registered BlockManager
Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distsim.tagger ... done [0.6 sec].
17/07/10 07:42:06 WARN BLAS: The input data is not directly cached, which may hurt performance if its parent RDDs are also uncached.

Corpus summary:
Training set size: 124 documents
Vocabulary size: 35422 terms
Preprocessing time: 32.42410496 sec

17/07/10 07:42:07 WARN BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeSystemBLAS

Compilation completed successfully in 3s 406ms (2 minutes ago)
```

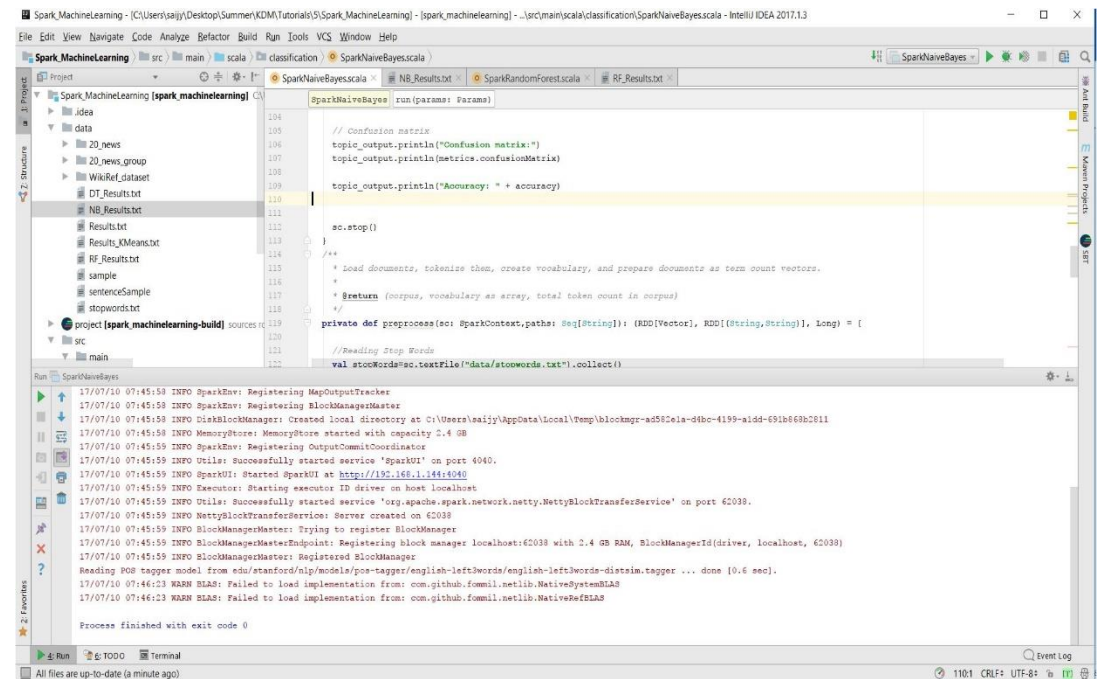
Classification:

Classification is nothing but a problem of classifying the observations, to which category it goes, on the base of the training dataset.

Here we used Decision Tree, Naïve Bays classifier and Random Forest Classification algorithms.

Here are the screen Shots of the program and execution of the Machine Learning clustering and classification for the chosen dataset:

Naïve Base:

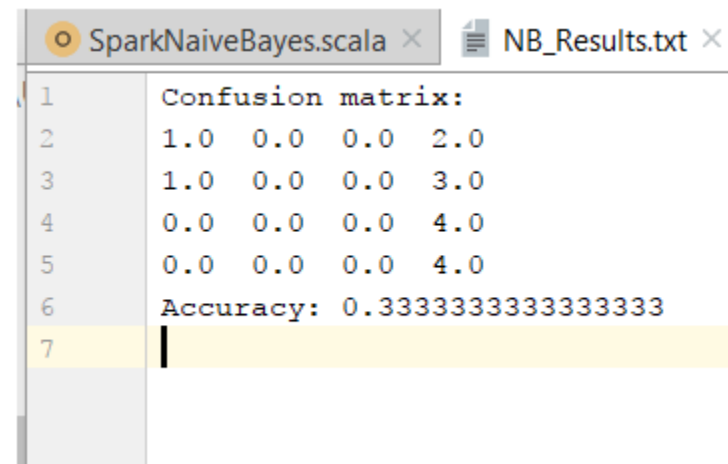


The screenshot shows an IDE window titled "Spark_MachineLearning - [C:\Users\saaj\Desktop\Summer\KDM\Tutorials\Spark_MachineLearning] - [src\main\scala\classification\SparkNaiveBayes.scala - IntelliJ IDEA 2017.1.3]". The editor displays the `SparkNaiveBayes.scala` file with the following code:

```
104 // Confusion matrix
105 topic_output.println("Confusion matrix:")
106 topic_output.println(metrics.confusionMatrix)
107
108 topic_output.println("Accuracy: " + accuracy)
109
110
111
112 sc.stop()
113 }
114 /**
115  * Load documents, tokenize them, create vocabulary, and prepare documents as term count vectors.
116  *
117  * @return (corpus, vocabulary as array, total token count in corpus)
118  */
119 private def preprocess(sc: SparkContext, paths: Seq[String]): (RDD[Vector], RDD[(String, String)], Long) = {
120 //Loading Stop Words
121 val stopWords = sc.textFile("data/stopwords.txt").collect()
122 }
```

The console output shows the execution of the program, including Spark environment setup, block manager registration, and the final output:

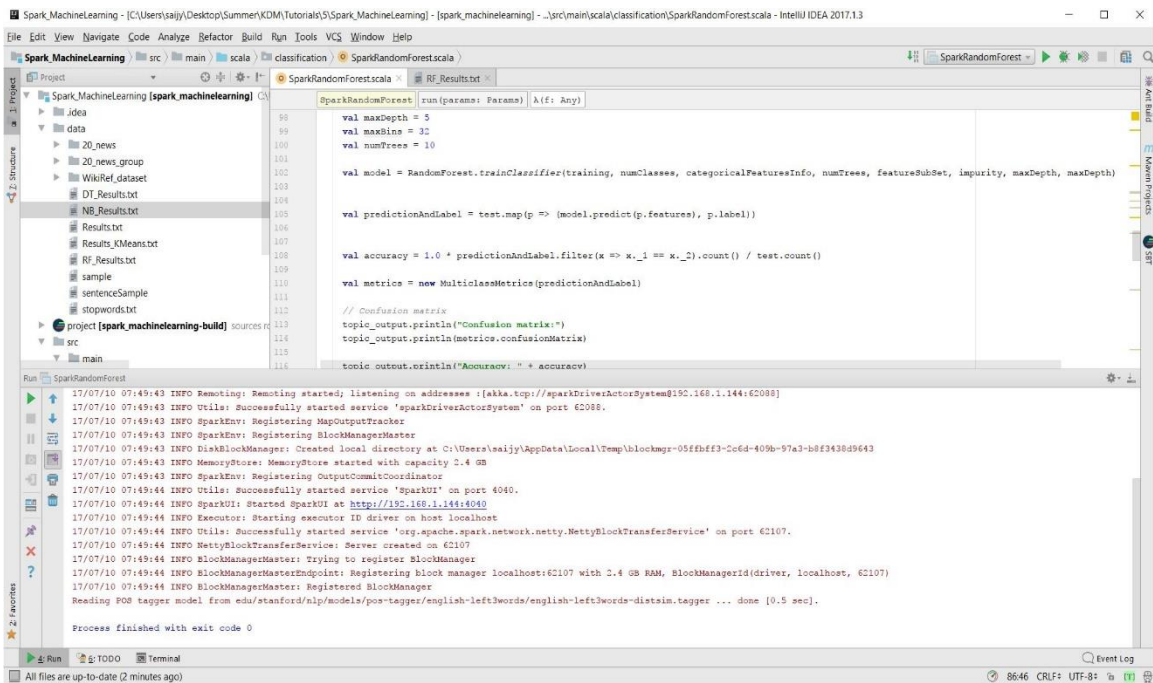
```
17/07/10 07:45:58 INFO SparkEnv: Registering MapOutputTracker
17/07/10 07:45:58 INFO SparkEnv: Registering BlockManagerMaster
17/07/10 07:45:58 INFO DiskBlockManager: Created local directory at c:\Users\saaj\AppData\Local\Temp\blockmgr-ad582c1a-8fbc-4199-a1dd-691b669b2611
17/07/10 07:45:58 INFO MemoryStore: MemoryStore started with capacity 2.4 GB
17/07/10 07:45:59 INFO SparkEnv: Registering OutputCommitCoordinator
17/07/10 07:45:59 INFO Utils: Successfully started service 'SparkUI' on port 4040.
17/07/10 07:45:59 INFO SparkUI: Started SparkUI at http://192.168.1.144:4040
17/07/10 07:45:59 INFO Executor: Starting executor ID driver on host localhost
17/07/10 07:45:59 INFO Utils: Successfully started service 'org.apache.spark.network.netty.NettyBlockTransferService' on port 62038.
17/07/10 07:45:59 INFO NettyBlockTransferService: Server created on 62038
17/07/10 07:45:59 INFO BlockManagerMaster: Trying to register BlockManager
17/07/10 07:45:59 INFO BlockManagerMasterEndpoint: Registering block manager localhost:62038 with 2.4 GB RAM, BlockManagerId(driver, localhost, 62038)
17/07/10 07:45:59 INFO BlockManagerMaster: Registered BlockManager
17/07/10 07:45:59 INFO BlockManager: Registered BlockManager
17/07/10 07:46:23 WARN ELAS: Failed to load implementation from: com.github.fommil.netlib.NativeSystemELAS
17/07/10 07:46:23 WARN ELAS: Failed to load implementation from: com.github.fommil.netlib.NativeRefELAS
Process finished with exit code 0
```



The screenshot shows the `NB_Results.txt` file with the following output:

```
1 Confusion matrix:
2 1.0 0.0 0.0 2.0
3 1.0 0.0 0.0 3.0
4 0.0 0.0 0.0 4.0
5 0.0 0.0 0.0 4.0
6 Accuracy: 0.3333333333333333
7
```

Random Forest:



SparkRandomForest.scala × RF_Results.txt ×	
1	Confusion matrix:
2	2.0 0.0 1.0 0.0
3	2.0 0.0 1.0 1.0
4	1.0 1.0 1.0 1.0
5	0.0 0.0 0.0 4.0
6	Accuracy: 0.4666666666666667
7	

Decision Tree:

```
object SparkDecisionTree {
  main(args: Array[String])

  private case class Params(
    input: Seq[String] = Seq.empty
  )

  def main(args: Array[String]) {
    val defaultParams = Params()

    val parser = new OptionParser(Params) {
      head("XMeansExample: an example XMeans app for plain text data.")
      arg[String]("input") { "input" }
      .text("input paths (directories) to plain text corpora." +
        " Each text file line should hold 1 document.")
      .unbounded()
      .required()
      .action((x, c) => c.copy(input = c.input :+ x))
    }

    parser.parse(args, defaultParams)
  }
}
```

Run SparkDecisionTree

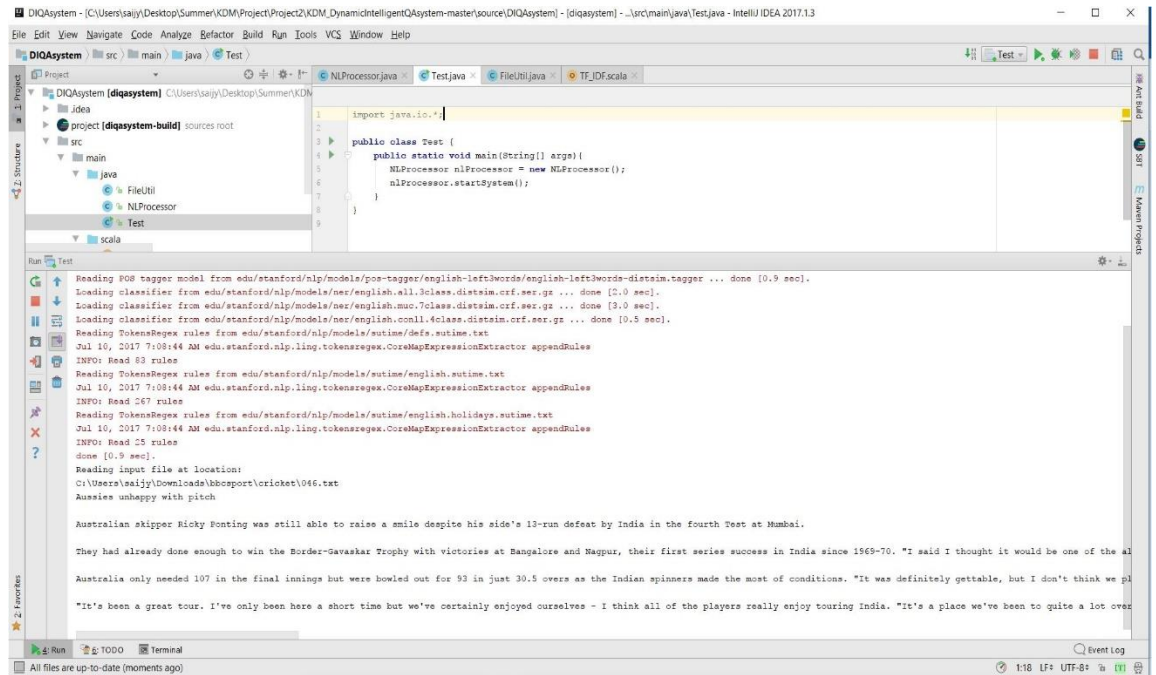
```
17/07/10 07:57:12 INFO Utils: Successfully started service 'sparkDriverActorSystem' on port 62350.
17/07/10 07:57:12 INFO SparkEnv: Registering MapOutputTracker
17/07/10 07:57:12 INFO SparkEnv: Registering BlockManagerMaster
17/07/10 07:57:12 INFO DiskBlockManager: Created local directory at C:\Users\saajy\AppData\Local\Temp\blockmgr-45eb8c6f-635d-4667-a44d-ec26a9e93d9
17/07/10 07:57:12 INFO MemoryStore: MemoryStore started with capacity 2.4 GB
17/07/10 07:57:12 INFO SparkEnv: Registering OutputCommitCoordinator
17/07/10 07:57:12 INFO Utils: Successfully started service 'SparkUI' on port 4040.
17/07/10 07:57:13 INFO SparkUI: Started SparkUI at http://192.168.1.144:4040
17/07/10 07:57:13 INFO Executor: Starting executor ID driver on host localhost
17/07/10 07:57:13 INFO Utils: Successfully started service 'org.apache.spark.network.netty.NettyBlockTransferService' on port 62411.
17/07/10 07:57:13 INFO NettyBlockTransferService: Server created on 62411
17/07/10 07:57:13 INFO BlockManagerMaster: Trying to register BlockManager
17/07/10 07:57:13 INFO BlockManagerMasterEndpoint: Registering block manager localhost:62411 with 2.4 GB RAM, BlockManagerId(driver, localhost, 62411)
17/07/10 07:57:13 INFO BlockManagerMaster: Registered BlockManager
17/07/10 07:57:13 INFO POS tagger model from edu.stanford.nlp/models/pos-tagger/english-left3words/english-left3words-distsim.tagger ... done [0.6 sec].
17/07/10 07:57:36 WARN DecisionTreeMetadata: DecisionTree reducing maxBins from 32 to 25 (= number of training instances)

Process finished with exit code 0
```

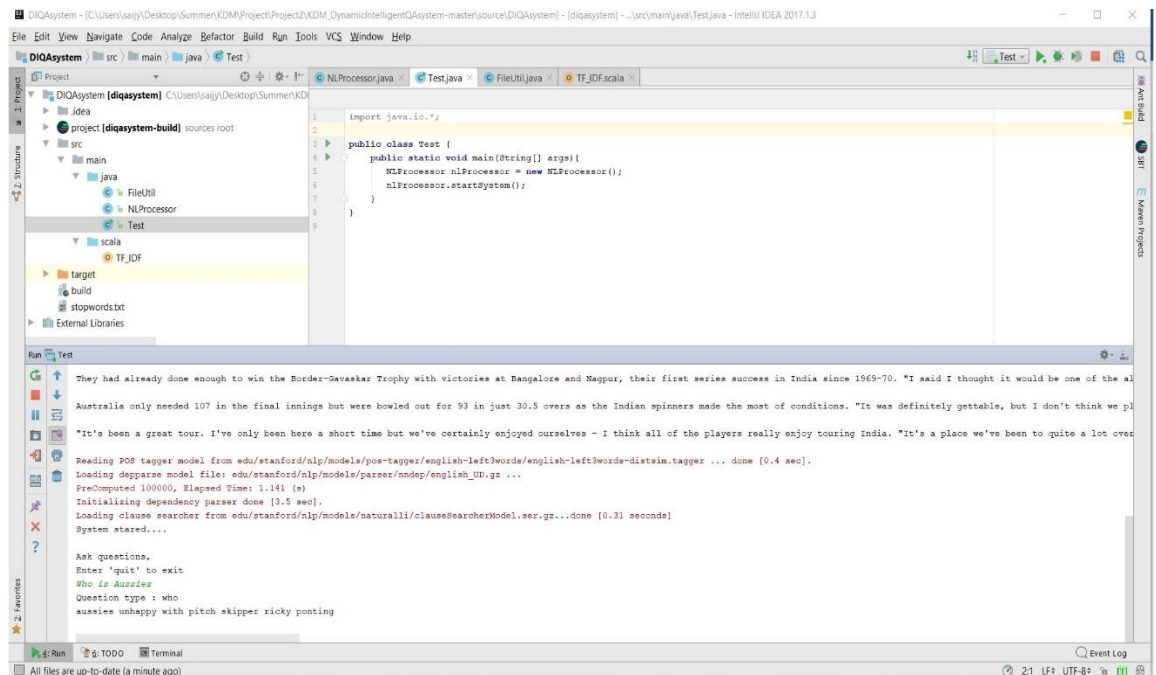
	SparkDecisionTree.scala	DT_Results.txt
1		Confusion matrix:
2		1.0 2.0 0.0 0.0
3		3.0 0.0 1.0 0.0
4		1.0 1.0 1.0 1.0
5		0.0 0.0 0.0 4.0
6		Accuracy: 0.4
7		

e. Question Answering:

Here are the screen Shots of the program and execution of the Question Answering for the chosen dataset:



```
DIQASystem - [C:\Users\saij\Desktop\Summer\KDM\Project\Project2\KDM_DynamicIntelligentQASystem-master\source\DIQASystem] - [diqasystem] - ...src\main\java\Test.java - IntelliJ IDEA 2017.1.3
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
DIQASystem [diqasystem] C:\Users\saij\Desktop\Summer\KDM\Project\Project2\KDM_DynamicIntelligentQASystem-master\source\DIQASystem
src main java Test
NLProcessor.java Test.java FileUtil.java TF_IDF.scala
import java.io.*;
public class Test {
    public static void main(String[] args){
        NLProcessor nlProcessor = new NLProcessor();
        nlProcessor.startSystem();
    }
}
Run Test
Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distisim.tagger ... done [0.9 sec].
Loading classifier from edu/stanford/nlp/models/nec/english.all3class.distisim.crf.ser.gz ... done [2.0 sec].
Loading classifier from edu/stanford/nlp/models/nec/english.muc.7class.distisim.crf.ser.gz ... done [3.0 sec].
Loading classifier from edu/stanford/nlp/models/nec/english.conll.4class.distisim.crf.ser.gz ... done [0.5 sec].
Reading TokensRegex rules from edu/stanford/nlp/models/sutime/defs.sutime.txt
INFO: Read 63 rules
Reading TokensRegex rules from edu/stanford/nlp/models/sutime/english.sutime.txt
INFO: Read 267 rules
Reading TokensRegex rules from edu/stanford/nlp/models/sutime/english.holidays.sutime.txt
INFO: Read 25 rules
done [0.9 sec]
Reading input file at location:
C:\Users\saij\Downloads\bbsport\cricket\046.txt
Aussies unhappy with pitch
Australian skipper Ricky Ponting was still able to raise a smile despite his side's 13-run defeat by India in the fourth Test at Mumbai.
They had already done enough to win the Border-Gavaskar Trophy with victories at Bangalore and Nagpur, their first series success in India since 1969-70. "I said I thought it would be one of the al
Australia only needed 107 in the final innings but were bowled out for 93 in just 39.5 overs as the Indian spinners made the most of conditions. "It was definitely gettable, but I don't think we pl
"It's been a great tour. I've only been here a short time but we've certainly enjoyed ourselves - I think all of the players really enjoy touring India. "It's a place we've been to quite a lot over
All files are up-to-date (moments ago) 1:18 LF: UTF-8
```



```
DIQASystem - [C:\Users\saij\Desktop\Summer\KDM\Project\Project2\KDM_DynamicIntelligentQASystem-master\source\DIQASystem] - [diqasystem] - ...src\main\java\Test.java - IntelliJ IDEA 2017.1.3
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
DIQASystem [diqasystem] C:\Users\saij\Desktop\Summer\KDM\Project\Project2\KDM_DynamicIntelligentQASystem-master\source\DIQASystem
src main java Test
NLProcessor.java Test.java FileUtil.java TF_IDF.scala
import java.io.*;
public class Test {
    public static void main(String[] args){
        NLProcessor nlProcessor = new NLProcessor();
        nlProcessor.startSystem();
    }
}
Run Test
They had already done enough to win the Border-Gavaskar Trophy with victories at Bangalore and Nagpur, their first series success in India since 1969-70. "I said I thought it would be one of the al
Australia only needed 107 in the final innings but were bowled out for 93 in just 39.5 overs as the Indian spinners made the most of conditions. "It was definitely gettable, but I don't think we pl
"It's been a great tour. I've only been here a short time but we've certainly enjoyed ourselves - I think all of the players really enjoy touring India. "It's a place we've been to quite a lot over
Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distisim.tagger ... done [0.4 sec].
Loading depparse model file: edu/stanford/nlp/models/parser/nndep/english_UD.gz ...
PreComputed 100000, Elapsed Time: 1.141 (s)
Initializing dependency parser done [3.5 sec].
Loading clause searcher from edu/stanford/nlp/models/natural11/clauseSearcherModel.ser.gz...done [0.31 seconds]
System started....
Ask questions.
Enter 'quit' to exit
Who is Aussie?
Question type : who
aussies unhappy with pitch skipper ricky ponting
All files are up-to-date (a minute ago) 2:1 LF: UTF-8
```

9. Project Management

a. Work Completed

Work completed on NLP, TFIDF, Word2Vec, OpenIE, WordNet, Clustering, Classification, Question Answer system for the chosen dataset.

b. Contributions

SaiJyothi Gudibandi

Kalyan Kilaru

Chaitanya Kumar Peravalli

33.33 % ■
Kalyan Kilaru

33.33 % ■
Chaitanya Peravalli

33.34 % ■
Saijyothi Gudibandi

Contributions



c. Statics

Here are the screen Shots Project Management:

NLP Design & Implementation

Start: Jul 4, 2017 [Change](#) Due: Jul 10, 2017 [Change](#)

[Edit Milestone](#) [NLP Design & I...](#)

[Labels](#) [Hide Pull Requests](#)

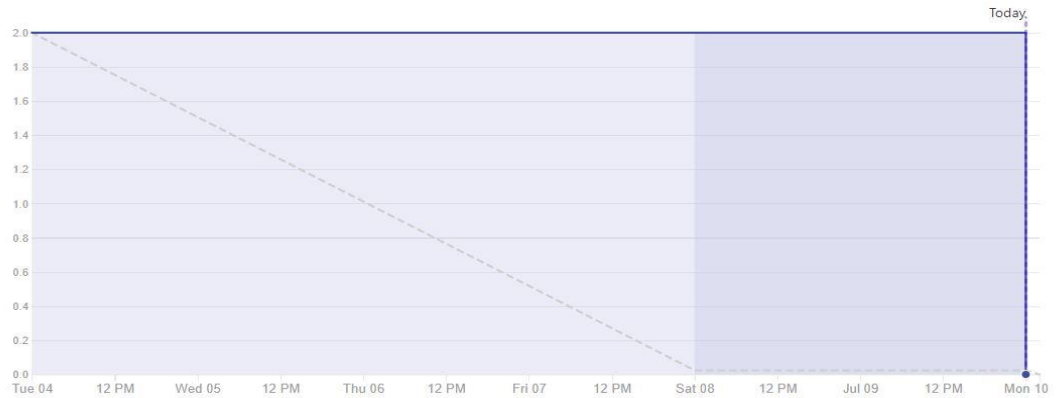
[Burn Pipelines](#)

Burndown report

☐ Weekends

☐ Ideal

☐ Completed



Knowledge Graph & Questions and Answers

Start: **Jul 6, 2017** [Change](#) Due: **Jul 10, 2017** [Change](#)

[Edit Milestone](#) [Knowledge Gra...](#)

[Labels](#) [Hide Pull Requests](#)

[Burn Pipelines](#)

Burndown report

☒ Weekends

☐ Ideal

☒ Completed



4 Total Story Points

4 Completed / 0 Remaining

2 Total Issues and Pull Requests

2 Completed / 0 Remaining

Information Retrieval, Extraction & Machine Learning

Start: **Jul 5, 2017** [Change](#) Due: **Jul 10, 2017** [Change](#)

[Edit Milestone](#) [Information Ret...](#)

[Labels](#) [Hide Pull Requests](#)

[Burn Pipelines](#)

Burndown report

☒ Weekends

☐ Ideal

☒ Completed



7 Total Story Points

7 Completed / 0 Remaining

3 Total Issues and Pull Requests

3 Completed / 0 Remaining

d. Concerns/Issues

NA

e. Future Work

Focus on implementation of the Question Answer System to build a best system which will give the better answers for the user queries.