

CS5560 Knowledge Discovery and Management

Project Report - 2

Dynamic Question Answering System using Natural Language Processing

Team 2

Mudunuri, Sai Sarat Chandra Varma (14)

Pabolu, Sandeep (19)

Pallepati, Rakesh Reddy (20)

Surparaju, Lava Kumar (27)

Motivation:

On the onset, man is after the quest for the knowledge. This thrust led us to the invention of lot many things including the most precious computers. This thrust starts with some questions like” Why?”,” What”,” Who” and so on. This finally led the working on question answering system.

Objective:

The main objective of the project is to set up a code which generates approximate answers for user’s quest based on the dataset. This sharing of ideas, experience and information should be available in the right place at the right time thereby reducing the user’s wait time.

Significance:

The major significances can be listed out as under:

- It increases the interactivity of the user with the database.
- Searched answers can be viewed frequently.
- Sentiment analysis can be performed for the better understanding.
- A proper usage of the linguistic resources.

Related Work - Journal Club:**1. Knowledge Vault :**

Knowledge Vault will use the algorithm at mass-scale to form a single base of facts with the combination of all information including structured, unstructured and semi structured data. From the web. It is as effective as knowledge graph with organized information from the web.

Knowledge vault is larger than the knowledge base. It uses multiple extraction from different types of sources and various systems. Components used in the Knowledge graph are as follows: Extractor, Graph-base prior and knowledge fusion. It has various types of extraction methods such as txt ,HTML tree(DOM),HTML Tables, Human Annotated Pages. For the link predictions, they have used the Path Ranking Algorithm(PRA), Neural Network Model(NNM).

These are used in a way whether they have a set of existing edges and will predict which other edges are having chance to exist. PRA is used for set-of-pairs of various entities that are connected to each other. This will perform the random walks from beginning of source nodes. It is formed in the form of pairs. Neural network will view the link prediction problem compares to the matrix form.

Local Closed World Assumption(LCWA) will be used to approximate the truth. One of the sources for Knowledge vault will be the Freebase where most of the information is extracted from it whereas the other data sources will be YAGO and DBpedia. It uses the fixed ontology-schemes such as Deepdive,NELL,Prospera and Readtheweb. Probbase is a constructor used here for hierarchies. The basic difference between Knowledge vault and present approach is here, we combine facts extracted from data with the previous knowledge graph. It can have the uncertainty from the facts it has derived.

Knowledge vault can be improvised by the mutual exclusion from different tweets and the correlation between the facts. The values can be represented from the multi-level abstraction, Dealing from the correlated data sources. There will be temporary facts which are true for only certain limits. It can be improved by adding various entities and relations in the graph.

2. Knowledge Graph Refinement:

The main objective of this paper is about the how to refine the existing Knowledge graph which has been constructed using some methods. There are numerous number of ways for building a knowledge graph or an ontology using the existing techniques. The base idea of building a knowledge graph is to have a good understanding of data. In the current trend, there is lots of amount of data that has been generated in audio, video and other domains. Handling of large amount of data and information available is a challenging task in the IT industry. Building a good Knowledge graph with the existing data and information is an effective way of understanding it.

This paper discusses about a survey which has been conducted on different approaches and evaluation methods that has been used for Knowledge graph refinement. Knowledge graph is the base for the IT industry which describes the domain specific knowledge. Google is the first tech company which has built its knowledge graph for their search engine. Following this announcement, a lot of

other companies started building their knowledge graphs. Some good examples of applications that are built based on Knowledge graphs are Siri, Google search engine, Amazon Echo and few others. The efficiency of the systems that are built using the Knowledge graphs is not so accurate.

Knowledge graph or an Ontology is a related graph which relates the real world entities along with the connections which describes the relationships between those entities. There are lot of different schemas for building the Knowledge graphs. Google first used the term Knowledge graph for the prototype that they have developed for their web search. The Knowledge graphs covers various domains and is not specific to any domain. There are lot of existing databases available in the data domain, some being Cyc and OpenCyc, Freebase, DBpedia, Wikidata, Wikipedia, YAGO, NELL, Google Knowledge graph and Knowledge Vault, Microsoft Satori, Yahoo's Knowledge graph and Facebook entities graph.

This paper describes lot of approaches for Knowledge graph refinement. Some of the evaluation methods that are being discussed include Partial Gold Standard, Silver Standard and Retrospective approaches. The main idea of this evaluation methods are to find out how accurate the Knowledge graph is, as we know its not so perfect. The Partial gold standard approach consumes lot of energy to produce but the results are good enough and can be reused for other approaches also. The Silver standard method yields less reliable results. Retrospective evaluation yields better reusable results when compared to other approaches and significant. There are lot of other approaches that has been discussed in the paper about the completeness of the Knowledge graph. None of the existing methods will not tell about the completeness of the Knowledge graph accurately as the amount of data that has been utilized to build the Knowledge graph is huge. It also discussed the error detection techniques that are used to evaluate the Knowledge graph.

The paper concludes the following results: A Single approach cannot tell us about the completeness and correctness of the Knowledge graph, It discusses about the error detection and error correction techniques for the Knowledge graphs, it mostly describes about the Knowledge graph refinement than building a Knowledge graph, it discusses and evaluates the different Knowledge graph techniques.

3. Deep Dive Declarative Knowledge Base Construction:

The paper was basically introduced to tackle with the current problem being faced in handling the humungous amount of data everywhere. The key feature where the focus has been made is the deep dive. What it does is it organizes the unstructured data into structured and well-organized form and identify the relation between them. On the onset, deep dive resembles a trained system which with the help of the machine learning tackles with the noise and the imprecision, this makes the work of easy for the end users and the representation of the data is more accurate.

Moreover, in the naming and stuff of different entities mistakes are expected by the humans but this can be easily being overcome by this. Deep dive is well fond of the fact that the data here is a mixture of noise and imprecise. It checks that if the probability of the data is nine parts in ten then it will very much be trending to assume that the selected fact is accurate.

Paleontology is related to the explanation of the biological classification related to the fossils. The same feature is currently being done by the humans which is a more expensive and more wastage of the time. Many tests should be done to assess the quality. In some of the cases the data is hidden not in the text format but in the pictorial or the tabular format in such cases the exact extraction of the data with the related joint is a difficult task. For solving this the KBC model was introduced consisting of the entities, relation, mention, relation mention. Here the feature extractors play a very important role in contribution to the candidates. Thus, using the deep dive, we can connect evidence for any relation from the training data.

It can use the training data for any relation indicating whether the particular entity is relationally true or false. In the inference phase, generally the factor graph is generated. The application is like in the error analysis where the major focus plays around the segregation of the common mistakes made by the humans. It has an added advantage of adding an additional probability for each entity relation search. The deep diving has an additional feature of the feature extraction which increases the data searches and takes the search to a new level of search.

The efficiency of deep dive is also noticeable fact. There are two types of efficiencies that are possible with the deep dive it is hardware and statistical. Thus on the whole the paper provides us with a clear understanding regarding the grounding and also the statistical interfacing in learning. Moreover, the decisions

made in the deepdive program provide some unambiguous design choices. They have inaugurated the KBC and the main console's framework in a very straightforward way. There are some places where the authors have gone off the grounds for example the assumption made that the readers have all the knowledge regarding the fields where the application is possible. Similarly, lot many technical terms are unexplained and are left clueless for the readers. Moreover, the usage of the numerical way of explanation is time killing and tiresome.

4. Semantic data integration for knowledge graph construction at query time

The paper majorly was published focusing on the ways for integrating the data present in various forms and domains. They have conceived a new technique which has the ability to explore the web domains and produce a logically and relationally connected data. It helps in the development of the search quality where the most accurate results can be obtained. They have used the RDF vocabularies in this process and obviously the triplets-present related to the same entity.

Here the jacard coefficient has a major application. It actually is used in the FUHSEN for obtaining the uniqueness in between the two triplets. When compared to the traditional approach the data is more core-related. They have taken an example where the data regarding a particular painting is obtained from different data bases, so all this data has to be integrated together for a complete connected data set and be published at the time of the request. So, this semantic similarity is what is tackled using this paper. The steps taken for that are a newly developed layer is placed on top of the existing layers therefore masking the original layer.

The process of molecule integration is that when there is a search item provided by the user with some pre-defined threshold value. Then the next step is the writing process where the input given to the search engine is processed and then sent to the search engine for the searching purpose. The engine verifies various data bases and converts the output into rdf molecules. Meanwhile these molecules are summed up with the additional data.

This molecule integration is mainly of three stages. Firstly, the similarity in the molecules is observed. Secondly, the relation of one to one is done with that perfect matching. Finally, once these molecules are added with the additional data then this data will be integrated in likely molecules. The various forms of words used in the datasets are understood by the engine using the dictionary. It is separated into

three various types like search engine, data source, domain specific. The query re-writing uses the vocabulary and transforms the keyword into the language that the wrappers will be able to understand.

As soon as the rdf values are ready then for the further improvement of the facts the relations with the other entities in the data domains. It is possible that the annotations can be attached to the information's. On the whole, this approach helps for the data in integrating the molecules which are distributed over various data bases. This technique is very well suited in the search engine.

There are some instances where I felt like the author has gone off the roads for example take the situation where I was questioning the is there a reason jc was used as a similarity measure only. There was also a place where the data source is defined manually is that a compulsion. But on the whole the paper was well explained and was well applied in the application point of view. Moreover the practical results were very strong supporting in those cases.

5.Fonduer: knowledge base construction from richly formatted data

The paper basically speaks about the knowledge base formation using the formulated data. This is also referred to the process of populating a database. Basically, kbc works on the unstructured data which helps in the down-stream applications. This works to use the datasheets to build a transistor collecting a maximum current. The main resistance present in its development are the document level relations are present before and it just uses them this happens in case of the unformatted data set. In case on the formatted data many relations are present on the linked up data sets.

The approach is made in such a way that the codes and the related data is automatically copied into a new working space data model. Then the facts are obtained from the data set. Using these large data sets more amount of training is necessary for the data. Due to the large number of the text modality and variety is obtained. Many industries have taken steps in the installations of learning model. The kbc is a collection of various different types documents and as a whole the output is a relational database. We have the data programing where the large data set is developed from the existing small sources and domain. This generates an estimate of different labeling functions for the generation of the large training labels set.

Matchers are the python functions used for taking a set of text and checking whether the given set of text is matching the conditions or not.

Throttles are used as the filters for the reduction of the candidates, these are also python functions. Labeling functions are supervising functions in python coding. There are different stages in the data pipeline kbc initialization, candidate generation, probabilistic relation. For the evaluation purposes, the state-of-art and the curated knowledge bases are used. There were few studies made in the experiments where the results projected that the increasing content scope will improve the F1 score. Thus, the multimodal feature library can be evaluated, decrease in the development time by eliminating the input feature, incorporation of all the features with highest extraction quality.

To sum up everything, we can extract info from formatted data overcoming the various challenges. Kbc is published for the better extraction process. Unified data model is enabled. The results were posted on four of the real-world domains which showed a good improvement. Though the paper was well implemented there were some places where wheels were off the road. For suppose though the tests were performed in quite a big data set but for ultimate check very large and real data sets must be used. Candidates-reasoning was not locally done might lead for a combinatorial blast.

Domain and Datasets:

The data sets that we used for this project are available in the links followed. Basically, we have used two datasets in this project one of which is the news dataset and the other being sports. The links to the datasets are posted here.

BBC News - <http://mlg.ucd.ie/datasets/bbc.html>

This dataset describes about the data that discusses mostly about the news.

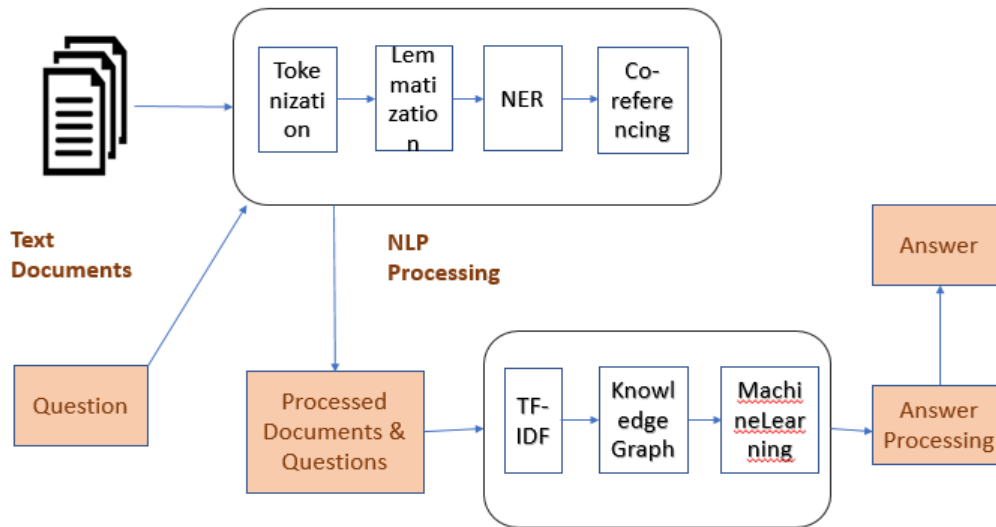
BBC Sports - <http://mlg.ucd.ie/datasets/bbc.html>

This dataset describes about the sports data which includes the domains such as athletics, cricket, football, rugby, tennis.

Design:

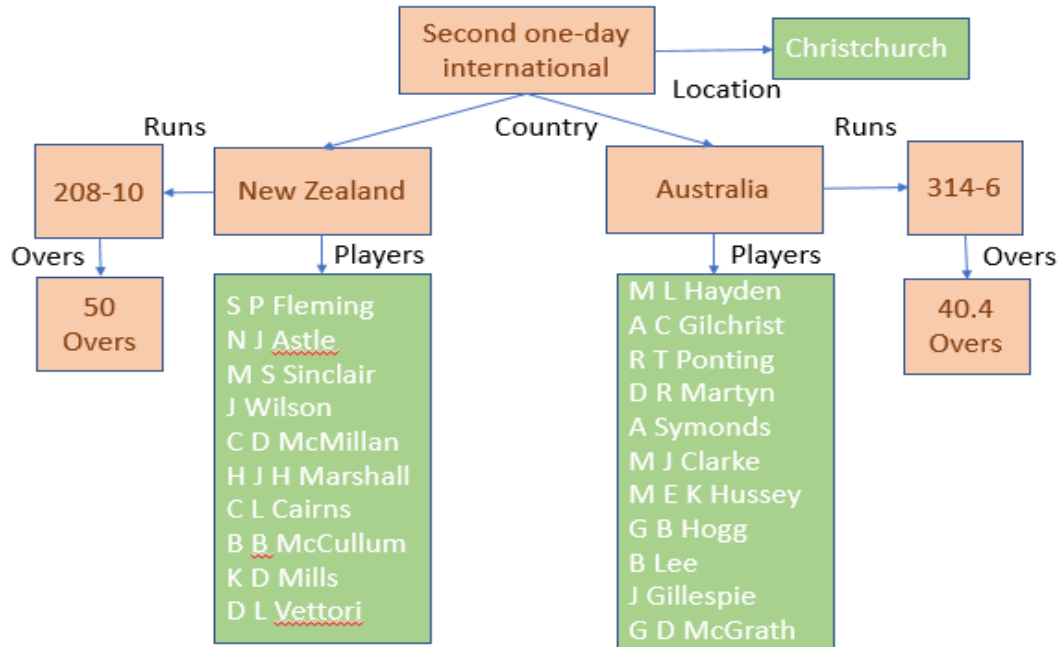
a. Workflow:

Workflow of Question Answering System



b. Knowledge Graph:

Knowledge Graph - Question Answering System



c. Set of questions and answers:

- What sport is discussed in the dataset?
Cricket.
- How many players are playing from Australia?
12.
- Who has won the match?
Australia.
- When was the match played?
May 10, 2015
- Which player has scored most number of runs?
Hayden.

- Are there any players injured in the match?

No.

- Is there any possibility of Australia winning the match based on the commentary?

Better Chances for winning the match.

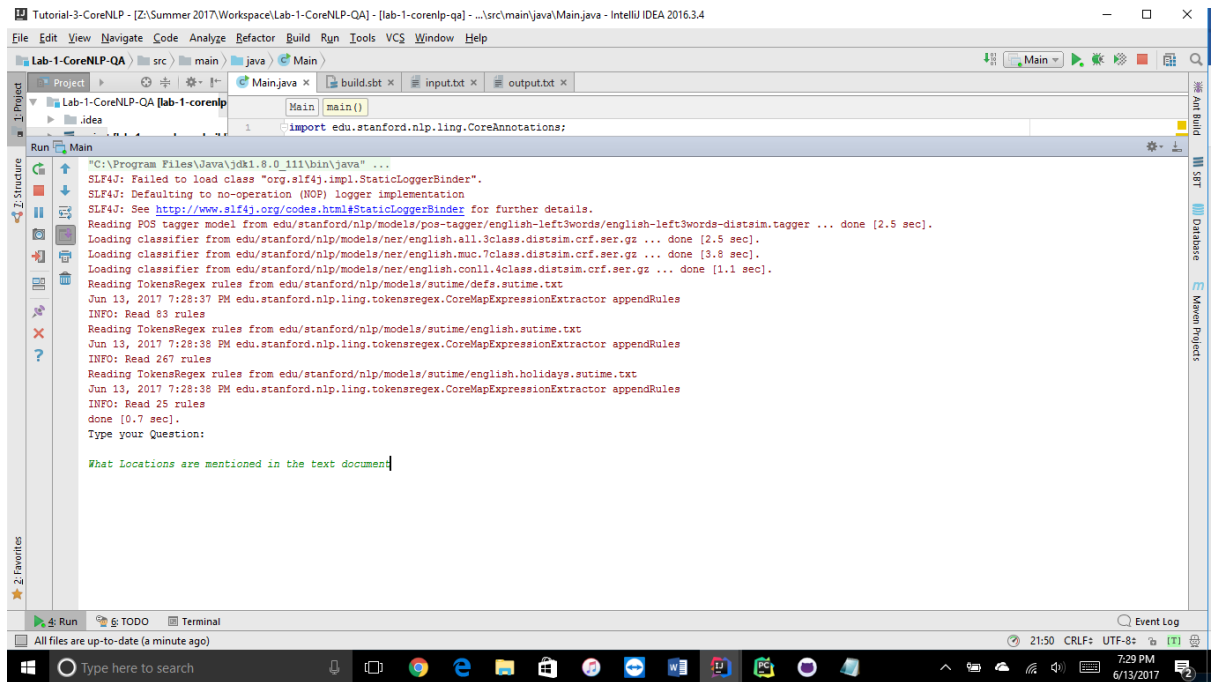
Implementation:

- We have used Natural Language processing (NLP) for tokenization, lemmatization, POS tagging, NER and Co-referencing.
- Used Stanford Core NLP for processing the selected dataset.
- The dataset used is related to BBC sports data on a match between Australia and New Zealand.
- Processed the input data taken from a text document and stored the information in different text files.
- Once after processing the question, we have used the data stored the output text files for answering them.
- We have used the techniques TF-IDF to determine the prominent words in the dataset along with their scores and used Word2Vec to determine the similar words for the prominent words which will better help while understanding the questions asked and about how to answer them efficiently.
- We have used the techniques OpenIE, WordNet, Clustering and Classification techniques to enrich the input dataset and Question Answering System.

Technologies Used: Java, Python and Scala.

Please find the implemented Question Answering System in the below screenshots:

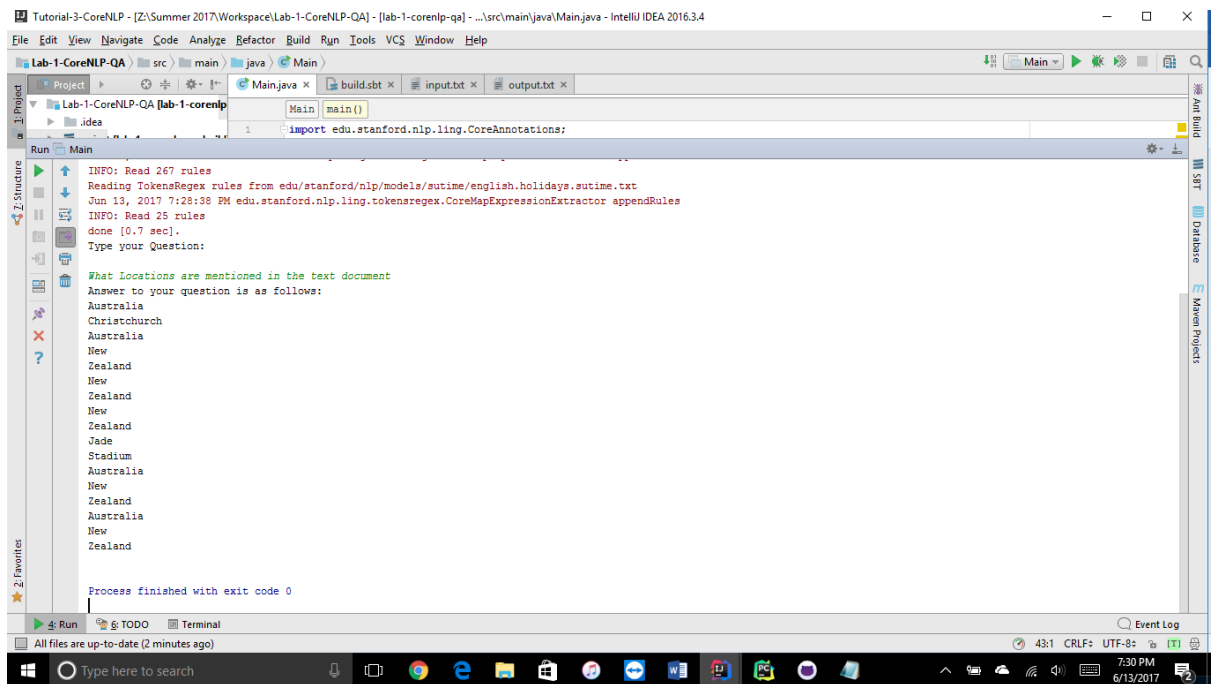
Question 1:



```
Tutorial-3-CoreNLP - [Z:\Summer 2017\Workspace\Lab-1-CoreNLP-QA] - [lab-1-corenlp-qa] - ...src\main\java\Main.java - IntelliJ IDEA 2016.3.4
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Lab-1-CoreNLP-QA | src | main | java | Main
Main.java | build.sbt | input.txt | output.txt
Main
import edu.stanford.nlp.ling.CoreAnnotations;

"C:\Program Files\Java\jdk1.8.0_111\bin\java" ...
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distisim.tagger ... done [2.5 sec].
Loading classifier from edu/stanford/nlp/models/ner/english.all3class.distisim.crf.ser.gz ... done [2.5 sec].
Loading classifier from edu/stanford/nlp/models/ner/english.muc.7class.distisim.crf.ser.gz ... done [3.8 sec].
Loading classifier from edu/stanford/nlp/models/ner/english.conll.4class.distisim.crf.ser.gz ... done [1.1 sec].
Reading TokenRegex rules from edu/stanford/nlp/models/sutime/defs.sutime.txt
Jun 13, 2017 7:28:37 PM edu.stanford.nlp.ling.tokensregex.CoreMapExpressionExtractor appendRules
INFO: Read 83 rules
Reading TokenRegex rules from edu/stanford/nlp/models/sutime/english.sutime.txt
Jun 13, 2017 7:28:38 PM edu.stanford.nlp.ling.tokensregex.CoreMapExpressionExtractor appendRules
INFO: Read 267 rules
Reading TokenRegex rules from edu/stanford/nlp/models/sutime/english.holidays.sutime.txt
Jun 13, 2017 7:28:38 PM edu.stanford.nlp.ling.tokensregex.CoreMapExpressionExtractor appendRules
INFO: Read 25 rules
done [0.7 sec].
Type your Question:

What Locations are mentioned in the text document
```



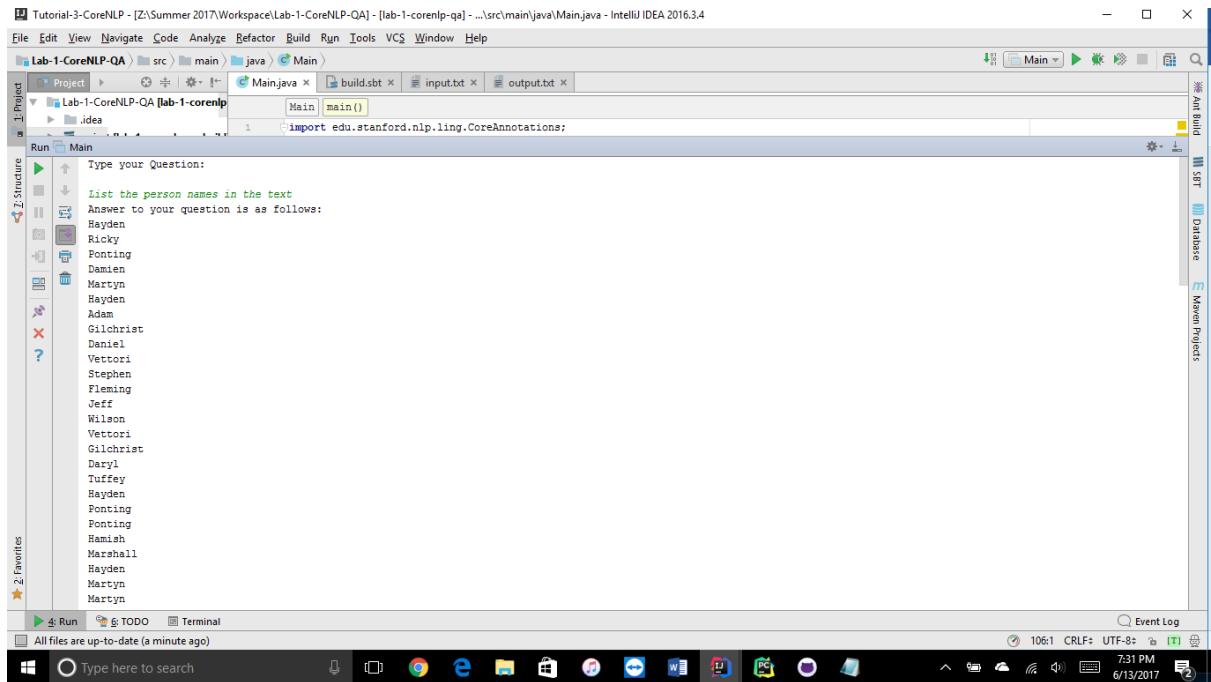
```
Tutorial-3-CoreNLP - [Z:\Summer 2017\Workspace\Lab-1-CoreNLP-QA] - [lab-1-corenlp-qa] - ...src\main\java\Main.java - IntelliJ IDEA 2016.3.4
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Lab-1-CoreNLP-QA | src | main | java | Main
Main.java | build.sbt | input.txt | output.txt
Main
import edu.stanford.nlp.ling.CoreAnnotations;

INFO: Read 267 rules
Reading TokenRegex rules from edu/stanford/nlp/models/sutime/english.holidays.sutime.txt
Jun 13, 2017 7:28:38 PM edu.stanford.nlp.ling.tokensregex.CoreMapExpressionExtractor appendRules
INFO: Read 25 rules
done [0.7 sec].
Type your Question:

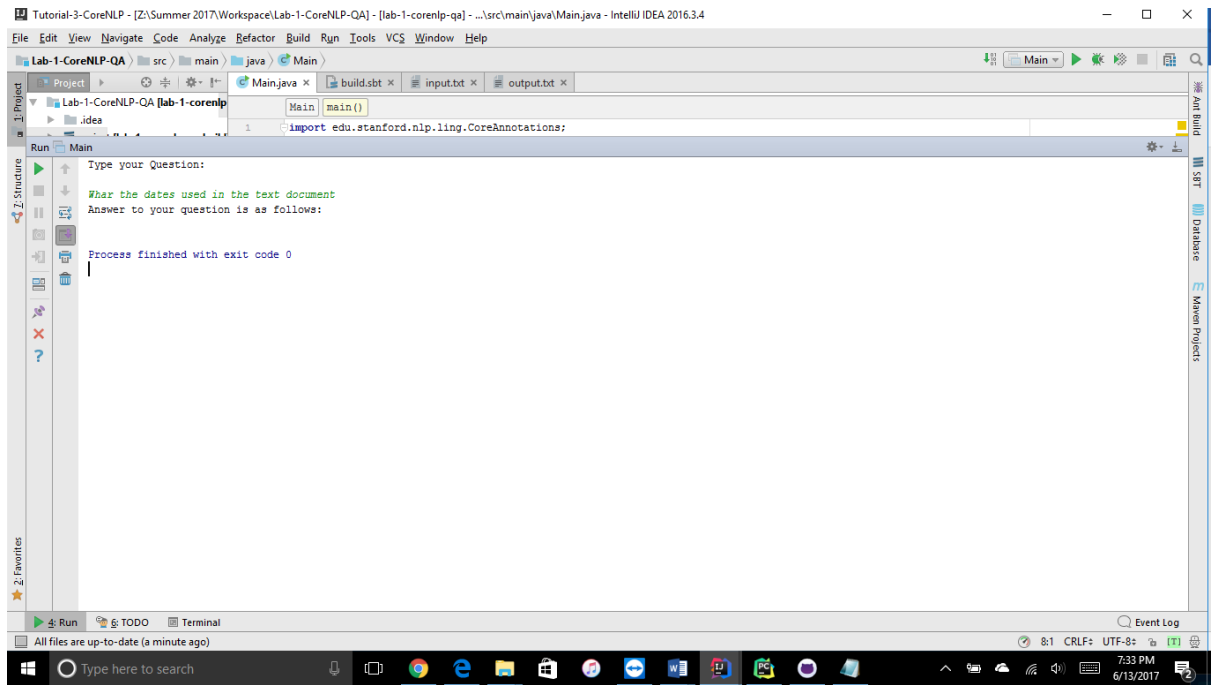
What Locations are mentioned in the text document
Answer to your question is as follows:
Australia
Christchurch
Australia
New
Zealand
New
Zealand
New
Zealand
Jade
Stadium
Australia
New
Zealand
Australia
New
Zealand

Process finished with exit code 0
```

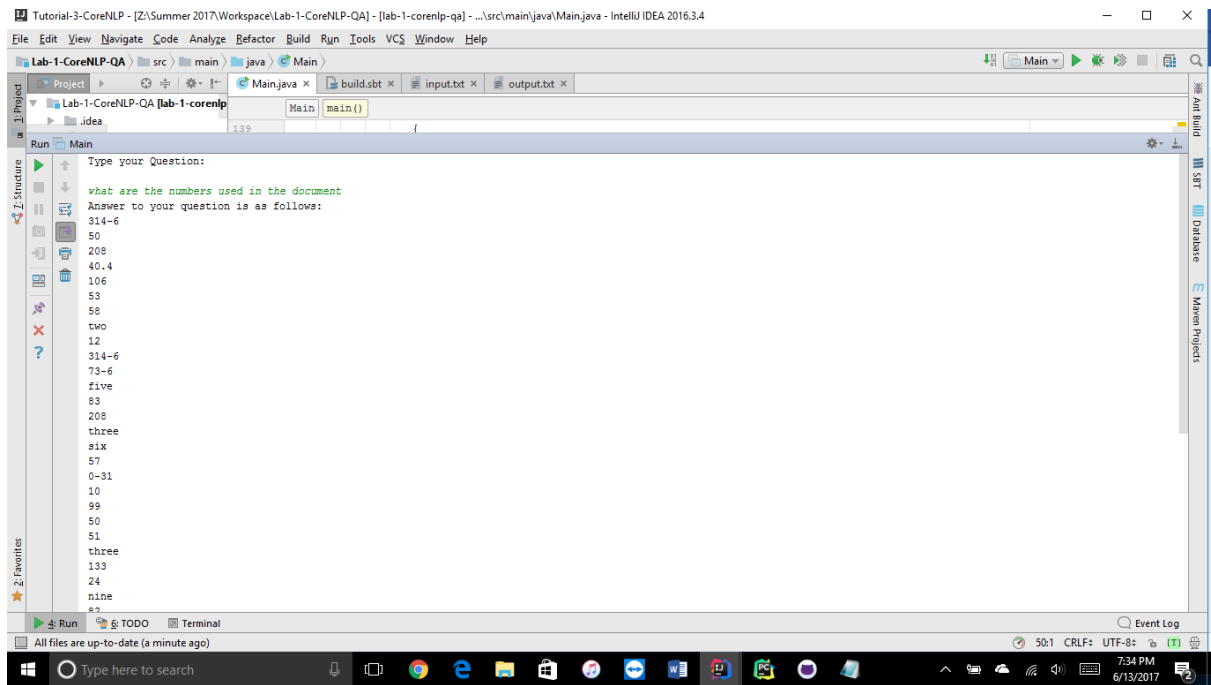
Question 2:



Question 3:



Question 4:

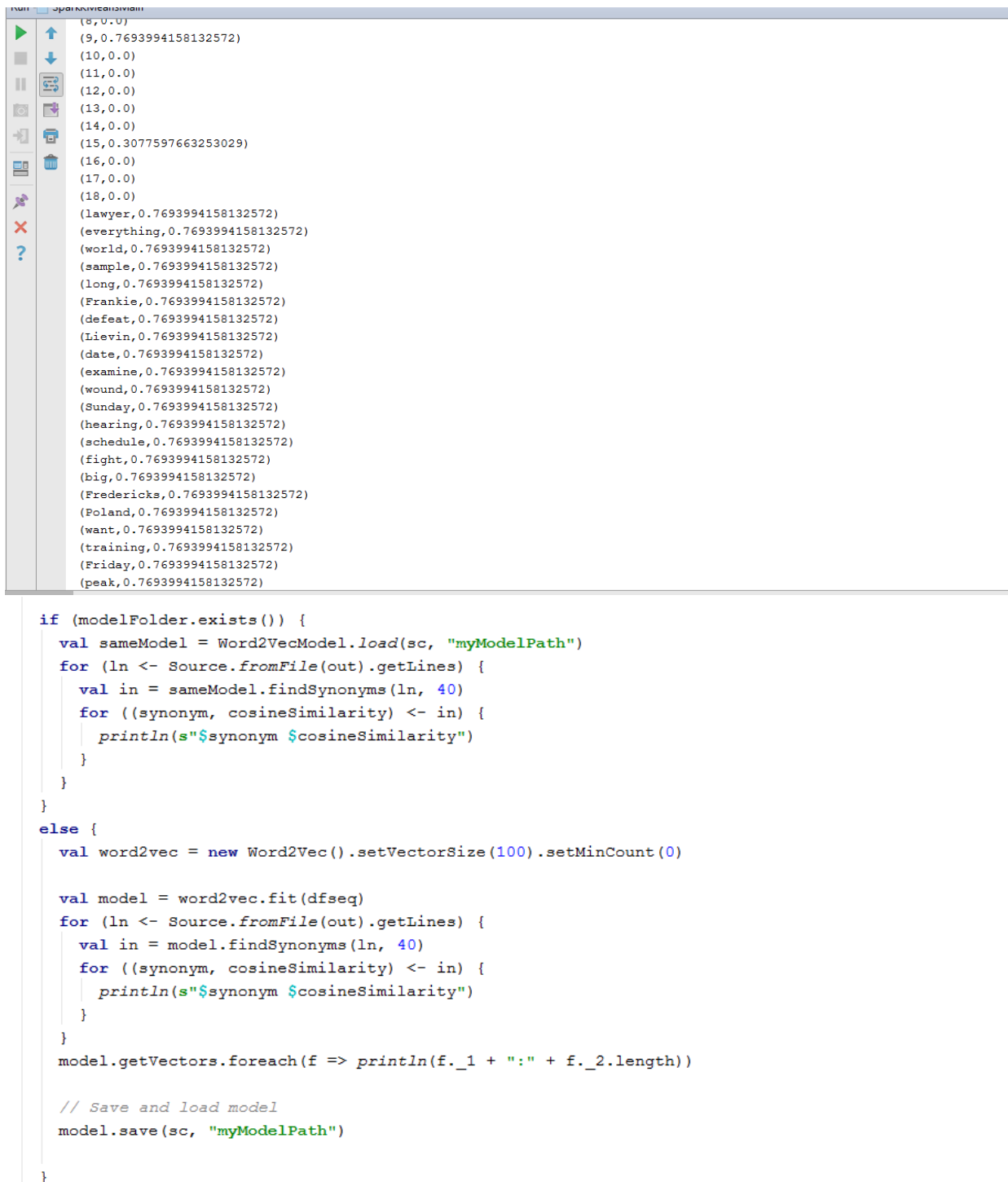


The screenshot shows the IntelliJ IDEA 2016.3.4 interface. The main editor displays the output of a Java program. The program prompts the user to "Type your Question:" and then outputs the following text:

```
what are the numbers used in the document
Answer to your question is as follows:
314-6
50
208
40.4
106
53
58
two
12
314-6
73-6
five
83
208
three
six
57
0-31
10
99
50
51
three
133
24
nine
87
```

The interface includes a menu bar (File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, Help), a toolbar, and a sidebar with Project, Run, and Event Log tabs. The status bar at the bottom shows the file encoding as UTF-8 and the date as 6/13/2017.

Screenshots:



The screenshot displays a Scala REPL window with a list of word vectors and their cosine similarities. The words are listed in two columns, with the first column containing words and the second column containing their corresponding cosine similarity values. The words are: (8,0.0), (9,0.7693994158132572), (10,0.0), (11,0.0), (12,0.0), (13,0.0), (14,0.0), (15,0.3077597663253029), (16,0.0), (17,0.0), (18,0.0), (lawyer,0.7693994158132572), (everything,0.7693994158132572), (world,0.7693994158132572), (sample,0.7693994158132572), (long,0.7693994158132572), (Frankie,0.7693994158132572), (defeat,0.7693994158132572), (Lievin,0.7693994158132572), (date,0.7693994158132572), (examine,0.7693994158132572), (wound,0.7693994158132572), (Sunday,0.7693994158132572), (hearing,0.7693994158132572), (schedule,0.7693994158132572), (fight,0.7693994158132572), (big,0.7693994158132572), (Fredericks,0.7693994158132572), (Poland,0.7693994158132572), (want,0.7693994158132572), (training,0.7693994158132572), (Friday,0.7693994158132572), and (peak,0.7693994158132572). Below the list, the source code for the model is shown, which includes loading a model, finding synonyms, and saving the model.

```
if (modelFolder.exists()) {
  val sameModel = Word2VecModel.load(sc, "myModelPath")
  for (ln <- Source.fromFile(out).getLines) {
    val in = sameModel.findSynonyms(ln, 40)
    for ((synonym, cosineSimilarity) <- in) {
      println(s"$synonym $cosineSimilarity")
    }
  }
}
else {
  val word2vec = new Word2Vec().setVectorSize(100).setMinCount(0)

  val model = word2vec.fit(dfseq)
  for (ln <- Source.fromFile(out).getLines) {
    val in = model.findSynonyms(ln, 40)
    for ((synonym, cosineSimilarity) <- in) {
      println(s"$synonym $cosineSimilarity")
    }
  }
  model.getVectors.foreach(f => println(f._1 + ":" + f._2.length))

  // Save and load model
  model.save(sc, "myModelPath")
}
```

rs	76	Admission
	77	bronze
	78	stand
	79	winner
	80	extra
	81	prepared
	82	man
	83	evening
	84	rrb
	85	happy
	86	Jamaica
	87	matter
	88	Run
	89	Qatar
	90	airport
	91	<u>Olsson</u>
	92	Justin
	93	<u>yearold</u>
	94	faze
	95	two
	96	bit
	97	represent
	98	thought
	99	tank
	100	Alan
	101	<u>halfmarathon</u>
	102	ability
	103	crowd
	104	whole
	105	beneficial
	106	comment
	107	Steve
	108	Army
	109	Roger
	110	conserve
	111	meeting
	112	ban
	113	AllTime
	114	Davies
	115	audience


```

//Creating Term Frequency of the document
val tf = hashingTF.transform(dfseq)
tf.cache()

val idf = new IDF().fit(tf)

//Creating Inverse Document Frequency
val tfidf = idf.transform(tf)
val tfidfvalues = tfidf.flatMap(f => {
  val ff: Array[String] = f.toString.replace(", ", ";").split(";")
  val values = ff(2).replace("]", "").replace(")", "").split(",")
  values
})

val tfidfindex = tfidf.flatMap(f => {
  val ff: Array[String] = f.toString.replace(", ", ";").split(";")
  val indices = ff(1).replace("]", "").replace(")", "").split(",")
  indices
})

val stopWordRemovedDF = df.map(f => {
  //Filtered numeric and special characters out
  val filteredF = f._2.map(_.replaceAll("[^a-zA-Z]", ""))
  //Filter out the Stop Words
  .filter(ff => {
    if (stopWordsBroadcast.value.contains(ff.toLowerCase))
      false
    else
      true
  })
  (f._1, filteredF)
})

def getNGrams(sentence: String, n: Int): Array[Array[String]] = {
  val words = sentence
  val ngrams = words.split(' ').sliding(n)
  ngrams.toArray
}

val df = sc.wholeTextFiles(paths.mkString(",")).map(f => {
  val lemmatised = CoreNLP.returnLemma(f._2)
  val splitString = lemmatised.split(" ")
  (f._1, splitString)
})

```

Corpus summary:

Training set size: 19 documents
Vocabulary size: 3220 terms
Preprocessing time: 13.038592142 sec

Finished training KMeans model. Summary:

Training time: 0.774973025 sec
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/002.txt;1
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/004.txt;0
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/005.txt;2
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/006.txt;2
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/007.txt;1
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/008.txt;0
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/009.txt;4
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/010.txt;2
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/011.txt;1
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/012.txt;1
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/013.txt;0
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/014.txt;3
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/015.txt;1
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/016.txt;0
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/017.txt;2
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/018.txt;2
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/019.txt;2
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/020.txt;1
file:/C:/Users/USER/Downloads/Spark_MachineLearning/data/bbc sport/sport.txt;0

```
val inputPath = Seq("data/bbc sport/*")
```

```
Logger.getRootLogger.setLevel(Level.WARN)
```

```
val topic_output = new PrintStream("data/Results_KMeans.txt")
```



```

val dd=data.map(f=>{
    val wordnet = new RiWordNet("C:\\Users\\user\\Desktop\\WordNet-3.0")
    val farr=f.split(" ")
    getSynonyms(wordnet, "entertainment")
})
dd.take(1).foreach(f=>println(f.mkString(" ")))
}

```

```

val input = sc.textFile("C:\\Users\\user\\Desktop\\KDM\\bbc\\entertainment\\019.txt").map(line => {
    println(CoreNLP.returnTriplets(line))
    //println(t)
})

```

```

Run SparkKMeansMain
17/07/10 09:32:00 INFO Utils: Successfully started service 'SparkUI' on port 4040.
17/07/10 09:32:00 INFO SparkUI: Started SparkUI at http://192.168.56.1:4040
17/07/10 09:32:00 INFO Executor: Starting executor ID driver on host localhost
17/07/10 09:32:00 INFO Utils: Successfully started service 'org.apache.spark.network.netty.NettyBlockTransferService' on port 53487.
17/07/10 09:32:00 INFO NettyBlockTransferService: Server created on 53487
17/07/10 09:32:00 INFO BlockManagerMaster: Trying to register BlockManager
17/07/10 09:32:00 INFO BlockManagerMasterEndpoint: Registering block manager localhost:53487 with 1125.8 MB RAM, BlockManagerId(driver, localhost, 53487)
17/07/10 09:32:00 INFO BlockManagerMaster: Registered BlockManager
17/07/10 09:32:00 INFO BlockManagerMaster: Registered BlockManager
Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distsim.tagger ... done [1.9 sec].

Corpus summary:
  Training set size: 40 documents
  Vocabulary size: 29757 terms
  Preprocessing time: 38.137583533 sec

17/07/10 09:32:39 WARN KMeans: The input data is not directly cached, which may hurt performance if its parent RDDs are also uncached.
17/07/10 09:32:41 WARN BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeSystemBLAS
17/07/10 09:32:41 WARN BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeRefBLAS
17/07/10 09:32:41 WARN KMeans: The input data was not directly cached, which may hurt performance if its parent RDDs are also uncached.
Finished training KMeans model. Summary:
  Training time: 1.607287882 sec

Process finished with exit code 0

```

Project management:

a. Contribution of each member:

Name	Work Contribution	Percentage Contributed
Sandeep Pabolu	NLP and Spark Coding, Dataset Processing, Journal Club, Documentation	25%
Sri Sai Sarat Chandra Varma Mudunuri	OpenIE, WordNet Coding, Dataset Processing, Journal Club, Documentation	25%
Lava Kumar Surparaju	Machine Learning Coding, Dataset Processing, Journal Club, Documentation	25%
Rakesh Reddy Pallepati	TF-IDF and Word2Vec Coding, Dataset Processing, Journal Club, Documentation	25%

b. ZenHub and GitHub 's:

<https://github.com/SaratM34/KDM-Lab-Assignments>

Iteration 1:

Clear current search query, filters, and sorts

<input type="checkbox"/>	0 Open ✓ 9 Closed	Author ▾	Labels ▾	Projects ▾	Milestones ▾	Assignee ▾	Sort ▾
<input type="checkbox"/>	<div>Exceptions 3</div> <div>#9 by SaratM34 was closed 2 days ago</div> <div>Testing 3</div>						
<input type="checkbox"/>	<div>Function failure 13</div> <div>#8 by SaratM34 was closed 2 days ago</div> <div>Testing 3</div>						
<input type="checkbox"/>	<div>Network Failure 5</div> <div>#7 by SaratM34 was closed 2 hours ago</div> <div>Testing 3</div>						
<input type="checkbox"/>	<div>Validation errors 5</div> <div>#6 by SaratM34 was closed 2 days ago</div> <div>Testing 2</div>						
<input type="checkbox"/>	<div>Login errors 21</div> <div>#5 by SaratM34 was closed 2 days ago</div> <div>Testing 2</div>						
<input type="checkbox"/>	<div>Compile time errors 8</div> <div>#4 by SaratM34 was closed 2 hours ago</div> <div>Testing 2</div>						
<input type="checkbox"/>	<div>Errors 13</div> <div>#3 by SaratM34 was closed 2 days ago</div> <div>Testing 1</div>						
<input type="checkbox"/>	<div>Bug Fixing 8</div> <div>#2 by SaratM34 was closed 2 days ago</div> <div>Testing 1</div>						
<input type="checkbox"/>	<div>UI Enhancements 5</div> <div>#1 by SaratM34 was closed 2 hours ago</div> <div>Testing 1</div>						

SaratM34 / KDM-Lab-Assignments

Unwatch 1

Star 0

Fork 0

<> Code

Issues 0

Pull requests 0

Boards

Reports

Projects 0

Wiki

Insights ▾

View ▾

Repos (1/1) ▾

Show one

Labels ▾

Milestones ▾

Assignees ▾

Epics ▾

Releases ▾

Search (/)

New Issue +

0 issues - 0 story points

In Progress

0 issues - 0 story points

Review/QA

0 issues - 0 story points

Done

9+ issues - 81 story points

Closed

KDM-Lab-Assignments #4

Compile time errors

Testing 2

0

KDM-Lab-Assignments #1

UI Enhancements

Testing 1

5

KDM-Lab-Assignments #7

Network Failure

Testing 3

6

Add a Pipeline ...

Labels

Milestones

New milestone

3 Open 0 Closed

Sort

Testing 2

Past due by 1 day Last updated about 2 hours ago

100% complete 0 open 3 closed

Edit Close Delete

Testing 1

Past due by 1 day Last updated about 2 hours ago

100% complete 0 open 3 closed

Edit Close Delete

Testing 3

Past due by 1 day Last updated about 2 hours ago

100% complete 0 open 3 closed

Edit Close Delete

Burndown

Velocity tracking

Release report

Testing 1

Start: Jun 10, 2017 Edit Due: Jun 12, 2017 Edit

Edit Milestone

Milestones

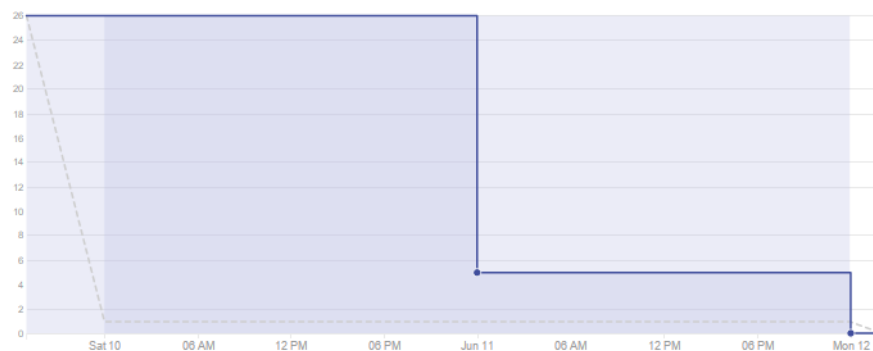
Labels

Hide Pull Requests

Burn Pipelines

Burndown report

Weekends Ideal Completed



26 Total Story Points
26 Completed / 0 Remaining

3 Total Issues and Pull Requests
3 Completed / 0 Remaining

Testing 2

Start: Jun 10, 2017 Edit Due: Jun 12, 2017 Edit

Edit Milestone

Milestones

Labels

Hide Pull Requests

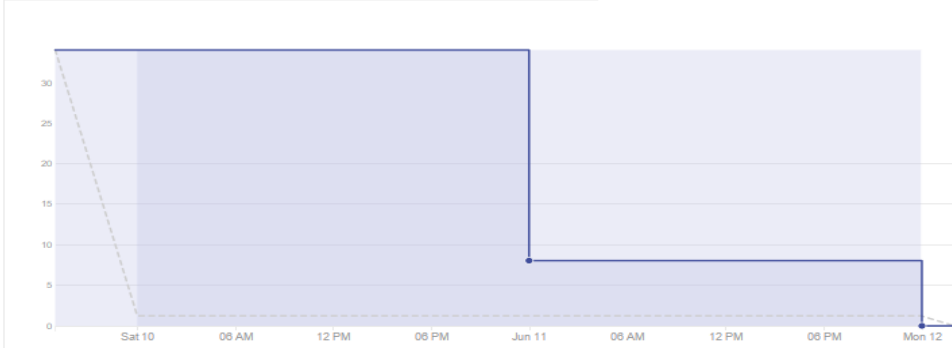
Burn Pipelines

Burndown report

Weekends

Ideal

Completed



34 Total Story Points

34 Completed / 0 Remaining

3 Total Issues and Pull Requests

3 Completed / 0 Remaining

Burndown

Velocity tracking

Release report

Testing 3

Start: Jun 10, 2017 [Edit](#) Due: Jun 12, 2017 [Edit](#)

[Edit Milestone](#)

[Milestones](#)

[Labels](#)

[Hide Pull Requests](#)

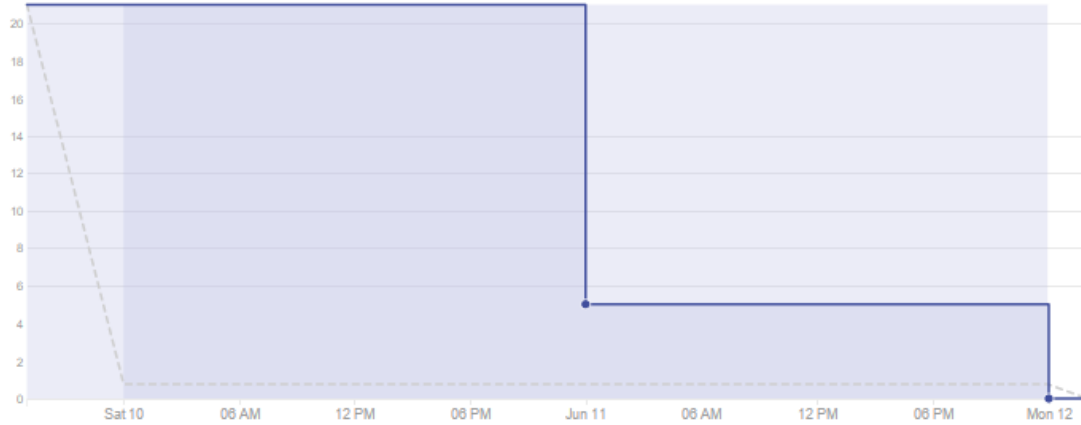
[Burn Pipelines](#)

Burndown report

☐ Weekends

☐ Ideal

☒ Completed



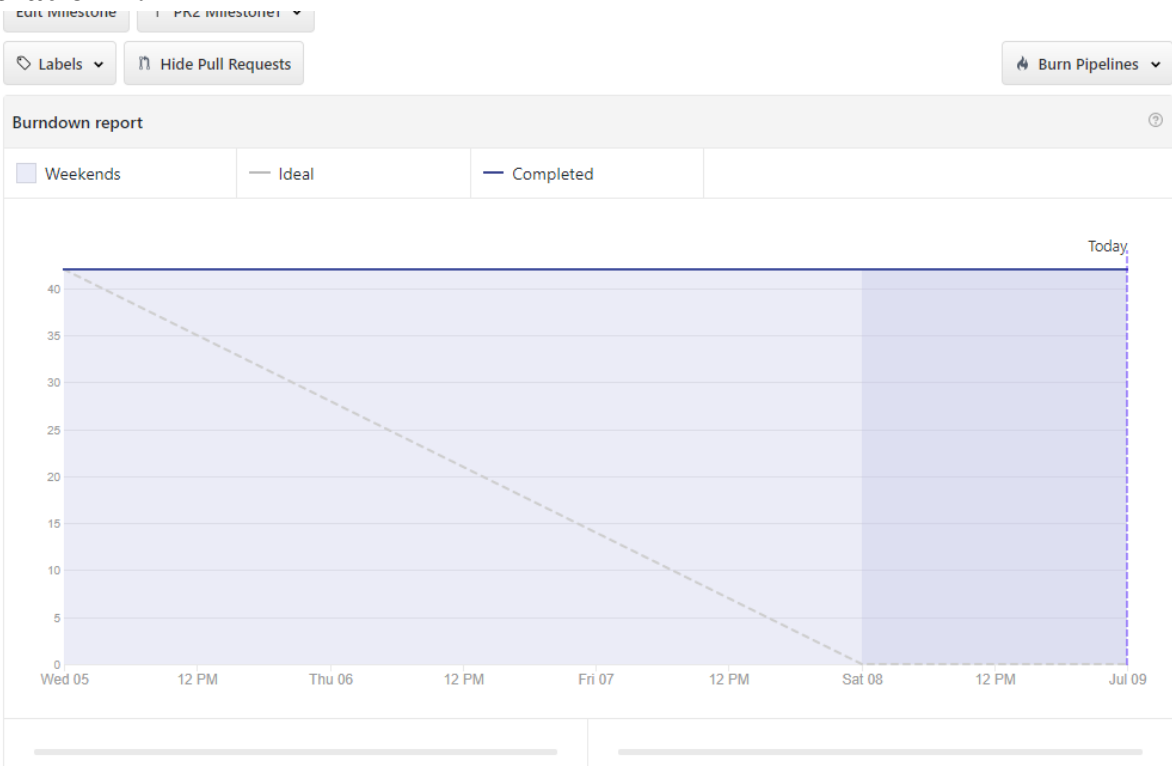
21 Total Story Points

21 Completed / 0 Remaining

3 Total Issues and Pull Requests

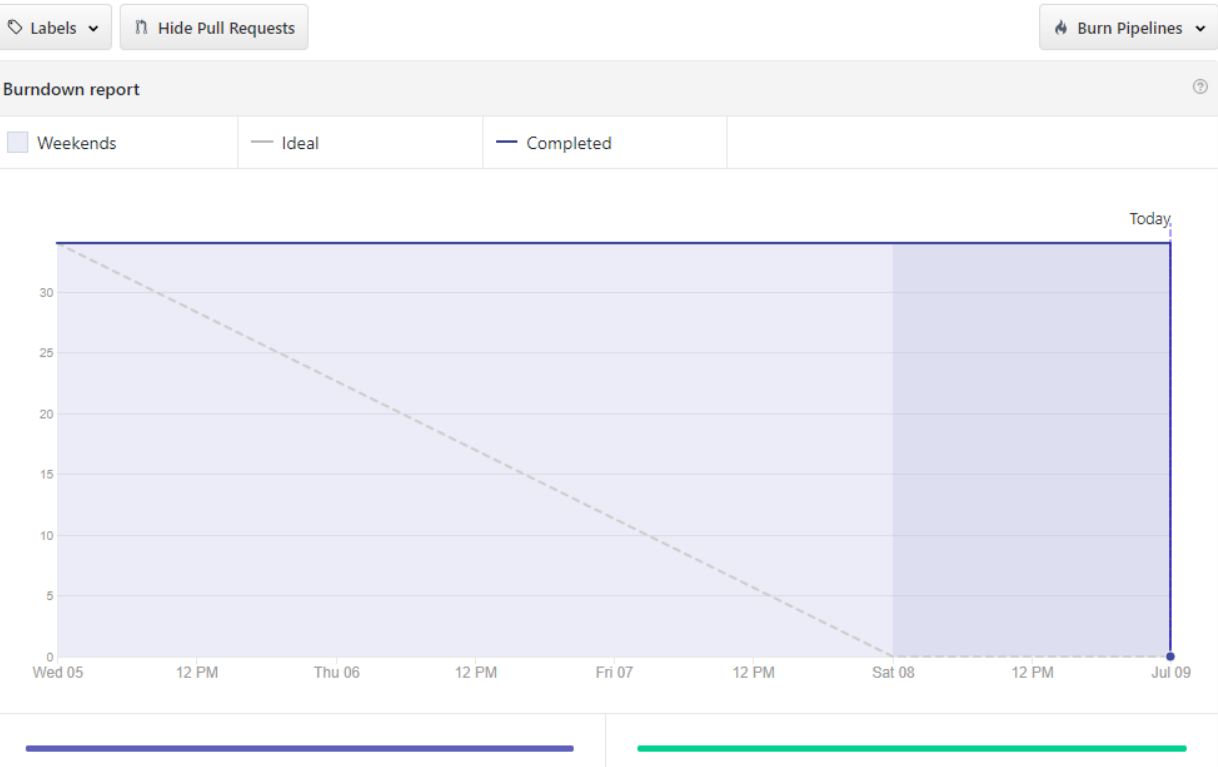
3 Completed / 0 Remaining

Iteration 2:



<input type="checkbox"/>	0 Open ✓ 6 Closed	Author ▾	Labels ▾	Projects ▾	Milestones ▾	Assignee ▾	Sort ▾
<input type="checkbox"/>	🔔 PR2 ML Classification 21	#6 by SaratM34 was closed a minute ago 📅 PR2 Milestone 2					
<input type="checkbox"/>	🔔 PR2 ML Clustering 5	#5 by SaratM34 was closed a minute ago 📅 PR2 Milestone 2					
<input type="checkbox"/>	🔔 PR2 WordNet Implementation 8	#4 by SaratM34 was closed a minute ago 📅 PR2 Milestone 2					
<input type="checkbox"/>	🔔 PR2 OpenIE implementation 21	#3 by SaratM34 was closed a minute ago 📅 PR2 Milestone1					
<input type="checkbox"/>	🔔 PR2 Word2Vec Implementation 13	#2 by SaratM34 was closed a minute ago 📅 PR2 Milestone1					
<input type="checkbox"/>	🔔 PR2 TFIDF Implementation 8	#1 by SaratM34 was closed a minute ago 📅 PR2 Milestone1					

💡 ProTip! What's not been updated in a month: [updated:<2017-06-09](#).



c. Concerns and Issues:

- Notable issues are regarding processing data for some set of questions.
- Filtering stop words, synonyms.
- Co-referencing for complex datasets.
- Relation between the persons in the text.
- Related words questions need to be achieved.

d. Future Work:

In our next project report we will be extending our current questions set to much more complex set. Also, Processing data using Machine learning and Deep learning techniques. We need to implement the relation analysis, enrich the question type with yes/no answers. Full-Fledged Implementation of the Knowledge Graph for the Dataset . Text classification for the dataset using Shallow/Deep Learning.