

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

KNOWLEDGE DISCOVERY AND MANAGEMENT

SUMMARIZATION

INSTRUCTOR:

Dr. Yugyung Lee

TEAM 8:

VILAS MAMIDYALA

VIKESH PADARTHI

DINESH KUMAR BANDAM

BHUMIREDDY RANJITHA REDDY

FINAL REPORT - SUMMARIZATION

1. Motivation:

We know that the whole world is awaiting to hear the result of US election which are going to be released by the end of this year. Everyone would like to see how these elections are going to be held. One has an anxiety that who is going to win and what actually the people opinion is and who has more probability to win. These questions stimulate our work towards collecting data about politics which clears all our skeptic things about elections. Since many of the things related to students and their future who have more excitement and worry to get to know the result. Our main motivation behind this project is to analyze the data present in social media like twitter and plot some graphs which shows about which candidate is more famous in social media, the probability of who will be getting elected.

Objective:

Main objective of this project is to use NLP, machine learning knowledge to predict the outcome of election result. Using these we can summarize the result of various blogs, news, and editorial matters in newspapers which are related to elections. We will first plot some graphs based on the twitter data which we have collected. And we want to analyze various text data present in the World Wide Web like Wikipedia and summarize these papers.

Expected outcomes:

By performing these operations using NLP, Machine Learning we want to predict the outcome of the US elections and various views about US elections by the people around the world. The output will be ontology graphs which are developed by analyzing the data sets which are related to US elections.

2. Domain:

Data Set: Twitter Data, provided data sets by Lee.

Technologies: Java, Scala.

Topic: US Politics

IDE : IntelliJ

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

3. Data Collection:

Twitter data using JAVA and Linux.

4. Task and Features:

- Source data
https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/output_word2vec.txt
- Collected Twitter data using Java code. Link for the source code is:
https://github.com/vilasmamidyla/KDM_SM16_SM/tree/master/Source/twit
- NLP processing has been applied to the sample input collected above .
https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/Nlp%20Output.txt
https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/Simplecorenlpoutput.txt
- Word count has been applied to the given same input :
https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/wordcount_output.txt
- Information Extraction/Retrieval technologies :
https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/wordcount_TFIDF.txt
- Name Entity Extraction/Relation Extraction
https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/sparkner.pdf

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTH (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

```

Run SparkNER
(El_Universal_Mx : ,0 0 )
(#entérate ,0 )
(Lynch ,PERSON )
(y ,0 )
(Comey ,PERSON )
(comparecerán ,0 )
(ante ,0 )
(Congreso ,PERSON )
(por ,0 )
(caso ,0 )
(HillaryClinton ,PERSON )
(https://t.co/wj1hipatd6 ,0 )
,(POLL)
(rt ,0 )
(@Reince : ,0 0 )
(. @hillaryclinton ,0 0 )
(spend ,0 )
(the ,0 )
(last ,0 )
(16 ,NUMBER )
(month ,DURATION )
(deliberately ,0 )
(lie ,0 )
(to ,0 )
(the ,0 )
(American ,MISC )
(people . ,0 0 )

```

```

SparkNER
(spend ,0 )
(the ,0 )
(last ,0 )
(16 ,NUMBER )
(month ,DURATION )
(deliberately ,0 )
(lie ,0 )
(to ,0 )
(the ,0 )
(American ,MISC )
(people . ,0 0 )
(watch : ,0 0 )
(https://t.co/10kiu6b69a ,0 )
,(POLL)
16/07/08 21:58:36 INFO SparkContext: Invoking stop() from shutdown hook
16/07/08 21:58:36 INFO SparkUI: Stopped Spark web UI at http://192.168.0.20:4040
16/07/08 21:58:36 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
16/07/08 21:58:36 INFO MemoryStore: MemoryStore cleared
16/07/08 21:58:36 INFO BlockManager: BlockManager stopped
16/07/08 21:58:36 INFO BlockManagerMaster: BlockManagerMaster stopped
16/07/08 21:58:36 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
16/07/08 21:58:36 INFO SparkContext: Successfully stopped SparkContext
16/07/08 21:58:36 INFO ShutdownHookManager: Shutdown hook called
16/07/08 21:58:36 INFO ShutdownHookManager: Deleting directory C:\Users\vilas\AppData\Local\Temp\spark-d3b84157-3504-4f3e-ac9c-54f198249045

Process finished with exit code 0

```

■ WordNet

https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/Wordnet_output.docx

■ Topic Discovery

LDA:

https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/LDA_Results.txt

COMP-SCI 5560 (SUMMER 2016) – KNOWLEDGE DISCOVERY AND MANAGEMENT

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

The screenshot shows the IntelliJ IDEA interface with the project SparkOpenIE selected. The code editor displays the `WordNetMain.java` file, which contains Java code for interacting with a WordNet database. Below the code editor is a terminal window showing the execution of the program. The terminal output includes parts of speech for words like "win" and "loss", definitions for "loss", and a list of synonyms for "win". The system tray at the bottom right indicates the date and time as 7/8/2016, 8:36 PM.

```

String[] poss = wordnet.getPos(word);
for (int j = 0; j < poss.length; j++) {
    System.out.println("\n\nSynonyms for " + word + " (" + poss[j] + ")");
    String[] synonyms = wordnet.getAllSynonyms(word, poss[j], 10);
    for (int i = 0; i < synonyms.length; i++) {
        System.out.println(synonyms[i]);
    }
}

String[] poss = wordnet.getPos(word);
for (int j = 0; j < poss.length; j++) {
    System.out.println("\n\nSynonyms for " + word + " (" + poss[j] + ")");
    String[] synonyms = wordnet.getAllSynonyms(word, poss[j], 10);
    for (int i = 0; i < synonyms.length; i++) {
        System.out.println(synonyms[i]);
    }
}

```

The screenshot shows the IntelliJ IDEA interface with the project Spark_KMeans_FV selected. The code editor displays the `LDA_Results.txt` file, which contains the results of an LDA model training. The terminal window shows the summary of the corpus, the time taken to preprocess the data, and the log likelihood of the trained model. The output also lists the topics and their associated words. The system tray at the bottom right indicates the date and time as 7/8/2016, 8:36 PM.

```

Corpus summary:
Training set size: 10704 documents
Vocabulary size: 11485 terms
Training set size: 68775 tokens
Preprocessing time: 36.617139417 sec

Finished training LDA model. Summary:
Training time: 231.321711026 sec
Training data average log likelihood: -51.39299144407459

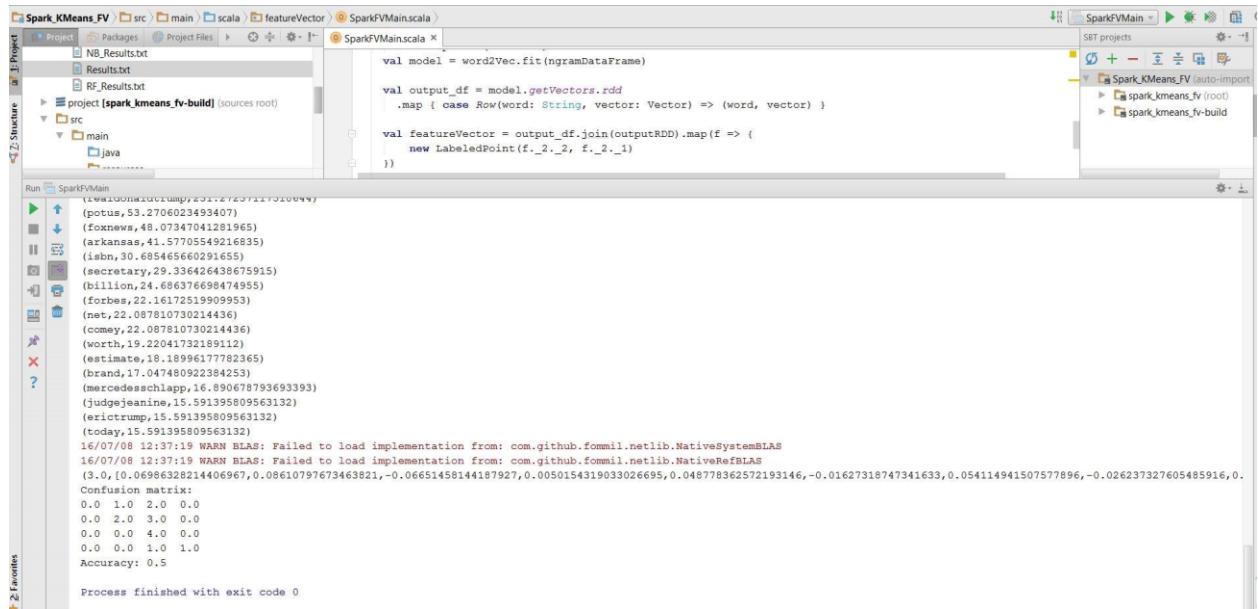
4 topics:
TOPIC_0:::0.06898491214163481
TOPIC_0::rt;0.05230523781160823
TOPIC_0::..0.0443182452233433

```

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

■ Feature Vector for Machine Learning

https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/spark_fv_output.pdf



The screenshot shows the IntelliJ IDEA interface with the following details:

- Project Structure:** The left sidebar shows the project structure for "Spark_KMeans_FV". It includes a "src" folder containing "main" and "scala" subfolders, and a "featureVector" folder which contains "SparkFVMain.scala". Other files like "NB_Results.txt" and "Results.txt" are also listed.
- Code Editor:** The main window displays the content of "SparkFVMain.scala". The code defines a word2Vec model and creates a feature vector from it.
- Logs:** The bottom pane shows the run logs for "SparkFVMain". The logs output a list of labeled points and conclude with "Process finished with exit code 0".
- IDE Features:** The top right shows the SBT projects panel with "Spark_KMeans_FV" selected. The bottom right shows the Scala code editor interface.

```

val model = word2Vec.fit(ngramDataFrame)

val output_df = model.getVectors.rdd
    .map { case Row(word: String, vector: Vector) => (word, vector) }

val featureVector = output_df.join(outputRDD).map(f =>
    new LabeledPoint(f._2._3, f._2._1)
)

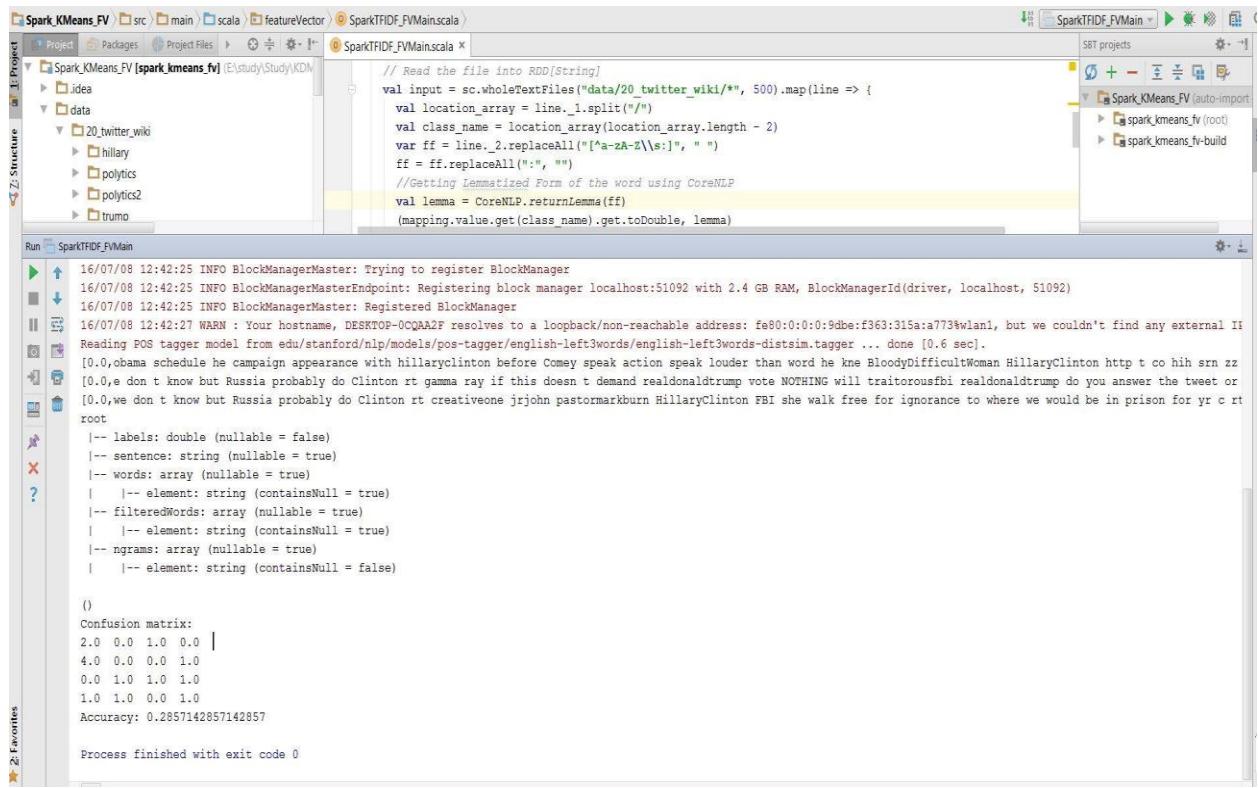
```

```

16/07/08 12:37:19 WARN BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeSystemBLAS
16/07/08 12:37:19 WARN BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeRefBLAS
(3.0, [0.0696328214406967, -0.08610797673463821, -0.06651450144187927, 0.0050154319033026695, 0.048778362572193146, -0.01627318747341633, 0.054114941507577896, -0.026237327605485916, 0.
Confusion matrix:
0.0 1.0 2.0 0.0
0.0 2.0 3.0 0.0
0.0 0.0 4.0 0.0
0.0 0.0 1.0 1.0
Accuracy: 0.5

```

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)
https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/spark_fv_out.pdf



```

// Read the file into RDD[String]
val input = sc.wholeTextFiles("data/20_twitter_wiki/*", 500).map(line => {
    val location_array = line._1.split("/")
    val class_name = location_array(location_array.length - 2)
    var ff = line._2.replaceAll("[^a-zA-Z\\s:]", " ")
    ff = ff.replaceAll(":", "")
    //Getting Lemmatized Form of the word using CoreNLP
    val lemma = CoreNLP.returnLemma(ff)
    (mapping.value.get(class_name).get.toDouble, lemma)
})

//-- labels: double (nullable = false)
//-- sentence: string (nullable = true)
//-- words: array (nullable = true)
//|   |-- element: string (containsNull = true)
//-- filteredWords: array (nullable = true)
//|   |-- element: string (containsNull = true)
//-- ngrams: array (nullable = true)
//|   |-- element: string (containsNull = false)

()
Confusion matrix:
2.0 0.0 1.0 0.0 |
4.0 0.0 0.0 1.0
0.0 1.0 1.0 1.0
1.0 1.0 0.0 1.0
Accuracy: 0.2857142857142857

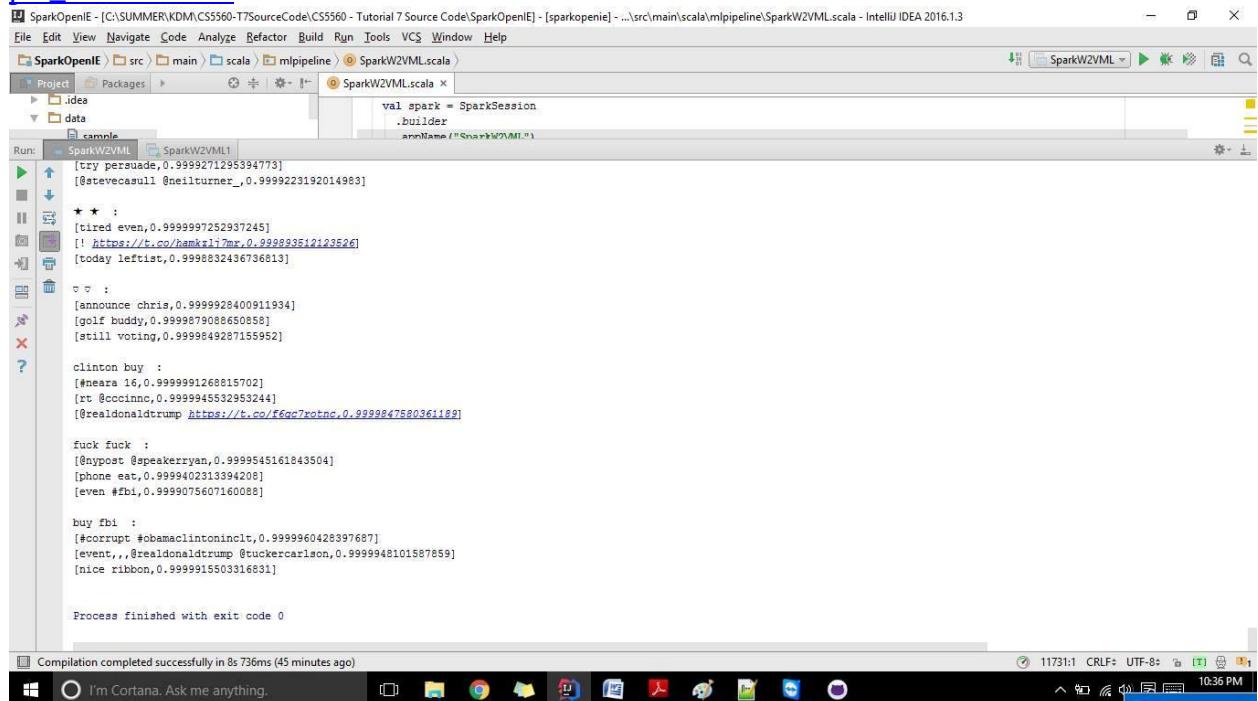
Process finished with exit code 0

```

■ NGram and Word2Vec:

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

- https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/output_word2vec.txt



```

SparkOpenE - [C:\SUMMER\KDM\CS5560-T7\SourceCode\CS5560 - Tutorial 7 Source Code] SparkOpenE - [sparkopenie] - ...\\src\\main\\scala\\mlpipeline\\SparkW2VML.scala - IntelliJ IDEA 2016.1.3
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
SparkOpenE > src > main > scala > mlpipeline > SparkW2VML.scala
Project Packages > SparkW2VML.scala x
Run: SparkW2VML SparkW2VML1
[try persuade,0.9999271295394773]
[@stevecasull @neilturner_,0.9999223192014983]
* * :
[tired even,0.999997252937245]
[! https://t.co/hanice17mr,0.999983510123526]
[today leftist,0.9998832436736513]
* * :
[announce chris,0.9999928400911934]
[golf buddy,0.9999879088650858]
[still voting,0.9999849287155952]
clinton buy :
[#neara 16,0.9999991268815702]
[rt @ccinnc,0.9999945532953244]
[@realdonaldtrump https://t.co/FKqc7rotnc,0.9999847580361189]
fuck fuck :
[@nypost @speakeerryan,0.9999545161843504]
[phone eat,0.9999402313394208]
[even #fbi,0.9999075607160088]
buy fbi :
[#corrupt @obamaclintoninc1t,0.999960428397687]
[event,, @realdonaldtrump @tuckercarlson,0.9999948101587859]
[nice ribbon,0.999915503316831]

Process finished with exit code 0

```

Compilation completed successfully in 8s 736ms (45 minutes ago)

I'm Cortana. Ask me anything.

Ontology Execution outputs:

https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/Ontology%20execution%20outputs.pdf

SOURCE CODE URL:

https://github.com/vilasmamidyla/KDM_SM16_SM/tree/master/Source/Spark2Ont

```

Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distsim.tagger ... done [1.0 sec].
root
|-- location: string (nullable = true)
|-- docs: string (nullable = true)
|-- rawTokens: array (nullable = true)
|   |-- element: string (containsNull = true)
|-- tokens: array (nullable = true)
|   |-- element: string (containsNull = true)
|-- features: vector (nullable = true)
|-- idfFeatures: vector (nullable = true)

```

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

Corpus summary:

Training set size: 2 documents
Vocabulary size: 13 terms
Training set size: 15 tokens
Preprocessing time: 0.558577732 sec

Finished training LDA model. Summary:

Training time: 4.352653153 sec
Training data average log likelihood: -20.115197054435626

3 topics:

president win usa election state currently majority female clinton lead oppostion trump virtue
usa win president virtue trump oppostion lead majority female currently clinton election state
usa win president clinton state lead currently election female majority virtue trump oppostion
0.0
0.0
0.0

SparkNaiveBayes

```
↑ holds ...
↓ Ontology Created
→ Process finished with exit code 0
↓
```

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

Election.java is where we write code for generating our ontology using the classes mentioned in the spark program using naïve Bayes algorithm

```
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Spark2Ont src main scala onthinterface Election
Project Packages SBT projects Art Build Maven Projects SBT
Election.java
public void saveOntology() {
    try {
        OWLOntologyManager manager = OWLManager.createOWLOntologyManager();
        OutputStream os = new FileOutputStream("data/Election2.owl");
        OWLXMLDocumentFormat owlxmlFormat = new OWLXMLDocumentFormat();
        manager.saveOntology(ont, owlxmlFormat, os);
        System.out.println("Ontology Created");
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}
```

The screenshot shows the IntelliJ IDEA interface with the following details:

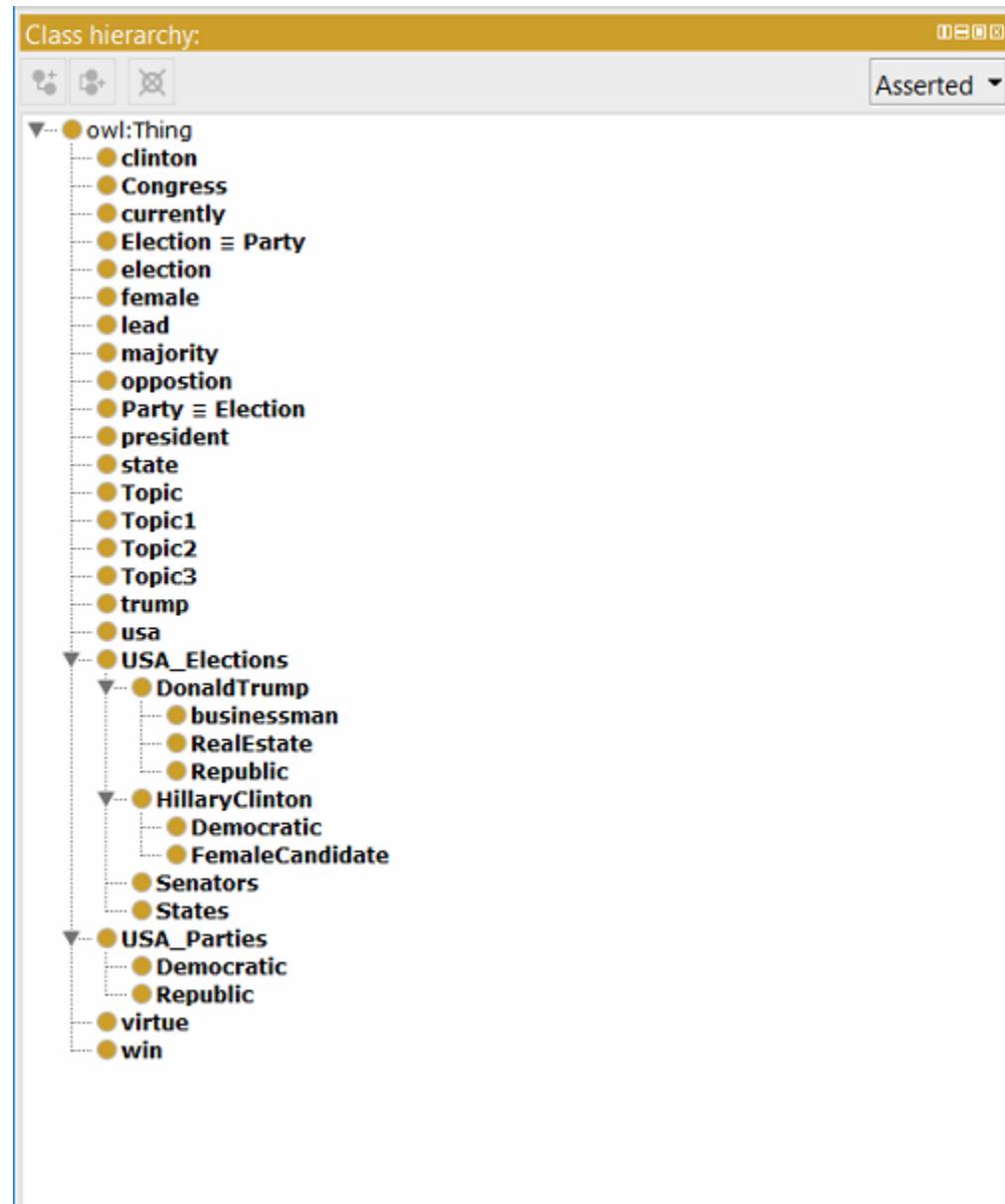
- Project Structure:** Shows the project tree with modules like data, Categories, Clinton, Trump, test, and catalog-v001.xml.
- Code Editor:** Displays the `Election.java` file containing Java code for saving an ontology to a file named `Election2.owl`.
- Run Tool Window:** Shows the output of the run command:
 - Output: "Ontology Created"
 - Status: "Process finished with exit code 0"

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

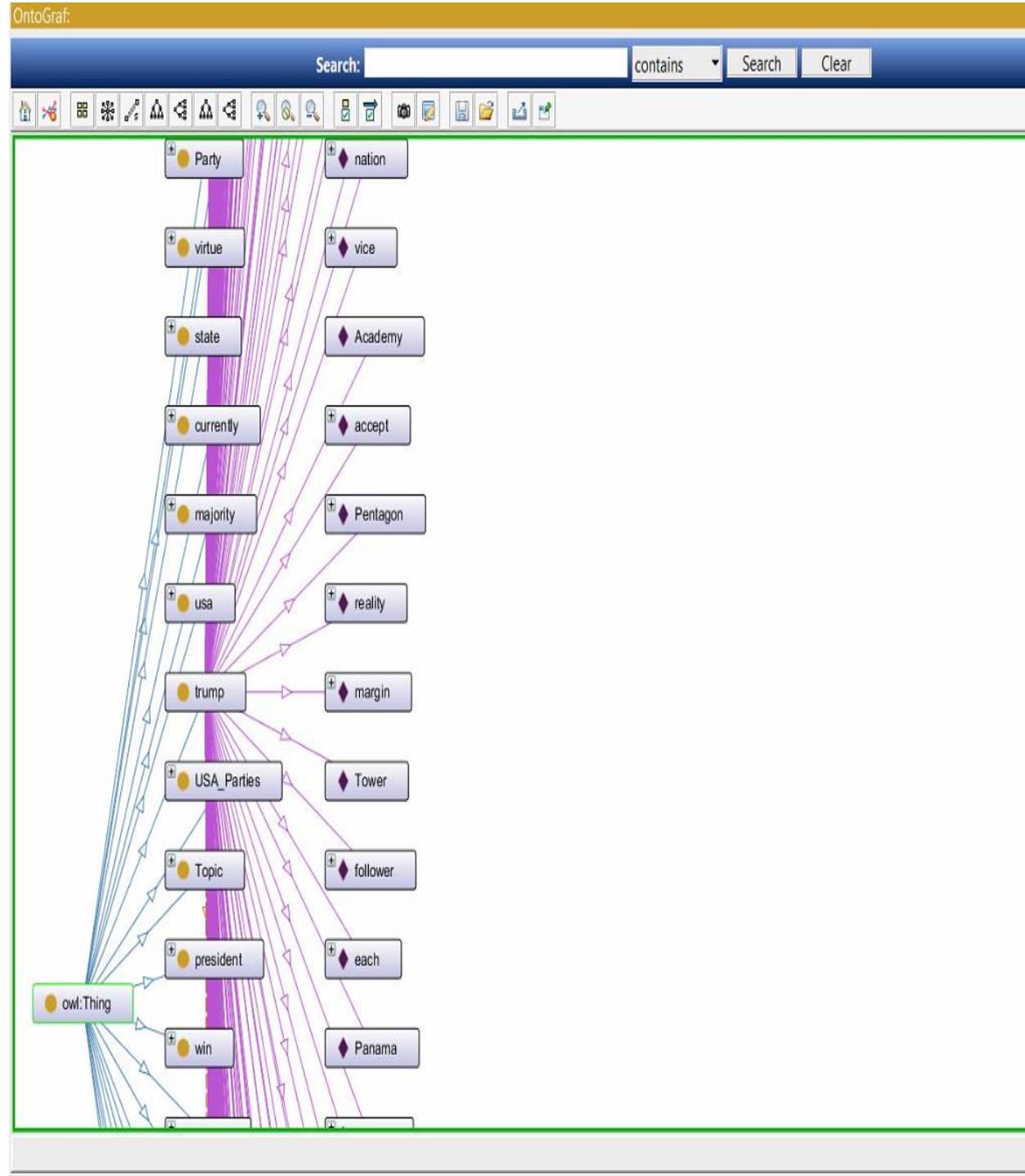
Ontology Election OWL File in Protégé Outputs:

https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/Ontoin%20protege_oututs.pdf

Class Hierarchy:

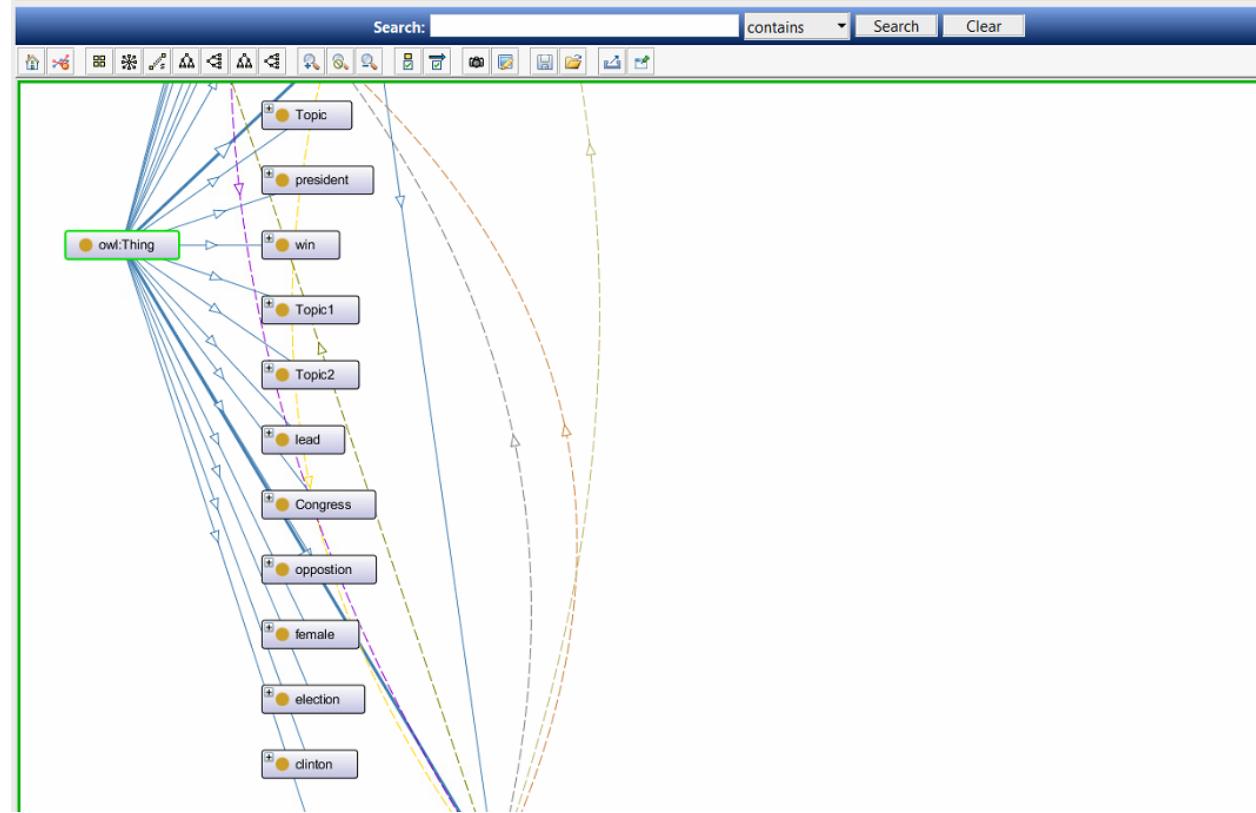


TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)
Ontograpf for our project Ontology:

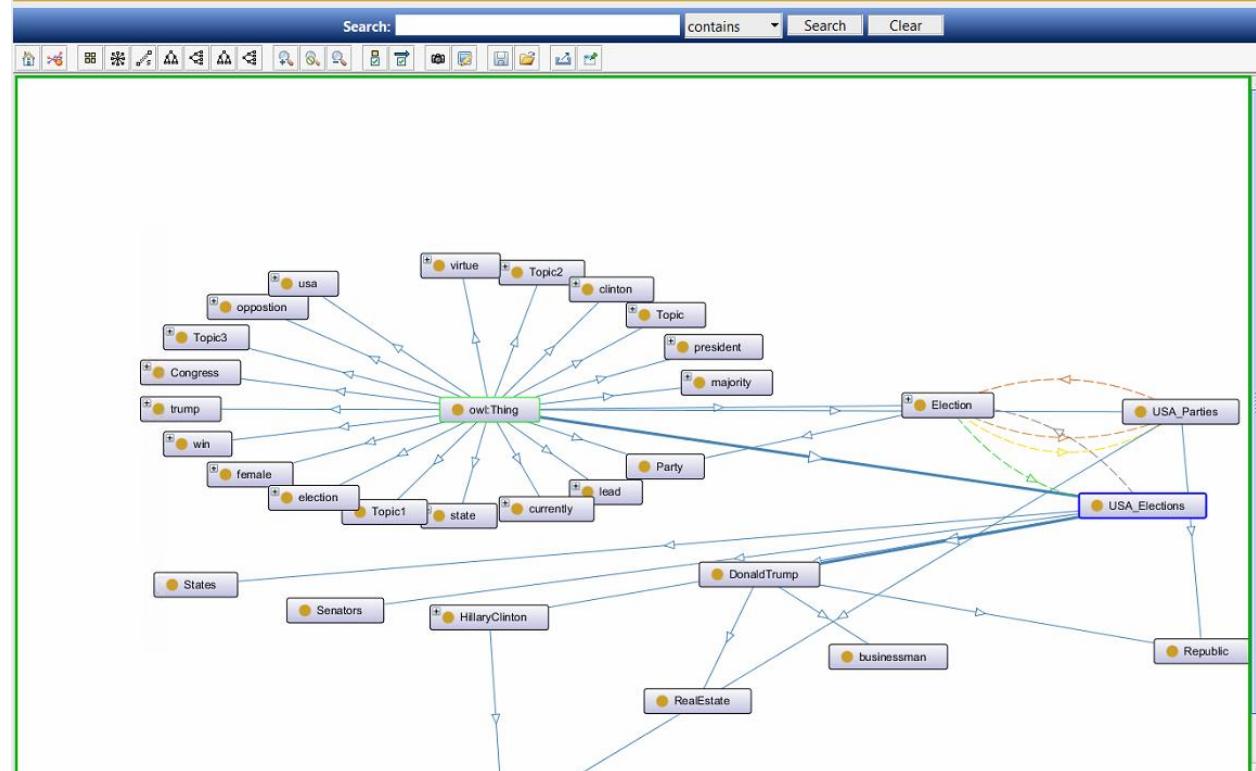


TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

OntoGraf:



OntoGraf:



TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

Ontology Metrics:

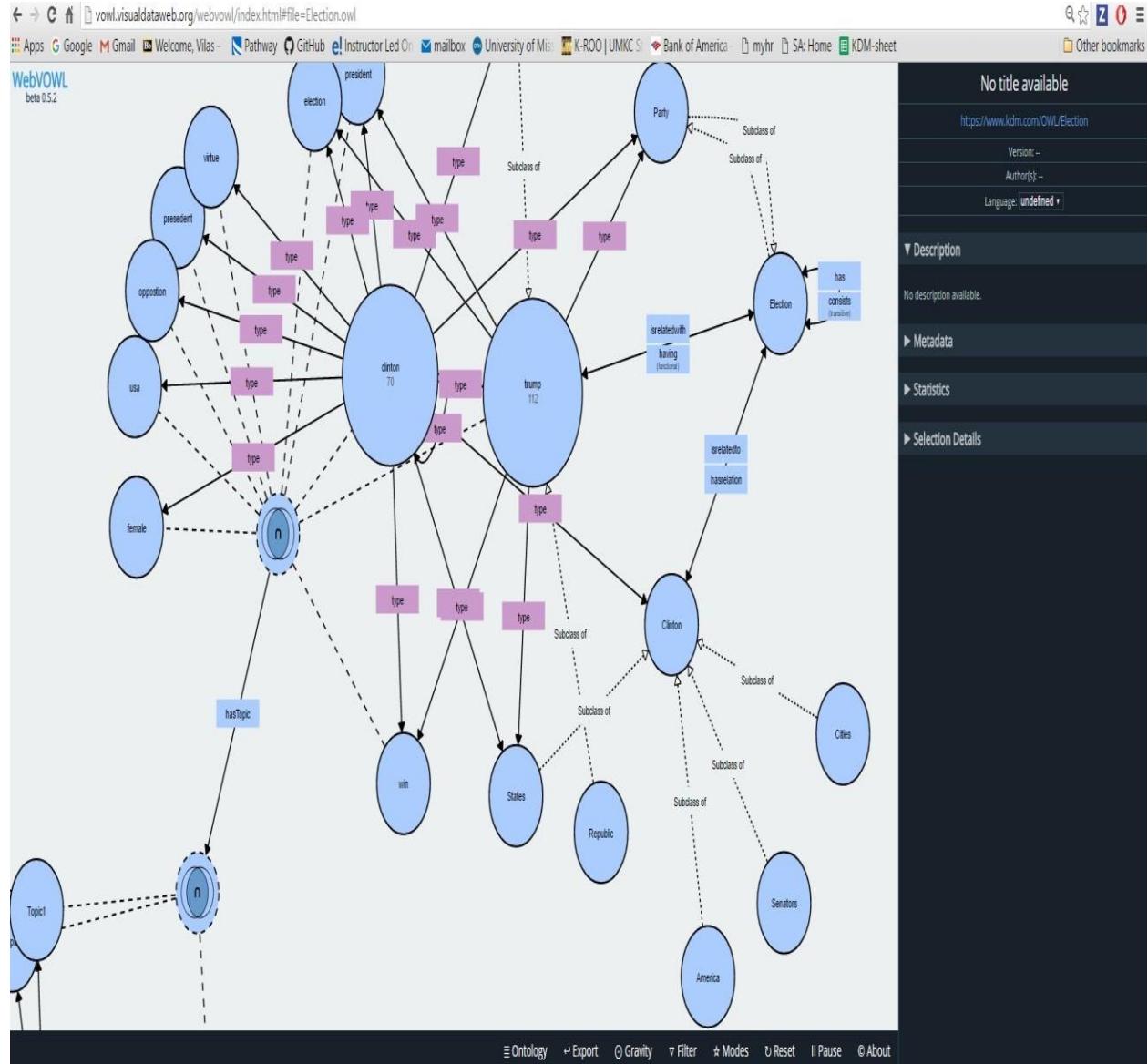
Ontology metrics:	
Metrics	
Axiom	36154
Logical axiom count	18276
Declaration axioms count	17877
Class count	31
Object property count	7
Data property count	1
Individual count	17838
DL expressivity	SHIF(D)
Class axioms	
SubClassOf	15
EquivalentClasses	0
DisjointClasses	1
GCI count	0
Hidden GCI Count	0
Object property axioms	
SubObjectPropertyOf	4
EquivalentObjectProperties	0
InverseObjectProperties	3
DisjointObjectProperties	0
FunctionalObjectProperty	1
InverseFunctionalObjectProperty	0
TransitiveObjectProperty	1

TEAM 8 : VILAS MAMIDYALA (18)

VIKESH PADARTHI (27)

DINESH KUMAR BANDAM (2)

BHUMIREDDY RANJITHA REDDY (4)

Web View Of our OWL file:

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

Apache jena fuseki and sparql:**1. Sample Select query and its output in Table Response format**

The screenshot shows a web browser window with the URL `localhost:3030/dataset.html?tab=upload&ds=/election`. The top part of the page contains a code editor with a SPARQL query:

```

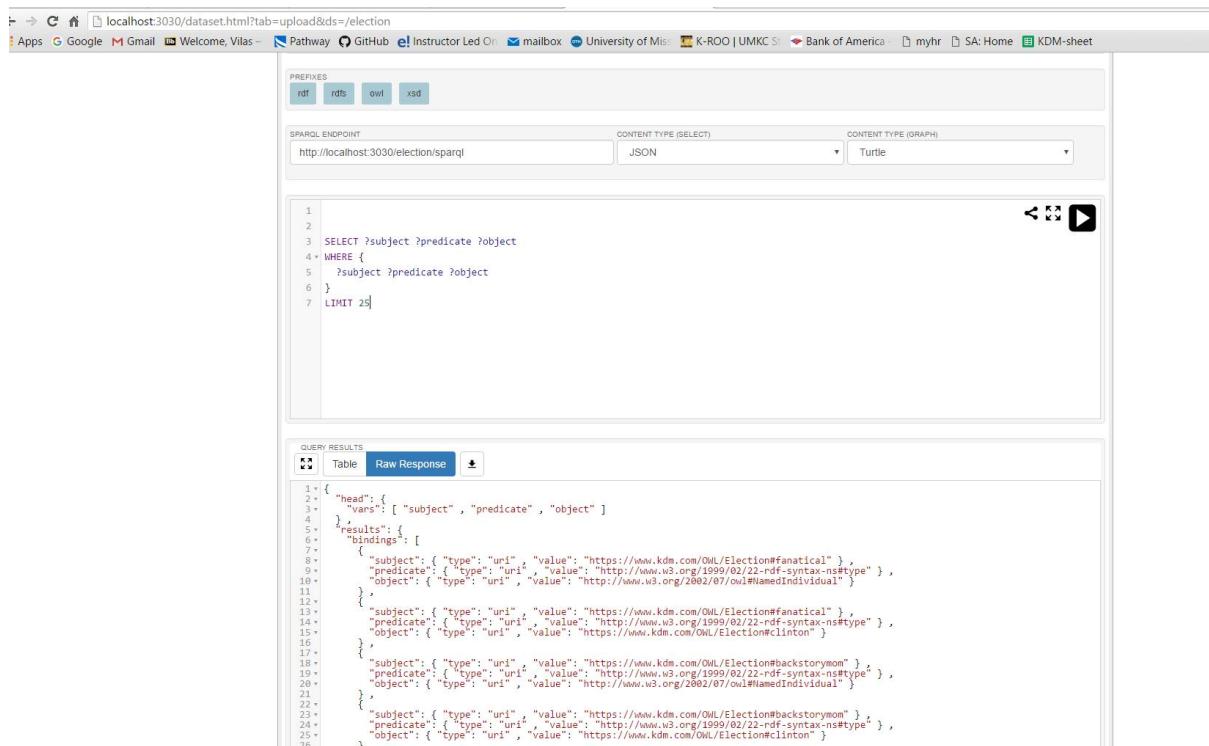
1
2
3 SELECT ?subject ?predicate ?object
4 WHERE {
5   ?subject ?predicate ?object
6 }
7 LIMIT 25
  
```

Below the code editor is a table titled "QUERY RESULTS". The table has three columns: "subject", "predicate", and "object". It displays 13 rows of data, each corresponding to a triple from the dataset. The "subject" column contains URIs like `<https://www.kdm.com/OWL/Election#fanatical>`, the "predicate" column contains URIs like `<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>`, and the "object" column contains URIs like `<http://www.w3.org/2002/07/owl#NamedIndividual>`.

subject	predicate	object
1 <https://www.kdm.com/OWL/Election#fanatical>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2002/07/owl#NamedIndividual>
2 <https://www.kdm.com/OWL/Election#fanatical>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<https://www.kdm.com/OWL/Election#clinton>
3 <https://www.kdm.com/OWL/Election#backstorymom>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2002/07/owl#NamedIndividual>
4 <https://www.kdm.com/OWL/Election#backstorymom>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<https://www.kdm.com/OWL/Election#clinton>
5 <https://www.kdm.com/OWL/Election#clioy>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2002/07/owl#NamedIndividual>
6 <https://www.kdm.com/OWL/Election#clioy>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<https://www.kdm.com/OWL/Election#clinton>
7 <https://www.kdm.com/OWL/Election#jeremiahwright>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2002/07/owl#NamedIndividual>
8 <https://www.kdm.com/OWL/Election#jeremiahwright>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<https://www.kdm.com/OWL/Election#clinton>
9 <https://www.kdm.com/OWL/Election#bnjby>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2002/07/owl#NamedIndividual>
10 <https://www.kdm.com/OWL/Election#bnjby>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<https://www.kdm.com/OWL/Election#clinton>
11 <https://www.kdm.com/OWL/Election#weak>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2002/07/owl#NamedIndividual>
12 <https://www.kdm.com/OWL/Election#weak>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<https://www.kdm.com/OWL/Election#clinton>
13 <https://www.kdm.com/OWL/Election#ftzil>	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	<http://www.w3.org/2002/07/owl#NamedIndividual>

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

2. Sample select query and its output in Raw Response format



The screenshot shows a web-based SPARQL endpoint interface. At the top, there are tabs for 'PREFIXES' (with options for rdf, rdfs, owl, xsd), 'SPARQL ENDPOINT' (set to http://localhost:3030/election/sparql), 'CONTENT TYPE (SELECT)' (set to JSON), and 'CONTENT TYPE (GRAPH)' (set to Turtle). Below this, the query text is displayed:

```

1
2
3 SELECT ?subject ?predicate ?object
4 WHERE {
5   ?subject ?predicate ?object
6 }
7 LIMIT 25

```

Below the query text is a 'QUERY RESULTS' section. It includes tabs for 'Table' and 'Raw Response'. The 'Raw Response' tab is selected, showing the following JSON output:

```

1 {
2   "head": {
3     "vars": [ "subject" , "predicate" , "object" ]
4   },
5   "results": {
6     "bindings": [
7       {
8         "subject": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#fanatical" },
9         "predicate": { "type": "uri" , "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type" },
10        "object": { "type": "uri" , "value": "http://www.w3.org/2002/07/owl#NamedIndividual" }
11      },
12      {
13        "subject": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#fanatical" },
14        "predicate": { "type": "uri" , "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type" },
15        "object": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#clinton" }
16      },
17      {
18        "subject": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#backstorymom" },
19        "predicate": { "type": "uri" , "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type" },
20        "object": { "type": "uri" , "value": "http://www.w3.org/2002/07/owl#NamedIndividual" }
21      },
22      {
23        "subject": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#backstorymom" },
24        "predicate": { "type": "uri" , "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type" },
25        "object": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#clinton" }
26      }
    ]
  }
}

```

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

3. Select query on subjects Clinton, trump and using predicate Election

The screenshot shows a web browser window with the URL `localhost:3030/dataset.html?tab=upload&ds=/election`. The query entered is:

```

1+
2+
3+ SELECT ?clinton ?trump
4+ WHERE {
5+   ?clinton ?election ?trump
6+ }
7+ LIMIT 25

```

The results section displays the query results in JSON format:

```

1+ {
2+   "head": {
3+     "vars": [ "clinton" , "trump" ]
4+   },
5+   "results": {
6+     "bindings": [
7+       {
8+         "clinton": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#fanatical" } ,
9+         "trump": { "type": "uri" , "value": "http://www.w3.org/2002/07/owl#NamedIndividual" }
10+      },
11+      {
12+        "clinton": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#fanatical" } ,
13+        "trump": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#clinton" }
14+      },
15+      {
16+        "clinton": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#backstorymom" } ,
17+        "trump": { "type": "uri" , "value": "http://www.w3.org/2002/07/owl#NamedIndividual" }
18+      },
19+      {
20+        "clinton": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#backstorymom" } ,
21+        "trump": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#clinton" }
22+      },
23+      {
24+        "clinton": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#ci้อย" } ,
25+        "trump": { "type": "uri" , "value": "http://www.w3.org/2002/07/owl#NamedIndividual" }
26+      },
27+      {
28+        "clinton": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#clioy" } ,
29+        "trump": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#clinton" }
30+      },
31+      {
32+        "clinton": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#jeremiahwright" } ,
33+        "trump": { "type": "uri" , "value": "http://www.w3.org/2002/07/owl#NamedIndividual" }
34+      },
35+      {
36+        "clinton": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#jeremiahwright" } ,
37+        "trump": { "type": "uri" , "value": "https://www.kdm.com/OWL/Election#clinton" }
38+      }
39+

```

Naïve Bayes Approach for Twitter Sentimental Analysis :

https://github.com/vilasmamidyla/KDM_SM16_SM/tree/master/Source/NaiveBayes

Naive Bayes Model output:

The below images refers to the training data output. We are here first training the model by giving it both positive tweets and negative tweets. Once we execute the program we will get the below output.

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTH (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

The screenshot shows the IntelliJ IDEA interface with the following details:

- Project Structure:** The left sidebar shows the project structure under "Spark2Ont [root]".
- Code Editor:** The main window displays the file `SparkNaiveBayes.scala` with the following code snippet:

```
val x=testDir.toString.split("\n")  
//val topicData = SparkLDAMain.main(sc, testDir, 3, "es")  
  
val testFV = getTFIDFVector(sc, x)  
  
println("testFV")  
testFV.foreach(f=>println(f))  
  
val result = model.predict(testFV)
```
- Run Log:** The bottom-left panel shows the run log with several SLF4J messages indicating failed class loading and defaulting to no-operation logger implementation.
- File Status:** The bottom status bar indicates "All files are up-to-date (28 minutes ago)".
- System Tray:** The bottom right shows system icons for battery, signal, and time (5:32 PM).

COMP-SCI 5560 (SUMMER 2016) – KNOWLEDGE DISCOVERY AND MANAGEMENT

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTH (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

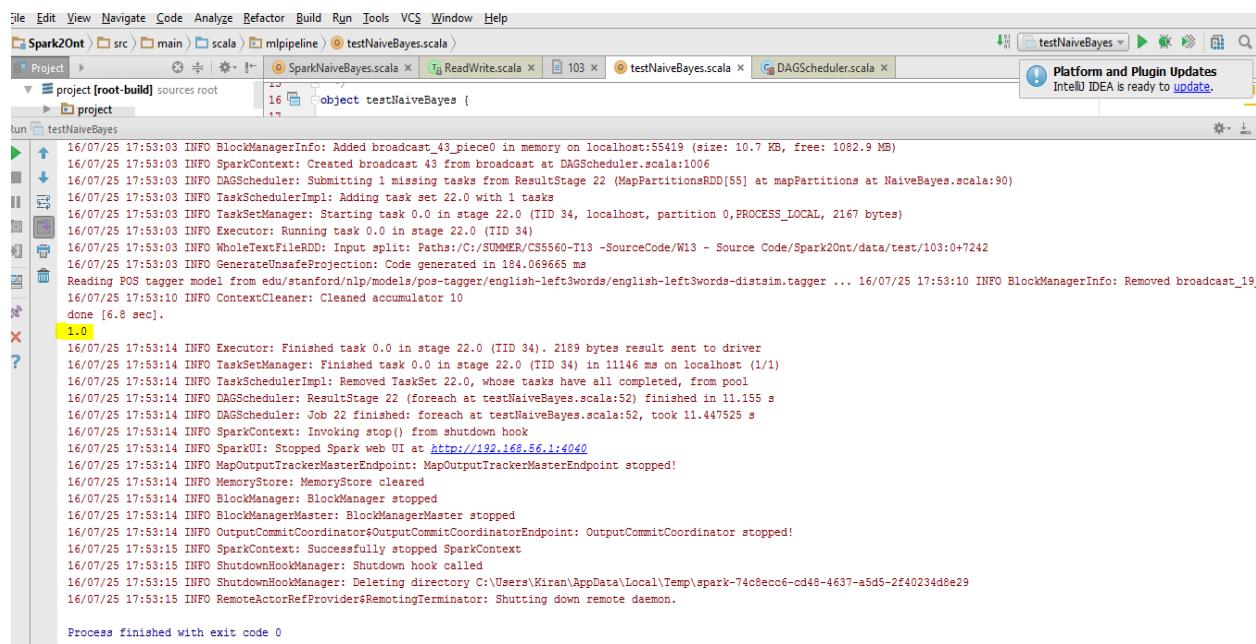
TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

After training this model, we have checked the working of this model by giving the positive tweets and negative tweets as input. It worked perfectly. Below image is the example of Negative tweets. Here we are getting 1.0 as output which tells us the file contains negative tweets.

Output:

https://github.com/vilasmamidyla/KDM_SM16_SM/blob/master/Sampleoutputs/Twitter_Sentiment%20_Analysis.pdf

when negative tweets as given input:



```

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Spark2OnT > src > main > scala > mlpipeline > testNaiveBayes.scala
Project > project [root-build] sources root > project > project
16/07/25 17:53:03 INFO BlockManagerInfo: Added broadcast_43_piece0 in memory on localhost:55419 (size: 10.7 KB, free: 1082.9 MB)
16/07/25 17:53:03 INFO SparkContext: Created broadcast 43 from broadcast at DAGScheduler.scala:1006
16/07/25 17:53:03 INFO DAGScheduler: Submitting 1 missing tasks from ResultStage 22 (MapPartitionsRDD[55] at mapPartitions at NaiveBayes.scala:90)
16/07/25 17:53:03 INFO TaskSchedulerImpl: Adding task set 22.0 with 1 tasks
16/07/25 17:53:03 INFO TaskSetManager: Starting task 0.0 in stage 22.0 (TID 34, localhost, partition 0, PROCESS_LOCAL, 2167 bytes)
16/07/25 17:53:03 INFO Executor: Running task 0.0 in stage 22.0 (TID 34)
16/07/25 17:53:03 INFO WholeTextFileRDD: Input split: Paths:/C:/SUMMER/C55560-T13 -SourceCode/W13 - Source Code/Spark2OnT/data/test/103:0+7242
16/07/25 17:53:03 INFO GenerateOnsiefeProjection: Code generated in 184.069665 ms
Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distsim.tagger ...
16/07/25 17:53:10 INFO BlockManagerInfo: Removed broadcast_19
16/07/25 17:53:10 INFO ContextCleaner: Cleaned accumulator 10
done [6.8 sec].
1.0
16/07/25 17:53:14 INFO Executor: Finished task 0.0 in stage 22.0 (TID 34). 2189 bytes result sent to driver
16/07/25 17:53:14 INFO TaskSetManager: Finished task 0.0 in stage 22.0 (TID 34) in 11146 ms on localhost (1/1)
16/07/25 17:53:14 INFO TaskSchedulerImpl: Removed TaskSet 22.0, whose tasks have all completed, from pool
16/07/25 17:53:14 INFO DAGScheduler: ResultStage 22 (foreach at testNaiveBayes.scala:52) finished in 11.155 s
16/07/25 17:53:14 INFO DAGScheduler: Job 22 finished: foreach at testNaiveBayes.scala:52, took 11.447525 s
16/07/25 17:53:14 INFO SparkContext: Invoking stop() from shutdown hook
16/07/25 17:53:14 INFO SparkUI: Stopped Spark web UI at http://192.168.56.1:4040
16/07/25 17:53:14 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
16/07/25 17:53:14 INFO MemoryStore: MemoryStore cleared
16/07/25 17:53:14 INFO BlockManager: BlockManager stopped
16/07/25 17:53:14 INFO BlockManagerMaster: BlockManagerMaster stopped
16/07/25 17:53:14 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
16/07/25 17:53:15 INFO SparkContext: Successfully stopped SparkContext
16/07/25 17:53:15 INFO ShutdownHookManager: Shutdown hook called
16/07/25 17:53:15 INFO ShutdownHookManager: Deleting directory C:\Users\Kiran\AppData\Local\Temp\spark-74c8ecc6-cd48-4637-a5d5-2f40234d8e29
16/07/25 17:53:15 INFO RemoteActorRefProvider$RemotingTerminator: Shutting down remote daemon.

Process finished with exit code 0

```

we can see 1.0 as output.

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

when positive tweets are given as input

```

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Spark2Ont src main scala mlpipeline testNaiveBayes.scala
Project sources root target
Run testNaiveBayes
16/07/25 17:58:51 INFO TaskSchedulerImpl: Adding task set 22.0 with 1 tasks
16/07/25 17:58:51 INFO TaskSetManager: Starting task 0.0 in stage 22.0 (TID 34, localhost, partition 0,PROCESS_LOCAL, 2167 bytes)
16/07/25 17:58:51 INFO Executor: Running task 0.0 in stage 22.0 (TID 34)
16/07/25 17:58:51 INFO WholeTextFileRDD: Input split: Paths:/C:/SUMMER/CSS5560-T13 -SourceCode/W13 - Source Code/Spark2Ont/data/test/103:0+10766
16/07/25 17:58:51 INFO GenerateUnsafeProjection: Code generated in 80.418579 ms
Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distsim.tagger ... 16/07/25 17:58:54 INFO BlockManagerInfo: Removed broadcast_41
16/07/25 17:58:54 INFO ContextCleaner: Cleaned accumulator 24
16/07/25 17:58:54 INFO BlockManagerInfo: Removed broadcast_40_piece0 on localhost:55856 in memory (size: 13.8 KB, free: 1082.9 MB)
done [2.7 sec]
16/07/25 17:58:58 INFO Executor: Finished task 0.0 in stage 22.0 (TID 34). 2189 bytes result sent to driver
0.0
16/07/25 17:58:58 INFO TaskSetManager: Finished task 0.0 in stage 22.0 (TID 34) in 7064 ms on localhost (1/1)
16/07/25 17:58:58 INFO TaskSchedulerImpl: Removed TaskSet 22.0, whose tasks have all completed, from pool
16/07/25 17:58:58 INFO DAGScheduler: ResultStage 22 (foreach at testNaiveBayes.scala:52) finished in 7.067 s
16/07/25 17:58:58 INFO DAGScheduler: Job 22 finished: foreach at testNaiveBayes.scala:52, took 7.352581 s
16/07/25 17:58:58 INFO SparkContext: Invoking stop() from shutdown hook
16/07/25 17:58:58 INFO SparkUI: Stopped Spark web UI at http://192.168.56.1:4040
16/07/25 17:58:58 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
16/07/25 17:58:58 INFO MemoryStore: MemoryStore cleared
16/07/25 17:58:58 INFO BlockManager: BlockManager stopped
16/07/25 17:58:58 INFO BlockManagerMaster: BlockManagerMaster stopped
16/07/25 17:58:58 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
16/07/25 17:58:58 INFO SparkContext: Successfully stopped SparkContext
16/07/25 17:58:58 INFO ShutdownHookManager: Shutdown hook called
16/07/25 17:58:58 INFO ShutdownHookManager: Deleting directory C:\Users\Kiran\AppData\Local\Temp\spark-4f0b12a9-09f1-4ef5-a0b5-388c584b97b2
16/07/25 17:58:58 INFO RemoteActorRefProvider$RemotingTerminator: Shutting down remote daemon.

Process finished with exit code 0

```

we can see we got 0.0 as output

Now we have given 30 positive and 30 negative tweets as input to know how this model works. It has given output as 1.0 which tells us that if the file contains negative data with equal number of positive tweets, it considers as negative.

```

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Spark2Ont src main scala mlpipeline testNaiveBayes.scala
Project sources root target build.properties
Run testNaiveBayes
16/07/25 18:03:24 INFO BlockManagerInfo: Added broadcast_43_piece0 in memory on localhost:56217 (size: 10.7 KB, free: 1082.9 MB)
16/07/25 18:03:24 INFO SparkContext: Created broadcast 43 from broadcast at DAGScheduler.scala:1006
16/07/25 18:03:24 INFO DAGScheduler: Submitting 1 missing tasks from ResultStage 22 (MapPartitionsRDD[55] at mapPartitions at NaiveBayes.scala:90)
16/07/25 18:03:24 INFO TaskSchedulerImpl: Adding task set 22.0 with 1 tasks
16/07/25 18:03:24 INFO TaskSetManager: Starting task 0.0 in stage 22.0 (TID 34, localhost, partition 0,PROCESS_LOCAL, 2167 bytes)
16/07/25 18:03:24 INFO Executor: Running task 0.0 in stage 22.0 (TID 34)
16/07/25 18:03:25 INFO WholeTextFileRDD: Input split: Paths:/C:/SUMMER/CSS5560-T13 -SourceCode/W13 - Source Code/Spark2Ont/data/test/103:0+6752
16/07/25 18:03:25 INFO GenerateUnsafeProjection: Code generated in 230.297384 ms
Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distsim.tagger ... done [6.7 sec].
16/07/25 18:03:38 INFO Executor: Finished task 0.0 in stage 22.0 (TID 34). 2189 bytes result sent to driver
16/07/25 18:03:38 INFO TaskSetManager: Finished task 0.0 in stage 22.0 (TID 34) in 13807 ms on localhost (1/1)
16/07/25 18:03:38 INFO TaskSchedulerImpl: Removed TaskSet 22.0, whose tasks have all completed, from pool
16/07/25 18:03:38 INFO DAGScheduler: ResultStage 22 (foreach at testNaiveBayes.scala:52) finished in 13.814 s
16/07/25 18:03:38 INFO DAGScheduler: Job 22 finished: foreach at testNaiveBayes.scala:52, took 14.209522 s
1.0
16/07/25 18:03:38 INFO SparkContext: Invoking stop() from shutdown hook
16/07/25 18:03:38 INFO SparkUI: Stopped Spark web UI at http://192.168.56.1:4040
16/07/25 18:03:39 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
16/07/25 18:03:39 INFO MemoryStore: MemoryStore cleared
16/07/25 18:03:39 INFO BlockManager: BlockManager stopped
16/07/25 18:03:39 INFO BlockManagerMaster: BlockManagerMaster stopped
16/07/25 18:03:39 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
16/07/25 18:03:39 INFO SparkContext: Successfully stopped SparkContext
16/07/25 18:03:39 INFO ShutdownHookManager: Shutdown hook called
16/07/25 18:03:39 INFO ShutdownHookManager: Deleting directory C:\Users\Kiran\AppData\Local\Temp\spark-458c08b0-dcd7-45d7-a3c5-25d438747b21

Process finished with exit code 0

```

We are getting 1.0 as output which says us that data is having /having meaning like negative tweets.

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

when positive 50 negative 30 tweets are given as input

```

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Spark2Ont src main scala mlpipeline testNaiveBayes.scala
Project [root-build] sources root
  project
  target
Run testNaiveBayes
16/07/25 18:08:34 INFO WholeTextFileRDD: Input split: Paths:/C:/SUMMER/CS5560-T13 -SourceCode/W13 - Source Code/Spark2Ont/data/test/103:0+8967
16/07/25 18:08:34 INFO GenerateUnsafeProjection: Code generated in 167.563577 ms
Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distsim.tagger ...
16/07/25 18:08:38 INFO ContextCleaner: Cleared accumulator 24
16/07/25 18:08:38 INFO BlockManagerInfo: Removed broadcast_40_piece0 on localhost:56735 in memory (size: 13.8 KB, free: 1082.9 MB)
16/07/25 18:08:38 INFO BlockManagerInfo: Removed broadcast_39_piece0 on localhost:56735 in memory (size: 15.1 KB, free: 1082.9 MB)
16/07/25 18:08:38 INFO ContextCleaner: Cleared accumulator 23
done [4.8 sec].
16/07/25 18:08:47 INFO BlockManagerInfo: Removed broadcast_28_piece0 on localhost:56735 in memory (size: 3.3 KB, free: 1082.9 MB)
16/07/25 18:08:47 INFO ContextCleaner: Cleared accumulator 15
16/07/25 18:08:47 INFO BlockManagerInfo: Removed broadcast_27_piece0 on localhost:56735 in memory (size: 13.8 KB, free: 1082.9 MB)
0.0
16/07/25 18:08:47 INFO Executor: Finished task 0.0 in stage 22.0 (TID 34). 2189 bytes result sent to driver
16/07/25 18:08:47 INFO TaskSetManager: Finished task 0.0 in stage 22.0 (TID 34) in 13514 ms on localhost (1/1)
16/07/25 18:08:47 INFO TaskSchedulerImpl: Removed TaskSet 22.0, whose tasks have all completed, from pool
16/07/25 18:08:47 INFO DAGScheduler: ResultStage 22 (foreach at testNaiveBayes.scala:52) finished in 13.521 s
16/07/25 18:08:47 INFO DAGScheduler: Job 22 finished: foreach at testNaiveBayes.scala:52, took 13.814247 s
16/07/25 18:08:47 INFO SparkContext: Invoking stop() from shutdown hook
16/07/25 18:08:47 INFO SparkUI: Stopped Spark web UI at http://192.168.56.1:4040
16/07/25 18:08:47 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
16/07/25 18:08:47 INFO MemoryStore: MemoryStore cleared
16/07/25 18:08:47 INFO BlockManager: BlockManager stopped
16/07/25 18:08:47 INFO BlockManagerMaster: BlockManagerMaster stopped
16/07/25 18:08:47 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
16/07/25 18:08:47 INFO SparkContext: Successfully stopped SparkContext
16/07/25 18:08:47 INFO ShutdownHookManager: Shutdown hook called
16/07/25 18:08:47 INFO ShutdownHookManager: Deleting directory C:/Users/Kiran/AppData/Local/Temp/spark-b2bf8673-9c2f-4d48-9ea5-3d6b03df2524

Process finished with exit code 0

```

we can see that output is 0.0 which tells us that here positive tweets are more in number and it is domination the negative tweets.

when negative 50 positive 30 is given as input

```

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Spark2Ont src main scala mlpipeline testNaiveBayes.scala
Project [root-build] sources root
  project
  target
  build.properties
  plugins.sbt
Run testNaiveBayes
16/07/25 18:13:09 INFO SparkContext: Created broadcast 43 from broadcast at DAGScheduler.scala:1006
16/07/25 18:13:09 INFO DAGScheduler: Submitting 1 missing tasks from ResultStage 22 (MapPartitionsRDD[55] at mapPartitions at NaiveBayes.scala:90)
16/07/25 18:13:09 INFO TaskSchedulerImpl: Adding task set 22.0 with 1 tasks
16/07/25 18:13:09 INFO TaskSetManager: Starting task 0.0 in stage 22.0 (TID 34, localhost, partition 0,PROCESS_LOCAL, 2167 bytes)
16/07/25 18:13:09 INFO Executor: Running task 0.0 in stage 22.0 (TID 34)
16/07/25 18:13:09 INFO WholeTextFileRDD: Input split: Paths:/C:/SUMMER/CS5560-T13 -SourceCode/W13 - Source Code/Spark2Ont/data/test/103:0+8839
16/07/25 18:13:09 INFO GenerateUnsafeProjection: Code generated in 165.796989 ms
Reading POS tagger model from edu/stanford/nlp/models/pos-tagger/english-left3words/english-left3words-distsim.tagger ...
16/07/25 18:13:18 INFO Executor: Finished task 0.0 in stage 22.0 (TID 34). 2189 bytes result sent to driver
1.0
16/07/25 18:13:18 INFO TaskSetManager: Finished task 0.0 in stage 22.0 (TID 34) in 8601 ms on localhost (1/1)
16/07/25 18:13:18 INFO TaskSchedulerImpl: Removed TaskSet 22.0, whose tasks have all completed, from pool
16/07/25 18:13:18 INFO DAGScheduler: ResultStage 22 (foreach at testNaiveBayes.scala:52) finished in 8.605 s
16/07/25 18:13:18 INFO DAGScheduler: Job 22 finished: foreach at testNaiveBayes.scala:52, took 9.371377 s
16/07/25 18:13:18 INFO SparkContext: Invoking stop() from shutdown hook
16/07/25 18:13:18 INFO SparkUI: Stopped Spark web UI at http://192.168.56.1:4040
16/07/25 18:13:18 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
16/07/25 18:13:18 INFO MemoryStore: MemoryStore cleared
16/07/25 18:13:18 INFO BlockManager: BlockManager stopped
16/07/25 18:13:18 INFO BlockManagerMaster: BlockManagerMaster stopped
16/07/25 18:13:22 INFO SparkContext: Successfully stopped SparkContext
16/07/25 18:13:22 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
16/07/25 18:13:22 INFO ShutdownHookManager: Shutdown hook called
16/07/25 18:13:22 INFO ShutdownHookManager: Deleting directory C:/Users/Kiran/AppData/Local/Temp/spark-e383bc6c-36c5-48fd-a30b8b438860

Process finished with exit code 0

```

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

we can see that output is 1.0 which tells us that here positive tweets are more in number and it is domination the negative tweets.

Summarization outputs:

https://github.com/vilasmamidyala/KDM_SM16_SM/blob/master/Sampleoutputs/Summarization_output.pdf

Source code URL:

https://github.com/vilasmamidyala/KDM_SM16_SM/tree/master/Source/Spark2Ont/src/main/java

Summarization For a given URL:

```
Url_summary
-----
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/fag.html#noconfig for more info.
Important summary the URL:

No matter where you are, you can help Hillary win in November.
and you deserve a country and a president and Commander-in-Chief who honors your service."
<b>People</b> fought and bled and died for them.
I don't understand <b>people</b> who trash talk about Americaâ¢ who act as if we are not the greatest country that has ever been created.â¢
"Americans aren't just choosing a president, we're also choosing a Commander-in-Chief."
<b>People</b> had some fun with a new game we invented: Sad!
Virginians have known and loved @timkaine for decadesâ¢ and we're proud to have him on <b>Team Hillary: hrc</b>.io/2a9MmqX
Our goal: 3 million <b>people</b> to register and commit to vote by Nov. 8
We stand with the <b>Afghan people</b> against terror.
Tough times don't last, but <b>tough people</b> do."
@TimKaine in his first speech as Hillary's running mate

Process finished with exit code 0
```

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

Spark2Ont [M:\Spark2Ont] - [root] - ..\src\main\java\Url_summary.java - IntelliJ IDEA 2016.2

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

Spark2Ont > src > main > java > Url_summary

Url_summary.java

```

1:  package com.intellexer.spark2ont;
2:
3:  import org.json.JSONArray;
4:  import org.json.JSONObject;
5:
6:  import java.io.BufferedReader;
7:  import java.io.InputStreamReader;
8:  import java.net.HttpURLConnection;
9:  import java.net.URL;
10: import java.util.StringBuilder;
11:
12: public class Url_summary {
13:
14:     public static void main() {
15:         HttpURLConnection client = null;
16:         URL url = null;
17:         BufferedReader rd = null;
18:         String line = null;
19:         StringBuilder s = new StringBuilder();
20:
21:         try {
22:             url = new URL("http://api.intellexer.com/summarize?apikey=bbbfb162-ee2e-4c5b-8945-45ded0377a98");
23:             client = (HttpURLConnection) url.openConnection();
24:             rd = new BufferedReader(new InputStreamReader(response.getEntity().getContent()));
25:             line = rd.readLine();
26:
27:             while ((line = rd.readLine()) != null) {
28:                 //System.out.println(line);
29:                 s.append(line);
30:
31:             }
32:             String output = s.toString();
33:             JSONObject j = new JSONObject(output);
34:             JSONArray jarray = j.getJSONArray("items");
35:
36:             String result = "";
37:
38:             for (int i = 0; i < jarray.length(); i++) {
39:                 result = result + jarray.getJSONObject(i).getString("text") + "\n";
40:             }
41:         } catch (Exception e) {
42:             e.printStackTrace();
43:         }
44:     }
45: }

```

Run Url_summary

log4j:WARN Please initialize the log4j system properly.
log4j:WARN See <http://logging.apache.org/log4j/1.2/faq.html#noconfig> for more info.

Important summary the URL:

No matter where you are, you can help Hillary win in November.
and you deserve a country and a president and Commander-in-Chief who honors your service."
< b>People fought and bled and died for them.
â I donâ t understand **< b>people** who trash talk about Americaâ who act as if we are not the greatest country that has ever been created.â
"Americans arenâ t just choosing a president, weâ re also choosing a Commander-in-Chief."
< b>People had some fun with a new game we invented: Sad!
Virginians have known and loved @timkaine for decadesâ and we're proud to have him on **< b>Team Hillary: hrc**.io/2a9MmqX
Our goal: 3 million **< b>people** to register and commit to vote by Nov. 8
We stand with the **< b>Afghan people** against terror.
"ough times don't last, but **< b>tough people** do."
â @TimKaine in his first speech as Hillary's running mate

Process finished with exit code 0

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

Summarization For a given Text:

The screenshot shows an IDE interface with the following details:

- File Bar:** File, Edit, View, Navigate, Code, Analyze, Refactor, Build, Run, Tools, VCS, Window, Help.
- Project Bar:** Spark2Ont > src > main > java > text_summary.
- Toolbars:** Project, Packages, SBT projects.
- Left Sidebar:** Z: Structure, 1: Project, 2: Favorites.
- Central Area:**
 - Code Editor:** text_summary.java


```

text_summary.java
main()

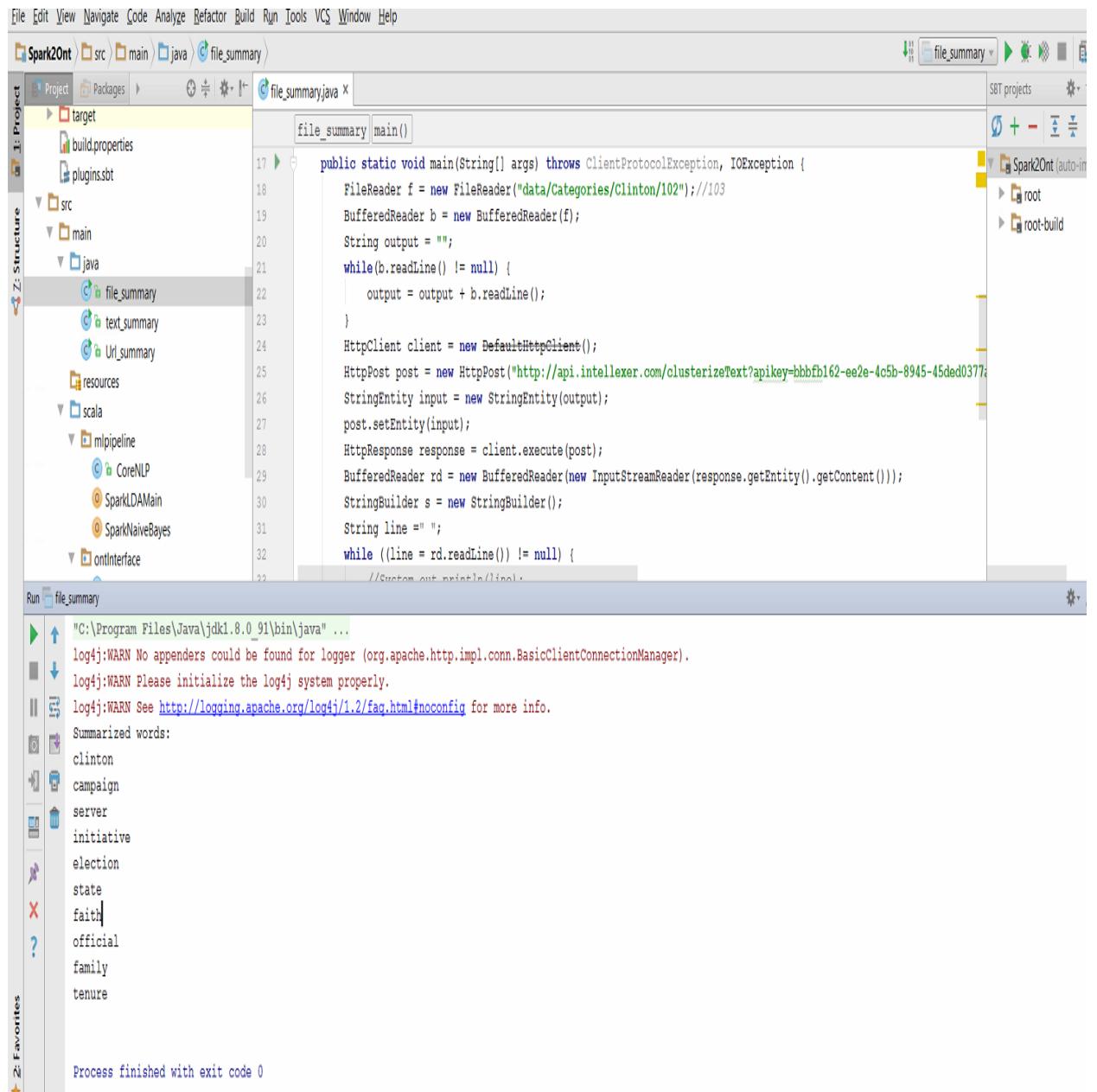
56   "Invest in good-paying jobs. In her first 100 days as president, Hillary will work with both parties to
57   "Restore collective bargaining rights for unions and defend against partisan attacks on workers' rights.
58   "Prevent countries like China from abusing global trade rules, and reject trade agreements, like the TPP,
59   "Raise the minimum wage and strengthen overtime rules. Hillary will work to raise the federal minimum wage
60   "Invest in high-quality training, apprenticeships, and skill-building for workers. Read the fact sheet at
61   "Encourage companies to invest in workers. Hillary will reward companies that share profits and invest in
62   "Protect workers from exploitation, including employer misclassification, wage theft, and other forms of
63   "Ensure policies meet the challenges families face in the 21st century economy. Hillary will fight for a
64   "Protect retirement security. After working hard for decades, Americans deserve Url_summary secure and
65 post.setEntity(input);
66 HttpResponse response = client.execute(post);
67 BufferedReader rd = new BufferedReader(new InputStreamReader(response.getEntity().getContent()));
68 StringBuilder s = new StringBuilder();
69 String line = "";
70 while ((line = rd.readLine()) != null) {
71 //    System.out.println(line);
72     s.append(line);
      
```
 - Run Tab:** text_summary


```

C:\Program Files\Java\jdk1.8.0_91\bin\java" ...
log4j:WARN No appenders could be found for logger (org.apache.http.impl.conn.BasicClientConnectionManager).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/fag.html#noconfig for more info.
Important summary of the text:

senator
secretary
state
hillary
clinton
comprehensive immigration reform
security
detention
enforcement
attack
      
```
- Bottom Status Bar:** Process finished with exit code 0

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

Summarization For a given File:


The screenshot shows the IntelliJ IDEA interface with the following details:

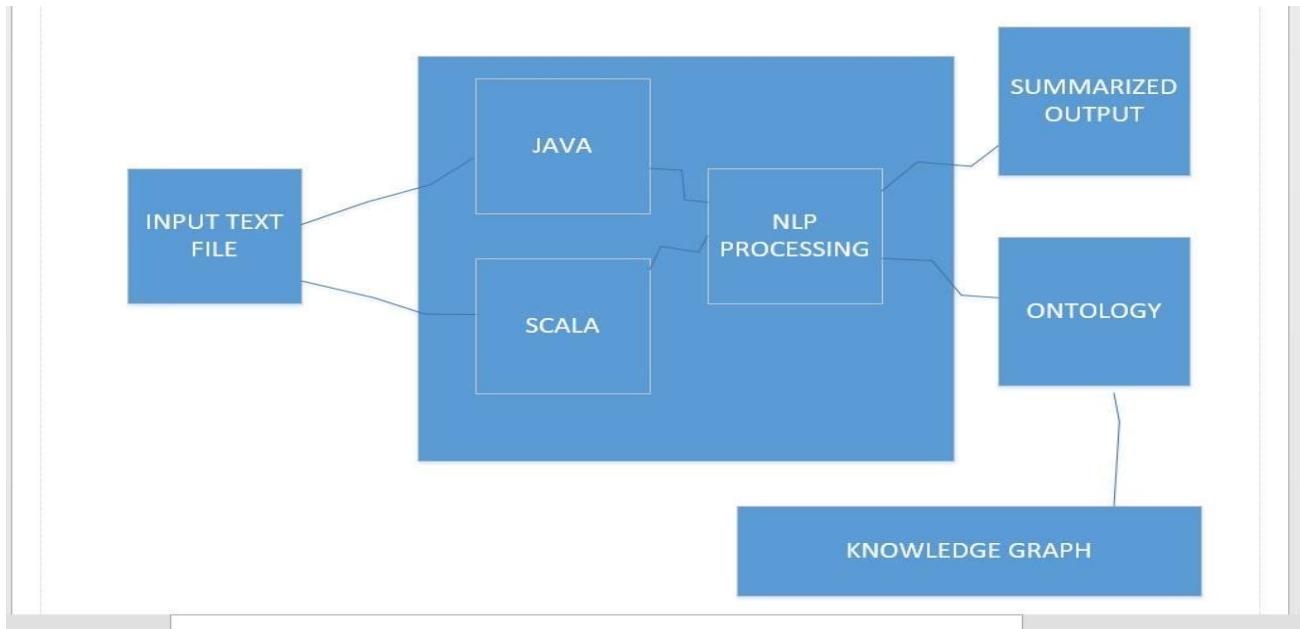
- File Structure:** The left sidebar shows the project structure for "Spark20nt". The "file_summary.java" file is selected in the "Project" view.
- Code Editor:** The main editor window displays the Java code for "file_summary.java". The code reads a file named "Clinton/102", sends a POST request to a specified URL, and prints the summarized words to the console.
- Run Tab:** The bottom tab bar shows the run configuration for "file_summary". The output pane displays the following log messages:


```
"C:\Program Files\Java\jdk1.8.0_91\bin\java" ...
log4j:WARN No appenders could be found for logger (org.apache.http.impl.conn.BasicClientConnectionManager).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
Summarized words:
clinton
campaign
server
initiative
election
state
faith
official
family
tenure
Process finished with exit code 0
```

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

5. Implementation specification:

Software Architecture:



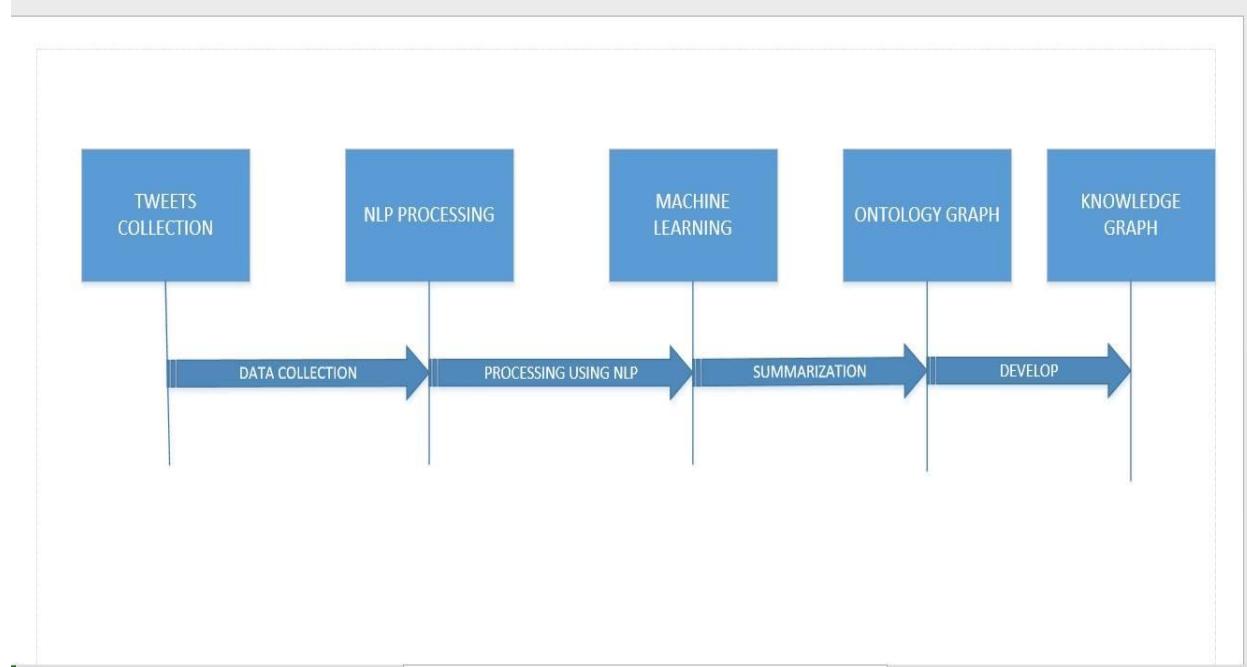
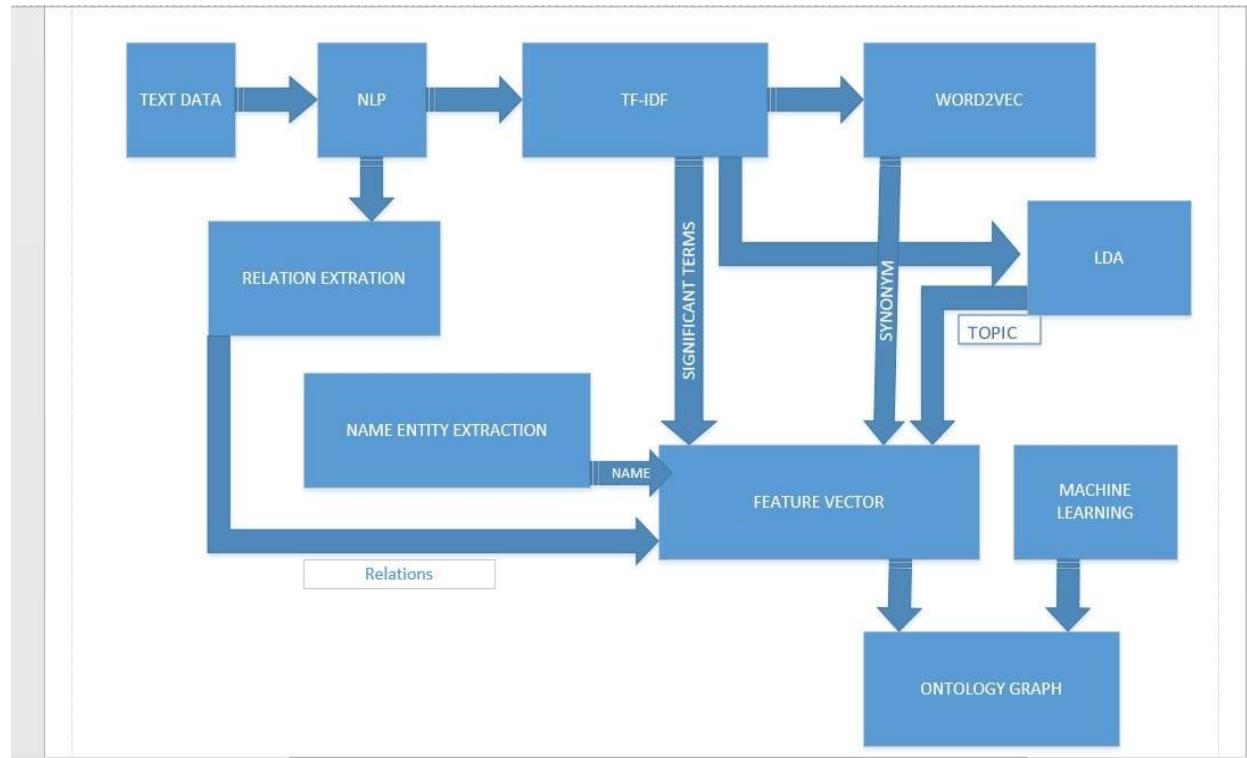
Sequence Diagram:

TEAM 8 : VILAS MAMIDYALA (18)

VIKESH PADARTHI (27)

DINESH KUMAR BANDAM (2)

BHUMIREDDY RANJITHA REDDY (4)

**Workflow Diagram:**

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

Existing Services Used:

- Implemented word count program using Scala.
- Implemented NLP program.
- Implemented TF-IDF.
- Implemented Word2vec
- Implemented Wordnet
- Implemented NER
- Implemented Feature vector generation
- Generated ontology and used protégé tool for its visualization.
- API services are implemented for summarization

New Services:

Tweet collection using Java Code.

API Services for summarization

6. Results and Evaluation:

- **Accuracy:**

```
0.08015114814043045,0.07753439247608185,-0.19245687127113342,-0.008618769235908985,-  
0.002883358858525753,0.13036096096038818,0.07886315882205963,0.04046628996729851,-  
0.0062410058453679085,0.08720554411411285,0.00431081373244524,-  
0.01477606780270454,0.008751815184950829,0.04563858360052109,-5.725307273678482E-  
4,0.04987286403775215,-0.0967053771018982,0.05071503296494484,0.12420203536748886,-  
0.06976064294576645,0.04827743023633957,0.10278143733739853,0.036327339708805084,0.05513  
7474089860916,0.0542316734790802,0.050917837768793106,-0.034305866807699203,-  
0.05986405536532402,0.026683181524276733,-  
0.09663253277540207,0.018960680812597275,0.020075950771570206,0.05938071385025978,-  
0.011799460276961327,-  
0.06337083876132965,0.010859455913305283,0.10203225165605545,0.045340344309806824,-  
0.029840435832738876,0.010707934387028217,0.1364029198884964,-0.016686882823705673]
```

Confusion matrix:

0.0 1.0 2.0 0.0

0.0 2.0 3.0 0.0

0.0 0.0 4.0 0.0

0.0 0.0 1.0 1.0

Accuracy: 0.5

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

Here when we ran the Naïve bayes algorithm we got the above feature vector and the confusion matrix. We could also observe that it has given the accuracy of 0.5 as well.

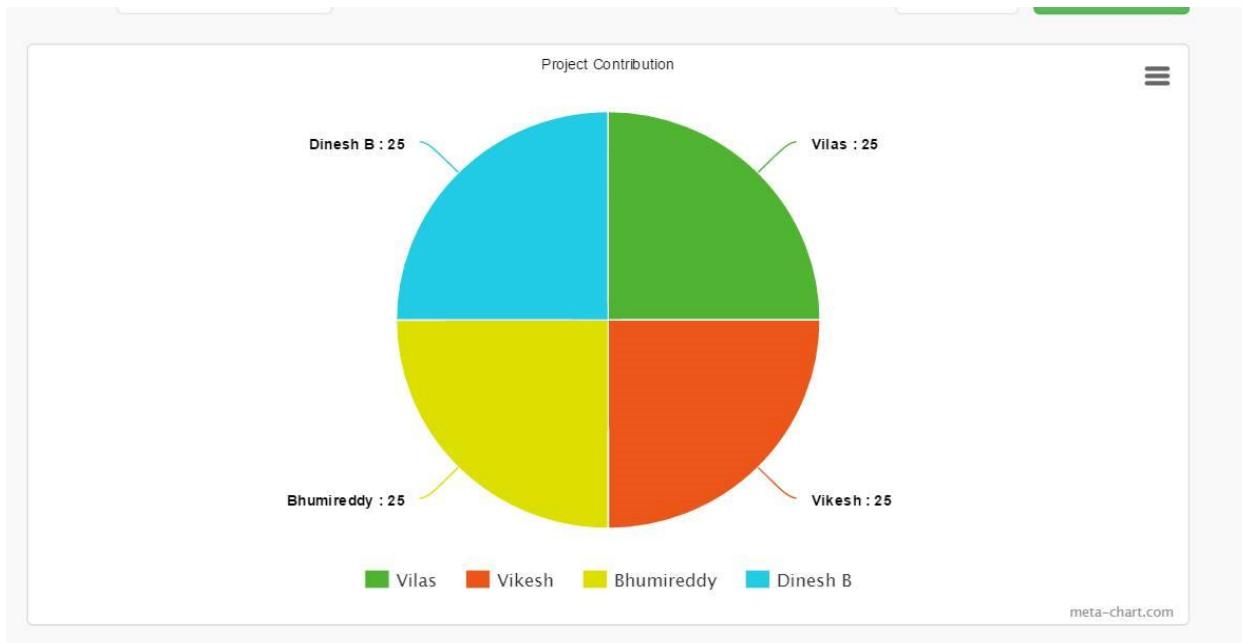
When we performed clustering for the tweets with positive and negative we had achieved 100% accuracy for the classification. This is a good achievement with respect to clustering.

- **Run Time Performance:**

- ✓ Ontology Run Time: 15 min
- ✓ Summarization:
 - For a given file: 1minute
 - For a given URL: 1minute
 - For a given text: 1minute
- ✓ Clustering of tweets with respect to positive and negative words using Naïve Bayes Algorithm: 7 min for training and 3 min for testing.
- ✓ OWL file opening through Protégé: 3 min for viewing Onto graph.
- ✓ SparQL query execution: 1minute.

7. Project Management:

Contribution of Each member:



TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

Zenhub and Github Screen shots:

The screenshot shows the GitHub repository page for 'vilasmamidyla / KDM_SM16_SM'. The repository has 4 stars, 2 forks, and 0 issues. The search bar shows a filter for 'is:issue is:closed'. There are buttons for 'Labels' and 'Milestones'. A green button labeled 'New issue' is visible. Below the search bar, there is a link to 'Clear current search query, filters, and sorts'. The main area displays a list of 14 closed issues, each with a title, a small profile picture, a comment icon, and a number indicating the number of comments. The issues are as follows:

- documentation** 1 (opened 12 hours ago by vilasmamidyla)
- Generate Feature Vector for Machine Learning (Tutorial 8)** 2 (opened 12 hours ago by vilasmamidyla)
- Conduct Topic Discovery (Tutorial 8)** 1 (opened 13 hours ago by vilasmamidyla)
- Conduct Name Entity Extraction/Relation Extraction** 1 (opened 13 hours ago by vilasmamidyla)
- Conduct NGram and Word2Vec (Tutorial 6)** 1 (opened 13 hours ago by vilasmamidyla)
- Try to use WordNet (Tutorial 7)** 1 (opened 13 hours ago by vilasmamidyla)

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTH (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

Milestones:**Documenattion par2**

Closed 2 minutes ago ⓘ Last updated less than a minute ago
continue editing the document 2

100% complete 0 open 1 closed

[Edit](#) [Reopen](#) [Delete](#)**Try to use WordNet (Tutorial 7)**

Closed 2 minutes ago ⓘ Last updated less than a minute ago
Please complete this task at the possible earliest and once you are... ([more](#))

100% complete 0 open 1 closed

[Edit](#) [Reopen](#) [Delete](#)**Conduct NGram and Word2Vec**

Closed 2 minutes ago ⓘ Last updated less than a minute ago
Please complete this task at the possible earliest and once you are... ([more](#))

100% complete 0 open 1 closed

[Edit](#) [Reopen](#) [Delete](#)**Conduct Name Entity Extraction/Relation Extraction**

Closed 2 minutes ago ⓘ Last updated less than a minute ago
Please complete this task at the possible earliest and once you are... ([more](#))

100% complete 0 open 1 closed

[Edit](#) [Reopen](#) [Delete](#)**Conduct Topic Discovery (Tutorial 8)**

Closed 2 minutes ago ⓘ Last updated less than a minute ago
Please complete this task at the possible earliest and once you are... ([more](#))

100% complete 0 open 1 closed

[Edit](#) [Reopen](#) [Delete](#)**Generate Feature Vector for Machine Learning (Tutorial 8)**

Closed 2 minutes ago ⓘ Last updated less than a minute ago
Please complete this task at the possible earliest and once you are... ([more](#))

100% complete 0 open 1 closed

[Edit](#) [Reopen](#) [Delete](#)**Documentation changes**

Closed 2 minutes ago ⓘ Last updated less than a minute ago
Please complete this task at the possible earliest and once you are... ([more](#))

100% complete 0 open 1 closed

[Edit](#) [Reopen](#) [Delete](#)

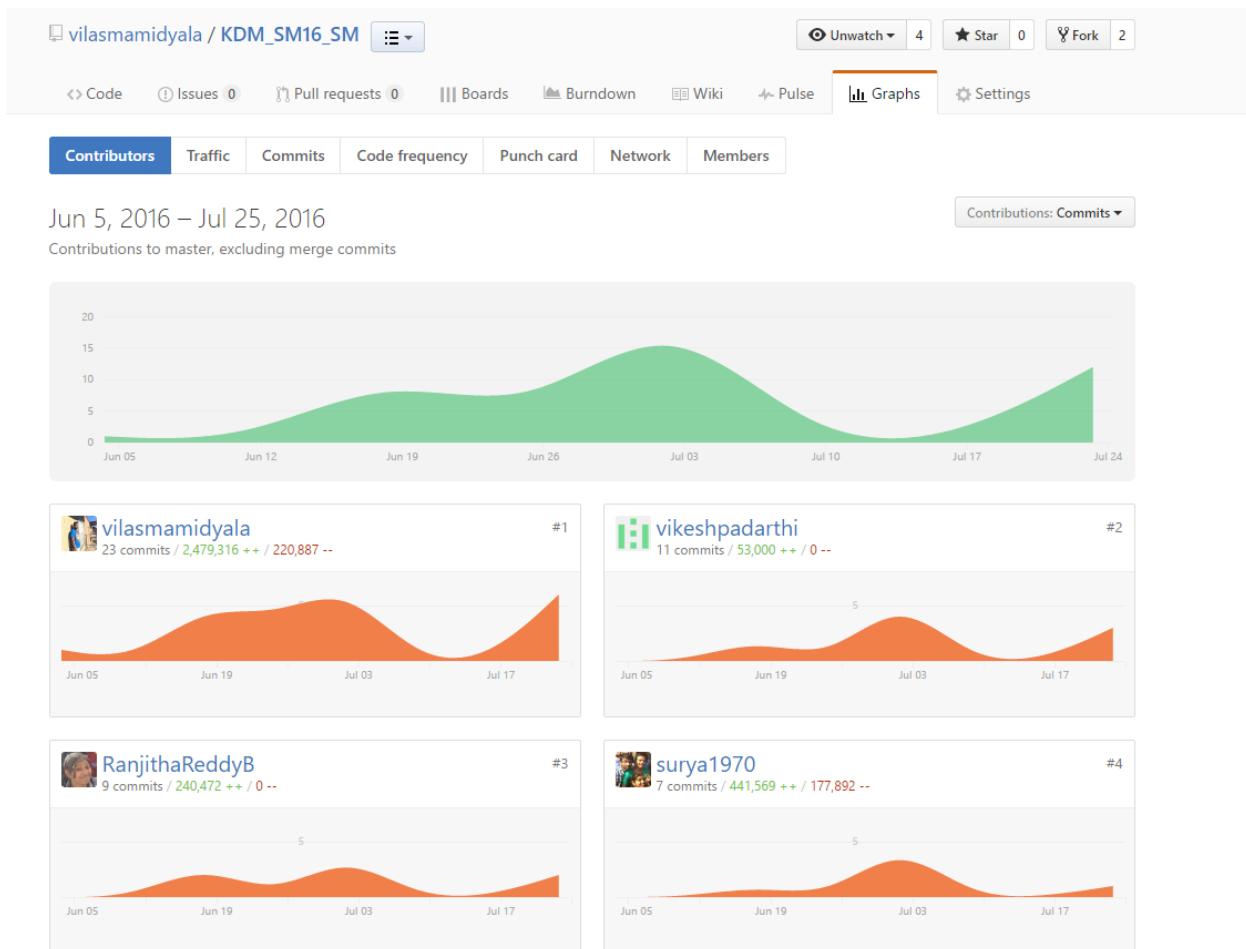
COMP-SCI 5560 (SUMMER 2016) – KNOWLEDGE DISCOVERY AND MANAGEMENT

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

The screenshot shows a GitHub repository named 'vilmamidyla / KDM_SM16_SM'. The interface displays a Kanban board with several columns: New Issues, Icebox, Backlog, In Progress, Review/QA, Done, and Closed. The 'Backlog' column contains two items: 'KDM_SM16_SM #5 Document part3' and 'KDM_SM16_SM #7 collect twitter data'. The 'In Progress' column contains two items: 'KDM_SM16_SM #1 architecture diagram and sequence diagram' and 'KDM_SM16_SM #7 collect twitter data'. The 'Closed' column contains four items: 'KDM_SM16_SM #6 task for wordcount', 'KDM_SM16_SM #3 Try NLP processing', 'KDM_SM16_SM #4 Try information Extraction/Retrieval/technologies', and 'KDM_SM16_SM #2 documentation-part1'. The top navigation bar shows tabs for Code, Issues (3), Pull requests (0), Boards, Burndown, Wiki, Pulse, Graphs, and Settings.

TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTHI (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

Contribution of source code GitHub:



TEAM 8 : VILAS MAMIDYALA (18) VIKESH PADARTH (27) DINESH KUMAR BANDAM (2) BHUMIREDDY RANJITHA REDDY (4)

8. Feature concerns/Issues:

- 1) For small amount of data given as input for NLP processing and for other code executions. We found that these programs are working well and giving better results. The issue has occurred when we had tried implement NLP operation on large amount of data the programs were not able to run properly.
- 2) We considered taking Twitter data for the first phase. But we want to know whether twitter data can be useful for summarization? Because each tweet will be independent of the other tweets most of the times. This data alone might not help us for summarization. we think we need to take other different source s of data as well. we will try to figure out about what are the other sources that can be included.

Future Work:

In our further increments we would like focus on how to implement NLP operations on a bit of huge amount of data. We would like to do Word2Vec and LDA analysis on our data and then to get the feature vector for the data. We would like to implement Machine learning and ontology to derive final graphs.

References:

- SparQL: <https://www.w3.org/TR/rdf-sparql-query/>
- Ontology: <http://homes.cs.washington.edu/~pedrod/papers/hois.pdf>
- Protégé: <http://protege.stanford.edu/>
- NaiveBayes Algorithm: <http://software.ucv.ro/~cmihaescu/ro/teaching/AIR/docs/Lab4-NaiveBayes.pdf>
- Summarization: <https://www.semanticscholar.org/paper/An-Ontology-Based-Approach-to-Text-Summarization-Hennig-Umbrath/ab138bc53af41bfc5f1a5b2ce5ab4f11973e50aa>
- <http://www.intellexer.com/>