CHAPTER I
INTRODUCTION

1.1 SYSTEM OVERVIEW

The increase in overall demand for better quality vehicles has caused customer value analysis to become a key factor determining the progress of an organization in the competitive market. In order to ensure better customer satisfaction and identify key areas of improvement, we perform sentimental analysis on online customer reviews of vehicles.

Sentiment analysis also known as Opinion Mining refers to the use of natural language processing, text analysis and computational linguistics to identify and extract subjective information in source materials. These linguistic algorithms decompose the user reviews into various dimensions based on the polarity of the review. The polarity of a review categorizes the review into three domains: positive, negative or neutral reviews. These three categories help in determining the tone of the review based on which the opinion of the consumer can be analyzed. Opinion mining can be useful in several ways. It can help marketers evaluate the success of an advertisement campaign or new product launch, determine which versions of a product or service are popular and identify which demographics like or dislike particular product features. For example, a review on a website might be broadly positive about a digital camera, but be specifically negative about how heavy it is. Being able to identify this kind of information in a systematic way gives the vendor a much clearer picture of public opinion than surveys or focus groups do, because the customer provides the data . Customer generated data can be hence manipulated accordingly and the analysis is used for various purposes such as guiding decisions to improve the products or marketing strategies.

1.2 SCOPE OF THE PROJECT

The increased usage of the Internet as a medium of communication and for expressing personal views has led to the collection of a large amount of user generated data that can be analyzed and utilized in a beneficial manner. This has led to the generation of the wildly acclaimed term referred to as “Big Data”. This huge data can be analyzed and monitored so as to predict data patterns and use them in multiple use cases.

The main aim of this project is to analyze consumer reviews for automobile industries and predict the flaws or areas of improvement in the various sectors of the industry. This is done with the help of the Natural Language Processing library (NLP) in Python that helps in dividing a sentence into positive, negative or neutral feedback based on the polarity of the sentence. A basic task in sentiment analysis is classifying the polarity of a given text at the document, sentence, or feature/aspect level—whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral. Advanced, "beyond polarity" sentiment classification looks, for instance, at emotional states such as "angry", "sad", and "happy". Existing approaches to sentiment analysis can be grouped into three main categories: knowledge-based techniques, statistical methods, and hybrid approaches. Knowledge-based techniques classify text by affect categories based on the presence of unambiguous affect words such as happy, sad, afraid, and bored. Some knowledge bases not only list obvious affect words, but also assign arbitrary words a probable "affinity" to particular emotions. This project aims to also take into consideration sarcastic comments that have conflicting meanings to a sentence which makes it arduous to determine the polarity of the comment.

CHAPTER 2
LITERATURE SURVEY

1) Sentiment Analysis in Social Networks: A Study on Vehicles

Author : Renata Maria Baracho and Gabriel Silva
Year : 2015
Transaction : IEEE Journal on Knowledge and Data Engineering

Renata and Silva presented a novel architecture on sentimental analysis that aims to create a process of sentiment analysis based on ontologies in the automobile domain and then to develop a prototype. The process aims at making a social media analysis, identifying feelings and opinions about brands and vehicle parts.

The method that guided the development process consisted of the construction of ontologies and a dictionary of terms that reflect the structure of the vocabulary domain. The dictionary was ideally a large repository of words phrases along with the possible contextual synonyms that these words meant . Moreover, The proposed process was capable of generating information that answered questions regarding the degree of relevance or likelihood between entities. The answer to these questions were provided by comparison and they have shown a general view reflected on different social networks, indicating, for example, that for a given vehicle, a certain percentage of responses are considered positive, while for others, the percentage is considered negative. The results were used for various purposes such as guiding decisions to improve the products or marketing strategies.

2) Analysis of review helpfulness based on Consumer perspective.

Author : Yuanlin Chen, Yueting Chai, Yi Liu, and Yang Xu

Year : 2015

Transaction: IEEE Conference sponsored by Tsinghua University Press on Science and Technology

The work of **Yuanlin Chen, Yueting Chai , Yi Liu, and Yang Xu** includes the exploration of the applications of reviews based on consumer perspective. When consumers make purchase decisions, they generally refer to the reviews generated by other consumers who have already purchased similar products in order to get more information. Online transaction platforms provide a highly convenient channel for consumers to generate and retrieve product reviews.

In addition, consumers could also vote reviews perceived to be helpful in making their decision. However, due to diverse characteristics, consumers could have different preferences on products and reviews. Their voting behavior might be influenced by reviews and existing review votes. To explore the influence mechanism of the reviewer, the review, and the existing votes on review helpfulness, they have proposed three hypotheses based on the consumer perspective and performed statistical tests to verify these hypotheses with real review data from Amazon. Their empirical study indicates that review helpfulness has significant correlation and trend with reviewers, review valance, and review votes. They have also discussed the implications of their findings on consumer preference and review helpfulness. In conclusion , they have provided the degree of relevance and the correlation trend between reviews and reviewers.

3) Co-Extracting Opinion Targets and Opinion Words from Online Reviews Based on the Word Alignment Model.

Author : Kang Liu, Liheng Xu, and Jun Zhao

Year : 2015

Transaction: IEEE Journal on Knowledge and Data Engineering

Kang Liu, Liheng Xu, and Jun Zhao have proposed a paper which deals with opinion targets from online reviews based on the Word Alignment Model. Mining opinion targets and opinion words from online reviews are important tasks for fine-grained opinion mining, the key component of which involves detecting opinion relations among words. To this end, this paper proposes a novel approach based on the partially supervised alignment model, which regards identifying opinion relations as an alignment process. Then, a graph-based co-ranking algorithm is exploited to estimate the confidence of each candidate. Finally, candidates with higher confidence are extracted as opinion targets or opinion words.

Compared to previous methods based on the nearest-neighbor rules, their model captures opinion relations more precisely, especially for long-span relations. In particular, compared to the traditional unsupervised alignment model, the proposed model obtains better precision because of the usage of partial supervision. Their experimental results on three corpora with different sizes and languages show that their approach effectively outperforms state-of-the-art methods. The research is based on the further assumption that the sarcasm probably arises from contrasting polarity .

4) Social Media Sentimental Analysis with Sarcasm Detection

Author : Edwin Lunando and Ayu Purwarianti

Year : 2015

Transaction : IEEE Transaction on Big Data

The work of **Lunando** and **Purwarianti** concentrates on identifying key aspects of natural language that cannot be normally identified by polarized lexicons such as sarcasm .Sarcasm is considered one of the most difficult problem in sentiment analysis. In their observation on Indonesian social media, for certain topics, people used to to criticize something using sarcasm. Here, they proposed two additional features to detect sarcasm after a common sentiment analysis is conducted. The features were the negativity information and the number of interjection words. They also employed translated SentiWordNet in the sentiment classification. All the classifications were reconducted with machine learning algorithms. The experimental results showed that the additional features are quite effective in the sarcasm detection.

The goal was to create a sarcasm classifier for tweets that explicitly recognized contexts that contain a positive sentiment contrasted with a negative situation. Their approach learned rich phrasal lexicons of positive sentiments and negative situations using only the seed a particular word and a collection of sarcastic tweets as input. A key factor that made the algorithm work was the presumption that if you find a positive sentiment or a negative situation in a sarcastic tweet, then you have found the source of the sarcasm.

5) Mining Online User-Generated Content: Using Sentiment Analysis Technique to Study Industrial Service Quality.

Author : Wenjing Duan, Qing Cao, Yang Yu and Stuart Levy

Year : 2014

Transaction: IEEE Journal on Affective Computing

The work of **Wenjing Duan, Qing Cao, Yang Yu and Stuart Levy** aims to look beyond the quantitative summary to provide a more comprehensive view of online user-generated content. They have obtained a unique and extensive dataset of online user reviews for various industries such as hotels, across various review sites and over long time periods. They use the sentiment analysis technique to decompose user reviews into five dimensions to measure industrial service quality. Those dimensions are then incorporated into econometrics models to examine their effect in shaping users' overall evaluation and content generating behavior. The results suggest that different dimensions of user reviews have significantly differential impact in forming user evaluation and driving content generation.

The sentiment analysis results show high level of accuracy in capturing and measuring service quality dimensions compared with existing text-mining studies. They have used an econometric modeling technique to examine the potential differential effect of different service quality dimensions. The results suggest that different dimensions of service quality embedded in user reviews account for significantly different weight in the review textual content.

6) Research on Brand Equity of Automobile Industry - Based on Customer Experience and Modern Service

Author : X Zhang , Guoqin Bu , Shuzen Wu

Year : 2013

Transaction : IEEE Journal on Management and Service Science.

Zhang and **Guoqin Bu** have contributed their work towards the rapid development of China's Automobile industry, the construction of independent brands, and accumulation of brand equity, expansion of brand influence which became the key factors for China's Automobile industry's further development. The brand equity, as enterprise important intangible asset, lacks accurate measurement standards and there was an unavailability of quantitative measure indicators.

Automobile industry collected characteristics both from manufacturing and service industry, which makes the customer experience become an important dimensions of measure for brand equity of automobile industry. They define brand equity as part of enterprise assets, and from the angle of customer experience and perception measure brand equity. They further explored customer experience as a breakthrough point and key factors, using empirical analysis of the data collection, applying calculating statistics method to analyse customer's satisfaction index from multidimensions. In conclusion, they proposed that more attention should be paid to the modern automobile comprehensive service level, which is mainly based on customer experience.

7) Qualitative risk avoidance methodology for categorization of mined opinions from online reviews.

Author : B. Dhanalakshmi and A. Chandrasekar

Year : 2015

Transaction : IEEE Conference on Advanced Computing (ICoAC),

The work done by **Chandrasekar** and **Dhanalakshmi** involved the rapid development of the Internet which spectacularly changed the mode that people articulate their opinions. People can liberally send reviews on any aspect of various websites to convey their individual opinions. As the opinions communicate the subjective thoughts, estimation, and conjectures of people in natural language, these type of contents contributed by users have been well acknowledged as valuable information. It can be oppressed to evaluate public reviews on a specific topic or product in order to classify out user like or dislike, etc.

Opinion blend and mining were the methods to explore and summarize opinions from reviews in order to understand public perception on a specific topic or an entity. The intent was to determine opinions from online reviews and managing risk in future. This methodology involved phases such as Data preprocessing, Content discovery, Review mining and Risk investigation. Initially the formless data from the websites was extracted and preprocessed. This stage was then used for formatting the fact before sentiment classification and mining.

CHAPTER 3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Sentiment analysis refers to automatically extracting the sentiment present in a given natural language text. The existing systems use supervised learning using features based on various polarity lexicons. Polarity lexicons denote the nature of a particular text or a phrase in relevance to the context.

The existing lexicon-based technique adopts the idea of determining the review sentiment by obtaining word polarities from a lexicon. This lexicon can be domain-independent or domain-specific. One can use a domain-specific lexicon whenever available, to get a better performance by obtaining the correct word polarities in the given domain . This is majorly because of the reason that the contextual meaning of a word provides various meanings across multiple domains. On the other hand, establishing a domain-specific lexicon is costly, so many systems use a domain-independent lexicon, such as the SentiWordNet, shortly and SenticNet . Part of Speech (POS) information is commonly indicated in polarity lexicons, partly to overcome word-sense disambiguity and therefore help achieve a better sentiment classification performance.

3.1.1 DISADVANTAGES

The following are the disadvantages of the existing system :

- Inability to identify contextual meanings of phrases that denote sarcastic behavior.
- Dependence on generalization from the training data set.
- Domain specific lexicons are costly in the existing system.

3.2 PROPOSED SYSTEM

Our system proposes a domain independent supervised learning methodology that uses machine learning techniques to build models or discriminators for the different classes such as positive or negative reviews using a large corpus for an automobile organization in order to increase the overall productivity.

The training data consists of a set of training examples of the product reviews of automobiles that is segregated based on various models and years of manufacturing. This dataset is collected by the usage of a RESTful API wherein the data is accumulated as a very large repository of data from which data can be accessed as a service and then is used as the Master Dataset. Since the dataset is potentially large supervised learning techniques , there is a need to use a Data Store Environment . Hadoop File System (HDFS) is used for this purpose. This training dataset is fed into the system which analyzes commonly occurring data patterns and identifies the polarity of each review provided by the user. This identification can be effectively used when the reviews are sarcastic and cannot be easily categorized based on the polarity. In such situations we perform comparisons to the previous and next word and analyze the contextual reference of that particular phrase. The analyzed data is then used to predict the nature of possible outcomes from previous data and provide recommendations to improve efficiency.

3.2.1 ADVANTAGES

These are the various advantages of our proposed system :

- Usage of Supervised Learning in place of conventional domain specific lexicon approaches.
- Identification of sarcastic reviews with better efficiency.

3.3 REQUIREMENTS SPECIFICATION

3.3.1 Hardware Requirements

• **Hard Disk**	: 256 GB and above
• **RAM**	: 8GB and above
• **Processor**	: Intel Core i3 and above

3.3.2 Software Requirements

- Linux Operating System
- Hadoop File System

3.4 LANGUAGE SPECIFICATION

3.4.1 Python

Python is a widely used high-level, general-purpose, interpreted, dynamic programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than possible in languages such as C++ or Java. The language provides constructs intended to enable clear programs on both a small and large scale.

Python supports multiple programming paradigms, including object-oriented, imperative and functional programming or procedural styles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library. Python uses dynamic typing and a mix of reference counting and a cycle-detecting garbage collector for memory management.

Rather than requiring all desired functionality to be built into the language's core, Python was designed to be highly extensible. Python can also be embedded in existing applications that need a programmable interface. This design of a small core language with a large standard library and an easily extensible interpreter was intended by Van Rossum from the start because of his frustrations with ABC, which espoused the opposite mindset.

Python Features

- **Easy-to-learn, read and maintain:** Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly. Python code is more clearly defined and visible to the eyes.Python's source code is fairly easy-to-maintain.

- **A broad standard library:** Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

- **Interactive Mode:**Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

- **Portable:** Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

- **Extendable:** You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

- **Scalable:** Python provides a better structure and support for large programs than shell scripting.

3.4.2 Hadoop Environment

Hadoop is an open-source framework that allows to store and process big data in a distributed environment across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage.

Hadoop Architecture

Hadoop framework includes following four modules:

- **Hadoop Common:** These are Java libraries and utilities required by other Hadoop modules. These libraries provides filesystem and OS level abstractions and contains the necessary Java files and scripts required to start Hadoop.

- **Hadoop YARN:** This is a framework for job scheduling and cluster resource management.

- **Hadoop Distributed File System (HDFS™):** A distributed file system that provides high-throughput access to application data.

- **Hadoop MapReduce:** This is YARN-based system for parallel processing of large data sets.

The following are the components of the Hadoop Framework :

MapReduce

Hadoop **MapReduce** is a software framework for easily writing applications which process big amounts of data in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner.

MapReduce actually refers to the following two different tasks that Hadoop programs perform:

- **The Map Task:** This is the first task, which takes input data and converts it into a set of data, where individual elements are broken down into tuples (key/value pairs).
- **The Reduce Task:** This task takes the output from a map task as input and combines those data tuples into a smaller set of tuples. The reduce task is always performed after the map task.

Typically both the input and the output are stored in a file-system. The framework takes care of scheduling tasks, monitoring them and re-executes the failed tasks.

The MapReduce framework consists of a single master **JobTracker** and one slave **TaskTracker** per cluster-node. The master is responsible for resource management, tracking resource consumption/availability and scheduling the jobs component tasks on the slaves, monitoring them and re-executing the failed tasks. The slaves TaskTracker execute the tasks as directed by the master and provide task-status information to the master periodically.

Hadoop Distributed File System

Hadoop can work directly with any mountable distributed file system such as Local FS, HFTP FS, S3 FS, and others, but the most common file system used by Hadoop is the Hadoop Distributed File System (HDFS).

HDFS uses a master/slave architecture where master consists of a single **NameNode** that manages the file system metadata and one or more slave **DataNodes** that store the actual data.

A file in an HDFS namespace is split into several blocks and those blocks are stored in a set of DataNodes. The NameNode determines the mapping of blocks to the DataNodes. The DataNodes takes care of read and write operation with the file system. They also take care of block creation, deletion and replication based on instruction given by NameNode.

HDFS provides a shell like any other file system and a list of commands are available to interact with the file system. These shell commands will be covered in a separate chapter along with appropriate examples.

Working of Hadoop

A user/application can submit a job to the Hadoop (a hadoop job client) for required process by specifying the following items:

1. The location of the input and output files in the distributed file system.
2. The java classes in the form of jar file containing the implementation of map and reduce functions.
3. The job configuration by setting different parameters specific to the job.

Then , The Hadoop job client then submits the job (jar/executable etc) and configuration to the JobTracker which then assumes the responsibility of distributing the software/configuration to the slaves, scheduling tasks and monitoring them, providing status and diagnostic information to the job-client.

The TaskTrackers on different nodes execute the task as per MapReduce implementation and output of the reduce function is stored into the output files on the file system.

Advantages of Hadoop

- Hadoop framework allows the user to quickly write and test distributed systems. It is efficient, and it automatic distributes the data and work across the machines and in turn, utilizes the underlying parallelism of the CPU cores.
- Hadoop does not rely on hardware to provide fault-tolerance and high availability (FTHA), rather Hadoop library itself has been designed to detect and handle failures at the application layer.
- Servers can be added or removed from the cluster dynamically and Hadoop continues to operate without interruption.
- Another big advantage of Hadoop is that apart from being open source, it is compatible on all the platforms since it is Java based.

3.4.3 NoSQL

A NoSQL database provides a mechanism for storage and retrieval of data which is modeled in means other than the tabular relations used in relational databases applications. NoSQL systems are also sometimes called "Not only SQL" to emphasize that they may support SQL-like query languages .

NoSQL document database fully supports agile development, because it is schema-less and does not statically define how the data must be modeled. Instead, it defers to the applications and services, and thus to the developers as to how data should be modeled. With NoSQL, the data model is defined by the application model. Applications and services model data as objects.

CHAPTER 4
SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

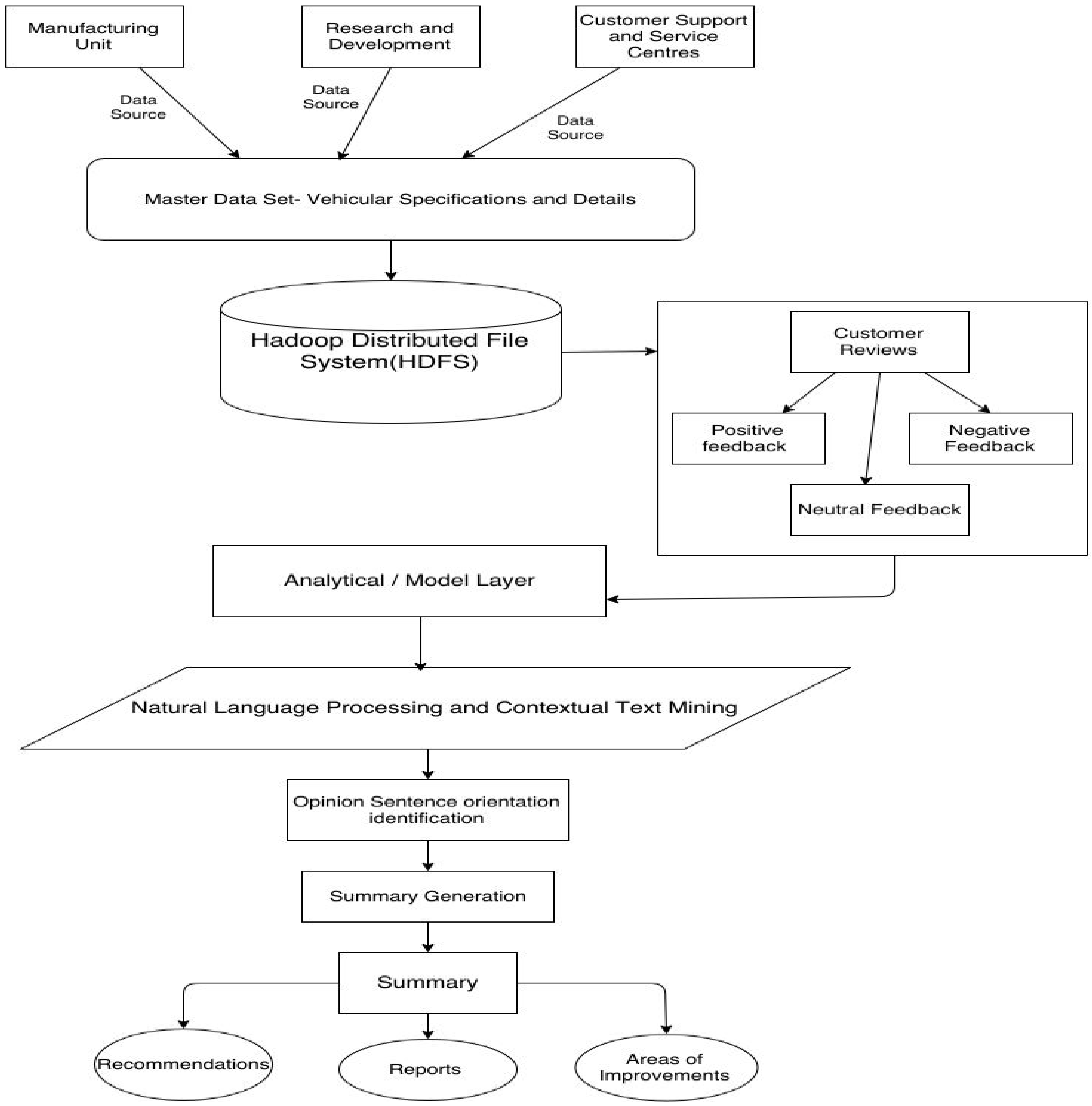


Fig 4.1:Architechture Diagram

The above diagram denotes the architecture of the proposed system. It displays three primary data sources i.e. Manufacturing units, Research and Development and Customer support and Service Centers. These three form the primary sources for data preparation. The master data set consists of the Vehicular Specifications and details gathered from the three data sources. This huge chunk of data is stored in the Hadoop Distributed File System (HDFS). The customer reviews are segregated into three categories viz. Positive feedback, Negative feedback and Neutral feedback. The Natural Language Processing (NLP) library in python enables us to determine the polarity of the review. This polarity is used to determine the tone of the comment that can help in identifying the intricate details of the review. The opinion sentence orientation identification is used in order to arrange the review based on the sentence orientation and identify the key features of the comment.

The opinion sentence orientation identification process is followed by summary generation wherein the final feedback is offered based on the deciphered comment. The summary can be in the form of Reports, Recommendations or Areas of improvement that will help the automobile firms to analyze their areas of improvement and take necessary steps to achieve customer satisfaction. It will help in determining the major faults in their system and can assist in strategizing in order to rectify faults/ errors in their production system.

4.2 ACTIVITY DIAGRAM

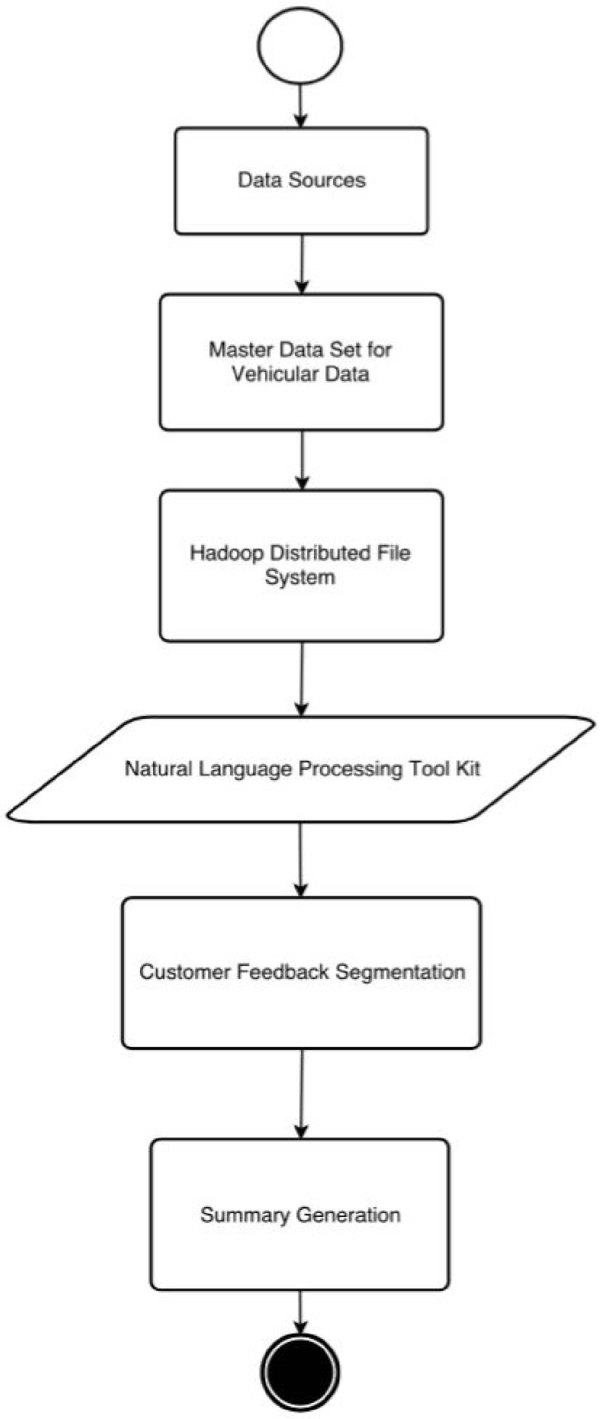

Data Sources
Master Data Set for Vehicular Data
Hadoop Distributed File System
Natural Language Processing Tool Kit
Customer Feedback Segmentation
Summary Generation

Fig 4.2:Activity Diagram

Activity diagram is another important diagram to describe dynamic behavior. Activity diagram consists of activities, links, relationships etc. It models all types of flows like parallel, single, concurrent. Activity diagram describes the flow control from one activity to another without any messages. These diagrams are used to model high-level view of business requirements. Before drawing an activity diagram we should identify the following elements: Activities, Association, Conditions, and Constraints. Once the above-mentioned parameters are identified we need to make a mental layout of the entire flow. This mental layout is then transformed into an activity diagram.

The first step is to identify the data sources and accumulate the relevant data from these data sources. These data sources are used in order to provide raw data for the master data set that contains the vehicular specifications and details pertaining to the automobile industry. This data is stored in the Hadoop Distributed File System (HDFS). Python has an inbuilt library called the Natural Language Processing library (NLP). This library is used to extract the relevant words from a sentence and disseminate it into positive, negative and neutral based on the polarity of the sentence. This is used to analyze the review and help to identify the areas of improvement. The feedback is then segmented and oriented to make it seem logical. Based on this data, a summary of reports is generated that provides an overall review of the comments from the different sectors of the automobile industry. This helps in providing the industries with a processed information that can be used to amend the faulty areas and ameliorate glitches in their automobile parts.

4.3 DATA FLOW DIAGRAM

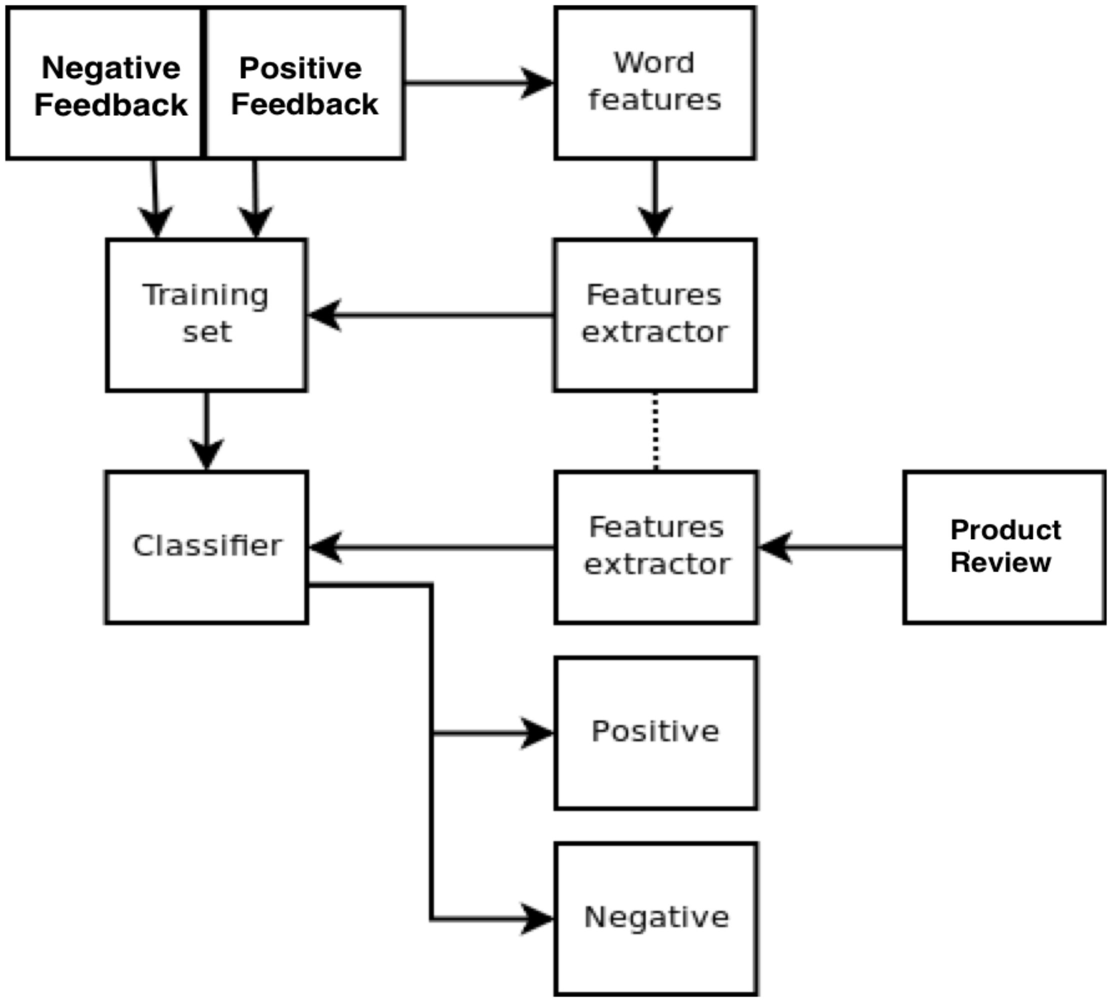


Fig 4.3:Dataflow Diagram

Dataflow diagram shows the flow of data between different processes in a business. Data flow diagrams use four primary symbols. They are process, data flow, data store, external entity. DFD can be represented in many levels. Each level can be considered as an incrementation of the previous level.

The features extractor extracts the relevant data from the product review and the classifier bifurcates them into positive and negative. A training set is used to train the system in order to identify the polarity for different scenarios and contexts. The word features are then extracted and further processes are performed.

CHAPTER 5
SYSTEM IMPLEMENTATION

5.1 MODULE I – DATA PREPARATION and STORAGE ENVIRONMENT

5.1.1 Edmunds Developer API:

In order to create the training data set we use a RESTful (Representational State Transfer) API for accessing the dataset. RESTful web services are built to work best on the Web. Representational State Transfer (REST) is an architectural style that specifies constraints, such as the uniform interface, that if applied to a web service induce desirable properties, such as performance, scalability, and modifiability, that enable services to work best on the Web. In the REST architectural style, data and functionality are considered resources and are accessed using Uniform Resource Identifiers (URIs), typically links on the Web. The resources are acted upon by using a set of simple, well-defined operations. The REST architectural style constrains an architecture to a client/server architecture and is designed to use a stateless communication protocol, typically HTTP. In the REST architecture style, clients and servers exchange representations of resources by using a standardized protocol.

The API provides the test data set based on various car makes, models and years of manufacturing in the required JSON Format as response to every request. The response received is in turn stored in the data store environment for further analysis and processing.

5.1.2 HADOOP INSTALLATION

Hadoop requires a working Java 1.5+ (aka Java 5) installation. This is because the Map-Reduce functionality in the HDFS is based completely on Java.

Installing Java JDK

```
user@ubuntu:~$ sudo apt-get install sun-java6-jdk
```

Adding a dedicated Hadoop system user

```
user@ubuntu:~$ java -version
user@ubuntu:~$ sudo addgroup hadoop_group
user@ubuntu:~$ sudo adduser --ingroup hadoop_group hduser1
user@ubuntu:~$ sudo adduser hduser1 sudo
```

Configuring SSH

The hadoop control scripts rely on SSH to peform cluster-wide operations. For example, there is a script for stopping and starting all the daemons in the clusters. To work seamlessly, SSH needs to be setup to allow password-less login for the hadoop user from machines in the cluster. The simplest way to achive this is to generate a public/private key pair, and it will be shared across the cluster.

Access key Generation

```
user@ubuntu:~$ su – hduser1
hduser1@ubuntu:~$ ssh-keygen -t rsa -P ""
hduser1@ubuntu:~$   cat $HOME/.ssh/id_rsa.pub >>
$HOME/.ssh/authorized_keys
hduser@ubuntu:~$ ssh localhost
```

Set Hadoop-related environment variables

```
     hduser@ubuntu:~$ su - hduser1
export HADOOP_HOME=/usr/local/hadoop
export PATH= $PATH:$HADOOP_HOME/bin
#export JAVA_HOME=/usr/lib/j2sdk1.5-sun
# export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-amd64  (for 64 bit)
# export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-amd64  (for 32 bit)
```

Creating a directory in HDFS :

```
hduser@ubuntu:~$ sudo mkdir -p /app/hadoop/tmp
hduser@ubuntu:~$ sudo chown hduser:hadoop /app/hadoop/tmp
hduser@ubuntu:~$ sudo chmod 750 /app/hadoop/tmp
hduser@ubuntu:~$ /usr/local/hadoop/bin/hadoop namenode –format
```

Starting your single-node cluster

```
hduser@ubuntu:~$ sudo chmod -R 777 /usr/local/Hadoop

hduser@ubuntu:~$ /usr/local/hadoop/bin/start-all.sh

hduser@ubuntu:/usr/local/hadoop$ jps
```

This will startup a Namenode, Datanode, Jobtracker and a Tasktracker on the machine. Thus the Hadoop Environment is installed.

APPENDIX – I

SAMPLE CODE

```
import nltk

class Splitter(object):

def __init__(self):

self.nltk_splitter = nltk.data.load('tokenizers/punkt/english.pickle')

self.nltk_tokenizer = nltk.tokenize.TreebankWordTokenizer()

def split(self, text):

input format: a paragraph of text

output format: a list of lists of words.

sentences = self.nltk_splitter.tokenize(text)

tokenized_sentences = [self.nltk_tokenizer.tokenize(sent) for sent in sentences]

return tokenized_sentences

class POSTagger(object):

def __init__(self):

pass

def pos_tag(self, sentences):
    pos = [nltk.pos_tag(sentence) for sentence in sentences]
    pos = [[(word, word, [postag]) for (word, postag) in sentence] for sentence in
    pos]
return pos

splitter = Splitter()

postagger = POSTagger()

splitted_sentences = splitter.split(text)

print splitted_sentences

pos_tagged_sentences = postagger.pos_tag(splitted_sentences)

print pos_tagged_sentences
```

positive.yml

```
nice: [positive]
awesome: [positive]
cool: [positive]
superb: [positive]
```

negative.yml

```
bad: [negative]
uninspired: [negative]
expensive: [negative]
dissapointed: [negative]
```

TAGGING THE TEXT WITH DICTIONARIES

```
class DictionaryTagger(object):

def __init__(self, dictionary_paths):

files = [open(path, 'r') for path in dictionary_paths]

dictionaries = [yaml.load(dict_file) for dict_file in files]

map(lambda x: x.close(), files)

self.dictionary = { }

self.max_key_size = 0

for curr_dict in dictionaries:

for key in curr_dict:

if key in self.dictionary:

self.dictionary[key].extend(curr_dict[key])

else:

self.dictionary[key] = curr_dict[key]

self.max_key_size = max(self.max_key_size, len(key))

def tag(self, postagged_sentences):

return [self.tag_sentence(sentence) for sentence in postagged_sentences]

def tag_sentence(self, sentence, tag_with_lemmas=False):
```

```
tag_sentence = []

N = len(sentence)

if self.max_key_size == 0:

self.max_key_size = N

i = 0

while (i < N):

j = min(i + self.max_key_size, N) #avoid overflow

tagged = False

while (j > i):

expression_form = ' '.join([word[0] for word in sentence[i:j]]).lower()

expression_lemma = ' '.join([word[1] for word in sentence[i:j]]).lower()

if tag_with_lemmas:

literal = expression_lemma

else:

literal = expression_form

if literal in self.dictionary:

#self.logger.debug("found: %s" % literal)

is_single_token = j - i == 1

original_position = i

i = j
```

```
taggings = [tag for tag in self.dictionary[literal]]

tagged_expression = (expression_form, expression_lemma, taggings)

if is_single_token: #if the tagged literal is a single token, conserve its previous
taggings:

original_token_tagging = sentence[original_position][2]

tagged_expression[2].extend(original_token_tagging)

tag_sentence.append(tagged_expression)

tagged = True

else:

j = j - 1

if not tagged:

tag_sentence.append(sentence[i])

i += 1

return tag_sentence
def sentence_score(sentence_tokens, previous_token, acum_score):

token_score = sum([value_of(tag) for tag in tags])

if previous_token is not None:

previous_tags = previous_token[2]

if 'inc' in previous_tags:

token_score *= 2.0
```

```
elif 'dec' in previous_tags:
token_score /= 2.0
elif 'inv' in previous_tags:
token_score *= -1.0
return sentence_score(sentence_tokens[1:], current_token, acum_score +
token_score)

def sentiment_score(review):
return sum([sentence_score(sentence, None, 0.0) for sentence in review])
```

SENTIMENT MEASURE

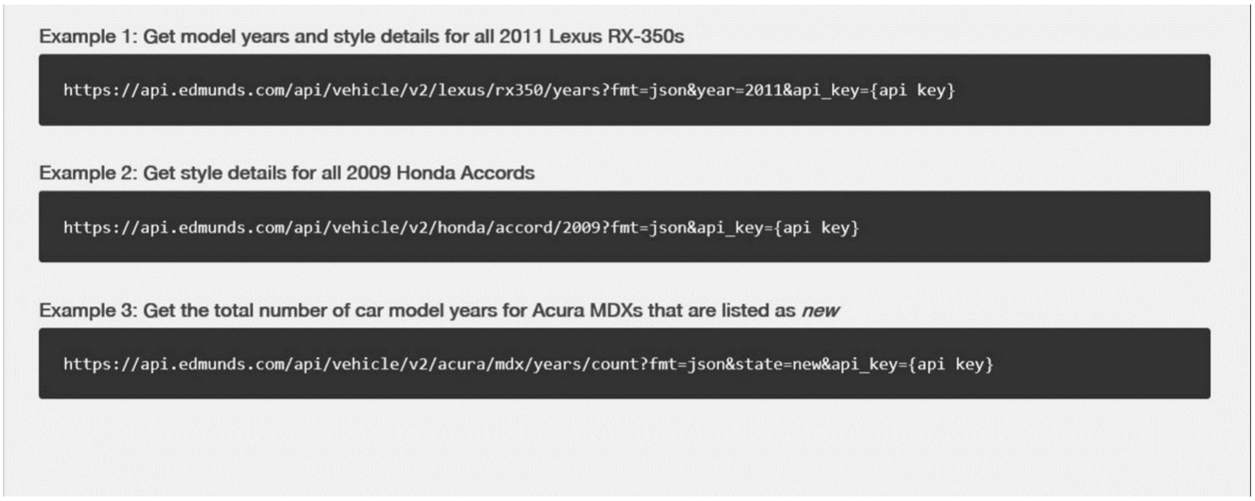
```
def value_of(sentiment):
if sentiment == 'positive': return 1
if sentiment == 'negative': return -1
return 0

def sentiment_score(review):
return sum ([value_of(tag) for sentence in dict_tagged_sentences for token in
sentence for tag in token[2]])
```

APPENDIX – II

SCREENSHOTS

Edmunds API



Example 1: Get model years and style details for all 2011 Lexus RX-350s

```
https://api.edmunds.com/api/vehicle/v2/lexus/rx350/years?fmt=json&year=2011&api_key={api key}
```

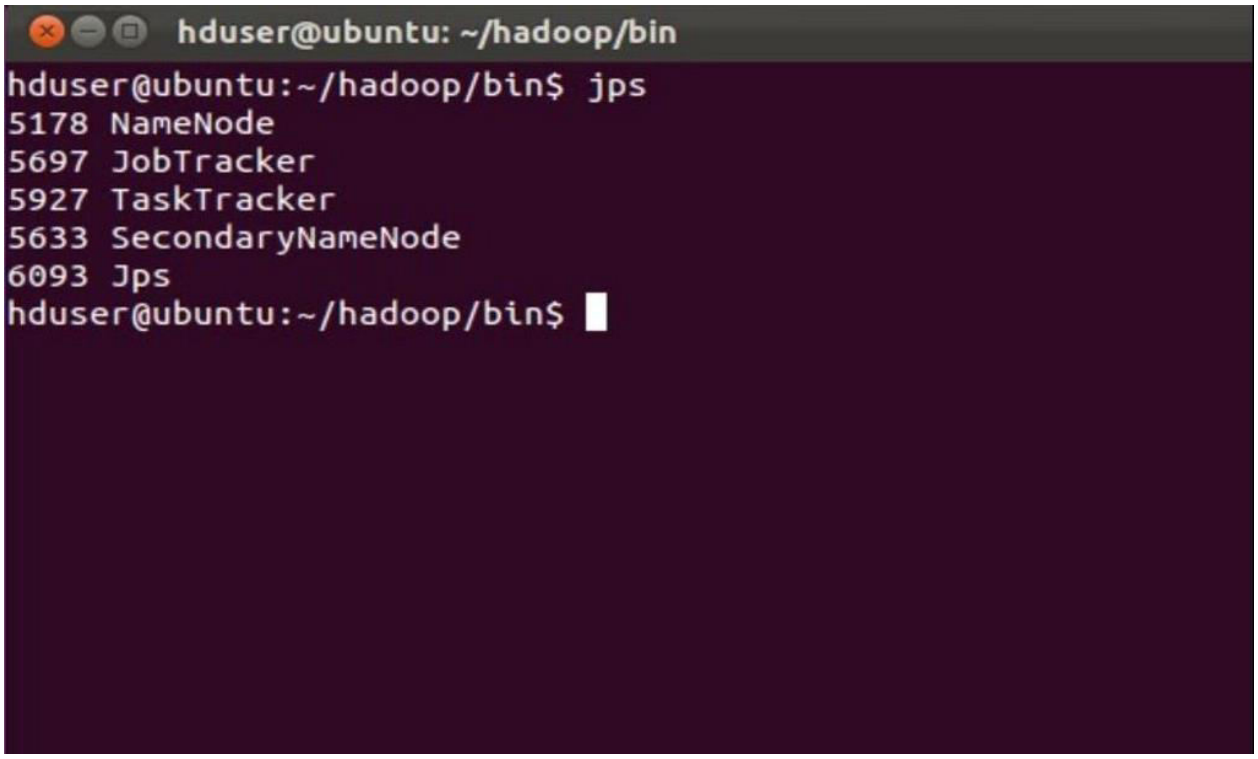
Example 2: Get style details for all 2009 Honda Accords

```
https://api.edmunds.com/api/vehicle/v2/honda/accord/2009?fmt=json&api_key={api key}
```

Example 3: Get the total number of car model years for Acura MDXs that are listed as *new*

```
https://api.edmunds.com/api/vehicle/v2/acura/mdx/years/count?fmt=json&state=new&api_key={api key}
```

Hadoop Installation



hduser@ubuntu: ~/hadoop/bin

```
hduser@ubuntu:~/hadoop/bin$ jps
5178 NameNode
5697 JobTracker
5927 TaskTracker
5633 SecondaryNameNode
6093 Jps
hduser@ubuntu:~/hadoop/bin$
```

REFERENCES

[1] **Considerations on construction ontologies.**

Author **:** Cicortas, A., Iordan, V., and Fortis, A.

Published **:** 2014

Page(s) **:** 115-132

[2] **Context related derivation of word senses.**

Author **:** Kunze` M ., Rozner D

Published **:** 2014

Page(s) **:** 1-17

[3] **Sentiment analysis and subjectivity. Handbook of Natural Language Processing, 2nd Edition.**

Author **:** Liu, B.

Published **:** 2010

Page(s) **:** 45-73

[4] **Negativity and extremity biases in impression formation: A review of Explanations 3rd Edition.**

Author **:** J. Skowronksi and D.E Carlston

Published **:** 2011

Page(s) **:** 284-310