1. Introduction

The Internet offers an effective, global platform for Ecommerce, communication, and opinion sharing. It has several blogs devoted to diverse topics like finance, politics, travel, education, sports, entertainment, news, history, environment, and so forth on which people frequently express their opinions in natural language. Mining through these terabytes of user review data is a challenging knowledge engineering task. However, automatic opinion mining has several useful applications. Hence, in recent years researchers have proposed approaches for mining user-expressed opinions from several domains such as movie reviews, political debates, restaurant food reviews, and product reviews, and so forth Generating user-query specific summaries is also an interesting application of opinion mining. Our focus in this paper is efficient feature extraction, sentiment polarity classification, and comparative feature summary generation of online product reviews.

Nowadays, several websites are available on which a variety of products are advertised and sold. Prior to making a purchase an online shopper typically browses through several similar products of different brands before reaching a final decision. This seemingly simple information retrieval task actually involves a lot of feature-wise comparison and decision making, especially since all manufacturers advertise similar features and competitive prices for most products.

However, most online shopping sites also allow users to post reviews of products purchased. There are also dedicated sites that post product reviews by experts as well as end users. These user reviews if appropriately classified and summarized can play an instrumental role in influencing a buyers' decision.

The main difficulty in analyzing these online users' reviews is that they are in the form of natural language. While natural language processing is inherently difficult, analyzing online unstructured textual reviews is even more difficult. Some of the major problems with processing unstructured text are dealing with spelling mistakes, incorrect punctuation, use of non dictionary words or slang terms, and undefined abbreviations. Often opinion is expressed in terms of partial phrases rather than complete grammatically correct sentences. So, the task of summarizing noisy, unstructured online reviews demands extensive Preprocessing.

In this paper, we apply a multistep approach to the problem of automatic opinion mining that consists of various phases like Preprocessing, Semantic feature-set extraction followed by opinion summarization and classification. The Multiword based approach for feature extraction used in the paper offers significant advantages over other contemporary approaches like the Apriori based approach and the seed-set expansion approach. Our approach significantly reduces the overhead of pruning compared to the Apriori-based approach and does not require prior domain knowledge for selecting an initial seed-set of features like the seed-set expansion approach. We have demonstrated empirically that the approach proposed in the paper can identify opinionated sentences from blog reviews with a high average precision of 91% outperforming the other two feature extraction strategies. The multistep approach can also classify the polarity of the reviews with a good average accuracy of 86%.

2. Problem Statement

- The growth of E-commerce has led to the invention of several websites that market and sells products as well as allows users to post reviews. It is typical for an online buyer to refer to these reviews before making a buying decision. Hence, automatic summarization of users' reviews has a great commercial significance. However, since the product reviews are written by no experts in an unstructured, natural language text, the task of summarizing them is challenging.

- The main difficulty in analyzing these online users' reviews is that they are in the form of natural language. While natural language processing is inherently difficult, analyzing online unstructured textual reviews is even more difficult.

3. Scope

The main difficulty in analyzing these online users' reviews is that they are in the form of natural language. While natural language processing is inherently difficult, analyzing online unstructured textual reviews is even more difficult. Some of the major problems with processing unstructured text are dealing with spelling mistakes, incorrect punctuation, use of non-dictionary words or slang terms, and undefined abbreviations. Hence the aim of this project is to perform natural language processing and guide the user to take buying decisions.

4. Literature Review

Classification and summarization of online blog reviews are very important to the growth of E-commerce and social networking applications. Earlier work on automatic text summarization has mainly focused on extraction of sentences that are more significant in comparison to others in a document corpus. The main approaches used to generate extractive summaries are:

(1) combinations of heuristics such as cue words, key words, title words or position.

(2) lexical chains, and

(3) rhetorical parsing theory.

However, it is important to note that the task of summarizing online product reviews is very different from traditional text summarization, as it does not involve extracting significant sentences from the source text. Instead, while summarizing user reviews, the aim is to first of all identify semantic features of products and next to generate a comparative summary of products based on feature-wise sentiment classification of the reviews that will guide the user in making a buying decision. In the authors have demonstrated that traditional unsupervised text classification techniques like naive Bayes, maximum entropy, and support vector machine do not perform well on sentiment or opinion classification and pointed out the necessity for feature-oriented classification. Thus, recent research work in opinion mining has focused on feature based extraction and summarization.

Opinion mining from users' reviews involves two main tasks—(1) identification of the opinion feature set and (2) sentiment analysis of users' opinions based on the identified features.

It has been observed that nouns and noun phrases (N and NP) frequently occurring in reviews are useful opinion features, while the adjectives and adverbs describing them are useful in classifying sentiment.

In order to extract nouns, noun phrases, and adjectives from review text, parts-of-speech (POS) tagging is performed. However, all nouns and noun phrases are not useful in mining and cannot directly be included in the feature set. So, the feature set is subsequently extracted using approaches that involve frequency analysis and/or use of domain knowledge as is discussed next.

Various methods exist in the literature to associate features with their corresponding descriptors. Hu and Liu proposed the nearby-adjective heuristic. Although this method is simple and fast, it may result in inaccuracies. So, supervised approaches to determine association have been proposed in recent years such as syntactic dependency parsing and syntactic tree templates.

5. Present Investigation

5.1 Methodology

In this section we will explain the system design of the opinion summarizer cum classifier implemented by us. We generated an opinion review database by crawling Some popular websites that categorically post product reviews by actual users. As shown in Figure 1, our product opinion summarizer has three main phases. These phases are

(1) preprocessing phase,

(2) feature extraction phase, and

(3) opinion summarization and classification phase.

These phases are briefly described next**.**

5.2 Preprocessing Phase

Online blog reviews posted by users frequently contain spelling errors and incorrect punctuation. Our next phase—the feature-extraction phase—requires parts-of-speech tagging which works at the sentence level. Thus, it becomes important to detect end of sentences. So, in this phase we performed basic cleaning tasks like sentence boundary detection and spell-error correction. Sentences normally end with punctuations like period (.), question mark (?), or exclamation mark (!). Sometimes bloggers overuse the "?" and "!" symbols for emphasis. For example, a blogger may post a review that says:

"It's surprising that the ebook reader does not have a touch screen !!!!"

In such cases we conflate the repetitive punctuation symbols to a single occurrence (i.e., "!!!!" is replaced by a single "!").

Several other considerations arise during the Preprocessing phase. The period (.) requires to be disambiguated as it may mean a full stop or a decimal point or an abbreviation (e.g., "Dr.," "Ltd."). Sometimes a single sentence straddles multiple lines as the user presses unnecessary return keys. In such cases we apply the sentence merge rules as proposed by Dey and Haque [14]. After sentence boundary detection, we perform spell-error correction using a word processor.

5.3 Feature Extraction Phase

In this phase we extract opinion features from the pre-processed review text obtained From the previous phase. We treat frequently occurring nouns (N) and noun phrases (NP) as possible opinion features and associated adjectives describing them as indicators of their opinion orientation.

We perform parts-of-speech (POS) tagging on the review sentences using the Link Grammar Parser [34]. The Link Grammar Parser is a well-known and efficient syntactic parser for English language (http://www.abisource.com/projects/link-grammar/). First, we extract all nouns (N) and noun phrases (NP) tagged by the Link Grammar Parser and identify the frequently occurring N and NP as possible

opinion features. By frequently occurring N and NP we mean those Ns and NP which occur at least five times in the users' reviews. We do not extract frequent itemsets from review sentence database using the Apriori based approach, since this method mines frequent features using a BOW (bag-of-words) approach and does not take into account the order in which the words of a phrase occur. Moreover, mining in this way would require ordering besides compactness and redundancy pruning [4, 8, 19]. We also do not use the seed-set expansion approach as it would require prior domain knowledge to specify a seed set. Instead we generate a frequent feature set using the multiword approach.

5.3.1 Feature-Based Opinion Mining Example elaborated.

Basic component of an opinion as three main components:

1) Opinion holder: A person who gives opinion about a particular object.

2) Object: An entity on which the opinion is given.

3) Opinion: A view, sentiment on an object given by an opinion holder

Classifying evaluative texts at the document level or the sentence level does not tell what the opinion holder likes and dislikes. A positive document on an object does not mean that the opinion holder has positive opinions on all aspects or features of the object. Likewise, a negative document does not mean that the opinion holder dislikes everything about the object. In an evaluative document (e.g., a product review), the opinion holder typically writes both positive and negative aspects of the object, although the general sentiment on the object may be positive or negative. To obtain such detailed aspects, going to the feature level is needed, three key mining tasks are:

1. Identifying *object features*: For instance, in the sentence “The picture quality of this camera is amazing,” the object feature is “picture quality”. A supervised pattern mining method is proposed. An unsupervised method is used. The technique basically finds frequent nouns and noun phrases as features, which are usually genuine features. Clearly, many information extraction techniques are also applicable, e.g., conditional random fields (CRF), hidden Markov models (HMM), and many others.

2. Determining opinion orientations*:* This task determines whether the opinions on the features are positive, negative or neutral. In the above sentence, the opinion on the feature “picture quality” is positive. Again, many approaches are possible. A lexicon-based approach has been shown to perform quite well. The lexicon-based approach basically uses opinion words and phrases in a sentence to determine the orientation of an opinion on a feature. A relaxation labeling based approach is given. Clearly, various types of supervised learning are possible approaches as well.

3. Grouping synonyms: As the same object features can be expressed with different words or phrases, this task groups those synonyms together. Not much research has been done on this topic.

5.4 Opinion Summarization and Classification Phase

In the previous phase we extracted opinion features, adjectives describing them, and any modifiers if present. We also generate a statistical feature-wise summary for each product which enables comparison of different brands selling similar products. In order to determine the sentiment polarity of an adjective describing an opinion feature we make use of SentiWordNet which is a lexical resource for opinion mining. SentiWordNet assigns three normalized sentiment scores: positivity, objectivity, and negativity to each synset of WordNet. Let us revisit the review sentence:

“The processor[.n] is[.v] significantly faster[.a], and the text[.n] is[.v] clear[.a].”

In this example, the SentiWordNet scores assigned to the appropriate usage of adjective clear is indicated as (P:0.625; O: 0.375; N: 0). Since the value of the positive polarity is highest, the adjective “clear” can be assigned a positive polarity. In this way, we generate a feature-orientation table (FO table) that records the opinion features and their corresponding descriptors of positive and negative polarities. The Table 1 shows the FO table entries for some of the features of product “Tablet.” The FO table, thus generated, enables us to generate feature-wise summary of a product or comparative summaries of different brands of similar products.

6. System Design

6.1 System Approach

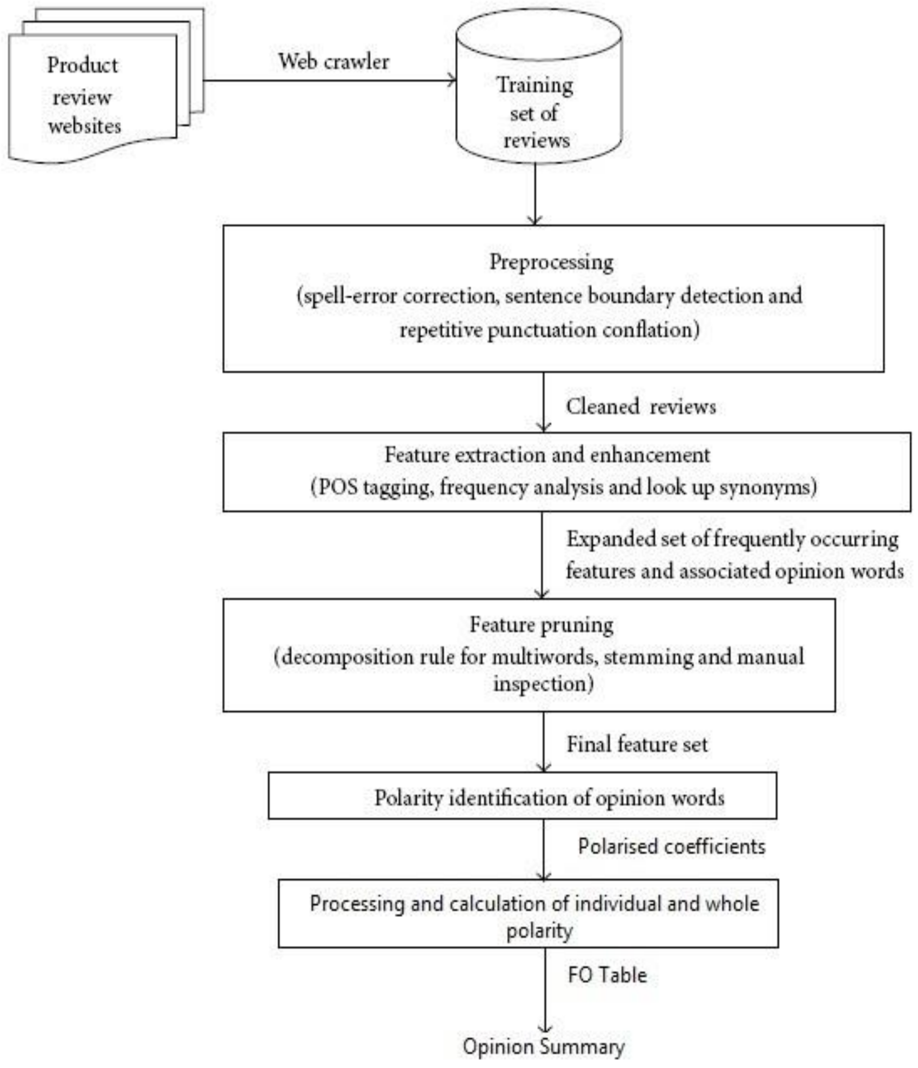

Product review websites
Web crawler
Training set of reviews
Preprocessing (spell-error correction, sentence boundary detection and repetitive punctuation conflation)
Cleaned reviews
Feature extraction and enhancement (POS tagging, frequency analysis and look up synonyms)
Expanded set of frequently occurring features and associated opinion words
Feature pruning (decomposition rule for multiwords, stemming and manual inspection)
Final feature set
Polarity identification of opinion words
Polarised coefficients
Processing and calculation of individual and whole polarity
FO Table
Opinion Summary

Fig.6.1

6.2 Block Diagram

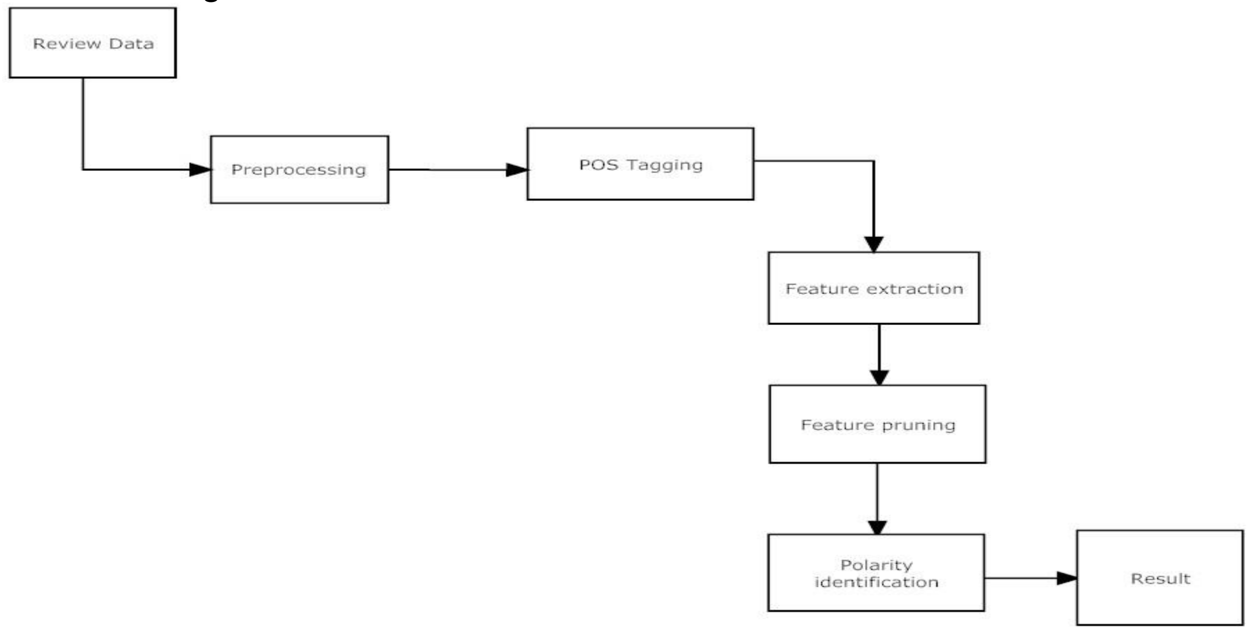


Fig:6.2

6.3 DFD Diagram

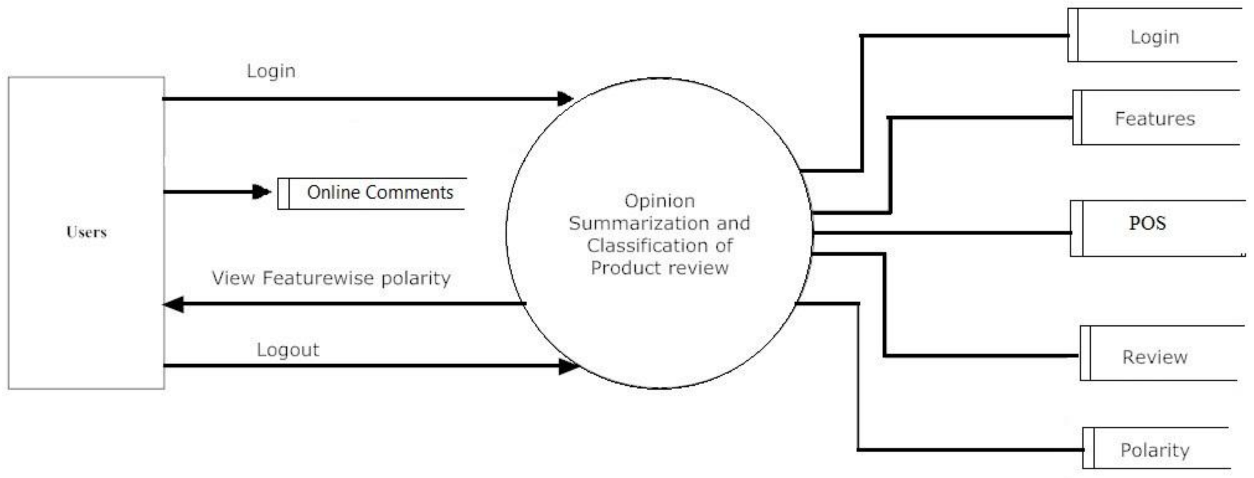

Users
Login
Online Comments
View Featurewise polarity
Logout
Opinion Summarization and Classification of Product review
Login
Features
POS
Review
Polarity

Fig:6.3

6.4 Use-Case Diagram

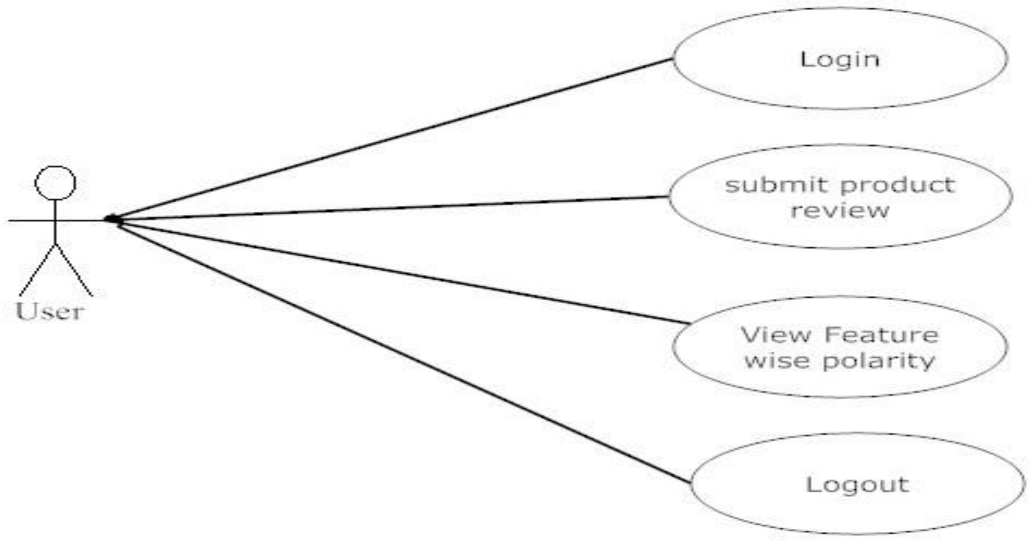


Fig:6.4

6.5 Component Diagram

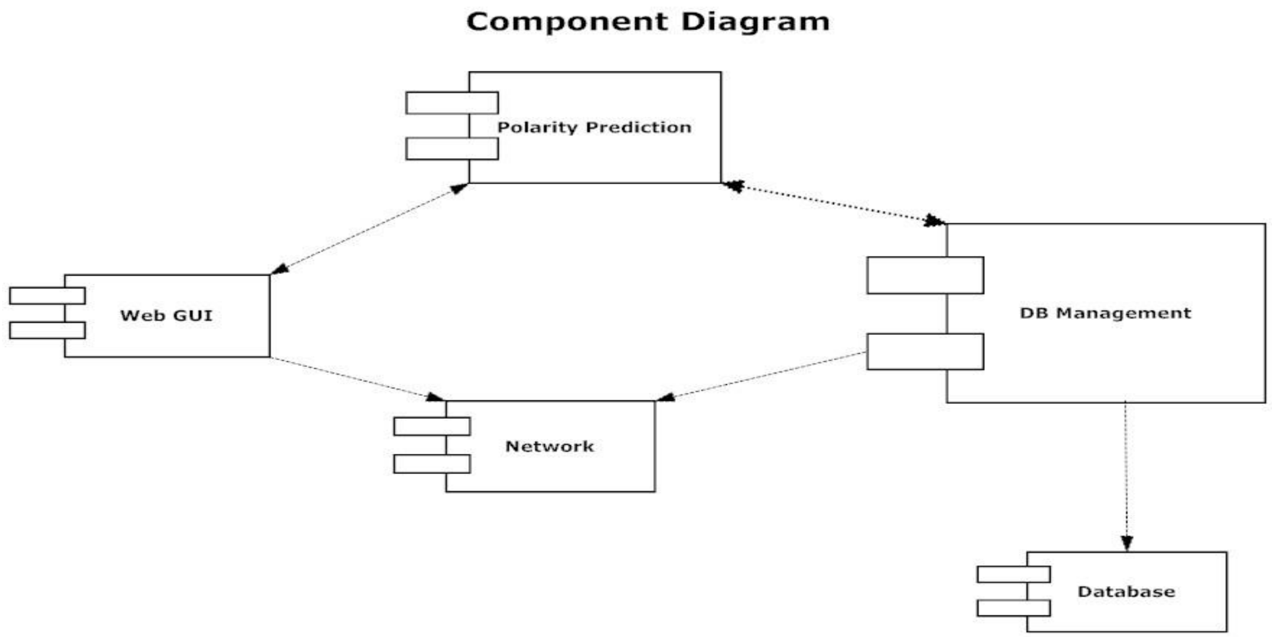


Fig:6.5

6.6 Activity Diagram

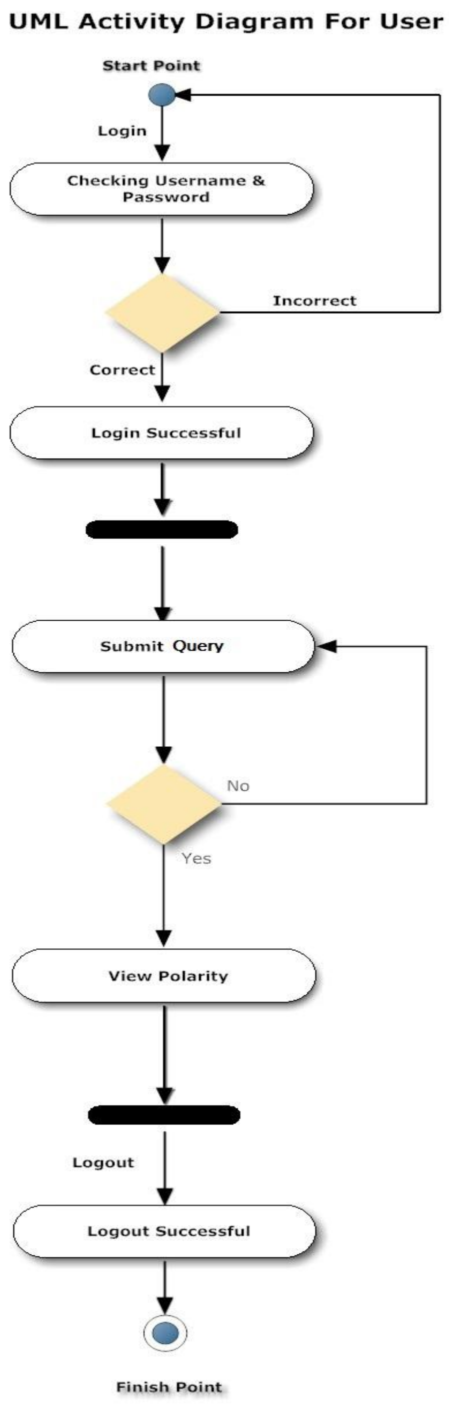


Fig: 6.6

7. Hardware and software requirements

7.1 Hardware Requirements

- 1 GB RAM.
- 200 GB HDD.
- Intel 1.66 GHz Processor Pentium 4.
- Internet Connectivity.

7.2 Software Requirements

- Windows XP, Windows 7,8.
- Visual Studio 2010+.
- MS SQL Server 2008+.
- Windows Operating System.

8. Testing Technology

System testing is a critical phase implementation. Testing of the system involves hardware devise and debugging of the computer programs and testing information processing procedures. Testing can be done with text data, which attempts to stimulate all possible conditions that may arise during processing. If structured programming Methodologies have been adopted during coding the testing proceeds from higher level to lower level of program module until the entire program is tested as unit. The testing methods adopted during the testing of the system were unit testing and integrated testing.

8.1 Unit Testing

Unit testing focuses first on the modules, independently of one another, to locate errors. This enables the tester to detect errors in coding and logical errors that is contained within that module alone. Those resulting from the interaction between modules are initially avoided.

8.2 Integration Testing

Integration testing is a systematic technique for constructing the program structure while at the same time to uncover the errors associated with interfacing. The objective is to take unit-tested module and build a program structure that has been detected by designing. It also tests to find the discrepancies between the system and its original objectives. Subordinate stubs are replaced one at time actual module. Tests were conducted at each module was integrated. On completion of each set another stub was replaced with the real module.

8.3 Functional Testing:

Functional testing is a technique in which all the functionalities of the program are tested to check whether all the functions that where proposed during the planning phase are full filled.

This is also to check that if all the functions proposed are working properly.

This is further done in two phases:

- One before the integration to see if all the unit components work properly
- Second to see if they still work properly after they have been integrated to check if some functional compatibility issues arise.

8.4 Performance Testing

- **Expected Result**
    - The client should be able to connect to the server properly without any problems.
    - The connection establishment between the mobile device and the server should take minimal time.
    - The mobile device should be able receive data from the server uninterruptedly.
    - Information provided by the application should be correct and as per the user's need.
- **Observation**
    - Connection can be established easily provided that the server is on.
    - The connection with the server takes time as it uses Internet connection.
    - Receiving data from the server takes time.
    - Information coming from the database is correct.

8.5 Load/Stress Testing

- **Expected Result**

  - Response time should be unaffected irrespective of the no of users.
  - The introduction of the newer clients should not make the server to work hap hazardously.
  - Continuous use of the server by different clients should not result into the server getting slowed down.
  - Response time should not be degraded if there is congestion in network.

- **Observation**

  - The speed of transmission was fine even when the newer clients were getting added. The response of the server was satisfying even with the introduction of newer client.

9. Technology

9.1 ASP.NET

ASP.NET is more than the next version of Active Server Pages (ASP); it is a unified Web development platform that provides the services necessary for developers to build enterprise-class Web applications. While ASP.NET is largely syntax-compatible with ASP, it also provides a new programming model and infrastructure that enables a powerful new class of applications. You can migrate your existing ASP applications by incrementally adding ASP.NET functionality to them.

ASP.NET is a compiled .NET Framework -based environment. You can author applications in any .NET Framework compatible language, including Visual Basic and Visual C#. Additionally, the entire .NET Framework platform is available to any ASP.NET application. Developers can easily access the benefits of the .NET Framework, which include a fully managed, protected, and feature-rich application execution environment, simplified development and deployment, and seamless integration with a wide variety of languages.

9.2 Microsoft SQl Server 2012

Business today demands a different kind of data management solution. Performance scalability, and reliability are essential, but businesses now expect more from their key IT investment. SQL Server 2008 exceeds dependability requirements and provides innovative capabilities that increase employee effectiveness, integrate heterogeneous IT ecosystems, and maximize capital and operating budgets. SQL Server 2008 provides the enterprise data management platform your organization needs to adapt quickly in a fast changing environment. Benchmarked for scalability, speed, and performance, SQL Server 2008 is a fully enterprise-class database product, providing core support for Extensible Markup Language (XML) and Internet queries.

10. Results and discussions

Opinions are so important that whenever one needs to make a decision, one wants to hear others' opinions. This is true for both individuals and organizations. The technology of opinion mining thus has a tremendous scope for practical applications.

- ***Individual consumers*:**

If an individual wants to purchase a product, it is useful to see a summary of opinions of existing users so that he/she can make an informed decision. This is better than reading a large number of reviews to form a mental picture of the strengths and weaknesses of the product. He/she can also compare the summaries of opinions of competing products, which is even more useful.

- ***Organizations and businesses*:**

Opinion mining is equally, if not even more, important to businesses and organizations. For example, it is critical for a product manufacturer to know how consumers perceive its products and those of its competitors. This information is not only useful for marketing and product benchmarking but also useful for product design and product developments.

TABLE 1: Partial feature orientation table for tablets.

Feature	Descriptors with positive polarity	Descriptors with negative polarity
Screen/display/ touchscreen	Good, sensitive, nice, bright	Fragile, bad
Price/cost	Affordable, low, good, cheap	High, expensive, more, unaffordable
Processor	Fast, efficient	Slow, incompatible
Front camera/face time camera	Awesome, great, high resolution	Inferior, low resolution
Battery life	Good, long, sufficient	Bad, limited, poor, short
Wireless connectivity/wi-fi	Good, fast, simple, free, 3G, 4G, seamless, reliable	Poor, slow, cumbersome

11. Historical Background And Need

Textual information in the world can be broadly classified into two main categories, *facts* and *opinions*. Facts are objective statements about entities and events in the world. Opinions are subjective statements that reflect people's sentiments or perceptions about the entities and events.

Much of the existing research on text information processing has been (almost exclusively) focused on mining and retrieval of factual information, e.g., information retrieval, Web search, and many other text mining and natural language processing tasks. Little work has been done on the processing of opinions until only recently. Yet, opinions are so important that whenever one needs to make a decision one wants to hear others' opinions. This is not only true for individuals but also true for organizations. One of the main reasons for the lack of study on opinions is that there was little opinionated text before the World Wide Web. Before the Web, when an individual needs to make a decision, he/she typically asks for opinions from friends and families. When an organization needs to find opinions of the general public about its products and services, it conducts surveys and focused groups. With the Web, especially with the explosive growth of the user generated content on the Web, the world has changed. One can post reviews of products at merchant sites and express views on almost anything in Internet forums, discussion groups, and blogs, which are collectively called the *user generated content*.

Now if one wants to buy a product, it is no longer necessary to ask one's friends and families because there are plentiful of product reviews on the Web which give the opinions of the existing users of the product. For a company, it may no longer need to conduct surveys, to organize focused groups or to employ external consultants in order to find consumer opinions or sentiments about its products and those of its competitors. Finding opinion sources and monitoring them on the Web, however, can still be a formidable task because a large number of diverse sources exist on the Web and each source also contains a huge volume of information. In many cases, opinions are hidden in long forum posts and blogs. It is very difficult for a human reader to find relevant sources, extract pertinent sentences, read them, summarize them and organize them into usable forms. An automated opinion mining and summarization system is thus needed. *Opinion mining*, also known as *sentiment analysis*, grows out of this need.

11.1 DIFFERENT LEVELS OF OPINION MINING

Document level Opinion Mining The basic information unit is a single document of opinionated text where document level classification is a single review about a topic is considered. But in the case of forums or blogs, comparative sentences may appear and customers may compare one product with another that has similar characteristics and hence document level analysis is not desirable in forums and blogs. Therefore subjectivity/objectivity classification is very important in this type of classification.

Sentence level Opinion Mining In sentence level Opinion Mining, the polarity of each sentence is calculated. The same document level classification methods can be applied to the sentence level classification problem also but Objective and subjective sentences must be found out. The subjective sentences contain opinion words which help in determining the sentiment about the entity. After which the polarity classification is done into positive and negative classes.

Phrase level Opinion Mining The phrase level sentiment classification is a much more pinpointed approach to opinion mining. The phrases that contain opinion words are found out and a phrase level classification is done. But in some other cases, where contextual polarity also matters, the result may not be fully accurate. Negation of words can occur locally. But if there are sentences with negating words which are far apart from the opinion words, phrase level analysis is not desirable. The process is Identifying Opinion Words, the role of negation words and But Clauses.

12. CODE

12.1 Login.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Login.aspx.cs"
Inherits="Login" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Login Here!!!</title>
    <link href="Styles/Site.css" rel="stylesheet" type="text/css" />
    <script src="Scripts/jquery-1.4.1.min.js" type="text/javascript"></script>
</head>
<body>
    <form id="form1" runat="server">
    <div class="page">
        <div class="header">
            <div class="title">
                <h1>
                    Sentiment Analysis For Car Review
                </h1>
            </div>
            <div class="loginDisplay">
                <asp:LoginView ID="HeadLoginView" runat="server"
EnableViewState="false">
                    <AnonymousTemplate>
                        [ <a href="~/Account/Login.aspx" id="HeadLoginStatus"
runat="server">Log In</a>
                        ]
                    </AnonymousTemplate>
                    <LoggedInTemplate>
                        Welcome <span class="bold">
                            <asp:LoginName ID="HeadLoginName" runat="server" />
                        </span>! [
                        <asp:LoginStatus        ID="HeadLoginStatus"        runat="server"
LogoutAction="Redirect" LogoutText="Log Out"
                            LogoutPageUrl="~/" />
                        ]
                    </LoggedInTemplate>
                </asp:LoginView>
            </div>
            <div class="clear hideSkiplink">
                <asp:Menu   ID="NavigationMenu"   runat="server"   CssClass="menu"
EnableViewState="false"
                    IncludeStyleBlock="false" Orientation="Horizontal">
                    <Items>
                        <%--<asp:MenuItem NavigateUrl="~/Default.aspx" Text="Home" />
                    <asp:MenuItem NavigateUrl="~/About.aspx" Text="About" />--%>
                    </Items>
                </asp:Menu>
            </div>
        </div>
        <div class="clear">
        </div>
        <div class="main">
            <fieldset>
                <legend>Login Information</legend>
                <table> <tr><td></td><td>
                            <asp:Label             ID="lblShow"             runat="server"
ForeColor="Red"></asp:Label>
                        </td><td></td></tr><tr>
                        <td>
                            Username :
                        </td>
                        <td>
                            <asp:TextBox  ID="txtUsername"  runat="server"  Height="30px"
Width="150px"></asp:TextBox>
                        </td><td>  </td></tr><tr><td>
                            Password :
                        </td><td>
                            <asp:TextBox  ID="txtPassword"  runat="server"  Height="30px"
Width="150px"
                                TextMode="Password"></asp:TextBox>
                        </td><td></td></tr><tr><td></td>
                        <td>
                            <asp:Button    ID="btnLogin"    runat="server"    Height="32px"
Text="Login" Width="82px"
                                OnClick="btnLogin_Click" />
                        </td><td></td></tr><tr><td></td><td>
                        <a href="Registration.aspx">Create New Account Here</a>
                    </td><td></td> </tr></table>
            </fieldset></div><div class="clear">
        </div></div></form>
</body>
</html>
```

12.2 Login.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Data.SqlClient;


public partial class Login : System.Web.UI.Page
{
    SqlConnection conn = new
SqlConnection(System.Configuration.ConfigurationManager.ConnectionStrings["
ConnString"].ConnectionString);
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            if (Request.QueryString["msg"] == "Add")
            {
                lblShow.Text = "Information Save Successfully";
            }
            if (Request.QueryString["msg"] == "logout")
            {
                Session.Abandon();
            }
        }
    }
    protected void btnLogin_Click(object sender, EventArgs e)
    {
        string query = "select * from Users where username='" + txtUsername.Text
+ "' and password='" + txtPassword.Text + "'";
        SqlDataAdapter da = new SqlDataAdapter(query, conn);
        DataTable dt = new DataTable();
        da.Fill(dt);
        if (dt.Rows.Count > 0)
        {
            Session["ID"] = Convert.ToInt32(dt.Rows[0]["Id"]);
            Session["name"]     =     dt.Rows[0]["Fname"].ToString()     +     "     "     +
dt.Rows[0]["lName"].ToString();
            Response.Redirect("index.aspx");
        }
        else
        {
            lblShow.Text = "Invalid Username and Password";
            txtUsername.Text = "";
            txtPassword.Text = "";
            txtUsername.Focus();
        }
    }
}
```

12.3 Review.aspx

```
<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.master"
AutoEventWireup="true"
    CodeFile="Review.aspx.cs" Inherits="_Default" %>

<asp:Content                    ID="HeaderContent"                    runat="server"
ContentPlaceHolderID="HeadContent">
</asp:Content>
<asp:Content                    ID="BodyContent"                      runat="server"
ContentPlaceHolderID="MainContent">
    <asp:ScriptManager ID="ScriptManager1" runat="server">
    </asp:ScriptManager>
    <table>
        <tr><td></td><td>
                Company name
            </td><td>
                Model Name
            </td></tr><tr><td>
                Select Car:
            </td>   <td>
                <asp:UpdatePanel ID="UpdatePanel1" runat="server">
                    <ContentTemplate>
                        <asp:DropDownList        ID="DropDownList1"        runat="server"
Height="28px" Width="121px" AutoPostBack="True"

OnSelectedIndexChanged="DropDownList1_SelectedIndexChanged"
DataSourceID="SqlDataSource1"
                            DataTextField="Name" DataValueField="C_Id">
                        </asp:DropDownList>
```

```
          <asp:SqlDataSource     ID="SqlDataSource1"     runat="server"
ConnectionString="<%$ ConnectionStrings:ConnString %>"
            SelectCommand="SELECT     [C_Id],     [Name]     FROM
[CompanyMaster]"></asp:SqlDataSource>
        </ContentTemplate>
        <%--  <Triggers>
          <asp:AsyncPostBackTrigger           ControlID="DropDownList1"
EventName="DropDownList1_SelectedIndexChanged" />
        </Triggers>--%>
      </asp:UpdatePanel>
    </td><td>
      <asp:UpdatePanel ID="UpdatePanel2" runat="server">
        <ContentTemplate>
          <asp:DropDownList      ID="DropDownList2"      runat="server"
Height="28px" Width="121px" AutoPostBack="True"
            DataSourceID="SqlDataSource2"        DataTextField="Model"
DataValueField="M_Id"
OnSelectedIndexChanged="DropDownList2_SelectedIndexChanged" />
          <asp:SqlDataSource     ID="SqlDataSource2"     runat="server"
ConnectionString="<%$ ConnectionStrings:ConnString %>"
            SelectCommand="SELECT    c.c_id,m.m_id,m.model    FROM
[ModelMaster]m join [companymaster]c on(m.c_id=c.c_id) WHERE (m.[C_Id] =
@C_Id)">
            <SelectParameters>
              <asp:ControlParameter           ControlID="DropDownList1"
DefaultValue="1" Name="C_Id" PropertyName="SelectedValue"
                Type="Int32" />
            </SelectParameters>
          </asp:SqlDataSource>
        </ContentTemplate>
        <Triggers>
          <asp:AsyncPostBackTrigger           ControlID="DropDownList1"
EventName="SelectedIndexChanged" />
        </Triggers>
      </asp:UpdatePanel>
    </td></tr><tr> <td></td>
    <td>
      <asp:Button ID="btnSubmit" runat="server" OnClick="btnSubmit_Click"
Text="Submit"
        Width="67px" />
    </td><td></td></tr>
  </table>
  <div>
    <asp:Panel ID="Panel1" runat="server">
    </asp:Panel>
  </div>
```

```
    <div>
        <asp:TextBox ID="txtPara" runat="server" Height="132px" Width="926px"
            TextMode="MultiLine"></asp:TextBox></div>
    <div>
        <br />
        <asp:Button ID="btnPos" runat="server" Height="25px" Width="120px"
OnClick="btnPos_Click"
            Text="POS Taging" />
        <br /><br />
        <asp:Button ID="btnFeature" runat="server" Height="25px" Width="120px"
OnClick="btnFeature_Click"
            Text="Feature Score" />
    </div>
</asp:Content>
```

12.4 Review.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Net;
using System.Text.RegularExpressions;
using Newtonsoft.Json;
using Newtonsoft.Json.Linq;
using HtmlAgilityPack;
using java.io;
using edu.stanford.nlp.tagger.maxent;
using edu.stanford.nlp.ling;
using javac = com.sun.tools.javac.util;
using java.util;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;
using System.Text.RegularExpressions;
public partial class _Default : System.Web.UI.Page
{
    String posstring;
    string newString;
    SqlConnection conn = new
SqlConnection(System.Configuration.ConfigurationManager.ConnectionStrings["C
onnString"].ConnectionString);
```

```
    public const string Model = @"C:\Users\Royal\Documents\Visual Studio 2013\Projects\CarReview_App\Bin\english-caseless-left3words-distsim.tagger";
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void btnSubmit_Click(object sender, EventArgs e)
    {
        string htmlCode;

        List<string> lst = new List<string>();
        // var result = arr[0]["imdbID"];
        string uri = String.Format("http://www.zigwheels.com/reviews/user-reviews/{0}/{1}", DropDownList1.SelectedItem, DropDownList2.SelectedItem);
        var getHtmlWeb = new HtmlWeb();
        var document = getHtmlWeb.Load(uri);
        Literal Literal1 = new Literal();
        Literal1.ID = "Literal1";


        foreach (HtmlNode para in document.DocumentNode.SelectNodes ("//div[@class='col-lg-12 col-md-12 col-sm-12 rw-description MT10']//div[@class='rw-heading']//a[@href]"))
        {
            HtmlAttribute att = para.Attributes["href"];

            lst.Add(att.Value);
        }
        foreach (var item in lst)
        {
            string newuri = string.Format("{0}", item);
            var htmlwebdoc = new HtmlWeb();
            var doc = htmlwebdoc.Load(newuri);
            foreach (HtmlNode para in doc.DocumentNode.SelectNodes("//div[@class='col-lg-12 col-md-12 col-sm-12 rw-description MT10']//p[@class='rwt-para']"))
            {
                int paracount = doc.DocumentNode.SelectNodes("//div[@class='col-lg-12 col-md-12 col-sm-12 rw-description MT10']//p[@class='rwt-para']").Count;
                Literal1.Text += String.Format(@"<p>{0}</p>", para.InnerText);
            }
        }
        Panel1.Controls.Add(Literal1);
    }
    private void TagReader(Reader reader)
    {
        var tagger = new MaxentTagger(Model);
        //List obj = (List)MaxentTagger.tokenizeText(reader);
        foreach (ArrayList sentence in MaxentTagger.tokenizeText(reader).toArray())
        {
            var tSentence = tagger.tagSentence(sentence);
            System.Console.WriteLine(Sentence.listToString(tSentence, false));
            posstring = (Sentence.listToString(tSentence, false));
            newString = newString + posstring;
            System.Console.WriteLine();
        }

    }

    protected void DropDownList1_SelectedIndexChanged(object sender,
EventArgs e)
    {

    }
    protected void DropDownList2_SelectedIndexChanged(object sender,
EventArgs e)
    {

    }
    protected void btnPos_Click(object sender, EventArgs e)
    {

        string replacestr = Regex.Replace(txtPara.Text, "[^a-zA-Z0-9_]+", " ");
        TagReader(new StringReader(replacestr));
        txtPara.Text = newString;
    }
    protected void btnFeature_Click(object sender, EventArgs e)
    {
        truncate();
        string[] para = txtPara.Text.Split('.');
        List<string> lstNN = new List<string>();
        List<string> lstJJ = new List<string>();
        for (int i = 0; i < para.Length; i++)
        {
            //if (lstNN.Count > lstJJ.Count && lstNN.Count > 0 && lstJJ.Count > 0)
            //{
            //    lstJJ.RemoveAt(lstJJ.Count - 1);
            //}
            //else if (lstNN.Count < lstJJ.Count && lstJJ.Count > 0 && lstNN.Count > 0)
            //{ lstNN.RemoveAt(lstNN.Count - 1); }
            string line = para[i];
            string[]         word         =         line.Split(new[]         {         '         '         },
StringSplitOptions.RemoveEmptyEntries);


            for (int j = 0; j < word.Length; j++)
            {

                int startindex = word[j].IndexOf("/");
                if (startindex < 0) { startindex = 0; }
                int lastindex = word[j].Length;
                string NewPos = word[j].Remove(0, startindex);
                string NewWord = word[j].Substring(0, startindex);
                switch (NewPos)
                {
                    case "/NN":
                        {
                            System.Console.Write("NN");
                            lstNN.Add(NewWord);

                            break;
                        }
                    case "/NNS":
                        {
                            System.Console.Write("NNS");
                            lstNN.Add(NewWord);

                            break;
                        }
                    case "/JJ":
                        {
                            System.Console.Write("JJ");
                            lstJJ.Add(NewWord);
                            string query = "select * from SentiWordNet where SynsetTerms='"
+ NewWord + "'";

                            break;
                        }
                    case "/JJR":
                        {
                            System.Console.Write("JJR");
                            lstJJ.Add(NewWord);
                            string query = "select * from SentiWordNet where SynsetTerms='"
+ NewWord + "'";
```

```
                            break;
                        }

                    case "/JJS":
                        {
                            System.Console.Write("JJS");
                            lstJJ.Add(NewWord);
                            string query = "select * from SentiWordNet where SynsetTerms='"
+ NewWord + "'";

                            break;
                        }

                    default:
                        { break; }

                }


            }
            for (int jj = 0; lstJJ.Count < lstNN.Count; jj++)
            {
                if (lstNN.Count > lstJJ.Count && lstNN.Count > 0)
                {
                    lstNN.RemoveAt(lstNN.Count - 1);
                }
            }
            for (int nn = 0; lstNN.Count < lstJJ.Count; nn++)
            {
                if (lstJJ.Count > lstNN.Count && lstJJ.Count > 0)
                {
                    lstJJ.RemoveAt(lstJJ.Count - 1);
                }
            }

            for (int k = 0; k < lstNN.Count; k++)
            {
                // int n= lstJJ.Count;


                SqlDataAdapter da = new SqlDataAdapter("insert into
FO_Table(Feature) values('" + lstNN[k] + "')", conn);
                conn.Open();
                da.SelectCommand.ExecuteNonQuery();
                conn.Close();
                string query = "select * from SentiWordNet where SynsetTerms='" +
lstJJ[k] + "'";
               // string query = "select max(PosScore)as PosScore,max(NegScore)as
NegScore from SentiWordNet where SynsetTerms='" + lstJJ[k] + "'";
                SqlDataAdapter da1 = new SqlDataAdapter(query, conn);
                conn.Open();
                da1.SelectCommand.ExecuteNonQuery();
                DataTable dt = new DataTable();
                da1.Fill(dt);
                conn.Close();
                if (dt.Rows.Count > 0)
                {
                    double posvalue = Convert.ToDouble(dt.Rows[0]["PosScore"]);
                    double Negvalue = Convert.ToDouble(dt.Rows[0]["NegScore"]);
                    if (posvalue > Negvalue)
                    {
                        string feature = lstNN[k];
                        string NewWord = lstJJ[k];
                        conn.Open();
                        SqlDataAdapter da3 = new SqlDataAdapter(new
SqlCommand("update FO_Table set PositivePolarity=" + 1 + ",value='" + NewWord
+ "' where Feature='" + feature + "'", conn));
                        da3.SelectCommand.ExecuteNonQuery();
                        conn.Close();
                    }
                    else if (Negvalue > posvalue || Negvalue != 0)
                    {
                        string feature = lstNN[k];
                        string NewWord = lstJJ[k];
                        conn.Open();
                        SqlDataAdapter da3 = new SqlDataAdapter("update FO_Table set
NegativePolarity=" + 1 + ",value='" + NewWord + "' where feature='" + feature + "'",
conn);
                        da3.SelectCommand.ExecuteNonQuery();
                        conn.Close();
                    }
                }

            }

            lstNN.RemoveAll(item => item == lstNN[0]);
            lstJJ.RemoveAll(item => item == lstJJ[0]);

        }
        Response.Redirect("Result.aspx");
```

```
    }

    protected void truncate()
    {
        SqlCommand cmd = new SqlCommand("truncate table FO_Table", conn);
        conn.Open();
        cmd.ExecuteNonQuery();
        conn.Close();
    }
}
```

12.5 Result.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.master"
AutoEventWireup="true"
    CodeFile="Result.aspx.cs" Inherits="Default2" %>

<%@ Register Assembly="System.Web.DataVisualization, Version=4.0.0.0,
Culture=neutral, PublicKeyToken=31bf3856ad364e35"
    Namespace="System.Web.UI.DataVisualization.Charting" TagPrefix="asp" %>
<asp:Content ID="Content1" ContentPlaceHolderID="HeadContent"
runat="Server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="MainContent"
runat="Server">
    <div>
        <h3 style="background: none !important;">
            Result</h3>
        <br />
        <asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="False"
BackColor="White"
            BorderColor="#CCCCCC" BorderStyle="None" BorderWidth="1px"
CellPadding="4" ForeColor="Black"
            Width="100%" GridLines="Horizontal">
            <Columns>
                <asp:BoundField DataField="Feature" HeaderText="Feature"
SortExpression="Feature">
                    <ItemStyle HorizontalAlign="Center" />
                </asp:BoundField>
                <asp:BoundField DataField="PositivePolarity" HeaderText="Positive
Polarity" SortExpression="PositivePolarity">
                    <ItemStyle HorizontalAlign="Center" />
                </asp:BoundField>
                <asp:BoundField DataField="NegativePolarity" HeaderText="Negative
Polarity" SortExpression="NegativePolarity">
                    <ItemStyle HorizontalAlign="Center" />
                <asp:BoundField>
            </Columns>
            <FooterStyle BackColor="#CCCC99" ForeColor="Black" />
            <HeaderStyle  BackColor="#333333"  Font-Bold="True"  ForeColor="White"
/>
            <PagerStyle BackColor="White" ForeColor="Black" HorizontalAlign="Right"
/>
            <SelectedRowStyle            BackColor="#CC3333"            Font-Bold="True"
ForeColor="White" />
            <SortedAscendingCellStyle BackColor="#F7F7F7" />
            <SortedAscendingHeaderStyle BackColor="#4B4B4B" />
            <SortedDescendingCellStyle BackColor="#E5E5E5" />
            <SortedDescendingHeaderStyle BackColor="#242121" />
        </asp:GridView>
    </div><div><div>
            <h3>
                Graph</h3>
        </div>
        <br />
        <asp:Chart ID="Chart1" runat="server" Width="325px">
            <ChartAreas>
                <asp:ChartArea Name="ChartArea1">
                </asp:ChartArea>
            </ChartAreas>
            <Series>
                <asp:Series             IsValueShownAsLabel=true             Name="Series1"
BorderColor="180, 26, 59, 105" ChartType="Pie">
                </asp:Series>
            </Series>
        </asp:Chart>
    </div>
</asp:Content>
```

12.6 Result.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Data.SqlClient;
using System.Web.UI.DataVisualization.Charting;
```

```
public partial class Default2 : System.Web.UI.Page
{
   SqlConnection conn = new SqlConnection(System.Configuration.ConfigurationManager.ConnectionStrings["connString"].ConnectionString);
   protected void Page_Load(object sender, EventArgs e)
   {
      string query = "select Feature,COUNT(*) as countOfColoumn , COUNT(PositivePolarity)as PositivePolarity,COUNT(NegativePolarity)as NegativePolarity from FO_Table  where value is not null group by feature";

      SqlDataAdapter da = new SqlDataAdapter(query, conn);
      DataTable dt = new DataTable();
      da.Fill(dt);

      GridView1.DataSource = dt;
      GridView1.DataBind();
      int xAxisValue = 0;
      int yAxisPositiveValue = 0;
      int yAxisNegativeValue = 0;
      for (int i = 0; i < dt.Rows.Count; i++)
      {
         xAxisValue += Convert.ToInt32(dt.Rows[i]["countOfColoumn"]);
         yAxisPositiveValue += Convert.ToInt32(dt.Rows[i]["PositivePolarity"]);
         yAxisNegativeValue += Convert.ToInt32(dt.Rows[i]["NegativePolarity"]);
      }

      Chart1.Series["Series1"].IsValueShownAsLabel = true;
      Chart1.Series["Series1"].Points.AddXY(xAxisValue, yAxisPositiveValue);
      Chart1.Series["Series1"].Points.AddXY(xAxisValue, yAxisNegativeValue);
      Chart1.DataBind();
      Chart1.Series["Series1"].Points[0].LabelToolTip = "Positive Polarity";
      Chart1.Series["Series1"].Points[1].LabelToolTip = "Negative Polarity";
   }
}
```

13. WORKING

13.1 Login

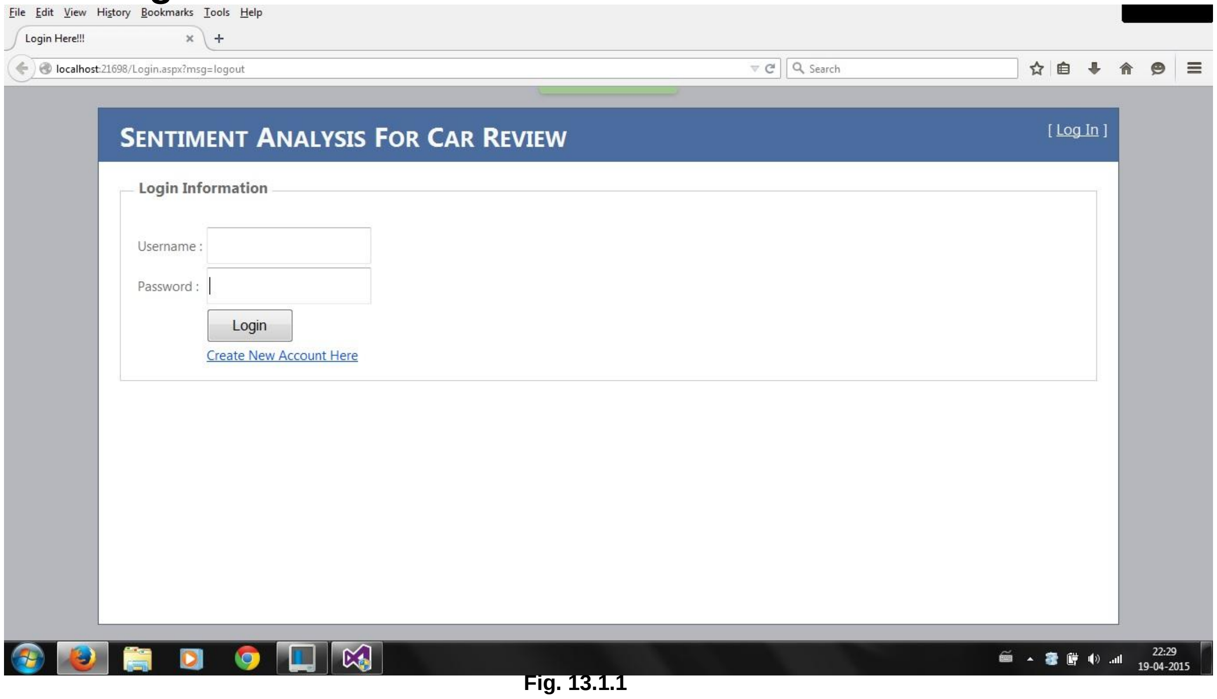

[redacted]
File Edit View History Bookmarks Tools Help
Login Here!!!
localhost:21698/Login.aspx?msg=logout
Search
SENTIMENT ANALYSIS FOR CAR REVIEW
[Log In]
Login Information
Username :
Password :
Login
Create New Account Here
22:29
19-04-2015

Fig. 13.1.1

13.2 Home Page

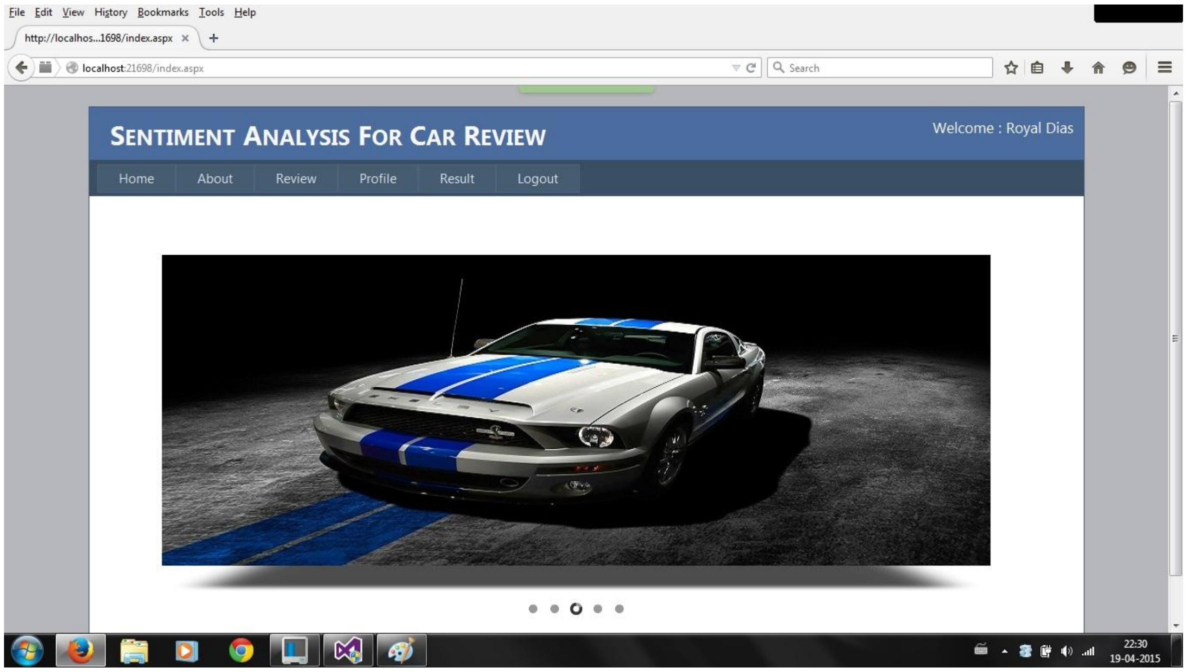
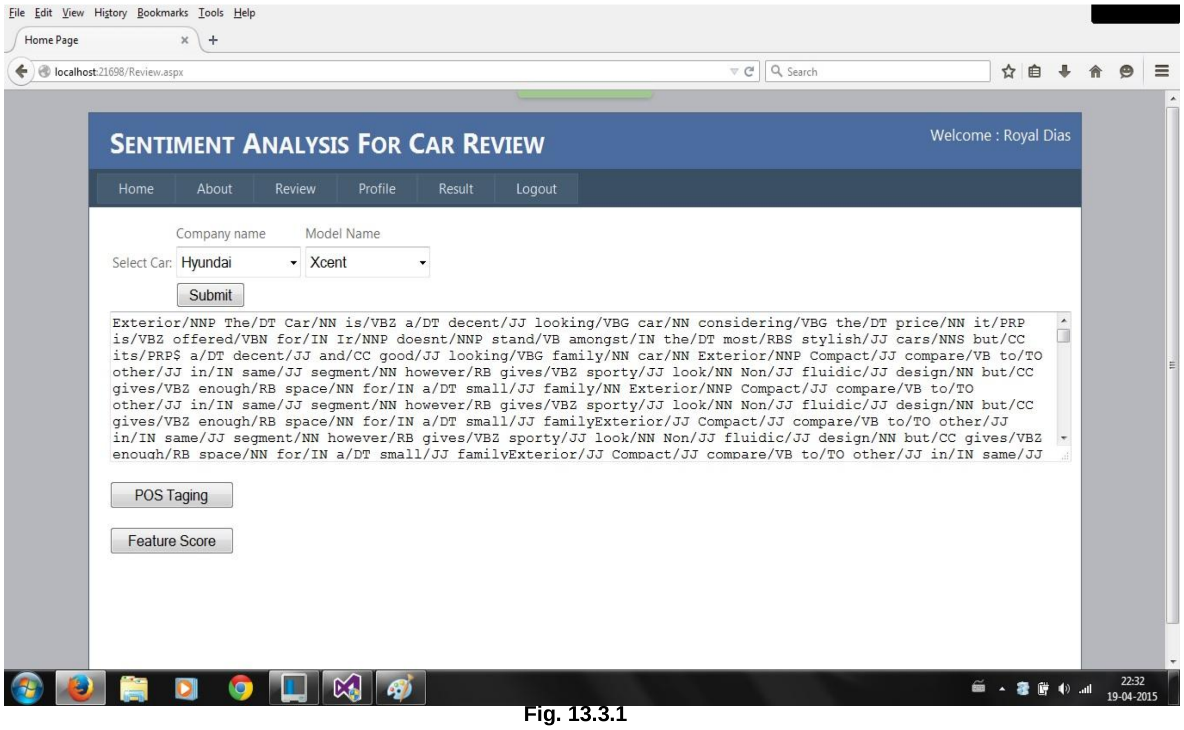

[redacted]
File Edit View History Bookmarks Tools Help
http://localhos...1698/index.aspx
localhost:21698/index.aspx
Search
SENTIMENT ANALYSIS FOR CAR REVIEW
Welcome : Royal Dias
Home
About
Review
Profile
Result
Logout
22:30
19-04-2015

Fig. 13.2.1

13.3 Review Page



[redacted]

Company name: Hyundai

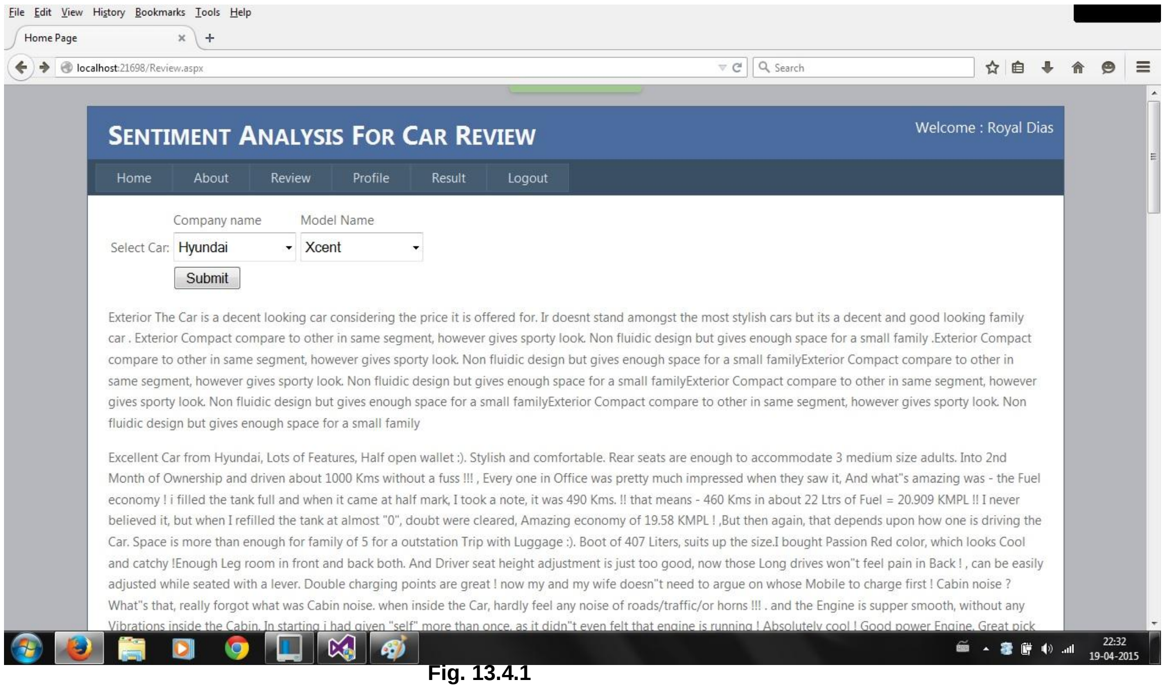
Model Name: Xcent

Select Car: Hyundai Xcent

```
Exterior/NNP The/DT Car/NN is/VBZ a/DT decent/JJ looking/VBG car/NN considering/VBG the/DT price/NN it/PRP
is/VBZ offered/VBN for/IN Ir/NNP doesnt/NNP stand/VB amongst/IN the/DT most/RBS stylish/JJ cars/NNS but/CC
its/PRP$ a/DT decent/JJ and/CC good/JJ looking/VBG family/NN car/NN Exterior/NNP Compact/JJ compare/VB to/TO
other/JJ in/IN same/JJ segment/NN however/RB gives/VBZ sporty/JJ look/NN Non/JJ fluidic/JJ design/NN but/CC
gives/VBZ enough/RB space/NN for/IN a/DT small/JJ family/NN Exterior/NNP Compact/JJ compare/VB to/TO
other/JJ in/IN same/JJ segment/NN however/RB gives/VBZ sporty/JJ look/NN Non/JJ fluidic/JJ design/NN but/CC
gives/VBZ enough/RB space/NN for/IN a/DT small/JJ familyExterior/JJ Compact/JJ compare/VB to/TO other/JJ
in/IN same/JJ segment/NN however/RB gives/VBZ sporty/JJ look/NN Non/JJ fluidic/JJ design/NN but/CC gives/VBZ
enough/RB space/NN for/IN a/DT small/JJ familyExterior/JJ Compact/JJ compare/VB to/TO other/JJ in/IN same/JJ
```

Fig. 13.3.1

13.4 POS Tagging


File Edit View History Bookmarks Tools Help
Home Page
localhost:21698/Review.aspx
Search
SENTIMENT ANALYSIS FOR CAR REVIEW
Welcome : Royal Dias
Home
About
Review
Profile
Result
Logout
Submit
22:32
19-04-2015

[redacted]

Company name: Hyundai

Model Name: Xcent

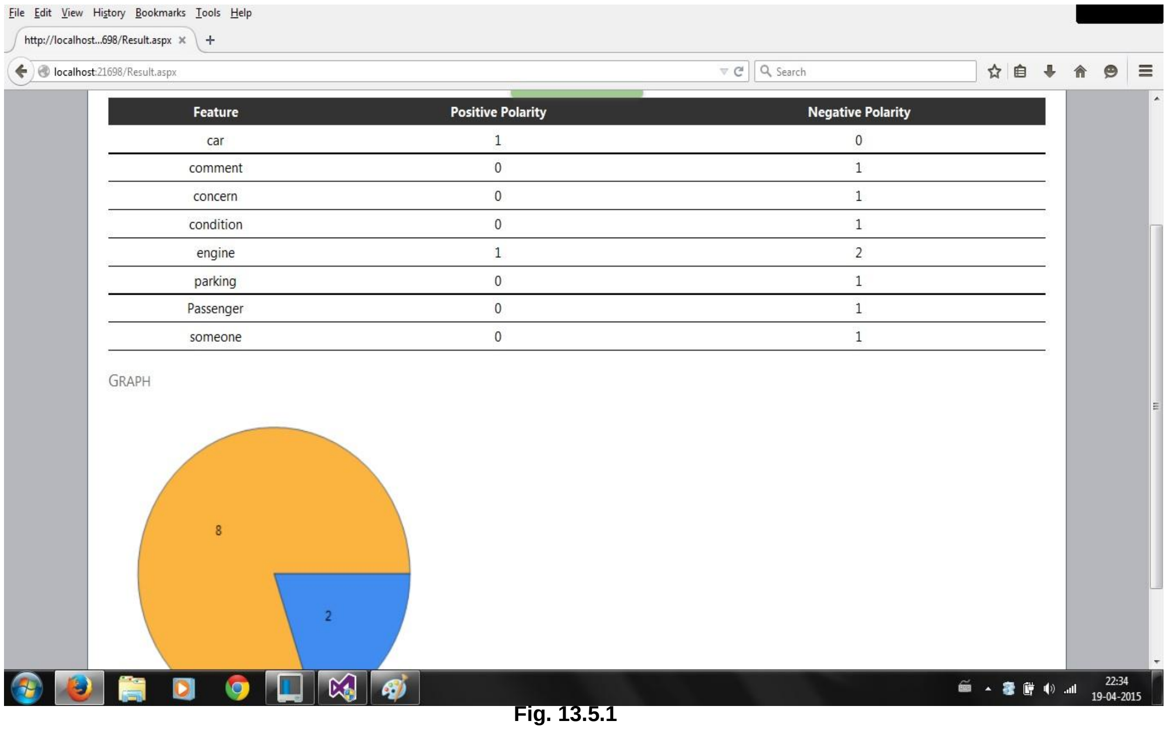
Select Car: Hyundai Xcent

Exterior The Car is a decent looking car considering the price it is offered for. Ir doesnt stand amongst the most stylish cars but its a decent and good looking family car . Exterior Compact compare to other in same segment, however gives sporty look. Non fluidic design but gives enough space for a small family .Exterior Compact compare to other in same segment, however gives sporty look. Non fluidic design but gives enough space for a small familyExterior Compact compare to other in same segment, however gives sporty look. Non fluidic design but gives enough space for a small familyExterior Compact compare to other in same segment, however gives sporty look. Non fluidic design but gives enough space for a small familyExterior Compact compare to other in same segment, however gives sporty look. Non fluidic design but gives enough space for a small family

Excellent Car from Hyundai, Lots of Features, Half open wallet :). Stylish and comfortable. Rear seats are enough to accommodate 3 medium size adults. Into 2nd Month of Ownership and driven about 1000 Kms without a fuss !!! , Every one in Office was pretty much impressed when they saw it, And what"s amazing was - the Fuel economy ! i filled the tank full and when it came at half mark, I took a note, it was 490 Kms. !! that means - 460 Kms in about 22 Ltrs of Fuel = 20.909 KMPL !! I never believed it, but when I refilled the tank at almost "0", doubt were cleared, Amazing economy of 19.58 KMPL ! ,But then again, that depends upon how one is driving the Car. Space is more than enough for family of 5 for a outstation Trip with Luggage :). Boot of 407 Liters, suits up the size.I bought Passion Red color, which looks Cool and catchy !Enough Leg room in front and back both. And Driver seat height adjustment is just too good, now those Long drives won"t feel pain in Back ! , can be easily adjusted while seated with a lever. Double charging points are great ! now my and my wife doesn"t need to argue on whose Mobile to charge first ! Cabin noise ? What"s that, really forgot what was Cabin noise. when inside the Car, hardly feel any noise of roads/traffic/or horns !!! . and the Engine is supper smooth, without any Vibrations inside the Cabin. In starting i had given "self" more than once, as it didn"t even felt that engine is running ! Absolutely cool ! Good power Engine, Great pick

Fig. 13.4.1

13.5 Summarized Result


File Edit View History Bookmarks Tools Help
[redacted]
http://localhost...698/Result.aspx
localhost:21698/Result.aspx
Search
GRAPH
8
2
22:34
19-04-2015

Feature	Positive Polarity	Negative Polarity
car	1	0
comment	0	1
concern	0	1
condition	0	1
engine	1	2
parking	0	1
Passenger	0	1
someone	0	1

Fig. 13.5.1

14. FEASIBILITY STUDY

The very first phase in any system developing life cycle is preliminary investigation. The feasibility study is a major part of this phase. A measure of how beneficial or practical the development of any information system would be to the organization is the feasibility study.

The feasibility of the development software can be studied in terms of the following aspects:

1. Operational Feasibility.
2. Technical Feasibility.
3. Economical feasibility.
4. Motivational Feasibility.
5. Legal Feasibility.

14.1 Operational Feasibility

The site will reduce the time consumed to maintain manual records and is not tiresome and cumbersome to maintain the records. Hence operational feasibility is assured.

- **Technical Feasibility**
- At least 166 MHz Pentium Processor or Intel compatible processor.
- At least 512 MB RAM.
- 14.4 kbps or higher modem.
- A mouse or other pointing device.
- At least 50 GB free hard disk space.
- Microsoft Internet Explorer 4.0 or higher.

14.2 Economical Feasibility

Once the hardware and software requirements get fulfilled, there is no need for the user of our system to spend for any additional overhead.

For the user, the web site will be economically feasible in the following aspects:

- The web site will reduce a lot of paper work. Hence the cost will be reduced.
- Our web site will reduce the time that is wasted in manual processes.
- The storage and handling problems of the registers will be solved.

14.3 Legal Feasibility

The licensed copy of the required software is quite cheap and easy to get. So from legal point of view the proposed system is legally feasible.

15. Conclusion

Classifying and summarizing opinions of bloggers has several interesting and commercially significant applications. However, this task is much more difficult than classifying regular text and requires intensive Preprocessing. The success of the opinion mining task is mainly dependent on the efficiency and sophistication of the Preprocessing and feature extraction steps. We empirically proved that the proposed approach for product feature set extraction, that is, using frequent multi words with decomposition strategy outperforms other contemporary approaches like the Apriori-based approach and the seed-set expansion approach.

Empirical results indicate that the multistep feature based semi supervised opinion mining approach used in this project can successfully identify opinionated sentences from unstructured user reviews and classify their orientation with acceptable accuracy. This enables reliable review opinion summarization which has several commercially important applications.

16. Future Work

In the future, we want to perform opinion mining on larger and more varied blog data sets. We would also like to extend our work to fuzzy opinion classification with support for fuzzy user querying. We intend to do this by learning the strength of various adjective descriptors along with corresponding linguistic hedges and include them in the feature-orientation table generated during the mining process. The classification technique proposed in the paper can be naturally extended to support fuzzy classification.

17. References:

[1] L. Zhao and C. Li, “Ontology based opinion mining for movie reviews,” in Proceedings of the 3rd International Conference on Knowledge Science, Engineering and Management, pp. 204–214, 2009.

[2] A. Balahur, Z. Kozareva, and A. Montoyo, “Determining the polarity and source of opinions expressed in political debates,” in Proceedings of the 10th International Conference on Intelligent Text Processing and Computational Linguistics, vol. 5449 of Lecture Notes in Computer Science, pp. 468–480, Springer, 2009.

[3] Y. H. Gu and S. J. Yoo, “Mining popular menu items of a restaurant from web reviews,” in Proceedings of the International Conference on Web Information Systems and Mining (WISM’11), vol. 6988 of Lecture Notes in Computer Science, pp. 242–250, Springer, 2011.

[4] M.Hu and B. Liu, “Mining and summarizing customer reviews,” in Proceedings of the 10th ACMSIGKDD International Conference on Knowledge Discovery and Data Mining (KDD ‘04), pp. 168–177, August 2004.

[5] M. A. Jahiruddin, M. N. Doja, and T. Ahmad, “Feature and opinion mining for customer review summarization,” in Proceedings of the 3rd International Conference on Pattern Recognition and Machine Intelligence (PReMI ‘09), vol. 5909 of Lecture Notes in Computer Science, pp. 219–224, 2009.

[6] S. Shi and Y. Wang, “A product features mining method based on association rules and the degree of property co-occurrence,” in Proceedings of the International Conference on Computer Science and Network Technology (ICCSNT ’11), pp. 1190–1194, December 2011.

[7] S. Huang, X. Liu, X. Peng, and Z. Niu, “Fine-grained product features extraction and categorization in reviews opinion mining,” in Proceedings of the 12th IEEE International Conference on Data Mining Workshops (ICDMW’12), pp. 680–686, 2012.

[8] C.-P.Wei,Y.-M.Chen,C.-S.Yang, and C. C.Yang, “Understanding what concerns consumers: a semantic approach to product feature extraction from consumer reviews,” Information Systems and e-Business Management, vol. 8, no. 2, pp. 149–167, 2010.