TECHNICAL REPORT

on

REVIEW RANKING SYSTEM BASED ON UTILITY

by

*Kunal Bharadwaj (D19016)*

*Veera Vignesh (D19036)*

*Vighnesh Tamse (D19037)*

*Durjay Das (D19039)*

Under the guidance of

*Prof. Gourab Nath*

*Assistant Professor, Data Science at Praxis Business School, Bangalore*



Department of Data Science

**Praxis Business School**

**ACKNOWLEDGEMENT**

We would like to express our deep and sincere gratitude to our guide and mentor **Prof. Gourab Nath** forgiving us the opportunity to do the project and providing invaluable guidance throughout the lifecycle of this project. His sincerity, hard work and motivation has inspired us in completing this project successfully. It was a great privilege and honour to be able to work under his guidance.

We would also like to extend our sincere gratitude to **Utkarsh Vardhan** sirfor taking time out of his busy schedule and guiding us throughout this project with his valuable advice and knowledge.

Last but not the least, we extend our heartfelt thanks to all the teaching as well as the non-teaching staff at **Praxis Business School, Bangalore** for providing us with whatever was necessary for this project and to our friends, colleagues and families for always helping and supporting us whenever in need.

Kunal Bharadwaj (D19016)

Veera Vignesh (D19036)

Vighnesh Tamse (D19037)

Durjay Das (D19039)

TABLE OF CONTENTS

Table of Contents

[Introduction 4](#_Toc38645343)

[1. Experiments 5](#_Toc38645344)

[1. Problem - To create features for our model from the extracted reviews using UI Path. 5](#_Toc38645345)

[2. Problem – Filtering stop words, punctuations and emoticons. Feature engineering using NER tagging, POS tagging, TF-IDF etc. Applying SVR model on the thus created data. 5](#_Toc38645346)

[3. Problem – Fitting Random Forest Regressor and XGBoost Regressor models on the data. 6](#_Toc38645347)

[4. Problem – Remove Upvote and Downvote columns and observe the MAPE score. 7](#_Toc38645348)

[5. Problem – Getting the review sentiment. 7](#_Toc38645349)

[6. Problem – Create 1st version of GUI for Review Ranking System. 7](#_Toc38645350)

[7. Problem – Add warning message if user clicks the Submit button without selecting any file. 8](#_Toc38645351)

[8. Problem – Change warning message to error message and some UI changes. GUI version 2. 9](#_Toc38645352)

[9. Problem – Web scraping the reviews data using Selenium Webdriver. 9](#_Toc38645353)

[10. Problem – To create a GUI to accept the URL of the reviews page of the product and rank them. GUI version 3. 10](#_Toc38645354)

[11. Problem – To select the best model from Random Forest or XG Boost or SVR to rank the reviews of products in different categories such as Mobile Phones, Laptops and Cameras. 10](#_Toc38645355)

CHAPTER 1

# Introduction

The demand for online purchase is increasing day by day because people want to shop without much hassle. People can purchase a product anytime from anywhere without facing the crowd, also having the advantages to explore a variety of different products in the same place at a competitive price. Customers also share their valuable feedback about their purchased products in the form of reviews and ratings. These reviews are also evaluated by other customers by upvoting or downvoting them.

These reviews are helpful in many ways, for example, when someone wishes to buy a new product, they can estimate whether the product is apt for them by reading the reviews. This will make it easy for them to understand the pros and cons of the product and accordingly they can decide whether to buy the product or not. Also, these reviews are helpful for the manufacturers to understand the customer’s sentiment. This may help the manufacturers to modify the product as per the customer’s needs to sustain in the aggressive market

There may be thousands of different opinions of customers. Some of the reviews are intended to understand the pros and cons of a product while others are not much beneficial. It is cumbersome to go through all the reviews to understand the product. However, we can read a small number of the most relevant and meaningful reviews to have a quicker understanding of the product.

The application can be utilized by customers as well as manufacturers to assess only the most useful reviews to take relevant business decisions.

Chapter 2

# Data Collection

**Aim:** To Extract the reviews of the product from Flipkart for further processing and modelling.

**Tools Used:** We are using **UiPath Studio** for data scrapping from Flipkart in the required format. We have used Selenium to automate the process of review scraping from Flipkart.

**About UIPath Studio:**

**UiPath Studio** - an advanced tool that enables you to design automation processes in a visual manner, through diagrams.

**UiPath Robot** - executes the processes built in Studio, as a human would. Robots can work **unattended** (run without human supervision in any environment, be it virtual or not) or as **assistants** (a human initiates the process).

**Workflow Designer:** It is essential for Robotic process automation (RPA) products to give their users an amazing experience. One **way** to do this is to make visual **workflow** automation straightforward and intuitive. **UiPath** is providing automated **workflow design** which can be used without programming knowledge.

**Activities:** An Activity is an action we can add in your workflow - like clicking the OK button or typing a text in an input box. Activity box is found on the left panel. An activity is added into the workflow by dragging it to the designer workspace.

**Data Domain:** We have extracted the data for the mobile reviews in initial stages and extended the same for any product present on Flipkart. Below are few examples of the links

1. [apple-iphone-7-black-32-gb](https://www.flipkart.com/apple-iphone-7-black-32-gb/product-reviews/itmen6daftcqwzeg?pid=MOBEMK62PN2HU7EE&lid=LSTMOBEMK62PN2HU7EEINTGNU&marketplace=FLIPKART)
2. [redmi-8-onyx-black-64-gb](https://www.flipkart.com/redmi-8-onyx-black-64-gb/product-reviews/itm74a5b975b3bdf?pid=MOBFKPYDZJQHGJXA&lid=LSTMOBFKPYDZJQHGJXA5X8Q5G&marketplace=FLIPKART)
3. [samsung-galaxy-s10-lite-prism-blue-512-gb](https://www.flipkart.com/samsung-galaxy-s10-lite-prism-blue-512-gb/product-reviews/itmbd1470b934615?pid=MOBFPDVHAJMVXUYZ&lid=LSTMOBFPDVHAJMVXUYZNP3GGC&marketplace=FLIPKART)

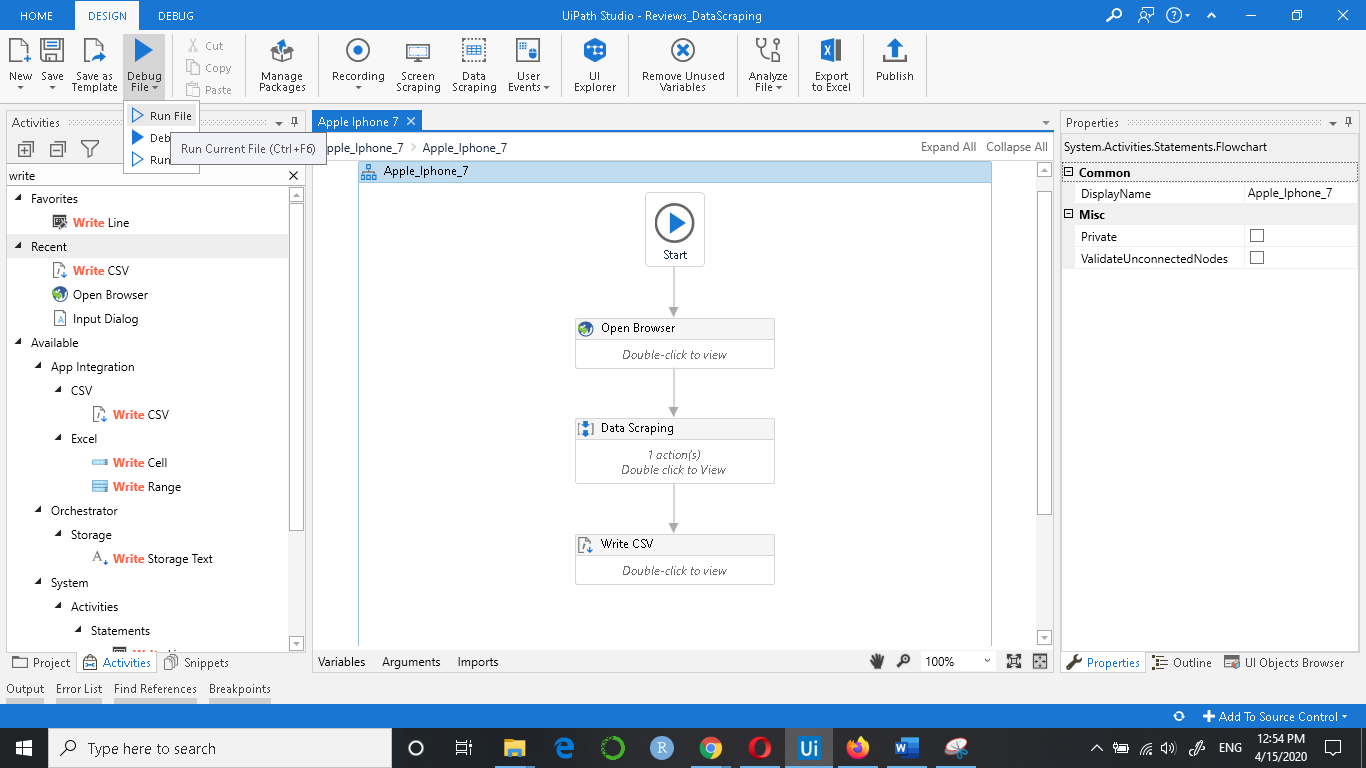
**Experiment:** For our experiments, we started scraping the mobile phone reviews from Flipkart and storing the same in csv files for further analysis.

**UiPath Studio steps:**

We are providing below the document containing the steps to scape the reviews from Flipkart with the help of UiPath Studio



**UIPath Studio workflow:**



**Result:**

## Experiments

Date: 06-03-2020

### Problem - To create features for our model from the extracted reviews using UI Path.

**Github** - <https://github.com/veeravignesh1/Capstone-Reviews-Ranker/blob/master/Experiments/Iphone7.ipynb>

**Experiment No. 1** – Used NLTK library to create features such as Length of review, Number of sentences, Number of emoticons used, Number of exclamation marks, Number of Question marks etc.

**Findings** – Most of the reviews were of length between 1-70 words. The average length of the reviews was about 52 words per review. The average number of sentences was 4. We found that there were very a small number of reviews that used exclamation or question marks. Also, not many reviewers have used Proper Case in their reviews. The maximum number of words per review that we got after web scraping was 500. This was because there were reviews that had READ MORE section in them which UI Path was not able to handle.

**Experiment No. 2** – Manually copying the reviews that had READ MORE section in them and performing the steps again as done in Experiment No.1.

**Findings** – After manually copying the reviews that had READ MORE section in them, the average length of the reviews came to be 58 words per review. The maximum number of words per review jumped to 633. All the features were extracted in a csv file named ‘withfeatures.csv’.

Date: 13-03-2020

### Problem – Filtering stop words, punctuations and emoticons. Feature engineering using NER tagging, POS tagging, TF-IDF etc. Applying SVR model on the thus created data.

**Github** - <https://github.com/veeravignesh1/Capstone-Reviews-Ranker/blob/master/Experiments/Modelling-Iphone.ipynb>

**Experiment No. 1** – Removing stop words, punctuations, emoticons and spell check. Also, we removed those reviews that had less than 2-3 words in it.

**Findings** – Using the ‘withfeatures.csv’ file that we created in Experiment No. 2 in Problem 1 , we removed the punctuations (question marks, exclamation marks, full stop etc), stop words and emoticons. For spell correction we experimented using autocorrect package of python and textblob package. Since textblob gave us better result we went ahead with textblob.

**Experiment No. 2** – Removed the stop words and applied POS and NER tagging. Used TF-IDF Vectorizer to create a TF-IDF matrix. Created the target variable named ‘h’ which is used to rank the reviews and is calculated as below:

h = Upvote/(Upvote + Downvote)

**Findings** – The TF-IDF matrix thus created consisted of all the unique words found in all the reviews as columns plus the columns we created during feature engineering stage. For iphone7, for a total of 264 reviews, we had a total of 1610 unique words. This dataframe was then exported to a csv file named ‘Unigram\_Feature.csv’.

**Experiment No. 3** – We split the data in ‘Unigram\_Features.csv’ thus created in Experiment No. 2 into Train-Validation-Test. We then used the SVR (Support Vector Regression) model to fit our data.

**Findings** – After fitting the SVR model on training data we predicted the test data and calculated the R^2 score which came out to be just 13%.

Date: 15-03-2020

### Problem – Fitting Random Forest Regressor and XGBoost Regressor models on the data.

**Github** -

1. <https://github.com/veeravignesh1/Capstone-Reviews-Ranker/blob/master/Experiments/Modelling_Iphone_RandomForest.ipynb>
2. <https://github.com/veeravignesh1/Capstone-Reviews-Ranker/blob/master/Experiments/Modelling_Iphone_XGBoost.ipynb>

**Experiment No. 1** – We fitted the Random Forest Regressor model on the Unigram\_Features data and calculated the MAPE score. We also found the most important features by using the feature\_importances\_ function provided by the Random Forest Regressor model.

**Findings** – We found that Upvote and Downvote are the most important features found by our model to predict the outcome. Also, we got a MAPE score of 17.28% means the accuracy of our model was around 82.7%.

**Experiment No. 2** – We fitted the XGBoost Regressor model on the Unigram\_Features data and calculated the MAPE score.

**Findings** – We got a MAPE score of 16.5% means the accuracy of our model was around 83.5%.

Date: 16-03-2020

### Problem – Remove Upvote and Downvote columns and observe the MAPE score.

**Github** ­–

<https://github.com/veeravignesh1/Capstone-Reviews-Ranker/blob/master/Experiments/Modelling_Exclude_Upvote_Downvote.ipynb>

**Experiment No. 1** – We removed the Upvote and Downvote columns which our model ranked as the most important features for predicting the outcome and then observed the MAPE score using both Random Forest Regressor and XGBoost Regressor.  
**Findings** – When we drop Upvote and Downvote columns, fit our model on train data and test on test data, we found that MAPE score increased to 29% for Random Forest Regressor and 30% for XGBoost Regessor.

Date: 22-03-2020

### Problem – Getting the review sentiment.

**Github** –

<https://github.com/veeravignesh1/Capstone-Reviews-Ranker/blob/master/Experiments/Review_Title_Sentiments.ipynb>

**Experiment No. 1** – We used the Vader Sentiment Analysis for getting the review sentiments such as positive, negative or neutral.

**Findings** – We found that a few of the users who have rated the product very low (less than 2 ratings) have positive sentiments. On checking the reviews of these users manually we found that the reviews were actually positive but the ratings given were less. This can be misleading.

Date: 27-03-2020

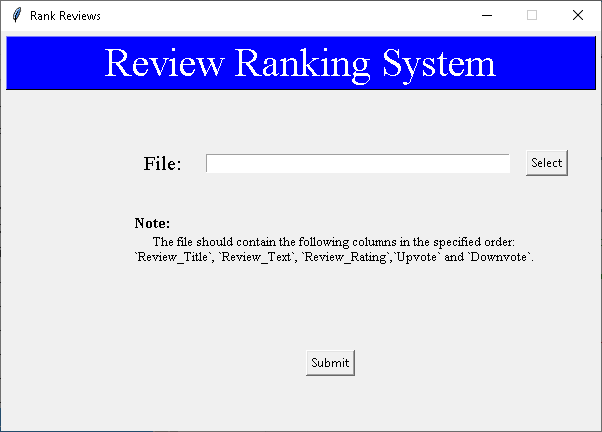
### Problem – Create 1st version of GUI for Review Ranking System.

**Github** –

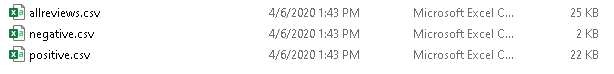
1. [https://github.com/veeravignesh1/Capstone/blob/ master/Experiments/Review\_GUI\_v1.ipynb](https://github.com/veeravignesh1/Capstone/blob/master/Review_GUI_v1.ipynb)
2. [https://github.com/veeravignesh1/Capstone/blob/ master/Experiments/Review\_GUI\_v1.py](https://github.com/veeravignesh1/Capstone/blob/master/Review_GUI_v1.py)

**Experiment No. 1 –**

* Used ‘Tkinter’ library to create GUI for our project. It is designed such that it takes the extracted reviews file from the user as input with columns in order of: Review\_Title, Review\_Text, Review\_Rating, Upvote and Downvote. Converted the ipython notebook (.ipynb) to python script (.py)



* This system outputs 3 csv files namely: allreviews, negative and positive each containing reviews which are ranked.



Date: 04-04-2020

### Problem – Add warning message if user clicks the Submit button without selecting any file.

**Experiment No. 1** – Added a warning message that pops up when a user tries to click on Submit button without selecting any file.

**Findings** – The system throwed the warning message even if the file was selected by the user.

**Experiment No. 2** – Tweaked the program to rectify the defect encountered in Experiment No. 1 to give warning message only when file is not selected.

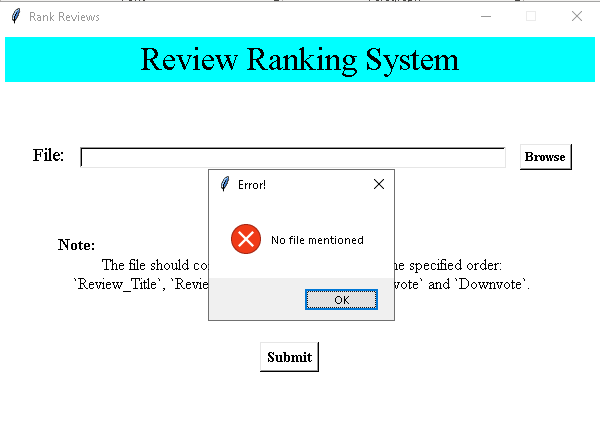
Date: 05-04-2020

### Problem – Change warning message to error message and some UI changes. GUI version 2.

**Github** - [https://github.com/veeravignesh1/Capstone/blob/ master/Experiments/Review\_GUI\_v2.ipynb](https://github.com/veeravignesh1/Capstone/blob/master/Review_GUI_v2.ipynb)

**Experiment No. 1** – Changed the warning message to an error message when user does not select any file. Changed the look and feel of the UI.

**Findings** – The system showed the error message accurately when user intends to click the submit button without selecting any file.



Date 15-04-2020

### Problem – Web scraping the reviews data using Selenium Webdriver.

**Github** -

**Experiment No. 1** – In order to rank the reviews using just the URL of the reviews page of the product for the new version of the GUI, we scraped the reviews data by using Selenium Webdriver.

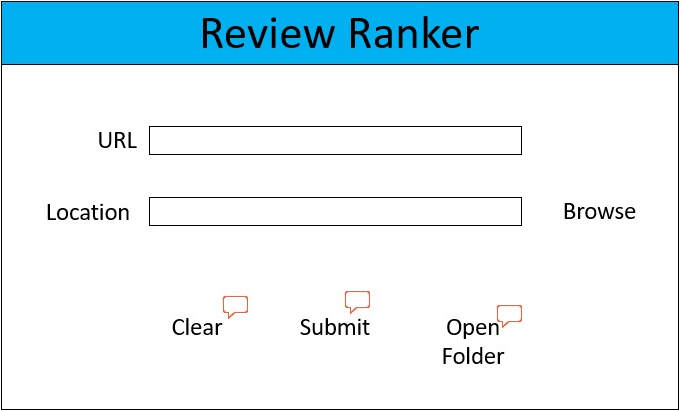
**Findings** – We created a function that would get the reviews just by passing the URL of the reviews page of the product.

Date 16-04-2020

### Problem – To create a GUI to accept the URL of the reviews page of the product and rank them. GUI version 3.

**Github** –

**Experiment No. 1** –Created a GUI that will accept the URL of the reviews page of the product and a folder location where you want to place the files in. On click of Submit button, the ranked reviews (all, positive and negative) will be stored in the specified folder.



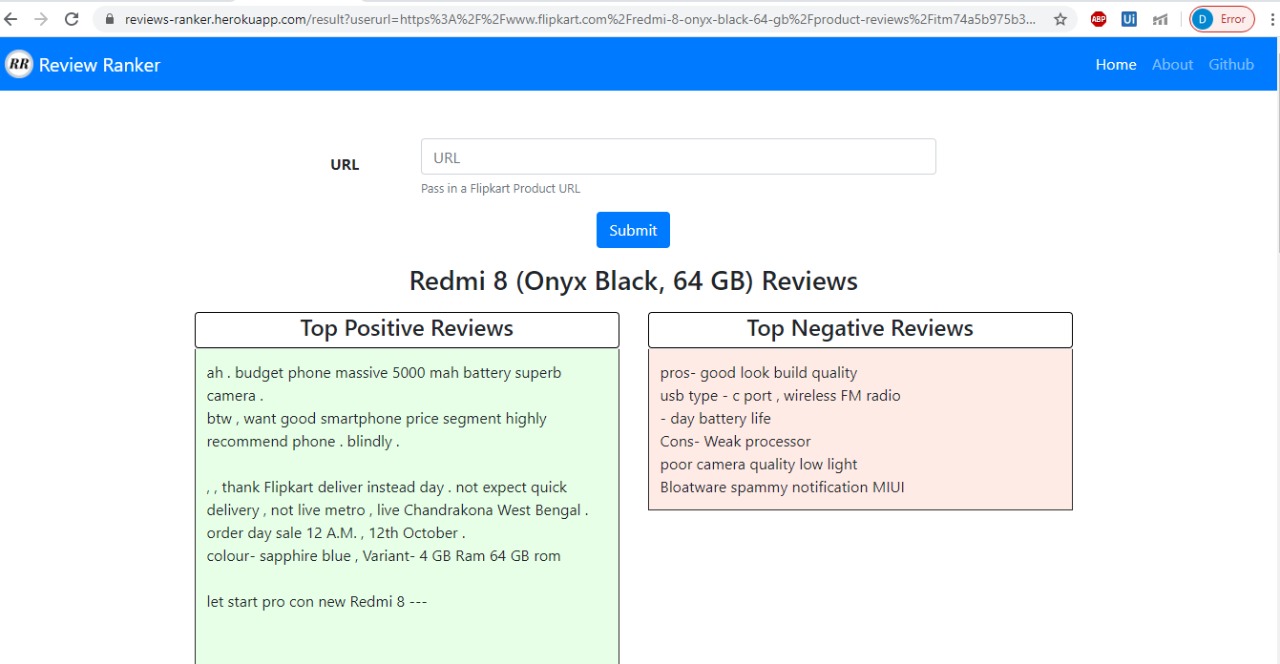
Date 22-04-2020

### Problem – To create a deployed version to accept the URL of the reviews page of the product and rank them. GUI version 4.

**Github** –

**Experiment No. 1** –Created a GUI and deployed on Heroku app that will accept the URL of the reviews page of the product on click of Submit button, the ranked reviews (positive and negative) will be displayed.

<https://reviews-ranker.herokuapp.com/>



Date 18-04-2020

### Problem – To select the best model from Random Forest or XG Boost or SVR to rank the reviews of products in different categories such as Mobile Phones, Laptops and Cameras.

Acronyms for the features experimented with

|  |  |
| --- | --- |
| **Column\_Name** | **Acronym** |
| Len\_after | LA |
| Len\_before | LB |
| Num\_Sentence | NS |
| No\_QMark | NQ |
| No\_ExMarch | NEx |
| No\_Upper | NU |
| No\_proper | NPr |
| Perc\_Noun | PN |
| Perc\_Verb | PV |
| Perc\_Adverb | PAdv |
| Perc\_Adj | PAdj |
| Target | h |

Acronyms for the available features

|  |  |
| --- | --- |
| **Column\_Name** | **Acronym** |
| Review\_Title | Title |
| Review\_Text | RT |
| Review\_Rating | RR |
| Num\_Photos | NPh |
| Upvote | Up |
| Downvote | Down |

**Github** –

**Experiment No. 1** –Scraped the reviews for a mobile phone, passed it through the get\_features() function containing TF-IDF vectorizer, created different combinations of feature sets, fitted the Random Forest Regressor Model on all the three products and calculated the MAPE score.

**Experiment No. 2** –Scraped the reviews for a mobile phone, passed it through the get\_features() function containing TF-IDF vectorizer, created different combinations of feature sets, fitted the XG Boost Regressor Model on all the three products and calculated the MAPE score.

**Experiment No. 3** –Scraped the reviews for a mobile phone, passed it through the get\_features() function containing TF-IDF vectorizer, created different combinations of feature sets, fitted the Support Vector Machine Regressor Model on all the three products and calculated the MAPE score.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Sr. No.** | **Features\*** | **Random Forest** | | **XG Boost** | | **SVR** | |
|  |  | **Train\_MAPE** | **Test\_MAPE** | **Train\_MAPE** | **Test\_MAPE** | **Train\_MAPE** | **Test\_MAPE** |
| 1 | [RT,RR,LA,h] | 4.76 | 14.02 | 0.59 | 16.18 | 11.17 | 12.61 |
| 2 | [RT,RR,NS,h] | 4.8 | 14.43 | 0.81 | 16.27 | 10.52 | 12.38 |
| 3 | [RT,RR,LB,h] | 4.61 | 15.09 | 0.42 | 15.09 | 11.23 | 12.45 |
| 4 | [RT,RR,NPh,h] | 4.89 | 14.08 | 0.64 | 16.24 | 9.79 | 12.79 |
| 5 | [RT,RR,NPh,LB,LA,h] | 4.49 | 13.19 | 0.29 | 15.85 | 10.88 | 12.4 |
| 6 | [RT,PN,PV,h] | 5.01 | 14.9 | 0.91 | 17.89 | 11.81 | 14.01 |
| 7 | [RT,PN,PV,PAdv,h] | 5.03 | 14.8 | 0.9 | 17.34 | 11.75 | 14.17 |
| 8 | [RT,PN,PV,PAdv,PAdj,h] | 4.97 | 14.86 | 0.79 | 16.86 | 11.81 | 14.92 |
| 9 | [RT,NQ,h] | 5.22 | 15.4 | 1.08 | 17.67 | 9.23 | 14.12 |
| 10 | [RT,NQ,NEx,h] | 5.13 | 14.98 | 0.98 | 17.04 | 9.49 | 14.38 |
| 11 | [RT,NQ,NEx,NU,h] | 5.13 | 14.81 | 0.98 | 17.04 | 9.52 | 14.38 |
| 12 | [RT,NQ,NEx,NU,NPr,h] | 4.96 | 14.78 | 0.56 | 16.47 | 9.21 | 14.03 |
| 13 | [RT,NPh,NS,h] | 4.68 | 13.62 | 0.92 | 15.49 | 10.23 | 13.4 |
| 14 | [RT,NPh,NS,PN,h] | 4.65 | 13.37 | 0.78 | 15.61 | 11.3 | 13.71 |
| 15 | [RT,NPh,NS,PV,h] | 4.88 | 13.65 | 0.9 | 14.72 | 10.88 | 12.16 |
| 16 | [RT,NPh,NS,PN,PV,h] | 4.81 | 13.52 | 0.77 | 15.42 | 11.11 | 13.25 |
| 17 | [RT,NPh,NS,PAdj,h] | 4.5 | 13.27 | 0.53 | 15.94 | 11.94 | 13.97 |
| 18 | [RT,NPh,NS,PAdj,PN,h] | 4.54 | 12.95 | 0.53 | 15.52 | 11.64 | 14.95 |
| 19 | [RT,NPh,NS,PAdj,PN,PV,h] | 4.67 | 13.21 | 0.53 | 14.39 | 11.58 | 15.01 |
| 20 | [RT,RR,NPh,NS,PAdj,PN,h] | 4.58 | 12.85 | 0.47 | 15.38 | 11.64 | 15.01 |

**Model and feature experimentation with Mobile phone reviews**

*\* Review\_Text (RT) column is needed in every combination. 'h' is target variable*

**Model and feature experimentation with Laptop reviews**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Sr. No.** | **Features** | **Random Forest** | | **XG Boost** | | **SVR** | |
|  |  | **Train\_MAPE** | **Test\_MAPE** | **Train\_MAPE** | **Test\_MAPE** | **Train\_MAPE** | **Test\_MAPE** |
| 1 | [RT,RR,LA,h] | 5.37 | 14.69 | 0.08 | 17.22 | 15.06 | 16.59 |
| 2 | [RT,RR,NS,h] | 5.42 | 14.6 | 0.26 | 17.02 | 13.05 | 15.58 |
| 3 | [RT,RR,LB,h] | 5.32 | 14.9 | 0.07 | 17.18 | 15.01 | 16.94 |
| 4 | [RT,RR,NPh,h] | 4.95 | 14.88 | 0.07 | 15.56 | 10.59 | 15.47 |
| 5 | [RT,RR,NPh,LB,LA,h] | 4.93 | 15.04 | 0.07 | 16.48 | 15.16 | 16.8 |
| 6 | [RT,PN,PV,h] | 5.53 | 15.8 | 0.25 | 18.89 | 14.42 | 16.74 |
| 7 | [RT,PN,PV,PAdv,h] | 5.58 | 15.66 | 0.25 | 18.15 | 14.18 | 16.68 |
| 8 | [RT,PN,PV,PAdv,PAdj,h] | 5.66 | 15.6 | 0.24 | 17.96 | 14.05 | 16.03 |
| 9 | [RT,NQ,h] | 5.65 | 14.55 | 0.49 | 15.29 | 9.97 | 15.44 |
| 10 | [RT,NQ,NEx,h] | 5.69 | 14.92 | 0.49 | 15.19 | 9.75 | 15.54 |
| 11 | [RT,NQ,NEx,NU,h] | 5.71 | 14.64 | 0.49 | 15.19 | 9.75 | 15.54 |
| 12 | [RT,NQ,NEx,NU,NPr,h] | 5.71 | 14.79 | 0.49 | 15.19 | 9.77 | 15.48 |
| 13 | [RT,NPh,NS,h] | 5.16 | 14.72 | 0.08 | 15.4 | 13.62 | 16.19 |
| 14 | [RT,NPh,NS,PN,h] | 5.21 | 16.01 | 0.08 | 17.54 | 14.15 | 16.88 |
| 15 | [RT,NPh,NS,PV,h] | 5.16 | 14.81 | 0.08 | 15.16 | 13.84 | 16.07 |
| 16 | [RT,NPh,NS,PN,PV,h] | 5.26 | 15.78 | 0.08 | 17.87 | 14.05 | 16.93 |
| 17 | [RT,NPh,NS,PAdj,h] | 5.21 | 14.22 | 0.08 | 15.44 | 14.17 | 15.72 |
| 18 | [RT,NPh,NS,PAdj,PN,h] | 5.31 | 15.9 | 0.07 | 16.74 | 13.97 | 16.51 |
| 19 | [RT,NPh,NS,PAdj,PN,PV,h] | 5.38 | 15.48 | 0.07 | 17.4 | 13.93 | 16.35 |
| 20 | [RT,RR,NPh,NS,PAdj,PN,h] | 5.06 | 15.78 | 0.07 | 16.21 | 13.5 | 15.86 |

*\* Review\_Text (RT) column is needed in every combination. 'h' is target variable*

**Model and feature experimentation with Camera reviews**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Sr. No.** | **Features** | **Random Forest** | | **XG Boost** | | **SVR** | |
|  |  | **Train\_MAPE** | **Test\_MAPE** | **Train\_MAPE** | **Test\_MAPE** | **Train\_MAPE** | **Test\_MAPE** |
| 1 | [RT,RR,LA,h] | 5.03 | 12.43 | 1.64 | 13.44 | 10.2 | 11.88 |
| 2 | [RT,RR,NS,h] | 5.43 | 12.68 | 2.12 | 13.23 | 9.55 | 11.69 |
| 3 | [RT,RR,LB,h] | 4.95 | 12.6 | 1.48 | 13.3 | 10.24 | 12.42 |
| 4 | [RT,RR,NPh,h] | 4.75 | 12.51 | 0.89 | 12.31 | 9.42 | 12.59 |
| 5 | [RT,RR,NPh,LB,LA,h] | 4.48 | 12.3 | 0.68 | 12.27 | 10.23 | 12.27 |
| 6 | [RT,PN,PV,h] | 5.33 | 12.64 | 2.06 | 12.6 | 10.67 | 12.23 |
| 7 | [RT,PN,PV,PAdv,h] | 5.31 | 12.7 | 2.06 | 12.37 | 10.53 | 12.08 |
| 8 | [RT,PN,PV,PAdv,PAdj,h] | 4.93 | 13.21 | 1.58 | 12.96 | 10.63 | 11.84 |
| 9 | [RT,NQ,h] | 5.65 | 12.5 | 2.55 | 12.77 | 8.6 | 11.65 |
| 10 | [RT,NQ,NEx,h] | 5.66 | 12.33 | 2.55 | 12.77 | 8.6 | 11.75 |
| 11 | [RT,NQ,NEx,NU,h] | 5.62 | 12.38 | 2.55 | 12.77 | 8.6 | 11.77 |
| 12 | [RT,NQ,NEx,NU,NPr,h] | 5.54 | 12.51 | 2.36 | 12.85 | 8.56 | 11.83 |
| 13 | [RT,NPh,NS,h] | 4.71 | 12.6 | 0.94 | 12.8 | 8.73 | 12.39 |
| 14 | [RT,NPh,NS,PN,h] | 4.58 | 12.64 | 0.74 | 13.17 | 10.87 | 11.73 |
| 15 | [RT,NPh,NS,PV,h] | 4.57 | 12.7 | 0.98 | 12.44 | 9.72 | 12.81 |
| 16 | [RT,NPh,NS,PN,PV,h] | 4.48 | 12.86 | 0.74 | 12.16 | 10.57 | 12.31 |
| 17 | [RT,NPh,NS,PAdj,h] | 4.59 | 13.05 | 0.79 | 13.43 | 10.78 | 12.07 |
| 18 | [RT,NPh,NS,PAdj,PN,h] | 4.52 | 12.94 | 0.72 | 12.84 | 10.9 | 11.78 |
| 19 | [RT,NPh,NS,PAdj,PN,PV,h] | 4.4 | 13.1 | 0.71 | 12.81 | 10.55 | 12.21 |
| 20 | [RT,RR,NPh,NS,PAdj,PN,h] | 4.57 | 12.88 | 0.72 | 12.99 | 10.79 | 11.73 |

*\* Review\_Text (RT) column is needed in every combination. 'h' is target variable*

**Findings** – Out of all the three experiments, we found that SVR model gave least MAPE scores, so we went ahead with SVR as our final model.