**Gürkan Soykan – Uğur Gül – Hasret Taşkın – Halil Çetinkaya**

**SENTIMENT ANALYSIS - TURKISH TWEETS**

**Project Report**

**Problem Definition**

Given a tweet about banks written in Turkish, the team was asked to classify its polarity (how positive a review is it) as a score between -1 (very negative) to +1 (very positive); so this is a regression problem.

E.g. “alt tarafi ATM karti istiyoruz. 5 aydir ne gelen ne giden. Para cekmek icin subeye gitmek

zorundamiyim???” -> -0.8 (quite negative comment)

“sorun çözüldü, tesekkürler!”-> +1 (fully positive comment)

**Our Approach**

After proper investigation of the train and test sets, the team devised a way to tackle with this problem. Since we would be dealing with tweets our approach revolved around that. Initial step to be taken was “text cleaning”. That was an essential step because the data was highly unstructured and noisy meaning the tweets were raw data and far away from the proper Turkish language. After this step, we needed to extract features which then combined with the given features. The extracted features are as follows:

* + Number of question marks
  + Number of exclamation marks
  + Number of negative smileys
  + Number of positive smileys
  + Number of very positive tokens
  + Number of positive tokens
  + Number of neutral tokens
  + Number of minor negative tokens
  + Number of negative tokens
  + Number of very negative tokens
  + Number of extremely negative tokens
  + General sentiment weight

Apart from those extracted features the team has also find features based on bag of words method and tf-idf and tried regression with those features as well. Our aim was combine all of the features and then find the best possible predictions and minimize errors.

After the team found the proper features, multitude of regression algorithms were executed for the task. Some of them are as follow:

* Decision Tree Regression
* k-nearest neighbors
* Support Vector Machine Regression
* Stochastic Gradient Descent Regression
* Random Forests Regression
* Multi-layer Perceptron Regression
* Ridge Regression
* Bayesian Ridge Regression
* Ensemble Regression with Ada Boost Regressor

Then based on the results of the regressions some of the parameters are changed iteratively in order to find maximum accuracy.

**Text Cleaning and Preprocessing Step**

Even before the team started to work on data something had been noticed. That was some letters in tweets were not appropriate in terms of Turkish language. Meaning that the tweets were containing less Turkish letters maybe because of the twitter medium or maybe because of the users keyboards. However, that needed to be handled because otherwise those text can give rise to problems when it comes to NLP processes. An example is presented below.

“@garantiyesor Devletmemuruym maasimi garantdn aliyorm. 300-500 tl arasibir paraya ihtiyç duydm. hiçbirücretvefaiz ödemyecegm kampnyanz varmi 0”

***Deasciified version:*** “@garantiyesor Devletmemuruym maaşımı garantdn alıyorm. 300-500 tl arasibir paraya ihtiyç duydm. hiçbirücretvefaiz ödemyecegm kampnyanz varmı” 0

For the deasciification process the team used Online Turkish deasciifier <http://www.deasciifier.com/>. Even though there were other options such as ITU Turkish Natural Language Processing Pipelines Deasciifier that needed an user account which the team did not have and send email to the authorized people yet, no reply was sent back.

After going through deasciification process the team had knew that would not be enough to get the correct intended meaning in sentences. For that the team come up with the idea of eliminating stop words. Since, they would not carry any information related to given problem at hand. Because stop words are the words occurring frequently in a language irrespective of the context. For instance “gibi”, “hepsi”, “biz” are some of the Turkish stop words. In order to get rid of stop words the team found Turkish stop words list from the internet and created a dictionary by using hash table. Then every stop word in a tweet filtered through this dictionary.

For the tokenization process the used the “Zemberek” which is an open-source general purpose Natural Language Processing library and toolset designed for Turkic languages, especially for Turkish. By exploiting the features of this library the team found a way to tokenize words in tweets. Moreover, “Zemberek” was also used for spell checking. That was because if we could not get the words in their proper form, the team would not be able to realize stemming and lemmatization processes as well.

While getting tokens from each tweet, the team checked that if these tokens are all upper case, emoticons, exclamation marks, question marks, and spelled correctly or not if not correct then get the correct form. As a finding the team concluded that tweets do not contain all uppercase tokens. If otherwise were correct than that might have been used as a feature as well. After getting tokens, lemmatization was applied. After applying that inflectional endings was removed from tokens and tokens are reduced to their dictionary form. As a side effect of this process the tokens which have no meaning are eliminated. Because they would return “UNKNOWN” value. So this process had multiple positive outcome on this project. This step lead to transformation of tweets in a way that is shown in the below part.

* ***Original Forms of Some Tweets***
  + @buketsarionder Buket Hanim islemiz sonuçlandirilmis olup her türlü görüs ve sorulariniz için @GarantiyeSor hesabimizi takip edebilirsiniz. 0.3
  + @umreak @garanti @GarantiyeSor ben onlara hicbisey sormuyorum ve hesaplarimi kapattim en akillica is bu oldu!!!! -1
  + @GarantiyeSor Bana geri dönüs yapacaktin, hala bekliyorum. -0.3
  + @GarantiyeSor Sizi en son Allah'a havale edeyim diyordum.Ondan bile 5 TL havale ücreti istediniz...!!! -1
  + @GarantiyeSor gönderdim, bu sorunun hizlica çözümlenmesini bekliyorum. -0.2
  + hizil dönüs ve ilginiz için tesekkür ederim @GarantiyeSor 1
* ***Preprocessed Forms of Those Tweets***
  + buket hanım işlem sonuçlandırmak olmak türlü görüş soru hesap takip etmek . 0.3
  + sormak hesap kapatmak Akıllı iş oldu ExclamationMark ExclamationMark ExclamationMark ExclamationMark -1
  + geri dönüşmek yapmak , hâlâ beklemek . -0.3
  + Allah havale etmek demek . 5 Tl havale ücret istemek ... ExclamationMark ExclamationMark ExclamationMark -1
  + göndermek , Sor hızlı çözümlemek beklemek . -0.2
  + dönüşmek ilgi teşekkür etmek 1

**Feature Extraction**

In this step of the project, knowing that the given 21 features would not be enough for this problem, the team has come up with different ideas for extracting more features from the given train set. Initially, the basic features of number of exclamation marks, number of question marks, number of negative smileys, and number of positive smileys are used as features. As a side no the positive and negative smileys were as follows:

* Pozitives: ":-)", ":)", "=)", ":D"
* Negatives: ":-(", ":(", "=(", ";("

When devising this features the team was influenced by the work of “Sentiment Analysis for Turkish Twitter Feeds” by Önder Çoban, Barış Özyer, Gülşah Tümüklü Özyer.

After detecting those features the bag of words method with tf-idf was used. In that model every distinct word in the data corpus were abstracted as a dimension in the vector space. So that when a sentence is given it is represented as a vector which contains the frequencies of these words matching to the index of the vector. After this model is being applied the team detected that the train set has 1304 distinct word and the matrix for all tweets has dimension of ( 757, 1304 ). Then with the availability of these features the team has conducted Bayesian Ridge Regression. The regression score was 0.117477844719 on the test set, test set was used because they also had labels. Since positive score was obtained from the test set, the sign of possible use of this features for later was shown. For instance the most important of those 1304 features could be used and merged with the other relevant features.

Then there was another idea for features which was finding the tendencies in the training set for each word and then mapping them on basic categories such as very positive tokens, positive tokens, neutral tokens, minor negative tokens, negative tokens, very negative tokens, extremely negative tokens. The called this operation “Bucketing” or “Bucketizing”. According to this method each token is associated with its sentence sentiment score. Then sum of those scores are divided by the number of occurrences of that token. As a result of that each token have average sentiment score. If these score were in certain boundaries they transformed to the name of the bucket that they are in. For instance if a token has score in between 1 and 0.5 then it had “VeryPositive” remark and if it as average score of -0.8 then it is in the bucket of “VeryNegative”. Because the range of that bucket is from -0.75 to -0.9.

Another interesting thing that the team had observed was to general score of whole training set. When it is measured the average score of training set was calculated as -0.3011889035667103. That showed us the fact that training set was skewed towards negative feeling or sentiment. Same thing also observed in bag of words method when the predictions are checked with their original values. Predictions were usually more negative than the original value. In order to prevent that the team increased the bucket size of the positive features. So that their weights would be higher. After this steps the “Word Sentiment Dictionary” was created. Some examples from the dictionary are as follows:

|  |
| --- |
| akbank: Positive |
| akbaş: Neutral |
| akrep: ExtremelyNegative |
| emeklilik: Positive |
| emin: Positive |
| enayi: ExtremelyNegative |
| enteresan: Negative |
| esen: Positive |
| eski: Neutral |
| fahiş: ExtremelyNegative |
| hoş: VeryPositive |
| kesinlikle: ExtremelyNegative |
| mağdur: VeryNegative |
| mağduriyet: Negative |
| medya: Positive |
| merhaba: Neutral |
| mesut: VeryPositive |
| oldukça: VeryPositive |
| peki: VeryPositive |
| positivesmiley: Positive |
| rezalet: Negative |

Getting those values for each word we could then translate each sentence to this abstracted form by using the dictionary as an abstraction. However, this mapping is used to obtain this features such as o number of very positive tokens,number of positive tokens, number of neutral tokens, number of minor negative tokens, number of negative tokens, number of very negative tokens, number of extremely negative tokens and general sentiment weight which is the average of this abstracted tokens.

**Results**

After acquiring the related features from the data a multitude of machine learning algorithms were applied with those features and their labels. The teams strategy was to be able to see the patterns with correlating with features so that best results can be obtained from the given test data. In order to realize that first a series of experiments were done by using those 21 given features. By using that a base for next experiments was created. Then, the extracted features from the teams bucketize method and the other generated features were run to check if they really were suitable for to be used in the end. After validating our features by doing regression with only those features, given features and those extracted ones were combine. Thus, creating a feature space with 33 parameters for each sample. However, after conducting series of experiment with that, the team has concluded that even these features were not sufficient to train the algorithms. That was mainly because of the fact that score of those experiments were very low meaning that the scores were below 0.20. The score is evaluated by the function of “Scikit-learn”. The scoring function returns “the coefficient of determination R^2 of the prediction. The coefficient R^2 is defined as (1 - u/v), where u is the residual sum of squares ((y\_true - y\_pred) \*\* 2).sum() and v is the total sum of squares ((y\_true - y\_true.mean()) \*\* 2).sum(). The best possible score is 1.0 and it can be negative (because the model can be arbitrarily worse). A constant model that always predicts the expected value of y, disregarding the input features, would get a R^2 score of 0.0”.

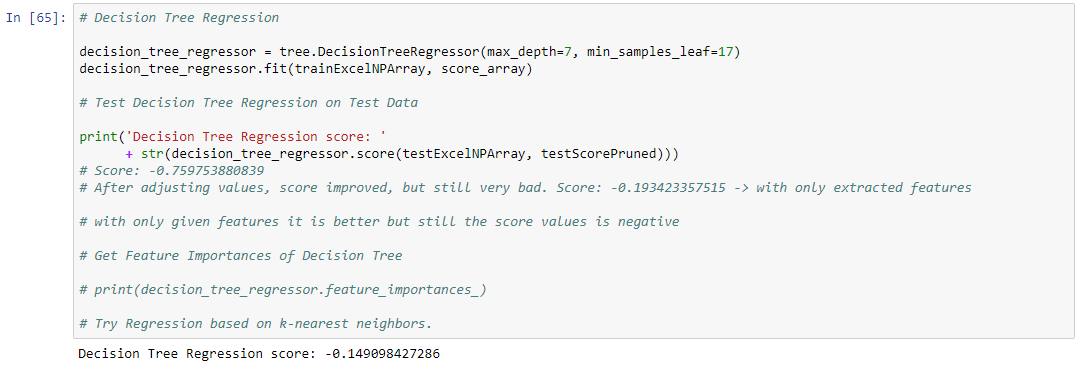
Hence, features coming from bag of words method were included in the feature vectors as well. However, It was not logical to use all the features coming from bag of words method. That could lead to exaggeration of those features and could lead to have not all relevant features in the feature space. Considering that most of those over a thousand features were eliminated by their relevancy. The relevant features are selected from “Select From Model” method. Only those features which are well above the threshold of 2\*mean were selected. To check the validation of those features, they were tested with several regression algorithms as well. By only having those features the limit of score were not surpassed. Thus, it was decided to combine all the features at hand to surpass that limit. By combining all of the features total of 175 features are gathered. However, even with them the best score was still around the value of 0.20. In the appendix the results of those experiments can be found. And most of them are in the code file submitted with this report file. Moreover, the best results obtained are shared within the same folder.

Note: Since the test data had also labels, it was possible to obtain scores of those regressions. As can be seen in the appendix. Some of the algorithms were failed and returned minus scores. Whereas others were able to generate comparably better results.

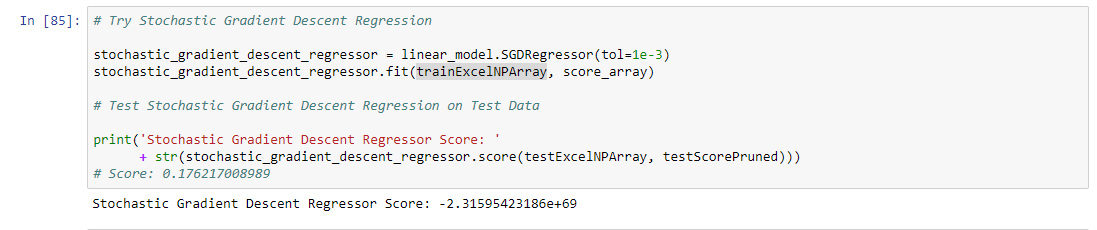
IMPORTANT NOTE: Last late submitted file contains new features and a new approach. The new one is that during the tokenization process the first 6 characters of strings were processed. That lead to an increase in the scores such that new best score is around 0.29. Compared to the given 21 features’ score which is 0.15 the score is doubled making our predictions better. Both previous and new java codes are added.

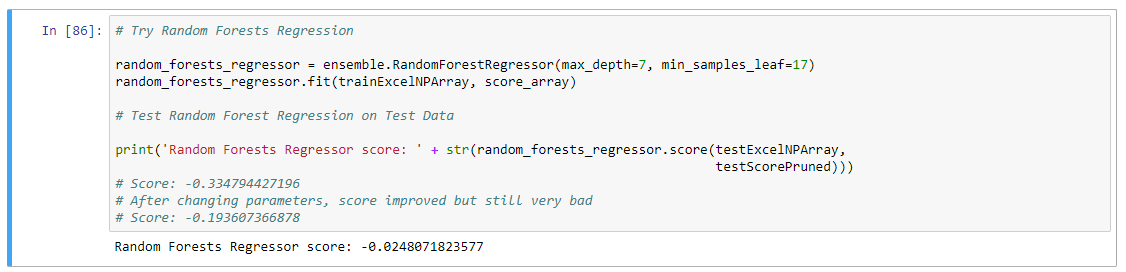
**Appendix**

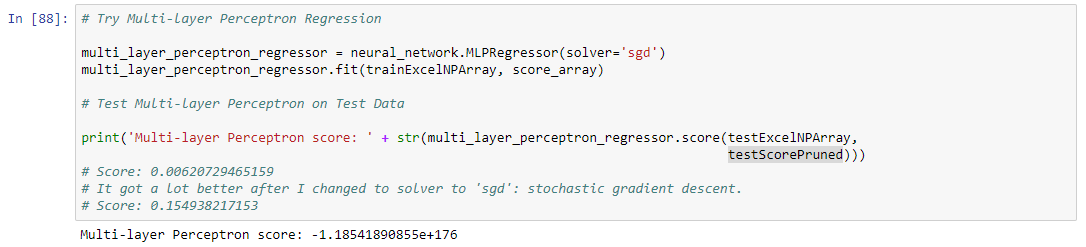
**Regressions with given 21 features**

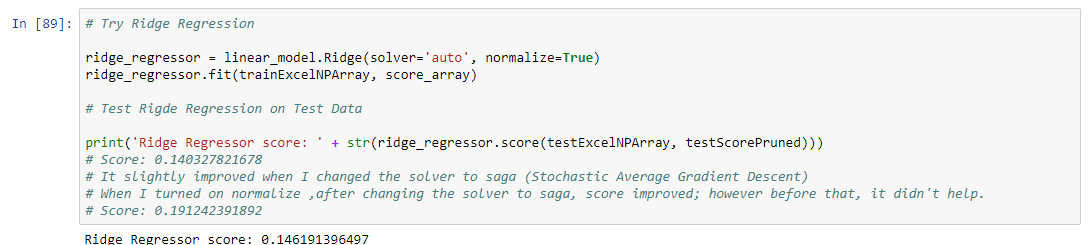


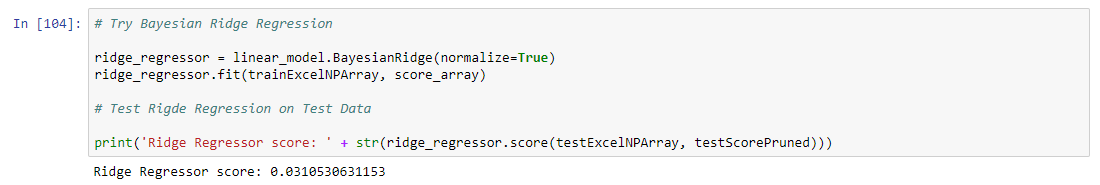








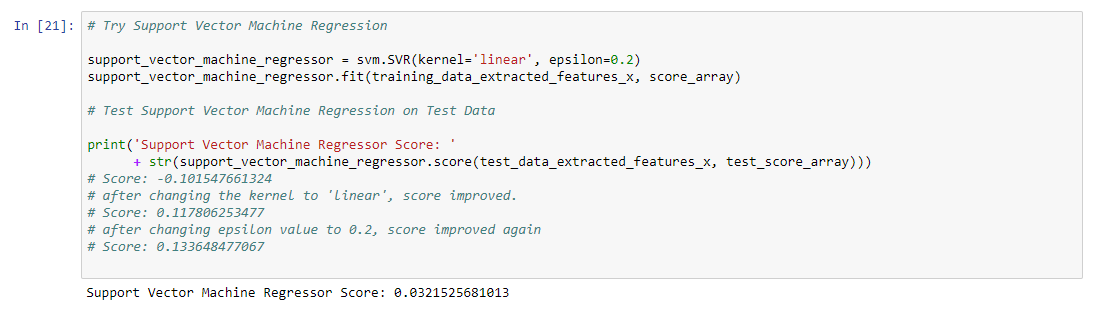


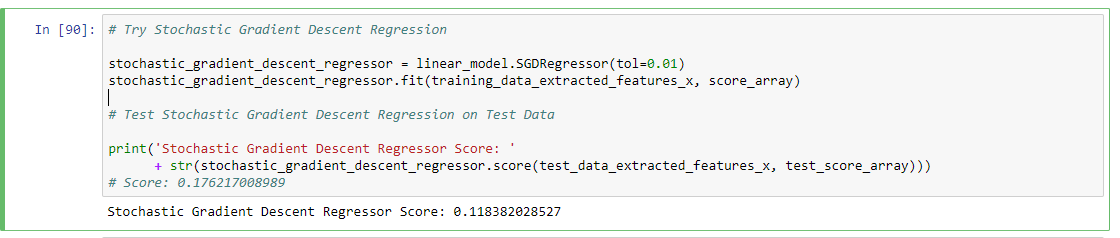


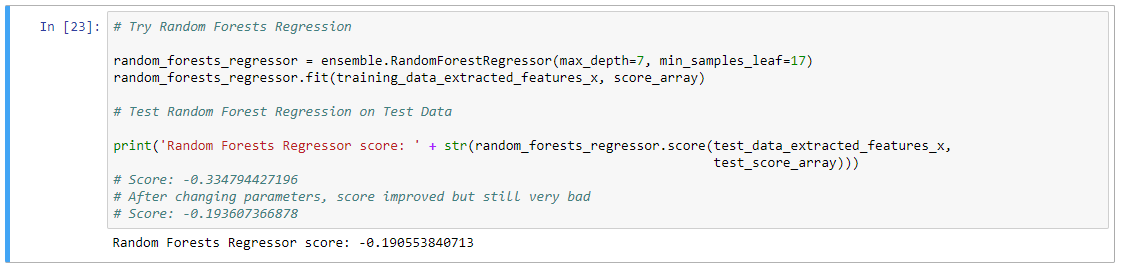
**Regression with only extracted features**

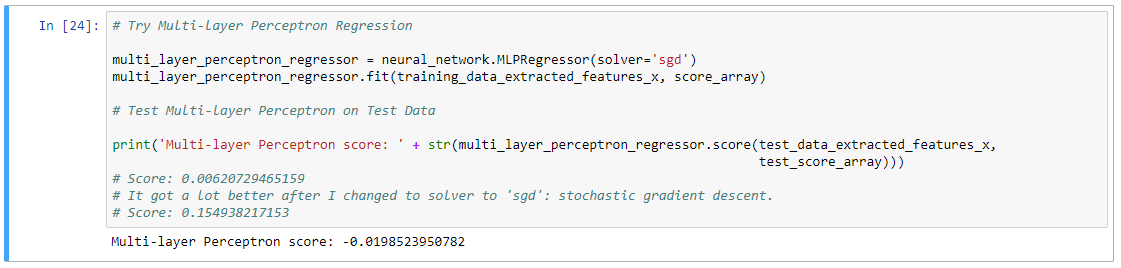


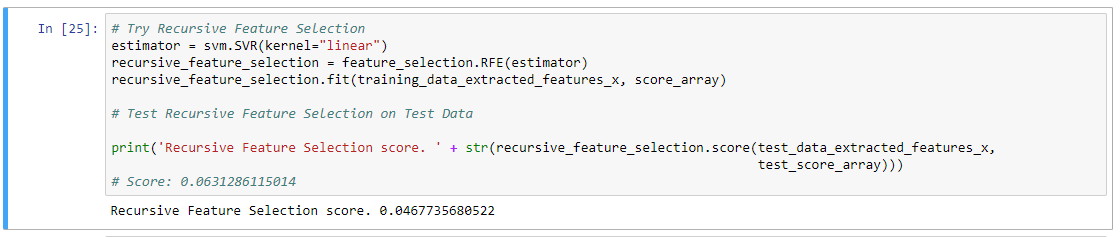


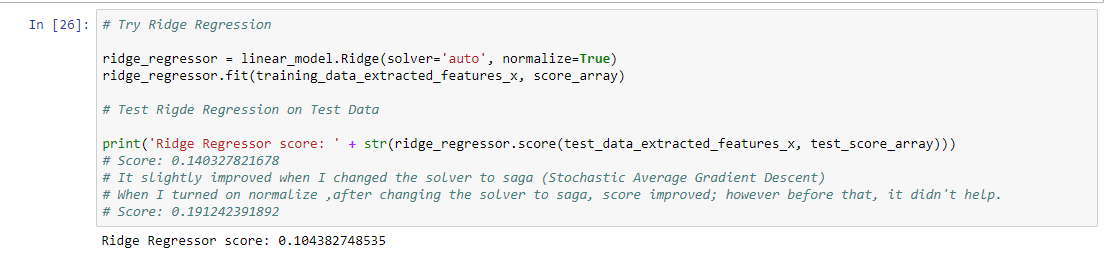






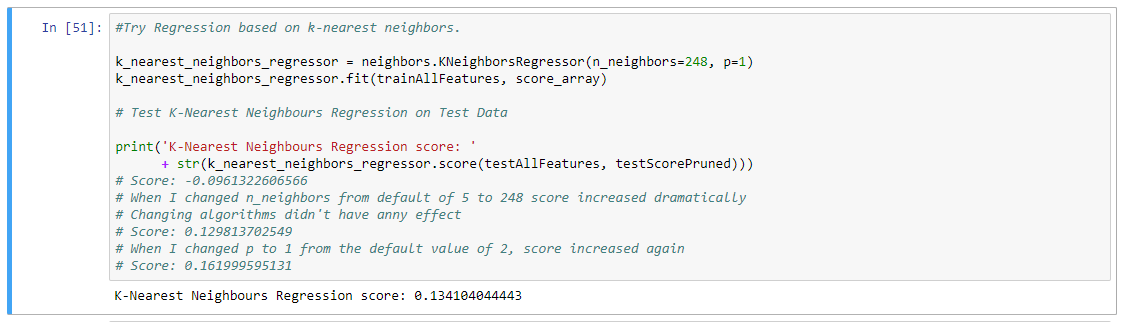




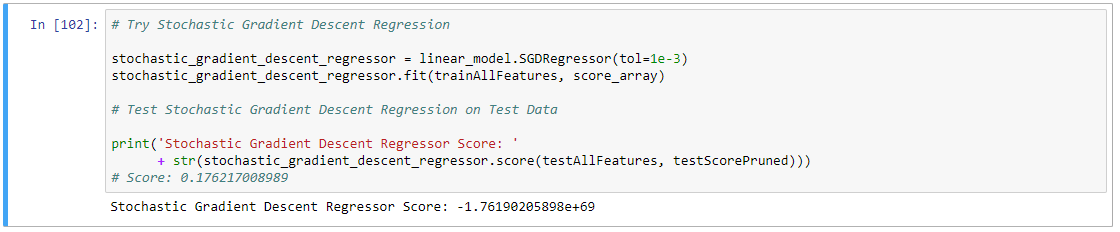
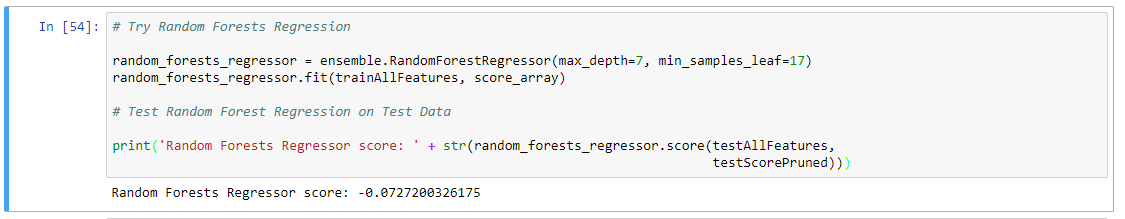


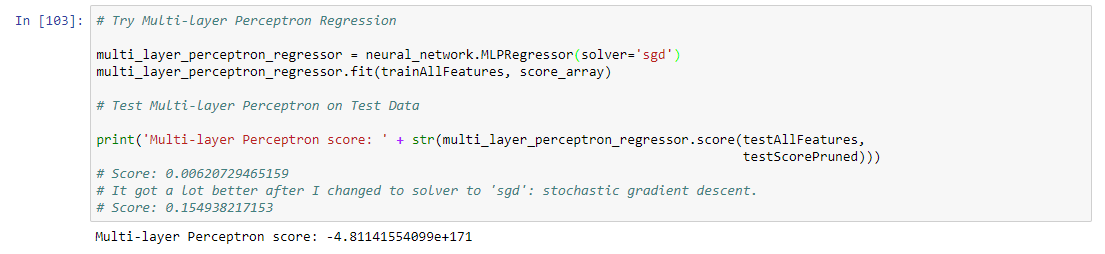
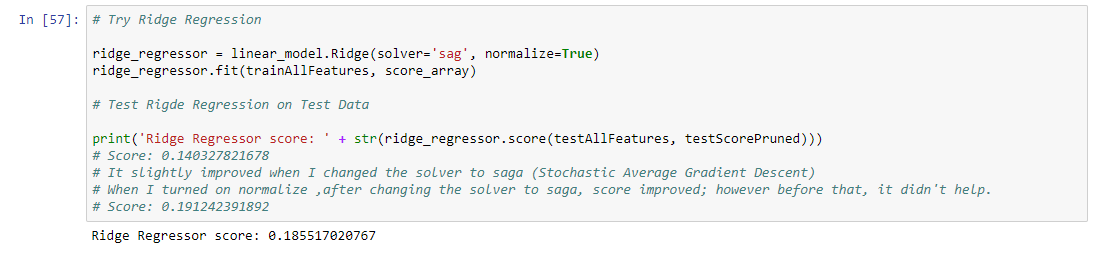
**Regressions with given 21 features and extracted features**

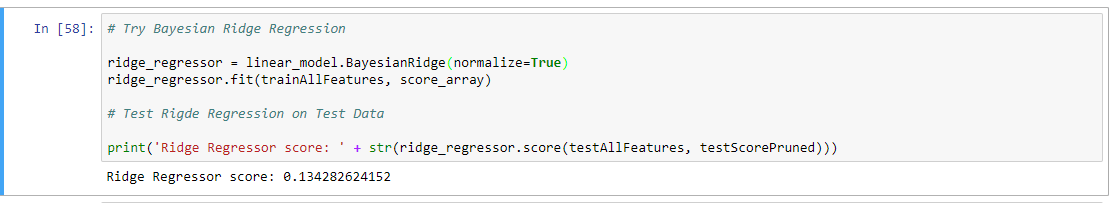




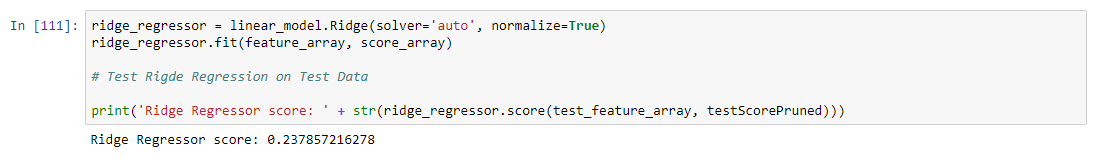


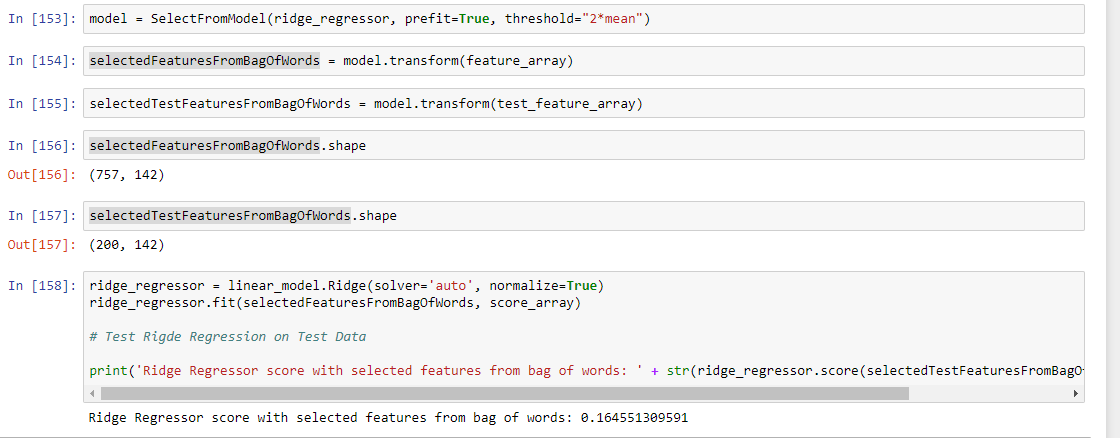
 



**Regressions with all features combined including those coming from bag of words**



**Note: After obtaining this score, it is decided to add some of those features as well.**



Threshold value was changed respectively to optimize number of features to be added and their possible significance on the results.

Combination of all those features are as follows;

