**NATURAL LANGUAGE PROCESSING USING R**

***INTRODUCTION***

*In this project we will cover basics of clustering, topic modeling, and classifying documents in R using both unsupervised and supervised machine learning techniques and compare some different methodologies.*

***WHY R STUDIO FOP NATURAL LANGUAGE PROCESSING?***

***R is***

1. *A library of statistical tools*
2. *An interactive environment for statistical analyses and graphics*
3. *A programming language*
4. *Public free software derived from the commercial system*

*R is becoming more and more popular especially for its effective data handling and storage facility and large, coherent, integrated collection of tools for data analysis*

*Well-developed, simple and effective programming language*

****

**WHY NATURAL LANGUAGE PROCESSING IS IMPORTANT?**

Applications for processing large amount of texts require NLP expertise.

***Classify text into categories, index and search large texts***:

Classify documents by topics, language, author, spam filtering, information retrieval (relevant, not relevant), and sentiment classification (positive, negative).

***Extracting data from text***: converting unstructured text into data

***Information extraction***: discover names of people and events they participate in, from a document.

***Automatic summarization***: Condense 1 book to 1 page

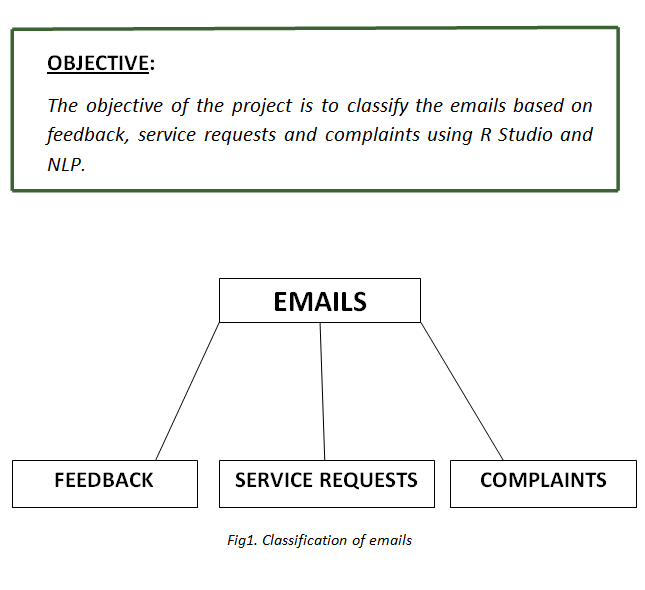
***Speech processing, artificial voice***: E.g. Book hotel over a phone.

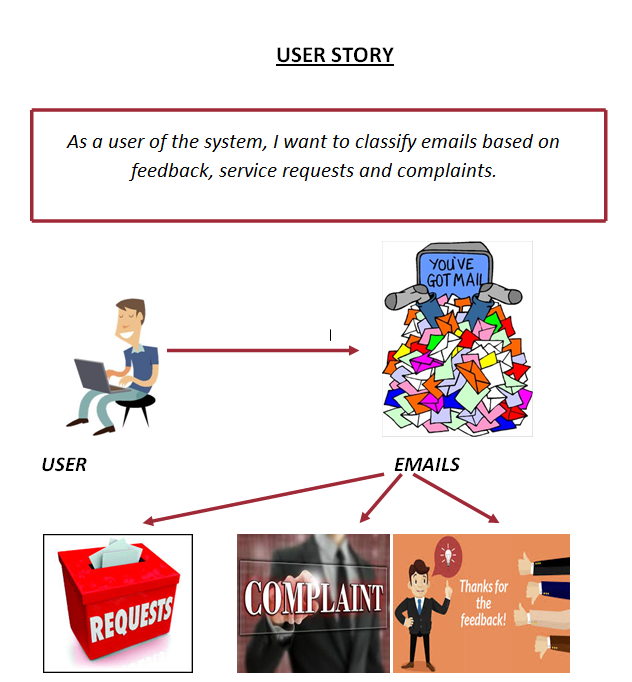
***Question answering***: find answers to natural language questions in a text collection or database.

***Spelling and Grammar Corrections***

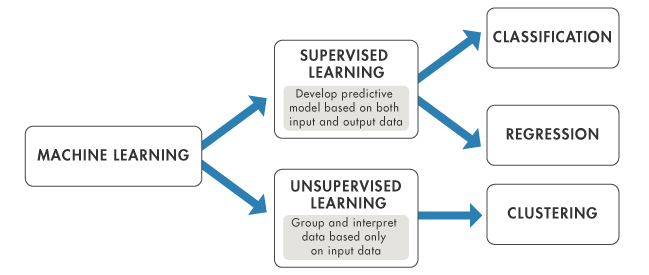
***Plagiarism detection***

***Automatic translation***





***MACHINE LEARNING CLASSIFICATION:***

******

*Figure1. Machine learning classification*

***SUPERVISED MACHINE LEARNING AND UNSUPERVISED MACHINE LEARNING***

### *Supervised Learning*

*This is predictive ML where the main goal is to use the existing knowledge of the data set to predict an unknown property for new data set.*

*It is like statistics where you have a hypothesis and you are trying to prove it - you use properties of a subset of data and apply it to more real world data.*

### *Unsupervised Learning*

*This is exploratory ML where you do not know what you are looking for.*

*Previous knowledge of data is not required.*

*It explores the raw data for you and gives information of any existing patterns or trends*.

## *What is a label?*

*Label is the feature you want to learn from the known data set and predict for the unknown data set.*

***EXAMPLE***

*Suppose you have 500 fruits of type apples, oranges and bananas.*

*For each fruit you have the weight, sweetness & acidity measures recorded. Now, say that, while in transit 200 fruits fall off of the truck.*

*So if you want to predict the type of fruit for the missing 200 using ML, then 'fruit type' is the label; and weight, sweetness & acidity are the features. And based on the knowledge of features of 500 fruits and label information of 300 fruits we can predict the label for the missing 200 fruits.*

*Columns = label + features and are like class definition  
Rows = instances of your label & features and are like objects*

*The above "fruit type" prediction is an example of* ***supervised learning*** *- we start with labels, we have a property we know for some data and we predict that property or label for new data. When label is categorical we use classification, and when label is numerical we use regression, supervised learning algorithms.*

*Unsupervised learning is exploratory analysis and there is no 'label' associated with it.*

*Clustering is unsupervised learning method where the clusters are formed using the features.*

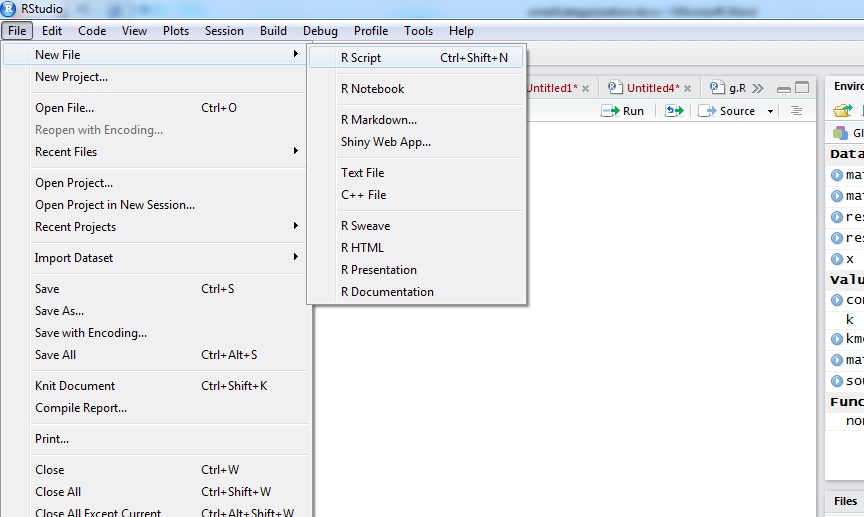
*In our fruit example, if we just scatter plot sweetness vs. acidity, we will be able to see high density regions, called clusters, separated by low density regions. These clusters group similar objects but we do not know what they correspond to in the real world.*

*Say, the three clusters are c1 c2 c3  
- c1 is cluster with high acidity and low sweetness  
- c2 is cluster with medium acidity and medium sweetness  
- c3 is cluster with low acidity and high sweetness*

*So fruit type with high acidity and low sweetness will be in c1, medium acidity and medium sweetness in c2; and low acidity and high sweetness in c3.*

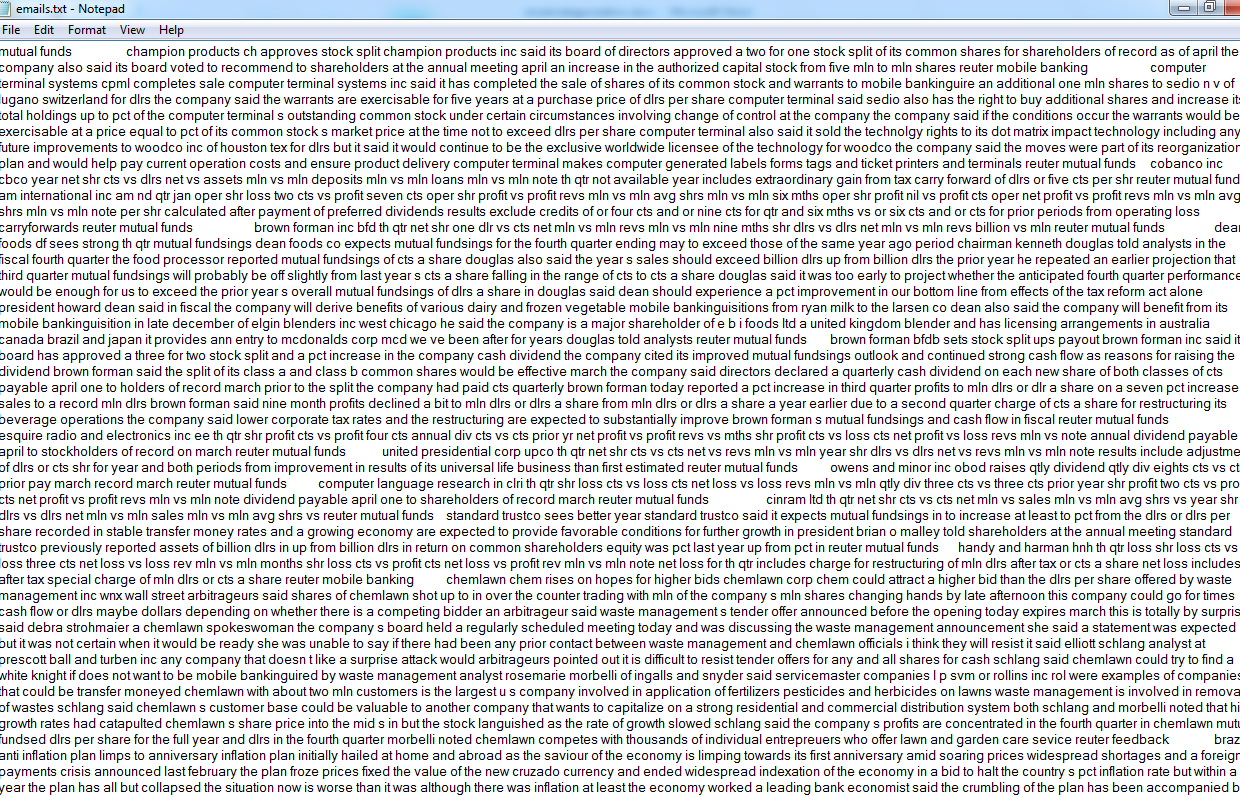
***IMPLEMENTATION:***

1. ***CREATE A NEW PROJECT IN R STUDIO***



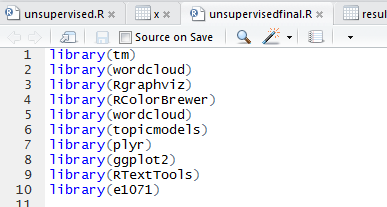
*Fig1. Creating a new R Script*

***2. CREATING A TEXT FILE***

******

*Figure2. Creating a text file*

1. ***CODE (Unsupervised)***

******

*Fig3. Loading the Libraries*

***Libraries used:***

***library(tm)****- A framework for text mining applications within R.*

***library(wordcloud)****-* ***Text mining*** *methods allow us to highlight the most frequently used keywords in a paragraph of texts. One can create a* ***word cloud****, also referred as* text cloud *or* tag cloud*, which is a visual representation of text data.*

***library(Rgraphviz)****- Interfaces R with the AT and T graphviz library for plotting R graph objects from the graph package.*

***library(RColorBrewer)-*** *Provides color schemes for maps (and other graphics).*

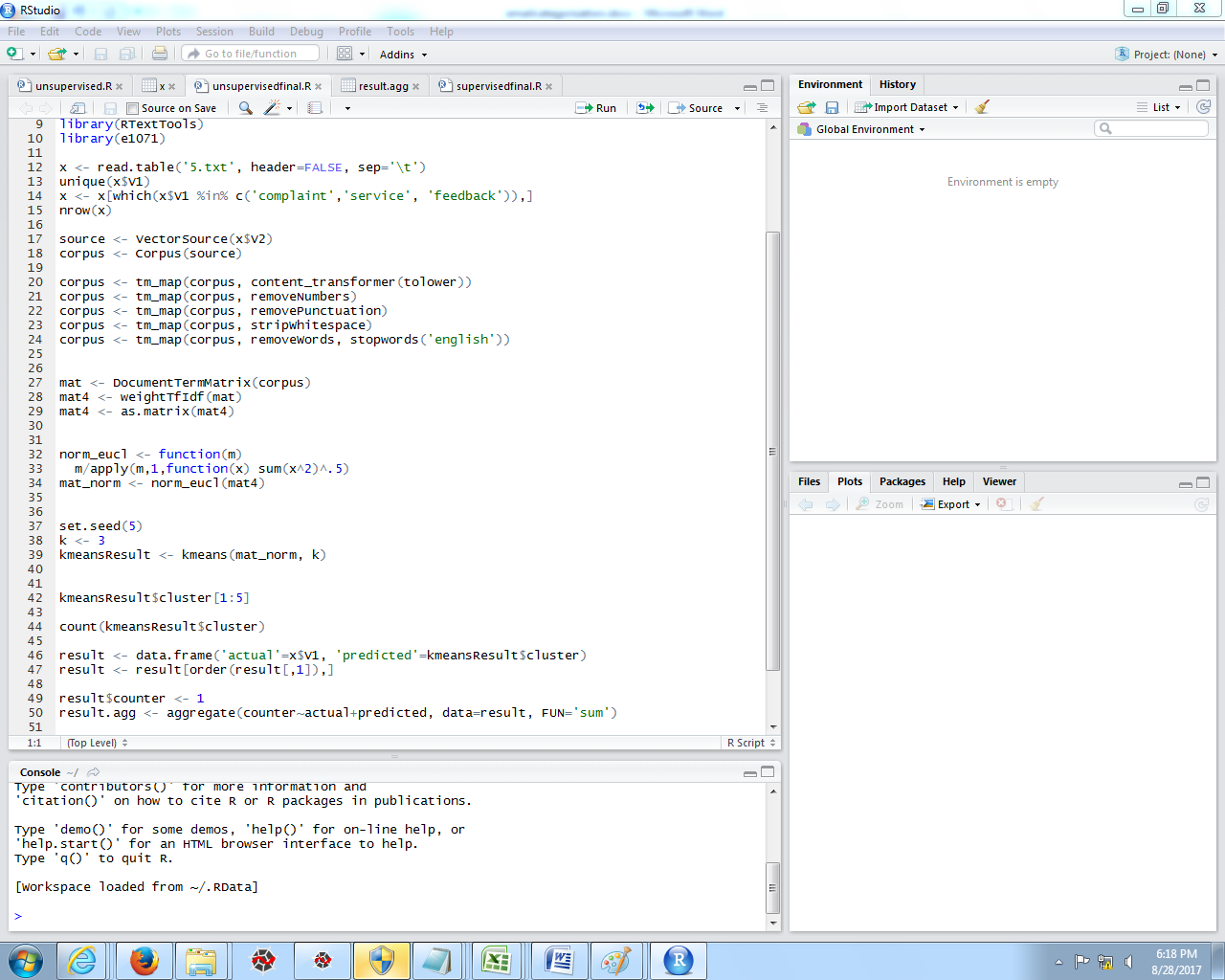
***library(topicmodels)****- Provides an interface to the C code for Latent Dirichlet Allocation (LDA) models and Correlated Topics Models (CTM)*

***library(plyr)****- Tools for Splitting, Applying and Combining Data.*

***library(ggplot2)****- ggplot2 is a plotting system for R, based on the grammar of graphics, which tries to take the good parts of base and lattice graphics.*

***library(RTextTools)****- RTextTools is a machine learning package for automatic text classification that makes it simple for users to get started with machine learning, while allowing experienced users to easily experiment with different settings and algorithm combinations. The package includes nine algorithms for ensemble classification (svm, slda, boosting, bagging, random forests, glmnet, decision trees, neural networks, maximum entropy), comprehensive analytics, and thorough documentation.*

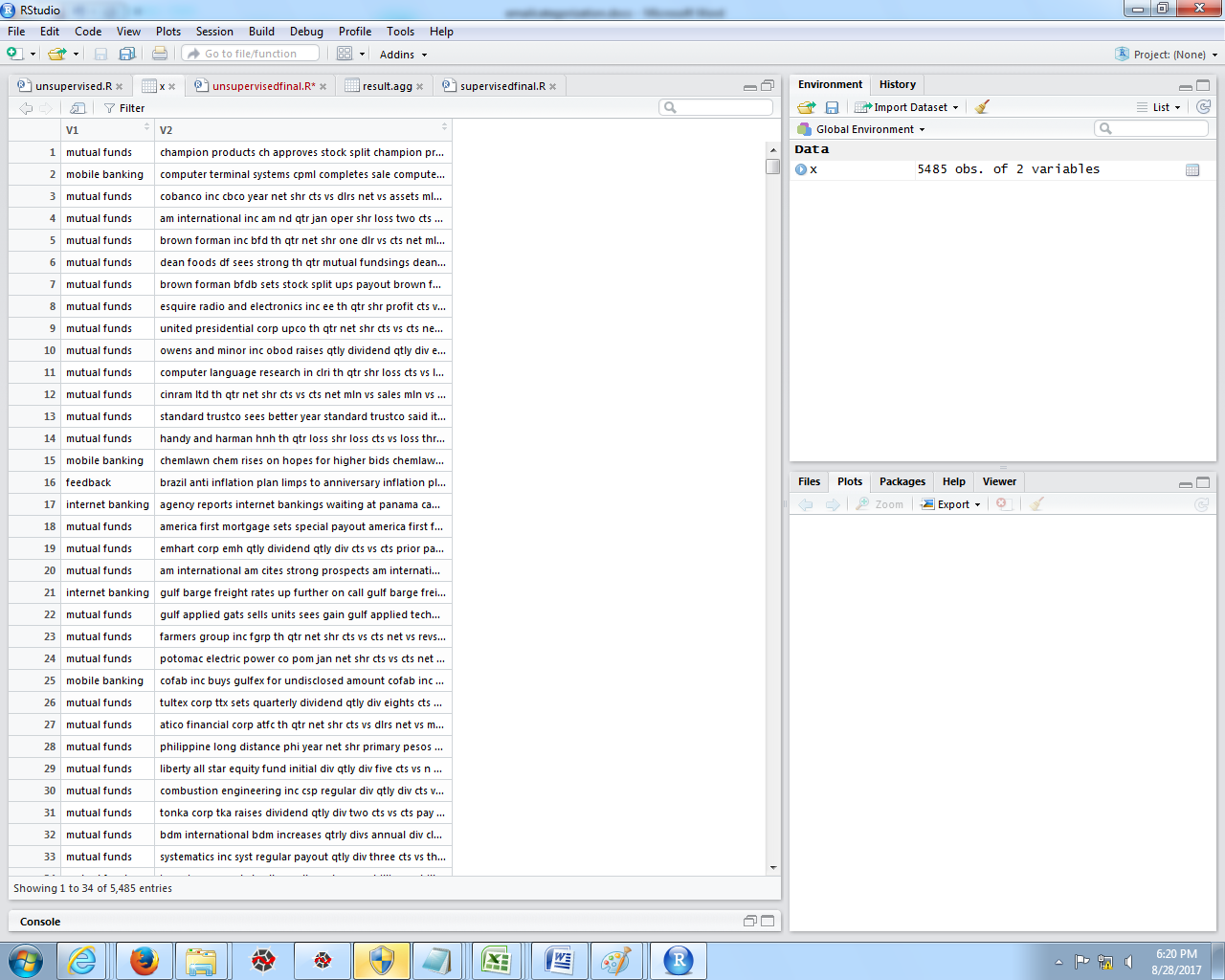
***library(e1071)****- Functions for latent class analysis, short time Fourier transform, fuzzy clustering, support vector machines, shortest path computation, bagged clustering, naive Bayes classifier, etc.*

****

*Figure5*

*Reading the file:*

***x<-read.table(‘emails.txt’, header=FALSE, sep=’\t’)***

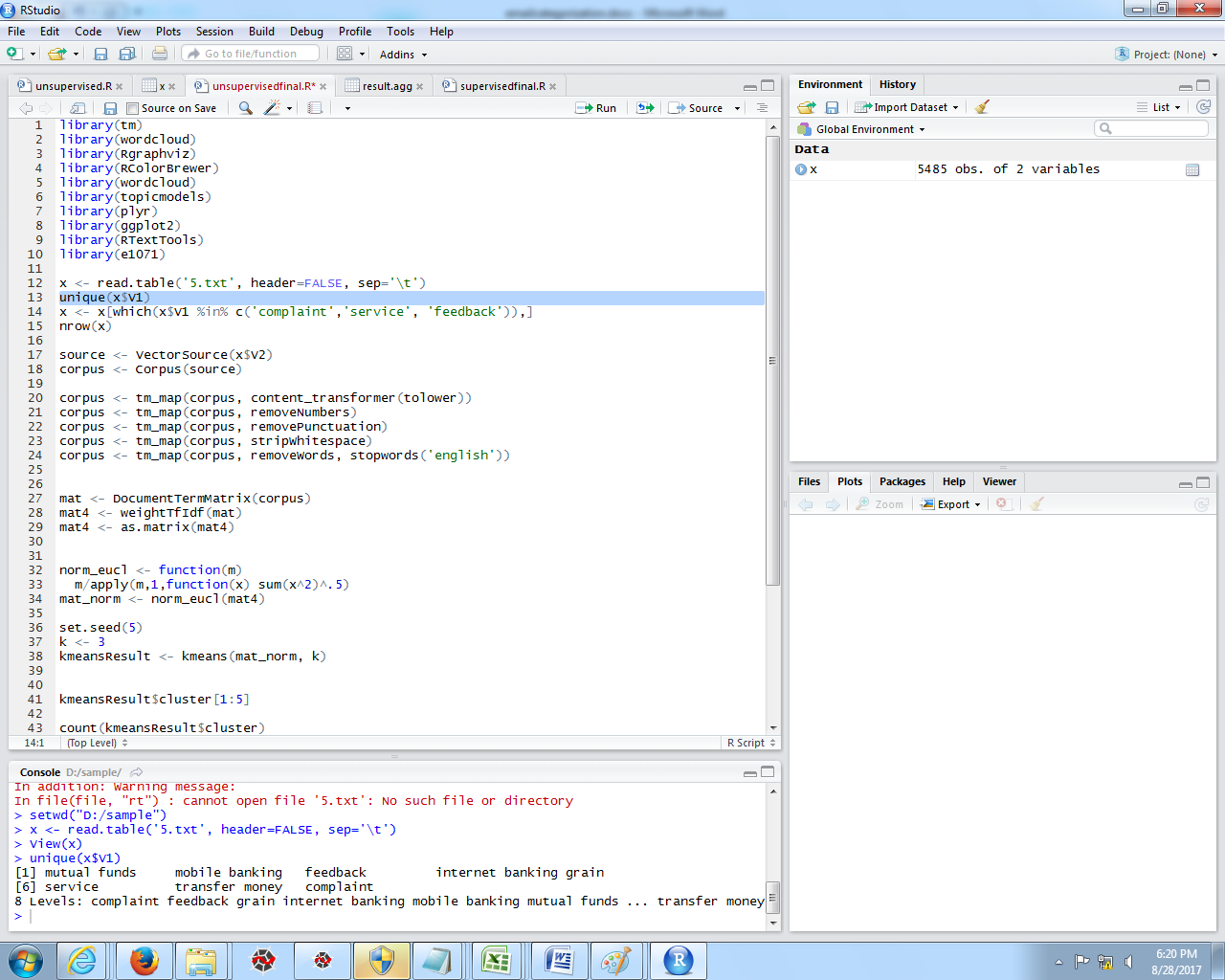
****

*Figure4: Reading the file*

*Now we have a data frame where a row represents a document with one column containing the document text (V2) and another column containing the tag for the topic (V1). The tags we have are:*

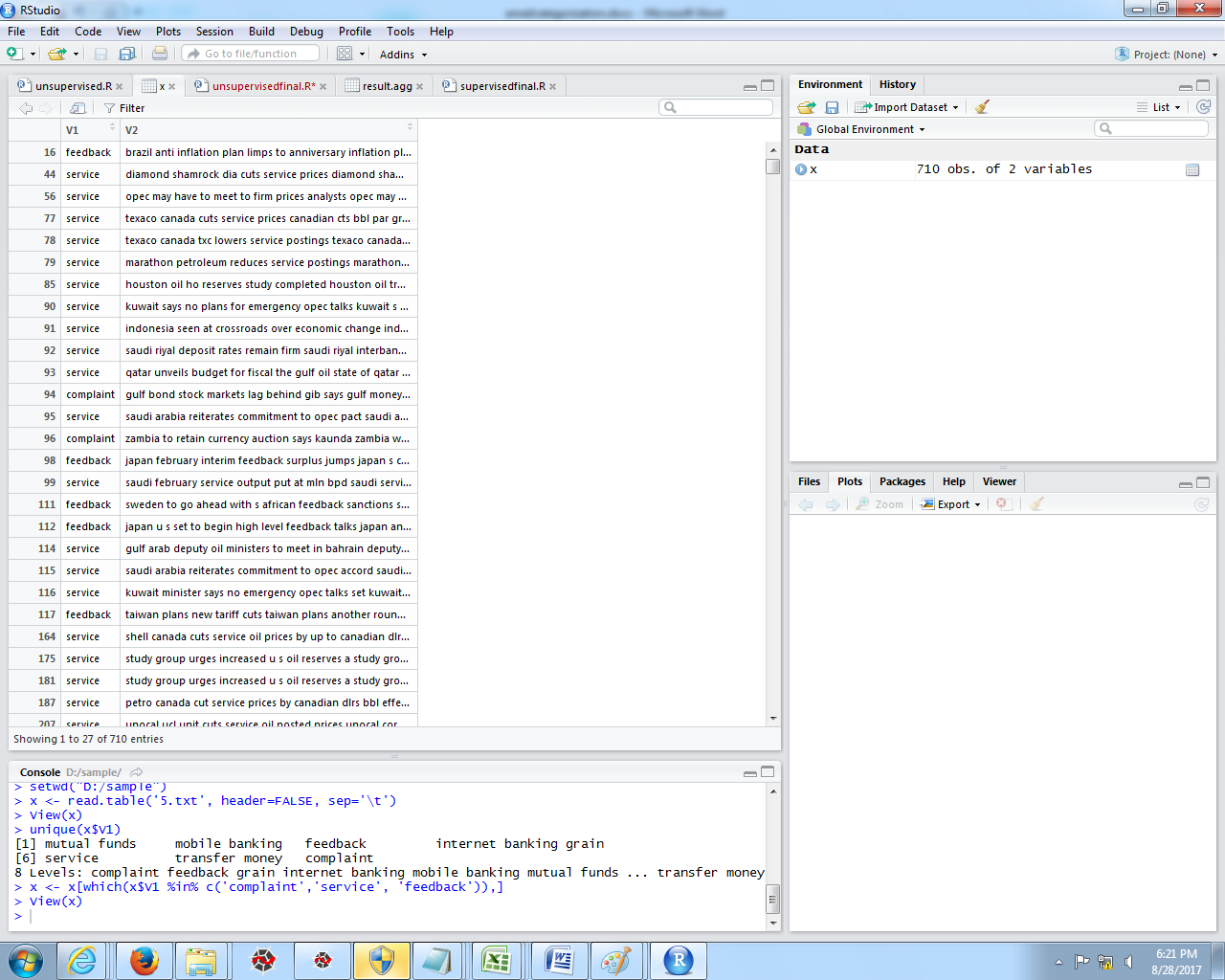
***unique(x$V1)***

*The tags are : mutual funds, mobile banking, feedback, complaint, service request, transfer money and internet banking.*

****

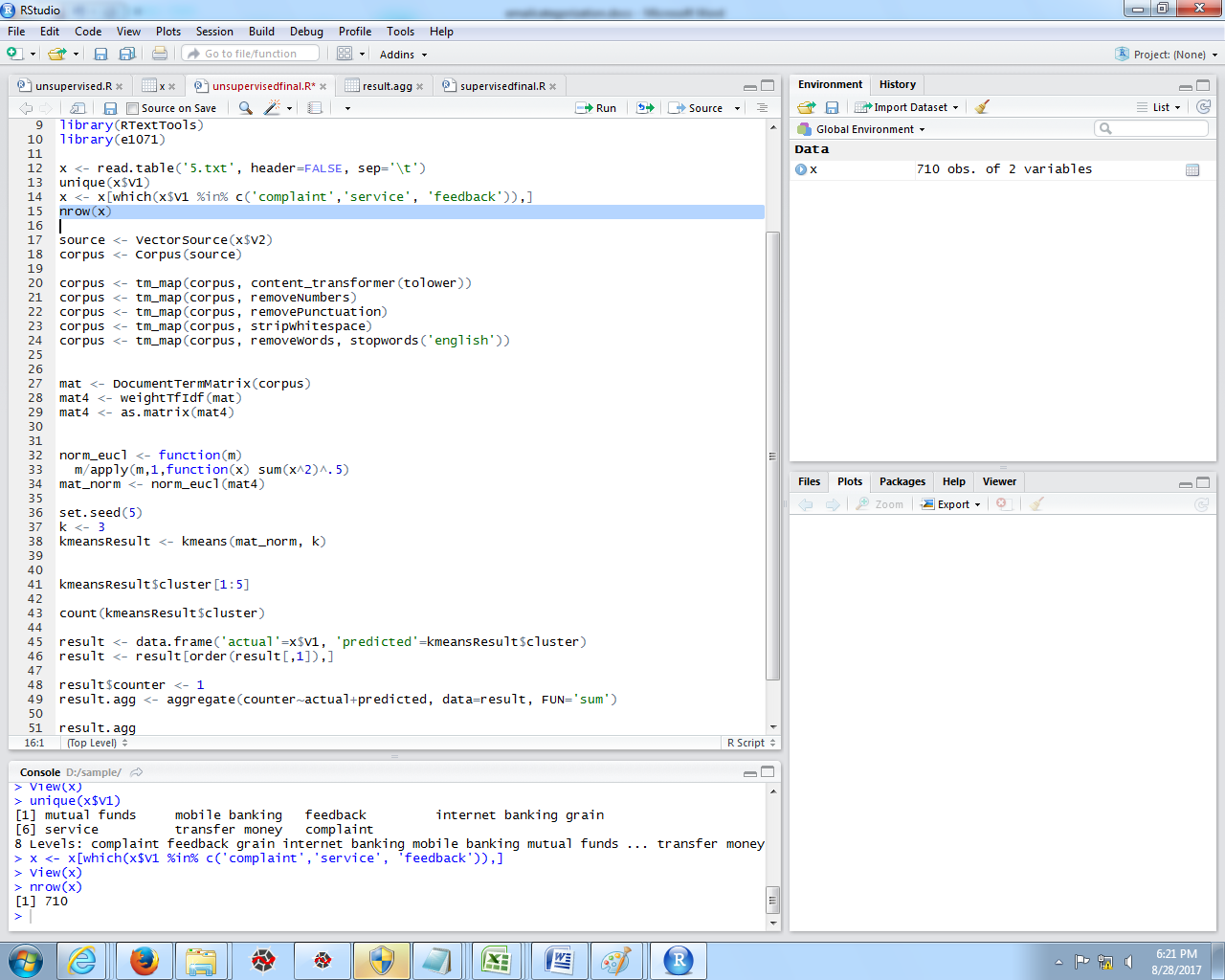
*Figure 6*

*Our data set contains 5,485 documents. This is good - especially for supervised analysis - because we will have plenty of documents on which to base our model. In the interest computational expense, lets limit to three of the document tags: feedback, service request, and complaint.*

****

*Figure7:Observation*

*x<- x[which(x&V1 %in% c(‘feedback’, ‘service request’, ‘complaint’)*

****

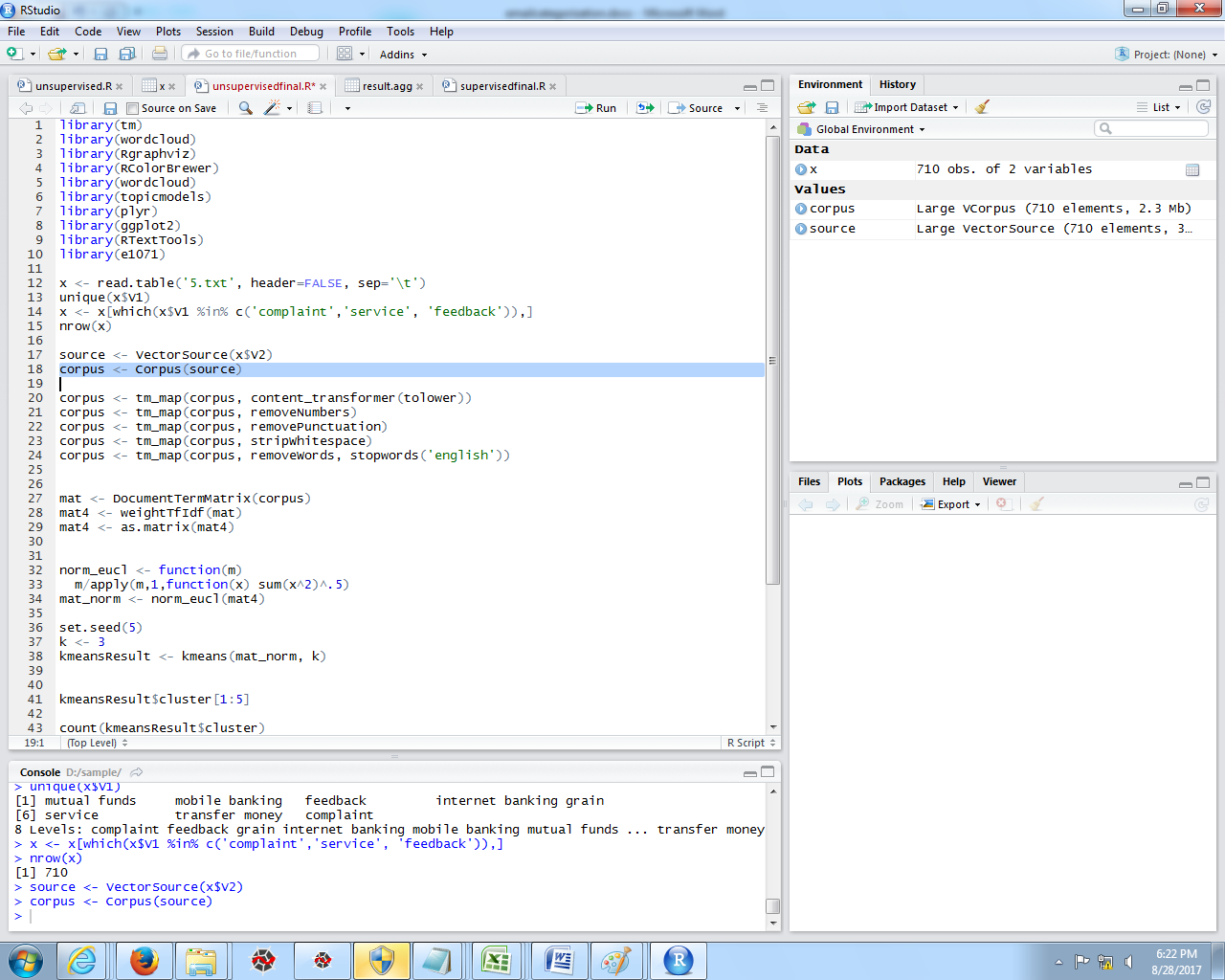
*Figure 8*

*Now we are down to 710 documents about our three selected subjects.*

*We can make a corpus from the column containing the document text:*

***source <- VectorSource(x$V2)***

***corpus <- Corpus(source)***

****

*Figure9. Cleaning the data*

*We will take the standard steps to clean and prepare the data:*

*corpus <- tm\_map(corpus, content\_transformer(tolower))*

*corpus <- tm\_map(corpus, removeNumbers)*

*corpus <- tm\_map(corpus, removePunctuation)*

*corpus <- tm\_map(corpus, stripWhitespace)*

*corpus <- tm\_map(corpus, removeWords, stopwords(‘english’))*

***UNSUPERVISED ANALYSIS***

1. ***CLUSTERING***

*We will use K means clustering.*

*K-means basically tries to cluster the individuals in a data set by comparing them across many variables.*

*In the text mining case, these variables come from word frequencies.*

*Lets first create a document-term matrix:*

***mat <- DocumentTermMatrix(corpus)***

*At this point, we could just move on to our clustering with this matrix, but we will instead create a weighted T-Ida version of the matrix.*

*This method - short for term-frequency/inverse-term-frequency - takes into account how often a term is used in the entire corpus as well as in a single document.*

*The logic here is that if a term is used in the entire corpus frequently, it is probably not as important when differentiating documents.*

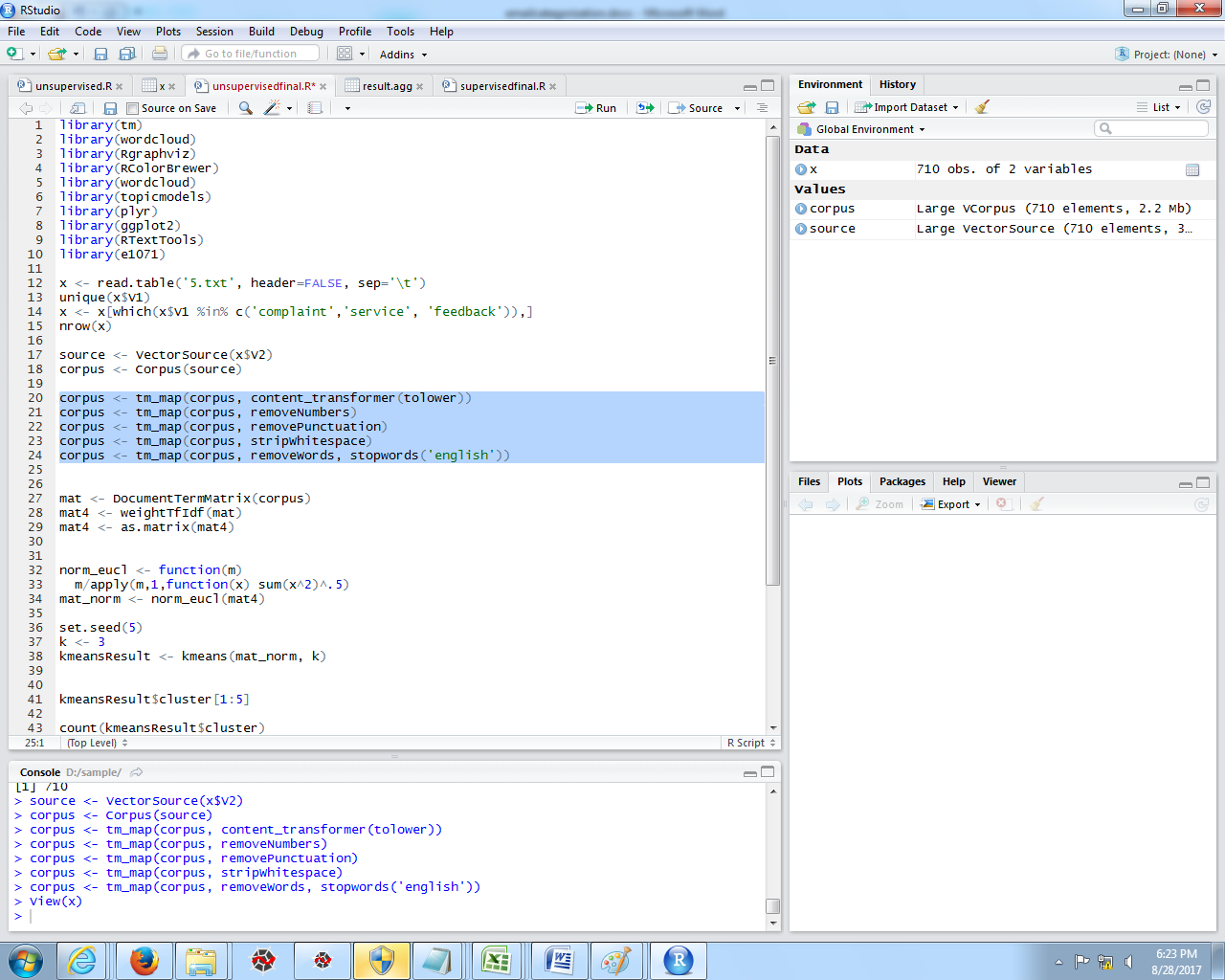
*Alternatively, if a word appears rarely in the corpus, it may be an important differentiation even if it only occurs a few times in a document.*

***Mat4 <- weightTfIdf(mat)***

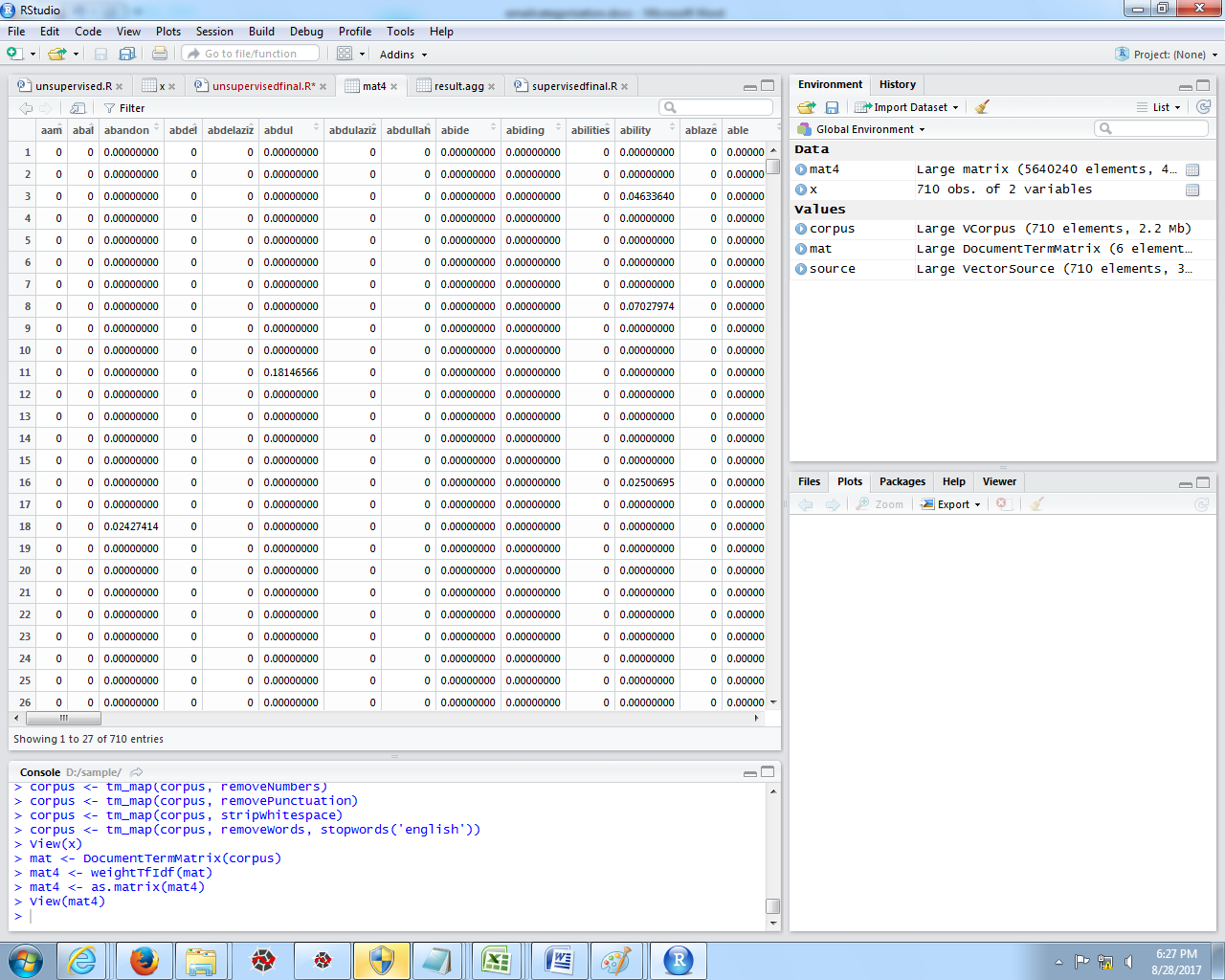
***Mat4 <- as.matrix(mat4)***

*Finally, we will normalize the T-Ida scores by euclidean distance. This is one of many scoring methods in text mining.*

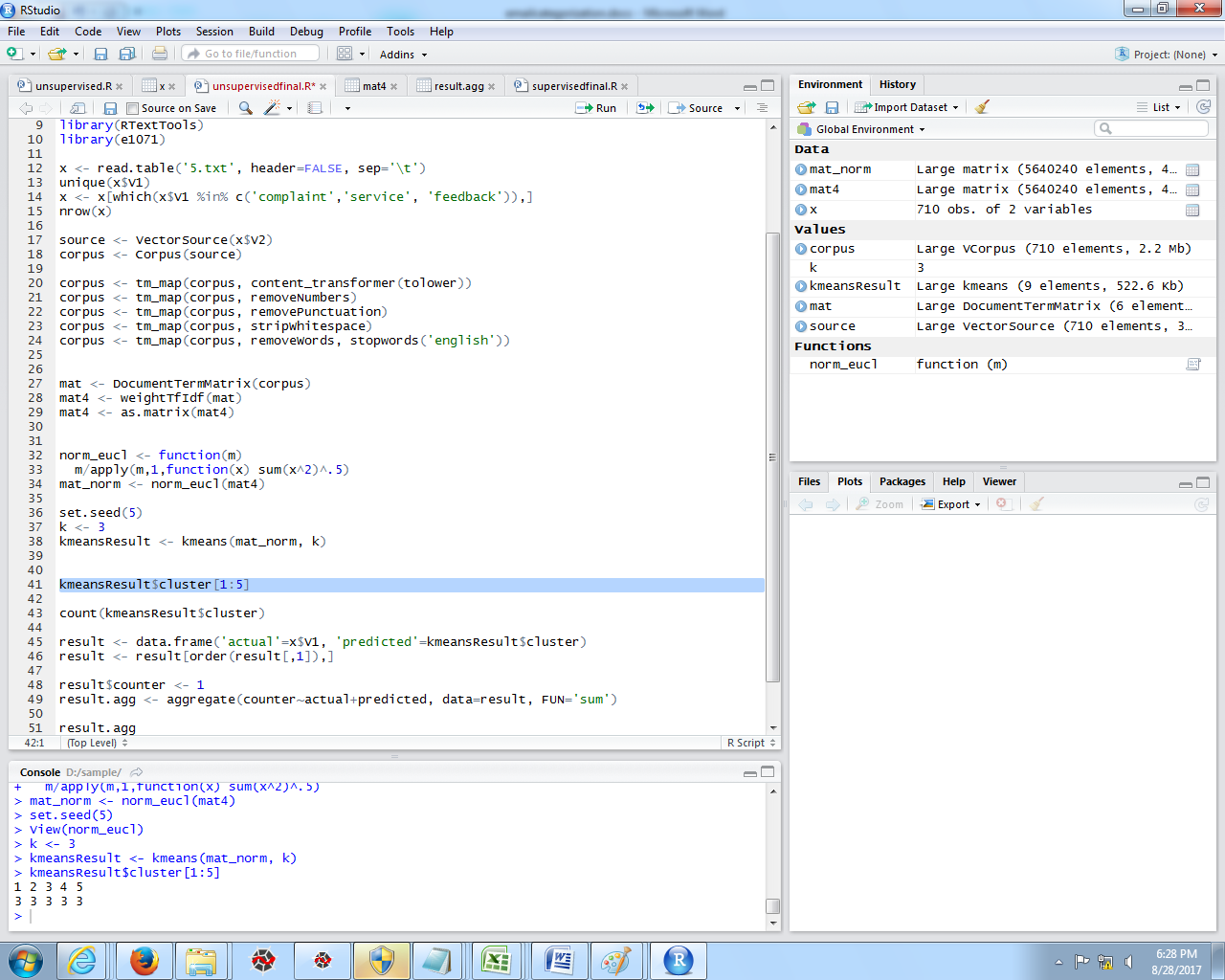
*Now we can run the k-means algorithm.*

****

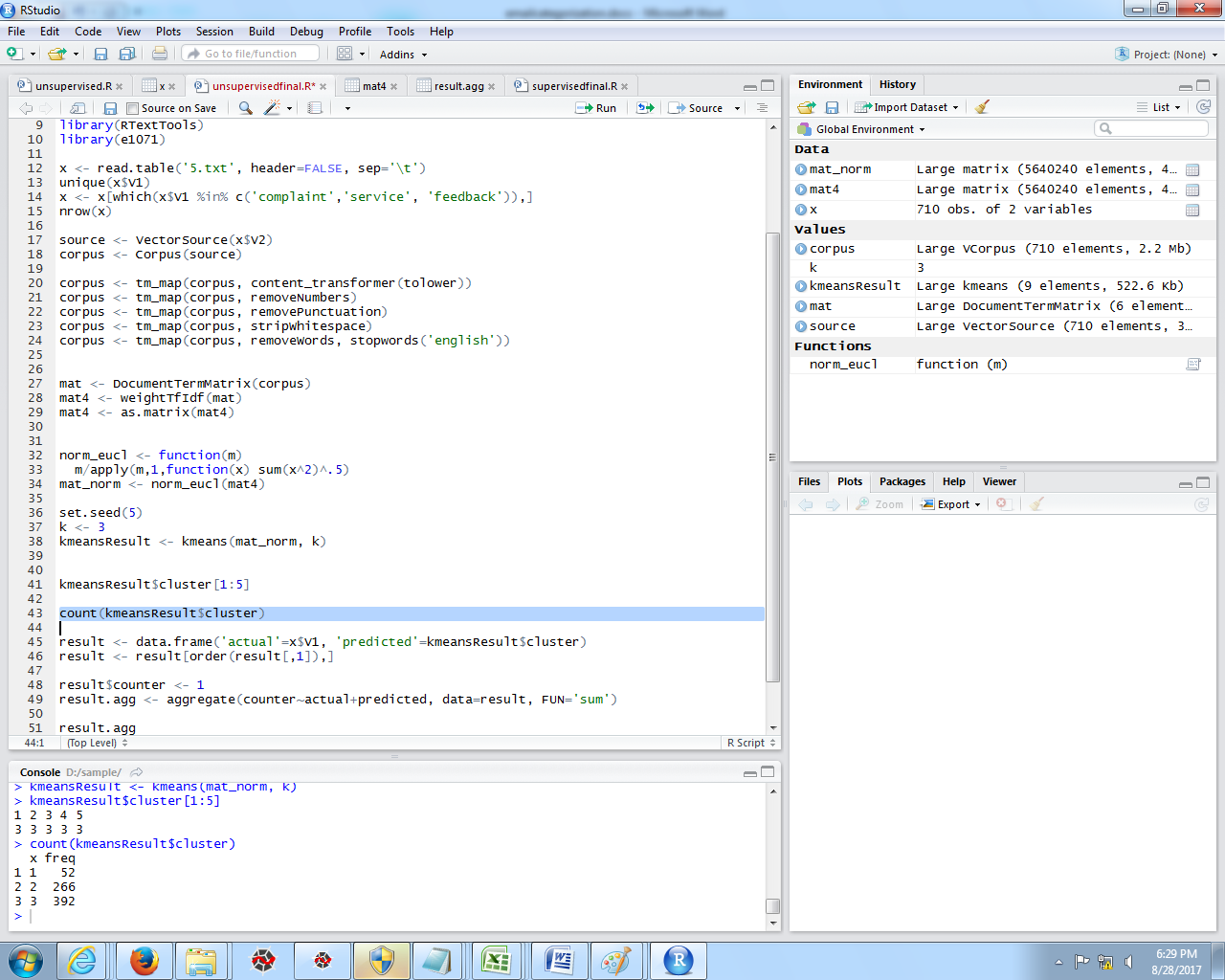
*Figure 10*

****

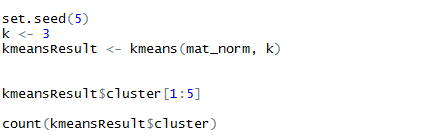
*Figure 8:matrix*

****

*Figure 11*

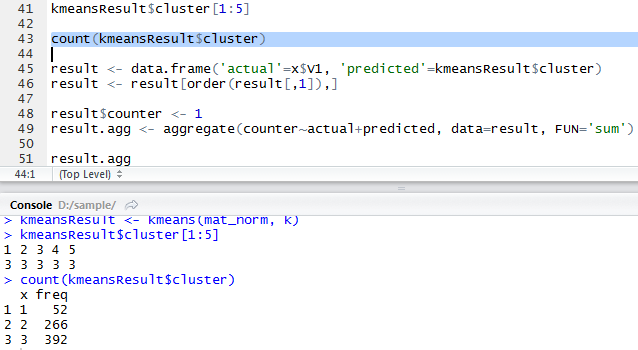
****

*Figure 12.Result of K cluster*



*Now we have a model object called kmeansResult.*

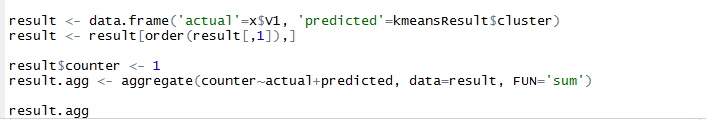
*The****set****.****seed****()function in* ***R****takes an (arbitrary) integer argument. So we can take any argument, say, 1 or 123 or 300 or 12345 to get the reproducible random numbers.*

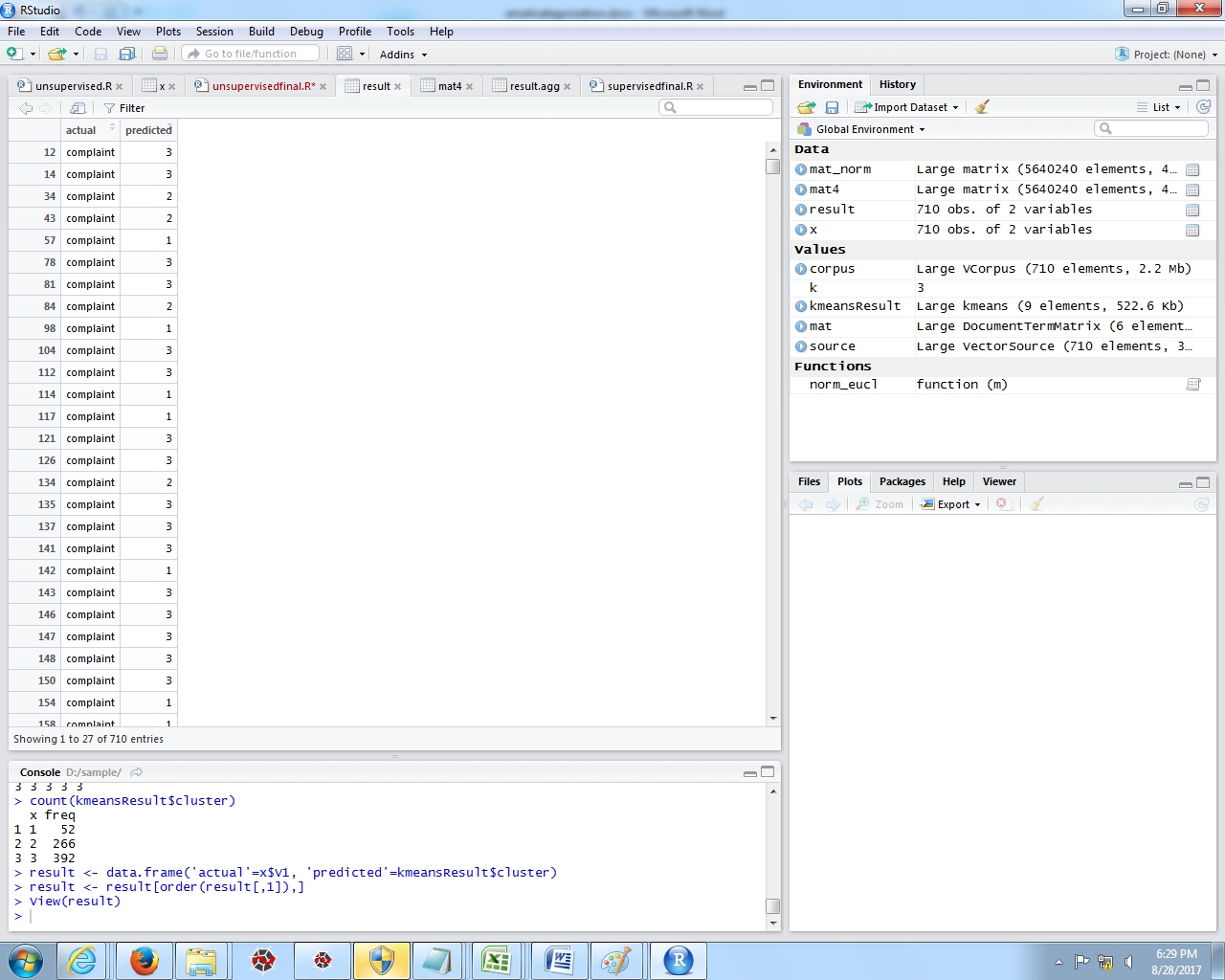


*We can see that documents 1 through 5 are all in cluster 3. We have 52 documents in cluster 1, 266 documents in cluster 2, and 392 documents in cluster 3.*

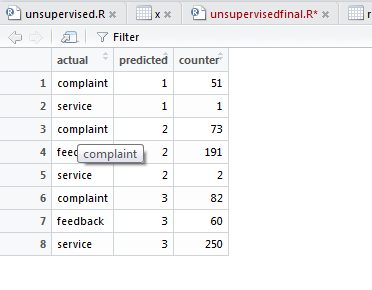
*In true unsupervised analysis, this is all we could get. We would know which documents were grouped together, but would need to dive into the actual documents to see what (if anything) this means.*

***MODEL PERFORMANCE***

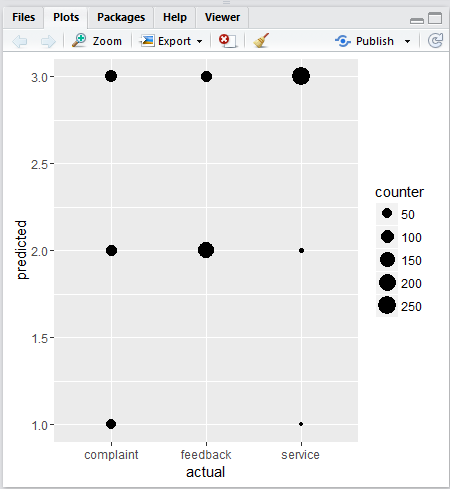


****

*Figure13. Actual and Predicted values*







*Figure 15. Plotting the graph.*

***SUPERVISED ANALYSIS***

*Now using supervised analysis.*

*These methods are only applicable if you have tagged data.*

*We will need to split our data into training and test sets in supervised analysis - both of which need to be tagged.*

*We first train a model on the training data, then apply it to the test data and see how well it did (otherwise, we could just be over-fitting to the training data).*

*If our data is sufficiently large and representative, our accuracy numbers should give us a good idea of how well the model will perform on data we don’t have.*

*A good rule of thumb is to use 80% of the data to train and 20% to test.*

*We should also look into n fold cross validation for further machine learning applications.*

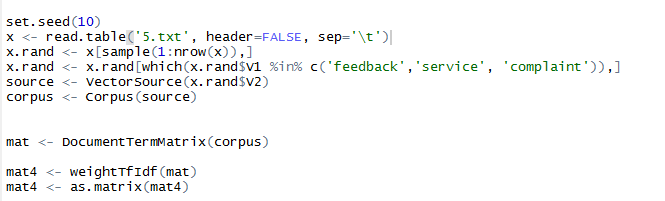
*For a quick example: If we use 10 fold cross validation, our data will be split into 10 groups.*

*The algorithm we choose will run 10 time, each time using a different group (fold) as the training data.*

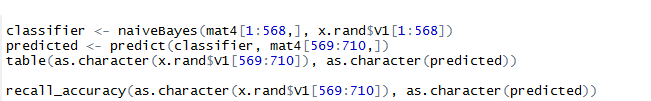
*The results are then compared to give a final accuracy number. This has the benefit of using all of the data to train the model, but is computationally more expensive.*

*For now, lets stick to the 80-20 split. In our case, that is 568 for training, 142 for test.*

*We will need to reprocess the data and randomize it to make sure we get a good split:*



 Now using naive Bayes model.





*As we can see by the confusion matrix and the recall accuracy, this is not a great model.*

*In a confusion matrix, the rows represent the actual group of the data, while the columns represent the predicted group of the data.*

*We can see that this model classified many of the documents as money-f regardless of their actual group.*

*To quantify the results another way, the recall accuracy tells us: of the documents that were truly in a given class how many were correctly labeled in that class by the algorithm.*

*Precision (not shown here for simplicity) is another interesting measure that shows us: of the documents that were labeled in a given class, how many were correctly placed there.*

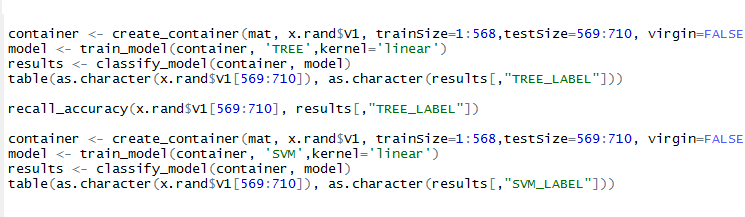
*A third measure, F1, combines precision and recall.*

*A major benefit of supervised analysis is that there are many different machine learning algorithms that can be applied.*

*Lets try a Tree next.*

*RTextTools allows us to create a container with our training and test data already defined.*

*We can then call this object with train\_model and classify\_model.*



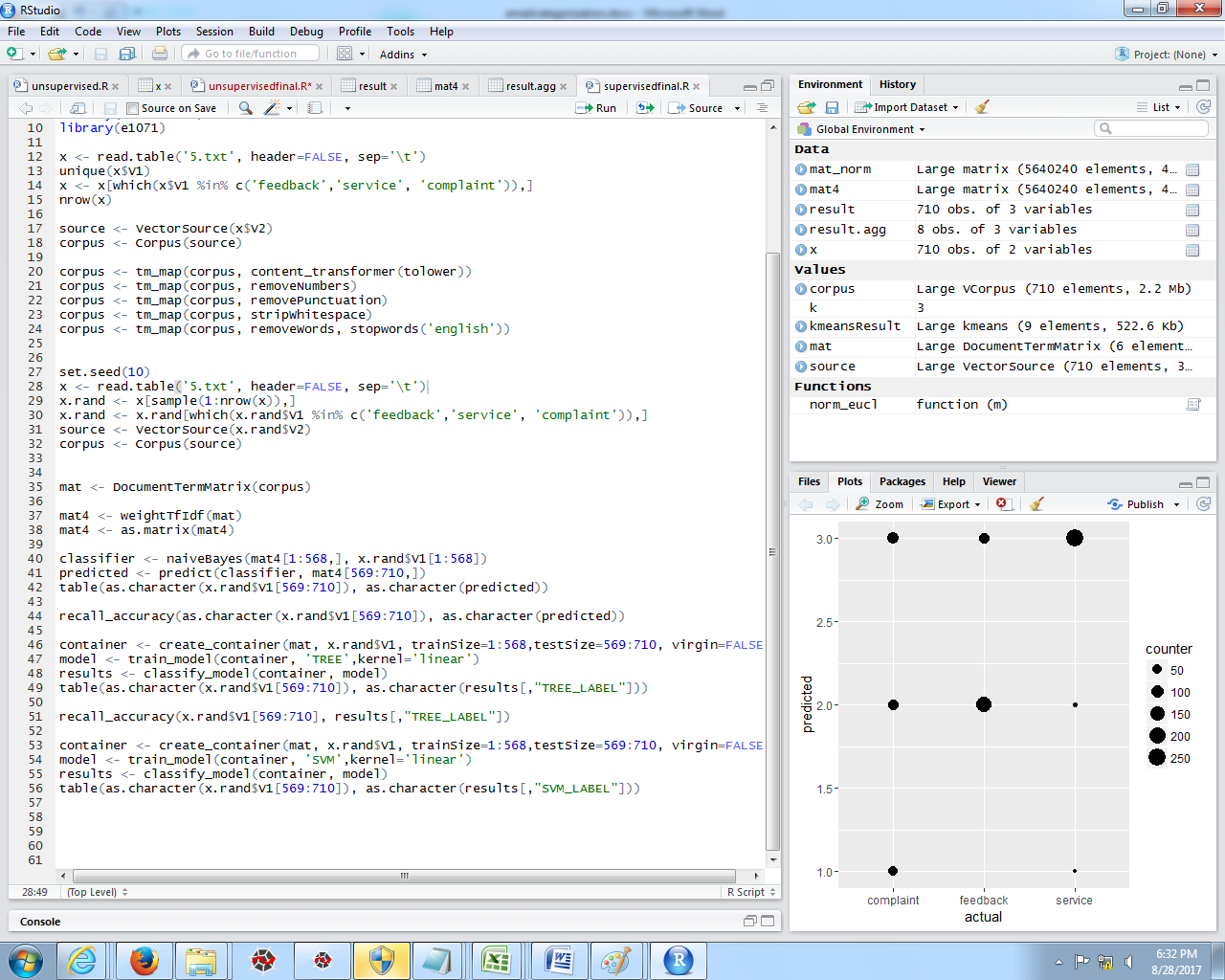
*The results here look much better, with a recall accuracy of 92%.*

*We also try Support Vector Machines.*

*SVM tends to work well with text mining classification and it also happens to be very fast.*

*It seems that SVM is the best model here. The recall accuracy was a substantial 97%.*

*If our data is representative, we would expect that we could classify documents of these three classes with 97% accuracy.*

****

*Figure 20.Full code*

**SOFTWARE REQUIREMENTS**

1. R language software
2. R studio