Sentiment Analysis

M. Martínez

November 25, 2021

Abstract

I use four different machine learning (ML) models (k-Nearest Neighbors, C5.0, random forest, and support vector machine) to predict customer sentiments on Samsung galaxy and iPhone smartphones. The train and test data is a small matrix data set ($\sim$ 3000 observations) for Samsung galaxy. Exploring several features selections (cross-correlation, feature variance, and recursive feature elimination), I find that the best results are when using the data set obtained from applied a recursive feature elimination (called $galaxyRFE$) and the C5.0 model (Accuracy=0.77 and Kappa=0.53). I built a new data set for the iPhone from the public Common Crawl database through Amazon Web Services that contains 30000 observations. This data set is used with the best-trained model C5.0 to predict the customers' sentiments on the iPhone. I find that, in general, customers have a very positive sentiment on the Samsung galaxy (78%), while a not-negligible minority ($\sim$ 10%) have a very negative sentiment. In the case of the iPhone, I find more polarized results, where 68% of customers have a very negative sentiment and 23% a very positive one. Based on these results, I suggest keeping both smartphones.

1 Introduction

Sentiment analysis or opinion mining is a natural language processing (NLP) technique used to identify (and quantify) subjective data information as positive, negative, or neutral. NLP is a powerful technique that allows monitoring brand and product sentiment in customer feedback to understand their needs.

Helio, a smartphone and tablet app developer, makes medical apps for aid workers in developing countries. These apps will allow aid workers to manage local health conditions. A government agency supporting this project requested that the app suite be bundled with one model of smartphone. Therefore, Helio is in the process of evaluating potential handset models to determine which one to bundle their software with.

In this project, we will perform a sentiment analysis that helps to know customers' preferences between Samsung galaxy and iPhone. Our general approach includes counting words, associated with customers' sentiment on these devices, within relevant documents on the web. Then, we will leverage this data and machine learning models (ML) to look for patterns in the documents that enable us to label each of these documents with a value that represents the level of positive or negative sentiment on of these devices. Specifically, we will be analyzing sentiment data for Samsung galaxy and iPhone.

2 Data

2.1 Train and test data: small matrix-galaxy

Helio created a short list of five devices that are all capable of executing the app suite's functions. The data is called small matrix data-set and is structured as follow:

- columns A-E give information about the relevancy of the webpage toward each device.
- columns F-G give information about the sentiment toward the operating system used on the phone.
- columns BA-BF list the number of positive, negative, and neutral words about the operating systems.
- columns H-V give information about the sentiment toward a phone's camera.
- columns W-AK give information about the sentiment toward a phone's display.

The attributes in columns A-E answer the question, "Does this webpage express meaningful sentiment about a device we care about?". Attributes that collect information about the sentiment toward a phone's performance are in columns AL-BF. In the first set of columns (AL-AZ), the

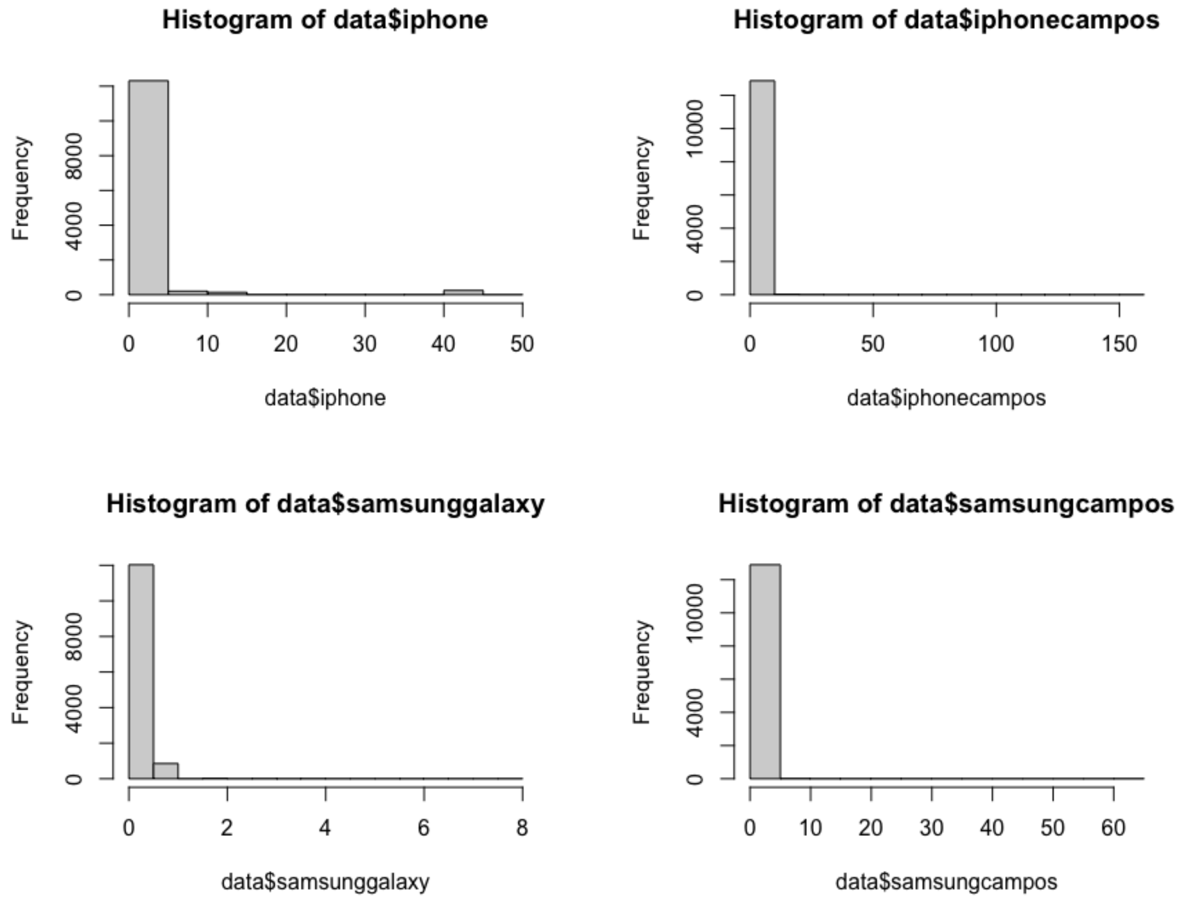

Figure 1: **Features**. **Left panels**: show the number of times the term iPhone (top) and the term Samsung galaxy (bottom) appears in the document. **Right panels**: number of positive words or expressions that are present near the term camera or its synonyms for iPhone (top) and Samsung galaxy (bottom).

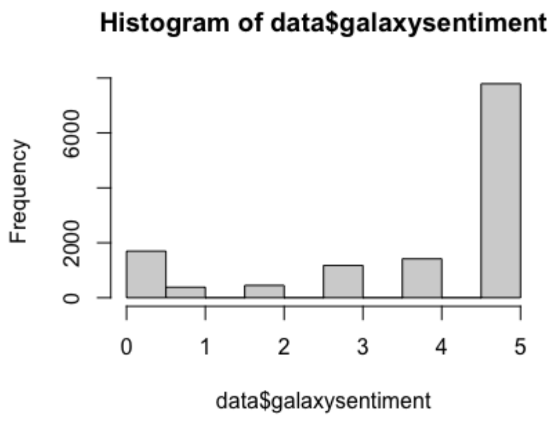

Figure 2: **Dependent variable for Samsung datad-set**.

information is about the sentiment towards a phone's hardware performance. These columns record the number of positive, negative, and neutral words within reasonable proximity to the word performance or alternative words for performance. The next set of columns (BA-BF) is the number of positive, negative, and neutral words in reasonable proximity to words related to the operating system (e.g., iOS, Android operating system, etc.). We use the galaxy data set for training and testing the ML models (file: *galaxy_smallmatrix_labeled_*9d.csv). Note that the last column BG (called *galaxysentiment*) is the dependent variable, which represents the overall sentiment toward the device on a scale of 0-5 as follow:

- 0: very negative
- 1: negative
- 2: somewhat negative
- 3: somewhat positive
- 4: positive
- 5: very positive

2.2 New data: large matrix-iphone

We extend the analysis by building a larger data set for the iPhone brand to constrain the customer sentiment. Using the cloud computing platform from Amazon Web Services (AWS) we analyze the data set from Common Crawl, which is an open repository of web crawl data (over 5 billion pages so far) stored on Amazon's Public Data Sets.

Using some scripts (developed by the team), we built a new data set called large-matrix. One of the scripts is called "mapper", it examines and counts data from portions of the Common Crawl data. Another is the scripts "reducer", which accumulates the analysis from the individual mapper jobs. The "aggregation" script helps stitch together the raw output from the multiple job flows.

Running the scripts for 250 Common Crawl archive files, I collected 35438 web pages. Then I create the large matrix with a structure like the small matrix. This data set will be used to validate the best-trained ML model. Note that in this case, the data is for the iPhone brand and that the last column is called *iphonesentiment*.

3 ML models for classification

In this section I briebly explain the technical details on the four ML models that we use. The algorithm is writen in the programing language R and are part of the CARET library; k-Nearest Neighbors (KKNN), C5.0, random forest (RF), and support vector machine (SVM). In the following a brief description is giveng only for these three models.

Some algorithms can be based on classification trees and/or ruled-based models. An example of a classification tree might be,

```
if    Predictor B >=0    then
||      if    Predictor A >=-100    then  Class =1
||       else   Class =2
else   Class =2
```

In the previous example, the two-dimensional predictor has been divided into three regions (also called nodes), and each region has been categorized according to two classes (Class 1 and Class 2). This nested IF-THEN statements can be written as rules, in the following way,

if **Predictor A**>=-100 and **Predictor B**>=0 then **Class**=1
if **Predictor A**>=-100 and **Predictor B**<0 then **Class**=2
if **Predictor A**<-100 then **Class**=2.

All algorithms are based on these classifications. However, their differences lie in the process of finding the optimal splits and rules, which depend on the optimization criteria assumed.

3.1 C5.0

This algorithm is based on trees and rules classifications. In the former case, the algorithm combines non occurring conditions for splits with several categories, resulting in a final global pruning procedure that removes the sub-trees based. In the latter, a first tree is built (based on rules), then the rules are simplified considering the most likely class. In CARET the method C5.0 works only for classification and allow the tunning of three parameters; the number of boosting iterations (*trials*), the model type (*model*: rules or tree), and *winnow*, which gives the feature selection approach. For example, predictors are considered unimportant if they are not in any split in the winnowing tree.

3.2 Random forest, RF

In this algorithm, each tree in the forest picks a vote for new sample classification. Then, the proportion of votes in each class are used to build the predicted probability vector. This algorithm has a single tunning parameter called *mtry*, which is the number of randomly selected predictors within each split.

3.3 k-Nearest Neighbors, kknn

This algorithm assume that elements are similar in close distance. The distances are determined based on different approximations. For example, the Euclidian distance, which measure the straight-lenght of a line between two points (or samples). Indeed, this is the most recomended approach when there is a high density data. The parameter k represent the number of neighbors and range from 1 to 20. Note that, small values of K tend to over-fit, while large values underfit. One way to improve the prediction is to weight neighbors' contribution to the prediction of a new sample based on their distance to the new sample. This means that closer samples will contribute more to the predicted response than those that are farther away. The other two parameters in this model are the distance and the kernel. The kernel is the approach that measure the distance.

3.4 Support Vector Machine, svm

This method is based on a metric called the *margin*, which measure the distance between the classification boundary and the closest training set data. In this method, chosing an appropiate kernel funtion (σ value) and cost value is important in order to avoid over-fitting the training data. Note that underfitted data will have low values of the *Cost* (C) parameter , contary , relatively high value of C will produce over-fitted data.

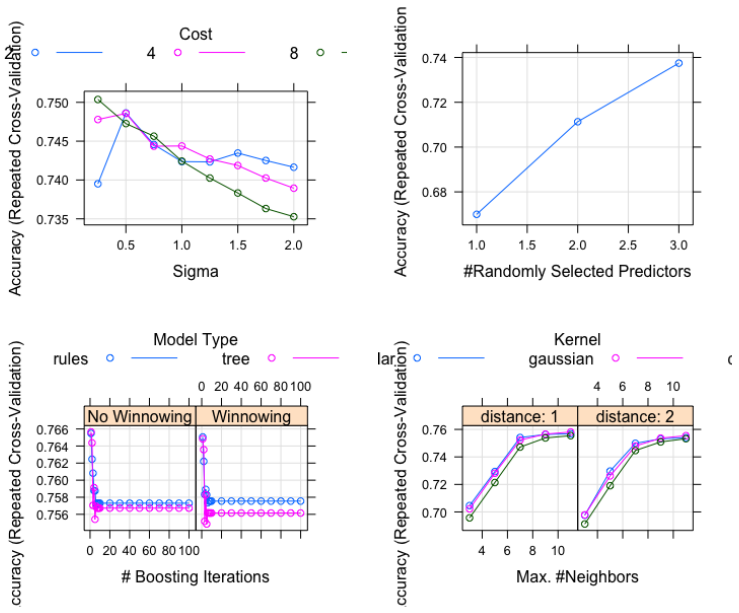

Figure 3: **Accuracy for all models using the RFE date-set.** The panel are for SVM (left-top), RF (left-bottom), C5.0 (right-top), and kknn (right-bottom)

4 Analysis

In order to look for the best combination of features and ML model we explore three methods for the feature selection and four ML models. For each feature selection method I build a new data-set. Then I use all them to train the four ML models and select the best one.

4.1 Featuring

For the **first method** we build the correlation matrix and look for those features highly correlated with the dependent variable. I found that none of the features show some correlation. Therefore, we use all the features and namely the new data set *data_all*. For the **second method** we explore the feature varianza, since features with very little or "near zero variance", may or may not have useful information. For this case the data-set is namely $galaxyNZV$. For the **third method** we follow a recursive feature elimination. We use the RFE() function from the R library CARET, which use a random forest approach to try all combination of feature subsets and return a final list of recommended features. The new data set is namely $galaxyRFE$.

Model	Grid of parameters
SVM	C = $2^{(1:3)}$, sigma = seq(0.25, 2, length = 8)
RF	*mtry*=3
C5	trials=40, model=rules, winnow=false
KKNN	kmax = c(3, 5, 7 ,9, 11), distance = c(1, 2), kernel = c(”rectangular”, ”gaussian”, ”cos”)

Table 1: Range of model parameters.

Model	Accuracy mean	Kappa mean	Best fitted parameters	Resampling Accuracy (min, max)	Resampling Kappa (min, max)
			Data_all		
SVM	0.75	0.51	C = 4, sigma = 0.25	0.72, 0.77	0.44, 0.56
RF	0.71	0.39	*mtry*=3	0.69, 0.74	0.33, 0.45
C5	0.76	0.53	trials=1, model=rules, winnow=true	0.74, 0.79	0.47, 0.60
KKNN	0.76	0.52	kmax=11, distance=1, kernel=gaussian	0.73, 0.78	0.46, 0.57
			galaxyRFE		
SVM	0.75	0.51	C = 8, sigma = 0.25	0.72, 0.79	0.44, 0.59
RF	0.74	0.45	*mtry*=3	0.71, 0.77	0.39, 0.53
C5	**0.77**	**0.53**	trials=1, model=tree, winnow=false	0.75, 0.78	0.49, 0.58
KKNN	0.76	0.52	kmax=11, distance=1, kernel=gaussian	0.73, 0.78	0.45, 0.57
			galaxyNZV		
SVM	0.74	0.47	C = 2, sigma = 1.75	0.71, 0.77	0.40, 0.55
RF	0.75	0.50	*mtry*= 3	0.73, 0.79	0.45, 0.58
C5	0.75	0.49	trials=1, model=rules, winnow=false	0.72, 0.77	0.43, 0.55
KKNN	0.75	0.49	kmax=11, distance=1, kernel=gaussian	0.72, 0.78	0.41, 0.56

Table 2: Metrics and best fitted parameters.

Reference	0	1	2	3	4	5
Prediction						
0	360	1	1	3	6	32
1	0	0	0	0	0	1
2	0	0	15	0	3	2
3	2	2	0	211	2	28
4	4	0	1	5	143	11
5	135	105	121	134	281	2264

Table 3: Confusion Matrix for the best model.

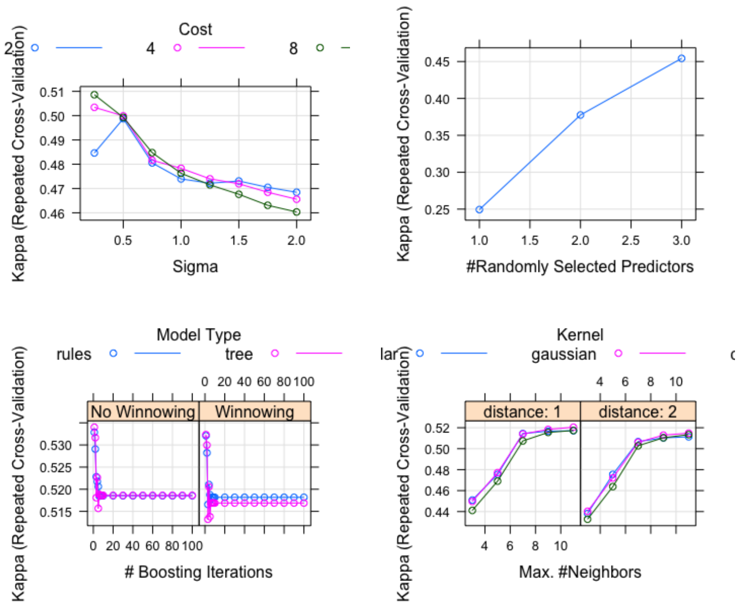

Figure 4: **Kappa metric for all ML models using the RFE date set**. The panel are for SVM (left-top), RF (left-bottom), C5.0 (right-top), and kknn (right-bottom). Note that a larger Kappa indicates a better prediction of the classes.

5 Training and testing models

Bootstrapping each data set, we define k-fold cross-validation with ten groups and ten repetitions. Then, we defined the grids of parameters to be tuned for each model, see Table 1. In Table 2 we list the metrics and values obtained from the best-fitted parameters for each data set.

Using the data set *Data_all* I find that accuracy range from 71% to 76% and Kappa from 0.39 to 0.53, using the data set *galaxyRFE* the accuracy range from 74% to 77% and Kappa from 0.45 to 0.53, and using the data set *galaxyNZV* the accuracy range from 74% to 75% and Kappa from 0.47 to 0.50. In general, the results from using the *galaxyRFE* are among the best, for all models, see Figures 3 and 4. However, C5.0 model gives the highest accuarcy= 77% and Kappa=0.53. The tunned parameters for this model are $trials = 1$, $model = tree$, and $winnow = false$. Based on these results, the C5.0 model trained with the *galaxyRFE* is the best option to do predictions on other datasets.

The data set *testSet_RFE* is used to test all models. However, here I present the results from testing only the best model. Table 3 show the confusion matrix obtained by comparing the dependent variable *galaxysentiment* and the predicted variable *fitted*. As mentioned above, there are six classes that indicate the sentiment, 0 (very negative), 1 (negative), 2 (somewhat negative), 3 (somewhat positive), 4 (positive), 5 (very positive). Classes 0 and 5 have the highest sensitivities (0.72 and 0.97) and detection rates (0.09, 0.58). Class 1 is unpredicted by the model, while intermediate classes 2, 3, and 4 are marginally predicted. Figure 5 shows the prediction of the C5.0 model, which can be interpreted as most customers (78%) tend to have a very positive sentiment on the Samsung galaxy smartphone. Although, a not-negligible fraction (10%) could also manifest a very negative sentiment.

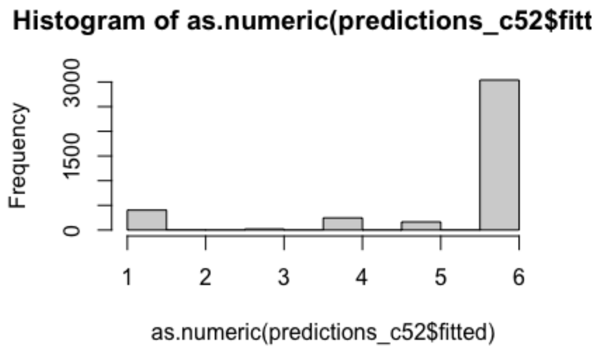

Figure 5: **Predictions for RFE data-set (Sansumg galaxy data) and C5.0 model**.

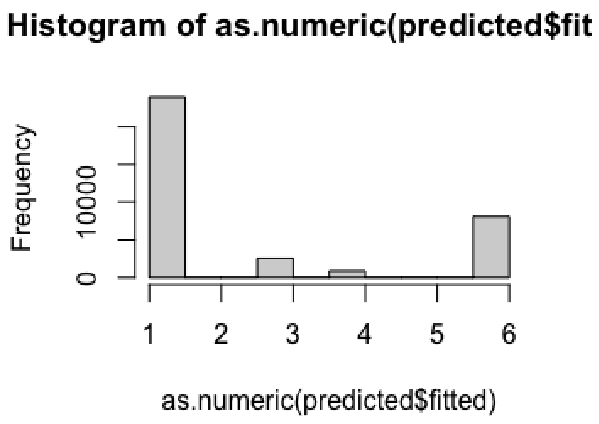

Figure 6: **Predictions for large matrix data set (iPhone data) and C5.0 model**.

5.1 Prediction with the model C5.0: large-matrix data set

Using the large matrix data set for iPhone data and the best-trained model (C5.0, $c5Fit2$ in the R file), I look for the prediction of the iPhone sentiment. Figure 6 shows the predictions, from the plot, it is possible to see some polarization between customers' sentiments. Somes have a very negative sentiment (68%) and other a very positive one (23%).

5.2 Recomendations

We find that Samsung galaxy customers generally tend to have a very positive sentiment. While, the scenario is much more polarized for iPhone. Therefore, considering these result, I highlight the following recommendations:

- keep both smart phone brands,
- builds a large matrix data-set for Samsung galaxy and make new predictions and,
- simplify the sentiment classification. I recommend only three cases; negative, positive, and neutral.

References

[1] https://en.wikipedia.org/wiki/Sentiment_analysis

[2] https://monkeylearn.com/sentiment-analysis/

[3] Max Kuhn, The CARET Package, 2019, http://topepo.github.io/caret/index.html

[4] https://towardsdatascience.com