**Online**



**Group 22**: William Leonard, Susan Mikhail, Matthias Ronnau, Lokesh Nandni Sood, Parsa Tahmasebi

Table of Contents

[1. Introduction 2](#_Toc58435935)

[1.1 Executive summary 2](#_Toc58435936)

[1.2 Business Idea 3](#_Toc58435937)

[2. Data 3](#_Toc58435938)

[2.1 Data Overview 3](#_Toc58435939)

[2.2 Exploratory Data Analysis 3](#_Toc58435940)

[2.3 Data Processing 6](#_Toc58435941)

[3. Analysis 6](#_Toc58435942)

[3.1 Naïve Bayes 7](#_Toc58435943)

[3.2 Random Forest 8](#_Toc58435944)

[3.3 K-Nearest Neighbors (KNN) 8](#_Toc58435945)

[4. Results 9](#_Toc58435946)

[4.1 Inferences and Key Takeaways 9](#_Toc58435947)

[5. Conclusion 11](#_Toc58435948)

[Appendices 12](#_Toc58435949)

[Appendix I: Heatmap for Correlation 12](#_Toc58435950)

[Appendix II: Distribution of the Number of Shares 13](#_Toc58435951)

[Appendix III: Count and Percentage of “High” vs “Low” Popularity Across Data Channels 14](#_Toc58435952)

[Appendix IV: Mutual Information Table 15](#_Toc58435953)

[Naïve Bayes 15](#_Toc58435954)

[Random Forest 15](#_Toc58435955)

[K-Nearest Neighbors 15](#_Toc58435956)

[Appendix V: Mutual Information Table 16](#_Toc58435957)

# 1. Introduction

## 1.1 Executive summary

The growth of the internet and technology has contributed to the popularity of online news articles and blogs. While print news is not completely dead yet, a growing number of people prefer to search the web for the day’s happenings, as online news is free, immediate, and convenient. In fact, in 2018 Pew Research noted that just over 50% of Americans get their news from some online format.[[1]](#footnote-2) According to Forbes, the number had increased to 55% in 2019.[[2]](#footnote-3)

Our team used data on Mashable articles from a two-year period to build a few machine learning models that predict the popularity of an article given a set of features about that article, such as the number of words in the article, the day of the week the article was published on, and the average sentiment polarity of the article content. In order to run these models, we discretized the continuous prediction variable “shares” into two categories: “High” popularity for those articles with a number of shares greater than or equal to the median number of shares for all articles, and “Low” popularity for those articles with a number of shares less than the median number of shares for all articles.

In order to predict the popularity of a given article, we built three machine learning models using Naïve Bayes, Random Forest, and K-Nearest Neighbors (KNN) algorithms. Overall, the accuracies we received from all three models were low and were unable to be improved much with the various data processing methods that we utilized. In fact, different method of data processing actually reduced the accuracies for various models. However, we found that implementing a Random Forest model while using supervised discretization (while not “looking” at the test set) gave us the best results at 64.91% overall.

## 1.2 Business Idea

We sought to understand the different variables that contribute to the popularity of online articles. To create hit news pieces that receive many shares, it is beneficial for publishers to understand what qualities tend to make their pieces more popular. To increase the number of shares these articles receive, and thus boost their company’s revenue, publishers could focus in on “nailing” the important factors that tend to increase shares. Additionally, they could focus on reducing the number of articles that contain features that lead to low shares.

# 2. Data

## 2.1 Data Overview

We used the “Online News Popularity” dataset on Mashable articles from the UCI Machine Learning Repository, found here: <https://archive.ics.uci.edu/ml/datasets/online+news+popularity>. The data contains articles from a two-year period.

In total, the original dataset contains 61 variables: 58 are used for prediction, 2 are used for identification, and 1 is the target variable (“shares”). Some of the variables are quantitative, like the number of words in the title, the number of images, and the number of videos, while others are binary categorical variables, such as indicators for the day of the week the article was published or the data channel the article is part of.

## 2.2 Exploratory Data Analysis

We began our exploratory analysis by formatting the columns of the dataframe into a usable format (remove the space that preceded each column name), as well as removing the two identification variables from the data (“id” and “timedelta”), as these would not be helpful in our analysis or in building our models.

Next, we investigated the correlation amongst the variables, so as to see if there were any that we could remove later when we were building and assessing our models (See Appendix I). The variables “n\_non\_stop\_words" (the rate of non-stop words in the article), “n\_non\_stop\_unique\_tokens” (the rate of unique non-stop words in the article), and "kw\_avg\_min" (the keyword of the article that generated the lowest amount of shares) exhibited very high correlation with other variables, so we decided to remove these later on when we ran our models with and without data processing. Strangely, no variable exhibited a strong correlation with the number of shares an article received, and this likely contributed to the poor performance that our models exhibited.

Next, we examined the distribution of the dependent variable, “shares” (See Appendix II). Based on our histogram, we can see that the distribution of the number of shares for each article does not appear to follow a normal distribution; we found its skewness to be 34.95 and its kurtosis to be 1909.98.

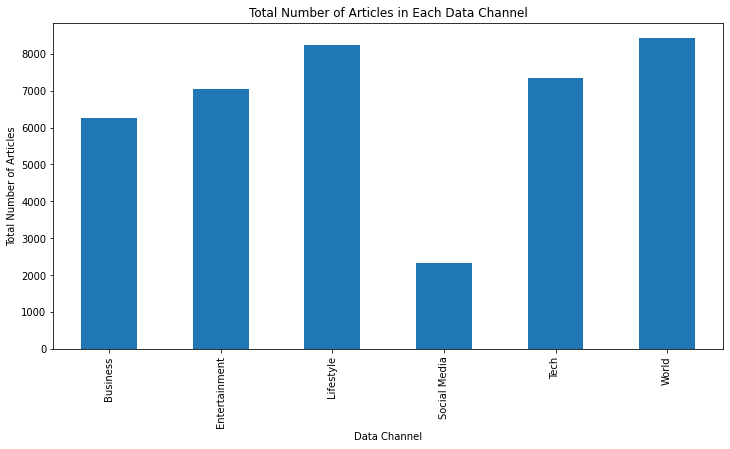


Figure : The total number of articles in each data channel. Excluding social media articles, the distribution is roughly uniform.

Because of this heavy skew, we opted to remove articles with an extremely low number of shares as well as articles with an extremely high number of shares later on when we analyzed different processing methods.

Ignoring social media articles, the distribution of the number of articles in each data channel is roughly uniform. There were significantly fewer social media articles, but these articles showed the highest percentage of popularity amongst all data channels; 71.42% of articles in this category had a “High” number of shares (See Appendix III).

World news performed the worst, with 65.16% of world news articles having a “Low” number of shares. We found that articles published on weekdays on average received the greatest number of shares, with articles published in the middle of the week (Tuesday-Thursday) performing the best.

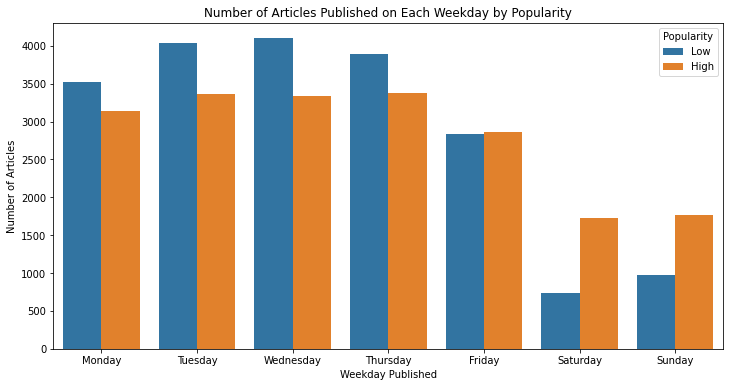


Figure 1: A bar graph illustrating the number of articles published on each weekday, broken up by their popularity. Articles published midweek seemed to do the best.

## 2.3 Data Processing

In total, we used six different methods to process our data to try and improve our model results: five different techniques individually and then a combined run. As stated above, for our first method we tried removing the correlated variables “n\_non\_stop\_words”, “n\_non\_stop\_unique\_tokens”, and “kw\_avg\_min” to see if the accuracy of the models would improve without these redundant variables. For our second method, we discovered that some of the articles in the data had a value of “n\_tokens\_content” equal to 0, which indicated that there were no words in the article. It did not make sense to us to use these values to try and predict the number of shares that an article will receive, as an underlying assumption we made is that an individual will read an article and then share it based on how they felt about the content. We then decided to run the models without these rows present to see if we could generate better predictions. In our third method, we decided to exclude data points with “shares” greater than 2500. As shown above, the distribution of “shares” was heavily skewed right, so we wanted to see if removing some of these “extreme” values would improve the models’ accuracy. We used the mutual information of the independent variables and the article popularity to determine the most “relevant” attributes for our fourth method (See Appendix IV for our list of features). We then used supervised discretization on the training set to bin some of the continuous variables that existed in our data. Finally, we combined our first four methods for an “omnibus” approach to try and improve the model accuracy.

# 3. Analysis

For both the Naïve Bayes and Random Forest models that we built, we tested each using a training and a test set (with 67% of the data in the training set and 33% in the test set), as well as 10-fold cross validation. For each K Nearest Neighbors classifier that we ran, we scaled the data so that it had a mean of 0 and a standard deviation of 1, as each variable is originally on a different scale with different units, and we did not want some variable to over-influence the KNN results. We then plotted elbow plots for values of k between 1 and 25 to choose the optimal number of nearest neighbors.

## 3.1 Naïve Bayes

We decided to use a Naïve Bayes classifier as our “baseline” model to compare the other two models to as it is relatively easy to implement and very quick to run.[[3]](#footnote-4) Naïve Bayes works by calculating the conditional probability of a specific output class given input data; it utilizes Bayes’ Theorem:

The results of our different data processing methods on our Naïve Bayes model are listed in the table below.

|  |  |  |
| --- | --- | --- |
| **Naïve Bayes** | | |
| **Data Processing Method** | **Accuracy** | **Change From Benchmark** |
| *Baseline (None)* | 60.40% | NA |
| *Remove Correlated Columns* | 60.40% | 0 |
| *Remove Articles With No Words* | 60.77% | + 0.37 |
| *Remove Articles With Extreme Amount of Shares* | 61.18% | + 0.78 |
| *Feature Selection Using Mutual Information* | 55.53% | - 4.87 |
| *Supervised Discretization* | 64.73% | + 4.33 |
| *Omnibus* | 56.47% | - 3.93 |

## 3.2 Random Forest

The second model that we decided to implement was a Random Forest classification. A Random Forest is an ensemble learning method for classification or regression, and it is fairly easy to implement and gives fairly decent results a lot of the time.[[4]](#footnote-5) Random Forests work by implementing feature bagging, where a randomly selected subset of the entire attributes are selected and a decision tree is built based on this random subset. Multiple decision trees are made with different random subsets of the features, hence the name *Random Forest*.[[5]](#footnote-6) The results of our different data processing methods on our Random Forest model are listed in the table below.

|  |  |  |
| --- | --- | --- |
| **Random Forest** | | |
| **Data Processing Method** | **Accuracy** | **Change From Benchmark** |
| *Baseline (None)* | 63.92% | NA |
| *Remove Correlated Columns* | 63.92% | 0 |
| *Remove Articles With No Words* | 64.04% | + 0.12 |
| *Remove Articles With Extreme Amount of Shares* | 61.76% | - 2.16 |
| *Feature Selection Using Mutual Information* | 61.77% | - 2.15 |
| *Supervised Discretization* | 64.91% | 0.99 |
| *Omnibus* | 60.27% | - 3.65 |

## 3.3 K-Nearest Neighbors (KNN)

For our final model, we decided to implement a K-Nearest Neighbors algorithm. K-Nearest Neighbors (KNN) is a classification/prediction algorithm that classifies instances based on the distance between a known data point and a point that it is trying to predict. By specifying the number of neighbors, “k”, the model will calculate the “k” closest neighbors to the new point using a given distance metric and decide the class of the unknown point by majority vote. A downside to implementing KNN, which we encountered, is the increase in runtime as more data is included. It takes quite a while to iterate through various values of “k” and calculate the error rate for each value of “k” so that an elbow plot can be generated so as to choose the optimal “k” to run the final model on. The results from our various data processing methods on our KNN model are listed in the table below.

|  |  |  |
| --- | --- | --- |
| **K-Nearest Neighbors** | | |
| **Data Processing Method** | **Accuracy** | **Change From Benchmark** |
| *Baseline (None)* | 62.13% | NA |
| *Remove Correlated Columns* | 62.13% | 0 |
| *Remove Articles With No Words* | 62.14% | + 0.01 |
| *Remove Articles With Extreme Amount of Shares* | 59.15% | - 2.98 |
| *Feature Selection Using Mutual Information* | 61.42% | - 0.71% |
| *Supervised Discretization* | 62.55% | + 0.42 |
| *Omnibus* | 57.84% | -4.29 |

# 4. Results

## 4.1 Inferences and Key Takeaways

Overall, our model prediction accuracies hovered between 60% and 65% accuracy, despite numerous data processing methods. One possibility for this is that the data was very “processed” when we received it. All categorical variables had been converted into 0-1 binary variables, and there were no missing values. Perhaps with the original, unstructured data, initial models would have given worse results, and after transforming it into the format that we received it, we would have observed the results that we did. Unfortunately, we are not able to test this hypothesis.

Overall, the Random Forest model performed the best of the three models we built. K-Nearest Neighbors performed second best overall, and Naïve Bayes was the worst. Interestingly, some data processing methods that we used caused our models to perform worse, in some cases *very* worse (Using Naïve Bayes with feature selection based on mutual information dropped the accuracy by almost 5 points).

The best results for all models came about with supervised discretization (See Appendix IV for our confusion matrices generated in WEKA for each model using supervised discretization). On our Naïve Bayes model, the overall accuracy was 64.73%; there was a stratified accuracy of 62.35% for classifying articles with “low” popularity, and a stratified accuracy of 67.17% for classifying articles with “high” popularity. For our Random Forest Classifier, supervised discretization gave an overall accuracy of 64.91% (this was the best result for any model across all data processing methods); there was a stratified accuracy of 65.87% for classifying articles of “low” popularity, and a stratified accuracy of 63.91% for classifying articles of “high” popularity. Finally, for K-Nearest Neighbors, supervised discretization gave an overall accuracy of 62.55%; there was a stratified accuracy of 70.57% for classifying “low” popularity articles, but a stratified accuracy of only 54.31% when classifying “high” popularity articles.

If there was a high cost associated with writing articles, we would recommend running a K-Nearest Neighbors model with supervised discretization, as this model was the most accurate at predicting articles with “low” popularity. This would prevent spending time, energy, and money on writing articles that are not destined to receive a lot of shares and generate revenue for the company.

Our mutual information table that we constructed (See Appendix V) shows that no variable has a very large amount of mutual information with our dependent variable. Similarly, our heatmap of correlations between variables does not show *any* variable that has a strong correlation with the number of shares. This could be due to the above hypothesis about a cleaned dataset, or it could simply be due to the fact that none of the predictors give much information about the number of shares a given article receives.

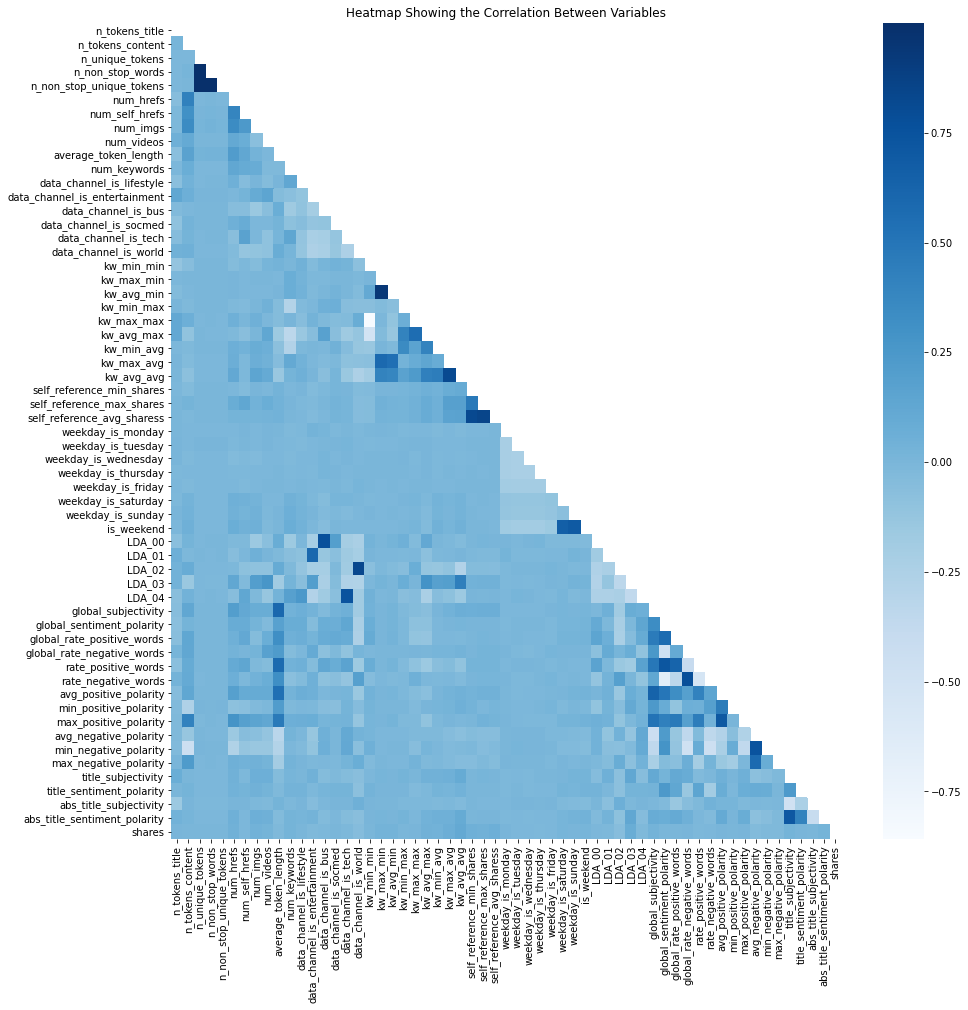
Although we do not know what these “topics” are, we would suggest that Mashable focus on publishing articles that fall into LDA Topic 2 and Topic 1, as these are two of the top three variables with largest mutual information with article popularity, and is something that Mashable has control over adjusting. Additionally, Mashable could focus on the features we analyzed while exploring the data, namely social media articles and publishing during the middle of the week.

# 5. Conclusion

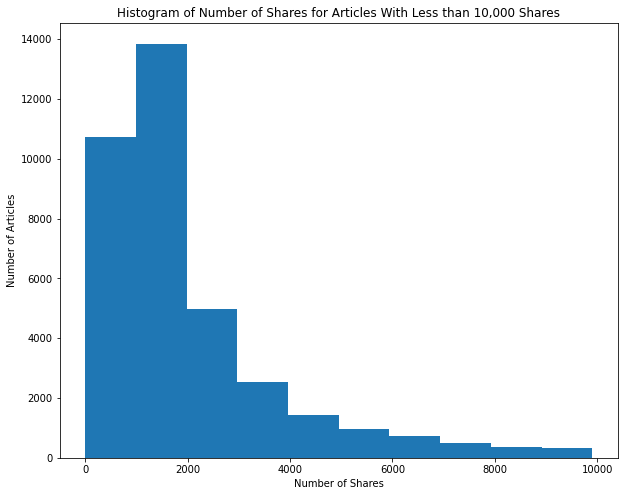
For our term project for BANA-273, Machine Learning for Analytics, we sought to apply what we learned in class this quarter to build a handful of predictive models that could be used to predict the popularity of online news articles. We used data on Mashable articles over a two-year period to build, train, and evaluate Naïve Bayes, Random Forest, and K-Nearest Neighbor models. Despite employing multiple methods to improve our model accuracy, the baseline models more or less plateaued in performance, and very little improvement was garnered. Nonetheless, we were able to achieve accuracies just under 70% while demonstrating the techniques we learned in class.

# Appendices

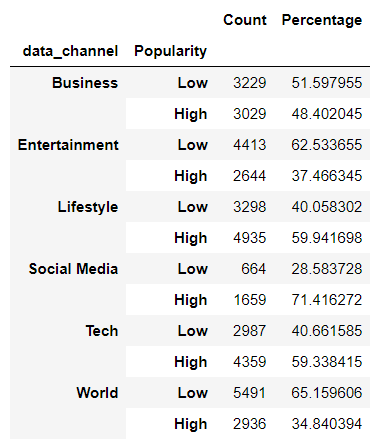
## Appendix I: Heatmap for Correlation



## Appendix II: Distribution of the Number of Shares

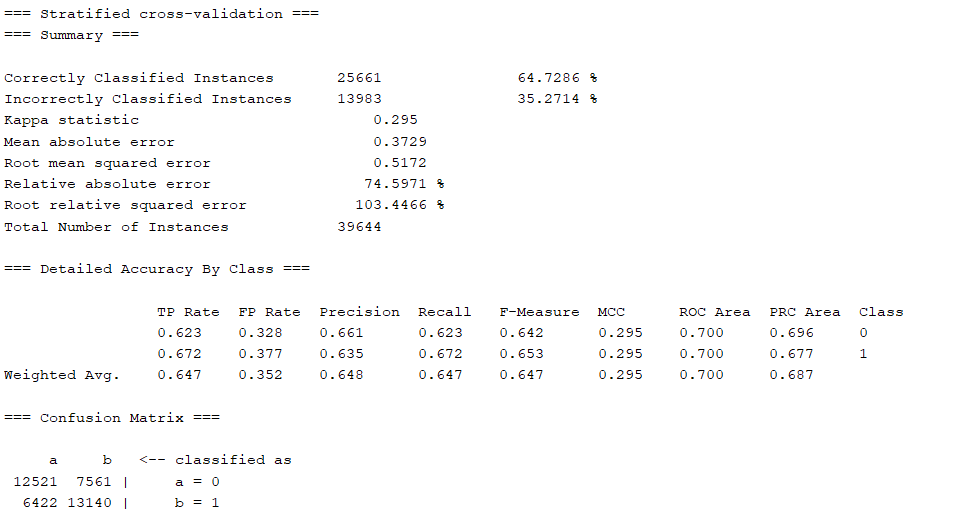


## Appendix III: Count and Percentage of “High” vs “Low” Popularity Across Data Channels

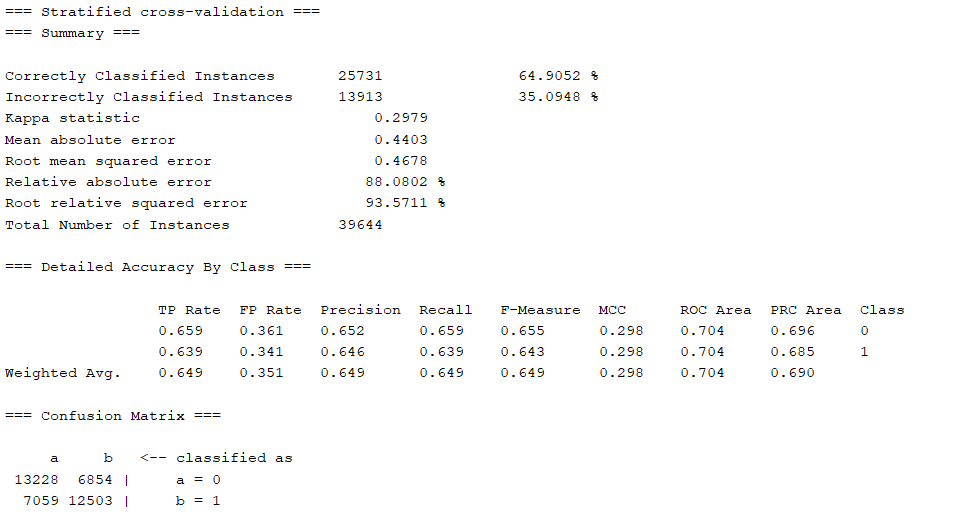


## Appendix IV: Mutual Information Table

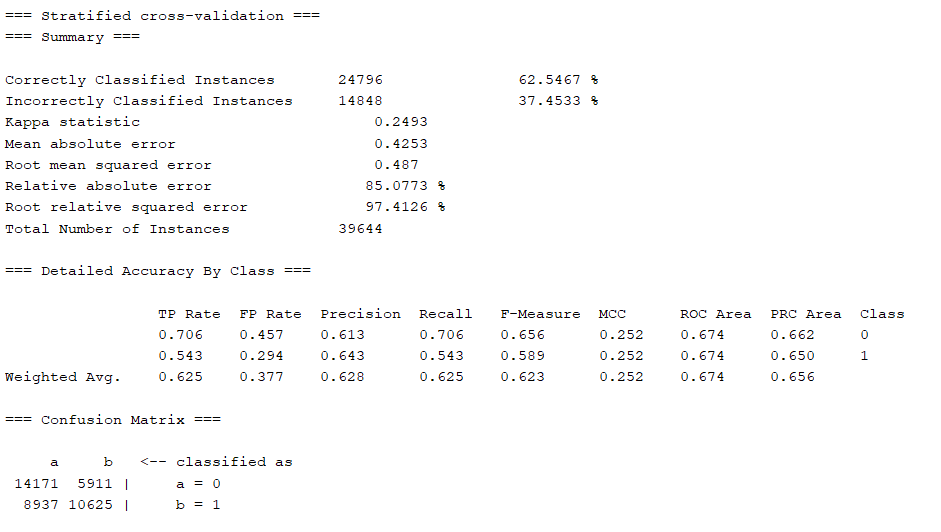
### Naïve Bayes



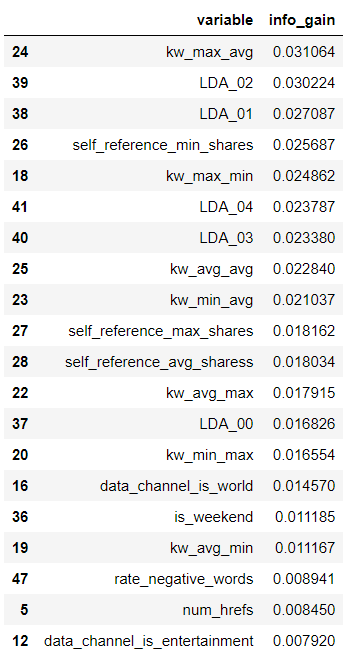
### Random Forest



### K-Nearest Neighbors



## Appendix V: Mutual Information Table



1. Elisa Shearer, “Social Media Outpaces Print Newspapers in the U.S. as a News Source,” Pew Research Center (Pew Research Center, August 27, 2020), https://www.pewresearch.org/fact-tank/2018/12/10/social-media-outpaces-print-newspapers-in-the-u-s-as-a-news-source/. [↑](#footnote-ref-2)
2. Peter Suciu, “More Americans Are Getting Their News From Social Media,” Forbes (Forbes Magazine, October 11, 2019), https://www.forbes.com/sites/petersuciu/2019/10/11/more-americans-are-getting-their-news-from-social-media/?sh=1979a41c3e17. [↑](#footnote-ref-3)
3. Class Lecture, October 21, 2020. [↑](#footnote-ref-4)
4. Niklas Donges, “A Complete Guide to the Random Forest Algorithm,” Built In, accessed December 9, 2020, https://builtin.com/data-science/random-forest-algorithm. [↑](#footnote-ref-5)
5. Class Lecture, December 3, 2020. [↑](#footnote-ref-6)