INFOMDM Assignment 2

Pascal Verkade 6045057 | Idan Grady 7304447 | Marc de Fluiter 5928087

1 Introduction

People are increasingly using IT systems to connect with each other, including through Computer-Mediated-Communication (CMC) [1]. In a virtual environment, people can form connections that were previously impossible. This raises some fundamental questions regarding trust, and the role trust plays in the making of decisions in the virtual environment. To what extent do people trust strangers and their opinions? According to (Diana, 2020) 91% of 18-34 year olds trust online reviews as much as personal recommendations, and 93% of consumers say they rely on online reviews when making purchases. However, these virtual relationships are not without their faults as well. During the 2016 US presidential campaigns, researchers observed a drastic increase in unprecedented issues such as 'fake campaign news', creating psychological profiles to target certain voters, and more (Groshek and Koc-Michalska, 2017, Brianne, 2019). In light of the fact that we are using digital technologies more and more during the COVID 19 pandemic (De' et al., 2020), studying whether we can classify a true or false story is essential so that we can stop misleading stories as well as events. One huge business in which this is important is in the online review business. Whether it concerns products, services, food or hotels, reviews are often key to a consumer as 93% of the consumers say that online reviews influenced their decision (Kaemingk, 2020).

In this report, we look into hotel reviews and apply four different algorithms, to classify whether or not negative reviews are fake or genuine. The four algorithms, each based on both a unigram and a bi-gram representations are: MultinomialNB(MNB), LogisticRegression(LR), DecisionTreeClassifier(DT), RandomForestClassifier(RF). We fitted each model with 5 as well as 10 folds cross validation. This results in 12 different models which we have compared and analysed. This is done by doing hyperparameter tuning first on the training set, and then using the best model for testing and evaluation. But before going into the answer of the question of how well each algorithm can differentiate between truthful and false reviews, we will first look at the data we worked with.

[1] https://en.wikipedia.org/wiki/Computer-mediated_communication/

2 Data Preparation

The data we used is the data collected by Ott et al., 2011 and Ott et al., 2013. The data consists of a total of 1600 online reviews, of which true reviews have been obtained from several popular online review communities and fake reviews have been produced by Ott et al. using Amazon's Mechanical Turk. The full corpus contains 400 true negative reviews, 400 false negative reviews, 400 true positive reviews and 400 false positive reviews. For this analysis, we will consider the negative reviews and focus on classifying and differentiating between true negative and false negative reviews only.

2.1 Data Description and Manipulation

We first checked the list of reviews for duplicates. We found 4 reviews belonging to the true category that were duplicates. Since we want to train the classifier on completely different instances of reviews, as duplicate reviews are a rare occurence in reality and should not happen, we removed these duplicates to make sure that there was no 'bias' to (the words in) these reviews. We also randomized the full list of reviews to provide a more fair basis for training, testing and evaluating.

Before we can use the data as input, we first have to clean and transform the reviews into something usable. For this we use a bag of words representation, in which we vectorize the features (=words) of each document and create a matrix of those features/words. This vectorization is done by first tokenizing on white-spaces and punctuation, then Counting the occurences of tokens in each document and then normalizing and weighting with diminishing importance tokens that occur in the majority of samples / documents. We used scikit's basic CountVectorizor to convert the collection of text documents to a matrix of token counts. To ensure that we focus on the important vocabulary and to prevent sparsity in our matrix, we used the paramter min_df to ignore the terms that have a lower document occurence than 2. This also meant that we had to set the parameter max_df to 0.7 to limit the terms that have a frequency strictly higher than that given threshold. After we vectorized the data, we made sure to counterbalance the term-frequency with the inverse document frequencies, which is called Tf-idf term weighting. This is to make sure that very frequently occurring words aren't overshadowing the frequencies of rarer yet more interesting terms. For this we used the basic sklearn TfidfVectorizer. Since we already made sure that we only use words with higher document occurences than 2, we set the parameter smooth_idf to False. Smooth_idf normally adds 1 to the document frequencies to prevent zero, but we don't need it because of our settings of the CountVectorizer. For the bigram vectorizers we used similar settings.

After having prepared our data, we are able to use it for training and testing the four algorithms. In order to do this, we splitted the data into 5 folds. Four of which are used in training and hyper parameter tuning, one of which is used for testing. The trainingset itself is also splitted into respectively 5 and 10 folds

for the hyperparameter tuning.

3 Analysis

3.1 Hyper parameter tuning

Before an estimation and comparison can be done on the performance of the algorithms, parameters of the algorithms have to be set. In order to do this, cross fold validation is used on the training data (the original 4 folds) to train and test the models with different parameters. Below is an overview of the hyperparameters and their values that are being looked at in this process. These combined with the amount of folds (5, 10) and the text representation (unigram, unigram and bigram) formed the grid over which we performed grid search and cross validation to evaluate the performance of each combination.

3.1.1 Multinomial Naive Bayes:

For MultinomialNB, we looked at the parameter alpha and the values [0.01, 0.1, 0.25, 0.5, 1, 2.5, 10]. This alpha is the additive (smoothing) parameter for (maximum likelihood) calculations of the model. It is to make sure that all probabilities are non-zero, but the actual value does affect the outcome slightly, hence the needed tuning.

3.1.2 Logistic Regression:

For logistic regression we also looked at two parameters: Penalty, with its values [l1, l2], and C, with the values [0.01, 0.1, 0.25, 0.5, 1, 2.5, 10]. The penalty in logistic regression regularizes the model and decreases variance. l1 or lasso regression uses the absolute value inside the loss function to make sure the model can (easily) pick the important variables. L2 or ridge regression uses the square and allows to keep all variables and shrinks coefficients of correlated variables. C represents the inverse of the regulation strengths described above, smaller values specify stronger regularizations here.

3.1.3 Decision Tree Classifier:

For the Decision Tree classifier we looked at the parameters max_depth and min_sample_split and their respective valuse [None, 2, 4, 8, 16] and [2, 4, 8, 16]. The max_depth parameter marks the maximum depth of the tree. This parameter is needed to prevent overfitting, as our data is quite sparse. The min_samples_split parameter is also important because of this and it signifies the number of samples required for a node to be considered for a split.

3.1.4 Random Forest Classifier:

For the random forest classifier we look at three parameters. Two of which, namely max_depth ([None, 2, 4, 8, 16, 30]) and min_samples_split ([2, 4, 6, 8,

16, 30]), we have already seen in the single decision tree classifier, but we added a bigger value here to check if it would make a difference for random forests. The n_estimators parameter with values [10, 50, 100, 500] represents the number of trees in the forest.

4 Results

Together with the hyperparameter tuning, the different amount of folds and the two different representations of the text, we received the results shown in Table 1.

Model	Acc	Prec	Rec	F1	Folds	n-gram	Best Parameters
MNB	0.887	0.699	0.788	0.7312	5	Unigram	alpha: 0.25
LR	0.856	0.667	0.801	0.702	5	Unigram	C: 2.5, penalty: $l2$
DT	0.731	0.581	0.702	0.575	5	Unigram	max_depth: 4, min_samples_split: 2
RF	0.893	0.724	0.881	0.771	5	Unigram	max_depth: 16, min_samples_split: 8, n_estimators: 100
MNB	0.862	0.666	0.774	0.698	10	Unigram	alpha: 0.1
LR	0.861	0.667	0.801	0.702	10	Unigram	C: 2.5, penalty: $l2$
DT	0.743	0.586	0.709	0.584	10	Unigram	max_depth: *None*, min_samples_split: 2
RF	0.831	0.645	0.787	0.674	10	Unigram	max_depth: 30, min_samples_split: 30, n_estimators: 100
MNB	0.8	0.614	0.740	0.632	5	BigramUnigram	alpha: 0.1
LR	0.865	0.681	0.834	0.720	5	BigramUnigram	C: 10, penalty: $l2$
DT	0.662	0.549	0.634	0.516	5	BigramUnigram	max_depth: 8, min_samples_split: 4
RF	0.887	0.705	0.818	0.742	5	BigramUnigram	max_depth: 16, min_samples_split: 8, n_estimators: 100
MNB	0.891	0.699	0.788	0.731	10	BigramUnigram	alpha: 0.25
LR	0.862	0.681	0.834	0.720	10	BigramUnigram	C: 10, penalty: $l2$
DT	0.668	0.571	0.697	0.538	10	BigramUnigram	max_depth: 2, min_samples_split: 2
RF	0.856	0.667	0.801	0.702	10	BigramUnigram	max_depth: 16, min_samples_split: 8, n_estimators: 100

Table 1: Accuracy, Recall, Precision and F1 Score for Multinomial Naive Bayes (MNB), Logistic Regression (LR), Decision Tree (DT) and Random Forest (FR) with their best parameters on different folds and n-grams.

4.1 The Parameters

Before diving into a comparison of the algorithms and a further analysis, let's first look at the algorithms themselves, and more specifically, their tuned parameters. Firstly, note that when we look at the parameters of the algorithms in general, that there isn't an algorithm that had the same parameters across all n-gram/fold combinations. Some individual parameters are consistent, for instance the penalty parameter of logistic regression is l2 across all sets, but others, like max_depth from the decision tree classifier, aren't consistent among any subset.

4.1.1 Multinomial Naive Bayes:

For the Munltinomial Naive Bayes classifier we just had the main parameter alpha. It can be seen that this one is consistent for neither the folds, nor the n-grams. It is however important to acknowledge that the the numbers between

the values, 0.25 and 0.1 have not been used for tuning and thus the alpha might lay between these and due to small variances between the fold/n-gram tests, the alpha value might have been swayed towards 0.1 or 0.25.

4.1.2 Logistic Regression:

For the Logistic Regression classifier we had two parameters. The penalty variable, as already said above, was consistent across all sets. For the C parameter, we see the two highest values, meaning that a less strong regularisation was preferred. C was consistent with the n-grams, and where the data in which both were used required the least amount of regularisation, the unigram data could do with slightly more.

4.1.3 Decision Tree Classifier:

Looking at the parameters for the Decision Tree classifier, we see that the parameter min_sample_split is quite consistent with 2, with only the bigram/unigram version on 5 folds being slightly different with just one move up on 4. The interesting parameter here is the max_depth parameter, as this is completely different across all combinations. Reasons for this are unclear and our best guess is that due to randomization in the data and coincidental folds, the data differed enough such that the parameter was neither consistent for the amount of folds, nor for the n-grams.

4.1.4 Random Forest Classifier:

Looking at the last classifier: Random Forests, we see the two recurrent parameters from the single decision tree and the number of estimators (trees). The number of estimators is 100 and consistent across all combinations. The other two parameters behave the same, consistent across all combinations, except for the 10 folds on a unigram dataset. It is no surprise to see these parameters behave the same, as they almost balance each other out. A lower value of max_depth would prevent overfitting, while a lower value of min_samples_split might make the tree more specific on this data. And apparently these two required a balance of the high values of 30 on the 10x folded unigram data.

It is important to note that, now that we have seen the difference within the models, the changes in single parameters are not analyzed. Therefore we weren't able explain to the full extent why some parameters had different values. However, this was never the main goal, as we intended to look into the difference between the models. And to do this properly, instead selecting the same parameters for a model across the different folds and n-grams, we used the best versions of each classifiers on the different data. This means a comparison between them was more fair (and a little more complex), since each algorithm is the best version and representative for the different folds and n-grams.

4.2 Generative and Discriminative Linear Models

If we consider the linear models, we see that the multinomial Bayes model and the logistic regression model perform relatively the same. The multinomial Bayes model, the generative linear model, gives slightly more accurate results in the case of 5 folds and considering just unigrams and in the case of 10 folds, considering both unigrams and bigrams. Meanwhile the regularized logistic regression model, the discriminative linear model, performs quite a bit better in the case of 5 folds with both unigrams an bigrams. For the remaining case, considering 10 folds and only unigrams, they perform equally accurate.

Looking at the precision and the F1 score of these models, we see that these measures follow the same trend as the accuracies.

For the recall of these models we see different results compared to the other performance measures. Here we see that the recall of the regularized logistic regression model always outperforms the recall of the multinomial bayes model by a varying amount. On average the recall is approximately bigger by 0.05, which is rather significant. This means that the regularized logistic regression model predicts more items from the total set of items correct.

Thus overall the linear models do not seem to outperform each other really, but based on the recall of the models the logistic regression model has a slight edge over the multinomial naive bayes model, meaning that the logistic regression model has, based on these data, a higher preference if you would like to reduce the False Negatives. This is rather important in the context of the data, since you would not want to remove truthful reviews of customers.

4.3 Classification Trees and Random Forests

Comparing the results of the classification tree model and the random forests model we can clearly see that the random forests perform way better than the single classification tree in all cases. This is also to be expected since the random forests prevent overfitting and capture the most important main structure of the data.

We introduced random forests to see if they would outperform the linear models, as those linear models did with just a single decision tree. Because the single classification is also outperformed by the linear models and the random forests capture the true strength of the classification trees, we will compare the results of the random forests model to the linear models to see if we can achieve a performance gain by using the idea of classification trees, but now incorporated into the random forest classifier

In the case of 5 folds we see that the random forests model performs visibly better than the linear models on all performance measures, except for the recall on the case where we consider both unigrams and bigrams. Only in this case we see that the logistic regression model has a better score than the random forests model by just a little amount.

In the case of 10 folds we get almost the opposite trend. The random forests score marginally worse for almost all performance measures. Which is probably

due to the fact that the trees might not have been able to generalize well. Only for the recall measure we see that the random forests give a better performance than the multinomial naive bayes model, but even in this case the random forests model is outperformed by the logistic regression model.

Generally, except for a few cases, we see that using 10 folds gives a performance increase for the linear models (as well as the single classification tree model) compared to using 5 folds. Though when we only consider unigrams, the multinomial naive Bayes model on the contrary has a performance decrease. Meanwhile the random forests model seems to suffer from using 10 folds compared to 5 folds, because the performance of this model decreases everywhere when switching to using 10 instead of 5 folds. So the random forests seem only to be able to improve on the performance of the linear classifiers with 5 folds.

4.4 Bigram Features

Overall the addition of bigrams introduce more accurate results in the case of using 10 folds. The single classification tree is the only model that gives less accurate results when introducing bigram features in this case. This could be caused by its vulnerability to overfitting. When using 5 folds this reduction in accuracy is also seen for the single classification tree. The multinomial naive bayes and random forests models also give less accurate results in the case of 5 folds, while the accuracy loss of the random forests is very small. On the contrary the logistic regression model seems to perform better in the case of 5 folds as well, when we also consider bigrams instead of just considering unigrams. So adding bigrams to improve the performance is rather situational, as it might also decrease the performance or have too little effect.

4.5 Most Important Terms

In order to understand the contribution each feature has on the classification, we use formula 1, in which we calculate the score of a feature f by looking at the conditional probabilities over the class labels $Truthful$ and $Deceptive$.

$$score(f) = \log \hat{P}(f|Truthful) - \log \hat{P}(f|Deceptive) \tag{1}$$

This formula will help us to find the association between features and class labels. A high positive value for this score indicates a tendency toward the class of deceptive reviews, while a low negative value for the score indicates a tendency toward the class of truthful reviews.

To determine the log probability of the presence of a feature given a certain class we use the multinomial naive Bayes model. We train a seperate MultinomialNB classifier on the full dataset, which calculates these log probabilities. Then we retrieve these values from this model for all features/terms given the class labels corresponding to the deceptive and truthful reviews. To calculate the score of every feature we simply subtract the log probability of this feature given the deceptive class from the log probability of this feature given the truthful class. We have done this both considering just unigrams and also considering

both unigrams an bigrams. In this report we show for both cases the five most important terms pointing to deceptive reviews as well as the five most important terms pointing to truthful reviews. These terms can be found in Table 2. The scores, calculated with the log probabilities found by the model, can also be found in this table.

As expected the scores are in general lower when considering bigrams as well, because there are a lot more features to consider. Consequently the log probabilities will decrease significantly. Furthermore we see that the top five terms found are almost the same terms when we also include bigrams. In the truthful reviews only the term *security* is replaced by the bigram *the conference*, where *smell* and *millennium* are replaced by *chicago* and *my room* in the case of deceptive reviews. We note here that the order of terms are not consistent, for example when looking at the terms *smell* and *recently* for the deceptive reviews. For unigrams *smell* is considered as a better indication for a deceptive review than *recently*, while this is the opposite when considering both unigrams and bigrams. This is due to the randomness in the data preparation.

When considering bigrams next to unigrams the data is processed and prepared once more, because there are more features needed to consider as well as the randomness in the data preparation these results can be slightly different each time the code is executed. A consequence of this is also that the top five most important terms can also change every time the code is run, but the general score of the features will be relatively the same each time. Therefore these terms are valid examples of the most important terms, but there could be other terms which could give similar or even more extreme scores.

	Unigrams		**Unigrams & Bigrams**	
Review Class	**Terms**	**Score**	**Terms**	**Score**
Deceptive	luxury	−1.508	luxury	−1.239
	millennium	−1.363	chicago	−1.128
	smell	−1.275	seemed	−1.089
	recently	−1.244	my room	−1.083
	seemed	−1.243	recently	−1.074
Truthful	priceline	1.308	elevators	0.923
	elevators	1.216	priceline	0.910
	conference	1.202	conference	0.905
	security	1.123	the conference	0.884
	elevator	1.073	elevator	0.810

Table 2: The five most important terms pointing to deceptive and truthful reviews when considering only unigrams and also when considering bigrams next to the unigrams. For every term the corresponding score, as calculated by using Formula 1, is also provided.

4.5.1 Terms Pointing to Fake Reviews

In Table 2 we see that the most important terms pointing to fake, or deceptive, reviews are *luxury*, *millennium*, *smell*, *recently* and *seemed*. Including bigrams we also found the term *my room* as an important term pointing to deceptive reviews, where the unigram term *chicago* was also uncovered in that specific run.

So negative reviews are indicated as being deceptive if they contain one of the terms *luxury*, *millennium*, *smell*, *recently*, *seemed*, *chicago* or *my room*. These terms will therefore often be used by people who write fake negative reviews, or at least a lot more often than usual in genuine negative reviews.

The terms *millennium* and *chicago* are often used referring to the names of the given hotel or other places near the hotel containing these words in their name. Here the term *chicago* of course also points to the city of the hotel, so it is very obvious why that term could often appear. The reviews are for example for the hotels Fairmont Chicago Millenium Park Hotel, Hotel Monaco Chicago, Millenium Hotel Chicago and many more. For people writing fake negative reviews it would of course be very alluring to explicitly mention the hotel a few times trying to really bash this hotel. Along the lines of these terms we also find the term *my room* as a very important term pointing to deceptive reviews. Of course the most hotel reviews, whether they are negative or positive, will say something about the room where the specific person stayed in. But as the very negative score shows, this term is a lot more common in deceptive negative reviews, compared to truthful negative reviews. This probably also has to do with the fact that the person making up fake complaints about the hotel room will explicitly state the term *my room* a couple of times to strengthen their argument. But in trying to write a negative review they evidently use this term a lot more than is done in the truthful reviews.

An easy thing to think about when wanting to complain about a hotel is the smell in a hotel or hotel room. Nobody likes a smelly hotel room and if a review mentions that then probably a lot of people will reconsider their decision because of this. Therefore we also see that the term *smell* is a strong indication of a fake negative review. In genuine reviews this term is used far less, so this is evidently not such a big problem in hotel rooms.

In the deceptive reviews the term *recently* is also used a lot more than in the truthful reviews. The writer of the deceptive review probably mentions this term often to convince the reader of the review that they recently stayed at that given hotel, while this is evidently not true. A genuine reviewer does not feel the need to explicitly mention this, since they mainly want to share their experience and don't have convincing other people as their main goal when writing the review. So this term in most cases points to a deceptive negative review.

Furthermore the terms *luxury* and *seemed* are indicative of deceptive reviews. These terms are presumably used to convince the readers of the review that the accommodations at the hotel were not as they seemed and they did not experience the luxury they expected at the given hotel. This is likely done

by fake reviewers to convince people that the hotel room does not meet the conditions as advertised, while genuine reviewers will more often just state what is missing or wrong with the hotel room instead of making a full comparison with how the hotel rooms are advertised.

4.5.2 Terms Pointing to Genuine Reviews

The most important terms pointing to genuine, or truthful, reviews we found are *priceline*, *elevators*, *conference*, *security* and *elevator* as can be seen in Table 2. In the run where we also consider bigrams we also got the term *the conference* as one of five most important terms pointing to truthful reviews, which is of course highly correlated to the unigram term *conference*, where this term also includes the article before the actual word.

Thus it appears that negative reviews which mention elevators, conferences, the security or priceline are the most indicative terms pointing to a genuine negative review. Therefore a review with complaints about these concepts will be a truthful review most of the time. So when writing a fake negative review it will be very unlikely that people think about complaining about the elevators, conferences, the security or priceline. So problems coming up due to these concepts are not really expected to occur, but in practice do occur often, or at least more often than expected. For elevators we could think of the malfunctioning of an elevator or more likely busy elevators and thus having to wait a long time to get on the elevator. The terms regarding conferences could indicate that conferences in the given hotel are not organised well enough resulting in complaints about them. Regarding the term security, there could have been some incidents where the security did not act accordingly. Lastly we have the term priceline referring to Priceline.com which is a company that offers for example discounts for staying at hotels. Because this is a term which indicates true negative reviews, this company supposedly does not really deliver on their promise of arranging discounts for staying at hotels.

4.6 Significant

To test between models, we use a statistical test called McNemar's[2] test. The test is based on and makes use of a contingency table, which is depicted in table 3. This table marks how many items were correctly classified by each algorithm. The more commonly used and by Edwards corrected (continuity correction) formula to calculate the statistical result is as follows:

$$x^2 = \frac{(|B - C| - 1)^2}{(B + C)}$$

[2]https://wikistatistiek.amc.nl/index.php/$McNemar\_toets$/

Model	MNB	LR
Model 1 Correct	A	B
Model 1 Wrong	C	D

Table 3: McNemar Visual Explanation with model 1 = Bagging and model 2 = Random Forest

As you can see the test calculates a significance over the differences of the models, as it is only interested in items that the models performed different on. The statistic is reporting on the different correct or incorrect predictions between the two models. To perform this analysis we have to make a selection, because running all combinations of the 16 models goes both to far and was not possible due to randomization in the train and testset and the individual folds. Mcnemar's test requires the same sample items, requiring us to choose the folds and n-grams on which we were gonna perform this analysis. We choose to do this on the Bigram/unigram set, since this contained both the unigrams and bigrams. It also has a higher likelihood of being used in practice (from a NLP point of view), as unigram isn't always that usefull and from the bigrams you could infer other relations between words as well. We then also picked the 10-fold version, as we believe overfitting on the data is a bigger problem, especially considering the small data set containing just 400 items for each label. We had the following statistical results:

Model	MNB	LR	DT	RF
MNB		12.033	0.018	0.235
LR	12.033		0.083	14.702
DT	0.018	0.083		0.062
RF	0.235	14.702	0.062	

Table 4: McNemar P Value results

The null hypothesis for each of the combination is that the models would perform similarly. We would reject this null hypothesis if the p-values of the table are smaller than 0.05. We see this only occur once with the multinomial naive bayes classifier and the decision tree classifier. This signifies that there is a significant performance difference between these two models. As a side note we do see marginally significance performance differences between the decision tree and logistic regression and the decision tree and the random forest, signifying that there could be a performance difference, but we can't just make that claim from a statistical point of view. This supports the claim and idea that the decision tree classifier has the worst performance. But since we can't really differentiate between the other algorithms, which has been the seen in the other section of this report as well, we cannot determine which algorithms performs best

5 Conclusion

The increase of online shopping, using online services and an online presence during the corona outbreak offered online perpetrator a great opportunity to defraud users(Akdemir and Yenal, 2021). According to (Brianne, 2019),people are likely to value the product purely based on its reviews. Thus, the ability to differentiate between false and truthful reviews might be crucial in the future.

We trained and tested four different classifiers on a total of 796 negative reviews, of which 400 were false. A tf-idf analysis was performed on both unigram representations as well as uni- and bigram representations of the data. A hyper parameter tuning procedure was conducted during training to find the best versions of the Multinomial Naive Bayes, the Logistic Regression, the Decision Tree and the Random Forest Classifiers.

Comparing the generative and discriminative linear models, we found no big differences between the algorithms. Although the single decision tree classifier under performed others, random forest classifier was able to compete with the two linear models. Bigrams generally perform better on more folds, but they are very situational on fewer folds. We also found a ranking of words that were most predictive of either a deceptive or a truthful review. These words were rather consistent, but didn't necessarily gave much insight. For instance, luxury and recently were found to be best predictors for deceptive reviews and priceline, elevator, conference for negative reviews.

To the question of how the different algorithms perform and which might be the best, a rather vague answer would be 'it depends on the situation'. The decision tree classifier performs significantly worse than multinomial naive bayes but only marginally worse than random forests and logistic regression. The random forest performs very well, but gets weaker when you add more folds. We cannot reject the null hypothesis when comparing the p-values between the other models from a significant point of view. However, the naive bayes multinomial classifier scores extremely high on some evaluation measurements, but has some of the lowest results in a given context. Logistic regression and random forests performed similarly, but both performed marginally significantly better than the single decision tree classifier. Finally, the logistic regression classifier appears to be most consistent with overall high values in the measurements, and especially with recall. Statistically, however, we found no significant differences between logistic regression and the other models. Given the current context, the recall parameter is crucial, since marking an honest review of a customer or user as fake (and removing it) would be detrimental to the company's image. Here, logistic regression was very effective and consistent.

Our conclusion is that Multinomial Naive Bayes tends to outperform other methods in various trials. However, in the current situation with the high recall score being this important, Logistic Regression, as it is the most consistent as well, should also be considered.

References

Akdemir, N., & Yenal, S. (2021). How phishers exploit the coronavirus pandemic: A content analysis of COVID-19 themed phishing emails [Publisher: SAGE Publications]. *SAGE Open*, *11*(3), 21582440211031879. https://doi.org/10.1177/21582440211031879

Brianne, S. (2019, August 15). *Why people trust strangers when buying online* [Business 2 community]. Retrieved October 27, 2021, from https://www.business2community.com/ecommerce/why-people-trust-strangers-when-buying-online-02229487

De', R., Pandey, N., & Pal, A. (2020). Impact of digital surge during covid-19 pandemic: A viewpoint on research and practice. *International Journal of Information Management*, *55*, 102171. https://doi.org/10.1016/j.ijinfomgt.2020.102171

Diana, K. (2020, October 30). *20 online review stats to know in 2019* [Qualtrics]. Retrieved October 27, 2021, from https://www.qualtrics.com/blog/online-review-stats/

Groshek, J., & Koc-Michalska, K. (2017). Helping populism win? social media use, filter bubbles, and support for populist presidential candidates in the 2016 US election campaign [Publisher: Routledge _eprint: https://doi.org/10.1080/1369118X.2017.1329334]. *Information, Communication & Society*, *20*(9), 1389–1407. https://doi.org/10.1080/1369118X.2017.1329334

Kaemingk, D. (2020, October 30). *20 online review stats to know in 2019* [Qualtrics]. Retrieved November 4, 2021, from https://www.qualtrics.com/blog/online-review-stats/

Ott, M., Cardie, C., & Hancock, J. T. (2013). Negative deceptive opinion spam. *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: human language technologies*, 497–501.

Ott, M., Choi, Y., Cardie, C., & Hancock, J. T. (2011). Finding deceptive opinion spam by any stretch of the imagination. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 309–319. https://aclanthology.org/P11-1032