The Impact of Data Noise on a Naive Bayes Classifier

Reinier H. Stribos
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
r.h.stribos@student.utwente.nl

ABSTRACT

Data from the real world often contains noise. Mistakes made by humans, incorrect measurements or equipment malfunctioning are just a few examples of how data noise arises. There has been a lot of research on how to clean such noise from databases, but there is a shortage of research on the effect of data noise on the accuracy of different classification algorithms. This research aims to study this effect on a Naive Bayes classifier and to compare it to a Random Forest classifier. In this paper, both classification algorithms are explained, as are the different types of data noise, and how such noise is added to the different data sets for the experiments. Furthermore, the effect of data noise on the accuracy will be discussed and both algorithms will be compared to each other. This research shows that Naive Bayes is robust against data noise in the training data until around the 90 percent of data noise, whereas noise in the testing data has an intermediate effect. In both cases however, it is more robust than a Random Forest classifiers which is immediately and more significantly affected by noise.

Keywords

Naive Bayes, Random Forest, Data Noise, Classification Algorithms

1. INTRODUCTION

In the modern world, machine learning classifiers have become more and more present to solve complex problems. These classifiers are trained with, and tested on data, both generated in ideal conditions and taken from the real world. Ideally, this data is clean. In the real world however, the data will often contain noise. Such noise can occur in both the data with which the classifiers are trained —training noise—, as well as in the data which the classifiers have to classify —testing noise—, and in both the attributes of the data, as well as the classes of the data. A lot of research has been done on how to clean data sets from such noise [1]. However, not enough research has been done on the effect of data noise on the accuracy of different classification algorithms.

Even though Naive Bayes is quite simple, it can outperform many sophisticated algorithms [2, 3, 4]. However, these comparisons were all done without taking data noise in consideration. The goal of this research is to complement these studies by investigating how data noise affects a Naive Bayes classification algorithms.

In one of the few studies that did look at how the accuracy of different classification algorithms was affected by data noise, it was shown that Random Forest classification performed best compared to other classification methods [5]. Note that Naive Bayes was not discussed in this research. That is why, in this research, Naive Bayes is compared to Random forest classification to see how well Naive Bayes reacts to data noise.

1.1 Research questions

In more detail, this research aims to answer the following research questions:

1. What is the impact of different levels of noise on the accuracy of a Naive Bayes classification algorithm?
   (a) Which has a bigger impact on the accuracy; attribute noise or class noise?
   (b) Which has a bigger impact on the accuracy; training noise or test noise?
2. What is the difference of impact between randomly added noise and structurally added noise on the accuracy of a Naive Bayes classification algorithm?
3. How does Naive Bayes classification compare to Random forest classification when dealing with data noise?

The remainder of the paper is divided as follows: Section 2 gives an in-depth description of the used algorithms and data noise. Section 3 gives a brief overview of other researches with similar subjects. In section 4, the complete process of this research is explained. The results are presented in section 5, while section 6 reflects on the total research. In section 7 the research questions will be answered. Finally, possible interesting future researches will be discussed in section 8.

2. BACKGROUND

In this section a more technical background of the techniques used in this research will be discussed. First the different algorithms are explained, followed by an explanation of the different types of data noise.

2.1 Algorithms

34th Twente Student Conference on IT Jan. 29th, 2021, Enschede, The Netherlands.

2.1.1 Naive Bayes

The Naive Bayes algorithm uses the following formula [6], which states that the chance that a value x belongs to a class c can be calculated using prior knowledge;

$$P(c|x) = \frac{P(x|c) \cdot P(c)}{P(x)}$$

Here $P(x|c)$ is the probability that value x is reported in class c, and $P(x)$ and $P(c)$ are the probabilities that x and c occur respectively. These variables can all be calculated with the training data. When presented with data to classify, Naive Bayes will calculate the probability for each class and select the class with the highest probability.

Even though Naive Bayes assumes independence between the variables —an assumption which does not often hold in the real world— it scores high accuracy levels [7].

2.1.2 Random Forest

A Random Forest tree Classifier is an improvement on the Decision Tree Classifier [8]. A decision tree is a tree graph with on each node a threshold to split the data. E.g. if a value is higher than the threshold the decision tree takes the right path, and otherwise the left path. The tree ends in classification nodes, which, when reached, will classify the data into their respective classes.

A Random Forest contains a multitude of such trees. Each tree in the forest is trained using a different subset of the complete data set [8]. Because every tree had a different training data set, different classifications can be given by different trees. The forest looks at all these classifications and selects the class with the most votes.

2.2 Data noise

Generally speaking, a data set can be divided into two groups: Attribute data and Class data [9]. Attribute data contains the information, e.g. the answers in a survey are attribute data. Class data assigns a class to that information, e.g. the conclusion of a survey puts the respondent in a class. Noise can be found in both groups. Attribute noise is noise in the attribute data and class noise is noise in the class data.

There are multiple ways to classify the generation of data noise [10]:

1. By its distribution; distributed relative to the data value or the data variable.
2. By its location; whether the noise is introduced into the attribute data or class data, or into the training data or test data, or a combination of these options.
3. Completely at random.

By generation by distribution, also called structured noise, the noise will be based on the original data. For example, when looking at cheese, most cheese is made out of cow milk, some goat milk or sheep milk, and milk of a very limited number of other animals is used. Consequently, when adding structured noise into the variable which describes which animal's milk was used, cow milk has a much higher chance to be added than goat or sheep milk.

3. RELATED WORK

Brodley et al. [11] looked at ways to identify mislabeled data in a data set. They used so called *filter algorithms* to identify mislabeling. These filter algorithms divide the data set into n parts. For every split, they are trained with the other $n-1$ parts. The resulting classification was then used to tag every entry in the used part as either mislabeled or correct. data got tagged as mislabeled if the classifiers failed to classify it correctly. These filter algorithms filtered out about 85% of mislabeled data and 5% of correctly labeled data.

Lodder [12] gave an overview of different techniques on how to deal with missing values, while Zhu et al. [9] looked at different methods on how to handle data noise. Both discussed techniques which improved the accuracy of the classifiers by deleting or correcting the data with noise or missing values. They recommended different techniques for different situations but recommended above all to try to prevent data noise from entering the data sets.

Cortes et al. [13] proposed a method for estimating the impact on performance imposed by the quality of the data set. Their result is independent on the machine learning algorithm, as it is expressed as a characteristic of the data. However, certain conditions need to be met before this method becomes reliable.

Multiple researchers have compared the effect of data noise on different classifier algorithms [5, 10]. They determined the sensitivity of certain classifiers through experiments. This paper aims to continue with the latter approach and to provide new insights into unexplored classifiers. These results will be compared to the least sensitive classifier, which is, as section 1 mentioned, the Random Forest Classifier.

4. METHODOLOGY

In this section the general approach to this research is illustrated. Firstly, a brief description is given of the different data sets which were used, and an explanation on why those particular data sets were selected, after which the insertion of the noise into the data sets is explained. Finally, the experiments to test the sensitivity are described.

4.1 The data sets

For this research, three different data sets were selected from Kaggle[1], which is an online platform with a multitude of public data sets. The first data set is about whether a mushroom is poisonous or not. It contains categorical values about the colour, cap-shape, odor, habitat, population, the stalk and so forth, as well as a boolean value which indicates whether the mushroom is poisonous. This data set was selected because there is a strong correlation between the characteristics about a mushroom and its edibility.

The second data set looks at the quality of wine. It contains numerical values about the acidity, sugar, amount of alcohol, pH, sulphates, and the quality of wine. The quality of the wine was then divided in good wine —with a quality of six and higher— and bad wine to create groups of approximately equal sizes. This data set was selected because the correlation between the information about a wine and its quality was a lot less evident. Making it considerably different than the previous data set.

The last data set distinguishes spam emails from emails that are not spam, also known as ham. It contains the text in the mails and a boolean value indicating whether it is spam or not. This data set was selected because Naive Bayes is often used for filtering spam [7]. In this data set, the ham emails are over-represented, making up around 80% of the data set.

[1] `https://www.kaggle.com/uciml/mushroom-classification`

4.2 Generating noise

To test different levels of noise there will be multiple tests with increasing data noise; 0%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, and 100%. The percentage is the ratio of data noise versus correct data. As is mentioned in section 2.2, there is a difference between random noise and structured noise, as well as a difference between attribute noise and class noise.

4.2.1 Random noise

The first type of noise that will be added, will be random noise. Random noise was added to the data sets by looking at the possible options and randomly generating one. Looking at the data set describing wines with numerical values, this meant to find the maximum and the minimum value of a category and to generate a random number in between. For the spam data set, this meant to replace a word with a random word from the English dictionary. Algorithm 1 shows how random noise was added to the data set about edibility of mushrooms. It starts with putting all the different options for a variable into a list. When replacing a value with noise, a random value got picked from that list and put into the data set. The same algorithm could be used for the other data sets, by changing the part where the noise gets added to match the description given above.

Algorithm 1 Adding random noise to mushroom data set

```
function AddRandomNoise(dataset, ratio)
    for all column ∈ dataset do
        vars ← countVariables(column)
        for all value ∈ column do
            if random() ≤ ratio then
                value ← vars[random() · len(vars)]
    Return dataset
```

4.2.2 Structured noise

In the real world, noise is hardly ever random. It is often based on the nature of the category. For example, in the English language, the word "the" occurs much more often than the word "impeachment". When adding structured noise, these distributions were kept in mind. For the data sets about mushrooms and spam, the occurrence of each variable was counted and considered when adding the noise to the data set. Algorithm 2 shows how this was applied to the wine data set. At first, the mean and the standard deviation of a category were calculated. After that, a random, normally distributed number was generated. This number was modified with the standard deviation and the mean of the category and was added as data noise.

Algorithm 2 Adding structured noise to wine data set

```
function AddStructuredNoise(dataset, ratio)
    for all column ∈ dataset do
        mean ← mean(column)
        std ← standardDeviaton(column)
        for all value ∈ column do
            if random() ≤ ratio then
                value ← randomNormal()· std + mean
    Return dataset
```

4.2.3 Class noise

The function to add class noise was the same for each data set because in every data set, the different classes were represented in the same way, with a 0 representing one class, and a 1 the other class. The function simply switched a 0 to a 1 and the other way around. This function is given in algorithm 3. In class noise, there is no difference between random noise and structured noise because there are only two options; to switch or not to switch, and that is determined by the ratio of noise.

Class noise is only added to the training data. Testing the sensitivity of classifiers to class noise in testing data has no use because the classifiers only look at the labels of test data after the data has been classified, to check whether they were correct or not. Changing these labels therefore has no impact on the classification itself.

Algorithm 3 Switching classes

```
function SwitchClasses(dataset, ratio)
    for all row ∈ dataset do
        class ← getClass(row)
        if random() ≤ ratio then
            class ← (class −1)^2
    Return dataset
```

4.3 Experiments

To set up the experiments, a training set and a test set needed to be generated, which was done by 10-fold cross validation, meaning that 10 percent of the data was selected randomly and extracted to be used as the test sets. This process was repeated every time at the start of a new experiment to reduce the chance of an accidental coincidence.

Five different experiments were carried out on every data set. In every experiment, noise was added in the data set to either the training data or the testing data, either randomly or structured and either in the class data or the attribute data, starting with 10% and increasingly getting more. The accuracy of both classifiers was reported and compared to one another and previous experiments. Because a lot depends on randomness, every experiment was executed ten times to decrease the chance of an accidental coincidence and the average results, together with the standard deviation, are reported. Afterwards, the confusion matrices were calculated to get a more detailed view on the mistakes of the classifiers.

5. RESULTS

The results of every individual experiment can be found in the appendices. The results have been grouped together in graphs which will be presented in this section to make it easier to compare different settings to each other.

In figure 1 the effect of random noise in the training data in the three different data sets has been plotted. This graph shows three different things: it shows that Naive Bayes is much worse at predicting the quality of wine than it is at the other two data sets. This was predicted, as that data set had been selected for that very reason. The second thing we notice, is that the different levels of data noise do not have a very big impact on the accuracy, until around 80 percent for the wine, and 100 percent for the other two data sets, when the accuracy plummets. The last thing this graph shows, is that when classifying wine and mushrooms, Naive Bayes finishes around 50 percent accuracy, which happens because when there is 100 percent noise in the data set, the classifier can not learn anything about what differs the two classes and will make an almost random guess, which will achieve an accuracy level of around 50 percent. The spam data set was not evenly divided

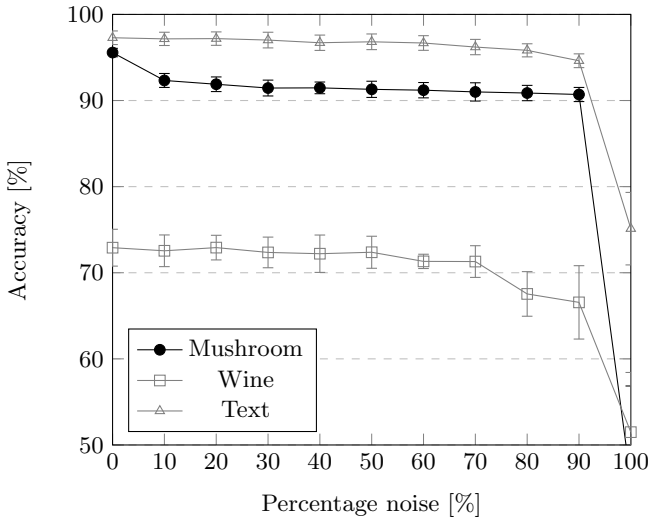

Figure 1. Effect of random training noise on Naive Bayes

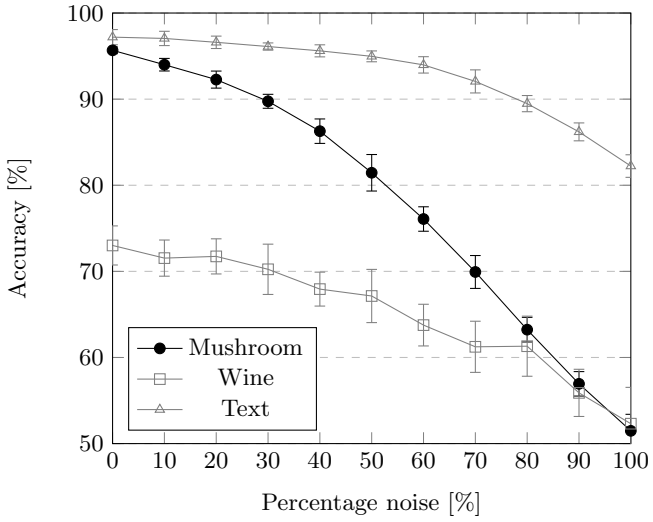

Figure 2. Effect of random testing noise on Naive Bayes

because the ham emails were over-represented, which resulted in the fact that the classifier would classify every mail as ham when there was no clear match. As the ham emails made up about 80 percent of the data set, and Naive Bayes classified most emails as ham, it had an accuracy of about 80 percent at 100 percent noise.

Figure 2 shows the effect of random data noise in test data on the accuracy of a Naive Bayes classifier. The accuracy at the start is the same as figure 1, but here the accuracy immediately and gradually declines instead of the sudden and extreme decrease in the training data. The accuracy at 100 percent noise is around the same as the experiments with data noise in the training data, with small deviations.

Figure 3 shows the effect of structured noise in testing data on the accuracy of a Naive Bayes classifier. The effect of structured noise in training data is not shown here because comparing that to figure 1 results in the same conclusion as comparing figure 2 and 3, the results of the experiments testing this effect can be found in the appendices. Comparing figure 2 and 3 shows that there is not much difference between structured noise and random noise. The accuracy on the wine data set and the spam data set seem to decrease slightly faster in the beginning to end at the

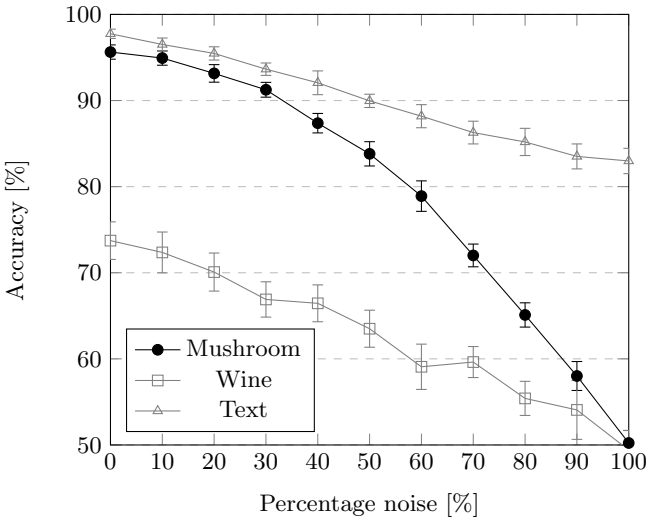

Figure 3. Effect of structured testing noise on Naive Bayes

same point, whereas the accuracy on the mushroom data set seems to increase slightly.

Figure 4 shows the effect of noise in label data on the accuracy of a Naive Bayes classifier. The graph is almost symmetrical, with small differences due to randomness. This effect happens because 100 percent noise is the inverse of 0 percent noise, since if the classifier gets told that it should classify every spam email as ham instead of the correct spam, and vice versa, it will do so. At 50 percent noise, half of all the labels are switched, and both classes are almost evenly represented, resulting in a classifier that can only guess, which results in an accuracy of about 50 percent. The graph is comparable to figure 1 for the first 30 to 40 percent noise, albeit that the accuracy is twice as sensitive when applying data noise. This is due to the fact that the labels of the training data were adjusted, giving the same effect as data noise in the training data. Only the data set on spam emails seems to be more sensitive towards noise in the labels.

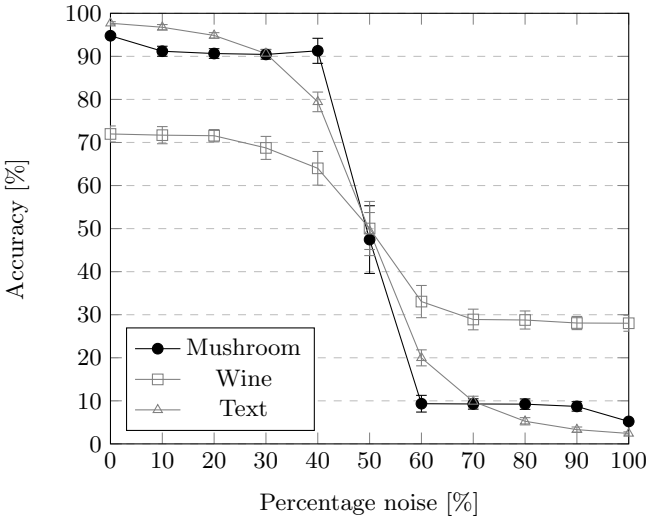

Figure 4. Effect of class noise on Naive Bayes

5.1 Comparison with Random Forest

Figures 5 and 6 compare the sensitivity of a Naive Bayes and a Random Forest classifier. Figure 5 shows the effect

of structured data noise in the training data of the mushroom and the wine data sets, whereas figure 5 shows the effect of random noise in the testing data of the wine and the text data sets. More experiments were carried out, the results of which can be found in the appendices. The Random Forest classifier performed better when no noise was added when classifying mushrooms or wine, whereas Naive Bayes performed slightly better on the spam data set. While the Random Forest classifier outperforms Naive Bayes at the start, it is more sensitive. When adding noise to the training data, the accuracy of the Random Forest classifier immediately decreases, while the Naive Bayes remains relatively unaffected until the very end. Because of this, Naive Bayes starts to outperform the Random Forest classifier around 50 to 60 percent data noise in the training data, until 100 percent, when they are both equally bad. Naive Bayes starts with a higher accuracy on the spam data set, and it keeps outperforming until the very end, when it is overtaken by the Random Forest.

Data noise in testing data has more of an immediate effect on the accuracy of the Naive Bayes classifier, but the accuracy of the Random Forest decreases faster and it gets overtaken by the Naive Bayes again, although how fast this happens and how big the difference is, differs between the three data sets. When testing on the wine and the text data sets, the accuracy between the two classifiers are somewhat even, but the results of the experiment with the mushroom data set, shows a slightly better performance of Naive Bayes.

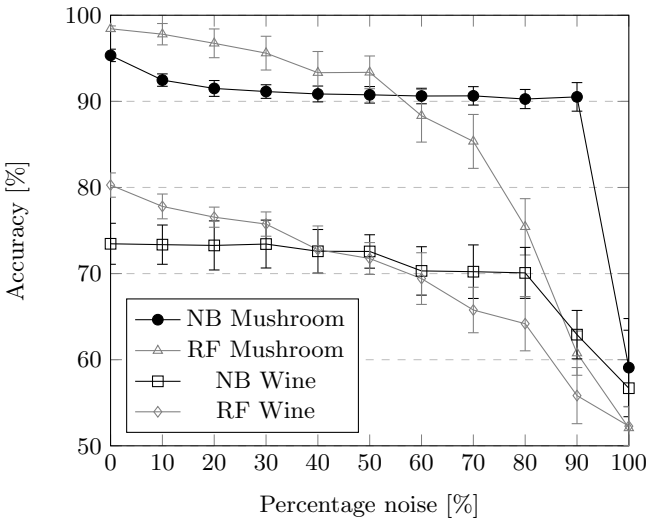

Figure 5. Effect of structured noise in training data

6. DISCUSSION

Naive Bayes performed better on the text data set, both with and without data noise. This was as expected because Naive Bayes performs strongly on text classification [7]. Random Forest performed better on the other two data sets when no noise was added. Naive Bayes showed to be more robust against data noise and the two classifiers cross each other around the 50% percent noise. Therefore, when dealing with a data set with over 50% noise, Naive Bayes is recommended. In practise however, data sets with such an amount of noise are practically nonexistent and it would be more impactful to try to reduce the data noise, making Random Forest the recommended classifier for all data sets but text classifications.

Certain choices have been made during this research, which could have impacted the conclusions. Firstly, the chosen data sets have a big influence on the reported accuracy. As shown, different types of data sets have different results, in both the starting accuracy and the sensitivity to data noise. Next to that only data sets with two classes have been used in this research. More classes have a big impact on the accuracy because it is easier to make a wrong decision. Therefore, the results could be different when testing a new type of data sets. In a real world scenario, the data and the noise will most probably deviate from what is used for this research, which will result in different results.

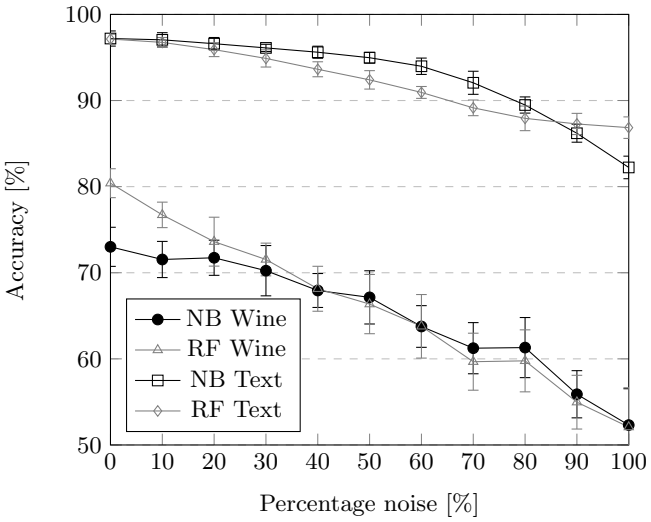

Figure 6. Effect of random noise in testing data

Furthermore, the experiments have only been carried out with one configuration of the Random Forest classifier and one type of Naive Bayes. Things as a different amount of trees or the function used to decide on what attributes to split, can have an impact on the sensitivity of a Random Forest classifier. Next to that, there are a couple of improvements on Naive Bayes which increase its sensitivity in certain situations [14]. In this research only the base formula is used. In real world scenario's, these improvements on Naive Bayes will be used and a Random Forest will be optimally configured for the situation, both of which will result in different results.

7. CONCLUSION

7.1 Training versus testing

As shown in figure 1 and 2, data noise in the training data does not have a very big impact on a Naive Bayes classifier until around 80 to 90 percent, when the accuracy plummets. Whereas data noise in the test data has an immediate effect on the accuracy, starting at 10 percent data noise. Both end around the same accuracy level at 100 percent, but noise in the training data remains unaffected longer. Therefore, we can conclude that data noise in the testing data has a bigger effect on the accuracy of a Naive Bayes classifier than data noise in training data.

7.2 Random versus structured

Figures 2 and 3 show the difference between the impact of random data noise and structured data noise in the test data on the accuracy. The effect on the mushroom and the wine data sets look almost identical, with only small differences. The effect on the text data set is a bit more present, but a significant difference remains absent. The difference between random and structured noise in

training data can be found in the appendices, and these results also show no clear difference. Therefore, we can conclude that there is no significant difference between the effects of random data noise and structured data noise on the accuracy of a Naive Bayes classifier.

7.3 Attribute versus class

For the first percentages of noise, figure 4 shows no clear difference with figure 1. However, starting from 30 percent noise, the difference becomes significant, and the accuracy in figure 4 decreases faster than all the other graphs. Furthermore, at 100 percent noise, figure 4 shows the lowest accuracy of every graph. Therefore, we can conclude that data noise in the labels has a significant bigger impact on the accuracy of a Naive Bayes classifier than data noise in the attributes.

7.4 Naive Bayes versus Random Forest

Figures 5 and 6 compare Naive Bayes and Random Forest classifiers. While Random Forest generally starts with a higher accuracy, data noise has a bigger impact on it and Naive Bayes overtakes it, at the latest, between 50 and 60 percent noise. Data noise in testing data has a bigger impact on Naive Bayes than noise in training data, and here the two classifiers show less difference but Naive Bayes remains less sensitive. Therefore, we can conclude that data noise has a bigger impact on the accuracy of a Random Forest classifier than that of a Naive Bayes classifier.

8. FUTURE WORK

This paper discusses a subset of algorithms and data sets. A lot more research can be done on this topic, by researching other algorithms or different types of data sets.

One of the future possible topics is different classifiers. In this paper only two classifiers are studied, without any optimization. More classifiers have been tested in other papers, but there are still many out there which have yet to be analysed. It would be interesting to see how these other classifiers compare to the classifiers presented in this paper. Furthermore, it would be interesting to see whether the known optimisations could improve the accuracy of Naive Bayes and Random Forests.

Next to different classifiers, different types of data could also be studied more intensely. This paper showed big difference between three types of data sets, more research needs to be done into more types of data sets. Interesting examples could be questionnaires or image recognition. Especially data sets with more than two classes, because differ significantly from the data sets tested in this research and it would be interesting to see what the effect of data noise is on those types of data sets.

Another interesting topic would be the impact of missing values in a data set. Missing values are another type of data noise, which was not tested in this research. Research needs to be done to study the effect of missing values on the accuracy and to compare those results with other studies.

9. REFERENCES

[1] A. C. Lorena and de Carvalho, "Evaluation of noise reduction techniques in the splice junction recognition problem," *Genetics and Molecular Biology*, vol. 27, no. 4, pp. 665–672, 2004.

[2] A. Ashari, I. Paryudi, and A. M. Tjoa, "Performance comparison between naïve bayes, decision tree and k-nearest neighbor in searching alternative design in an energy simulation tool," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 4, no. 11, 2013.

[3] D. Xhemali, C. J HINDE, and R. G STONE, "Naïve bayes vs. decision trees vs. neural networks in the classification of training web pages," *D. XHEMALI, CJ HINDE and Roger G. STONE," Naive Bayes vs. Decision Trees vs. Neural Networks in the Classification of Training Web Pages", International Journal of Computer Science Issues, IJCSI, Volume 4, Issue 1, pp16-23, September 2009*, vol. 4, no. 1, 2009.

[4] R. M. Rahman and F. Afroz, "Comparison of various classification techniques using different data mining tools for diabetes diagnosis," *Journal of Software Engineering and Applications*, vol. 6, no. 3, pp. 85–97, 2013.

[5] W. Schooltink, "Testing the sensitivity of machine learning classifiers to attribute noise in training data," 2020.

[6] I. Ben-Gal, "Bayesian networks," *Encyclopedia of statistics in quality and reliability*, vol. 1, 2008.

[7] V. Metsis, I. Androutsopoulos, and G. Paliouras, "Spam filtering with naive bayes-which naive bayes?," in *CEAS*, vol. 17, pp. 28–69, Mountain View, CA, 2006.

[8] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[9] X. Zhu and X. Wu, "Class noise vs. attribute noise: A quantitative study," *Artificial intelligence review*, vol. 22, no. 3, pp. 177–210, 2004.

[10] D. F. Nettleton, A. Orriols-Puig, and A. Fornells, "A study of the effect of different types of noise on the precision of supervised learning techniques," *Artificial intelligence review*, vol. 33, no. 4, pp. 275–306, 2010.

[11] C. E. Brodley, M. A. Friedl, *et al.*, "Identifying and eliminating mislabeled training instances," in *Proceedings of the National Conference on Artificial Intelligence*, pp. 799–805, 1996.

[12] P. Lodder, "To impute or not impute: That's the question," *Advising on research methods: Selected topics*, pp. 1–7, 2013.

[13] C. Cortes, L. D. Jackel, and W.-P. Chiang, "Limits on learning machine accuracy imposed by data quality," *Advances in Neural Information Processing Systems*, vol. 7, pp. 239–246, 1994.

[14] L. Jiang, D. Wang, Z. Cai, and X. Yan, "Survey of improving naive bayes for classification," in *International Conference on Advanced Data Mining and Applications*, pp. 134–145, Springer, 2007.

APPENDIX

A. MUSHROOM DATA SET

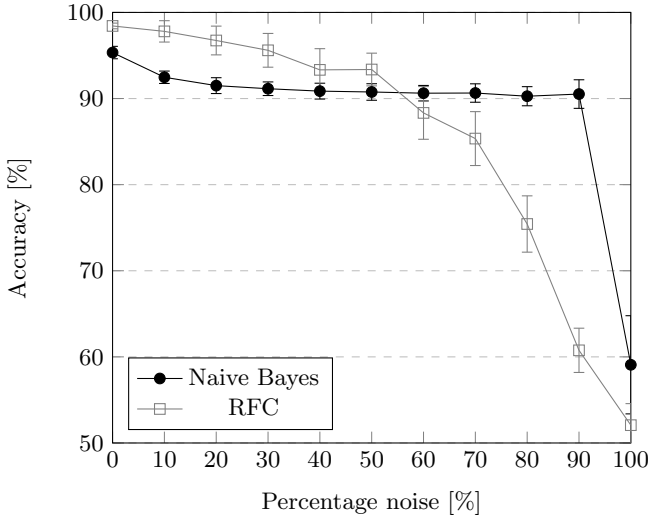

Figure 7. Structured training noise on mushrooms

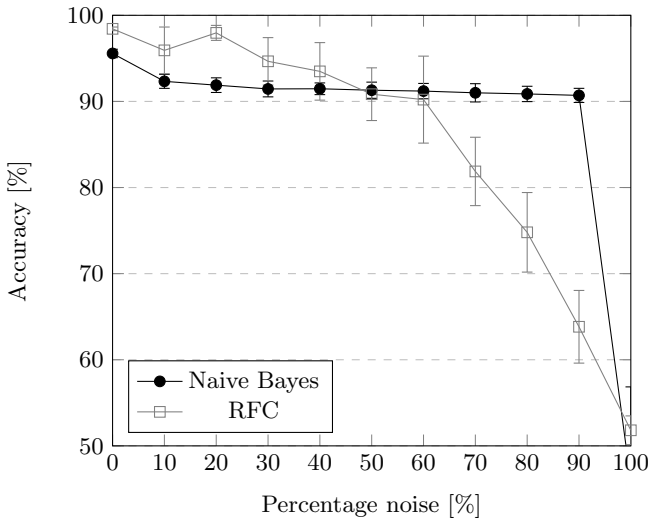

Figure 8. Random training noise on mushrooms

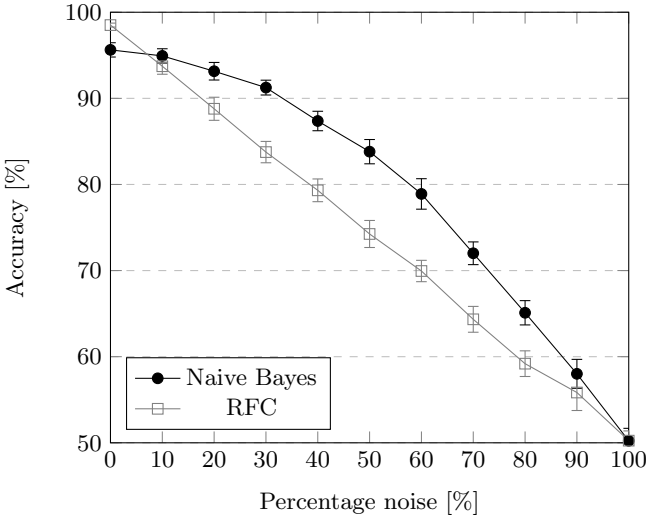

Figure 9. Structured test noise on mushrooms

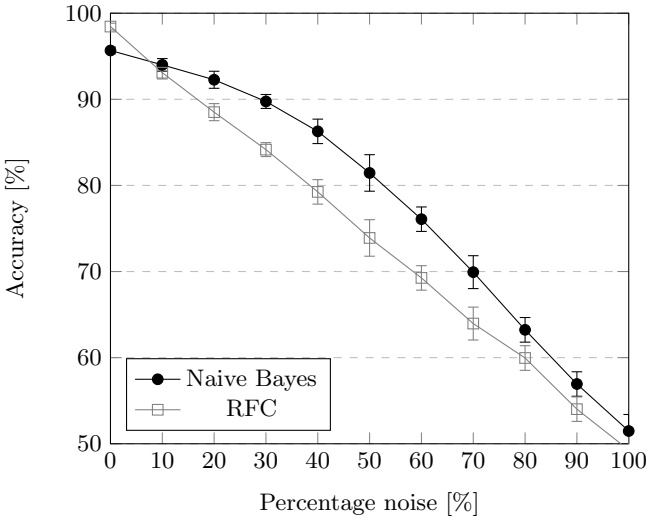

Figure 10. Random test noise on mushrooms

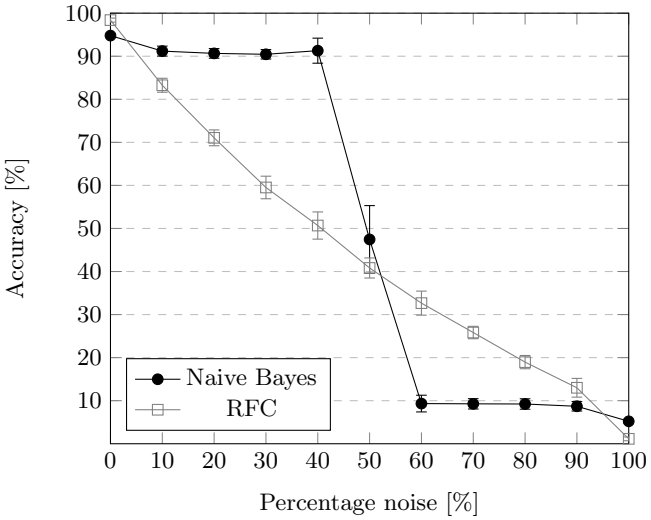

Figure 11. Training label noise on mushrooms

B. WINE DATA SET

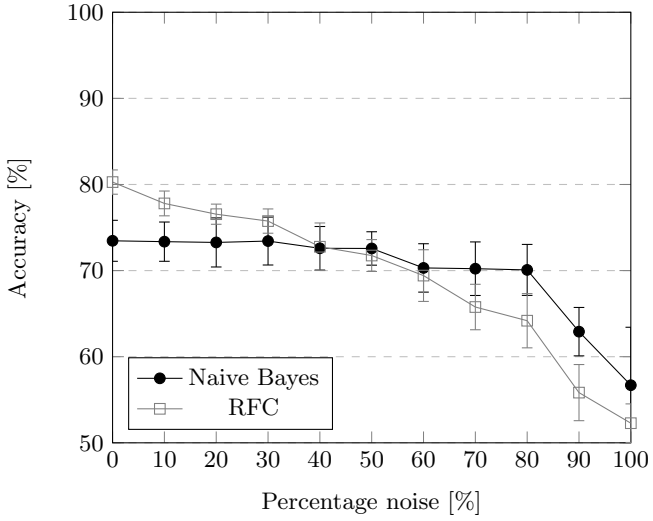

Figure 12. Structured training noise data on wine

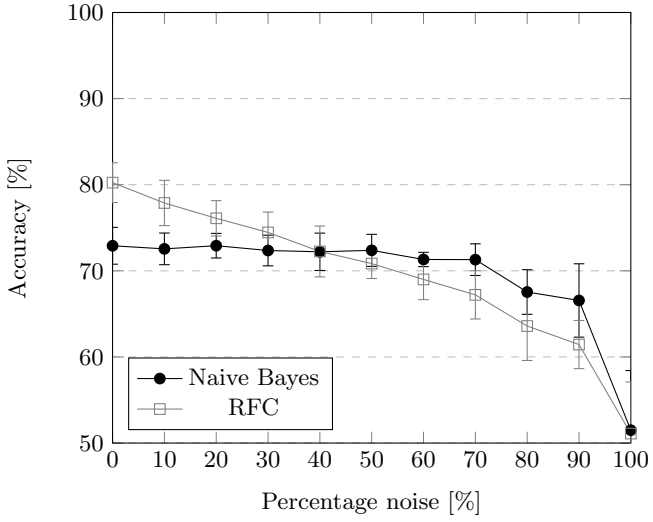

Figure 13. Random training noise data on wine

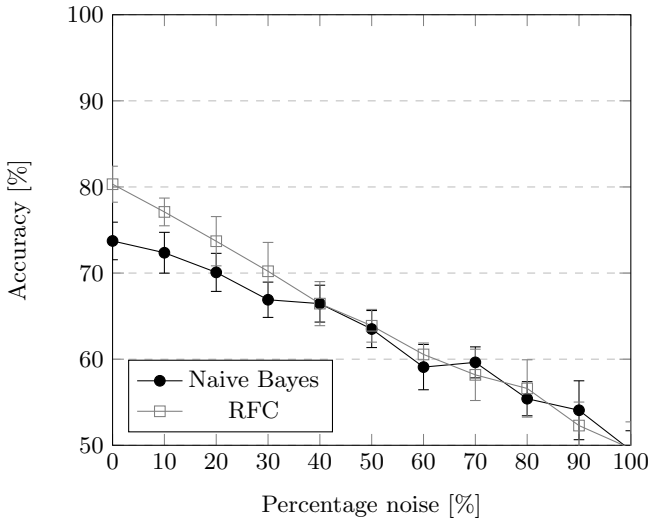

Figure 14. Structured testing data noise on wine

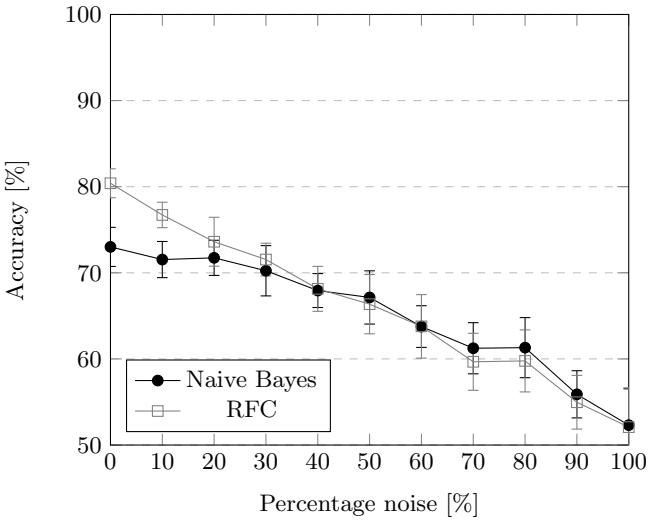

Figure 15. Random testing data noise on wine

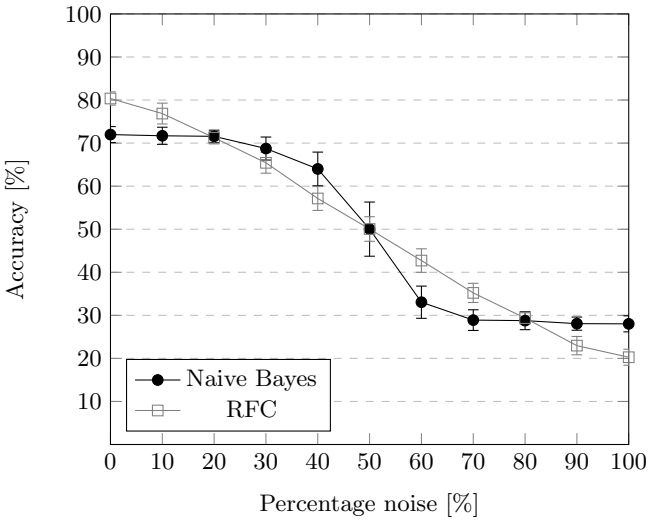

Figure 16. Training label noise on wine

C. SPAM DATA SET

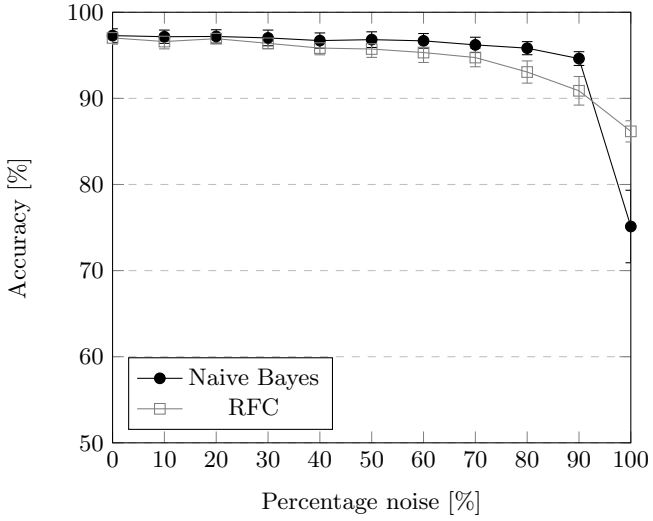

Figure 17. Random training data noise on text

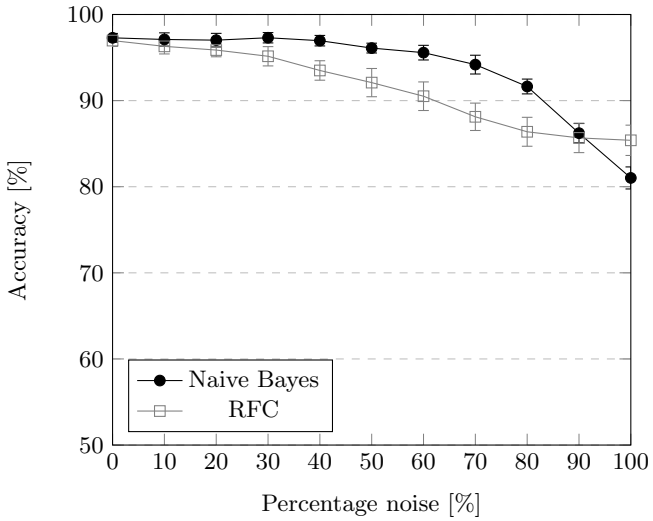

Figure 18. Structured training data noise on text

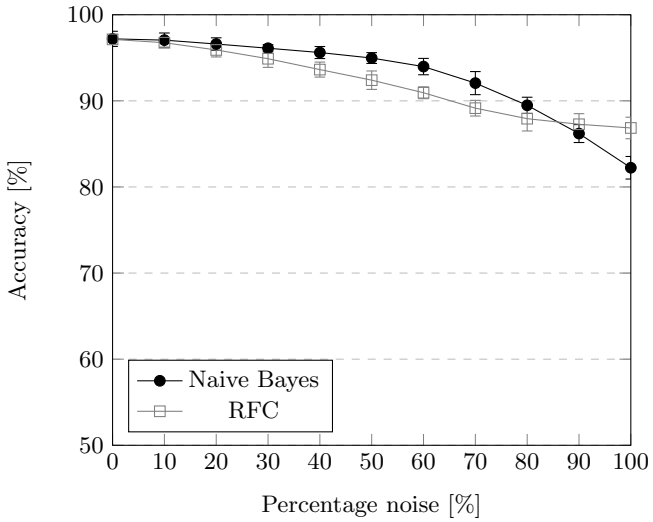

Figure 19. Random testing data noise on text

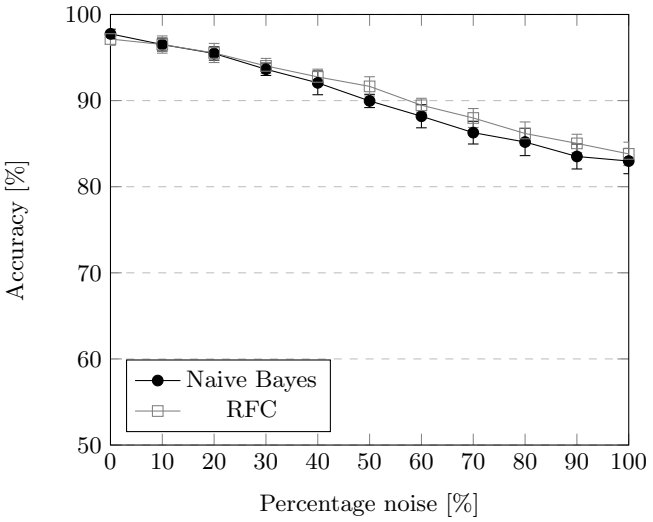

Figure 20. Structured testing data noise on text

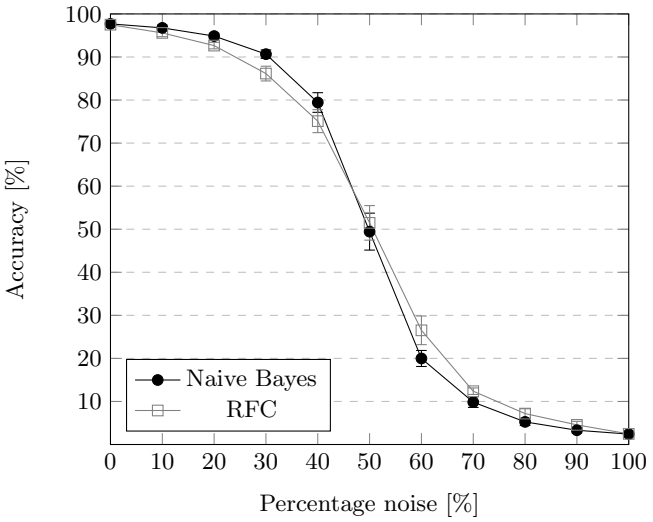

Figure 21. Training label noise on text