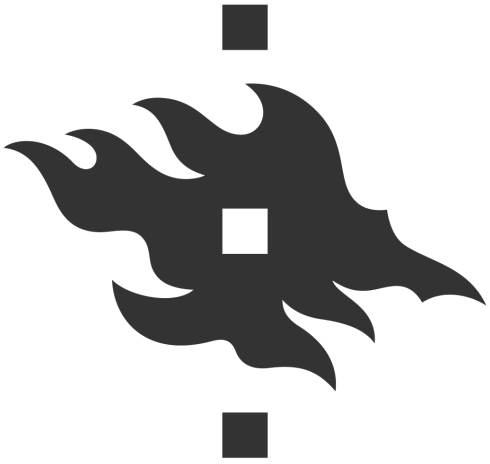


HELSINGIN YLIOPISTO

DATA1002: INTRODUCTION TO MACHINE LEARNING

TERM PROJECT
REPORT

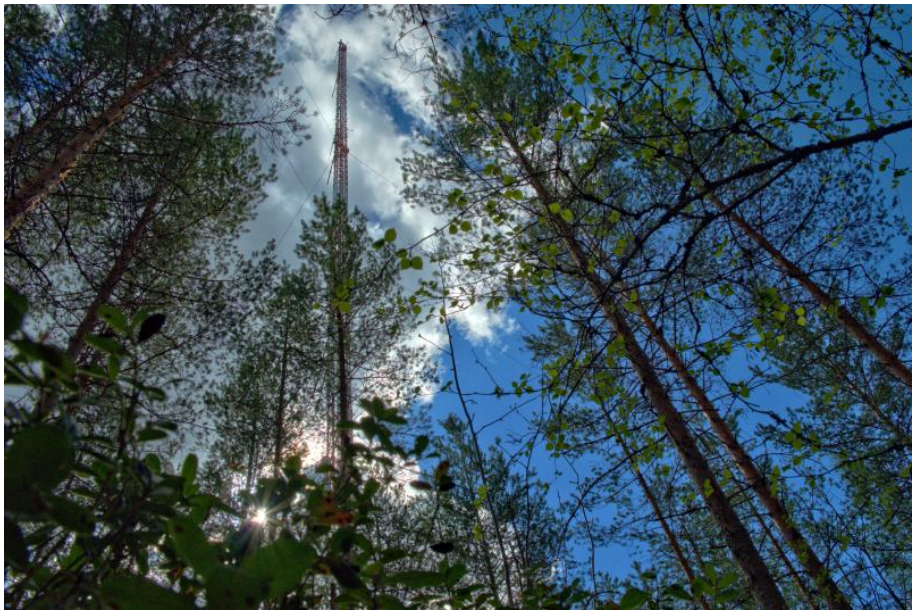
GROUP 69

Alexandros Matakos Theo Blauberg Rasmus H. Rasmussen

December, 2021

CHAPTER 1

INTRODUCTION



Hyytiälä forestry field station measurement tower. Image source: DEIMS

1.1 PROBLEM STATEMENT

The term project involved building a machine learning model for classification of NPF phenomena. These occur on days when small particles begin to form larger new particles and by spreading out affect weather and air pollution, among other things. The primary task was to predict if an event were to occur on a certain day, and the exact type of event could be found as a secondary task. Data from measurements at the Hyytiälä forestry field station (seen above) were made available for the purposes of the project.

1.2 MOTIVATION

Besides the apparent motivation for completing the course, our team intended to try its best to win the unofficial challenge in binary and multi-class classification accuracy. Apart from that, it was intriguing to tackle a real problem with real data, and see how we could apply our knowledge for solving a practical task.

1.3 GOALS

Our goal was to complete all of the project's tasks to the best of our capabilities, namely the challenge submission, the presentation and the report. At the same time, our goals for the project were inherently tied to our goals for the course, which were in-depth understanding of the various techniques and methods spanning all of the typical machine learning pipeline, such as feature selection, model selection and cross-validation. Finally, a common goal we all set from the beginning, was to do our best through teamwork, respecting and enabling all members to contribute freely.

CHAPTER 2

METHODS

Since we were instructed not to focus on the nature of the phenomena and measurements to build our model, we focused on general statistical and machine learning methods to conduct our analysis. Therefore, our approach was similar to that of a usual machine learning pipeline, and consisted of roughly three steps: exploratory data analysis/feature engineering, model selection and validation.

2.1 DATA EXPLORATION - FEATURE ENGINEERING

The available data was already clean, with no missing entries and labels assigned to all dates. That is why most of the data pre-processing regarded feature selection. It became obvious from the very beginning, that the data contained many over-abundant variables, several of which were highly correlated. The heatmap below shows this clearly. One reason for this, is that there are variables containing measurements

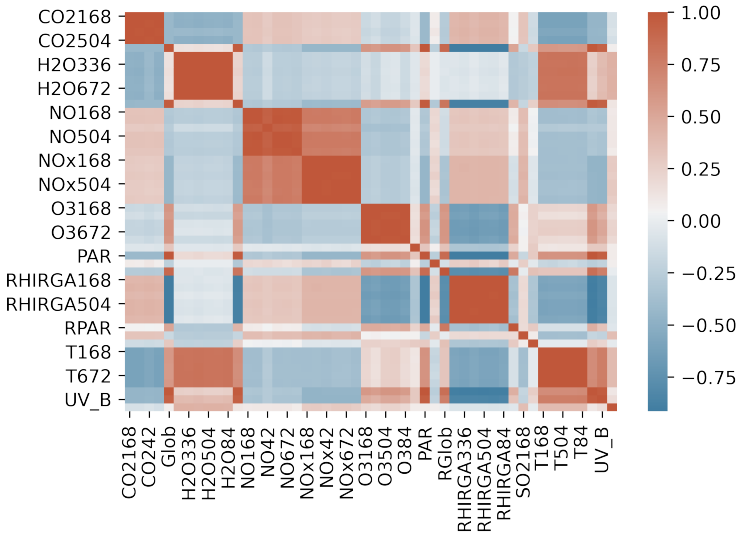


FIGURE 2.1
Some variables are highly correlated.

from different heights for each chemical element, making them very similar. One hypothesis we made in our analysis was that the measurements from different heights do not contribute any further information, and as such we chose to omit all heights except those made at 17 meters. Our hypothesis was confirmed by comparing the values of the variables from different heights, and by the increased performance of our model on the pruned data.

Our aim in the feature analysis was to end up with a dataset that contained as much information with the least amount of over-abundant variables (in line with James et al., Sec. 3.3.6). This was also motivated by the fact that the total available data had only 458 days with 100 variables, and a lower data to parameters ratio is known to lead to many issues related with high dimensionality. LASSO Polynomial and Logistic regression revealed that most of the variables related with the standard deviations ended up having coefficients equal to 0, and as such we decided to omit the std variables completely. This was again confirmed to improve the performance of the model later on.

Lastly, we looked at another heatmap with the remaining variables, as well as several pair-wise heatmaps and Kernel Density Estimation (KDE) plots with manually selected variables, and discarded the ones we deemed most unnecessary. For our visualizations, we reduced the dimensionality of the data using PCA on both train and test data. We then used *seaborn*'s KDE plots, which provide smoother looking results than histograms. For example, we can see in Figure 2.2 below that there is greater variation in ultraviolet radiation (type A) between events and nonevents than in C_2O or H_2O concentrations.

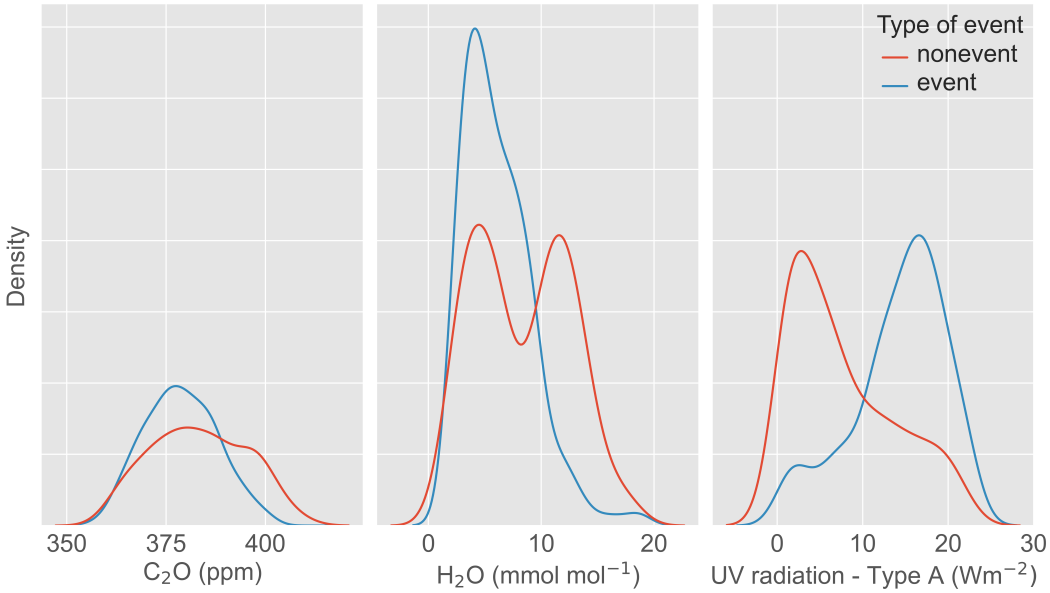


FIGURE 2.2
Some variables are better indicators of events.

Regarding discriminating between different types of events, we could see from the beginning that this would be a difficult task, since their respective measurements are very similar. Nonevents on the other hand show differences, and can be visualized for lower dimensions, as can be seen in Figure 2.3.

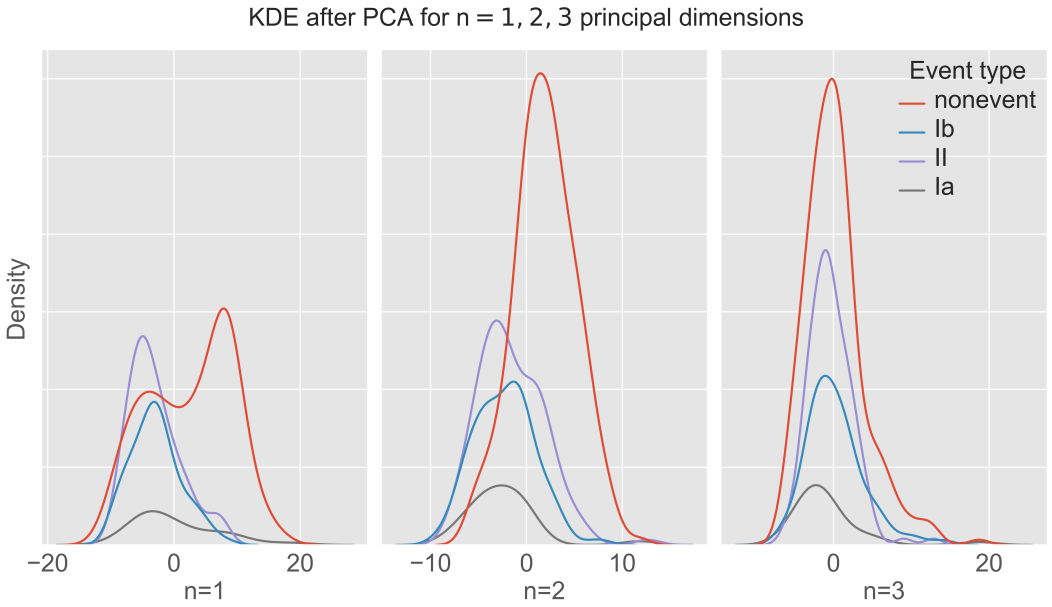


FIGURE 2.3
Nonevents are visually separable in lower dimensions.

This concluded the feature selection process and produced a dataset which was quite heterogeneous statistically-wise, with 18 predictor variables. A heatmap of the final variables can be seen in Figure 2.4. The last part was partly based on gut feeling towards the nature of some of the variables at hand. Even though we are by no means atmospheric scientists, this gut feeling was proved somewhat correct by the model performance, described next.

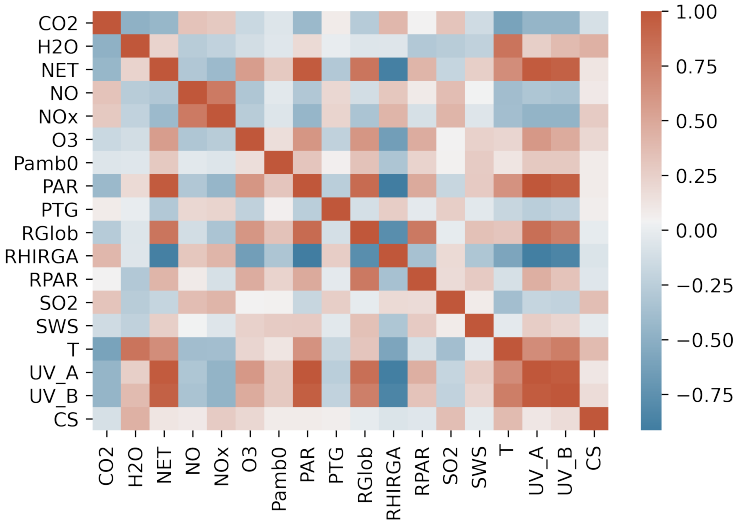


FIGURE 2.4
Heatmap of the final selected variables.

On a final note regarding the data, in order to have compatibility with *sklearn*'s methods, we converted the event types from strings to numeric classes: 0 for nonevent, and $1, 2, 3$ for the different event types. Furthermore, since the various measurements were in different units, data normalization was necessary to enable meaningful comparisons and inference from the data. As such, we normalized each variable by subtracting its mean from the respective observations and divided them by the standard deviation.

2.2 MODEL SELECTION

The model selection process mostly consisted of trial and error between various classifiers available from the sklearn library, and brute-force hyperparameter search. We only tried the models which we had seen in the course, as we possessed more in-depth understanding for those and wanted to avoid using black-box methods as much as possible. The discriminative classifiers we considered were Support Vector Machine, Random Forest and Logistic Regression, while we also considered a generative classifier, Naive Gaussian Bayes.

Before tackling the problem we expected SVM with radial kernel and Random Forest to perform the best, since they can practically capture any complexity in the data. We experimented a lot with Gradient Boosted Trees and Random Forest, but we did not get the expected results. Figures 2.5 and 2.6 contain a summary of our experimentation with various classifiers.

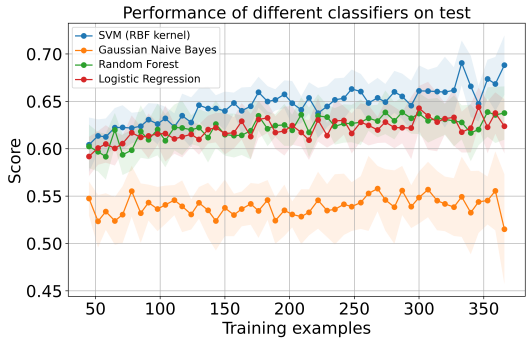


FIGURE 2.5
Performance of various classifiers on test

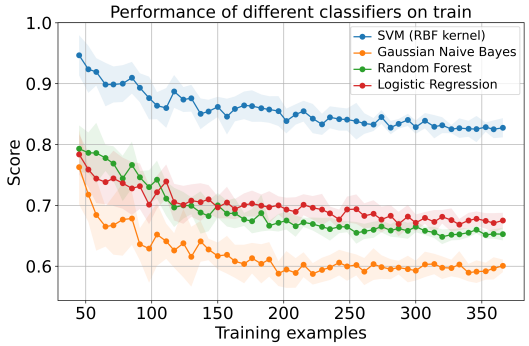


FIGURE 2.6
Performance of various classifiers on train

We can see that SVM seemed to perform the best, so we chose to focus on it and find the optimal parameter C. We describe our process in the next section.

2.3 VALIDATION

Our validation process was inherently tied with the relatively small amount of available data, and consequently the train test split procedure. We noticed that when splitting randomly all of the data and only afterwards dividing each train and test set into classes, there were cases where observations belonging to event type "Ia" (only 29 observations in total) would get heavily unevenly split, sometimes ending up completely in one of the two sets.

In order to ensure that our training set was representative of the phenomenon, we devised the following strategy. First, we separated each class. Then, we split each class randomly into two equal sized sets. Finally, we combined the resulting halves to form training and test sets. This ensured that the number of observations from each class was the same in the two sets, while the individual observations could end up in any of the two sets. Note, that the function responsible for this could be given any train test split ratio as input. We experimented on various splits, such as 80-20, 70-30, or even 90-10, but the

average performance was more or less the same, with 50-50 splits yielding the lowest variance. Therefore, we chose to go with that. We can see below in Figures 2.7, 2.8 that increasing the train set size has an increasing effect on the test accuracy up to about half of the total data.

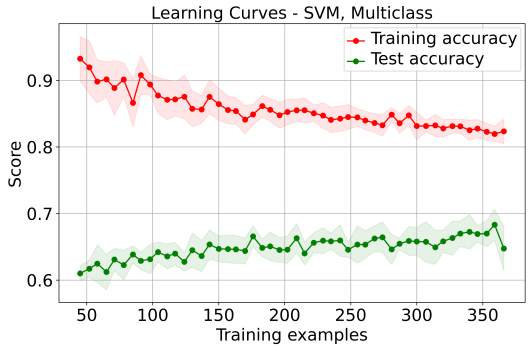


FIGURE 2.7
Normal classifier

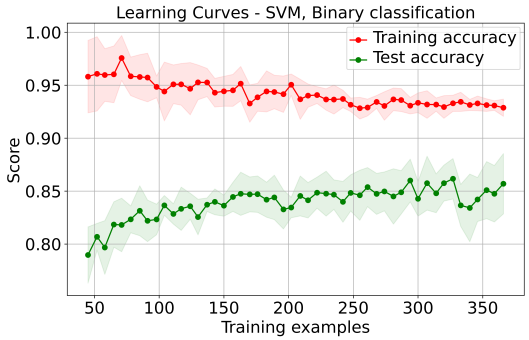


FIGURE 2.8
Overfit classifier

Since we did not have any information regarding which observations were the most informative, we repeated this procedure several thousand times and trained a classifier for each. We found that this captured better the overall structure of the data than sklearn's *StratifiedKFold()* method of splitting the data, since by splitting randomly thousands of times we guarantee covering most of the different combinations that the data can end up as training and test sets.

We used the aforementioned process to conduct brute-force search for the parameter C. This parameter has an intuitive meaning, since increasing it regulates how much the classifier should try to classify each sample correctly, or put differently, how small the margin of the fit hyperplane gets. The value that seemed to achieve the highest accuracy on the test split was $C = 3.75$. Values of C higher than that kept increasing the accuracy on the training set, but the test accuracy remained the same, which meant that the classifier was overfitting the data. Hence, we chose the highest value of C before overfitting took off. We can see this below in Figure 2.9.

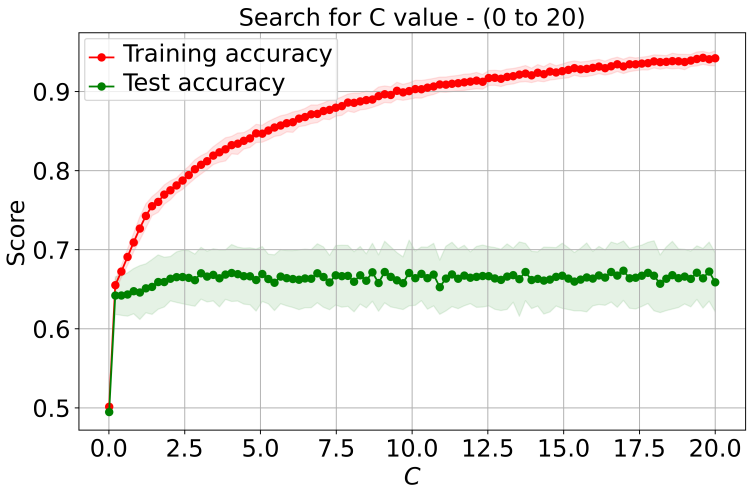


FIGURE 2.9
Accuracy for different values of C (multi-class).

Lastly, we confirmed the previous by drawing the decision boundaries for the PCA reduced data. In Figure 2.10, we can see the smooth decision boundaries for $C = 3.75$, while in Figure 2.11, with $C = 500$, the decision boundary is reminiscent of an overfit classifier.

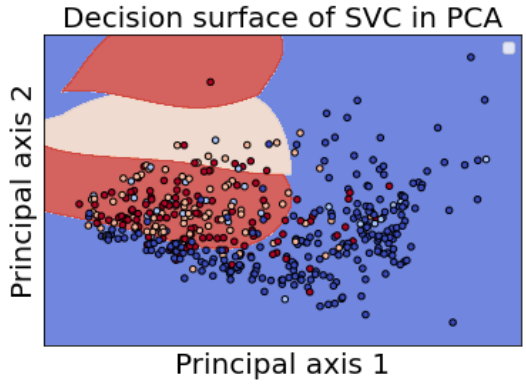


FIGURE 2.10
Multi-class classification

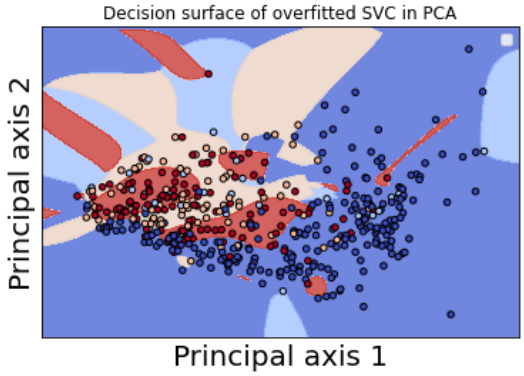


FIGURE 2.11
Binary classification

CHAPTER 3

RESULTS

3.1 PERFORMANCE

Having chosen a Support Vector Machine with a Radial Basis Function kernel and $C = 3.75$ as our classifier, we received some impressive results. Our performance on the validation sets was 84.6% on average for the binary classification task, and 65.5% on average for the multi-class classification task. These averages are over the thousands of classifiers trained on different splits described in section 2.3. The maximum performance over these splits was 91.3% and 73.6% for binary and multi-class respectively, which means that there were some "appropriate" train test splits, which allowed the classifier to learn the underlying structure of the data better.

The "impressive" part of the results came when the hidden test labels were published. Our classifier achieved an accuracy of 90.2% on the binary classification task, and 73.7% on the multi-class. From what we saw, the multi-class accuracy was significantly higher compared to other groups. Unfortunately, the reason our result was not on the published list was due to some miscommunication between our group members, which resulted in no one uploading the csv file with our answers to Moodle. Regarding the other two metrics of the challenge, we guessed 87% for our binary accuracy, and the perplexity came up to be 1.63, which was quite average compared to other groups. The reasoning behind our binary guess was that since the classifier would be trained in all of the data for the final challenge task, it would perform a bit better than the average 84.6% performance it had on validation, which was trained on half of the data. The table below summarizes our results.

	Validation	Test
Binary	84.6%	90.1%
Multi-class	65.5%	73.7%
Perplexity	-	1.63
Estimated	-	87%

TABLE 3.1
Performance metrics.

We noticed that our performance on the challenge was similar to the maximum performances during the validation phase. This led us to suspect that the classifiers trained on the "appropriate" random splits, managed to learn as well as when using all of the 458 observations to prepare for the challenge. Therefore, we attribute the higher performance of our classifier to the following reasons:

- Our feature selection removed noise from the data and allowed the classifier to learn the important features of the data. In fact, training the same classifier on the whole data (before feature selection) led to similar performance with the other groups.
- Our validation process took into account all of the data via a statistically sound splitting process, and as such the hyperparameter search based on it ended up producing good results.

One interesting thing we discovered after the publication of the hidden labels, was that when using a very large value of C for our classifier (eg $C > 500$), we would get even higher performance on the challenge set: 76.7% for multi-class. This value of C completely overfit the training dataset, with the accuracy on train being 100%. Even though this achieves a higher accuracy overall, we would avoid going with this value when deploying a real model, since even the challenge set data can be not fully representative of the phenomenon.

3.2 CONCLUSION

To summarize, we had the problem of classifying NPF phenomena. After extensive analysis of the available data, data exploration and feature selection, we prepared a condensed dataset for our classifier. We used an SVM classifier for predicting future events, which ended up performing very well in the challenge submission. We have found that the distinction between the various types of events is a hard one, and that typical data science methods are insufficient. We could find certain variables that were more important in the binary prediction task, but the multi-class task proved to be much harder. Perhaps this confirms that relevant area expertise is very important for data science projects.

All of the project code and material is available on our group's Github project page.

CHAPTER 4

GRADING

Here is how we evaluate our deliverables:

- Challenge submission: 5
- Presentation: 5
- Report: 5

4.1 CHALLENGE SUBMISSION

Our classifier achieved the highest multi-class and binary accuracy. Our binary estimation and perplexity were average. Based on the statistically significant difference on the multi-class accuracy, we believe we did excellent in the challenge. We do not believe that the slight hiccup regarding the submission should negatevily affect our grade.

4.2 PRESENTATION

We covered all of the important parts of our work, supplementing our presentation with clear and informative slides. In contrast with most of the pitches we saw from other groups, we met physically for the presentation and recorded ourselves, without simply showing the slides. Our presentation was within the time frame of 3 minutes.

4.3 REPORT

There was a serious effort for the report to be clear and professional, from its visual aspect, to the organization of the content and the explanation of the various concepts. We believe the report manages to explain our work fully, while having enough arguments and sources to back up its claims. It fully reflects our group's hard work on this project.

Overall, we believe we have managed to crack several aspects of the project's task, documenting and explaining each step. There were several insights that arose from our analysis, which allowed our classifier to perform significantly better than the other groups. We have shown in our analysis our in-depth understanding of the whole spectrum of the course's topics, and we were able to apply this knowledge to solve a real problem.