Project 2 - Introduction to Machine Learning and Data Mining

Jens Collatz Laustrup (s204431) Gustav Morgensol (s205428)
Toshini Iyer (s236312)

14/11/2023

Constributions:

	Section 1	**Section 2**	**Section 3**	**Section 4**	**Exam quest.**
s204431	20%	20%	60%	40%	33.33%
s205428	20%	60%	20%	30%	33.33%
s236312	60%	20%	20%	30%	33.33%

1 Regression, part a

We will start by performing linear regression on the SPAMBASE dataset[1].

The main goal of the data set is to predict a spam binary attribute using classification. However, for the regression part, we will attempt to predict one of the continuous ratio variables. In particular we will try to predict the capital_run_length_total attribute, which is the total number of capital letters in the email.

To do this, we will use most of the attributes in the data set, such as frequencies of the words, character counts. The only three that we will not include are capital_run_length_average, capital_run_length_longest and the spam/not spam attribute. This is because we think that the capital_run_length attributes are too similar to the attribute we are predicting, and we would like to predict the attribute without knowing whether the email is a spam email.

The goal of the regression model would therefore be to determine the number of capital letters contained in the email with the help of the attributes mentioned above.

First, we will standardize the data. This will ensure that each feature in our data set has a mean of 0 and a standard deviation of 1.

[1]Hopkins,Mark, Reeber,Erik, Forman,George, and Suermondt,Jaap. (1999). Spambase. UCI Ma- chine Learning Repository. https://doi.org/10.24432/C53G6X.

After standardizing our data, we perform K-fold cross-validation. The goal is to investigate how the model performs for different regularization strengths. We introduce a regularization parameter (λ). We have chosen to consider the range $\lambda = 0, 200, ..., 3000$. That is, we consider values of λ from 0 to 3000 with a gap of 200 between each value. We use 10 fold cross validation. For each value of λ we estimate the generalization error from the cross-validation using the test errors for the 10 splits.

Figure 1 shows the results as a graph of the generalization error as a function of the value of λ.

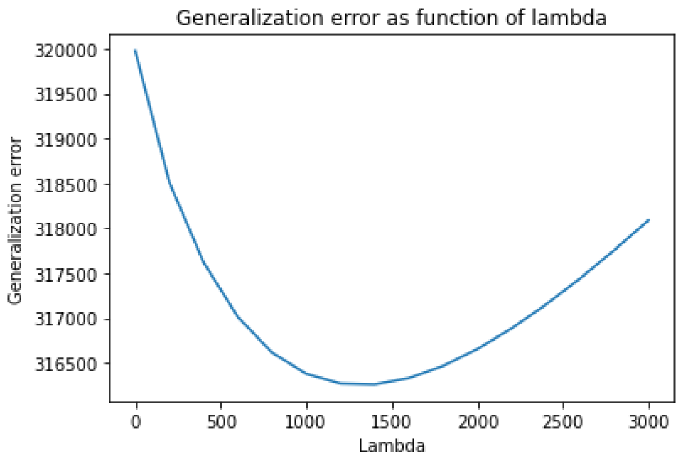

Figure 1: Generalization error as a function of λ for linear regression model.

From figure 1, we can see that initially, the generalization error decreases as λ increases, which indicates that overfitting is being reduced. This leads to better generalization to unseen data. At a specific point in the graph, we reach minimal generalization error. This minimum generalization error is found to be 316260 at $\lambda = 1400$. At this point the model performs the best on test data. Then as the λ further increases the generalization error again begins to increase and the model becomes too simple to capture the underlying patterns in the data (too strong regularization). We also mention that the estimated baseline generalization error (which simply always predicts the mean of the target attribute) is found to be 367922, which is indeed larger than the generalization error found for the linear regression model, as we would hope.

In a linear regression model, the formula for predicting a variable y from m attributes is as follows:

$$y = w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 + \ldots + w_m \cdot x_m$$

Where:

y is the predicted output (in our case, "capital_run_length_total").

w_0 is the intercept term, also called the bias term.

$w_1, w_2, \ldots, w_m$ are the coefficients associated with each attribute.

$x_1, x_2, \ldots, x_m$ are the input attributes (features).

We now train a linear regression model on all data using the best value of λ found previously (1400). Figure 2 shows a histogram with the coefficients found for each of the attributes in the data set by the model.

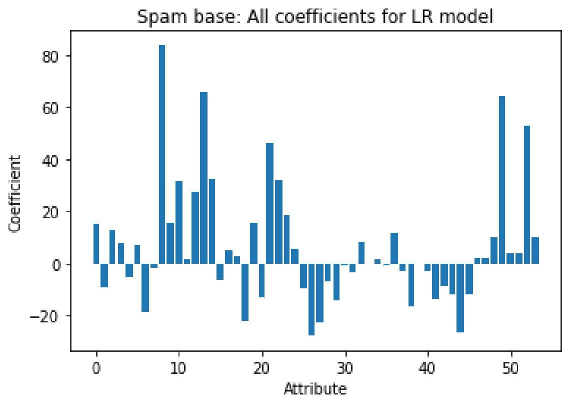

Figure 2: All coefficient for linear regression model trained on all data.

This additionally finds an intercept of 283.289.

The effect of an individual attribute, say x_i, on the output, y, is determined by the corresponding coefficient, w_i. If w_i is positive, an increase in the value of x_i will lead to an increase in the predicted value of y. If w_i is negative, an increase in x_i will lead to a decrease in y.
In order to better get an overview of the most important attributes according to the model, figure 3 shows the 17 coefficients with largest absolute values and the corresponding attributes. These are considered most important by the model.

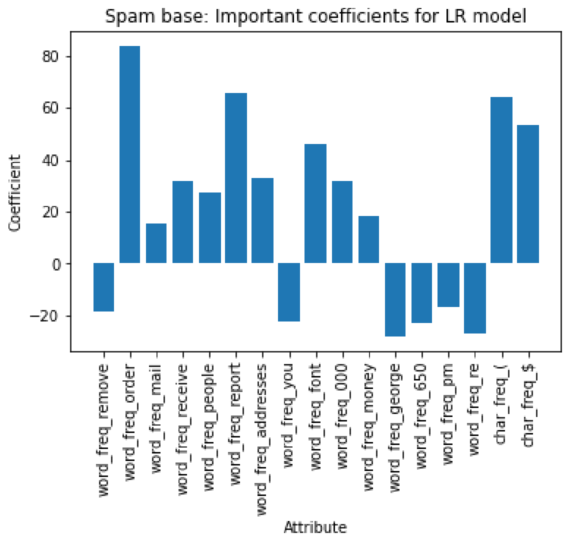

Figure 3: The 17 most important weights for linear regression model trained on all data.

Positive coefficients highlight attributes that indicate a higher number of capital letters in the email, while negative coefficients highlight attributes that suggest a lower number of capital letters in the email. Here, we see that words like 'order' and 'report' as well as the characters '(' and '$' indicate a large number of capital letters. On the other hand, words like 'george' or 'remove' indicate a lower number of capital letters.
Some of the attributes deemed relevant make sense based on our understanding. For example, it seems that spam-like words and characters often indicate a large number of capital letters (such as '$', '000', 'money', etc.). On the other hand, some attributes may indicate that an email is short which in turn should contain a low number of capital letters ('re' and 'pm' for example).

2 Regression, part b

We will now perform two-level K-fold cross validation to compare linear regression, artificial neural networks and a baseline for regression, with different complexity controlling parameters for linear regression and neural networks. We use 10 outer folds and 10 inner folds. We record the mean squared error for each inner split for each parameter for linear regression and neural networks. After the inner loop we estimate the generalization error for each parameter and select the parameter with lowest error as the best parameter for the outer fold. We then test the three models on the outer split using the best values of the parameters.
For linear regression we have used the regularization parameter (λ) with the same tested values as part a (0, 200,..., 3000).
For the artificial neural network we use one hidden layer and the number of hidden nodes (h) as complexity controlling parameter and based on trial runs we test the values 1, 300, 400, 500. We use the transfer function tanh, and no function on the output. Mean squared error is used as loss function to train the neural networks. For the neural network we will train using 10,000 max iterations and 1 replicate for each training.
The best parameter and test error found for each model in each outer fold can be seen in the table below.

Outer fold	ANN		Linear Regression		baseline
i	h_i^*	E_i^{test}	λ_i^*	E_i^{test}	E_i^{test}
1	400	54366	1400	148101	184622
2	500	68803	1200	203153	276603
3	400	321769	600	432955	507950
4	400	132022	1200	269064	308136
5	500	63896	1000	244062	261280
6	400	63928	1200	153008	209476
7	400	174349	1200	341316	371283
8	400	91442	1200	279152	352366
9	500	539200	1200	644190	672409
10	400	264683	1200	456653	532882

We see in the table above that the chosen values of h is 400 or 500 in all cases. This suggests that a high complexity model is beneficial for the problem. The value 400 is most present, which suggests that the model likely starts to overfit above this value.
For linear regression we see that the parameter is chosen around 1000-1400 in almost all cases with 1200 being the most common choice. The model most likely starts to overfit below this value. This result is quite similar to what we found in part a.
We additionally see that ANN has the lowest test error in all cases, which is much lower than the error for the other two models in most cases. The linear regression model additionally has a lower test error than the baseline in all cases. We will now statistically evaluate whether there is a difference in performance between the three models.

For the statistical analysis we will use $\alpha = 0.05$. We use the found errors in the outer splits (the errors seen in the table).
We have chosen to perform tests using setup II in the book (the correlated t-test). That is, the randomness of the training sets is taken into account such that it can be generalized to replicas of the same experiment.
We will test the Null-hypothesis that there is no difference between each pair of the three models. In case of a p-value of less than 0.05 we will reject the null-hypothesis. We will also show 95% confidence intervals for the difference in errors between the models.
Let θ_{ANN}, θ_{LR} and θ_{base} be the test errors for the artificial neural network, linear regression and baseline respectively.
For the difference between ANN and the baseline ($\theta_{ANN} - \theta_{base}$) we get:

$$\text{95\% CI: } [-237363, -143147]$$
$$\text{p-value: } 7.55 \cdot 10^{-6}$$

Given the p value of $7.55 \cdot 10^{-6}$ we reject the null-hypothesis. We conclude that there is a statistical significant difference between ANN and baseline with very high confidence. From the confidence interval we can say with 95% confidence that the error is between 237363 and 143147 lower for the ANN model compared to the baseline.
For the difference between linear regression and baseline ($\theta_{LR} - \theta_{base}$) we get:

$$\text{95\% CI: } [-73050, -28021]$$
$$\text{p-value: } 0.000665$$

Given the p value of 0.000665 we reject the null-hypothesis. We conclude that there is a statistical significant difference between linear regression and the baseline with very high confidence. Based on the confidence interval we can say with 95% confidence that the error is between 73050 and 28021 lower for the linear regression model compared to the baseline.
For the difference between ANN and linear regression ($\theta_{ANN} - \theta_{LR}$) we get:

$$\text{95\% CI: } [-178813, -100626]$$
$$\text{p-value: } 2.03 \cdot 10^{-5}$$

Given the p value of $2.03 \cdot 10^{-5}$ we reject the null-hypothesis. We conclude that there is a statistical significant difference between ANN and linear regression with very high confidence. Based on the confidence interval we can say with 95% confidence that the error is between 178813 and 100626 lower for the ANN model compared to the linear regression model.

We have therefore established that both ANN and linear regression perform significantly better than the baseline. Additionally, we see that the ANN model performs significantly better than the linear regression model, which suggests that a more complex model is suitable for this task. We conclude that it is very likely that the ANN model performs better than both the Linear regression model, and the baseline, and the linear regression model performs better than the baseline.
Based on what we have learned we recommend using a quite complex model for the task such as a neural network with around 400 hidden nodes.

3 Classification

We will now perform classification on the data set. We have chosen to classify the binary spam/not spam attribute based on all other attributes. That is, we plan to carry out the primary aim of predicting whether an email is spam or not based on all attributes. Thus, it is a binary classification problem.

We have chosen to compare a logistic regression model to an artificial neural network model. For the ANN model, we will use one hidden layer with tanh as the transfer function. For the output, we will use the sigmoid function to produce a probability. We will use the binary classification error to train the network. Since it is a binary classification problem, the model will only have one output. The prediction will then be that the email is spam (1) if the output probability is greater than 0.5 and that the email is not spam (0) otherwise. The logistic regression model will also have one probability as the output, which will be interpreted in the same way.
As complexity controlling parameters we will choose the regularization parameter (λ) for the logistic regression model and the number of hidden nodes in the hidden layer (h) for the neural network.
Based on trial runs we have chosen to examine the values $\lambda = 10^i$ for $i \in [-5, 4]$ (where i is an integer). This is a total of 10 different values.
For the neural network we have chosen to examine the values 1, 2, 3 and 10. This is a total of 4 different values. We have chosen not to examine any more values since training a neural network is computationally slow.
For the neural network we will train using 10,000 max iterations and 1 replicate for each training.

We will now perform two-level K-fold cross validation with 10 outer folds and 10 inner folds. For each inner fold, we record the classification error for each of the examined parameters (for both logistic regression and ANN). After the inner loop, we then estimate the generalization error for each choice of the parameters and select the parameter with the lowest generalization error as the best parameter for this inner loop. The three models are then tested on the outer split using the best values found for the two parameters. The best values for the parameters and the test errors for the outer splits can be seen in the table below for each of the three models and each of the outer folds.

Outer fold	ANN		Logistic Regression		baseline
i	h_i^*	E_i^{test}	λ_i^*	E_i^{test}	E_i^{test}
1	3	0.0716	0.01	0.0781	0.390
2	3	0.0957	0.1	0.0957	0.413
3	1	0.0609	0.01	0.0696	0.380
4	2	0.0717	0.1	0.0652	0.402
5	3	0.0652	0.1	0.0587	0.380
6	3	0.0804	0.001	0.0696	0.365
7	3	0.0587	0.1	0.0717	0.376
8	3	0.0630	0.01	0.0717	0.409
9	1	0.0609	0.01	0.0674	0.417
10	2	0.0500	0.1	0.0739	0.407

From the table above we see that ANN has an error rate of around 5% to 10% for the tests. The chosen value of h varies. However, the value h = 3 is chosen the majority of the times. It is also seen that the value 10 is not chosen for any of the folds which suggests that the neural network likely over-fits for this value of the parameter.
For the logistic regression model we see a similar result with an error rate of around 5% to 10%. The chosen λ varies. However we see that only 3 of the 10 possible values of λ are present in the table. Of these, the values 0.01 and 0.1 are by far the most present ones.
The baseline achieves an error rate of around 36% to 42%.
In conclusion, the ANN model and logistic regression model appear to have similar accuracies at around 90% to 95% accuracy, which appears to be much better than the baseline model.

In order to investigate whether there is a performance difference between the three models, we will now show results from performing statistical evaluation for difference in performance between the models. For the evaluation we have reused the same splits, that were used to generate the table above. That is, each model is tested on the same splits.
We have again chosen to perform tests using setup II in the book (the correlated t-test). That is, the randomness of the training sets is taken into account such that it can be generalized to replicas of the same experiment. For all three pairs of the models, we show a 95% confidence interval for the difference in error rate between the two models. The p-value is the p-value for the null hypothesis stating that there is no difference between the error rates. Let θ_{ANN}, θ_{LR} and θ_{base} be the error rates for the artificial neural network, logistic regression and baseline models respectively.

For the difference between ANN and baseline ($\theta_{ANN} - \theta_{base}$) we get:

$$\text{95\% CI: } [-0.348, -0.305]$$
$$\text{p-value: } 7.82 \cdot 10^{-11}$$

Thus, from the confidence interval, we see that there is a 95% confidence that the ANN model has an error rate that is between 0.305 and 0.348 less than the error rate of the baseline. Additionally, since the p-value is extremely low, there is a very high probability that there is a difference in performance between the ANN and the baseline. In conclusion, the ANN most likely performs much better than the baseline.

For the difference between logistic regression and baseline ($\theta_{LR} - \theta_{base}$) we get:

$$\text{95\% CI: } [-0.339, -0.305]$$
$$\text{p-value: } 8.86 \cdot 10^{-12}$$

From this confidence interval, we see that there is a 95% confidence that the logistic regression model has an error rate that is between 0.305 and 0.339 less than the error rate of the baseline. Additionally, the p-value is again extremely low. Thus, there is a very high probability that there is a difference in performance between the logistic regression model and the baseline. In conclusion, the logistic regression model most likely performs much better than the baseline.

For the difference between ANN and logistic regression ($\theta_{ANN} - \theta_{LR}$) we get:

$$\text{95\% CI:} [-0.0147, 0.00601]$$
$$\text{p-value: } 0.367$$

From this confidence interval, we see that there is a 95% confidence that the difference between the error rate of the ANN model and the logistic regression model is between -0.0147 and 0.00601, which includes 0. Thus, there is no significant difference between the models with 95% confidence. The p-value is 0.367 in this case which suggests that there is only a 63.3% chance that the models have a difference in error rates, which is not very significant. In conclusion, we have not found a significant difference between the ANN model and the logistic regression model.
Based on what we have learned we would recommend to use a logistic regression model with $\lambda = 0.1$, since it seems to perform around as well as the neural network model and is much faster to run.

We will now train a logistic regression model and investigate how the model makes a prediction. We will choose the regularization value $\lambda = 0.1$ since this was the most commonly chosen value for the cross validation that we presented in the table earlier.
Since we are not interested in validating the performance in this case, we will simply train the model on the whole data set.
Overall, a logistic regression model works by assigning a weight to each attribute in the data set and an intercept. It then predicts a point by first calculating $\mathbf{x}^T \cdot \mathbf{w} + b$, where $\mathbf{x}$ is the point, $\mathbf{w}$ are the weights and b is the intercept. It then uses a sigmoid function to convert the result into a percentage (between 0 and 1), which is then interpreted as the probability that the point belongs to class 1 (spam).
In order to investigate how the model makes a prediction we will have a look at the weights found after training.
Figure 4 shows the weights found for all attributes. The intercept found is -5.02.

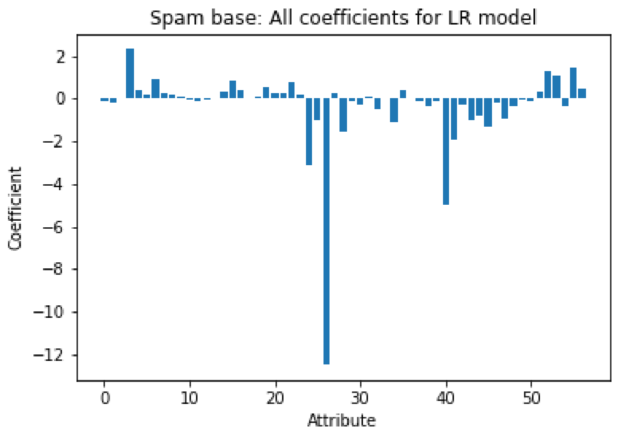

Figure 4: All weights for logistic regression model trained on all data.

As seen, some weights are negative and some are positive. The negative weights are assigned to the attributes that the model considers as signs of non-spam, while the positive weights are assigned to attributes that are considered signs of spam. The weights also have a large difference in magnitude. Attributes with large magnitude weights are considered more important than attributes with low magnitude weights. We now consider only the most important attributes. Figure 5 shows the weights of the 17 most important attributes (that is, with largest magnitude weights). The name of each attribute can be seen on the figure.

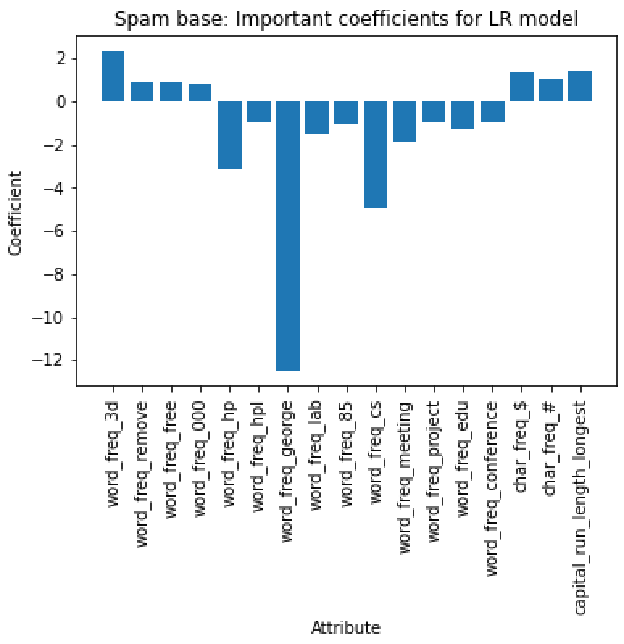

Figure 5: Most important weights for logistic regression model trained on all data.

We can now see which attributes are considered most important for the model. We see that the 10 words "hp", "hpl", "george", "lab", "85", "cs", "meeting", "project", "edu" and "conference" are all considered strong signs that the email is not spam according to the model, since there weights are negative. This makes sense, as many of these words would often be associated with work emails.

We also see that the words "3d", "remove", "free" and "000", and the characters "$" and "#", as well as the longest string of capital letters are strong signs that an email is spam according to the model. This is also what you would expect.

In addition to these weights, the model has assigned the intercept a value of around -5. This suggests, that very short emails (or emails that have low magnitudes for the attributes) would often be labeled as non-spam. For example, an empty email would be considered non-spam. This is the model's way of saying that there must be some sign of the email being spam for it to be labeled as spam.

The goal of this part is to classify the spam attribute, while the goal of the regression part was to predict the number of capital letters in the email. These are quite different goals, so we would in general not expect that the same features are considered important. However, we do see that the words 'remove', '000' and 'george' as well as the character '$' are considered important in both cases, while the rest are different.

4 Discussion

We will now discuss what we have learned from this project.

For the first part of the regression, we investigated linear regression models to predict the number of capital letters in an email. We found an optimal value of the regularization parameter λ to be 1400. Additionally, we looked at the most important attributes for the model and were able to somewhat make sense of these based on our understanding.

For the second part of the regression, we compared linear regression with artificial neural networks for predicting the number of capital letters. Here, we found that linear regression performs significantly better than a baseline, while artificial neural networks with a large number of hidden nodes performed significantly better than both the baseline and the linear regression models. This indicates that it is possible to predict the number of capital letters quite well using machine learning.

For the classification part, we compared logistic regression with artificial neural networks for predicting whether an email is spam. Here, we found that both logistic regression and artificial neural networks perform significantly better than a baseline, but we did not find a significant difference between logistic regression and artificial neural networks. We found an accuracy of around 90% to 95% for both logistic regression and ANNs, which we are quite satisfied with. We additionally looked at the most important attributes for a logistic regression model and were able to make sense of these quite well.

The authors of the data set have provided results from studies [2]. This shows results when performing classification to predict the spam/not spam attribute, as we did in the classification part. Here, they found an accuracy for both logistic regression and neural networks of around 90% to 93%. This matches very well with our result of 90% to 95% for both. This both shows no significant difference between the two and a similar accuracy to what we found. The only difference is that our result has more uncertainty (a larger range of accuracy) than their results.
We have not been able to find any studies performing the same regression task as we have.

5 Exam Questions

1. Option C: At FPR = 1 we have a case where TPR = 0.75, which means that the left-most point must be red, which leaves just option A and C. Additionally, at FPR = 0.75 we have TPR = 0.75, which means that the second point from the left must be black. This leaves us with just option C.

2. Option C: We calculate the impurity gain from the class error as:

$$I(r) - I(v_{x_7=2}) - I(v_{x_7\neq 2}) = (1 - \frac{37}{135}) - 0 - (1 - \frac{37}{134}) \cdot \frac{134}{135} = 0.0074$$

   And thus option C is correct.

3. Option A: Each hidden node contains one weight for each input and one extra weight. Thus, these contain $7 \cdot 10 + 10 = 80$ weights. There are 4 output nodes. Each one has a weight for each hidden node and one extra weight. This is $10 \cdot 4 + 4 = 44$ weights. Thus, the network contains $80 + 44 = 124$ weights in total and A is correct.

4. Option D: Option D is the only option that properly separates the white class from the rest and thus, this must be the correct option.

5. Option C: Both the neural network and the logistic regression model are trained $5 \cdot 4 \cdot 5 + 5 = 105$ times in total each. Each time, they are both trained and tested. Thus, the total time taken is $105 \cdot 25 + 105 \cdot 9 = 3570$ ms.

6. Option B: We calculate the probability of y = 4 for the observation as:

$$\frac{1}{1 + (e^{1.2+2.1\cdot 0.6-3.2\cdot 1.6} + e^{1.2+1.7\cdot 0.6-2.9\cdot 1.6} + e^{1.3+1.1\cdot 0.6-2.2\cdot 1.6})} = 0.73$$

   Since this must be the highest probability for any class the observation will be assigned to class 4.

[2]Hopkins,Mark, Reeber,Erik, Forman,George, and Suermondt,Jaap. (1999). Spambase. UCI Ma- chine Learning Repository. https://doi.org/10.24432/C53G6X.