UPPSALA
UNIVERSITET

MAT-VET-F-22016
Examensarbete 15 hp
Maj 2022

Email classification using machine learning algorithms

Isak Jonsson

UPPSALA
UNIVERSITET

Teknisk- naturvetenskaplig fakultet
UTH-enheten

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

Abstract

Email classification using machine learning algorithms

Isak Jonsson

The goal of this project is to construct a machine learning algorithm that improves over time. This was done by first constructing a dataset that reflects real world messages, that would simulate receiving emails from two different sources. The data set was constructed by combining data from two different online forums. Two application programming interrfaces were used to collect and send data to the program. The dataset was tested on 4 different methods where the best one would be used for the final product. The 4 different methods were: k-nearest neighbors, adaptive boosting, random forest and artificial neural network. All the above methods were tested and tuned to achieve the best accuracy. From the result it became clear that the artificial neural network outperformed the other methods by a large margin and would be most suited for the final product. The final product was an algorithm that would improve over time. This was achieved by using a feedback loop on the new data that was collected over time from the online forums. If the algorithm was sure that a new datapoint was the right class it would incorporate it into the dataset and over time the dataset would grow larger and the algorithm would adapt to new data and trends. The final result became a growing dataset that started on a 1000 data points and ended up at 8464 data points, where the total amount of misclassification ended up at 74.

Handledare: Elvira Boman
Ämnesgranskare: Maria Strömme
Examinator: Martin Sjödin
ISSN: 1401-5757, MAT-VET-F-22016

Table Of Contents

1 Introduction **1**

2 Theory **1**
2.1 String to numbers 1
2.2 Bias–variance tradeo 2
2.3 Cross validation 3
2.4 k-nearest neighbors 3
2.5 Adaptive boosting vs Random forest 4
2.6 Artificial neural network 5
2.7 API 7

3 Implementation **8**
3.1 Python libraries 8
3.2 Implementation of k-nearest neighbors, Adaptive boosting and Random forest 9
3.3 Artificial neural network 9
3.4 Assembling the data 9
3.5 Implementation of the API's 10
3.6 Final implementation 10

4 Results and Discussion **11**
4.1 Resulting dataset 11
4.2 k-nearest neighbors 12
4.3 Adaptive boosting 12
4.4 Random Forrest 14
4.5 Artificial neural networks 16
4.6 Model implementation 18
4.7 Result of final implementation 18

5 Conclusions **20**
5.1 Model conclusion 20
5.2 Feedback loop conclusion 20
5.3 Final conclusion 20

6 Further work **20**

7 Populärvetenskaplig sammanfattning **22**

8 Appendix **22**

1 Introduction

In a more digitizing world, building and constructing models that are designed to separate data into categories has become a relevant design space for di□erent models and algorithms. The goal of this project is to construct a machine learning algorithm that separates emails into two di□erent categories. Furthermore an algorithm will be constructed that will improves over time and become better at categorising emails.

This will be done by first constructing a dataset reflecting real world data, since it will be nearly impossible to collect enough data from a normal email this will be solved by collecting the data from two di□erent online forums. The data collected will be used to train and tune the di□erent machine learning algorithms. The 4 di□erent methods were: k-nearest neighbors, adaptive boosting, random forest and artificial neural network. The best preforming algorithm will be incorporated in the final product. The final product will be a feedback loop machine learning algorithm. Where new data will be presented to the algorithm and tested. If the algorithm is sure that a new data point is the right class it would incorporate it into the dataset and over time the dataset would grow larger and the algorithm would adapt to new data and trends.

The goals of the project is to understand what makes a good machine learning algorithm for email classification and to understand how it can be incorporated into something that grows over time. The project main limitation is time since only 4 di□erent methods will be tested and the growing algorithm will not be able to run for a substantial time.

2 Theory

2.1 String to numbers

In the concept of machine learning, most of the known methods uses numerical numbers as its parameters when classifying. This presents problems when it comes to email classification since it mostly contains strings of letters and numbers. With the help of Scikit-learn library this problem can be solved with the build in function TfidfVectorizer which builds on the theory of TF-IDF transformation. TF-IDF stands for Term frequency inverse document frequency and it transforms text in to a numerical vector. It is built on two concepts, Term Frequency (TF) and Inverse document frequency (IDF).

Term Frequency calculates the occurrence of a specific term in a string and converts it to a matrix whose rows represents the number of strings and its columns is the number of distinct terms throughout all documents. Document frequency calculates the number of a specific term and this tells the frequency of a specific term. Inverse document frequency indicates the weight of the term. It calculates and reduces the terms if it shows up in throughout the data-set. Idf is presented in equation (1)

$$idf_i = log\frac{n}{df_i} \tag{1}$$

where idf_i is the IDF score for specific term i, df_i is the number of is the number of terms found in the data-set, and n is the total number of data-sets. The final TF-IDF score is the combined matrixes of TF and IDF and is calculated with the equation (2). [8]

$$w_{i,j} = tf_{i,j} \times idf_i \tag{2}$$

In this project a mail containing the following message: "What video games can I play with my father? I am 17M and my father is 45, I am wondering what games would be good to play together. I ask because most of the games I play aren39;t his style, He is" would be transformed to a sparse matrix, see figure(1):

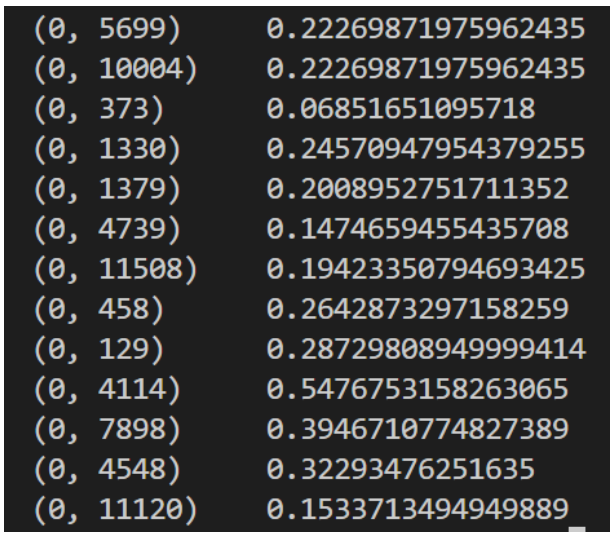

Figure 1: TF-IDF transformation for "What video games can I play with my father? I am 17M and my father is 45, I am wondering what games would be good to play together. I ask because most of the games I play aren39;t his style, He is"

figure(1) shows a sparse matrix where every row is corresponding to a distinct word with a value.

2.2 Bias–variance tradeoff

In machine learning the concept of bias and variance becomes important when discussing a model prediction and how to understand the prediction error (bias and variance). The goal of most machine learning algorithms is to minimize the bias and variance. This is often refereed as the bias–variance tradeo

Bias is the di erence between the correct value and the prediction our model produces. A higher bias leads to a model that oversimplifies the results and pays less attention to the training data. This leads to a higher error on the training and test data.

Variance tells how spread out the fitted data is and a model with higher variance pays more attention to the training data. This leads to lower generalization on data it has not seen

yet. A model with higher variance will preform well on training data but not on test data.

The total error of a model can be calculated with equation(3)

$$Err(x) = Bias^2 + Variance + irreducible\ error \tag{3}$$

Where the irreducible error is the noise of the data. [11]

2.3 Cross validation

When construction a machine leering algorithm some of the data is reserved for training data and the rest is reserved as test data. This will not be an accurate representation for the model accuracy, since di□erent portions of the data-set can be less useful then others. It is here it is good to incorporate a cross validation method that testes the method on the whole data set. In this project the k-fold cross validation method is incorporated.

K-fold divides the data in to k di□erent splits and uses one of the splits as test data and the rest as training data, this is preformed for every split and will result in an even distribution when testing the accuracy for the given method. In this project the chosen k is 10 and the mean accuracy and the standard deviation is calculated for the di□erent methods. [4]

2.4 k-nearest neighbors

One of the methods explored in this project is k-nearest neighbors also known as KNN which is one of the more easily interpreted machine learning algorithms. The method classifies depending on the distance between the test data and the training data and works for both regression and classification. Depending on the chosen k the algorithm calculates the k closes neighbors and makes it's decision depending on what classes it most closely resembles. In this project this would resemble the di□erent words (terms). The distance between the test data and the training data is the Euclidean distance and is calculated with the equation(4)

$$D = \sqrt{(\mathbf{x}_1 - \mathbf{x}_2)^2 + (\mathbf{y}_1 - \mathbf{y}_2)^2} \tag{4}$$

Where D is the Euclidean distance, x_1 and y_1 is the coordinates for one of the data point and x_2, y_2 is the coordinates for the second data point. This distance is calculated for all the data points and helps the algorithm classify depending in the distance. [3]

With help of Scikit-learn library the algorithm can be executed with the help of python using the function *KNeighborsClassifier(n_neighbors = k)*. The parameter k was calculated with help of k-fold cross validation and depending on the k di□erent accuracy's could be achieved, this will be presented in the result and generally a lower k means higher variance and a higher k means a higher bias. [7]

2.5 Adaptive boosting vs Random forest

Adaptive boosting and random forest builds on the theory of bagging and boosting which is methods were the data set is tested multiple times. Bagging is where the data set is split in to n di□erent splits and all the di□erent splits are trained and the result become the mean of all the di□erent splits. Boosting is where the data set is trained and improved over n iterations. Depending on the errors from the previous model the algorithm learns from its mistakes.

Both adaptive boosting and random forest uses decision tree as its method of learning. Which is a method where the data uses tree structures as it's base of classifying, as an example see figure (6).

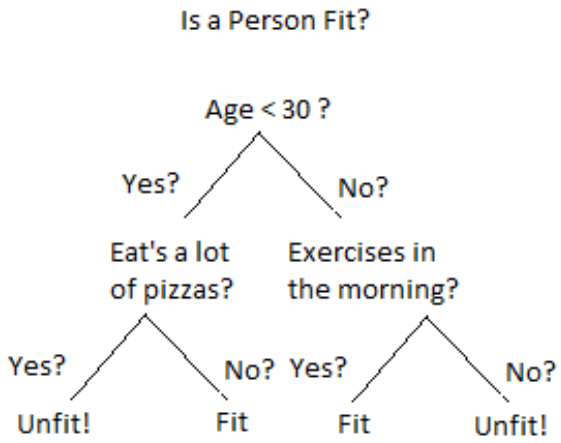

Figure 2: Simple tree structure [5]

For this project di□erent tree depths will be tested and compared. The split are determined by minimizing the gini index, which is calculated by equation(5).

$$\text{Gini index} = \sum_{m=1}^{M} \hat{\pi}_{lm}(1 - \hat{\pi}_{lm}) \tag{5}$$

Where M is the total number of classes and π_{lm} is the probability of picking a certain class. In the case of Adaptive boosting the weights are calculated with the help of equation (6).

$$\text{Weight} = \text{learning rate} * log(\frac{1 - error}{error}) \tag{6}$$

Where the error is the percentage of errors in the prediction and the learning rate is the original weight of the tree. [5]

Random forest and adaptive boosting are implemented with the help of Scikit-learn library and with help of the function RandomForestClassifier() for random forest and AdaBoostClassifier() for adaptive boosting. Hyper parameters are n estimators, random state for random forest and n estimators, learning rate for adaptive boosting. Both algorithms are using DecisionTreeClassifier() as its classification model with different depth. N estimators are how many different splits/estimators, random states controls the random states, learning rate is the weight applied to each classifier. The different hyper parameters are tuned later to maximise the accuracy.

In general random forest reduces the variance in the model and adaptive boosting reduces the bias in the model. Depending on the result it will become clear witch method is most suited for the data set. [7] [1]

2.6 Artificial neural network

In machine learning the section on artificial neural network is a subset of methods that mimics the structure of how a real human brain works and mimics the biological features that are present in the brain. A artificial neural network is built on having a input layer that is follow by one or more hidden layers and finally a output layer, see figure(3). Each layer contains a set number of nodes that is a computational unit that is built up on input values, output values, weighted input connections and a transfer function. Since this project builds on classifying two different types of emails the last output layer will only have two nodes.

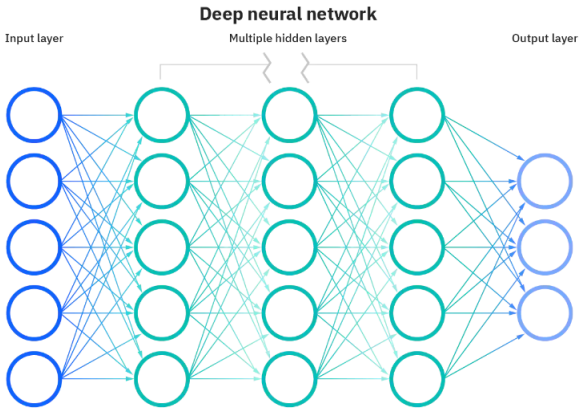

Figure 3: Example of a artificial neural network [6]

The Input layer is the first layer and has a set number of nodes and are corresponding to each training value for the data. Each node in each layer is connected to each node in the second layer. When training the network the weights of each node are changed to reduce the cost function. The output and weights are calculated with equation(7) and equation(8),

$$\sum_{i=1}^{m} w_i x_i + \text{bias} = w_1 x_1 + w_2 x_2 + w_3 x_3 + \text{bias} \tag{7}$$

$$\text{Output} = f(x) = \begin{cases} 1, & \text{if } \sum_{i=1}^{m} w_i x_i + b_i \geq 0, \\ 0, & \text{if } \sum_{i=1}^{m} w_i x_i + b_i < 0 \end{cases} \tag{8}$$

Where i is the index of the sample and m is the number of samples. In equation(7) the w_i describes the weights that are tuned during training and the final output in equation(8) is either a 1 or a 0, this could be interpreted as if the node is on or o. When the data passes trough a node the given weights are applied and the output is passed through a given activation function. If the values sent to the node satisfies the given requirements the node sends data to the next layer. Each nodes input values are determined on its predecessors. This structure is defined as a $feedforward$ network and is the structure that will be used in this project.

When training the model the goal is to maximise the accuracy and evaluate it with help of a cost function. In this project di erent cost functions will be tested and compared. The cost functions that will be tested are: mean square error (equation(9)), Poisson (equation(10)) and Binary crossentropy (equation(11))

$$\text{Mean square error} = \frac{1}{2m} + \sum_{i=1}^{m} (\hat{y}_i - y_i)^2 \tag{9}$$

$$\text{Poisson} = \frac{1}{m} \sum_{i=1}^{m} (\hat{y}_i - y_i log(\hat{y}_i)) \tag{10}$$

$$\text{Binary crossentropy} = \frac{1}{m} \sum_{i=1}^{m} (y_i - log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)) \tag{11}$$

Where i is the index of the sample, $\hat{y}$ is the predicted outcome, y is the real value and m is the number of samples. These cost functions are used together with gradient descent find local minimum. Gradient descent has the equation(12)

$$a_{n+1} = a_n - \lambda \nabla F(a_n) \tag{12}$$

Where a is a point on the function and λ is the step size. [6]

In this project the Keras library will be implemented and optimised. Where the the model will be defined as $model = Sequential()$ and the layers will be inputted as

$model.add(Dense(7490, activation = x))$. Where x is the activation function, the two activation function used in this project is sigmoid and relu. Sigmoid is described in equation(13) and relu in equation(14). [2]

$$f(x) = \frac{e^x}{e^x - 1} \tag{13}$$

$$f(x) = max(0, x) \tag{14}$$

The two activation function are plotted in figure(4)

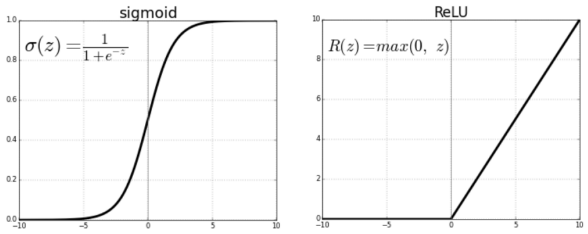

Figure 4: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

The model will be trained using epochs and batch size. One Epoch is when all the training data is trained on one time and passed forward and backwards through the artificial neural network. The network cant be trained on all the data at once, therefor the data is sent through the network in batches, this is called batch size. For this project the model with be trained on with 20 epochs and a batch size of 10. That in other word mean that every data point will be trained on exactly 20 times and every epoch will use 10 data points at a time when training. [10]

2.7 API

API stands for Application Programming Interface and is a software that connects an application to a server (website). For the Gmail API an OAuth agreement has to be signed and is a standard access delegation that gives the user permission to share data between the two programs. The API's used in this project is a REST API and stands for representational state transfer API. A REST API Uses HTTP requests, which i other words mean that the API sends a request to the server where the data is stored and extracting data like reading it, updating and more. In this project reading the data will be the main focus since the REST API will mainly focus on reading emails from a certain email account. Rest API uses a GET request to collect data. Which in other words mean requesting data from a specific server. The data can be transferred in several di□erent formats, JSON is one of the more popular ones and is the one that Gmail API builds on. JSON is a compact text based format that is used when data is exchanged. [9]

3 Implementation

3.1 Python libraries

For the implementation of the k-nearest neighbors, Adaptive boosting and Random forest algorithms the Scikit-Learn Python library was used. Scikit-Learn is a free library designed for machine learning and is used extensively through out the project. Some of the more used function are:

- Train test split: Witch is used to divide the test data in to training data and test data.
- KFold: Kfold is used for cross validation.
- TfidfVectorizer: Is used for converting string to numbers.
- AdaBoostClassifier: Is used for the Adaptive boosting algorithm.
- RandomForestClassifier: Is used for the Random forest algorithm.
- DecisionTreeClassifier: Is used for the Adaptive boosting and Random forest algorithm.
- KNeighborsClassifier: IS used for the KNeighborsClassifier algorithm.

For the artificial neural network the TensorFlow-Keras library was implemented. Keras is a free library designed for artificial neural network and some of the more notable functions are:

- Sequential: Is the model name for the model used for this artificial neural network.
- Dense: Is the layers that are used in between the input and output layers.
- Dropout: is a layer that dropout some of the data to avoid overfitting.

Other notable Python libraries used were Panda and Numpy. Pandas is used to read the data from csv files and and convert it into the Panda format. Numpy is used to calculate some of the necessary calculations in the code. Pandas is allso used to transfer some of the data into other files. Notable functions are:

- Read csv (Pandas): Lets the program read csv files
- DataFrame (Panda): converts the data into a format that can be transferred to a csv file

To extract the emails from a specifics email address, the Google python library had to be implemented. Google python library uses the credentials that are given when signing up for the google cloud service and gives the code access to the specific email address and uses it with the help of a build function to extract the email messages. Notable functions are:

- Credentials: Checks the credential for the given email.
- build: Builds the model that extracts the email form the given email.

3.2 Implementation of k-nearest neighbors, Adaptive boosting and Random forest

When implementing a machine learning algorithm the first step is to process the data and split it into training and test data, this was done with help of the panda library and it's read csv function and the train test split function. The data used for this part is displayed in the appendix(Originalmail.csv). When testing the di□erent parameters the kfold function is used instead of train test split when splitting the data. The text strings in the emails are transformed into numerical vectors and after that the model is initialize and trained on. The model is thereafter complete and the training data is run through the model and a classification accuracy can be calculated.

The hyper parameters are calculated with taking the mean accuracy after running it through out the data with help of k-fold. For the di□erent methods this correlates to: value of neighbors (k) for k-nearest neighbors, n estimators, random state for random forest and n estimators, learning rate for adaptive boosting. For both Random forest and Adaptive boosting di□erent tree depths are also tuned.

3.3 Artificial neural network

To construct the artificial neural network the first step is to set up the data and split it into training and test data. This was done with help of the panda library's read csv function and the train test split function and the TfidfVectorizer to convert the emails to numerical matrixes. The network is constructed by initialising the model and adding the necessary layers. The first layer needs to have the same input dimensions as the amount of emails and the last layer can only have two since there is only two classes. The layers in between is what constructs the model and decides how well it will preform. The di□erent parameters tuned in the layers are activation function, cost function, number of in between layers and number of nodes. The final step to create the artificial neural network is to compile the model and train it on the training data. The training accuracy and the model loss will be plotted with the help of matplotlib and di□erent configurations of layers and loss functions will tested to achieve the highest possible accuracy.

3.4 Assembling the data

To simulate how a real world email classifier would work, real data would be needed and created. This was done with the help of two APIs. The Google Gmail API was used to extract the messages from a email and a IFTTT script that connected the email to two di□erent forums that would simulate receiving two di□erent types of emails. The di□erent forums were both from the website Reddit, which is one of the biggest online forum. The two di□erent forums were r/worldnews and r/gaming.

- r/worldnews: Russia cuts gas to Poland, Bulgaria, West vows arms for Kyiv | AP News
- r/worldnews: France39;s Atos moves Russian services to India and Turkey amid Ukraine war
- r/gaming: Should I buy a Nintendo 64 to play games with friends?
- r/gaming: I still love how BoTW's 2014 demo looked back then. Wish they could keep it that way.

This data was collected until a dataset of around 1000 samples was assembled. The dataset distribution was 50/50 and the data would be used to tune the di□erent models.

3.5 Implementation of the API's

Googles Gmail API was used to extract emails from a specified email address. This was done by first enable the Google cloud service Gmail API service. To enable the Gmail API to share it's data a OAuth delegation had to be created that had the connecting emails within it. A Python script could be created that used the OAuth credentials to extract the emails from the given Gmail address. The emails was later transferred to a separate csv file.

IFTTT is a website that works like an API between di□erent websites. For this project the IFTTT API was used to extract Reddit posts and sending them to a specific email address. This was done by creating a short script that told the API to send an email everytime a post was created on r/worldnews or r/gaming.

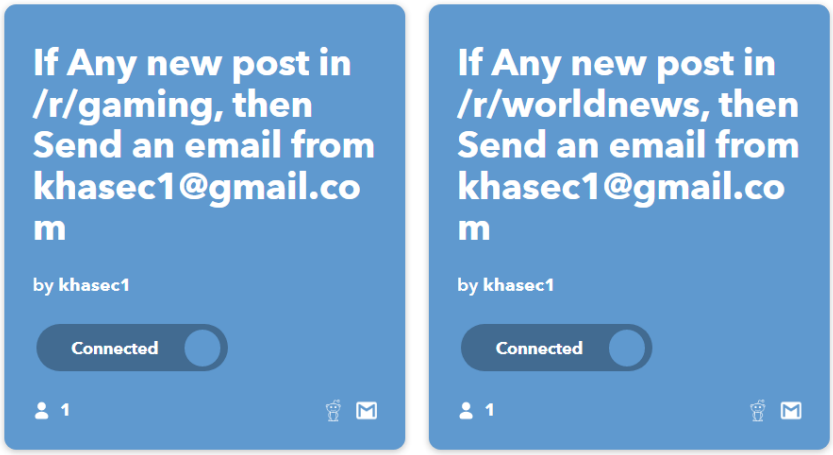

Figure 5: The two IFTTT scripts

3.6 Final implementation

To simulate how a growing network would work the final task of the project is to set up a machine learning algorithm that grows as it assembles more data. This will incorporate all

of the above parts and the final result will be a network that grows and becomes better over time. This will be done by first establishing the most suitable machine learning algorithm. After testing all the above algorithms the method that yields the best accuracy will be Incorporated in the feedback loop. The Google gmail API and the IFTTT API will supply the new data for the algorithm. The model will be trained on the original data and the new data will be tested on the model, if the model is determined with a certain thresholds that the new data is r/worldnews or r/gaming it will incorporate it into the original dataset. The correct class for every data point will be stored so future testing can be done.

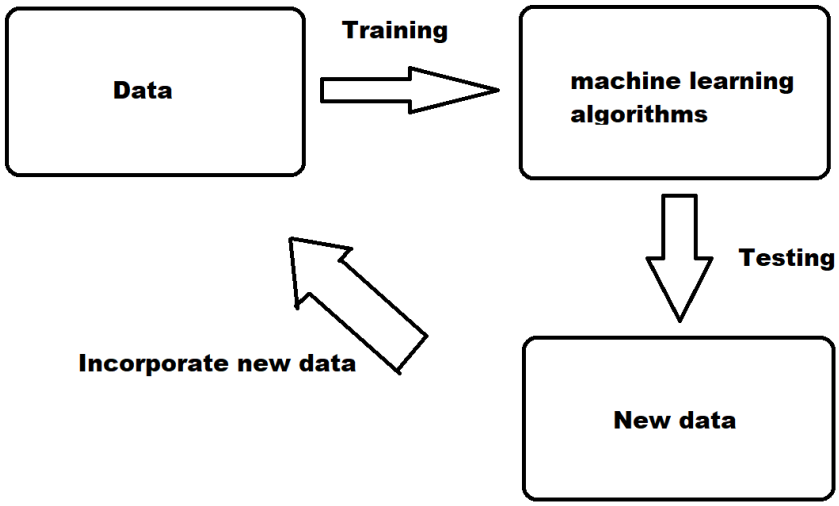

Figure 6: Feedback loop machine learning

The algorithm will be uploaded to a server where the dataset can grow and become larger. After 2 weeks the dataset will be tested and a conclusion for how well it preformed will be displayed in the result. The data set will also be tested against the same algorithm without the feedback loop to conclude if it decreases the number of misclassifications.

4 Results and Discussion

4.1 Resulting dataset

After using the Gmail API combined with the IFTTT API the resulting dataset is displayed in the appendix and some of the more notable features are:

- 1000 data points
- 508 data points from r/worldnews
- 492 data points from r/gaming

4.2 k-nearest neighbors

The k-nearest neighbors algorithm was tested for di□erent values of k and the resulting accuracy's are displayed in figure(7) and table(1)

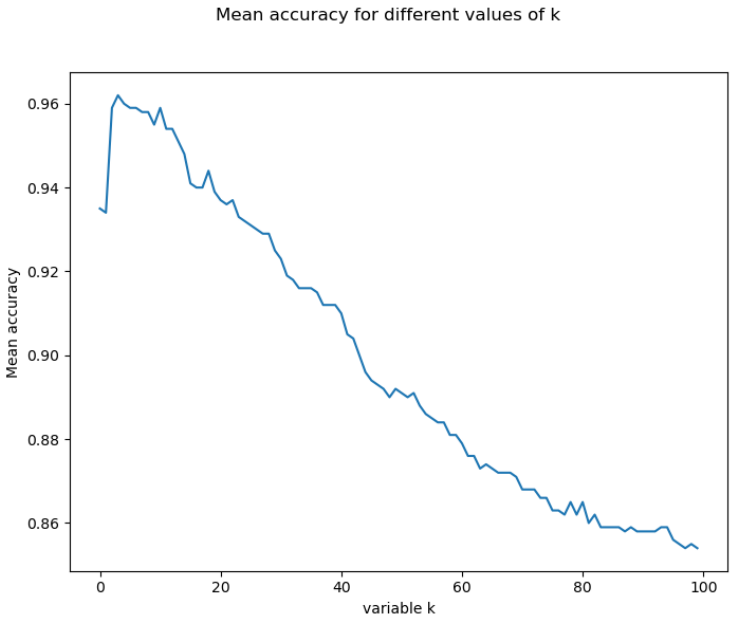

Figure 7: Mean accuracy for di□erent values of k

Table 1: Best value of k and it's mean accuracy and mean standard deviation.Using k-fold = 10

Value of k	Accuracy	STD
k = 4	0.962	0.018868 height

Setting up the model with $k = 4$ and using 80 % of the data as training data, this yields the confusion matrix:

		True diagnosis		
		r/gaming	r/worldnews	Total
Screening test	r/gaming	87	10	97
	r/worldnews	5	98	103
	Total	92	108	200

Form this results it is clear that the model is not bias toward any of the di□erent classes. The final result of the model is a misclassification of 15 out of 200.

4.3 Adaptive boosting

To achieve the most accurate model as possible the parameter hyper tuning was done with the help of k-fold with 100 di□erent folds since the dataset is only 1000 data points. The

3 hyper parameters tuned were max depth, n estimators and learning rate. The result of the hyper tuning is displayed in figure(8)(9)(10)

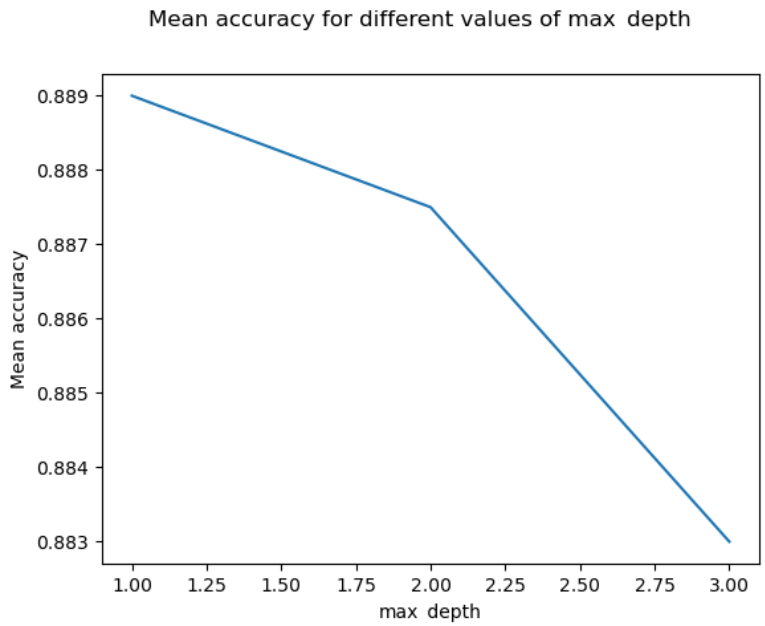

Figure 8: Mean accuracy for di□erent value for max depth

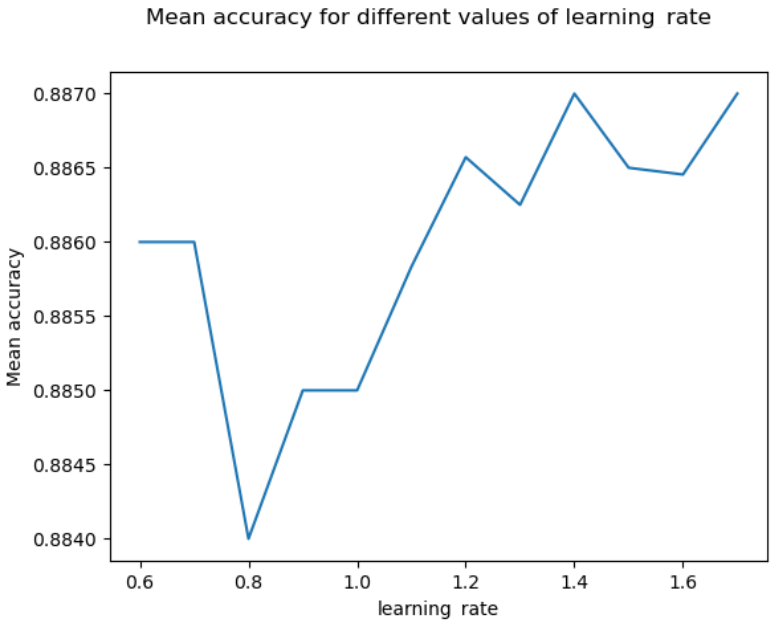

Figure 9: Mean accuracy for di□erent values for learning rate

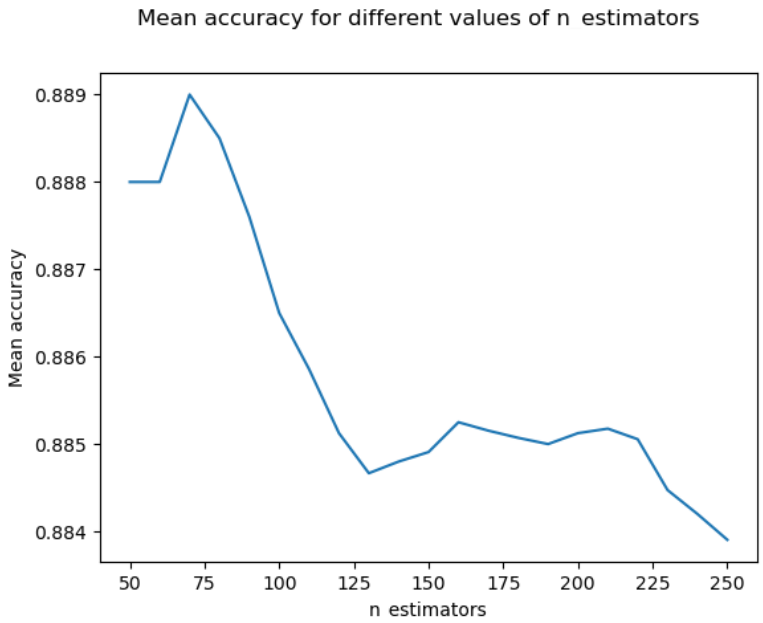

Figure 10: Mean accuracy for di□erent values of n estimators

From the hyper parameter tuning the values chosen for the model were: max depth $=$ 1, learning rate $=$ 1.4 and n estimators $=$ 70. The result of this model is displayed in table(2).

Table 2: Best value for n estimator, mac depth and learning rate and it's mean accuracy and mean standard deviation. Using k-fold $=$ 10

n estimators	max depth	learning rate	Accuracy	STD
70	1	1.4	0.878	0.0632

Setting up the model with the hyper parameters tuned and using 80% of the data as training data, this yields the confusion matrix:

		True diagnosis		
		r/gaming	r/worldnews	Total
Screening test	r/gaming	89	23	112
	r/worldnews	3	85	88
	Total	92	108	200

The result from the confusion matrix shows that there is some bias in the results since it wrongly classifies 23 data points as r/worldnews. The final result of the model is a misclassificationsification of 26 out of 200.

4.4 Random Forrest

To achieve the most accurate model as possible the parameter hyper tuning was done with the help of k-fold with 100 di□erent folds. The 3 hyper parameters tuned were max

depth, n estimators and max samples. The result of the hyper tuning is displayed in figure(11)(12)(13)

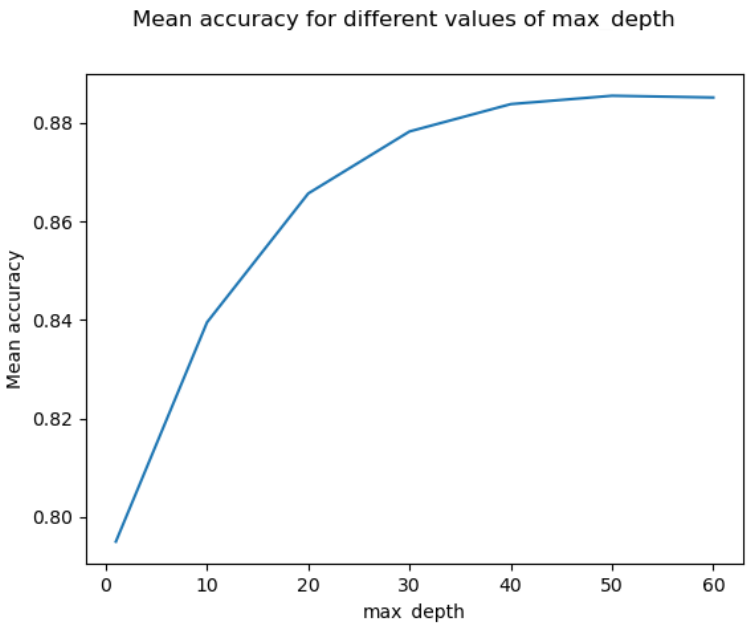

Figure 11: Mean accuracy for di□erent value for max depth

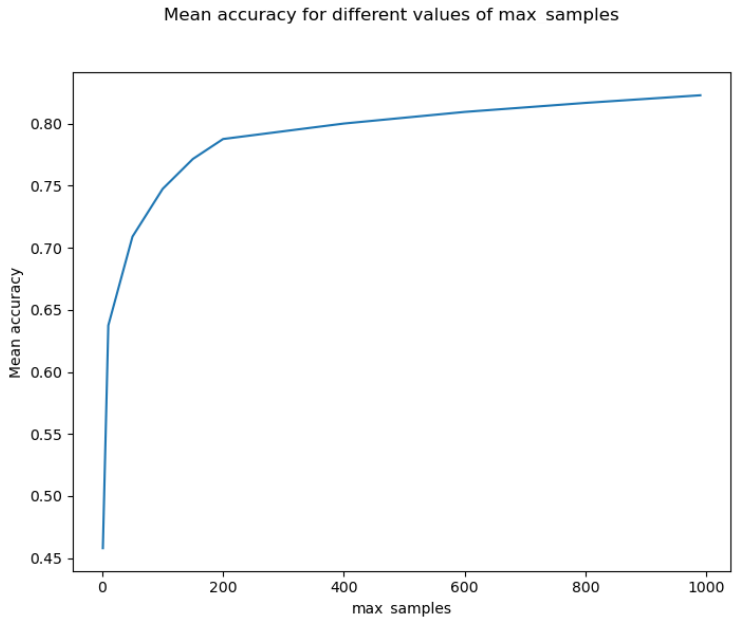

Figure 12: Mean accuracy for di□erent values for max samples

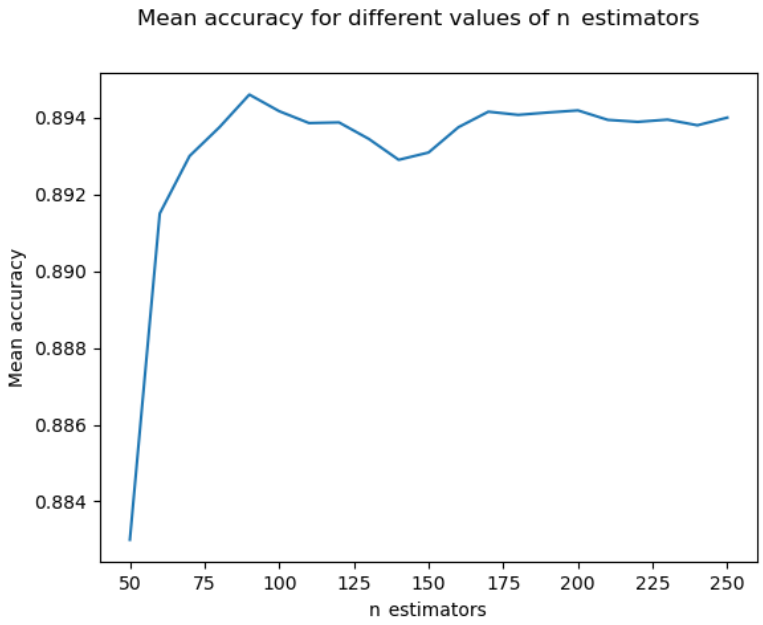

Figure 13: Mean accuracy for di□erent values of n estimators

From the hyper parameter tuning the values chosen for the model were: max depth = 50, max samples = 900 and n estimators = 90. The result of this model is displayed in table(3).

Table 3: Best value for n estimator, mac depth and learning rate and it's mean accuracy and mean standard deviation. Using k-fold = 10

n estimators	max depth	max samples	Accuracy	STD
90	50	900	0.881	0.05147

Setting up the model with the hyper parameters tuned and using 80% of the data as training data, this yields the confusion matrix:

		True diagnosis		
		r/gaming	r/worldnews	Total
Screening test	r/gaming	89	26	115
	r/worldnews	3	82	85
	Total	92	108	200

The result from the confusion matrix shows that there is some bias in the results since it wrongly classifies 26 data points as r/worldnews. The final result of the model is a misclassificationsification of 26 out of 200.

4.5 Artificial neural networks

Constructing the most accurate artificial neural network consists of testing allot of di□erent configurations. The di□erent configurations tested were: Di□erent loss function, activation

function, Number of layers and total number of nodes. Di□erent configuration are tested using 20 epochs and 10 as batch size and the result displayed in table(4)

Table 4: Di□erent configuration of the artificial neural network and it's resulting accuracy

Activation function	Cost function	layers	Nodes	Accuracy	STD
sigmoid	binary crossentropy	1	383,901	0.975	0.0988
sigmoid	binary crossentropy	2	404,101	0.975	0.1307
relu	binary crossentropy	1	404,101	0.975	0.129
sigmoid	Poisson	1	394,001	0.975	0.581
sigmoid	Mean Squared Error	1	394,001	0.975	0.0239
sigmoid	Mean Squared Error	1	39,581	0.975	0.0239
sigmoid	Mean Squared Error	1	39,581	0.975	0.0241
sigmoid	Mean Squared Error	1	7,717	0.980	0.099
sigmoid	Mean Squared Error	2	10,679,001	0.460	0.540

From the result in table(4), the best configuration for the artificial neural networks were using: activation function: sigmoid, cost function: Mean Squared Error, number of layers: 1, total number of nodes: 7,717, accuracy: 0.980 and std: 0.099. Setting up the model using 80% of the data as training data and training the model using 20 epochs and 10 as batch size yields the figure(15)(14) and the confusion matrix (4.5)

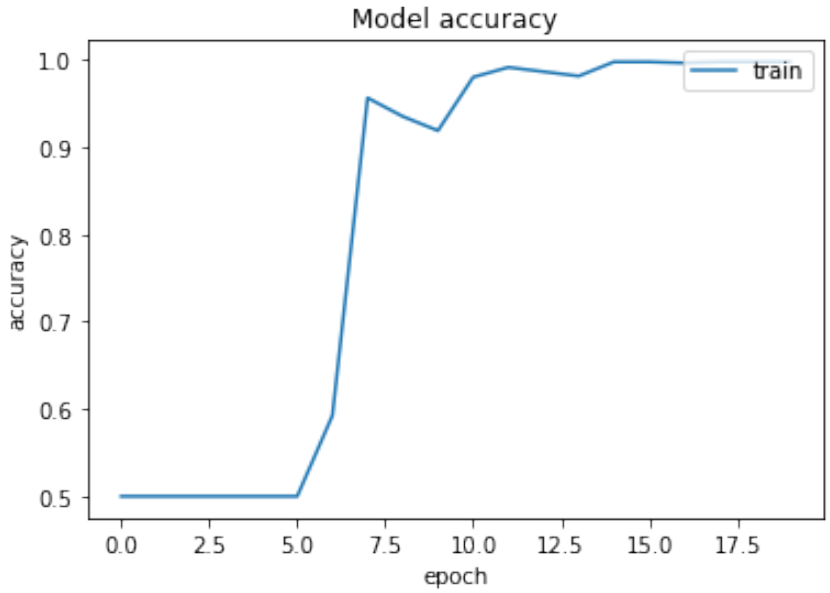

Figure 14: Mean accuracy for di□erent epochs

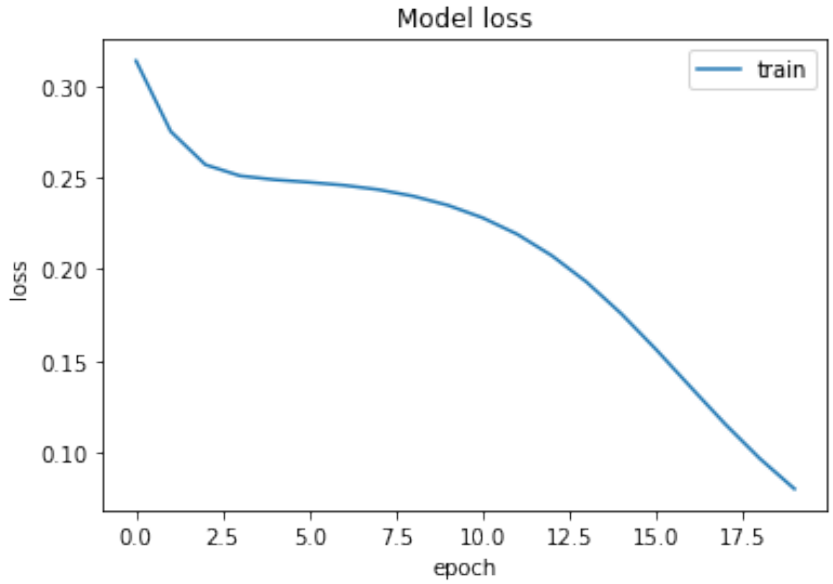

Figure 15: Mean loss for di□erent epochs

		True diagnosis		
		r/gaming	r/worldnews	Total
Screening test	r/gaming	89	1	90
	r/worldnews	3	107	110
	Total	92	108	200

Form the confusion matrix is clear that the model correctly classifies and is the most accurate of all the above models. The result is that the model classified 197 out of 200 correctly.

4.6 Model implementation

After testing all the above methods it is clear that the artificial neural networks outperformed the other two methods. And will be incorporated in the final feedback loop machine learning. Other notably features of the results is that the adaptive boosting and the random forest preformed likewise and showed that the there were no notable bias or variance in the decision tree model, furthermore the k-nearest neighbors algorithm preformed surprisingly well and shows that the data set was simple and easy to separate. Since $k = 4$ preformed the best for the k-nearest neighbor model had a larger variance and a lower bias.

4.7 Result of final implementation

The original dataset used in the feedback loop was the same as the one that the model was tuned and trained on. With the help the API's the dataset grew to 4231 after 1 week of it running and the total number of misclassifications that ended up in the data set were 29. To get a better understanding if the data converges a plot of the total misclassifications over iterations is displayed in figure(16)

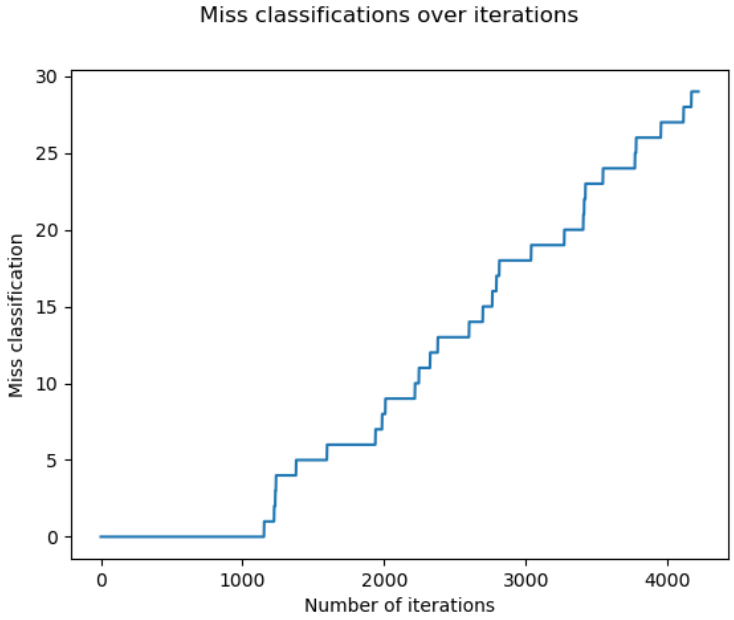

Figure 16: misclassifications over iterations

from figure(16) it is clear that the misclassifications can be deducted as noise since it only makes up 0.7% of the data and the figure looks like to be linear. After 2 weeks the dataset grew to 8464 and the total number of misclassifications s grew to 74 and gave the figure(17).

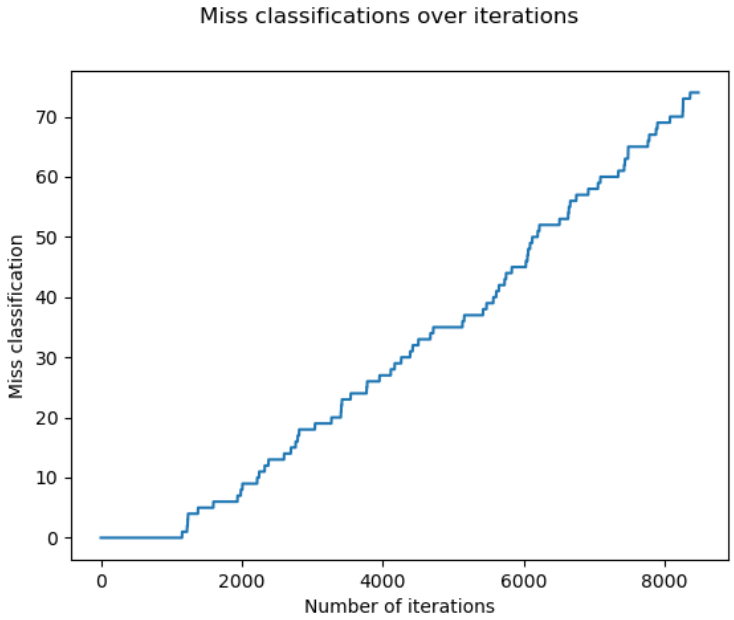

Figure 17: misclassifications over iterations

From figure(17) it is also clear that the misclassifications can be deducted as noise and the total number of errors are now 0.8% and it still looks to be linear.

When testing the algorithm against itself without a feedback loop, it produced the following result displayed in figure(5).

Table 5: Total number of misclassifications for two di□erent Models

Feedback loop	Total number of misclassifications
yes	74
no	143

From figure(5) it is clear that the incorporated feedback loop archived a lower misclassifications rate then without it. The result is around 50 % less misclassifications and shows that the feedback loop is something that is necessary when classifying emails.

5 Conclusions

5.1 Model conclusion

The final result of all the di□erent machine learning methods showed that using machine learning algorithms is a suited method to classify emails and the results showed that the most suited method was the artificial neural networks and it followed the theory since generally it should preform the best out of the 4 methods. The two methods that stood out was random forrest and adaptive boosting since they preformed underwhelming compared to the other methods. This could be a result of the dataset being to small and future testing would be needed to conclude if random forrest and adaptive boosting would be suited for email classification.

5.2 Feedback loop conclusion

The results for the feedback loop machine learning was promising for future implementations since the result showed that it did not converge towards one of the di□erent classes and the total number of misclassification was significant lower then when using the same model without a feedback loop. This proves the concept of using the feedback loop and letting the program run on it's own on a server showed a promising solution for how it would be implemented on a email service.

5.3 Final conclusion

The results showed that machine learning was a promising tool when classifying emails and the results showed that it allso could be incorporated into a final product that grew and became more intelligent as time goes on.

6 Further work

This project explores some of the adaptations of machine learning algorithms and it is clear that there are many avenues that is yet not explored. Some notable actions that

can be done in the future is:

- Add more classes to explore how it e□ects the modules and explore how it would e□ect the growing network. Adding more classes would greatly improve the complexity of the models and make the final product harder to construct.
- Test more machine learning algorithms. There are many more machine learning algorithm that would be needed testing to come to a conclusion for what the best algorithm is for classifying emails.
- Start with a bigger or smaller initial dataset. To truly understand the conversion rate of the growing network di□erent initial data sets would be needed to be tested.
- let the feedback loop machine learning run for a longer time. The final result for the growing network was only run for 2 weeks and could be explored further.
- Using BERT instead of TF-IDF transformation, BERT is a already trained model that gives numerical values to words. This could prove to be a better solution then using TF-IDF transformation.

References

[1] Lilly Chen. *Basic Ensemble Learning (Random Forest, AdaBoost, Gradient Boosting)- Step by Step Explained*. URL: https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725.

[2] Francois Chollet et al. *Keras*. 2015. URL: https://github.com/fchollet/keras.

[3] Antony Christopher. *K-Nearest Neighbor*. URL: https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4.

[4] *Cross-validation: evaluating estimator performance*. URL: https://towardsdatascience.com/understanding-the-bias-variance-tradeoff-165e6942b229.

[5] *Decision Trees for Classification: A Machine Learning Algorithm*. URL: https://www.xoriant.com/blog/product-engineering/decision-trees-machine-learning-algorithm.html.

[6] *Neural Networks*. URL: https://www.ibm.com/cloud/learn/neural-networks.

[7] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[8] Luthfi Ramadhan. *TF-IDF Simplified*. January 20, 2021. URL: https://towardsdatascience.com/tf-idf-simplified-aba19d5f5530.

[9] *REST APIs*. URL: https://www.ibm.com/cloud/learn/rest-apis.

[10] SAGAR SHARMA. *Epoch vs Batch Size vs Iterations*. URL: https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9.

[11] Seema Singh. *Understanding the Bias-Variance Tradeo* . URL: https://towardsdatascience.com/understanding-the-bias-variance-tradeoff-165e6942b229.

7 Populärvetenskaplig sammanfattning

I en mer digitaliserad värld där mycket av kommunikationen sker genom internet är det allt mer viktigt med moduler och algoritmer som kan separera och klassifiera inehåll. Varje dag sickas det milljontalls medelanden och dessa medelande kan med hjälp av olika program separeras i olika kategorier. Företag som Google och Microsoft har lagt mycket resurser på att tillverka algoritmer som kan särskilja innehåll och klassificera emails. I denna kamp har statistisk maskininlärning varit i största fokus och det finns alltid plats för förbättringar.

Maskininlärning är ett verktyg som andvänds för att separera data och i detta projekt kommer detta andvändas för att separera emails i två olika klasser. Datan är tagen från två olika online formum(reddit) och de två olika forumen var r/worldnews och r/gaming. I projektet testades 4 olika metoder som var följande: k-nearest neighbors, adaptive boosting, random forest och artificial neural network. Resultatet visade på att artificial neural network kunde klassifiera datan med minsta felmarginal. Projektet utbyggdes med att bygga ett program som kunde växa och bli bättre över tid. Detta gjordes med att andvända den besta maskininlärnings modulen och sätta upp en feedback loop. Resultatet blev ett program som gissade på ny data som kom in och om den var tillräckligt säcker så uppdaterades programet. Detta gjorde programet smartare och över tid blev den bättre på att klassifiera data som den alldrig tidigare har sett.

8 Appendix

Github repository for code and data.
https://github.com/khasec/Email-classification-using-machine-learning-algorithms-Appendix