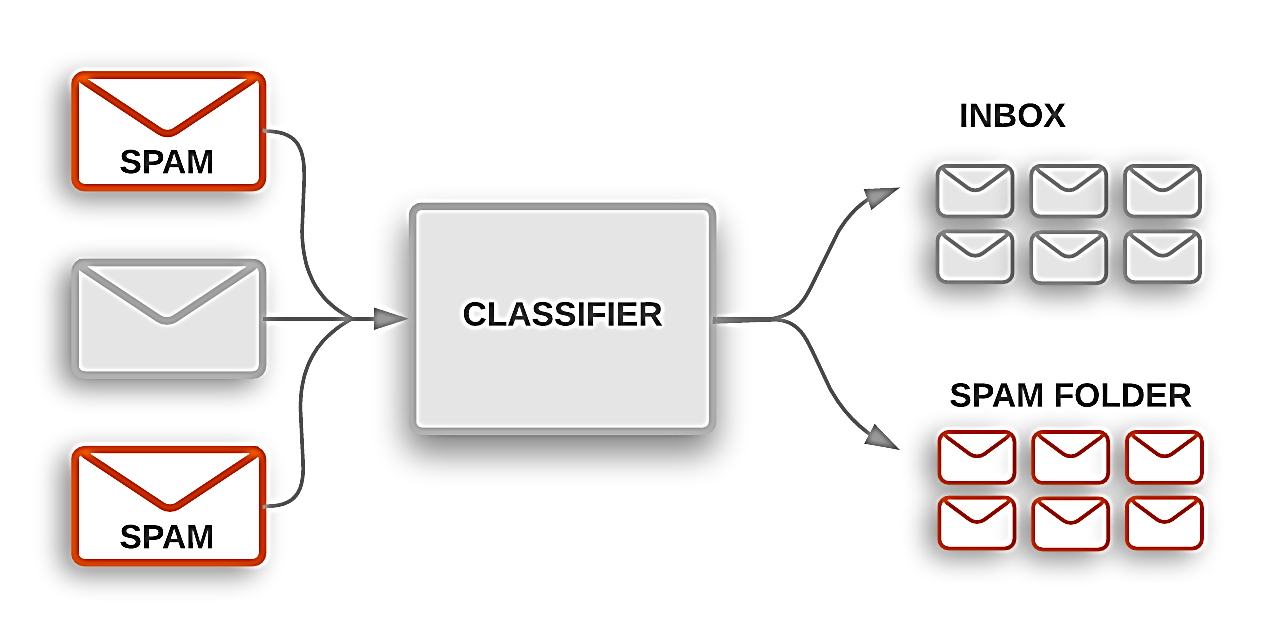
**ML METHODS FOR SPAMSMER AND NON-SPAMSMER E- MAIL**

**CLASSIFICATION**



**Abstract**

The expanding volume of spontaneous mass email (otherwise called spams) has created a requirement for reliableanti spams channel. Machine Learing method now days used to consequently channel the spams email in an extremely operative rate. In this paper we audit the absolute most mainstream Machine Learning approaches ( Bayesian N/w, k-nearest neighbors, ANNs, support vector machine, Artificial invulnerable framework ) and of their appropriateness to the problem of spams Email arrangement. Portrayals of the calculation are demonstrated.

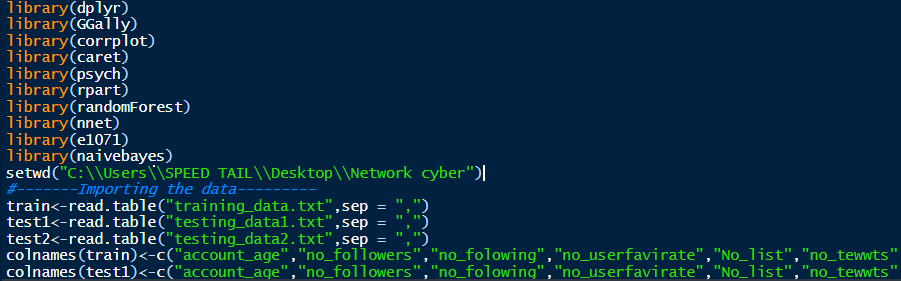
1. **Project Info:** As of late unconstrained business/mass email generally called spams, become a significant issue over the web. Spams is pointless activity, additional room and correspondence move speed. The issue of spams email has been growing for a significant period of time. In late estimations, 40% of all messages are spams which about 15.4 billion email for consistently and that cost web customers about $355 million consistently. Modified email isolating is apparently the best procedure for countering spams at this moment and a tight challenge among spammer's and spams-filtering methodologies is going on. Just a serious extended period of time back an enormous part of the spams could be constantly prevailing by blocking messages beginning from explicit areas or filtering through messages with persuaded titles. Spammer's begun to use a couple of questionable procedures to vanquish the isolating methods like using discretionary sender locations or theoretically connect sporadic characters to the beginning or the piece of the planning title. Data structuring and AI are the two general techniques used in email filtering. In getting the hang of structuring strategy a great deal of principles must be resolved agreeing to which messages are arranged as spams or ham. A great deal of such standards should be made either by the customer of the channel, or by some other expert (for instance the item association that gives a particular rule based spams-isolating contraption). By applying this technique, no promising results shows up since the standards must be constantly invigorated and kept up, which is a pointless activity and it isn't useful for the most part customers. Artificial intelligence approach is more capable than getting the hang of planning technique; it doesn't require deciding any standards . Or maybe, a great deal of getting ready tests, these models is a ton of pre assembled email messages. A specific computation is by then used to take in the gathering rules from these email messages. ML approach has been by and large examined and there are heaps of computations can be used in email isolating. They incorporate Naïve Bayes, SVM(Support vector machines), Neural Networks, K-nearest neighbor, arbitrary woodland calculation and the fake safe system.

**Objective:** The goal of this project is to test different classification and supervised machine learning algorithms that might hypothetically be used as spamss filter in the Email . Fundamentally we want to see which algorithms have the best chance to predicting a spams email in given certain-criteria.

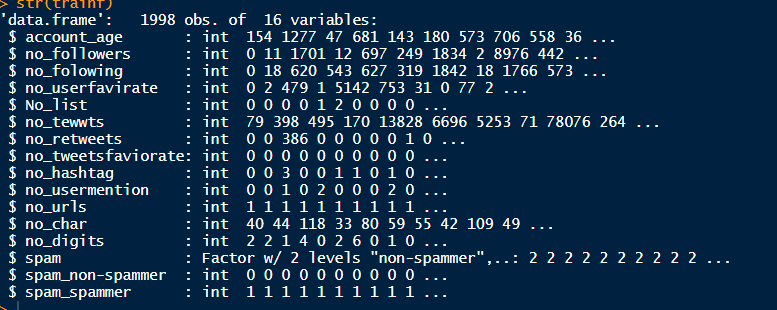
**MACHINE-LEARNING IN E-MAILS CLASSIFICATION:** Machine-Learning (ML) field is a subfield from the wide field of man-made cognizance, this expects to prepare machines to learn like human. Learning here techniques fathomed, watch and address information about some accurate marvel. In independent learning one endeavors to uncover covered regularities (bundles) or to perceive peculiarities in the data like spams messages or framework interference. In email filtering task a couple of features could be the pack of words or the title examination. Consequently, the commitment to email course of action task can be viewed as a two dimensional system, whose tomahawks are the messages and the features. Email request tasks are every now and again parceled into a couple of sub-assignments. In any case, Data social affair and depiction are generally issue express (for instance email messages), second, email incorporate decision and feature lessening try to reduce the dimensionality (for instance the amount of features) for the remainder of the methods for this venture.

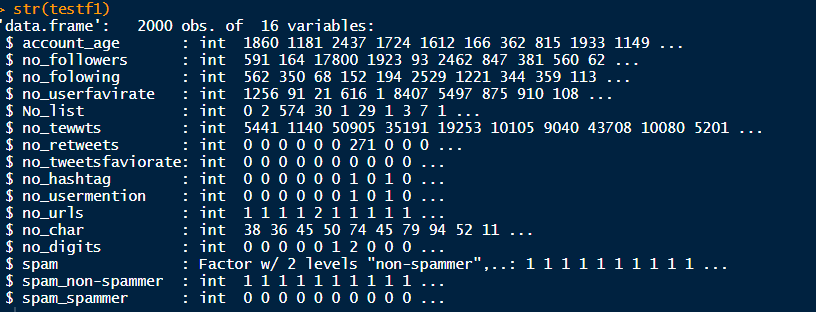
**We have follow the following steps to perform the email –classification:**

**1: Import data set and installed all required packages in R studio:**

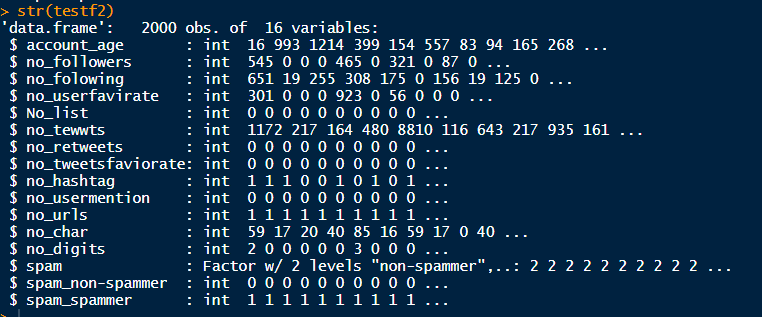


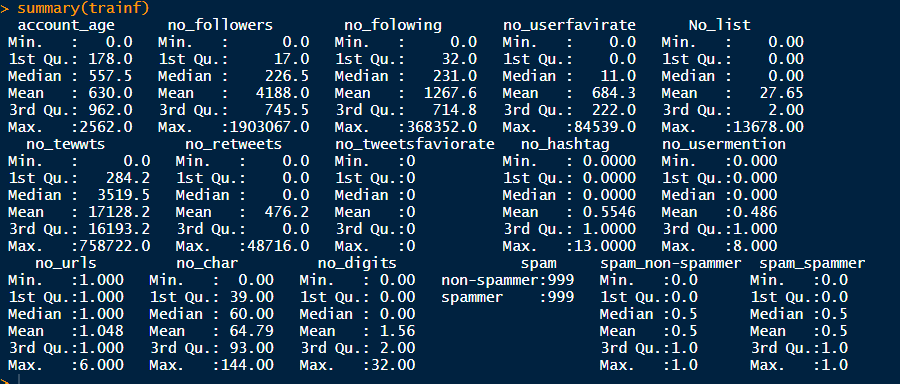
**2. Then perform the Explanatory data analysis:**

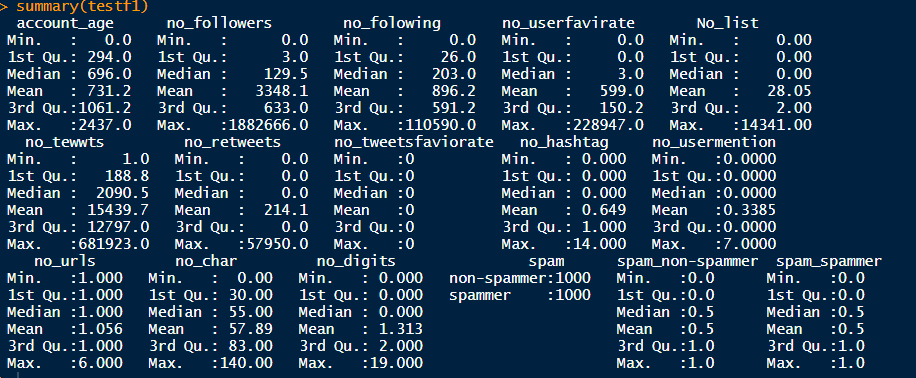


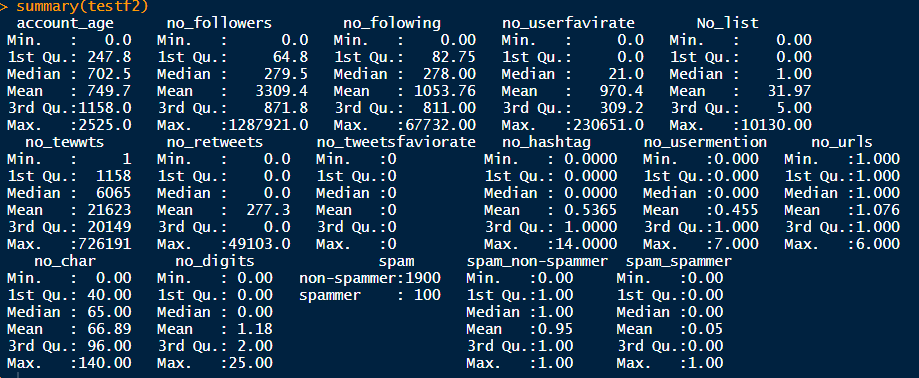


**We have only spams variable that is factor data type**, **and other are integers:**

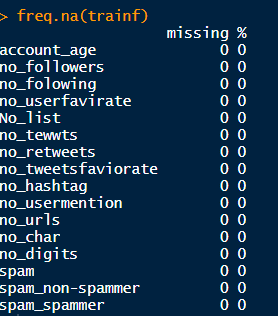


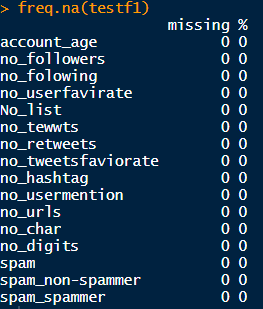


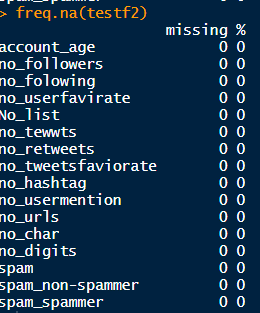




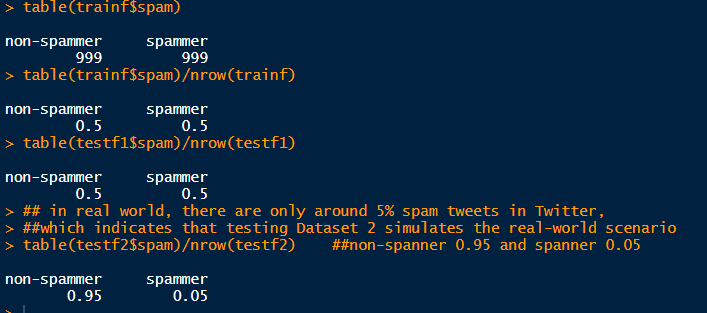
**Data set has cleaned, here none of missing values:**

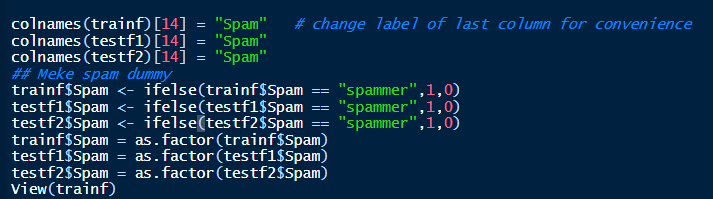






**By using table function we have analysis that 5% email are spams in testing table2 dataset:**

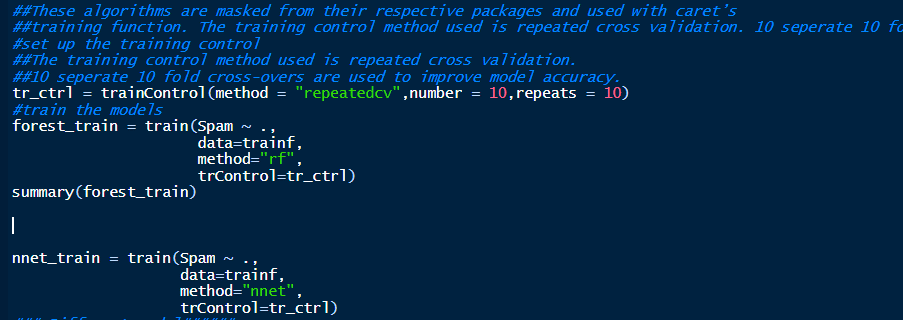




**Training the Models:** After some-research we have decided to test the Random Forest and Neural –Networks. These algorithm required the least tunings and is designed to work with the categorical data. These algorithms are disguised from their respective package and used with caret’s training -function. The training control method used is repeated cross-validation. 10 separate 10 fold cross over are used to mend model accuracy. Have use nnet library in R to perform the model.

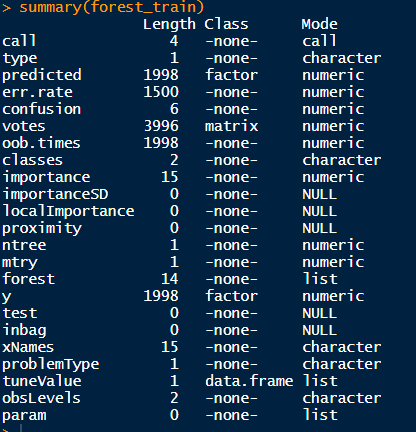
We have set up the training-control. The trainings control method repeated the cross validations. We have use 10 separate 10 fold cross overs are used to improve model the model accuracy.

**Machine-Learning and classification model:-**



**We have perform the random forest and Neural network :**

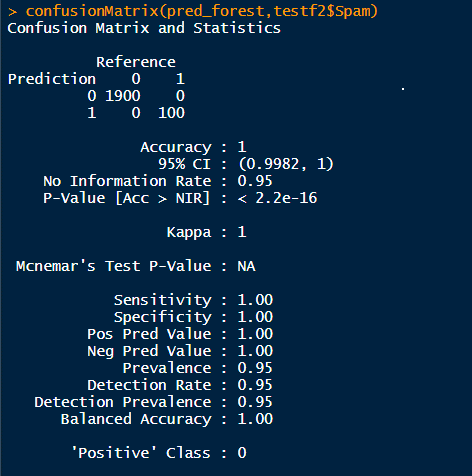
**Random forest Output:**



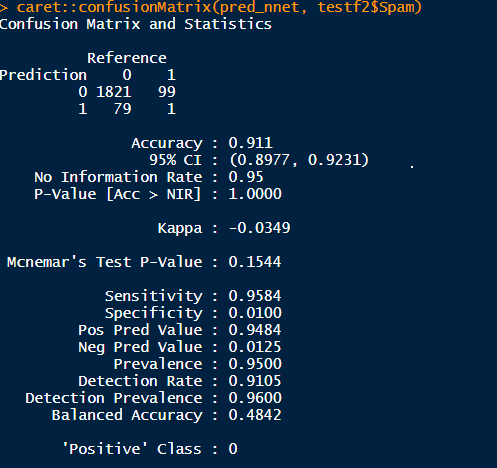
**Result:** Now that the model are trained we test them against the testing set.

The confusion Matrix () display the result and relevant-statistics for each-model.

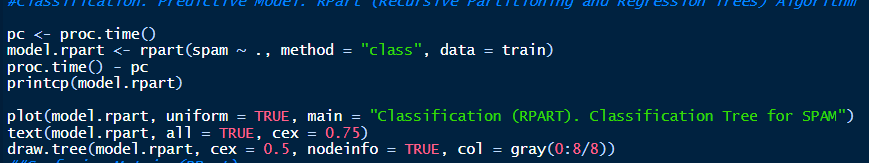
**Confusion-Matrix for Random-forest Model: We got 100% accuracy.**



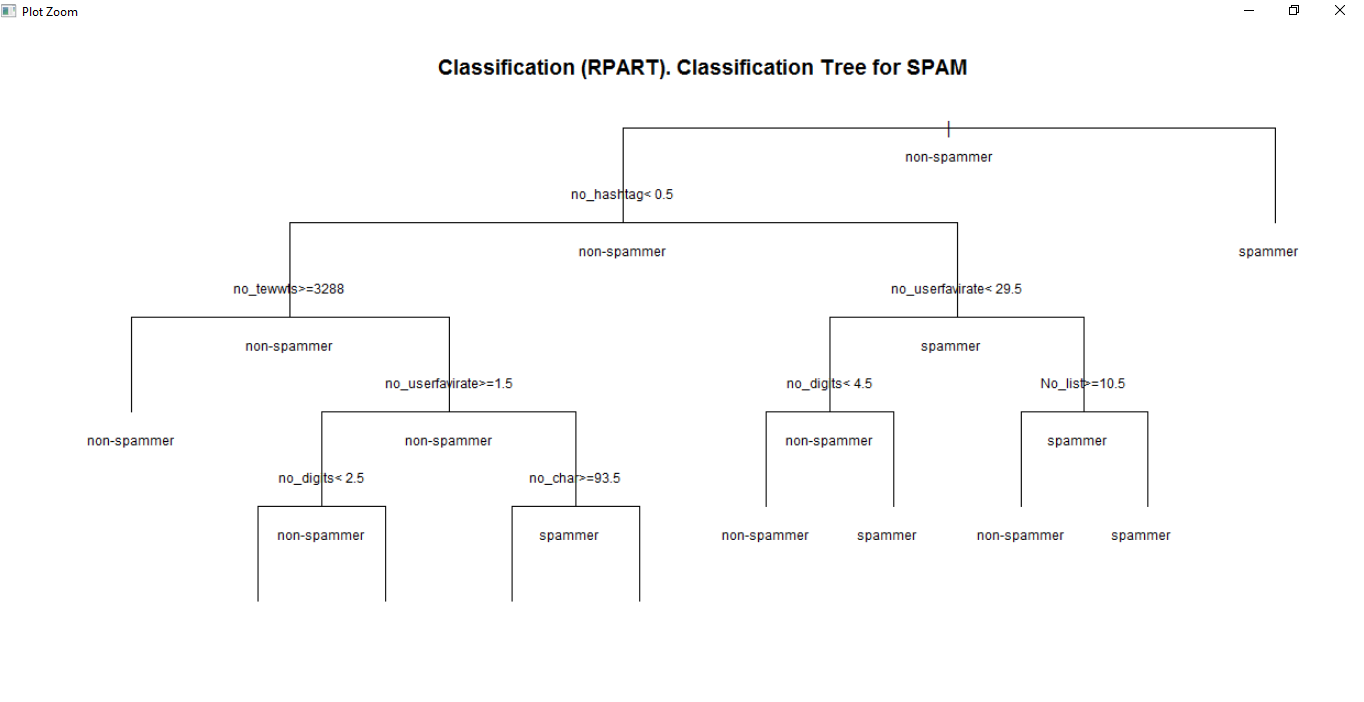
**Confusion-Matrix for Neural-Network: we got 91% model accuracy**

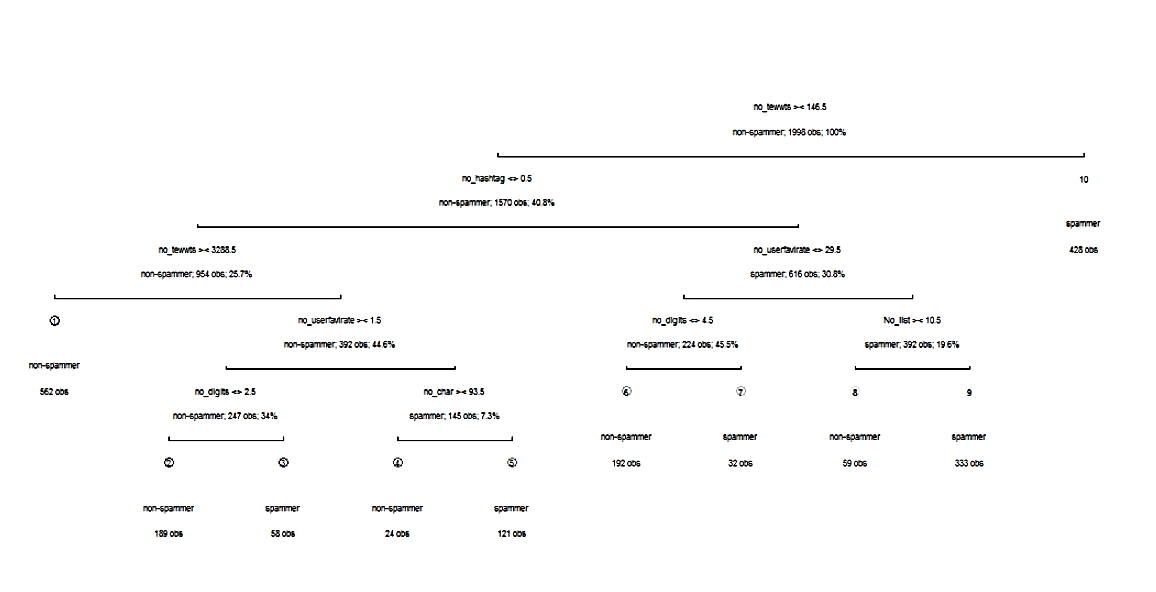


**Classification RPart (Recursive Partitioning and Regression Trees) Algorithm:**



**Classification Tree for Email (non spamsmer)-SPAMS:**





table(`Actual Class` = test2$spams, `Predicted Class` = prediction.rpart)

Predicted Class

Actual Class non-spamsmer spamsmer

non-spamsmer 1464 436

spamsmer 30 70

> error.rate.rpart <- sum(test2$spams != prediction.rpart)/nrow(test2)

> print(paste0("Accuary (Precision): ", 1 - error.rate.rpart))

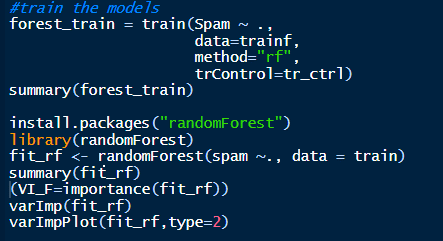
[1] "Accuary (Precision): 0.767"

**We got 76% accuracy .**

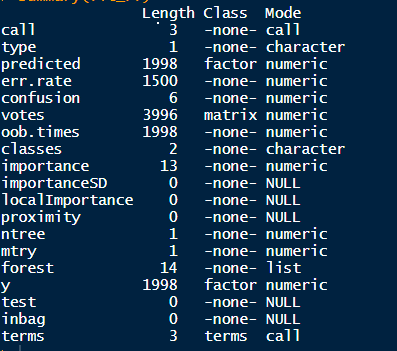
With this-sample, Random-Forest model seemed to performed the best with 100% accuracy and Neural-Networks model performed slightly-lower , have got 91% accuracy but with fine-tuning the result would most likely improved.

**Random-Forest Classifier:** We have used random forest classifier to classify the Spams. Random Forest model is method to do arrangements dependent on highlights. This infers you need highlights and orders. Random-Forest model generates a lot of characterization trees (a troupe) in light of parting a subset of highlights at areas which augment data gain. This strategy is along these lines truly reasonable for disseminated parallel calculation.

Data addition can be controlled by how exact the parting point is in deciding the characterization. Information is part founded on the element at a particular point and the characterization on the left and right of the parting point are checked. On the off chance that for instance the parting point parts all information of a first order from all information of a subsequent grouping, the certainty is 100%; most extreme data gain.



**Random forest Model Summary:**



**FeatureImportance We have found by using Random forest classifier** :

> varImp(fit\_rf)

Overall

account\_age 105.012981

no\_followers 119.133972

no\_folowing 96.111533

no\_userfavirate 78.090408

No\_list 64.312603

no\_tewwts 160.054259

no\_retweets 34.287561

no\_tweetsfaviorate 0.000000

no\_hashtag 113.802207

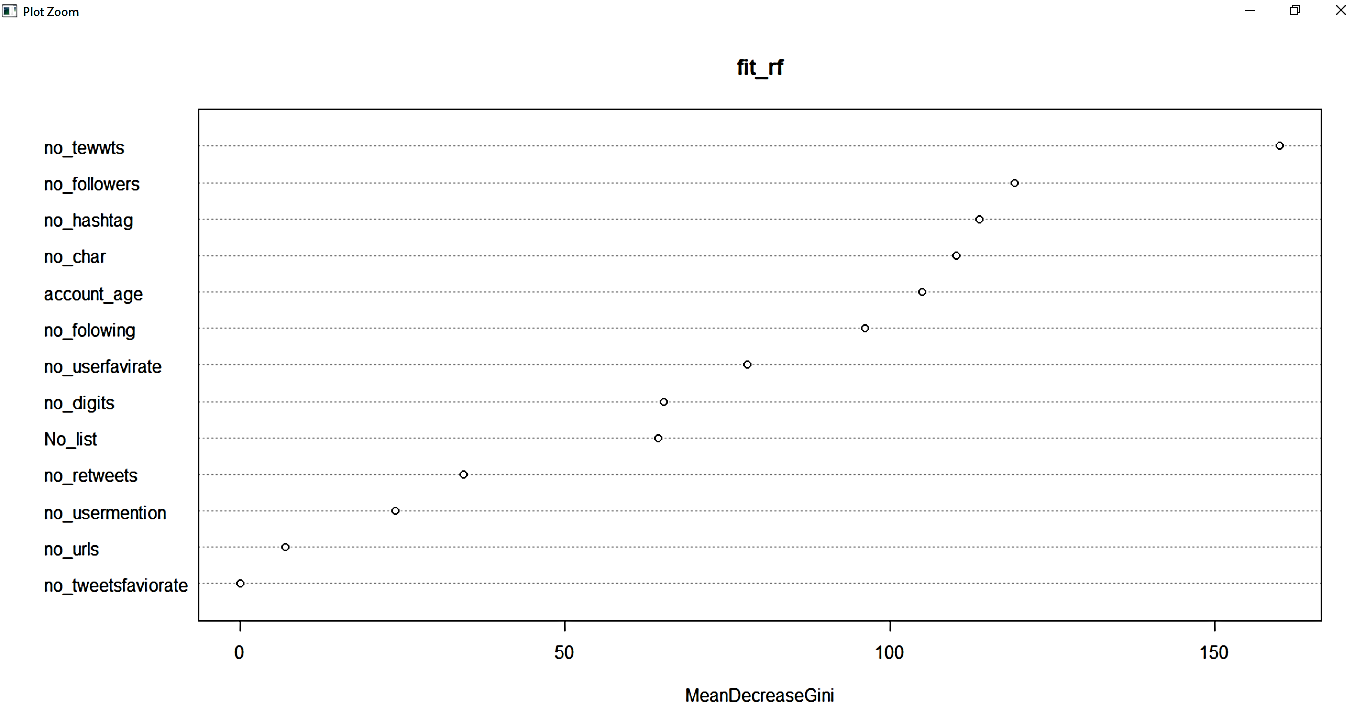
no\_usermention 23.919947

no\_urls 6.936656

no\_char 110.178964

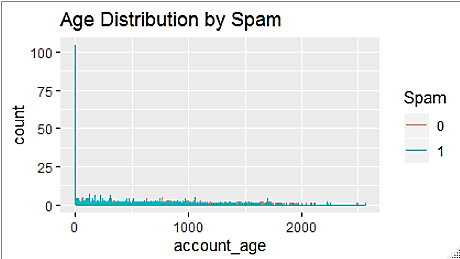
no\_digits 65.188742

**Feature Importance Plot by random Forest:**

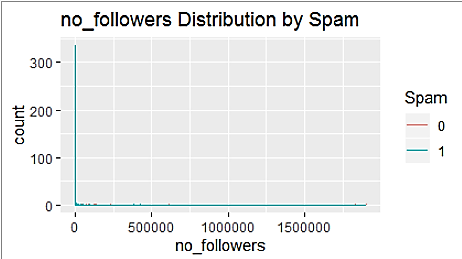


**Visualization for SPAMS Data: here 0 is spams and 1 is non-spamsmer**

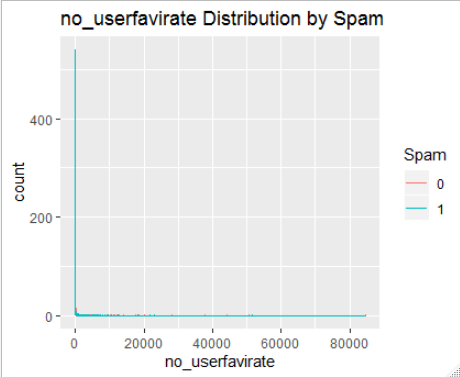
**No\_age distribution by Spams:**



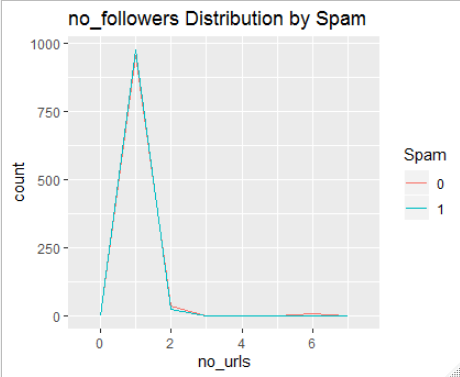
**No-followers distribution by Spams:**



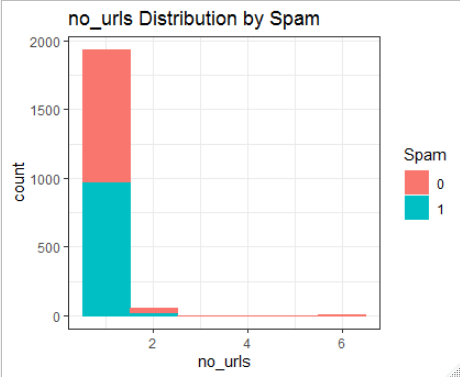
**No \_user favorite distribution by Spams:**



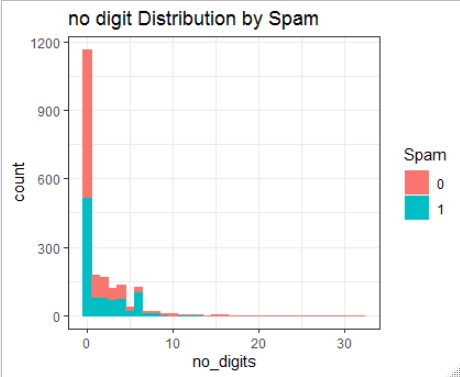
**No\_foloowers distribution by Spams:**



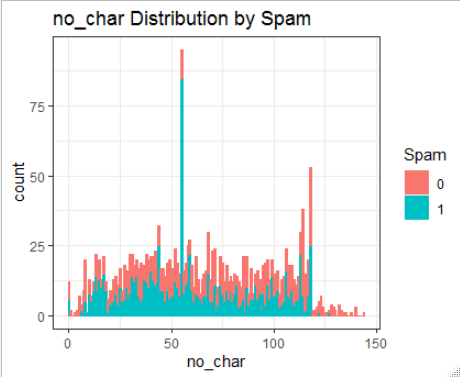
**No\_urls distribution by Spams:**



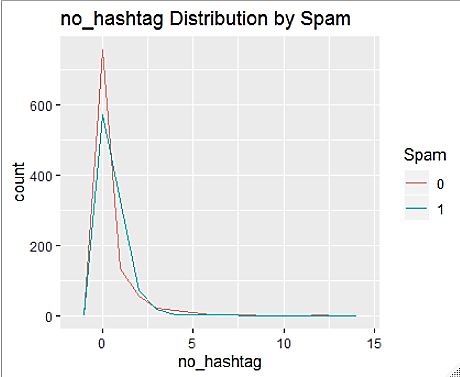
**No-digits distribution by Spams:**



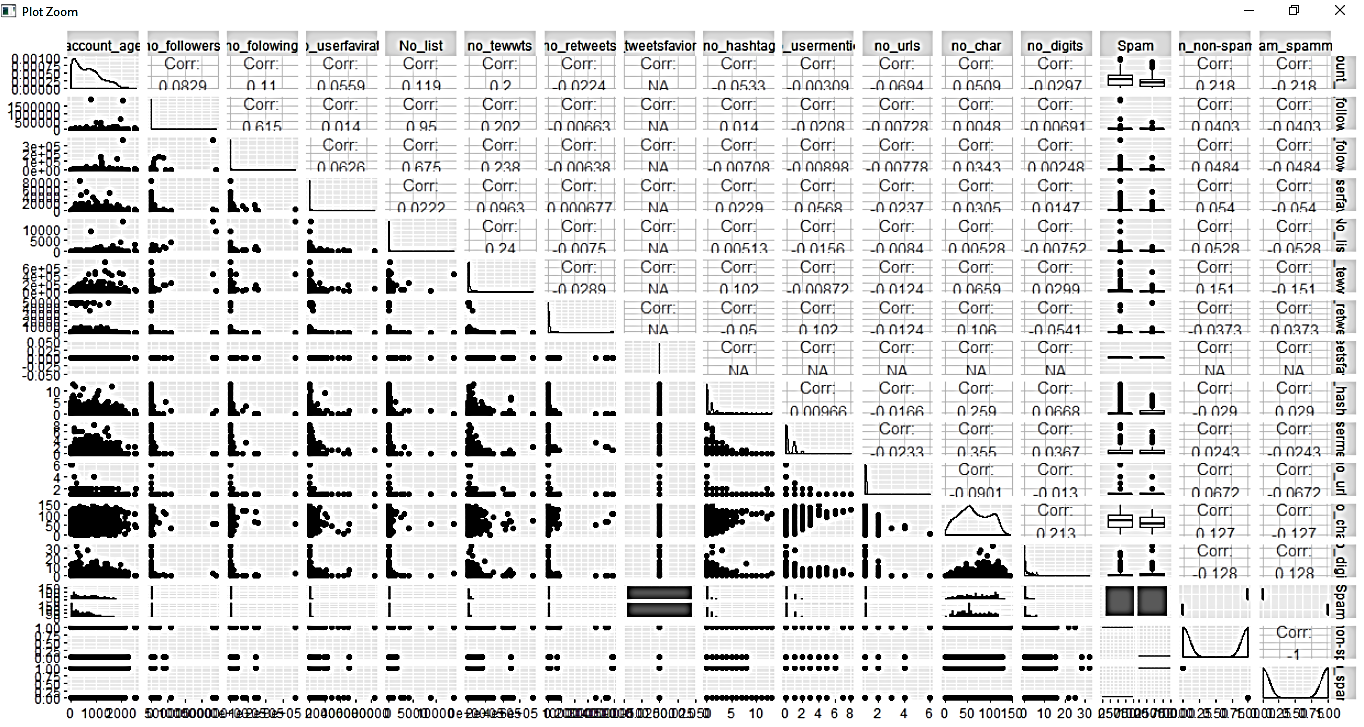
**No\_char distribution by Spams:**



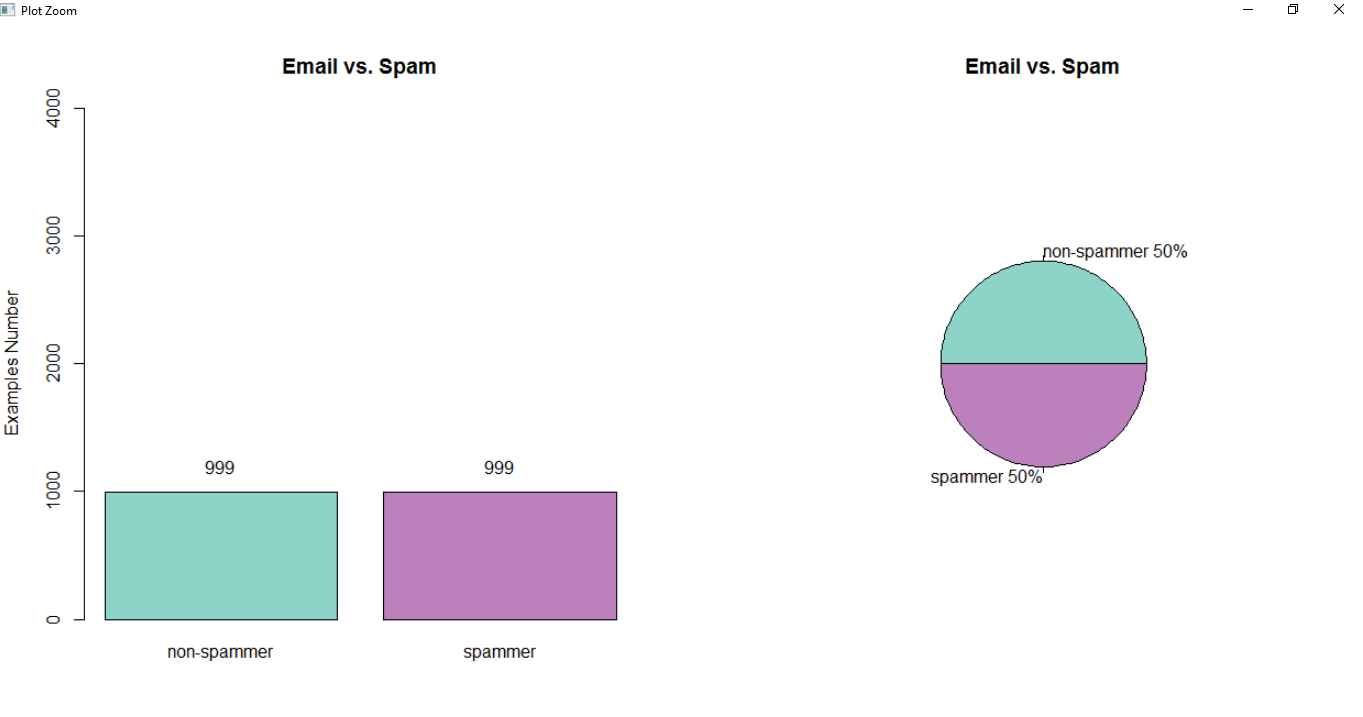
**No\_hashtag distribution by Spams:**



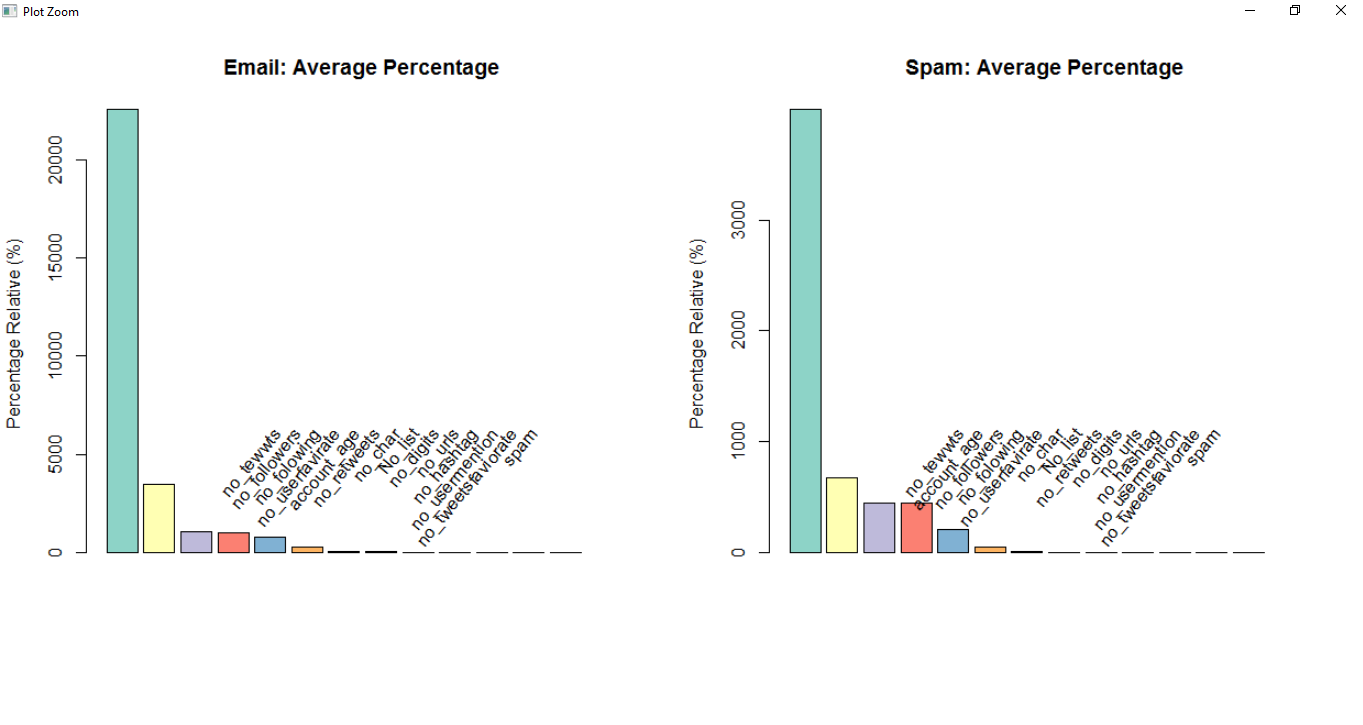
**Plot of Train data :**



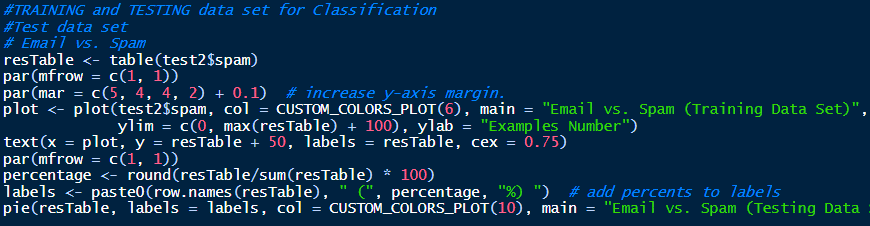
**In train dataset Spams-percentage: Here email -** **non-spamsmer**



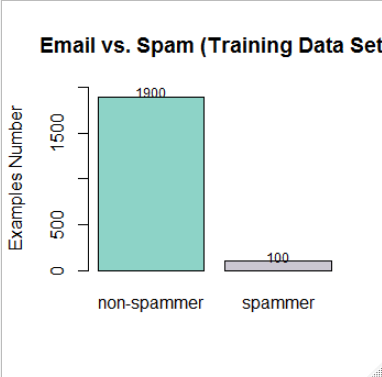
**Spams: Average-Percentage in the data:**



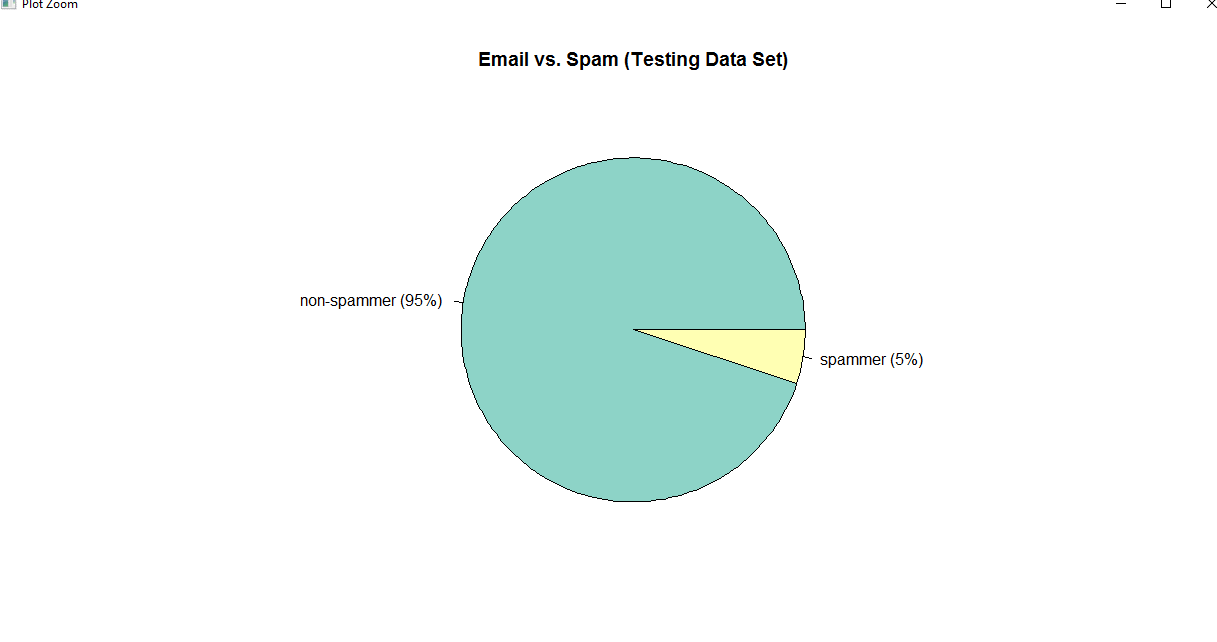
**Email**(**non-spamsmer**)  **vs. Spams (Training-Data Set):**

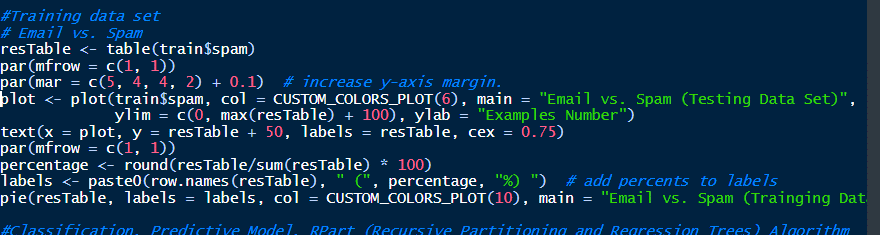


**In Test table 2:**

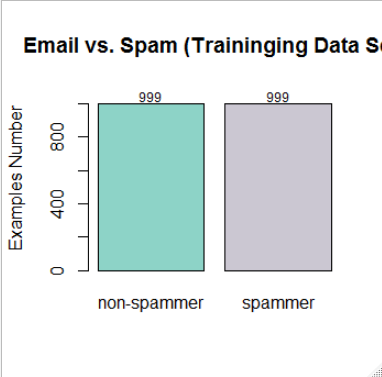


**Email (non-spamsmer) vs. Spams (Testing Data Set):**

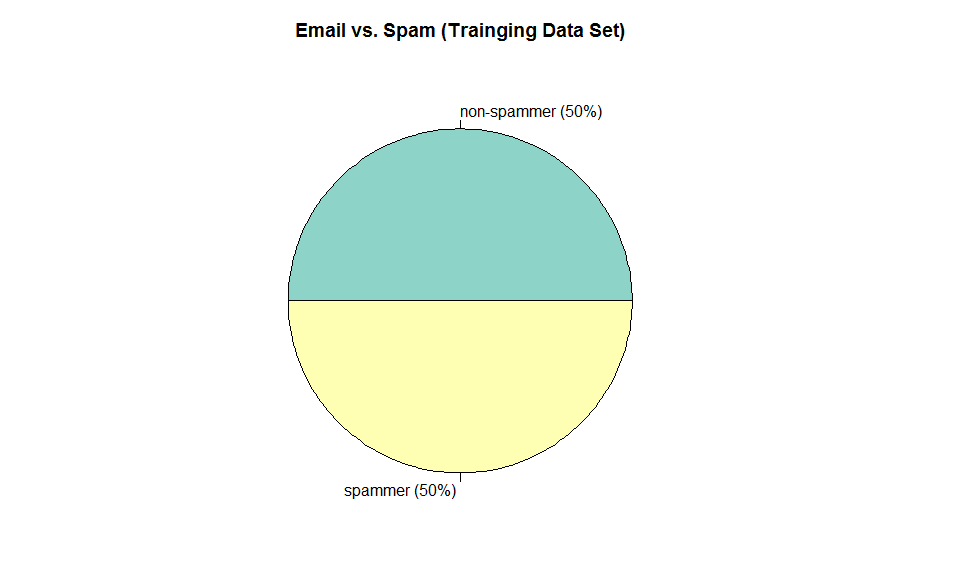




**In Train-data set:**

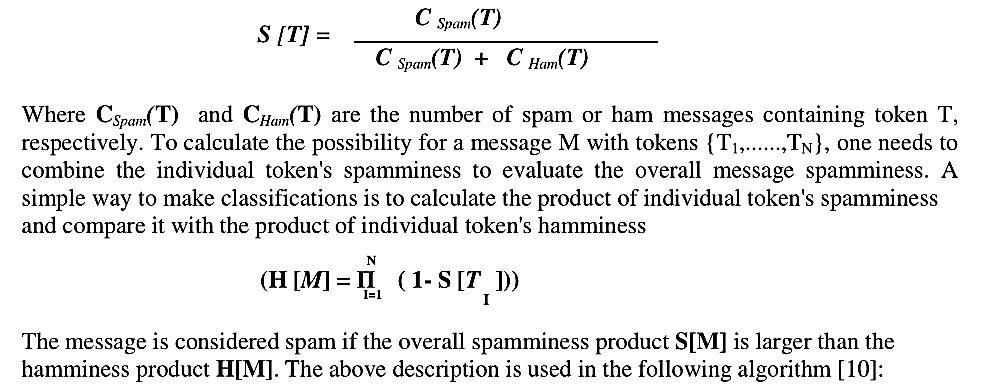


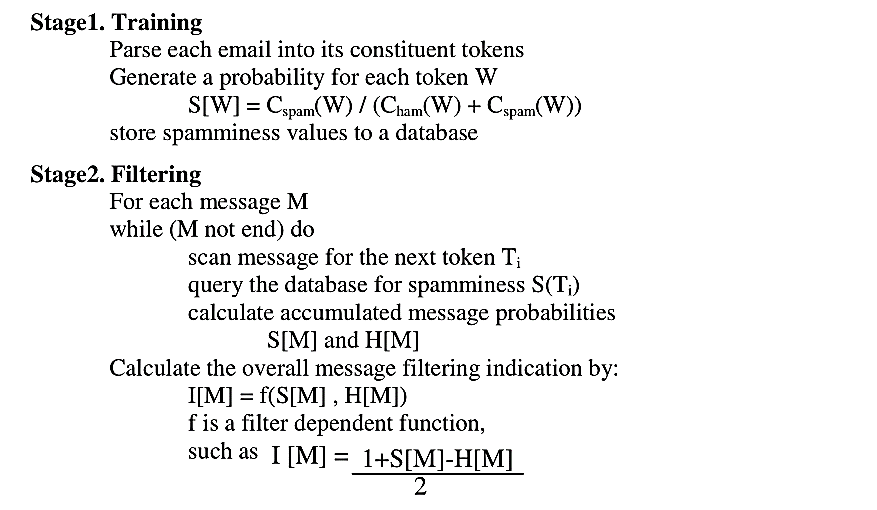
**Pie Chart for Email (non-spamsmer) and Spams:**

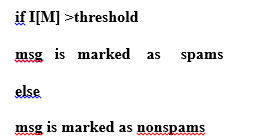


**Naïve-Bayes Machine-Learning classifier method:**

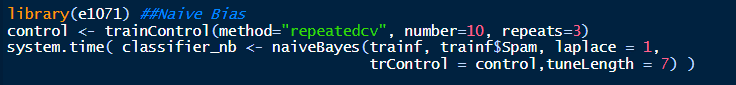
In 1998 the NaïveBayes classifier was projected for spams acknowledgment. Bayesian-classifier is taking a shot at the needy occasions and the likelihood of an occasion happening later on that can be recognized from the past happening of a similar occasion . Guileless Bayes is a generative grouping technique that depends on Bayes hypothesis. It computes the earlier probabilities of each class and probabilities of each quality in each class. It accept that the probabilities of each quality are free of one another. At the hour of order it utilizes the earlier probabilities of each class and the probabilities of the watched traits. The class with most astounding likelihood is relegated to the case being characterized. This method can be utilized to characterize spams messages; words probability play the primary standard here. On the off chance that a few word occur regularly in spams yet not in ham, at that point this approaching email is likely spams. Innocent bayes classifier procedure has turned into a mainstream strategy in mail sifting programming. Bayesian channel ought to be prepared to work adequately. Each word has certain likelihood of happening in spams or ham email in its database. In the event that the aggregate of words probabilities surpasses a specific farthest point, the channel will check the email to either classification. Here, just two classifications are essential: spams or ham. Practically all the measurement based spams channels utilize Bayesian likelihood figuring to join singular token's insights to a general score , and settle on separating choice dependent on the score. The measurement we are for the most part intrigued for a token T is its spamsmers , determined as pursues:







**Naïve0Bias Model:**



**Now Making Predictions and evaluating the NaiveBayes Classifier:**

Confusion Matrix and Statistics

Reference

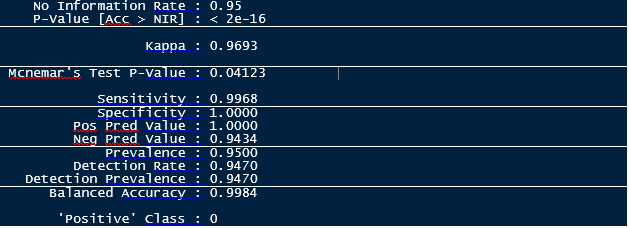
Prediction 0 1

0 1894 0

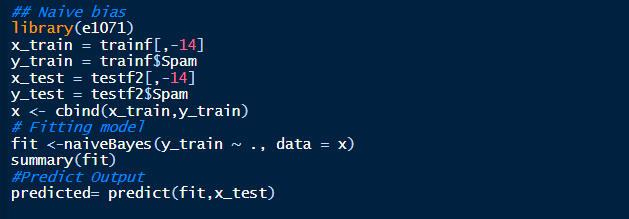
1 6 100

Accuracy : 0.997

95% CI : (0.9935, 0.9989)

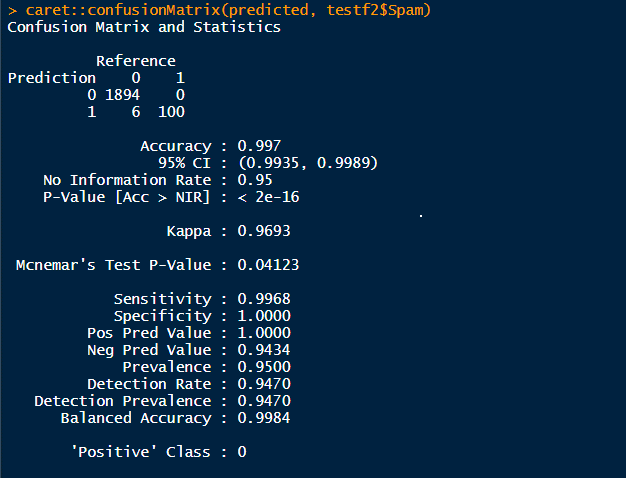


The Naïve-Bayes Classifier also performed-very well on the-training set by achieve 99.70% accuracy which-means we got misclassifications out possible remark. While the model has a 100% sensitivity rate; the proportions of the positive class predicted as positive, it was able to achieve about 100% on specificity which is the proportions of the negative class predicted accurately



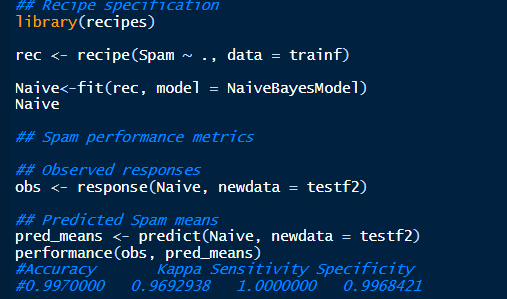
**By using naïve Bias We have got best ever accuracy : 99%**



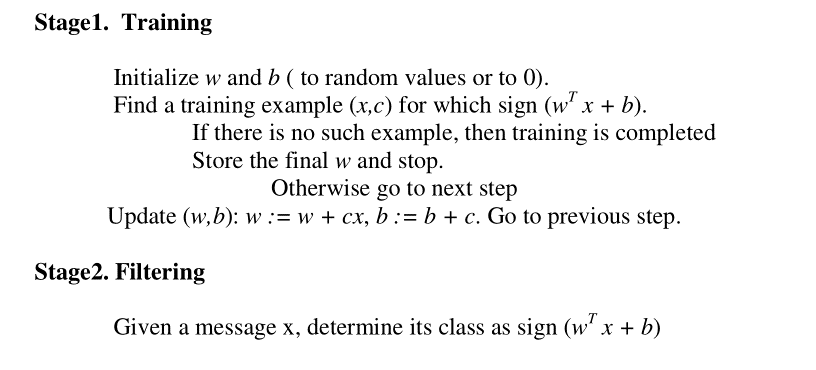


**By using Naïve Bias we have got 99% accuracy.**

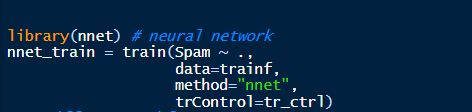
**###Model-Evalution for naïve Bias Machine Learning Model:**



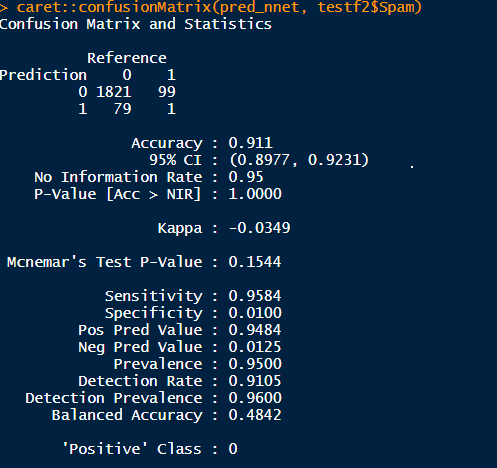
**Artificial-Neural Networks classifier method:** An artificial-neural-network (ANN) is computational-model dependent on natural-neural system. It comprises of an interconnected gathering of counterfeit neurons. A fake neural system is a versatile framework that changes its structure dependent on data that moves through the fake system during a learning stage. The ANN depends on the guideline of learning by model. There are, anyway the two old style sort of the neural systems, perceptron and the multilayer-perceptron.



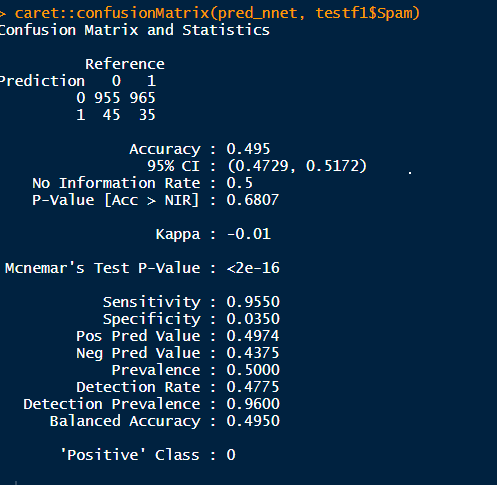
**Neural N/w Machine0Learning Model:**



**Model result: We have got 91% accuracy Neural N/w model for Email spams detection**



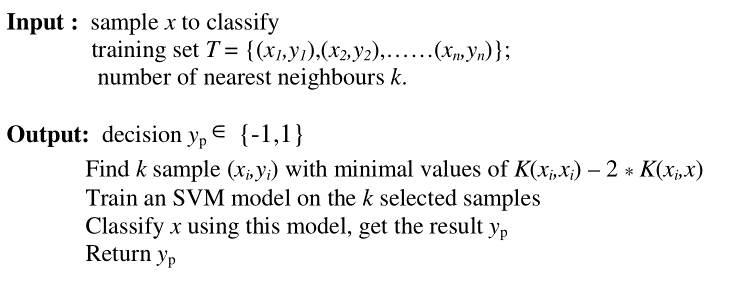
**For test data 1 We have got less accuracy: got 49.5% accuracy**

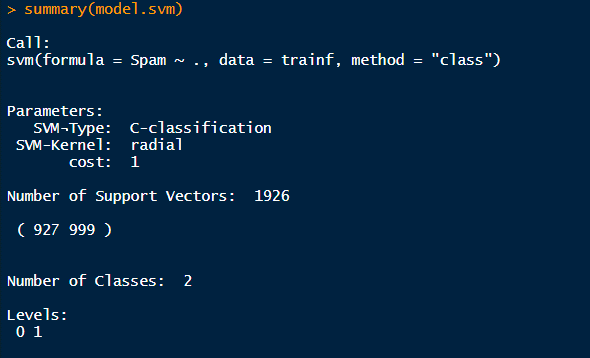


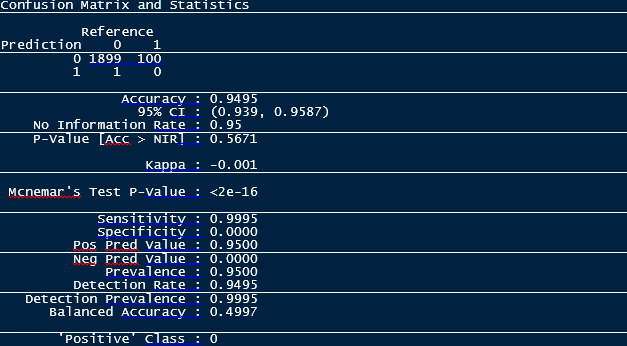
Neural-Networks performed slightly lower , we got 91% accuracy but with fine tuning the results would most likely improve.

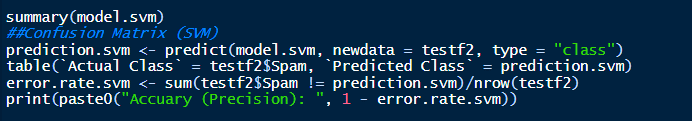
**Support-Vector Machines-classifier Model:**

SVM rely upon the possibility of decision planes that portray decision limits. A decision plane is one that disconnects between a ton of articles having assorted class enlistments, the SVM demonstrating calculation finds an ideal hyper plane with the maximal edge to isolate two classes, which requires taking care of the accompanying advancement issue. SVM is a discriminative regulated AI system of grouping. SVM applies0Vapnik's measurable learning hypothesis to prepare classifiers. SVM has some striking highlights for which it has been considered as condition of craftsmanship in the grouping undertakings. SVM has been utilized for content order, written by hand digit location and numerous other characterization errands. A portion of its one of a kind highlights are: it can function admirably in an exceptionally high dimensional element space, it exploit just a subset of unique preparing set to settle on choice limit called bolster vector and it is likewise appropriate for non-directly detachable information (it utilizes piece stunt).



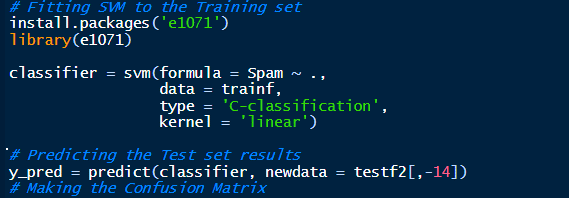




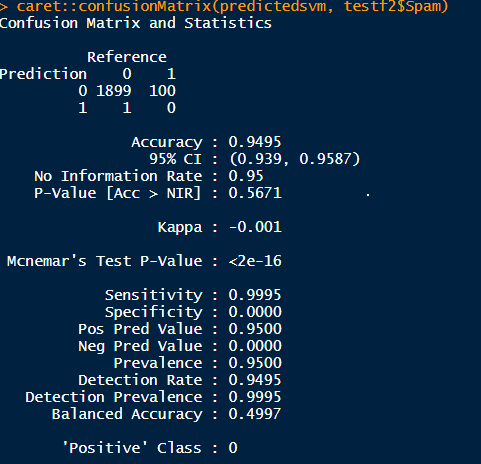


We got 94% accuracy by SVM.

By using linear kernel we have build SVM model:



We have again perform the SVM model then we have got this result:



Confusion Matrix For SVM:

> conf\_matrix

predictions

targets 1 2

1 1552 82

2 348 18

**SVM (Support Vector Machine) Algorithm:**

**Output:**

svm(formula = spams ~ ., data = train, method = "class")

Parameters:

SVM-Type: C-classification

SVM-Kernel: radial

cost: 1

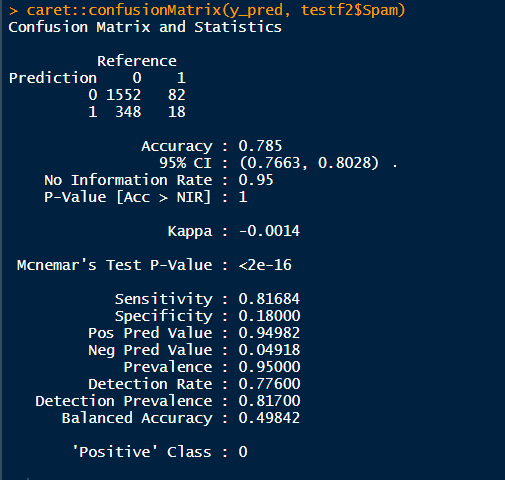
Number of Support Vectors: 1928

( 929 999 )

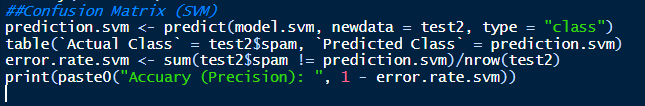
Number of Classes: 2

Levels:

non-spamsmer spamsmer



We have got less accuracy by using SVM , got 78%



print(paste0("Accuary (Precision): ", 1 - error.rate.svm))

[1] "Accuary (Precision): 0.9495"

We have goted 94% Precisian accuracy

**Performance-Comparison:** NaïveBayes system has the most surprising accuracy among the six Calculations while the kNN neighbor has the most perceptibly dreadful precision rate and amazingly the cruel sets procedure has an incredibly forceful percent, finally we can find that the audit is the less rate among the six classifiers while the Naïve Bayes still has the most astonishing execution anyway saw as low when appeared differently in relation to precision and exactness while the upsetting sets has the most exceedingly horrible introduction.

**Validity of the-model:**

The essence of building the spams classifier is for the model to be able to effectively categories an incoming email as either spamsmer or non-spamsmer. The Random Forest and Naive Bayes performed exceptionally well in this project.

**CONCLUSION:** In Email-Spam classification project we scrutinize the absolute most protuberant Machine-learning techniques and of their materialness to the issue of spams email characterization. Email spams order has gotten a huge consideration by lion's share of the individuals as it recognizes the undesirable data and dangers. Thusly, the greater part of the analysts focus in finding the best classifier for distinguishing spams messages. We can discovery that the Naïv- bayes and harsh sets techniques has an exceptionally fulfilling exhibition among different strategies, more research must be done to raise the presentation of the Naïve-bayes and Artificial resistant framework either by crossover framework or by purpose the component reliance issue in the gullible bayes classifier, or half and half the Immune by unpleasant sets. At long last half and half frameworks seem to be the most productive approach to create a antispams channel these days.

**References**

* Meda, C., Bisio, F., Gastaldo, P. and Zunino, R., 2014, October. A machine learning approach for Twitter spammer’s detection. In 2014 International Carnahan Conference on Security Technology (ICCST) (pp. 1-6). IEEE.
* Song, J., Lee, S. and Kim, J., 2011, September. Spams filtering in twitter using sender-receiver relationship. In International workshop on recent advances in intrusion detection (pp. 301-317). Springer, Berlin, Heidelberg.
* Tan, E., Guo, L., Chen, S., Zhang, X. and Zhao, Y., 2012, June. Spamsmer behavior analysis and detection in user generated content on social networks. In 2012 IEEE 32nd International Conference on Distributed Computing Systems (pp. 305-314). IEEE.
* Ye, Q., Zhang, Z. and Law, R., 2009. Sentiment classification of online reviews to travel destinations by supervised machine learning approaches. Expert systems with applications, 36(3), pp.6527-6535.
* Chen, J., Huang, H., Tian, S. and Qu, Y., 2009. Feature selection for text classification with Naïve Bayes. Expert Systems with Applications, 36(3), pp.5432-5435.
* Bijalwan, V., Kumar, V., Kumari, P. and Pascual, J., 2014. KNN based machine learning approach for text and document mining. International Journal of Database Theory and Application, 7(1), pp.61-70.

The year 2016 witnessed advancements in artificial intelligence in self-driving cars, language translation, and big data. That same time period, however, also witnessed [the rise of ransomware, botnets, and attack vectors as popular forms of malware attack](https://www.malwarebytes.com/pdf/white-papers/stateofmalware.pdf), with cybercriminals continually expanding their methods of attack (e.g., [attached scripts to phishing emails and randomization](https://www.malwarebytes.com/pdf/white-papers/stateofmalware.pdf)), according to [Malware Byte's State of Malware report](https://www.malwarebytes.com/pdf/white-papers/stateofmalware.pdf). To complement the skills and capacities of human analysts, organizations are turning to [machine learning (ML)](https://en.wikipedia.org/wiki/Machine_learning) in hopes of providing a more forceful deterrent. ABI Research [forecasts](https://www.abiresearch.com/press/machine-learning-cybersecurity-boost-big-data-inte/) that "machine learning in cybersecurity will boost big data, intelligence, and analytics spending to $96 billion by 2021." At the SEI, machine learning has played a critical role across several technologies and practices that we have developed to reduce the opportunity for and limit the damage of cyber attacks. In this post--the first in a series highlighting the application of machine learning across several research projects--I introduce the concept of machine learning, explain how machine learning is applied in practice, and touch on its application to cybersecurity throughout the article.

Machine learning refers to systems that are able to automatically improve with experience. Traditionally, no matter how many times you use software to perform the same exact task, the software won't get any smarter. Always launch your browser and visit the same exact website? A traditional browser won't "learn" that it should probably just bring you there by itself when first launched. With ML, software can gain the ability to learn from previous observations to make inferences about both future behavior, as well as guess what you want to do in new scenarios. From thermostats that optimize heating to your daily schedule, autonomous vehicles that customize your ride to your location, and advertising agencies seeking to keep ads relevant to individual users, ML has found a niche in all aspects of our daily life.

To understand how ML works we first need to understand the fuel that makes ML possible: data. Consider an email spam detection algorithm. Original spam filters would simply blacklist certain addresses and allow other mail through. ML enhanced this considerably by comparing verified spam emails with verified legitimate email and seeing which "features" were present more frequently in one or the other. For example, intentionally misspelled words ("V!AGR4"), the presence of hyperlinks to known malicious websites, and virus-laden attachments are likely features indicative of spam rather than legitimate email. (More discussion on "features" below.) This process of automatically inferring a label (i.e., "spam" vs "legitimate") is called [*classification*](https://en.wikipedia.org/wiki/Statistical_classification), and is one of the major applications of ML techniques. It is worth mentioning that one other very common technique is [*forecasting*](https://en.wikipedia.org/wiki/Forecasting), the use of historical data to predict future behavior. While considerable research and technology has been developed to perform forecasting, the remainder of this post will focus on classification.

There are two major types of ML classification techniques: [*supervised learning*](https://en.wikipedia.org/wiki/Supervised_learning) and *unsupervised learning*, which are differentiated by the data (i.e., input) that they accept. *Supervised learning* refers to algorithms that are provided with a set of labeled training data, with the task of learning what differentiates the labels. While in our previous example there were only two labels--"spam" and "legitimate"--other scenarios may contain many, many more. For example, modern image recognition algorithms, such as [Google Image search](https://images.google.com/), can accurately distinguish tens of thousands of objects, and modern facial recognition algorithms [exceed the performance of human beings](https://medium.com/the-physics-arxiv-blog/the-face-recognition-algorithm-that-finally-outperforms-humans-2c567adbf7fc). By learning what makes each category unique, the algorithm can then be presented with new, unlabeled data and apply a correct label. Note the criticality in choosing a representative training dataset; if the training data contains only dogs and cats, but the new photo is a fish, the algorithm will have no way of knowing the proper label.

[*Unsupervised learning*](https://en.wikipedia.org/wiki/Unsupervised_learning) refers to algorithms provided with unlabeled training data, with the task of inferring the categories all by itself. Sometimes labeled data is very rare, or the task of labeling is itself very hard, or we may not even know if labels exist. For example, consider the case of network flow data. While we have enormous amounts of data to examine, attempting to label data would be extremely time-intensive, and it would be very hard for a human to determine what label to assign. Given how good machines are at finding patterns in large datasets, it is often much easier to simply have the machine separate data into groups for us.

Note that separating data into groups assumes that the relevant data is present. Determining the color of someone's skin is fairly trivial for a sighted person, but a blind person will find that task much harder since they are lacking the most important sensor. They will have to rely on other information, such as the person's voice, to correctly "label" the individual. Machines are no different in this regard.

We mentioned earlier the concept of a *feature*. This concept can be understood fairly straightforwardly: if our data is stored in a spreadsheet where a single row represents one data point, then the features are the columns. For our email example, some features may be the sender, recipient, date, and content of the email. From our network flow example, features include packet size, remote IP address, network port, packet content, or any of the hundreds of different attributes that network traffic can have. Having useful features is a critical prerequisite for being able to successfully apply machine learning techniques. Simultaneously, having too many non-informative features may degrade algorithm performance, as the overabundance of noise can hide more useful information.

To that extent, there is an entire branch within machine learning referred to as [*feature engineering*](https://en.wikipedia.org/wiki/Feature_engineering). The goal of this practice is to extract the maximum information from the available features so as to maximize our ability to predict or categorize unknown data. Frequently these techniques will take multiple features and combine or transform them in complex ways to obtain new, more informative features. While a full treatment of these approaches is outside the scope of this article, interested readers are encouraged to read up on [Principle Component Analysis](https://en.wikipedia.org/wiki/Principal_component_analysis) (PCA), a fairly straightforward yet highly useful technique for both creating new data from existing features, as well reducing the number of total features required for the algorithm to function.

One last topic to address is that of [*big data*](https://insights.sei.cmu.edu/sei_blog/big-data/). From the above cases we can understand that more data is almost always a good thing; it allows algorithms to be aware of many more varieties of categories. Continuing our email example, while one person may get a lot of spam, many people get a tremendous amount of spam, providing that many more examples for the ML algorithm to train against. Within the past ten years, as the value of data has been realized, enormous databases for all types of imaginable data have sprung up containing sometimes billions of rows of data, with hundreds of thousands of features. Such enormous datasets are technically hard to work with, and an entire field of research and tooling has developed with the specific intent of simplifying the process of working with data of this size. This is the field of big data.

The steps required to create a ML tool are varied, but typically proceed as follows:

1. **Data collection**. While it's possible to run and even create ML algorithms based on streaming, real-time data (e.g., trading decision based on stock market data), the majority of techniques involve collecting data ahead of time and creating a model using stored data.
2. **Data cleaning**. Raw data is often unusable for ML purposes. There may be missing data, inconsistent data use (e.g., a cardinality feature may contain "North", "north", and "N", all identical in meaning), and numeric data with non-numeric characters, among many other possible problems. This step also involves combining multiple data sources to a single usable source. Cleaning is often a time-consuming and iterative process, as fixing one issue often uncovers another.
3. **Feature engineering**. After all the data is ready for use it's time to ensure that maximum information is extracted from the data itself, as described above. This process usually takes place prior to creating the ML algorithm.
4. **Model building/model validation**. This set of steps involves building the model and testing to ensure it works properly on unlabeled data. There are many statistical considerations to consider when testing the model. When working with supervised ML, a chief concern is whether the model is *[overfit](https://en.wikipedia.org/wiki/Overfitting)* to the training data, i.e., whether the model that was produced takes into account properties that are unique to the training data. There are many statistical techniques used to minimize this risk, which are often employed during model validation.
5. **Deployment/Monitoring**. Deployment of an ML model is rarely a "once-and-done" event. Generally, and especially in the case of network traffic, historical observations do not necessarily match future activity. For that reason, even after deployment, models are monitored and periodically rerun through the build/validate step to ensure top performance.

**Looking Ahead**

In future posts in this series, I will be exploring the application of machine learning to various research areas to reduce the opportunity for, and limit the damage of, various types of cyber attacks. Some specific areas include:

* insider threat
* malware analysis
* network analytics
* secure coding
* situational awareness

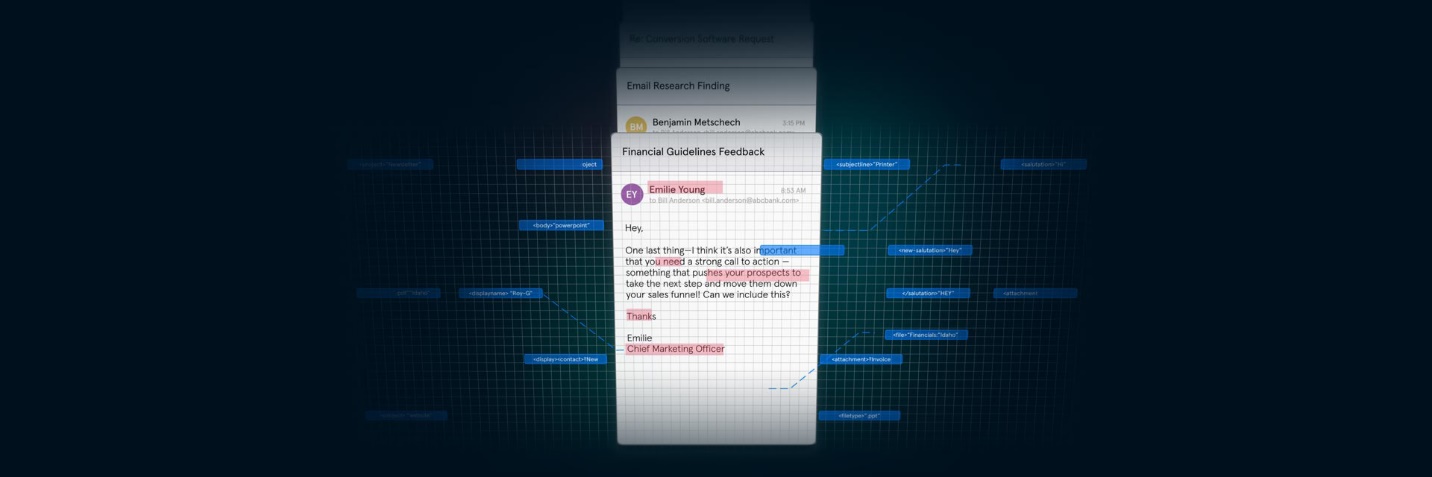
In addition to the above, we will also describe how we are applying ML to training new cybersecurity analysts.

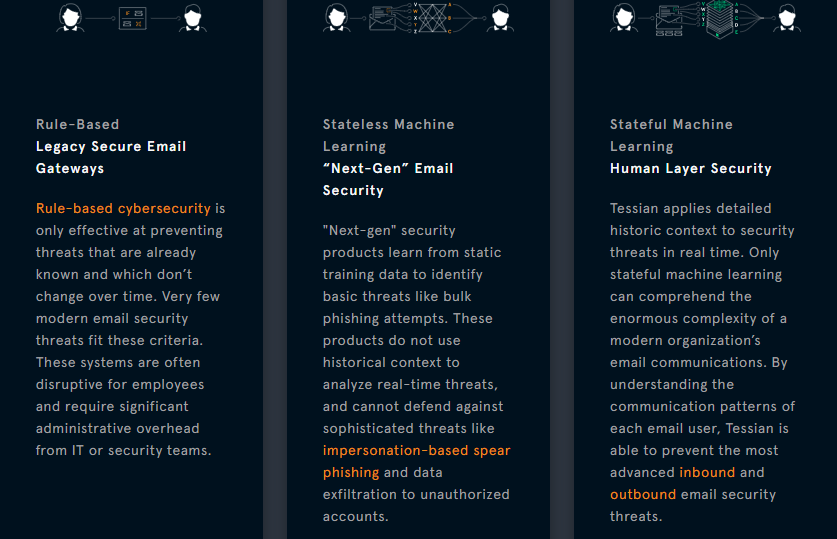
I welcome your feedback on this work in the comments section below.

**Additional Resources**

Further reading for those interested in a more in-depth treatment of this topic:

* ZDNet's "[Inside the Black Box: Understanding AI Decision-Making](http://www.zdnet.com/article/inside-the-black-box-understanding-ai-decision-making/)"
* MARTECH Today's "[How Machine Learning Works, As Explained By Google](https://martechtoday.com/how-machine-learning-works-150366)"
* Science Magazine's [overview of Facebook's DeepFace](http://www.sciencemag.org/news/2015/02/facebook-will-soon-be-able-id-you-any-photo))
* Stateful machine learning to secure people using email
* Trained on over 1 billion emails, Tessian’s stateful machine learning engine learns the difference between normal and abnormal from historical email data. In real time, Tessian automatically prevent the most advanced forms of spear phishing, accidental data loss and data exfiltration. Enterprises can also design and deploy their own customized email filters to solve for specific risks and ensure email usage remains compliant.





**Machine Learning for Cybersecurity 101**

Machine Learning is aiding greatly with cybersecurity. Let's get more familiar with the basics of how this is happening.

[**[](https://dzone.com/users/2952208/apolyakov-1.html)**](https://dzone.com/users/2952208/apolyakov-1.html)**by**

[**Alexander Polyakov**](https://dzone.com/users/2952208/apolyakov-1.html)

**·**

**Oct. 28, 18 ·**[**AI Zone**](https://dzone.com/artificial-intelligence-tutorials-tools-news)**· Opinion**

**Like** [**(9)**](https://dzone.com/articles/machine-learning-for-cybersecurity-101)

**Comment (0)**

**Save**

[**Tweet**](https://dzone.com/articles/machine-learning-for-cybersecurity-101)

**17.43k Views**

Join the DZone community and get the full member experience.

[**JOIN FOR FREE**](https://dzone.com/articles/machine-learning-for-cybersecurity-101)

The considerable number of articles cover Machine Learning for cybersecurity and the ability to protect us from cyber attacks. Still, it’s important to scrutinize how actually Artificial Intelligence (AI), Machine Learning (ML), and Deep Learning (DL) can help in cybersecurity right now and what this hype is all about.

First of all, I have to disappoint you. Unfortunately, Machine Learning will never be a silver bullet for cybersecurity compared to image recognition or natural language processing, two areas where Machine Learning is thriving. There will always be a man trying to find weaknesses in systems or ML algorithms and to bypass security mechanisms. What’s worse, now hackers are able to use Machine Learning to carry out all their nefarious endeavors.

Fortunately, Machine Learning can aid in solving the most common tasks including regression, prediction, and classification. In the era of an extremely large amount of data and cybersecurity talent shortage, ML seems to be an only solution.

This article is an introduction written to give the practical technical understanding of the current advances and future directions of ML research applied to cybersecurity.

**Machine Learning Terminology**

Stop calling everything "AI" — learn the terms.

* **AI (Artificial Intelligence)** — a broad concept. A *Science* of making things smart or, in other words, human tasks performed by machines (e.g., Visual Recognition, NLP, etc.). The main point is that AI is not exactly Machine Learning or smart things. It can be a classic program installed in your robot cleaner like edge detection. Roughly speaking, AI is a thing that somehow carries out human tasks.
* **ML (Machine Learning)** — an *Approach* (just one of many approaches) to AI that uses a system that is capable of learning from experience. It is intended not only for AI goals (e.g., copying human behavior) but it can also reduce the efforts and/or time spent for both simple and difficult tasks like stock price prediction. In other words, ML is a system that can recognize patterns by using examples rather than by programming them. If your system learns constantly, makes decisions based on data rather than algorithms, and change its behavior, it’s Machine Learning.
* **DL (Deep Learning)** — a set of *Techniques* for implementing Machine Learning that recognize patterns of patterns – like image recognition. The systems identify primarily object edges, a structure, an object type, and then an object itself. The point is that Deep Learning is not exactly Deep Neural Networks. There are other algorithms, which were improved to learn patterns of patterns, such as Deep Q Learning in Reinforcement task.

The definitions show that cybersecurity field refers mostly to Machine Learning (not to AI). And a large part of the tasks are not human-related.

Machine Learning means solving certain tasks with the use of an approach and particular methods based on data you have.

Most of the tasks are subclasses of the most common ones, which are described below:

* Regression (or prediction) — a task of predicting the next value based on the previous values.
* Classification — a task of separating things into different categories.
* Clustering — similar to classification but the classes are unknown, grouping things by their similarity.
* Association rule learning (or recommendation) — a task of recommending something based on the previous experience.
* Dimensionality reduction — or generalization, a task of searching common and most important features in multiple examples.
* Generative models — a task of creating something based on the previous knowledge of the distribution.

There are different approaches in addition to these tasks. You can use only one approach for some tasks, but there can be multiple approaches for other tasks.

**Approaches to Solving ML Tasks**

**Trends of the Past**

* **Supervised learning.** Task Driven approach. First of all, you should label data like feeding a model with examples of executable files and saying that this file is malware or not. Based on this labeled data, the model can make decisions about the new data. The disadvantage is the limit of the labeled data.
* **Eensemble learning.**This is an extension of supervised learning while mixing different simple models to solve the task. There are different methods of combining simple models.

**Current Trends**

* **Unsupervised Learning.** Data Driven approach. The approach can be used when there are no labeled data and the model should somehow mark it by itself based on the properties. Usually, it is intended to find anomalies in data and considered to be more powerful in general as it’s almost impossible to mark all data. Currently, it works less precisely than supervised approaches.
* **Semi-supervised learning.** As the name implies, semi-supervised learning tries to combine benefits from both supervised and unsupervised approaches, when there are some labeled data.

**Future Trends (Well, Probably)**

* **Reinforcement learning.** Environment Driven approach can be used when the behavior should somehow react on the changing environment. It’s like a kid who is learning environment by trial and error.
* **Active learning.** It’s more like a subclass of Reinforcement learning that probably will grow into a separate class. Active learning resembles a teacher who can help correct errors and behavior in addition to environmental changes.

**Machine Learning Tasks and Cybersecurity**

Let’s see examples of different methods that can be used to solve Machine Learning tasks and how they are related to cybersecurity tasks.

**Regression**

Regression (or prediction) is simple. The knowledge about the existing data is utilized to have an idea of the new data. Take an example of house prices prediction. In cybersecurity, it can be applied to fraud detection. The features (e.g., the total amount of suspicious transaction, location, etc.) determine a probability of fraudulent actions.

As for the technical aspects of regression, all methods can be divided into two large categories: Machine Learning and Deep Learning. The same is used for other tasks.

For each task, there are examples of ML and DL methods.

**Machine Learning for Regression**

Below is a short list of Machine Learning methods (having their own advantages and disadvantages) that can be used for regression tasks.

* Linear regression
* Polynomial regression
* Ridge regression
* Decision trees
* SVR (Support Vector Regression)
* Random forest

You can find out the detailed explanation of each method [here](https://www.superdatascience.com/wp-content/uploads/2017/02/Regression-Pros-Cons.pdf).

**Deep Learning for Regression**

For regression tasks, the following Deep Learning models can be used:

* Artificial Neural Network (ANN)
* Recurrent Neural Network (RNN)
* Neural Turing Machines (NTM)
* Differentiable Neural Computer (DNC)

**Classification**

Classification is also straightforward. Imagine you have two piles of pictures classified by type (e.g., dogs and cats). In terms of cybersecurity, a spam filter separating spams from other messages can serve as an example. Spam filters are probably the first ML approach applied to Cybersecurity tasks.

The supervised learning approach is usually used for classification where examples of certain groups are known. All classes should be defined in the beginning.

Below is the list related to algorithms.

**Machine Learning for Classification**

* LogisticRegression (LR)
* K-Nearest Neighbors (K-NN)
* Support Vector Machine (SVM)
* KernelSVM
* NaiveBayes
* DecisionTreeClassification
* Random Forest Classification

It’s considered that methods like SVM and random forests work best. Keep in mind that there are no one-size-fits-all rules, and they probably won’t operate properly for your task.

**Deep Learning for Classification**

* Artificial Neural Network
* Convolutional Neural Networks

Deep Learning methods work better if you have more data. But they consume more resources especially if you are planning to use it in production and re-train systems periodically.

**Clustering**

Clustering is similar to classification with the only but major difference. The information about the classes of the data is unknown. There is no idea whether this data can be classified. This is unsupervised learning.

Supposedly, the best task for clustering is [forensic analysis](https://www.peerlyst.com/tags/forensic-analysis). The reasons, course, and consequences of an incident are obscure. It’s required to classify all activities to find anomalies. Solutions to [malware analysis](https://www.peerlyst.com/tags/malware-analysis) (i.e., [malware protection](https://www.peerlyst.com/tags/malware-protection) or [secure email](https://www.peerlyst.com/tags/secure-email) gateways) may implement it to separate [legal](https://www.peerlyst.com/tags/legal) files from outliers.

Another interesting area where clustering can be applied is user behavior analytics. In this instance, application [users](https://www.peerlyst.com/tags/users) cluster together so that it is possible to see if they should belong to a particular group.

Usually, clustering is not applied to solving a particular task in cybersecurity as it is more like one of the subtasks in a pipeline (e.g., grouping users into separate groups to adjust risk values).

**Machine Learning for Clustering**

* K-nearest neighbours (KNN)
* K-means
* Mixturemodel(LDA)
* DBSCn
* Bayesian
* GaussianMixtureModel
* Agglomerative
* Mean-shift

**Deep Learning for Clustering**

* Self-organized Maps (SOM) or Kohonen Networks

**Association Rule Learning (Recommendation Systems)**

Netflix and SoundCloud recommend films or songs according to your movies or music preferences. In cybersecurity, this principle can be used primarily for incident response. If a company faces a wave of incidents and offers various types of responses, a system learns a type of response for a particular incident (e.g., mark it as a false positive, change a risk value, run the investigation). [Risk management](https://www.peerlyst.com/tags/risk-management) solutions can also have a benefit if they automatically assign [risk](https://www.peerlyst.com/tags/risk) values for new [vulnerabilities](https://www.peerlyst.com/tags/vulnerabilities) or misconfigurations built on their description.

There are algorithms used for solving recommendation tasks.

**Machine Learning for Association Rule Learning**

* Apriori
* Euclat
* FP-Growth

**Deep Learning for Association Rule Learning**

* Deep Restricted Boltzmann Machine (RBM)
* Deep Belief Network (DBN)
* Stacked Autoencoder

The latest recommendation systems are based on restricted Boltzmann machines and their updated versions, such as promising deep belief networks.

**Dimensionality Reduction**

Dimensionality reduction or generalization is not as popular as classification, but necessary if you deal with complex systems with unlabeled data and many potential features. You can’t apply clustering because typical methods restrict the number of features or they don’t work. Dimensionality reduction can help handle it and cut unnecessary features. Like clustering, dimensionality reduction is usually one of the tasks in a more complex model. As to cybersecurity tasks, dimensionality reduction is common for face detection solutions — the ones you use in your iPhone.

**Machine Learning for Dimensionality Reduction**

* Principal Component Analysis (PCA)
* Singular-value decomposition (SVD)
* T-distributed Stochastic Neighbor Embedding (T-SNE)
* Linear Discriminant Analysis (LDA)
* Latent Semantic Analysis (LSA)
* Factor Analysis (FA)
* Independent Component Analysis (ICA)
* Non-negative Matrix Factorization (NMF)

You can find more on dimensionality reduction [here](https://arxiv.org/pdf/1403.2877.pdf) (including the general description of the methods and their features).

**Generative Models**

The task of generative models differs from the above-mentioned ones. While those tasks deal with the existing information and associated decisions, generative models are designed to simulate the actual data (not decisions) based on the previous decisions.

The simple task of offensive cybersecurity is to generate a list of input parameters to test a particular application for Injection vulnerabilities.

Alternatively, you can have a vulnerability scanning tool for web applications. One of its modules is testing files for unauthorized access. These tests are able to mutate existing filenames to identify the new ones. For example, if a crawler detected a file called login.php, it’s better to check the existence of any backup or test its copies by trying names like *login\_1.php, login\_backup.php, login.php.2017*. Generative models are good at this.

**Machine Learning for Generative Models**

* Markov Chains
* Genetic algorithms

**Deep Learning for Generative Models**

* Variational Autoencoders
* Generative adversarial networks (GANs)
* Boltzmann Machines

Recently, GANs showed impressive results. They successfully mimic a video. Imagine how it can be used for generating examples for fuzzing.

**Cybersecurity Tasks and Machine Learning**

Instead of looking at ML tasks and trying to apply them to cybersecurity, let’s look at the common cybersecurity tasks and Machine Learning opportunities. There are three dimensions (Why, What, and How).

The first dimension is a goal, or a task (e.g., detect threats, predict attacks, etc.). According to [Gartner’s PPDR](https://www.gartner.com/document/3286317) model, all security tasks can be divided into five categories:

1. Prediction;
2. Prevention;
3. Detection;
4. Response;
5. Monitoring.

The second dimension is a technical layer and an answer to the “What” question (e.g., at which level to monitor issues). Here is the list of layers for this dimension:

* network (network traffic analysis and intrusion detection);
* endpoint (anti-malware);
* application (WAF or database firewalls);
* user (UBA);
* process (anti-fraud).

Each layer has different subcategories. For example, network security can be Wired, Wireless or Cloud. Rest assured that you can’t apply the same algorithms with the same hyper parameters to both areas, at least in near future. The reason is the lack of data and algorithms to find better dependencies of the three areas so that it’s possible to change one algorithm to different ones.

The third dimension is a question of “How” (e.g., how to check security of a particular area):

* in transit in real time;
* at rest;
* historically;
* etc.

For example, if you are about endpoint protection, looking for the intrusion, you can monitor processes of an executable file, do static binary analysis, analyze the history of actions in this endpoint, etc.

Some tasks should be solved in three dimensions. Sometimes, there are no values in some dimensions for certain tasks. Approaches can be the same in one dimension. Nonetheless, each particular point of this three-dimensional space of cybersecurity tasks has its intricacies.

It’s difficult to detail them all so let’s focus on the most important dimension — technology layers. Look at the cybersecurity solution from this perspective.

**Machine Learning for Network Protection**

Network protection is not a single area but a set of different solutions that focus on a protocol such as Ethernet, wireless, SCADA, or even virtual networks like SDNs.

Network protection refers to well-known Intrusion Detection System (IDS) solutions. Some of them used a kind of ML years ago and mostly dealt with signature-based approaches.

ML in network security implies new solutions called Network Traffic Analytics (NTA) aimed at in-depth analysis of all the traffic at each layer and detect attacks and anomalies.

How can ML help here? There are some examples:

* regression to predict the network packet parameters and compare them with the normal ones;
* classification to identify different classes of network attacks such as scanning and spoofing;
* clustering for forensic analysis.

You can find at least 10 papers describing diverse approaches in academic research papers.

More resources:

* [Machine Learning Techniques for Intrusion Detection](https://arxiv.org/abs/1312.2177v2)
* [Long Short Term Memory Networks for Anomaly Detection in Time Series](https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2015-56.pdf)
* [Anomaly Detection Framework Using Rule Extraction for Efficient Intrusion Detection](https://arxiv.org/abs/1410.7709v1)
* [A survey of network anomaly detection techniques](https://www.gta.ufrj.br/~alvarenga/files/CPE826/Ahmed2016-Survey.pdf)
* [Shallow and Deep Networks Intrusion Detection System: A Taxonomy and Survey](https://arxiv.org/abs/1701.02145v1)
* [Deep Packet: A Novel Approach For Encrypted Traffic Classification Using Deep Learning](https://arxiv.org/abs/1709.02656v3)
* [Performance Comparison of Intrusion Detection Systems and Application of Machine Learning to Snort System](https://arxiv.org/pdf/1710.04843v2.pdf)
* [Evaluation of Machine Learning Algorithms for Intrusion Detection System](https://arxiv.org/pdf/1801.02330v1.pdf)
* [One Class collective Anomaly Detection based on LSTM](https://arxiv.org/pdf/1802.00324.pdf)
* [Network Traffic Anomaly Detection Using Recurrent Neural Networks](https://arxiv.org/abs/1803.10769v1)
* [Sequence Aggregation Rules for Anomaly Detection in Computer Network Traffic](https://arxiv.org/pdf/1805.03735v2.pdf)
* [Big collection of all approaches for IDS](https://arxiv.org/pdf/1806.03517v1.pdf)

**Machine Learning for Endpoint Protection**

The new generation of anti-viruses is Endpoint Detection and Response. It’s better to learn features in executable files or in the process behavior.

Keep in mind that if you deal with Machine Learning at endpoint layer, your solution may differ depending on the type of endpoint (e.g., workstation, server, container, cloud instance, mobile, PLC, IoT device). Every endpoint has its own specifics but the tasks are common:

* regression to predict the next system call for executable process and compare it with real ones;
* classification to divide programs into such categories as malware, spyware and ransomware;
* clustering for malware protection on secure email gateways (e.g., to separate legal file attachments from outliers).

Academic papers about endpoint protection and malware specifically are gaining popularity. Here are a few examples:

* [Malware Detection by Eating a Whole EXE](https://arxiv.org/pdf/1710.09435v1.pdf)
* [Deep Learning at the shallow end: Malware classification for non-domain experts](https://arxiv.org/abs/1807.08265v1)
* [TESSERACT: Eliminating Experimental Bias in Malware Classification across Space and Time](https://arxiv.org/abs/1807.07838v1)

**Machine Learning for Application Security**

Application security is my favorite area, by the way, especially ERP Security.

Where to use ML in app security? — WAFs or Code analysis, both static and dynamic. To remind you, Application security can differ. There are web applications, databases, ERP systems, SaaS applications, micro services, etc. It’s almost impossible to build a universal ML model to deal with all threats effectively in near future. However, you can try to solve some of tasks.

Here are examples what you can do with Machine Learning for application security:

* regression to detect anomalies in HTTP requests (for example, XXE and SSRF attacks and auth bypass);
* classification to detect known types of attacks like injections (SQLi, XSS, RCE, etc.);
* clustering user activity to detect DDOS attacks and mass exploitation.

More resources:

* [Adaptively Detecting Malicious Queries in Web Attacks](https://arxiv.org/pdf/1701.07774.pdf)
* [Neural Classification of Malicious Scripts: A study with JavaScript and VBScript](https://arxiv.org/abs/1805.05603v1)
* [URLNet: Learning a URL Representation with Deep Learning for Malicious URL Detection](https://arxiv.org/abs/1802.03162v2)

**Machine Learning for User Behavior**

This started as Security Information and Event Management (SIEM).

SIEM was able to solve numerous tasks if configured properly including user behavior search and ML. Then the UEBA solutions declared that SIEM couldn’t handle new, more advanced types of attacks and constant behavior change.

The market has accepted the point that a special solution is required if the threats are regarded from the user level.

However, even UEBA tools don’t cover all things connected with different user behavior. There are domain users, application users, SaaS users, social networks, messengers, and other accounts that should be monitored.

Unlike malware detection focusing on common attacks and the possibility to train a classifier, user behavior is one of the complex layers and unsupervised learning problem. As a rule, there is no labeled dataset as well as an idea of what to look for. Therefore, the task of creating a universal algorithm for all types of users is tricky in user behavior area. Here are the tasks that companies solve with the help of ML:

* regression to detect anomalies in User actions (e.g., login in unusual time);
* classification to group different users for peer-group analysis;
* clustering to separate groups of users and detect outliers.

More resources:

* Detecting Anomalous User Behavior Using an Extended Isolation Forest Algorithm: An Enterprise Case Study
* Deep Learning for Unsupervised Insider Threat Detection in Structured Cybersecurity Data Streams

**Machine Learning for Process Behavior**

The process area is the last but not least. While dealing with it, it’s necessary to know a business process in order to find something anomalous. Business processes can differ significantly. You can look for fraud in banking and retail system or a plant floor in manufacturing. The two are totally different, and they demand a lot of domain knowledge. In Machine Learning feature engineering (the way you represent data to your algorithm) is essential to achieve results. Similarly, features are different in all processes.

In general, there are examples of tasks in the process area:

* regression to predict the next user action and detect outliers such as credit card fraud;
* classification to detect known types of fraud;
* clustering to compare business processes and detect outliers.

You can find research papers related to banking fraud as ICS and SCADA systems security is much less represented.

More resources:

* [Fraud with autoencoders](https://shiring.github.io/machine_learning/2017/05/01/fraud)
* [A Survey of Credit Card Fraud Detection Techniques: Data and Technique Oriented Perspective](https://arxiv.org/abs/1611.06439v1)
* [Anomaly detection; Industrial control systems; convolutional neural networks](https://arxiv.org/abs/1806.08110v1)

**Machine Learning Cybersecurity Books**

If you want to learn more about Machine Learning in cybersecurity, there are books that can help:

* [AI for Cybersecurity by Cylance](https://pages.cylance.com/en-us-introduction-to-ai-book.html?_ga=2.89683291.1595385041.1538052662-139740503.1538052662) (2017)- Short but good introduction to basics of ML for Cybersecurity. Good practical examples.

* [Machine Learning and Security by O’reilly](http://shop.oreilly.com/product/0636920065555.do) (January 2018) — Best book so far about this topic but very few examples of Deep Learning and mostly general ML.
* [Machine Learning For Penetration Testers, by Packt](https://www.packtpub.com/networking-and-servers/mastering-machine-learning-penetration-testing) (July 2018)- Less fundamental than the previous one, but has more Deep Learning approaches.
* [Malware Data Science: Attack Detection and Attribution](https://nostarch.com/malwaredatascience) (September 2018) — As seen from the title, this book is focused on malware. It was just released by the time of writing this article, so I can’t give any feedback so far. But I bet it is a must for everyone from endpoint protection teams.

**Conclusion**

There are more areas left. I have outlined the basics. On the one hand, Machine Learning is definitely not a silver-bullet solution if you want to protect your systems. Undoubtedly, there are many issues with interpretability (particularly for Deep Learning algorithms), but humans also cannot interpret their own decisions, right?

On the other hand, with the growing amount of data and decreasing number of experts, ML is an only remedy. It works now and will be mandatory soon. It is better to start right now.

Keep in mind, hackers are also starting to use ML in their attacks. My next article will reveal how exactly attackers can utilize ML.