

**Computer machine learning**

**COMP4388**

Machine learning

Dr: [Radi Jarrar](https://ritaj.birzeit.edu/bzu-msgs/type?mttid=104&classid=183185)

Second Project

* **Group member:** 
  + Mohammad AbuBader-1190478.
  + Amir Eleyan-1191076.
* **Date:**
* Monday, January 10, 2022.

Table of Contents

[ ***Introduction:*** 3](#_Toc92680727)

[ ***EDA steps:*** 4](#_Toc92680728)

[ ***Algorithm used*** 5](#_Toc92680729)

[ ***What is a Gradient Boosting algorithm?*** 5](#_Toc92680730)

[ ***Results:*** 5](#_Toc92680731)

# ***Introduction:***

Initially, we analyzed and processed the data that the module will train on, and we did this through several steps (checking for null values, selecting the most important features, etc...). After that, we started training the module on the data, starting from the complex module back to the simple module, and then back to using the complex module. Gradient Boosting algorithm, which gave us the best result and the highest accuracy.

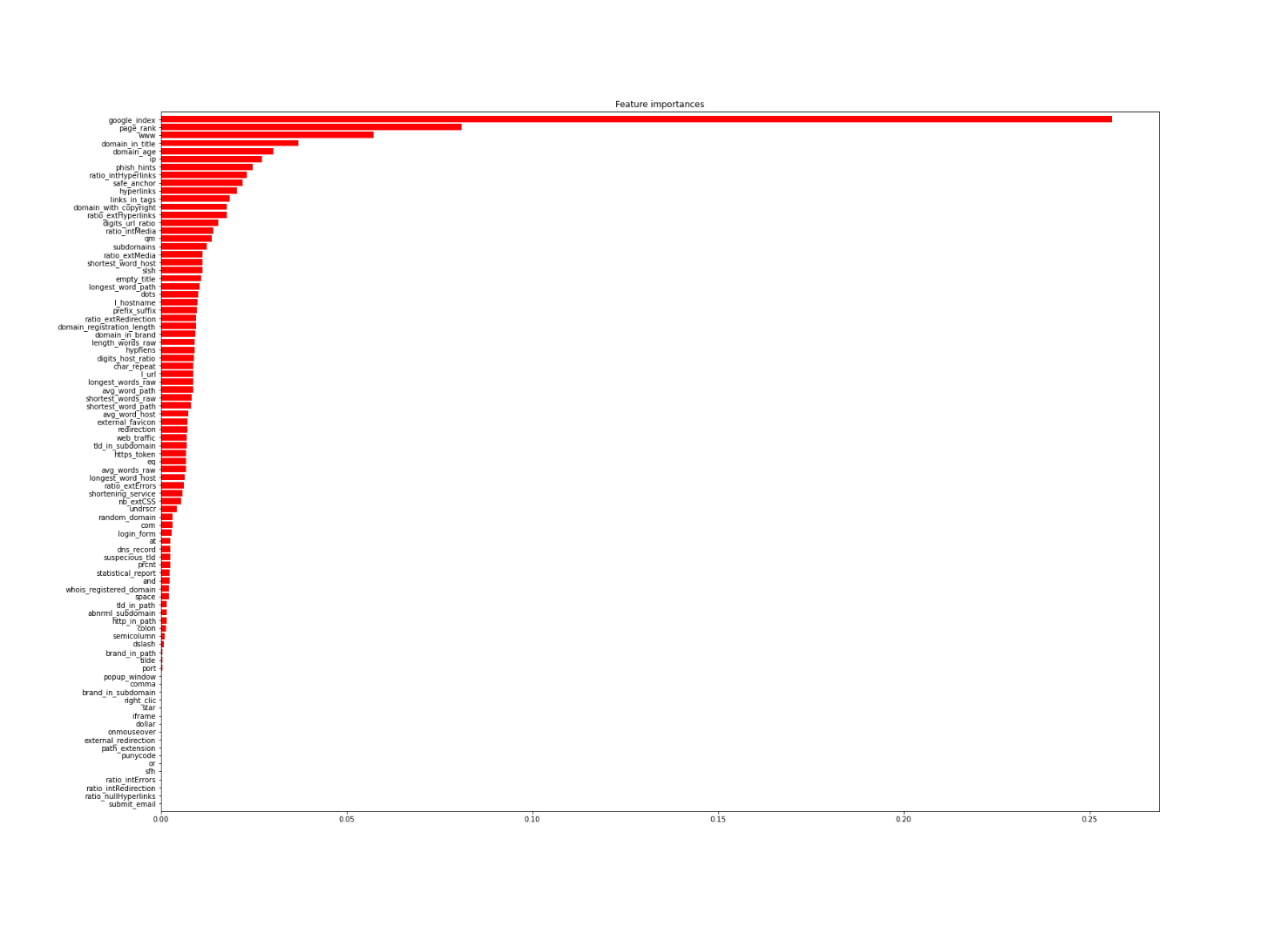
# ***EDA steps:***

First of all, the training and testing data was read, delete the spaces from the cells if they exist, and then the data was described (mean, standard deviation, max, etc...). Any row all his value is zero was deleted, it was checked that there were no empty cells (missing values). The target variable was transformed into a binary classification (1 = good, 0 = bad).

The feature selection process started by calculating the "Correlation" and "Feature importance" between the dependent variables and the independent (status) variables.

Feature importance scores can provide insight into the dataset. The relative scores can highlight which features may be most relevant to the target, and the converse, which features are the least relevant. This may be interpreted by a domain expert and could be used as the basis for gathering more or different data.

Based on the above two results, the features that have a significant impact on the target class were selected, and the features that had no effect (i.e. negligible) were deleted.



# ***Algorithm used***

Several modules were used such as the SVM algorithm, KNN algorithm, decision tree, logistic regression, naive basis, and gradient boosting algorithm. Gradient boosting algorithm had better accuracy and better module.

# ***What is a Gradient Boosting algorithm?***

Gradient Boosting is an iterative functional gradient algorithm, i.e an algorithm that minimizes a loss function by iteratively choosing a function that points towards the negative gradient; a weak hypothesis.

Gradient Boosting has three main components:

Loss Function - The role of the loss function is to estimate how good the model is at making predictions with the given data. This could vary depending on the problem at hand. For example, if we're trying to predict the weight of a person depending on some input variables (a regression problem), then the loss function would be something that helps us find the difference between the predicted weights and the observed weights. On the other hand, if we're trying to categorize if a person will like a certain movie based on their personality, we'll require a loss function that helps us understand how accurate our model is at classifying people who did or didn't like certain movies.

Weak Learner - A weak learner is one that classifies our data but does so poorly, perhaps no better than random guessing. In other words, it has a high error rate. These are typically decision trees (also called decision stumps, because they are less complicated than typical decision trees).

Additive Model - This is the iterative and sequential approach of adding the trees (weak learners) one step at a time. After each iteration, we need to be closer to our final model. In other words, each iteration should reduce the value of our loss function.

# ***Results:***

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Measure name** | **Model score** | **Accuracy** | **Precision** | **Recall** | **F-score** | **MSE** | **Bias** | **Variance** |
| **Result** | 99.9 | 97.2 | 97.3 | 97.1 | 97.2 | 0.034 | 0.026 | 0.009 |

So, our module is a good balance.