Machine Learning

Course Project Report (Classification)

(Final Report, Team No: 10)

Title of the project: Website Phishing Classification
Student 1: Tarun Kumar Reddy
Email: tarun.g-25@scds.saiuniversity.edu.in

1. Introduction

- Phishing is a cyber-attack where attackers deceive targeted people by stealing their sensitive information by making them open links to malicious sites or ransomware.
- In today's world, the internet has developed to a very sophisticated level and is available everywhere. It has increased connectedness and the risk of cyber-attacks.
- There are widespread phishing attacks worldwide, and hence, it is necessary to detect a website's legitimacy and trustworthiness.
- Websites contain some identifiers through which they can be flagged as being phishy or legitimate.
- In this report, we seek to classify the websites collected through different attributes such as url length, IP address, character repeat, etc., using different classifier models such as Softmax Regression, Linear SVMs, Polynomial SVMs, Radial Basis SVMs, Decision Trees, Random Forest, AdaBoost, and Gradient Boost.

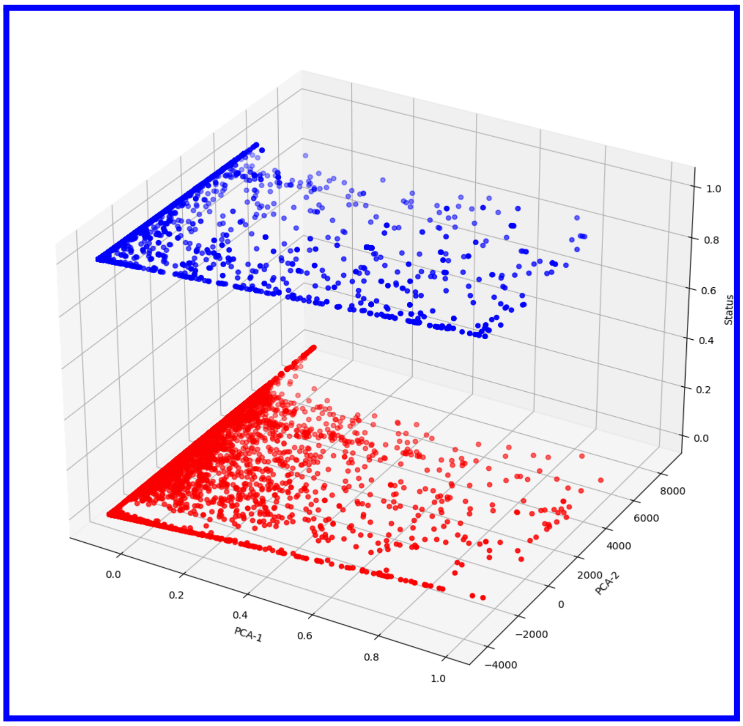
2. Dataset and Features

- The given dataset contains 11430 data samples with 89 features. Out of 89 features 87 features are of numerical data type and 2 out of them are of categorical type.
- The URL of the websites is one of the objects and it is dropped out of the dataset since it has no value to our machine learning model.
- The target variable, "status", is an object type and has two unique values, 'phishing' and 'legitimate'. Later, they are mapped to numbers using LabelEncoder()
- Used "describe()" to understand the descriptive statistics of the data frame.
- Since this is a classification problem, we can only investigate the data separation based on the target variable using dimensionality reduction techniques.

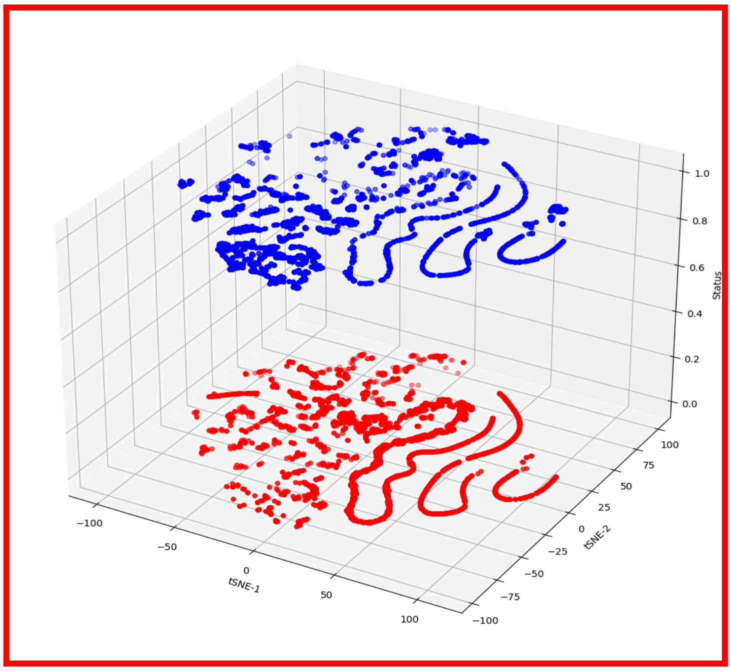
Principal Component Analysis:

Principal Component Analysis is the most famous dimensionality reduction technique that works on the principle of Singular Value Decomposition.

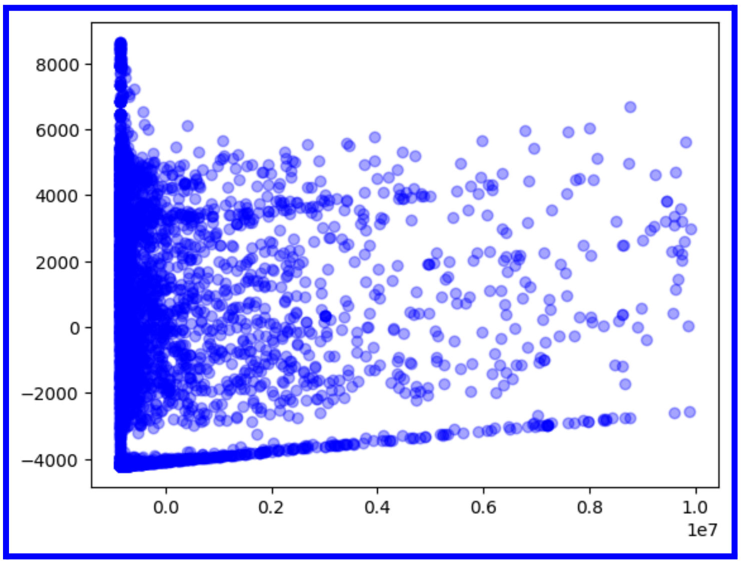
Reduced the data in two dimensions using PCA and plotted the data against the target variable to see the data separation.



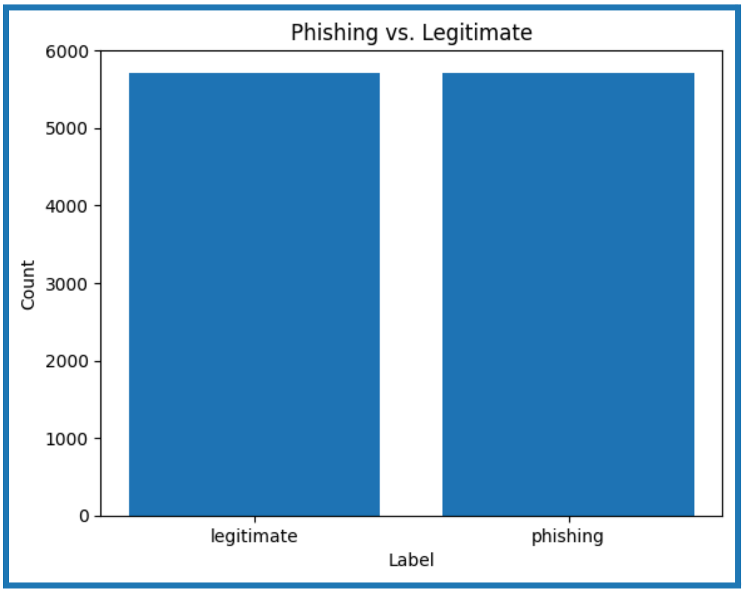
t-SNE Algorithm:
t-SNE is another algorithm that is used in dimensionality reduction; it works on the basis of probability distributions. Plotted the data against the target variable after reducing the data into two dimensions using t-SNE. Following is the visualisation:



→ Also plotted the PCA vs PCA plot, and that gives the following visualisation:



→ A bar plot is plotted on the status attribute(Target variable) and the count. Since both the labels' counts are equal, the plot has bars of the same height.



3. Methods

Following are the various methods used in this project.

3.1 Baseline - Logistic/Softmax Regression

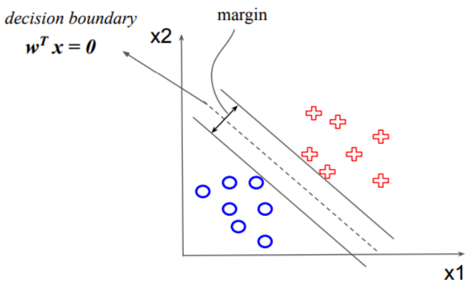
- Logistic regression is a binary class classifier that works on estimates of probabilities of the given instances. It makes decision boundaries similar to regression, but it provides a sigmoid output instead of a continuous value.
- It minimizes the cross entropy, and it is the cost function for the logistic regression.

$$J(\boldsymbol{\theta}) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}log(\hat{p}^{(i)}) + (1-y^{(i)})log(1-\hat{p}^{(i)})]$$

3.2 Support Vector Machine

- SVM is one of the most popular choices for Machine Learning in the context of Classification.
- SVMs are capable of performing both binary and multi-class classification.
- They use extreme data points as support vectors to make decision boundaries and find a hyperplane that separates the point cloud.

  Decision boundaries:



- Linear SVM Regression:
  - Linear SVMs try to find the hyperplane that best represents the point cloud.
- Kernel SVM Regression:

  Kernel SVMs transform the data into higher dimensions, and then a hyperplane that best represents the point cloud is found. It can be achieved in two ways:
  - Polynomial kernel
    - It will transform the data into higher dimensions by converting the data into polynomial features

      $$\phi(x_1,x_2)=(z_1,z_2,z_3)=(x_1,x_2,x_1^2+x_2^2)$$
  - Radial Basis Function kernel
    - Rbf is the widely used Kernel.
    - It is also called the Gaussian Kernel since it uses the Gaussian function to transform the data into higher dimensions

      $$K(X_1,X_2)=\exp(-\frac{\|X_1-X_2\|^2}{2\sigma^2})$$
    - The two parameters used in the above equation are $\mathbf{X_{i,}}$ which is the **mean** and $\boldsymbol{\sigma}$**,** which is the **standard deviation**

3.3 Decision Tree Classifier

- A decision tree classifier is a decision tree technique that solves classification problems in machine learning.
- This machine-learning approach builds a binary tree with Root nodes, split nodes and leaf nodes.
- Every node has a threshold value of a particular feature and navigates across the tree by comparing the instance value with the threshold.

- The measure of impurity within a node is called Gini. This is the proportion of class instances among all the instances in the node

$$G_i = 1 - \sum_{k=1}^{n} {p_{i,k}}^2$$

$\boldsymbol{p_{i,k}}$ is the ratio of class $\boldsymbol{k}$ instances among training instances in $\boldsymbol{i^{th}}$ node

- Decision trees are trained using the CART algorithm. The CART algorithm minimizes the cost function to obtain the threshold values in the nodes for the respective features.

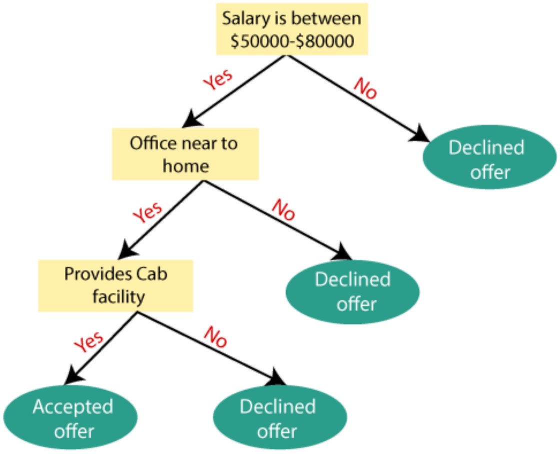
$$J(k, t_k) = \frac{m_{\text{left}}}{m} G_{\text{left}} + \frac{m_{\text{right}}}{m} G_{\text{right}}$$

Where:

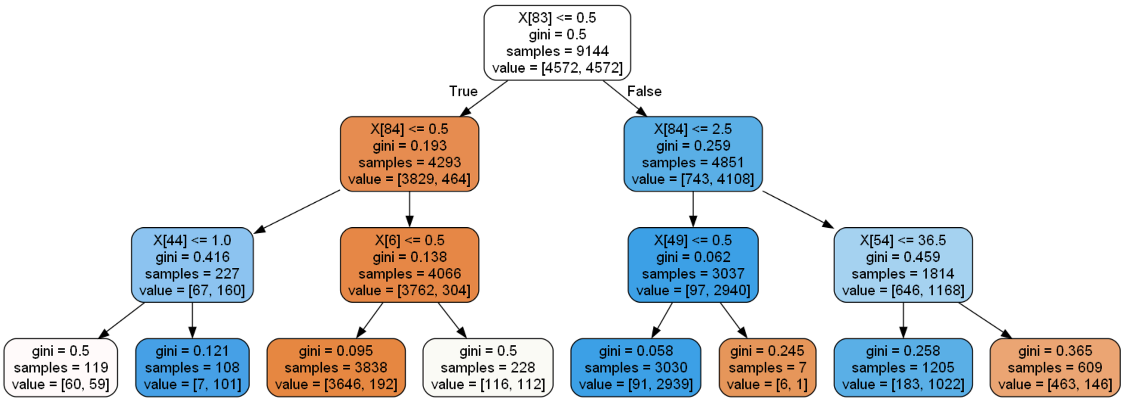
$G_{\textbf{left/right}}$ captures the impurity in the left/right subset

$M_{\textbf{left/right}}$ captures the number of instances in the left/right subset

- The CART algorithm starts by splitting the Dataset into two based on a single feature, $\boldsymbol{K}$ and a threshold $\mathbf{T_k}$.
- ($\boldsymbol{K}$ and $\boldsymbol{T_k}$) are chosen to produce the purest subset. This process recursively until the given max_depth or mean squared error becomes zero.
- Below example is an example of a Decision Tree classifier:

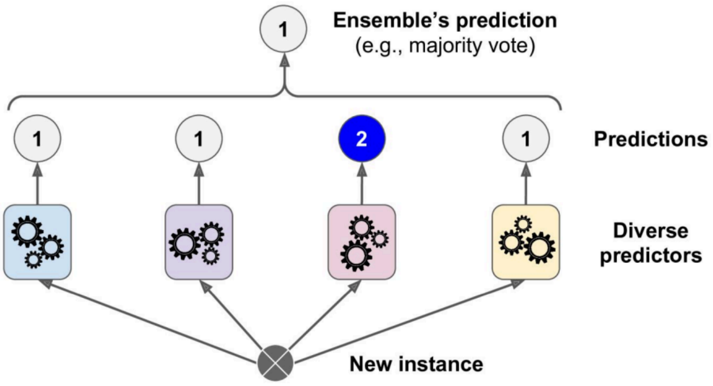


- Decision Trees can be visualized using the **export_graphviz** method in the **sklearn.tree** module. It is hard to visualize trees with a depth of more than 3, but the decision tree with a depth of 3 can be visualized as follows:



3.4 Random Forest Classifier

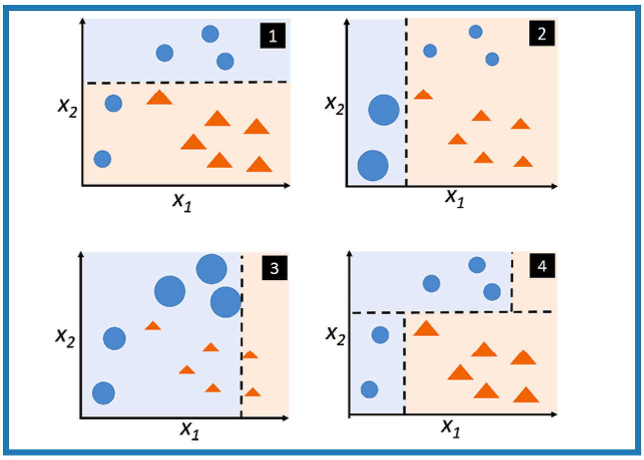
- A random forest classifier is an ensemble learning technique that trains different models and aggregates them to predict a single output.
- In the case of the Random Forest Classifier, different Decision Trees are trained on the random subsets split from the total data set(With replacement). This type of Ensemble learning is called Bagging(Bootstrap Aggregating)
- Trained models use statistical measure Mode to predict the final output



3.5 AdaBoost Classifier

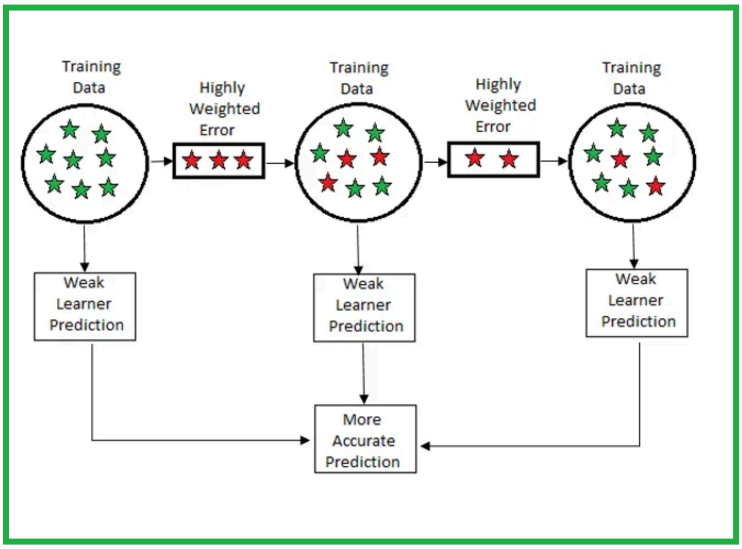
- AdaBoost (Adaptive Boosting) is another type of ensemble learning technique.
- It is sequential learning, unlike Bagging. It combines several weak learners to make a strong learner.

- It trains different models but the trained model is a corrected model from the predecessor model. It gives more weight to the misclassified data points.



3.6 GradientBoost Classifier

- Gradient boosting classifier is also one of the ensemble learning techniques.
- It is also a sequential learning technique; it trains several models, and the new model is produced by correcting the error in the previous model. It gives more weight to the error in the last model and aggregates all the models to make new predictions.



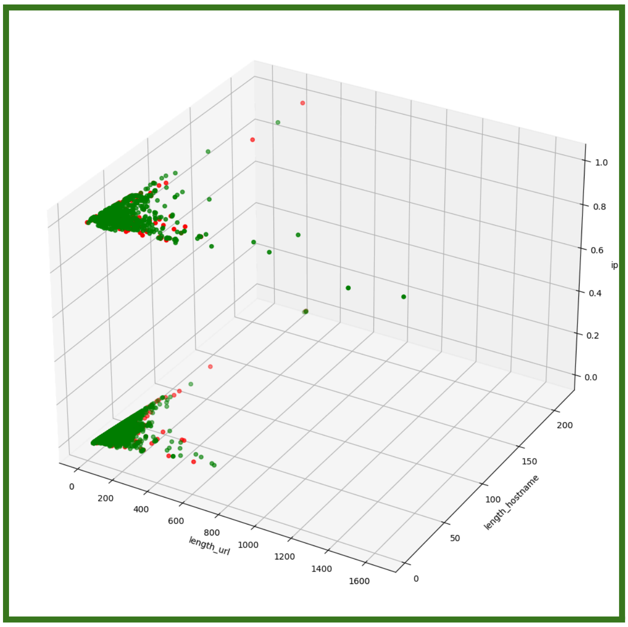
3.7 Hyperparameter Tuning

- A hyperparameter is a parameter of the learning algorithm which is not affected by the learning algorithm itself
- It is set before training and remains constant throughout the training
- It is an important part of building a machine learning system(Géron)
- Examples of hyperparameters:
  - In RBF Kernel SVM, we encounter the hyperparameters of C and gamma.
  - In Polynomial Kernel SVM, we encounter the hyperparameters of C, Gamma and degree.
  - In the Random Forest Classifier, we encounter the hyperparameter max_depth.

4. Experiments & Results

4.1 Protocol

- The dataset was divided into two subsets namely the training and the testing data with a proportion of 80% and 20% respectively.
- The split data set can be visualized by taking any 3 features in 3-dimensional space:



- The training data is used to train the model and the testing data can be used to evaluate the model performance.
- This can be achieved using the built-in function in the Scikit Learn library.
- To increase the performance of the Gradient Descent Algorithm and the efficiency of the model, we perform standard scaling (Standardisation) on the data set.
- Standardization scales the data into a particular range. It is done by subtracting every data point from its mean and dividing it by the standard deviation.
- Standardisation: $\mathbf{X}' = (\mathbf{X} - \mu) / \sigma$
- We use the StandardScaler class from Scikit Learn to perform standardization.
- We also use the MinMaxScaler class from Scikit Learn to perform normalization.
- Generally, we noticed that with feature scaling, optimization is faster and optimized slower without feature scaling.

Feature Selection:

- Feature selection is a way to select a few features that are highly correlated with the model performance.
- There are different techniques for feature selection, and Sklearn provides methods for all the techniques.
- The purpose of feature selection is to increase the model accuracy but for this classification problem feature selection is not helping to increase the accuracy and hence feature selection is performed to compare.

4.2 Results

- Different models are trained using the Logistic Regression with LBFGS and Newton CG solvers, Linear, RBF, and Polynomial SVMs.
- The Accuracy Scores and F1 Scores of different models are represented in the form of two tables below:

Table 4.2.1 Accuracy Scores of Different Models:

Model		Without Standard Scaling	With Standard Scaling
Logistic Regression	LBFGS solver	93.00%	93.70%
	Newton CG solver	93.13%	93.70%
Linear SVM		NA	94.0%
Kernel SVM (RBF)		95.18%	96.45%
Kernel SVM (Polynomial)		93.00%	93.70%
Decision Trees		94.1%	NA
Random Forest Classifier		96.2%	NA
AdaBoost Classifier		96.4%	90.1%
Gradient Boost Classifier		95.7%	95.6%

Table 4.2.2 F1 Scores of Different Models:

Model		Precision	Recall	F1 Score
Logistic Regression	LBFGS	0.93	0.93	0.93
	Newton CG	0.93	0.93	0.93
Linear SVM		0.93	0.94	0.94
Kernel SVM (RBF)		0.95	0.96	0.95
Kernel SVM (Polynomial)		0.95	0.96	0.95
Decision Trees		0.94	0.94	0.94
Random Forest		0.96	0.96	0.96
Ada Boost		0.96	0.96	0.96
Gradient Boost		0.96	0.96	0.96

Table 4.2.3 Cross Validation Scores of Different Models:

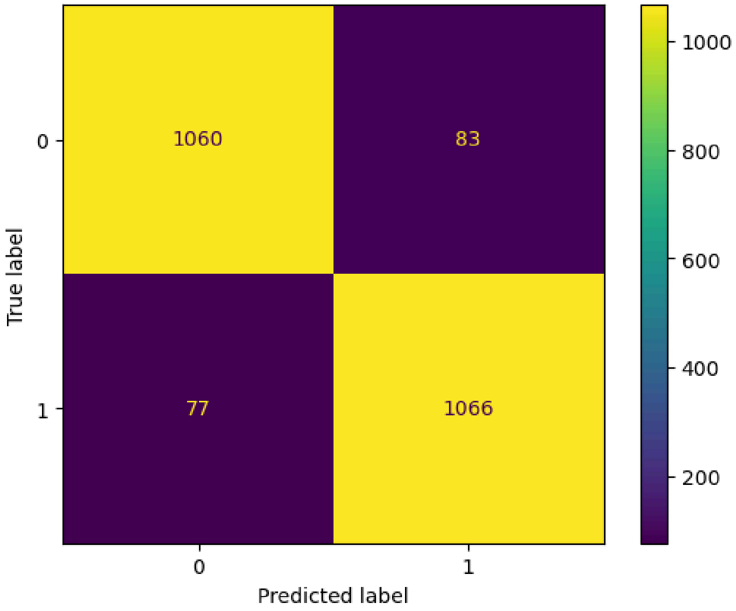
Cross Validation:

Cross-validation is a technique to identify the sensitivity of the model to new instances. It divides the dataset into N subsets and trains the models on different subsets. It calculates the mean and standard deviation of the scores. If the mean is low or the standard deviation is high or both low and high respectively the model will not perform well since it is sensitive to new instances. The trained models for this classification problem are performed well in a 5 fold Cross-validation which means the trained models are not sensitive to new instances.

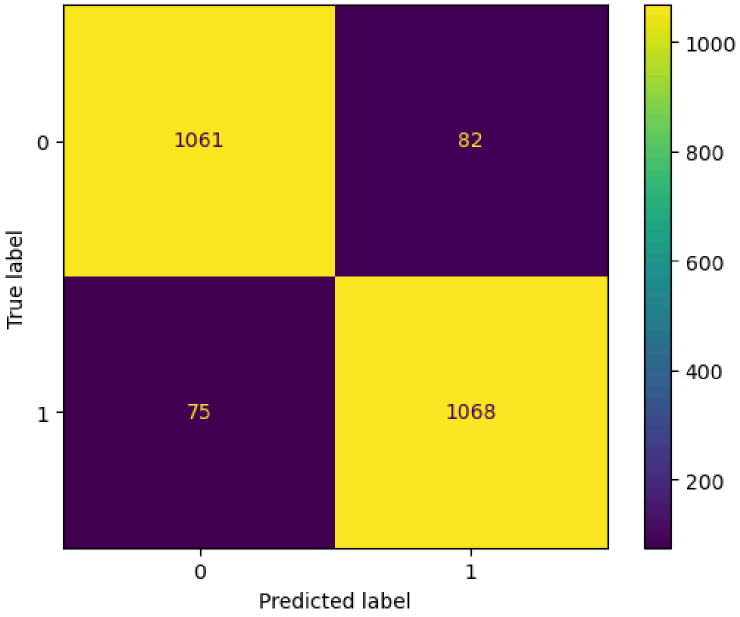
Model		Mean	Standard Deviation
Logistic Regression	LBFGS	0.7845596055765787	0.007840880781187946
	Newton CG	0.9426956653861385	0.006728249789364023
Linear SVM		Not Available	Not Available
Kernel SVM (RBF)		0.705380611184024	0.004953408222273073
Kernel SVM (Polynomial)		Not Available	Not Available
Decision Trees		0.9374458786413401	0.0073684064683657785
Random Forest		0.9374458786413401	0.0073684064683657785
Ada Boost		0.9668639102808747	0.005821304767024926
Gradient Boost		0.9635827711332017	0.0055708499780746795

The confusion matrix for the different models is attached below:

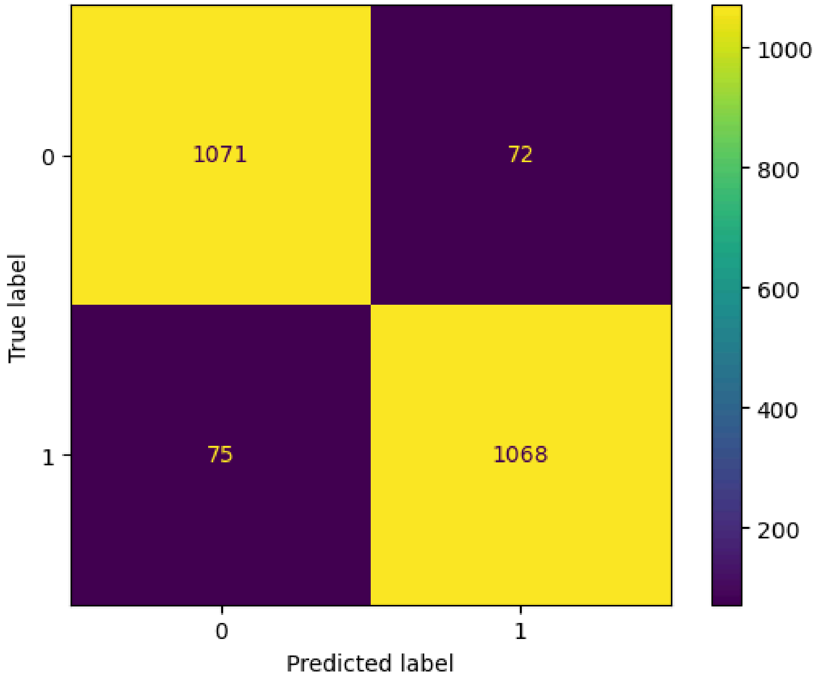
Logistic Regression- LBFGS Solver:



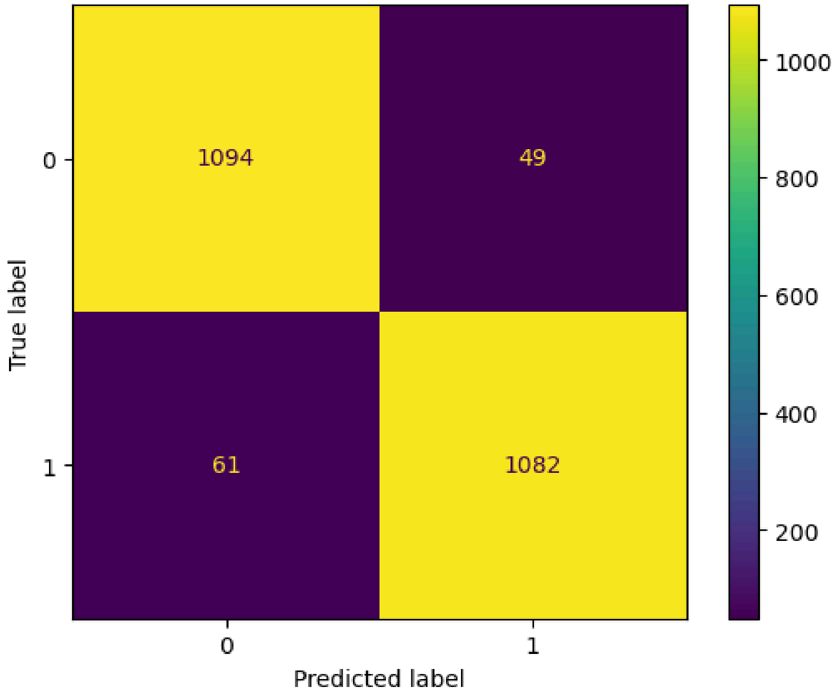
Logistic regression- Newton CG Solver:



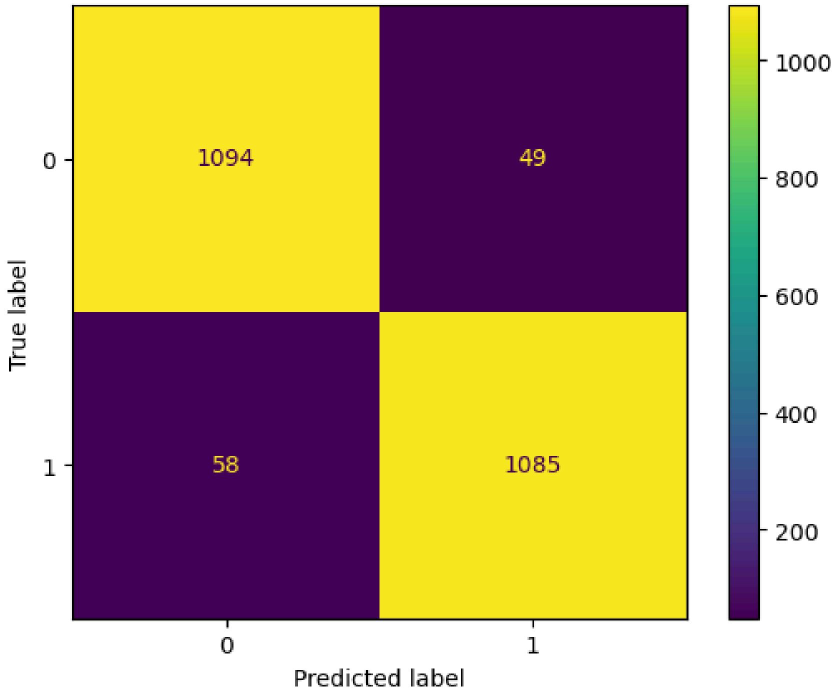
Linear SVM:



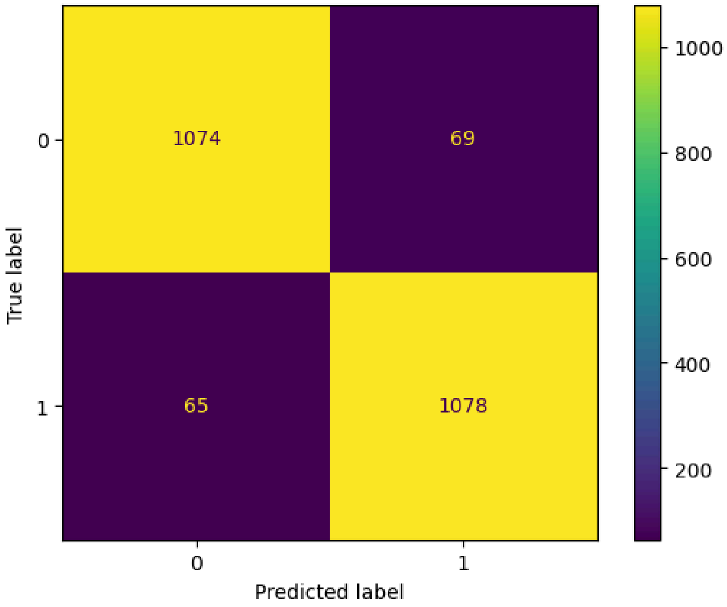
Radial Basis Function SVM:



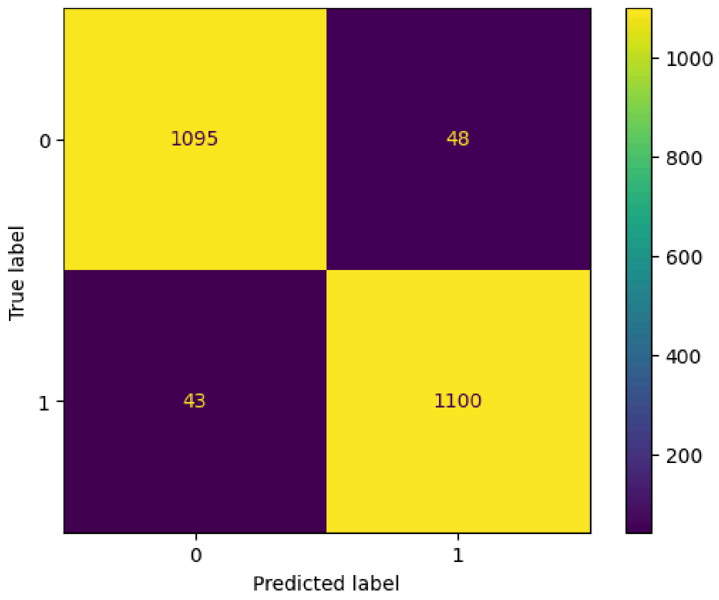
Polynomial SVM:



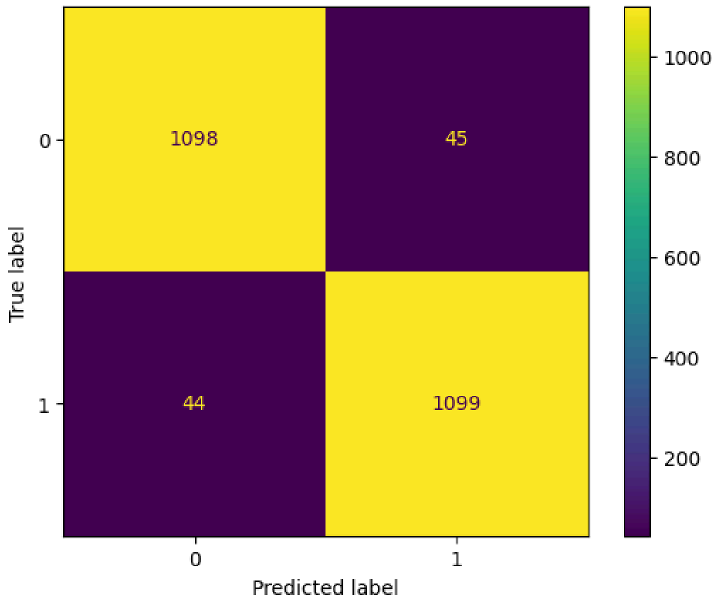
Decision Trees:



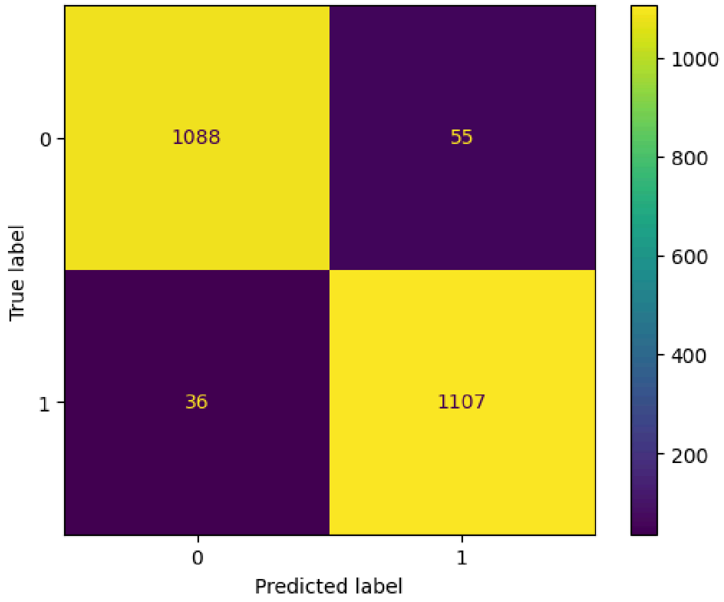
Random Forest Classifier:



AdaBoost Classifier:



Gradient Boost Classifier:



5. Discussion

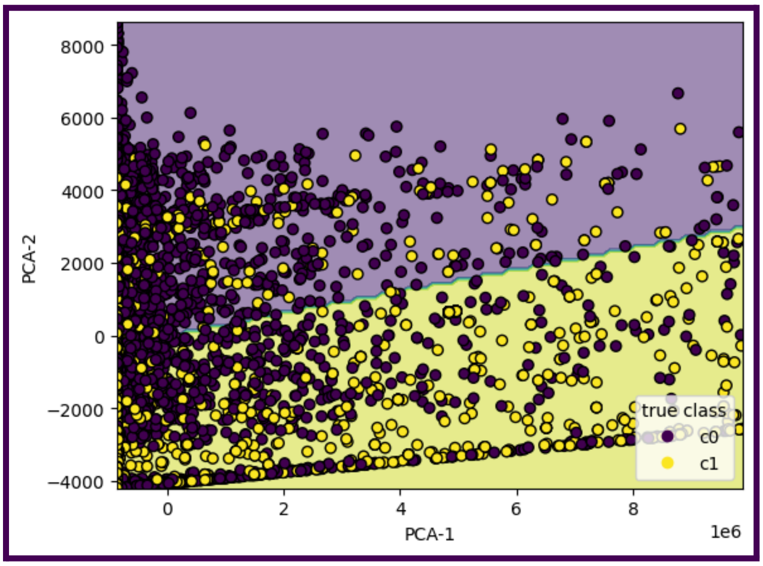
- From the above and preliminary analysis of the dataset, two models seem to give us some good accuracy scores of greater than 96%. Which is pretty good accuracy in the given Classification problem.
- Hyperparameter tuning is performed on the different models that are required. Since it is a large dataset Grid Search technique is not very efficient whereas the Random Search technique was helpful. A manual hyperparameter was also used in some places since the random search technique was not precise.
- With tuned hyperparameters, Random Forest, AdaBoost, and the Gradient Boost classifier are promising very good scores.
- SVMs also require hyperparameter tuning; however, they take a very long time to run, and sometimes, convergence is doubtful. So, SVMs are unsuitable for a huge dataset like this and have minimal inclusion in the project.
- 5-fold cross-validation is performed on all the models(Except Linear and Polynomial SVMs) to evaluate and select the models since it is a large dataset 10 fold cross-validation is not efficient. And all the models excelled in cross-validation.

6. Conclusion

- After performing all the models and performing cross-validation AdaBoost classifier is the well-performed model among all other models with accuracy and an F1 Score of 96%.
- The baseline model, logistic regression, is better, giving an accuracy of 93%.

Additional:

Classification boundaries are visualized after training the data with logistic regression using PCA-1 and PCA-2 as the features of the data set and status features as the target variable.



References:

1. **Image used for Decision Tree Classifier:** **https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm**
2. **Image used for AdaBoost: https://vitalflux.com/adaboost-algorithm-explained-with-python-example/**
3. **Image used for Gradient Descent Classifier:** **https://pub.towardsai.net/fully-explained-gradient-boosting-technique-in-supervised-learning-d3e293ca70e1**
4. **"The Machine Learning Landscape."** ***Hands-on Machine Learning with Scikit-Learn, Keras and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*****, by Aurélien Géron, 3rd ed., O'Reilly, Sebastopol, CA, 2023, pp. 70–71.**