EasyChair Preprint

№ 9623

An Experiment on Feature Selection Using Logistic Regression

Raisa Islam, Subhasish Mazumdar and Rakibul Islam

January 26, 2023

An Experiment on Feature Selection using Logistic Regression

Raisa Islam[1], Subhasish Mazumdar[2], and Rakibul Islam[3]
Dept of Computer Science & Engineering
New Mexico Institute of Mining and Technology
Socorro, NM 87801 USA
[1]raisa.islam@student.nmt.edu, [2]subhasish.mazumdar@nmt.edu, [3]mdrakibul.islam@student.nmt.edu

***Abstract*—In supervised machine learning, feature selection plays a very important role by potentially enhancing explainability and performance as measured by computing time and accuracy-related metrics. In this paper, we investigate a method for feature selection based on the well-known L1 and L2 regularization strategies associated with logistic regression (LR). It is well known that the learned coefficients, which serve as weights, can be used to rank the features. Our approach is to synthesize the findings of L1 and L2 regularization. For our experiment, we chose the CIC-IDS2018 dataset [1] owing partly to its size and also to the existence of two problematic classes that are hard to separate. We report first with the exclusion of one of them and then with its inclusion. We ranked features first with L1 and then with L2, and then compared logistic regression with L1 (LR+L1) against that with L2 (LR+L2) by varying the sizes of the feature sets for each of the two rankings. We found no significant difference in accuracy between the two methods once the feature set is selected. We chose a synthesis, i.e., only those features that were present in both the sets obtained from L1 and that from L2, and experimented with it on more complex models like Decision Tree and Random Forest and observed that the accuracy was very close in spite of the small size of the feature set. Additionally, we also report on the standard metrics: accuracy, precision, recall, and f1-score.**

***Index Terms*—logistic regression, L1 regularization, L2 regularization, feature selection**

I. Introduction

Feature selection is an effective and efficient data preprocessing method in machine learning (ML) employed to reduce the dimensionality of the data, which can improve the performance of machine learning models and make them more interpretable. Moreover, it has the potential of providing more accurate predictions by removing noise in the data arising from irrelevant features. The challenge is to select a subset of the interesting data features that are most relevant and able to correctly differentiate samples from different classes.

A major problem in supervised ML is overfitting, i.e., the situation where the model learns the dataset "too well", resulting in a high training data accuracy but a poor test data accuracy during cross-validation. Regularization is one of the most popular methods used to avoid overfitting; it adds a penalty term to the loss function of the ML model that reflects the complexity of the model. L1 regularization tends to assign the coefficients or weights of less important features to zero, thus producing a sparse vector and equivalently a smaller set of good-enough features. On the other hand, L2 regularization tends to shrink the coefficients more uniformly; thus, a sparse vector is not realizable.

In our proposed method, we attempt to synthesize the two approaches. First, we obtain a ranked ordering of features using logistic regression with L1 regularization, then another ordering using L2. We choose those features that are present in both of these sets and experimentally test the performance of selected ML models using our set.

Our experiment is on one large dataset which is a non-trivial real-world dataset that is large in volume. Also, it is challenging because there is a problematic class that is hard to separate from another.

Regarding supervised ML models, we targeted the Decision Tree because it is explainable. Since we are studying accuracy, a metric that is not always the highest for these classifiers, we also chose the Random Forest classifier, which is related to the Decision Tree, exhibits excellent accuracy, but sacrifices explainability. We have found that using our method on Decision Trees and Random Forest, there is a loss of only 0.8 and 0.6% mean accuracy while reducing the feature size by 72%, and that regardless of the inclusion of the problematic class.

The rest of the paper is organised as follows. In section II, we present some background material and prior feature selection techniques. In section III, we describe the experiment setup and procedure; and in section IV present the results and analysis of this case study. Finally, in section V, we offer concluding remarks along with future research directions.

II. Background & Related Work

A. Background

1) Logistic regression and Regularization: Logistic regression (LR) is one of the most popular supervised ML algorithms used for solving the classification problems. Regression analysis models uses coefficients to estimate the features. If the estimates can be narrowed towards zero, then the impact of insignificant features might be reduced. L1 and/or L2 regularization is used with LR to prevent overfitting by adding a penalty term to the cost function. In the context of feature selection, L1 regularization is known to perform feature selection by shrinking the coefficients of less important features to zero which will make some features obsolete. This results in a sparse model where only a subset of features are used. On the other hand, L2 regularization shrinks the coefficients of less important features but does not set them to zero. This results in a model where all features are used, but the less important features have smaller coefficients.

- *L1 Regularization* adds a penalty term to the loss function equal to the sum of the absolute values of the coefficients. The cost function with L1 penalty is given by

$$\sum_{i=1}^{n}(y_i - \sum_{j=1}^{m} x_{ij}.W_j)^2 + \lambda \sum_{j=1}^{m} |W_j|$$

- *L2 Regularization* adds a penalty term to the loss function equal to the sum of the squares of the coefficients. Thus cost function with L2 penalty becomes

$$\sum_{i=1}^{n}(y_i - \sum_{j=1}^{m} x_{ij}.W_j)^2 + \lambda \sum_{j=1}^{m} {W_j}^2$$

Here, W_j is the weight/coefficient of the feature x_j and λ is known as the *regularization parameter*. The terms $\lambda \sum_{j=1}^{m} |W_j|$ and $\lambda \sum_{j=1}^{m} {W_j}^2$ are called *penalty terms* for L1 and L2 respectively.

2) Random Forest: Random Forest (RF) [2] is a *divide-and-conquer* based supervised learning algorithm that can scale the volume while maintaining statistical efficiency of information. It can be applied comprehensively to any prediction problem with few calibrated parameters. RF is a collection of tree predictors that allows the ensemble of trees to choose the most popular class in order to improve classification accuracy. The ensemble generates random vectors to regulate tree growth. Each tree is constructed using random observations from the original dataset.

3) Decision Tree: A Decision Tree (DT) is a tree-based technique which learns a model by generating the same labeling for the provided data. Similar to RF, DT also uses a *divide-and-conquer* strategy to find the best split points within a tree by employing a greedy search. The main idea is to continually split the dataset into yes/no questions using its features, until each data point is recognized as belonging to a particular class.

4) Performance evaluation parameters: The most common way to analyze the performance of any classification model is to use the confusion matrix [3]. Confusion matrix consists of actual label vs predicted labels which helps to visualize the distribution of each class along with a breakdown of error categories. Performance evaluation parameters i.e., accuracy, precision, recall, F1-score, etc., can be computed using the components of confusion matrix.

a) Accuracy: It is the ratio of the number of accurately predicted instances to the number of total instances. Despite being the most common performance measure, it is not an useful metric for unevenly distributed data.

b) Precision: The precision is an indicator measuring the exactness of each class. It is the ratio of the number of positive instances that were predicted accurately to the total number of instances that were predicted positive. When a high cost is associated with false positive, precision is a preferable metric.

c) Recall: The recall is an indicator measuring the completeness of each class. It is the ratio of the number of positive instances that were predicted accurately to the number of actually positive instances. Recall is a preferable metric when the associated cost of false negative is high.

d) F1-score: F1-score is a function of precision and recall which represents the harmonic mean. F1-score is more appropriate measure for uneven classification problems where a balance between precision and recall is needed.

B. Related Work

Features selection techniques are crucial for reducing training time and improving performance by removing irrelevant features. [4]. There have been proposed a lots of feature selection mechanisms in the past for supervised learning. The currently available feature selection methods can be categorized into the followings:

- *Filter Methods*: Features are selected based on the results of various statistical methods i.e., information gain, chi-square test, ANOVA, etc., are used to evaluate their association with actual results. These methods are faster and computationally less expensive, hence when dealing with high-dimensional data, it is suggested to use filter methods [5].

- *Wrapper Methods*: Wrapper methods generates subsets of all the features, and selects the best-performing subsets of features. However, these methods are usually computationally very expensive. Followings are the most popular wrapper methods:
  - Forward Selection: An iterative technique that starts with no features and in each iteration adds a feature that best improves the model until no additional feature is available to enhance performance [6].
  - Backward Elimination: Backward elimination begins with all features and, with each iteration, eliminates the least relevant one. This procedure is continued until there is no improvement after the elimination of features [6].
  - Recursive Feature elimination (RFE): A small sample classifier that increases performance by deleting characteristics with the least impact on training errors [7, 8]. In RFE, features are ranked by coefficient value or feature importance.
- *Embedded Methods*: Embedded methods generally combine two different approaches [9] thus encompass the benefits of both filter and wrapper methods. It intelligently drop unnecessary features during training phase. Both L1 and L2 regularization is embedded methods. Some other common algorithms include:
  - Tree-based feature selection: Tree-based estimators compute impurity-based feature importance, the best performing features are the closest nodes of the root of the tree.

III. Experiment

A. Dataset and pre-processing

1) Dataset: The CIC-IDS2018 dataset represents observations gathered over a span of 10 days of network traffic; it is huge in volume representing 16,233,002 samples and is spread over ten CSV files [10]. For each sample, it gives values for 79 features along with one of 15 different target classes of which it is known to be a member. Table I lists their names and sample sizes.

Among the 15 classes, the number of samples for *DDOS Attack LOIC UDP* is small: 1730, while those for *Brute Force XSS, Brute Force Web*, and *SQL Injection* are extremely small: 230, 611, and 87 respectively. Most real-world datasets are class imbalanced, i.e., the number of samples of some classes are grossly different from that of others. This is a key issue since most ML algorithms assume that data is evenly distributed across classes. The domination of majority classes over minority ones can make ML classifiers biased towards the former resulting in misclassification of the former.

2) pre-processing: As a part of a cleaning phase, observations with feature values *Infnity*, *NaN* (i.e., a missing value) were dropped. Also, there were 59 entries for which the class entry was *Label*, i.e., unknown; these too were dropped. Finally, we chose not to use the timestamp feature since it is not relevant to the classification [11]; this left us with 78 features.

We attempted to extract 5,000 random samples from each class that survived the cleaning phase. To avoid drastic class imbalance, we excluded the three classes with very small sample size: *Brute Force XSS, Brute Force Web*, and *SQL Injection*, leaving us with 12 classes. We had 5,000 samples each from eleven of them but only 1,730 from the last (*DDOS Attack LOIC UDP*); thus, we had 56,730 samples.

Moreover, *DoS attacks-SlowHTTPTest* and *FTP-BruteForce* classes are very hard to separate. We identified *DoS attacks-SlowHTTPTest* as a *problematic* class. To handle it, we split our experiment into two parts, excluding the problematic class in the first part of the experiment and including it in the second. Thus, for the first part, the size of the dataset became 51,730 from 11 classes.

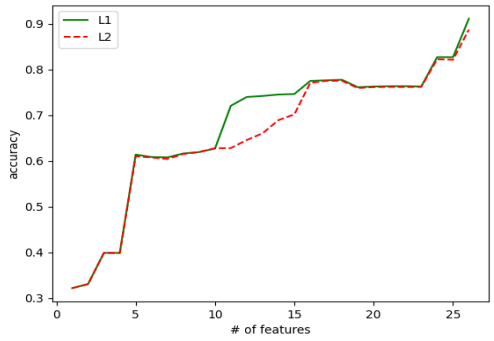
target class	record count
Benign	13484708
DDOS attack-HOIC	686012
DDoS attacks-LOIC-HTTP	576191
DoS attacks-Hulk	461912
Bot	286191
FTP-BruteForce	193360
SSH-Bruteforce	187589
Infilteration	161934
DoS attacks-SlowHTTPTest	139890
DoS attacks-GoldenEye	41508
DoS attacks-Slowloris	10990
DDOS Attack LOIC UDP	1730
Brute Force Web	611
Brute Force-XSS	230
SQL Injection	87

TABLE I: target classes

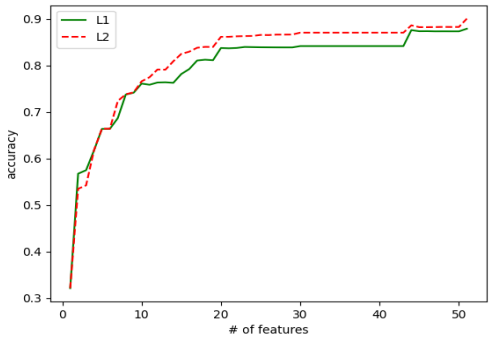
B. Experimental setup

Our experiments were implemented in Python 3 using the scikit learn (sklearn) package. This allowed us to execute standard code for training the machine learning models of interest, as well as for testing and obtaining results; we used matplotlib for plotting graphs.

We used a 70 : 30 ratio for training data : test data; consequently, in the first part of the experiment, the training dataset had 36,211 observations and testing set had 15,519 observations. For the second part of the experiment, we added 5000 random samples of the problematic class *DoS attacks-SlowHTTPTest*. The 70:30 ratio for training data : test data remained unchanged leading to 39,711 training samples and 17,019 test ones.



(a) Mean accuracy vs top features from L1-rank ordering.



(b) Mean accuracy vs top features from L2-rank ordering.

Fig. 1: Accuracy vs number of top features from L1 and L2 orderings.

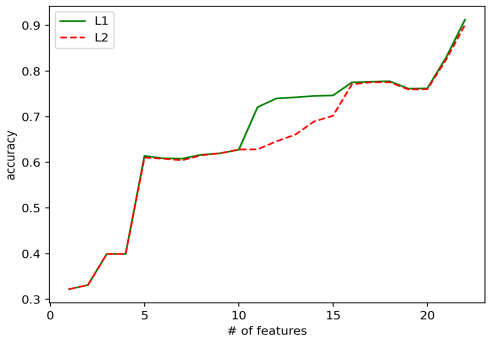


Fig. 2: Accuracy vs number of top common features ordered as per Table III.

The various class sizes are shown in Table II; the numbers in the first 11 rows correspond to the first part of the experiment; the last row was obtained from the second part (the changed values for the other 11 classes in that part are not shown).

target class	notation	sample size	training set	test set
Benign	cls-1	5000	3531	1469
Bot	cls-2	5000	3450	1550
DDOS attack-HOIC	cls-3	5000	3518	1482
DDOS attack-LOIC-UDP	cls-4	1730	1212	518
DDoS attacks-LOIC-HTTP	cls-5	5000	3476	1524
DoS attacks-GoldenEye	cls-6	5000	3442	1558
DoS attacks-Hulk	cls-7	5000	3545	1455
DoS attacks-Slowloris	cls-8	5000	3491	1509
FTP-BruteForce	cls-9	5000	3570	1430
Infilteration	cls-10	5000	3501	1499
SSH-Bruteforce	cls-11	5000	3475	1525
DoS attacks-SlowHTTPTest	cls-prb	5000	3518	1482

TABLE II: Sample dataset description

As noted earlier, we are interested in Logistic Regression. We used the *SAGA* solver (a version of SAG, which is stochastic average gradient) with an inverse regularization parameter equal to 0.5.

Next, we trained it on the dataset prepared for the first part of the experiment with all the features; this generated coefficient values of each feature and mean accuracy of the model.

L1 rank	L2 rank	feature name	L1 rank	L2 rank	feature name
1	1	Fwd Seg Size Min	12	15	Init Bwd Win Byts
2	3	URG Flag Cnt	13	16	Fwd IAT Max
3	5	SYN Flag Cnt	14	17	Bwd IAT Std
4	6	Fwd PSH Flags	15	18	Flow IAT Max
5	10	Bwd IAT Mean	16	12	Flow Byts/s
6	9	Bwd IAT Min	17	27	Flow IAT Mean
7	4	Fwd Pkt Len Min	18	45	Protocol
8	21	Fwd IAT Min	19	2	Pkt Len Std
9	23	Flow IAT Std	22	13	FIN Flag Cnt
10	24	Fwd IAT Mean	25	51	RST Flag Cnt
11	22	Bwd IAT Max	26	44	PSH Flag Cnt

TABLE III: common features

As outlined in the Introduction, we are interested in the coefficients learned from logistic regression using L1 and L2 regularization. Towards that end, we first applied logistic regression with L1 regularization (LR+L1) and sorted the learned coefficients to get an L1-rank ordering of features. Similarly, we trained an LR+L2 model and obtained an L2-rank ordering of features. Next, we sorted both sets of features in decreasing order of coefficient value.

1) Three experiments: Accuracy vs number of features:

(a) We tested the learned LR+L1 logistic regression model repeatedly with an increasing number of features form the L1-rank ordering: the top feature, the top two, the top three, etc., and observed the improvement in accuracy with increase in the number of features. This gave us **MAX_L1**, the number of features at which the accuracy reached approximately 95% of the maximum accuracy (the one obtained using all features). We took this accuracy level as the baseline value for the next component.

To find out how different the LR+L2 model would

be, we tested it for the same sequence of features, i.e., observed the accuracy of the learned LR+L2 model with an increasing number of top features from the same L1-rank ordering.

(b) Next, we repeated the above using the L2-rank ordering. Using the learned LR+L2 model, we first obtained **MAX_L2**, the number of features at which the accuracy equaled the baseline value obtained earlier and then observed the improvement of accuracy with increasing features, and found out how different the LR+L1 model would be for the same feature sets.

Finally, we attempted to synthesize the findings of LR+L1 and LR+L2 by computing the intersection of the L1-rank ordering and the L2-rank ordering obtaining a set of features we call *common features*.

(c) We then re-did the accuracy versus number of features experiment using these common features ordered as per Table III (taking the first eleven from the left column and the last eleven from the right).

2) Twelve experiments: We then conducted twelve experiments which we have given names listed in Table IV. Each name consists of a ML model (one of LR+L1, LR+L2, RF for Random Forest, or DT for Decision Tree) followed by a letter (one of *A, B,* or *C*). The letters denote the feature set used: *A* denotes the top MAX_L1 features from the L1-rank ordering; *B* the MAX_L2 features from the L2-rank ordering; and *C* all the common features.

For example, in *Experiment LR+L1-A*, we ran the LR+L1 model using features ranked 1 through MAX_L1 in the L1-rank ordering. We noted the mean accuracy, the corresponding confusion matrix, as well as precision, recall, and F1-score metrics.

As indicated in the last two columns, we experimented similarly with Random Forest (RF) and Decision Tree (DT) classifiers.

Features	ML Models			
	LR+L1	LR+L2	RF	DT
L1-rank ordering [1...MAX_L1]	*LR+L1-A*	*LR+L2-A*	*RF-A*	*DT-A*
L2-rank ordering [1...MAX_L2]	*LR+L1-B*	*LR+L2-B*	*RF-B*	*DT-B*
common features (all)	*LR+L1-C*	*LR+L2-C*	*RF-C*	*DT-C*

TABLE IV: Experiment names.

IV. Result and Analysis

The common features are listed in Table III along with their ranks in both L1-rank and L2-rank orderings.

The number of common features was 22 drawn from ranks 1 through 26 with average rank of 12 from the L1-rank and from ranks 1 through 51 with average rank of 18 from the L2-rank. This showed that the common features primarily follow the L1-rank ordering.

	cls-1	cls-2	cls-3	cls-4	cls-5	cls-6	cls-7	cls-8	cls-9	cls-10	cls-11
cls-1	948	141	42	0	1	15	9	0	6	112	195
cls-2	2	1547	0	0	0	0	0	0	0	1	0
cls-3	0	155	1327	0	0	0	0	0	0	0	0
cls-4	0	0	0	518	0	0	0	0	0	0	0
cls-5	0	0	178	0	1346	0	0	0	0	0	0
cls-6	0	0	0	0	0	1558	0	0	0	0	0
cls-7	0	0	0	0	0	0	1455	0	0	0	0
cls-8	19	0	0	0	0	1	0	1398	84	6	1
cls-9	0	0	0	0	0	0	0	0	1430	0	0
cls-10	159	98	75	40	28	3	2	5	0	1088	1
cls-11	0	0	0	0	0	0	0	0	0	0	1525

(a) LR+L1-A

	cls-1	cls-2	cls-3	cls-4	cls-5	cls-6	cls-7	cls-8	cls-9	cls-10	cls-11
cls-1	954	153	47	0	0	16	12	6	20	101	160
cls-2	1	1547	2	0	0	0	0	0	0	0	0
cls-3	0	741	661	0	80	0	0	0	0	0	0
cls-4	0	0	0	514	0	0	0	0	0	4	0
cls-5	0	0	0	0	1524	0	0	0	0	0	0
cls-6	1	0	0	0	0	1552	0	5	0	0	0
cls-7	0	0	0	0	0	0	1455	0	0	0	0
cls-8	17	0	0	0	0	15	13	1374	84	6	0
cls-9	0	0	0	0	0	0	0	0	1430	0	0
cls-10	146	130	89	0	23	1	0	8	0	1101	1
cls-11	0	0	0	0	0	0	0	0	0	0	1525

(b) LR+L1-B

	cls-1	cls-2	cls-3	cls-4	cls-5	cls-6	cls-7	cls-8	cls-9	cls-10	cls-11
cls-1	955	141	42	0	0	19	9	1	6	118	178
cls-2	3	1547	0	0	0	0	0	0	0	0	0
cls-3	0	156	1326	0	0	0	0	0	0	0	0
cls-4	0	0	0	518	0	0	0	0	0	0	0
cls-5	0	0	145	0	1379	0	0	0	0	0	0
cls-6	1	0	0	0	0	1557	0	0	0	0	0
cls-7	0	0	0	0	0	0	1455	0	0	0	0
cls-8	20	0	0	0	0	0	0	1398	84	6	1
cls-9	0	0	0	0	0	0	0	0	1430	0	0
cls-10	159	98	72	44	46	2	2	9	0	1066	1
cls-11	0	0	0	0	0	0	0	0	0	0	1525

(c) LR+L1-C

Fig. 3: Confusion matrix of LR+L1 model.

A. First part: excluding the problematic class

As mentioned earlier, the first part of the experiment excludes the problematic class.

1) Accuracy vs number of features: We described three experiments in Subsection III-B1. The results of the first, i.e., (a) are displayed in Fig 1a. We learned that MAX_L1 = 26, i.e., with the 26 top L1-ranked features, we obtained a 91.11% accuracy, which was approximately 95% of the maximum value of 95.69%

	cls-1	cls-2	cls-3	cls-4	cls-5	cls-6	cls-7	cls-8	cls-9	cls-10	cls-11
cls-1	947	144	38	0	1	16	9	1	6	112	195
cls-2	2	1547	0	0	0	0	0	0	0	1	0
cls-3	0	510	972	0	0	0	0	0	0	0	0
cls-4	0	0	0	518	0	0	0	0	0	0	0
cls-5	0	0	179	0	1345	0	0	0	0	0	0
cls-6	0	0	0	0	0	1558	0	0	0	0	0
cls-7	0	0	0	0	0	0	1455	0	0	0	0
cls-8	19	0	0	0	0	1	0	1398	84	6	1
cls-9	0	0	0	0	0	0	0	0	1430	0	0
cls-10	160	99	75	42	42	2	2	5	0	1071	1
cls-11	0	0	0	0	0	0	0	0	0	0	1525

(a) LR+L2-A

	cls-1	cls-2	cls-3	cls-4	cls-5	cls-6	cls-7	cls-8	cls-9	cls-10	cls-11
cls-1	1109	147	41	0	0	14	8	1	5	110	34
cls-2	3	1547	0	0	0	0	0	0	0	0	0
cls-3	0	741	661	0	80	0	0	0	0	0	0
cls-4	0	0	0	514	0	0	0	0	0	4	0
cls-5	0	0	0	0	1524	0	0	0	0	0	0
cls-6	1	0	0	0	0	1554	0	3	0	0	0
cls-7	0	0	0	0	0	0	1455	0	0	0	0
cls-8	17	0	0	0	0	0	0	1486	0	6	0
cls-9	0	0	0	0	0	0	0	0	1430	0	0
cls-10	160	91	72	0	1	3	0	5	0	1167	0
cls-11	0	0	0	0	0	0	0	0	0	0	1525

(b) LR+L2-B

	cls-1	cls-2	cls-3	cls-4	cls-5	cls-6	cls-7	cls-8	cls-9	cls-10	cls-11
cls-1	939	141	41	0	0	19	9	1	6	118	195
cls-2	3	1547	0	0	0	0	0	0	0	0	0
cls-3	0	321	1161	0	0	0	0	0	0	0	0
cls-4	0	0	0	518	0	0	0	0	0	0	0
cls-5	0	0	137	0	1387	0	0	0	0	0	0
cls-6	0	0	0	0	0	1558	0	0	0	0	0
cls-7	0	0	0	0	0	0	1455	0	0	0	0
cls-8	20	0	0	0	0	0	0	1398	84	6	1
cls-9	0	0	0	0	0	0	0	0	1430	0	0
cls-10	159	98	65	44	55	2	2	9	0	1064	1
cls-11	0	0	0	0	0	0	0	0	0	0	1525

(c) DT-C

Fig. 4: Confusion matrix of LR+L2 model.

	cls-1	cls-2	cls-3	cls-4	cls-5	cls-6	cls-7	cls-8	cls-9	cls-10	cls-11
cls-1	1342	4	9	0	0	2	0	0	3	109	0
cls-2	3	1546	0	0	0	0	0	0	0	1	0
cls-3	4	1	1477	0	0	0	0	0	0	0	0
cls-4	0	0	0	518	0	0	0	0	0	0	0
cls-5	0	0	0	0	1524	0	0	0	0	0	0
cls-6	0	0	0	0	0	1558	0	0	0	0	0
cls-7	0	0	0	0	0	0	1455	0	0	0	0
cls-8	1	0	0	0	0	0	0	1504	0	4	0
cls-9	0	0	0	0	0	0	0	0	1430	0	0
cls-10	70	2	3	0	0	0	0	4	0	1420	0
cls-11	0	0	0	0	0	0	0	0	0	0	1525

(a) RF-A

	cls-1	cls-2	cls-3	cls-4	cls-5	cls-6	cls-7	cls-8	cls-9	cls-10	cls-11
cls-1	1351	4	9	0	0	2	0	0	3	100	0
cls-2	4	1545	0	0	0	0	0	0	0	1	0
cls-3	4	1	1477	0	0	0	0	0	0	0	0
cls-4	0	0	0	518	0	0	0	0	0	0	0
cls-5	0	0	0	0	1524	0	0	0	0	0	0
cls-6	0	0	0	0	0	1558	0	0	0	0	0
cls-7	2	0	0	0	0	0	1453	0	0	0	0
cls-8	1	0	0	0	0	0	0	1506	0	2	0
cls-9	0	0	0	0	0	0	0	0	1430	0	0
cls-10	77	2	3	0	0	0	0	3	0	1414	0
cls-11	0	0	0	0	0	0	0	0	0	0	1525

(b) RF-B

	cls-1	cls-2	cls-3	cls-4	cls-5	cls-6	cls-7	cls-8	cls-9	cls-10	cls-11
cls-1	1345	4	9	0	0	2	0	0	3	106	0
cls-2	2	1546	1	0	0	0	0	0	0	1	0
cls-3	4	1	1477	0	0	0	0	0	0	0	0
cls-4	0	0	0	518	0	0	0	0	0	0	0
cls-5	0	0	0	0	1524	0	0	0	0	0	0
cls-6	0	0	0	0	0	1558	0	0	0	0	0
cls-7	0	0	0	0	0	0	1455	0	0	0	0
cls-8	2	0	0	0	0	0	0	1503	0	4	0
cls-9	0	0	0	0	0	0	0	0	1430	0	0
cls-10	67	2	3	0	0	0	0	4	0	1423	0
cls-11	0	0	0	0	0	0	0	0	0	0	1525

(c) RF-C

Fig. 5: Confusion matrix of RF model.

that we found possible with logistic regression with L1 regularization using all features.

Furthermore, we observed that the performances of LR with L1 and L2 are very similar (the divergence around 11 – 16 features can be attributed to the ordering of features being L1-ranked).

Similarly, the results of the second part, (outlined in Subsection III-B1(b)) are displayed in Fig. 1b. Here, we found MAX_L2 = 51, i.e., with the 51 top L2-ranked features, we obtained a 90.03% accuracy, on par with the LR+L1 observations.

As in part (a), we observed similar performances of LR with L1 and L2 (the divergence can be similarly attributed to the use of the L2-ranked ordering).

The results of the third part (outlined in Subsection III-B1(c)) are displayed in Fig 2. Here we observed that we 22 features, we obtained an accuracy of 91.22% with LR+L1 and 90.1% with LR+L2, both slightly higher than their corresponding counterparts with 26 and 51 features respectively). These values are summarized

	cls-1	cls-2	cls-3	cls-4	cls-5	cls-6	cls-7	cls-8	cls-9	cls-10	cls-11
cls-1	1338	7	10	0	0	2	0	1	2	109	0
cls-2	4	1544	1	0	0	0	0	0	0	1	0
cls-3	6	1	1475	0	0	0	0	0	0	0	0
cls-4	0	0	0	518	0	0	0	0	0	0	0
cls-5	0	0	0	0	1524	0	0	0	0	0	0
cls-6	0	0	0	0	0	1558	0	0	0	0	0
cls-7	2	0	0	0	0	0	1452	1	0	0	0
cls-8	1	0	0	0	0	0	0	1506	0	2	0
cls-9	4	0	0	0	0	0	0	0	1426	0	0
cls-10	106	3	3	0	0	0	0	3	0	1384	0
cls-11	0	0	0	0	0	0	0	0	0	0	1525

(a) DT-A

	cls-1	cls-2	cls-3	cls-4	cls-5	cls-6	cls-7	cls-8	cls-9	cls-10	cls-11
cls-1	1341	6	10	0	0	3	1	1	2	105	0
cls-2	4	1544	1	0	0	0	0	0	0	1	0
cls-3	5	1	1476	0	0	0	0	0	0	0	0
cls-4	0	0	0	518	0	0	0	0	0	0	0
cls-5	0	0	0	0	1524	0	0	0	0	0	0
cls-6	0	0	0	0	0	1558	0	0	0	0	0
cls-7	2	0	0	0	0	0	1453	0	0	0	0
cls-8	1	0	0	0	0	0	0	1506	0	2	0
cls-9	4	0	0	0	0	0	0	0	1426	0	0
cls-10	104	1	3	0	0	0	0	3	0	1388	0
cls-11	0	0	0	0	0	0	0	0	0	0	1525

(b) DT-B

	cls-1	cls-2	cls-3	cls-4	cls-5	cls-6	cls-7	cls-8	cls-9	cls-10	cls-11
cls-1	1334	7	9	0	0	2	0	0	2	115	0
cls-2	5	1544	1	0	0	0	0	0	0	0	0
cls-3	4	1	1477	0	0	0	0	0	0	0	0
cls-4	0	0	0	518	0	0	0	0	0	0	0
cls-5	0	0	0	0	1524	0	0	0	0	0	0
cls-6	0	0	0	0	0	1558	0	0	0	0	0
cls-7	2	0	0	0	0	0	1452	1	0	0	0
cls-8	1	0	0	0	0	0	0	1506	0	2	0
cls-9	4	0	0	0	0	0	0	0	1426	0	0
cls-10	112	2	4	0	0	0	0	3	0	1378	0
cls-11	0	0	0	0	0	0	0	0	0	0	1525

(c) DT-C

Fig. 6: Confusion matrix of DT model.

in Table V. It is apparent that using common features (which is only 28% of the original features), the drop in mean accuracy is not significant when compared with the values generated with all 78 features.

2) Twelve experiments: We described 12 experiments in Subsection III-B2. Their results for mean accuracy are listed in Table V under the columns 'L1 features', 'L2 features', and 'common features' corresponding to the suffix *A, B*, and *C* in the experiment names (see Table IV). In other words, 0.9111 is the accuracy obtained in *Experiment LR+L1-A*, etc.

The last two columns provide more observations per row: where the experiments using L1-rank features (-A) and L2-rank features (-B) are restricted to the top 22 of their features to compare with an equal number of features from the set of common features. They show the superiority of the set of common features does not come from their size.

The accuracy along with precision, recall, and F1-scores are summarized for all four ML models we have used in Table VI. It is visible that both RF and DT works very well with the selected features; all performance metrics have a score of 98% or higher. For completeness, we also report the corresponding confusion matrices in Fig. 3–6.

B. Second part: including the problematic class

For the second part, we performed 12 experiments described in Subsection III-B2 after including the problematic class in the dataset (56,730 samples). The comparison of mean accuracy using different ML models for this case is presented in Table VII. Comparing Table V and Table VII, it is clear that the mean accuracy dropped drastically for all the models. On top of that, none of the feature subsets performed well for either LR+L1 or LR+L2. However, for with RF and DT, the mean accuracy achieved using the 22 common features are very close to that achieved with all 78 features, showing the effectiveness of the set of common features.

The confusion matrix after inclusion is very similar for all the classes except for the problematic class *DoS attacks-SlowHTTPTest* (cls-prb) and *FTP-BruteForce* (cls-9); these are the two classes that are hard to separate. We will refer to the latter (cls-9) as the *confounding class*.

So, instead of showing the entire confusion matrix, we focus on the sub-matrix corresponding to these two classes. Table VIII shows the values of that sub-matrix. Both LR+L1 and LR+L2 using L1-ranked ordering or common features, identified all the samples of confounding class as the problematic class (see Table VIIIa, Table VIIIc, Table VIIId, Table VIIIf). The situation is better for both RF and DT regardless of the feature set: both models identified the problematic class correctly 97% of the time although they correctly identify the confounding class only 42% of the time. It is slightly better for L2-rank ordering features with both LR+L1 and LR+L2 since they identify the problematic class correctly 55% of the time and the confounding class 80%.

Since these appear to be tradeoffs, we present the values of recall for both the problematic class and the confounding class in Table IX and Table X.

	all 78 features	*A* (L1 features)	*B* (L2 features)	*C* (common features)	top 22 L1 features	top 22 L2 features
LR+L1-	0.9569	0.9111	0.8787	0.9122	0.7634	0.8375
LR+L2-	0.9689	0.887	0.9003	0.901	0.7615	0.8625
RF-	0.9917	0.9858	0.9826	0.986	0.9823	0.9826
DT-	0.9903	0.9827	0.9832	0.9822	0.9796	0.9785

TABLE V: Mean accuracy comparison excluding problematic class.

	accuracy	precision	recall	F1-score
LR+L1	0.91	0.91	0.92	0.92
LR+L2	0.90	0.91	0.91	0.90
RF	0.99	0.99	0.99	0.99
DT	0.98	0.98	0.98	0.98

TABLE VI: Performance metrics excluding problematic class using common features.

V. Conclusion & Future Work

In this research, we experimented on a logistic regression based feature selection method to reduce the number of features required to train a supervised ML model. We chose the CIC-IDS2018 dataset to analyze the performance of feature selection method on both linear and complex machine learning models. Using our proposed method synthesizing top features from LR+L1 and LR+L2, we obtained a small feature set of size 22, a 72% reduction from the original. (22 of 78) and observed that (a) it was the most important set of that size since it performed better than subsets of that size from either the L1-rank ordering or the L2-rank ordering for every ML model — both linear and complex; (b) we could reach or exceed the baseline accuracy target set (see Section III-B1) for LR with less features — 15% less for L1 and 57% less for L2. Note that it may appear that the accuracy for our feature set is 5 to 7% lower compared with full 78 feature set, but that is not a reasonable conclusion since we had set a lower baseline accuracy target. When the problematic class was included, the mean accuracy dropped owing to that class and one confounding class as we have shown with the confusion submatrix; (c) going beyond linear models to Random Forest and Decision Trees, our feature set showed an accuracy loss of less than 1% with a 72% reduction in feature size. This was valid even when the problematic class was included.

One limitation this scheme may encounter is a dataset in which the common set is the same size as the L1-ranked ordering in which case it would become equivalent to the latter. Also, our experiment was on a single dataset, which is not enough to prove the efficacy of this method in general.

In the future, we will do performance analysis on other datasets and provide a heuristic to predict the resulting number of features number.

References

[1] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *ICISSP*, 2018.

[2] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[3] M. P. Akhter, Z. Jiangbin, I. R. Naqvi, M. Abdelmajeed, A. Mehmood, and M. T. Sadiq. Document-level text classification using single-layer multisize filters convolutional neural network. *IEEE Access*, 8:42689–42707, 2020.

[4] A. Kumar and A. Jaiswal. Swarm intelligence based optimal feature selection for enhanced predictive sentiment accuracy on twitter. *Multimedia Tools and Applications*, 78, Feb. 2019.

[5] A. Bommert, X. Sun, B. Bischl, J. Rahnenführer, and M. Lang. Benchmark for filter methods for feature selection in high-dimensional classification data. *Computational Statistics & Data Analysis*, 143:106839, 2020.

[6] B. Walczak and D. Massart. Chapter 15 - calibration in wavelet domain. In B. Walczak, editor, *Wavelets in Chemistry*. Volume 22, Data Handling in Science and Technology, pages 323–349. Elsevier, 2000.

[7] X.-W. Chen and J. C. Jeong. Enhanced recursive feature elimination. In *Sixth International Conference on Machine Learning and Applications (ICMLA 2007)*, pages 429–435, 2007.

[8] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1):389–422, 2002.

[9] K. Yan and D. Zhang. Feature selection and analysis on correlated gas sensor data with recursive feature elimination. *Sensors and Actuators B: Chemical*, 212:353–363, 2015.

[10] CIC-IDS2018 dataset. https://www.unb.ca/cic/datasets/ids-2018.html.

[11] M. Catillo, M. Rak, and V. Umberto. *2l-zed-ids: a two-level anomaly detector for multiple attack classes*. In Mar. 2020, pages 687–696.

	all 78 features	*A* (L1 features)	*B* (L2 features)	*C* (common features)	top 22 L1 features	top 22 L2 features
LR+L1-	0.8927	0.7983	0.8307	0.8012	0.6965	0.8013
LR+L2-	0.9115	0.7979	0.8469	0.8019	0.6988	0.8113
RF-	0.9385	0.9333	0.9328	0.933	0.9297	0.9295
DT-	0.9381	0.9304	0.9307	0.9306	0.9277	0.9263

TABLE VII: Mean accuracy comparison including problematic class.

	cls-prb	cls-9
cls-prb	1482	0
cls-9	1490	0

(a) LR+L1-A

	cls-prb	cls-9
cls-prb	812	670
cls-9	316	1174

(b) LR+L1-B

	cls-prb	cls-9
cls-prb	1482	0
cls-9	1490	0

(c) LR+L1-C

	cls-prb	cls-9
cls-prb	1482	0
cls-9	1490	0

(d) LR+L2-A

	cls-prb	cls-9
cls-prb	812	670
cls-9	316	1174

(e) LR+L2-B

	cls-prb	cls-9
cls-prb	1482	0
cls-9	1490	0

(f) LR+L2-C

	cls-prb	cls-9
cls-prb	1433	49
cls-9	859	631

(g) RF-A

	cls-prb	cls-9
cls-prb	1433	49
cls-9	859	631

(h) RF-B

	cls-prb	cls-9
cls-prb	1435	47
cls-9	861	629

(i) RF-C

	cls-prb	cls-9
cls-prb	1435	47
cls-9	861	629

(j) DT-A

	cls-prb	cls-9
cls-prb	1435	47
cls-9	861	629

(k) DT-B

	cls-prb	cls-9
cls-prb	1435	47
cls-9	861	629

(l) DT-C

TABLE VIII: Confusion (sub-)matrix only for the problematic class and its confounding class.

	L1 features	L2 features	common features	top 22 L1 features	top 22 L2 features
LR+L1	1.00	0.55	1.00	1.00	0.54
LR+L2	1.00	0.55	1.00	1.00	0.55
RF	0.97	0.97	0.97	0.97	0.97
DT	0.97	0.97	0.97	0.97	0.97

TABLE IX: Comparison of recall values only for the problematic class *DoS attacks-SlowHTTPTest*.

	L1 features	L2 features	common features	top 22 L1 features	top 22 L2 features
LR+L1	0.00	0.79	0.00	0.00	0.79
LR+L2	0.00	0.79	0.00	0.00	0.79
RF	0.42	0.42	0.42	0.42	0.42
DT	0.42	0.42	0.42	0.42	0.42

TABLE X: Comparison of recall values only for the confounding class *FTP-BruteForce*.