Intrusion detection using logistic regression: A comparison of regularization methods and PCA-logistic regression method

Nowmin Naj Manisha
School of Computing and Information Technology
University of Wollongong
CSCI933, SN:8248862, nnm031@uowmail.edu.au

April 19, 2024

Abstract

In this report we study four regularization methods and PCA-logistic regression, and compare their performances under an intrusion detection task. Specifically, we cast the intrusion task as a binary classification problem and used logistic regression along with several regularizations to explore the performances. Using a a real-time IoT dataset, it was found that No Regularization selected an appropriate number of useful features and provided a good accuracy of 99.34% . The PCA-logistic regression method with 99% of features retained resulted in the best accuracy of 98.17%.

1 Introduction

Intrusion detection refers to the process of monitoring and analyzing network or system activities for sign of unauthorised or malicious behaviour. The main goal of the intrusion detection is to identify and respond to security incidents. Sharmila (2024) generated the RT-IoT2022 dataset using an IoT environment detecting anomalies using an auto-encoder framework. They proposed two optimized auto-encoder models that calculates reconstruction error and thresholds for detecting anomalies in IoT infrastructure. Our objective is to develop a series of binary classifier models capable of distinguishing between normal and attack traffic on dataset features. Logistic regression is a regression model suitable for binary classification. We explore the effectiveness of four regularization methods and Principle Component Analysis (PCA) to asses the best accuracy obtained with each of the model.

2 Theory and properties of regression

Regression analysis is a fundamental statistical method used to model the relationship between one or more independent variables and a dependent variable. Regression models are based on several assumptions, including linearity, independence of error, constant variance of errors and normal distributed errors. From the book of Geron (2019), we can conclude that the Root Mean Square(RMSE) is commonly used as a performance measure. It provides an indication of the typical prediction errors made by the system, giving more emphasis to large errors. Equation 1 shows the mathematical formula to compute the RMSE. In this equation, $h(x_i)$ represent predictions and y_i represent true value. m is the number of observations , x_i is the i-th input feature vector.

$$\text{RMSE}(X, h) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(h(x_i) - y_i)^2} \tag{1}$$

RMSE(X,h) is the cost function measured on the set of examples using prediction function which is also called hypothesis.

Even though RMSE generally preferred performance measure for regression task, Mean Absolute Error(MAE) is frequently utilized as a performance metric to evaluate the accuracy of predictions generated by a model. Equation 2 shows the mathematical formula to compute the MAE.

$$\text{MAE}(X, h) = \frac{1}{m}\sum_{i=1}^{m}|(h(x_i) - y_i)| \tag{2}$$

Here m is the number of observations, x_i is the i-th input feature vector, $h(x_i$ is the predicted value for x_i, and y_i is the true value corresponding to x_i.Geron (2019)

2.1 L1, L2 and Elastic-net penalties

2.1.1 L1

L1 regularization also known as Lasso regularization. The primary goal is to reduce sparsity in the solution weight vector. This means some of the coefficients corresponding to less important features are driven to exactly zero. This property makes L1 regularization particularly useful models with high-dimension data where feature selection is importantGeron (2019).
Mathematically, the L1 term added to cost function is represented as:

$$J(\theta) = \text{MSE}(\theta) + (\alpha \sum_{i=1}^{n} |\theta_i|) \tag{3}$$

Where-
α is the regularization parameter
θ_i are coefficients of the model.

2.1.2 L2

In Ridge Regression, which is also known as L2 Regression, we adjust the weights of the model by including an extra term in the cost function. This term encourages the learning algorithm to not only accurately predict the data bust also to keep the model weights relatively small. Importantly, this additional term should only be included in the cost function during training phase. Equation 5 presents the Ridge Regression cost function:

$$J(\theta) = \text{MSE}(\theta) + (\alpha \frac{1}{2} \sum_{i=1}^{n} \theta_i^2) \tag{4}$$

Here, MSE(θ) is the Mean Squared Error, α is the regularization parameter, θ_i are the coefficients, and n is the number of coefficients. The hyper parameter α determines the extent of regularization applied to the model. When α = 0, Ridge Regression behaved exactly like Linear Regression. However, as α increases, the weights of the model tend to converge towards zero, eventually resulting in a flat line that passes through the mean of the data.Geron (2019)

2.1.3 Elastic Net

Elastic Net serves as a comprise between Ridge Regression and Lasso Regression. It combines elements from both regularization techniques, allowing to control the balance between the two using the mix ratio r, when r = 0, Elastic Net behaved like Ridge Regression, while r = 1 makes it equivalent to Lasso Regression.
The cost function for Elastic Net is given by:

$$J(\theta) = \text{MSE}(\theta) + r(\alpha \sum_{i=1}^{n} |\theta_i| + \frac{(1-r)}{2} \alpha \sum_{i=1}^{n} \theta_i^2 \tag{5}$$

Here, MSE(θ) is the Mean Squared Error Term,*alpha* is the regularization parameter, r is the mix ratio, $|\theta_i|$ is the absolute value of the i-th coefficients, θ_i^2 is the squared value of the i-th coefficient, and n is the number of coefficients.Geron (2019)

2.2 Logistic regression

Logistic Regression also known as Logit Regression, stand out as a widely utilised method for gauging the likelihood of an instance belonging to a specific class. From the book of Geron (2019), we conclude that similar to Linear Regression, it compared a weighted sum of the input features, along with a bias term. Equation 6 is the Logistic Regression model estimated probability (vectorized form)

$$\hat{p} = h_\theta(x) = \sigma(x^T \theta) \tag{6}$$

However, unlike Linear Regression, Logistic Regression employs a sigmoid function, denoted as $sigma(.)$, to transform this result. It produces an output between 0 and 1, representing the estimated probability. Equation 7 is the Logistic function

$$\sigma(t) = \frac{1}{1 + e^{-t}} \tag{7}$$

The training and cost function of logistic regression aim to optimize the model parameters to best fit data and make accurate predictions. The cost function measures the error between the predicted probabilities and the actual class labels. The cost function for logistic regression, also known as the log loss or cross-entropy loss, is defined as:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log(h_\theta(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right] \tag{8}$$

The partial derivative of the cost function with respect to θ_j is given by:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} \tag{9}$$

This derivative is used in the gradient descent algorithm to update the parameters in each iteration of training.Geron (2019)

2.3 PCA-logistic regression

Principal Component Analysis (PCA) is a dimensionality reduction technique commonly used in data analysis and machine learning. From the book of Buduma (2017) we can get the fundamental idea behind PCA is to discover a set of axes that capture the most essential information about the dataset. Specifically, in the case of d-dimensional data, we aim to identify a new set of m ¡ d dimensions that retain as much valuable information from the original dataset as possible. Lets simplify this by considering d=2 and m=1. We start by finding a unit vector representing the direction of maximum variance in the dataset, which becomes our fist axis. Then from the orthogonal vectors to this first choice, we select another unit vector representing the direction of maximum variance , forming our second axis. This iterative process continues until we find a total of d new vectors representing the new axes. Mathematically, we can interpret this operation as projecting onto the vector space defined by the top m eigenvectors of the covariance matrix of the dataset.

3 Experiments

In this experiment, we are developing a series of binary classifier models that can detect normal and attack traffic based on the features in the dataset. We are using Logistic regression and developing logistic regression models using scikit-learn, considering- no regularization, ridge, lasso and elastic-net. We are performing PCA dimentionality reduction on the features and building logistic regression model using the reduced feature set. Finally, we evaluate each model's performance and report the best accuracy obtained.

3.1 Internet-of-things dataset

Sharmila (2023) generated a real-time internet-of-things(IOT) dataset (Sharmila (2024) for normal and attack traffic. This dataset is being used to train and test the model. This RT-IoT2022 dataset is derived from real-time IoT infrastructure, offers a comprehensive collection of data encompassing various IoT device and sophisticated network attack techniques. The data set does not have any missing value. The data set has 2 columns that have strings and other columns have numeric values in X data features. We are going to represent this 2 columns as categorical variable columns. In Y data target there is only one column that has categorical variables.

3.2 Experimental setup

To load the dataset into the code, we have to install ucimlrepo package in the environment. In the dataset X represents data features and Y represents data target. We are using jupyter notebook to develop the code. After loading the dataset we have to pre-process the data so that we can use this data set for different regression models. In this dataset there are no missing values.
OneHotEncoder class is used from Scikit-Learn to handle categorical variables in X data set. This categorical variables doesn't have natural ordering. It transforms categorical variables into a numerical format. It creates binary column for each category in the original variable. Categorical columns are then replaced by the binary column.
Label encoder class from Scikit-Learn is used in Y data set to transform categorical labels into numerical format. Label encoding assign a unique integer to each category.
Then we perform standard scaler to X data set to standardize features by removing the mean and scaling to unit variance.
We used train_test_split function from scikit-learn to split dataset into training and testing set for model evaluation. We split X in X_test, X_train and y in y_test, y_train. Here test_size is 0.2. It means that 20% of data will be use for testing and 80% of data is for training.

3.3 Experiment 1

We train no regularization, ridge, Lasso, elastic net model with X_train and y_train and predict X_test and compare predicted labels with y_test which has true label. Then we calculate the accuracy. In logistic regression model, there are fixed penalties for each model. We fixed parameter solver to 'saga', random state to '42' for every model.
In PCA the parameter 'n_components' specifies the number of principal components to retain after dimensionality reduction. We choose n_components=0.99, it means it will retain 99% of the original features. 'pca.fit_transform(X_train)' is used to train data and transform it into a lower dimensional space. 'pca.transform(X_test)' applies the transformation learned from training data to the testing data to achieve consistent dimentionality reduction. Then we perform logistic regression with fixed solver 'saga' and random state '42' to calculate the accuracy score.

We train these model by changing parameter max_iter to 50,100,200,500 and 1000 to converge the optimal solution for accuracy score.

3.4 Experiment 2

In experiment 2, we select the max_iter='1000' which gives the highest accuracy score in the experiment 1. To improve accuracy score, we experiment with different solvers for the logistic regression model. scikit-learn solver includes 'newton-cg', 'lbfgs', 'liiblinear', 'newton-cholesky', 'sag' and 'saga'. The choice of the algorithm depends on the penalty chosen. Supported penalties by solver:

- 'lbfgs' - ['l2', None]
- 'liblinear' - ['l1', 'l2']
- 'newton-cg' - ['l2', None]
- 'newton-cholesky' - ['l2',None]
- 'sag' - ['l2', None]
- 'saga' - ['elasticnet', 'l1', 'l2', None]

We can conclude that not every penalty supports every solver. Only solver 'saga' is supported by all the penalties.

3.5 Results

From the Experiment 1, after experimenting different iterations in different model we can conclude that iteration 1000 gives the highest accuracy score for solver saga. The

Table 1: Accuracy of different models in different Iterations

	Models				
Iter	None	(L2)	(L1)	L1-L2	PCA
50	0.9785	0.9785	0.9786	0.9786	0.9800
100	0.9768	0.9785	0.9768	0.9768	0.9810
200	0.9791	0.9791	0.9791	0.9791	0.9798
500	0.9801	0.9801	0.9801	0.9801	0.9810
700	0.9801	0.9801	0.9801	0.9801	0.9814
1000	0.9802	0.9802	0.9802	0.9802	0.9817

Table 1 and the Figure 1 display the accuracy scores for different regularization methods (no regularization, ridge, Lasso, elastic net, PCA) across various iteration counts. We can observe the convergence behaviour of the accuracy score in no regularization, ridge, Lasso, elastic net after iteration 100. But when the iteration is 100, L2 performs better than other models. We get the best accuracy from

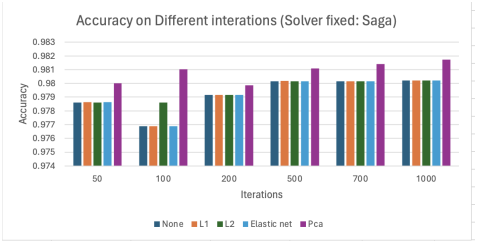

Figure 1: Accuracy on different iteration

all the models in iteration 1000.
One the other hand, when we select n_components=0.99 in PCA it gives the highest accuracy than all other model. PCA gives 98.17% after retaining 99% of original data.
In Experiment 2, we choose iteration 1000 to experiment with different solver to find out which solver gives the best accuracy.

Table 2: Accuracy of different models in different Solver

	Methods				
Solver	None	(L2)	(L1)	L1-L2	PCA
Saga	0.9802	0.9802	0.9802	0.9802	0.9817
Sag	0.9809	00.9809	NA	NA	NA
Newton-chol	0.9922	0.9894	NA	NA	NA
Newton-cg	0.9934	0.9915	NA	NA	NA
liblniear	NA	0.9870	0.9916	NA	NA
lbfgs	0.9929	0.9917	NA	NA	NA

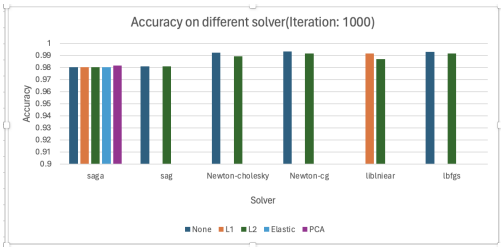

Figure 2: Accuracy on different Solver

From the Table 2 and Figure 2, we can conclude that we can not use all solver for all models. We have to use solver 'saga' for elastic net and PCA as we can not use other solver for them. Solver 'Newton-cg' gives the highest accuracy for No Regularization and L2. Among them, No Regularization gives the highest result which is 99.34% after using Newton-cg as solver and iteration 1000.

4 Discussion

The result obtained from the experiments offer valuable insights into the performance of logistic regression models with different regularization methods and solvers, as well

as the impact of PCA on dimension reduction.

The experiments highlighted the importance of selecting appropriate hyper-parameters, such as the number of iterations and the choice of solver, in logistic regression models. The exceptional results observed with iteration 1000 underscored the importance of ensuring thorough convergence to attain the highest accuracy possible. Moreover, the versatility of the Saga solver across different regularization methods emphasised its suitability for various model configuration.

5 Conclusion

The experiment demonstrates the impact of different iterations and solver on the accuracy of logistic regression with various regularization techniques and PCA dimensionality reduction.

From Experiment 1, it was evident that iteration 1000 yielded the highest accuracy scores for Saga solver. The table and graph presented illustrate the convergence behaviour of accuracy scores across different regularization methods and iterations. Notably, L2 regularization outperformed other models in solver Saga and at iteration 100, while the highest accuracy was achieved by all models at iteration 1000. Additionally, PCA with n-components = 0.99 showcased superior accuracy, achieving 98.17%.

Experiment 2 focused on evaluating different solvers for logistic regression models. The results indicated that not all solvers are compatible with all models, with the solver 'saga' being the most versatile across different regularization model, solver 'Newton-cg' demonstrated the highest accuracy for No Regularization and L2, with No Regularization achieving the highest accuracy of 99.34% at iteration 1000.

In summary, the meticulous adjustment of parameters such as iterations and solvers plays an important role in shaping the effectiveness of logistic regression model. These results emphasize the necessity for throughout exploration and fine-tuning to attain peak accuracy levels in classification.

References

Buduma, N. (2017). *Fundamentals of deep learning: Designing next-generation machine intelligence algorithms.*

Geron, A. (2019). *Hands-on machine learning with scikit-learn, keras and tensorflow: Concepts, tools, and techniques to build intelligent systems (2nd ed.). ca, usa: O'reilly media, incw.*

Sharmila, . N. R., B. (2023). *Quantized autoencoder (qae) intrusion detection system for anomaly detection in resource-constrained iot devices using rt-iot2022 dataset. cyber- security, 6(41(2023)), 1-15.*

Sharmila, . N. R., B. (2024). Retrieved from `RT-IoT2022.UCIMachineLearningRepository .(DOI:https://doi.org/10.24432/C5P338)`