Impact of Noise on Machine Learning Algorithms Performance

How noise in datasets affect the performance of ML algorithms

Neha Tiwari
MSc Computer Science - Data Science
Trinity College Dublin
Dublin, Ireland
tiwarin@tcd.ie

Krishna Hariramani
MSc Computer Science - Data Science
Trinity College Dublin
Dublin, Ireland
hariramk@tcd.ie

Ankur Rangwala
MSc Computer Science - Data Science
Trinity College Dublin
Dublin, Ireland
rangwala@tcd.ie

INTRODUCTION

We never get perfect data in real-world, hence our data models and decisions based on corrupted (noisy) data tends to be inaccurate. Presence of noise can make a negative impact on system performance by increasing the complexity and training time and decreasing the efficiency of the resulting model [1]. Majority of machine learning (ML) algorithms have already incorporated different tactics to handle noise, despite that, noise still can affect algorithms performance adversely. Hence, knowing the exact impact of noise in our dataset is an essential issue and should be evaluated to make an effective model to take a better decision accordingly.

RELATED WORK

Noise in dataset affects the performance of ML algorithms and thus motivated the study related to different ways of creating and introducing noise in the dataset to measure its impact on different ML algorithms [2].

1. Noise generation:

There are different ways to characterize noise generation. Noise can be characterized by either its distribution (Normal (Gaussian)) or where the noise is introduced (individual or combination of training and test data or input attributes or output class) or differentiating whether the generated values are related to each data value or min, max and standard deviation of each feature.

2. Impact of noise in algorithms:

In [3], Zhu investigated different types of noise in training and test data, input attributes, output class and identified a comprehensive analysis of noise using the C4.5 algorithm. Also, performed few comparative testing using various techniques (Prism, 1R, C4.5 and HCV).

METHODOLOGY

Figure 1 shows the project workflow at a macro level.

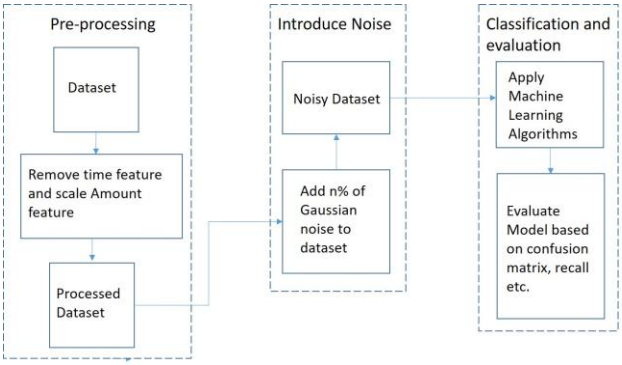

Figure 1: Project Workflow Diagram

1. Dataset and Pre-Processing

We have used credit card fraud dataset from Kaggle [4] with 284807 records of transactions (made in September 2013 over two days) and 28 normalized numerical features which were obtained by using PCA on a set of original features, due to confidentiality issues the names of the features are not revealed by the author, other than these 28 features, two additional features 'Time' and 'Amount' is also provided. We dropped the Time feature as it follows an increasing trend (from 0 to 172792) and does not give any information for labels, we normalized and scaled the 'Amount' feature as it can give information like the amount of fraud and the value which can be saved by employing the fraud detection models. Finally, we get 29 scaled and normalized numerical features and one label column 'Class' containing 0's and 1's (1 denoting fraud transaction).

2. Introduce Noise to the dataset

We added noise to dataset after pre-processing. For adding n% of noise to the dataset, we use the algorithm shown in Figure 2[5]. Figure 3 shows an example for the same.

```
Find the standard deviation SD of each feature f
Pick n% of random rows from the original dataset and store in a new variable ND
FOR each instance in ND,
        FOR each feature value of feature f,
                1. Choose a random number x taken from the interval (-SD, SD)
                2. Add x to that instance
CONCATINATE ND at the end of original dataset
SHUFFLE all dataset and Randomly SELECT number of rows in the original dataset
```

Figure 2: Algorithm for adding noise

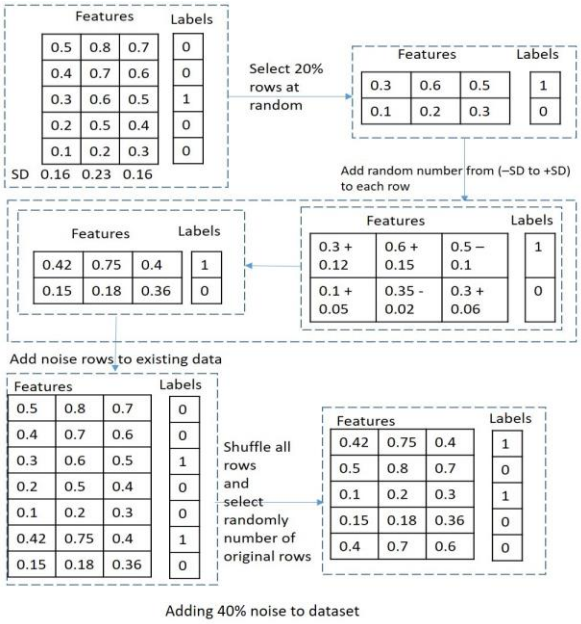

Figure 3: Example of adding 40% noise to a dataset

3. Classification and Evaluation

After adding noise to the dataset, it was split into train set and test set with a 70:30 split. We use Logistic Regression, NB Classifier and RF Classifier with default hyper-parameters at the different noise level. We will include hyper-parameter tuning in future work.

Since our dataset is highly unbalanced, the negative class (frauds) account for only 0.172% of all transactions, evaluation metrics like accuracy and AUC under ROC should not be considered for evaluating the models. As we are interested in correctly predicting the frauds, we have used confusion matrix and recall for negative class (TN/TN+ FN) as a metric to evaluate our model's performance. We are also showing specificity (TN/TN+FP) at different noise levels.

RESULTS & DISCUSSIONS

Results of our experiments on different algorithms are discussed below.

1. Logistic Regression

Figure 4 demonstrates the confusion matrix at different noise levels, which shows as noise increases, the ratio of True-Negative to False-Negative decreases.

```
For noise level % 0.00:
                    Predicted Positive        Predicted Negative
Actually Positive     85294                     15
Actually Negative        46                     88

             Positive       Negative
Num Cases:  85309.000000          134.00
precision:      1.00             0.85
Recall:         1.00             0.66
```

```
For noise level % 0.30:
                    Predicted Positive        Predicted Negative
Actually Positive     85283                     10
Actually Negative        65                     85

             Positive       Negative
Num Cases:  85293.000000          150.00
precision:      1.00             0.89
Recall:         1.00             0.57
```

```
For noise level % 0.90:
                    Predicted Positive        Predicted Negative
Actually Positive     85293                     9
Actually Negative        72                     69

             Positive       Negative
Num Cases:  85302.000000          141.00
precision:      1.00             0.88
Recall:         1.00             0.49
```

Figure 4: Confusion Matrix & Performance Evaluation at noise levels - 0%, 30%, and 90%.

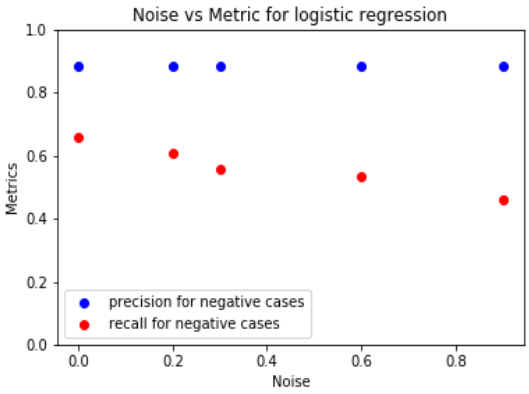

Figure 5: Noise Vs Metrics for Logistic Regression

Fig 4 and 5 depicts that value of recall for negative class follows a decreasing trend (66% to 49%) with an increase in noise level (0% to 90%). Also, we can see that specificity remains the same due to class imbalance.

2. Naive Bayes (NB) classifier

Figure 6 and 7 shows the confusion matrix and performance metrics at different noise levels which shows as noise increases, the ratio of true-negative to false-negative do not follow any trend signifying that classification for fraud is more robust to noise but it comes at the cost of specificity since this model classifies many regular transactions as fraud, i.e., number of false-negatives are more for NB when compared to other two models.

```
For noise level % 0.60:
                    Predicted Positive        Predicted Negative
Actually Positive     83804                     1475
Actually Negative        27                      137

            Positive       Negative
Num Cases:  85279.000000          164.00
precision:     1.00             0.08
Recall:        0.98             0.84
```

```
For noise level % 0.00:
                    Predicted Positive        Predicted Negative
Actually Positive     83444                     1865
Actually Negative        24                      110

            Positive       Negative
Num Cases:  85309.000000          134.00
precision:     1.00             0.06
Recall:        0.98             0.82
```

```
For noise level % 0.90:
                    Predicted Positive        Predicted Negative
Actually Positive     83990                     1300
Actually Negative        33                      120

            Positive       Negative
Num Cases:  85290.000000          153.00
precision:     1.00             0.08
Recall:        0.98             0.78
```

Figure 6: Confusion Matrix & Performance Evaluation at noise levels - 0%, 60%, and 90%

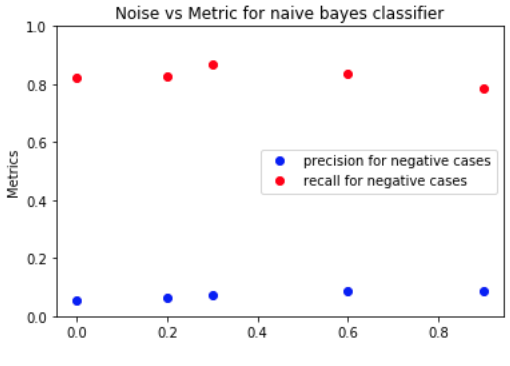

Figure 7: Noise Vs Metrics for Naïve Bayes

3. Random forest (RF) classifier

Figure 8 demonstrates the confusion matrix at different noise levels, which shows as noise level increases, the ratio of true-negative to false-positive decreases. Figure 9 illustrates that as noise increases, recall decreases from 75% to 64% whereas specificity at all noise levels is good and does not follow any trend.

From above experiments, when classifying frauds, we conclude that:

1. Logistic Regression is the least robust to noise.
2. NB classifier is most robust to noise but provides low specificity.
3. RF classifier is more robust to noise than Logistic Regression but less robust than NB classifier. Also, it maintains high specificity throughout different noise levels.

```
For noise level % 0.60:
                    Predicted Positive        Predicted Negative
Actually Positive     85286                        3
Actually Negative        40                      114

            Positive       Negative
Num Cases:  85289.000000          154.00
precision:     1.00             0.97
Recall:        1.00             0.74
```

```
For noise level % 0.90:
                    Predicted Positive        Predicted Negative
Actually Positive     85297                       11
Actually Negative        48                       87

            Positive       Negative
Num Cases:  85308.000000          135.00
precision:     1.00             0.89
Recall:        1.00             0.64
```

```
For noise level % 0.00:
                    Predicted Positive        Predicted Negative
Actually Positive     85303                        6
Actually Negative        33                      101

            Positive       Negative
Num Cases:  85309.000000          134.00
precision:     1.00             0.94
Recall:        1.00             0.75
```

Figure 8: Confusion Matrix & Performance Evaluation at noise levels - 0%, 60%, and 90%

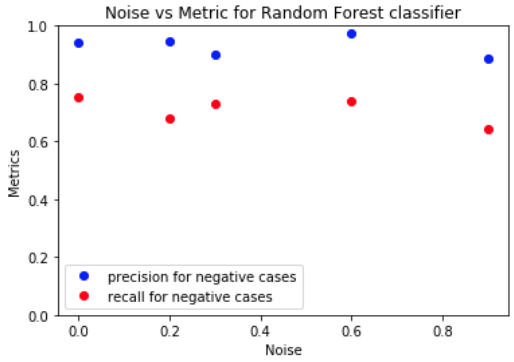

Figure 9: Noise Vs Metrics for RF

LIMITATIONS & OUTLOOKS

Due to time constraints, currently, we have evaluated the impact of noise on only three ML algorithms (NB classifier, Logistic Regression and RF classifier). Also, we calculated the noise impact based on default hyper-parameters for each algorithm. However, we would try to investigate the impact after performing hyper-parameter tuning on the existing algorithms, in addition to experimenting with other algorithms like decision tree and k-nearest neighbors (KNN) and investigating other forms of noise as part of future work.

REFERENCES

[1] Kalapanidas, Elias, Nikolaos Avouris, Marian Craciun, and Daniel Neagu. "Machine learning algorithms: a study on noise sensitivity." In *Proc. 1st Balcan Conference in Informatics*, pp. 356-365. 2003.

[2] Nettleton, David F., Albert Orriols-Puig, and Albert Fornells. "A study of the effect of different types of noise on the precision of supervised learning techniques." *Artificial intelligence review* 33, no. 4 (2010): 275-306.

[3] Zhu, Xingquan, and Xindong Wu. "Class noise vs. attribute noise: A quantitative study." *Artificial intelligence review* 22, no. 3 (2004): 177-210.

[4] Bozsolik, Timo, Credit Card Fraud Detection, Retrieved from https://www.kaggle.com/mlg-ulb/creditcardfraud
[5] Lee, T., 2015. How to add some noise data to my classification datasets. Retrieved from https://www.researchgate.net/post/How_to_add_some_noise_data_to_my_classification_datasets.