Advanced Machine Learning Assignment 2 / Final practical part report

Valeriy Novossyolov
Innopolis University
Innopolis, Russia
v.novossyolov@innopolis.university

Github repository:
https://github.com/vallre/AML_Final_practical

1 Introduction

The aim of this paper is to report on the findings of a Advanced Machine Learning assignment. The main idea of this assignment is to get practical experience of working with autoencoder and GAN models for the tasks of data regeneration and data balancing.

In the first part of the assignment, the aim was develop three different types of autoencoder models, namely Undercomplete Autoencoder, Regularized Autoencoder, and Variational Autoencoder. The Undercomplete Autoencoder is an autoencoder that tries to find a lower dimension representation of a higher dimensional input data, such that this representation could be decoded back into the original form with minimal loss of information. The Regularized Autoencoder is an autoencoder, which could be both undercomplete, or overcomplete, with a modified loss function. The main modification is the addition of regularization parameter, which should keep model from overfitting. Meaning that this type of autoencoder also prioritizes the problem of generalization to unseen examples. The Variational Autoencoder is one of the generational machine model. In comparison to a regular autoencoder model, the variational counterpart learns the distribution of the data. Thus, the decoding function is sampling from the learned distribution. All three machine learning models were developed for the task of filling in the missing values in domain of fraud transaction detection.

In second part of the assignment, the aim was to create a Generative Adversarial Network (GAN) for the purpose of the balancing of imbalanced datasets. The problem of imbalanced datasets leads to the low precision of rare classes. In theory GAN can be used to generate new unseen examples of the given classes to solve the imbalance problem.

2 Methodology

2.1 Task 1

For the first part of the assignment the data comes from the Vesta's real-world e-commerce transactions dataset. The dataset contains two classes: normal transactions (96.5%) and fraudulent transactions (3.5%). The data contains a significant number of missing values, therefore, it is difficult to use this kind of data in the raw format with machine learning approaches. Since the main problem is filling in the missing values, there minimal data reprocessing. The only two features that were dropped from the dataset are TransactionID and TransactionDT because they were unique for each datapoint and did not contain any useful information.

After the autoencoder models finish training, the performance of the models will be tested against the statistical approach to fill in the missing values using f1 score with a Random Forest Classifier. In addition, the effect of dimension reduction on performance will be analyzed using Principal Component Analysis and Linear Discriminant Analysis methods.

2.2 Task 2

For the second par of the assignment the data comes from the UNSW-NB15 network intrusion benchmark dataset. The dataset contains ten classes: Analysis, Backdoor, DoS, Exploits, Fuzzers, Generic, Normal, Reconnaissance, Shell Code, and Worms. Also, each class can be categorized into two classes: normal sequence and intrusion, however, these two classes will not be used in this work. The dataset contains no empty values, however, some of the values of the categorical features can be interpreted as a missing value. Nevertheless, there are only 4 categorical values out 39 legitimate features. After further analysis, I decided to keep all categorical as they are and encode them to numerical representation. Three of the features were dropped because more than 98% of the data inside of them were a specific value. These features are: is_ftp_login, ct_ftp_cmd, and is_sm_ips_ports. After that, id feature was dropped from the dataset and the final dataset was scaled using Standard Scaler.

3 Results

3.1 Task 1

The results of the first task showed mixed success. The best performing autoencoder was the Undercomplete Autoencoder. The performance scores can be seen in the table 1. Considering that the all three autoencoders were training for 20 epochs and batch size of 128, all of them spaced relatively far apart from each other, which can also be observed in the figure 1. According to my observations, the Regularized Autoencoder struggled to improve the value of test loss with higher values of regularization coefficient. This probably means that the underlying implementation of the autoencoder does not overfit the data, thus, regularization only slows the learning process. Comparing the Undercomplete Autoencoder to the statistical method of filling in the missing values with the mean of the column showed, that the model was not able to recognize the pattern of the fraudulent transactions, which can be seen in the table 1.

After assessing the performance of the autoencoders, I performed dimension reduction on the regenerated data to see the effect of the dimension reduction. At first, I analyzed how much variance is explained by each component, as shown in the figure 2. In both cases around 46% of the variance was explained by the the first 2 components, as can be seen in the figure 3. I tests different values of components up until 100 components. At 100 components both dimension reduction algorithms explained almost 99% of the variance, however, dimension reduction only resulted in decrease in performance.

3.2 Task 2

For the second part of the assignment I implemented and trained a Generative Adversarial Network (GAN) for 500 epochs. The progress of the training can be seen in the figure 4. As can be seen from the graph, the Generator loss starts to rapidly grow near epoch 300. After further testing, the Generator at epoch 300 showed the most optimal results. Meaning that the after epoch number 300 the Discriminator started to outpace the Generator, which started to negatively impact the learning process.

In order to test the performance of the GAN model, I performed classification on the test dataset before and after the train dataset was balanced with the model. For the classification, I used three different models: Random Forest Classifier from *sklearn*, Explainable Boosting Classifier from *interpret*, and my implementation of simple 2-layer Fully Connected Neural Network written in *pytorch*. The results of the comparison can be observed in the table 2. According to the results of the classification, the classifiers trained on the regenerated dataset had nearly the same performance as the classifiers trained on the imbalanced dataset. Further investigation shows, that the classifiers trained on regenerated dataset have slightly higher F1 score on the rare classes.

4 Discussion

Looking at the results of the both task 1 and task 2, the both machine learning approaches were able to complete their tasks with mixed success. In the task 1, the autoencoder model was not able to outperform a simple substitution of the mean into the columns. The fact that the test error of the regularized and the variational autoencoders always stayed higher than the error value of the undercomplete autoencoder might suggest that the autoencoder model was not complex enough. In the task 2, the conditional GAN model was able to consistently regenerate the dataset, which can be proven by relatively small value of std in the table 2. However, the generated data did not significantly impact the F1 score of the classification algorithms. This might be due to nature of the data, thus, the classifier cannot learn to distinguish between rare and common classes. Furthermore, the figure 4 suggests that the current implementation the GAN will not improve its results with increased number of training epochs.

5 Limitations

Both of the datasets used in this assignments contained significantly imbalanced classes and Vesta's dataset consisted mostly of missing values, which impacted the performance of analyzed models. Another limitation stated in the Vesta's dataset description is that this dataset is not robust to false labeling, meaning that fraud transactions could have been labeled as normal.

Another limiting factor was Explainable Boosting Classifier from *interpret* library. It was only possible to run this classifier with the minimal number of "bins", which is an important hyper-parameter that significantly affects the classification performance. If the classifier is set to the number of bins larger than 2, it will try to fit the data seemingly indefinitely. Thus, the value for this classifier in the table 2 is not representative. The documentation of *interpret* library states that multi-class functionality of this classifier will change in the future, thus, this limitation might change in the future.

6 Conclusion

In conclusion, both task of the assignment were complete and uploaded to github. The investigated machine learning models did not improve on the baseline approaches, however, they were able to show similar performance. For potential future iteration, it would beneficial to try more complex and deeper neural network models.

Appendices

A Task 1

Table 1. Performance of the autoencoders from the Task 1

Approach	Test loss	F1 class 1	F1 class 2
Undercomplete Autoencoder	0.1257	0.99	0.51
Regularized Autoencoder	0.4952		
Variational Autoencoder	0.8471		
Fill with mean		0.99	0.60

Figure 1. Loss values by batches

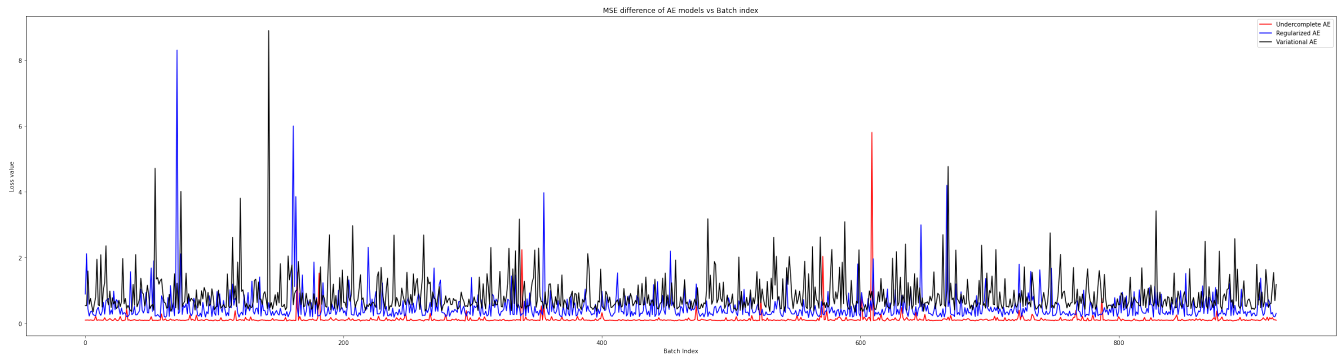


Figure 2. Explained variance by each component

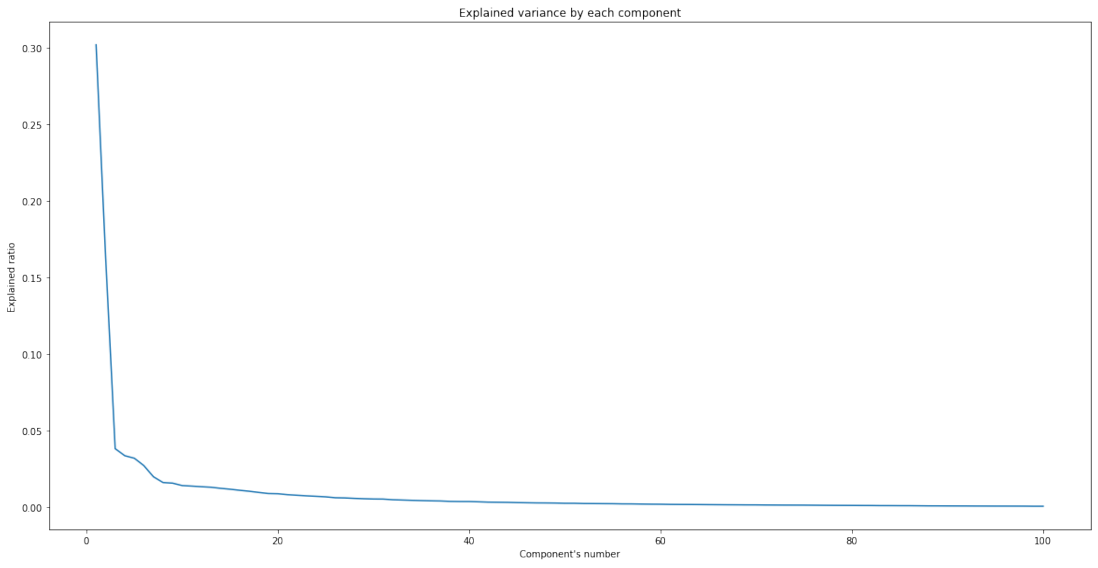
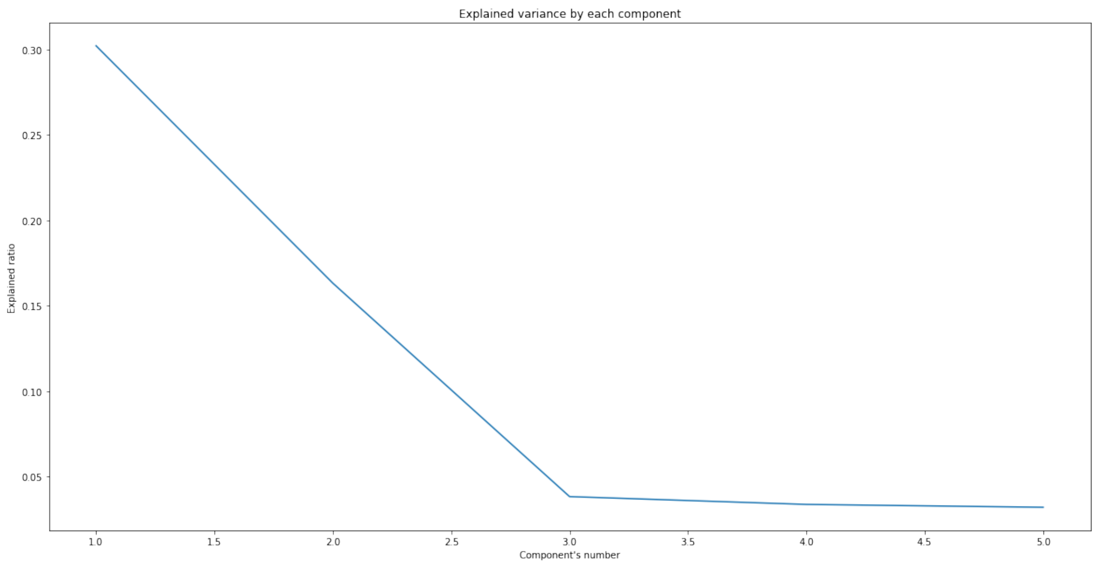


Figure 3. Explained variance by each component zoom in on the interval [1, 5]



B Task 2

Figure 4. Generator and Discriminator loss progression over epochs

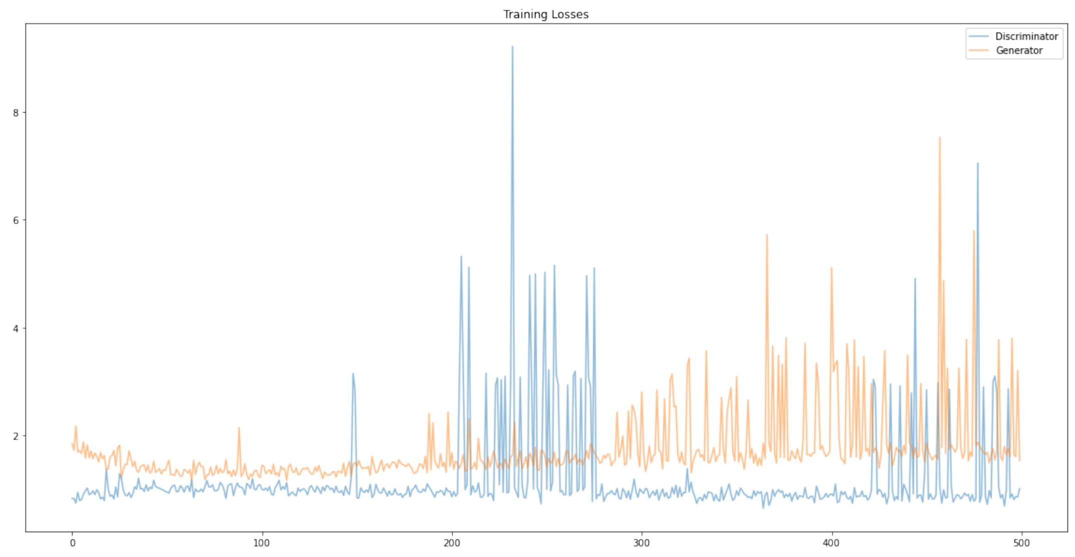


Table 2. Performance of the GAN from the Task 2

Metric	**Imbalanced dataset**	**Regenerated Dataset**
Random Forest F1	0.7788	0.7790
Random Forest std		0.0003
Explainable Boosting F1	0.2787	0.0129
Explainable Boosting std		0.0157
Neural Network F1	0.7151	0.7022
Neural Network std		0.0044
Class 0 F1	0.85	0.85
Class 1 F1	0.03	0.04
Class 2 F1	0.00	0.03
Class 3 F1	0.39	0.38
Class 4 F1	0.46	0.46
Class 5 F1	0.86	0.86
Class 6 F1	0.69	0.69
Class 7 F1	0.19	0.19
Class 8 F1	0.28	0.29
Class 9 F1	0.98	0.98