Introduction to Machine Learning (SS 2024)
Project: Programming Project

Author 1
Last name: Rieser
First name: David
Matrikel Nr.: 12141689

Author 2
Last name: Sillaber
First name: Cedric
Matrikel Nr.: 12211124

This template outlines the sections that your report must contain. Inside each section, we provide pointers to what you should write about in that section (in blue text).

Please remove all the text in blue in your report! Your report should be 2 pages for regular teams (excluding references!) and 3 pages for the three person team.

I. Introduction

The task of this project is to implement machine learning models to detect fraudulent transactoins. The problem type is a binary classification task. Based on the timestamp, value of transactions and 28 features the model should predict if a transaction is fraudulent or not. The dataset consists of 227845 instances with 30 features and flag indicating if the transaction is fraudulent or not. The dataset is highly imbalanced with 0.15% of the transactions being fraudulent. There is no information about the 28 features, so it is not clear what they represent.

As there the dataset is labelled, we can use supervised learning to solve this problem. We decided two use two methods for this problem. The first choice was to use a neural network for this task. Neural networks can extract the most important features from the data and are able to learn complex patterns. The second chance was a decision tree. In order to achieve even better results, we also implemeted a random forest classification.

- What is the nature of your task (regression/classification)? Is it about classifying types of birds, or deciding the number of cookies an employee receives?
- Describe the dataset (number of features, number of instances, types of features, missing data, data imbalances, or any other relevant information).

II. Implementation / ML Process

As mentioned in the introduction the dataset is highly imbalanced. There are 227845 datapoints and less than 500 are fraudulent. This is about 0.173% of the dataset. A naive implementation of the neural network was not able to detect frauds corretly. With an accuracy of 99.83% most likely the frauds were not detected. In order to solve this problem, we decided to do simple data augmentation. As other methods were not adequate, we simply duplicated the fraudulent transactions. This was done the following way: The dataset was split into train and validation set. Then the fraudulent transactions in the train set were duplicated randomly by a factor. Using this augmentation method it was possbile to give the frauds more weight in the train set, without changing the validation set. Additionally we scaled the data using a normalization method. This functionality is achieved by calling scikit-learn's 'StandardScaler'.

The timestamp in the dataset was removed as it is most likely not important. For the problem we used two Methods: Multilayer Perceptron and Decision Tree.

A. Multilayer Perceptron

We used a Neural Network for the classification task. The Neural Network is based on 'pytorch''s 'nn.Module'. For the Network there are three layers. A input layer with 30 nodes, a hidden layer with dimensionality 94 and a learning rate of 0.0004301150216793739. The activation function is a ReLU function. The output layer has one node and a sigmoid activation function.

A Neural Network is suitable for this problem as it can learn complex patterns and extract the most important features from the data. We don't have any information about the features, so a Neural Network is a good choice. The programmer does not have to make any choices concerning the features. The dataset contains labeled data, so supervised learning and thus neural networks seem a good option.

The choice of hyperparameters was really important and made a difference. In this case the hyperparameters where factor for data augmentation, number of nodes in hidden layer, learning rate and learning epochs. To find the best hyperparameters we used brute force method to train using randomly selected parameters. The parameters where randomly selected from a specified range. Given the results the best hyperparameters where chosen. The final hyperparameters are: factor for data augmentation: 50, number of nodes in hidden layer: 94, learning rate: 0.0004301150216793739, learning epochs: 36.

B. Decision Tree

The Decision Tree is based on the ‘sklearn‘ library. The Decision Tree is a simple model that can be used for classification tasks. ...

Further we tested a modification of decision trees - the random forest classification. This method is based on the decision tree, but uses multiple trees to improve the accuracy. The accuracy was about 99.959% and the model was able to detect the frauds. This is about the same as the accuracy of the decision tree.

- Did you need to pre-process the dataset (e.g. augmenting data points, extracting features, reducing the dimensionality, etc.)? If so, describe how you did this.
- Specify the method (e.g. linear regression, or neural network, etc.). You do not have to describe the algorithm in detail, but rather the algorithm family and the properties of the algorithm within that family, e.g. which distance functions for a decision tree, what architecture (layers and activations) for a neural network, etc.
- State (in 2-5 lines) what makes the algorithm you chose suitable for this problem. What are the reasons for choosing your ML method over others?
- If you used a method that was not covered in the VO, describe how it is different from the closest method described in the VO.
- How did you choose hyperparameters (other design choices) and what are the values of the hyperparameters you chose for your final model? How did you make sure that the choice of hyperparameters works well?

III. Results

For the MLP model the final accuracy on the validation set was 99.9414%, while the train accuracy is only about 98%. This rather bad accuracyon the train set is caused by the data augmentation. Nevertheless, the model is able to detect the frauds in the validatoion set, this is whats important.

The decision tree impelentation has a validation accuracy of 99.96%. The train accuracy is about...

As a naive baseline predictor we present a model that only outputs 0, means no fraud. Because the dataset is imbalanced and data augmentaiton does not affect the validation set there are only maximum only 0.15% frauds in the validation set. Thus the naive predictor has a accuracy of at least 99.85%. Both implementations are significantly better than the naive baseline predictor. The decision tree and random forest classification are able to detect the frauds.

- Describe the performance of your model (in terms of the metrics for your dataset) on the training and validation sets with the help of plots or/and tables.
- You must provide at least two separate visualizations (plot or tables) of different things, i.e. don’t use a table and a bar plot of the same metrics. At least three visualizations are required for the 3 person team.

IV. Discussion

As mentioned above, our models are significantly better than a model that only predicts no fraudulent transactions. With an accuracy of 99.96 % the decision tree and random forest classification are able to detect some of the frauds. However as there is still

- Analyze the results presented in the report (comment on what contributed to the good or bad results). If your method does not work well, try to analyze why this is the case.
- Describe very briefly what you tried but did not keep for your final implementation (e.g. things you tried but that did not work, discarded ideas, etc.).
- How could you try to improve your results? What else would you want to try?

V. Conclusion

- Finally, describe the test-set performance you achieved. Do not optimize your method based on the test set performance!
- Write a 5-10 line paragraph describing the main take-away of your project.