Association Rules for Common Prediction Failure

Elad Shechter, Noam Lahmani

March 7, 2023

1 Abstract

Learning in tabular data science involves techniques such as data preprocessing, feature engineering, and model training and evaluation.
It is widely used in various industries such as finance, healthcare, and retail, where businesses rely on insights derived from previous data to make informed decisions.

The goal of learning previous data is to extract meaningful information from it, and make accurate predictions, while also understanding the limitations and potential biases of the data and models used.

Our goal is to use machine learning algorithms to make accurate predictions on data presented in tables or spreadsheets.
By cleaning, preprocessing, and transforming the data, selecting relevant features, and training models, we aim to extract insights, patterns, and trends from the data. Ultimately, our goal is to create a model that can be used to make informed decisions and predictions on new data.

Summary -
· Problem Overview : Creating overfit by training the model on big / repetitive data.
· Solution Overview : Filtering the trained data by taking only common errors.

1.1 Problem Description

The pipeline involves the following stages:
· Loading the dataset.
· Preprocessing the dataset (handling missing values and encoding categorical variables).
· Splitting the dataset into training and testing sets.
· Scaling the features of the dataset.
· Training a machine learning model on the training set.
· Evaluating the performance of the trained model on the testing set.

The code uses various techniques such as pre-processing the table, hyperparameter tuning to improve the model's performance.

The primary problem that the pipeline suffers from is overfitting, which occurs when the model performs well on the training data but poorly on the testing data. This is a common issue in machine learning tasks, especially when training with small datasets.

1.2 Solution Description

The general element of the data science pipeline that this article is attempting to improve/change is the model training and evaluation step.

To address the overfitting issue and improve the general performance of the model, we try to filter incoming new data, by searching for the common denominator in the model's failures and simplify the model's complexity.

Current implementation compares between 2 regression models - our new suggested model, and the "normal" approach, which includes training on every known error the model made.

The following actions were carried out on the tabular data:

We addressed NULL values by filling them in, as the model is unable to process them. Then, we utilized binning primarily for association rules since they work best with categorical data.

We performed One-Hot Encoding on the data so the model could process the numerical values the fastest.
Therefore, we converted each categorical value, of a single column, into columns that indicate the column's initial categorical value.

In order to test the effectiveness of a machine learning model, it is crucial to split the dataset into two categories: training data and testing data.
The model is then trained and evaluated.

The first stage involves training the model and obtaining the predicted results, which is then used for cross-checking against the actual results in the test data to identify any discrepancies and the mean absolute error. By doing so, we can determine the accuracy of the model in predicting the values.
In the second stage, association rules are utilized to identify errors that share similarities. This process enables us to derive insights from the mistakes and develop strategies to enhance the model's performance. By utilizing these insights, we can refine the model and make it more accurate in its predictions.

Overall, the process of evaluating a machine learning model involves training it on a subset of data, testing its performance on another subset, and identifying errors to improve its performance. Using association rules to identify similarities among errors provides valuable insights. Ultimately, the effectiveness of a machine learning model is determined by its ability to accurately predict values and make informed decisions.

Our model differs by demanding the user's feedback before any further training and error analysis. Only then our model could perform best.

2 Solution Overview

Our solution introduces a tool which can be used by the general case we've made. The general case utilizes a script that can be used with any dataset by inserting the target feature followed by the filename of the dataset. It performs by dividing the dataset into a number of parts defined by the user, and generates several tests, a subset for validation, and a subset for initial training.
(*For future implementations, we have saved binary files to continue the same process / dataset, in case where the host pc is shutdown).

The model runs by using the train file generated in previous steps, and as many tests as desired, with the model outputting predictions in an output file with an appropriate name. The results of each test are then provided to the model for continued learning, focusing on the prediction errors that have a common denominator.

After the learning process is complete, the function returns the trained model, and the performance of the filtered and unfiltered models is compared through the use of graphs. On the other hand, the normal model's learning process is trained upon the dataset, and similar to our model, also learns from the entirety of its mistakes.

This allows for a comparison of the performance of a model that filters and takes errors with a common denominator, against a model that does not perform this filtering at all. Our work mainly utilized the pandas module for working with tabular data, the xgboost module for creating and training models upon the data provided, and sklearn's inner class - OneHotEncoder, to preprocess the data, and finally efficient-apriori, to calculate all the itemsets of the prediction failures.

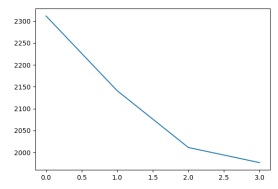
3 Experimental Evaluation

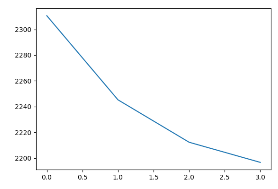
We tested our model on 4 datasets and compared the results of our model to the normal model.

3.1 India flight prices

The database on Kaggle provides data on airline ticket prices and related parameters for use in building predictive models for flight price prediction.

Our result: Normal result:

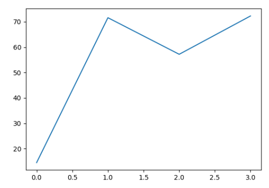


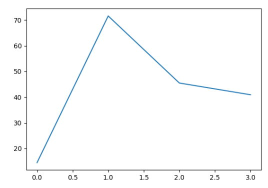


3.2 Energy

This database contains wind turbine energy generation data in kilowatts.

Our result: Normal result:

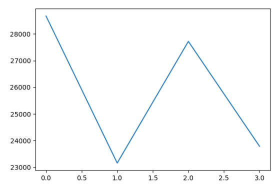


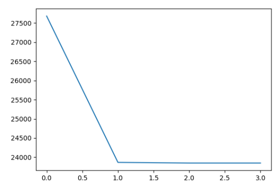


3.3 House prices

This database contains information on various attributes of residential homes in Ames, Iowa.

Our result: Normal result:

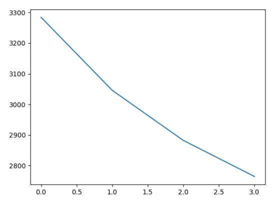




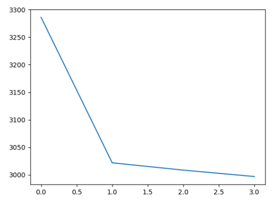
3.4 Used cars

This database provides information about used cars in the US, including their make, model, year of manufacture, and other relevant features such as mileage and price.

Our result:



Normal result:



4 Related Work

According the article “What is Overfitting in Deep Learning”[1], overfitting occurs when a model becomes too complex, such as when additional neural layers are added, which then require more training time and execution per test sample. We believe that the performance of our model is better than the normal model because of the complexity of the normal model, and the overall MAE.

Additional article that called “Summary of Association Rules”[2] data science techniques are show for their ability to inform decision-making in areas such as drought management, through the analysis of vast amounts of data. Conversely, in our project, we implement association rules to decrease the amount of training data, focusing solely on rows that share common errors.

5 Conclusion

In conclusion, we were amazed by the very small number of articles talking about optimizing the amount and quality of trained data. The solution is in fact rather simple in terms of execution - We have done the following:

· Preprocessed the data by filling null-values, so the model could process it easily.
· Categorize, and bin if needed be, the entire data so the apriori execution could go on every feature in the data, not to skip / halt on numerical features.
· Experiment and use various method to calculate feature’s correlation between one another (To decrease the complexity of calculating the association rules).
· Filter the new data by using common denominators across a subset sample.

We have found our solution to be extremely effective when dealing with small train samples and a large portion of new data flowing in, such as Real-Time environments. For future implementations we prepare to save the faulty predictions within a container, and to test each time whether there was a common denominator across executions, rather than per execution. Also having treatment towards system-shutdown, such as saving binary files to run the application again as soon as the system is up.

6 References

[1] “What is Overfitting in Deep Learning” - https://www.v7labs.com/blog/overfitting
[2] “Summary of Association Rules” - https://iopscience.iop.org/article/10.1088/1755-1315/252/3/032219/pdf
[3] “Used Cars dataset” - https://www.kaggle.com/datasets/ananaymital/us-used-cars-dataset
[4] “Wind Turbines Power Generation dataset” - https://www.kaggle.com/datasets/psycon/wind-turbine-energy-kw-generation-data
[5] “Flight Prices in India dataset” - https://www.kaggle.com/datasets/shubhambathwal/flight-price-prediction
[6] “House Prices dataset” - https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques