Automating The Analysis Of Model Faults And Mispredictions Results

Tabular Data Science Final Course Project 2023

Samuel Memmi 342677358, Moria Rais 300977782

Abstract

The project includes an automated process to streamline the model building, evaluation, and training phases of K-means clustering and the SVM algorithm, aimed at achieving the most accurate analysis of model faults and misprediction results. The process involves data cleaning, conversion, and feature selection, followed by the use of K-means clustering and PyTorch to identify the mispredicted data, group it into similar clusters, and select the most important features. In Addition to K-Means, we used the SVM algorithm and automated it using the H2O AutoML framework. The aim is to prevent overfitting or underfitting of the model and to remove irrelevant or redundant features to achieve a more robust and accurate model. The results for both are visualized using a bar plot, histogram plot, and kernel density estimate plot. The process is applied to several datasets, including Telco Customer Churn, Company Bankruptcy Prediction, Credit card fraud, and Pima Indians Diabetes.

Problem description

Analyzing model faults and mispredictions can be challenging due to the complexity of machine learning models and their training process. For instance, during the course, we learned about "linear regression," where the fit, train, and scale stages were carried out manually. This involved understanding the models, and their predictions, and identifying factors causing misclassification, from data cleaning to training. To address this challenge in our project, we developed an automated process to streamline these steps. The aim is to obtain the best model for training and consequently, to achieve the most accurate analysis of model faults and misprediction results.

The element of the data science pipeline that the code is trying to improve is the model building, evaluation, and training phase for both K-means clustering and the SVM algorithm. The code aims to enhance the performance of a logistic regression model by leveraging K-means clustering and the SVM algorithm. The automatic feature selection aims to prevent potential overfitting or underfitting of the model and remove irrelevant or redundant features that can influence the model's performance. The code identifies the most important features for each cluster of mispredictions and eliminates the least important features, resulting in a more robust and accurate model for both K-means clustering and the SVM algorithm.

Solution overview

1. Process data

We used the following datasets during the automation process and implantation:

A. Telco Customer Churn
B. Company Bankruptcy Prediction
C. Credit card fraud
D. Pima Indians Diabetes

For each dataset, we conducted cleaning, conversion, and feature selection in preparation for machine learning. Our aim was to streamline the process as much as possible and minimize manual intervention. However, we discovered that the data was imbalanced, as our accuracy was high but other metrics were poor. To address this, we used the SMOTE (Synthetic Minority Over-sampling Technique) method to oversample the data and balance it. We then split the oversampled data into training and testing sets using the train_test_split function from the scikit-learn library.

2. Machine learning methods

2.1 K-means & PyTorch

In this automation we send two goals: the first one was to use the k-means algorithm to group together similar examples of mispredictions and then use this information to identify common characteristics or features that may be contributing to the errors, and the second one was to use PyTorch as a tool for automation, evaluation, and improvement of the model.

The k-means is an unsupervised machine-learning technique used for clustering data. It groups data points into k clusters based on similarity. We wanted to use PyTorch to automate the analysis of model faults and mispredictions because it has the ability to handle large amounts of data. It also provides a lot of functionality for data processing and manipulation, which is important for the analysis of model faults and mispredictions.

Our main goal is to automate the analysis of model faults and mispredictions. We used the k-means algorithm to achieve the main goal as described in the following paragraph. First, the model we choose to automate the analysis is logistic regression. So the first step is to train a logistic regression model on the training data and evaluate its performance on the testing data using accuracy, precision, recall, and f1 score metrics. These metrics provide a measure of how well the model is making predictions and how well it is able to identify positive cases (for example customers who have churned), a bar plot is created to visualize the values of these metrics. We want to implement a process to improve the performance of this model. The purpose of this process is to identify and correct mispredictions made by the logistic regression model. The next step is to identify the mispredictions made by the model and their associated feature values. For that, we want to identify the instances where the model made incorrect predictions by

comparing the model's predictions with the actual target variables. The mispredicted data is then grouped into similar clusters based on their feature values using k-means clustering (we use the elbow method to determine the optimal number of clusters before we do the k-means). The following step is the feature selection. For each cluster of mispredictions, we use the SelectKBest method to identify the most important features that contribute to the misprediction. This function performs a statistical test to determine the significance of each feature in relation to the target variable. We can make changes to the model by removing the least important features from the training and testing data. After that we can retrain the model, the logistic regression model is then retrained on the modified training data without the least important features. The retrained model's accuracy, precision, recall, and f1 score are calculated. We calculated the improvement in accuracy, precision, recall, and f1 score of the retrained model compared to the original model. Finally, we can use the results_visualization function to visualize the results of the new logistic regression model to better understand the model faults and mispredictions. We use the predictions of the best model that we have and convert the predictions and actual outcomes to PyTorch tensors. The PyTorch tensors are then used to create two different plots to visualize the distribution of the predictions and actual outcomes. The first plot is a histogram plot that shows the distribution of model predictions and actual outcomes. It plots the frequency of the target variables for both the predictions and actual outcomes in different bins. The second plot is a kernel density estimate plot that shows the probability density of the target variables for both the predictions and actual outcomes.

2.2 SVM & H20AutoML

For automating support vector machine(SVM) training we used the H2O AutoML framework. Choosing this framework was done because of the possibility to run the library over the windows operating system. The function ‘svm_training’ is responsible for training the data. First, we standardized the training and testing data using the StandardScaler class from the ‘scikit-learn’ library. It then initializes an H2O cluster and converts the training data to an H2OFrame object, which is the data structure used by H2O. The target and predictor variables are specified, and H2O AutoML is run to automatically select, train, and optimize an SVM model. The max_models parameter is set to 10, the sort_metric is set to 'mse', and the max_runtime_secs is set to 5 minutes. MSE is a common metric used in regression problems to evaluate the performance of a model. It measures the average squared difference between the predicted and actual values of the target variable.| The H2O AutoMLinitial object is set to the maximum runtime to frame the training duration. Also, we will set max_models = 15 to make sure that AutoML trains all 15 models in constant framed time. Next, the step displays the leaderboard of trained models, which is the performance metrics of the trained models, and selects the best model based on the sort_metric. The testing data is then converted to an H2OFrame object, and the prediction method of the best model is used to make predictions on the testing data. After a second train is executed and done with the SVM model. This model

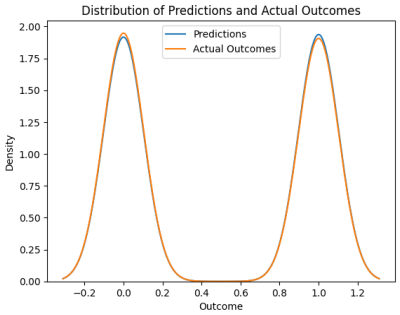
performs feature selection using the SequentialFeatureSelector class and uses an SVM model with the same hyperparameters as the best model from the H2O AutoML training. The best model from the H2O AutoML training is set as the base estimator for the SVM model using the named_steps attribute of the Pipeline object. As a final step, the SVM model is trained on the standardized training data and the prediction method is used to make predictions on the testing data. A comparison between the best model and the final SVM model evaluation parameters is taken.

Experimental evaluation

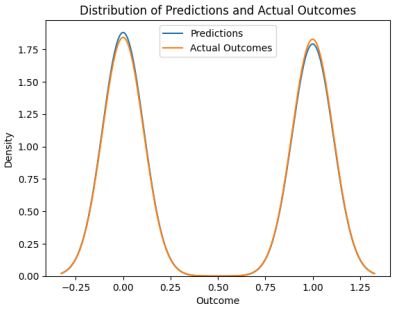
1. Inner comparison between the best model and second training with the machine learning method

We analyze each of the datasets trained and compare the results between the two training outcomes, we have received the following from the algorithms methods:

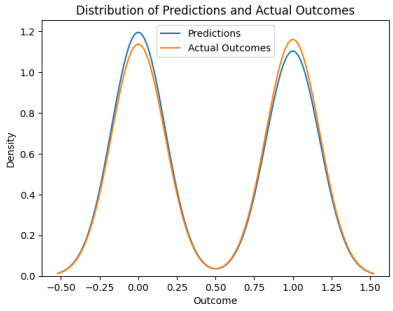
K-means



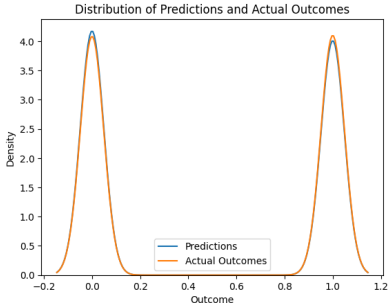
Company bankruptcy: Better accuracy of the K-means model of 0.757%, better precision of the model of 0.81%, better recall of the model of 0.689%, better f1 score of the model of 0.75%.



Customer churn: same here, Better accuracy of the K-means model of 0.0%, better precision of the model of 0.687%, better recall of the model of 0.68%, better f1 score of the model of 0.0%.



Diabetes disease: Better accuracy of the K-means model of 0.5%, better precision of the model of 1.04%, better recall of the model of 0.99%, better f1 score of the model of 0.65%.



Credit card fraud: Better accuracy of the K-means model of 0.0035%, better precision of the model of 0.0014%, better recall of the model of 0.0087%, better f1 score of the model of 0.0036%.

The functions performed as expected and have the potential to improve the performance of the model automatically. However, despite our efforts to optimize the function, we only achieved a marginal improvement in the model's performance. In the visualization, the curve almost overlaps which is very good but we can have very little underfitting (when the actual outcome is over the predictions) and we can have very little overfitting (when the inverse is happening).

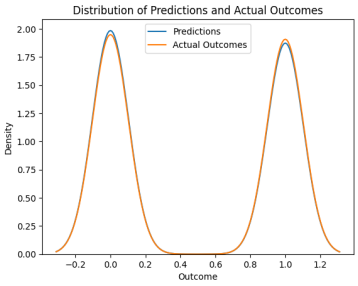
SVM

During the training of the SVM model and getting the best model we received a mix of binary and continuous targets. Therefore, we had to round the values and calculate the evaluation values above. In 2 out of 4 models, we received low values. We assume it might be because of the method we used to round the values (we have tried many methods and this was the recommended one). Another option is that the prediction wasn't successful and influenced next on the continued training with the SVM model. As a result of that, in addition to the metrics mentioned, we add analyze for the best model with the measurement:

MSE (Mean Squared Error): This measures the average squared difference between the predicted and actual values. Lower values indicate better performance.

MAE (Mean Absolute Error): This measures the average absolute difference between the predicted and actual values. Lower values indicate better performance.

Mean Residual Deviance: This is a measure of the quality of the fit of the model. Lower values indicate better performance.

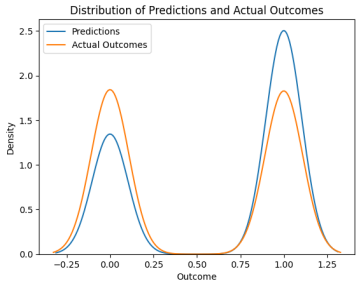


Company bankruptcy:

We gain a better accuracy of the model of -58.9%, better precision of the model of -81.59%, better recall of the model of -81.85%, and a better f1 score of the model of -81.86%.

MSE: 0.0039, MAE: 0.0301, MRD:0.0039

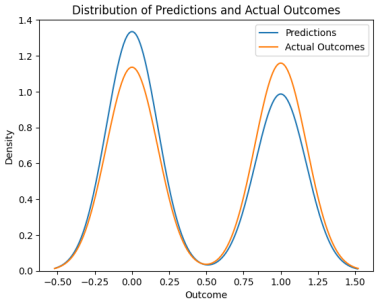
There is a contradiction in this model since the MSE, MAE, and mRD measurements were good but the accuracy, prediction, recall, and F1 score were lower compared to the graph. We assume that in this case converting the model to binary made this gap.



Customer churn:

We gain a better accuracy of the model of -29.38%, better precision of the model of -45.008%, better recall of the model of -80.369%, and a better f1 score of the model of -65.508%.

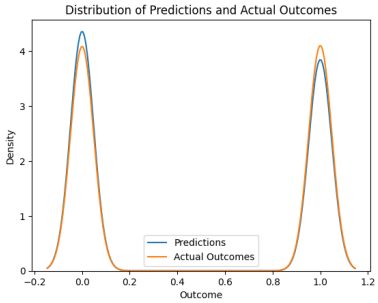
MSE:0.0935 , MAE: 0.1059, MRD:0.0935. In this model the performance of the training was low and we can see that by the graph and the bad prediction rates compared to the other datasets models.



Diabetes disease:

We gained a better accuracy of the model of 14%, better precision of the model of 7.579%, better recall of the model of 27.72%, and a better f1 score of the model of 17.867%.

MSE:0.0249, MAE:0.1146, MRD:0.0249. The Values are lower which indicated good performance



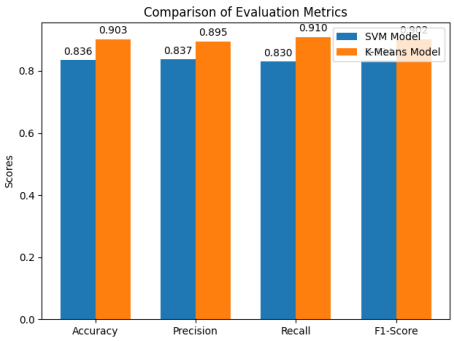
Credit card fraud:

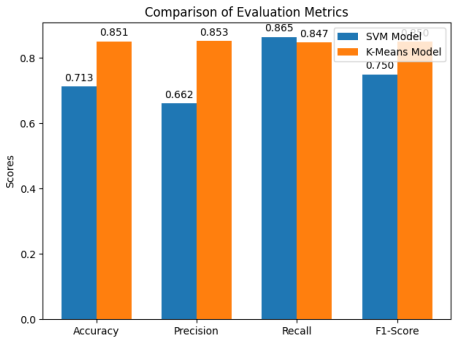
We gain a better accuracy of the model of 5.507%, better precision of the model of 2.408%, a better recall of the model of 8.744%, and a better f1 score of the model of 5.68%

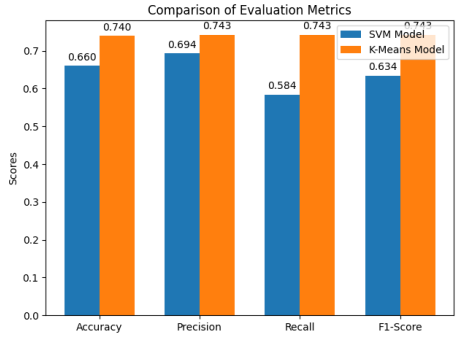
MSE: 0.0044, MAE: 0.0082, MRD:0.0004. The Values are lower which indicated good performance

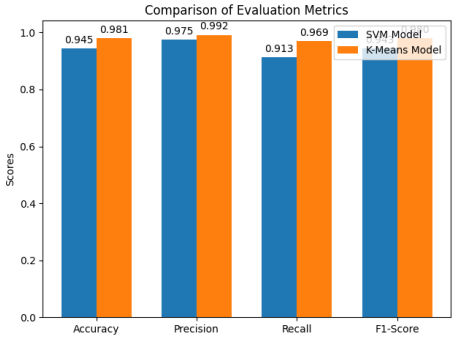
2. <u>Comparison between the 2 models</u>

Both K-Means and SVM are used to improve predictive models. K-means identifies similar customer groups and removes the least important features that contribute to the misprediction, while SVM optimizes the model using feature selection and scaling. K-means uses the elbow method to find the optimal number of clusters, while SVM uses H20 AutoML to select the best model. The choice between the two depends on the requirements of the specific problem and will be determined by comparing their performance on the 4 datasets. We will perform the 4 datasets in order: Company bankruptcy, Customer churn, Diabetes disease, and Credit card fraud. The graphs are arranged in order from left to right, following the list.









When trying to search for the reason why the K-Means incline the results for the training we assume it can occur for several reasons:

1. Way of implementation: The SVM training was implemented with H2O AutoML while setting the best model as the base estimator for the SVM model. Different settings than the training with K-Means models. It might affect the results and the ability to create better values such as accuracy, precision, recall, and f1.
2. Number of Features: SVM can struggle when dealing with high-dimensional data, especially if the number of features is much greater than the number of samples. This is because SVM tries to find a hyperplane that separates the data into classes, and in high-dimensional spaces, the data may be spread out in a way that makes it difficult to find a clear separation. K-means, on the other hand, is less affected by the curse of dimensionality and can often handle high-dimensional data more effectively.
3. Data Clustering: k-means is a clustering algorithm, meaning that it is designed to partition data into groups or clusters based on similarity. If your data naturally forms distinct clusters that correspond to different classes or categories, then k-means may be a better choice than SVM, which is a supervised learning algorithm that relies on labeled data to make predictions.
4. Data size: SVM might be computationally expensive for large datasets and hence, k-means may perform better on a larger dataset due to its simpler computational complexity.

Related work

Medical - "Early-Stage Alzheimer's Disease Prediction Using Machine Learning Models" (link: Front Public Health. 2022)

We compare our k-means machine learning approach with this related work that focused on predicting Alzheimer's disease. However, as our solution does not specifically target Alzheimer's disease, we will only be examining the differences in the machine learning techniques utilized. There have been several studies that aim to diagnose Alzheimer's disease using machine learning algorithms. Martinez-Murcia used deep convolutional autoencoders to analyze MRI images and extract features that represent an individual's cognitive symptoms and neurodegeneration process. Other studies have used deep neural networks for binary classification and used k-folds validation to choose the best-performing model. Several studies have attempted to diagnose Alzheimer's disease using different machine learning algorithms, contradictory to our solution that takes a unique approach to improve the accuracy of the predictions. Unlike other studies that have used deep convolutional autoencoders or deep neural networks, our solution uses a logistic regression model as the base for analysis. This allows for a simpler, more interpretable model that can be improved through feature selection and retraining. In addition, the studies have not explicitly addressed the issue of mispredictions made by the model. Our solution addresses this issue by identifying instances where the model made incorrect predictions and clustering the mispredicted data based on feature values.

This allows for a targeted approach to improving the model's accuracy by removing the least important features that contribute to the misprediction. The visual representation of the results provided in our solution has also a unique aspect that sets it apart from this study. The histogram and kernel density estimate plots offer a clear and intuitive understanding of the model's predictions and actual outcomes. This can be useful in identifying areas where the model can be improved and gaining insights into the underlying distribution of the target variables.

To conclude the comparison, our solution offers a unique and effective approach to machine learning techniques by improving the accuracy of a logistic regression model through feature selection and retraining. The solution also provides a clear and intuitive visual representation of the results to better understand the model's performance.

Conclusion

The project aims to make the training process automated. The process includes streamlining the model building, tuning hyperparameters, evaluation, and training phase for machine learning models. They use the K-means clustering and SVM algorithms to enhance the performance of a logistic regression model and prevent potential overfitting or underfitting of the model. We demonstrate the process using four different datasets, conduct cleaning, conversion, and feature selection, and use the SMOTE method to balance the imbalanced data. They then evaluate the performance of the model using metrics such as accuracy, precision, recall, and f1 score and identify the mispredictions made by the model. The authors use k-means clustering to group similar mispredictions together and identify the most important features contributing to the misclassification. They then eliminate the least important features and retrain the model, achieving an improvement in accuracy, precision, recall, and f1 score. Finally, the authors use PyTorch and H2O AutoML to automate the analysis of model faults and mispredictions and create visualizations to better understand the model's performance. Overall, this automated process improves the efficiency and accuracy of the model building, evaluation, and training phase, making it easier to identify and correct mispredictions and ultimately achieve better results. But as for getting the best results the K-Means algorithm was the leading method.

Other insights

1. It's important to note that data management cannot be fully automated as each dataset may require some level of cleaning and preprocessing before training can begin.
2. As we worked with the Windows framework, some Python libraries were readily available. However, it's possible that different results could be obtained in a Linux environment or with different libraries.