AI351 - Machine Learning Lab

Taiwanese Bankruptcy Statistics

Ali Hasan Khan
Artificial Intelligence(FCSE)
Ghulam Ishaq Khan Institute of Engineering and Technology
Topi,Swabi, Pakistan
u2021079@giki.edu.pk

I. Introduction

This code is a comprehensive script written in Python, primarily for data analysis and machine learning tasks. It involves various data preprocessing steps, model training and evaluation, as well as the use of the 'evalml' library for automated machine learning.

The script begins with the import of necessary libraries, including pandas, seaborn, and matplotlib for data analysis and visualization. It then loads a dataset from a CSV file, conducts preprocessing steps such as label encoding and handling missing values, and performs random under-sampling for dealing with imbalanced data.

Next, the script fits a decision tree model and a logistic regression model, evaluating their performance using various metrics such as accuracy, precision, recall, and F1 score. It further compares the performance of multiple classification models, including but not limited to Random Forest, Support Vector Machine, K-Nearest Neighbors, Naive Bayes, XGBoost, LightGBM, and others. ROC curves are plotted for each model, with their corresponding AUC scores.

Furthermore, the script leverages the 'evalml' library for automated machine learning. It conducts an automated search for the best algorithm using the 'AutoMLSearch' function, ranks the pipelines based on performance, and selects the best pipeline for scoring on holdout data. The script also demonstrates how to save and load the best-performing model.

Lastly, the script generates various visualizations, including scatter plots, histograms, and line plots using the 'autoviz' library to provide insights into the data and model predictions.

In summary, this code represents a complete end-to-end data analysis and machine learning pipeline, showcasing various techniques for data preprocessing, model training, evaluation, and automated machine learning.

II. Logistic Regression

The provided code snippet appears to be for a binary classification task using a logistic regression model. The model is trained using the 'X-train' and 'y-train' data and then evaluated on the 'X-test' data. The evaluation metrics, including accuracy, precision, recall, and F1 score, are calculated and printed. Additionally, the confusion matrix and classification report are generated and displayed.

978-1-6654-5992-1/22/$31.00 ©2022 IEEE

The output shows that the model achieves an accuracy of 0.66, indicating that it correctly predicts 66 percentage of the samples in the test set. The precision score of 0.74 suggests that when the model predicts a positive result, it is correct 74 percentage of the time. The recall score of 0.71 implies that the model can identify 71 percentage of the actual positive samples. The F1 score, which is the harmonic mean of precision and recall, is 0.73, signifying a balanced performance between precision and recall.

The confusion matrix provides additional insight into the model's performance, showing that it correctly predicts 9 instances of the first class and 20 instances of the second class. However, it misclassifies 7 instances of the first class and 8 instances of the second class. The classification report presents a detailed summary of the model's performance for each class, including precision, recall, and F1 scores, along with the support for each class.

Overall, the model demonstrates moderate performance in this binary classification task, indicating the potential for further optimization or the exploration of more complex models to enhance predictive accuracy.

III. All Possible Classifier

Certainly! The provided code involves the evaluation of several machine learning models for a binary classification task. Each model is trained using the 'X-train' and 'y-train' data and subsequently evaluated on the 'X-test' data. Here's an in-depth analysis of the methods and results:

- Logistic Regression:

This model yields moderate performance with an accuracy of 0.66. While it demonstrates a relatively higher precision of 0.74, suggesting that when it predicts a positive outcome, it is likely to be correct, its recall value of 0.71 indicates that it may not effectively capture all positive cases in the dataset.

- Decision Tree Classifier:

The decision tree model shows exceptional performance, achieving an accuracy of 1.0. This perfect accuracy may indicate potential overfitting, suggesting that the model might have memorized the training data, which may not generalize well to unseen data.

Random Forest Classifier: The random forest model exhibits robust performance, with an accuracy of 0.98. It shows balanced precision and recall values, indicating that it effectively captures both positive and negative cases without overfitting.

- Support Vector Classifier (SVC):

This model demonstrates decent performance, with an accuracy of 0.70. The precision value of 0.74 and the recall value of 0.82 suggest a balanced trade-off between accurately predicting positive cases and effectively capturing all positive instances in the dataset.

- K-Nearest Neighbors Classifier (KNN):

The KNN classifier's performance is modest, with an accuracy of 0.59. It exhibits relatively lower precision and recall values, indicating its limitations in effectively capturing both positive and negative cases in the dataset.

- Gaussian Naive Bayes (GaussianNB):

The Gaussian Naive Bayes model performs poorly, achieving an accuracy of 0.34. Its low precision and recall values of 0.33 and 0.04, respectively, indicate significant difficulties in capturing the complexities and nuances of the data.

- XGBoost Classifier:

The XGBoost model shows exceptional performance, with an accuracy of 1.0, suggesting robust generalization capabilities without overfitting. Its precision and recall values of 1.0 further support its effectiveness in accurately capturing both positive and negative cases in the dataset.

- LightGBM Classifier:

Similar to XGBoost, the LightGBM model demonstrates perfect performance with an accuracy of 1.0, indicating robust and accurate predictions without overfitting. Its precision and recall values of 1.0 further support its strong generalization capabilities.

- Multi-Layer Perceptron (MLP) Classifier:

The MLP classifier shows moderate performance, with an accuracy of 0.64. While its precision and recall values of 0.71 suggest a relatively balanced performance in capturing both positive and negative cases, there is room for further improvement, potentially through hyperparameter tuning and model optimization.

These results underscore the importance of selecting appropriate models based on the specific characteristics of the dataset, as well as the desired trade-offs between accuracy, precision, recall, and generalization capabilities. Furthermore, the results suggest the need for further analysis, including hyperparameter tuning and cross-validation, to optimize the performance of each model and improve their generalization capabilities.

IV. ROC-CURVES

The updated code includes the implementation and evaluation of additional machine learning models for the binary classification task. Here is a detailed analysis of the methods and results:

- Gradient Boosting Classifier:

The Gradient Boosting model exhibits strong performance, with an AUC of 0.94. Its high AUC value suggests that the model effectively distinguishes between positive and negative cases, indicating robust predictive capabilities.

- AdaBoost Classifier:

The AdaBoost model demonstrates moderate performance, achieving an AUC of 0.76. While its AUC value is lower compared to other models, it still shows potential for effectively classifying instances, albeit with some limitations.

Ridge Classifier: The Ridge Classifier performs reasonably well, with an AUC of 0.81. Its AUC value indicates its ability to effectively classify instances, although it may not be as robust as some other models.

- SGD Classifier:

The SGD Classifier exhibits modest performance, with an AUC of 0.71. While it demonstrates the ability to distinguish between positive and negative cases, its performance is relatively weaker compared to some other models.

- Linear Discriminant Analysis:

The Linear Discriminant Analysis model shows strong performance, with an AUC of 0.91. Its high AUC value indicates its effectiveness in distinguishing between positive and negative cases, suggesting robust predictive capabilities.

- Quadratic Discriminant Analysis:

The Quadratic Discriminant Analysis model performs well, with an AUC of 0.87. Its AUC value indicates its ability to effectively classify instances, although it may not be as robust as some other models.

- Bagging Classifier:

The Bagging Classifier demonstrates strong performance, with an AUC of 0.94. Its high AUC value suggests its effectiveness in distinguishing between positive and negative cases, indicating robust predictive capabilities.

- Extra Trees Classifier:

The Extra Trees model exhibits strong performance, with an AUC of 0.95. Its high AUC value suggests its effectiveness in distinguishing between positive and negative cases, indicating robust predictive capabilities.

- HistGradientBoosting Classifier:

The HistGradientBoosting model shows exceptional performance, with an AUC of 0.96. Its high AUC value suggests robust predictive capabilities, indicating its effectiveness in distinguishing between positive and negative cases.

- 1Gaussian Process Classifier:

The Gaussian Process model demonstrates moderate performance, with an AUC of 0.79. While its AUC value is relatively lower compared to some other models, it still shows potential for effectively classifying instances, albeit with some limitations.

- 1Multi-Layer Perceptron (MLP) Classifier:

The MLP classifier exhibits strong performance, with an AUC of 0.92. Its high AUC value suggests its effectiveness in distinguishing between positive and negative cases, indicating robust predictive capabilities.

- 1Voting Classifier:

The Voting Classifier combines the predictions from multiple models, including Random Forest, XGBoost, and LightGBM, to achieve an AUC of 0.95. This combined approach

effectively leverages the strengths of each model to enhance overall predictive capabilities.

- 1Stacking Classifier:

The Stacking Classifier combines the predictions from Random Forest, XGBoost, and LightGBM, using Logistic Regression as the final estimator, to achieve an AUC of 0.95. This approach effectively leverages the complementary strengths of each base model to enhance overall predictive capabilities.

The generated ROC curve graphically represents the performance of each model, with the AUC values providing a quantitative measure of the models' predictive capabilities. The varying AUC values highlight the differences in performance among the different classifiers, emphasizing the importance of selecting appropriate models based on the specific characteristics of the dataset and the desired trade-offs between predictive accuracy and generalization capabilities. Furthermore, the warnings logged during the execution of the LightGBM model training suggest potential issues or limitations that may require further investigation and troubleshooting for improved model performance.

V. EVAL ML

The provided script involves a series of operations aimed at automating the machine learning pipeline, optimizing model performance, and visualizing the results. Let's delve into the details of each step to gain a comprehensive understanding of the process.

First, the script loads the necessary libraries and imports the dataset using Pandas. The dataset is then split into training and testing sets using the EvalML library, which enables automated machine learning. The AutoML search is initiated, wherein different algorithms are evaluated for their performance on the provided dataset. This includes the computation of various evaluation metrics such as AUC, F1 score, precision, and recall.

The script then proceeds to rank the different pipelines based on their respective scores and provides a detailed description of the highest-ranking pipeline, highlighting its key components and parameters. Another AutoML search is performed, this time with a focus on optimizing the AUC metric. The rankings and details of the best pipeline based on the AUC optimization are presented.

Moreover, the script generates visualizations to facilitate a better understanding of the model's predictions and behavior. Scatter plots, histograms, and line plots are created to depict the distribution of predicted probabilities or scores, enabling a detailed analysis of the model's performance.

Furthermore, the script saves the generated visualizations as PDF files, allowing for easy access and sharing with relevant stakeholders or team members. This step ensures that the insights derived from the visualizations can be effectively communicated and utilized for decision-making purposes.

Overall, the script demonstrates a comprehensive approach to building, evaluating, and visualizing a machine learning model for binary classification. By leveraging the capabilities of the EvalML library, the script enables an automated and systematic exploration of different algorithms, ultimately leading to the selection of the most suitable model for the given dataset. The inclusion of visualizations further enhances the interpretability of the model's results and provides valuable insights for informed decision-making.