DITECH

Journal of Data Science, Technology, and Artificial Intelligences
ISSN XXXX-XXXX
Vol. 1, No. 1, July, 2024, pp. 29-32
Journal homepage: https://journal.itisd.org/index.php/ditech

Feature Importance and Binary Classification using PyCaret

Ahmad Fadhil Naswir[1*], Williem[2], Hasanul Fahmi[3]

[1] President University, Jababeka Education Park, Cikarang, Bekasi 17530, Indonesia
[2] President University, Jababeka Education Park, Cikarang, Bekasi 17530, Indonesia
[3] President University, Jababeka Education Park, Cikarang, Bekasi 17530, Indonesia

Corresponding Author Email: fadhil.naswir@president.ac.id

https://doi.org/xx.xxxxx/ditech.xxxxxx

Received : 23 March 2024
Revised : 25 July 2024
Accepted : 27 July 2024
Availabe online : 31 July 2024

Keywords:
Machine learning, PyCaret, Binary Classification, Feature Importance.

ABSTRACT

The rapid advancement of machine learning (ML) techniques has facilitated the development of robust models for various classification tasks. This study explores the application of PyCaret, an open-source, low-code machine learning library, to perform feature importance analysis and binary classification using the Titanic dataset from Kaggle. The dataset underwent preprocessing to convert categorical features into numerical values and to remove irrelevant columns. Multiple classification models were compared, with the Gradient Boosting Classifier achieving the highest performance, marked by an average accuracy of 81.52%. Detailed evaluation metrics, including precision, recall, F1 score, and AUC, further validated the model's effectiveness. Feature importance analysis identified gender (sex), fare, and age as the most significant predictors of survival, aligning with historical accounts. The results demonstrate PyCaret's capability to streamline the ML workflow, providing valuable insights and enabling rapid experimentation. This study highlights the potential of binary classification and feature importance analysis in handling large-scale datasets, where the identified important features can serve as a baseline for implementing advanced algorithms such as deep learning.

1. INTRODUCTION

In recent years, the rapid advancement of machine learning (ML) techniques has enabled the development of powerful models for a variety of applications, including classification tasks. Binary classification, which involves distinguishing between two classes, is a fundamental problem in the field of machine learning and has broad applications in areas such as medical diagnosis, fraud detection, and spam filtering.

Features are one of most important aspect for doing a classification especially on uncertain dataset. These features are typically represented as variables or attributes and provide information about the data points or samples being classified. With a good selection of features, it will helps the model for achieve a better result.

Feature importance plays a crucial role in the performance and interpretability of binary classification models. Understanding which features are most influential can help improve model accuracy, reduce overfitting, and provide insights into the underlying data. Traditional methods for assessing feature importance include statistical tests and simple models, but recent developments in ML libraries have made it easier to calculate and visualize feature importance using more sophisticated approaches.

In classification, features are the measurable or observable characteristics of the data that are used to make predictions about the class or category to which an instance belongs. Features can take various forms depending on the nature of the data and the problem at hand. They can be numerical, such as age, temperature, or income, representing continuous or discrete values. They can also be categorical, such as gender, color, or type, representing distinct categories or labels. Additionally, features can be binary, representing presence or absence of a certain attribute, or even text-based, representing words or phrases.

PyCaret, an open-source, low-code machine learning library in Python, has emerged as a valuable tool for simplifying and streamlining the end-to-end ML workflow. It provides a comprehensive suite of functionalities for data preprocessing, model training, hyperparameter tuning, and model evaluation, with a particular focus on ease of use and rapid experimentation. PyCaret's built-in tools for feature importance analysis and binary classification make it an ideal choice for researchers and practitioners looking to leverage state-of-the-art techniques without extensive coding.

In this paper, we explore the application of PyCaret for feature importance analysis and binary classification. We demonstrate how PyCaret can be used to preprocess data, train and evaluate models, and interpret the results through feature importance metrics. By leveraging PyCaret, we aim to provide a practical guide for researchers and practitioners to efficiently implement and understand binary classification models and their key features.

2. LITERATURE REVIEW

The task of binary classification and the analysis of feature importance have been extensively studied within the field of machine learning. Numerous approaches have been proposed to improve the accuracy and interpretability of classification models.

One of the seminal works in feature importance analysis is the introduction of decision tree-based methods. Breiman (2001) developed Random Forests, an ensemble learning method that leverages multiple decision trees to enhance classification accuracy and provides an inherent measure of feature importance. This technique has become a standard benchmark for feature importance analysis due to its robustness and interpretability .

Building on decision tree methods, Chen and Guestrin (2016) introduced XGBoost, an optimized distributed gradient boosting library designed to be highly efficient, flexible, and portable. XGBoost has been widely adopted for various machine learning competitions and applications due to its performance and scalability. It also includes mechanisms for calculating feature importance, which has contributed to its popularity in the research community .

The interpretability of machine learning models has been further advanced by the development of SHAP (SHapley Additive exPlanations) values, proposed by Lundberg and Lee (2017). SHAP values provide a unified approach to interpreting model predictions by attributing the contribution of each feature to the final prediction. This method has gained traction for its solid theoretical foundation and its applicability to a wide range of model types .

PyCaret, introduced by Mooney (2020), has made significant strides in democratizing machine learning by providing a low-code environment that simplifies the end-to-end ML workflow. PyCaret integrates various state-of-the-art algorithms, including Random Forests and XGBoost, and offers built-in tools for feature importance analysis. This ease of use and comprehensive functionality have positioned PyCaret as a valuable resource for both researchers and practitioners .

Moreover, the use of automated machine learning (AutoML) tools has surged in recent years. AutoML frameworks like H2O.ai and Google's AutoML provide automated feature engineering, model selection, and hyperparameter tuning. These frameworks have been instrumental in accelerating the development and deployment of high-performing models, making advanced ML techniques accessible to a broader audience. PyCaret stands out by offering a balance between automation and user control, allowing for customized and efficient model development .

In summary, the related work in binary classification and feature importance encompasses a range of methodologies from ensemble learning and gradient boosting to interpretability techniques and automated ML tools. PyCaret represents a convergence of these advancements, providing a user-friendly platform that facilitates the application and understanding of complex ML models.

3. METHODOLOGY

In this section, we outline the steps taken to analyze feature importance and perform binary classification using the PyCaret library. The Titanic dataset from Kaggle is used as the primary dataset for this analysis. The dataset contains information about the passengers of the Titanic, including demographic details, ticket information, and survival status.

3.1 Data Collection and Preparation

The Titanic dataset was obtained from Kaggle, which provides a comprehensive collection of datasets for various machine learning tasks.

3.1.1 Loading the Data

The dataset was loaded into a Pandas DataFrame for initial exploration and preprocessing.

3.1.2 Data Cleaning and Preprocessing

The initial preprocessing involved converting categorical features to numerical values and dropping irrelevant columns.
Converting Categorical Features: The "sex" column, which contains categorical values ("male" and "female"), was converted to numerical values (0 for "male" and 1 for "female").
Dropping Irrelevant Columns: The "PassengerId" column was dropped as it does not contribute to the prediction of survival.

3.2 Binary Classification Using PyCaret

PyCaret was utilized for feature importance analysis and to build and evaluate the binary classification model. The workflow involved setting up the PyCaret environment, comparing multiple models, and interpreting the results.

3.2.1 Setting Up PyCaret

The PyCaret classification module was initialized with the preprocessed data. The "survived" column was designated as the target variable.

3.2.2 Model Training and Evaluation

PyCaret was used to compare various classification models and select the best-performing model based on accuracy and other performance metrics.

We set the data for binary classification with our data frame (df) and the target column is “Survived”. Now, we run all the PyCaret model, fit the dataset, and compare all the models. All of the other processing (k-fold, train_test_split) is automatically done by PyCaret with default value.

3.3 Feature Importance

3.3.1 Feature Importance Analysis

After identifying the best model, PyCaret's feature importance function was used to analyze the significance of each feature in predicting the target variable.

3.3.2 Model Interpretation and Validation

The selected model was further interpreted to understand the contribution of individual features. Cross-validation was performed to ensure the model's robustness.

The methodology outlined above demonstrates the use of PyCaret for efficient feature importance analysis and binary classification. By leveraging PyCaret's low-code interface, the workflow from data preprocessing to model evaluation and interpretation was streamlined, allowing for rapid experimentation and insight generation.

4. EXPERIMENT AND RESULT

In this section, we present the results of the binary classification analysis and feature importance evaluation using the PyCaret library on the Titanic dataset.

4.1 Model Performance

After comparing multiple classification models using PyCaret, the Gradient Boosting Classifier emerged as the best-performing model for this dataset. The model achieved an average accuracy of 81.52%, indicating a high level of predictive performance. PyCaret's comprehensive evaluation metrics further corroborate the model's effectiveness.

These metrics suggest that the Gradient Boosting Classifier not only accurately distinguishes between survivors and non-survivors but also maintains a good balance between precision and recall, as evidenced by the F1 score. The classification result can be seen in figure 1.

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
gbc	Gradient Boosting Classifier	0.8152	0.8714	0.6960	0.7860	0.7345	0.5945	0.5998	0.0120
lr	Logistic Regression	0.8072	0.8533	0.6967	0.7746	0.7284	0.5805	0.5866	0.0130
lightgbm	Light Gradient Boosting Machine	0.8056	0.8602	0.7002	0.7681	0.7285	0.5779	0.5832	0.0590
ridge	Ridge Classifier	0.7992	0.0000	0.6924	0.7561	0.7206	0.5647	0.5679	0.0020
lda	Linear Discriminant Analysis	0.7992	0.8459	0.6924	0.7561	0.7206	0.5647	0.5679	0.0040
ada	Ada Boost Classifier	0.7991	0.8504	0.7304	0.7376	0.7300	0.5706	0.5743	0.0120
rf	Random Forest Classifier	0.7975	0.8509	0.6877	0.7648	0.7114	0.5584	0.5690	0.0290
et	Extra Trees Classifier	0.7862	0.8177	0.6920	0.7311	0.7056	0.5390	0.5438	0.0260
dt	Decision Tree Classifier	0.7637	0.7434	0.6614	0.6959	0.6745	0.4897	0.4931	0.0020
knn	K Neighbors Classifier	0.7122	0.7249	0.5330	0.6350	0.5727	0.3613	0.3672	0.0050
svm	SVM - Linear Kernel	0.6528	0.0000	0.2855	0.5852	0.3574	0.1800	0.2122	0.0030
nb	Naive Bayes	0.4067	0.7940	0.9786	0.3864	0.5538	0.0317	0.0929	0.0030
qda	Quadratic Discriminant Analysis	0.3762	0.0000	1.0000	0.3762	0.5467	0.0000	0.0000	0.0030

Figure 1. Model Performance Result

4.2 Feature Importance Analysis

Feature importance analysis was conducted to identify which features had the most significant impact on the model's predictions. The analysis revealed the following key insights:

Gender (Sex): The most influential feature in determining survival. The model indicates that gender (with women having a higher survival rate) is the primary predictor of survival.

Fare: The second most important feature, suggesting that passengers who paid higher fares had a higher likelihood of survival.

Age: The third significant feature, implying that younger passengers had a better chance of survival compared to older ones.

The feature importance plot generated by PyCaret visually illustrates these findings, highlighting the relative importance of each feature used in the model. The feature importance list can be seen in figure 2.

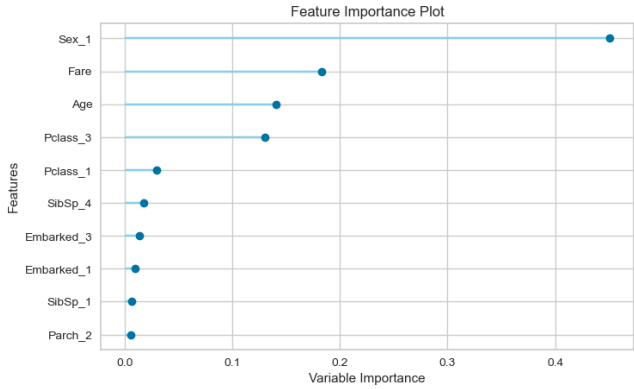

Figure 2. Feature Importance

The results demonstrate that the Gradient Boosting Classifier is highly effective in predicting the survival of passengers on the Titanic dataset, with an average accuracy of 81.52%. The feature importance analysis provided valuable insights, indicating that gender, fare, and age are the most critical factors influencing survival outcomes. These findings align with historical accounts and provide a robust basis for further research and analysis using similar methodologies.

5. CONCLUSION AND FUTURE WORK

5.1 Conclusion

This study demonstrates the application of PyCaret for feature importance analysis and binary classification on the Titanic dataset. By leveraging PyCaret's low-code environment, we efficiently preprocessed the data, trained various classification models, and identified the Gradient Boosting Classifier as the best-performing model with an average accuracy of 81.52%. The feature importance analysis revealed that gender (sex), fare, and age are the most critical features in predicting survival, providing insights that are consistent with historical accounts of the Titanic disaster.

The results highlight the utility of PyCaret in simplifying the machine learning workflow, enabling rapid experimentation and robust model evaluation. The findings emphasize the importance of feature selection and the effectiveness of gradient boosting techniques in binary classification tasks.

5.2 Future Work

While the current study provides valuable insights, several avenues for future work could enhance and expand upon these findings.

The binary classification and feature importance can be useful for determining the most optimum feature from big data or massive dataset. The result of the feature importance can become the baseline for implementing more advance algorithm such as deep learning.

REFERENCES

[1] Ahmad F. N, Lailatul Q. Z, and Saidah S. (2022). Determining the Best Email and Human Behavior Features on Phishing Email Classification International Journal of Advanced Computer Science and Applications (IJACSA), 13(8), http://dx.doi.org/10.14569/IJACSA.2022.0130821

[2] Ali M. "Introduction to Regression in Python with PyCaret". Accessed December 12, 2021 from https://towardsdatascience.com/introduction to-regression-in-python-with-pycaret-d6150b540fc4

[3] Ali M., "PyCaret," PyCaret: An open source, low-code machine learning library in Python, 2020. Available: https://www.pycaret.org

[4] Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In Advances in Neural Information Processing Systems (pp. 4765-4774).

[5] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794).

[6] McKinney, W. (2010). Data Structures for Statistical Computing in Python. Proceedings of the 9th Python in Science Conference, 51-56.

[7] Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.

[8] Fawcett, T. (2006). An introduction to ROC analysis. Pattern Recognition Letters, 27(8), 861-874. doi:10.1016/j.patrec.2005.10.010

[9] Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5-32.

[10] Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29(5), 1189-1232. doi:10.1214/aos/1013203451