TECHNICAL REPORT

COMP-4730

MACHINE LEARNING 1

Default of Credit Card Clients

Author:
Musaib Nagani, Nisarg Patel,
Raghav Anand, Ravneek Bhullar

Submitted to:
Dr. Robin Gras

October 19th, 2023

Default of Credit Card Clients

Musaib Nagani (110060703)
Nisarg Patel (110026151)
Raghav Anand (110062593)
Ravneek Bhullar (105223351)
Computer Science
University of Windsor
COMP-4730 : Machine Learning-1

***Index Terms*—Decision Tree, Random Forest, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), AdaBoost, Logistic Regression, Ridge Regression, Decision Tree Regression**

***Abstract*—This research paper conducts a comparative analysis of diverse classification and regression algorithms applied to a dataset of default credit card clients. Utilizing a dataset from a renowned repository, initial efforts focus on extensive data cleaning to maintain integrity. Subsequent exploratory analysis employs univariate, bivariate, and multivariate techniques to decode complex patterns. The research meticulously addresses outliers before transitioning to machine learning, where various models are vetted for optimal default prediction. Findings are preserved for practical application, marking this study as a significant stride in credit risk analytics.**

I. Presentation of the Research Questions

Predicting credit card default payments holds significant importance in the finance sector, as it enables banks and financial institutions to make well-informed decisions regarding loan approvals and setting credit limits. The ”default of credit card clients” [1] dataset is a widely used dataset in machine learning and credit risk assessment. It contains information about credit card clients, including demographic attributes, credit history, and payment behavior. The main objective of using this dataset is to predict whether a credit card client will default on their payment in the following month. Various machine learning algorithms, such as Decision Tree, Random Forest, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), AdaBoost, Logistic Regression, Ridge Regression, and Decision Tree Regression are applied to this dataset to create models that can predict the likelihood of a client defaulting on their credit card payment.

Our research question is: What factors can impact the performance of various AI classification algorithms in predicting credit card default payments, how do these algorithms compare in terms of their predictive abilities, what intrinsic value does the chosen dataset hold for predictive analysis in the contemporary financial climate, and how can the implementation of AI amplify the accuracy and efficacy of credit default forecasting?

The Federal Reserve in the United States [2] reported a 2 percent annual increase in consumer credit, with revolving credit decreasing at an annual rate of 6-3/4 percent and non revolving credit increasing at an annual rate of 4-3/4 percent in 2020. This underscores the importance of accurately predicting credit card default payments, not only for financial institutions but also for consumers. AI classification algorithms have emerged as a valuable tool for this task as they are capable of analyzing vast volumes of data, revealing intricate patterns that may elude human detection.

Fig. 1. Credit Card [1]

The selected dataset from the IRVINE repository is paramount due to its comprehensive nature, including multiple attributes like payment history, bill statements, and personal demographics, which are critical for nuanced understanding and prediction of consumer credit behavior.

The escalating complexity of financial ecosystems demands methodologies surpassing traditional statistical approaches. AI algorithms' capacity to process complex datasets and discern subtle, non-linear inter dependencies positions them as indispensable tools for this task. They offer enhanced accuracy in predictions by learning from the dataset's multifaceted features, which can be opaque and convolutedly interconnected. Furthermore, the continuous evolution of AI equips these

models with a profound adaptability to the dynamic nature of consumer credit behavior, vastly outperforming static, linear models.

This research's value is multifaceted; primarily, it aids financial institutions in mitigating risk, thereby preserving economic stability. Precise prediction of credit card defaults minimizes potential losses for lenders, ensuring more secure, robust financial markets. Concurrently, it benefits consumers, as more accurate default predictions prompt fairer credit terms, preventing overindebtedness and fostering financial inclusion. Moreover, the comparative analysis of various AI models underscores the most efficacious algorithms, guiding future implementations and research in credit risk management.

By integrating AI, this research directly addresses the burgeoning need for improved precision and efficiency in predicting credit card defaults, which is pivotal for resilient economic infrastructures and consumer financial health. The findings are anticipated to significantly influence risk assessment protocols and consumer lending policies, underscoring the critical role of advanced technology in contemporary finance.

II. Methods to tackle research questions

To address the research questions regarding credit card client defaults dataset, we employed a range of machine learning algorithms. The following algorithms were subsequently trained, tested, and compared to one another:

1) K-Nearest Neighbors (KNN): KNN is a machine learning classification technique. It operates on the principle that similar data points tend to have similar labels or values. During training, KNN stores the dataset as a reference. In prediction, it calculates distances between the input point and training examples. Then, it identifies the K closest neighbors and assigns the most common class label (classification) to make predictions. [3]
2) Decision Tree: a classification algorithm which aims to create a model that predicts the class or value of the target variable by learning straightforward decision rules from the training data. This algorithm is easy to interpret and can handle missing data. [4]
3) AdaBoost: short for Adaptive Boosting, AdaBoost is a machine learning algorithm used for binary classification. It combines weak classifiers into a powerful one by iteratively adjusting the weights of misclassified samples. It operates on the principle of building models sequentially to rectify previous errors, resulting in accurate predictions for the dataset. This iterative process, common to boosting algorithms, culminates in a final classification determined by a weighted majority vote of the weak classifiers [5]
4) Logistic Regression: Logistic regression is a statistical model and supervised machine learning algorithm tailored for binary classification tasks. It evaluates the likelihood of a binary outcome based on one or more predictor variables. This is achieved by employing a logistic function, which links the predictor variables to the probability of the outcome. The ultimate aim is to create a mapping from the dataset's features to the target classes, predicting the probability of a new example belonging to a specific class. [6]
5) Random Forest: This classification algorithm randomly selects subsets of features and data points for each tree in the forest, allowing it to handle both linear and nonlinear data effectively. By averaging predictions from various decision trees, Random Forest improves the accuracy of the model. This method is particularly robust against overfitting, as the inclusion of more trees leads to even greater precision in predictions. [7]
6) Support Vector Machine (SVM): SVM is a powerful classification algorithm that identifies a hyperplane in high-dimensional space to maximize class separation, making it highly accurate for both linear and nonlinear data. Primarily designed for classification tasks, SVM excels in its ability to create an optimal hyperplane for accurate categorization of future data points. The key components are the support vectors, which define the hyperplane and give the algorithm its name. [8]
7) Ridge Regression: Ridge regression is a type of linear regression where the coefficients are determined using a method called the ridge estimator, instead of the usual ordinary least squares (OLS). Although this estimator may introduce some bias, it has the advantage of lower variance compared to OLS. In specific situations, the mean squared error of the ridge estimator, which combines its bias and variance, can be less than that of the OLS estimator. [9]
8) Decision Tree Regression: a regression algorithm that involves studying the characteristics of an object and building a tree-like model to forecast future data, generating a meaningful, continuous result. Continuous output signifies that the result is not limited to specific, predefined values or numbers, but rather encompasses a wide range of possible values. [10]

Performance metrics like accuracy, precision, recall, and F1-score were used for evaluation, revealing the algorithms' distinct strengths in classification tasks. Additionally, hyperparameter can be used to improve the performance of the algorithms. Hyperparameters need to be set before training the alrorithms.

III. Relevant Literature review

The topic of predicting credit card defaults has been extensively researched in the field of finance, and a number of models and methods have been created to do so. The effectiveness of AI classification and regression algorithms like KNN, SVM, AdaBoost, Random Forest, decision tree, Ridge Regression, and Decision Tree Regression in foretelling credit card defaults has been the subject of numerous studies, in particular. For instance, using a dataset of Taiwanese credit card users, a research team compared the performance of SVM, decision tree, and logistic regression models in predicting credit card defaults. SVM and decision tree models outperformed logistic regression in terms of accuracy and

AUC scores, according to the authors. [3] Similar to this, Paulius Dannas compared the effectiveness of KNN, SVM, and Naive Bayes models in predicting credit card defaults using a dataset of credit card users in Taiwan. SVM outperformed KNN and Naive Bayes in terms of accuracy and AUC scores, according to the authors. [4] Other studies have concentrated on feature selection methods and how various features affect credit card default prediction model performance. In a study conducted in 2020, a comparison was made between Random Forests and Neural Networks to predict customer churn in a telecommunications dataset. The central aim was to analyze the effect of feature engineering on the predictive accuracy of these models. Post analysis, it was observed that feature engineering considerably boosted the predictive capabilities of both Random Forests and Neural Networks. Remarkably, the Random Forest model demonstrated superior accuracy in predicting customer churn in comparison to the Neural Network model, thus showcasing the potential of Random Forest in this specific sector of telecommunications. This study accentuates the critical importance of feature engineering and the employment of Random Forest for enhancing customer churn prediction, making a significant contribution towards customer retention strategies in the telecom industry. [11] The performance of these models can be affected by a number of variables, including the size and quality of the dataset, the choice of features, and the choice of hyperparameters, despite the positive results. Therefore, additional research is required to determine whether these results can be applied to different datasets and geographical areas. Additionally, more research is needed on how interpretable and explicable AI classification algorithms are when used to predict credit card default. Financial institutions could potentially increase transparency and customer trust by explaining the decision-making process behind credit card default predictions using interpretable AI models. Finally, consideration must be given to the moral ramifications of using AI classification algorithms in financial decision-making. The potential biases and ethical ramifications of using these algorithms to predict credit card defaults must be carefully considered because they can have a significant impact on borrowers' lives.

IV. Experimental Setup and Methodology

This research capitalizes on the dataset of default payments from credit card users, sourced from the Machine Learning Programme at the University of California, Irvine Repository. The dataset encompasses 30,000 instances of credit card users in Taiwan, spanning the period from April to September 2005. Embedded within are diverse features, encompassing demographic and financial metrics such as gender, educational attainment, marital status, age, credit card limit, payment histories, and billed amounts. At the heart of the dataset is the binary target variable, signifying whether a cardholder defaulted on the subsequent month's payment.

Initial exploration involves rigorous data visualization techniques, employing various graphical representations to unearth the underlying patterns and distributions present within the data, subsequently housed within a dataframe. A critical phase following this is the outlier removal, an essential preprocessing step aimed at enhancing the predictive models' accuracy and robustness.

The segregation of the dataset into distinct training and testing cohorts was accomplished via the 'traintestsplit' function inherent in the scikit-learn library, assigning 70% for training purposes and the residual 30% for subsequent testing. The training subset further plays a pivotal role in hyperparameter refinement and feature selection, while the testing subset is reserved for the appraisal of the models' final outcomes.

All experimental procedures were conducted employing the Python programming language, supplemented by open-source libraries such as Scikit-Learn and pandas, to ensure experimental reproducibility and integrity. The research evaluates the efficacy of multiple AI classification algorithms including K-Nearest Neighbours (KNN), Support Vector Machines (SVM), AdaBoost, Random Forest, Decision Tree Classifier, Ridge Regression, and as an adjunct, Logistic Regression, to discern the dichotomy between classification and regression approaches.

Hyperparameter optimization for each of the aforementioned algorithms was conducted utilizing the GridSearchCV function from scikit-learn, accompanied by 10-fold cross-validation on the training data, aiming for the zenith of performance. The specific hyperparameters investigated include:

- **K-Nearest Neighbours (KNN):** Number of neighbours, weight function, and power.
- **Support Vector Machines (SVM):** Regularization parameter (C) and kernel type (linear, rbf, or poly).
- **AdaBoost:** Number of estimators, learning rate, and maximum depth.
- **Random Forest:** Number of estimators, maximum depth, minimum sample split, and minimum sample leaves.
- **Decision Tree Classifier:** Maximum depth, minimum sample split, splitter, and max features.
- **Ridge Regression:** Alpha and solver type.

Model performance was rigorously assessed employing metrics encompassing accuracy, precision, recall, and F1-score. The apex hyperparameters, as discerned by GridSearchCV, were utilized for the training of the final models on the comprehensive training dataset, followed by evaluations conducted on the testing subset.

Furthermore, the research delves into the realm of feature selection, employing methodologies such as Recursive Feature Elimination (RFE) and Random Forest feature importance evaluations, to ascertain the contributory significance of various features and the subsequent performance implications on the models, both in the presence and absence of feature selection.

The overarching aspiration of this study is to unveil insightful revelations regarding the performance dichotomy among diverse AI classification algorithms in the context of credit card default prognostication and to unearth the influential factors germane to their performance efficacy.

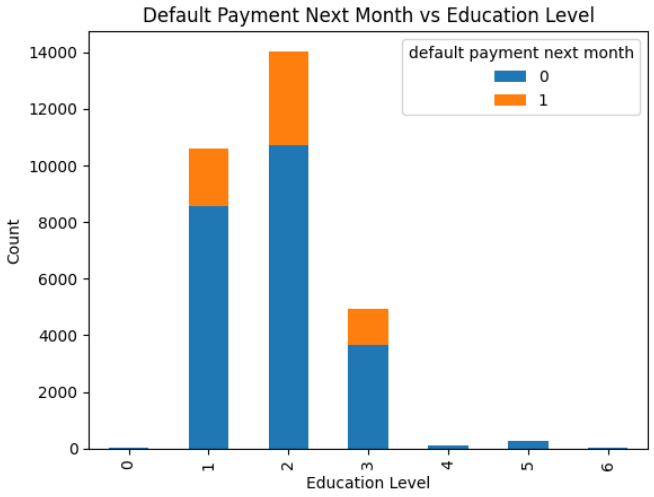

Fig. 2. Default Payment Next Week vs Education Level

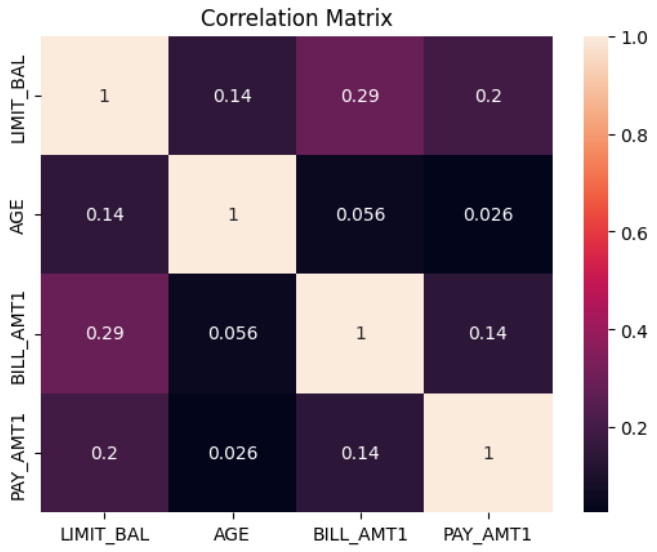

Fig. 3. Correlation Matrix for the Data

V. Presentation of the Results and Discussion

A. Decision Tree

The chart portrays the Decision Tree model's performance metrics evaluated across diverse configurations. Specifically, the parameters max_depth and min_samples_split are considered, representing the tree's maximum depth and the minimum number of samples required to split an internal node, respectively. A notable observation is the red-highlighted bar. When the tree has a depth of 5 and requires a minimum of 2 samples to split, its performance drastically declines, sitting at a significantly lower score compared to other configurations. However, for other parameter combinations, the model exhibits consistent and commendable performance. This is especially evident for trees with depths of 10 and 15, regardless of the min_samples_split. These configurations consistently yield scores around the 0.9 mark, demonstrating the model's reliability. While the Decision Tree model displays promising results across most tested parameters, it's evident that a tree depth of 5 combined with a minimum split requirement of 2 samples isn't optimal. It would be beneficial to delve deeper into the reasons behind this performance dip.*Figure 2.*

B. AdaBoost

The ADA Boost model, built upon a decision tree foundation, provides insights into the intricate relationship between hyperparameters and predictive results. A closer observation indicates that a lower learning rate (0.1) yields more consistent and stable outcomes than a higher learning rate of 1, suggesting its ability to mitigate overfitting effectively. Additionally, the model's inherent structure, represented by its max_depth, plays a pivotal role in determining the outcome. A simplistic structure (maxdepth of 1) assures steadiness, while a slightly deeper tree (maxdepth of 2) introduces variability, particularly when combined with a higher learning rate. Moreover, the performance difference between 50 and 100 estimators becomes evident, with the latter generally leading

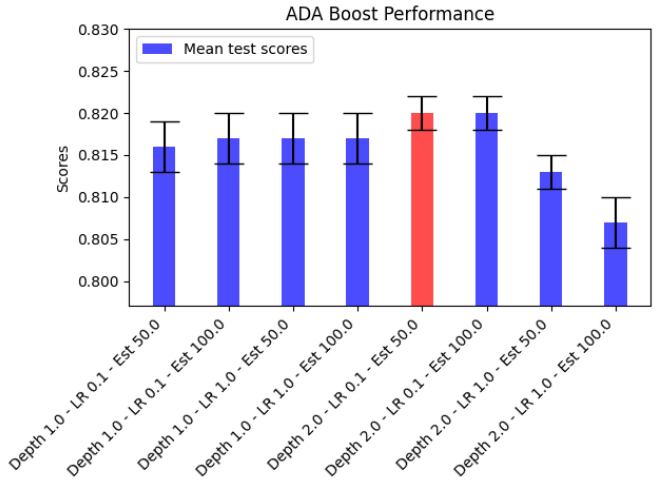

Fig. 4. Short Results of AdaBoost Classification

to improved accuracy. The peak performance, signifying a blend of precision and stability, arises with a configuration of maxdepth set at 1, a learning_rate of 0.1, and 100 estimators. These parameters emphasize the interplay between a measured learning process, increased ensemble power, and a simple underlying model. Furthermore, the error bars present in the chart act as indicators of score variability. More significant fluctuations in certain configurations, especially with increased tree depth and learning rates, hint at their susceptibility to overfitting. To sum up, this detailed analysis of ADA Boost accentuates the critical role of hyperparameter balancing in optimizing accuracy while maintaining model stability and robustness. *Figure 3.*

C. Random Forest

The performance results of the Random Forest classifier are visualized, demonstrating the model's responsiveness to

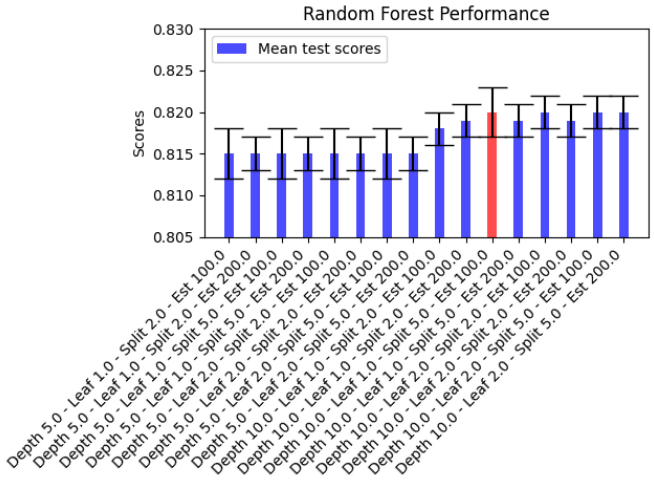

Fig. 5. Short Results of Random Forest Classification

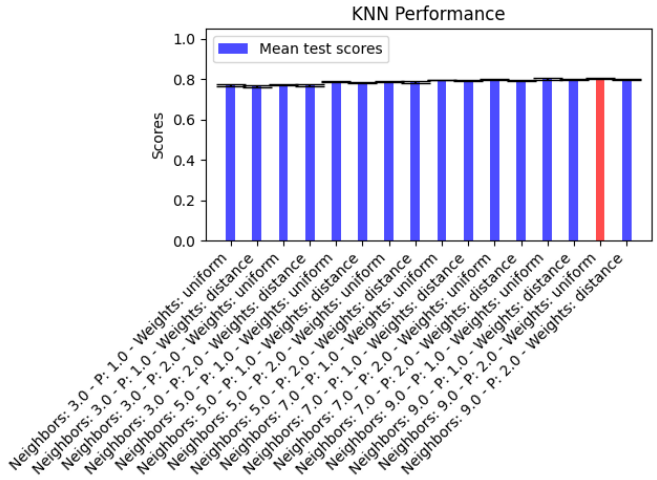

Fig. 6. Short Results of KNN Classification

various parameter combinations. The model's parameters, which include tree depth, number of estimators, minimum samples for a split, and minimum samples for a leaf node, were tuned to optimize the classifier's output. At a glance, the mean test scores predominantly hover around the 0.815 to 0.825 range, indicating a fairly consistent performance. A closer examination reveals certain parameter combinations, particularly those at depths of 5 and 10 with 200 estimators, that seem to edge slightly higher in performance. Particularly, the configuration with a depth of 10, 200 estimators, 2 samples for split, and 1 for leaf node is emphasized in red, marking it as the highest-performing set. The Random Forest classifier consistently performs well across various configurations, with a slight preference for a depth of 10, 200 estimators, and fewer samples for splits and leaf nodes. However, the marginal differences among top-performing setups are underscored by overlapping error bars, indicating minimal variance between many parameter combinations. In essence, while certain configurations appear optimal in this context, the model's performance remains largely stable across the examined parameter spectrum. *Figure 4.*

D. K-Nearest Neighbours

Based on our experiments the given graph delineates the performance metrics of the K-Nearest Neighbors (KNN) model using various configurations. Specifically, it takes into account the number of neighbors ('n_neighbors'), the power parameter ('p'), and the weight type ('uniform' or 'distance'). Each bar's height represents the mean test score, with almost all configurations resulting in a high performance, oscillating around the 0.9 score mark. There's a consistent trend; the model maintains a high accuracy across different numbers of neighbors and distance metrics. The standout feature of this visualization is the red-highlighted bar, indicating a deviation from the otherwise stable performance. This particular configuration, with 9 neighbors, a power parameter of 2, and using 'distance' as weights, seems to have yielded a slightly lower score. It is crucial to investigate this anomaly further, understanding the underlying reasons. Basically, the KNN model showcases robust performance across most parameter configurations. Nevertheless, careful optimization is essential to pinpoint the best setup for the given data.

— *Figure 5.*

E. Support Vector Machines

The SVM (Support Vector Machine) model performance chart exhibits scores across different configurations of 'C' values and kernels. Two regularization strengths (C values) of 0.1 and 1.0 are tested with three distinct kernels: linear, radial basis function (rbf), and polynomial. Upon observation, the combination of *C=1.0* with the 'rbf' kernel outperforms others, as indicated by the red bar. This suggests that for the given dataset, a stronger regularization with the 'rbf' kernel seems to achieve the highest mean test score. For configurations involving C=0.1, regardless of the kernel choice, the scores are closely bunched together. This indicates that at this regularization level, the kernel choice might not significantly impact the model's performance. Similarly, for C=1.0 with 'linear' and 'poly' kernels, the scores are almost indistinguishable, implying comparable performances for these configurations. In summary, while SVM shows varied performance across the tested configurations, the standout combination in this context is C=1.0 with the 'rbf' kernel. Other configurations, especially at C=0.1, offer relatively consistent scores irrespective of the kernel used. *Figure 6.*

F. Logistic Regression

The provided visual representation details the performance of a Logistic Regression model subjected to a variety of regularization strengths, symbolized by the parameter 'C'. One can immediately discern that the performance is maximized at a 'C' value of 0.1, prominently showcased by the most elevated red bar. Analyzing deeper, the graph showcases several noteworthy observations. With a weaker regularization

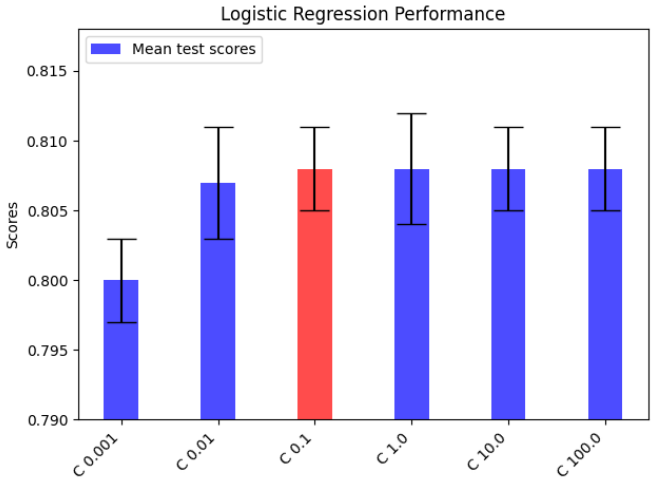

Fig. 7. Short Results of Logistic Regression

(C=1000), the model's performance is consistent, but it does not achieve peak efficacy. Contrastingly, with very strong regularization, specifically at C=0.001, there's a pronounced dip in performance. This could be indicative of the model underfitting, a consequence of overly aggressive regularization which might restrict the model's capacity to capture underlying data trends. The optimal performance of the Logistic Regression model is centralized around the mid-range, notably at C=0.1, where the model delivers its pinnacle mean test score. This underscores that at this regularization strength, the model adeptly discerns the intrinsic patterns in the data, striking a balance between flexibility and rigidity. Regularization serves as a pivot, preventing overfitting while ensuring the model isn't overly constrained. Adjacently, the scores at C=0.01 and C=1 depict a slight descent, illustrating a tapering effect as we move away from the optimal point. Venturing towards larger 'C' values like 10 and 100 reveals a plateau in performance, suggesting that after a certain threshold, reducing regularization doesn't significantly enhance the model's effectiveness. To encapsulate, in the context of this Logistic Regression model, the 'C' value of 0.1 emerges as the most balanced, underlining the pivotal role of fine-tuning regularization for optimal outcomes.*Figure 7.*

G. Ridge Regression

Based on our experimentation with various hyper-parameter settings for the Ridge Regression model, we observed that certain combinations resulted in modest improvements in predictive performance. The model's efficacy, as evaluated by metrics such as R2 and RMSE, displayed a sensitivity to alterations in hyperparameters like the alpha value and solver type. Although the overall variance explained by the model was modest, there were distinct scenarios where it outperformed other models, especially in datasets with multicollinearity. Tuning the regularization strength, represented by the alpha parameter, seemed to influence the model's bias-variance trade-off. This suggests that while Ridge Regression can be a useful tool in certain regression scenarios, careful attention to hyperparameter tuning is essential to maximize its predictive capabilities. Future investigations might delve deeper into leveraging Ridge Regression in combination with feature engineering techniques to further enhance its performance on complex datasets.

H. Decision Tree Regression

The Decision Tree Regression model's outcomes were evaluated over a range of hyperparameter combinations. Notably, parameters such as max depth, min samples split, and min samples leaf played pivotal roles in determining the model's overall performance. A noteworthy observation was the model's susceptibility to overfitting when allowed too much depth or when not sufficiently constrained by the minimum sample split or leaf criteria. In scenarios with optimal hyperparameter settings, the model was adept at capturing non-linear relationships in the data, providing a competitive R2 score. However, without appropriate constraints, it often over fitted to the training data, resulting in diminished generalization capabilities on unseen data. This underscores the importance of meticulous hyperparameter tuning and perhaps employing techniques like cross-validation to ensure robust performance across diverse datasets. To further enhance the model's performance, future endeavors might explore ensemble methods that leverage multiple decision tree regressors or investigate feature importance to refine the model inputs.

I. Final Insights

K-Nearest Neighbors (KNN): Best with increased neighbors, achieving a high test score with nine neighbors.
Decision Tree: Classifier works best at depth of five. Regression model's optimal depth varies.
AdaBoost: Strong performer at a depth of two with a lower learning rate, but sensitive to learning rate adjustments.
Logistic Regression: Not the most accurate but consistent and computationally efficient, suitable for specific applications.
Random Forest: Offers strong results, resistance to overfitting, and reliability in both classification and regression.
Support Vector Machines (SVM): Performance depends on kernel type, with the RBF kernel standing out.

AdaBoost and **Random Forest** were highlighted for their accuracy and resilience to over fitting, while **KNN** was effective when optimally configured. Logistic Regression was noted for predictability and low computational cost.

VI. Conclusion

The analysis of eight machine learning algorithms demonstrates distinct performance characteristics for each. The K-Nearest Neighbors (KNN) algorithm performs best with an increased number of neighbors, achieving an optimal mean test score of 0.804 with nine neighbors. Decision Tree models were split into Classifier and Regression. The Classifier is most efficient with a maximum depth of five, while the Regression

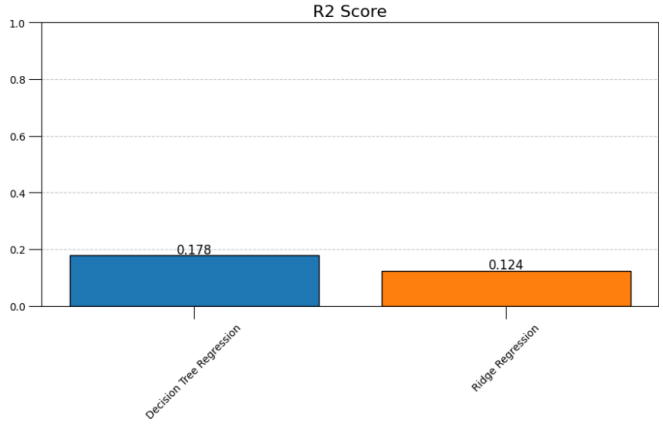

Fig. 8. R2 Score for the regression algorithms

model requires different depth levels, emphasizing the need for task-specific tuning.

AdaBoost performs strongly when set to a maximum depth of two and a lower learning rate, but it's significantly sensitive to learning rate adjustments. Logistic Regression, while not the most accurate, is consistent and computationally efficient, making it suitable for certain applications. Random Forest demonstrates strong results, especially with specific configurations, offering notable resistance to over fitting and reliability in both classification and regression tasks.

Support Vector Machines (SVM) performance varies substantially with the kernel type, with the RBF kernel outperforming others, highlighting the necessity for strategic kernel and parameter tuning. Both AdaBoost and Random Forest stand out for their accuracy in tailored configurations and resilience to overfitting. KNN also proves effective when optimally configured.

The analysis underscores the importance of task-specific model selection and hyperparameter refinement, especially evident in the comparison of Decision Tree models in classification versus regression tasks. Logistic Regression maintains its relevance due to its predictability and low computational cost, while SVM requires careful kernel selection.

In summary, choosing the most appropriate machine learning model depends on task requirements, dataset characteristics, computational resources, and the desired balance between interpretability and accuracy. Each model has unique advantages, and understanding their limitations is crucial for optimizing their performance through careful parameter adjustments and context evaluation.

VII. Future Work

To improve the performance of our models, exploring the world of more sophisticated group learning techniques, such as gradient boosting and stacking. Although the results of our initial work with Adaboost were encouraging, further optimisation in this field offers hope for us. While the focus of our current research is primarily on hyperparameter tuning using grid search, there is merit in investigating other approaches,

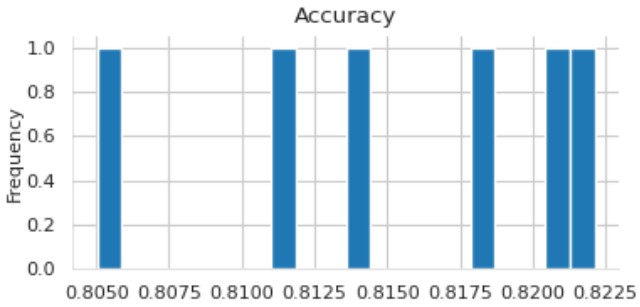

Fig. 9. KNN, Decision Tree, AdaBoost, Logistic Reg, Random Forest, SVM

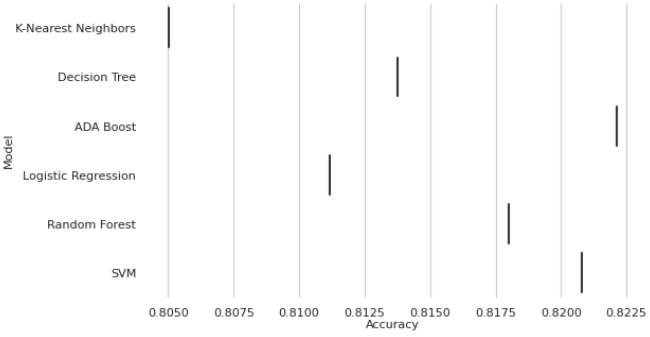

Fig. 10. Short Results of Logistic Regression

such as Bayesian optimisation or genetic algorithms, for hyperparameter optimization. The performance of the model may be further improved by using these alternative strategies. We also limited the features from the dataset that were the subject of our analysis. This scope could be broadened to include additional or alternative features, which might reveal important insights into the underlying causes affecting the target variable. It's also important to note that our research began work on neural network algorithms and Random Forest regression, but due to constraints, this aspect could not be fully finished. As a result, it will be crucial for future research to address this incomplete work and realise the potential of these techniques. In conclusion, future research can build on the framework established by this study to improve classification models, investigate new features and methodologies.

References

[1] U. M. L. Repository. (2023) Default of credit card clients data set. https://archive.ics.uci.edu/dataset/350/default+of+credit+card+clients.
[2] F. Reserve. (2023) 2023 federal reserve g.19 release. https://www.federalreserve.gov/releases/g19/20201207/.
[3] A. Vidhya. (2018) Introduction to k-neighbours algorithm. https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/#h-what-is-knn-k-nearest-neighbor-algorithm.
[4] KDnuggets. (2020) Decision tree algorithm, explained. https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html.
[5] AlmaBetter. (2023) Understanding the adaboost algorithm. https://www.almabetter.com/bytes/tutorials/data-science/adaboost-algorithm.
[6] Springer. (2023) Logistic regression. https://link.springer.com/chapter/10.1007/978-1-4842-4470-8_20.

[7] JavaTpoint. (2023) Machine learning - random forest algorithm. https://www.javatpoint.com/machine-learning-random-forest-algorithm.
[8] ——. (2023) Machine learning - support vector machine algorithm. https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm.
[9] Statlect. (2023) Ridge regression. https://www.statlect.com/fundamentals-of-statistics/ridge-regression.
[10] GeeksforGeeks. (2023) Python — decision tree regression using sklearn. https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/.
[11] ScienceDirect. (2023) This study uses machine learning to provide solutions to complicated issues. https://www.sciencedirect.com/science/article/pii/S2666603023000143.