Life Assurance Application Conversion Prediction using Supervised Machine Learning

Greg Langella

A Thesis Submitted in Partial Fulfilment

of the requirements for the

Degree of

Master of Science in Data Analytics

Diagram

Description automatically generated with medium confidence

September 2023

Supervisor: Sam Weiss

**Acknowledgements**

**Abstract**

This research project aims to develop predictive models for classifying the conversion of life assurance applications in to active policies. The research incorporates feature correlation analysis, feature importance evaluation, machine learning model evaluation, and hyperparameter tuning. The importance of features such as WorkflowStatus and Agency in conversion prediction is one of the key findings. The performance of the models vary, with Gradient Boosting and LightGBM standing out. These findings provide useful information for both the fields of data analytics and machine learning as well as the life assurance sector.

Contents

[1. Introduction 7](#_Toc145889802)

[1.1 Background 7](#_Toc145889803)

[1.2 Existing Research and Research Gaps 8](#_Toc145889804)

[1.3 Research Problem 9](#_Toc145889805)

[1.4 Research Objectives 10](#_Toc145889806)

[1.5 Scope 11](#_Toc145889807)

[2. Literature Review 13](#_Toc145889808)

[2.1 Machine Learning in Insurance 13](#_Toc145889809)

[2.2 Correlation Analysis 19](#_Toc145889810)

[2.3 Feature Importance 19](#_Toc145889811)

[2.4 Feature Selection 22](#_Toc145889812)

[2.5 Hyperparameter Optimisation 24](#_Toc145889813)

[2.6 Experimental Design 26](#_Toc145889814)

[2.7 Validation 28](#_Toc145889815)

[2.7 Model Evaluation Metrics 33](#_Toc145889816)

[2.8 Conclusions 34](#_Toc145889817)

[3. Methodology 36](#_Toc145889818)

[3.1 Sampling Strategy 36](#_Toc145889819)

[3.2 Primary Research Methodology 36](#_Toc145889820)

[3.3 Data Collection 37](#_Toc145889821)

[3.4 Data Cleansing 37](#_Toc145889822)

[3.5 Data Exploration 38](#_Toc145889823)

[3.6 Feature Correlation & Feature Importance 39](#_Toc145889824)

[3.7 Machine Learning 42](#_Toc145889825)

[3.8 Performance Validation 43](#_Toc145889826)

[4. Results 44](#_Toc145889827)

[4.1 Feature Correlation & Feature Importance 44](#_Toc145889828)

[4.1.1 Imbalanced Training Data 44](#_Toc145889829)

[4.1.1.1 Association Analysis 44](#_Toc145889830)

[4.1.1.2 Random Forest 46](#_Toc145889831)

[4.1.1.3 Gradient Boosting Classifier 49](#_Toc145889832)

[4.1.1.4 LightGBM Classifier 53](#_Toc145889833)

[4.1.1.4 Class Aware Feature Importance 57](#_Toc145889834)

[4.1.1.5 RFECV 58](#_Toc145889835)

[4.1.1.6 Compare with and without Feature 60](#_Toc145889836)

[4.1.2 Oversampled Training Data 63](#_Toc145889837)

[4.1.2.1 Association Analysis 63](#_Toc145889838)

[4.1.2.2 Random Forest 66](#_Toc145889839)

[4.1.2.3 Gradient Boosting Classifier 69](#_Toc145889840)

[4.1.2.4 LightGBM Classifier 71](#_Toc145889841)

[4.1.2.5 Class Aware Feature Importance 74](#_Toc145889842)

[4.1.2.6 RFECV 75](#_Toc145889843)

[4.2 Machine Learning Models & Hyperparameter Tuning 78](#_Toc145889844)

[4.2.1 Imbalanced Training Data 78](#_Toc145889845)

[4.2.1.1 All Features 78](#_Toc145889846)

[4.2.1.1.1 Logistic Regression 78](#_Toc145889847)

[4.2.1.1.2 Decision Tree 78](#_Toc145889848)

[4.1.1.1.3 Random Forest 79](#_Toc145889849)

[4.1.1.1.4 Gradient Boosting Classifier 80](#_Toc145889850)

[4.1.1.1.5 LightGBM 81](#_Toc145889851)

[4.1.1.1.6 Model Validation 81](#_Toc145889852)

[4.1.1.1.7 Evaluation on Unseen Data 82](#_Toc145889853)

[4.2.1.2 Selected Features 84](#_Toc145889854)

[4.2.1.2.1 Logistic Regression 84](#_Toc145889855)

[4.2.1.2.2 Decision Tree 85](#_Toc145889856)

[4.2.1.2.3 Random Forest 85](#_Toc145889857)

[4.2.1.2.4 Gradient Boosting Classifier 86](#_Toc145889858)

[4.2.1.2.5 Light GBM 87](#_Toc145889859)

[4.2.1.2.6 Model Validation 88](#_Toc145889860)

[4.2.1.2.7 Evaluation on Unseen Data 88](#_Toc145889861)

[4.2.2 Oversampled Dataset 90](#_Toc145889862)

[4.2.2.1 All Features 90](#_Toc145889863)

[4.2.2.1.1 Logistic Regression 90](#_Toc145889864)

[4.2.2.1.2 Decision Tree 91](#_Toc145889865)

[4.2.2.1.3 Random Forest 91](#_Toc145889866)

[4.2.2.1.4 Gradient Boosting Classifier 92](#_Toc145889867)

[4.2.2.1.5 Light GBM 93](#_Toc145889868)

[4.2.2.1.6 Model Validation 94](#_Toc145889869)

[4.2.2.1.7 Evaluation on Unseen Data 95](#_Toc145889870)

[4.2.2.2 Selected Features 97](#_Toc145889871)

[4.2.2.2.1 Logistic Regression 97](#_Toc145889872)

[4.2.2.2.2 Decision Tree 97](#_Toc145889873)

[4.2.2.2.3 Random Forest 98](#_Toc145889874)

[4.2.2.2.4 Gradient Boosting Classifier 99](#_Toc145889875)

[4.2.2.2.5 Light GBM 99](#_Toc145889876)

[4.2.2.2.6 Model Validation 100](#_Toc145889877)

[4.2.2.2.7 Evaluation on Unseen Data 101](#_Toc145889878)

[5. Discussion 104](#_Toc145889879)

[5.1 Feature Correlation And Feature Importance 104](#_Toc145889880)

[5.1.1 Imbalanced Dataset 104](#_Toc145889881)

[5.1.2 Oversampled Dataset 105](#_Toc145889882)

[5.2 Machine Learning Models & Hyperparameter Tuning 109](#_Toc145889883)

[5.2.1 Original Imbalanced Dataset 109](#_Toc145889884)

[5.2.1.1 All Features 109](#_Toc145889885)

[5.2.1.2 Selected Features 110](#_Toc145889886)

[5.2.2 Oversampled Dataset 112](#_Toc145889887)

[5.2.2.1 All Features 112](#_Toc145889888)

[5.2.2.2 Selected Features 113](#_Toc145889889)

[5.3 Future Research 118](#_Toc145889890)

[6. Conclusion 119](#_Toc145889891)

[References 121](#_Toc145889892)

1. **Introduction**
   1. **Background**

In today's ever-changing business world, the role of data analytics and data science has evolved from supplemental tools to vital assets for firms seeking to maintain a competitive advantage. The insurance and financial services sectors is a perfect example of an industry in which these disruptive technologies have not only found use, but have thrived.

A plethora of complex issues entice organisations in the life assurance market to continually adapt and innovate. One of the most serious of these difficulties is the requirement to optimally manage resources in order to maximise incoming business while carefully rationing operational expenses. This mission of efficiency and profitability is at the heart of a fine balancing act that life assurance businesses must execute on a daily basis.

To comprehend the scale of this challenge, one must first comprehend the complexities of the life assurance sector. It is a business where success depends on more than simply providing financial security. It is about building long-term relationships with policyholders, recognising their changing demands, and staying ahead of volatile market forces. As a result, allocating resources, both human and financial, becomes crucial.

Historically, the life assurance industry relied primarily on experience, intuition, and traditional actuarial procedures to make decisions. A critical shift has happened, however, as the volume, velocity, and diversity of data available to these organisations has expanded greatly. In a data-rich world, organisations have realised that they cannot afford to depend entirely on human judgment.

This awareness has resulted in widespread use of advanced data analytics and machine learning technologies. Organisations can make data-driven decisions using data analytics, allowing them to traverse the complicated and highly regulated world of life assurance with more speed and accuracy.

* 1. **Existing Research and Research Gaps**

In general, the use of supervised machine learning models in insurance has mostly focused on predicting client purchasing behaviour. While this research has provided useful information, there is still a gap in knowing whether a current life assurance application would eventually become an active policy. Furthermore, the amount of study and information about the factors that influence customers’ decisions to purchase non-life insurance exceeds that of life assurance. This study intends to utilise known best practices in machine learning within the insurance industry in order to bridge these gaps and broaden the scope of research in the life assurance domain. Correlation analysis, feature importance assessment, feature selection approaches, hyperparameter optimisation, experimental design, and model evaluation metrics are examples of these practices.

Existing research has identified feature selection and other data pre-processing techniques as having proven useful in addressing issues such as imbalanced class distributions and identifying characteristics that significantly influence the likelihood of insurance policy purchases. Furthermore, the Decision Tree algorithm and its derivatives, such as Random Forest, continue to be successful in predicting client purchase intentions, highlighting its interpretability and importance in finding the determinants of customer behaviour.

The existing research also highlights the effectiveness of machine learning algorithms in a variety of classification tasks in the insurance business, such as churn prediction, claim prediction, and fraud detection. Ensemble approaches, such as Random Forest and deep learning models, have showed promise in terms of accuracy and interpretability, moving predictive modelling in the insurance business ahead. Validation is becoming an increasingly important aspect of predictive modelling and machine learning. External validation, in addition to internal validation, is required to check the anticipated correctness and resilience of models. Rigorous validation approaches, as well as transparent reporting, are required for realistic comparative performance assessments and assuring predictive model generalisability and replicability.

While the existing body of research in the domain of insurance analytics and machine learning has made considerable progress in understanding consumer behaviour and predicting outcomes in the broader insurance sector, there remains a clear gap when it comes to life assurance-specific focus. Given the specific traits and complexity involved with life assurance products, as well as the distinct client behaviours that they imply, this gap in research focus is notable. Unlike other types of insurance, life assurance has a unique set of factors that necessitate extensive research. Life assurance products sometimes entail long-term commitments and financial preparation, which distinguishes them from non-life insurance plans. As a result, the variables impacting decision-making for life assurance products differ and cover a larger range of concerns than those for non-life insurance.

In addition, the life assurance sector confronts unique issues such as policyholder behaviour, lapse prediction, premium optimisation, and mortality rate forecasting. These difficulties are vastly different from those experienced in non-life insurance. Understanding client behaviours in relation to life insurance, predicting whether an existing application will become an active policy, and determining the drivers of these behaviours need specific attention and research dedicated exclusively to the life assurance business.

* 1. **Research Problem**

This research project aims to implement a supervised machine learning model that can predict the conversion of a life assurance application into an active policy. This prediction can help the life assurance company allocate resources effectively and improve their overall business efficiency.

The project will involve analysing various data points, including product selections and application details to identify the factors that affect the conversion of life assurance applications. By leveraging machine learning algorithms, the model will be able to provide a classify whether each application will become an active policy or not.

The project is pertinent to Data Analytics because it involves the use of supervised machine learning techniques to tackle a real-world problem in the life assurance sector. To build an accurate and efficient machine learning model, the project will necessitate the usage of several data analysis techniques such as data cleansing, feature importance identification, and model selection. Furthermore, the project will involve the application of numerous best practices methodologies in data analytics to assure the model’s validity and reliability.

**1.4 Research Objectives**

Based on the research topic, four research objectives that will be addressed using the Problem Definition model have been identified:

1. Evaluate the impact of the independent features’ correlation with the dependent variable and the impact of feature importance score to determine the variables that have the greatest impact on the conversion of life assurance applications, and develop supervised machine learning models using the features that have the greatest impact.
2. Design and execute experiments to test the impact of features that are most highly correlated with application conversion or have the highest feature importance scores. The findings of these experiments should help to identify the features that are most strongly associated with application conversion, validate the importance of specific features in predicting application conversion, and refine the predictive model to improve its performance.
3. Critically evaluate and examine the effectiveness of multiple supervised machine learning models and their hyperparameters in predicting the conversion of life assurance accurately and how the hyperparameters contribute to the performance of the machine learning models.
4. Validate the performance of the developed machine learning models for predicting the conversion of life assurance accurately to evaluate their generalisability, by testing them on unseen data, and comparing this performance to the performance achieved in training of the models. The findings of this can provide valuable insights into the performance and reliability of the machine learning models, and be used to indicate areas for additional model improvement and modification to improve performance and applicability.

These objectives align with the research topic, as they aim to develop and implement supervised machine learning models that can accurately predict the conversion of a life assurance application into an active policy. The objectives also address the key challenges and requirements of the project, such as data analysis techniques, model selection, and evaluation metrics.

**1.5 Scope**

This research project is dedicated to a thorough examination of the application of supervised machine learning models in the life assurance business, with a particular emphasis on predicting the conversion of an existing life insurance application into an active policy. In this regard, the study’s scope includes a number of essential factors that will aid in the construction of specialised prediction models for life assurance.

A detailed investigation of the features influencing the conversion of life assurance applications is an essential aspect of the research scope. The most significant variables that impact consumer decisions in the context of life assurance will be identified using feature selection methodologies. This thorough examination will serve as the foundation for model creation and refinement.

The scope will include multiple data pre-processing approaches due to the inherent difficulties of insurance data. These will address issues such as unequal class distributions and data quality concerns. Techniques for dealing with missing data, detecting outliers, and transforming data will be carefully utilised to maintain the prediction models’ integrity.

To evaluate the performance of the constructed predictive models, the research will use a comprehensive model assessment approach. A variety of assessment criteria will be used, including accuracy, precision, recall, F1-score, and Area Under the Curve of the Receiver Operating Characteristic (ROC AUC). This detailed study will provide a thorough report of the models’ performance and ability to anticipate life assurance application conversions appropriately.

Validation and generalisation will be critical components of the research, with both internal and external validation methodologies used. The study will go beyond theoretical correctness to measure the prediction models’ real-world applicability and robustness, which will be tested on previously unseen data. To benchmark the performance of the created prediction models, a comparison study will be performed. The interpretability of the predictive models will be stressed, allowing for a better understanding of the elements influencing consumer decisions in the life assurance industry.

1. **Literature Review**

Life assurance companies can receive vast volumes applications every day, and it is vital to determine which applications are most likely to be converted into a policy. Predictive analytics, and more specifically, supervised machine learning models have proven to be effective tools for such tasks. This literature review aims to provide an overview of the current research related to the development and implementation of supervised machine learning models for insurance application conversion prediction. In the context of life assurance application conversion prediction, supervised machine learning models can be trained on application data to identify the factors that impact conversion and predict the likelihood of a new application being converted.

**2.1 Machine Learning in Insurance**

Machine Learning techniques have been used within the insurance industry for some time to address a range of classification problems and feature importance identification, including prediction of customer purchase behaviour.

van der Putten et al. (2000) conducted a predictive modelling competition, the CoIL Challenge 2000, with the goal of identifying potential customers for insurance policies and explaining the factors that influence their purchasing decisions. The results showed that machine learning algorithms can be used effectively to correctly predict insurance policy purchasing, with accuracy rates as high as 88%. The best predicting features of policy ownership included demographic variables as well as variables related to the insurance product benefits. However, the effectiveness of the models depended on the specific characteristics of the sample population. This study provided a yardstick for evaluating the performance of machine learning models for predicting insurance policy purchasing, and informed the feature selection and engineering process.

Chang & Lai (2021) adopted a neural network-based approach to predict customer intentions of purchasing insurance policies. The researchers utilized the dataset from the CoIL Challenge 2000 (van der Putten et al.,2000) and employed three data pre-processing approaches to address the issue of imbalanced class distributions. The results obtained were found to be comparable with the top performing entries of the CoIL Challenge 2000, indicating the efficiency of the proposed model in predicting customer intentions. The study also aimed to identify factors that impact probability to purchase insurance policies via feature selection. Neighbourhood component analysis (NCA), sequential forward selection (SFS), and sequential backward selection (SBS) were used. The results of implementing SFS and eliminating socio-demographic features were found to be comparable with other submissions of the CoIL Challenge 2000. The study highlighted the importance of using machine learning approaches, such as artificial neural networks, in predicting intention of purchasing insurance policies. The utilisation of various data pre-processing techniques, including feature selection, feature construction, and under-sampling, proved useful in addressing the issue of imbalanced datasets.

Rubi et al. (2022) evaluated ten classification algorithms to select a model with the highest accuracy in predicting whether a customer would purchase insurance or not. Random Forest, Decision Tree Classifier, and Stochastic Gradient Descent models provided the highest levels of accuracy. The research provides important insights into identifying the features that have impact on decisions of customers when considering whether or not to by an insurance policy. The use of machine learning techniques to predict insurance purchasing behaviour is specifically relevant as it provides a data-driven approach to understanding customers behaviour.

Ampt (2017) aimed to investigate the potential of machine learning techniques in predicting customer interest in insurance products. The study utilised ten classification algorithms and conducted six experiments to determine which machine learning technique had the highest potential for predicting insurance product interest. It was found that the Decision Tree and Logistic Regression algorithms showed the highest potential for predicting insurance product interest. The accuracy achieved by machine learning techniques was up to 94%, allowing for confident predictions of customer insurance product interest. Moreover, machine learning showed ability in the handling of irrelevant features, eliminating the need for data scientists to comb through data to pick relevant features. Overall, the study suggests that machine learning can be an effective tool for predicting customer interest in insurance products, and recommends the use of Decision Tree and Logistic Regression algorithms for this purpose.

An et al. (2021) proposed a predictive model to predict whether existing health insurance customers were likely to purchase car insurance. Using logistic regression and boosted decision tree algorithms, they were able to develop a model with a high level of accuracy. They further concluded that this model could be useful for predicting customer behaviour, particularly insurance policy purchase intention.

Mau et al. (2018) aimed to accurately forecast the likelihood of purchasing life insurance using digital consumer data. The study discovered that customer data fed into a random forest model, yielded a prediction accuracy of over 90%. Overall, the study demonstrated that using consumer reaction data might significantly improve the accuracy of predicting purchase behaviour in the insurance industry. Through the research of Mau et al. (2018), Rubi et al. (2022), Ampt (2017), and Mau et al. (2018), it is clear that there is high level of support for the use of Decision Tree algorithm and their extensions, such as Random Forests, for accurately predicting customer purchasing intention in the insurance industry. There is also an element of support for Logistic Regression for prediction purposes.

Jaiswal (2022) explored the use of big data and machine learning in predicting the intention of a customer to purchase an insurance policy. Among six machine learning models, it was concluded that LightGBM was the most suitable for predicting purchase intention. Jaiswal further recommends the use of personal, geographical, and regional factors to predict the likelihood of a customer completing the policy purchase.

Mai et al. (2020) found that purchase intention, attitudes, financial awareness, and product accessibility all influence life insurance purchasing behaviour. Also, Nomi & Sabbir (2020) investigated the characteristics that influence consumer purchasing intentions for life insurance. According to the findings, attitude, subjective norms, risk aversion motives, saving motives, and financial literacy all have a significant favourable impact on customer purchase intention for life insurance. Saving motives have been recognized as a mediator in the association between risk aversion motives and purchase intention, as well as the relationship between financial literacy and purchase intention.

A range of studies have been carried out on the use of machine learning for other classification purposes within the non-life insurance industries, mostly churn prediction, claim prediction, and fraud detection.

Random Forest and AdaBoost have proven to be effective for classification purposes within the insurance industry, particularly in predicting customer churn (Stucki, 2019). It was found that that machine learning in general was a better and more feasible method of predicting customer churn than methods traditionally used within the insurance industry. Groll et al. (2022) explored the use of machine learning to predict policy cancellation likelihood. They found no significant difference observed between the performances of tree-based and logistic regression approaches to classify the life insurance policies. Mauritsius et al. (2020) evaluated the ability of each of Decision Tree, Naïve Bayes, and Artificial Neural Network as classification methods for a customer churn problem of an insurance company. In this case Decision Tree was found to be the most suitable approach for the creation of a customer churn model. Zhang et al. (2017) proposed a combined Deep & Shallow model for a classification task related to customer churn prediction within the insurance industry. They argue that this type of model has advantages such as generalisation and memorisation being present in one model. It was also concluded that the combined model outperformed both the deep-only and shallow-only methods in the classification task.

Pesantez-Narvaez et al. (2019) compared the use of logistic regression and XGBoost algorithms for classification purposes in the prediction of claims. Better predictive capacity and interpretability meant that logistic regression was the more suitable approach for problem presented in this study. Among a range of machine learning methods, Random Forest was found to have the best accuracy and generalisablity when it came to the classification task based on the insurance policy and customer data (Hanafy & Ming, 2021). The use of decision trees, and their subsequent extensions, such as gradient boosting and random forests was explored by Quan & Valdez (2018) as potential predictive models for insurance claim prediction. They found that multivariate tree-based models generally outperform univariate tree-based models. Frempong et al. (2017) developed a decision tree predictive model to predict the likelihood of a claim being made based on a number of risk factors within the insurance industry. While developing the model, they discovered that certain features had a greater impact on the likelihood of an insurance claim being made. Differences have been observed between how traditional machine learning methods deal with classification tasks in the insurance industry, and how deep learning approaches deal with the same (McDonnell et al., 2023). In claim prediction tasks, a deep learning architecture called TabNet outperformed more traditional machine learning models such as GLMs and XGBoost in terms of interpretability and accuracy. However, it was noted that the time to run TabNet and effort needed for hyperparameter tuning are possible limitations that must be considered.

Severino & Peng (2021) found that ensemble methods were most effective for fraud detection within property insurance, outperforming a range of other models, including logistic regression. The best suited model, used along with feature selection techniques, can be adapted for a probabilistic approach and improved with spatial analysis and other machine learning algorithms. However, their study did not use imbalanced classification methods or hyperparameter tuning, resulting in a gap in their research. Xia et al. (2022) determined that deep learning models combining CNN, LSTM, and DNN can perform better than traditional machine learning models in classification task, such as fraud detection, within insurance. They explain that deep learning is better suited to deal with the high dimensionality and large amounts of data that are often present within insurance dataset. Deep learning classification models have had issues with classification tasks within the insurance industry when the dataset is imbalanced. Muranda et al. (2021) addressed this problem by using sampling techniques while pre-processing the data in order to give balance to the dataset. In their experiments, the deep learning classification models performed well in detecting fraudulent claims when the dataset was balanced, but performed less well on imbalanced datasets.

Taha et al. (2022) considered the importance of feature selection when applying machine learning in the insurance industry. One particular challenge is the amount of noise often present within insurance datasets, and the subsequent negative impact this can have on performance of machine learning models. Taha et al. propose that this can be dealt with by using a selected set of features over the use of an entire dataset without feature selection applied. The most powerful variables on the intention to purchase life insurance are financial knowledge and attitude toward the purchase of life insurance. The influence of product accessibility, risk perception, and subjective norms on insurance intention is quantified and explored. Li (2019) highlighted the difficulties that insurance firms confront in staying competitive and discovering worthwhile consumers. The article discussed the use of customer data for descriptive statistical analysis and data cleaning in order to improve data quality. The paper then discussed how to estimate client preferences for life insurance products using logistic regression models, decision trees, and random forests. The results suggest that the combination model of random forest and logistic regression predicts customer behaviour the best. Dragos et al. (2020) presented an empirical study designed to better understand the impact of behavioural and socio-demographic characteristics on purchasing. The study discovered that specific behavioural characteristics and insurance knowledge are significantly significant for the purchase choice but not for the purchase intention. The study also discovered that financial education, as measured by a self-constructed Index of Insurance Knowledge, has a significant impact in explaining financial decisions. Life insurance is strongly encouraged by marital status, high levels of education, and income.

There are a number of ethical issues to consider when using data analytics and machine learning within the insurance industry, particularly in relation to issues around discrimination and fairness (Barry & Charpentier, 2022). Biases such as the use of irrelevant features and correlated but not causal features can be seen within machine learning approaches. It was concluded that contestability and transparency should be adopted when using machine learning within insurance in order to ensure fairness within in insurance. According to Anagol et al. (2017), instead of focusing on the coverage customers require, agents overwhelmingly propose unsuitable, high commission products and cater to the assumptions of misinformed consumers. Poor advice is motivated by commission incentives and agents' insufficient product expertise. The study also argues that financial product disclosure standards should be similar across the menu of substitutable items, as concealing information may be a significant component of agents' sales approach. The study asks how emerging markets with new investors might get excellent information on making financial decisions.

Due to the sheer amount of data now being produced within the insurance industry, there is an awareness that this has create a requirement for adequate technologies to effectively leverage this data for business benefit (Paruchuri, 2020). Machine learning can have a number of uses within the insurance industry including underwriting, fraud detection, entitlements management, and client capability.

**2.2 Correlation Analysis**

Gogtay & Thatte (2017) list a number of considerations that should be taken into account when using correlation analysis. These include the limitations of correlation with repeated measures, the impact of outliers, the presence of non-linear relationships, potential for false correlations, and the importance of sample size.

In explaining the usefulness of linear correlation coefficient between two variables in order to find the multicollinearity of variables in a model, Senthilnathan (2019) also warns of how the interpretation of correlation must not be conflated to state that it incorrectly represents as causation effect. It is insisted that correlation only explores and indicates the type and degree association between variables, but does not explicitly explain the relationship between them or causal effect.

When conducting correlation analysis, appropriate consideration should be given to the size of the sample being used in order to ensure that the results of the correlation analysis are able to achieve the required minimum correlation coefficient value with adequate power and type I error or p-value (Bujang & Baharum, 2016).

**2.3 Feature Importance**

Feature importance has been described as one of the most prevalent methods of explaining the way in which machine learning models behave (Saarela & Jauhiainen, 2021). Simple classification itself is not always the desired outcome, but rather knowing the importance of how specific features in a model can in some ways be explained, and furthermore how certain actions can either prevent or increase the likelihood of a certain classified outcome.

Gopagoni et al. (2020) evaluated important features and factors for better insurance sale conversion rates. The logistic regression model achieved a predictive accuracy of 84% and a cross-validation score of 81%. The SVM algorithm achieved a predictive accuracy of 80% accuracy. This provides valuable insights into the use of machine learning algorithms for selling insurance and the importance of selecting the right algorithm to improve the success rate of selling campaigns in insurance.

More recently, Merikanto (2022) focused on developing machine learning models to predict which customer attributes affected purchase decisions. Separate machine learning models were created for each product using LightGBM. Merikanto found that the models had high accuracy, with one product model having an almost 80% accuracy for predicting whether a customer would purchase the product or not.

RemOve And Retrain (ROAR) was developed as a benchmark to provide a way to evaluate the accuracy of feature importance identification methods in deep learning networks (Hooker et al., 2018). One interesting finding in this piece of research was that there were some instances a number of feature importance estimators were found to be less accurate at identifying feature importance than randomly assigning feature importance values the features in a dataset. This only further highlights the importance of using the correct and most suitable feature importance detection method.

Relative Feature Importance (RFI) has been said to grant a more nuanced approach to calculating feature importance (Konig et al., 2021). RFI has the ability to calculate the importance of a given feature relative to any other subset possible of features present in the dataset. There is, however, more research and development needed in relation to the challenges presented in sampling from unknown continuous variables and in using RFI on datasets with high dimensionality.

Wojtas & Chen (2020) introduced the concept of a dual-net architecture, where an operator and a selector work collectively in order to identify an optimal feature importance ranking and feature subset for feature importance ranking and subset selection for the purposes of deep learning. This method was used to develop an algorithm that learns by training both nets concurrently, resulting in an approach out-performs many of the best-in-class methods of ranking feature importance and feature selection.

Another proposed method of feature selection is the use of a feature selection algorithm called Dynamic Feature Importance based Feature Selection (DFIFS) (Wei et al., 2020). In addition to this, DFIFS can be used along with a traditional filter to create an algorithm known as Modified-Dynamic Feature Importance based Feature Selection (M-DFIFS). After applying it to 14 different high dimensional datasets, Wei et al. concluded that M-DFIFS performed better in relation to computational time and accuracy in comparison to a range of other feature selection algorithms.

When adopting feature selection in classification problems, classifier specific and classifier agnostic methods should be considered in order to calculate feature importance ranks (Rajbahadur et al., 2022). However, the limitations of using these should be kept in mind, as the results of using these measures do not always produce results that agree with each other. For example, where classification dataset consists of a high number of features that interact with each other, this can have an impact on how classifier specific feature importance methods performs, but not on the classifier agnostic method.

Zhou & Hooker (2021) discussed how bias has an impact in relation to split-improvement variable importance measures in tree-based methods, particularly Random Forest. This bias can be seen in the way split-improvement variable importance measures often give too much weight to features with more potential splits, leading to a skew in the derived rankings of feature importance. However, it was shown how this issue can be addressed by incorporating the split-improvement measured on out-of-sample data in order to correct the bias.

Using the Random Forest algorithm along with correlated predictors as a method of feature selection was evaluated by Gregorutti et al. (2016). They highlighted that high dimensionality in a dataset can be seen as a limitation for this type of approach in both classification and regression frameworks. However, they concluded that the use of Recursive Feature Elimination (RFE) can aid the performance of Random Forest when used for feature selection purposes.

Greenwell et al. (2018) proposed the development of a model-based approach to deriving feature importance that could be used with any supervised machine learning model. This approach would attempt to identify the level of interaction between variables by taking the effect of all features into account, and would then result in the same interpretation regardless of the selected supervised machine learning algorithm. However, there could be limitations to this approach, such as the impact of outliers and computational resources required to deal with large datasets.

**2.4 Feature Selection**

Feature selection has been found to be an effective way to reduce data dimensionality when pre-processing a dataset for machine learning (Li et al., 2017). When done correctly, feature selection can result in more comprehensive, simpler machine learning models. Li et al. state one area of concern to be that most feature selection models require the number of selected features to be specified prior to actually knowing the optimal number of features for the given problem. But the reward for getting it right is the reduction of noise within the dataset.

Chandrashekar & Sahin (2014) noted that comparisons between different types of feature selection methods can only be done when the methods are applied to the same dataset, and then based on the when comparing to baseline classification performance metrics, the most suitable feature selection approach can be chosen. They further conclude that a number of factors should be considered when making selecting a feature selection method, including simplicity, stability and classification accuracy. When applied correctly, feature selection can result in improved classification, enhanced generalisation and identification of noise.

Another benefit of feature selection and dimensionality reduction can be seen in how the these approaches help deal with the ‘curse of dimensionality’ by reducing noise and therefore helps models to avoid overfitting (Venkatesh & Anuradha, 2019). When reviewing feature selection methods, Venkatesh & Anuradha inferred that while wrapper methods are computationally more costly than filter-based methods, they are generally more accurate.

Huang et al. (2019) discuss how dimensionality reduction can be successful in identifying the essential characteristics needed in a dataset for machine learning purposes, particularly classification and clustering. However, it is warned that current dimensionality reduction techniques can lack efficiency due to their complex nature, particularly as the dimensionality of the data grows.

In a review of feature selection methods among datasets with a high level of dimensionality, it was observed that selecting the optimal feature selection method can be of great importance to improve the performance of machine learning algorithms, reduce the time needed for the model to learn, and increase the accuracy of the learning (Asir et al., 2016). While evaluating feature selection methods, it was found that subset-based methods were computationally inefficient, and therefore not suitable for high-dimensional data, while ranking methods showed improved generalisability as well more efficient computational performance. However, it was concluded that filter methods are ultimately the optimal choice for dealing with high-dimensional data as they require less computational power, and can perform better across classification algorithms.

To address high-dimensionality difficulties, Ke et al. (2017) proposed two novel techniques, Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB), which reduce the data size and number of features, respectively. They implemented these techniques in a new GBDT algorithm called LightGBM, and demonstrated its efficiency and scalability compared to other Gradient Boosted Decision Trees (GBDT) algorithms, such as XGBoost. The experimental results showed that LightGBM can achieve up to 20 times faster training time than conventional GBDT algorithms, while maintaining almost the same level of accuracy.

In a review of dimensionality reduction techniques, it was concluded that adopting Principal Component Analysis (PCA) along with machine learning algorithms generally produce in better results than machine learning alone when a dataset has a high level of dimensionality (Reddy et al., 2020). However, the review also determined that dimensionality reduction techniques should only be used where appropriate, such as datasets with high dimensionality, as applying dimensionality reduction techniques to datasets with low dimensionality resulted in poorer performance than using machine learning alone.

While there is much support for the benefits associated with feature selection, there are a number of limitations to be considered (Heinze & Dunkler, 2016). It can sometimes cause an unnecessary level of complication to analysis, while also invalidating statistical tools such as P-values and confidence intervals. Heinze & Dunkler further argue that expert knowledge is more valuable than over-complicated feature selection techniques.

**2.5 Hyperparameter Optimisation**

Andonie (2019) highlighted the significance of hyperparameter optimization in machine learning models, as well as the need to utilise a combination of optimization, search space, and training time reduction strategies to identify the optimum hyperparameters. According to Andonie, there is no quantitative procedure for selecting the right hyperparameters for a specific dataset, and the selection is based on trial and error.

The advantages of adopting automated hyperparameter optimization (HPO) in model-based reinforcement learning (MBRL) was discussed by Zhang et al. (2021). Because MBRL algorithms are sophisticated and have many hyperparameters and architectural options, they are difficult to apply to new problems without significant human input. Zhang et al. demonstrated that automated HPO can greatly outperform human tuning, and that dynamically tweaking hyperparameters during training can further increase performance. The trials shed light on the influence of various hyperparameters on training stability and the subsequent rewards.

Franceschi et al. (2017) investigated two methods (reverse-mode and forward-mode) for calculating the gradient of the validation error with regard to the hyperparameters of any iterative learning algorithm. The reverse-mode technique is related to past work but does not require reversible dynamics, whereas the forward-mode procedure is appropriate for real-time hyperparameter updates, which can accelerate hyperparameter optimization on big datasets. Franceschi et al. referred to research on data cleaning and learning task interactions and demonstrated that if the number of hyperparameters is minimal, forward-mode computing may be preferred to reverse-mode computation.

Yang & Shami, (2020) explored the significance of hyperparameter optimization in machine learning and proposed several cutting-edge optimisation approaches for common machine learning models. It also examined the performance of various optimisation approaches using benchmark datasets. According to Yang & Shami, if randomly selected subsets are highly representative of the given dataset, BOHB were the best choice for optimising a machine learning model, while BO models were recommended for small hyperparameter configuration space and PSO was the best choice for large configuration space.

Using Bayesian optimization, Joy et al. (2016) presented a novel paradigm for hyperparameter tuning on big data. The method separates large amounts of data into smaller chunks and uses typical Bayesian optimisation to build hyperparameter configurations for each chunk in parallel. Using a transfer learning configuration, the knowledge collected from the chunks is then used to tune the hyperparameters for the entire big dataset. The suggested method outperforms state-of-the-art hyperparameter tuning methods with less computing time when tested on two machine learning algorithms and two real-world datasets.

The importance of hyperparameters in machine learning algorithms and the difficulties in optimising them were discussed by Wu et al. (2019). To characterise the problem as an optimisation problem, the Wu et al. offered a hyperparameter tuning approach based on Bayesian optimisation and Gaussian processes. The approach proved to be effective in discovering the appropriate hyperparameters for frequently used machine learning models such as random forest and neural networks while drastically lowering runtime when compared to manual search.

Using 94 classification datasets from OpenML, Mantovani et al. (2018) analysed the effects of hyperparameter adjustment on three Decision Tree induction algorithims (CART, C4.5, and CTree). The goal was to assess the importance of hyperparameters and to identify the best optimisation approaches for hyperparameter tuning. According to the study, hyperparameter adjustment yielded statistically significant gains for C4.5 and CTree in only one-third of the datasets, and for CART in the majority of the datasets. The Irace approach was the best for all algorithms, and tweaking a specific small group of hyperparameters contributed the majority of the achievable ideal predictive performance.

The Random Forest (RF) algorithm is affected by a number of hyperparameters. While the default values are adequate, tweaking hyperparameters can increase performance, and the package tuneRanger automates this process using model-based optimisation (Probst et al., 2019). The number of trees should be set to a high value, and mtry is the most important hyperparameter. Sample and node sizes have a minimal impact but are worth adjusting. Large-scale comparison studies on hyperparameters and their impact on variable significance measures are lacking in the literature. Comparison studies are vital for evaluating and comparing the behaviours and performances of RF variations and hyperparameter choices. Although RF tuning can boost performance, the effect is smaller than that of other machine learning approaches. tuneRanger outperformed standard RF and other software that implements RF tuning.

Deep neural network (DNN) performance depends on hyperparameter optimization, and manual tuning can be time-consuming and inconvenient. Bayesian Optimization (BO)-based automated methods have been established, and Cho et al. (2020) analysed four strategies to improve BO for DNN hyperparameter optimization: diversification, early termination, parallelization, and cost function transformation. DEEP-BO, a simple yet resilient technique, outperformed well-known solutions on six DNN benchmarks. Research of the four techniques showed that diversity, conservative early termination, using partial training performance while parallelising, and heuristic cost function modification can all increase BO's performance. DEEP-BO performed at or near the top of all benchmarks examined.

**2.6 Experimental Design**

The automatic tweaking of design flow parameters was presented by Xie et al. (2020) as a machine learning-based solution. The suggested approach makes use of approximation sampling and clustering approaches to boost tuning effectiveness and reuses feature extraction information from earlier designs. The method makes use of a XGBoost model and suggests a novel dynamic tree methodology to get around overfitting. When compared to earlier techniques, experimental results on benchmark circuits and two industrial designs reveal a considerable gain in design quality or decrease in sampling cost.

The usefulness of label-specific justifications for digits in a convolutional neural network representation was assessed by Ahern (2019). A 5-layer convolutional network was used in the experiment, which was run on MNIST, and it attained a test accuracy of 99.04%. Instead of evaluating a specific prediction on a specific image, the study assessed how well the explanations conveyed the critical properties for each digit in the dataset.

The effectiveness feature selection algorithms (FSAs) was evaluated by Molina et al. (2002). The experimental methodology was described in depth, and many experiment parameters were quantified. To evaluate the effectiveness of the FSAs, twelve families of data sets were created and examined. The FSAs were modified, and a filtering standard was developed to reduce their output to a subset of features. It was argued that all FSAs should have roughly the same possibilities to compete in terms of the computational resources in order to select which algorithm to utilise in specific circumstances.

Miao & Niu (2016 ) examined the most recent feature selection algorithms. They tested feature selection techniques on 12 publicly accessible datasets and evaluated the results using normalized mutual information and clustering accuracy. MaxVar, Laplacian Score, SPEC, SPFS-SFS, MCFS, UDFS, NDFS, and EUFS are a few of the algorithms. The experiment employed the K-means algorithm with numerous random initializations, and it presented the mean findings together with the standard deviation. The findings demonstrated that feature selection strategies are advantageous for machine learning tasks and enhance clustering performance.

In order to discover causal relationships between word characteristics and class labels in document classification, Paul (2017) suggested a matching strategy. The method seeks to find more significant and broadly applicable features than only correlational approaches. The study made use of datasets of reviews from the medical, film, and product industries. Results revealed that the suggested strategy, especially when used with non-domain data, significantly improves classification performance and identifies interpretable word connections with sentiment. Propensity score matching outperforms McNemar's test in two out of three datasets where feature selection is concerned, according to comparison of the two methods.

Imai et al. (2012) discussed the limitations of experiments in identifying causal mechanisms and proposed alternative experimental designs to overcome these limitations. The proposed designs involved manipulating the mediator variable and assuming that the manipulation does not directly affect the outcome. They emphasised the importance of identifying assumptions directly linked to experimental design and highlighted recent social science experiments to illustrate the proposed designs. It was expected that the designs would open up possibilities for identifying causal mechanisms through clever manipulations and future technological developments in various scientific disciplines, including social and medical sciences.

**2.7 Validation**

While prediction models can be vital for decision-making and measuring performance, external validation is required in order to confirm the predictive accuracy of the model (Hickey & Blackstone, 2016). To carry out a suitably rigorous external validation study, a number of elements must be present, including appropriate study design, correct statistical methods, and clear and transparent reporting. Internal validation may not be sufficient to demonstrate predictive accuracy, and overfitting can lead to poor performance in external validation.

Adibi et al. (2020) gave warning on the importance of model validation, such that a lack of external validation among large amounts of data leads to many tests being untested and unvalidated, meaning that there could be a challenge to identify and select the most useful models. Fragmented efforts that assess only one model at a time do not allow for a reliable ranking of comparative performance.

Steyerberg & Harrell (2016) explored the significance of validation in predictive modelling, particularly highlighting the fact that model development studies are often not large enough, and that internal validation is of utmost importance, even more so than random split sample methods. They argue in favour of internal-external validation and direct tests for heterogeneity in predictor effects, concluding that fully independent external validation with data not available at the time of prediction model development is important.

Rahman et al. (2017) reviewed and evaluated a number of performance measures for external validation of prediction models. They recommended using Uno's concordance measure or Gönen and Heller's measure for quantifying concordance, Royston's D for assessing discrimination, and the calibration slope for assessing calibration. Also, investigating the characteristics of the validation data before choosing performance measures was recommended as a validation approach.

Accounting for competing events when developing and validating prediction models is also of great importance in model development (van Geloven et al., 2022). Failing to account for competing events can lead to overestimation of the cumulative incidence of an event of interest and distorted model performance. van Geloven et al. suggest methods of calculating and interpreting performance measures relating to the full risk distribution and a decision analytic perspective, consistent with TRIPOD guidelines for reporting prediction models. It is also noted that large sample sizes would generally be needed for a reliable performance assessment.

Debray et al. (2015) presented a framework for examining and improving the interpretation of prediction model external validation findings. By analysing their respective case-mix differences, the proposed methodological approach quantifies the degree of relatedness between development and validation samples on a scale spanning from reproducibility to transportability. The model's performance in the validation sample is evaluated and interpreted in light of case-mix changes, and the model is changed to the validation setting if necessary. The suggested framework improves the comprehension of results obtained during external validation of prediction models.

Validation Experiment Design Optimization (VEDO) method for prediction model design was proposed by Ao et al., (2017). This method was developed to maximise he information gain for model validation within the available testing constraints. In order to improve the robustness of the validation experiment design, a number of sources of uncertainty are included during the optimisation process.

Chen et al. (2007) provided a model validation strategy based on design, with the goal of increasing confidence in design decisions using a Bayesian prediction model. This method uses data from physical experiments and computer models to provide a framework for making predictions in the intended design domain. The proposal gives a fresh and enhanced perspective on model validation by connecting its definition to a specific design choice related to a specific design purpose, as well as direct estimations of the global influence of uncertainty sources on confidence in a design decision.

In the context of predictive modeling, Morrison et al. (2013) presented a systematic technique for splitting legacy data into calibration and validation sets, adopted from cross-validation. The approach is illustrated through an example using generated experiments of a nonlinear one degree-of-freedom oscillator. The proposed framework is broad in scope and can be used to a variety of challenges. The method is computationally intensive and needs to be improved.

Azpurua et al. (2014) discussed the significance of validation techniques in simulation tools for complicated situations, as well as the shortcomings of the standardized feature selected validation (FSV) method. By evaluating its shortcomings and complexities, it was hoped to uncover improvement opportunities to make FSV a more robust tool for data validation.

Parvandeh et al. (2020) explored how to utilise feature selection to increase machine learning model accuracy while avoiding overfitting. A consensus nested cross-validation (cnCV), a new approach that combines feature stability from differential privacy and nested cross-validation (nCV) were presented. The cnCV approach picks fewer features than nCV and has comparable accuracy to other methods such as private evaporative cooling (pEC). Parvandeh et al. compared these methods using simulated and real data and come to the conclusion that cnCV is an excellent and efficient way for combining feature selection and classification. The cnCV methodology can be combined with other feature selection and classification approaches, and it can handle overfitting by adjusting the threshold in the inner folds.

Misra & Yadav (2020) addressed how to use feature selection to increase the predictive accuracy of machine learning models. The Recursive Feature Elimination with Cross-Validation (RFECV) method was suggested and tested on a dataset using five distinct machine learning methods. According to the results, the simplest model, Logistic Regression, had the best accuracy. The study also implies that simpler models can outperform sophisticated models if the problem nature and appropriate feature selection strategies are thoroughly investigated. The research suggests that while feature selection is vital in enhancing the accuracy of ML models, the nature of the data, its quality, and volume should also be taken into account.

Demircioğlu (2021) examined how skewed results in radiomics datasets can be caused by poor feature selection prior to cross-validation. The researchers ran two experiments on ten publicly accessible radiomics datasets to assess the amount of bias introduced by feature selection prior to cross-validation. The findings revealed a significant positive bias, with higher dimensionality datasets more prone to overfitting. The study emphasized the necessity of avoiding data leakage and using feature selection correctly. The paper also analyses the effect of feature selection on classifier selection and compares the bias of various feature selection algorithms.

The need of validation in building robust multivariate models was discussed by Shi et al. (2018), as is the requirement for algorithms that can choose both minimal-optimal and all-relevant variables while effectively cross-validating. The MUVR algorithm used recursive variable elimination in a repeated double cross-validation procedure to uncover both minimal-optimal and all-relevant variables for regression, classification, and multilevel analysis. When compared to other methods, MUVR supported partial least squares and random forest modelling and has been found to provide prudent models with low overfitting and enhanced performance.

Cabitza et al. (2021) presented a meta-validation method for evaluating the reliability of external validation procedures for machine learning models. To inform the dependability of a validation approach, the suggested method takes dataset cardinality and similarity between training and validation sets into account. The methodology is demonstrated by validating a COVID-19 diagnostic model on 8 external validation sets. The validation datasets were determined to be adequate in terms of dataset cardinality and similarity, and the validated model reported good discrimination, usefulness, and calibration, implying that the results were sound. The research emphasizes the need of adequate external validation and presents a qualitative guideline for evaluating the reliability of validation techniques.

The bias and variance of model validation procedures has been investigated in the context of defect prediction models used by software quality assurance teams (Tantithamthavorn et al., 2017). The study discovered, through a case study of 18 systems, that single-repetition holdout validation produces estimates with greater bias and variation than the top-ranked model validation procedures, and advised out-of-sample bootstrap validation instead. The relevance of adopting an effective model validation technique as a major experimental design decision for accurate and reliable defect prediction was also emphasised.

The relevance of verifying predictive models was discussed by Ivanescu et al. (2015). It discussed why predictive validity decreases and presents metrics that are routinely used to estimate predictive validity. The research emphasises the need of reporting a model's projected loss of predictive power in new samples and gives methods for measuring and reporting validity shrinkage and predicted predictive validity. According to Ivanescu et al., future predictive modelling research should always report the projected decrease in predictive power of a model in new samples.

Steyerberg & Vergouwe (2014) presented a methodology for constructing and evaluating prediction models, with seven critical processes and four model performance measures: calibration-in-the-large, calibration slope, discrimination, and clinical applicability. They also explored model validation issues such as miscalibration and minor improvements in discrimination with additional markers, emphasising the significance of involving statistical specialists. The suggested approach aims to increase the methodological rigour and predictive model quality.

Ali & Gravin (2021) analysed various model validation methods for datasets containing software development effort estimation (SDEE) and software fault prediction (SFP). The study analysed estimate strategies' prediction accuracy and stability using ten different cross-validation (CV) and bootstrap validation methods. The results demonstrated that the model validation procedures that yield the best prediction accuracy are repeated 10-fold CV with SDEE data and optimistic boot with SFP data. The most stable model validation method for both SDEE and SFP datasets is repeated 5-fold CV. The study recommended employing model-agnostic methodologies to identify essential variables and instance-level interpretations to explain whether software systems are clean or flawed.

Adler & Painsky (2022) described a weakness in the commonly used Gradient Boosting Machines (GBM) technique that causes bias in its feature importance (FI) estimates due to the usage of decision trees that are biased towards categorical variables with large cardinalities. A cross-validated unbiased base learner framework (CVB) that addresses this issue and is effective in a variety of synthetic and real-world settings is proposed. According to the study, GBM FI is unique to each implementation, but CVB provides impartial FI without sacrificing generalization capabilities.

Altmann et al. (2010) highlighted the significance of interpretability in machine learning models and how linear models are frequently employed to evaluate feature relevance. However, it has been discovered that more complicated models, such as support vector machines and Random Forest (RF) models, have biased feature importance measurements. Altmann et al. offer a solution for normalizing feature significance measures in a non-informative context by using repeated permutations of the outcome vector to estimate the distribution of measured importance for each variable. This updated measure of feature importance enhanced model interpretability and is applicable to different learning methods.

**2.7 Model Evaluation Metrics**

Bylinskii et al. (2019) analysed and suggested 8 distinct evaluation measures and their properties under specified assumptions and for specific applications. The research stated that the choice of metric is determined by the qualities of the inputs, and that multiple metrics may be required for different tasks and applications.

Zhang et al. (2020) discussed the significance of assessment metrics in batch evaluations of information retrieval (IR) systems. The findings provided suggestions for fine-tuning assessment metric parameters and promote the consistency of user behaviour modelling and satisfaction measurement.

The area under the receiver operating characteristic curve (AUC) is a typical measure of discrimination for binary outcome prediction models, but it has been criticised for its shortcomings. Under the assumption of multivariate normality, Pencina et al. (2012) analysed this claim by linking the AUC to clinical performance indicators based on sensitivity and specificity. They found that, unless where good specificity is required, the change in the AUC is an appropriate predictor of the change in clinical performance indicators. In such circumstances, the discrimination slope may be a more accurate predictor of model improvement than AUC. However, if the baseline model performs well, increasing the AUC may be more difficult. There are some limitations to the study, such as the assumption of multivariate normality, linear discriminant analysis, and the restricted number of clinical measurements and risk thresholds considered. Nonetheless, the study implies that reporting the AUC increment is fair because changes in the AUC are proportionate to changes in clinical measures of prediction performance. If clinically meaningful metrics can be discovered, they should also be reported.

Marcot (2012) presented a variety of existing and new metrics for evaluating the performance and uncertainty of Bayesian Network (BN) models, including metrics for conducting model sensitivity analysis, evaluating scenarios, depicting model complexity, assessing prediction performance, and evaluating model posterior probability distributions' uncertainty. Marcot emphasised the value of metrics in enhancing model credibility, acceptance, and suitable application. The research emphasised the significance of balancing model performance and prudence. In addition, the study advises that metrics be chosen early in the model-building process to avoid post-hoc selection bias, and that metrics of performance and uncertainty can be used to assist select the best model from a group of competing models in a multi-model approach.

Tian et al. (2016) suggested that a correct error model should be used instead of metrics to evaluate models. Traditional metrics are interdependent, imperfect, and incapable of accurately assessing uncertainty because they are based on linear, additive, Gaussian errors. A accurate error model, on the other hand, contains the entire error information, conveys the error structure more naturally, and explicitly quantifies uncertainty. The error modelling methodology applies to both linear and nonlinear errors, however the metrics only apply to linear errors. The error model contains all of the information needed to evaluate the prediction model and can be used to build the conditional distribution between the data and the reference.

**2.8 Conclusions**

While supervised machine learning models have been used within the insurance industry to predict purchase intention behaviour of customers, there has been little research into the effectiveness of supervised machine learning models to predict whether an existing life insurance application will eventually become an active policy. Also, while there has been some research into the features that contribute to the likelihood of a customer purchasing non-life insurances, the same level of knowledge and research is not present in relation to life insurance. By satisfying the proposed research objective, this study can address these research gaps, while also taking into account previously researched best practices in machine learning in insurance, correlation analysis, feature importance, feature selection, hyperparameter optimisation, experimental design, and model evaluation metrics.

This literature review provided useful insights into the significance of several elements connected to the usage of machine learning techniques in the insurance industry. The research examined have shown that machine learning algorithms are useful at accurately predicting client purchase behaviour and handling classification challenges in the insurance industry. The importance of feature selection and feature engineering has emerged as a significant subject in the literature. Different data pre-processing approaches have proved helpful in resolving issues like uneven class distributions and finding characteristics that impact the likelihood of purchasing insurance policies.

Furthermore, the literature review demonstrated the efficacy of machine learning approaches in various classification tasks in the insurance industry, such as churn prediction, claim prediction, and fraud detection. The literature review emphasised the need of validation in predictive modelling and machine learning. External validation, in addition to internal validation, is required to check the predicted accuracy and resilience of models. For trustworthy comparative performance assessment and assuring the generalisability and repeatability of predictive models, rigorous validation techniques and transparent reporting are essential.

Finally, the literature review stressed the significance of adopting proper evaluation criteria for model evaluation. Different assessment measures have different qualities and may be better suited to different tasks and applications. The assessment metrics used are determined by the features of the inputs, the model's aims, and the environment in which the model will be implemented. To guarantee a complete assessment of model performance and to make informed judgments in model creation and selection, researchers must carefully evaluate these elements.

1. **Methodology**

**3.1 Sampling Strategy**

A population of life assurance policy applications, made up of every application received over a six year period, was selected as a suitable population of interest based on the primary research approach of conducting experiments to test the impact of highly correlated features and features with high feature importance scores on the performance of machine learning models. Life assurance applications contain a plethora of that can be utilised to examine the influence of different features on model performance.

Due to the likelihood of the presence of a diverse pool of applicants for life assurance, stratified random sampling is be an appropriate sampling strategy for this research when splitting the population dataset into training, validation and test datasets. By using the stratified technique, it is possible to ensure that the training, validation, and test splits chosen for the experiment is representative of the whole population and that each record of the population has an equal chance of being included in the sample. This eliminates the chance of selection bias, which happens when particular groups are either overrepresented or underrepresented, and helps to ensure that the research findings are applicable to the entire population of interest. This is a form of probability sampling as each member of the population has an equal chance of being chosen for the sample.

**3.2 Primary Research Methodology**

A quantitative type of experimentation is used as the principal form of primary research. To carry out this experiment, only the highly correlated features and features with high feature importance scores are included for the life assurance policy applications, and then the classification results of the machine learning models run only with the selected features are compared to those of a control group in which all features are included. The control group serves as a baseline against which the selected features findings can be evaluated to determine the causal influence of the features on the classification models’ performance. The results of the experiment can help validate the significance of the features in predicting application conversion and improve the machine learning models’ predictive performance.

**3.3 Data Collection**

The study methodology includes both descriptive and analytical components. Descriptive statistics are used to describe and illustrate the features of the life assurance applications, while analytical approaches are used to look for patterns, correlations, and trends in the dataset.

The data is provided by a life insurance organisation and covers all applications submitted between 2017 and 2022. The researcher worked with the life insurance company to extract pertinent data from their records. This information included product details, application dates, application statuses, and any other relevant features. Because the full information over a six-year period is available, it gives a comprehensive perspective of all applications and enables for in-depth study.

The dataset consists of diverse features related to insurance policies, including Product, ProductGroup, ProductType, Agency, WorkflowStatus, Indexation, NoOfLives, CommDateProvided, PaymentFreq, UWDecision, ComissionSacrifice, CommissionSacrificeType, RenewalSacrificeType, CommissionTerms, Discount, BonusCommission, FreeCover, SeriousIllnessType, and SignedDecReceived. PolicyIssued is the variable of interest.

**3.4 Data Cleansing**

The first stage of data cleansing requires preparing the dataset. For the sake of this research, columns having the data type ‘int64’ are considered categorical characteristics. Following that, these categorical columns are transformed to the ‘category’ data type. This conversion not allows for more efficient categorical data handling. Then, by choosing columns with numeric data types, numerical properties are segregated. For each numeric parameter, the Median Absolute Deviation (MAD), a robust measure of data variability, is determined. MAD gives a more trustworthy assessment of data dispersion than traditional metrics such as standard deviation and is less susceptible to outliers. A threshold multiplier (denoted as ‘k’) is chosen to identify probable outliers. In this investigation, a multiplier of three is used. The threshold for each characteristic is calculated by multiplying the MAD by the multiplier of choice, establishing a standard for finding data points that differ considerably from the norm.

A comparison method is used to identify outliers. Individual data points’ absolute variances from their respective attribute medians are compared to a predetermined threshold. The result is a binary matrix that highlights the existence or absence of outliers for each data point. For features identified as potentially containing outliers, the np.log1p function is used to perform a logarithmic transformation on these features. This change reduces the influence of extreme values and brings them closer to the middle of the distribution.

**3.5 Data Exploration**

Descriptive statistics are produced using the numeric\_data.describe() method to acquire a basic overview of the dataset. This produces important statistical measures including mean, median, standard deviation, and quartiles, which provide insight into the central tendency and dispersion of numerical variables. In addition, preliminary observations on the data’s features are made.

Understanding data patterns requires visualising the distribution and change of important variables. To show the distribution of numeric features side by side, box plots and violin plots are constructed. The box plot shows the quartiles and outliers, but the violin plot shows the distribution’s form in greater detail. These plots graphically represent the variability and range of the variables, assisting in the detection and comprehension of potential outliers.

Density plots and histograms can reveal information about the distribution of numerical data. Density plots provide the data’s estimated probability density function, whereas histograms show data frequency in bins. For each numeric variable, a density map is created to visually analyse the underlying distribution and potential multimodality. Histograms augment this evaluation by displaying the frequency of data points inside predetermined bins.

Quantile-Quantile (Q-Q) plots are used to determine if data follows a given theoretical distribution, most often the normal distribution. Deviations from the predicted distribution are shown by comparing the actual data quantiles to those of a theoretical distribution. The Q-Q plot of each numeric variable is constructed to examine its deviation from normalcy.

**3.6 Feature Correlation & Feature Importance**

To understand the composition of categorical variables, their frequency distributions are calculated. There are cross-tabulations between category variables and the target variable. These tables provide insights into the relationship between variables and aid in the identification of patterns. The chi-square test evaluates the independence of categorical variables and the target variable, assessing if actual and predicted frequencies differ considerably.

The strength of correlations between pairs of categorical variables is determined by Cramer's V, a measure of association for categorical variables. This demonstrates the extent to which variables are dependent on one another beyond the reported frequencies. To show correlations among categorical variables, a matrix of Cramer's V values is produced and presented as a heatmap.

Correlation analysis investigates the connections between numerical variables. To understand the strength and direction of relationships, correlation matrices are constructed, shown via heatmaps, and analysed. The emphasis is on identifying variables that are highly correlated with the target variable and with one another, indicating possible predictive power or multicollinearity problems.

To aid robust model evaluation, the dataset is separated into two subsets: a training set and a test set. The train\_test\_split method was used, with a test size of 20%, and stratified on the target column. This method assures that the class distribution is consistent in both sets and reduces the possibility of overfitting. This test dataset is set aside to be used as unseen data, for future external validation of the trained models. The training dataset is used to calculate feature importance, using several techniques.

A Random Forest classifier is created for each value of n\_estimators (50, 100, and 200) using the RandomForestClassifier class from the sklearn.ensemble module. The classifier is then trained using the training data. Following Random Forest classifier training, feature importances are computed using the classifier's feature\_importances function. These significance scores are saved in a Pandas Series called feature\_importances, which is indexed by name and sorted in decreasing order. The relevance of each feature is presented in the terminal for each n\_estimators setting. A loop is used to iterate over a set of n values that reflect the number of top-ranked characteristics to choose from. The most significant n characteristics are picked for each n value based on their significance scores. A new dataset (X\_train\_selected and X\_test\_selected) is constructed with only the selected characteristics. Following that, a new Random Forest classifier (rf\_classifier\_selected) is trained and tested on the test data. The ROC AUC score is calculated with the roc\_auc\_score function from the sklearn.metrics module and saved in the dictionary feature\_importances\_mdrauc. The same approach is used with a Gradient Boosted Classifier, and a LightGBM Classifier.

In order to calculate Class Aware Feature Importance, a RandomForestClassifier is constructed. The classifier is then trained using the training data. Following that, feature importance scores are determined using the trained classifier’s feature\_importances function. The relevance of each characteristic in formulating predictions is represented by these ratings. The study goes beyond evaluating global feature relevance by computing class-specific importance ratings. The dataset has several classes, and significance ratings are calculated separately for each class. A loop iterates over the target variable’s unique class labels. A mask is built for each class label to separate data points that belong to that class. The overall feature importance scores are multiplied by the mask’s mean to determine class-specific relevance. This provides insights into which features are particularly relevant for each class. To assess the overall performance of the RandomForestClassifier, the ROC AUC for the entire dataset is generated using the predict\_proba function. The expected probabilities for the positive class are retrieved from the classifier’s predictions. The ROC AUC score evaluates the classifier’s ability to differentiate between positive and negative classes. To record feature importances determined using the Mean Decrease in ROC AUC (MDRAUC) approach, an empty dictionary is established. A loop iterates over the dataset features. By permuting the feature’s values, a new dataset (X\_feature) is constructed for each feature. The feature-target variable link is significantly disrupted by this permutation. The ROC AUC for the dataset with the permuted feature (roc\_auc\_permuted) is computed. The difference between the overall ROC AUC and the permuted ROC AUC is then used to calculate MDRAUC for the current feature. This computation measures the impact of each feature on the performance of the classifier. The MDRAUC scores obtained are saved in the dictionary, which associates each feature with its appropriate importance in terms of performance variation.

Recursive Feature Elimination with Cross-Validation (RFECV) is also used to calculate feature importance. A RandomForestClassifier is trained using the training data. To develop predictive models, the training procedure requires maximising an ensemble of decision trees inside a random forest framework. RFECV is setup with key parameters and initialised using the rf\_classifier as the estimator. It takes a step-by-step method, removing one feature at a time, and applies a StratifiedKFold cross-validation strategy with five folds to ensure class balance during the selection phase. The ROC AUC scoring metric was used for assessment since it measures the classifier’s ability to discriminate across classes. RFECV examines the relevance of each feature systematically, discarding the least informative ones repeatedly until the optimum subset of features is discovered. After the RFECV procedure is completed, the indices corresponding to the selected features are retrieved using rfecv.support. These indices reveal which features have the most impact on the classifier’s performance. Following that, the relevant feature names from the original feature set (X.columns) are retrieved based on the specified indices, resulting in a subset of features judged most important for predictive modelling. The technique culminates with a straightforward reporting of the selected feature names, providing clarity on which features have been recognised as critical for the maximum performance of the RandomForestClassifier. This systematic method to feature selection and classification helps to advance the field of model optimisation by improving the interpretability and efficacy of prediction models. The same RFECV approach is taken with GradientBoostingClassifer and LightGBM respectively.

The same approaches of RandomForestClassifier, GradientBoostingClassifer, LightGBM, Class Aware Feature Importance, RandomForestClassifier with RFECV, GradientBoostingClassifer with RFECV, and LightGBM with RFECV are carried out on a BorderlineSMOTE oversampled training dataset, in order to investigate if this results in a better performance than the original imbalanced dataset.

For each of the features identified as having the most importance, hypotheses are developed to imply that its existence improves model performance. The training dataset is divided further into training and validation sets using an 80-20 split ratio to analyse the influence of these variables. An experimental setup with a control group and an experimental group is developed for each feature and model combination. The control group is untouched, while the experimental group is subjected to feature alteration depending on the corresponding hypothesis. Each machine learning model is trained on both groups, and predictions are generated, allowing ROC AUC values to be calculated. The significance of the results is then determined using statistical significance testing. To compare the control and experimental groups, independent two-sample t-tests are used, and p-values are calculated to determine the significance of the observed differences.

**3.7 Machine Learning**

First, the dataset is constructed by extracting feature variables (X) and the target variable (y), with the target variable PolicyIssued. Following that, using the train\_test\_split function, the dataset is divided into training and testing sets, with 80% of the data allocated to the training set, and 20% set aside to be used as unseen data, for future external validation of the trained models. The training dataset is divided further into training and internal validation sets using an 80-20 split ratio. Logistic Regression, Decision Tree, Random Forest, Gradient Boosting, and LightGBM are examined for assessment. A series of steps are carried out for each model. The model is first fitted to the training data. Following that, predictions are generated on the validation data, and each of accuracy, precision, recall, F1 score, and ROC AUC are calculated. The results for each model are displyed, together with the model name and related assessment metric outputs.

Following that, each model’s hyperparameters are tuned using GridSearchCV from scikit-learn. For each model, hyperparameter grids are created to define the hyperparameters for optimisation. The best hyperparameters are saved, and the model with the best hyperparameters is chosen. The best-performing model is re-evaluated after hyperparameter adjustment using the same evaluation measures as previously. The findings are shown, along with the optimal hyperparameters. Each model goes through this iterative procedure.

A systematic framework to evaluate each machine learning model using only the identified most important features for each model respectively is implemented. Feature selection, which involves selecting significant predictor variables from the dataset based on the feature importance investigations, is applied. For best model performance, a hyperparameter tuning grid is defined. After that, the model is fitted, and predictions are assessed. The results are reported, and model comparisons are made.

The above machine learning approaches are applied in the same way to a BorderlineSMOTE oversampled version of the same dataset. This is done in order to take into account the imbalanced target variable in the original dataset and to compare the machine learning results of the oversampled data against the results of the original imbalanced dataset.

**3.8 Performance Validation**

This research’s methodology is intended to construct and assess predictive models for policy issuance prediction utilising previously unseen data. The previously mentioned unseen test dataset is used to test for model generalisability to unseen data. ROC AUC, accuracy, precision, recall, and F1 score, were computed on the unseen test dataset. When applied to real-world, previously unseen data, these measures give a better assessment of each model's predictive ability.

The best model for policy issuance prediction is chosen based on the model with the greatest ROC AUC score on the test dataset. The major criteria was chosen because of its usefulness in gauging a model’s discriminating power when applied to entirely new data. This thorough process ensures a rigorous approach to model creation and validation, resulting in the selection of the most effective predictive model.

1. **Results**

**4.1 Feature Correlation & Feature Importance**

**4.1.1 Imbalanced Training Data**

**4.1.1.1 Association Analysis**

|  |  |  |  |
| --- | --- | --- | --- |
| **PolicyIssued vs. Feature** | **Cramer's V** | **Chi2 Statistic** | **P-value** |
| Product | 0.1197 | 1809.3459 | 0 |
| ProductGroup | 0.0966 | 1175.8404 | 1.28E-254 |
| ProductType | 0.0945 | 1125.4886 | 1.07E-243 |
| Agency | 0.2138 | 7343.7851 | 0 |
| WorkflowStatus | 0.407 | 20840.0214 | 0 |
| Indexation | 0.0377 | 179.9643 | 4.93E-41 |
| NoOfLives | 0.0697 | 612.0287 | 4.05E-135 |
| CommDateProvided | 0.2196 | 6066.4651 | 0 |
| PaymentFreq | 0.0076 | 9.3465 | 0.0093 |
| UWDecision | 0.3446 | 14941.0955 | 0 |
| ComissionSacrifice | 0.0195 | 48.6988 | 2.98E-12 |
| CommissionSacrificeType | 0.0201 | 53.0057 | 3.09E-12 |
| RenewalSacrificeType | 0.0155 | 33.1672 | 2.97E-07 |
| CommissionTerms | 0.0007 | 1.0562 | 0.3041 |
| Discount | 0.0053 | 4.4829 | 0.0342 |
| BonusCommission | 0.0104 | 14.6602 | 0.0001 |
| FreeCover | 0.0452 | 258.4565 | 3.72E-58 |
| SeriousIllnessType | 0.0143 | 27.8336 | 9.04E-07 |
| SignedDecReceived | 0.0927 | 1082.656 | 1.94E-237 |
| day\_of\_week | 0.0079 | 13.8562 | 0.0313 |
| day\_of\_month | 0.007 | 36.2092 | 0.2013 |
| month | 0.0091 | 21.3243 | 0.0302 |
| year | 0.0318 | 131.9064 | 9.37E-27 |

**Figure 1 – Association Analysis, Imbalanced Training Dataset**

**A screenshot of a computer screen

Description automatically generated**

**Figure 2 – Association Analysis P-Value Matrix, Imbalanced Training Dataset**

**A screenshot of a computer screen

Description automatically generated**

**Figure 3 – Association Analysis Chi Squared Statistic Matrix, Imbalanced Training Dataset**

**A screenshot of a computer screen

Description automatically generated**

**Figure 4 – Association Analysis Cramer’s V Matrix, Imbalanced Training Dataset**

**4.1.1.2 Random Forest**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **n\_estimators: 50** | **n\_estimators: 100** | **n\_estimators: 200** |
| Agency | 0.568212 | 0.567376 | 0.567266 |
| WorkflowStatus | 0.177457 | 0.174382 | 0.192298 |
| UWDecision | 0.099782 | 0.105368 | 0.089511 |
| CommDateProvided | 0.037087 | 0.034927 | 0.03397 |
| SignedDecReceived | 0.014222 | 0.014224 | 0.013956 |
| NoOfLives | 0.013113 | 0.01352 | 0.013573 |
| BonusCommission | 0.01271 | 0.012435 | 0.012193 |
| ProductGroup | 0.012109 | 0.011068 | 0.010103 |
| SeriousIllnessType | 0.011081 | 0.011337 | 0.011183 |
| CommissionTerms | 0.01093 | 0.011093 | 0.010966 |
| Product | 0.010863 | 0.011422 | 0.011795 |
| Indexation | 0.009646 | 0.009994 | 0.00965 |
| ProductType | 0.005605 | 0.005567 | 0.005328 |
| CommissionSacrificeType | 0.003944 | 0.003942 | 0.003827 |
| RenewalSacrificeType | 0.003862 | 0.003927 | 0.003923 |
| ComissionSacrifice | 0.003201 | 0.003182 | 0.003242 |
| Discount | 0.002941 | 0.0031 | 0.003098 |
| PaymentFreq | 0.002637 | 0.002632 | 0.002655 |
| FreeCover | 0.000598 | 0.000504 | 0.000463 |

**Figure 5 – Random Forest Feature Importance Scores**

|  |  |  |
| --- | --- | --- |
| **n\_estimators** | **No of Features** | **ROC AUC** |
| 50 | 1 | 0.513652518 |
| 50 | 2 | 0.641456228 |
| 50 | 3 | 0.645852392 |
| 50 | 4 | 0.647158385 |
| 50 | 5 | 0.649896148 |
| 50 | 6 | 0.646723597 |
| 50 | 7 | 0.650177346 |
| 100 | 1 | 0.514175626 |
| 100 | 2 | 0.641763159 |
| 100 | 3 | 0.640881501 |
| 100 | 4 | 0.64258462 |
| 100 | 5 | 0.650422605 |
| 100 | 6 | 0.645342128 |
| 100 | 7 | 0.648090703 |
| 200 | 1 | 0.513894784 |
| 200 | 2 | 0.643598959 |
| 200 | 3 | 0.641212846 |
| 200 | 4 | 0.643082754 |
| 200 | 5 | 0.651537394 |
| 200 | 6 | 0.648096488 |
| 200 | 7 | 0.647948886 |

**Figure 6 – Random Forest ROC AUC by n\_estimators and number of features**

**A graph with a line

Description automatically generated**

**Figure 7 – Random Forest ROC AUC vs. No of Features (n Estimators: 50)**

**A graph with a line

Description automatically generated**

**Figure 8 – Random Forest ROC AUC vs. No of Features (n Estimators: 100)**

**A graph with a line

Description automatically generated**

**Figure 9 – Random Forest ROC AUC vs. No of Features (n Estimators: 200)**

**4.1.1.3 Gradient Boosting Classifier**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **n\_estimators: 50** | **n\_estimators: 100** | **n\_estimators: 200** |
| WorkflowStatus | 0.732134 | 0.717006 | 0.702411 |
| Agency | 0.123139 | 0.135537 | 0.148086 |
| UWDecision | 0.0566 | 0.056161 | 0.056156 |
| ProductGroup | 0.028173 | 0.027444 | 0.026817 |
| CommDateProvided | 0.017467 | 0.017573 | 0.017355 |
| Product | 0.012451 | 0.012961 | 0.013004 |
| BonusCommission | 0.010935 | 0.011256 | 0.011202 |
| NoOfLives | 0.01082 | 0.010808 | 0.010719 |
| ProductType | 0.003844 | 0.004737 | 0.004851 |
| SignedDecReceived | 0.003841 | 0.004333 | 0.004382 |
| Indexation | 0.000516 | 0.000777 | 0.00105 |
| CommissionTerms | 0.000078 | 0.000308 | 0.000899 |
| PaymentFreq | 0 | 0.00005 | 0.000151 |
| ComissionSacrifice | 0 | 0.000033 | 0.000445 |
| CommissionSacrificeType | 0 | 0.000026 | 0.000585 |
| RenewalSacrificeType | 0 | 0.000192 | 0.000319 |
| Discount | 0 | 0.000214 | 0.000394 |
| FreeCover | 0 | 0 | 0 |

**Figure 10 – Gradient Boosting Classifier Feature Importance Scores**

|  |  |  |
| --- | --- | --- |
| **n\_estimators** | **No of Features** | **ROC AUC** |
| 50 | 1 | 0.500179276 |
| 50 | 2 | 0.571261497 |
| 50 | 3 | 0.586884373 |
| 50 | 4 | 0.575311596 |
| 50 | 5 | 0.584246299 |
| 50 | 6 | 0.585113797 |
| 50 | 7 | 0.583316572 |
| 50 | 8 | 0.58191815 |
| 50 | 9 | 0.58191815 |
| 50 | 10 | 0.581867088 |
| 50 | 11 | 0.590673578 |
| 50 | 12 | 0.590225388 |
| 50 | 13 | 0.590225388 |
| 50 | 14 | 0.590225388 |
| 50 | 15 | 0.590225388 |
| 50 | 16 | 0.590225388 |
| 50 | 17 | 0.590225388 |
| 50 | 18 | 0.590225388 |
| 50 | 19 | 0.590225388 |
| 100 | 1 | 0.500179276 |
| 100 | 2 | 0.573105875 |
| 100 | 3 | 0.591318555 |
| 100 | 4 | 0.590289495 |
| 100 | 5 | 0.597394541 |
| 100 | 6 | 0.592200213 |
| 100 | 7 | 0.598981374 |
| 100 | 8 | 0.596557802 |
| 100 | 9 | 0.596557802 |
| 100 | 10 | 0.596481767 |
| 100 | 11 | 0.59705817 |
| 100 | 12 | 0.597083701 |
| 100 | 13 | 0.597301552 |
| 100 | 14 | 0.597915973 |
| 100 | 15 | 0.597916531 |
| 100 | 16 | 0.598236507 |
| 100 | 17 | 0.598377207 |
| 100 | 18 | 0.598377207 |
| 100 | 19 | 0.598377207 |
| 200 | 1 | 0.500179276 |
| 200 | 2 | 0.575336569 |
| 200 | 3 | 0.599354087 |
| 200 | 4 | 0.594468366 |
| 200 | 5 | 0.596559477 |
| 200 | 6 | 0.603299829 |
| 200 | 7 | 0.606696932 |
| 200 | 8 | 0.609349725 |
| 200 | 9 | 0.609349725 |
| 200 | 10 | 0.606772408 |
| 200 | 11 | 0.610500297 |

**Figure 11 – Gradient Boosting Classifier ROC AUC by n\_estimators and number of features**

A graph with blue dots

Description automatically generated

**Figure 12 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 50)**

A graph on a white surface

Description automatically generated

**Figure 13 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 100)**

**A graph with blue lines

Description automatically generated**

**Figure 14 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 200)**

**4.1.1.4 LightGBM Classifier**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **n\_estimators: 50** | **n\_estimators: 100** | **n\_estimators: 200** |
| Agency | 963 | 1681 | 2450 |
| UWDecision | 78 | 112 | 164 |
| CommDateProvided | 74 | 111 | 177 |
| SignedDecReceived | 58 | 126 | 367 |
| ProductType | 56 | 137 | 320 |
| WorkflowStatus | 40 | 62 | 86 |
| NoOfLives | 40 | 117 | 390 |
| ProductGroup | 34 | 119 | 395 |
| BonusCommission | 32 | 100 | 334 |
| Product | 28 | 58 | 111 |
| SeriousIllnessType | 26 | 126 | 390 |
| CommissionTerms | 16 | 59 | 220 |
| CommissionSacrificeType | 15 | 31 | 75 |
| ComissionSacrifice | 14 | 41 | 126 |
| Indexation | 13 | 67 | 266 |
| Discount | 9 | 20 | 40 |
| PaymentFreq | 2 | 12 | 25 |
| FreeCover | 2 | 3 | 9 |
| RenewalSacrificeType | 0 | 18 | 55 |

**Figure 15 – LightGBM Classifier Feature Importance Scores**

|  |  |  |
| --- | --- | --- |
| **n\_estimators** | **No of Features** | **ROC AUC** |
| 50 | 1 | 0.5118 |
| 50 | 2 | 0.6188 |
| 50 | 3 | 0.6232 |
| 50 | 4 | 0.6214 |
| 50 | 5 | 0.6245 |
| 50 | 6 | 0.6415 |
| 50 | 7 | 0.6424 |
| 50 | 8 | 0.6379 |
| 50 | 9 | 0.6411 |
| 50 | 10 | 0.6399 |
| 50 | 11 | 0.6405 |
| 50 | 12 | 0.6379 |
| 50 | 13 | 0.6415 |
| 50 | 14 | 0.6415 |
| 50 | 15 | 0.6411 |
| 50 | 16 | 0.6397 |
| 50 | 17 | 0.6399 |
| 50 | 18 | 0.6404 |
| 50 | 19 | 0.6404 |
| 100 | 1 | 0.5126 |
| 100 | 2 | 0.5197 |
| 100 | 3 | 0.5223 |
| 100 | 4 | 0.521 |
| 100 | 5 | 0.5223 |
| 100 | 6 | 0.5224 |
| 100 | 7 | 0.6203 |
| 100 | 8 | 0.6212 |
| 100 | 9 | 0.6257 |
| 100 | 10 | 0.625 |
| 100 | 11 | 0.6425 |
| 100 | 12 | 0.6427 |
| 100 | 13 | 0.6426 |
| 100 | 14 | 0.6437 |
| 100 | 15 | 0.6428 |
| 100 | 16 | 0.6431 |
| 100 | 17 | 0.6436 |
| 100 | 18 | 0.6428 |
| 100 | 19 | 0.6434 |
| 200 | 1 | 0.5126 |
| 200 | 2 | 0.5132 |
| 200 | 3 | 0.5144 |
| 200 | 4 | 0.5163 |
| 200 | 5 | 0.5165 |
| 200 | 6 | 0.5226 |
| 200 | 7 | 0.524 |
| 200 | 8 | 0.5249 |
| 200 | 9 | 0.5251 |
| 200 | 10 | 0.564 |
| 200 | 11 | 0.6264 |
| 200 | 12 | 0.6273 |
| 200 | 13 | 0.6266 |
| 200 | 14 | 0.6437 |
| 200 | 15 | 0.6448 |
| 200 | 16 | 0.6446 |
| 200 | 17 | 0.6464 |
| 200 | 18 | 0.6447 |
| 200 | 19 | 0.644 |

**Figure 16 – Light GBM Classifier ROC AUC by n\_estimators and number of features**

A graph with blue dots

Description automatically generated

**Figure 17 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 50)**

A graph with blue dots

Description automatically generated

**Figure 18 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 100)**

A graph with a line and dots

Description automatically generated

**Figure 19 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 200)**

**4.1.1.4 Class Aware Feature Importance**

|  |  |
| --- | --- |
| **Feature** | **MDRAUC** |
| WorkflowStatus | 0.1325 |
| Agency | 0.0302 |
| UWDecision | 0.0215 |
| CommDateProvided | 0.0118 |
| SignedDecReceived | 0.0077 |
| NoOfLives | 0.0037 |
| CommissionTerms | 0.0034 |
| SeriousIllnessType | 0.0015 |
| PaymentFreq | 0.0002 |
| Indexation | 0.0002 |
| FreeCover | 0 |
| Discount | -0.0005 |
| RenewalSacrificeType | -0.0005 |
| BonusCommissionPercentage | -0.0005 |
| CommissionSacrificePercentage | -0.0006 |
| ProductType | -0.0021 |
| CommissionSacrificeType | -0.0026 |
| BonusCommission | -0.0035 |
| ComissionSacrifice | -0.0041 |
| ProductGroup | -0.0062 |
| Product | -0.0097 |

**Figure 20 – Feature Weighted Importance Scores**

**A blue and white rectangle with a black border

Description automatically generated**

**Figure 21 – Feature Weighted Importance Scores Bar Chart**

**4.1.1.5 RFECV**

|  |  |  |  |
| --- | --- | --- | --- |
| **Features** | **RFECV with Random Forest** | **RFECV with Gradient Boosting** | **RFECV with LightGBM** |
| Agency | X | X | X |
| WorkflowStatus | X | X | X |
| Product |  | X | X |
| ProductGroup |  | X |  |
| ProductType |  | X | X |
| NoOfLives |  | X |  |
| CommDateProvided |  | X | X |
| UWDecision |  | X | X |
| BonusCommission |  | X | X |
| BonusCommissionPercentage |  | X | X |
| SignedDecReceived |  | X | X |
| CommissionSacrificePercentage |  | X | X |
| CommissionTerms |  | X | X |

**Figure 22 – RFECV by Model and Feature**

A graph with colorful lines

Description automatically generated

**Figure 23 – Random Forest RFECV – Cross-Validated ROC AUC vs. Number of Features**

A graph of a graph with different colored lines

Description automatically generated with medium confidence

**Figure 24 – Gradient Boosted Classifier RFECV – Cross-Validated ROC AUC vs. Number of Features**

A graph with colorful lines

Description automatically generated

**Figure 25 – Light GBM Classifier RFECV – Cross-Validated ROC AUC vs. Number of Features**

**4.1.1.6 Compare with and without Feature**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **Model** | **Control ROC AUC** | **Experiment ROC AUC** |
| Agency | Decision Tree | 0.6986 | 0.5948 |
| Random Forest | 0.7622 | 0.7276 |
| Gradient Boosting | 0.811 | 0.7908 |
| WorkflowStatus | Decision Tree | 0.6965 | 0.5582 |
| Random Forest | 0.7619 | 0.6006 |
| Gradient Boosting | 0.811 | 0.5962 |
| UWDecision | Decision Tree | 0.6985 | 0.6572 |
| Random Forest | 0.7619 | 0.7604 |
| Gradient Boosting | 0.811 | 0.8045 |
| CommDateProvided | Decision Tree | 0.6987 | 0.6944 |
| Random Forest | 0.7617 | 0.7541 |
| Gradient Boosting | 0.811 | 0.8075 |
| SignedDecReceived | Decision Tree | 0.6974 | 0.6994 |
| Random Forest | 0.7622 | 0.7574 |
| Gradient Boosting | 0.811 | 0.8086 |

**Figure 26 – Model ROC AUC Comparison with and without Selected Features**

|  |  |
| --- | --- |
| **Feature** | **ROC AUC p-value** |
| Agency | 0.1715 |
| WorkflowStatus | 0.976 |
| UWDecision | 0.0662 |
| CommDateProvided | 0.0635 |
| SignedDecReceived | 0.0577 |

**Figure 27 – Statistical Significane of Selected Features**

A screenshot of a graph

Description automatically generated

**Figure 28 – Model Performance with Feature Manipulation**

**4.1.2 Oversampled Training Data**

**4.1.2.1 Association Analysis**

|  |  |  |  |
| --- | --- | --- | --- |
| **PolicyIssued vs Feature** | **Cramer's V** | **Chi2 Statistic** | **P-value** |
| Product | 0.090355855 | 2194.900645 | 2.8559E-181 |
| ProductGroup | 0.067835172 | 1226.194876 | 1.172E-104 |
| ProductType | 0.078374602 | 1231.981091 | 6.0827E-241 |
| Agency | 0.338325369 | 63970.19619 | 0 |
| WorkflowStatus | 0.502656856 | 50046.46186 | 0 |
| Indexation | 0.021121711 | 284.3717897 | 4.56033E-05 |
| NoOfLives | 0.029293062 | 752.0526197 | 2.9376E-06 |
| CommDateProvided | 0.310985179 | 19001.3257 | 0 |
| PaymentFreq | 0.013002182 | 42.10862445 | 3.13964E-06 |
| UWDecision | 0.34806568 | 25370.33255 | 0 |
| ComissionSacrifice | 0.005793469 | 34.57347291 | 0.182625774 |
| CommissionSacrificeType | 0.006572057 | 32.45896158 | 0.116013744 |
| RenewalSacrificeType | 0.014835844 | 57.10556049 | 3.75042E-07 |
| CommissionSacrificePercentage | 0.115091257 | 12947.18872 | 5.11066E-63 |
| CommissionTerms | 0.024202963 | 223.722142 | 6.36419E-10 |
| Discount | 0.003621209 | 3.568124911 | 0.058898784 |
| BonusCommission | 0.006631597 | 118.6133597 | 0.270635393 |
| BonusCommissionPercentage | 0.100518442 | 11341.83708 | 3.632E-42 |
| FreeCover | 0.067736363 | 899.5697793 | 1.2172E-197 |
| SeriousIllnessType | 0.011751498 | 227.0464885 | 0.09196612 |
| SignedDecReceived | 0.110476513 | 2823.277783 | 0 |

**Figure 29 – Association Analysis, Oversampled Training Dataset**

**A screenshot of a computer screen

Description automatically generated**

**Figure 30 – Association Analysis P-Value Matrix, Oversampled Training Dataset**

**A screen shot of a computer

Description automatically generated**

**Figure 31 – Association Analysis Chi Squared Statistic Matrix, Oversampled Training Dataset**

**A screenshot of a computer screen

Description automatically generated**

**Figure 32 – Association Analysis Cramer’s V Matrix, Oversampled Training Dataset**

**4.1.2.2 Random Forest**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **n\_estimators: 50** | **n\_estimators: 100** | **n\_estimators: 200** |
| Agency | 0.427256 | 0.42763 | 0.428032 |
| WorkflowStatus | 0.239154 | 0.23987 | 0.238666 |
| BonusCommissionPercentage | 0.068366 | 0.068365 | 0.067978 |
| UWDecision | 0.06199 | 0.065162 | 0.072202 |
| CommDateProvided | 0.057668 | 0.053308 | 0.047973 |
| CommissionSacrificePercentage | 0.057188 | 0.057073 | 0.057039 |
| SignedDecReceived | 0.018066 | 0.017908 | 0.018131 |
| NoOfLives | 0.01451 | 0.015271 | 0.014955 |
| CommissionTerms | 0.008953 | 0.00879 | 0.008809 |
| SeriousIllnessType | 0.008596 | 0.008615 | 0.008558 |
| Indexation | 0.008116 | 0.007916 | 0.007829 |
| Product | 0.007305 | 0.007214 | 0.00722 |
| ProductGroup | 0.00515 | 0.00515 | 0.005058 |
| BonusCommission | 0.00374 | 0.003774 | 0.003847 |
| RenewalSacrificeType | 0.0026 | 0.002566 | 0.002543 |
| CommissionSacrificeType | 0.00239 | 0.002398 | 0.002357 |
| Discount | 0.002129 | 0.002186 | 0.002217 |
| ProductType | 0.002042 | 0.002009 | 0.001935 |
| ComissionSacrifice | 0.001942 | 0.001996 | 0.001955 |
| PaymentFreq | 0.00158 | 0.001591 | 0.001623 |
| FreeCover | 0.001261 | 0.001207 | 0.001076 |

**Figure 33 – Random Forest Feature Importance Scores**

**A graph with blue lines and dots

Description automatically generated**

**Figure 34 – Random Forest ROC AUC vs. No of Features (n Estimators: 50)**

**A screenshot of a graph

Description automatically generated**

**Figure 35 – Random Forest ROC AUC vs. No of Features (n Estimators: 100)**

**A screen shot of a graph

Description automatically generated**

**Figure 36 – Random Forest ROC AUC vs. No of Features (n Estimators: 200)**

**4.1.2.3 Gradient Boosting Classifier**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **n\_estimators: 50** | **n\_estimators: 100** | **n\_estimators: 200** |
| WorkflowStatus | 0.85146 | 0.819108 | 0.787252 |
| UWDecision | 0.047175 | 0.045731 | 0.045035 |
| CommDateProvided | 0.029293 | 0.029219 | 0.028415 |
| Agency | 0.025378 | 0.034882 | 0.050419 |
| SignedDecReceived | 0.021661 | 0.022202 | 0.022056 |
| BonusCommissionPercentage | 0.009086 | 0.026179 | 0.036314 |
| ProductGroup | 0.006151 | 0.006267 | 0.006296 |
| NoOfLives | 0.005301 | 0.007104 | 0.007569 |
| BonusCommission | 0.001152 | 0.001134 | 0.001179 |
| ProductType | 0.00092 | 0.001235 | 0.001488 |
| CommissionTerms | 0.000811 | 0.001619 | 0.002274 |
| ComissionSacrifice | 0.000774 | 0.001204 | 0.001532 |
| Product | 0.000597 | 0.000926 | 0.002598 |
| CommissionSacrificePercentage | 0.000224 | 0.002206 | 0.003787 |
| CommissionSacrificeType | 0.000016 | 0.000015 | 0.000015 |
| RenewalSacrificeType | 0 | 0.000315 | 0.000537 |
| Discount | 0 | 0.000422 | 0.000525 |
| PaymentFreq | 0 | 0 | 0.000152 |
| Indexation | 0 | 0.000063 | 0.000574 |
| FreeCover | 0 | 0 | 0 |

**Figure 37 – Gradient Boosting Classifier Feature Importance Scores**

A screen shot of a graph

Description automatically generated

**Figure 38 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 50)**

A screen shot of a graph

Description automatically generated

**Figure 39 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 100)**

A screen shot of a computer

Description automatically generated

**Figure 40 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 200)**

**4.1.2.4 LightGBM Classifier**

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **n\_estimators: 50** | **n\_estimators: 100** | **n\_estimators: 200** |
| WorkflowStatus | 0.787252 | 0.787252 | 0.787252 |
| Agency | 0.050419 | 0.050419 | 0.050419 |
| UWDecision | 0.045035 | 0.045035 | 0.045035 |
| BonusCommissionPercentage | 0.036314 | 0.036314 | 0.036314 |
| CommDateProvided | 0.028415 | 0.028415 | 0.028415 |
| SignedDecReceived | 0.022056 | 0.022056 | 0.022056 |
| NoOfLives | 0.007569 | 0.007569 | 0.007569 |
| ProductGroup | 0.006296 | 0.006296 | 0.006296 |
| CommissionSacrificePercentage | 0.003787 | 0.003787 | 0.003787 |
| Product | 0.002598 | 0.002598 | 0.002598 |
| CommissionTerms | 0.002274 | 0.002274 | 0.002274 |
| SeriousIllnessType | 0.001983 | 0.001983 | 0.001983 |
| ComissionSacrifice | 0.001532 | 0.001532 | 0.001532 |
| ProductType | 0.001488 | 0.001488 | 0.001488 |
| BonusCommission | 0.001179 | 0.001179 | 0.001179 |
| Indexation | 0.000574 | 0.000574 | 0.000574 |
| RenewalSacrificeType | 0.000537 | 0.000537 | 0.000537 |
| Discount | 0.000525 | 0.000525 | 0.000525 |
| PaymentFreq | 0.000152 | 0.000152 | 0.000152 |
| CommissionSacrificeType | 0.000015 | 0.000015 | 0.000015 |
| FreeCover | 0 | 0 | 0 |

**Figure 41 – Light GBM Classifier Feature Importance Scores**

A screen shot of a computer

Description automatically generated

**Figure 42 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 50)**

A screen shot of a computer

Description automatically generated

**Figure 42 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 100)**

A screen shot of a computer

Description automatically generated

**Figure 42 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 200)**

**4.1.2.5 Class Aware Feature Importance**

|  |  |
| --- | --- |
| **Feature** | **MDRAUC** |
| CommDateProvided | 0.0382 |
| WorkflowStatus | 0.0329 |
| SignedDecReceived | 0.0253 |
| CommissionSacrificePercentage | 0.0093 |
| BonusCommissionPercentage | 0.0039 |
| BonusCommission | 0.0032 |
| Indexation | 0.0022 |
| ComissionSacrifice | 0.0017 |
| Agency | 0 |
| Product | 0 |
| CommissionSacrificeType | 0 |
| ProductGroup | 0 |
| NoOfLives | 0 |
| PaymentFreq | 0 |
| Discount | 0 |
| FreeCover | 0 |
| SeriousIllnessType | -0.0001 |
| RenewalSacrificeType | -0.0004 |
| CommissionTerms | -0.0006 |
| ProductType | -0.0016 |
| UWDecision | -0.0513 |

**Figure 43 – Feature Weighted Importance Scores**

**A white and blue rectangle with black border

Description automatically generated**

**Figure 44 – Feature Weighted Importance Scores Bar Chart**

**4.1.2.6 RFECV**

|  |  |  |  |
| --- | --- | --- | --- |
| **Features** | **RFECV with Random Forest** | **RFECV with Gradient Boosting** | **RFECV with LightGBM** |
| Agency | X | X | X |
| WorkflowStatus | X | X | X |
| Product |  | X | X |
| ProductGroup |  | X | X |
| ProductType |  | X | X |
| Indexation |  | X | X |
| NoOfLives |  | X | X |
| CommDateProvided |  | X | X |
| UWDecision |  | X | X |
| ComissionSacrifice |  | X | X |
| CommissionSacrificePercentage |  | X | X |
| CommissionTerms |  | X | X |
| BonusCommissionPercentage |  | X | X |
| SeriousIllnessType |  | X | X |
| SignedDecReceived |  | X | X |

**Figure 45 – RFECV by Model and Feature**

A graph with different colored lines

Description automatically generated

**Figure 46 – Random Forest RFECV – Cross-Validated ROC AUC vs. Number of Features**

A graph with different colored lines

Description automatically generated

**Figure 47 – Gradient Boosted RFECV – Cross-Validated ROC AUC vs. Number of Features**

A graph with colored lines

Description automatically generated

**Figure 48 – Light GBM RFECV – Cross-Validated ROC AUC vs. Number of Features**

**4.2 Machine Learning Models & Hyperparameter Tuning**

**4.2.1 Imbalanced Training Data**

**4.2.1.1 All Features**

**4.2.1.1.1 Logistic Regression**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'C': 0.001, 'penalty': 'l1'} | nan |
| {'C': 0.001, 'penalty': 'l2'} | 0.79 |
| {'C': 0.01, 'penalty': 'l1'} | nan |
| {'C': 0.01, 'penalty': 'l2'} | 0.79 |
| {'C': 0.1, 'penalty': 'l1'} | nan |
| {'C': 0.1, 'penalty': 'l2'} | 0.78 |
| {'C': 1, 'penalty': 'l1'} | nan |
| {'C': 1, 'penalty': 'l2'} | 0.79 |
| {'C': 10, 'penalty': 'l1'} | nan |
| {'C': 10, 'penalty': 'l2'} | 0.79 |

**Figure 49 – Logistic Regression Mean Cross Validation Score by Hyperparameter Combination**

**4.2.1.1.2 Decision Tree**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'max\_depth': None, 'min\_samples\_split': 2} | 0.69 |
| {'max\_depth': None, 'min\_samples\_split': 5} | 0.7 |
| {'max\_depth': None, 'min\_samples\_split': 10} | 0.72 |
| {'max\_depth': 10, 'min\_samples\_split': 2} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 5} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 10} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 2} | 0.73 |
| {'max\_depth': 20, 'min\_samples\_split': 5} | 0.74 |
| {'max\_depth': 20, 'min\_samples\_split': 10} | 0.75 |
| {'max\_depth': 30, 'min\_samples\_split': 2} | 0.69 |
| {'max\_depth': 30, 'min\_samples\_split': 5} | 0.71 |
| {'max\_depth': 30, 'min\_samples\_split': 10} | 0.73 |

**Figure 50 – Decision Tree Mean Cross Validation Score by Hyperparameter Combination**

**4.1.1.1.3 Random Forest**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'max\_depth': None, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.76 |
| {'max\_depth': None, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.76 |
| {'max\_depth': None, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.76 |
| {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.77 |
| {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.77 |
| {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.77 |
| {'max\_depth': None, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.78 |
| {'max\_depth': None, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.78 |
| {'max\_depth': None, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.78 |
| {'max\_depth': 10, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.81 |
| {'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.79 |
| {'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.79 |
| {'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.79 |
| {'max\_depth': 20, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.76 |
| {'max\_depth': 30, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.77 |
| {'max\_depth': 30, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.77 |
| {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.77 |
| {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.77 |
| {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.77 |
| {'max\_depth': 30, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.78 |
| {'max\_depth': 30, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.78 |
| {'max\_depth': 30, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.78 |

**Figure 51 – Random Forest Mean Cross Validation Score by Hyperparameter Combination**

**4.1.1.1.4 Gradient Boosting Classifier**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 100} | 0.8 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 200} | 0.8 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 100} | 0.8 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 300} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 300} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | 0.81 |

**Figure 52 – Gradient Boosted Classifier Mean Cross Validation Score by Hyperparameter Combination**

**4.1.1.1.5 LightGBM**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 300} | 0.82 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 300} | 0.82 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 300} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 300} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | 0.81 |

**Figure 53 – Light GBM Classifier Mean Cross Validation Score by Hyperparameter Combination**

**4.1.1.1.6 Model Validation**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Best Hyperparameters** | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **ROC AUC Score** |
| Logistic Regression | {'C': 10, 'penalty': 'l2'} | 0.78 | 0.79 | 0.98 | 0.88 | 0.54 |
| Decision Tree | {'max\_depth': 10, 'min\_samples\_split': 10} | 0.8 | 0.81 | 0.96 | 0.88 | 0.6 |
| Random Forest | {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.8 | 0.81 | 0.97 | 0.88 | 0.58 |
| Gradient Boosting | {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 300} | 0.8 | 0.82 | 0.95 | 0.88 | 0.62 |
| LightGBM | {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 100} | 0.81 | 0.83 | 0.95 | 0.88 | 0.63 |

**Figure 54 – Model Performances on Validation Dataset**

A close-up of a color chart

Description automatically generated

**Figure 55 – Accuracy & ROC AUC Validation Comparison by Model**

**4.1.1.1.7 Evaluation on Unseen Data**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Best Hyperparameters** | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **ROC AUC Score** |
| Logistic Regression | {'C': 10, 'penalty': 'l2'} | .76 | .83 | .87 | .85 | .72 |
| Decision Tree | {'max\_depth': 10, 'min\_samples\_split': 10} | .77 | .83 | .9 | .86 | .78 |
| Random Forest | {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 100} | .79 | .82 | .94 | .88 | .80 |
| Gradient Boosting | {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 300} | .79 | .83 | .92 | .87 | .78 |
| LightGBM | {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 100} | .81 | .83 | .94 | .88 | .82 |

**Figure 56 – Model Performances on Unseen Test Dataset**

A blue and white rectangular object

Description automatically generated

**Figure 57 – Model Accuracy Comparison on Unseen Test Dataset**

A pink rectangular object with white lines

Description automatically generated

**Figure 58 – Model ROC AUC Comparison on Unseen Test Dataset**

**4.2.1.2 Selected Features**

**4.2.1.2.1 Logistic Regression**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'C': 0.001, 'penalty': 'l1'} | nan |
| {'C': 0.001, 'penalty': 'l2'} | 0.79 |
| {'C': 0.01, 'penalty': 'l1'} | nan |
| {'C': 0.01, 'penalty': 'l2'} | 0.79 |
| {'C': 0.1, 'penalty': 'l1'} | nan |
| {'C': 0.1, 'penalty': 'l2'} | 0.79 |
| {'C': 1, 'penalty': 'l1'} | nan |
| {'C': 1, 'penalty': 'l2'} | 0.79 |
| {'C': 10, 'penalty': 'l1'} | nan |
| {'C': 10, 'penalty': 'l2'} | 0.79 |

**Figure 59 – Logistic Regression Mean Cross Validation Score by Hyperparameter Combination**

**4.2.1.2.2 Decision Tree**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'max\_depth': None, 'min\_samples\_split': 2} | 0.8 |
| {'max\_depth': None, 'min\_samples\_split': 5} | 0.8 |
| {'max\_depth': None, 'min\_samples\_split': 10} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 2} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 5} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 10} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 2} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 5} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 10} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 2} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 5} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 10} | 0.8 |

**Figure 60 – Decision Tree Mean Cross Validation Score by Hyperparameter Combination**

**4.2.1.2.3 Random Forest**

|  |  |
| --- | --- |
|  |  |
| **Hyperparameters** | **Mean CV Score** |
| {'max\_depth': None, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': None, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': None, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': None, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': None, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': None, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.8 |

**Figure 61 – Random Forest Mean Cross Validation Score by Hyperparameter Combination**

**4.2.1.2.4 Gradient Boosting Classifier**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 100} | 0.8 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 200} | 0.8 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 100} | 0.8 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 300} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 300} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 300} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 300} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 300} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | 0.82 |

**Figure 62 – Gradient Boosting Mean Cross Validation Score by Hyperparameter Combination**

**4.2.1.2.5 Light GBM**

|  |
| --- |
|  |
| **Hyperparameters** | | **Mean CV Score** |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 100} | | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 200} | | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 300} | | 0.82 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 100} | | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 200} | | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 300} | | 0.82 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 100} | | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 200} | | 0.82 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 300} | | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 100} | | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 200} | | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 300} | | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 100} | | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 200} | | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 300} | | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 100} | | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 200} | | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 300} | | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 100} | | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 200} | | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 300} | | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 100} | | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 200} | | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 300} | | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 100} | | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 200} | | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | | 0.81 |

**Figure 63 – Light GBM Mean Cross Validation Score by Hyperparameter Combination**

**4.2.1.2.6 Model Validation**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Best Hyperparameters** | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **ROC AUC Score** |
| Logistic Regression | {'C': 0.1, 'penalty': 'l2'} | 0.78 | 0.8 | 0.97 | 0.87 | 0.55 |
| Decision Tree | {'max\_depth': 20, 'min\_samples\_split': 10} | 0.8 | 0.84 | 0.92 | 0.88 | 0.65 |
| Random Forest | {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.8 | 0.84 | 0.93 | 0.88 | 0.65 |
| Gradient Boosting Classifier | {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 300} | 0.81 | 0.83 | 0.94 | 0.88 | 0.63 |
| LightGBM | {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 100} | 0.81 | 0.83 | 0.95 | 0.89 | 0.63 |

**Figure 64 – Model Performances on Validation Dataset**

**4.2.1.2.7 Evaluation on Unseen Data**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Best Hyperparameters** | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **ROC AUC Score** |
| Logistic Regression | {'C': 0.1, 'penalty': 'l2'} | .69 | .83 | .76 | .79 | .72 |
| Decision Tree | {'max\_depth': 20, 'min\_samples\_split': 10} | .81 | .8 | .84 | .83 | .88 |
| Random Forest | {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 200} | .80 | .83 | .93 | .88 | .81 |
| Gradient Boosting Classifier | {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 300} | .8 | .83 | .94 | .88 | .81 |
| LightGBM | {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 100} | .8 | .82 | .97 | .88 | .81 |

**Figure 65 – Model Performances on Unseen Test Dataset**

**A blue and white bar chart

Description automatically generated**

**Figure 66 – Model Accuracy Comparison on Unseen Test Dataset**

**A pink bars with white stripes

Description automatically generated with medium confidence**

**Figure 67 – Model ROC AUC Comparison on Unseen Test Dataset**

**4.2.2 Oversampled Dataset**

**4.2.2.1 All Features**

**4.2.2.1.1 Logistic Regression**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'C': 0.001, 'penalty': 'l1'} | nan |
| {'C': 0.001, 'penalty': 'l2'} | 0.77 |
| {'C': 0.01, 'penalty': 'l1'} | nan |
| {'C': 0.01, 'penalty': 'l2'} | 0.77 |
| {'C': 0.1, 'penalty': 'l1'} | nan |
| {'C': 0.1, 'penalty': 'l2'} | 0.77 |
| {'C': 1, 'penalty': 'l1'} | nan |
| {'C': 1, 'penalty': 'l2'} | 0.77 |
| {'C': 10, 'penalty': 'l1'} | nan |
| {'C': 10, 'penalty': 'l2'} | 0.77 |

**Figure 68 – Logistic Regression Mean Cross Validation Score by Hyperparameter Combination**

**4.2.2.1.2 Decision Tree**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'max\_depth': None, 'min\_samples\_split': 2} | 0.79 |
| {'max\_depth': None, 'min\_samples\_split': 5} | 0.8 |
| {'max\_depth': None, 'min\_samples\_split': 10} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 2} | 0.79 |
| {'max\_depth': 10, 'min\_samples\_split': 5} | 0.79 |
| {'max\_depth': 10, 'min\_samples\_split': 10} | 0.79 |
| {'max\_depth': 20, 'min\_samples\_split': 2} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 5} | 0.81 |
| {'max\_depth': 20, 'min\_samples\_split': 10} | 0.81 |
| {'max\_depth': 30, 'min\_samples\_split': 2} | 0.8 |
| {'max\_depth': 30, 'min\_samples\_split': 5} | 0.81 |
| {'max\_depth': 30, 'min\_samples\_split': 10} | 0.81 |

**Figure 69 – Decision Tree Mean Cross Validation Score by Hyperparameter Combination**

**4.2.2.1.3 Random Forest**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'max\_depth': None, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.85 |
| {'max\_depth': None, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.85 |
| {'max\_depth': None, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.85 |
| {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.85 |
| {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.85 |
| {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.85 |
| {'max\_depth': None, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.85 |
| {'max\_depth': None, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.85 |
| {'max\_depth': None, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.85 |
| {'max\_depth': 10, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.8 |
| {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.8 |
| {'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.84 |
| {'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.84 |
| {'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.84 |
| {'max\_depth': 20, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.84 |
| {'max\_depth': 20, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.84 |
| {'max\_depth': 20, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.84 |
| {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.84 |
| {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.84 |
| {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.84 |
| {'max\_depth': 30, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.85 |
| {'max\_depth': 30, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.85 |
| {'max\_depth': 30, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.85 |
| {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.85 |
| {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.85 |
| {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.85 |
| {'max\_depth': 30, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.85 |
| {'max\_depth': 30, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.85 |
| {'max\_depth': 30, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.85 |

**Figure 70 – Random Forest Mean Cross Validation Score by Hyperparameter Combination**

**4.2.2.1.4 Gradient Boosting Classifier**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 100} | 0.78 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 200} | 0.78 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 300} | 0.79 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 100} | 0.78 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 200} | 0.79 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 300} | 0.79 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 100} | 0.79 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 200} | 0.79 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 300} | 0.8 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 100} | 0.8 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 300} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 300} | 0.83 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 200} | 0.83 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 300} | 0.84 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 300} | 0.83 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 200} | 0.84 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 300} | 0.85 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 100} | 0.83 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 200} | 0.85 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | 0.86 |

**Figure 71 – Gradient Boosting Mean Cross Validation Score by Hyperparameter Combination**

**4.2.2.1.5 Light GBM**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 100} | 0.8 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 300} | 0.83 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 200} | 0.83 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 300} | 0.84 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 200} | 0.83 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 300} | 0.84 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 100} | 0.86 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 200} | 0.87 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 300} | 0.88 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 100} | 0.87 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 200} | 0.88 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 300} | 0.89 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 100} | 0.87 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 200} | 0.89 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 300} | 0.89 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 100} | 0.88 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 200} | 0.88 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 300} | 0.89 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 100} | 0.88 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 200} | 0.89 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 300} | 0.89 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 100} | 0.89 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 200} | 0.89 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | 0.9 |

**Figure 72 – Light GBM Mean Cross Validation Score by Hyperparameter Combination**

**4.2.2.1.6 Model Validation**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Best Hyperparameters** | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **ROC AUC Score** |
| Logistic Regression | {'C': 0.1, 'penalty': 'l2'} | 0.68 | 0.94 | 0.62 | 0.75 | 0.75 |
| Decision Tree | {'max\_depth': 30, 'min\_samples\_split': 10} | 0.71 | 0.89 | 0.71 | 0.79 | 0.71 |
| Random Forest | {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.72 | 0.89 | 0.73 | 0.81 | 0.71 |
| Gradient Boosting | {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | 0.72 | 0.92 | 0.7 | 0.79 | 0.74 |
| LightGBM | {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | 0.22 | 0.88 | 0 | 0.01 | 0.5 |

**Figure 73 – Model Performances on Validation Dataset**

**A close-up of a color chart

Description automatically generated**

**Figure 74 – Accuracy & ROC AUC Validation Comparison by Model**

**4.2.2.1.7 Evaluation on Unseen Data**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Best Hyperparameters** | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **ROC AUC Score** |
| Logistic Regression | {'C': 0.1, 'penalty': 'l2'} | .62 | .95 | .55 | .69 | .71 |
| Decision Tree | {'max\_depth': 30, 'min\_samples\_split': 10} | .7 | .89 | .7 | .78 | .72 |
| Random Forest | {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 300} | .73 | .89 | .75 | .81 | .78 |
| Gradient Boosting | {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | .72 | .89 | .74 | .8 | .78 |
| LightGBM | {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | .77 | .88 | .81 | .85 | .8 |

**Figure 75 – Model Performances on Unseen Test Dataset**

**A blue and white bar chart

Description automatically generated**

**Figure 76 – Model Accuracy Comparison on Unseen Test Dataset**

**A pink and white bar chart

Description automatically generated**

**Figure 77 – Model ROC AUC Comparison on Unseen Test Dataset**

**4.2.2.2 Selected Features**

**4.2.2.2.1 Logistic Regression**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'C': 0.001, 'penalty': 'l1'} | nan |
| {'C': 0.001, 'penalty': 'l2'} | 0.77 |
| {'C': 0.01, 'penalty': 'l1'} | nan |
| {'C': 0.01, 'penalty': 'l2'} | 0.77 |
| {'C': 0.1, 'penalty': 'l1'} | nan |
| {'C': 0.1, 'penalty': 'l2'} | 0.77 |
| {'C': 1, 'penalty': 'l1'} | nan |
| {'C': 1, 'penalty': 'l2'} | 0.77 |
| {'C': 10, 'penalty': 'l1'} | nan |
| {'C': 10, 'penalty': 'l2'} | 0.77 |

**Figure 78 – Logistic Regression Mean Cross Validation Score by Hyperparameter Combination**

**4.2.2.2.2 Decision Tree**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'max\_depth': None, 'min\_samples\_split': 2} | 0.83 |
| {'max\_depth': None, 'min\_samples\_split': 5} | 0.83 |
| {'max\_depth': None, 'min\_samples\_split': 10} | 0.83 |
| {'max\_depth': 10, 'min\_samples\_split': 2} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 5} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 10} | 0.81 |
| {'max\_depth': 20, 'min\_samples\_split': 2} | 0.83 |
| {'max\_depth': 20, 'min\_samples\_split': 5} | 0.83 |
| {'max\_depth': 20, 'min\_samples\_split': 10} | 0.83 |
| {'max\_depth': 30, 'min\_samples\_split': 2} | 0.83 |
| {'max\_depth': 30, 'min\_samples\_split': 5} | 0.83 |
| {'max\_depth': 30, 'min\_samples\_split': 10} | 0.83 |

**Figure 79 – Decision Tree Mean Cross Validation Score by Hyperparameter Combination**

**4.2.2.2.3 Random Forest**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'max\_depth': None, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.83 |
| {'max\_depth': None, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.83 |
| {'max\_depth': None, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.83 |
| {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.83 |
| {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.83 |
| {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.83 |
| {'max\_depth': None, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.83 |
| {'max\_depth': None, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.83 |
| {'max\_depth': None, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.83 |
| {'max\_depth': 10, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.81 |
| {'max\_depth': 10, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.81 |
| {'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.83 |
| {'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.83 |
| {'max\_depth': 20, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.83 |
| {'max\_depth': 20, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.83 |
| {'max\_depth': 20, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.83 |
| {'max\_depth': 20, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.83 |
| {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.83 |
| {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.83 |
| {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.83 |
| {'max\_depth': 30, 'min\_samples\_split': 2, 'n\_estimators': 100} | 0.83 |
| {'max\_depth': 30, 'min\_samples\_split': 2, 'n\_estimators': 200} | 0.83 |
| {'max\_depth': 30, 'min\_samples\_split': 2, 'n\_estimators': 300} | 0.83 |
| {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.83 |
| {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 200} | 0.83 |
| {'max\_depth': 30, 'min\_samples\_split': 5, 'n\_estimators': 300} | 0.83 |
| {'max\_depth': 30, 'min\_samples\_split': 10, 'n\_estimators': 100} | 0.83 |
| {'max\_depth': 30, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.83 |
| {'max\_depth': 30, 'min\_samples\_split': 10, 'n\_estimators': 300} | 0.83 |

**Figure 80 – Random Forest Mean Cross Validation Score by Hyperparameter Combination**

**4.2.2.2.4 Gradient Boosting Classifier**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 100} | 0.78 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 200} | 0.78 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 300} | 0.79 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 100} | 0.79 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 200} | 0.79 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 300} | 0.79 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 100} | 0.79 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 200} | 0.8 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 300} | 0.8 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 100} | 0.8 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 200} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 300} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 300} | 0.83 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 200} | 0.83 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 300} | 0.84 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 300} | 0.83 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 200} | 0.84 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 300} | 0.85 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 100} | 0.83 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 200} | 0.85 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | 0.86 |

**Figure 81 – Gradient Boosting Mean Cross Validation Score by Hyperparameter Combination**

**4.2.2.2.5 Light GBM**

|  |  |
| --- | --- |
| **Hyperparameters** | **Mean CV Score** |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 100} | 0.8 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 200} | 0.82 |
| {'learning\_rate': 0.01, 'max\_depth': 3, 'n\_estimators': 300} | 0.83 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 100} | 0.81 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 200} | 0.83 |
| {'learning\_rate': 0.01, 'max\_depth': 4, 'n\_estimators': 300} | 0.84 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 100} | 0.82 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 200} | 0.84 |
| {'learning\_rate': 0.01, 'max\_depth': 5, 'n\_estimators': 300} | 0.85 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 100} | 0.86 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 200} | 0.88 |
| {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 300} | 0.88 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 100} | 0.87 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 200} | 0.88 |
| {'learning\_rate': 0.1, 'max\_depth': 4, 'n\_estimators': 300} | 0.89 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 100} | 0.88 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 200} | 0.89 |
| {'learning\_rate': 0.1, 'max\_depth': 5, 'n\_estimators': 300} | 0.89 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 100} | 0.88 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 200} | 0.88 |
| {'learning\_rate': 0.2, 'max\_depth': 3, 'n\_estimators': 300} | 0.89 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 100} | 0.88 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 200} | 0.89 |
| {'learning\_rate': 0.2, 'max\_depth': 4, 'n\_estimators': 300} | 0.9 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 100} | 0.89 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 200} | 0.9 |
| {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | 0.9 |

**Figure 82 – Light GBM Mean Cross Validation Score by Hyperparameter Combination**

**4.2.2.2.6 Model Validation**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Hyperparameters** | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **ROC AUC Score** |
| Logistic Regression | {'C': 1, 'penalty': 'l2'} | 0.68 | 0.94 | 0.62 | 0.75 | 0.74 |
| Decision Tree | {'max\_depth': None, 'min\_samples\_split': 5} | 0.71 | 0.91 | 0.69 | 0.79 | 0.73 |
| Random Forest | {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 100} | 0.71 | 0.91 | 0.7 | 0.79 | 0.72 |
| Gradient Boosting | {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | 0.72 | 0.92 | 0.69 | 0.79 | 0.74 |
| LightGBM | {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | 0.23 | 0.94 | 0.01 | 0.02 | 0.5 |

**Figure 83 – Model Performances on Validation Dataset**

**4.2.2.2.7 Evaluation on Unseen Data**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Hyperparameters** | **Accuracy** | **Precision** | **Recall** | **F1 Score** | **ROC AUC Score** |
| Logistic Regression | {'C': 1, 'penalty': 'l2'} | .61 | .95 | .53 | .68 | .71 |
| Decision Tree | {'max\_depth': None, 'min\_samples\_split': 5} | .71 | .91 | .69 | .79 | .77 |
| Random Forest | {'max\_depth': None, 'min\_samples\_split': 5, 'n\_estimators': 100} | .71 | .91 | .7 | .79 | .77 |
| Gradient Boosting | {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | .22 | 1 | 0 | 0 | .42 |
| LightGBM | {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | .22 | 1 | 0 | 0 | .5 |

**Figure 84 – Model Performances on Unseen Test Dataset**

**A blue and white bar chart

Description automatically generated**

**Figure 85 – Model Accuracy Comparison on Unseen Test Dataset**

**A graph of pink bars

Description automatically generated with medium confidence**

**Figure 86 – Model ROC AUC Comparison on Unseen Test Dataset**

1. **Discussion**

**5.1 Feature Correlation And Feature Importance**

**5.1.1 Imbalanced Dataset**

The results of the feature correlation and feature importance investigations and experiments shed light on the relationship between various features and the predicted conversion of a life assurance application into an active policy. All independent features were analysed for their Cramer's V values and significance levels in the association analysis. These findings show which factors have a strong relationship with conversion, with higher Cramer's V values and lower p-values suggesting greater relevance. The Cramer’s V values and chi-squared statistics indicate significant connections between particular variables and policy issuance. WorkflowStatus, Agency, and CommDateProvided display particlarly strong connections, as evidenced by low p-values and relatively high Cramer’s V values. This would indicate that these features are important in determining whether or not an application is converted into an active policy.

Furthermore, feature relevance scores were computed using Random Forest, Gradient Boosting, and LightGBM classifiers with varying numbers of estimators. Features with higher significance ratings are judged essential for prediction performance across various models. Across all models and estimator settings, Agency constantly stands out as one of the the most important feature. WorkflowStatus and UWDecision are similarly highly ranked, indicating that they have a significant impact on the predictive performance of these models.

The Class Aware Feature Importance analysis emphasises the impact of each feature, with positive scores suggesting a positive contribution to model performance and negative scores indicating a negative contribution. The results of this method highlights the positive and negative implications of features. WorkflowStatus, Agency, and UWDecision have a favourable impact, but ComissionSacrifice, ProductGroup, and Product have a negative impact.

The results of Recursive Feature Elimination with Cross-Validation (RFECV) highlight the features chosen by various methods, assisting in the identification of the most relevant characteristics for each model. This further emphasises the relevance of Agency, WorkflowStatus, and some other characteristics in the modelling process.

Finally, the comparison of feature influence on model performance, as evaluated by ROC AUC, reveals which features have a substantial impact on model results. Low p-values for ROC AUC differences suggest characteristics that have a significant impact on model performance. WorkflowStatus, for example, considerably improves ROC AUC values across multiple models when added, suggesting its significance in forecasting policy issuance.

When the results of all the selected feature importance methods are considered, there is evidence to state that WorkflowStatus, Agency, UWDecision, and CommDateProvided are among the most significant factors influencing the conversion of life assurance applications. These insights can help drive decision-making and feature selection in the life assurance area in order to create reliable prediction models.

**5.1.2 Oversampled Dataset**

The evaluation of the impact of independent features on the conversion of life assurance applications yields substantial insights from a variety of analytical methodologies. Cramer’s V and Chi2 Statistic association reveals strong associations between certain features and application conversion. WorkflowStatus and Agency are the most closely associated, followed by UWDecision.

Consistent feature importance rankings are identified by machine learning methods such as Random Forest, Gradient Boosting, and LightGBM. These models’ key aspects are Agency, WorkflowStatus, BonusCommissionPercentage, UWDecision, CommDateProvided, and CommissionSacrificePercentage. Class Aware Feature Importance confirms the importance of CommDateProvided, WorkflowStatus, and SignedDecReceived but implies that UWDecision, ProductType, and CommissionTerms have a negative influence. Agency and WorkflowStatus are consistently identified as important features using Recursive Feature Elimination with Cross-Validation.

Between these feature importance identification methods, there is support to state that WorkflowStatus and Agency have a considerable impact on application conversion, but other features such as UWDecision, CommDateProvided, and BonusCommissionPercentage are also important. These findings are useful in developing an accurate prediction model for life assurance application conversion.

Correlation analysis is an important technique in data analysis, although it has several limitations. When utilising correlation analysis, Gogtay & Thatte (2017) underline the need of accounting for restrictions such as repeated measurements, outliers, and non-linear connections. Sample size is also important for obtaining statistically significant findings. It is critical to realise that correlation coefficients describe relationships between variables but do not indicate causation, as Senthilnathan (2019) emphasises.

The role of feature importance in machine learning has grown Saarela & Jauhiainen (2021). It extends beyond classification results, providing insights into the value of distinct traits in various tasks. To measure feature significance, many approaches such as RFE can be employed. Feature selection is a useful strategy for decreasing data dimensionality in machine learning, but it must be used with caution. According to Heinze & Dunkler (2016), overly-complicated selection strategies can create needless complexity and have an impact on statistical tools such as P-values. They argue for specialist knowledge in data analysis to achieve a balance between complexity and interpretability.

The insights gained from this research coincide with various important aspects highlighted in the existing literature on correlation analysis, feature importance, and feature selection, while also adding new insights and nuances. The results demonstrate the relevance of particular features (WorkflowStatus, Agency, UWDecision, CommDateProvided) in determining the conversion of life assurance applications into live policies. This is consistent with the prior research’s discussion of feature importance, which emphasises the impact of certain features in model prediction. The use of machine learning techniques to estimate feature significance is consistent with previous research. Because of their efficacy, these algorithms are often used for feature selection and significance ranking. The use of Class Aware Feature Importance to measure the influence of features on model performance is consistent with the discussion of various techniques of assessing feature significance. It emphasises both the positive and negative impacts of features, offering a more nuanced perspective. Using RFECV to determine the most important features for each model relates to the already described feature selection idea. It aids in narrowing down the most significant features for modelling.

Overall, the results of the research are consistent with previous literature on feature importance and correlation analysis, but they also add useful expansions and nuances. They emphasise the significance of addressing dataset balance, the varying impact of features, and the impact of various analytic approaches. These findings add to a better understanding of feature importance in the context of predicting life assurance application conversion.

A comprehensive set of analyses and approaches efficiently addresses the research problem, which is based on understanding the factors impacting the conversion of life assurance applications. The research objectives, which are to analyse the influence of independent characteristics on application conversion, are achieved systematically using a variety of investigative methodologies.

Using Cramer's V values and significance levels, the study begins with association analysis and digs into the links between features and application conversion. By determining which variables are highly related with conversion likelihood, these findings directly contribute to the research objectives. WorkflowStatus, Agency, UWDecision, and CommDateProvided are underlined as critical criteria in deciding conversion outcome.

Machine learning methods such as Random Forest, Gradient Boosting, and LightGBM are used to study feature importance further. The consistent feature significance rankings across these models give clear insights into which factors have the greatest influence on prediction performance. The variables Agency, WorkflowStatus, BonusCommissionPercentage, UWDecision, CommDateProvided, and CommissionSacrificePercentage constantly emerge as essential features, meeting the research goals connected to feature impact evaluation directly.

The examination of Class Aware Feature Importance adds to the attainment of the research objectives by stressing the effects of each feature on model performance. This analysis highlights the favourable contributions of WorkflowStatus, Agency, and UWDecision, as well as the negative implications of features such as ComissionSacrifice, ProductGroup, and Product.

Recursive Feature Elimination with Cross-Validation increases the attainment of research objectives by consistently identifying Agency and WorkflowStatus as important features across several machine learning models.

Furthermore, the comparison of feature impact on model performance, as measured by ROC AUC, emphasises the relevance of WorkflowStatus, Agency, UWDecision, and CommDateProvided in predicting policy issuance. These findings are directly related to the research objectives, proving the importance of these features in determining conversion outcomes. Notably, the research approaches the research problem and objectives from both imbalanced and balanced dataset perspectives, assuring the findings’ robustness and application in many settings.

Ultimately, the research effectively addresses the research problem and objectives using a multifaceted strategy that includes association analysis, feature importance assessment, class-aware analysis, recursive feature elimination, and model performance evaluation. The findings continuously highlight the importance of WorkflowStatus, Agency, UWDecision, and CommDateProvided in predicting the conversion of life assurance applications, directly contributing to the research’s overall objectives.

**5.2 Machine Learning Models & Hyperparameter Tuning**

**5.2.1 Original Imbalanced Dataset**

**5.2.1.1 All Features**

The performance of various machine learning models and their hyperparameters, especially in relation to ROC AUC scores, is critical for determining their suitability for solving the research problem and objectives.

The performance of Logistic Regression varies depending on the hyperparameters used. On the original imbalanced dataset, the ROC AUC score varies from 0.78 to 0.79. C: 10, penalty: l2 is the best-performing hyperparameter configuration. On unseen data, however, the ROC AUC value decreases to 0.72, showing a considerable reduction in generalisation performance. This shows that, while Logistic Regression showed some promise on the initial dataset, it may require more tuning or different methodologies to properly handle unseen data.

The performance of the Decision Tree model is influenced by hyperparameters such as max\_depth and min\_samples\_split. With the hyperparameters max\_depth: 10 and min\_samples\_split: 10, the best ROC AUC score of 0.8 is obtained. With a ROC AUC score of 0.78, this model performs well on the original dataset but has slight limits in terms of generalisation to new data. It suggests that Decision Tree performs quite well but might benefit from more generalisation enhancing approaches.

When compared to the Decision Tree, the Random Forest outperforms it. With the hyperparameters max\_depth: 10, min\_samples\_split: 10, and n\_estimators: 100, the best ROC AUC score of 0.81 is attained. On unseen data, it maintains a reasonably good ROC AUC score of 0.80, showing greater generalisation ability. Given its balanced performance on both the original dataset and unseen data, Random Forest looks to be a suitable candidate for handling the research problem.

On the original dataset, the Gradient Boosting Classifier works well, with ROC AUC values ranging from 0.8 to 0.82. learning\_rate: 0.1, max\_depth: 4, n\_estimators: 300 is the optimum hyperparameter choice. Furthermore, on unseen data, this model retains a strong ROC AUC score of 0.78. Gradient Boosting is a suitable alternative for handling the research problem and objectives since it has significant generalisation capabilities.

LightGBM routinely outperforms its competitors, with ROC AUC scores ranging from 0.81 to 0.82 on the original dataset. learning\_rate: 0.1, max\_depth: 3, and n\_estimators: 100 are the optimum hyperparameter settings. LightGBM earns a great ROC AUC score of 0.82 on unseen data, demonstrating its exceptional generalisation skills. Given its constant good performance on both the training and unseen datasets, this model appears as a very viable alternative for tackling the research problem.

Overall, when evaluating the effectiveness of machine learning models using ROC AUC values, Logistic Regression shows certain limits in generalisation to unseen data. Although Decision Tree and Random Forest perform well, they may require additional measures to improve generalisation. Gradient Boosting Classifier and LightGBM, on the other hand, stand out as having good potential, continuously doing well on both the original dataset and unseen data. In addressing the research problem and objectives, these models demonstrate robustness and effectiveness.

**5.2.1.2 Selected Features**

On the original unbalanced dataset with selected features, Logistic Regression produces consistent ROC AUC values of roughly 0.79 across multiple hyperparameters. While these results are fairly consistent, Logistic Regression has room for improvement. On unknown data, it maintains a reasonable ROC AUC score of around 0.72, although there is some reduction in performance, indicating a problem with generalisation. This shows that further hyperparameter tuning or resolving class imbalance may help Logistic Regression.

On the original dataset with selected features, the Decision Tree classifier consistently achieves a ROC AUC value of 0.8. This implies that it successfully captures the underlying patterns in the data. Furthermore, when tested on unseen data, the Decision Tree retains a high ROC AUC score of roughly 0.88. This demonstrates its robustness and capacity to generalise to new, previously unseen samples, making it an attractive candidate for this job.

On the original dataset with selected features, Random Forest, like Decision Tree, earns a ROC AUC score of 0.8. When compared to a single Decision Tree, this ensemble strategy reduces overfitting and improves model stability. Random Forest retains a good ROC AUC score of roughly 0.81 when tested on unknown data, suggesting its capacity to generalise.

The Gradient Boosting Classifier outperforms Logistic Regression in terms of ROC AUC, with values ranging from 0.8 to 0.82 on the original dataset. This demonstrates its capacity to detect complex patterns in data. On unseen data, it retains a respectable ROC AUC score of roughly 0.81, highlighting its generalisability.

LightGBM, another boosting method, performs well on the original dataset, with ROC AUC scores ranging from 0.81 to 0.82. This shows that it may be slightly more successful than Gradient Boosting at capturing complicated correlations within data. LightGBM retains a ROC AUC score of around 0.81 on unseen data, showing its robustness.

During model validation, all models had ROC AUC scores more than 0.63, with Decision Tree and Random Forest scoring around 0.65. However, while Logistic Regression yields excellent recall, it suffers from poor accuracy, resulting in lower F1 scores Decision Tree excels in the evaluation on unseen data, with a ROC AUC score of roughly 0.88, suggesting its robustness and competence in handling unseen samples. Random Forest likewise has a high ROC AUC score of around 0.81, exhibiting its generalisation ability. Gradient Boosting and LightGBM both have ROC AUC ratings in the 0.81 range, suggesting their dependability in real-world circumstances.

Overall, the Decision Tree, Random Forest, Gradient Boosting, and LightGBM models perform well on both the original dataset and unseen data, with robust ROC AUC values indicating their ability to handle imbalanced data and generalise to new samples. While Logistic Regression has a high recall, it may require further tuning or feature engineering to improve accuracy and overall performance, particularly on unseen data.

**5.2.2 Oversampled Dataset**

**5.2.2.1 All Features**

Various techniques were tested with different hyperparameters on the BorderlineSMOTE Oversampled Dataset with all features in the assessment of machine learning models, with a focus on the critical measure of ROC AUC score. While Logistic Regression consistently achieved a ROC AUC score of 0.77, there was little variance based on hyperparameter modifications. It performed best with the hyperparameters C: 0.1 and penalty: l2. Decision Tree and Random Forest, on the other hand, showed more significant variations in ROC AUC values, ranging from 0.79 to 0.81 and consistently at 0.85, respectively. The Decision Tree model worked best with max\_depth: 20, min\_samples\_split: 10, and n\_estimators: 100, whereas Random Forest performed best with max\_depth: None, min\_samples\_split: 2, and n\_estimators: 100. Based on hyperparameter choices, Gradient Boosting and LightGBM displayed versatility, with ROC AUC values ranging from 0.78 to 0.90 and 0.80 to 0.90, respectively. For both models, the best parameters were learning\_rate: 0.2, max\_depth: 5, and n\_estimators: 300.

Further evaluation of these models on both training and unseen data revealed some noteworthy findings. On training data, Logistic Regression had a ROC AUC score of 0.75 and on unseen data, it had a score of 0.71. Decision Tree achieved a training ROC AUC score of 0.71 and excelled on unseen data with 0.72. Random Forest maintained its excellent performance on unseen data, scoring a strong 0.78 with a training ROC AUC of 0.71. Gradient Boosting Classifer demonstrated its robustness once again, with constant ROC AUC values of 0.74 on training data and 0.78 on unseen data. LightGBM was the best performance, with a training ROC AUC of 0.50 and an even better 0.80 on unseen data.

In summary, when dealing with the BorderlineSMOTE Oversampled Dataset with all features, Gradient Boosting and LightGBM are the preferred models because to their impressive performance, especially when implemented with the optimal hyperparameters discovered. These models generalise well to previously unseen data, making them viable candidates for classification tasks in this context.

**5.2.2.2 Selected Features**

Various techniques were tested with different hyperparameters on the BorderlineSMOTE Oversampled Dataset with selected features in the evaluation of machine learning models, with a focus on the critical measure of ROC AUC score.

Logistic Regression was tested using a variety of hyperparameters, the most important of which were the regularisation strength (C) and penalty (l1 and l2). Across all hyperparameters, the ROC AUC score remained constant at 0.77. While it remained stable, there was no significant improvement in the ROC AUC score.

The performance of the Decision Tree model was evaluated using several hyperparameters, notably the maximum depth and minimum samples necessary to divide a node. With numerous hyperparameter combinations, the best ROC AUC score obtained was 0.83.

Random Forest also achieved a ROC AUC score of 0.83, which is comparable to the Decision Tree model. It was consistent across many hyperparameter settings, including varying maximum depth, minimum samples for splitting, and number of trees (n\_estimators) values.

The performance of the Gradient Boosting Classifier was influenced by the learning rate, maximum depth, and number of estimators. It achieved ROC AUC values ranging from 0.78 to 0.86, with the greatest value obtained using a learning rate of 0.2, a maximum depth of 5, and 300 estimators. This model was sensitive to hyperparameter adjustment and showed the potential for improved performance.

LightGBM was likewise sensitive to hyperparameter changes. It produced ROC AUC values ranging from 0.8 to 0.9, with a learning rate of 0.2, maximum depth of 5, and 300 estimators producing the best results. LightGBM outperformed other models, especially when hyperparameters were used to optimise it.

Logistic Regression (0.74), Decision Tree (0.73), Random Forest (0.72), Gradient Boosting (0.74), and LightGBM (0.5) were the ROC AUC scores on the training data for model validation. Among them, LightGBM obtained the lowest ROC AUC value, indicating that its performance on the training set may be improved.

When assessing the models on unseen data, the following ROC AUC values were obtained: Logistic Regression (0.71), Decision Tree (0.77), Random Forest (0.77), Gradient Boosting (0.42), and LightGBM (0.5) were the ROC AUC scores on the unseen data for model validation. On unseen data, both Gradient Boosting had much lower ROC AUC scores than on training data, indicating possible overfitting difficulties.

Overall, LightGBM and Gradient Boosting Classifier are promising models when using ROC AUC as the major assessment measure, but they may require further adjustment to avoid overfitting on unseen data. Random Forest and Decision Tree models also performed well on this dataset and should be considered. While Logistic Regression was stable, it did not reach the same ROC AUC performance as the other models. Additional feature engineering and hyperparameter tuning approaches may improve the models performance on both training and unseen data.

This existing literature emphasises the importance of hyperparameter optimization (HPO) and model validation in the field of machine learning. The studies underscore the importance of HPO by demonstrating automated procedures that outperform manual tuning across multiple algorithms, as well as the influence of different optimisation approaches on model performance. Furthermore, model validation appears as an important step in confirming the predicted accuracy of machine learning models, notably through external validation, accounting for competing events, and using sophisticated approaches to analyse and understand validation findings. Evaluation metrics are also investigated, with an emphasis on the use of error models and a range of metrics adapted to certain activities. These findings highlight the need of thorough hyperparameter tuning, robust model validation, and cautious metric selection in ensuring the reliability and performance of machine learning models across a wide range of applications.

The findings for machine learning models and hyperparameter tuning give important insights into the performance of various models, notably in terms of ROC AUC scores. In several essential ways, the findings are consistent with previous studies. To begin, using ROC AUC values as the primary metric for model evaluation is consistent with accepted best practices. Because it gives a full perspective of model classification performance, ROC AUC is a solid measure for analysing models, especially when working with imbalanced datasets.

The findings highlight the hyperparameter sensitivity of machine learning models. This is consistent with the knowledge that the selection of hyperparameters may have a considerable influence on the performance of a model. The varied ROC AUC values under different hyperparameter setups emphasize the need of fine-tuning hyperparameters. Furthermore, the evaluation of model performance on unseen data adheres to recognised model validation criteria. It is critical to assess how effectively models generalise to previously unknown data in order to ensure their practical utility and trustworthiness.

However, there are some differences in the findings when compared to previous literature. Notably, the ROC AUC score for Logistic Regression on the initial imbalanced dataset with all features decreases on unseen data. This discrepancy indicates that the model's generalisation capabilities may face difficulties. Overfitting or difficulties connected to class imbalance might be significant contributors to this disparity. Furthermore, while Decision Tree outperforms on the original dataset with chosen features, attaining a high ROC AUC score on unseen data, it underperforms on the BorderlineSMOTE Oversampled Dataset. This variant suggests that Decision Tree may be affected by dataset distribution and oversampling strategies. Random Forest's persistent good performance is impressive, although its performance gain on unseen data compared to Decision Tree is quite minor. This implies that, while Random Forest reduces overfitting, it may fall short of entirely addressing the complexity presented by the BorderlineSMOTE Oversampled Dataset. Both the Gradient Boosting and the LightGBM models are hyperparameter sensitive, and their performance on unseen data occasionally falls short of their training performance. This emphasizes the significance of fine-tuning hyperparameters to avoid overfitting. When compared to the initial imbalanced dataset, the BorderlineSMOTE Oversampled Dataset shows a distinct data situation. While it effectively tackles class imbalance, it may create complexity that have an influence on model performance. This emphasises the need of taking individual dataset features into account when selecting and modifying models. Furthermore, the usage of selected features effects model performance, with some models performing well with feature selection but requiring extra tuning. This highlights the importance of feature engineering and selection during the modelling phase.

In summary, these findings both confirm and contradict previous studies. They demonstrate the significance of ROC AUC as a performance indicator, hyperparameter sensitivity, and the necessity to analyse model generalization. They do, however, add new difficulties relating to dataset properties and feature selection, which have an influence on model performance. These findings add to our understanding of how various models perform under different settings and highlight the need of careful model selection and tweaking in real machine learning applications.

The comprehensive evaluation of machine learning models, which addresses the primary research problem of predicting life assurance application conversion, is a critical step toward achieving the research objectives.

The main focus on ROC AUC scores, a relevant metric for binary classification tasks, directly addresses the research goal of objectively evaluating the efficacy of several supervised machine learning models and their hyperparameters. The aim is to determine the best model to solve the conversion prediction issue by rigorously assessing the performance of models such as Logistic Regression, Decision Tree, Random Forest, Gradient Boosting Classifier, and LightGBM.

The evaluation provides vital insights into the performance of these models in the context of the original imbalanced dataset. Logistic Regression, for example, shows promise, with ROC AUC values ranging between 0.78 and 0.79. However, as seen by a significant decline in ROC AUC on unseen data, this model has generalisation issues. This finding strongly correlates with research objective of evaluating how hyperparameters contribute to model performance, as the performance of Logistic Regression varies dramatically with different parameter configurations.

Similarly, the Decision Tree model’s variable performance as a result of hyperparameters emphasises the significance of parameter selection. It achieves its greatest ROC AUC score with specified parameters, but the observed decline in generalisation to unseen data shows the need for additional optimisation, which is a critical component of addressing the research problem.

On unseen data, the Random Forest model not only beats the Decision Tree, but it also retains high generalisation. This is consistent with the research objective, illustrating the significance of not only model selection but also how effectively a model generalises to new data.

The Gradient Boosting Classifier consistently performs well, with ROC AUC values ranging from 0.8 to 0.82 on the original dataset and a strong ROC AUC of 0.78 on unseen data. By showcasing a model with considerable generalisation potential, this finding directly meets the research objectives.

LightGBM surpasses its competitors on a constant basis, with ROC AUC values ranging from 0.81 to 0.82 on the original dataset and a notable ROC AUC of 0.82 on unseen data. Its steady performance matches the study aims flawlessly, demonstrating a highly viable answer to the research problem.

In relation to feature selection, the analysis continues to give insight on how different models perform under different settings, which is well aligned with the research aims. While Logistic Regression is stable, it still faces generalisation issues on the original dataset and unseen data. This underscores the need of dealing with generalisation in the context of feature selection.

The Decision Tree excels the original dataset, and it also excels in generalisation, achieving a ROC AUC of 0.88 on unseen data. This finding clearly supports the research objective of evaluating model generalisation in various settings. Random Forest continues its reasonable performance on the original dataset and robust generalisation with a ROC AUC of about 0.81 on unseen data, coinciding with the study aims by emphasising the model’s appropriateness in feature-selected datasets.

Overall, our evaluation not only provides a thorough understanding of how different machine learning models perform, but it also directly meets the research problem and objectives. It highlights the crucial role of model selection, hyperparameter tuning, and generalisation in predicting life assurance application conversion, providing useful insights for practical data analytics and supervised machine learning application in the life assurance industry.

**5.3 Future Research**

Future research in predicting life insurance application conversion offers a number of promising avenues. The use of ensemble modelling, which integrates many machine learning techniques, has the potential to enhance accuracy. Deep learning algorithms are worth investigating because of their ability to capture patterns. For improved model performance, feature engineering and feature selection strategies should continue to grow.

Addressing imbalanced datasets remains critical. Sophisticated approaches such as SMOTE variants can assist with this. Future research could adopt artificial Intelligence could improve model interpretability. Also, time-series analysis can give insights into changing conversion rates for temporal datasets. Collaboration with industry professionals is essential because domain-specific information improves models, while external data sources, such as economic indicators, can help to improve predictions. Dynamic models that can respond to market changes are required. Understanding the impact of consumer behaviour on conversion rates, particularly interactions and communication channels, could inform customised strategy. Regional variances could be shown using geospatial analysis. Customer feedback and sentiment analysis could provide insights into the influence of satisfaction. Decision-making processes can be clarified using behavioural economics. Long-term policy results may help to refine models.

1. **Conclusion**

By utilising a wide range of analytical approaches and machine learning techniques, this research project has methodically tackled the challenging challenge of predicting the conversion of life assurance applications. The are a number of significant results and insights from the research.

WorkflowStatus, Agency, UWDecision, and CommDateProvided are among the most significant features impacting the conversion of life assurance applications, according to the feature correlation and significance research. These features showed strong association and consistently ranked high in relevance across a variety of analytical approaches and machine learning models. The Class Aware Feature Importance analysis presented a more nuanced view, showing both the positive and negative effects of features on predictive model performance. For various machine learning models, Recursive Feature Elimination with Cross-Validation (RFECV) consistently identified Agency and WorkflowStatus as essential features.

The performance of machine learning models was tested using ROC AUC as the key metric, demonstrating how different models performed on both the original and oversampled datasets. Logistic Regression showed promise, but it has limits in applicability to unseen data. Both the Decision Tree and Random Forest models performed well on the original dataset, however the Decision Tree model excelled on unseen data, while Random Forest indicated good generalisation. Gradient Boosting and LightGBM consistently outperformed on both training and unseen data, with LightGBM demonstrating excellent generalisation skills. Model performance was affected by the use of feature selection, with some models benefiting from feature selection yet requiring more tuning.

The research not only gave useful insights into the performance of several machine learning models, but it also stressed the importance of hyperparameter tuning, model generalisation, and feature selection in addressing the research problem. These findings have practical consequences for the life insurance business, providing direction on the selection of predictive models and attributes for the development of accurate conversion prediction models.

Furthermore, the research revealed the importance of ROC AUC as an assessment parameter, which is consistent with best practices in binary classification tasks. It has also emphasised the complexity imposed by dataset attributes and oversampling approaches, emphasizing the need of taking these elements into account when developing models.

Finally, by presenting a thorough and data-driven method to tackling a real-world problem in the life assurance business, the research contributes to the field of data analytics and machine learning. The findings provide practical insights that may help the industry with decision-making and resource allocation, eventually boosting the efficiency of life assurance application conversion procedures.

**References**

Adibi, A., Sadatsafavi, M. and Ioannidis, J.P. (2020) ‘Validation and utility testing of clinical prediction models’, JAMA, 324(3), p. 235. doi:10.1001/jama.2020.1230.

Adler, A.I. and Painsky, A. (2022) ‘Feature importance in gradient boosting trees with cross-validation feature selection’, Entropy, 24(5), p. 687. doi:10.3390/e24050687.

Andonie, R. (2019) ‘Hyperparameter optimization in Learning Systems’, Journal of Membrane Computing, 1(4), pp. 279–291. doi:10.1007/s41965-019-00023-0.

Ahern, I., Noack, A., Guzman-Nateras, L., Dou, D., Li, B. and Huan, J., 2019. NormLime: A new feature importance metric for explaining deep neural networks. arXiv preprint arXiv:1909.04200.

Ali, A. and Gravino, C. (2021) ‘An empirical comparison of validation methods for software prediction models’, Journal of Software: Evolution and Process, 33(8). doi:10.1002/smr.2367.

Altmann, A. et al. (2010) ‘Permutation importance: A corrected feature importance measure’, Bioinformatics, 26(10), pp. 1340–1347. doi:10.1093/bioinformatics/btq134.

Ampt, A.B.F. (2017) *On the potential for machine learning in prediction of insurance policy sales: Helping Insurance Intermediaries Get Insights in their clients' insurance needs*. thesis.

AN, S.H., YEO, S.H. and KANG, M., 2021. A Study on a car Insurance purchase Prediction Using Two-Class Logistic Regression and Two-Class Boosted Decision Tree. Korea Journal of Artificial Intelligence, 9(1), pp.9-14.

Anagol, S., Cole, S. and Sarkar, S. (2017) ‘Understanding the advice of commissions-motivated agents: Evidence from the Indian Life Insurance Market’, Review of Economics and Statistics, 99(1), pp. 1–15. doi:10.1162/rest\_a\_00625.

Ao, D., Hu, Z. and Mahadevan, S. (2017) ‘Design of validation experiments for Life prediction models’, Reliability Engineering & System Safety, 165, pp. 22–33. doi:10.1016/j.ress.2017.03.030.

Asir, D., Appavu, S. and Jebamalar, E. (2016) ‘Literature review on feature selection methods for high-dimensional data’, International Journal of Computer Applications, 136(1), pp. 9–17. doi:10.5120/ijca2016908317.

Azpurua, M.A. et al. (2014) ‘A review on the drawbacks and enhancement opportunities of the feature selective validation’, IEEE Transactions on Electromagnetic Compatibility, 56(4), pp. 800–807. doi:10.1109/temc.2014.2304622.

Barry, L. and Charpentier, A., 2022. The Fairness of Machine Learning in Insurance: New Rags for an Old Man?. arXiv preprint arXiv:2205.08112.

Bujang, M.A. and Baharum, N. (2016) ‘Sample size guideline for Correlation Analysis’, World Journal of Social Science Research, 3(1), p. 37. doi:10.22158/wjssr.v3n1p37.

Cabitza, F. et al. (2021) ‘The importance of being external. methodological insights for the external validation of machine learning models in medicine’, Computer Methods and Programs in Biomedicine, 208, p. 106288. doi:10.1016/j.cmpb.2021.106288.

Chandrashekar, G. and Sahin, F. (2014) ‘A survey on feature selection methods’, Computers & Electrical Engineering, 40(1), pp. 16–28. doi:10.1016/j.compeleceng.2013.11.024.

Chang, W.T. and Lai, K.H. (2021) “A neural network-based approach in predicting consumers' intentions of purchasing insurance policies,” Acta Informatica Pragensia, 10(2), pp. 138–154. Available at: <https://doi.org/10.18267/j.aip.152>.

Chen, W. et al. (2007) ‘A design-driven validation approach using Bayesian prediction models’, Journal of Mechanical Design, 130(2). doi:10.1115/1.2809439.

Cho, H. et al. (2020) ‘Basic enhancement strategies when using Bayesian optimization for hyperparameter tuning of deep neural networks’, IEEE Access, 8, pp. 52588–52608. doi:10.1109/access.2020.2981072.

Debray, T.P.A. et al. (2015) ‘A new framework to enhance the interpretation of external validation studies of Clinical Prediction Models’, Journal of Clinical Epidemiology, 68(3), pp. 279–289. doi:10.1016/j.jclinepi.2014.06.018.

Demircioğlu, A. (2021) ‘Measuring the bias of incorrect application of feature selection when using cross-validation in radiomics’, Insights into Imaging, 12(1). doi:10.1186/s13244-021-01115-1.

Dragos, S.L., Dragos, C.M. and Muresan, G.M. (2020) ‘From intention to decision in Purchasing Life Insurance and private pensions: Different effects of knowledge and behavioural factors’, Journal of Behavioral and Experimental Economics, 87, p. 101555. doi:10.1016/j.socec.2020.101555.

Franceschi, L., Donini, M., Frasconi, P. and Pontil, M., 2017, July. Forward and reverse gradient- based hyperparameter optimization. In International Conference on Machine Learning (pp. 1165-1173). PMLR.

Frempong, N.K., Nicholas, N. and Boateng, M.A., 2017. Decision tree as a predictive modeling tool for auto insurance claims. International Journal of Statistics and Applications, 7(2), pp.117-120.

Greenwell, B.M., Boehmke, B.C. and McCarthy, A.J., 2018. A simple and effective model-based variable importance measure. arXiv preprint arXiv:1805.04755.

Gregorutti, B., Michel, B. and Saint-Pierre, P. (2016) ‘Correlation and variable importance in random forests’, Statistics and Computing, 27(3), pp. 659–678. doi:10.1007/s11222-016-9646-1.

Groll, A., Wasserfuhr, C. and Zeldin, L., 2022. Churn modeling of life insurance policies via statistical and machine learning methods--Analysis of important features. arXiv e-prints, pp.arXiv-2202.

Gogtay, N.J. and Thatte, U.M. (2017) Statistics for Researcher: Principles of Correlation Analysis. Journal of the Association of Physicians of India, 65, 78- 80.

Gopagoni, D.R., Lakshmi, P.V. and Siripurapu, P. (2020) “Predicting the sales conversion rate of car insurance promotional calls,” Rising Threats in Expert Applications and Solutions, pp. 321–329. Available at: https://doi.org/10.1007/978-981-15-6014-9\_37.

Hanafy, M. and Ming, R. (2021) ‘Machine learning approaches for auto insurance big data’, Risks, 9(2), p. 42. doi:10.3390/risks9020042.

Heinze, G. and Dunkler, D. (2016) ‘Five myths about variable selection’, Transplant International, 30(1), pp. 6–10. doi:10.1111/tri.12895.

Hickey, G.L. and Blackstone, E.H. (2016) ‘External model validation of binary clinical risk prediction models in cardiovascular and Thoracic Surgery’, The Journal of Thoracic and Cardiovascular Surgery, 152(2), pp. 351–355. doi:10.1016/j.jtcvs.2016.04.023.

Hooker, S. et al. (2018) Evaluating feature importance estimates, Google Research. Available at: https://research.google/pubs/pub47088/ (Accessed: 11 May 2023).

Huang, X., Wu, L. and Ye, Y. (2019) ‘A review on dimensionality reduction techniques’, International Journal of Pattern Recognition and Artificial Intelligence, 33(10), p. 1950017. doi:10.1142/s0218001419500174.

Imai, K., Tingley, D. and Yamamoto, T. (2012) ‘Experimental designs for identifying causal mechanisms’, Journal of the Royal Statistical Society Series A: Statistics in Society, 176(1), pp. 5–51. doi:10.1111/j.1467-985x.2012.01032.x.

Ivanescu, A.E. et al. (2015) ‘The importance of prediction model validation and assessment in obesity and Nutrition Research’, International Journal of Obesity, 40(6), pp. 887–894. doi:10.1038/ijo.2015.214.

Jaiswal, R., 2022. PROGNOSTICATING CUSTOMERS’INTENTION TO PURCHASE AN INSURANCE PLAN WITH MACHINE LEARNING. Fostering Resilient Business Ecosystems and Economic Growth: Towards the Next Normal, p.292.

Joy, T.T., Rana, S., Gupta, S. and Venkatesh, S., 2016, December. Hyperparameter tuning for big data using Bayesian optimisation. In 2016 23rd International Conference on Pattern Recognition (ICPR) (pp. 2574-2579). IEEE.

Ke, G. et al. (2017) Lightgbm: Proceedings of the 31st International Conference on Neural Information Processing Systems, Guide Proceedings. Available at: https://dl.acm.org/doi/10.5555/3294996.3295074 (Accessed: April 1, 2023).

Konig, G. et al. (2021) ‘Relative feature importance’, 2020 25th International Conference on Pattern Recognition (ICPR) [Preprint]. doi:10.1109/icpr48806.2021.9413090.

Li, J. et al. (2017) ‘Feature selection’, ACM Computing Surveys, 50(6), pp. 1–45. doi:10.1145/3136625.

Li, S. (2019) ‘Insurance customer purchase prediction based on data optimization’, Statistics and Application, 08(05), pp. 784–796. doi:10.12677/sa.2019.85089.

Mai, T.H. et al. (2020) ‘A study on behaviors of purchasing life insurance in Vietnam’, Management Science Letters, pp. 1693–1700. doi:10.5267/j.msl.2020.1.011.

Mantovani, R.G., Horváth, T., Cerri, R., Junior, S.B., Vanschoren, J. and de Carvalho, A.C.P.D.L.F., 2018. An empirical study on hyperparameter tuning of decision trees. arXiv preprint arXiv:1812.02207.

Mau, S., Pletikosa, I. and Wagner, J. (2018) ‘Forecasting the next likely purchase events of insurance customers’, International Journal of Bank Marketing, 36(6), pp. 1125–1144. doi:10.1108/ijbm-11-2016-0180.

Mauritsius, T. et al. (2020) ‘Customer churn prediction models for PT. XYZ Insurance’, 2020 8th International Conference on Orange Technology (ICOT) [Preprint]. doi:10.1109/icot51877.2020.9468788.

McDonnell, K. et al. (2023) ‘Deep learning in insurance: Accuracy and model interpretability using TabNet’, Expert Systems with Applications, 217, p. 119543. doi:10.1016/j.eswa.2023.119543.

Merikanto, K (2022) Using Machine Learning To Predict Purchase Potential From Customer Data. Thesis.

Miao, J. and Niu, L. (2016) ‘A survey on feature selection’, Procedia Computer Science, 91, pp. 919–926. doi:10.1016/j.procs.2016.07.111.

Misra, P. and Yadav, A.S., 2020. Improving the classification accuracy using recursive feature elimination with cross-validation. Int. J. Emerg. Technol, 11(3), pp.659-665.

Molina, L.C., Belanche, L. and Nebot, A. (2002) ‘Feature selection algorithms: A survey and experimental evaluation’, 2002 IEEE International Conference on Data Mining, 2002. Proceedings. [Preprint]. doi:10.1109/icdm.2002.1183917.

Morrison, R.E. et al. (2013) ‘Data Partition Methodology for validation of Predictive Models’, Computers & Mathematics with Applications, 66(10), pp. 2114–2125. doi:10.1016/j.camwa.2013.09.006.

Muranda, C., Ali, A. and Shongwe, T. (2021) ‘Deep learning method for detecting fraudulent motor insurance claims using unbalanced data’, 2021 62nd International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS) [Preprint]. doi:10.1109/itms52826.2021.9615264.

Nomi, M. and Sabbir, Md.M. (2020) ‘Investigating the factors of consumers’ purchase intention towards Life Insurance in Bangladesh: An application of the theory of reasoned action’, Asian Academy of Management Journal, 25(2). doi:10.21315/aamj2020.25.2.6.

Paruchuri, H. (2020) ‘The impact of machine learning on the future of insurance industry’, American Journal of Trade and Policy, 7(3), pp. 85–90. doi:10.18034/ajtp.v7i3.537.

Parvandeh, S. et al. (2020) ‘Consensus features nested cross-validation’, Bioinformatics, 36(10), pp. 3093–3098. doi:10.1093/bioinformatics/btaa046.

Paul, M.J. (2017) ‘Feature selection as causal inference: Experiments with text classification’, Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017) [Preprint]. doi:10.18653/v1/k17-1018.

Pesantez-Narvaez, J., Guillen, M. and Alcañiz, M. (2019) ‘Predicting motor insurance claims using telematics data—XGBoost versus logistic regression’, Risks, 7(2), p. 70. doi:10.3390/risks7020070.

Probst, P., Wright, M.N. and Boulesteix, A. (2019) ‘Hyperparameters and tuning strategies for Random Forest’, WIREs Data Mining and Knowledge Discovery, 9(3). doi:10.1002/widm.1301.

Quan, Z. and Valdez, E.A. (2018) ‘Predictive analytics of insurance claims using multivariate decision trees’, Dependence Modeling, 6(1), pp. 377–407. doi:10.1515/demo-2018-0022.

Rahman, M.S. et al. (2017) ‘Review and evaluation of performance measures for survival prediction models in external validation settings’, BMC Medical Research Methodology, 17(1). doi:10.1186/s12874-017-0336-2

Rajbahadur, G.K. et al. (2022) ‘The impact of feature importance methods on the interpretation of defect classifiers’, IEEE Transactions on Software Engineering, 48(7), pp. 2245–2261. doi:10.1109/tse.2021.3056941.

Rubi, M.A. et al. (2022) “Machine learning prediction of consumer travel insurance purchase behavior,” 2022 13th International Conference on Computing Communication and Networking Technologies (ICCCNT) [Preprint]. Available at: https://doi.org/10.1109/icccnt54827.2022.9984470.

Reddy, G.T. et al. (2020) ‘Analysis of dimensionality reduction techniques on Big Data’, IEEE Access, 8, pp. 54776–54788. doi:10.1109/access.2020.2980942.

Saarela, M. and Jauhiainen, S. (2021) ‘Comparison of feature importance measures as explanations for classification models’, SN Applied Sciences, 3(2). doi:10.1007/s42452-021-04148-9.

Senthilnathan, S. (2019) ‘Usefulness of correlation analysis’, SSRN Electronic Journal [Preprint]. doi:10.2139/ssrn.3416918.

Severino, M.K. and Peng, Y. (2021) ‘Machine learning algorithms for fraud prediction in Property Insurance: Empirical Evidence Using Real-World Microdata’, Machine Learning with Applications, 5, p. 100074. doi:10.1016/j.mlwa.2021.100074.

Shi, L. et al. (2018) ‘Variable selection and validation in multivariate modelling’, Bioinformatics, 35(6), pp. 972–980. doi:10.1093/bioinformatics/bty710.

Steyerberg, E.W. and Harrell, F.E. (2016) ‘Prediction models need appropriate internal, internal–external, and external validation’, Journal of Clinical Epidemiology, 69, pp. 245–247. doi:10.1016/j.jclinepi.2015.04.005.

Steyerberg, E.W. and Vergouwe, Y. (2014) ‘Towards better clinical prediction models: Seven steps for development and an ABCD for validation’, European Heart Journal, 35(29), pp. 1925–1931. doi:10.1093/eurheartj/ehu207.

Stucki, O. (2019) Predicting the customer churn with machine learning methods - CASE: private insurance customer data. thesis.

Taha, A., Cosgrave, B. and Mckeever, S. (2022) ‘Using feature selection with machine learning for generation of insurance insights’, Applied Sciences, 12(6), p. 3209. doi:10.3390/app12063209.

Tantithamthavorn, C. et al. (2017) ‘An empirical comparison of model validation techniques for defect prediction models’, IEEE Transactions on Software Engineering, 43(1), pp. 1–18. doi:10.1109/tse.2016.2584050.

van der Putten, P., de Ruiter, M. and van Someren, M. (2000) CoIL Challenge 2000 Tasks and Results: Predicting and Explaining Caravan Policy Ownership, Coil Challenge 2000 Report. Available at: https://liacs.leidenuniv.nl/~puttenpwhvander/library/cc2000/ (Accessed: April 1, 2023).

van Geloven, N. et al. (2022) ‘Validation of prediction models in the presence of competing risks: A guide through modern methods’, BMJ [Preprint]. doi:10.1136/bmj-2021-069249.

Venkatesh, B. and Anuradha, J. (2019) ‘A review of feature selection and its methods’, Cybernetics and Information Technologies, 19(1), pp. 3–26. doi:10.2478/cait-2019-0001.

Wei, G. et al. (2020) ‘A novel hybrid feature selection method based on dynamic feature importance’, Applied Soft Computing, 93, p. 106337. doi:10.1016/j.asoc.2020.106337.

Wojtas, M. and Chen, K., 2020. Feature importance ranking for deep learning. Advances in Neural Information Processing Systems, 33, pp.5105-5114.

Wu, J., Chen, X.Y., Zhang, H., Xiong, L.D., Lei, H. and Deng, S.H., 2019. Hyperparameter optimization for machine learning models based on Bayesian optimization. Journal of Electronic Science and Technology, 17(1), pp.26-40.

Xia, H., Zhou, Y. and Zhang, Z. (2022) ‘Auto Insurance Fraud Identification based on a CNN-LSTM fusion deep learning model’, International Journal of Ad Hoc and Ubiquitous Computing, 39(1/2), p. 37. doi:10.1504/ijahuc.2022.120943.

Xie, Z. et al. (2020) ‘Fist: A feature-importance sampling and tree-based method for automatic design flow parameter tuning’, 2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC) [Preprint]. doi:10.1109/asp-dac47756.2020.9045201.

Yang, L. and Shami, A. (2020) ‘On hyperparameter optimization of Machine Learning Algorithms: Theory and practice’, Neurocomputing, 415, pp. 295–316. doi:10.1016/j.neucom.2020.07.061.

Zhang, B., Rajan, R., Pineda, L., Lambert, N., Biedenkapp, A., Chua, K., Hutter, F. and Calandra, R., 2021, March. On the importance of hyperparameter optimization for model-based reinforcement learning. In International Conference on Artificial Intelligence and Statistics (pp. 4015-4023). PMLR.

Zhang, R. et al. (2017) ‘Deep and shallow model for insurance churn prediction service’, 2017 IEEE International Conference on Services Computing (SCC) [Preprint]. doi:10.1109/scc.2017.51.

Zhou, Z. and Hooker, G. (2021) ‘Unbiased measurement of feature importance in tree-based methods’, ACM Transactions on Knowledge Discovery from Data, 15(2), pp. 1–21. doi:10.1145/3429445.