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**

There are a number of people who have helped and guided me throughout the process of completing this project. Firstly, my project supervisor, Sam Weiss, for always being available to provide his assistance and expertise. All of the staff and lecturers at CCT Dublin for giving me the basis need to be able to get to this stage. My employer deserves recognition for their understanding and flexibility in facilitating my academic pursuits. And last, but not least, my sincerest thank you goes to my family and friends for their never ending support.

**Abstract**

This research project tackles the difficult challenge of predicting the conversion of life assurance applications by doing a thorough analysis that incorporates multiple analytical methodologies and machine learning techniques. The research produces significant findings and useful insights for the life assurance sector. Key factors influencing the conversion of life assurance applications are identified using feature correlation and significance analysis, including the workflow status, agency, underwriting status, and presence of a policy commencement date. These features consistently show significant associations and high relevance across a variety of analytical approaches and machine learning models. A different perspective is provided by Class Aware Feature Importance analysis, which implies both the positive and negative impacts among the predictor variables on model performance. Recursive Feature Elimination with Cross-Validation proves useful and consistent among several models in identifying WorkflowStatus and Agency as being the most important features. The Receiver Operating Characteristic Area Under the Curve is used as the primary measure in order to evaluate the performance of several models across datasets with both imbalanced target variables and oversampled but balanced target variables, respectively. Logistic Regression proved to have poor generalisability, despite reasonable training performance. Decision Tree, Random Forest, Gradient Boosting Classifer, and LightGBM all performed well on training models, as well proving to have strong generalisability, performing to a reasonably high level on unseen data. Using only the identified most important features has a positive impact on some models, but proves that others may still require further hyperparameter tuning. Hyperparameter tuning, feature selection, and generalisability of trained models are essential for solving the research problem. The life assurance industry could practically benefit from these findings as they can help identify important features and guide the selection of the most appropriate machine learning models.

Table of Contents

[1. Introduction 1](#_Toc146234140)

[1.1 Background 1](#_Toc146234141)

[1.2 Existing Research and Research Gaps 2](#_Toc146234142)

[1.3 Research Problem 2](#_Toc146234143)

[1.4 Research Objectives 3](#_Toc146234144)

[2. Literature Review 5](#_Toc146234145)

[2.1 Machine Learning in Insurance 5](#_Toc146234146)

[2.2 Correlation Analysis 10](#_Toc146234147)

[2.3 Feature Importance 11](#_Toc146234148)

[2.4 Feature Selection 14](#_Toc146234149)

[2.5 Hyperparameter Optimisation 16](#_Toc146234150)

[2.6 Experimental Design 18](#_Toc146234151)

[2.7 Validation 20](#_Toc146234152)

[2.7 Model Evaluation Metrics 25](#_Toc146234153)

[2.8 Conclusions 26](#_Toc146234154)

[3. Methodology 28](#_Toc146234155)

[3.1 Sampling Strategy 28](#_Toc146234156)

[3.2 Primary Research Methodology 28](#_Toc146234157)

[3.3 Data Collection 29](#_Toc146234158)

[3.4 Data Cleansing 29](#_Toc146234159)

[3.5 Data Exploration 30](#_Toc146234160)

[3.6 Feature Correlation & Feature Importance 31](#_Toc146234161)

[3.7 Machine Learning 34](#_Toc146234162)

[3.8 Performance Validation 35](#_Toc146234163)

[3.9 Ethical Considerations 35](#_Toc146234164)

[3.9 Project Schedule 36](#_Toc146234165)

[4. Results 37](#_Toc146234166)

[4.1 Feature Correlation & Feature Importance 37](#_Toc146234167)

[4.1.1 Imbalanced Training Data 37](#_Toc146234168)

[4.1.1.1 Association Analysis 37](#_Toc146234169)

[4.1.1.2 Random Forest 40](#_Toc146234170)

[4.1.1.3 Gradient Boosting Classifier 43](#_Toc146234171)

[4.1.1.4 LightGBM Classifier 47](#_Toc146234172)

[4.1.1.4 Class Aware Feature Importance 51](#_Toc146234173)

[4.1.1.5 RFECV 52](#_Toc146234174)

[4.1.1.6 Compare with and without Feature 54](#_Toc146234175)

[4.1.2 Oversampled Training Data 57](#_Toc146234176)

[4.1.2.1 Association Analysis 57](#_Toc146234177)

[4.1.2.2 Random Forest 61](#_Toc146234178)

[4.1.2.3 Gradient Boosting Classifier 64](#_Toc146234179)

[4.1.2.4 LightGBM Classifier 67](#_Toc146234180)

[4.1.2.5 Class Aware Feature Importance 70](#_Toc146234181)

[4.1.2.6 RFECV 72](#_Toc146234182)

[4.2 Machine Learning Models & Hyperparameter Tuning 74](#_Toc146234183)

[4.2.1 Imbalanced Training Data 74](#_Toc146234184)

[4.2.1.1 All Features 74](#_Toc146234185)

[4.2.1.1.1 Logistic Regression 74](#_Toc146234186)

[4.2.1.1.2 Decision Tree 75](#_Toc146234187)

[4.2.1.1.3 Random Forest 76](#_Toc146234188)

[4.2.1.1.4 Gradient Boosting Classifier 77](#_Toc146234189)

[4.2.1.1.5 LightGBM 78](#_Toc146234190)

[4.2.1.1.6 Model Validation 79](#_Toc146234191)

[4.2.1.1.7 Evaluation on Unseen Data 80](#_Toc146234192)

[4.2.1.2 Selected Features 82](#_Toc146234193)

[4.2.1.2.1 Logistic Regression 82](#_Toc146234194)

[4.2.1.2.2 Decision Tree 82](#_Toc146234195)

[4.2.1.2.3 Random Forest 83](#_Toc146234196)

[4.2.1.2.4 Gradient Boosting Classifier 84](#_Toc146234197)

[4.2.1.2.5 Light GBM 85](#_Toc146234198)

[4.2.1.2.6 Model Validation 86](#_Toc146234199)

[4.2.1.2.7 Evaluation on Unseen Data 86](#_Toc146234200)

[4.2.2 Oversampled Dataset 88](#_Toc146234201)

[4.2.2.1 All Features 88](#_Toc146234202)

[4.2.2.1.1 Logistic Regression 88](#_Toc146234203)

[4.2.2.1.2 Decision Tree 88](#_Toc146234204)

[4.2.2.1.3 Random Forest 89](#_Toc146234205)

[4.2.2.1.4 Gradient Boosting Classifier 90](#_Toc146234206)

[4.2.2.1.5 Light GBM 91](#_Toc146234207)

[4.2.2.1.6 Model Validation 92](#_Toc146234208)

[4.2.2.1.7 Evaluation on Unseen Data 93](#_Toc146234209)

[4.2.2.2 Selected Features 94](#_Toc146234210)

[4.2.2.2.1 Logistic Regression 94](#_Toc146234211)

[4.2.2.2.2 Decision Tree 95](#_Toc146234212)

[4.2.2.2.3 Random Forest 96](#_Toc146234213)

[4.2.2.2.4 Gradient Boosting Classifier 97](#_Toc146234214)

[4.2.2.2.5 Light GBM 98](#_Toc146234215)

[4.2.2.2.6 Model Validation 99](#_Toc146234216)

[4.2.2.2.7 Evaluation on Unseen Data 100](#_Toc146234217)

[4.2.3 Comparison Performance on Unseen Data 101](#_Toc146234218)

[5. Discussion 105](#_Toc146234219)

[5.1 Feature Correlation And Feature Importance 105](#_Toc146234220)

[5.1.1 Imbalanced Dataset 105](#_Toc146234221)

[5.1.2 Oversampled Dataset 106](#_Toc146234222)

[5.2 Machine Learning Models & Hyperparameter Tuning 109](#_Toc146234223)

[5.2.1 Original Imbalanced Dataset 109](#_Toc146234224)

[5.2.1.1 All Features 109](#_Toc146234225)

[5.2.1.2 Selected Features 110](#_Toc146234226)

[5.2.2 Oversampled Dataset 111](#_Toc146234227)

[5.2.2.1 All Features 111](#_Toc146234228)

[5.2.2.2 Selected Features 112](#_Toc146234229)

[5.2.3 Model Performance Comparison 114](#_Toc146234230)

[5.3 Future Research 118](#_Toc146234231)

[6. Conclusion 119](#_Toc146234232)

[References 121](#_Toc146234233)

[Figure 1 - Association Analysis P-Value Matrix, Imbalanced Training Dataset 38](#_Toc146234234)

[Figure 2 - Association Analysis Chi Squared Statistic Matrix, Imbalanced Training Dataset 38](#_Toc146234235)

[Figure 3 – Association Analysis Cramer’s V Matrix, Imbalanced Training Dataset 39](#_Toc146234236)

[Figure 4 – Random Forest ROC AUC vs. No of Features (n Estimators: 50) , Imbalanced Training Dataset 42](#_Toc146234237)

[Figure 5 – Random Forest ROC AUC vs. No of Features (n Estimators: 100) , Imbalanced Training Dataset 42](#_Toc146234238)

[Figure 6 – Random Forest ROC AUC vs. No of Features (n Estimators: 200) , Imbalanced Training Dataset 43](#_Toc146234239)

[Figure 7 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 50) , Imbalanced Training Dataset 45](#_Toc146234240)

[Figure 8 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 100) , Imbalanced Training Dataset 46](#_Toc146234241)

[Figure 9 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 200) , Imbalanced Training Dataset 46](#_Toc146234242)

[Figure 10 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 50) , Imbalanced Training Dataset 49](#_Toc146234243)

[Figure 11 - Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 100) , Imbalanced Training Dataset 50](#_Toc146234244)

[Figure 12 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 200) , Imbalanced Training Dataset 50](#_Toc146234245)

[Figure 13 - Feature Weighted Importance Scores Bar Chart, Imbalanced Training Dataset 52](#_Toc146234246)

[Figure 14 – Random Forest RFECV – Cross-Validated ROC AUC vs. Number of Features, Imbalanced Training Dataset 53](#_Toc146234247)

[Figure 15 – Gradient Boosted Classifier RFECV – Cross-Validated ROC AUC vs. Number of Features, Imbalanced Training Dataset 53](#_Toc146234248)

[Figure 16 – Light GBM Classifier RFECV – Cross-Validated ROC AUC vs. Number of Features, Imbalanced Training Dataset 54](#_Toc146234249)

[Figure 17 – Model Performance with Feature Manipulation, Imbalanced Training Dataset 56](#_Toc146234250)

[Figure 18 – Model Performance with Feature Manipulation, Oversampled Training Dataset 58](#_Toc146234251)

[Figure 19 – Association Analysis Chi Squared Statistic Matrix, Oversampled Training Dataset 59](#_Toc146234252)

[Figure 20 – Association Analysis Cramer’s V Matrix, Oversampled Training Dataset 60](#_Toc146234253)

[Figure 21– Random Forest ROC AUC vs. No of Features (n Estimators: 50) , Oversampled Training Dataset 62](#_Toc146234254)

[Figure 22 – Random Forest ROC AUC vs. No of Features (n Estimators: 100) , Oversampled Training Dataset 62](#_Toc146234255)

[Figure 23 – Random Forest ROC AUC vs. No of Features (n Estimators: 200) , Oversampled Training Dataset 63](#_Toc146234256)

[Figure 24 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 50) , Oversampled Training Dataset 65](#_Toc146234257)

[Figure 25 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 100) , Oversampled Training Dataset 65](#_Toc146234258)

[Figure 26 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 200) , Oversampled Training Dataset 66](#_Toc146234259)

[Figure 27 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 50) , Oversampled Training Dataset 68](#_Toc146234260)

[Figure 28 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 100) , Oversampled Training Dataset 68](#_Toc146234261)

[Figure 29 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 200) , Oversampled Training Dataset 69](#_Toc146234262)

[Figure 30 – Feature Weighted Importance Scores Bar Chart, Oversampled Training Dataset 71](#_Toc146234263)

[Figure 31 – Random Forest RFECV – Cross-Validated ROC AUC vs. Number of Features, Oversampled Training Dataset 73](#_Toc146234264)

[Figure 32 – Gradient Boosted RFECV – Cross-Validated ROC AUC vs. Number of Features, Oversampled Training Dataset 73](#_Toc146234265)

[Figure 33 – Light GBM RFECV – Cross-Validated ROC AUC vs. Number of Features, Oversampled Training Dataset 74](#_Toc146234266)

[Figure 34 – Accuracy & ROC AUC Validation Comparison by Model, Imbalanced Training Dataset with all features 79](#_Toc146234267)

[Figure 35 – Model Accuracy Comparison on Unseen Test Dataset, Imbalanced Training Dataset with all features 80](#_Toc146234268)

[Figure 36 – Model ROC AUC Comparison on Unseen Test Dataset, Imbalanced Training Dataset with all features 81](#_Toc146234269)

[Figure 37 – Model Accuracy Comparison on Unseen Test Dataset, Imbalanced Training Dataset with selected features 87](#_Toc146234270)

[Figure 38 – Model ROC AUC Comparison on Unseen Test Dataset, Imbalanced Training Dataset with selected features 87](#_Toc146234271)

[Figure 39 – Accuracy & ROC AUC Validation Comparison by Model, Oversampled Training Dataset with all features 92](#_Toc146234272)

[Figure 40 – Model Accuracy Comparison on Unseen Test Dataset, Oversampled Training Dataset with all features 93](#_Toc146234273)

[Figure 41 – Model ROC AUC Comparison on Unseen Test Dataset, Oversampled Training Dataset with all features 94](#_Toc146234274)

[Figure 42 – Model Accuracy Comparison on Unseen Test Dataset, Oversampled Training Dataset with selected features 100](#_Toc146234275)

[Figure 43 – Model ROC AUC Comparison on Unseen Test Dataset, Oversampled Training Dataset with selected features 101](#_Toc146234276)

[Figure 44 – Unseen Data, ROC AUC Score Comparison Heatmap 101](#_Toc146234277)

[Figure 45 – Unseen Data, Accuracy Comparison Heatmap 102](#_Toc146234278)

[Figure 46 – Unseen Data, Precision Comparison Heatmap 102](#_Toc146234279)

[Figure 47 – Unseen Data, Recall Comparison Heatmap 103](#_Toc146234280)

[Figure 48 – Unseen Data, F1 Score Comparison Heatmap 103](#_Toc146234281)

[Figure 49 – Unseen Data, ROC AUC Score Comparison by Model and Dataset 104](#_Toc146234282)

[Figure 50 – Unseen Data, Accuracy Comparison by Model and Dataset 104](#_Toc146234283)

[Table 1 – Project Schedule 36](#_Toc146234284)

[Table 2 – Association Analysis, Imbalanced Training Dataset 37](#_Toc146234285)

[Table 3 – Random Forest Feature Importance Scores, Imbalanced Training Dataset 40](#_Toc146234286)

[Table 4 – Random Forest ROC AUC by n\_estimators and number of features, Imbalanced Training Dataset 41](#_Toc146234287)

[Table 5 – Gradient Boosting Classifier Feature Importance Scores, Imbalanced Training Dataset 43](#_Toc146234288)

[Table 6 – Gradient Boosting Classifier Feature Importance Scores, Imbalanced Training Dataset 45](#_Toc146234289)

[Table 7 – Gradient Boosting Classifier Feature Importance Scores, Imbalanced Training Dataset 47](#_Toc146234290)

[Table 8 – Gradient Boosting Classifier Feature Importance Scores, Imbalanced Training Dataset 49](#_Toc146234291)

[Table 9 – Feature Weighted Importance Scores, Imbalanced Training Dataset 51](#_Toc146234292)

[Table 10 – RFECV by Model and Feature, Imbalanced Training Dataset 52](#_Toc146234293)

[Table 11 – Model ROC AUC Comparison with and without Selected Features, Imbalanced Training Dataset 54](#_Toc146234294)

[Table 12 – Statistical Significance of Selected Features, Imbalanced Training Dataset 55](#_Toc146234295)

[Table 13 – Association Analysis, Oversampled Training Dataset 57](#_Toc146234296)

[Table 14 – Random Forest Feature Importance Scores, Oversampled Training Dataset 61](#_Toc146234297)

[Table 15 – Gradient Boosting Classifier Feature Importance Scores, Oversampled Training Dataset 64](#_Toc146234298)

[Table 16 – Light GBM Classifier Feature Importance Scores, Oversampled Training Dataset 67](#_Toc146234299)

[Table 17 – Feature Weighted Importance Scores, Oversampled Training Dataset 70](#_Toc146234300)

[Table 18 – RFECV by Model and Feature, Oversampled Training Dataset 72](#_Toc146234301)

[Table 19 – Logistic Regression Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with all features 74](#_Toc146234302)

[Table 20 – Decision Tree Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with all features 75](#_Toc146234303)

[Table 21 – Random Forest Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with all features 76](#_Toc146234304)

[Table 22 – Gradient Boosted Classifier Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with all features 77](#_Toc146234305)

[Table 23 – Light GBM Classifier Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with all features 78](#_Toc146234306)

[Table 24 – Model Performances on Validation Dataset, Imbalanced Training Dataset with all features 79](#_Toc146234307)

[Table 25 – Model Performances on Unseen Test Dataset, Imbalanced Training Dataset with all features 80](#_Toc146234308)

[Table 26 – Logistic Regression Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with selected features 82](#_Toc146234309)

[Table 27 – Decision Tree Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with selected features 82](#_Toc146234310)

[Table 28 – Random Forest Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with selected features 83](#_Toc146234311)

[Table 29 – Gradient Boosting Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with selected features 84](#_Toc146234312)

[Table 30 – Light GBM Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with selected features 85](#_Toc146234313)

[Table 31 – Model Performances on Validation Dataset, Imbalanced Training Dataset with selected features 86](#_Toc146234314)

[Table 32 – Model Performances on Validation Dataset, Imbalanced Training Dataset with selected features 86](#_Toc146234315)

[Table 33 – Logistic Regression Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with all features 88](#_Toc146234316)

[Table 34 – Decision Tree Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with all features 88](#_Toc146234317)

[Table 35 – Random Forest Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with all features 89](#_Toc146234318)

[Table 36 – Gradient Boosting Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with all features 90](#_Toc146234319)

[Table 37 – Light GBM Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with all features 91](#_Toc146234320)

[Table 38 – Model Performances on Validation Dataset, Oversampled Training Dataset with all features 92](#_Toc146234321)

[Table 39 – Model Performances on Unseen Test Dataset, Oversampled Training Dataset with all features 93](#_Toc146234322)

[Table 40 – Logistic Regression Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with selected features 94](#_Toc146234323)

[Table 41 – Decision Tree Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with selected features 95](#_Toc146234324)

[Table 42 – Random Forest Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with selected features 96](#_Toc146234325)

[Table 43 – Gradient Boosting Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with selected features 97](#_Toc146234326)

[Table 44 – Light GBM Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with selected features 98](#_Toc146234327)

[Table 45 - Model Performances on Validation Dataset, Oversampled Training Dataset with selected features 99](#_Toc146234328)

[Table 46 - Model Performances on Unseen Test Dataset, Oversampled Training Dataset with selected features 100](#_Toc146234329)

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

The role of data analytics and data science has evolved from that of supplementary tools to crucial assets for organisations seeking to maintain a competitive advantage. The insurance and financial services sectors are perfect examples of industries in which these disruptive technologies have the potential to thrive.

A plethora of complex issues encourage life assurance organisations to continually adapt and innovate. One of the most serious of these challenges is the need to optimally manage resources in order to maximise incoming business. This mission is at the heart of a balancing act that life assurance businesses must perform on a daily basis.

To comprehend the scale of this challenge, the complexities of the life assurance sector must first be understood. It is a sector 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 both human and financial resources becomes crucial.

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

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

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

In general, supervised machine learning models have mostly been employed in general insurance to predict client purchasing behaviour. However, the use of this machine learning models in this way has not been extended in the same way to the purchase of life assurance. Overall, the amount of research on the variables impacting customer purchase intentions in non-life insurances greatly exceed that of life assurance. One of the aims of the current research is to bridge this research gap, and enhance the body of research that currently exists in relation to the use of supervised machine learning in life assurance. These supervised machine learning practices include, but are not limited to, correlation analysis, feature importance identification, hyperparameter tuning, and model performance metrics.

The existing body of research has outlines how feature selection and other pre-processing measures can be used to deal with challenges such as imbalanced class distributions and the identification of variables that have the greatest impact on the purchase of insurance policies. Within the existing research, Decision Trees and Random Forest stand out as having some of the most consistent strong performances when predicting purchase behaviour. These models also have the benefits of being easy to interpret, while also having their own built in methods of feature importance identification. Churn prediction, claim prediction, and fraud detection have also proven to be among the benefits of using machine learning for classification tasks. In addition to Decision Trees and Random Forests, deep learning models have started to show some potential in such classification tasks.

* 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. **Literature Review**

By simply using human judgement or traditional work practices, it can be very difficult for life assurance organisations to determine which of the applications they have received are likely to become active policies. This is a gap which data analytics and , more specifically, supervised machine learning could possibly fill. The aim of this literature review is to give an overview of the current state of play in relation to how supervised machine learning is currently used in both the life and non-life insurance industries.

**2.1 Machine Learning in Insurance**

While there is little research into the use of data analytics and supervised machine learning in life assurance in particular, there is a vast amount of research into how these approaches are adopted in the areas of general insurances. There is a broad range of existing research into the use of machine learning in general insurances in relation to classification and feature importance identification. This research included the prediction on purchasing insurance policies.

The performance capabilities of machine learning in classification of insurance policy purchasing behaviour, with accuracy scores of up to 88%, is observed among predictive models demonstrated as part of the CoIL Challenge 2000 (van der Putten et al., 2000). In addition to showing how machine learning can be used successfully to predict the purchase of insurance policies, the variables which have the greatest impact on policy purchasing are also identified, with customer demographics and insurance product benefits deemed as being of the most importance when making such predictions. Chang and Lai (2021) build on these findings by using the same dataset but instead applying neural network techniques for insurance policy purchasing classification. The results of these techniques are comparable to those of the CoIL Challenge 2000’s more traditional methods. There were further similarities to the CoIL Challenge in that demographic variables were identified as being among the most influential variables when predicting policy purchasing outcomes.

Random Forest, Decision Tree, and Stochastic Gradient Descent produce the highest levels of accuracy when evaluating and comparing ten different machine learning classification algorithims used to predict whether or not a customer would purchase an insurance policy (Rubi et al., 2022). These findings are consistent with those of Ampt (2017), who evaluated ten different classification models and found that Decision Tree and Random Forest models performed with accuracies of up to 94%.

Both Logistic Regression and boosted Decision Tree models achieved high levels of accuracy in predicting customer intent to purchase car insurance, indicating that they are among the best models for this type of insurance classification task (An et al., 2021). When using personal data as predictor variables in Random Forest models to predict insurance policy purchase behaviour, prediction accuracies of more than 90% are achieved (Mau et al., 2018). This not only demonstrates Random Forest’s suitability for policy purchase prediction, but it also strengthens support for personal data, such as demographics, as important in the prediction outcome.

Experimentation with six different machine learning models and how they perform in predicting the purchase of insurance policies results in LightGBM as being the best performing model, with demographic and geographical features deemed to be the most important predictors of policy purchasing (Jaiswal, 2022).

Financial knowledge and overall attitudes towards insurances have been found to influence life assurance policy purchasing behaviour (Mai et al, 2020), with customer risk appetite and financial literacy both also having significant impacts (Nomi & Sabbir, 2020).

A broad range of research has been carried out on the use of machine learning for other classification purposes within the non-life insurance industries, such as churn prediction, claim prediction, and fraud detection.

Stucki (2019) discovered that machine learning techniques, particularly Random Forest and AdaBoost, are more accurate than traditional methods in predicting client churn in the insurance sector. When used to anticipate insurance policy cancellation, both logistic regression and tree-based techniques perform similarly (Groll et al., 2022). Mauritsius et al. (2020) assessed Decision Tree, Nave Bayes, and Artificial Neural Networks for their ability to classify customer churn in the insurance sector. Decision Tree is determined to be the most effective of them. A combined Deep & Shallow classification was proposed by Zhang et al. (2017) in order to predict customer churn in the insurance industry, as they argues that this approach would be able to generalise well to real-world data. In addition to this, they concluded that this Deep & Shallow method outperformed classification performance of both deep only and shallow only approaches.

Pesantez-Narvaez et al. (2019) compared Logistic Regression and XGBoost for claim classification prediction in the insurance sector. They conclude that Logistic Regression performed better in this categorisation classification. Random Forest is determined by Hanafy and Ming (2021) to have the best accuracy and generalisability for predicting insurance claim occurrence, while Quan and Valdez (2018) go on to state that multivariate tree-based models often outperform univariate tree-based models for this classification task. Frempong et al. (2017) identify how some features may have a higher impact on whether or not a claim would be lodged. McDonnell et al. (2023) discovered how deep learning approached could outperform more traditional machine learning methods for predicting claims.

When used to detect fraud in property insurance, ensemble approaches outperform alternative models such as Logistic Regression (Severino & Peng, 2021). However, a mixture of deep learning models could beat traditional methods in insurance fraud classification tasks because they are deemed to be better able to deal with the high dimensionality that is commonly present in insurance datasets. However, there was a word of warning, as these deep learning algorithms may struggle when confronted with imbalanced datasets. Muranda et al. (2021) discovered the issue of imbalanced datasets when their research indicated that deep learning classification approaches to fraud detection performed better on balanced datasets than imbalanced datasets.

Taha et al. (2022) claim that suitable feature selection strategies could aid in dealing with challenges due to the negative impact that can occur when an insurance dataset contains a high level of noise. They go on to state that the most critical variables influencing insurance policy purchasing behaviour are the customer’s financial understanding and attitude towards insurance. Li (2019) discovered that Logistic Regression, Decision Trees, and Random Forest may be utilised effectively to predict insurance consumers behaviour. Dragos et al. (2020) conducted an empirical study that supported the assumption that financial and insurance knowledge is a key variable in the policy purchasing outcomes of insurance policies. They also found that personal customer information such as marital status, education levels, and income levels have an association with life insurance purchasing behaviour.

According to Barry and Charpentier (2022), there are a lot of ethical concerns to address when employing machine learning techniques in the insurance sector, particularly in terms of discrimination. They warn against employing irrelevant correlated traits, which might lead to biases even if they are not demonstrated to be causal. Transparency is proposed as a means of mitigating such biases. Because of the massive amount of data increasingly being produced within the insurance industry, there is a growing realisation that sufficient technologies are required to successfully use this data for commercial gain (Paruchuri, 2020).

**2.2 Correlation Analysis**

According to Gogtay & Thatte (2017), correlation analysis covers a variety of factors. Some of these include the importance of sample size, the impact of outliers, the existence of non-linear connections, the possibility of inaccurate correlations, and the appearance of misleading correlations.

Describing the usefulness of the linear correlation coefficient between two variables to find multicollinearity of variables in a model, Senthilnathan (2019) cautions that the interpretation of correlation should not be conflated with claims that it is incorrectly represented as a causal effect. According to this argument, correlation only reveals such associations, but does not examine or reveal the nature or strength of the links between variables.

The sample size used in conducting the examination of relationships should be appropriately considered to ensure that the correlatuion results can achieve the minimum necessary coefficient value with sufficient power and type I error or p-value (Bujang & Baharum, 2016).

**2.3 Feature Importance**

Feature importance can be described as one of the most prevalent methods of explaining the way in which machine learning models behave (Saarela & Jauhiainen, 2021). Classification itself is not always the only desired outcome, but rather knowing the importance of how specific features in a model can in some ways be measured, and then how the presence of certain features can either prevent or increase the likelihood of a certain outcome.

Gopagoni et al. (2020) evaluated important features and factors for insurance sale conversion. Logistic Regression achieves a predictive accuracy of 84% and a cross-validation score of 81%. Support Vector Machine achieves a predictive accuracy of 80%. 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. It is found that the models have a reasonably high accuracy, with one 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 is that in some instances a number of feature importance estimators are found to be less accurate at identifying feature importance than randomly assigning feature importance values to 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) introduce the concept of a dual-net architecture, where an operator and a selector work together 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 resulting in an approach out-performs many of the best-in-class methods of ranking feature importance and feature selection.

Another propose a method of feature selection is the use of a feature selection algorithm called Dynamic Feature Importance based Feature Selection (DFIFS) (Wei et al., 2020). DFIFS can also be used along with a traditional method 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. conclude that M-DFIFS performs 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) discuss 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 is shown how this issue can be addressed by incorporating the split-improvement measured on unseen data in order to correct the bias.

Using the Random Forest algorithm along with correlated features as a method of feature selection was evaluated by Gregorutti et al. (2016). They highlight 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 conclude 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) propose 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 impact 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 stressed 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) propose 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 is concluded that adopting Principal Component Analysis (PCA) along with machine learning algorithms generally results in better outcomes than machine learning alone when a dataset has a high level of dimensionality (Reddy et al., 2020). However, the review also determines 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 results 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 domain knowledge is more valuable than over-complicated feature selection techniques.

**2.5 Hyperparameter Optimisation**

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

The advantages of adopting automated hyperparameter optimization (HPO) in model-based reinforcement learning (MBRL) is 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. demonstrate that automated HPO can greatly outperform manual human tuning, and that dynamically tuning hyperparameters during training can further increase performance. This sheds light on the influence of various hyperparameters on training stability and the subsequent rewards.

Franceschi et al. (2017) investigate 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 optimisation on large datasets. Franceschi et al. referr to research on data cleaning and learning task interactions and demonstrate that if the number of hyperparameters is minimal, forward-mode computing may be preferred to reverse-mode computation.

Using Bayesian optimization, Joy et al. (2016) present a novel concept 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 portion in parallel. Using a transfer learning configuration, the knowledge collected from the portions 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 automated tuning of design flow parameters is presented by Xie et al. (2020) as a machine learning-based solution to experimental design. This 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 avoid overfitting. When compared to earlier techniques, experimental results on benchmark circuits reveal a considerable gain in design quality or decrease in sampling cost.

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) give warning on the importance of model validation, and how a lack of external validation among large amounts of data leads to many models 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) explore 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 great importance, even more so than random split sample methods. They argue in favour of internal and 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) review and evaluate a number of performance measures for external validation of prediction models. They recommend 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 distort 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 are generally 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) provide 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 modelling, Morrison et al. (2013) present 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 non-linear 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) discuss the significance of validation techniques in simulation tools for complicated situations, as well as the shortcomings of the standardised feature selected validation (FSV) method. By evaluating its shortcomings and complexities, it is hoped to uncover improvement opportunities to make FSV a more robust tool for data validation.

Parvandeh et al. (2020) explor how to utilise feature selection to increase machine learning model accuracy while avoiding overfitting. A consensus nested cross-validation (cnCV), a novel approach that combines feature stability from differential privacy and nested cross-validation (nCV) are 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. compare 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) address how to use feature selection to increase the predictive accuracy of machine learning models. The Recursive Feature Elimination with Cross-Validation (RFECV) method is suggested and tested on a dataset using five distinct machine learning methods. 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 machine learning models, the nature of the data, its quality, and volume should also be taken into account.

Demircioğlu (2021) examines 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 reveal a significant positive bias, with higher dimensionality datasets more prone to overfitting. The study emphasised 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 multivariate 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, the multivariate model 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) present 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 eight external validation sets. The validation datasets are 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 emphasises 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 discovers, 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 advises 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 is also emphasised.

The relevance of verifying predictive models is discussed by Ivanescu et al. (2015). It discusses 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) present 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 explore 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) analysr various model validation methods for datasets containing software development effort estimation (SDEE) and software fault prediction (SFP). The study analyses estimate strategies’ prediction accuracy and stability using 10 different cross-validation (CV) and bootstrap validation methods. The results demonstrate 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 recommends employing model-agnostic methodologies to identify essential variables and instance-level interpretations to explain whether software systems are clean or flawed.

Adler & Painsky (2022) describe a weakness in the commonly used Gradient Boosting Machines (GBM) technique that causes bias in its feature importance 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 feature importance is unique to each implementation, but CVB provides impartial feature importance without sacrificing generalisation capabilities.

Altmann et al. (2010) highlight 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 models, have biased feature importance measurements. Altmann et al. offer a solution for normalising 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) analyse and suggested eight distinct evaluation measures and their properties under specified assumptions and for specific applications. The research states 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) discuss the significance of assessment metrics in batch evaluations of information retrieval systems. The findings provide 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 (ROCAUC) 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 ROC AUC to clinical performance indicators based on sensitivity and specificity. They found that, unless where good specificity is required, the change in the ROC 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 ROC AUC. However, if the baseline model performs well, increasing the ROC 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 ROC AUC increment is fair because changes in the ROC 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) presents a variety of existing and new metrics for evaluating the performance and uncertainty of Bayesian Network models, including metrics for conducting model sensitivity analysis, evaluating scenarios, depicting model complexity, assessing prediction performance, and evaluating model posterior probability distributions’ uncertainty. Marcot emphasises the value of metrics in enhancing model credibility, acceptance, and suitable application. The research further emphasises the significance of balancing model performance. 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) suggest 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 provides useful insights into the significance of several elements connected to the usage of machine learning techniques in the insurance industry. The literature examined has 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 has emerged as a significant subject in the literature. Different data pre-processing approaches have proven helpful in resolving issues like imbalanced class distributions and identifying characteristics that impact the purchasing of insurance policies.

Furthermore, the literature review demonstrates 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 emphasises the need for validation in predictive modelling and machine learning. External validation, in addition to internal validation, is required to check the predicted accuracy and robustness 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, this literature review stresses 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 data that can be utilised to examine the influence of different features on model performance.

Due to the presence of a diverse pool of applicants for life assurance, stratified random sampling is an appropriate sampling strategy for this research when splitting the population dataset into training, validation and test datasets. This method ensures that each of the training and validation splits are representative of the population as a whole, meaning that every application has an equal chance of being included in each split. The reason why this equal chance is important is that can help to reduce the probability of selection bias occurring, meaning overfitting and underfitting should be avoided. All of these probabilistic sampling measures help to ensure that the findings of the research have the greatest opportunity to be applicable to the population of interest.

**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 findings of using only the selected features 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 importance of the features in predicting application conversion and improve the machine learning models’ predictive performance.

**3.3 Data Collection**

The features of the life assurance applications are illustrated by the use of descriptive statistics, while correlations and trends are identified with help of analytic techniques. The dataset comprises of every application received by a life assurance organisation between 2017 and 2022 inclusive, with relevant features extracted, including, but not limited to product details, application dates, application workflow statuses. For data privacy purposes, personal and identifiable data has not been included. However, the presence of all data over a six year period enables a comprehensive and in-depth level of research to be carried out.

**3.4 Data Cleansing**

The first stage of data cleansing requires preparing the dataset. For the purpose of this research, columns having the data type ‘int64’ are considered categorical features. Following that, these categorical columns are transformed to the ‘category’ data type. This allows for more efficient categorical data handling. Then, by choosing columns with numeric data types, numerical properties are segregated. For each numeric feature, 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 (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 produced. The box plot shows the quartiles and outliers, but the violin plot shows the distribution’s form in greater detail, graphically representing the variability and range of the features, assisting in the detection 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 plot is created to visually analyse the underlying distribution and potential multimodality. Histograms strengthen 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 normal distribution. Deviations from the predicted distribution are shown by comparing the actual data quantiles to those of a normal distribution. The Q-Q plot of each numeric variable is constructed to examine its deviation from normality.

**3.6 Feature Correlation & Feature Importance**

Firstly, frequency distributions are calculated on categorical variables. Cross-tabulations are produced to compare the same categorical variables to the target variable. Chi-square tests are adopted in order to determine how independent each categorical variable is to the target variable. This is done to determine if there is a significant difference between the predicted frequencies and the actual frequencies. The strength of correlations between categorical variables is determined by Cramer’s V. Similar to the chi-squared tests, this helps to understand the extent to which variables are dependent on each other beyond the predicted frequencies. To show correlations among categorical variables, a matrix of Cramer’s V values is generated and presented as a heatmap for interpretability. Feature correlation analysis is used to determine the relationships between numerical variables. To understand the strength and direction of relationships, correlation matrices are produced and illustrated on heatmaps.

To aid robust model evaluation, the dataset is separated into two subsets. A training set and a test set. The train\_test\_split method is used, with a test size of 20%, and stratified on the target column. This stratification method assures that the class distribution is consistent in both sets and reduces the possibility of overfitting. The 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 determine feature importance, with several techniques adopted in order to get the most comprehensive and thorough insights.

A Random Forest classifier is created for each value of selected n\_estimators (50, 100, and 200) using the RandomForestClassifier 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 printed for each n\_estimators configuration. A loop is used to iterate over a set of n values that reflect the number of top-ranked characteristics to select from. The most important n characteristics are picked for each n value based on their importance scores. A new dataset (X\_train\_selected and X\_test\_selected) is constructed with only the selected characteristics. Then a new RandomForestClassifier (rf\_classifier\_selected) is trained and tested on the test data. The Area under the Receiver Operating Characteristic Curve (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 GradientBoostedClassifier, and a LightGBMClassifier.

In order to calculate Class Aware Feature Importance, a RandomForestClassifier is built. 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 scores. 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 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 relationship between feature and target variables 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 score in relation to 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 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 uses 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 calculates the relevance of each feature systematically, discarding the least informative features repeatedly until the optimum subset of features is found. 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 determine 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 selection 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 formed 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 assessed. 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 displayed, along with the model name and related assessment metric outputs.

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 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 the most important 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 performance of the oversampled data against the performance of the original imbalanced dataset.

**3.8 Performance Validation**

This research’s methodology is intended to develop 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 previously unseen data, these measures give a better assessment of each model’s predictive abilities. This thorough process ensures a rigorous approach to model creation and validation, resulting in the selection of the most effective predictive model.

**3.9 Ethical Considerations**

Among the most important ethical concerns to be considered when dealing with a life assurance dataset is that of the privacy and security of any individuals that the data relates to. One of the agreements made with the life assurance organisation is that such data would not be used in this research. The extract that the life assurance organisation agreed to release for the purposes of this research only includes data related to products, sales, and workflow statuses. This eliminates any risks associated with using private or sensitive data. This also helps to eliminate other ethical concerns in relation to biases against certain groups based on characteristics such as age, gender, colour, religion or any other characteristics of a personal or sensitive nature. Any biases or limitations in the data or analytic approach should be fully stated, and the research findings should be honestly and transparently reported.

The findings of the research must be reported accurately and transparently. This can be achieved by clearly outlining the methodology to be used, as well as declaring both the benefits and limitations of the data and methodology used to analyse it. Following these procedures allows ethical difficulties to be addressed and research to be carried out in compliance with best ethical practice.

**3.9 Project Schedule**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Milestone** | **Week 1** | **Week 2** | **Week 3** | **Week 4** | **Week 5** | **Week 6** | **Week 7** | **Week 8** | **Week 9** | **Week 10** | **Week 11** |
| Finalise research objectives |  |  |  |  |  |  |  |  |  |  |  |
| Preprocess dataset. Conduct EDA |  |  |  |  |  |  |  |  |  |  |  |
| Identify highly correlated and important features |  |  |  |  |  |  |  |  |  |  |  |
| Develop and train supervised machine learning models |  |  |  |  |  |  |  |  |  |  |  |
| Finalise the development of the artefact |  |  |  |  |  |  |  |  |  |  |  |
| Analyse and interpret the experimental results |  |  |  |  |  |  |  |  |  |  |  |
| Fine-tune the hyperparameters. Validate generalisability |  |  |  |  |  |  |  |  |  |  |  |
| Analyse results. Evaluate performance of the models |  |  |  |  |  |  |  |  |  |  |  |
| Write the methodology, results sections |  |  |  |  |  |  |  |  |  |  |  |
| Write the discussion section |  |  |  |  |  |  |  |  |  |  |  |
| Write conclusion section and revise the entire document |  |  |  |  |  |  |  |  |  |  |  |

Table 1 – Project Schedule

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 |

Table 2 – Association Analysis, Imbalanced Training Dataset

**A screenshot of a computer screen

Description automatically generated**

Figure 1 - Association Analysis P-Value Matrix, Imbalanced Training Dataset

**A screenshot of a computer screen

Description automatically generated**

Figure 2 - Association Analysis Chi Squared Statistic Matrix, Imbalanced Training Dataset

**A screenshot of a computer screen

Description automatically generated**

Figure 3 – 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 |

Table 3 – Random Forest Feature Importance Scores, Imbalanced Training Dataset

|  |  |  |
| --- | --- | --- |
| **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 |

Table 4 – Random Forest ROC AUC by n\_estimators and number of features, Imbalanced Training Dataset

**A graph with a line

Description automatically generated**

Figure 4 – Random Forest ROC AUC vs. No of Features (n Estimators: 50) , Imbalanced Training Dataset

**A graph with a line

Description automatically generated**

Figure 5 – Random Forest ROC AUC vs. No of Features (n Estimators: 100) , Imbalanced Training Dataset

**A graph with a line

Description automatically generated**

Figure 6 – Random Forest ROC AUC vs. No of Features (n Estimators: 200) , Imbalanced Training Dataset

**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 |

Table 5 – Gradient Boosting Classifier Feature Importance Scores, Imbalanced Training Dataset

|  |  |  |
| --- | --- | --- |
| **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 |

Table 6 – Gradient Boosting Classifier Feature Importance Scores, Imbalanced Training Dataset

A graph with blue dots

Description automatically generated

Figure 7 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 50) , Imbalanced Training Dataset

A graph on a white surface

Description automatically generated

Figure 8 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 100) , Imbalanced Training Dataset

**A graph with blue lines

Description automatically generated**

Figure 9 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 200) , Imbalanced Training Dataset

**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 |

Table 7 – Gradient Boosting Classifier Feature Importance Scores, Imbalanced Training Dataset

|  |  |  |
| --- | --- | --- |
| **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 |

Table 8 – Gradient Boosting Classifier Feature Importance Scores, Imbalanced Training Dataset

A graph with blue dots

Description automatically generated

Figure 10 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 50) , Imbalanced Training Dataset

A graph with blue dots

Description automatically generated

Figure 11 - Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 100) , Imbalanced Training Dataset

A graph with a line and dots

Description automatically generated

Figure 12 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 200) , Imbalanced Training Dataset

**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 |

Table 9 – Feature Weighted Importance Scores, Imbalanced Training Dataset

**A blue and white rectangle with a black border

Description automatically generated**

Figure 13 - Feature Weighted Importance Scores Bar Chart, Imbalanced Training Dataset

**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 |

Table 10 – RFECV by Model and Feature, Imbalanced Training Dataset

A graph with colorful lines

Description automatically generated

Figure 14 – Random Forest RFECV – Cross-Validated ROC AUC vs. Number of Features, Imbalanced Training Dataset

A graph of a graph with different colored lines

Description automatically generated with medium confidence

Figure 15 – Gradient Boosted Classifier RFECV – Cross-Validated ROC AUC vs. Number of Features, Imbalanced Training Dataset

A graph with colorful lines

Description automatically generated

Figure 16 – Light GBM Classifier RFECV – Cross-Validated ROC AUC vs. Number of Features, Imbalanced Training Dataset

**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 |

Table 11 – Model ROC AUC Comparison with and without Selected Features, Imbalanced Training Dataset

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

Table 12 – Statistical Significance of Selected Features, Imbalanced Training Dataset

A screenshot of a graph

Description automatically generated

Figure 17 – Model Performance with Feature Manipulation, Imbalanced Training Dataset

**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 |

Table 13 – Association Analysis, Oversampled Training Dataset

**A screenshot of a computer screen

Description automatically generated**

Figure 18 – Model Performance with Feature Manipulation, Oversampled Training Dataset

**A screen shot of a computer

Description automatically generated**

Figure 19 – Association Analysis Chi Squared Statistic Matrix, Oversampled Training Dataset

**A screenshot of a computer screen

Description automatically generated**

Figure 20 – 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 |

Table 14 – Random Forest Feature Importance Scores, Oversampled Training Dataset

**A graph with blue lines and dots

Description automatically generated**

Figure 21– Random Forest ROC AUC vs. No of Features (n Estimators: 50) , Oversampled Training Dataset

**A screenshot of a graph

Description automatically generated**

Figure 22 – Random Forest ROC AUC vs. No of Features (n Estimators: 100) , Oversampled Training Dataset

**A screen shot of a graph

Description automatically generated**

Figure 23 – Random Forest ROC AUC vs. No of Features (n Estimators: 200) , Oversampled Training Dataset

**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 |

Table 15 – Gradient Boosting Classifier Feature Importance Scores, Oversampled Training Dataset

A screen shot of a graph

Description automatically generated

Figure 24 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 50) , Oversampled Training Dataset

A screen shot of a graph

Description automatically generated

Figure 25 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 100) , Oversampled Training Dataset

A screen shot of a computer

Description automatically generated

Figure 26 – Gradient Boosting Classifier ROC AUC vs. No of Features (n Estimators: 200) , Oversampled Training Dataset

**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 |

Table 16 – Light GBM Classifier Feature Importance Scores, Oversampled Training Dataset

A screen shot of a computer

Description automatically generated

Figure 27 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 50) , Oversampled Training Dataset

A screen shot of a computer

Description automatically generated

Figure 28 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 100) , Oversampled Training Dataset

A screen shot of a computer

Description automatically generated

Figure 29 – Light GBM Classifier ROC AUC vs. No of Features (n Estimators: 200) , Oversampled Training Dataset

**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 |

Table 17 – Feature Weighted Importance Scores, Oversampled Training Dataset

**A white and blue rectangle with black border

Description automatically generated**

Figure 30 – Feature Weighted Importance Scores Bar Chart, Oversampled Training Dataset

**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 |

Table 18 – RFECV by Model and Feature, Oversampled Training Dataset

A graph with different colored lines

Description automatically generated

Figure 31 – Random Forest RFECV – Cross-Validated ROC AUC vs. Number of Features, Oversampled Training Dataset

A graph with different colored lines

Description automatically generated

Figure 32 – Gradient Boosted RFECV – Cross-Validated ROC AUC vs. Number of Features, Oversampled Training Dataset

A graph with colored lines

Description automatically generated

Figure 33 – Light GBM RFECV – Cross-Validated ROC AUC vs. Number of Features, Oversampled Training Dataset

**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 |

Table 19 – Logistic Regression Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with all features

**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 |

Table 20 – Decision Tree Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with all features

**4.2.1.1.3 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 |

Table 21 – Random Forest Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with all features

**4.2.1.1.4 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 |

Table 22 – Gradient Boosted Classifier Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with all features

**4.2.1.1.5 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 |

Table 23 – Light GBM Classifier Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with all features

**4.2.1.1.6 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 |

Table 24 – Model Performances on Validation Dataset, Imbalanced Training Dataset with all features

A close-up of a color chart

Description automatically generated

Figure 34 – Accuracy & ROC AUC Validation Comparison by Model, Imbalanced Training Dataset with all features

**4.2.1.1.7 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 | .77 |
| Decision Tree | {'max\_depth': 10, 'min\_samples\_split': 10} | .77 | .83 | .90 | .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 |

Table 25 – Model Performances on Unseen Test Dataset, Imbalanced Training Dataset with all features

A blue and white rectangular object

Description automatically generated

Figure 35 – Model Accuracy Comparison on Unseen Test Dataset, Imbalanced Training Dataset with all features

A pink rectangular object with white lines

Description automatically generated

Figure 36 – Model ROC AUC Comparison on Unseen Test Dataset, Imbalanced Training Dataset with all features

**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 |

Table 26 – Logistic Regression Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with selected features

**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 |

Table 27 – Decision Tree Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with selected features

**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 |

Table 28 – Random Forest Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with selected features

**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 |

Table 29 – Gradient Boosting Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with selected features

**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 |

Table 30 – Light GBM Mean Cross Validation Score by Hyperparameter Combination, Imbalanced Training Dataset with selected features

**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} | 0.80 | 0.84 | 0.92 | 0.88 | 0.65 |
| Random Forest | {'max\_depth': 20, 'min\_samples\_split': 10, 'n\_estimators': 200} | 0.80 | 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 |

Table 31 – Model Performances on Validation Dataset, Imbalanced Training Dataset with selected features

**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} | .80 | .84 | .92 | .88 | .81 |
| 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} | .80 | .83 | .94 | .88 | .81 |
| LightGBM | {'learning\_rate': 0.1, 'max\_depth': 3, 'n\_estimators': 100} | .80 | .82 | .97 | .88 | .81 |

Table 32 – Model Performances on Validation Dataset, Imbalanced Training Dataset with selected features

**A blue and white bar chart

Description automatically generated**

Figure 37 – Model Accuracy Comparison on Unseen Test Dataset, Imbalanced Training Dataset with selected features

**A pink bars with white stripes

Description automatically generated with medium confidence**

Figure 38 – Model ROC AUC Comparison on Unseen Test Dataset, Imbalanced Training Dataset with selected features

**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 |

Table 33 – Logistic Regression Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with all features

**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 |

Table 34 – Decision Tree Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with all features

**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 |

Table 35 – Random Forest Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with all features

**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 |

Table 36 – Gradient Boosting Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with all features

**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 |

Table 37 – Light GBM Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with all features

**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 | 0.00 | 0.01 | 0.50 |

Table 38 – Model Performances on Validation Dataset, Oversampled Training Dataset with all features

**A close-up of a color chart

Description automatically generated**

Figure 39 – Accuracy & ROC AUC Validation Comparison by Model, Oversampled Training Dataset with all features

**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} | .70 | .89 | .70 | .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 | .80 | .78 |
| LightGBM | {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | .77 | .88 | .81 | .85 | .80 |

Table 39 – Model Performances on Unseen Test Dataset, Oversampled Training Dataset with all features

**A blue and white bar chart

Description automatically generated**

Figure 40 – Model Accuracy Comparison on Unseen Test Dataset, Oversampled Training Dataset with all features

**A pink and white bar chart

Description automatically generated**

Figure 41 – Model ROC AUC Comparison on Unseen Test Dataset, Oversampled Training Dataset with all features

**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 |

Table 40 – Logistic Regression Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with selected features

**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 |

Table 41 – Decision Tree Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with selected features

**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 |

Table 42 – Random Forest Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with selected features

**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 |

Table 43 – Gradient Boosting Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with selected features

**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 |

Table 44 – Light GBM Mean Cross Validation Score by Hyperparameter Combination, Oversampled Training Dataset with selected features

**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 | 0.70 | 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 | 0.50 |

Table 45 - Model Performances on Validation Dataset, Oversampled Training Dataset with selected features

**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 | .70 | .79 | .77 |
| Gradient Boosting | {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | .22 | 1.00 | 0.00 | 0.00 | .42 |
| LightGBM | {'learning\_rate': 0.2, 'max\_depth': 5, 'n\_estimators': 300} | .22 | 1.00 | 0.00 | 0.00 | .50 |

Table 46 - Model Performances on Unseen Test Dataset, Oversampled Training Dataset with selected features

**A blue and white bar chart

Description automatically generated**

Figure 42 – Model Accuracy Comparison on Unseen Test Dataset, Oversampled Training Dataset with selected features

**A graph of pink bars

Description automatically generated with medium confidence**

Figure 43 – Model ROC AUC Comparison on Unseen Test Dataset, Oversampled Training Dataset with selected features

**4.2.3 Comparison Performance on Unseen Data**

A screenshot of a computer

Description automatically generated

Figure 44 – Unseen Data, ROC AUC Score Comparison Heatmap

A screenshot of a computer

Description automatically generated

Figure 45 – Unseen Data, Accuracy Comparison Heatmap

A screenshot of a computer

Description automatically generated

Figure 46 – Unseen Data, Precision Comparison Heatmap

A screenshot of a computer

Description automatically generated

Figure 47 – Unseen Data, Recall Comparison Heatmap

A screenshot of a computer

Description automatically generated

Figure 48 – Unseen Data, F1 Score Comparison Heatmap

A colorful rectangular objects on a white background

Description automatically generated

Figure 49 – Unseen Data, ROC AUC Score Comparison by Model and Dataset

A colorful bars on a white background

Description automatically generated

Figure 50 – Unseen Data, Accuracy Comparison by Model and Dataset

1. **Discussion**

**5.1** **Feature Correlation And Feature Importance**

The results of the feature correlation and feature importance investigations and experiments reveal 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, as higher Cramer's V values and lower p-values suggest greater relevance. The Cramer’s V values and chi-squared statistics indicate significant connections between particular variables and policy issuance.

**5.1.1 Imbalanced Dataset**

WorkflowStatus, Agency, and CommDateProvided display particularly 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. However, as per Senthilnathan (2019), these correlations should not be conflated to be understood as causality.

Feature relevance scores were computed using Random Forest, Gradient Boosting, and LightGBM classifiers with varying numbers of estimators. Features that return higher importance scores are deemed to have the most influence prediction performance on the respective model. Across all models and estimator configurations, the Agency feature is consistently deemed to be one of the most important features. WorkflowStatus and UWDecision are similarly highly ranked across models, indicating that they have a significant impact on predictive performance.

Class Aware Feature Importance analysis calculates the impact of each feature. In this method, positive scores suggest a positive contribution to model performance and negative scores indicate a negative contribution. Just like with the Random Forest, Gradient Boosting, and LightGBM classifiers, WorkflowStatus, Agency, and UWDecision are all shown to have a positive impact However, Class Aware Feature Importance analysis identifies ComissionSacrifice, ProductGroup, and Product as having a negative impact on predictive performance.

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

Finally, the comparison of feature importance 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 features that have a significant impact on model performance. WorkflowStatus, considerably improves ROC AUC values across multiple models when added, suggesting its significance in predicting 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 sector in order to create reliable prediction models.

**5.1.2 Oversampled Dataset**

The evaluation, using a range of methodologies, of the impact of independent features on the conversion of life assurance applications provides important insights. 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 features are deemed to be 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 the outputs of these feature importance identification methods, there is evidence to state that WorkflowStatus and Agency have a considerable impact on application conversion, but other features such as UWDecision, CommDateProvided, and BonusCommissionPercentage also have some importance. These findings are useful in developing an accurate prediction model for life assurance application conversion.

**5.1.3 Feature Importance Summary**

The insights gained from this research coincide with various important aspects described in the existing literature on correlation analysis, feature importance, and feature selection, while also adding new insights. 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 argument 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.

The effectiveness of these methods in correctly identifying the most important features for classification tasks means that they are often popular choices for the task. These finding would be support Hooker et al.’s (2018) conclusion that using effective feature importance identification techniques is of utmost importance in the feature selection process.

Overall, the results of the research are consistent with previous literature on feature importance and correlation analysis, but they also add useful insights. 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 impact of independent features on application conversion, are achieved systematically using various methods.

Using Cramer's V values and significance levels, and association analysis the research evaluates 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 predicting 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, directly meeting the research objectives related to feature impact evaluation.

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 CommissionSacrifice, 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.

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 problem and objectives.

**5.2 Machine Learning Models & Hyperparameter Tuning**

**5.2.1 Original Imbalanced Dataset**

**5.2.1.1 All Features**

The performance of several machine learning models and their hyperparameters, and how they impact on ROC AUC scores, is critical for determining their suitability of the model and hyperparameter combination best suited to solving the research problem and objectives.

The performance of Logistic Regression varies depending on the hyperparameter configuration tested. 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 test data, the ROC AUC value scores at 0.77, suggesting a reasonable generalisation performance.

With the hyperparameters max\_depth: 10 and min\_samples\_split: 10, the ROC AUC score of 0.8 is obtained for the Decision Tree. With a test ROC AUC score of 0.78, this model performs well on the original imbalanced dataset and has reasonable generalisation to new data. This suggests that Decision Tree performs quite well in this classification task.

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 respectable 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 reasonable ROC AUC score of 0.78. Gradient Boosting is a suitable alternative for handling the research problem and objectives since it has reasonable generalisation capabilities.

LightGBM 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 strong ROC AUC score of 0.82 on unseen data, demonstrating its solid generalisation abilities. Given its constant high performance on both the training and unseen datasets, this model appears as a very viable option for tackling the research problem.

Overall, when evaluating the effectiveness of machine learning models using ROC AUC values, while Logistic Regression, Decision Tree, Random Forest, and Gradient Boosting Classifier perform well, they may require additional measures to improve generalisation. LightGBM, on the other hand, stands out as having good potential, constantly performing well on both the original dataset and unseen data. In addressing the research problem and objectives, this models demonstrates robustness and effectiveness.

**5.2.1.2 Selected Features**

On the imbalanced dataset with only selected important features used as independent variables, Logistic Regression produces ROC AUC scores of 0.79 across multiple hyperparameter configurations. On unseen test data, it returns an ROC AUC score of around 0.72 indicating a problem with generalisation ability. Taking steps such as applying further hyperparameter tuning or addressing class imbalance may help improve the performance.

Decision Tree consistently achieves an ROC AUC score of 0.8 on the training data, indicating that it effectively captures the patterns within the training data. When applied to unseen data, it retains a reasonable ROC AUC score of 0.81, demonstrating a robustness and capacity to generalise to previously real-world data.

Random Forest, comparable to the Decision Tree performance, produces an ROC AUC score of 0.8 on the training dataset. Random Forest retains a good ROC AUC score of roughly 0.81 when tested on unseen data, suggesting it has capacity to generalise to real-world data.

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

LightGBM performs well on the training dataset, with ROC AUC scores between 0.81 and 0.82. This shows that it may be slightly more successful than Gradient Boosting at capturing complicated correlations within the data. LightGBM retains an ROC AUC score of around 0.81 on unseen data, showing its robustness.

When the performances and ROC AUC values on unseen data of each of the models is compared, it is clear that Decision Tree, Random Forest, Gradient Boosting, and LightGBM models perform well on both the training dataset and unseen test data, with robust ROC AUC values indicating their ability to handle imbalanced data and generalise to real-world data. While Logistic Regression does not generalise well to unseen data, and may require further tuning to improve overall performance.

**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 is little variance based on hyperparameter configurations. 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 strong 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.

When dealing with the BorderlineSMOTE Oversampled Dataset with all features, Gradient Boosting is the preferred models because to it’s impressive performance, especially when implemented with the optimal hyperparameters discovered. This model generalises well to previously unseen data, making it 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 important 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 is little variance based on hyperparameter configurations. 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 an ROC AUC score of 0.83, which is comparable to that of 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 achieved ROC AUC scores ranging from 0.78 to 0.86, with the best performing score obtained using a combination of 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 configuration, producing 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 on training data, 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 internal 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 previously unseen test data, 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 obtained. On unseen data, both Gradient Boosting and Light GBM had much lower ROC AUC scores than on training data, indicating possible overfitting difficulties.

**5.2.3 Model Performance Comparison**

Overall, LightGBM achieves the ROC AUC value across all investigations, with an external validation score of over 0.8 on the original imbalanced dataset with all features included. It also performs well on the imbalanced dataset with selected features and the oversampled dataset with all features. Random Forest and Gradient Boosting Classifier are also promising models when using ROC AUC as the assessment measure, but they may require further adjustment to avoid overfitting on unseen data. Decision Tree models also performed reasonably well on the imbalanced dataset with selected features, but could did not generalise as well on the other datasets. While Logistic Regression was stable, it did not reach the same ROC AUC performance as the other models. Additional hyperparameter tuning approaches may improve the models performance on both training and unseen data.

The previous research highlights the importance of hyperparameter optimisation and model validation in the area of machine learning, demonstrated by employing automated procedures that outperform manual tuning across multiple algorithms. Model validation is also described as an important step in verifying the performance of machine learning models, mostly through external validation, and the use of advanced approaches to analyse and understand validation findings. These findings highlight the need of thorough hyperparameter tuning, robust model validation, and careful metric selection in ensuring the reliability and performance of machine learning models.

The findings of this research provide important insights into the performance of various models, particularly in relation to their ability to result in reasonably ROC AUC scores. In several ways, the findings are consistent with previous studies. To begin with, using ROC AUC values as the primary metric for model evaluation is consistent with accepted best practices when measuring the ability to distinguish between classes in a binary classification task. Because it gives a full perspective of model classification performance, ROC AUC is a reliable measure for analysing models, especially when working with imbalanced datasets.

Consistent with the previous research conclusions that the selection of hyperparameters may have a considerable influence on the performance of a model, the findings of this research further confirm the sensitivity of hyperparameters in relation to machine learning models. The varying ROC AUC values produced under a range hyperparameter configurations emphasises the need and advantage of fine-tuning of hyperparameters. Further similarities to the existing corpus of research can be seen as the evaluation of model performance on unseen data adheres to recognised model validation criteria. This further supports how critical it is to assess how effectively models generalise to previously unseen data in order to ensure their practical use and trustworthiness.

However, there are some notable differences in the findings when compared to previous literature. For example, in contrast to previous research, 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 have difficulties. Overfitting or difficulties related to class imbalance might be significant contributors to this difference. Also, while Decision Tree performs well on the original dataset with selected important features, attaining a high ROC AUC score on unseen data, it underperforms on the oversampled dataset. This variation suggests that Decision Tree may be affected by dataset distribution and oversampling strategies. Random Forest’s constant good performance is impressive, although its performance gain on unseen data compared to Decision Tree is quite minor. This suggests that, while Random Forest reduces overfitting, it may fall short of entirely addressing the complexity or noise presented by the 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 further emphasises the significance of fine-tuning hyperparameters to avoid overfitting. When compared to the imbalanced dataset, the oversampled dataset presents a distinct data set of circumstances. While it effectively tackles class imbalance, it may create complexity that have an influence on model performance. This suggests that there is a need to take individual datasets on their merits when selecting and modifying models. Additionally, the usage of selected features impacts on model performance, with some models performing well with feature selection but others requiring additional fine-tuning, highlighting the importance of feature selection during the modelling phase.

The findings of this research demonstrates the significance of ROC AUC as a performance indicator, hyperparameter sensitivity, and the necessity to analyse model generalisation. However, they raise new challenges related to feature selection, which have an influence on model performance. These findings add to the understanding of how various models perform under different circumstances and highlight the need of careful model selection and tuning in machine learning models.

The thorough 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 directly addresses the research goal of objectively evaluating the performance of several supervised machine learning models and their hyperparameters. The aim is to determine the best model, or models, to solve the conversion prediction objective by rigorously assessing the performance of models such as Logistic Regression, Decision Tree, Random Forest, Gradient Boosting Classifier, and LightGBM.

The evaluation provides important 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 depending on the parameter configuration. 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 decline in generalisation when tested on 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 objectives, 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 training dataset and a strong ROC AUC of 0.78 on unseen test 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 training dataset and a notable ROC AUC of 0.82 on unseen data. The discovery of its performance addresses the research objectives, demonstrating a highly viable solution to the research problem.

In relation to feature selection, the analysis gives insights into how different models perform under different circumstances, which is well aligned with the research objectives. While Logistic Regression is stable on the training dataset, it still faces generalisation issues on the test dataset and unseen data. This highlights the need of dealing with generalisation in the context of feature selection.

The Decision Tree performs well the original dataset, and also maintains generalisation, achieving a ROC AUC of 0.81 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, addressing the research objectives by emphasising the model’s suitability to feature-selected datasets.

Overall, the research not only provides a thorough understanding of how different machine learning models perform, but it also directly addresses 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 data analytics and supervised machine learning applications in the life assurance industry.

**5.3 Future Research**

Future research in predicting life assurance application conversion offers a number of promising opportunities. The use of ensemble modelling, which integrates many machine learning techniques, has the potential to enhance performance. 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 expand.

Addressing imbalanced datasets remains critical. Sophisticated approaches such as SMOTE variants can assist with this. Future research could adopt Artificial Intelligence to 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 can improve models, while external data sources, such as economic indicators, can help to improve predictions. Dynamic models that can respond to market changes are equally required. Understanding the impact of consumer behaviour on conversion rates, particularly interactions and communication channels, could further inform machine learning strategy.

1. **Conclusion**

By utilising a wide range of analytical approaches and machine learning techniques, this research project has methodically tackled the challenging problem 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 importance research. These features showed strong association and consistently ranked high in relevance across a variety of 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 consistently identified Agency and WorkflowStatus as important features.

The performance of machine learning models was tested using ROC AUC as the primary evaluation metric, demonstrating how different models performed on both the imbalanced and oversampled datasets. Logistic Regression showed promise, but it has limitations in genralisability to unseen data. Both the Decision Tree and Random Forest models performed well on the original dataset, as well as on unseen data, while Random Forest indicated better generalisation. Gradient Boosting and LightGBM consistently performed well on both training and unseen data, with LightGBM demonstrating strong 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 gives important insights into the performance of several machine learning models in a life assurance context, it also stresses the importance of hyperparameter tuning, model generalisation, and feature selection in addressing the research problem. The research reveals the importance of ROC AUC as an assessment parameter, which is consistent with best practices in binary classification tasks. It has also emphasises the complexity imposed by dataset attributes, distributions and oversampling approaches, emphasising the need to take these elements into account when developing predictive models.

Finally, by presenting a thorough and data-driven method to tackling a real-world problem in the life assurance sector, the research contributes to the field of data analytics and machine learning. The findings provide practical insights that may help the life assurance 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.