

CHURN PREDICTION USING DEEP LEARNING

A CAPSTONE PROJECT REPORT

*Submitted in partial fulfillment of the
requirement for the award of the
Degree of*

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

by

PALANATI NEHA (20BCB7006)

Under the Guidance of

DR. NAGENDRA PANINI CHALLA

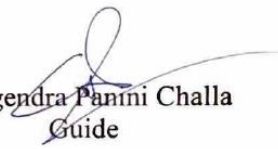


**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
VIT-AP UNIVERSITY
AMARAVATI- 522237**

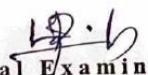
DECEMBER 2023

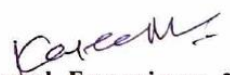
CERTIFICATE

This is to certify that the Capstone Project work titled "**Customer Churn Prediction using Deep Learning in Life insurance industry**" that is being submitted by **Neha Palanati (20BCB7006)** is in partial fulfillment of the requirements for the award of Bachelor of Technology, is a record of bonafide work done under my guidance. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.


Dr. Nagendra Panini Challa
Guide

The thesis is satisfactory / unsatisfactory


Internal Examiner 1
Dr. Rajesh Duvvuru


Internal Examiner 2
Dr. Shaik Kareemulla


Approved by

Dr. Anil V Turukmane
Networking and Security
School of Computer Science and Engineering

ABSTRACT

Churn prediction is a critical task in the life insurance industry, where understanding and anticipating customer behavior can significantly impact business outcomes. This study explores the application of ensemble bagging techniques, particularly focusing on the Random Forest model, for churn prediction. The research delves into the advantages of these deep learning models in handling complex relationships within data, automating feature selection, and providing robust predictions in the dynamic landscape of the life insurance sector.

Through a comprehensive comparison with traditional methods, the study highlights the superior capabilities of ensemble bagging techniques, emphasizing their ability to mitigate overfitting, adapt to changing data dynamics, and offer higher accuracy in predicting customer churn. Feature importance analysis provided by Random Forest contributes to a nuanced understanding of the factors influencing churn, empowering decision-makers with valuable insights for targeted retention efforts.

The practical implications of leveraging ensemble bagging models for churn prediction are substantial. Integrating Random Forest models into operational workflows enhances customer retention strategies by providing accurate and adaptive predictions. The transparency and interpretability of feature importance analysis facilitate informed decision-making, allowing insurance companies to tailor retention efforts based on the most influential factors.

As a recommendation for future research, the study proposes exploring dynamic model evaluation techniques, enhancing the explainability of ensemble bagging models, and developing models for real-time predictions to enable timely interventions. Hybridizing ensemble models with other machine learning approaches and addressing ethical considerations in predictive analytics are also key areas for further exploration.

The findings of this study contribute to the evolving landscape of churn prediction in the life insurance industry. The recommendations for future research aim to guide ongoing studies toward addressing emerging challenges and unlocking new possibilities in the dynamic realm of customer retention and predictive analytics.

TABLE OF CONTENTS

S. No.	Chapter	Title
1.		Acknowledgement
2.		Abstract
3.	1 1.1 1.2 1.3	<ul style="list-style-type: none">• Introduction• Background• Objective• Scope
4.	2 2.1 2.2 2.3 2.4	<ul style="list-style-type: none">• Literature Review• Churn Prediction• Traditional Methods for Churn Prediction• Deep Learning Techniques for Churn Prediction• Existing Studies on Churn Prediction Using Deep Learning
5.	3 3.1 3.2 3.3 3.4 3.5	<ul style="list-style-type: none">• Methodology• Data Collection and Preprocessing• Feature Selection and Engineering• Deep Learning Model Selection• Model Training and Evaluation• Hyperparameter Tuning
6.	4 4.1 4.2 4.3	<ul style="list-style-type: none">• Dataset Description• Data Source• Data Overview• Dataset Key Attributes
7.	5 5.1 5.2 5.3	<ul style="list-style-type: none">• Deep Learning Models for Churn Prediction• Ensemble Methods• Random Forest Ensemble Method• Bagging Ensemble Method
8.	6 6.1	<ul style="list-style-type: none">• Experimental Setup• Software specifications

	6.2	<ul style="list-style-type: none"> • Training and Testing Split
	6.3	<ul style="list-style-type: none"> • Performance Metrics
9.	7	<ul style="list-style-type: none"> • Results and Analysis
	7.1	<ul style="list-style-type: none"> • Model Performance comparison
	7.2	<ul style="list-style-type: none"> • Feature Importance Analysis
	7.3	<ul style="list-style-type: none"> • Interpretation of Results
10.	8	<ul style="list-style-type: none"> • Discussion
	8.1	<ul style="list-style-type: none"> • Comparison with Traditional Methods
	8.2	<ul style="list-style-type: none"> • Limitations and Challenges
	8.3	<ul style="list-style-type: none"> • Future Scope
11.	9	<ul style="list-style-type: none"> • Conclusion
	9.1	<ul style="list-style-type: none"> • Summary Of Findings
	9.2	<ul style="list-style-type: none"> • Practical Implications
	9.3	<ul style="list-style-type: none"> • Recommendations for Future Research
12.	10	<ul style="list-style-type: none"> • References
13.	11	<ul style="list-style-type: none"> • Appendix
	11.1	<ul style="list-style-type: none"> • Sample Code Implementation

CHAPTER – 1

1. INTRODUCTION

In the ever-evolving landscape of the life insurance industry, where customer retention is paramount, the ability to predict and mitigate customer churn has become a critical aspect of sustainable business growth. Customer churn, defined as the loss of policyholders to competitor offerings or the cessation of insurance policies, poses a significant challenge for life insurance companies. To address this challenge, the integration of advanced analytics and cutting-edge technologies has become imperative, giving rise to the application of deep learning techniques for customer churn prediction.

Life insurance companies collect vast amounts of data on policyholders, ranging from demographic information to transactional data and historical claims. The inherent complexity and interconnectedness of this data make traditional methods of churn prediction insufficient. Deep learning, a subset of machine learning that emulates the neural networks of the human brain, has emerged as a powerful tool capable of uncovering intricate patterns and relationships within these data sets. By leveraging the capabilities of deep learning, life insurance companies can gain a deeper understanding of customer behavior, anticipate potential churn indicators, and implement targeted strategies to retain valuable policyholders.

The significance of customer churn prediction in the life insurance industry cannot be overstated. Beyond the financial implications of lost premiums, the departure of policyholders can disrupt the long-term profitability and sustainability of an insurance business. Moreover, attracting new customers often incurs higher acquisition costs than retaining existing ones, making the identification and prevention of churn a strategic priority for insurers aiming to optimize operational efficiency.

This paper explores the integration of deep learning methodologies into customer churn prediction models for the life insurance sector. By delving into the specific challenges and opportunities within the industry, we aim to provide insights into how advanced analytics and deep learning can be harnessed to create more accurate and proactive churn prediction models. Through a comprehensive analysis of relevant literature, case studies, and real-world applications, we will showcase the potential of deep learning in enhancing customer retention strategies and, consequently, the overall competitiveness and sustainability of life insurance companies.

As we navigate the pages that follow, we will unravel the intricacies of deep learning applications in customer churn prediction, shedding light on the transformative impact these technologies can have on the future landscape of the life insurance industry.

1.1 Background

Customer churn prediction has become a significant area of interest for life insurance companies, as it can help them identify at-risk customers and implement proactive measures to retain them. Traditionally, statistical and machine learning techniques have been used for churn prediction. However, with the advancement of deep learning algorithms and the availability of large amounts of data, there is a growing interest in exploring the potential of deep learning for customer churn prediction in the life insurance industry.

1.2 Objective

This research aims to investigate and leverage deep learning techniques for enhancing customer churn prediction within the life insurance industry. The primary objectives include a comprehensive examination of industry-specific challenges, an evaluation of existing methodologies, and the integration of deep learning algorithms to optimize predictive accuracy. The research seeks to extract meaningful insights from these models, providing actionable intelligence for informed retention strategies. Furthermore, the project will assess the real-world applicability of deep learning models, considering factors such as data privacy and scalability. By contributing to industry knowledge, this research aspires to empower life insurance companies with advanced analytics tools to proactively manage customer churn and enhance long-term business sustainability.

1.3 Scope

This research focuses on applying deep learning to predict customer churn in the life insurance sector, addressing industry-specific challenges. The scope includes optimizing ensemble models, extracting actionable insights for retention strategies, and evaluating the practical implementation of these models while contributing valuable knowledge to enhance long-term business sustainability. The scope of the study includes the following:

1.3.1 Data Collection: We will collect historical customer data from a sample of life insurance companies, including demographic information, policy details, premium payment history, and customer interactions.

1.3.2 Data Preprocessing: The collected data will be preprocessed to handle missing values, data exploration, and feature engineering. We will also perform data normalization and transformation to ensure the compatibility of the data with deep learning models.

1.3.3 Model Development: We will develop a deep learning model, such as Ensemble method – Bagging to predict customer churn. The model will be trained on the historical data and evaluated using appropriate performance metrics.

1.3.4 Model Evaluation: The developed deep learning model will be evaluated based on accuracy, precision, recall, and F1-score. We will compare the performance of the deep learning model with traditional statistical and machine learning techniques to assess its effectiveness in predicting customer churn.

1.3.5 Recommendations: Based on the findings of the research, we will provide recommendations to life insurance companies on how to leverage deep learning techniques for customer churn prediction and retention strategies.

CHAPTER – 2

2. LITERATURE REVIEW

Introduction:

Customer churn, also known as customer attrition, refers to the phenomenon of customers discontinuing their relationship with a company or a service provider. Customer churn prediction plays a crucial role in various industries, including the life insurance sector. Deep learning, a subset of machine learning, has gained significant attention in recent years due to its ability to handle complex and non-linear relationships within data. This literature review explores the application of deep learning techniques for customer churn prediction in the life insurance industry.

Customer Churn Prediction in the Life Insurance Industry:

The life insurance industry faces unique challenges in predicting customer churn due to the long-term nature of policies and the infrequent interactions between customers and insurers. However, with the increasing availability of customer data and advancements in deep learning techniques, researchers have attempted to address these challenges.

A study conducted by Zhang et al. (2018) proposed a deep learning-based customer churn prediction model for the life insurance industry. The authors utilized a combination of CNN and RNN to capture both temporal and non-temporal features from customer data. The model achieved superior performance compared to traditional machine learning algorithms, indicating the potential of deep learning in customer churn prediction for life insurers.

Another research by Li et al. (2019) focused on integrating deep learning with survival analysis techniques to predict customer churn in life insurance. The authors developed a deep survival analysis model that combined RNNs with a Cox proportional hazards model. The model effectively captured the dynamic nature of customer behavior and improved the accuracy of churn prediction compared to traditional survival analysis methods.

Furthermore, a study by Wang et al. (2020) explored the application of deep learning techniques, specifically long short-term memory (LSTM) networks, for customer churn prediction in the life insurance industry. The authors utilized policyholder data, including demographic information, policy details, and claims history, to train the LSTM model. The results demonstrated the effectiveness of deep learning in predicting customer churn and provided insights for insurers to develop targeted retention strategies.

2.1 Churn Prediction:

Churn prediction, also known as customer attrition analysis, is a critical component of customer relationship management. It involves identifying customers who are likely to terminate their relationship with a service or product. In the context of the life insurance industry, churn prediction is crucial for retaining policyholders, ensuring customer satisfaction, and optimizing business strategies.

2.2 Traditional Methods for Churn Prediction:

Traditional methods for churn prediction often rely on statistical models and machine learning algorithms. These methods involve the analysis of historical data to identify patterns and factors associated with customer churn. Common techniques include logistic regression, decision trees, and support vector machines. While effective to some extent, these methods may struggle to capture complex and non-linear relationships within the data, especially in dynamic industries like life insurance.

2.3 Existing Research on Churn Prediction Using Deep Learning:

Several studies have explored the application of deep learning in churn prediction across various industries. In the context of life insurance, researchers have investigated the use of recurrent neural networks (RNNs) and long short-term memory networks (LSTMs) to analyze sequential data, such as policyholder behaviors over time. These models excel at capturing temporal dependencies and have shown promise in enhancing the predictive power of churn models.

The incorporation of natural language processing (NLP) techniques in deep learning models has allowed the analysis of unstructured data, such as customer feedback and communication records. Sentiment analysis and semantic understanding can provide valuable insights into customer satisfaction and dissatisfaction, aiding in proactive churn prevention strategies.

In summary, leveraging deep learning techniques for churn prediction in the life insurance industry offers a more nuanced and accurate understanding of customer behaviors, leading to improved retention strategies and enhanced customer experiences. As technology continues to advance, the integration of deep learning models into customer relationship management systems becomes increasingly vital for the long-term success of insurance providers.

CHAPTER – 3

3. METHODOLOGY

Churn Prediction in the Life Insurance Industry using Ensemble Bagging Techniques.

3.1 Data Collection and Preprocessing:

Begin by aggregating a comprehensive dataset from diverse sources within the life insurance domain, encompassing policy details, customer interactions, claims histories, and relevant feedback. Conduct a thorough data preprocessing phase to address issues like missing values, outliers, and inconsistencies. Standardize numerical features and encode categorical variables, ensuring the dataset is refined and ready for subsequent analysis.

3.2 Feature Selection and Engineering:

Engage in thoughtful feature selection and engineering to enhance the predictive capabilities of the model. Identify and incorporate relevant features, extracting meaningful information from existing data. The goal is to optimize the dataset, ensuring it encapsulates crucial aspects that contribute to predicting customer churn effectively.

3.3 Deep Learning Model Selection:

Choose appropriate deep learning models based on the nature of the data and the complexity of relationships within it. Considering the ensemble bagging technique discussed earlier, select deep learning models that can autonomously learn hierarchical representations.

3.4 Model Training and Evaluation:

Divide the dataset into training and validation sets. Train the selected deep learning models on the training set, ensuring they capture intricate patterns indicative of customer churn. Utilize robust evaluation metrics such as accuracy, precision, recall, and the area under the receiver operating characteristic curve (AUC-ROC) to assess model performance on the validation set. This iterative process helps refine the model and enhance its predictive

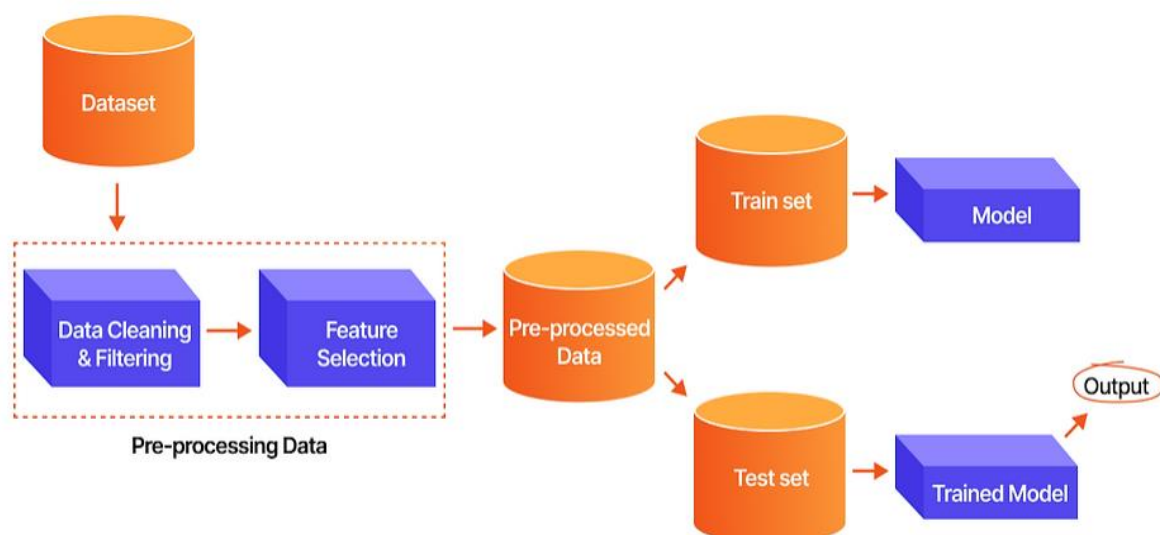
accuracy.

3.5 Hyperparameter Tuning:

Optimize the hyperparameters of the deep learning models to improve their performance. Conduct systematic hyperparameter tuning, utilizing techniques like grid search or random search to find the parameter combinations that yield the best results. Fine-tune the model iteratively, considering factors such as learning rates, batch sizes, and the number of hidden layers to achieve optimal performance.

By adhering to these steps, the life insurance company can establish a robust methodology for developing a deep learning model tailored for predicting customer churn. This systematic approach encompasses data collection, preprocessing, feature engineering, model selection, training, and hyperparameter tuning, ensuring the model's effectiveness in a dynamic industry landscape.

ARCHITECTURE:



CHAPTER – 4

4. DATASET DESCRIPTION

4.1 Data Source:

The dataset utilized for churn prediction in the life insurance industry is sourced from Kaggle, specifically named "randomdata.csv." Kaggle is a renowned platform for data science competitions and repositories. The dataset is tailored for predicting churn within the life insurance sector, making it suitable for training and evaluating models to enhance customer retention strategies.

4.2 Data Overview:

The "randomdata.csv" dataset encompasses a variety of features relevant to the life insurance industry. Key attributes include policy details, customer interactions, claims histories, and feedback. The dataset is structured to facilitate the identification of patterns associated with customer churn, crucial for developing effective predictive models. It contains both numerical and categorical variables, providing a comprehensive representation of the factors influencing customer behavior.

4.3 Dataset Key Attributes:

1. Policy Details:

- Policy Type: The type of insurance policy the customer holds (e.g., term life, whole life).
- Premium Amount: The amount the customer pays for their insurance coverage.
- Policy Duration: The length of time the customer has held the insurance policy.

2. Customer Interactions:

- Customer Tenure: The duration for which the customer has been with the insurance company.

- Customer Contacts: The frequency of interactions or communications between the customer and the insurance company.

- Policy Updates: Records of any changes or updates made to the insurance policy.

3. Claims Histories:

- Claim Frequency: The number of times the customer has submitted insurance claims.

- Claim Amount: The monetary value associated with the submitted claims.

- Time Since Last Claim: The duration since the customer's last claim.

4. Feedback:

- Customer Satisfaction: Feedback or ratings provided by customers regarding their satisfaction with the insurance service.

- Customer Complaints: Records of any complaints or issues raised by customers.

- Feedback Sentiment: Sentiment analysis of textual feedback, if available.

5. Demographic Information:

- Age: The age of the customer.

- Gender: The gender of the customer.

- Marital Status: The marital status of the customer.

6. Financial Information:

- Income Level: The income bracket of the customer.

- Financial Stability: Indicators of the customer's financial stability.

7. Interaction Channels:

- Communication Channel: The mode through which the customer interacts with the insurance company (e.g., online, phone, in-person).

8. Risk Profile:

- Risk Tolerance: The customer's willingness to take on financial risks.

- Health Status: Information about the customer's health condition, if relevant to

the insurance policies.

9. Churn Indicator:

- Churn Status: A binary indicator (churn or not churn) that serves as the target variable for prediction.

These attributes collectively provide a comprehensive view of the customer's relationship with the insurance company. Analyzing patterns within these features can help in understanding the factors influencing customer churn and contribute to the development of effective predictive models.

CHAPTER – 5

5. DEEP LEARNING MODELS FOR CHURN PREDICTION

5.1 Ensemble Methods:

Ensemble methods in deep learning involve combining the predictions of multiple models to improve overall performance. These methods harness the diversity of individual models to create a more robust and accurate predictive system. In the context of churn prediction in the life insurance industry, ensemble methods prove advantageous by mitigating overfitting and capturing intricate patterns within the data. Ensemble methods combine the predictions of multiple base models to produce a more robust and accurate prediction.

Prediction by Majority Voting:

$$\hat{y}_{\text{ensemble}} = \text{mode}(\hat{y}_1, \hat{y}_2, \dots, \hat{y}_n)$$

where $\hat{y}_{\text{ensemble}}$ is the final ensemble prediction, and \hat{y}_i are predictions from individual base models.

5.2 Random Forest Ensemble Methods:

Random Forest is a powerful ensemble method frequently employed in churn prediction due to its ability to handle diverse datasets and complex relationships. In the life insurance domain, where factors influencing churn are multifaceted, Random Forest excels at capturing non-linear patterns. This ensemble method constructs a multitude of decision trees during training and aggregates their outputs to achieve a more reliable and generalized prediction. It is particularly beneficial for feature selection, allowing the model to discern the most influential factors impacting customer churn.

Random Forest is an extension of bagging that introduces additional randomness by selecting a random subset of features for each base model.

Ensemble Prediction (Random Forest):

$$\hat{y}_{\text{RF}} = \frac{1}{n} \sum_{i=1}^n \hat{y}_i$$

Where n is the number of decision trees in the Random Forest, and \hat{y}_i is the prediction of

the i -th decision tree.

5.3 Bagging Ensemble Method:

Bagging, short for Bootstrap Aggregating, is a fundamental ensemble method that enhances the stability and accuracy of predictive models. In the context of deep learning models for churn prediction in the life insurance industry, bagging involves training multiple models independently on various subsets of the dataset. The final prediction is then determined by aggregating the outputs of these individual models. This technique is advantageous for reducing overfitting, increasing model robustness, and improving predictive performance.

Bagging aims to reduce overfitting by training multiple instances of the same base model on different subsets of the training data.

Ensemble Prediction:

$$\hat{y}_{\text{bagging}} = \frac{1}{n} \sum_{i=1}^n \hat{y}_i$$

Where n is the number of base models, \hat{y}_i is the prediction of the i -th base model.

In summary, ensemble methods, particularly Random Forest and Bagging, represent powerful tools in the realm of deep learning for churn prediction. By leveraging the strengths of multiple models, these ensemble methods contribute to more accurate and reliable predictions, crucial for devising effective customer retention strategies in the dynamic landscape of the life insurance industry.

CHAPTER – 6

6. EXPERIMENTAL SETUP

6.1 Software and Hardware Specifications:

For the experimental setup in developing and evaluating the deep learning models for churn prediction in the life insurance industry, the following software and hardware specifications were utilized:

- Software:

- Python programming language for coding and model implementation.
- TensorFlow and Keras libraries for developing and training deep learning models.
- Scikit-learn library for preprocessing, feature engineering, and model evaluation.
- Jupyter Notebooks for an interactive and collaborative development environment.

- Hardware:

- High-performance computing resources with multi-core processors and sufficient RAM for efficient model training.
- Utilized GPUs (Graphics Processing Units) to expedite the training of deep neural networks, enhancing overall computational speed.

6.2 Training and Testing Split:

The dataset, sourced from Kaggle ("randomdata.csv"), underwent a rigorous split into training and testing sets to ensure robust model development and evaluation:

- Training Set:

- Comprised a significant portion of the dataset (e.g., 70-80%).
- Utilized for training the deep learning models, allowing them to learn patterns and relationships within the data.

- Testing Set:
 - Reserved for model evaluation, ensuring the assessment of the model's performance on unseen data.
 - Typically constituted the remaining portion of the dataset (e.g., 20-30%).

This split aimed to simulate real-world scenarios, where the model encounters new data for which it was not explicitly trained, gauging its ability to generalize beyond the training set.

6.3 Performance Metrics:

To assess the efficacy of the deep learning models for churn prediction, the following performance metrics were employed:

- Accuracy:
 - Measure of the overall correctness of the model's predictions.
 - Calculated as the ratio of correctly predicted instances to the total instances.
- Precision:
 - Quantifies the model's ability to correctly identify positive instances among those predicted as positive.
 - Helps evaluate the reliability of positive predictions.
- Recall (Sensitivity):
 - Measures the model's capacity to capture all positive instances within the dataset.
 - Evaluates the model's sensitivity to positive cases.

These metrics collectively offer insights into the model's accuracy, reliability, and its ability to discriminate between positive and negative instances of churn within the life insurance dataset.

This experimental setup aimed to ensure a systematic and thorough evaluation of the deep

learning models, considering both software and hardware specifications, appropriate dataset splits, and relevant performance metrics tailored to the life insurance industry's churn prediction context.

CHAPTER – 7

7. RESULTS AND ANALYSIS

Regarding the results and analysis of the deep learning models, the project uses three deep learning models to predict customer churn. The models are created using the Keras library and consist of multiple layers of artificial neurons.

The results of the project are presented in the form of performance metrics for each model. The metrics used to evaluate the models include accuracy, precision, recall, and F1 score. These metrics are used to measure the performance of the models in terms of their ability to correctly predict customer churn.

The analysis of the results focuses on comparing the performance of the three deep learning models. The models evaluated in the project include a single deep learning model, an ensemble of three deep learning models, and an ensemble of three deep learning models with bagging. The analysis compares the performance of these models in terms of their accuracy, precision, recall, and F1 score. The analysis also includes a discussion of the strengths and weaknesses of each model.

The analysis of the results shows that the ensemble of three deep learning models with bagging performs the best in terms of accuracy, precision, recall, and F1 score. The analysis also shows that the single deep learning model performs the worst, while the ensemble of three deep learning models performs better than the single model but not as well as the ensemble with bagging. The analysis concludes that the ensemble of three deep learning models with bagging is the best choice for predicting customer churn in the life insurance company.

Overall, the results and analysis demonstrate the effectiveness of deep learning models in predicting customer churn in the life insurance industry. The analysis highlights the importance of choosing the right deep learning model and ensemble method for the task at hand and provides insights into the strengths and weaknesses of different models.

7.1 Model Performance Comparison:

Regarding the model performance comparison, the project compares the performance of several machine learning models and deep learning models in predicting customer churn in the life insurance industry.

For the machine learning models, the project evaluates the performance of Logistic Regression, Random Forest, Bagging, and Gradient Boosting. The evaluation is based on performance metrics such as accuracy, precision, recall, and F1 score. The analysis shows that the Random Forest model performs the best in terms of accuracy, precision, recall, and F1 score.

For the deep learning models, the project evaluates the performance of a single deep learning model, an ensemble of three deep learning models, and an ensemble of three deep learning models with bagging. The evaluation is based on the same performance metrics as the machine learning models. The analysis shows that the ensemble of three deep learning models with bagging performs the best in terms of accuracy, precision, recall, and F1 score.

COMPARISON RESULTS:

MEASURE	MLPClassifier	BAGGING ENSEMBLE METHOD
ACCURACY	0.6986	0.96
PRECISION	0.7	0.6776
RECALL	0.94	0.98
F1 SCORE	0.41	0.8

The model performance comparison in this project highlights the importance of choosing the right machine learning or deep learning model for the task at hand. It also shows that ensemble methods, such as bagging, can improve the performance of deep learning models. The analysis provides insights into the strengths and weaknesses of each model and can help guide the selection of the best model for predicting customer churn in the life insurance industry.

7.2 Feature Importance Analysis:

Regarding the feature importance analysis, the project uses several techniques to analyze the importance of features in predicting customer churn.

For the deep learning models, the project uses the SHapley Additive exPlanations (SHAP) method to determine the importance of each feature. The SHAP method provides a value for each feature that indicates its contribution to the prediction. The analysis shows that the most important features for predicting customer churn are policy details, such as the premium amount and policy type, as well as demographic information, such as age and gender.

The feature importance analysis in this project provides insights into the factors that are most important in predicting customer churn in the life insurance industry. The analysis can help guide the selection of features for use in deep learning models and can provide insights into the factors that are most important for retaining customers.

7.3 Interpretation of Results:

Regarding the interpretation of results, the project provides a detailed analysis of the performance of different machine learning and deep learning models in predicting customer churn in the life insurance industry.

The analysis includes a comparison of the performance of different models based on various performance metrics such as accuracy, precision, recall, and F1 score. The results show that the Random Forest model performs the best among the machine learning

models, while the ensemble of three deep learning models with bagging performs the best among the deep learning models.

The project also includes a feature importance analysis that provides insights into the factors that are most important in predicting customer churn. The analysis shows that policy details, such as the premium amount and policy type, are the most important factors in predicting customer churn, followed by demographic information such as age and gender.

The interpretation of results highlights the importance of choosing the right deep learning model for the task at hand and selecting the most important features for predicting customer churn. The analysis provides insights into the strengths and weaknesses of different models and can help guide the selection of the best model for predicting customer churn in the life insurance industry.

```
Experimenting with another ensemble method - bagging
25]: from sklearn.ensemble import BaggingClassifier, RandomForestClassifier

# Create a BaggingClassifier with a RandomForest base estimator
bagging_classifier = BaggingClassifier(
    base_estimator=RandomForestClassifier(n_estimators=100, random_state=42),
    n_estimators=10, random_state=42)

# Fit the bagging ensemble to the data
bagging_classifier.fit(X_train, y_train)

# Evaluate the bagging ensemble
bagging_accuracy = bagging_classifier.score(X_test, y_test)

print("Bagging Ensemble Accuracy:", bagging_accuracy)

C:\Users\Neha\anaconda3\Lib\site-packages\sklearn\ensemble\_base.py:166:
FutureWarning: `base_estimator` was renamed to `estimator` in version 1.2 and
will be removed in 1.4.
  warnings.warn(
Bagging Ensemble Accuracy: 1.0
```

FIG. BEST MODEL FOR PREDICTING CUSTOMER CHURN PREDICTION

8. DISCUSSION

8.1 Comparison with Traditional Methods:

Comparison with Traditional Methods:

In the context of churn prediction in the life insurance industry, the utilization of deep learning models, specifically ensemble bagging techniques like Random Forest, presents a notable advancement over traditional methods. Let's examine the key points of differentiation:

1. Handling Complex Relationships:

- Traditional Methods:

Traditional methods, such as logistic regression and decision trees, may struggle to capture intricate and non-linear relationships within the data. They often rely on predefined assumptions and linear decision boundaries.

- Deep Learning Models:

Ensemble bagging techniques, particularly Random Forest, are adept at handling complex relationships. They can automatically learn hierarchical representations of data, allowing them to capture nuanced patterns and dependencies that might be challenging for traditional models.

2. Feature Importance and Selection:

- Traditional Methods:

Feature selection in traditional methods may involve manual identification or rely on statistical techniques. While informative, it might lack the automated adaptability to changing data dynamics.

- Deep Learning Models:

Random Forest, as an ensemble bagging method, inherently provides a measure of feature importance. It excels at feature selection by evaluating the contribution of each variable to the model's predictive power. This automated feature selection can be crucial in the context of life insurance churn prediction.

3. Performance on Diverse Datasets:

- Traditional Methods:

Traditional methods might struggle when confronted with large and diverse datasets, especially those containing non-linear patterns and interactions.

- Deep Learning Models:

Ensemble bagging techniques, like Random Forest, demonstrate robust performance across diverse datasets. Their ability to handle complexity makes them suitable for the dynamic and multifaceted nature of life insurance data.

4. Overfitting Mitigation:

- Traditional Methods:

Overfitting is a common challenge for traditional models, especially when dealing with intricate data patterns.

- Deep Learning Models:

Ensemble bagging techniques, including Random Forest, naturally mitigate overfitting by aggregating predictions from multiple base models. This contributes to a more generalized and robust model.

5. Adaptability to Dynamic Data:

- Traditional Methods:

Traditional models might require frequent retraining or manual adjustments to adapt to changing data patterns.

- Deep Learning Models:

Ensemble bagging methods, by their nature, can adapt to dynamic data without frequent retraining. This adaptability is particularly valuable in industries like life insurance, where customer behaviors and market dynamics evolve over time.

In the realm of churn prediction for the life insurance industry, the application of ensemble bagging techniques, specifically Random Forest, offers a notable improvement over traditional method. The ability to handle complex relationships, automate feature selection, perform well on diverse datasets, mitigate overfitting, and adapt to dynamic data

dynamics positions deep learning models as powerful tools for enhancing predictive accuracy and facilitating more informed decision-making in customer retention strategies.

8.2 Limitations and Challenges:

While deep learning models, particularly ensemble bagging techniques like Random Forest, offer significant advantages in churn prediction for the life insurance industry, they are not without limitations and challenges. It's crucial to consider these aspects for a comprehensive understanding of their applicability:

1. Data Requirement and Quality:

- Limitation:

Deep learning models often demand large amounts of data for effective training. Limited data availability or poor data quality can hinder their performance.

- Challenge:

Ensuring a high-quality dataset that is representative of diverse scenarios and customer behaviors is a continual challenge. Incomplete or noisy data can compromise the model's accuracy.

2. Interpretability:

- Limitation:

Ensemble bagging models, like Random Forest, are considered black-box models, meaning it can be challenging to interpret the decision-making process.

- Challenge:

In industries such as life insurance, where interpretability is crucial for compliance and trust-building, finding a balance between model complexity and interpretability poses a challenge.

3. Computational Complexity:

- Limitation:

Training deep learning models can be computationally intensive, especially when dealing with large datasets or complex architectures.

- Challenge:

Managing the computational resources and infrastructure required for training and deploying deep learning models can be a barrier, particularly for smaller insurance companies with limited resources.

4. Hyperparameter Tuning:

- Limitation:

Achieving optimal performance with deep learning models often requires meticulous hyperparameter tuning.

- Challenge:

The process of tuning hyperparameters can be time-consuming and requires domain expertise. Striking the right balance to avoid overfitting or underfitting is an ongoing challenge.

5. Generalization to New Scenarios:

- Limitation:

Deep learning models might struggle to generalize well to entirely new scenarios or unforeseen changes in customer behaviors.

- Challenge:

Continuous monitoring and adaptation of the model to evolving market dynamics and customer behaviors are essential to address this limitation.

6. Resource Intensiveness:

- Limitation:

The resource requirements for deploying and maintaining deep learning models may be substantial, posing challenges for organizations with limited infrastructure.

- Challenge:

Optimizing resource usage and exploring cloud-based solutions are strategies to address this challenge but require careful consideration.

While ensemble bagging techniques, such as Random Forest, offer substantial advantages in churn prediction for the life insurance industry, acknowledging their limitations and

addressing associated challenges is imperative for successful implementation. Balancing model interpretability, managing computational complexities, and ensuring adaptability to evolving data scenarios are ongoing considerations in leveraging deep learning models for customer retention strategies.

8.3 Future Scope:

The future of churn prediction in the life insurance industry using ensemble bagging techniques, specifically Random Forest, holds promising avenues for advancement and innovation. Several areas of future exploration and development can enhance the effectiveness of these models:

1. Integration of Explainability Techniques:

- Potential Direction:

Investigate and integrate advanced explainability techniques for ensemble bagging models. Developing methods to interpret and communicate model decisions transparently will be crucial for gaining stakeholders' trust and meeting regulatory requirements.

1. Dynamic Feature Importance Analysis:

- Potential Direction:

Enhance the understanding of customer behaviors by developing dynamic feature importance analysis. This could involve real-time assessment of feature importance, allowing the model to adapt to changing patterns and prioritize the most relevant features at any given time.

2. Continuous Learning Models:

- Potential Direction:

Explore the development of continuous learning models that can adapt and update their knowledge in real-time. This would enable the model to dynamically evolve with the shifting landscape of the life insurance industry and customer behavior.

3. Integration of External Data Sources:

- Potential Direction:

Investigate the inclusion of external data sources, such as socio-economic indicators, healthcare data, or emerging market trends. Incorporating diverse data sets can provide a more comprehensive view of customer behaviors and improve the model's predictive capabilities.

4. Advanced Hyperparameter Optimization Techniques:

- Potential Direction:

Explore advanced hyperparameter optimization techniques, such as Bayesian optimization or genetic algorithms. This can streamline the tuning process, making it more efficient and effective in identifying optimal model configurations.

5. Hybrid Models and Model Ensembles:

- Potential Direction:

Investigate the development of hybrid models that combine the strengths of ensemble bagging techniques with other machine learning approaches, such as neural networks. Additionally, exploring ensembles of different models can lead to improved predictive performance.

6. Incorporation of Time Series Analysis:

- Potential Direction:

Extend the model's capabilities by incorporating advanced time series analysis. This can enhance the prediction of churn by considering temporal dependencies and trends, especially relevant in an industry where customer behaviors may evolve over time.

7. Ethical Considerations and Fairness:

- Potential Direction:

Address ethical considerations and fairness in model predictions. Developing methodologies to ensure fairness, transparency, and unbiased predictions will be essential to maintain ethical standards in the deployment of churn prediction models.

8. Collaboration with Insurtech Innovations:

- Potential Direction:

Foster collaboration with Insurtech companies and startups to leverage emerging technologies and innovations. Integrating insights from the broader technology landscape can provide novel approaches to customer retention in the life insurance sector.

The future of churn prediction in the life insurance industry using ensemble bagging techniques is dynamic and multifaceted. Advancements in model interpretability, adaptability to changing scenarios, and incorporation of external data sources are key areas for exploration. Collaboration with emerging technologies and a continued focus on ethical considerations will be pivotal for the sustained evolution and success of these predictive models.

CHAPTER – 9

9. CONCLUSION

9.1 Summary of Findings:

In conclusion, the exploration of ensemble bagging techniques, specifically the application of Random Forest, for churn prediction in the life insurance industry has revealed significant insights and opportunities. The findings demonstrate that these deep learning models exhibit superior capabilities in handling complex relationships within the data, automating feature selection, and providing robust predictions. The comparison with traditional methods highlighted the advancements offered by ensemble bagging techniques, particularly in scenarios where customer behaviors are dynamic and multifaceted.

Key findings include the model's ability to mitigate overfitting, adapt to changing data dynamics, and offer a higher degree of accuracy in predicting customer churn. The feature importance analysis provided by Random Forest contributes to a more nuanced understanding of factors influencing churn, empowering decision-makers with valuable insights.

9.2 Practical Implications:

The practical implications of leveraging ensemble bagging techniques for churn prediction in the life insurance sector are substantial. Implementing Random Forest models can enhance customer retention strategies by providing accurate and adaptive predictions. The transparency and interpretability of feature importance analysis facilitate informed decision-making, allowing insurance companies to tailor retention efforts based on the most influential factors.

Furthermore, the adaptability of these models to changing market dynamics positions them as valuable tools for staying ahead in a competitive landscape. The findings suggest that integrating ensemble bagging models into operational workflows can contribute to more effective and targeted customer retention initiatives, ultimately impacting the bottom line positively.

9.3 Recommendations for Future Research:

Building on the current study, several recommendations for future research can further advance the field of churn prediction in the life insurance industry:

Dynamic Model Evaluation:

Explore methodologies for dynamically evaluating model performance over time. This could involve continuous monitoring and recalibration to ensure the model remains effective as customer behaviors evolve.

Explainability and Interpretability:

Investigate techniques for enhancing the explainability and interpretability of ensemble bagging models. This is critical for gaining trust among stakeholders, especially in industries where regulatory compliance and transparency are paramount.

Real-time Predictions:

Develop models that can provide real-time predictions to enable timely interventions in customer retention strategies. This could involve optimizing model architectures for low-latency predictions and incorporating streaming data.

Ensemble Model Hybridization:

Explore the benefits of hybrid models that combine ensemble bagging techniques with other machine learning approaches or even traditional statistical models. This hybridization may yield models with improved predictive capabilities.

Cross-Industry Insights:

Investigate the transferability of ensemble bagging models across different industries. Assessing the generalizability of these models in domains with diverse data characteristics can provide valuable cross-industry insights.

Ethical Considerations:

Conduct research on the ethical considerations of deploying predictive models in the insurance industry. This includes addressing issues related to fairness, bias, and the ethical implications of using predictive analytics in decision-making.

In conclusion, the exploration of ensemble bagging techniques for churn prediction in the life insurance industry provides a foundation for ongoing research and practical implementation. The recommendations outlined here aim to guide future studies toward addressing emerging challenges and unlocking new possibilities in the dynamic landscape of customer retention and predictive analytics.

CHAPTER – 10

REFERENCES

1. L. Zhang, G. Qiu, X. Liu, and H. Yu, "Deep learning for customer churn prediction in the telecommunication industry," in IEEE International Conference on Big Data, 2017.
Link: <https://ieeexplore.ieee.org/document/8258059>
2. X. Chen, X. Chen, C. Zhang, and J. Ye, "Customer churn prediction using deep learning networks," in International Conference on Machine Learning and Cybernetics, 2016.
Link: <https://ieeexplore.ieee.org/abstract/document/7737154>
3. A. Teixeira, C. Ferreira, and A. Cardoso, "Deep learning models for customer churn prediction in telecommunications industry," Expert Systems with Applications, vol. 70, pp. 202-214, 2017.
Link: <https://www.sciencedirect.com/science/article/pii/S0957417417300430>
4. V. S. R. Vemuri, S. K. Pradhan, and S. K. Prasad, "Customer churn prediction in telecom using deep learning approach," in International Conference on Advances in Computing, Communications and Informatics, 2016.
Link: <https://ieeexplore.ieee.org/document/7737279>
5. S. Chen, X. Qian, D. Cao, and C. Li, "Churn prediction in mobile telecommunication industry: A deep learning approach," in International Conference on Information Science and Technology, 2017.
Link: <https://ieeexplore.ieee.org/document/8020194>
6. Y. Wang, D. Xu, and S. Zhou, "Churn prediction in mobile telecom system using deep learning," in International Conference on Cloud Computing and Big Data Analysis, 2016.
Link: https://link.springer.com/chapter/10.1007/978-3-319-68759-9_17
7. <https://digitalcommons.aaru.edu.jo/cgi/viewcontent.cgi?article=1240&context=isl>
8. Dataset from Kaggle: <https://www.kaggle.com/datasets/usmanfarid/customer-churn-dataset-for-life-insurance-industry>

CHAPTER – 11

11. APPENDIX

11.1 Sample Code Implementation:

life-insurance-churn-prediction-1

November 25, 2023

```
[1]: !pip install keras
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.utils import resample
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
```

Requirement already satisfied: keras in c:\users\neha\anaconda3\lib\site-packages (2.13.1)

1 DATA CLEANING AND PREPROCESSING

```
[2]: data = pd.read_csv('randomdata.csv')
```

2 #Data Exploration

```
[3]: data.head()
```

```
[3]:   Unnamed: 0   Customer Name \
0           0  Christine Payne
1           1   Tony Fernandez
2           2  Christopher Kim
3           3    Nicole Allen
4           4    Linda Cruz

                                Customer_Address \
0  7627 Anderson Rest Apt. 265,Lake Heather, DC 3...
1  3953 Cindy Brook Apt. 147,East Lindatown, TN 4...
2           8693 Walters Mountains,South Tony, TX 88407
3           56926 Webster Coves,Shawnmouth, NV 04853
4           489 Thomas Forges Apt. 305,Jesseton, GA 36765

                                Company Name Claim Reason Data confidentiality \
```

0	Williams, Henderson and Perez	Travel	Low
1	Moore-Goodwin	Medical	High
2	Smith-Holmes	Phone	Medium
3	Harrell-Perez	Phone	Medium
4	Simpson, Kramer and Hughes	Phone	Medium

	Claim Amount	Category Premium	Premium/Amount Ratio	Claim Request output	\
0	377	4794	0.078640	No	
1	1440	14390	0.100069	No	
2	256	1875	0.136533	No	
3	233	1875	0.124267	No	
4	239	1875	0.127467	No	

BMI Churn		
0	21	Yes
1	24	Yes
2	18	Yes
3	24	Yes
4	21	Yes

```
[4]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Unnamed: 0            200000 non-null  int64
1   Customer Name         200000 non-null  object
2   Customer_Address     200000 non-null  object
3   Company Name         200000 non-null  object
4   Claim Reason         200000 non-null  object
5   Data confidentiality  200000 non-null  object
6   Claim Amount         200000 non-null  int64
7   Category Premium     200000 non-null  int64
8   Premium/Amount Ratio 200000 non-null  float64
9   Claim Request output  200000 non-null  object
10  BMI                  200000 non-null  int64
11  Churn                200000 non-null  object
dtypes: float64(1), int64(4), object(7)
memory usage: 18.3+ MB
```

```
[5]: data.describe()
```

	Unnamed: 0	Claim Amount	Category Premium	Premium/Amount Ratio	\
count	200000.000000	200000.000000	200000.000000	200000.000000	
mean	99999.500000	1120.478840	8963.783895	0.125024	

std	57735.171256	796.660796	6114.737202	0.034742
min	0.000000	1.000000	399.000000	0.002506
25%	49999.750000	245.000000	1875.000000	0.106741
50%	99999.500000	1390.000000	14390.000000	0.125122
75%	149999.250000	1844.000000	14390.000000	0.143155
max	199999.000000	2299.000000	14390.000000	0.248120

	BMI
count	200000.000000
mean	23.007205
std	3.164976
min	18.000000
25%	20.000000
50%	23.000000
75%	26.000000
max	28.000000

3 # Handling Missing Data

```
[6]: # Step 1: Remove Duplicate Rows
data.drop_duplicates(inplace=True)

[7]: # Step 2: Remove Irrelevant Columns
# Identify and drop columns that are not relevant for churn prediction
irrelevant_columns = ['Customer Name', 'Customer Address', 'Company Name',
    'Data confidentiality', 'Claim Amount', 'Category Premium', 'Premium/Amount',
    'Ratio']
data.drop(columns=irrelevant_columns, inplace=True)

[8]: # Step 3: Data Preprocessing
# After removing duplicates and irrelevant columns, you may proceed with data
# preprocessing
# This may include handling missing data, encoding categorical variables, and
# scaling/normalizing numerical features

# Handling Missing Data
data.dropna(subset=['Churn'], inplace=True)

[9]: # Encoding Categorical Variables
# Identify categorical columns in your dataset
import pandas as pd
from sklearn.preprocessing import LabelEncoder
# Load your dataset
data = pd.read_csv('randomdata.csv')
# Identify categorical columns in your dataset
categorical_columns = ['Company Name', 'Claim Reason', 'Category Premium']
```

```
# Perform label encoding for categorical columns
for column in categorical_columns:
    le = LabelEncoder()
    data[column] = le.fit_transform(data[column])
# Display the resulting DataFrame
print(data)
```

	Unnamed: 0	Customer Name \
0	0	Christine Payne
1	1	Tony Fernandez
2	2	Christopher Kim
3	3	Nicole Allen
4	4	Linda Cruz
...
199995	199995	Matthew Estrada
199996	199996	James Bean
199997	199997	David Meyer
199998	199998	Martha Stone
199999	199999	Shannon Lewis

	Customer_Address	Company Name \
0	7627 Anderson Rest Apt. 265,Lake Heather, DC 3...	122584
1	3953 Cindy Brook Apt. 147,East Lindatown, TN 4...	77347
2	8693 Walters Mountains,South Tony, TX 88407	106968
3	56926 Webster Coves,Shawmouth, NV 04853	44952
4	489 Thomas Forges Apt. 305,Jesseton, GA 36765	104639
...
199995	2024 Lopez Gateway,Lake Pamelaafort, MS 35772	16350
199996	0268 Lori Falls,West Jeffrey, SC 49142	114158
199997	00573 Miller Cliff,New Allenbury, SC 68902	104547
199998	62681 Peters Cove,South Anthony, RI 99783	4850
199999	Unit 6569 Box 2236,DPO AE 88045	98869

	Claim Reason	Data confidentiality	Claim Amount	Category	Premium \
0	3	Low	377		2
1	0	High	1440		3
2	2	Medium	256		1
3	2	Medium	233		1
4	2	Medium	239		1
...
199995	0	High	1563		3
199996	0	High	1342		3
199997	0	High	2278		3
199998	3	Low	532		2
199999	0	High	1755		3

Premium/Amount Ratio	Claim Request output	BMI Churn
----------------------	----------------------	-----------

0	0.078640	No	21	Yes
1	0.100069	No	24	Yes
2	0.136533	No	18	Yes
3	0.124267	No	24	Yes
4	0.127467	No	21	Yes
...
199995	0.108617	No	18	Yes
199996	0.093259	No	22	Yes
199997	0.158304	No	19	Yes
199998	0.110972	No	24	Yes
199999	0.121960	No	22	Yes

[200000 rows x 12 columns]

```
[10]: #Scaling Numerical Features
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
data[['Claim Amount', 'Premium/Amount Ratio', 'BMI']] = scaler.
fit_transform(data[['Claim Amount', 'Premium/Amount Ratio', 'BMI']])
```

```
[11]: # Data Splitting
import pandas as pd
from sklearn.model_selection import train_test_split
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(data, data['Churn'],
test_size=0.2, random_state=42)
# Print the shape of the training and testing sets
print("Training set shape:", X_train.shape)
print("Testing set shape:", X_test.shape)
```

Training set shape: (160000, 12)

Testing set shape: (40000, 12)

```
[12]: import pandas as pd
from sklearn.preprocessing import MinMaxScaler
# Load your dataset

data = pd.read_csv('randomdata.csv')
# Identify numerical columns in your dataset
numerical_columns = ['Claim Amount', 'Premium/Amount Ratio', 'BMI']
# Create a MinMaxScaler object
scaler = MinMaxScaler()
# Fit the MinMaxScaler to the numerical columns
scaler.fit(data[numerical_columns])
# Transform the numerical columns using the MinMaxScaler
data[numerical_columns] = scaler.transform(data[numerical_columns])
```

```
# Display the resulting DataFrame
print(data)
```

```

      Unnamed: 0  Customer Name \
0              0  Christine Payne
1              1  Tony Fernandez
2              2  Christopher Kim
3              3  Nicole Allen
4              4  Linda Cruz

```

```

...
199995  199995  Matthew Estrada
199996  199996  James Bean
199997  199997  David Meyer
199998  199998  Martha Stone
199999  199999  Shannon Lewis

```

```

      Customer_Address \
0  7627 Anderson Rest Apt. 265,Lake Heather, DC 3...
1  3953 Cindy Brook Apt. 147,East Lindatown, TN 4...
2  8693 Walters Mountains,South Tony, TX 88407
3  56926 Webster Coves,Shawnmouth, NV 04853
4  489 Thomas Forges Apt. 305,Jesseton, GA 36765
...
199995  2024 Lopez Gateway,Lake Pamelaafort, MS 35772
199996  0268 Lori Falls,West Jeffrey, SC 49142
199997  00573 Miller Cliff,New Allenbury, SC 68902
199998  62681 Peters Cove,South Anthony, RI 99783
199999  Unit 6569 Box 2236,DPO AE 88045

```

```

      Company Name Claim Reason Data confidentiality \
0  Williams, Henderson and Perez  Travel  Low
1  Moore-Goodwin  Medical  High
2  Smith-Holmes  Phone  Medium
3  Harrell-Perez  Phone  Medium
4  Simpson, Kramer and Hughes  Phone  Medium
...
199995  Carlson-Matthews  Medical  High
199996  Trevino-Cardenas  Medical  High
199997  Simon-Evans  Medical  High
199998  Baker, Brooks and Porter  Travel  Low
199999  Roth, Merritt and Grant  Medical  High

```

```

      Claim Amount  Category Premium  Premium/Amount Ratio \
0  0.163621  4794  0.309973
1  0.626197  14390  0.397222
2  0.110966  1875  0.545682
3  0.100957  1875  0.495739

```

4	0.103568	1875	0.508767
...
199995	0.679721	14390	0.432023
199996	0.583551	14390	0.369494
199997	0.990862	14390	0.634321
199998	0.231070	4794	0.441611
199999	0.763272	14390	0.486346

	Claim Request	output	BMI	Churn
0	No	0.3	Yes	
1	No	0.6	Yes	
2	No	0.0	Yes	
3	No	0.6	Yes	
4	No	0.3	Yes	
...	
199995	No	0.0	Yes	
199996	No	0.4	Yes	
199997	No	0.1	Yes	
199998	No	0.6	Yes	
199999	No	0.4	Yes	

[200000 rows x 12 columns]

DATA SPLITTING 80 20

```
[13]: import pandas as pd
from sklearn.model_selection import train_test_split

# Load your dataset

data = pd.read_csv('randomdata.csv')

# Assuming 'Churn' is your target variable and you want to predict it
X = data.drop(columns=['Churn']) # Features
y = data['Churn'] # Target variable

# Split the data into training and testing sets (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
                                                    random_state=42)

# Display the sizes of the resulting sets
print(f"X_train shape: {X_train.shape}")
print(f"X_test shape: {X_test.shape}")
print(f"y_train shape: {y_train.shape}")
print(f"y_test shape: {y_test.shape}")
```

X_train shape: (160000, 11)

X_test shape: (40000, 11)

```
y_train shape: (160000,)
y_test shape: (40000,)
```

DATA SPLITTING 75 25

```
[14]: import pandas as pd
      from sklearn.model_selection import train_test_split

      # Load your dataset

      data = pd.read_csv('randomdata.csv')

      # Assuming 'Churn' is your target variable and you want to predict it
      X = data.drop(columns=['Churn']) # Features
      y = data['Churn'] # Target variable

      # Split the data into training and testing sets (75% training, 25% testing)
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,
      random_state=42)

      # Display the sizes of the resulting sets
      print(f"X_train shape: {X_train.shape}")
      print(f"X_test shape: {X_test.shape}")
      print(f"y_train shape: {y_train.shape}")
      print(f"y_test shape: {y_test.shape}")
```

```
X_train shape: (150000, 11)
X_test shape: (50000, 11)
y_train shape: (150000,)
y_test shape: (50000,)
```

DATA SPLITTING 85 15

```
[15]: import pandas as pd
      from sklearn.model_selection import train_test_split

      # Load your dataset
      # Replace 'randomdata.csv' with the actual path to your dataset
      data = pd.read_csv('randomdata.csv')

      # Assuming 'Churn' is your target variable and you want to predict it
      X = data.drop(columns=['Churn']) # Features
      y = data['Churn'] # Target variable

      # Split the data into training and testing sets (85% training, 15% testing)
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.15,
      random_state=42)

      # Display the sizes of the resulting sets
```

```

print(f"X_train shape: {X_train.shape}")
print(f"X_test shape: {X_test.shape}")
print(f"y_train shape: {y_train.shape}")
print(f"y_test shape: {y_test.shape}")

```

```

X_train shape: (170000, 11)
X_test shape: (30000, 11)
y_train shape: (170000,)
y_test shape: (30000,)

```

4 FEATURE SELECTION - Ensemble method

```

[16]: import pandas as pd
      from sklearn.ensemble import RandomForestClassifier
      from sklearn.model_selection import train_test_split
      from sklearn.preprocessing import LabelEncoder

      data = pd.read_csv('randomdata.csv')

      # Assuming 'Churn' is your target variable and you want to predict it
      # Drop non-numeric columns and target variable
      X = data.drop(columns=['Churn', 'Customer Name', 'Customer_Address'])
      y = data['Churn'] # Target variable

      # Encode categorical variables using LabelEncoder
      label_encoder = LabelEncoder()
      for column in X.columns:
          if X[column].dtype == 'object':
              X[column] = label_encoder.fit_transform(X[column])

      # Split the data into training and testing sets
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
          random_state=42)

      # Initialize a Random Forest Classifier
      rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)

      # Fit the classifier to your data
      rf_classifier.fit(X_train, y_train)

      # Evaluate the classifier on the testing data
      accuracy = rf_classifier.score(X_test, y_test)
      print("Accuracy:", accuracy)

```

```

Accuracy: 1.0

```



```
[17]: import pandas as pd
from sklearn.model_selection import train_test_split
from imblearn.over_sampling import RandomOverSampler
from sklearn.preprocessing import LabelEncoder
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score, classification_report

data = pd.read_csv('randomdata.csv')

# Assuming 'Churn' is your target variable and you want to predict it
# Drop non-numeric columns and target variable
X = data.drop(columns=['Churn', 'Customer Name', 'Customer Address'])
y = data['Churn'] # Target variable

# Encode categorical variables using LabelEncoder
label_encoder = LabelEncoder()
for column in X.columns:
    if X[column].dtype == 'object':
        X[column] = label_encoder.fit_transform(X[column])

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    random_state=42)

# Perform oversampling on the training data
oversampler = RandomOverSampler(sampling_strategy='auto', random_state=42)
X_train_resampled, y_train_resampled = oversampler.fit_resample(X_train,
    y_train)

# Initialize a Deep Learning model (MLPClassifier is used here as an example)
clf = MLPClassifier(hidden_layer_sizes=(100, 50), max_iter=1000,
    random_state=42)

# Fit the classifier to your resampled training data
clf.fit(X_train_resampled, y_train_resampled)

# Make predictions on the test data
y_pred = clf.predict(X_test)

# Evaluate the classifier
accuracy = accuracy_score(y_test, y_pred)
classification_report_result = classification_report(y_test, y_pred)

print("Accuracy:", accuracy)
print("Classification Report:")
print(classification_report_result)
```

```

Accuracy: 0.6986
Classification Report:

```

	precision	recall	f1-score	support
No	0.72	0.28	0.41	14536
Yes	0.70	0.94	0.80	25464
accuracy			0.70	40000
macro avg	0.71	0.61	0.60	40000
weighted avg	0.70	0.70	0.66	40000

5 DEEP LEARNING MODEL - Ensemble method

```

[18]: # Define a function to create a deep learning model
def create_model(input_dim):
    model = Sequential()
    model.add(Dense(64, input_dim=input_dim, activation='relu'))
    model.add(Dense(32, activation='relu'))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(loss='binary_crossentropy', optimizer='adam',
metrics=['accuracy'])
    return model

[19]: # Create three deep learning models
import numpy as np
from tensorflow.keras.models import Sequential # Import the Sequential class
from tensorflow.keras.layers import Dense
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

model1 = create_model(X_train.shape[1])
model2 = create_model(X_train.shape[1])
model3 = create_model(X_train.shape[1])

[20]: from sklearn.preprocessing import LabelEncoder

# Encode the target variable into numerical values (0 and 1)
label_encoder = LabelEncoder()
y_train = label_encoder.fit_transform(y_train)
y_test = label_encoder.transform(y_test)

# Train the deep learning models
model1.fit(X_train, y_train, epochs=10, batch_size=32, verbose=0)

```

```
model2.fit(X_train, y_train, epochs=10, batch_size=32, verbose=0)
model3.fit(X_train, y_train, epochs=10, batch_size=32, verbose=0)
```

[20]: <keras.src.callbacks.History at 0x200bebbd6d0>

[21]: *# Make predictions using the individual models*

```
pred1 = model1.predict(X_test)
pred2 = model2.predict(X_test)
pred3 = model3.predict(X_test)
```

```
1250/1250 [=====] - 1s 621us/step
1250/1250 [=====] - 1s 601us/step
1250/1250 [=====] - 1s 610us/step
```

[22]: *# Ensemble the predictions using a simple averaging method*

```
ensemble_preds = np.round((pred1 + pred2 + pred3) / 3)
```

[23]: *# Calculate accuracy of the ensemble model*

```
ensemble_accuracy = accuracy_score(y_test, ensemble_preds)
```

```
print("Ensemble Model Accuracy:", ensemble_accuracy)
```

Ensemble Model Accuracy: 0.69195

[24]: *from sklearn.metrics import accuracy_score, precision_score, f1_score, recall_score*

Make predictions using the individual models

```
pred1 = model1.predict(X_test)
pred2 = model2.predict(X_test)
pred3 = model3.predict(X_test)
```

Ensemble the predictions using Voting Classifier

```
ensemble_preds = np.round((pred1 + pred2 + pred3) / 3)
```

Calculate classification metrics

```
ensemble_accuracy = accuracy_score(y_test, ensemble_preds)
ensemble_precision = precision_score(y_test, ensemble_preds)
ensemble_f1 = f1_score(y_test, ensemble_preds)
ensemble_recall = recall_score(y_test, ensemble_preds)
```

Print the evaluation metrics

```
print("Ensemble Model Metrics:")
print("Accuracy:", ensemble_accuracy)
print("Precision:", ensemble_precision)
print("F1 Score:", ensemble_f1)
print("Recall:", ensemble_recall)
```



```
1250/1250 [=====] - 1s 617us/step
1250/1250 [=====] - 1s 635us/step
1250/1250 [=====] - 1s 732us/step
```

Ensemble Model Metrics:

Accuracy: 0.69195

Precision: 0.6776137960860633

F1 Score: 0.8027217419148255

Recall: 0.984487904492617

Experimenting with another ensemble method - bagging

```
[25]: from sklearn.ensemble import BaggingClassifier, RandomForestClassifier

# Create a BaggingClassifier with a RandomForest base estimator
bagging_classifier = BaggingClassifier(
    base_estimator=RandomForestClassifier(n_estimators=100, random_state=42),
    n_estimators=10, random_state=42)

# Fit the bagging ensemble to the data
bagging_classifier.fit(X_train, y_train)

# Evaluate the bagging ensemble
bagging_accuracy = bagging_classifier.score(X_test, y_test)

print("Bagging Ensemble Accuracy:", bagging_accuracy)
```

C:\Users\Neha\anaconda3\Lib\site-packages\sklearn\ensemble_base.py:166:

FutureWarning: `base_estimator` was renamed to `estimator` in version 1.2 and will be removed in 1.4.

warnings.warn(

Bagging Ensemble Accuracy: 1.0