Scientific Journal of Informatics
Vol. 12, No. 2, May 2025
p-ISSN 2407-7658 https://journal.unnes.ac.id/journals/sji/index e-ISSN 2460-0040

Evaluation of Ridge Classifier and Logistic Regression for Customer Churn Prediction on Imbalanced Telecommunication Data

Rofik[1*], Jumanto[2]
[1,2]Department of Computer Science, Universitas Negeri Semarang, Indonesia

Abstract.
Purpose: Customer churn is a crucial issue for companies, especially those in the telecommunications sector, as it has a direct impact on revenue and new customer acquisition costs. The purpose of this research is to create a customer churn prediction model through performance comparison between the Logistic Regression algorithm and Ridge Classifier, considering the effect of data balancing.
Methods: This study developed a churn classification model by comparing the Logistic Regression and Ridge Classifier algorithms in three scenarios: without data balancing, balancing using SMOTE, and balancing using GAN. The dataset used was Telco Customer Churn from Kaggle. Model evaluation was performed using a confusion matrix with accuracy, precision, recall, and F1-score metrics, with a primary focus on the accuracy metric.
Result: The results show that data balancing using SMOTE and GAN does not improve model accuracy. The highest accuracy was achieved by the Ridge Classifier without data balancing, at 82.47%, followed by Logistic Regression at 82.25%. However, the recall and F1-score metrics improved when using SMOTE. The highest recall was achieved by Ridge Classifier at 75.34% and Logistic Regression at 75.07% in the SMOTE 50:50 scenario. The highest F1-score was also achieved by Ridge Classifier at 64.76% and Logistic Regression at 64.68% followed by the SMOTE 50:30 scenario. Meanwhile, the precision metric tends to decrease after data balancing.
Novelty: The uniqueness of this study lies in the comparison of the performance of the Ridge Classifier and Logistic Regression in data balancing scenarios using SMOTE and GAN, which has not been widely discussed in previous studies. The main findings show that the highest accuracy is achieved when the Ridge Classifier model uses original data or without applying SMOTE or GAN data balancing. However, data balancing using SMOTE has been proven to significantly improve the recall and F1-score metrics.

Keywords: Customer churn, Telecommunications, Logistic regression, Ridge classifier, Classification.
Received May 2025 / **Revised** June 2025 / **Accepted** June 2025

INTRODUCTION

Customer churn is a condition where customers decide to stop subscribing to a company and choose to use services or subscribe to other companies [1]. The phenomenon of customer churn is a crucial problem in various sectors such as banking [2], e-commerce [3], and telecommunications [4], [5]. For service and subscription-based companies such as telecommunication companies, this phenomenon greatly impacts long-term revenue and increases the cost of acquiring new customers. The increasing number of churns and ineffective methods of dealing with customer churn will only cause huge losses that can disrupt the stability and growth of the company's business.

Previous research shows that annual churn reaches 20-40%, and the cost of acquiring new customers is 5-10 times greater than retaining existing customers [6], [7]. Yet 5% customer retention can increase profits by 25-95% [8]. An article from McKinsey & Company reveals that to replace 1 lost customer, companies need to acquire 3 new customers [9]. These data show that the churn phenomenon not only affects the loss of customers but also disrupts the stability of cash flow and long-term sustainability of a company. Therefore, identifying customer churn effectively and efficiently is key to avoiding sustained losses and designing timely intervention strategies [10].

Machine learning, which is a subset of Artificial Intelligence that allows computers to learn and perform tasks without being explicitly programmed [11], is a potential solution to the customer churn problem [12]. This method can overcome the limitations of conventional methods that require large resources and a long

*Corresponding author.
Email addresses: rofikn4291@students.unnes.ac.id (Rofik)
DOI: 10.15294/sji.v12i2.24620

time. In addition, it is vulnerable to human error. With the ability to process large amounts of data and find complex patterns automatically, machine learning can help companies predict customers who will churn quickly, accurately, and efficiently.

A number of previous studies have developed churn prediction models using machine learning approaches. When considering the type of algorithm used, Shaikhsurab et al. [12] employed XGBoost and achieved an accuracy of 79.8%, while Wang et al. [13] applied FCLCNN-LSTM with Maj-LASSO feature selection and recorded an accuracy of 81.21%. On the other hand, Haddadi et al. [14] combined the Random Forest algorithm with two-phase resampling techniques and achieved the highest accuracy of 82%. Meanwhile, in terms of data balancing approaches, research conducted by Wu et al. [4] compared six classification algorithms and found that Logistic Regression produced the highest accuracy (80.19%) on the original unbalanced data. Interestingly, the accuracy decreased to 74.82% after applying SMOTE. This finding serves as an important basis for this study to further explore the influence of balancing techniques on model performance.

One of the main challenges in building customer churn prediction models is data imbalance [15]. The number of customers who churn is usually less than customers who survive. This condition causes the model built to be biased towards the majority class, which in effect tends to lack the ability to detect customers who really churn [16]. Unbalanced data can be overcome with 2 approaches, namely undersampling, reducing the amount of data from the majority class to balance with the minority class. While oversampling, adding synthetic data from the minority class to balance with the majority class. The undersampling technique is very prone to causing loss of important information from the data. On the other hand, oversampling is often recommended because it preserves all the data [17].

The SMOTE (Synthetic Minority Over-sampling Technique) is one of the popular oversampling methods. It works by creating synthetic data from minority classes based on the similarity of features between samples [18]. GAN (Generative Adversarial Networks) is also a popular oversampling method due to its ability to produce synthetic data that is more varied and able to produce data that is more similar to the original data [19]. Both data balancing methods were chosen because they are compatible with algorithms such as Logistic Regression and Ridge Classifier. Both are capable of generating synthetic data that remains in the original feature space, so as not to disturb the distribution [20], [21] and data structure, which are important factors for linear models such as Logistic Regression and Ridge Classifier. In addition, not all algorithms can overcome the problem of feature multicollinearity or provide good prediction stability. Logistic Regression algorithm is a probability-based algorithm that is considered quite competent with good accuracy in several previous studies [22]. The Ridge Classifier itself is a variation of the Logistic Regression algorithm with the addition of L2 regularization, which has advantages in handling multicollinearity and improving prediction stability. In this case, the Ridge Classifier is an interesting alternative that has not been widely explored in churn prediction studies but provides competitive results. Therefore, this research aims to build a customer churn prediction model by finding the optimal algorithm model through the comparison of Logistic Regression and Ridge Classifier algorithms, both before and after data balancing using SMOTE and GAN.

METHODS

The research flow is carried out sequentially, starting from the data collection stage, then data preprocessing, modeling, to model evaluation, as shown in Figure 1.

1. Data Collection

The dataset used in this research is sourced from open source public data, namely kaggle.com, which can be accessed via the URL: https://www.kaggle.com/datasets/blastchar/telco-customer-churn, the same data used in several previous studies [4], [13], [14]. The dataset consists of 7043 data records, with 21 features containing customer demographic information and the type of service used. Of the total data, 5174 are non-churn customers and 1869 are churn customers. The customer label is written on the Churn feature, where a value of 0 indicates a non-churn customer and 1 indicates a churn customer. Details of the features and their descriptions can be seen in Table 1.

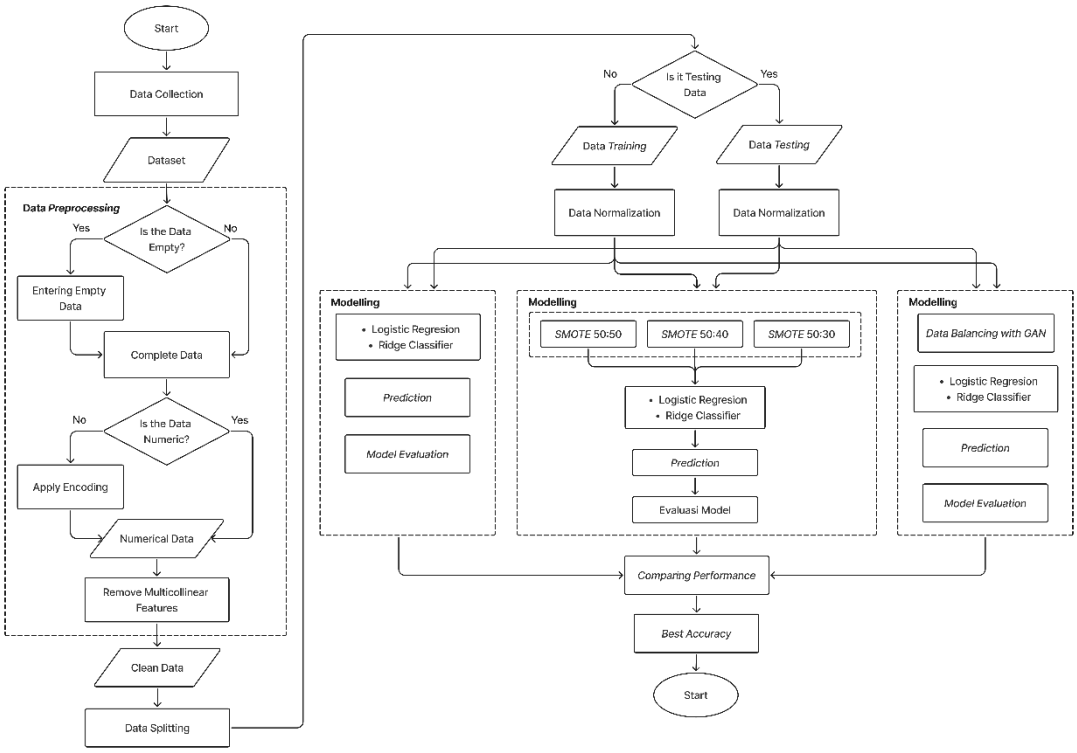


Figure 1. Research workflow for developing a churn prediction model

Table 1. Explanation of each feature in the dataset

Feature Name	Data Type	Description
Customer ID	Categorical	It is a unique identity used to distinguish each customer.
Gender	Categorical	Customer gender (male or female).
Senior Citizen	Categorical	Indicates whether the customer is 65 years old or above (1 means yes and 0 means no).
Partner	Categorical	State whether the customer has a spouse or not (Yes/No).
Dependents	Categorical	States whether the customer has dependents or not (Yes/No).
Tenure	Numeric	Length of time the customer has subscribed to the service, measured in months.
Phone Service	Categorical	Indicates whether the customer uses telephone service (Yes/No).
Multiple Lines	Categorical	Indicates whether the customer has more than one phone line (Yes/No/No Phone Service).
Internet Service	Categorical	Type of internet service used by the customer (DSL/Fiber Optic/No).
Online Security	Categorical	Status of whether the customer has an online security service (Yes/No/No Internet Service).
Online Backup	Categorical	Status of whether the customer has an online backup service (Yes/No/No Internet Service)
Device Protection	Categorical	Indicates whether the customer uses device protection services (Yes/No/No Internet Service)
Tech Support	Categorical	State whether the customer has access to technical support (Yes/No/No Internet Service)
Streaming TV	Categorical	Indicates whether the customer subscribes to a TV streaming service (Yes/No/No Internet Service)
Streaming Movies	Categorical	Indicates whether the customer subscribes to a movie streaming service (Yes/No/No Internet Service)
Contract	Categorical	Type of subscription contract the customer has (Month-to-month/One year/Two-year)
Paperless Billing	Categorical	Indicates whether the customer uses a paperless billing system (Yes/No)
Payment Method	Categorical	Payment method used by the customer (Electronic check/Mailed check/Bank transfer (automatic)/Credit card (automatic))
Monthly Charges	Numeric	Amount of the monthly subscription fee paid by the customer
Total Charges	Numeric	Total accumulated cost incurred by the customer during the subscription.
Churn	Categorical	Status of whether the customer has unsubscribed (Yes if Churn, No if not Churn)

2. Data Preprocessing

Data preprocessing is the initial stage in data analysis and machine learning development to clean, transform, and prepare data for use in algorithms. Data preprocessing is important because data obtained from the real world is often incomplete, inconsistent, or contains noise [23]. Data preprocessing usually includes handling empty data, removing irrelevant features, transforming categorical data, and detecting and removing outliers. Previous research revealed that data quality determines the quality of analysis results, and preprocessing is a very important stage in the machine learning pipeline.

The preprocessing stage carried out in the context of creating a customer churn prediction model in this study is to remove irrelevant features, handle empty data, encode data, and remove feature multicollinearity. The CustomerID feature is a feature that contains a unique value to distinguish Customer 1 from other customers. And in the context of customer churn prediction, this feature does not contribute or provide any information, so the CustomerID feature is removed. The data that previously had 21 features now has 20 features.

Missing value checking is done because if the missing data is left, it will reduce the quality of analysis and model accuracy [24]. It was found that there were 11 blank rows in the Total Charges feature. After further analysis, the missing data turns out to occur when the tenure is 0, which indicates that the customer still does not have total charges because the subscription is not even 1 month old. The empty data in total charges is inputted with the same value that is in the monthly charges. With this step, this research assumes that customers will still pay their bills in that month, and they still have total charges. This stage creates a complete dataset for each row and feature.

Next, categorical data is transformed into numerical form, which is called data encoding. Data encoding in this research uses 2 techniques, namely label encoding and one-hot encoding. Label encoding is applied to features that only have 2 unique values and that have a sequence or hierarchy. While one-hot encoding is applied to features that have no hierarchy or order. Label encoding is applied to Churn, Gender, Partner, Dependents, PhoneService, PaperlessBilling, and Contract features. One-hot encoding is applied to MultipleLines, InternetService, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, StreamingMovies, and PaymentMethod features. This method was chosen so that each category is treated equally without assuming any numerical relationship between categories.

The effect of data encoding is that the relationship between features can be very high, even if the information is the same. Features that have a very high correlation with other features are called feature multicollinearity. Feature multicollinearity can cause the model to become less stable and reduce the interpretability of the results if not handled properly [25]. This happens when two or more features in the dataset have a very high correlation, making the information redundant and confusing the model in determining the right weight. Identify feature pairs that have a high correlation (generally above 0.9). From each pair of features that have a very strong correlation, one feature from the pair will be removed. The selection of features to be removed is based on their relevance to the target, measured by the correlation between the feature and the target (churn). Thus, the retained features are those that provide stronger predictive contributions. Feature removal is not solely based on weak correlation with the target, but only in cases where the information is redundant within highly correlated pairs.

3. Data Splitting

Data splitting is done to divide the data into 2 parts, namely training data, for training the algorithm model, and testing data to evaluate the performance of the algorithm model. This process is done by utilizing the train_test_split function in the scikit-learn library with a proportion of 80:20, where 80% is used as training data and 20% as testing data. The division is done randomly, with a random_state control of 42 (a default number that has no special meaning) to ensure that the data division is done deterministically or consistently every time the code is run.

4. Data Balancing using SMOTE

Data balancing is done to overcome the problem of data imbalance, considering that the dataset used is not balanced between churn and non-churn customers. To avoid the loss of important information from the data, this study chose to use the oversampling method, which balances the data by creating new synthetic

data from the minority class. This research tested 2 data oversampling techniques, namely SMOTE and GAN.

SMOTE (Synthetic Minority Over-sampling Technique) is a data balancing technique by creates new samples synthetically by interpolating between similar samples. The method starts by finding k nearest neighbors (usually k = 5) for each sample from the minority class [26]. In this study, the default value of k = 5 was used, as is commonly applied in SMOTE implementations, assuming that this number is sufficient to form a representative local distribution without causing excessive noise. One of these neighbors is chosen at random. A new point is then created between the original point and its neighbor according to Equation (1).

$$x_{synthetic} = x_i + \delta . (x_j - x_i) \quad (1)$$

Where x_i is the original data of the minority class, x_j is the nearest neighbor of x_i, and δ is a random number between 0 and 1 (to determine the interpolation position). The random number is drawn from a uniform distribution. Randomization is done using a random generator or Pseudorandom Number Generator (PRNG) function, such as np.random.uniform (0,1) in Python. Since δ is random, the position of the synthetic point will be between x_i and x_j, but not exactly in the center, resulting in more varied and realistic synthetic data.

5. Data Balancing using GAN

Unlike SMOTE, GAN (Generative Adversarial Networks) generates synthetic data by using two competing neural networks, the Generator (G) and the Discriminator (D) [27]. The Generator is in charge of creating synthetic data that resembles the original data, while the Discriminator is in charge of distinguishing whether the data comes from the original data or the Generator's creation. The training process takes place iteratively in the form of a competition: The Generator tries to "fool" the Discriminator by generating increasingly realistic data, while the Discriminator keeps trying to distinguish with high accuracy. The method's process starts with a random noise input ($z \sim p_z(z)$), which is fed to the Generator. The Generator is a neural network model that is responsible for generating new synthetic data that resembles real data. The output of the generator is synthetic data from the minority class, which aims to balance the data distribution. The synthetic data is then combined with real data from the training dataset, which is then given to the Discriminator. The Discriminator is the second neural network model in charge of distinguishing whether the data comes from the real data or the Generator's creation. The Discriminator will give an output value in the form of a probability between "real" and "artificial" data.

During the training process, the Generator continues to improve its artificial results so that they resemble the original data and can "fool" the Discriminator. Likewise, the Discriminator continues to learn so that it can be better at distinguishing between real data and artificial data. This process continues until both reach an equilibrium point, which is when the Generator successfully produces synthetic data that cannot be significantly distinguished from the original data by the Discriminator.

In this study, the Generative Adversarial Network (GAN) architecture was utilized to generate synthetic data to balance class distribution. The data synthesis process was carried out by constructing a generator architecture consisting of three fully connected layers: the first layer with 128 neurons and ReLU activation, followed by the second layer with 256 neurons and ReLU activation, and the output layer with the number of neurons adjusted to the number of features and tanh activation function. Random noise input with the same dimension as the features ($z \sim N(0,1)$) is used to generate new data representations that resemble the characteristics of the original data. After being formed, this synthetic data is returned to its original scale using the inverse transform of MinMaxScaler, then combined with the initial training data to create a more balanced class distribution in the model training process.

6. Modeling

6.1 Logistic Regression

Logistic regression is a statistically based classification algorithm used to predict events between two possibilities, namely "yes" or "no". It predicts the probability of an event by restricting the output to a range between 0 and 1. The algorithm works by assigning small random initial values to all weights (w) and bias values (b). These weights act as a regulator of the level of influence each input feature (x) has on the output

of the model. After the initialization process is complete, the model starts making initial predictions using the sigmoid function. The value entered in the sigmoid function is the value of the linear combination (z) as in Equation (2).

$$z = w^T x + b \tag{2}$$

w^T is the transpose of the 1 × n-dimensional weight vector, x is the n × 1-dimensional input vector, b is the bias value (scalar), and z is the result of a linear combination between input features and weights before being converted into probabilities. The result of the z value is inputted into the sigmoid function (a function to convert the value into a probability between 0 and 1). The sigmoid function can be calculated as in Equation (3).

$$P(y = 1|x) = \sigma(z) = \frac{1}{1 + e^{-z}} \tag{3}$$

To measure how well the model predicts the target class, the log loss function or binary cross-entropy [28] is used to calculate the difference between the predicted probability and the actual label. The log loss function can be calculated as in Equation (4).

$$Log\ Loss = -[y . \log(p) + (1 - y). \log(1 - p)] \tag{4}$$

Where y ∈ {0,1} is the actual label, and p = σ (z) is the predicted probability for class 1. The greater the difference between the predicted and actual labels, the higher the error value. The weights and biases are updated incrementally using gradient descent to make the model more accurate [29]. This technique works by calculating the partial derivatives of the log loss function against the weight and bias parameters. Once the direction of the gradient is found, the parameters are shifted to the negative direction of the gradient. The weight and bias update formulas correspond to Equations (5) and (6).

$$\text{w} := \text{w} - \alpha . \frac{\partial J(w,b)}{\partial w} \tag{5}$$

$$\text{b} := \text{b} - \alpha . \frac{\partial J(w,b)}{\partial b} \tag{6}$$

Where α is the learning rate or the size of the update step in each iteration, and J (w,b) is the cost (log loss). The partial derivative value shows how fast the error value changes with each parameter. The optimization process continues until the cost function value reaches a minimum or approaches zero, or when the specified maximum number of iterations has been reached. After training is complete, the model will use the final weights to calculate the probabilities of new data. The probability is usually set with a threshold of 0.5. So, if the value of the probability is more than 0.5, then the data will be classified into class 1. If the probability value is less than 0.5, the data will be classified as class 0 [30].

6.2 Ridge Classifier

Ridge classifier is a classification algorithm based on linear regression, but equipped with the L2 regularization technique (ridge regularization) to avoid overfitting [31]. The algorithm starts by initializing the weights w and bias b randomly. Next, the value of the regularization parameter α is determined, which serves to control how much penalty is given to the weights. The ridge penalty is used to make the model less complicated and avoid overfitting. The prediction process is done with a linear formula as in Equation (7).

$$z = w^T x + b \tag{7}$$

w^T is the transpose of the 1 × n-dimensional weight vector, x is the n × 1-dimensional input vector, b is the bias value (scalar), and z is the linear prediction result before the class is determined. Prediction error is measured using the Ridge Regression loss function, which is a combination of the Mean Squared Error (MSE) with the L2 penalty. The function can be calculated as in Equation (8).

$$L(w,b) = \frac{1}{n}\sum_{i=1}^{n}(y_i - (w^T x_i + b))^2 + \alpha \| w \|^2 \quad (8)$$

y_i is the true label of the i-th data, x_i is the input feature for the i-th data, and $\alpha \| w \|^2$ is the L2 regularization penalty, which penalizes large weights. The loss function is optimized through a gradient descent algorithm to adjust the weights and bias so that the error (loss) becomes as small as possible. The update process continues until the weights become very small (convergent) or until the maximum number of iterations is reached. Once the training process is complete and the final weights are obtained, the final predicted value z is used to determine the output class. Since the Ridge Classifier is a linear model, the classification decision is made by comparing the z value to zero. If $z > 0$, then the data is classified as class 1. If $z < 0$, then it is classified as class 0.

7. Model Evaluation

Model evaluation is done using the confusion matrix, which is a table that describes the performance of the classification prediction model by comparing the predicted results to the actual labels. Table 2 is a confusion matrix.

Table 2. Confusion matrix

		Actual Values	
		1	0
Predictive	1	TP	FP
Values	0	FN	TN

From Table 2, it can be seen that the confusion matrix consists of 4 main components. TP (True Positive) is when the data is labeled positive and predicted positive by the model. Conversely, TN (True Negative) is when the data is labeled negative and predicted negative by the model. FP (False Positive) is when data that is negative but classified as positive by the model. Conversely, FN (False Negative) is when data that is positive but classified by the model as negative. From these four components, evaluation metrics such as accuracy, precision, recall, and F1-score can be calculated.

1. Accuracy

Accuracy is a metric that measures the proportion of the number of correct predictions (both positive and negative) compared to the total number of predictions made by the model. The formula for the accuracy metric is shown in Equation (9)

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \; x \; 100\% \quad (9)$$

2. Precision

Precision is a metric that measures how accurate the model is in making positive predictions, i.e., the number of all positive predictions that match the original label. The formula for the precision metric is shown in Equation (10)

$$Precision = \frac{TP}{TP + FP} \quad (10)$$

3. Recall

Recall is an evaluation metric that shows how much of the true positive data was correctly identified by the model. The formula for the recall metric is shown in Equation (11)

$$Recall = \frac{TP}{TP + FN} \quad (11)$$

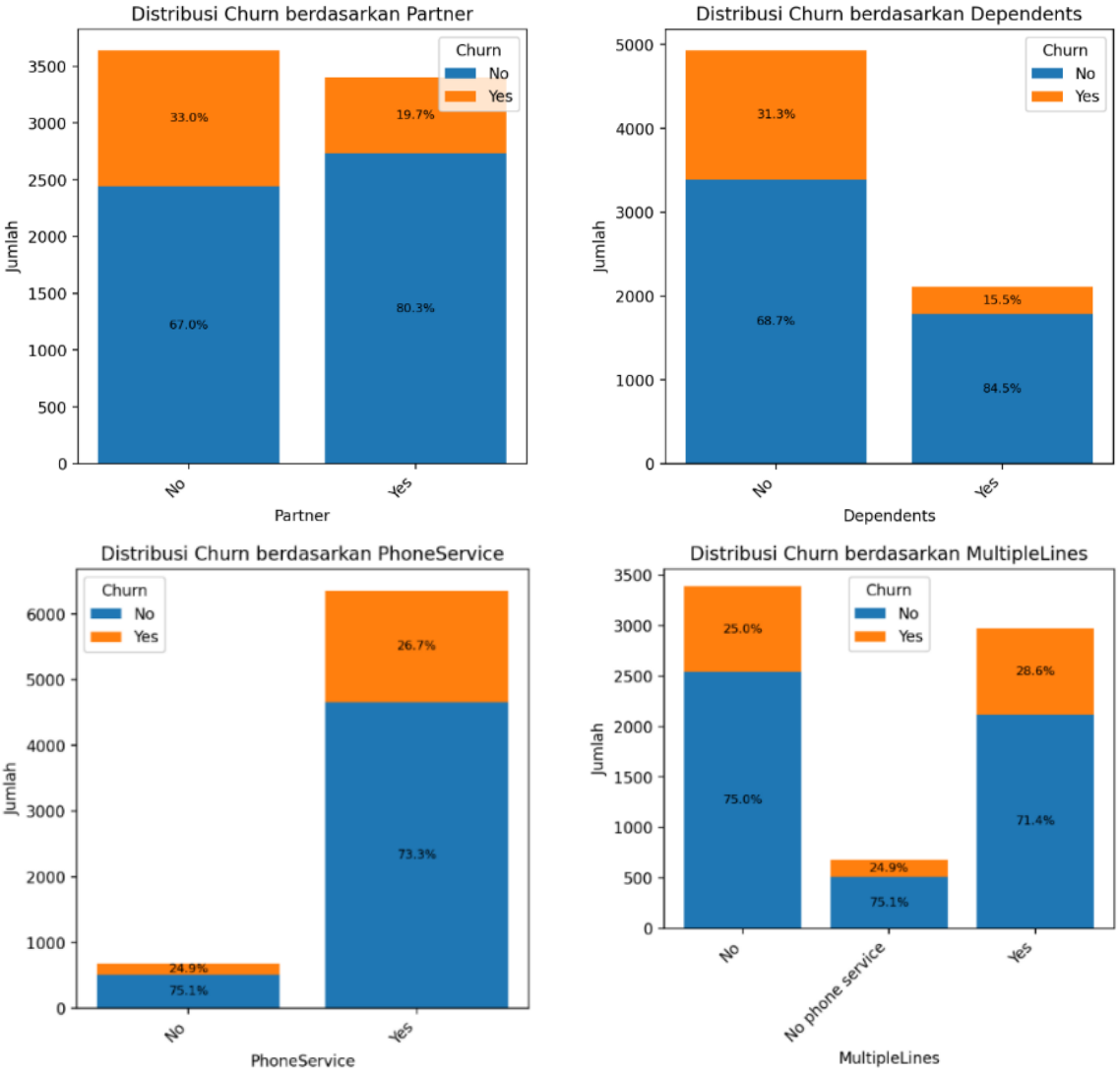
4. F1-Score

F1-score is a metric used to assess the balance between precision and recall values, thus providing an overall picture of the model's ability to classify data correctly. The formula for the accuracy F1-score is shown in Equation (12)

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (12)$$

RESULTS AND DISCUSSIONS

This research uses the Telco Churn Prediction dataset, which includes important customer information, including customer demographic information and services used. The features contained in the dataset provide important insights into analyzing the characteristics of customers who tend to churn and those who stay. This research conducted exploratory data analysis (EDA), the results of which can be seen in Figure 2.



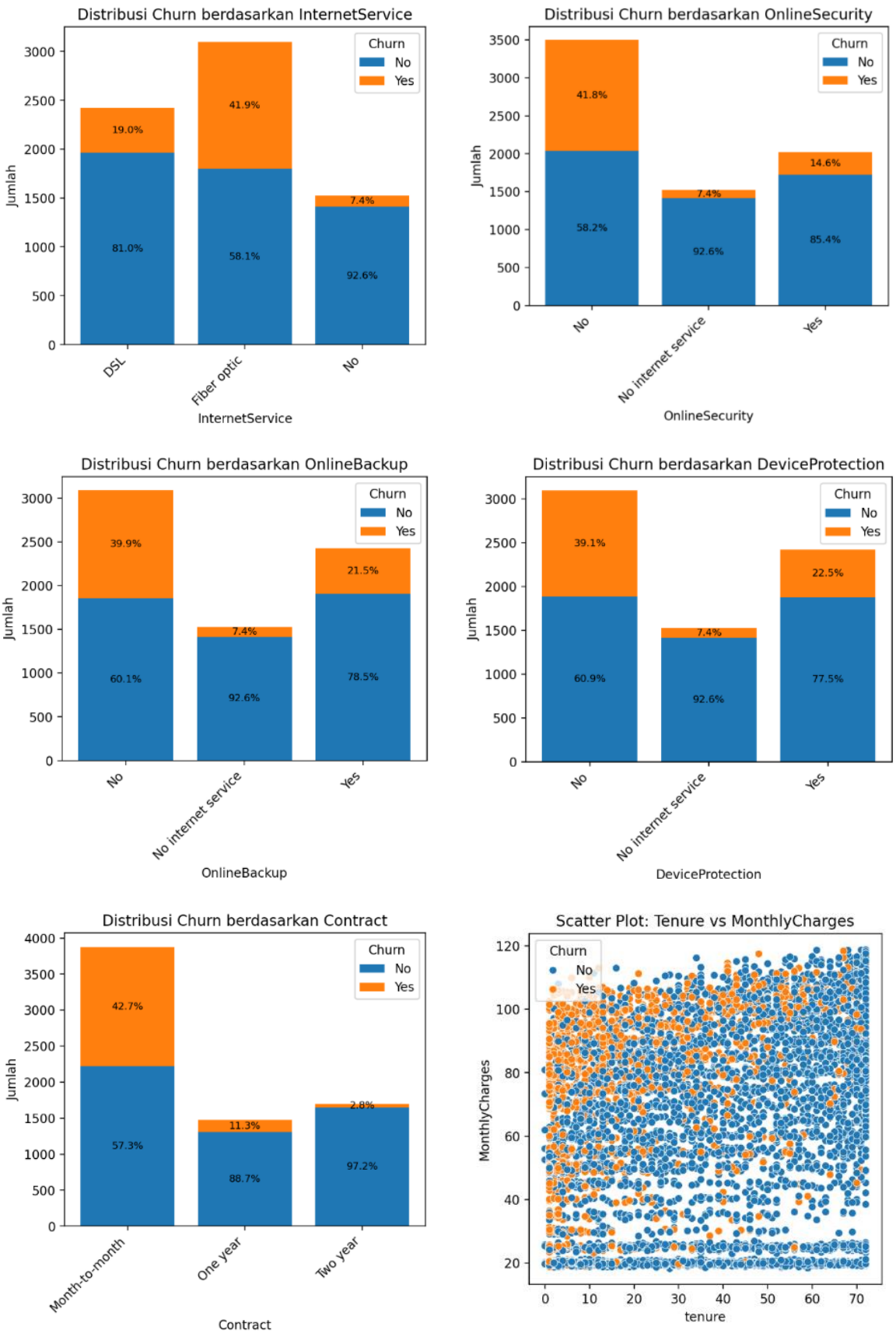


Figure 2. Exploratory data analysis (EDA) results

Figure 2, which consists of 10 charts of different features, shows important insights into the characteristics of customers who are likely to churn. It can be seen in Figure 2 that customers without spouses and dependents tend to have higher churn rates. The type of internet service also seems to affect the churn behavior of customers, with Fiber Optic users showing a higher churn rate than DSL users or customers without internet service. This suggests that the possible causes are customers' expectations of high service quality or the more expensive subscription fee.

In addition, customers who did not subscribe to additional services such as Online Security, Online Backup, Device Protection, and Tech Support were more likely to churn, indicating the importance of these support features in maintaining customer loyalty. It was also found that customers with high Monthly Charges and low Tenure are more at risk or likely to churn. Customers with Month-to-month contracts are more likely to churn than other contracts, such as one-year or Two-year, which indicates that the flexibility of short-term contracts is more likely to make customers switch services more easily.

Next, data preprocessing is performed. Deleting the customerID feature and inputting missing values in the total charges feature with the same value as that in monthly charges, creating a complete dataset of 20 features and 7043 rows. Categorical to numerical data conversion is done by implementing label encoding and one-hot encoding techniques. Label encoding is applied to Churn, Gender, Partner, Dependents, PhoneService, PaperlessBilling, and Contract features. One-hot encoding is applied to the MultipleLines, InternetService, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, StreamingMovies, and PaymentMethod features. Data mapping using Label encoding and One-hot encoding techniques can be seen in Tables 3, 4, and 5.

Table 3. Data mapping using label encoding

Feature	Value
Churn	No:0, Yes:1
Gender	Female:0, Male:1
Partner	No:0, Yes:1
Dependents	No:0, Yes:1
Phone Service	No:0, Yes:1
Paperless Billing	No:0, Yes:1
Contract	Month-to-month: 0, One year: 1, Two year: 2

Table 4. Example data before one-hot encoding implementation

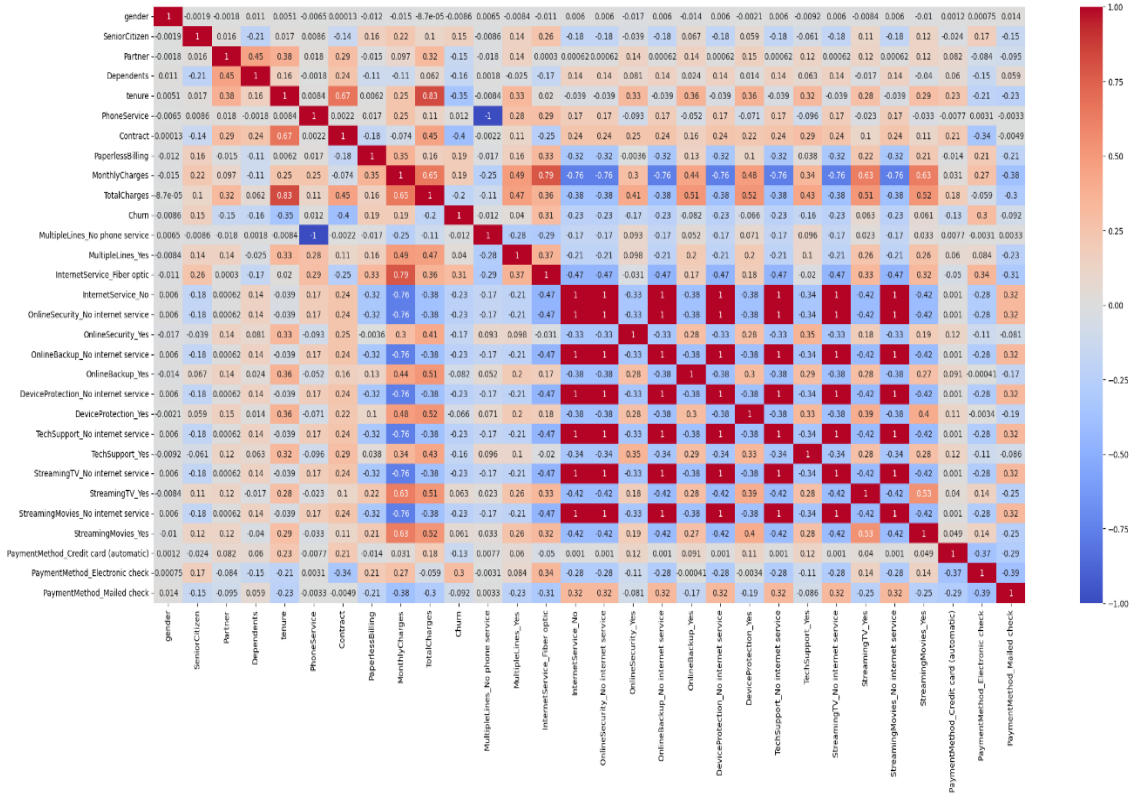
MultipleLines	***InternetService***	***OnlineSecurity***
No phone service	DSL	No
No	No	Yes
Yes	No	No internet service
No phone service	DSL	No internet service
No	Fiber optic	No

Table 5. Example data after the implementation of one-hot encoding

Multiple Lines_Yes	*Multiple Lines_No*	*Multiple Lines_No phone service*	*Internet Service_DSL*	*Internet Service_Fiber optic*	*Internet Service_No*	*Online Security_Yes*	*Online Security_No*	*Online Security_No internet*
0	0	1	1	0	0	0	1	0
0	1	0	0	0	1	1	0	0
1	0	0	0	0	1	0	0	1
0	0	1	1	0	0	0	0	1
0	1	0	0	1	0	0	1	0

Label encoding and One-hot encoding help convert previously categorical or object data into numerical values that can be used for modeling. However, the effect of this encoding will result in a very high

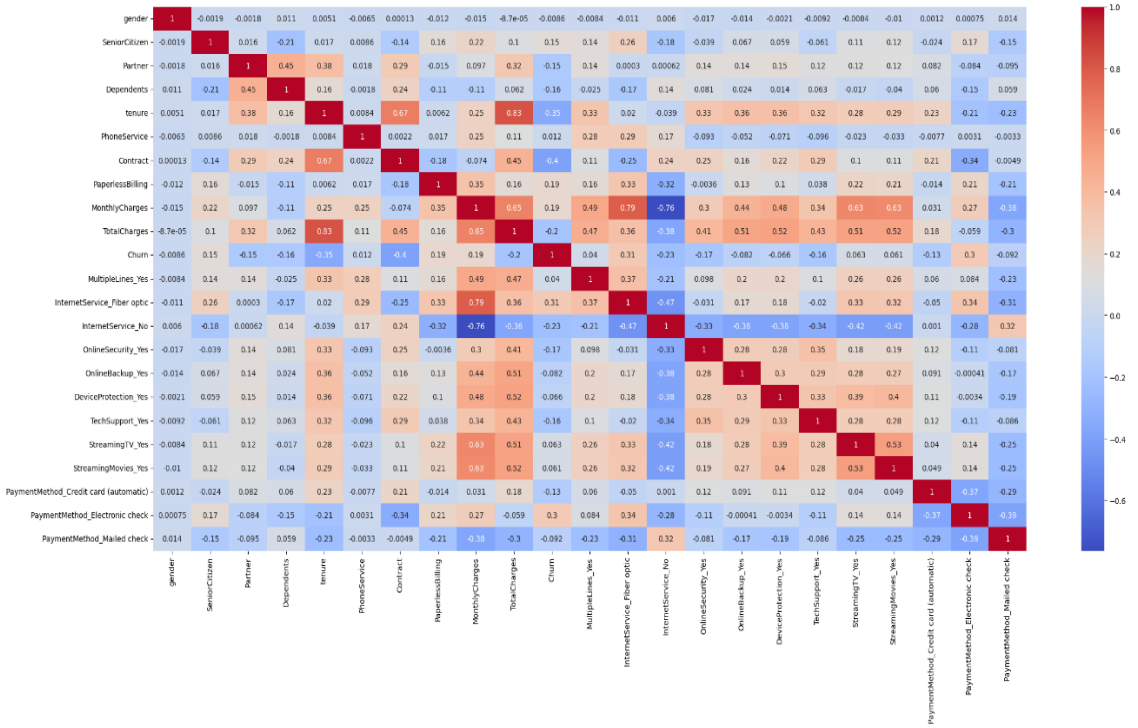
relationship between features (multicollinearity of features). A correlation heatmap was created using the Pearson technique to see the relationship between features in the dataset.



Gambar 3. Heatmap correlation

The correlation heatmap displays the strength of relationships between features in the dataset. The correlation value between features ranges from -1 to 1, where 1 indicates a perfect positive correlation (both features move in the same direction), while -1 indicates a perfect negative correlation (one feature rises when the other falls). The closer the value is to zero, the weaker the relationship between the features. The colors on the heatmap reinforce this understanding visually. Dark red indicates a strong positive correlation, while dark blue indicates a strong negative correlation. If the colors fade or are neutral, it indicates that the relationship between features is weak or insignificant.

It can be seen in Figure 3 that there are features that have a strong relationship with other features, even with a correlation number of 1. This indicates that the information between the feature pairs is identical or very similar. To avoid multicollinearity that may affect the stability and interpretability of the model, the features MultipleLines_No phone service, OnlineSecurity_No internet service, OnlineBackup_No internet service, DeviceProtection_No internet service, TechSupport_No internet service, StreamingTV_No internet service, StreamingMovies_No internet service were removed from the dataset.



Gambar 4. Correlation heatmap after feature multicollinearity removal

Figure 4 shows the condition of the dataset after the removal of highly correlated features (correlation value 1). It can be seen that each remaining feature in the dataset no longer stores the same information, and there is no redundancy of information. And after dividing the data into training data and testing data. Then the training and evaluation of the model can be carried out. Experiments were conducted to train the Ridge Classifier and Logistic Regression algorithm models to find the most optimal algorithm for predicting customer churn. The first experiment, which uses the original data, can be seen in Table 6.

Table 6. Model performance using original data

Model	Accuracy Test	Accuracy Train	Precision	Recall	F1-Score
Ridge Classifier	82.47%	79.93%	71.58%	56.03%	62.86%
Logistic Regression	82.26%	80.39%	68.92%	60.05%	64.18%

Table 6 shows that Ridge Classifier and Logistic Regression are capable of providing good performance in predicting customer churn with test accuracies of 82.47% and 82.26%, respectively. The difference in accuracy between training and testing is relatively small, indicating that there is no significant indication of overfitting. Ridge Classifier outperforms Logistic Regression in terms of accuracy, although its recall value is lower. This advantage of Ridge Classifier can be attributed to the use of L2 regularization, which is the main characteristic of the algorithm. The Ridge Classifier works by adding a penalty to large coefficient weights, making the model more stable when dealing with features that have high correlation. The Ridge Classifier can control model complexity and avoid overfitting compared to the standard Logistic Regression, which does not apply a penalty to large weights.

To improve the performance of the prediction model, this study tested the implementation of the data balancing method. This research wants to see whether, with balanced data, the model will be more reliable in making predictions or vice versa. Data balancing is done using SMOTE and GAN techniques. Data balancing with SMOTE was carried out with 3 scenarios, namely 50:50, 50:40, and 50:30. This approach was carried out with the aim of identifying the proportion of data that could produce the most optimal model performance. The 50:50 scenario is a condition where the model learns from very balanced data, which is 50% non-churn customers and 50% churn customers. Even though it is balanced, this approach causes the model to learn a fairly large proportion of the synthetic data resulting from oversampling with SMOTE. Likewise, the researcher wanted to test the model's performance when using data that was not too balanced, but also not too far apart in the number of churn and non-churn customers with scenarios of 50:40 and

50:30. This approach makes the model less likely to learn from synthetic data, and is expected to reduce the model's dependence on synthetic data, while maintaining generalizability. The results of modeling the Ridge Classifier and Logistic Regression algorithms using data that has been balanced using SMOTE, both with the 50:50, 50:40, and 50:30 scenarios, can be seen in Table 7.

Table 7. Model performance using SMOTE result data

SMOTE 50:50					
Model	Accuracy Test	Accuracy Train	Precision	Recall	F1-Score
Ridge Classifier	77.22%	80.72%	55.10%	75.34%	63.65%
Logistic Regression	76.72%	80.65%	54.37%	75.07%	63.06%
SMOTE 50:40					
Model	Accuracy Test	Accuracy Train	Precision	Recall	F1-Score
Ridge Classifier	78.99%	80.10%	58.28%	72.65%	64.68%
Logistic Regression	78.28%	80.22%	57.08%	72.39%	63.83%
Scenario 50:30					
Model	Accuracy Test	Accuracy Train	Precision	Recall	F1-Score
Ridge Classifier	80.77%	80.29%	62.88%	66.76%	64.76%
Logistic Regression	79.84%	79.14%	60.32%	69.71%	64.68%

Table 7 shows that the performance of both algorithms changes significantly according to the proportion of data balancing used. In general, the test accuracy increases as the proportion of synthetic data decreases, with the greatest accuracy achieved by the Ridge Classifier algorithm in the 50:30 scenario at 80.77%. However, the accuracy is still lower when compared to modelling using real data or data without data balancing using SMOTE, which is 82.47% for the Ridge Classifier and 82.26% for Logistic Regression.

When viewed from the recall metric, models trained on more balanced data (such as the 50:50 scenario) show better performance in detecting customers who will churn. For example, Ridge Classifier can produce a recall of 75.34% with a balancing proportion of 50:50, followed by Logistic Regression, which achieves a recall of 75.07%. This shows that the model with a balanced data proportion has a higher sensitivity to the minority class (churn customers), although the accuracy obtained is slightly lower. In contrast, the scenario with a 50:30 proportion makes the model produce higher accuracy, but the recall tends to decrease. This shows that the model is less able to recognize all potentially churn customers. Thus, the selection of data balancing scenarios needs to be adjusted to the business objectives, namely optimizing accuracy, recall, or other metrics. If the main focus is to minimize customer loss, then high recall is preferred, and a more balanced data scenario, such as 50:50, could be a good choice.

The experiments continued by implementing a more advanced data balancing method, the Generative Adversarial Network (GAN), which is a technique capable of generating more realistic synthetic data than traditional oversampling techniques such as SMOTE. The performance results of the models trained using the oversampled data with GAN can be seen in Table 8.

Table 8. Model performance using GAN result data

Model	**Accuracy Test**	**Accuracy Train**	**Precision**	**Recall**	**F1-Score**
Ridge Classifier	76.08%	84.27%	53.96%	67.38%	59.93%
Logistic Regression	75.66%	83.52%	53.11%	70.86%	60.71%

Table 8 shows that the performance of both models degrades when using data from data balancing with GAN, which is lower than modeling using SMOTE or no data balancing. The Ridge Classifier produced an accuracy of 76.08%, and Logistic Regression produced 75.66%. Recall of Ridge Classifier increased when using GAN, compared to no data balancing implementation. However, the resulting Logistic Regression recall also decreased. Modeling using the GAN data also unfortunately showed an indication of overfitting, where the difference between training and testing accuracy reached approximately 8%. This suggests that the synthetic data resulting from GAN may not successfully represent the original data distribution, so that the model becomes a little too familiar with the training data, but fails to generalize when faced with new data.

Although GAN has the potential to generate better synthetic data that is more similar to the original data, the results show that this technique still fails to produce optimal performance in the context of customer churn prediction. The results obtained from GAN are still lower than models that use SMOTE oversampling or models without data balancing. Possible reasons why the synthetic data generated by GAN is not better

include the fact that, although its implementation includes adversarial training of the Generator and Discriminator, the training process is simple, the number of epochs is limited, and no stabilization techniques such as batch normalization, dropout, label smoothing, or alternative loss functions such as Wasserstein loss are applied. There is a possibility that the Discriminator becomes too dominant from the start, as its architecture tends to be simple. As a result, the Generator struggles to generate synthetic data that truly approximates the original distribution. This is evident from the results obtained, where although recall improves, accuracy and F1-score decrease significantly, and there is a large discrepancy between training and testing accuracy (overfitting). This indicates that the synthetic data generated not only deviates from the distribution of real data but is also less effective as a representation of minority-class data.

CONCLUSION

This study demonstrates that, in the context of customer churn prediction in the telecommunications sector, simple models such as the Ridge Classifier can achieve the best performance, with an accuracy of 82.47% when applied to the original data without data balancing. This finding underscores that the use of data balancing techniques does not always provide benefits in terms of accuracy; in some cases, it can even reduce it. The main contribution of this study lies in proving that the Ridge Classifier model, which is rarely used in churn prediction research, can work optimally on unbalanced data, thereby challenging the common assumption that data balancing is a prerequisite for improving classification model performance. On the other hand, the SMOTE technique in this case is still able to improve the recall and F1-score metrics, particularly in certain distribution configurations. These findings have important implications for industry and the research community, particularly in terms of the efficiency of predictive model development. Amid the tendency to apply complex data preprocessing techniques by default, this study emphasizes that simpler solutions can deliver optimal performance. The research results also indicate that data balancing strategies should not be treated as standard procedures but must be empirically tested according to the characteristics of the dataset and the objectives of the analysis. For further development, it is recommended to explore a wider range of algorithms, including ensemble approaches, and apply advanced techniques such as feature engineering, hyperparameter tuning, and cross-dataset validation. This is important to test the consistency of the findings and expand their application in a broader industrial context.

REFERENCES

[1] E. Zdravevski, P. Lameski, C. Apanowicz, and D. Ślęzak, "From Big Data to business analytics: The case study of churn prediction," *Appl. Soft Comput. J.*, vol. 90, 2020, doi: 10.1016/j.asoc.2020.106164.

[2] R. Rajamohamed and J. Manokaran, "Improved credit card churn prediction based on rough clustering and supervised learning techniques," *Cluster Comput.*, vol. 21, no. 1, pp. 65–77, 2018, doi: 10.1007/s10586-017-0933-1.

[3] S. Baghla and G. Gupta, "Performance Evaluation of Various Classification Techniques for Customer Churn Prediction in E-commerce," *Microprocess. Microsyst.*, vol. 94, no. February, p. 104680, 2022, doi: 10.1016/j.micpro.2022.104680.

[4] S. Wu, W. C. Yau, T. S. Ong, and S. C. Chong, "Integrated Churn Prediction and Customer Segmentation Framework for Telco Business," *IEEE Access*, vol. 9, pp. 62118–62136, 2021, doi: 10.1109/ACCESS.2021.3073776.

[5] Rofik, J. Unjung, and B. Prasetiyo, "Enhancing costumer churn prediction with stacking ensemble and stratified k-fold," *Bull. Electr. Eng. Informatics*, vol. 14, no. 1, pp. 398–408, 2025, doi: 10.11591/eei.v14i1.8112.

[6] A. Amin *et al.*, "Cross-company customer churn prediction in telecommunication: A comparison of data transformation methods," *Int. J. Inf. Manage.*, vol. 46, no. June 2018, pp. 304–319, 2019, doi: 10.1016/j.ijinfomgt.2018.08.015.

[7] Y. Li, B. Hou, Y. Wu, D. Zhao, A. Xie, and P. Zou, "Giant fight: Customer churn prediction in traditional broadcast industry," *J. Bus. Res.*, vol. 131, no. February, pp. 630–639, 2021, doi: 10.1016/j.jbusres.2021.01.022.

[8] A. Gallo, "The Value of Keeping the Right Customers," *Bain & Company*, 2014. https://hbr.org/2014/10/the-value-of-keeping-the-right-customers?utm_source=chatgpt.com

[9] H. Fanderl, "Focusing on existing customers to unlock growth," *McKinsey & Company*, 2023. https://www.mckinsey.com/~/media/mckinsey/email/rethink/2023/08/2023-08-16c.html?utm_source=chatgpt.com

[10] A. Hassan and A. H. Mhmood, "Optimizing Network Performance, Automation, and Intelligent

Decision-Making through Real-Time Big Data Analytics," pp. 12–22.
[11] A. De Mauro, A. Sestino, and A. Bacconi, "Machine learning and artificial intelligence use in marketing: a general taxonomy," *Ital. J. Mark.*, vol. 2022, no. 4, pp. 439–457, 2022, doi: 10.1007/s43039-022-00057-w.
[12] M. A. Shaikhsurab and P. Magadum, "Enhancing Customer Churn Prediction in Telecommunications: An Adaptive Ensemble Learning Approach," 2017.
[13] C. Wang, C. Rao, F. Hu, X. Xiao, and M. Goh, "Risk assessment of customer churn in telco using FCLCNN-LSTM model," *Expert Syst. Appl.*, vol. 248, no. January, p. 123352, 2024, doi: 10.1016/j.eswa.2024.123352.
[14] S. J. Haddadi, A. Farshidvard, F. dos S. Silva, J. C. dos Reis, and M. da Silva Reis, "Customer churn prediction in imbalanced datasets with resampling methods: A comparative study," *Expert Syst. Appl.*, vol. 246, no. September 2023, p. 123086, 2024, doi: 10.1016/j.eswa.2023.123086.
[15] J. Brown, "Building Robust Customer Churn Prediction Models with Imbalanced Data: An AI/ML Expert's Perspective," *33rdsquare.com*, 2024. https://33rdsquare.com/tech/ai/building-customer-churn-prediction-model-with-imbalance-dataset/
[16] A. Amin *et al.*, "Comparing Oversampling Techniques to Handle the Class Imbalance Problem: A Customer Churn Prediction Case Study," *IEEE Access*, vol. 4, no. Ml, pp. 7940–7957, 2016, doi: 10.1109/ACCESS.2016.2619719.
[17] M. Mujahid *et al.*, "Data oversampling and imbalanced datasets: an investigation of performance for machine learning and feature engineering," *J. Big Data*, vol. 11, no. 1, 2024, doi: 10.1186/s40537-024-00943-4.
[18] H. Guan, L. Zhao, X. Dong, and C. Chen, "Extended natural neighborhood for SMOTE and its variants in imbalanced classification," *Eng. Appl. Artif. Intell.*, vol. 124, no. March, p. 106570, 2023, doi: 10.1016/j.engappai.2023.106570.
[19] J. Engelmann and S. Lessmann, "Conditional Wasserstein GAN-based oversampling of tabular data for imbalanced learning," *Expert Syst. Appl.*, vol. 174, no. September 2020, p. 114582, 2021, doi: 10.1016/j.eswa.2021.114582.
[20] G. S. Sayago, "Balancing Strategies in Machine Learning: Comparing SMOTE, Undersampling, and Class Weights in a Real-World Problem," *Medium*, 2025. https://medium.com/%40surribasg/balancing-strategies-in-machine-learning-comparing-smote-undersampling-and-class-weights-in-a-31d37106953a
[21] M. E. Sánchez-Gutiérrez and P. P. González-Pérez, "Addressing the class imbalance in tabular datasets from a generative adversarial network approach in supervised machine learning," *J. Algorithms Comput. Technol.*, vol. 17, 2023, doi: 10.1177/17483026231215186.
[22] R. Reddy and U. M. A. Kumar, "Classification of user's review using modified logistic regression technique," *Int. J. Syst. Assur. Eng. Manag.*, 2022, doi: 10.1007/s13198-022-01711-4.
[23] S. B. Kotsiantis and D. Kanellopoulos, "Data preprocessing for supervised leaning," *Int. J. …*, vol. 1, no. 2, pp. 1–7, 2006, doi: 10.1080/02331931003692557.
[24] N. G. Ramadhan, "Comparative Analysis of ADASYN-SVM and SMOTE-SVM Methods on the Detection of Type 2 Diabetes Mellitus," *Sci. J. Informatics*, vol. 8, no. 2, pp. 276–282, 2021, doi: 10.15294/sji.v8i2.32484.
[25] K. I. Sundus, B. H. Hammo, M. B. Al-Zoubi, and A. Al-Omari, "Solving the multicollinearity problem to improve the stability of machine learning algorithms applied to a fully annotated breast cancer dataset," *Informatics Med. Unlocked*, vol. 33, no. September, p. 101088, 2022, doi: 10.1016/j.imu.2022.101088.
[26] A. A. Soomro *et al.*, "Data augmentation using SMOTE technique: Application for prediction of burst pressure of hydrocarbons pipeline using supervised machine learning models," *Results Eng.*, vol. 24, no. October, p. 103233, 2024, doi: 10.1016/j.rineng.2024.103233.
[27] H. Ding, N. Huang, Y. Wu, and X. Cui, "Improving imbalanced medical image classification through GAN-based data augmentation methods," *Pattern Recognit.*, vol. 166, no. February, p. 111680, 2025, doi: 10.1016/j.patcog.2025.111680.
[28] T. S. Lee, C. C. Chiu, Y. C. Chou, and C. J. Lu, "Mining the customer credit using classification and regression tree and multivariate adaptive regression splines," *Comput. Stat. Data Anal.*, vol. 50, no. 4, pp. 1113–1130, 2006, doi: 10.1016/j.csda.2004.11.006.
[29] G. Nie, W. Rowe, L. Zhang, Y. Tian, and Y. Shi, "Credit card churn forecasting by logistic regression and decision tree," *Expert Syst. Appl.*, vol. 38, no. 12, pp. 15273–15285, 2011, doi: 10.1016/j.eswa.2011.06.028.
[30] H. Jain, A. Khunteta, and S. Srivastava, "Churn Prediction in Telecommunication using Logistic

Regression and Logit Boost," *Procedia Comput. Sci.*, vol. 167, no. 2019, pp. 101–112, 2020, doi: 10.1016/j.procs.2020.03.187.
[31] A. Singh, B. S. Prakash, and K. Chandrasekaran, "A comparison of linear discriminant analysis and ridge classifier on Twitter data," *Proceeding - IEEE Int. Conf. Comput. Commun. Autom. ICCCA 2016*, pp. 133–138, 2017, doi: 10.1109/CCAA.2016.7813704.