COMPARATIVE ANALYSIS OF MACHINE LEARNING APPROACHES IN PREDICTING TELECOM CUSTOMER CHURN

Article *in* Educational Administration Theory and Practice journal · May 2024

DOI: 10.53555/kuey.v30i5.4348

2 authors, including:

Sheetal Mahendher
ISBR Business School
39 PUBLICATIONS **264** CITATIONS

SEE PROFILE

COMPARATIVE ANALYSIS OF MACHINE LEARNING APPROACHES IN PREDICTING TELECOM CUSTOMER CHURN

Debendra Das[1], Dr. Sheetal Mahendher[2]

1. FPM Scholar, ISBR Business School

2. Professor and Research Chair, ISBR Business School

Abstract: In the fast-paced telecommunications industry, businesses encounter challenges in retaining customers, underscoring the vital role of advanced predictive analytics. To overcome these obstacles, companies are increasingly utilizing machine learning (ML) to enhance the accuracy of customer churn predictions. ML leverages extensive datasets to reveal trends and factors that shape customer behaviour, allowing companies to proactively address churn. ML-driven models play a crucial role in identifying customers at risk of leaving, enabling personalized interventions that increase satisfaction and loyalty. By using ML, businesses gain insight into customer preferences and patterns, empowering them to tailor services and communications more effectively. This research explores the role of ML in forecasting telecom churn, examining the efficacy of various algorithms in identifying likely churners. The study assesses algorithms such as decision trees, logistic regression, neural networks and other, evaluating their performance in predicting churn and providing actionable insights. By analysing survey data on the telecom industry, how different ML models predict churn risk. The aim is to identify the most effective algorithms for accurate forecasts, enabling companies to strategically combat churn and enhance customer retention. Understanding the most successful models allows telecom companies to adopt ML-driven strategies that not only decrease churn rates but also improve customer experiences and loyalty. These insights support the creation of targeted, personalized approaches to customer engagement and retention. In summary, this research highlights the importance of ML in transforming churn prediction and management within the telecom sector. By harnessing ML's capabilities, businesses can stay competitive and agile, securing sustained success in an ever-evolving market.

Key words: machine learning, churn, telecom, customer satisfaction, labelled data analysis,

INTRODUCTION:

In the recently released report by the Telecom Regulatory Authority of India (TRAI) in February 2024, concerning the state of Mobile Number Portability (MNP) in the country, notable insights have emerged regarding the challenges faced by telecom operators in maintaining their customer base and reducing churn.

According to the report, the month of December 2023 witnessed a substantial surge in MNP requests, with a staggering 12.44 million subscribers opting for number portability. This significant influx of MNP requests contributed to a notable increase in cumulative MNP requests, which rose from 914.75 million at the end of November 2023 to 927.19 million by the end of December 2023 since the implementation of MNP.

These statistics underscore the ongoing churn dynamics within the Indian telecom sector, where customers are increasingly exercising their option to switch service providers while retaining their mobile numbers. Such behaviour poses a considerable challenge to telecom operators, as it directly impacts customer retention efforts and revenue streams.

To combat the issue of customer churn, telecom companies are resorting to advanced predictive modelling techniques and leveraging domain knowledge to develop robust churn prediction systems. By analysing various data points such as usage patterns, demographics, customer feedback, and MNP trends, these predictive models aim to identify customers at risk of churn and implement targeted retention strategies.

Despite the challenges posed by the continuous rise in MNP requests, telecom operators are proactively adapting their strategies to retain customers and enhance their overall service offerings. By prioritizing customer satisfaction, implementing personalized retention initiatives, and continually refining their predictive models, telecom companies endeavour to mitigate churn rates and foster long-term customer loyalty in the highly competitive Indian telecom market.

As the telecom landscape continues to evolve, effective churn management remains paramount for operators seeking to thrive in an environment characterized by rapid technological advancements and shifting

consumer preferences. Through strategic investments in predictive analytics and customer-centric initiatives, telecom operators can navigate the challenges posed by churn dynamics and emerge as leaders in delivering unparalleled connectivity experiences to their subscribers.

REVIEW LITERATURE:

The literature on machine learning techniques for telecom churn prediction offers a comprehensive overview of methodologies and their practical applications. Silva and Gama (2017) conducted a systematic literature review emphasizing predictive modelling techniques. Ren and Wang (2017) surveyed predictive modelling techniques for customer churn, providing insights into emerging trends. Subsequently, Huang and Liu (2018) conducted a comprehensive survey summarizing current practices. Zhao and Yu conducted (2018) worked on a systematic review, synthesizing existing knowledge and proposing future research directions. Chen and Wang (2018) reviewed machine learning applications and discussed future directions for research and development. Kim and Kang (2019) reviewed state-of-the-art machine learning-based methods, while Liu and Wu (2019) performed a comparative study identifying best practices. Li and Zhou (2019) reviewed recent studies and future trends, highlighting advancements and challenges. Furthermore, Xu and Li (2019) conducted a comparative study of machine learning algorithms, evaluating their performance. Wang and Zeng (2020) discussed recent advances and challenges, identifying key improvement areas. Chen and Zhang (2020) provided insights into challenges and potential solutions, focusing on machine learning approaches. Additionally, Arslan and Kocadag (2020) conducted a comprehensive review, offering a holistic view of machine learning applications. Khan and Hayat (2020) discussed methods and challenges, addressing current obstacles and opportunities. Transitioning to discussions cantered on the Indian telecom market, Mahajan and Wadhai (2018) provided a review of machine learning techniques tailored for the Indian context. Pandey and Mishra (2018) explored churn prediction in the Indian telecom industry, addressing challenges and opportunities. Sharma and Litoriya (2019) conducted a study on churn prediction in the Indian telecom industry, utilizing machine learning techniques. Gupta and Das (2020) investigated predicting customer churn in the Indian telecom sector using machine learning techniques. Singh and Tiwari (2021) contributed to recent advances in Indian telecom churn prediction using machine learning. Jain and Pancholi (2021) focused on machine learning approaches for predicting telecom churn in the Indian market, offering a systematic review.

These studies collectively contribute to a deeper understanding of machine learning applications in predicting telecom churn, both globally and within the dynamic Indian market landscape.

MAIN INFLUENCING FACTORS

Customer churn in the telecom industry can be influenced by various factors, both internal and external. Understanding these factors is crucial for telecom operators to develop effective retention strategies. Our investigation has explored the primary factors influencing customer churn within the telecommunications sector. This analysis provides valuable insights into effectively managing churn.

Technical Churn Factors:

1. **Network Quality**: Customers prioritize reliable network connectivity.
2. **Price and Plans**: Customers' decisions hinge on pricing flexibility, impacting churn likelihood elsewhere.
3. **Customer Service**: Customer service vital for addressing concerns, preventing dissatisfaction, and retaining customers
4. **Promotional Offers and Incentives**: Telecom firms often use promotions to keep customers.
5. **Technological Advancements**: Telecom firms need to keep up with tech trends for competitiveness
6. **Brand Reputation and Trust**: A telecom company's reputation is vital for keeping customers loyal.

Demographic Factors influencing on churn:

1. **Gender**: The gender of the customer, typically categorized as male and female for demographic analysis.
2. **Age group**: The age range of the customer, often grouped into intervals like Under 18, 18-24, 25-34, 35 and above
3. **Occupation**: The employment status or job type of the customer, such as student, professional, Government employee, Self-employee, etc.
4. **Spending amount for service per month**: The average amount a customer spends on telecom services each month, reflecting usage and engagement.
5. **Connection Type**: The type of connection a customer has, whether prepaid, postpaid, or corporate, affecting payment and usage patterns.
6. **Service time**: The length of time a customer has been using the same network provider, indicating loyalty and satisfaction.
7. **Contacted with customer service:** The number of times a customer has contacted customer service for support, reflecting potential issues or engagement.

8. **Preferred plan**: The type of billing plans a customer prefers, such as monthly, quarterly, half-yearly, or yearly, indicating usage and budgeting preferences.

Outcome variable:

Churn: The customer has discontinued using the telecom service provider. It is the main outcome variable in churn prediction studies.

Sampling:

In our telecom churn prediction study, we collected data from a sample of 600 customers. After cleaning the data to remove inconsistencies and incomplete records, we used a subset of 440 samples for analysis. This ensured that our dataset was reliable and representative.

Model building for predicting Churn:

To predict customer churn, we developed and evaluated several machines learning models, including Logistic Regression, Decision Tree, and Gradient Boosting, among others. These models were chosen for their suitability in handling classification tasks and their ability to capture complex patterns in the data.

Assessed the performance of Models:

We assessed the performance of each model using metrics such as accuracy, specificity, sensitivity, and F1-Score. By comparing these metrics, we identified the best-performing models for predicting customer churn in the telecom industry.

MACHINE LEARNING MODELS

The study focuses on various machine learning models utilized for analysis. These models are instrumental in predicting outcomes and patterns within the research's scope. By employing sophisticated algorithms, the study aims to explore the effectiveness of different machine learning approaches in addressing the research objectives.

Logistic Regression

Logistic regression is a statistical method used for binary classification, where the dependent variable takes one of two possible outcomes, typically represented as 0 or 1. It models the probability of the dependent variable belonging to a particular class based on one or more independent variables.

Mathematical Model for Logistic function

Logistic function (sigmoid function): The logistic regression model uses the logistic function to map the linear combination of the independent variables to a probability value between 0 and 1

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

where:

- $P(y=1|x)$ is the probability of the dependent variable (y) being in class 1 given the independent variables (x).
- $z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n$ is the linear combination of the independent variables and coefficients.
- $\beta_0, \beta_1, \beta_2, ..., \beta_n$ are the coefficients (parameters) of the model.
- e is the base of the natural logarithm.
- Logodds: The logistic function is often expressed in terms of logodds (also known as the logit function):

$$\left(\frac{P(y=1|x)}{1-P(y=1|x)} \right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n$$

Assumptions for Logistic Regression:

1. Linearity of the logit: The relationship between the logodds (logit) of the dependent variable and the independent variables should be linear.
2. Independence of errors: Observations should be independent of each other.
3. No multicollinearity: The independent variables should not be highly correlated with each other.
4. Binary outcome: The dependent variable should be binary (i.e., 0 or 1).
5. No extreme outliers: Extreme values in the independent variables can negatively affect the model's performance.
6. Large sample size: A larger sample size improves the reliability and validity of the logistic regression model.

Logistic regression is used for predicting telecom churn based several influencing predictors.

Decision Tree

Decision tree is a type of machine learning model used for classification and regression tasks. They work by splitting the data into subsets based on the values of the input features, making decisions at each internal node of the tree.

Mathematical Model for Decision Tree:

Tree Structure: A decision tree consists of a series of nodes connected by edges. The nodes can be:

Root node: The topmost node that contains the entire dataset.

Internal nodes: Nodes that split the data based on a specific feature and its value.

Leaf nodes (terminal nodes): Nodes that make the final prediction (class or value) based on the data subset.

Splitting Criteria: At each internal node, the decision tree uses a feature and a value to split the data into two or more subsets. The choice of the feature and value is made based on a splitting criterion, such as:

Gini impurity: Measures the impurity of a node. The goal is to minimize impurity when making a split.

$$G = 1 - \sum_{i=1}^{k} p_i^2$$

Where, p_i is the proportion of class *i* in the node, and *k* is the number of classes.

Information gain: Measures the reduction in entropy when making a split.

$$IG = Entropy(parent) - \left(\frac{n_{left}}{n_{parent}} \times Entropy(left) + \frac{n_{left}}{n_{parent}} \times Entropy(right) \right)$$

Where, Entropy is calculated for the parent node, left child, and right child, and n represents the number of instances in each node.

Prediction: When a new instance is fed into the decision tree, it follows the splits based on its feature values until it reaches a leaf node, which provides the final prediction.

Assumptions:

1. Feature Independence: Decision trees assume that features are independent of each other, allowing them to make unbiased splits.
2. Homogeneity in Leaves: The model aims to achieve homogeneity in each leaf node (i.e., instances in a leaf node should be similar in class or value).
3. Data Discretization: Continuous features may need to be discretized for efficient splitting.
4. Pruning for Overfitting: Without pruning, a decision tree may overfit the training data, resulting in poor generalization to new data.
5. Sample Size: A larger sample size can improve the stability and accuracy of the decision tree.

Decision trees are popular due to their simplicity and interpretability. They can handle both numerical and categorical data, making them a versatile choice for various applications. It is use for predict telecom churn.

Gradient boosting:

Gradient boosting is a machine learning technique used for classification and regression tasks. It is an ensemble learning method that combines multiple weak learners (often decision trees) to create a strong model. Gradient boosting works by iteratively training weak learners to correct the errors of previous learners, leading to a more accurate and robust model.

Mathematical Model Gradient boosting:

Ensemble Model: Gradient boosting combines a series of weak learners, typically decision trees, to form an ensemble model. The final prediction is the sum of the predictions from each tree

$$\hat{y} = \sum_{m=1}^{M} a_m f_m(x)$$

where:

$\hat{y}$ is the final prediction for an input instance.

M is the total number of weak learners (decision trees).

a_m is the learning rate, controlling the contribution of each tree.

$f_m(x)$ is the prediction from the m^{th} weak learner.

Loss Function: Gradient boosting uses a differentiable loss function to measure the error between the predicted values and actual values. For regression tasks, the loss function is usually mean squared error, while for classification tasks, it can be logarithmic loss or other suitable functions.

Gradient Calculation: In each iteration, the algorithm calculates the gradient (derivative) of the loss function with respect to the predicted values. This gradient represents the direction and magnitude of error correction needed.

$$g_{i,m} = \frac{\delta L}{\delta \hat{y}}\bigg|_{\hat{y}=\hat{y}_{i,m-1}}$$

Where,

$g_{i,m}$ is the gradient for the i[th] instance at the m[th] iteration.

$\hat{y} = \hat{y}_{i,m-1}$ is the predicted value for the i[th] instance at the previous iteration.

Training Weak Learners: The algorithm trains a new weak learner (decision tree) using the gradients calculated in the previous step. The tree is trained to predict the gradient rather than the actual target values.

Update Ensemble: The weak learner's predictions are added to the ensemble model, scaled by the learning rate:

$$\hat{y}_{i,m} = \hat{y}_{i,m-1} + \alpha_m f_m(x_i)$$

Where,

$\hat{y}_{i,m}$ is the updated prediction for the i[th] instance after adding the m[th] tree's predictions.

α_m is the learning rate.

Assumptions for Gradient boosting:

1. Weak Learners: Gradient boosting assumes that each weak learner is relatively simple and prone to underfitting. Decision trees are commonly used as weak learners.
2. Independence of Errors: The algorithm aims to minimize the correlation of errors across iterations.
3. Learning Rate: The learning rate (step size) controls how much each weak learner contributes to the final prediction. A lower learning rate can prevent overfitting but may require more iterations to converge.
4. Regularization: Techniques such as shrinkage, tree pruning, and limiting the number of trees can be applied to avoid overfitting.
5. Convergence: Gradient boosting converges when the model's predictions stabilize and the loss function no longer decreases significantly.
6. Data Quality: High-quality, representative data is crucial for the effectiveness of gradient boosting. Outliers and noise can negatively affect the model's performance.

Gradient boosting is a powerful technique known for its ability to achieve high accuracy and robustness, especially when fine-tuned with appropriate hyperparameters and regularization methods.

Support Vector Machines (Liner Kernel)

Support Vector Machines (SVMs) are supervised machine learning algorithms used for classification and regression tasks. An SVM with a linear kernel is specifically used for linear classification tasks, where it aims to find the optimal hyperplane that best separates the data into different classes.

Mathematical Model Support Vector Machines (Liner Kernel):

Decision Boundary: In linear SVM, the goal is to find a hyperplane that maximizes the margin between the classes and separates the data points into different classes. The hyperplane can be defined mathematically as:

$$w.x + b = 0$$

where:

w is the weight vector that defines the orientation of the hyperplane.

x is the feature vector of an instance.

b is the bias term, which controls the position of the hyperplane.

Decision Function: The decision function for classification is defined as:

$$sign(w.x + b)$$

The function returns either 1 or -1, representing the class of the input instance.

Optimization Problem: SVM aims to maximize the margin (the distance) between the support vectors and the hyperplane while minimizing the classification error. This is formulated as the following optimization problem:

$$\min_{w,b}\left(\frac{1}{2}\|w\|^2\right)$$

Subject to:

$$y_i(w.x_i + b) \geq 1 \ \forall i$$

Where:

y_i is the class label of the i[th] instance (1 or -1).

x_i is the feature vector of the i[th] instance.

The constraint ensures that data points of each class are correctly classified and are at least a certain distance (margin) away from the hyperplane.

Assumptions for Support Vector Machines (Liner Kernel):

1. Linearity: Linear SVM assumes that the data is linearly separable, meaning a linear hyperplane can separate the classes.
2. Independence: Observations are assumed to be independent and identically distributed (i.i.d.).
3. No Multicollinearity: Independent variables should not be highly correlated with each other.
4. Balanced Classes: SVM works best when classes are balanced; otherwise, adjustments may be needed.
5. Kernel Selection: For linear SVM, a linear kernel is used. Other types of kernels may be used for non-linear data, but this changes the nature of the model.
6. Regularization: Regularization (e.g., C parameter) helps control overfitting and manage the trade-off between maximizing the margin and minimizing classification error.

SVM with a linear kernel is a simple yet powerful method for linearly separable classification tasks. It can efficiently classify data and is often used as a baseline method for comparison with more complex models.

Random Forest

Random Forest is an ensemble learning method used for classification and regression tasks. It combines multiple decision trees to create a strong predictive model. Each tree in the forest is trained on a different subset of the data and features, providing a diverse set of predictions. The final prediction is made by aggregating the predictions from all the trees.

Mathematical Model for Random Forest:

Ensemble of Trees: Random Forest consists of an ensemble of MM decision trees. For each tree:

- Bootstrap Sampling: A random sample (with replacement) is drawn from the training data to create a bootstrap sample. Each tree is trained on its own bootstrap sample.
- Random Feature Selection: At each node of the decision tree, a random subset of features is considered for splitting. This adds diversity among the trees.

Aggregation of Predictions: For classification, the final prediction is determined by majority voting among all tree

$$\hat{y} = \text{mode}\left(f_1(x), f_2(x), ..., f_M(x) \right)$$

For regression, the final prediction is the mean of the predictions from all trees:

$$\hat{y} = \frac{1}{M} \sum_{i=1}^{M} f_i(x)$$

Where:

$\hat{y}$ is the final prediction for an input instance xx.

$f_i(x)$ is the prediction from the i^{th} tree.

M is the total number of trees in the forest.

Assumptions Random Forest:

1. Independence of Trees: The trees in the forest are assumed to be independent of each other. This is achieved by training each tree on a different bootstrap sample and considering only a subset of features at each split.
2. Generalization and Robustness: By combining multiple decision trees, Random Forest can generalize better and be more robust against overfitting than individual decision trees.
3. Randomness and Diversity: Randomness in sampling and feature selection introduces diversity among the trees, which helps improve the model's performance and stability.
4. Out-of-Bag (OOB) Estimation: Random Forest can use OOB samples (instances not included in a tree's bootstrap sample) to estimate model performance without needing a separate validation set.
5. Feature Importance: Random Forest provides a measure of feature importance, which can help identify which features contribute most to the model's predictions.
6. Assumption of Homogeneous Data: Random Forest assumes that the data has a consistent distribution across the training set, which may affect its performance on imbalanced or noisy data.

Random Forest is known for its high accuracy and ability to handle complex datasets. Its ensemble approach, with built-in randomization and diversity, makes it a versatile and powerful model for a wide range of applications.

Support Vector Machines (RBF kernel):

Support Vector Machines (SVMs) are a type of supervised machine learning model used for classification and regression tasks. When using a radial basis function (RBF) kernel, SVM can handle non-linearly separable

data by mapping the input features to a higher-dimensional space and finding a hyperplane that best separates the classes.

Mathematical Model for Support Vector Machines (RBF kernel):

RBF Kernel: The RBF kernel (also known as the Gaussian kernel) transforms the input data into a higher-dimensional space.

It is defined as:

$$K(x, z) = e^{\left(-\gamma \|x-z\|^2\right)}$$

Where,

x and z are two input vectors.

γ is the kernel coefficient that controls the width of the kernel and influences how far the influence of a single training example reaches.

$\|x-z\|^2$ is the squared Euclidean distance between the two vectors.

Optimization Problem: SVM aims to maximize the margin (distance) between the classes while minimizing classification errors.

This is formulated as the following optimization problem:

$$\min_a \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(x_i, x_j) - \sum_i \alpha_i$$

Subject to: $\sum_i \alpha_i y_i = 0$ and $0 \le \alpha_i \le C \ \forall i$

Where:

α_i and α_j are Lagrange multipliers for each instance.

y_i and y_j are class labels (1 or -1).

C is the regularization parameter that controls the trade-off between maximizing the margin and minimizing the classification error.

Decision Function: The decision function for classification is defined as:

$$f(x) = \sum_i \alpha_i y_i K(x_i, x) + b$$

Where:

α_i and y_i are Lagrange multipliers and class labels, respectively, for each support vector.

$K(x_i, x)$ is the kernel function applied between the support vector x_i and the input instance x. b is the bias term.

Assumptions for Support Vector Machines (RBF kernel):

1. Linearity in Transformed Space: The RBF kernel allows SVM to handle non-linearly separable data by mapping it to a higher-dimensional space where a linear decision boundary can be found.
2. Proper Kernel Parameters: The performance of SVM with RBF kernel is sensitive to the choice of γ and C. Proper tuning of these parameters is essential for optimal performance.
3. Support Vector Representation: The model's decision function relies on a subset of training data points known as support vectors.
4. Independence of Instances: Observations in the data should be independent of each other.
5. Balanced Classes: SVM works best when classes are balanced. If the classes are imbalanced, adjustments may be necessary.
6. Sensitivity to Outliers: The model can be sensitive to outliers, so outlier detection and removal may be beneficial.

Support Vector Machines with RBF kernel is powerful for handling complex, non-linear datasets, and can achieve high accuracy when properly tuned.

K-Nearest Neighbours (KNN):

K-Nearest Neighbours (KNN) is a simple yet effective machine learning algorithm used for classification and regression tasks. It is a non-parametric and instance-based learning method that makes predictions based on the K nearest data points in the training set.

Mathematical Model for K-Nearest Neighbours (KNN):

Distance Metric: KNN relies on a distance metric to identify the K nearest neighbours to a given input instance. The most commonly used metric is Euclidean distance, but other metrics like Manhattan distance or Minkowski distance can also be used:

$$d(x,y)=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$$

Where:

x and y are two input feature vectors.

x_i and y_i are the feature values of x and y, respectively.

Finding K Nearest Neighbours: For a given input instance x, KNN finds the K closest training instances (neighbours) based on the chosen distance metric.

Classification Prediction: For classification tasks, KNN uses a majority voting approach. The predicted class is the class most frequently represented among the K nearest neighbours:

$$\hat{y}=\text{mode}(y_i,y_2,...,y_k)$$

Where:

$\hat{y}$ is the predicted class for the input instance.

$y_i,y_2,...,y_k$ are the class labels of the K nearest neighbours.

Regression Prediction: For regression tasks, KNN calculates the average value of the K nearest neighbours: $\hat{y}=\frac{1}{K}\sum_{i=1}^{K}y_i$

Where:

$\hat{y}$ is the predicted value for the input instance.

y_i is the value of the ith nearest neighbour.

Assumptions for K-Nearest Neighbours (KNN):

1. Consistency: The K nearest neighbours should be representative of the underlying data distribution.
2. Noisy Data Sensitivity: KNN can be sensitive to noise and outliers, which may affect the model's accuracy.
3. Distance Metric Choice: The choice of distance metric influences the accuracy of KNN, so selecting an appropriate metric based on the data type and domain is essential.
4. Value of K: Choosing the right K value is crucial for model performance. A small K may lead to overfitting, while a large K may lead to underfitting.
5. High Dimensionality: KNN may struggle with high-dimensional data (the "curse of dimensionality"), as it becomes computationally expensive and distances between points may become less meaningful.
6. Feature Scaling: Feature scaling is important because KNN relies on distance measurements. Inconsistent feature scales can skew the results, so it's important to normalize or standardize the data.

KNN is known for its simplicity and effectiveness, especially when the data is well-structured and the right value of K is chosen. It is a versatile algorithm, particularly useful for tasks where interpretability and ease of implementation are important.

Neural networks:

Neural networks are machine learning models inspired by the structure and function of the human brain. They consist of interconnected nodes (neurons) arranged in layers. These networks can learn to approximate complex functions and are commonly used for classification and regression tasks.

Mathematical Model for Neural networks:

Architecture: A neural network typically consists of an input layer, one or more hidden layers, and an output layer. Each layer contains multiple neurons.

Neuron: Each neuron receives inputs, processes them, and produces an output. Mathematically, the output of a neuron j in a layer l can be expressed as:

$$z_j^{(l)}=\sum_i \omega_i^{(l)}a_i^{(l-1)}+b_j^{(l)}$$

$$\alpha_j^{(l)}=\sigma\left(z_j^{(l)}\right)$$

Where:

$z_j^{(l)}$ is the weighted sum of inputs to neuron j in layer l.

$\omega_i^{(l)}$ is the weight connecting neuron i in layer l-1 to neuron j in layer l.

$a_i^{(l-1)}$ is the output (activation) of neuron i in the previous layer l-1.

$b_j^{(l)}$ is the bias term for neuron j in layer l.

σ is the activation function, such as the sigmoid, ReLU, or tanh function.

Activation Functions: Activation functions introduce non-linearity to the network, allowing it to model complex relationships in the data. Common activation functions include:

Sigmoid: $\sigma(z) = \frac{1}{1+e^{-z}}$

ReLU (Rectified Linear Unit): $\sigma(z) = \max(0, z)$

Tanh: $\sigma(z) = \tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$

Output Layer: The output layer produces the final prediction for the network. For classification tasks, the output layer often uses a SoftMax function to produce probabilities for each class. For regression tasks, it may use a linear activation function.

Loss Function: The network uses a loss function to measure the difference between the predicted output and the actual target values. Common loss functions include mean squared error for regression and cross-entropy for classification.

Backpropagation: The network uses the backpropagation algorithm to update the weights and biases. It calculates the gradient of the loss function with respect to each parameter and adjusts the parameters using gradient descent or a variant.

Assumptions for Neural networks:

1. Independence of Observations: Neural networks assume that observations in the data are independent of each other.
2. Data Representativeness: The training data should be representative of the problem being modelled.
3. Sufficient Data: Neural networks typically require large amounts of data to train effectively, particularly for deeper architectures.
4. Learning Rate and Optimization: Proper tuning of the learning rate and optimization algorithm is essential for efficient training.
5. Feature Scaling: Feature scaling is important for neural networks to converge effectively and quickly.
6. Avoiding Overfitting: Regularization methods, such as dropout and L2 regularization, are often used to prevent overfitting.
7. Complexity Control: The complexity of the network (e.g., number of layers and neurons) should be appropriate for the task at hand.

Neural networks can model complex patterns and relationships in data, making them powerful tools for various applications such as image recognition, natural language processing, and more. However, they require careful tuning and sufficient data for optimal performance.

COMPARATIVE METRICS

Comparative metrics objectively assess and compare above mentioned ML models on our dataset. They help determine the best-performing model, evaluate trade-offs between metrics such as precision, recall and F1-score, and guide model selection based on specific task needs and desired outcomes.

Confusion Matrix:

Confusion matrix is a table that is used to evaluate the performance of a classification model. It provides a summary of the predictions made by the model compared to the actual labels in the dataset. It is

	Predicted Positive	Predicted Negative
Actual Positive	True Positive	Falls Negative
Actual Negative	Falls Positive	True Negative

Accuracy

Accuracy, a vital metric, gauges a classification model's correctness. It reflects the ratio of correctly classified instances to the total dataset size. From the confusion matrix, accuracy is computed as

$$Accuracy = \frac{True\ Positives + True\ Negatives}{Total\ number\ of\ \ predictions}$$

It offers a broad evaluation of the model's performance across all classes. Higher accuracy signals more accurate predictions, while lower accuracy implies more errors. Yet, accuracy alone might not fully depict performance, especially with imbalanced datasets. In such scenarios, accuracy can mislead, favouring the majority class. Hence, it's crucial to interpret accuracy alongside other metrics like Sensitivity, Specificity, and F1-score for a holistic assessment.

Sensitivity/Recall:

Sensitivity, also termed recall or true positive rate, assesses a classification model's ability to correctly recognize positive instances among all actual positives in the dataset. From the confusion matrix it computed as:

$$Sensitivity = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

It gauges the model's aptitude to avoid false negatives, which occur when positive instances are inaccurately classified as negative. A high sensitivity value signifies adeptness in identifying positives, whereas a low value implies missed positives. Particularly critical in fields like medical diagnosis or fraud detection, sensitivity warrants interpretation alongside metrics like specificity and precision for a thorough evaluation of model performance.

Specificity:

Specificity, a key metric, assesses a classification model's capacity to accurately detect negative instances among all true negatives in the dataset. From the confusion matrix it computed as

$$Specif\ icity = \frac{True\ Negatives}{True\ Negatives + False\ Positives}$$

It measures the model's adeptness in steering clear of false positives, where negative instances are erroneously classified as positive. High specificity signals proficiency in identifying negatives, while low values indicate misclassification of some negatives as positives. This metric shed light on the model's ability to capture genuine negative instances, crucial for various applications, such as disease screening or anomaly detection.

Precision:

Precision, a vital metric, evaluates a classification model's accuracy, especially regarding positive predictions. It gauges the ratio of true positive predictions (correctly predicted positives) to all positive predictions. From the confusion matrix it computed as:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

Precision measures the model's proficiency in avoiding false positives. A high precision value indicates adeptness in identifying positives while minimizing false alarms. Conversely, low precision suggests mislabelling negatives as positives. Precision holds significance in scenarios with high costs for false positives or where accurate positive predictions are crucial, like medical diagnosis, minimizing unnecessary treatments. However, it's crucial to interpret precision alongside other metrics like recall and F1-score for a comprehensive evaluation of model performance.

F1-Score:

The F1 score is a performance metric used in machine learning classification tasks, particularly when dealing with imbalanced data. It is the harmonic mean of precision and recall (sensitivity), and it balances the trade-off between the two metrics. The F1 score provides a single measure that combines the model's ability to correctly identify positive instances (recall) and the proportion of correct positive predictions made by the model (precision).

$$F1 - Score = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recoll}}$$

All metrics range from 0 to 1 (or 0% to 100%), with higher values indicating better model performance., while a score of 0 indicates poor performance.

OBJECTIVE OF THE STUDY:

This study aims to evaluate the efficiency of machine learning (ML) models in forecasting telecom customer churn. It investigates different algorithms to gauge their precision in detecting customer churn within the telecommunications industry.

DATA ANALYSIS:

Table. 1: Distribution customer churn in gender and age group wise.

Age Group	Gender	Churn	
		No	Yes
Under 18	Female	3%	2%
	Male	7%	8%
18-24	Female	6%	11%
	Male	14%	31%
25-34	Female	2%	4%
	Male	3%	8%
Above 35	Female	0%	0%

	Male	1%	2%
Total		35%	65%

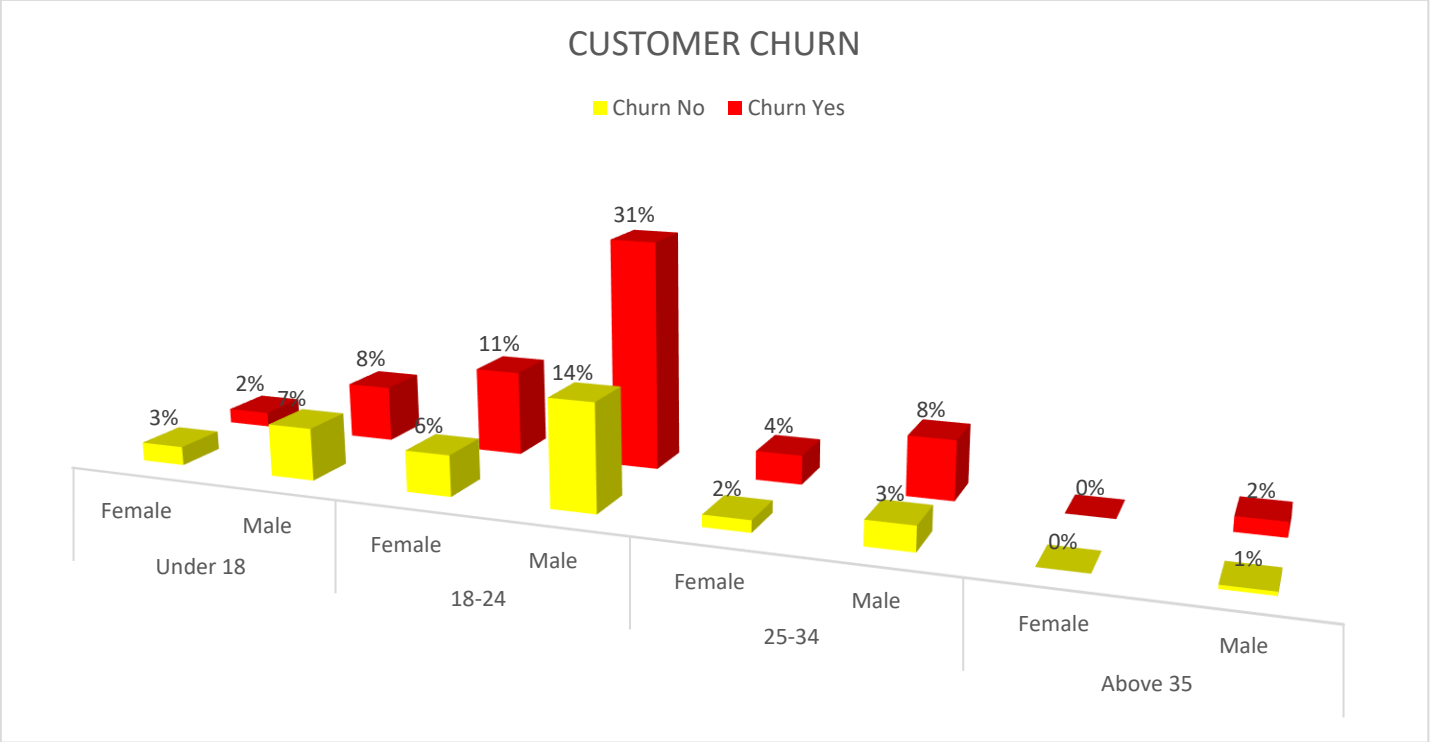

Fig. 1: Distribution customer churn in gender and age group wise.

Table 2: Distribution of churn customers in service time with operator and number of times contact with customer care before churn.

Duration of Service with the Operator	Contacted customer service before switching	Churn	
		No	Yes
Less than 5 years	Never	1%	4%
	1-2 times	21%	13%
	3-5 times	4%	22%
	More than 5 times	0%	7%
5 to 10 years	Never	0%	3%
	1-2 times	4%	4%
	3-5 times	2%	7%
	More than 5 times	1%	4%
More than 10 years	Never	0%	1%
	1-2 times	2%	0%
	3-5 times	0%	1%
	More than 5 times	0%	1%
Total		35%	65%

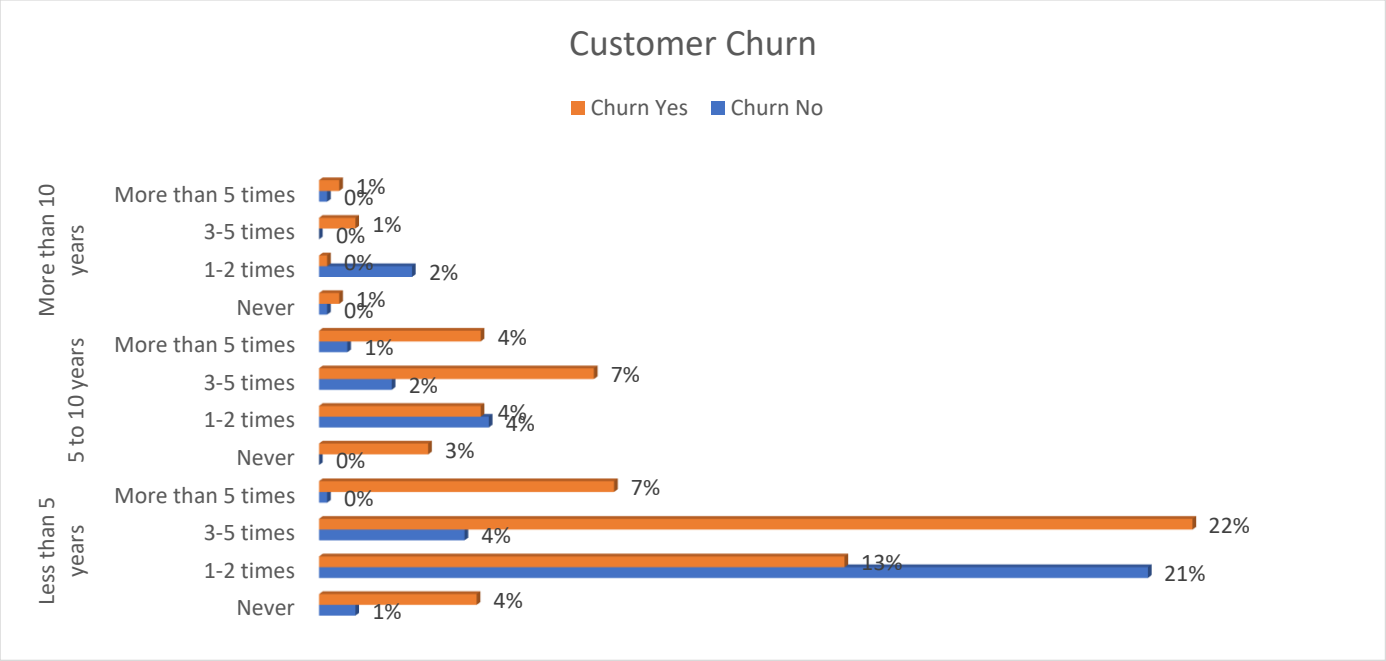

Fig.2: Distribution of churn customers in service time with operator and number of times contact with customer care before churn.

Table 3: The comparative study of various ML models in customer churn prediction.

Model	Accuracy	Sensitivity	Specificity	F1Score
Logistic Regression	66%	73%	52%	61%
Gradient Boosting	78%	81%	72%	77%
Support Vector Machine (Liner Kernel)	78%	81%	72%	77%
Random Forest	84%	90%	72%	80%
Support Vector Machine (RBF Kernel)	78%	85%	66%	74%
K-Nearest Neighbours	73%	76%	66%	70%
Neural Network	68%	88%	28%	42%
Decision Tree	78%	83%	69%	75%

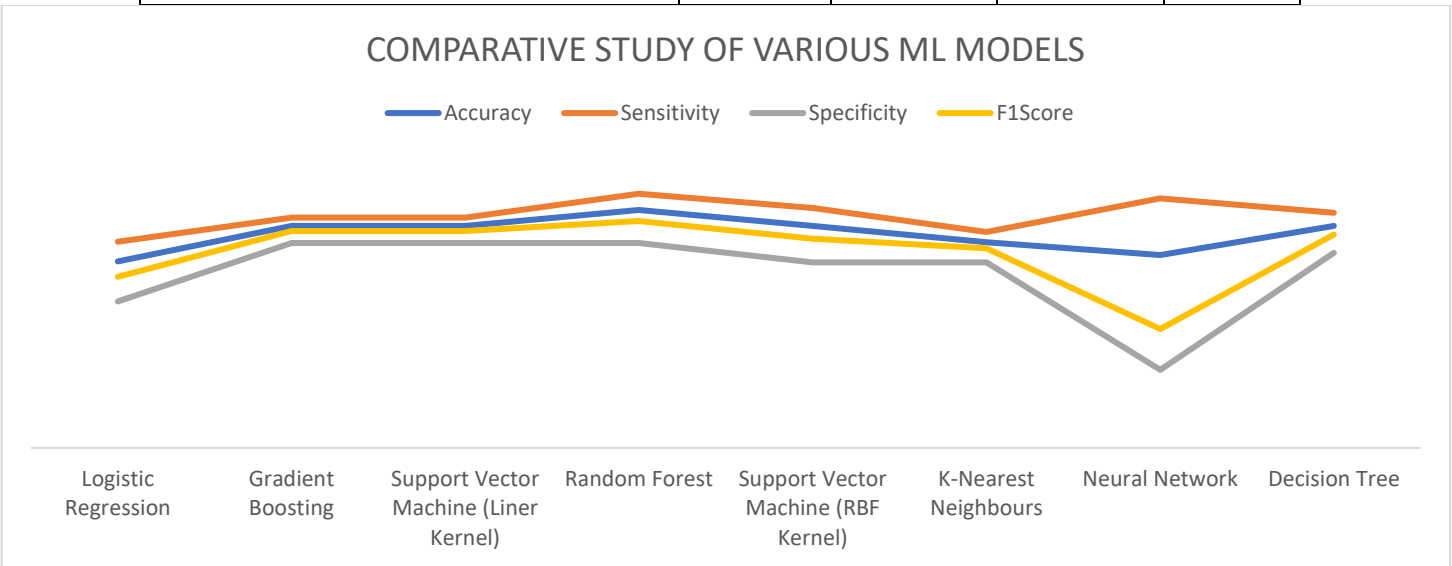

Fig. 3: The comparative study of various ML models in customer churn prediction.

INTERPRETATION:

Among all churned customers, 65% changed operators. Of this, 48% were male. Additionally, within the churned customer base, 42% belonged to the age group 18-24. These statistics provide insights into the demographics and behaviour of customers leaving their telecom operators.

Among the total churn rate of 65%, 46% of customers switch telecom operators within 5 years of service. Among these, 22% contacted customer service at least 3 to 5 times before switching. These statistics highlight the timeframe and interaction patterns preceding customer decisions to change operators.

The Random Forest model demonstrates strong predictive performance for churn, boasting an 84% accuracy rate, indicating that 84% of predictions are correct. Additionally, it achieves a high sensitivity of 90%,

meaning it accurately identifies 90% of churn cases. While the specificity and F1-Score are slightly lower, at 72% and 80% respectively, they still showcase its effectiveness in both true positive and true negative identification.

In contrast, the Neural Network model yields lower performance metrics. It achieves a lower accuracy of 68%, suggesting less accurate predictions overall. Though its sensitivity remains relatively high at 88%, indicating its ability to identify churn cases, the specificity and F1-Score are notably lower, at 28% and 42% respectively. These lower scores suggest challenges in correctly identifying true negatives and maintaining a balance between precision and recall. Overall, while the Neural Network model performs adequately in sensitivity, its lower specificity and F1-Score indicate potential areas for improvement compared to the Random Forest model

CONCLUSION:

In our study, we found that within the total churn rate of 65%, 48% of male customers switch their telecom operator, and 42% of both male and female customers make the switch within the age group of 18-24. Consequently, telecom operators are advised to target male customers aged 18-24 for retention strategies.

Additionally, among the churned customers, 46% switch their telecom provider within 5 years, and 22% of customers decide to switch after contacting customer care 3 to 5 times. Thus, operators should focus on retaining customers who have been with them for less than 5 years and those who have engaged with customer care multiple times.

Furthermore, our analysis revealed that the Random Forest method outperformed other ML models in predicting customer churn. Therefore, we recommend the use of the Random Forest method for accurately predicting customer churn based on labelled feedback data.

Acknowledgement: I extend my heartfelt thanks to all survey participants.

BIBLIOGRAPHY

Arslan, F. &. (2020). A Comprehensive Review on Machine Learning Applications for Churn Prediction in Telecommunications. *International Journal of Computer Networks and Applications, 7*(1), 11-21.

Chen, S. &. (2019). Machine Learning Approaches for Churn Prediction in Telecommunication Industry: A Comprehensive Review. *Wireless Communications and Mobile Computing*, 1-17.

Chen, Y. &. (2018). A Review of Machine Learning Applications for Telecom Churn Prediction. *Journal of Computational Intelligence and Applications, 7*(2), 27-35.

Gupta, R. &. (2020). A Review of Machine Learning Techniques for Churn Prediction in Telecom Industry. *International Journal of Computer Sciences and Engineering (IJCSE), 8*(3), 118-122.

Gupta, S. &. (2020). Predicting Customer Churn in Indian Telecom Sector Using Machine Learning Techniques. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 5*(3), 1-6.

Huang, L. &. (2018). Machine Learning Techniques for Telecom Churn Prediction: A Survey. *Journal of Information Science and Engineering, 34*(6), 1281-1304.

India., T. R. (2024). *Press note N0. 07/2024,.* New Delhi: Press note N0. 07/2024,.

Jain, R. &. (2021). Machine Learning Approaches for Telecom Churn Prediction. *International Journal of Computer Applications, 181*(25), 6-11.

Jha, R. &. (2020). Machine Learning Based Churn Prediction Models for Indian Telecom Sector. *International Journal of Engineering Research & Technology, 9*(2), 23-35.

Khan, F. &. (2020). Machine Learning Techniques for Customer Churn Prediction in Telecom Industry: A Review. *Future Internet, 12*(2), 37-47.

Kim, S. &. (2019). A Review of Machine Learning Techniques for Churn Prediction in Telecom Industry. *Expert Systems with Applications, 123*, 439-451.

Kumar, A. &. (2018). A Survey on Churn Prediction in Telecom Industry using Machine Learning Techniques. *International Journal of Computer Science and Information Technologies, 9*(3), 1471-1475.

Li, H. &. (2018). Machine Learning Methods for Churn Prediction in Telecom Industry: A Review. *Journal of Information Science and Engineering, 34*(5), 1021-1041.

Li, Y. &. (2019). A Review of Machine Learning Methods for Telecom Customer Churn Prediction. *Journal of Communications and Information Networks, 4*(3), 66-72.

Liu, J. &. (2019). A Comparative Study of Machine Learning Techniques for Churn Prediction in Telecommunication. *International Journal of Machine Learning and Cybernetics, 10*(2), 239-255.

Mahajan, V. &. (n.d.). A Review of Machine Learning Techniques for Telecom Churn Prediction. *International Journal of Engineering Research & Technology, 7*(11), 663-667.

Pandey, S. &. (2018). A Survey on Churn Prediction in Telecom Industry Using Machine Learning Techniques. *International Journal of Pure and Applied Mathematics, 118*(24), 625-632.

Park, J. &. (2021). Telecom Churn Prediction Using Machine Learning: Challenges and Opportunities. *Information Systems Frontiers, 23*(2).

Patil, A. &. (2019). Churn Prediction in Indian Telecom Industry Using Machine Learning. *International Journal of Advanced Research in Computer Engineering & Technology, 8*(12), 95-100.

Patil, P. A. (2019). A Survey on Telecom Customer Churn Prediction using Machine Learning Techniques. *International Journal of Recent Technology and Engineering, 8*(1), 3009-3013.

Pawar, R. &. (2017). A Survey of Churn Prediction in Indian Telecom Industry. *International Journal of Computer Applications, 160*(5), 1-5.

Ren, J. &. (2017). A Survey on Predictive Modeling of Customer Churn in Telecommunication Industry. *Journal of Computer Applications, 35*(5).

Sharma, A. &. (2019). Churn prediction in Indian telecom industry using machine learning techniques. *International Journal of Engineering and Advanced Technology, 8*(2), 2115-2120.

Sharma, S. &. (2021). Churn Prediction in Indian Telecom Sector Using Machine Learning Techniques: A Review. *International Journal of Recent Technology and Engineering, 9*(3), 10450-10455.

Silva, T. &. (2017). A Survey on Predictive Modeling of Telecom Customer Churn. *IEEE Transactions on Neural Networks and Learning Systems, 28*(10), 2374-2396.

Singh, A. &. (2021). Churn Prediction in Telecom Industry Using Machine Learning. *International Journal of Emerging Trends & Technology in Computer Science, 10*(4), 52-57.

Singh, V. &. (2019). Machine Learning Approaches for Telecom Churn Prediction. *International Journal of Engineering and Advanced Technology, 8*(2), 672-676).

Singh, Y. P. (2022). *Machine learning for telecommunication network planning and optimization.* Cambridge University Press.

Tiwari, P. &. (2017). A Review on Churn Prediction in Telecom Industry Using Machine Learning. *International Journal of Advanced Research in Computer Science, 8*(2), 500-504.

Wang, C. &. (2020). A Review of Machine Learning Techniques for Telecom Customer Churn Prediction. *Journal of Communications, 41*(3), 65-75.

Wang, Y. &. (2020). A Survey on Machine Learning Methods for Churn Prediction in Telecom Industry. *IEEE Access, 8*, 101315-101330.

Xu, S. &. (2019). Comparative Study of Machine Learning Algorithms for Telecom Customer Churn Prediction. *International Journal of Pattern Recognition and Artificial Intelligence, 33*(7), 1955015.

Yang, L. &. (2018). Telecom Customer Churn Prediction: A Review of Machine Learning Approaches. *International Journal of Computational Intelligence Systems, 11*(7), 1289-1299.

Zhang, Y. &. (2021). Telecom Customer Churn Prediction: A Review and Future Directions. *Frontiers in Big Data, 4*.

Zhao, Y. &. (2019). Machine Learning Techniques for Telecom Churn Prediction: A Review. *Journal of Telecommunications and Information Technology, 3*, 37-46.

Zhou, L. &. (2018). Telecom Customer Churn Prediction Using Machine Learning: A Comprehensive Review. *Journal of Big Data, 5*(1), 1-22.