CUSTOMER CHURN ANALYSIS AND PREDICTION

Dissertation submitted in fulfillment of the requirements for the Degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

ESHWAR REDDY

12114438

Supervisor

VED PRAKASH CHAUBEY

School of Computer Science and Engineering

Lovely Professional University

Phagwara, Punjab (India)

April,2024

DECLARATION STATEMENT

I hereby declare that the research work reported in the dissertation/dissertation proposal entitled " CUSTOMER CHURN ANALYSIS AND PREDICTION" in partial fulfillment of the requirement for the award of Degree for Bachelor of Technology in Computer Science and Engineering at Lovely Professional University, Phagwara, Punjab is an authentic work carried out under the supervision of my research supervisor Mr. Ved Prakash Chaubey. I have not submitted this work elsewhere for any degree ordiploma.

I understand that the work presented herewith is in direct compliance with Lovely Professional University's Policy on plagiarism, intellectual property rights, and the highest standards of moral and ethical conduct. Therefore, to the best of my knowledge, the contentof this dissertation represents an authentic and honest research effort conducted, in its entirety, by me. I am fully responsible for the contents of my dissertation work.

Signature of Candidate

ESHWAR REDDY

SUPERVISOR'S CERTIFICATE

This is to certify that the work reported in the B. Tech Dissertation/dissertation proposal entitled " **CUSTOMER CHURN ANALYSIS AND PREDICTION "** submitted by **ESHWAR REDDY** at **Lovely Professional University, Phagwara, India** is a bonafide record of his / her original work carried out under mysupervision. This work has not been submitted elsewhere for any other degree.

Signature of Supervisor

(Ved Prakash Chaubey)

Date:

Counter Signed by:

1) Concerned HOD:
HoD's Signature:_______________

HoD Name:___________________

Date:__________________

2) Neutral Examiners:

External Examiner

Signature:_______________

Name:________________

Affiliation: _______________

Date:__________________

Internal Examiner

Signature:_____________

Name:_______________

Date: __________________

Acknowledgment

"GOD HELPS THOSE WHO HELP THEMSELVES."
"ARISE! AWAKE! AND STOP NOT UNTIL THE GOAL IS REACHED."

Success often requires preparation, hard work, and perspiration. The path to success is a long journey that calls for tremendous effort with many bitter and sweet experiences. This can only be achieved by the Graceful Blessing from the Almighty on everybody. I want to submit everything beneath the feet of God.

I want to acknowledge my regards to my teacher, Mr. Ved Prakash Chaubey, for his constant support and guidance throughout my training. I would also like to thank HOD Ms. Harjeet Kaur, School of Computer Science and Engineering for introducing such a great program.

I may be failing in my duties if I do not thank my parents for their constant support, suggestion, inspiration and encouragement and best wishes for my success. I am thankful for their supreme sacrifice, eternal benediction, and ocean-like bowls full of love and affection.

Abstract

In today's fiercely competitive market, understanding and predicting customer churn have become pivotal for businesses striving to maintain sustainable growth and profitability. This abstract encapsulates the essence of a customer churn analysis and prediction project, shedding light on its significance, methodologies, and implications.

Customer churn, the phenomenon of customers discontinuing their association with a company, poses a substantial challenge across diverse industries. This project endeavors to unravel the intricate dynamics of churn by leveraging data-driven approaches to unearth underlying patterns and drivers.

The project embarks on a voyage of data exploration, navigating through extensive datasets to discern meaningful insights. Exploratory data analysis (EDA) techniques are employed to uncover correlations, trends, and anomalies, laying the foundation for subsequent analysis. Feature engineering assumes paramount importance in discerning pertinent predictors of churn. Through meticulous feature selection and extraction, the project endeavors to distill the most influential variables, empowering predictive models with actionable insights.

A diverse array of machine learning algorithms, including logistic regression, decision trees, random forest, and XGBoost, are harnessed to construct predictive models. These models are honed and fine-tuned to achieve optimal performance, facilitating accurate churn prediction. Rigorous evaluation methodologies, encompassing cross-validation, receiver operating characteristic (ROC) analysis, and precision-recall curves, are employed to gauge the efficacy of predictive models. Performance metrics are scrutinized to ascertain model robustness and generalization capabilities.

The culmination of the project yields invaluable insights into customer behavior and churn dynamics. Armed with predictive models and actionable insights, businesses are empowered to preemptively identify churn-prone customers and devise targeted retention strategies, thereby fostering customer loyalty and enhancing organizational resilience.

In conclusion, this abstract encapsulates the essence of a customer churn analysis and prediction project, elucidating its significance, methodologies, and implications. By unraveling the enigma of churn and empowering businesses with predictive insights, the project heralds a paradigm shift in customer retention strategies, heralding a future of sustained growth and competitiveness.

Introduction

In today's fast-paced business environment, maintaining a loyal customer base is just as crucial as acquiring new ones. Customer churn, where customers disengage from a company's services, presents a formidable challenge across industries. Its impact extends beyond mere revenue loss, affecting brand perception and competitive edge. Acknowledging the urgency of understanding and predicting customer churn, businesses are increasingly turning to cutting-edge analytics and machine learning.

This introduction serves as a prelude to a customer churn analysis and prediction initiative aimed at unraveling churn drivers and crafting predictive models. Leveraging historical data and advanced analytical tools, companies stand to glean invaluable insights into churn determinants, enabling them to proactively implement retention strategies.

The evolution of big data and the refinement of machine learning algorithms have reshaped how businesses tackle churn analysis. Traditional methods like manual segmentation are making way for data-driven approaches harnessing predictive analytics' prowess to forecast churn with precision.

Our project embarks on an exploration of churn intricacies, spanning from data preprocessing to model validation. By employing a blend of exploratory data analysis, feature engineering, and machine learning algorithms such as logistic regression, decision trees, random forest, and XGBoost, our objective is to unearth actionable insights and craft robust predictive models.

Through this endeavor, our aim is to arm businesses with the insights and tools required to mitigate churn, amplify customer satisfaction, and nurture enduring relationships. By anticipating churn and deploying proactive retention tactics, companies not only fortify their financial standing but also cultivate a steadfast customer base primed for sustained growth in today's fiercely competitive market landscape.

OBJECTIVES

1. Understand Churn Dynamics: Gain insights into the factors contributing to customer churn within the specific industry or business context. This involves analyzing historical churn patterns and identifying key drivers behind customer attrition.

2. Data Preparation and Pre-processing: Collect, clean, and preprocess relevant data sources to ensure data quality and consistency. This step involves handling missing values, encoding categorical variables, and transforming data into a suitable format for analysis.

3. Exploratory Data Analysis (EDA): Conduct comprehensive EDA to uncover patterns, trends, and correlations within the data. This involves visualizing data through graphs and charts to identify potential churn indicators and understand customer behavior dynamics.

4. Feature Engineering: Extract meaningful features from the data that are predictive of churn. This may involve creating new features, transforming existing ones, and selecting the most relevant variables for model building.

5. Model Building: Develop predictive models using machine learning algorithms to forecast customer churn. Experiment with a variety of models such as logistic regression, decision trees, random forest, XGBoost, etc., to identify the most effective approach for the specific business problem.

6. Model Evaluation and Selection: Assess the performance of the developed models using appropriate evaluation metrics such as accuracy, precision, recall, F1-score, and ROC curves. Select the best-performing model(s) based on their ability to accurately predict churn and generalize to unseen data.

7. Interpretation and Insights: Interpret the results of the predictive models to extract actionable insights for business stakeholders. Understand the relative importance of different features in predicting churn and identify potential intervention strategies to mitigate churn risk.

8. Deployment and Monitoring: Implement the chosen model(s) into operational systems for real-time churn prediction. Establish monitoring mechanisms to track model performance over time and incorporate feedback loops for continuous improvement.

9. Business Impact Assessment: Quantify the business impact of implementing churn prediction models by evaluating the reduction in churn rate, increase in customer retention, and potential revenue gains. Assess the return on investment (ROI) of the project to justify resource allocation and support decision-making.

10. Continuous Improvement: Continuously refine and update the churn prediction models based on evolving customer behavior and market dynamics. Incorporate feedback from model performance monitoring and adapt strategies to enhance predictive accuracy and maintain relevance over time.

Importance of Customer Churn Prediction

1. Retention of Revenue: Customer churn directly impacts a company's revenue and profitability. By analyzing and predicting churn, businesses can proactively identify at-risk customers and implement targeted retention strategies to mitigate revenue loss.

2. Cost Reduction: Acquiring new customers is typically more expensive than retaining existing ones. By focusing efforts on retaining current customers through churn prediction, businesses can reduce acquisition costs and allocate resources more efficiently.

3. Customer Satisfaction and Loyalty: High churn rates can indicate dissatisfaction among customers. By identifying churn predictors and addressing underlying issues, businesses can improve customer satisfaction, foster loyalty, and strengthen brand reputation.

4. Competitive Advantage: Businesses that effectively analyze and predict customer churn gain a competitive edge by being proactive rather than reactive. Anticipating churn allows companies to tailor their offerings, customer service, and marketing strategies to retain customers and differentiate themselves in the market.

5. Data-Driven Decision Making: Churn analysis provides valuable insights into customer behavior and preferences. By leveraging data analytics and predictive modeling, businesses can make informed decisions based on empirical evidence rather than intuition or guesswork.

6. Resource Allocation: Understanding which customers are likely to churn enables businesses to allocate resources more effectively. By focusing retention efforts on high-value customers with a high likelihood of churn, companies can maximize the impact of their interventions and optimize resource utilization.

7. Long-Term Growth and Sustainability: Sustained customer retention is essential for long-term growth and sustainability. By reducing churn rates and increasing customer lifetime value, businesses can secure a stable revenue stream and position themselves for continued success in the marketplace.

8. Customer Insights and Segmentation: Churn analysis provides valuable insights into customer segmentation and behavior patterns. By segmenting customers based on churn risk and preferences, businesses can tailor marketing campaigns, product offerings, and customer experiences to better meet individual needs and preferences.

9. Predictive Maintenance: Identifying early warning signs of churn allows businesses to take proactive measures to prevent customer defection. By intervening before customers reach a critical churn threshold, companies can retain valuable customers and prevent revenue loss.

10. Continuous Improvement: Churn analysis is an ongoing process that enables continuous improvement and optimization of customer retention strategies. By monitoring churn trends, evaluating the effectiveness of interventions, and adapting strategies based on feedback, businesses can iteratively improve their retention efforts and stay ahead of evolving customer preferences and market dynamics.

Scope of the project:

Data Collection and Preparation:
Gathering relevant data sources including customer demographics, transaction history, and interactions.
Data cleaning to handle missing values, outliers, and inconsistencies.
Data transformation and feature engineering to create meaningful predictors for churn analysis.
Exploratory Data Analysis (EDA):
Exploring the dataset to understand the distribution and relationships between variables.
Identifying potential churn predictors through graphical analysis and statistical tests.
Visualizing trends and patterns that may indicate customer behavior leading to churn.
Feature Selection and Engineering:
Selecting the most relevant features using techniques like correlation analysis, feature importance, and domain knowledge.
Engineering new features that capture complex relationships and interactions between variables.
Dimensionality reduction techniques such as PCA to streamline the feature space while preserving predictive power.
Model Building:
Implementing a variety of machine learning algorithms including logistic regression, decision trees, random forest, XGBoost, and neural networks.
Tuning hyperparameters to optimize model performance and generalization.
Ensemble methods to combine multiple models for improved predictive accuracy and robustness.
Model Evaluation and Validation:
Assessing model performance using appropriate evaluation metrics such as accuracy, precision, recall, F1-score, and ROC-AUC.
Conducting cross-validation to ensure the models' robustness and generalizability.
Validating model predictions on unseen data to estimate real-world performance.
Interpretability and Actionable Insights:
Interpreting model predictions to understand the factors driving customer churn.
Generating actionable insights for business stakeholders to devise targeted retention strategies.
Communicating findings through visualizations, reports, and presentations to facilitate informed decision-making.
Deployment and Monitoring:
Integrating the predictive models into existing business systems for real-time churn prediction.
Monitoring model performance over time and recalibrating as needed to adapt to changing customer behavior.
Continuously refining the churn prediction framework based on feedback and evolving business requirements.
Ethical Considerations:
Ensuring data privacy and security throughout the project lifecycle.
Mitigating biases in the data and models to prevent discriminatory outcomes.
Transparency in model decisions and accountability for their implications on customer relationships.

Methodology

1. Data Acquisition and Preprocessing:

- Gather relevant data about customer interactions, transactions, demographics, and churn status.

Ensure data integrity and quality by validating sources and addressing any inconsistencies.. Preprocessing involves standardization, normalization, and feature extraction from the images. Outliers and missing values are addressed through appropriate techniques. Finally, the dataset is split into training and testing sets for machine learning model development.

2. Exploratory Data Analysis (EDA):

- Perform EDA to gain insights into the distribution and characteristics of the dataset.
- Conduct exploratory data analysis to gain insights into the dataset's characteristics and distributions.
- Utilize graphical representations such as histograms, box plots, and correlation matrices to visualize relationships between variables.
- Identify potential churn predictors and patterns indicative of customer behavior.

3. Feature Selection:

-Extract relevant features from the dataset that are likely to influence customer churn.

Utilize domain knowledge and statistical techniques to create new features or transform existing ones.

- Utilize techniques such as correlation analysis, feature importance, or domain knowledge to select the subset of features.

4. Model Training:

- Split the dataset into training and testing sets.
- Train various machine learning algorithms on the training data, including:
  - Logistic Regression
  - Decision Tree Classifier
  - Random Forest Classifier
  - XGBoost Classifier
  - K-Nearest Neighbors (KNN)
  - Support Vector Machine (SVM)
  - Naive Bayes Classifier
  - Ada Boost
  - Bagging Classifier

5. Model Evaluation:

- Evaluate the performance of each model using metrics such as accuracy, precision, recall, F1-score.
- Compare the performance of different algorithms to determine the most effective approach for asteroid classification.

6. Results Interpretation and Discussion:

- Interpret the results obtained from model evaluation and discuss the strengths and weaknesses of each algorithm.
- Analyze the factors contributing to the predictive performance and provide insights into the classification process.

7. Conclusion:

In conclusion, this abstract encapsulates the essence of a customer churn analysis and prediction project, elucidating its significance, methodologies, and implications. By unraveling the enigma of churn and empowering businesses with predictive insights, the project heralds a paradigm shift in customer retention strategies, heralding a future of sustained growth and competitiveness.

8. Report Writing:

- Compile the results, methodology, and discussions into a comprehensive report format.
- Include visualizations, tables, and figures to support the analysis and conclusions.

Below is a simplified flowchart representing the methodology:

```
Start
|
|__Data Acquisition and Preprocessing
|  |
|  |__Exploratory Data Analysis (EDA)
|  |
```

```
|   |___Feature Selection
|
|___Model Training
|  |
|   |___Split Data into Training and Testing Set

| |
|   |___Train Various Machine Learning Algorithms
|
|___Model Evaluation
|  |
|   |___Evaluate Performance Metrics
|  |
|   |___Compare Model Performance
|
|___Results Interpretation and Discussion
|  |
|   |___Analyze Model Results
|  |
|   |___Discuss Implications and Insights
|
|___Conclusion
|  |
|   |___Summarize Findings
|  |
|   |___Discuss Future Research Directions
|
|___Report Writing
    |
    |___Compile Results and Methodology into Report Format
    |
    |___Include Visualizations and Table
```

CODE

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import BaggingClassifier

from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score, roc_curve, confusion_matrix, classificatio
```

IMPORTING DATASETS

[210]:
```
train_data = pd.read_csv("train.csv")
test_data = pd.read_csv("test.csv")
train_data.head()
```

[210]:

	state	account_length	area_code	international_plan	voice_mail_plan	number_vmail_messages	total_day_minutes	total_day_calls	total_day_charge	total_eve_minu
0	OH	107	area_code_415	no	yes	26	161.6	123	27.47	19
1	NJ	137	area_code_415	no	no	0	243.4	114	41.38	12
2	OH	84	area_code_408	yes	no	0	299.4	71	50.90	6
3	OK	75	area_code_415	yes	no	0	166.7	113	28.34	14
4	MA	121	area_code_510	no	yes	24	218.2	88	37.09	34

[211]:
```
test_data.head()
```

[211]:

	id	state	account_length	area_code	international_plan	voice_mail_plan	number_vmail_messages	total_day_minutes	total_day_calls	total_day_charge	total_eve_n
0	1	KS	128	area_code_415	no	yes	25	265.1	110	45.07	
1	2	AL	118	area_code_510	yes	no	0	223.4	98	37.98	
2	3	IA	62	area_code_415	no	no	0	120.7	70	20.52	
3	4	VT	93	area_code_510	no	no	0	190.7	114	32.42	
4	5	NE	174	area_code_415	no	no	0	124.3	76	21.13	

[212]:
```
print(train_data.shape)
print(test_data.shape)
```

```
(4250, 20)
(750, 20)
```

[213]:
```
train_data.info()
```

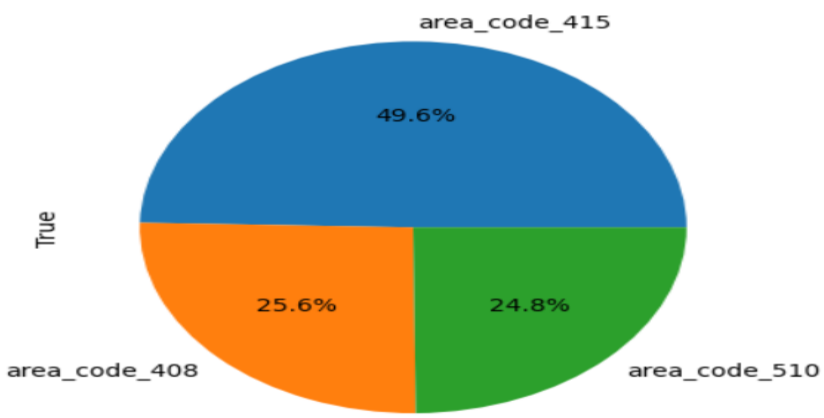
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4250 entries, 0 to 4249
Data columns (total 20 columns):
 #   Column                         Non-Null Count  Dtype
---  ------                         --------------  -----
 0   state                          4250 non-null   object
 1   account_length                 4250 non-null   int64
 2   area_code                      4250 non-null   object
 3   international_plan             4250 non-null   object
 4   voice_mail_plan                4250 non-null   object
 5   number_vmail_messages          4250 non-null   int64
 6   total_day_minutes              4250 non-null   float64
 7   total_day_calls                4250 non-null   int64
 8   total_day_charge               4250 non-null   float64
 9   total_eve_minutes              4250 non-null   float64
 10  total_eve_calls                4250 non-null   int64
 11  total_eve_charge               4250 non-null   float64
 12  total_night_minutes            4250 non-null   float64
 13  total_night_calls              4250 non-null   int64
 14  total_night_charge             4250 non-null   float64
 15  total_intl_minutes             4250 non-null   float64
 16  total_intl_calls               4250 non-null   int64
 17  total_intl_charge              4250 non-null   float64
 18  number_customer_service_calls  4250 non-null   int64
 19  churn                          4250 non-null   object
dtypes: float64(8), int64(7), object(5)
```

```python
train_data.describe()
```

	account_length	number_vmail_messages	total_day_minutes	total_day_calls	total_day_charge	total_eve_minutes	total_eve_calls	total_eve_charge	total_night_minu
count	4250.000000	4250.000000	4250.000000	4250.000000	4250.000000	4250.000000	4250.000000	4250.000000	4250.000
mean	100.236235	7.631765	180.259600	99.907294	30.644682	200.173906	100.176471	17.015012	200.527
std	39.698401	13.439882	54.012373	19.850817	9.182096	50.249518	19.908591	4.271212	50.353
min	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
25%	73.000000	0.000000	143.325000	87.000000	24.365000	165.925000	87.000000	14.102500	167.225
50%	100.000000	0.000000	180.450000	100.000000	30.680000	200.700000	100.000000	17.060000	200.450
75%	127.000000	16.000000	216.200000	113.000000	36.750000	233.775000	114.000000	19.867500	234.700
max	243.000000	52.000000	351.500000	165.000000	59.760000	359.300000	170.000000	30.540000	395.000

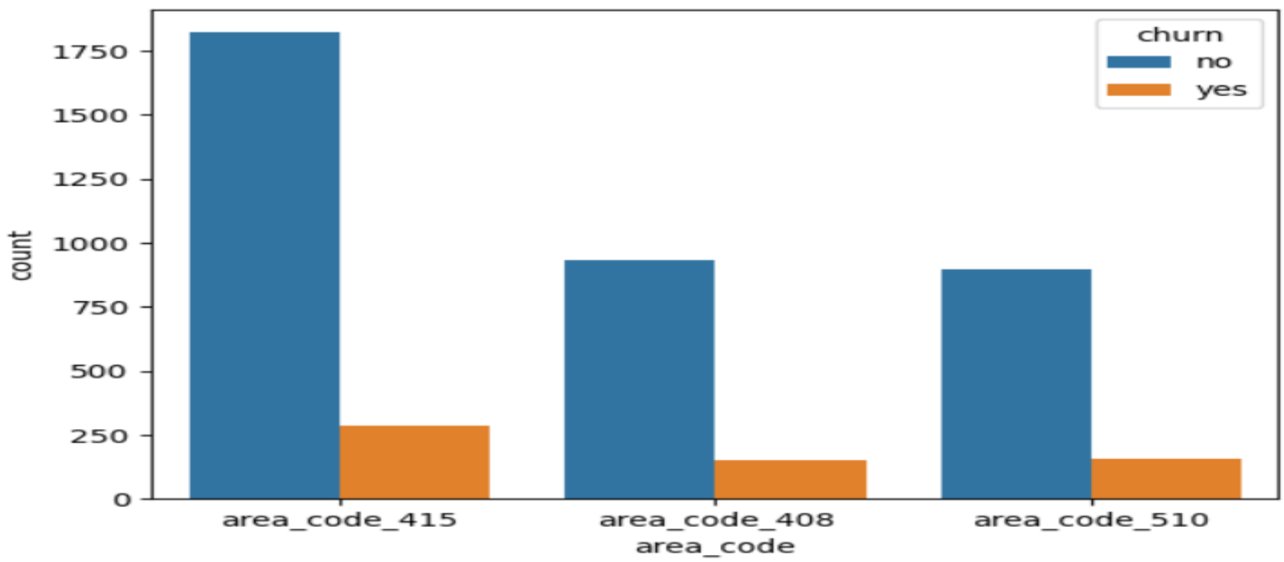
```python
train_data["area_code"].value_counts().plot(kind='pie', label=True, autopct = '%.1f%%')
```

<Axes: ylabel='True'>




```python
sns.countplot(data = train_data, x = 'area_code', hue = 'churn')
```

<Axes: xlabel='area_code', ylabel='count'>



CLEANING THE DATA

[10]:
```
train_data.isnull().sum()
```

[10]:
```
state                            0
account_length                   0
area_code                        0
international_plan               0
voice_mail_plan                  0
number_vmail_messages            0
total_day_minutes                0
total_day_calls                  0
total_day_charge                 0
total_eve_minutes                0
total_eve_calls                  0
total_eve_charge                 0
total_night_minutes              0
total_night_calls                0
total_night_charge               0
total_intl_minutes               0
total_intl_calls                 0
total_intl_charge                0
number_customer_service_calls    0
churn                            0
dtype: int64
```

[11]:
```
train_data.duplicated().sum()
```

[11]:
```
0
```

PRE-PROCESSING

[12]:
```
cat_cols = ['state','area_code','international_plan','voice_mail_plan','churn']

train_data[cat_cols] = train_data[cat_cols].astype('category')
test_data[cat_cols[:-1]] = test_data[cat_cols[:-1]].astype('category')
```

1. Calculate the total_net_minutes to reduce the number of features; we are going to do the same with calls, and charge
2. Convert all yes, no strings into ints such as in columns (voice_mail_plan, international_plan, and churn)
3. Convert the categorical values into onehote vectors such as (state, and area_code)
4. Drop all repeted features and useless columns such as area (code and state)

[13]:
```
def clean_data(df):
  df['total_net_minutes'] = df["total_day_minutes"] + df["total_eve_minutes"] + df["total_night_minutes"]
  df["total_net_calls"]   = df["total_day_calls"] + df["total_eve_calls"] + df["total_night_calls"]
  df["total_net_charge"] = df["total_day_charge"] + df["total_eve_charge"] + df["total_night_charge"]

  df['voice_mail_plan'] = df['voice_mail_plan'].map({'yes': 1, 'no': 0})
  df['international_plan'] = df['international_plan'].map({'yes': 1, 'no': 0})

  df.drop(columns = ["state", "area_code",'total_day_charge', 'total_eve_charge','total_night_charge',
                    'total_day_calls','total_eve_calls', 'total_night_calls', 'total_day_minutes',
                     'total_eve_minutes', 'total_night_minutes'], inplace=True)
  return df
clean_data(train_data)
clean_data(test_data)
```

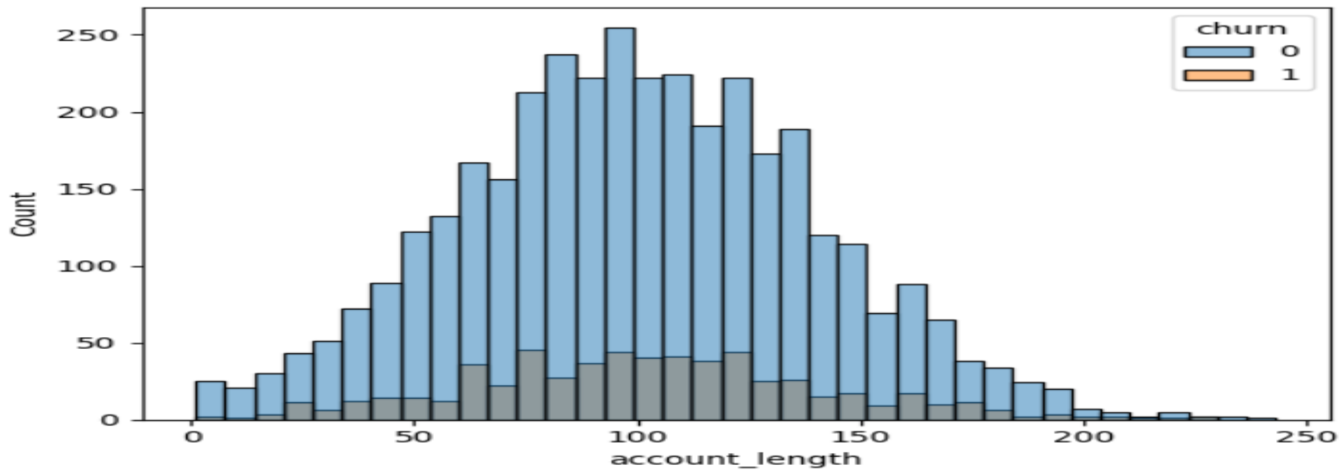
```
# convert "CHURN" column from categorical into numerical
train_data['churn'] = train_data['churn'].map({"yes": 1, "no": 0})
```

```
train_data
```

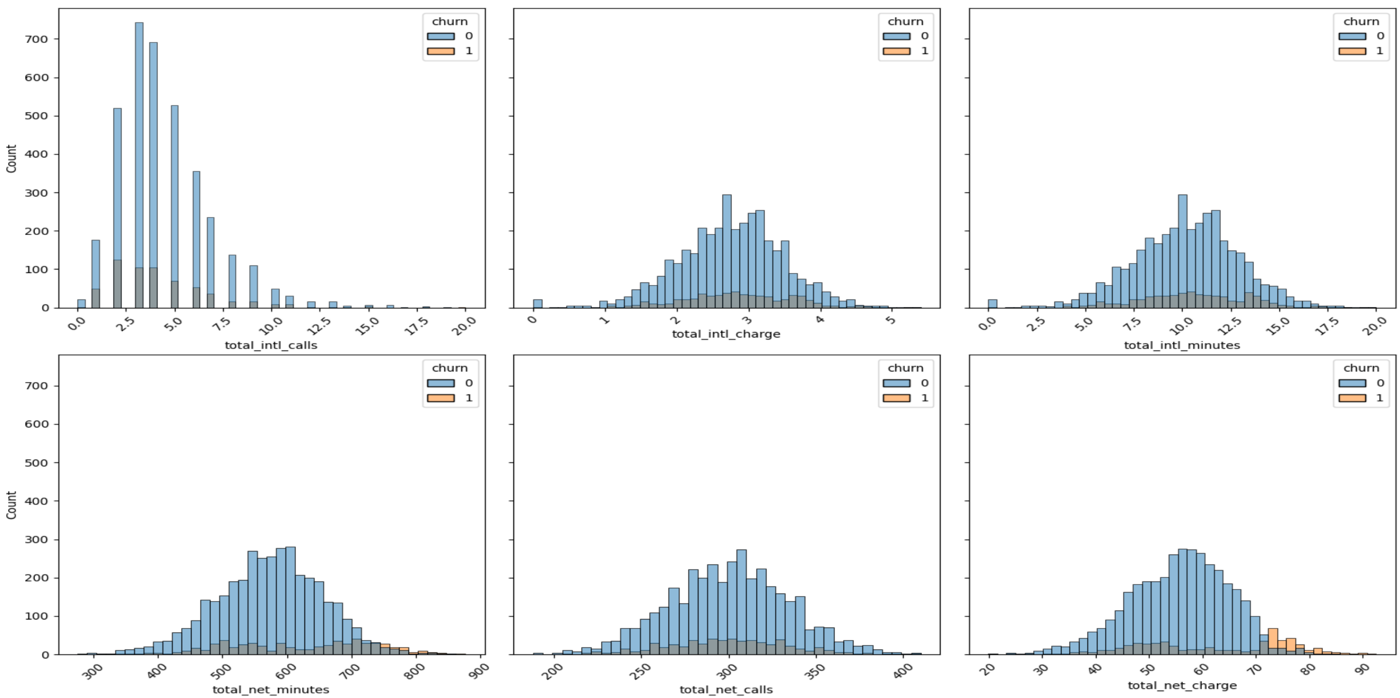
voice_mail_plan	number_vmail_messages	total_intl_minutes	total_intl_calls	total_intl_charge	number_customer_service_calls	churn	total_net_minutes	total_net_calls	tot
1	26	13.7	3	3.70	1	0	611.5	329	
0	0	12.2	5	3.29	0	0	527.2	328	
0	0	6.6	7	1.78	2	0	558.2	248	
0	0	10.1	3	2.73	3	0	501.9	356	
1	24	7.5	7	2.03	3	0	779.3	314	
...	...	...	...	...	...	...	...	...	
0	0	10.3	6	2.78	0	0	645.8	237	
0	0	11.5	6	3.11	3	0	495.3	260	
0	0	6.9	7	1.86	1	0	492.9	331	
1	40	9.9	5	2.67	2	0	756.2	369	
1	34	9.3	16	2.51	0	0	551.3	306	

```
sns.histplot(x=train_data['account_length'],hue = train_data['churn'])
```

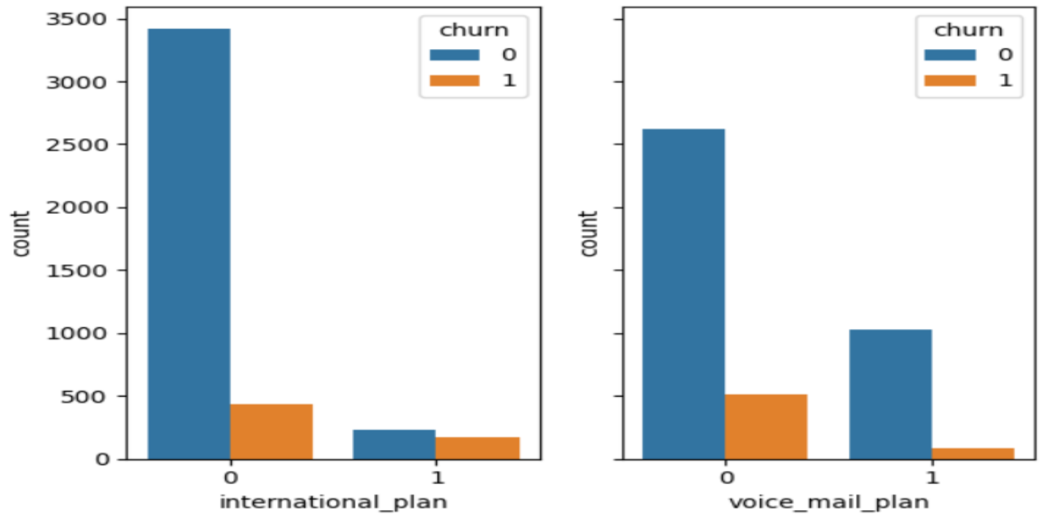
```
<Axes: xlabel='account_length', ylabel='Count'>
```




```
fig, axes = plt.subplots(nrows=2, ncols=3, figsize=(15, 10), sharey=True)

# Plotting
sns.histplot(data=train_data, x='total_intl_calls', hue='churn', ax=axes[0, 0])
sns.histplot(data=train_data, x='total_intl_charge', hue='churn', ax=axes[0, 1], color='orange')
sns.histplot(data=train_data, x='total_intl_minutes', hue='churn', ax=axes[0, 2], color='red')
sns.histplot(data=train_data, x='total_net_minutes', hue='churn', ax=axes[1, 0], color='red')
sns.histplot(data=train_data, x='total_net_calls', hue='churn', ax=axes[1, 1], color='green')
sns.histplot(data=train_data, x='total_net_charge', hue='churn', ax=axes[1, 2], color='blue')

for ax in axes.flat:
    ax.tick_params(axis='x', labelrotation=45)

plt.tight_layout()  # Adjust spacing between subplots
plt.show()
```

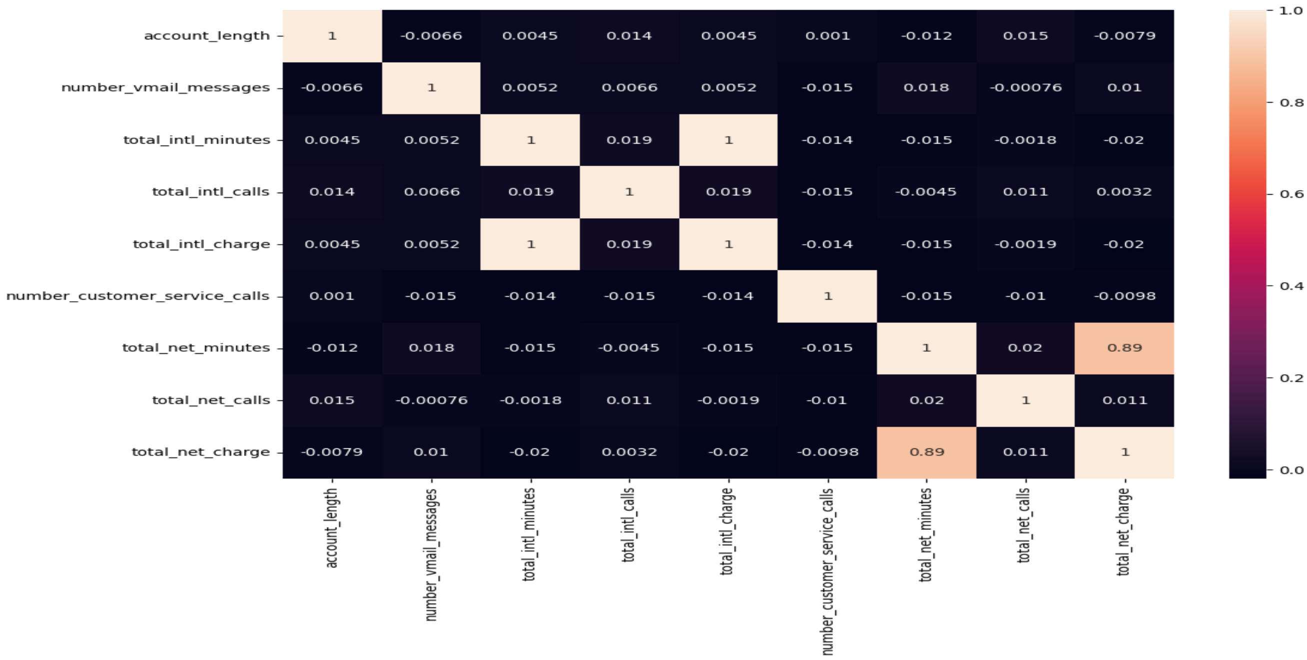


```python
fig, axes = plt.subplots(nrows=1, ncols=2, sharey=True)

sns.countplot(data=train_data, x="international_plan", hue="churn", ax=axes[0])
sns.countplot(data=train_data, x="voice_mail_plan", hue="churn", ax=axes[1]);
```



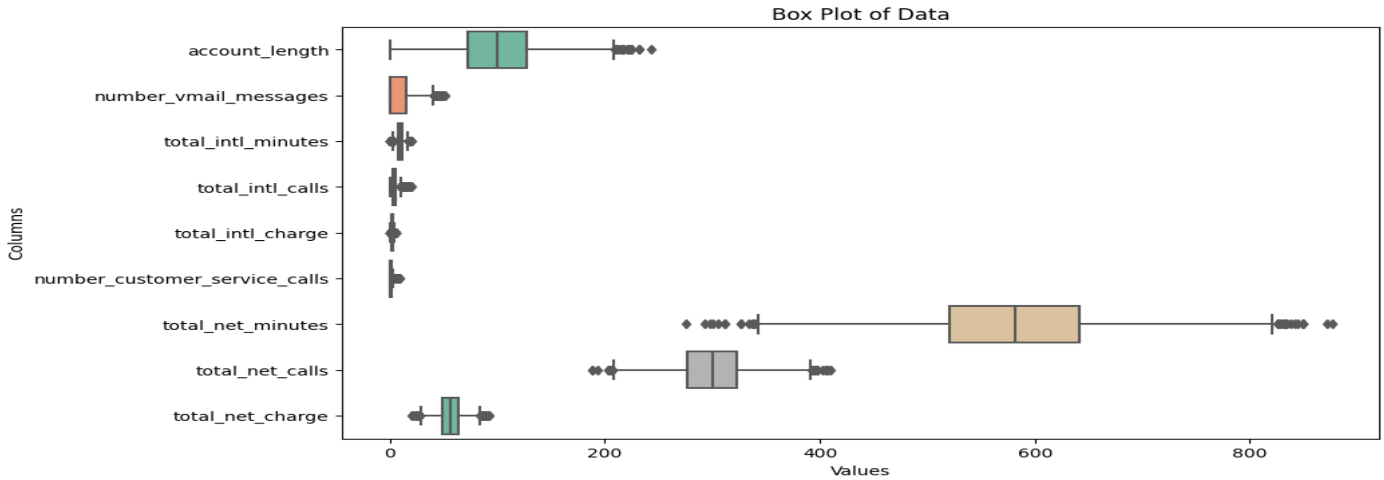
CORRELATION BETWEEN NUMERICAL COLUMS

```python
num_cols = train_data.select_dtypes(include = ["float", "int"]).columns
corr_data = train_data[num_cols].corr()
plt.figure(figsize = [12, 8])
sns.heatmap(corr_data, annot = True)
plt.show()
```



DEALING WITH OUTLIERS

[21]:
```python
plt.figure(figsize=(10, 6))
sns.boxplot(data=train_data, orient="h", palette="Set2")
plt.title("Box Plot of Data")
plt.xlabel("Values")
plt.ylabel("Columns")
plt.show()
```




```python
def replace_outliers_with_median(series):
    # Calculate quartiles and IQR
    Q1 = series.quantile(0.25)
    Q3 = series.quantile(0.75)
    IQR = Q3 - Q1

    # Calculate bounds for outliers
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Replace outliers with median
    series = series.mask((series < lower_bound) | (series > upper_bound), series.median())
    return series

# Apply outlier replacement to each column
for col in num_cols:
    train_data[col] = replace_outliers_with_median(train_data[col])

train_data
```

```python
train_data[num_cols] = train_data[num_cols].fillna(train_data[num_cols].median())
train_data
```

	account_length	international_plan	voice_mail_plan	number_vmail_messages	total_intl_minutes	total_intl_calls	total_intl_charge	number_customer_service_calls	ch
0	107	0	1	26	13.7	3	3.70	1	
1	137	0	0	0	12.2	5	3.29	0	
2	84	1	0	0	6.6	7	1.78	2	
3	75	1	0	0	10.1	3	2.73	3	
4	121	0	1	24	7.5	7	2.03	3	
...	...	...	...	...	...	...	...	...	
4245	83	0	0	0	10.3	6	2.78	0	
4246	73	0	0	0	11.5	6	3.11	3	
4247	75	0	0	0	6.9	7	1.86	1	
4248	50	0	1	40	9.9	5	2.67	2	
4249	86	0	1	34	9.3	4	2.51	0	

4250 rows × 12 columns

SCALLING THE TRAIN DATA

```python
num_cols = train_data.select_dtypes(include = ["float", "int"]).columns
cat_cols = ["voice_mail_plan", "international_plan"]

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

scaled_cols = scaler.fit_transform(train_data[num_cols])
train_scaled = pd.DataFrame(scaled_cols, columns = num_cols )
train_scaled[cat_cols] = train_data[cat_cols]
train_scaled.head()
```

	account_length	number_vmail_messages	total_intl_minutes	total_intl_calls	total_intl_charge	number_customer_service_calls	total_net_minutes	total_net_calls	total_ne
0	0.188764	1.550048	1.340576	-0.591544	1.341366	-0.295258	0.355208	0.867180	-
1	0.961465	-0.542948	0.749910	0.380074	0.743382	-1.361399	-0.623766	0.837428	
2	-0.403641	-0.542948	-1.455244	1.351692	-1.458952	0.770882	-0.263764	-1.542703	
3	-0.635452	-0.542948	-0.077023	-0.591544	-0.073378	1.837023	-0.917575	1.670474	-
4	0.549358	1.389048	-1.100845	1.351692	-1.094327	1.837023	2.303867	0.420905	

SCALLING THE TEST DATA

```
num_cols = test_data.select_dtypes(include = ["float", "int"]).columns
cat_cols = ["voice_mail_plan", "international_plan"]

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

scaled_cols = scaler.fit_transform(test_data[num_cols])
test_scaled = pd.DataFrame(scaled_cols, columns = num_cols )
test_scaled[cat_cols] = test_data[cat_cols]
test_scaled.head()
```

	id	account_length	number_vmail_messages	total_intl_minutes	total_intl_calls	total_intl_charge	number_customer_service_calls	total_net_minutes	total_net_cal
0	-1.729743	0.696065	1.172240	-0.106243	-0.613702	-0.106976	-0.497639	1.369617	-0.0413
1	-1.725124	0.444001	-0.599015	-1.442710	0.625821	-1.445297	-1.281734	0.713930	0.4730
2	-1.720505	-0.967554	-0.599015	1.013500	0.625821	1.017213	1.854646	0.525958	-1.7054
3	-1.715887	-0.186157	-0.599015	-0.792537	-0.613702	-0.789520	1.070551	-0.495721	1.3505
4	-1.711268	1.855557	-0.599015	1.880397	0.212647	1.887122	1.070551	0.760369	0.0494

SPLITING THE DATA

```
x= train_scaled
x[cat_cols] = x[cat_cols].astype(int)
y = pd.Series(train_data['churn'])
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3,stratify = y, random_state=42)
x_train.shape, x_test.shape, y_train.shape, y_test.shape
```

```
((2975, 11), (1275, 11), (2975,), (1275,))
```

BUILDING MACHINE LEARNING MODELS

LOGISTIC REGRESSION

```
lo_model = LogisticRegression()
lo_model.fit(x_train, y_train)
y_predict = lo_model.predict(x_test)
```

```
lo_accuracy = accuracy_score(y_predict, y_test)
lo_accuracy
```

```
0.8564705882352941
```

```
print(classification_report(y_test, y_predict))
```

```
              precision    recall  f1-score   support

           0       0.87      0.97      0.92      1096
           1       0.46      0.14      0.21       179

    accuracy                           0.86      1275
   macro avg       0.67      0.56      0.57      1275
weighted avg       0.82      0.86      0.82      1275
```

Naive bayes

```
nb_model = GaussianNB()
nb_model.fit(x_train, y_train)
y_predict = nb_model.predict(x_test)
```

```
nb_acc = accuracy_score(y_predict, y_test)
nb_acc
```

```
0.8627450980392157
```

```
print(classification_report(y_test, y_predict))
```

```
              precision    recall  f1-score   support

           0       0.92      0.92      0.92      1096
           1       0.51      0.50      0.50       179

    accuracy                           0.86      1275
   macro avg       0.71      0.71      0.71      1275
weighted avg       0.86      0.86      0.86      1275
```

SUPPORT VECTOR MACHINE

```
svm = SVC()
svm.fit(x_train, y_train)
y_preidct = svm.predict(x_test)
```

```
svm_acc = accuracy_score(y_predict, y_test)
svm_acc
```

```
0.8627450980392157
```

```
print(classification_report(y_test, y_predict))
```

```
              precision    recall  f1-score   support

           0       0.92      0.92      0.92      1096
           1       0.51      0.50      0.50       179

    accuracy                           0.86      1275
   macro avg       0.71      0.71      0.71      1275
weighted avg       0.86      0.86      0.86      1275
```

DECISION TREE CLASSIFIER

```
from sklearn.tree import DecisionTreeClassifier
dt_model = DecisionTreeClassifier()
dt_model.fit(x_train, y_train)
dt_model_predict = dt_model.predict(x_test)
```

```
accuracy = accuracy_score(dt_model_predict, y_test)
accuracy
```

```
0.9027450980392157
```

```
print(classification_report(y_test, dt_model_predict))
```

```
              precision    recall  f1-score   support

           0       0.95      0.94      0.94      1096
           1       0.65      0.67      0.66       179

    accuracy                           0.90      1275
   macro avg       0.80      0.81      0.80      1275
weighted avg       0.90      0.90      0.90      1275
```

RANDOM FOREST CLASSIFIER

```
from sklearn.ensemble import RandomForestClassifier
rf_model = RandomForestClassifier()
rf_model.fit(x_train, y_train)
rf_model_predict = rf_model.predict(x_test)
```

```
rf_accuracy = accuracy_score(rf_model_predict, y_test)
rf_accuracy
```

```
0.9427450980392157
```

```
print(classification_report(y_test, rf_model_predict))
```

```
              precision    recall  f1-score   support

           0       0.94      1.00      0.97      1096
           1       0.96      0.61      0.75       179

    accuracy                           0.94      1275
   macro avg       0.95      0.81      0.86      1275
weighted avg       0.94      0.94      0.94      1275
```

XGBOOST CLASSIFIER

```
xg_model = XGBClassifier()
xg_model.fit(x_train, y_train)
xg_model_predict = xg_model.predict(x_test)
```

```
xg_accuracy = accuracy_score(xg_model_predict, y_test)
xg_accuracy
```

```
0.9380392156862745
```

```
print(classification_report(y_test, xg_model_predict))
```

```
              precision    recall  f1-score   support

           0       0.94      0.99      0.96      1096
           1       0.89      0.64      0.74       179

    accuracy                           0.94      1275
   macro avg       0.92      0.81      0.85      1275
weighted avg       0.94      0.94      0.93      1275
```

K-NEAREST NEIGHBORS

```
knn = KNeighborsClassifier(n_neighbors=5)

knn.fit(x_train, y_train)

y_pred = knn.predict(x_test)

accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.8854901960784314
              precision    recall  f1-score   support

           0       0.90      0.98      0.94      1096
           1       0.72      0.30      0.43       179

    accuracy                           0.89      1275
   macro avg       0.81      0.64      0.68      1275
weighted avg       0.87      0.89      0.86      1275
```

ADA BOOST

```
adaboost = AdaBoostClassifier(n_estimators=40, random_state=42)

adaboost.fit(x_train, y_train)

y_pred = adaboost.predict(x_test)

accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.9090196078431373
              precision    recall  f1-score   support

           0       0.92      0.97      0.95      1096
           1       0.76      0.51      0.61       179

    accuracy                           0.91      1275
   macro avg       0.84      0.74      0.78      1275
weighted avg       0.90      0.91      0.90      1275
```

GRADIENT BOOSTING

```
gradient_boosting = GradientBoostingClassifier(n_estimators=100, random_state=42)

gradient_boosting.fit(x_train, y_train)

y_pred = gradient_boosting.predict(x_test)

accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.9427450980392157
              precision    recall  f1-score   support

           0       0.94      0.99      0.97      1096
           1       0.95      0.63      0.75       179

    accuracy                           0.94      1275
   macro avg       0.95      0.81      0.86      1275
weighted avg       0.94      0.94      0.94      1275
```

BAGGING CLASSIFIER

```
base_classifier = DecisionTreeClassifier(random_state=42)

# Initialize Bagging classifier
bagging_classifier = BaggingClassifier(estimator=base_classifier, n_estimators=10, random_state=42)

# Train the classifier
bagging_classifier.fit(x_train, y_train)

# Predict on the testing data
y_pred = bagging_classifier.predict(x_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.9333333333333333
              precision    recall  f1-score   support

           0       0.94      0.98      0.96      1096
           1       0.86      0.63      0.73       179

    accuracy                           0.93      1275
   macro avg       0.90      0.81      0.84      1275
weighted avg       0.93      0.93      0.93      1275
```

PREDICTION CHURN FOR THE TEST DATA

```
test_data.head()
```

	id	account_length	international_plan	voice_mail_plan	number_vmail_messages	total_intl_minutes	total_intl_calls	total_intl_charge	number_customer_service_calls	tc
0	1	128	0	1	25	10.0	3	2.70	1	
1	2	118	1	0	0	6.3	6	1.70	0	
2	3	62	0	0	0	13.1	6	3.54	4	
3	4	93	0	0	0	8.1	3	2.19	3	
4	5	174	0	0	0	15.5	5	4.19	3	

```
final_test = test_scaled[:5].drop(columns = 'id')
final_test
```

	account_length	number_vmail_messages	total_intl_minutes	total_intl_calls	total_intl_charge	number_customer_service_calls	total_net_minutes	total_net_calls	total_ne
0	0.696065	1.172240	-0.106243	-0.613702	-0.106976	-0.497639	1.369617	-0.041312	
1	0.444001	-0.599015	-1.442710	0.625821	-1.445297	-1.281734	0.713930	0.473065	
2	-0.967554	-0.599015	1.013500	0.625821	1.017213	1.854646	0.525958	-1.705472	-
3	-0.186157	-0.599015	-0.792537	-0.613702	-0.789520	1.070551	-0.495721	1.350532	-
4	1.855557	-0.599015	1.880397	0.212647	1.887122	1.070551	0.760369	0.049461	-

PREDICTING CHURN FOR TEST DATA WITH GRADIENT BOOSTING CLASSIFIER

```
gbt_pred = gradient_boosting.predict(final_test)
gbt_pred
```

```
array([0, 0, 0, 0, 0], dtype=int64)
```

```
sample_data = pd.read_csv('sampleSubmission.csv')
data = sample_data[:5].copy()
data.reset_index(drop=True, inplace = True)
data.churn = gbt_pred
data.churn = data.churn.map({1:'yes',0:'no'})
data
```

	id	churn
0	1	no
1	2	no
2	3	no
3	4	no
4	5	no

PREDICTING CHURN FOR TEST DATA WITH RANDOM FOREST CLASSIFIER

```
rf_pred = rf_model.predict(final_test)
rf_pred
```

```
array([0, 0, 0, 0, 0], dtype=int64)
```

```
data = sample_data[:5].copy()
data.reset_index(drop=True, inplace = True)
data.churn = rf_pred
data.churn = data.churn.map({1:'yes',0:'no'})
data
```

	id	churn
0	1	no
1	2	no
2	3	no
3	4	no
4	5	no

RESULT

In conducting customer churn analysis and prediction, the first step involves thorough data cleaning to ensure the dataset is accurate and reliable. This process includes handling missing values, correcting inconsistencies, and addressing any anomalies. Outliers, which can significantly impact model performance, are identified and imputed with robust measures such as the median to minimize their influence on subsequent analysis.

Following data cleaning, feature extraction is performed to derive meaningful insights from the available data. This step involves selecting or creating relevant features that may influence customer churn, such as demographic information, usage patterns, and engagement metrics.

With the preprocessed data, various machine learning models are built to predict customer churn. A range of algorithms is employed, including logistic regression, naive Bayes, decision trees, random forest, AdaBoost, gradient descent, k-nearest neighbors, XGBoost, bagging classifier, and support vector machine (SVM). Each algorithm is trained and evaluated to determine its predictive performance.

After rigorous evaluation, it is found that random forest and gradient boosting algorithms exhibit the highest accuracy, achieving an impressive 94.2%. These models demonstrate superior predictive power in identifying potential churners among the customer base. Therefore, they are selected as the final models for deploying in real-world scenarios to proactively manage customer retention efforts and mitigate churn risks.

ALGORITHM	ACCURACY
Logistic Regression	85.6
Naive Bayes	86.2
Support Vector Machine	86.2
Decision Tree	90.2
Random Forest	94.2
xgboost	93.5
Ada Boost	90.9
Gradient Boosting	94.2
K-Nearest Neighbors	88.8
Bagging Classifier	93.3

INSIGHTS TO REDUCE CUSTOMER CURN:

Reducing customer churn in the telecom industry involves understanding the reasons why customers leave and implementing strategies to address those reasons. Here are some insights and strategies to help reduce churn:

1. Improve Customer Service: Providing excellent customer service can significantly reduce churn. Ensure that customers can easily reach customer support through multiple channels (phone, email, chat) and that their issues are resolved quickly and effectively.

2. Personalized Offers and Discounts: Analyze customer data to understand preferences and usage patterns. Offer personalized deals, discounts, or upgrades to incentivize customers to stay.

3. Enhance Network Quality: Invest in improving network coverage, speed, and reliability. Dissatisfaction with network performance is a common reason for churn, so ensuring a high-quality network experience can help retain customers.

4. Monitor Usage Patterns: Track customer usage patterns and proactively reach out to customers who show signs of disengagement or dissatisfaction. Offer solutions tailored to their needs to encourage continued usage.

5. Loyalty Programs: Implement loyalty programs that reward customers for their continued patronage. Offer perks such as discounts on additional services, priority customer support, or exclusive access to content.

6. Regular Communication: Keep customers engaged through regular communication. Update them on new features, promotions, and improvements to the service. This helps customers feel valued and connected to the brand.

7. Simplify Billing and Payment Processes: Complicated billing processes or unexpected fees can frustrate customers and drive them to switch providers. Simplify billing statements and offer flexible payment options to improve customer satisfaction.

CONCLUSION

In conclusion, the customer churn analysis and prediction process involved several crucial steps, starting with data cleaning to ensure the dataset's integrity and accuracy. This phase addressed issues such as missing values, inconsistencies, and errors to establish a reliable foundation for analysis. Furthermore, outliers were identified and treated by imputing them with the median value to prevent them from unduly influencing the results.

Following data cleaning, feature extraction was undertaken to identify relevant predictors that could effectively characterize customer churn behavior. This involved selecting and transforming features to capture meaningful insights from the data. Subsequently, various machine learning models were constructed and evaluated for their predictive performance. These included logistic regression, naive Bayes, decision trees, random forest, AdaBoost, gradient descent, k-nearest neighbors, XGBoost, bagging classifier, and support vector machine algorithms.

Upon thorough evaluation, it was determined that random forest and gradient boosting algorithms exhibited superior performance, achieving an accuracy of 94.2%. These models demonstrated robustness in handling the complexities of the dataset and effectively capturing the underlying patterns associated with customer churn. As such, they represent the most suitable choices for predicting and addressing churn within the studied context. This outcome underscores the importance of systematically exploring various modeling approaches to identify the most effective solutions for real-world business challenges like customer churn prediction.

REFERENCES

[1] Chih Fong Tsai explores customer churn prediction through hybrid neural networks in "Expert Systems with Applications."

[2] Wouuter Verbeke and Bart Baesens focus on constructing intelligible customer churn prediction models using advanced rule induction techniques, also in "Expert Systems with Applications."

[3] Ning Lu, Hua Lin, and Guangquan Zhang present a customer churn prediction model for the telecom industry using boosting techniques in "IEEE Transactions on Industrial Informatics."

[4] H. Karamollaoğlu, İ. Yücedağ and İ. A. Doğru conducted a comparative analysis of machine learning methods for customer churn prediction in the "6th International Conference on Computer Science and Engineering."

[5] R.V.S. Rohit, D. Chandrawat, and D. Rajeswari explore smart farming techniques for new farmers using machine learning, as presented in the proceedings of the "6th International Conference on Recent Trends in Computing."

[6] [8] Ssu-Han Chen discusses the gamma CUSUM chart method for online customer churn prediction in "Electronic Commerce Research and Applications."

[7] Koen W. De Bock and Dirk Van den Poel evaluate rotation-based ensemble classifiers for customer churn prediction in "Expert Systems with Applications."