Naan mudhalvan project

Customer churn prediction – phase 5

Akshaya K 2021115011

Akshaya S M 2021115012

Preetham V 2021115077

Praveen M-2021115075

Akash M 2021115324

**PROBLEM STATEMENT:**

The project involves using IBM Cognos to predict customer churn and identify factors influencing customer retention. The goal is to help businesses reduce customer attrition by understanding the patterns and reasons behind customers leaving. This project includes defining analysis objectives, collecting customer data, designing relevant visualizations in IBM Cognos, and building a predictive model.

**DESIGN THINKING:**

**1.) Analysis Objectives**:

Using IBM Cognos, the specific objectives for predicting customer churn are:

Churn Prediction Model: Develop a predictive model within IBM Cognos to identify customers at risk of churning.

Churn Risk Score: Assign a churn risk score to each customer, facilitating targeted retention efforts.

Segmentation: Segment customers based on their churn risk levels for customized retention strategies.

Key Churn Drivers: Identify and visualize the primary factors contributing to customer churn using IBM Cognos analytics.

Retention Recommendations: Generate actionable insights and recommendations for retention strategies within the IBM Cognos environment.

Model Validation: Evaluate and validate the performance of the churn prediction model using IBM Cognos tools and visualizations.

**2.) Data Collection:**

We will be using the dataset provided by kaggle.com to carry on this project

[**https://www.kaggle.com/datasets/blastchar/telco-customer-churn**](https://www.kaggle.com/datasets/blastchar/telco-customer-churn)

The above dataset contains necessary data like day, date etc. It also contains number of unique visits, first visits and returning visits which will be very helpful for us to enhance the user experience by identifying what they need the most.

**3.) Visualization Strategy:**

Visualization is a powerful tool in understanding customer churn data and conveying insights effectively. Here are some visualization strategies to consider when working with customer churn data:

Churn Rate Trends:

Line charts or time series plots can show how churn rates have evolved over time.

Compare churn rates among different customer segments using stacked area charts or grouped bar charts.

Customer Segmentation:

Create pie charts, bar charts, or treemaps to visually represent customer segments based on churn risk levels or demographics.

Use heatmaps to display correlation between different customer attributes and churn.

Customer Journey Mapping:

Flowcharts or Sankey diagrams can illustrate the customer journey, highlighting touchpoints where churn is more likely to occur.

**4.) Predictive modeling**

Predictive modeling is a process used in data science and machine learning to develop and train algorithms that can make predictions or classifications based on data. In the context of customer churn prediction, predictive modeling involves building a model that can forecast whether a customer is likely to churn (leave) or stay with a product or service in the future. Here are the key steps involved in predictive modeling for customer churn prediction:

**Data Preparation:**

Collect and clean the customer data, ensuring it's of high quality and ready for analysis.

Split the data into training and testing sets to evaluate model performance.

**Feature Selection and Engineering:**

Identify relevant features (variables) that could influence customer churn, such as demographics, usage patterns, or customer interactions.

Create new features or transform existing ones to enhance the predictive power of the model.

**Model Selection:**

Choose an appropriate machine learning algorithm for the task. Common choices include logistic regression, decision trees, random forests, support vector machines, or gradient boosting methods.

Experiment with multiple models to find the one that performs best for your dataset.

**Model Training:**

Use the training data to train the selected model. The model learns from historical data patterns to make predictions about future churn.

**Model Evaluation:**

Assess the model's performance using evaluation metrics such as accuracy, precision, recall, F1-score, ROC AUC, and others.

Validate the model on the testing dataset to ensure it generalizes well to new, unseen data.

**Hyperparameter Tuning:**

Fine-tune the model's hyperparameters to optimize its performance. Techniques like cross-validation can be employed to find the best hyperparameters.

**Deployment:**

Implement the trained model into your operational systems, allowing it to make real-time predictions on new customer data.

Monitoring and Maintenance:

Continuously monitor the model's performance in the production environment and retrain it as necessary to adapt to changing customer behaviors.

**Interpretation and Actionable Insights:**

Understand the model's predictions and the factors that contribute to churn, enabling the development of targeted retention strategies.

**INTRODUCTION:**

In the previous phase, we clearly established our end goals, and we also explained what insights we intend to derive from this project.

In this phase, we will introduce cutting-edge techniques and tools to advance our comprehension of WEBSITE TRAFFIC ANALYSIS. The objective of this phase is to harness the power of machine learning algorithms to forecast service interruptions and extract insights from passenger feedback, ultimately enhancing the overall service quality and user experience.

Below is an in-depth description of the design and algorithms applied in this project.

**DATA INTEGRATION:**

Data integration is the process of collecting, transforming, and unifying data from various sources, such as web analytics tools, databases, and server logs, to create a coherent and standardized dataset for analysis within IBM Cognos. It involves identifying sources, collecting data, standardizing formats, mapping relationships, and loading data securely while ensuring data quality and synchronization. Effective data integration is fundamental for accurate and reliable insights in a website traffic analysis project, enabling seamless decision-making based on a consolidated and harmonized dataset.

**DATA PREPARATION:  
 Data Preparation** is a pivotal step in a website traffic analysis project using IBM Cognos. In this phase, raw data collected from various sources, including web analytics tools, server logs, and databases, undergoes rigorous cleaning, transformation, and structuring. The primary aim is to ensure data quality and consistency. This involves identifying and rectifying data inconsistencies, handling missing values, and standardizing data formats. Furthermore, data preparation may involve filtering out irrelevant or erroneous information to create a clean dataset that is ready for analysis. This process lays the foundation for accurate insights and informed decision-making, as it ensures that the data is reliable and free from anomalies.

**DATA MODELLING:** Data Modeling is the subsequent step in the analysis project, where the prepared data is organized into a structured framework within IBM Cognos Framework Manager. A well-designed data model defines the relationships between tables, establishes key metrics, and provides a foundation for efficient querying and reporting. For website traffic analysis, it's crucial to create a time-based dimension, allowing for trend analysis over specific periods. This data model serves as the backbone for generating reports and dashboards that deliver meaningful insights. Effective data modeling simplifies the complex task of data retrieval and presentation, ensuring that users can efficiently access and understand the relevant metrics and KPIs for website traffic analysis.

**DESIGNING REPORT :**  Designing reports is a pivotal phase in this project using IBM Cognos, where data insights are transformed into visually engaging and informative presentations. During this process, you select appropriate visualizations, such as tables, charts, and graphs, to represent key performance indicators (KPIs) and metrics. Effective report design is characterized by clarity, relevance, and user-friendliness, ensuring that stakeholders can quickly grasp the most crucial insights. Additionally, thoughtful layout and intuitive navigation enhance the user experience, making it easier to explore and understand the data. By employing IBM Cognos tools like Cognos Analytics or Report Studio, you can tailor reports to meet specific objectives, providing a dynamic and interactive platform for data-driven decision-making and continuous improvement in the context of website traffic analysis.

**ANALYSIS :**  In the analysis phase, data is scrutinized to uncover patterns and insights, revealing user behavior and performance metrics. These insights lead to concrete actions, which could involve content optimization, marketing strategy adjustments, or resolving identified issues. This iterative process ensures the website evolves in alignment with organizational goals and user satisfaction.

**MACHINE LEARNING MODELS :  
   
Time Series Forecasting Models:**

* ARIMA (AutoRegressive Integrated Moving Average): ARIMA models are suitable for predicting time series data, making them ideal for forecasting future website traffic patterns, such as page views or user sessions.
* Exponential Smoothing: Exponential smoothing methods, like Holt-Winters, are effective for capturing seasonality and trends in time series data.

**Regression Models:**

* Linear Regression: Linear regression can predict numerical outcomes, making it useful for estimating future traffic metrics or user engagement levels.
* Random Forest Regression: Random forest models can handle complex relationships and provide more accurate predictions, especially when dealing with non-linear data**.**

**CONCLUSION :**

Here, we have discussed the project design that we will follow and the machine learning models we will utilize in this project

**Naan Mudhalvan Project**

**customer churn prediction**

**Team Members:**

**~Akshaya K- 2021115011**

**~Akshaya SM - 2021115012**

**~ Preetham V -2021115077**

**~Praveen M - 2021115075**

**~ Akash M - 202111532419/10/2023, 04:12**

**telco-customer-churn.ipynb - Colaboratory**

**ψ**

**Naan Mudhalvan Project**

**customer churn prediction**

**Team Members:Naan Mudhalvan Project customer churn prediction Team Members:**

**~Akshaya K- 2021115011~Akshaya SM - 2021115012**

**~Akshaya SM - 2021115012~ Preetham V -2021115077**

**~ Preetham V -2021115077~Praveen M - 2021115075**

**~Praveen M- 2021115075**

**~ Akash M-**

**~Akshaya K- 2021115011**

**~ Akash M - 2021115324**

**2021115324**

**# This Python 3 environment comes with many helpful analytics libraries installed**

**# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python**

**# For example, here's several helpful packages to load**

**import numpy as np # linear algebra**

**import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)**

**# Input data files are available in the read-only "../input/" directory**

**# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory**

**import os**

**for dirname, \_, filenames in os.walk('/kaggle/input'):**

**for filename in filenames:**

**print(os.path.join(dirname, filename))**

**# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a versio**

**# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session**

****

**/kaggle/input/telco-customer-churn/WA\_Fn-UseC\_-Telco-Customer-Churn.csv**

**Double-click (or enter) to edit**

**import matplotlib.pyplot as plt**

**df = pd.read\_csv('/kaggle/input/telco-customer-churn/WA\_Fn-UseC\_-Telco-Customer-Churn.csv')**

**df.head(5)**

**customerIDgenderSeniorCitizenPartnerDependentstenurePhoneServiceMultipleLines**

**07590-**

**VHVEGFemale0YesNo1NoNo phone service**

**15575-**

**GNVDEMale0NoNo34YesNo**

**23668-QPYBKMale0NoNo2YesNo**

**37795-**

**CFOCWMale0NoNo45NoNo phone service**

**49237-HQITUFemale0NoNo2YesNo**

**5 rows × 21 columns**

**df.drop('customerID',axis='columns',inplace=True)**

**df.sample(5)**

**https://colab.research.google.com/drive/1PRbApcSyGqNb8-1Mf3ITWuV7VYv\_zNFV?usp=sharing#scrollTo=Uy3C5vAq7fSV&print… 1/919/10/2023, 04:12**

**telco-customer-churn.ipynb - Colaboratory**

**genderSeniorCitizenPartnerDependentstenurePhoneServiceMultipleLines**

**2128Male0YesYes41YesYes**

**384**

**df.dtypesMale0NoNo48YesYes**

**NoNo9NoNo phone service**

**NoNo1YesNo**

**YesNo56YesNo**

**gender**

**3946 Female**

**SeniorCitizen**

**Partner**

**Dependents**

**4081 Female**

**tenure**

**PhoneService**

**MultipleLines**

**3390**

**Male**

**InternetService**

**OnlineSecurity**

**OnlineBackup**

**DeviceProtection**

**TechSupport**

**StreamingTV**

**StreamingMovies**

**Contract**

**PaperlessBilling**

**PaymentMethod**

**MonthlyCharges**

**TotalCharges**

**Churn**

**dtype: object**

**object**

**0**

**int64**

**object**

**object**

**0**

**int64**

**object**

**object**

**0**

**object**

**object**

**object**

**object**

**object**

**object**

**object**

**object**

**object**

**object**

**float64**

**object**

**object**

**InternetS**

**Fi**

**Fi**

**df[pd.to\_numeric(df.TotalCharges,errors='coerce').isnull()]**

**genderSeniorCitizenPartnerDependentstenurePhoneServiceMultipleLines**

**488Female0YesYes0NoNo phone service**

**753Male0NoYes0YesNo**

**936Female0YesYes0YesNo**

**1082Male0YesYes0YesYes**

**1340Female0YesYes0NoNo phone service**

**3331Male0YesYes0YesNo**

**3826Male0YesYes0YesYes**

**4380Female0YesYes0YesNo**

**5218Male0YesYes0YesNo**

**6670Female0YesYes0YesYes**

**6754Male0NoYes0YesYes**

**InternetS**

**df1 = df[df.TotalCharges!=' ']**

**df1['TotalCharges'] = pd.to\_numeric(df['TotalCharges'],errors='coerce')**

**/tmp/ipykernel\_20/1694240200.py:1: SettingWithCopyWarning:**

**A value is trying to be set on a copy of a slice from a DataFrame.**

**Try using .loc[row\_indexer,col\_indexer] = value instead**

**See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-**

**df1['TotalCharges'] = pd.to\_numeric(df['TotalCharges'],errors='coerce')**

**df1.dtypes**

**gender**

**SeniorCitizen**

**Partner**

**Dependents**

**tenure**

**PhoneService**

**MultipleLines**

**object**

**int64**

**object**

**object**

**int64**

**object**

**object**

**https://colab.research.google.com/drive/1PRbApcSyGqNb8-1Mf3ITWuV7VYv\_zNFV?usp=sharing#scrollTo=Uy3C5vAq7fSV&print… 2/919/10/2023, 04:12**

**InternetService**

**OnlineSecurity**

**OnlineBackup**

**DeviceProtection**

**TechSupport**

**StreamingTV**

**StreamingMovies**

**Contract**

**PaperlessBilling**

**PaymentMethod**

**MonthlyCharges**

**TotalCharges**

**Churn**

**dtype: object**

**telco-customer-churn.ipynb - Colaboratory**

**object**

**object**

**object**

**object**

**object**

**object**

**object**

**object**

**object**

**object**

**float64**

**float64**

**object**

**df\_tenure\_no = df1[df1.Churn == 'No'].tenure**

**df\_tenure\_yes = df1[df1.Churn == 'Yes'].tenure**

**plt.hist([df\_tenure\_yes,df\_tenure\_no],color=['green','red'],label=['Churn=Yes','Churn=No'])**

**plt.legend()**

**plt.xlabel('Tenure')**

**plt.ylabel('Number of customers')**

**plt.title('Histogram based on Tenure of customers')**

**Text(0.5, 1.0, 'Histogram based on Tenure of customers')**

**df\_total\_charges\_no = df1[df1.Churn == 'No'].TotalCharges**

**df\_total\_charges\_yes = df1[df1.Churn == 'Yes'].TotalCharges**

**plt.hist([df\_tenure\_yes,df\_tenure\_no],color=['green','red'],label=['Churn=Yes','Churn=No'])**

**plt.legend()**

**plt.xlabel('Total Charges')**

**plt.ylabel('Number of customers')**

**plt.title('Histogram based on Total charges of customers')**

**Text(0.5, 1.0, 'Histogram based on Total charges of customers')**

**https://colab.research.google.com/drive/1PRbApcSyGqNb8-1Mf3ITWuV7VYv\_zNFV?usp=sharing#scrollTo=Uy3C5vAq7fSV&print… 3/919/10/2023, 04:12**

**telco-customer-churn.ipynb - Colaboratory**

**def print\_unique\_values(df):**

**for cols in df.columns:**

**print(cols,df[cols].unique())**

**print\_unique\_values(df1)**

**gender ['Female' 'Male']**

**SeniorCitizen [0 1]**

**Partner ['Yes' 'No']**

**Dependents ['No' 'Yes']**

**tenure [ 1 34 2 45 8 22 10 28 62 13 16 58 49 25 69 52 71 21 12 30 47 72 17 27**

**5 46 11 70 63 43 15 60 18 66 9 3 31 50 64 56 7 42 35 48 29 65 38 68**

**32 55 37 36 41 6 4 33 67 23 57 61 14 20 53 40 59 24 44 19 54 51 26 39]**

**PhoneService ['No' 'Yes']**

**MultipleLines ['No phone service' 'No' 'Yes']**

**InternetService ['DSL' 'Fiber optic' 'No']**

**OnlineSecurity ['No' 'Yes' 'No internet service']**

**OnlineBackup ['Yes' 'No' 'No internet service']**

**DeviceProtection ['No' 'Yes' 'No internet service']**

**TechSupport ['No' 'Yes' 'No internet service']**

**StreamingTV ['No' 'Yes' 'No internet service']**

**StreamingMovies ['No' 'Yes' 'No internet service']**

**Contract ['Month-to-month' 'One year' 'Two year']**

**PaperlessBilling ['Yes' 'No']**

**PaymentMethod ['Electronic check' 'Mailed check' 'Bank transfer (automatic)'**

**'Credit card (automatic)']**

**MonthlyCharges [29.85 56.95 53.85 ... 63.1 44.2 78.7 ]**

**TotalCharges [ 29.85 1889.5**

**108.15 ... 346.45 306.6 6844.5 ]**

**Churn ['No' 'Yes']**

**df1.replace('No phone service','No',inplace=True)**

**/tmp/ipykernel\_20/628100714.py:1: SettingWithCopyWarning:**

**A value is trying to be set on a copy of a slice from a DataFrame**

**See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-**

**df1.replace('No phone service','No',inplace=True)**

**df1.replace('No internet service','No',inplace=True)**

**/tmp/ipykernel\_20/4127402845.py:1: SettingWithCopyWarning:**

**A value is trying to be set on a copy of a slice from a DataFrame**

**See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-**

**df1.replace('No internet service','No',inplace=True)**

**print\_unique\_values(df1)**

**gender ['Female' 'Male']**

**SeniorCitizen [0 1]**

**Partner ['Yes' 'No']**

**Dependents ['No' 'Yes']**

**tenure [ 1 34 2 45 8 22 10 28 62 13 16 58 49 25 69 52 71 21 12 30 47 72 17 27**

**5 46 11 70 63 43 15 60 18 66 9 3 31 50 64 56 7 42 35 48 29 65 38 68**

**32 55 37 36 41 6 4 33 67 23 57 61 14 20 53 40 59 24 44 19 54 51 26 39]**

**PhoneService ['No' 'Yes']**

**MultipleLines ['No' 'Yes']**

**InternetService ['DSL' 'Fiber optic' 'No']**

**OnlineSecurity ['No' 'Yes']**

**OnlineBackup ['Yes' 'No']**

**DeviceProtection ['No' 'Yes']**

**TechSupport ['No' 'Yes']**

**StreamingTV ['No' 'Yes']**

**StreamingMovies ['No' 'Yes']**

**Contract ['Month-to-month' 'One year' 'Two year']**

**PaperlessBilling ['Yes' 'No']**

**PaymentMethod ['Electronic check' 'Mailed check' 'Bank transfer (automatic)'**

**'Credit card (automatic)']**

**MonthlyCharges [29.85 56.95 53.85 ... 63.1 44.2 78.7 ]**

**TotalCharges [ 29.85 1889.5**

**108.15 ... 346.45 306.6 6844.5 ]**

**Churn ['No' 'Yes']**

**df2 = pd.get\_dummies(data=df1,columns=['InternetService','Contract','PaymentMethod'])**

**df2.shape**

**(7032, 27)**

**https://colab.research.google.com/drive/1PRbApcSyGqNb8-1Mf3ITWuV7VYv\_zNFV?usp=sharing#scrollTo=Uy3C5vAq7fSV&print… 4/919/10/2023, 04:12**

**telco-customer-churn.ipynb - Colaboratory**

**print\_unique\_values(df2)**

**gender ['Female' 'Male']**

**SeniorCitizen [0 1]**

**Partner ['Yes' 'No']**

**Dependents ['No' 'Yes']**

**tenure [ 1 34 2 45 8 22 10 28 62 13 16 58 49 25 69 52 71 21 12 30 47 72 17 27**

**5 46 11 70 63 43 15 60 18 66 9 3 31 50 64 56 7 42 35 48 29 65 38 68**

**32 55 37 36 41 6 4 33 67 23 57 61 14 20 53 40 59 24 44 19 54 51 26 39]**

**PhoneService ['No' 'Yes']**

**MultipleLines ['No' 'Yes']**

**OnlineSecurity ['No' 'Yes']**

**OnlineBackup ['Yes' 'No']**

**DeviceProtection ['No' 'Yes']**

**TechSupport ['No' 'Yes']**

**StreamingTV ['No' 'Yes']**

**StreamingMovies ['No' 'Yes']**

**PaperlessBilling ['Yes' 'No']**

**MonthlyCharges [29.85 56.95 53.85 ... 63.1 44.2 78.7 ]**

**TotalCharges [ 29.85 1889.5**

**108.15 ... 346.45 306.6 6844.5 ]**

**Churn ['No' 'Yes']**

**InternetService\_DSL [ True False]**

**InternetService\_Fiber optic [False True]**

**InternetService\_No [False True]**

**Contract\_Month-to-month [ True False]**

**Contract\_One year [False True]**

**Contract\_Two year [False True]**

**PaymentMethod\_Bank transfer (automatic) [False True]**

**PaymentMethod\_Credit card (automatic) [False True]**

**PaymentMethod\_Electronic check [ True False]**

**PaymentMethod\_Mailed check [False True]**

**df\_true\_false\_cols = ['InternetService\_DSL','InternetService\_Fiber optic','InternetService\_No','Contract\_One year','Contract\_**

**for cols in df\_true\_false\_cols:**

**df2[cols].replace({True:1,False:0},inplace=True)**

**print\_unique\_values(df2)**

**gender ['Female' 'Male']**

**SeniorCitizen [0 1]**

**Partner ['Yes' 'No']**

**Dependents ['No' 'Yes']**

**tenure [ 1 34 2 45 8 22 10 28 62 13 16 58 49 25 69 52 71 21 12 30 47 72 17 27**

**5 46 11 70 63 43 15 60 18 66 9 3 31 50 64 56 7 42 35 48 29 65 38 68**

**32 55 37 36 41 6 4 33 67 23 57 61 14 20 53 40 59 24 44 19 54 51 26 39]**

**PhoneService ['No' 'Yes']**

**MultipleLines ['No' 'Yes']**

**OnlineSecurity ['No' 'Yes']**

**OnlineBackup ['Yes' 'No']**

**DeviceProtection ['No' 'Yes']**

**TechSupport ['No' 'Yes']**

**StreamingTV ['No' 'Yes']**

**StreamingMovies ['No' 'Yes']**

**PaperlessBilling ['Yes' 'No']**

**MonthlyCharges [29.85 56.95 53.85 ... 63.1 44.2 78.7 ]**

**TotalCharges [ 29.85 1889.5**

**108.15 ... 346.45 306.6 6844.5 ]**

**Churn ['No' 'Yes']**

**InternetService\_DSL [1 0]**

**InternetService\_Fiber optic [0 1]**

**InternetService\_No [0 1]**

**Contract\_Month-to-month [1 0]**

**Contract\_One year [0 1]**

**Contract\_Two year [0 1]**

**PaymentMethod\_Bank transfer (automatic) [0 1]**

**PaymentMethod\_Credit card (automatic) [0 1]**

**PaymentMethod\_Electronic check [1 0]**

**PaymentMethod\_Mailed check [0 1]**

**df\_yes\_no\_cols = ['Partner','Dependents','PhoneService','MultipleLines','OnlineSecurity','OnlineBackup','DeviceProtection','T**

**for cols in df\_yes\_no\_cols:**

**df2[cols].replace({'Yes':1,'No':0},inplace=True)**

**print\_unique\_values(df2)**

**gender ['Female' 'Male']**

**SeniorCitizen [0 1]**

**Partner [1 0]**

**Dependents [0 1]**

**tenure [ 1 34 2 45 8 22 10 28 62 13 16 58 49 25 69 52 71 21 12 30 47 72 17 27**

**5 46 11 70 63 43 15 60 18 66 9 3 31 50 64 56 7 42 35 48 29 65 38 68**

**32 55 37 36 41 6 4 33 67 23 57 61 14 20 53 40 59 24 44 19 54 51 26 39]**

**PhoneService [0 1]**

**MultipleLines [0 1]**

**OnlineSecurity [0 1]**

**https://colab.research.google.com/drive/1PRbApcSyGqNb8-1Mf3ITWuV7VYv\_zNFV?usp=sharing#scrollTo=Uy3C5vAq7fSV&print… 5/919/10/2023, 04:12**

**telco-customer-churn.ipynb - Colaboratory**

**OnlineBackup [1 0]**

**DeviceProtection [0 1]**

**TechSupport [0 1]**

**StreamingTV [0 1]**

**StreamingMovies [0 1]**

**PaperlessBilling [1 0]**

**MonthlyCharges [29.85 56.95 53.85 ... 63.1 44.2 78.7 ]**

**TotalCharges [ 29.85 1889.5**

**108.15 ... 346.45 306.6**

**Churn [0 1]**

**InternetService\_DSL [1 0]**

**InternetService\_Fiber optic [0 1]**

**InternetService\_No [0 1]**

**Contract\_Month-to-month [1 0]**

**Contract\_One year [0 1]**

**Contract\_Two year [0 1]**

**PaymentMethod\_Bank transfer (automatic) [0 1]**

**PaymentMethod\_Credit card (automatic) [0 1]**

**PaymentMethod\_Electronic check [1 0]**

**PaymentMethod\_Mailed check [0 1]**

**6844.5 ]**

**df2['gender'].replace({'Female':1,'Male':0},inplace=True)**

**cols\_to\_scale = ['tenure','MonthlyCharges','TotalCharges']**

**from sklearn.preprocessing import MinMaxScaler**

**scaler = MinMaxScaler()**

**df2[cols\_to\_scale] = scaler.fit\_transform(df2[cols\_to\_scale])**

**print\_unique\_values(df2)**

**gender [1 0]**

**SeniorCitizen [0 1]**

**Partner [1 0]**

**Dependents [0 1]**

**tenure [0.**

**0.46478873 0.01408451 0.61971831 0.09859155 0.29577465**

**0.12676056 0.38028169 0.85915493 0.16901408 0.21126761 0.8028169**

**0.67605634 0.33802817 0.95774648 0.71830986 0.98591549 0.28169014**

**0.15492958 0.4084507 0.64788732 1.**

**0.22535211 0.36619718**

**0.05633803 0.63380282 0.14084507 0.97183099 0.87323944 0.5915493**

**0.1971831 0.83098592 0.23943662 0.91549296 0.11267606 0.02816901**

**0.42253521 0.69014085 0.88732394 0.77464789 0.08450704 0.57746479**

**0.47887324 0.66197183 0.3943662 0.90140845 0.52112676 0.94366197**

**0.43661972 0.76056338 0.50704225 0.49295775 0.56338028 0.07042254**

**0.04225352 0.45070423 0.92957746 0.30985915 0.78873239 0.84507042**

**0.18309859 0.26760563 0.73239437 0.54929577 0.81690141 0.32394366**

**0.6056338 0.25352113 0.74647887 0.70422535 0.35211268 0.53521127]**

**PhoneService [0 1]**

**MultipleLines [0 1]**

**OnlineSecurity [0 1]**

**OnlineBackup [1 0]**

**DeviceProtection [0 1]**

**TechSupport [0 1]**

**StreamingTV [0 1]**

**StreamingMovies [0 1]**

**PaperlessBilling [1 0]**

**MonthlyCharges [0.11542289 0.38507463 0.35422886 ... 0.44626866 0.25820896 0.60149254]**

**TotalCharges [0.0012751 0.21586661 0.01031041 ... 0.03780868 0.03321025 0.78764136]**

**Churn [0 1]**

**InternetService\_DSL [1 0]**

**InternetService\_Fiber optic [0 1]**

**InternetService\_No [0 1]**

**Contract\_Month-to-month [1 0]**

**Contract\_One year [0 1]**

**Contract\_Two year [0 1]**

**PaymentMethod\_Bank transfer (automatic) [0 1]**

**PaymentMethod\_Credit card (automatic) [0 1]**

**PaymentMethod\_Electronic check [1 0]**

**PaymentMethod\_Mailed check [0 1]**

**df2.dtypes**

**gender**

**SeniorCitizen**

**Partner**

**Dependents**

**tenure**

**PhoneService**

**MultipleLines**

**OnlineSecurity**

**OnlineBackup**

**DeviceProtection**

**TechSupport**

**StreamingTV**

**StreamingMovies**

**PaperlessBilling**

**MonthlyCharges**

**TotalCharges**

**Churn**

**InternetService\_DSL**

**InternetService\_Fiber optic**

**int64**

**int64**

**int64**

**int64**

**float64**

**int64**

**int64**

**int64**

**int64**

**int64**

**int64**

**int64**

**int64**

**int64**

**float64**

**float64**

**int64**

**int64**

**int64**

**https://colab.research.google.com/drive/1PRbApcSyGqNb8-1Mf3ITWuV7VYv\_zNFV?usp=sharing#scrollTo=Uy3C5vAq7fSV&print… 6/919/10/2023, 04:12**

**InternetService\_No**

**Contract\_Month-to-month**

**Contract\_One year**

**Contract\_Two year**

**PaymentMethod\_Bank transfer (automatic)**

**PaymentMethod\_Credit card (automatic)**

**PaymentMethod\_Electronic check**

**PaymentMethod\_Mailed check**

**dtype: object**

**telco-customer-churn.ipynb - Colaboratory**

**int64**

**int64**

**int64**

**int64**

**int64**

**int64**

**int64**

**int64**

**X = df2.drop('Churn',axis='columns')**

**y = df2['Churn']**

**print('Shape of X:',X.shape)**

**print('Shape of y:',y.shape)**

**Shape of X: (7032, 26)**

**Shape of y: (7032,)**

**from sklearn.model\_selection import train\_test\_split**

**X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.2,random\_state=5)**

**print('Shape of X\_train:',X\_train.shape)**

**print('Shape of X\_test:',X\_test.shape)**

**print('Shape of y\_train:',y\_train.shape)**

**print('Shape of y\_test:',y\_test.shape)**

**Shape of X\_train: (5625, 26)**

**Shape of X\_test: (1407, 26)**

**Shape of y\_train: (5625,)**

**Shape of y\_test: (1407,)**

**import tensorflow as tf**

**from tensorflow import keras**

**model = keras.Sequential([**

**keras.layers.Dense(20,input\_shape=(26,),activation='relu'),**

**keras.layers.Dense(1,activation='sigmoid')**

**])**

**model.compile(optimizer='adam',loss='binary\_crossentropy',metrics=['accuracy'])**

**model.fit(X\_train,y\_train,epochs=10)**

**Epoch 1/10**

**176/176 [==============================] - 1s 1ms/step - loss: 0.5006 - accuracy: 0.7550**

**Epoch 2/10**

**176/176 [==============================] - 0s 1ms/step - loss: 0.4357 - accuracy: 0.7902**

**Epoch 3/10**

**176/176 [==============================] - 0s 2ms/step - loss: 0.4244 - accuracy: 0.7977**

**Epoch 4/10**

**176/176 [==============================] - 0s 1ms/step - loss: 0.4196 - accuracy: 0.8021**

**Epoch 5/10**

**176/176 [==============================] - 0s 2ms/step - loss: 0.4162 - accuracy: 0.8043**

**Epoch 6/10**

**176/176 [==============================] - 0s 2ms/step - loss: 0.4148 - accuracy: 0.8048**

**Epoch 7/10**

**176/176 [==============================] - 0s 1ms/step - loss: 0.4128 - accuracy: 0.8066**

**Epoch 8/10**

**176/176 [==============================] - 0s 1ms/step - loss: 0.4121 - accuracy: 0.8076**

**Epoch 9/10**

**176/176 [==============================] - 0s 1ms/step - loss: 0.4110 - accuracy: 0.8071**

**Epoch 10/10**

**176/176 [==============================] - 0s 1ms/step - loss: 0.4097 - accuracy: 0.8089**

**<keras.callbacks.History at 0x7e596b5d7df0>**

**model.evaluate(X\_test,y\_test)**

**44/44 [==============================] - 0s 1ms/step - loss: 0.4416 - accuracy: 0.7889**

**[0.44159233570098877, 0.7889125943183899]**

**from sklearn.metrics import classification\_report,confusion\_matrix**

**yp = model.predict(X\_test)**

**yp**

**44/44 [==============================] - 0s 1ms/step**

**array([[0.19795616],**

**[0.3622312 ],**

**[0.01491462],**

**...,**

**[0.72733843],**

**[0.6845046 ],**

**[0.55635047]], dtype=float32)**

**yp.shape**

**https://colab.research.google.com/drive/1PRbApcSyGqNb8-1Mf3ITWuV7VYv\_zNFV?usp=sharing#scrollTo=Uy3C5vAq7fSV&print… 7/919/10/2023, 04:12**

**telco-customer-churn.ipynb - Colaboratory**

**(1407, 1)**

**yp[:5]**

**array([[0.19795616],**

**[0.3622312 ],**

**[0.01491462],**

**[0.7472115 ],**

**[0.48185456]], dtype=float32)**

**y\_pred = []**

**for x in yp:**

**if x > 0.5:**

**y\_pred.append(1)**

**else:**

**y\_pred.append(0)**

**y\_pred[:5]**

**[0, 0, 0, 1, 0]**

**y\_test[:5]**

**2660**

**0**

**744**

**0**

**5579**

**1**

**64**

**1**

**3287**

**1**

**Name: Churn, dtype: int64**

**print('Classification Report:',classification\_report(y\_test,y\_pred))**

**Classification Report:**

**precision**

**recall**

**0**

**10.84**

**0.650.87**

**0.580.85**

**0.61999**

**408**

**accuracy**

**macro avg**

**weighted avg0.74**

**0.780.73**

**0.790.79**

**0.73**

**0.791407**

**1407**

**1407**

**f1-score**

**support**

**import seaborn as sns**

**cm = confusion\_matrix(y\_test,y\_pred)**

**sns.heatmap(cm,annot=True,fmt='.2f')**

**plt.title('Confusion Matrix')**

**plt.xlabel('Predicted')**

**plt.ylabel('True')**

**Text(50.722222222222214, 0.5, 'True')**

**https://colab.research.google.com/drive/1PRbApcSyGqNb8-1Mf3ITWuV7VYv\_zNFV?usp=sharing#scrollTo=Uy3C5vAq7fSV&print… 8/919/10/2023, 04:12**

**telco-customer-churn.ipynb - Colaboratory**

**https://colab.research.google.com/drive/1PRbApcSyGqNb8-1Mf3ITWuV7VYv\_zNFV?usp=sharing#scrollTo=Uy3C5vAq7fSV&print… 9/9**

Data preprocessing and visualization:-







Results:-





