SRI SRI UNIVERSITY

Customer Churn Prediction in Telecom Industry

FINAL PROJECT REPORT

Customer Churn Prediction and Telecom Industry

BACHELOR OF TECHNOLOGY

Computer Science & Engineering

(DS)

SUBMITTED BY

Riddhima Banwale- (FET-BDS-2022-26-008)

Aditya Narayan Purohit- (FET-BDS-2022-26-006)

Subhangee Das- (FET-BDS-2022-26-003)

Somya Ranjan Mishra (FET-BDS-2022-26-010)

GUIDED BY

Dr. Deepak Sahoo

Associate Professor, FET

SRI SRI UNIVERSITY

IBM Innovation Centre for Education

SRI SRI UNIVERSITY
LEARN LEAD SERVE

PHEMESOFT
Software That Matters

Certificate Of Submission

This is to certify that Subhangee Das, Aditya Narayan Purohit, Riddhima Banwale and Somya Ranjan Mishra enrolled in *BTech. Computer Science & Engineering with Specialization in "Data Science* "at Sri University have successfully completed and submitted their minor project titled "**Customer Churn Prediction in Telecom Industry**" as a part of their academic curriculum. The project was submitted on **April 15th 2024**.

Throughout the duration of the project, the learners have demonstrated a laudable understanding of the subject matter and have exhibited exceptional skills in research, analysis & prediction.

We acknowledge the efforts and dedication put forth by the students in completing this project and commend their diligence to academic proficiency.

_____ _____

Institution Supervisor **External Examiner**

Content

ABSTRACT

The telecommunications industry faces significant challenges in today's technology-driven era, particularly with the advent of cloud technology. This has intensified competition and highlighted the importance of effective customer communication and service. Amidst these challenges, telecom companies must also contend with existing issues such as outdated infrastructure and business models. To thrive in this demanding landscape, telecom companies must prioritize understanding and meeting customer needs. Employee attrition analysis emerges as a crucial tool in this endeavour. By examining employee turnover patterns, companies can gain insights into underlying issues affecting customer service, pricing strategies, and marketing efforts. The key lies not merely in surviving, but in leveraging attrition analysis to proactively address customer concerns and enhance overall service quality. Through this approach, telecom companies can optimize their operations, retain loyal customers, and adapt to the dynamic demands of the industry.

List of acronyms:

CM=churned-married

CUM=Churned-unmarried

SM= Stayed -married

SUM=Stayed-unmarried

JM=Joined-married

JUM=Joined-unmarried

ag1=Age group1(age<=20)

ag2=Age group2(21>=age<=30)

ag3=Age group3(31>=age<=40)

ag4=Age group4(age>=41)

CMAg1-Chruned-married in Age group1

CMAg2-Chruned-married in Age group2

CMAg3-Chruned-married in Age group3

CMAg4-Chruned-married in Age group4

CUMAg1-Chruned-unmarried in Age group1

CUMAg2-Chruned-unmarried in Age group2

CUMAg3-Chruned-unmarried in Age group3

CUMAg4-Chruned-unmarried in Age group4

CU_multi- churned customer that is using multilines

CNU_multi -churned customer that is not using multilines

SU_multi -stayed customer that is using multilines

SNU_multi -stayed customer that is not using multilines

TF=Term frequency

DF=Document frequency

IDF=Inverse document frequency

Introduction

Problem Statement:

Telecom companies face a significant challenge with customer churn, as retaining existing customers is crucial for profitability. Customer churn, which occurs when customers discontinue services, is costly for companies, as acquiring new customers involves substantial marketing and sales expenses. Customer retention relies on delivering excellent customer service and high-quality products, but understanding customers on a deeper level is paramount. By leveraging the vast amounts of customer data available, telecom companies can develop churn prediction models to identify customers most likely to defect. This enables companies to focus their marketing efforts on retaining these high-risk customers, ultimately improving customer retention rates and reducing churn-related costs.

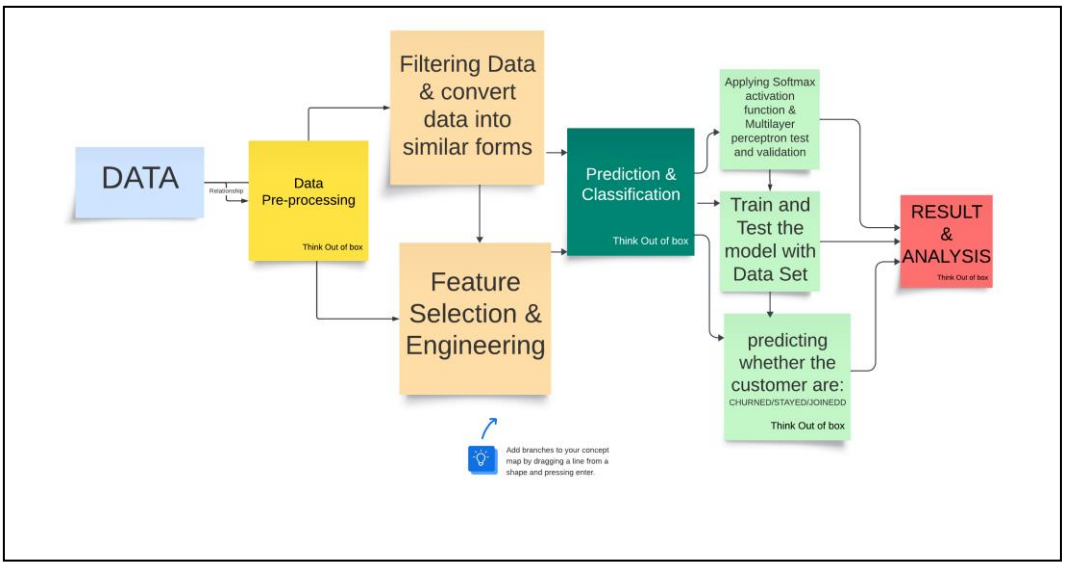
Motivation of work:

The motivation behind this project from the pressing issue of customer churn in the telecommunication industry. Telecom companies invest significant resources in acquiring new customers, making it imperative to minimize customer defection. Understanding the factors driving churn and developing effective strategies to retain customers is essential for sustainable growth and profitability. By addressing churn proactively, telecom companies can enhance customer satisfaction, reduce costs, and strengthen their competitive position in the market.

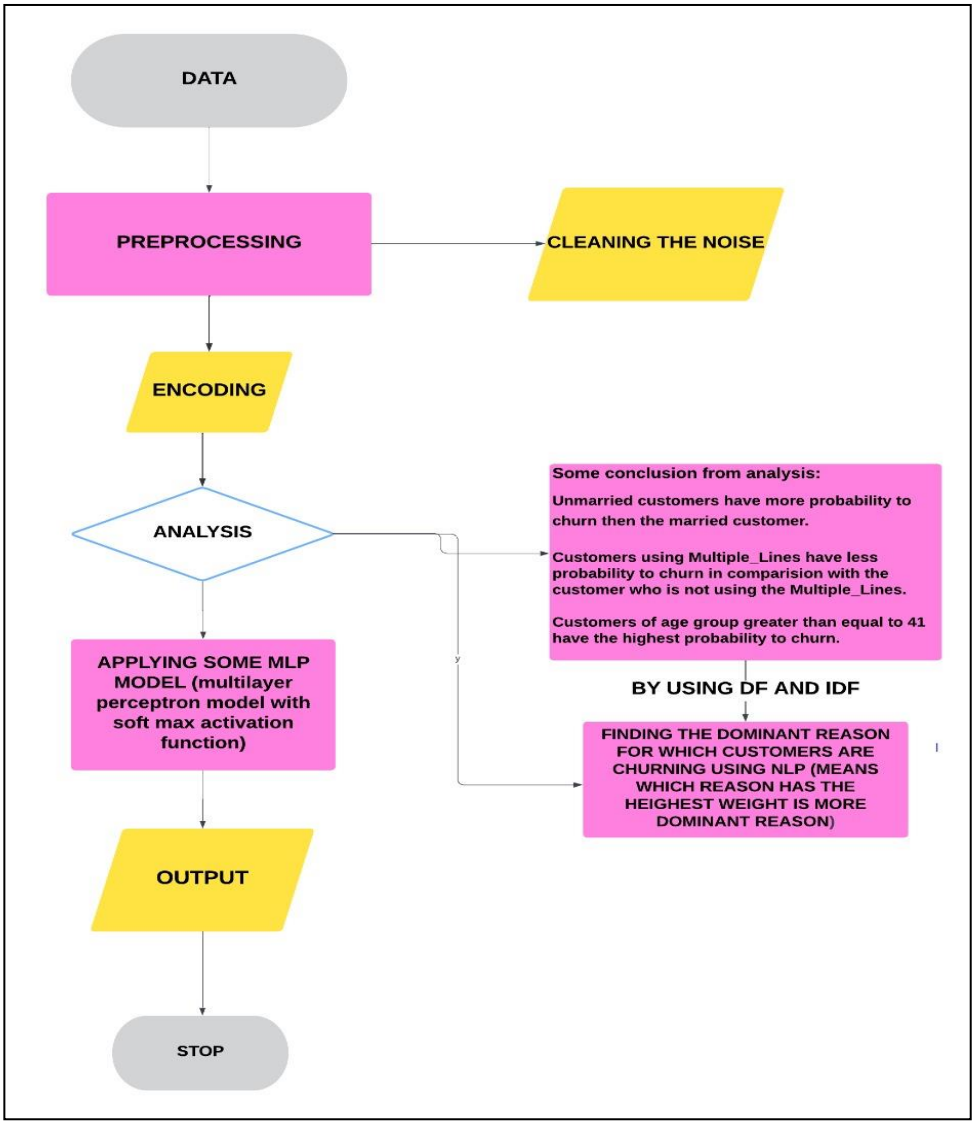
Objective of the project:

The main objective of this project is why the customer are churning from the existing telecom subscriptions for that we have to analyse telecom churn data and develop the predictive model to identify customer at risk of churn. The project aims to:

1. Understand the factor contributing to customer churn in the telecom industry.
2. Explore and preprocess the telecom dataset to prepare them for analysis.
3. Build and evaluate machine learning models to predict customer churn.
4. Identify key features and factors influencing churn prediction.
5. Develop actionable insights and recommendations to reduce churn and improve customer retention strategies.
6. Validate the predictive models and assess their effectiveness in real-world scenarios.
7. Provide telecom companies with practical solutions to mitigate churn and enhance customer satisfaction.



Proposed model to be work



OurApproach

Pre-processing steps

Data Reading: The 1st step of analysis consists of reading and storing the data in a pandas data frame using the (pandas. read_csv) function.

```
In [1]: import pandas as pd
        import numpy as np
        from category_encoders import *

In [2]: df=pd.read_csv(r"D:\Data Science\telecom_customer_churn.csv")

In [3]: df.columns

Out[3]: Index(['Customer ID', 'Gender', 'Age', 'Married', 'Number of Dependents',
               'City', 'Zip Code', 'Latitude', 'Longitude', 'Number of Referrals',
               'Tenure in Months', 'Offer', 'PhoneService',
               'Avg Monthly Long Distance Charges', 'Multiple_Lines',
               'Internet_Service', 'Internet  _Type', 'Avg Monthly GB Download',
               'Online_Security', 'Online_Backup', 'Device_Protection_Plan',
               'Premium _Tech _Support', 'Streaming TV', 'Streaming Movies',
               'Streaming Music', 'Unlimited Data', 'Contract', 'Paperless Billing',
               'Payment Method', 'Monthly Charge', 'Total Charges', 'Total Refunds',
               'Total Extra Data Charges', 'Total Long Distance Charges',
               'Total Revenue', 'Customer Status', 'Churn Category', 'Churn Reason'],
              dtype='object')
```

1. **Exploratory Data Analysis and Data Cleaning**: In EDA we want to know that as much information as possible about the data by using (df.info) method

```
 0   Customer ID                        7043 non-null   object
 1   Gender                             7043 non-null   object
 2   Age                                7043 non-null   int64
 3   Married                            7043 non-null   object
 4   Number of Dependents               7043 non-null   int64
 5   City                               7043 non-null   object
 6   Zip Code                           7043 non-null   int64
 7   Latitude                           7043 non-null   float64
 8   Longitude                          7043 non-null   float64
 9   Number of Referrals                7043 non-null   int64
 10  Tenure in Months                   7043 non-null   int64
 11  Offer                              7043 non-null   object
 12  PhoneService                       7043 non-null   object
 13  Avg Monthly Long Distance Charges  6361 non-null   float64
 14  Multiple_Lines                     6361 non-null   object
 15  Internet_Service                   7043 non-null   object
 16  Internet  _Type                    5543 non-null   object
 17  Avg Monthly GB Download            5517 non-null   float64
 18  Online_Security                    5524 non-null   object
 19  Online_Backup                      5521 non-null   object
 20  Device_Protection_Plan             5517 non-null   object
 21  Premium _Tech _Support             5517 non-null   object
 22  Streaming TV                       5517 non-null   object
 23  Streaming Movies                   5517 non-null   object
 24  Streaming Music                    5517 non-null   object
 25  Unlimited Data                     5531 non-null   object
 26  Contract                           7043 non-null   object
 27  Paperless Billing                  7043 non-null   object
 28  Payment Method                     7043 non-null   object
 29  Monthly Charge                     7043 non-null   float64
 30  Total Charges                      7043 non-null   float64
 31  Total Refunds                      7043 non-null   float64
 32  Total Extra Data Charges           7043 non-null   int64
 33  Total Long Distance Charges        7043 non-null   float64
 34  Total Revenue                      7043 non-null   float64
 35  Customer Status                    7043 non-null   object
```

We can fill the missing values by the help of principle of locality, it will fill all the missing values of the data set whether the datatype is integer or Boolean or categorical data.

```
In [4]: 1 df.isnull().sum()

Out[4]: Customer ID                          0
        Gender                               0
        Age                                  0
        Married                              0
        Number of Dependents                 0
        City                                 0
        Zip Code                             0
        Latitude                             0
        Longitude                            0
        Number of Referrals                  0
        Tenure in Months                     0
        Offer                                0
        PhoneService                         0
        Avg Monthly Long Distance Charges  682
        Multiple_Lines                     682
        Internet_Service                     0
        Internet _Type                    1526
        Avg Monthly GB Download           1526
        Online_Security                   1526
        Online_Backup                     1526
        Device_Protection_Plan            1526
        Premium _Tech _Support            1526
        Streaming TV                      1526
        Streaming Movies                  1526
        Streaming Music                   1526
        Unlimited Data                    1526
        Contract                             0
        Paperless Billing                    0
        Payment Method                       0
```

Code for prinple of locality:

```
principal_locality={}
for col in df.columns:
    principal_locality[col]=df[col].mode()[0]
    df[col].fillna(principal_locality[col],inplace=True)
    #print(df)
```

Encoding: We encode all the categorical attributes. By using the package called “Label Encoder” form sklearn library.

Code: “from sklearn. preprocessing import LabelEncoder

label encoder = LabelEncoder ()”

with the help of this we encode all the categorical columns.

```
In [16]: 1 df.head()
```

Out[16]:

	Customer ID	Gender	Age	Married	Number of Dependents	City	Zip Code	Latitude	Longitude	Number of Referrals	...	Payment Method	Monthly Charge	Total Charges	Total Refunds	Total Extra Data Charges	Total Long Distance Charges
0	1118	0	46	0	0	1	93510	34.501452	-118.207862	0	...	1	44.05	202.15	0.0	0	54.95
1	2312	1	69	1	0	1	93510	34.501452	-118.207862	0	...	0	79.30	2414.55	0.0	0	305.95
2	4359	1	34	0	0	2	92301	34.667815	-117.536183	0	...	1	45.80	436.20	0.0	0	154.50
3	6354	1	54	0	0	3	96006	41.171578	-120.913161	0	...	0	69.15	488.65	0.0	0	147.84
4	4967	1	53	1	0	4	91301	34.129058	-118.759788	0	...	0	108.65	6937.95	0.0	0	2722.20

5 rows × 38 columns

FEATURE ANALYSIS

Descriptive Statistics: after encoding we did some statistical analysis like:

1. No. of married customer that churned: 669
2. No. of unmarried customer that churned: 1200
3. No. of unmarried customer that stayed: 2071
4. No. of married customer that stayed: 2649
5. No. of married customer that joined: 84
6. No. of unmarried customer that joined: 370

On Multiple_Lines we count churned stayed and joined also:

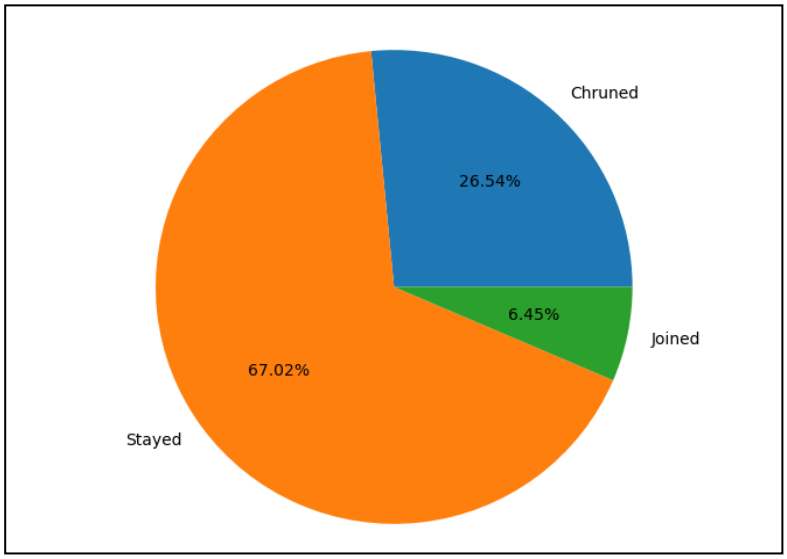
1. No. of customers using Multiple_Lines that churned: 850
2. No. of customers not using Multiple_Lines that churned: 1019
3. No. of customers not using Multiple_Lines that stayed: 2644
4. No. of customers using Multiple_Lines that stayed: 2076
5. No. of customers using Multiple_Lines that joined: 45
6. No. of customers not using Multiple_Lines that joined: 409

On age category we count which interval are stayed and churned:

1. No. of married customer of age less than 20, that churned: 14
2. No. of married customer of age between 21 to 30 that churned: 99
3. No. of married customer of age between 31 to 40 that churned: 106

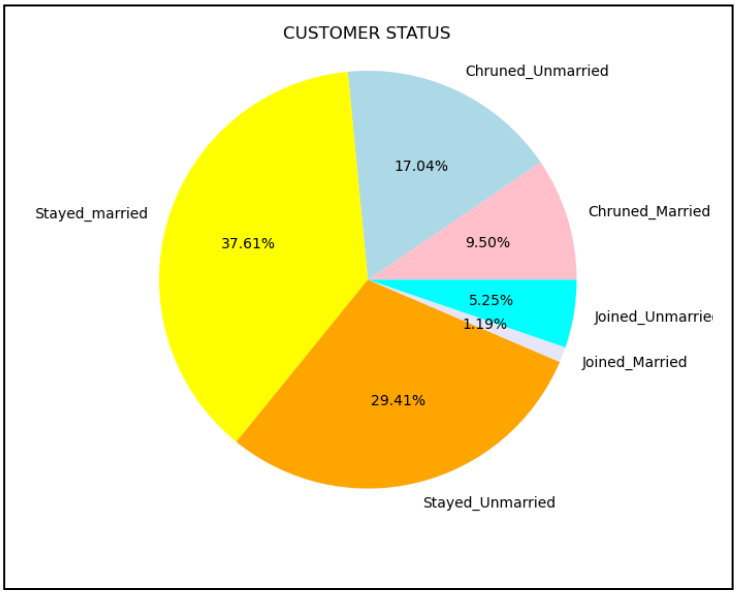
4. No. of married customer of age above 41, that churned: 438
5. No. of unmarried customer of age less than 20, that churned: 11
6. No. of unmarried customer of age between 21 to 30, that churned: 195
7. No. of unmarried customer of age between 31 to 40, that churned: 197
8. No. of unmarried customer of age above 41, that churned: 752

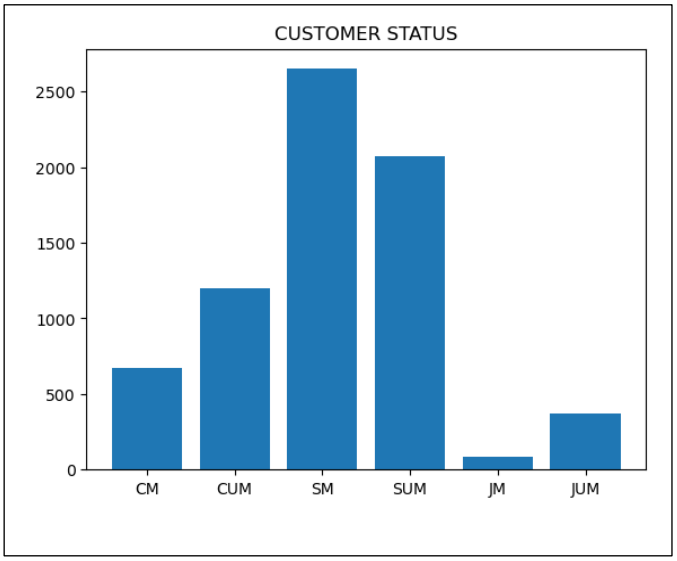
Data Visualization: We done some visualisation on the customer status, relation between the attribute Customer_Status and Married, Age and Multilines attribute.

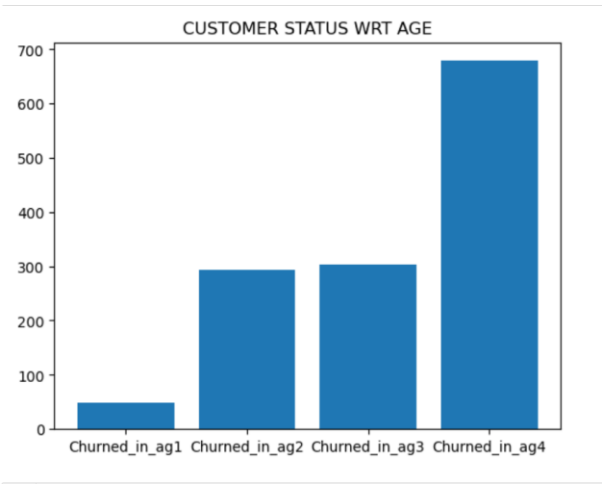


This graph tells us the percentage of customers churned, stayed and joined.

The graphs below tells us about percentage of customers status(stayed, joined, churned) of married and unmarried customers.

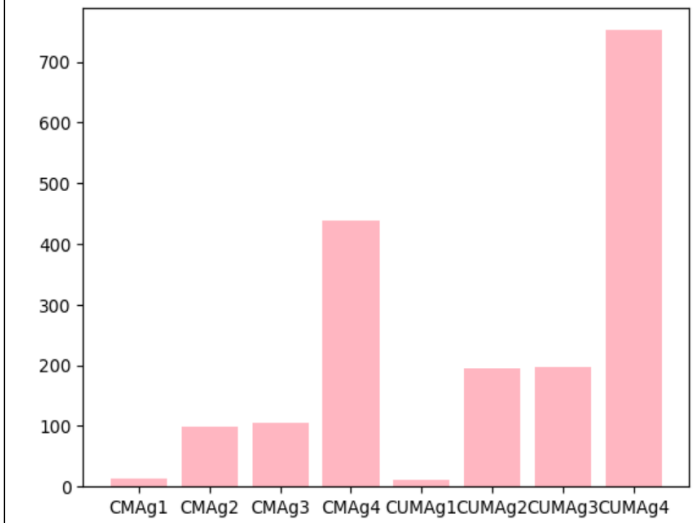


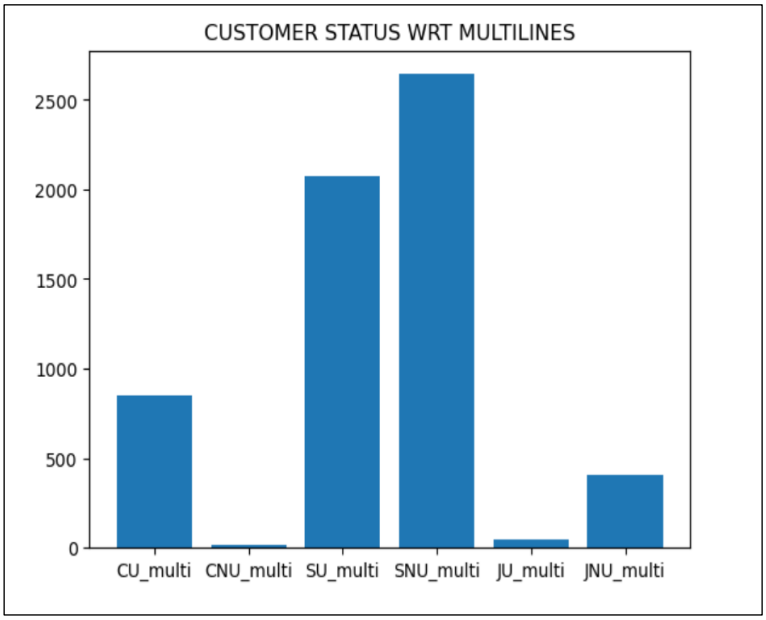




Here, in this graph we can see that the customers of age group 4 (i.e, age>=41) are more likely to churned.

Here, in this graph we can see that the customers of Age group 4 who are unmarried are more likely to churn.





This graph shows us that customers using multilines are more likely to stay and it also shows us that customers using mutlilines are more likely to churn in comparision to the customers who are not using multilines.

REMOVING STOP WORDS AND PRONOUNS

```
stop_words = set(stopwords.words('english'))

def remove_stopwords(text):
    tokens = nltk.word_tokenize(text)
    filtered_tokens = [word for word in tokens if word.lower() not in stop_words]
    return ' '.join(filtered_tokens)

df2['Churn_Reason']=df2['Churn_Reason'].apply(remove_stopwords)

nltk.download('averaged_perceptron_tagger')

def remove_pronouns(text):
    tokens = word_tokenize(text)
    tagged_words = nltk.pos_tag(tokens)
    pronouns = ['NNP']  # POS tags for pronouns
    filtered_tokens = [word for word, pos in tagged_words if pos not in pronouns]
    return ' '.join(filtered_tokens)

df2['Churn_Reason']=df2['Churn_Reason'].apply(remove_pronouns)
```

By using this code we remove the stopwords and pronouns so, that we get can find out the proper reason of customer churn, without the influence of stopwords and pronouns.

```
In [64]: 1 df2.head()
```

Out[64]:

	Age	Married	Customer_Status	Churn_Reason
4	53	1	0	support person
5	56	0	0	service provider
8	20	0	0	Competitor offered data
19	40	0	0	Competitor offered higher download speeds
24	43	0	0	Competitor offered higher download speeds

Here, all the stop words and pronoun are removed of Chrun_Reason attribute by using “nltk” library (natural language tool kit).

NATURAL LANGUAGE PROCESSING TASKS

NLP task :

- We find out the TF(Term frequency) of each main word in chruned_reason after removing the stop- words and pronouns, then we find out the DF(document frequency) of the each unique words, then after we find out the IDF(inverse document frequency) by using the formula [logTF/n]. From IDF we findout weight of each word with the help of formula [TF×IDF]. Then by adding the individual weights of the words to get the weight of the reason**.**
- After finding the weight of each reason, we findout the most prominent reason of the customer leaving by seeing the highest weight.We were able to do so, since we have changed our attribute (Chrun_Reason) in the from of vector by using Counter Vectorizer.

```
:  1 from sklearn.feature_extraction.text import CountVectorizer
   2 cv=CountVectorizer(max_features=5000)

:  1 vectors=cv.fit_transform(df2['Churn_Reason'])
   2
```

[Counter Vectorizer is a technique used for converting a collection of text documents into a numerical representation.]

```
In [68]:  1 from collections import Counter
          2 text=' '.join(df2['Churn_Reason'])

In [69]:  1 words = text.split()
          2
          3 frequency=Counter(words)
          4 tf=[]
          5 for word,freq in frequency.items():
          6     print(word,"-",freq)
          7     tf.append(freq)
          8 print(tf)

support - 263
person - 220
service - 94
provider - 94
Competitor - 528
offered - 217
data - 156
higher - 100
download - 100
speeds - 100
expertise - 43
online - 31
made - 311
better - 624
offer - 311
Limited - 37
range - 37
services - 37
```

Here, we get the Term Frequency of the each words in Churn_Reason.

Finding the DF (document frequency) of each words :

```
y=[]
for i in range(len(x)):
    unique_words = x[i]

    word_df = {word: [] for word in x}

    for word in x:

        for reason in lowercase_unique_reasons:
            if word in reason:
                word_df[word].append(1)

    word_count = {word: sum(counts) for word, counts in word_df.items()}

print("Word Document Frequency:")
for word, freq in word_count.items():
    z=f"{word}: {freq}"
    y.append(z)

print(y)

print(type(y))
```

```
Word Document Frequency:
['affordable: 1', 'better: 2', 'charges: 2', 'competitor: 4', 'data: 2', 'deceased: 1', 'devices: 1', 'dissatisfaction: 2', 'di
stance: 1', 'download: 2', 'expertise: 2', 'high: 2', 'higher: 1', 'know: 1', 'limited: 1', 'long: 1', 'made: 1', 'moved: 1',
'offer: 3', 'offered: 2', 'online: 1', 'person: 1', 'phone: 1', 'product: 1', 'provider: 1', 'range: 1', 'reliability: 1', 'sel
f: 1', 'service: 4', 'services: 1', 'speed: 2', 'speeds: 1', 'support: 3', 'upload: 1']
<class 'list'>
```

We got the DF of each word in Chrun_Reason.

Finding the IDF (inverse document frequency) of each words:

```
import math

def calculate_idf(term, documents):
    num_documents = len(documents)
    term_appearance_count = sum(1 for doc in documents if term in doc)

    if term_appearance_count == 0:
        return 0

    return math.log(num_documents / term_appearance_count)
IDF=[]
for term in x:
    idf = calculate_idf(term, lowercase_unique_reasons)
    print(f"IDF of '{term}': {idf}")
    IDF.append(idf)
#print(IDF)
```

```
IDF of 'affordable': 2.995732273553991
IDF of 'better': 2.302585092994046
IDF of 'charges': 2.302585092994046
IDF of 'competitor': 1.6094379124341003
IDF of 'data': 2.302585092994046
IDF of 'deceased': 2.995732273553991
IDF of 'devices': 2.995732273553991
IDF of 'dissatisfaction': 2.302585092994046
IDF of 'distance': 2.995732273553991
IDF of 'download': 2.302585092994046
IDF of 'expertise': 2.302585092994046
IDF of 'high': 2.302585092994046
IDF of 'higher': 2.995732273553991
```

We got IDF of each word using a formula [log TF/n].

Finding the weight of each words:

In [74]:
```
result=[]
for i,j in zip(tf,IDF):
    result.append(i*j)
#print(result)

for term,weight in zip(x,result):
        print(f"Weight of '{term}': {weight}")
```

```
Weight of 'affordable': 787.8775879446996
Weight of 'better': 506.5687204586901
Weight of 'charges': 216.4429987414403
Weight of 'competitor': 151.2871637688054
Weight of 'data': 1215.7649291008563
Weight of 'deceased': 650.073903361216
Weight of 'devices': 467.3342346744226
Weight of 'dissatisfaction': 230.25850929940458
Weight of 'distance': 299.57322735539907
Weight of 'download': 230.25850929940458
Weight of 'expertise': 99.01115899874398
Weight of 'high': 71.38013788281542
Weight of 'higher': 931.6727370752911
Weight of 'know': 1869.3369386976904
Weight of 'limited': 931.6727370752911
Weight of 'long': 110.84209412149767
Weight of 'made': 110.84209412149767
Weight of 'moved': 110.84209412149767
Weight of 'offer': 593.7985552692809
Weight of 'offered': 299.33606208922595
Weight of 'online': 389.4451955620188
Weight of 'person': 215.69272369588734
```

We got weight of the each word in Churn_Reason. By multiplying TF*IDF.

Finding the most given reason:

In [75]:
```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
import numpy as np
```

In [91]:
```
vectors=cv.fit_transform(df2['Churn_Reason'])

tfidf_transformer = TfidfTransformer()
tfidf_vectors = tfidf_transformer.fit_transform(vectors)
print("Weight of each reason:")
sum_of_weights = np.asarray(tfidf_vectors.sum(axis=1)).flatten()

reason_weights =[ (df2.iloc[idx]['Churn_Reason'],weight_sum) for idx, weight_sum in enumerate(sum_of_weights)]
sorted_reason_weights = sorted(reason_weights, key=lambda x: x[1], reverse=True)[:3]
for churn_reason, weight_sum in sorted_reason_weights:
    print(f"\n{churn_reason}: {weight_sum}")
```

```
Weight of each reason:

Competitor offered higher download speeds: 2.198711031088252

Competitor offered higher download speeds: 2.198711031088252

Competitor offered higher download speeds: 2.198711031088252
```

We got that [Competitor offered higher download speeds] is the top most reason for customers to churned. We got so, by adding the weights of each word in a reason.

Model building using multilayer perceptron

```
from sklearn.model_selection import train_test_split
X = df[predictors].values
y = df[target_column].values
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30)
print(X_train.shape)
print(X_test.shape)
```

```
(4930, 28)
(2113, 28)
```

```
from sklearn.neural_network import MLPClassifier

mlp = MLPClassifier(hidden_layer_sizes=(28, 28), activation='logistic', solver='adam', max_iter=500)
mlp.fit(X_train, y_train)

predict_train = mlp.predict(X_train)
predict_test = mlp.predict(X_test)
#print(predict_test)

```

```
C:\Users\ridam\anaconda3\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:1109: DataConversionWarning: A colu
mn-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\ridam\anaconda3\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:692: ConvergenceWarning: Stochastic
Optimizer: Maximum iterations (500) reached and the optimization hasn't converged yet.
  warnings.warn(
```

We use Multilayer perceptron model for the predictions of the Customer_Status.

In [84]:
```
from sklearn.metrics import classification_report,confusion_matrix
print(confusion_matrix(y_train,predict_train))
print(classification_report(y_train,predict_train))
```

```
[[ 877   87  362]
 [ 115  211    0]
 [ 225    0 3053]]
              precision    recall  f1-score   support

           0       0.72      0.66      0.69      1326
           1       0.71      0.65      0.68       326
           2       0.89      0.93      0.91      3278

    accuracy                           0.84      4930
   macro avg       0.77      0.75      0.76      4930
weighted avg       0.84      0.84      0.84      4930
```

In [85]:
```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_train, predict_train)
print("Accuracy:", accuracy)

```

```
Accuracy: 0.8399594320486815
```

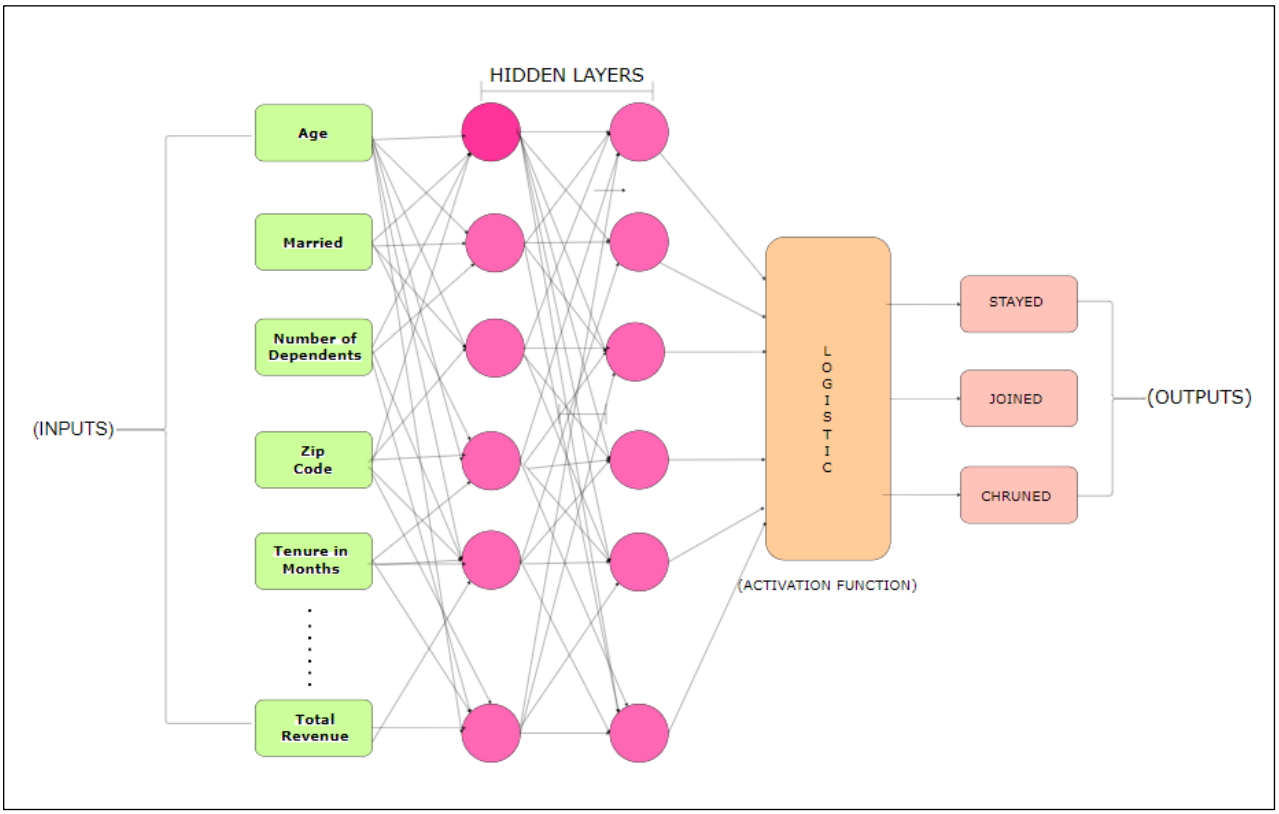
In [86]:
```
new_input = np.array([[46,0,0,93510,5,1,0,10.99,0,1,1,29,0,0,0,0,0,0,0,1,0,1,44.05,202.15,0.0,0,54.95,257.10]])
predicted_class = mlp.predict(new_input)
print(predicted_class)
```

```
[0]
```

We got 84% accuracy of the model.

And we also test the model by giving it the new input and ask it to predict the class. As a output is tells us that the customer belongs to class 0, i.e, Chruned.

Backend Working of Our Model



In this perceptron model inputs are our features we choose for model building, we take 2 hidden layers and logistics as activation function and as a output we tried to predict in which class [Stayed, Joined, Chruned] the customer will be in.

Model Extraction

Understanding Customer Churn Prediction: A Backend Journey

Data Collection:

The journey begins with raw data—a treasure trove of customer-related information. This data could come from various sources: customer databases, transaction logs, surveys, and more. Preprocessing: Before diving into analysis, we clean and prepare the data. This involves handling missing values, removing duplicates, and ensuring consistent formatting.

Noise Reduction:

Data can be noisy—contaminated with irrelevant or erroneous points. We apply techniques to cleanse the data, ensuring it's reliable for analysis.

Feature Encoding:

To feed data into machine learning models, we transform it into a suitable format. Techniques like one-hot encoding or label encoding are used for categorical variables.

Exploratory Data Analysis (EDA):

We explore the data, visualize distributions, correlations, and anomalies.
For example, we might analyze the churn behavior of different customer segments, such as unmarried vs. married customers.

NLP Model Application:

- Natural Language Processing (NLP) comes into play when dealing with text data, such as customer feedback or comments.
- A multilayer neural network with a SoftMax activation function can analyse this textual goldmine.
- DF: The number of documents containing a specific term.
- IDF: A measure of a term's importance across all documents. It's calculated by dividing the total number of documents by the number of documents containing the term and taking the logarithm.
- IDF = LOG(DF) / N, where N represents the total number of unique words.
- A multilayer neural network with a SoftMax activation function can analyse this textual goldmine.
- Update the Backend: As the field evolves, we continuously update and enhance the backed. New techniques, algorithms, and insights emerge, allowing us to refine our churn prediction mod.

Conclusion: -

- Marital status has an impact on churn, with more unmarried customers churning compared to married customers.
- Customers using Multiple Lines are less likely to churn than those not using Multiple Lines, indicating that this feature may be a retention factor.
- Age is also a factor in churn, with more customers above 41 years old churning, regardless of marital status.
- New customers are more likely to be unmarried and not using Multiple Lines, suggesting that these factors may be associated with customer acquisition.

To reduce churn, it may be beneficial to target marketing efforts towards married customers and those using Multiple Lines.

Reference: -

- Wikipedia [https://en.wikipedia.org/wiki/Churn_rate]
- research gate[https://www.researchgate.net]
- Google scholar [https://www.sciencedirect.com/science/article]