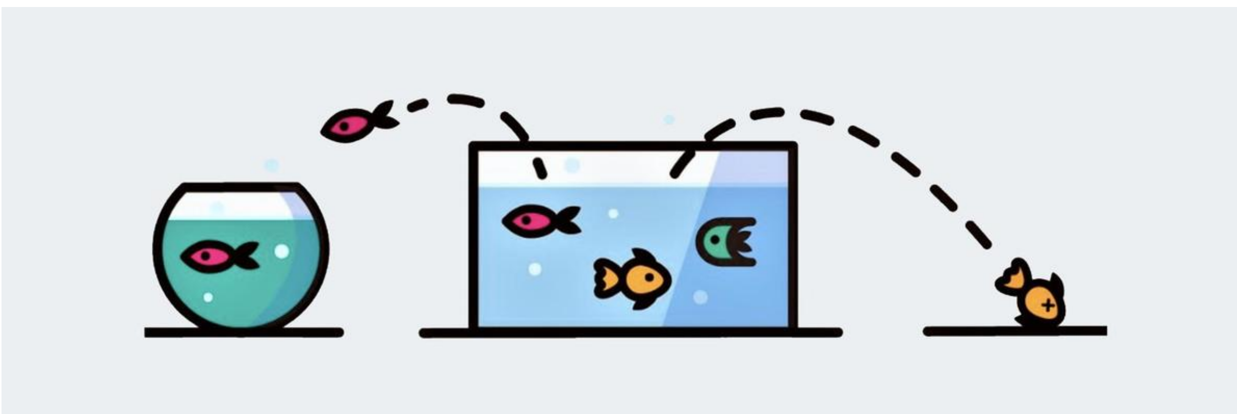
Churn Detector Report

Chatterbox Telco Pvt Ltd



CS3120 - Introduction to Data Science

Name : Laksika Tharmalingam
Index NO : 190346A
Date : 01.05.2022

Problem overview

Customer churn is the loss of customers by a business for different reasons such as poor service and better prices somewhere else. It is one of the most critical and challenging problems for telecommunication companies, credit card companies, cable service providers, etc.

CEO of Chatterbox Telecom Pvt Ltd in the Banana Republic wants to analyze this customer churn in his/her company.Finding solutions to reduce customer churn is important since it is more expensive to acquire new customers than it is to keep the ones we already have.

Therefore our objective is

- ❖ Analyze data to uncover new insights.
- ❖ Develop a dashboard that presents numerous insights into the provided dataset and enables the user to receive a prediction for a new customer.

This report describes how we preprocessed the provided dataset, as well as the insights we gleaned from the results of our analysis.

Dataset description

Chatterbox Telecom Pvt Ltd, has collected a dataset regarding the customers in their company. The objective of the dataset is to diagnostically predict whether they left Chatterbox or not. After preprocessing,there are 2312 Customer details in the train dataset. We have a test dataset also.The given train dataset consists of 19 predictor variables and one target variable and the train dataset consists of 19 predictor variables only without target variable(Churn).

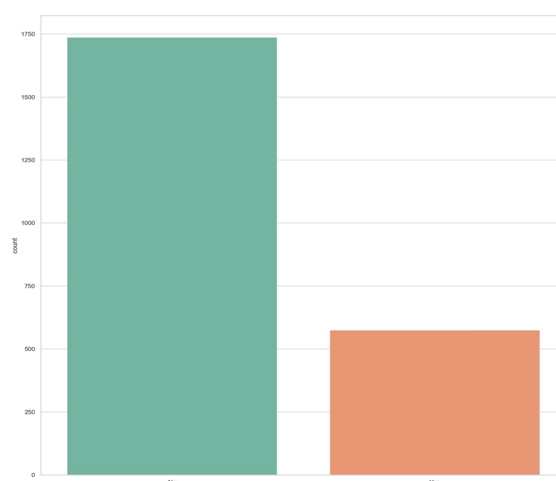
Target Variable

Churn: If the customer left or not

DataType**:** Categorical Nominal

Value ={Yes , No}

Yes: 1737 No: 575



Predictor Variables

We can see information on each predictor variable below.

Predictor Variable	**Description**	**DataType**	**Value (For CN)**
customer_id	Customer Identification Number	Categorical Nominal	
account_length	Number of months the customer has been with the current telco provider	Metric Discrete	
location_code	Customer location code	Categorical Nominal	{445,452,547}
international_plan	If the customer has international plan or not	Categorical Nominal	{yes,no}
voice_mail_plan	If the customer has voice mail plan or not	Categorical Nominal	{yes,no}
number_vm_messages	Number of voice-mail messages	Metric Discrete	
total_day_min	Total minutes of day calls	Metric Continuous	
total_day_calls	Total number of day calls	Metric Discrete	
total_day_charge	Total charge of day calls	Metric Continuous	
total_eve_min	Total minutes of evening calls	Metric Continuous	
total_eve_calls	Total number of evening calls	Metric Discrete	
total_eve_charge	Total charge of evening calls	Metric Continuous	
total_night_minutes	Total minutes of night calls	Metric Continuous	
total_night_calls	Total number of night calls	Metric Discrete	
total_night_charge	Total charge of night calls	Metric Continuous	
total_intl_minutes	Total minutes of international calls	Metric Continuous	
total_intl_calls	Total number of international calls	Metric Discrete	
total_intl_charge	Total charge of international calls	Metric Continuous	
customer_service_calls	Number of calls to customer service	Metric Discrete	

Also we can see the description about the variables on the train dataset below.

	count	mean	std	min	25%	50%	75%	max
customer_id	2321.0	2161.000000	670.159309	1001.00	1581.000	2161.00	2741.0000	3321.00
account_length	2319.0	101.400172	40.044985	1.00	74.000	101.00	127.0000	232.00
location_code	2321.0	473.470918	42.011853	445.00	445.000	452.00	452.0000	547.00
number_vm_messages	2318.0	7.557377	14.250001	-202.00	0.000	0.00	14.0000	51.00
total_day_min	2320.0	182.718103	73.332822	-179.90	144.000	180.35	221.0000	2283.90
total_day_calls	2318.0	105.324418	221.100535	-1.00	87.000	102.00	115.0000	10700.00
total_day_charge	2316.0	30.961524	9.830271	-25.60	24.480	30.60	37.5900	60.96
total_eve_min	2318.0	203.511734	115.552100	-103.30	165.925	202.40	236.4000	5186.40
total_eve_calls	2317.0	100.125162	20.536224	-80.00	87.000	101.00	114.0000	170.00
total_eve_charge	2313.0	17.123130	4.327327	0.00	14.180	17.21	20.0900	30.83
total_night_minutes	2319.0	209.543467	408.066120	23.20	167.350	201.10	235.0500	19700.00
total_night_calls	2316.0	87.641192	12.737232	33.00	79.000	90.00	98.0000	105.00
total_night_charge	2316.0	9.436710	18.656075	1.04	7.530	9.05	10.5825	900.15
total_intl_minutes	2319.0	10.247736	2.795472	-9.30	8.600	10.30	12.0000	18.30
total_intl_calls	2318.0	4.439172	2.461172	0.00	3.000	4.00	6.0000	20.00
total_intl_charge	2316.0	2.773364	0.733526	0.00	2.320	2.78	3.2400	4.94
customer_service_calls	2320.0	1.651724	1.429166	0.00	1.000	1.00	2.0000	9.00
Unnamed: 20	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Data pre-processing

We have both a train and a test dataset. However, in this section, I will discuss how I can preprocess the train dataset. As with the train dataset, the test dataset can be preprocessed as well.

Data Cleaning

Detect & Remove Duplicates

When I checked for duplicates without the customer_id column, I found four duplicate rows .Therefore I dropped those four rows.

Handling Missing Values

1. Check for null values in the dataset.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2321 entries, 0 to 2320
Data columns (total 21 columns):
customer_id               2321 non-null int64
account_length            2319 non-null float64
location_code             2321 non-null int64
intertiol_plan            2318 non-null object
voice_mail_plan           2315 non-null object
number_vm_messages        2318 non-null float64
total_day_min             2320 non-null float64
total_day_calls           2318 non-null float64
total_day_charge          2316 non-null float64
total_eve_min             2318 non-null float64
total_eve_calls           2317 non-null float64
total_eve_charge          2313 non-null float64
total_night_minutes       2319 non-null float64
total_night_calls         2316 non-null float64
total_night_charge        2316 non-null float64
total_intl_minutes        2319 non-null float64
total_intl_calls          2318 non-null float64
total_intl_charge         2316 non-null float64
customer_service_calls    2320 non-null float64
Churn                     2316 non-null object
Unnamed: 20               0 non-null float64
dtypes: float64(16), int64(2), object(3)
memory usage: 380.9+ KB
```

❖ When I checked for null values I found that Column named as “Unnamed:20” has null values for all rows.Therefore I dropped that column.

❖ After that I found that most of the columns had missing values.I used different techniques for each column to fill those

2. Fill the null values

Variable	Method of Handling missing Values
Customer_id	No null values
account_length	Filled with median values
location_code	No null values
international_plan	Filled with “no”
Voice_mail_plan	If number_vm_messages = 0 then filled with “no”. Otherwise filled with “yes”.
number_vm_messages	If Voice_mail_plan = “no” then filled with 0. Otherwise filled with median value based on Voice_mail plan = “yes” rows.
total_day_min	First sort the dataset with location code and total_day_charge. After that fill the null values with the forward fill method.
total_day_calls	First sort the dataset with total_day_min and total_day_charge. After that fill the null values with the forward fill method.
total_day_charge	First sort the dataset with location code and total_day_min. After that fill the null values with the forward fill method.
total_eve_min	First sort the dataset with location code and total_eve_charge. After that fill the null values with the forward fill method.
total_eve_calls	First sort the dataset with total_eve_min and total_eve_charge. After that fill the null values with the forward fill method.
total_eve_charge	First sort the dataset with location code and total_eve_min. After that fill the null values with the forward fill method.
total_night_minutes	First sort the dataset with location code and total_night_charge. After that fill the null values with the forward fill method.
total_night_calls	First sort the dataset with total_night_minutes and total_night_charge. After that fill the null values with the forward fill method.
total_night_charge	First sort the dataset with location code and total_night_calls. After that fill the null values with the forward fill method.
total_intl_minutes	First sort the dataset with location code and total_intl_charge. After that fill the null values with the forward fill method.
total_intl_calls	First sort the dataset with total_intl_minutes and total_intl_charge. After that fill the null values with the forward fill method.

total_intl_charge	First sort the dataset with location code and total_intl_minutes. After that fill the null values with the forward fill method.
customer_service_calls	Filled with median value.
Churn	Dropped the rows.

Handling Out-of-range values

In the dataset I found that some columns had negative values. Therefore I replace those values with the absolute values.

Handling Outliers

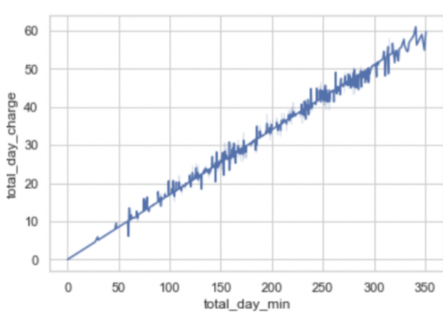
I discovered some outliers in the dataset using a pair plot. I substituted null values for them. And then I handled them similarly to how I would handle missing values.

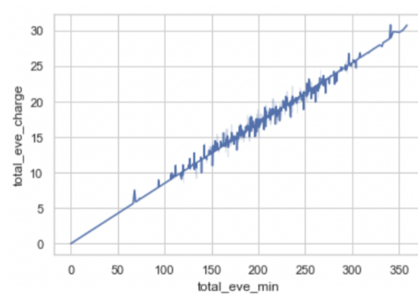
Data Transformation

There are four categorical columns in this dataset. For location_code I used one_hot_encoding. And for the other three columns (international_plan,voice_mail_plan,Churn) I used ordinal encoding.

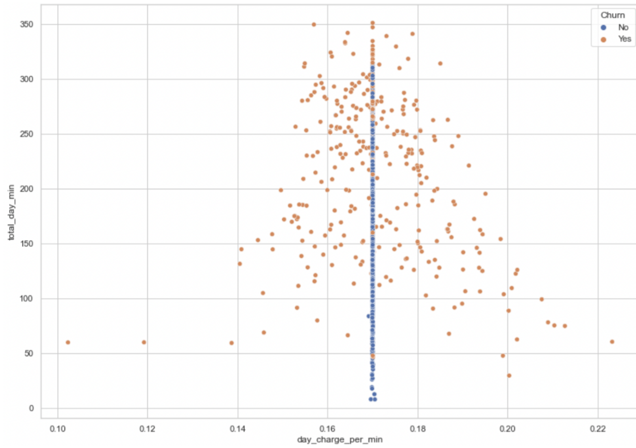
Insights from data analysis

1. total_day_min and total_day_charge have linear relationships with them. As like that, total_eve_min and total_eve_charge , total_night_minutes and total_night_charge, total_intl_min and total_intl_charge also have linear relationships with them.





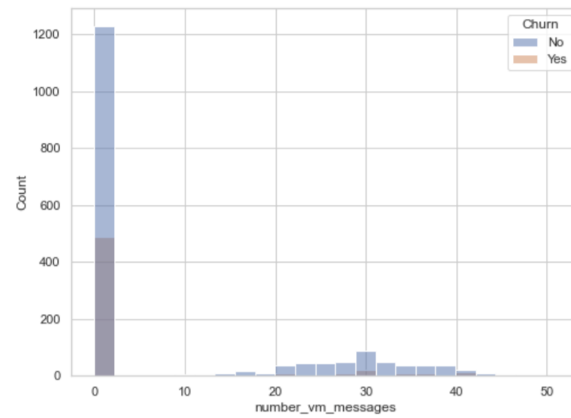
Here we can notice that some data points deviate from the linear line. But I found that the most customers who left the company are in the deviation points. So I add a new column named “day_charge_per_min” column [day_charge_per_min = total_day_charge / total_day_min] .After that I plot the graph for total_charge_per_min vs total_day_minutes.

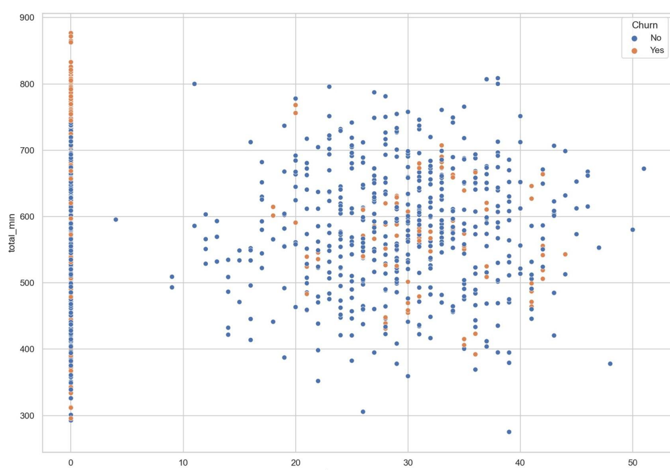


This graph shows that those who charged very low or higher for a min than an average customer have the most chance of leaving the company.

2. When I graph a counter plot for number_vm_messages, I found that most of the Churns didn't send any vm_messages.This means they don't have a voice_mail_plan.

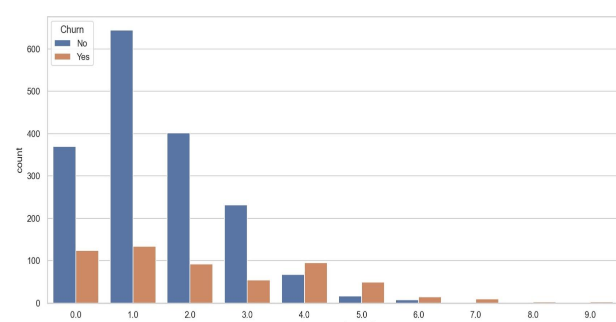
   From the idea of this graph, I graphed another graph for total_min (New column with the sum of all mins) and number_vm_messages.





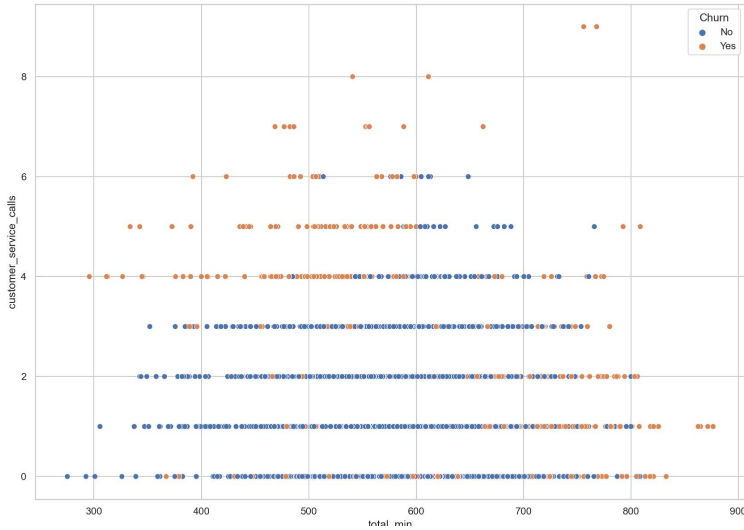
From the graph, I found that those who talk for more than 750 mins and don't send vm_messages have the most chance of leaving the company.

3. From the analysis, I can say that customers who make customer_service_calls more than 3 times have the most chance to leave the company.

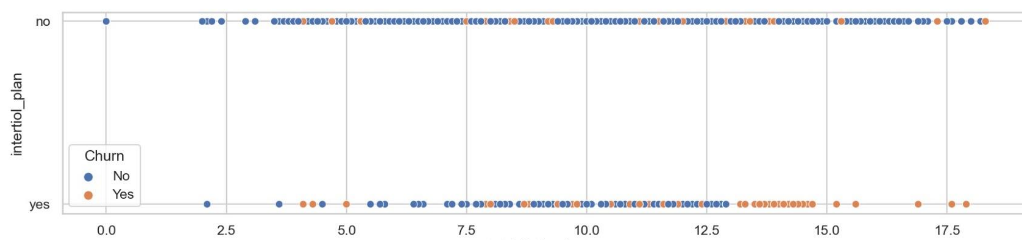


From the idea of a graph, I drew another graph for total_min vs customer service calls.

With this, we can identify those who talk less and make customer_service_calls more than 3 times have the most chance to leave the company.



4. Customers who have intl_plan and talk more intl_mins have the chance to leave the company. It may be because of the intl_plan. Improving the intl_plan may less the customers who leave the company.



5. I draw a graph between total_min vs total_day_mins. From that I found that those who talk in calls more minutes and talk more minutes in day_time have the most chance to leave the company. Making better day_time plans will reduce the number of customers becoming churns.