DataPrepreparation

March 8, 2024

1 Milestone 1: Data Prepreparation

In this file we have done the data preprocessing, we imported the raw data set here, dropped some columns and altered some data types for further useage. Lastly we converted the dataframe into a csv file and saved it in the same folder.

Before running this file we need to have the raw dataset in the same folder, or the path should be provided. This is the file which should be run first, before running any other notebook files as main useable data frame is created in the data prepreparation

[70]:
```
import pandas as pd
```

[71]:
```
df = pd.read_excel("Telco_customer_churn.xlsx")
df.head()
```

[71]:

	CustomerID	Count	Country	State	City	Zip Code	\
0	3668-QPYBK	1	United States	California	Los Angeles	90003	
1	9237-HQITU	1	United States	California	Los Angeles	90005	
2	9305-CDSKC	1	United States	California	Los Angeles	90006	
3	7892-POOKP	1	United States	California	Los Angeles	90010	
4	0280-XJGEX	1	United States	California	Los Angeles	90015	

	Lat Long	Latitude	Longitude	Gender	…	Contract	\
0	33.964131, -118.272783	33.964131	-118.272783	Male	…	Month-to-month	
1	34.059281, -118.30742	34.059281	-118.307420	Female	…	Month-to-month	
2	34.048013, -118.293953	34.048013	-118.293953	Female	…	Month-to-month	
3	34.062125, -118.315709	34.062125	-118.315709	Female	…	Month-to-month	
4	34.039224, -118.266293	34.039224	-118.266293	Male	…	Month-to-month	

	Paperless Billing	Payment Method	Monthly Charges	Total Charges	\
0	Yes	Mailed check	53.85	108.15	
1	Yes	Electronic check	70.70	151.65	
2	Yes	Electronic check	99.65	820.5	
3	Yes	Electronic check	104.80	3046.05	
4	Yes	Bank transfer (automatic)	103.70	5036.3	

	Churn Label	Churn Value	Churn Score	CLTV	Churn Reason
0	Yes	1	86	3239	Competitor made better offer

```
1          Yes           1          67  2701                            Moved
2          Yes           1          86  5372                            Moved
3          Yes           1          84  5003                            Moved
4          Yes           1          89  5340  Competitor had better devices

[5 rows x 33 columns]
```

Dropping some columns which we decided should not be used

removed location details to avoid models getting biased for the people of one region. However, we didnt had the data to represent the whole population. But we decided to still remove the location columns. count and customerID couldnt be used by a machine learning model. Lasly all the other Target variable related columns were dropped as well.

[72]:
```
columns_to_drop = ['CustomerID', 'Count', 'Country', 'State', 'City', 'Zip␣
 ↪Code', 'Lat Long',
                   'Latitude', 'Longitude', 'Dependents', 'Churn Label', 'Churn␣
 ↪Score', 'CLTV', 'Churn Reason']

# Remove leading and trailing whitespace from column names
columns_to_drop = [col.strip() for col in columns_to_drop]

# Drop the specified columns
df.drop(columns_to_drop, axis=1, inplace=True)

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 19 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Gender             7043 non-null   object
 1   Senior Citizen     7043 non-null   object
 2   Partner            7043 non-null   object
 3   Tenure Months      7043 non-null   int64
 4   Phone Service      7043 non-null   object
 5   Multiple Lines     7043 non-null   object
 6   Internet Service   7043 non-null   object
 7   Online Security    7043 non-null   object
 8   Online Backup      7043 non-null   object
 9   Device Protection  7043 non-null   object
 10  Tech Support       7043 non-null   object
 11  Streaming TV       7043 non-null   object
 12  Streaming Movies   7043 non-null   object
 13  Contract           7043 non-null   object
 14  Paperless Billing  7043 non-null   object
 15  Payment Method     7043 non-null   object
 16  Monthly Charges    7043 non-null   float64
```

```
 17  Total Charges      7043 non-null   object
 18  Churn Value        7043 non-null   int64
dtypes: float64(1), int64(2), object(16)
memory usage: 1.0+ MB
```

Total charges is a numerical column however its Dtype is object, so we decided to change that to numeric Dtype to avoid further confusion.

[73]:
```
df['Total Charges'] = pd.to_numeric(df['Total Charges'], errors='coerce')

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 19 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Gender             7043 non-null   object
 1   Senior Citizen     7043 non-null   object
 2   Partner            7043 non-null   object
 3   Tenure Months      7043 non-null   int64
 4   Phone Service      7043 non-null   object
 5   Multiple Lines     7043 non-null   object
 6   Internet Service   7043 non-null   object
 7   Online Security    7043 non-null   object
 8   Online Backup      7043 non-null   object
 9   Device Protection  7043 non-null   object
 10  Tech Support       7043 non-null   object
 11  Streaming TV       7043 non-null   object
 12  Streaming Movies   7043 non-null   object
 13  Contract           7043 non-null   object
 14  Paperless Billing  7043 non-null   object
 15  Payment Method     7043 non-null   object
 16  Monthly Charges    7043 non-null   float64
 17  Total Charges      7032 non-null   float64
 18  Churn Value        7043 non-null   int64
dtypes: float64(2), int64(2), object(15)
memory usage: 1.0+ MB
```

All the other columns apart from Monthly Charges, Total Charges, Tenure Months and Churn Value are categorical

after that, we are checking if any row have NaN values, as we previously changed Total Charges column from object to numeric data type, if there would be any rows with some alphabets in it, those will be converted to NaN.

[74]:
```
df.isnull().any()
```

[74]:
```
Gender              False
Senior Citizen      False
Partner             False
Tenure Months       False
Phone Service       False
Multiple Lines      False
Internet Service    False
Online Security     False
Online Backup       False
Device Protection   False
Tech Support        False
Streaming TV        False
Streaming Movies    False
Contract            False
Paperless Billing   False
Payment Method      False
Monthly Charges     False
Total Charges        True
Churn Value         False
dtype: bool
```

As we can see Total Charges came to be True, which means that there are possible some NaN values in the column

[75]:
```
df = df.fillna(df.mean())
```

```
/var/folders/4z/qh15q30n25nckhc10gfcfdwh0000gn/T/ipykernel_23155/114435927.py:1:
FutureWarning: Dropping of nuisance columns in DataFrame reductions (with
'numeric_only=None') is deprecated; in a future version this will raise
TypeError.  Select only valid columns before calling the reduction.
  df = df.fillna(df.mean())
```

[76]:
```
# checking again if the total charges value still have any NaN values

df.isnull().any()
```

[76]:
```
Gender              False
Senior Citizen      False
Partner             False
Tenure Months       False
Phone Service       False
Multiple Lines      False
Internet Service    False
Online Security     False
Online Backup       False
Device Protection   False
Tech Support        False
Streaming TV        False
```

```
Streaming Movies     False
Contract             False
Paperless Billing    False
Payment Method       False
Monthly Charges      False
Total Charges        False
Churn Value          False
dtype: bool
```

[77]:
```
df.to_csv('Assign1Data.csv', index=False)
```

Lastly we converted and saved the dataset into a csv file (in the same folder). The saved file is then used by the other files to run their specific codes on the new dataset.

[ ]:
```

```