Customer Churn Analysis and Prediction

Dierta Pasific - 2024

The primary goal of this analysis is to gain a comprehensive understanding of the factors that drive customer churn within a given dataset of customer information. This analysis combines data exploration, feature engineering, and machine learning techniques to achieve these objectives.

Objective

To identify key factors influencing customer churn and build a predictive model to improve customer retention strategies.

Approach

- Conducted Exploratory Data Analysis (EDA) to uncover relationships between features using correlation matrices and summary statistics.
- Trained a Random Forest Classifier to predict customer churn with a 70-30 train-test split.
- Evaluated feature importance to identify critical variables affecting churn.

Impact

This analysis highlights areas where targeted interventions, such as optimizing call plans or improving customer support, can significantly reduce churn rates.

About Dataset [Source]

- State: Customer's state of residence (string).
- Phone Number: Unique phone number assigned to the customer (integer).
- Account Length: Duration of the customer's account with the company in days (integer).
- Area Code: Area code corresponding to the customer’s phone number (integer).
- International Plan: Indicates whether the customer is subscribed to an international calling plan (0 or 1, integer).
- VoiceMail Plan: Indicates whether the customer is subscribed to a voicemail plan (0 or 1, integer).
- Number of Voicemail Messages: Number of voicemail messages the customer has received (integer).
- Total Day Minutes: Total number of minutes used for daytime calls (double).
- Total Day Calls: Total number of daytime calls made by the customer (integer).
- Total Day Charge: Total charge incurred for daytime calls (double).

- Total Evening Minutes: Total number of minutes used for evening calls (double).
- Total Evening Calls: Total number of evening calls made by the customer (integer).
- Total Evening Charge: Total charge incurred for evening calls (double).
- Total Night Minutes: Total number of minutes used for nighttime calls (double).
- Total Night Calls: Total number of nighttime calls made by the customer (integer).
- Total Night Charge: Total charge incurred for nighttime calls (double).
- Total International Minutes: Total number of minutes used for international calls (double).
- Total International Calls: Total number of international calls made by the customer (integer).
- Total International Charge: Total charge incurred for international calls (double).
- Customer Service Calls: Number of calls made by the customer to customer service (integer).
- Class: Target variable indicating whether the customer churned (0 or 1, integer).

1. Exploratory Data Analysis (EDA)

To begin the analysis, we first load the dataset into the telecom_df dataframe using the following code:

```
telecom_df = pd.read_csv('/content/drive/MyDrive/Learning/[UDM] Data Science dalam Satu Minggu/sample_data/telecom_churn.csv')
```

Next, we check the columns of the telecom_df dataframe by using the .columns function:

```
telecom_df.columns
```

```
Index(['state', 'account_length', 'area_code', 'phone_number',
       'international_plan', 'voice_mail_plan', 'number_vmail_messages',
       'total_day_minutes', 'total_day_calls', 'total_day_charge',
       'total_eve_minutes', 'total_eve_calls', 'total_eve_charge',
       'total_night_minutes', 'total_night_calls', 'total_night_charge',
       'total_intl_minutes', 'total_intl_calls', 'total_intl_charge',
       'number_customer_service_calls', 'class'],
      dtype='object')
```

To get an overview of the first few rows of the data, we use the .head() function:

```
telecom_df.head()
```

	state	account_length	area_code	phone_number	international_plan	voice_mail_plan	number_vmail_messages	total_day_minutes	total_day_calls	total_day_c
0	16	128	415	2845	0	1	25	265.1	110	
1	35	107	415	2301	0	1	26	161.6	123	
2	31	137	415	1616	0	0	0	243.4	114	
3	35	84	408	2510	1	0	0	299.4	71	
4	36	75	415	155	1	0	0	166.7	113	

5 rows × 21 columns

We can also generate descriptive statistics for the dataframe using the .describe() function, which provides key insights into the numerical data:

```
telecom_df.describe()
```

	state	account_length	area_code	phone_number	international_plan	voice_mail_plan	number_vmail_messages	total_day_minutes	total_day_calls
count	5000.00000	5000.00000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000
mean	25.99840	100.25860	436.911400	2499.500000	0.094600	0.264600	7.755200	180.288900	100.029400
std	14.80348	39.69456	42.209182	1443.520003	0.292691	0.441164	13.546393	53.894699	19.831197
min	0.00000	1.00000	408.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	13.00000	73.00000	408.000000	1249.750000	0.000000	0.000000	0.000000	143.700000	87.000000
50%	26.00000	100.00000	415.000000	2499.500000	0.000000	0.000000	0.000000	180.100000	100.000000
75%	39.00000	127.00000	415.000000	3749.250000	0.000000	1.000000	17.000000	216.200000	113.000000
max	50.00000	243.00000	510.000000	4999.000000	1.000000	1.000000	52.000000	351.500000	165.000000

8 rows × 21 columns

total_day_charge	...	total_eve_calls	total_eve_charge	total_night_minutes	total_night_calls	total_night_charge	total_intl_minutes
5000.000000	...	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000
30.649668	...	100.191000	17.054322	200.391620	99.919200	9.017732	10.261780
9.162069	...	19.826496	4.296843	50.527789	19.958686	2.273763	2.761396
0.000000	...	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
24.430000	...	87.000000	14.140000	166.900000	87.000000	7.510000	8.500000
30.620000	...	100.000000	17.090000	200.400000	100.000000	9.020000	10.300000
36.750000	...	114.000000	19.900000	234.700000	113.000000	10.560000	12.000000
59.760000	...	170.000000	30.910000	395.000000	175.000000	17.770000	20.000000

total_intl_calls	total_intl_charge	number_customer_service_calls	class
5000.000000	5000.000000	5000.000000	5000.000000
4.435200	2.771196	1.570400	0.141400
2.456788	0.745514	1.306363	0.348469
0.000000	0.000000	0.000000	0.000000
3.000000	2.300000	1.000000	0.000000
4.000000	2.780000	1.000000	0.000000
6.000000	3.240000	2.000000	0.000000
20.000000	5.400000	9.000000	1.000000

Finally, to check for any missing values in the dataset, we use the .isnull().sum() function, which will return the count of null values in each column:

```
telecom_df.isnull().sum()
```

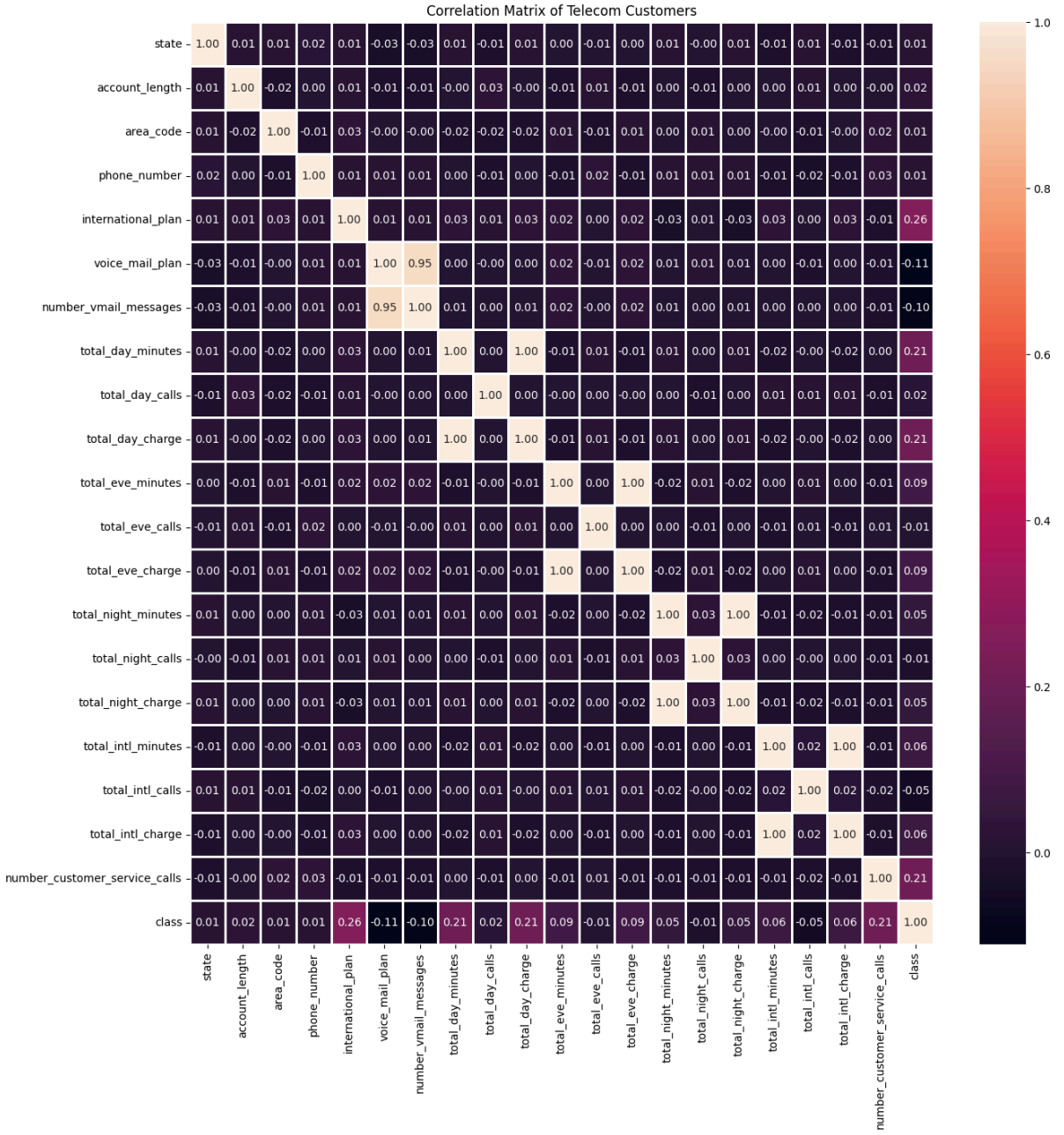
state	0
account_length	0
area_code	0
phone_number	0
international_plan	0
voice_mail_plan	0
number_vmail_messages	0
total_day_minutes	0
total_day_calls	0
total_day_charge	0
total_eve_minutes	0
total_eve_calls	0
total_eve_charge	0
total_night_minutes	0
total_night_calls	0
total_night_charge	0
total_intl_minutes	0
total_intl_calls	0
total_intl_charge	0
number_customer_service_calls	0
class	0

Now we can try to visualize our columns correlation with each other using correlation matrix using seaborn heatmap:

```
corr_matrix = telecom_df.corr()

plt.figure(figsize = (15, 15))

cm = sns.heatmap(corr_matrix, linewidths = 1, annot = True, fmt = '.2f')
plt.title('Correlation Matrix of Telecom Customers')
plt.show()
```



2. Predictive Modeling

Before training our model, we need to prepare the data by splitting it into features (X) and the target variable (y).

- X: This dataframe will contain all the independent variables (features) except for the columns we won't use in the model, such as class, area_code, and phone_number.
- y: This series will hold the target variable class, which indicates customer churn.

```
X = telecom_df.drop(['class', 'area_code', 'phone_number'], axis = 'columns')
y = telecom_df['class']
```

Using the train_test_split function from sklearn, we divide the data into training and testing sets. This allows us to train our model on one portion of the data and evaluate its performance on another.

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 150)
```

We will use the Random Forest Classifier, a popular machine learning algorithm known for its robustness and ability to handle both classification and regression tasks.

```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()
rf.fit(X_train, y_train.values.ravel())
```

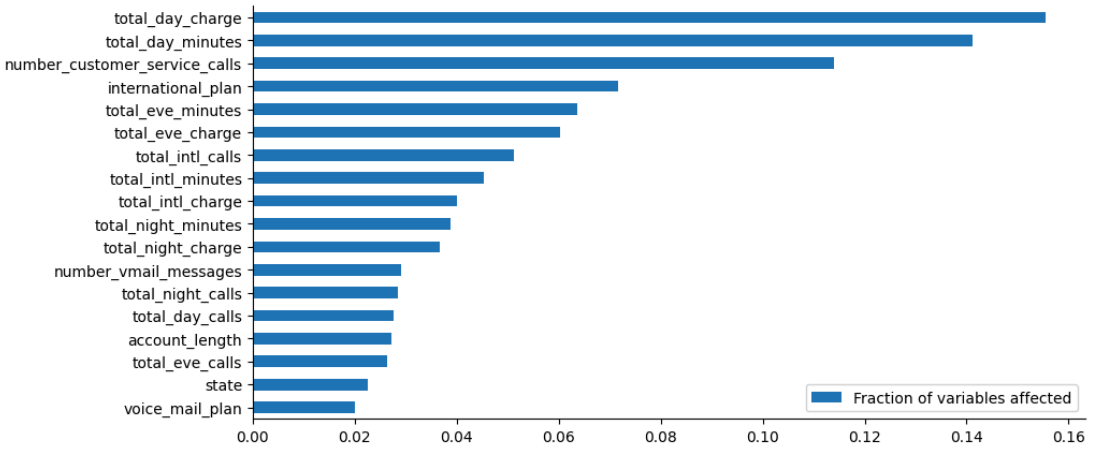
Feature importance indicates how much each feature contributes to the prediction. We can extract and visualize these importance scores:

```
rf.feature_importances_
```

```
array([0.02266921, 0.02728039, 0.07163607, 0.02014385, 0.02919192,
       0.14125408, 0.02760682, 0.1554769 , 0.06356852, 0.02628338,
       0.06030487, 0.03884839, 0.02852673, 0.03665167, 0.04538178,
       0.05121951, 0.04006776, 0.11388813])
```

Create a dataframe with feature names as the index and their corresponding importance scores as the values. Sort the dataframe by importance scores for better visualization.

```
feat_score = pd.DataFrame({"Fraction of variables affected" : rf.feature_importances_}, index = X.columns)
feat_score = feat_score.sort_values(by = "Fraction of variables affected")
feat_score.plot(kind = 'barh', figsize = (10, 5))
sns.despine()
```



This process helps us understand which features have the most influence on predicting customer churn. The visualization makes it easier to identify the most critical features.

3. Conclusion

The visualization of feature importance highlights the significance of each variable in predicting customer churn. From the analysis, we observe that the top three most influential features are:

- Total Day Charge
- Total Day Minutes
- Number of Customer Service Calls

These features play a critical role in determining whether a customer is likely to churn. For instance:

- High values in Total Day Charge & Total Day Minutes may indicate heavy usage, which could correlate with higher churn likelihood due to costs or dissatisfaction.
- An increased Number of Customer Service Calls might suggest unresolved issues, leading to customer dissatisfaction and eventual churn.