**UC Berkeley ML and AI Capstone Project**

**Telecommunication Customer Churn Analysis and Prediction**

Bowang Xiao, June 2025

Contents

[I. Problem Statement: 2](#_Toc199888227)

[II. The dataset 2](#_Toc199888228)

[III. Data Preprocessing/Preparation: 3](#_Toc199888229)

[1. Cleanup 3](#_Toc199888230)

[2. Remove Outliers 4](#_Toc199888231)

[3. Transformation 4](#_Toc199888232)

[IV. Data Understanding and visualization 4](#_Toc199888233)

[V. Correlation Analysis 6](#_Toc199888234)

[VI. Data analysis summary 6](#_Toc199888235)

[VII. Baseline Modeling: 7](#_Toc199888236)

[VIII. Model Evaluation: 7](#_Toc199888237)

[IX. Different modeling, tuning and evaluation 9](#_Toc199888238)

[X. Final Modeling findings 10](#_Toc199888239)

[XI. Next steps and recommendations 11](#_Toc199888240)

Jupyter notebook file is here: <https://github.com/jackxiao8/UC-Berkeley-AI-certificate-Capstone-Final-Report>

# Problem Statement:

A churn of “YES” is for the case when a customer left the telecommunication service within the last month. It is a lost for the telecommunication company and the company usually will offer a promotion to keep the customer when they think the customers is probably to leave.

This project is to analyze the dataset of the customer account info and predict whether the customer will stay in the program. Two objectives.

1. Analyze what factors affect the churn score significantly.
2. Build a model to predict the churn score so that the company can do something to keep the customer.

# The dataset

The dataset contains the customer account information, service type, tenue, etc. <https://www.kaggle.com/datasets/blastchar/telco-customer-churn?resource=download>

*Context*

*"Predict behavior to retain customers. You can analyze all relevant customer data and develop focused customer retention programs." [IBM Sample Data Sets] Content Each row represents a customer, each column contains customer’s attributes described on the column Metadata. The data set includes information about:*

* *Customers who left within the last month – the column is called Churn*
* *Services that each customer has signed up for – phone, multiple lines, internet, online security, online backup, device protection, tech support, and streaming TV and movies*
* *Customer account information – how long they’ve been a customer, contract, payment method, paperless billing, monthly charges, and total charges*
* *Demographic info about customers – gender, age range, and if they have partners and dependents*

**The dataset info.**

|  |
| --- |
| **churn = pd.read\_csv('data/WA\_Fn-UseC\_-Telco-Customer-Churn.csv')**  **churn.info()** |
| <class 'pandas.core.frame.DataFrame'>  RangeIndex: 7043 entries, 0 to 7042  Data columns (total 21 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----  0 customerID 7043 non-null object  1 gender 7043 non-null object  2 SeniorCitizen 7043 non-null int64  3 Partner 7043 non-null object  4 Dependents 7043 non-null object  5 tenure 7043 non-null int64  6 PhoneService 7043 non-null object  7 MultipleLines 7043 non-null object  8 InternetService 7043 non-null object  9 OnlineSecurity 7043 non-null object  10 OnlineBackup 7043 non-null object  11 DeviceProtection 7043 non-null object  12 TechSupport 7043 non-null object  13 StreamingTV 7043 non-null object  14 StreamingMovies 7043 non-null object  15 Contract 7043 non-null object  16 PaperlessBilling 7043 non-null object  17 PaymentMethod 7043 non-null object  18 MonthlyCharges 7043 non-null float64  19 TotalCharges 7043 non-null object  20 Churn 7043 non-null object  dtypes: float64(1), int64(2), object(18) |

# Data Preprocessing/Preparation:

## Cleanup

This is to remove the rows with null values, and drop the columns not related to churn score, such as the customerID. The totalCharges column data is also converted to float type.

|  |  |
| --- | --- |
| **code** | **churn['TotalCharges'] = churn['TotalCharges'].replace(' ', np.nan).astype('float')**  **## removes all rows from the churn DataFrame that contain any missing (NaN) values.**  **churn = churn.dropna()**  **## drop the customerID, which is not related to the churn score.**  **churn = churn.drop(columns='customerID')**  **churn.info()** |
| output | <class 'pandas.core.frame.DataFrame'>  Index: 7032 entries, 0 to 7042  Data columns (total 20 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----  0 gender 7032 non-null object  1 SeniorCitizen 7032 non-null int64  2 Partner 7032 non-null object  3 Dependents 7032 non-null object  4 tenure 7032 non-null int64  5 PhoneService 7032 non-null object  6 MultipleLines 7032 non-null object  7 InternetService 7032 non-null object  8 OnlineSecurity 7032 non-null object  9 OnlineBackup 7032 non-null object  10 DeviceProtection 7032 non-null object  11 TechSupport 7032 non-null object  12 StreamingTV 7032 non-null object  13 StreamingMovies 7032 non-null object  14 Contract 7032 non-null object  15 PaperlessBilling 7032 non-null object  16 PaymentMethod 7032 non-null object  17 MonthlyCharges 7032 non-null float64  18 TotalCharges 7032 non-null float64  19 Churn 7032 non-null object  dtypes: float64(2), int64(2), object(16) |

## Remove Outliers

The number data type columns are analyzed and see if there are some outliers. The zscore from scipy is used. The cretiera is abs(z\_scores) > 3

The analysis shows there is no outlier, so all the cleaned dataset is used.

|  |  |
| --- | --- |
| **code** | **from scipy.stats import zscore**  **z\_scores = churn[['tenure', 'MonthlyCharges', 'TotalCharges']].apply(zscore)**  **outliers = (abs(z\_scores) > 3)**  **outlier\_summary = outliers.sum()**  **print("Number of outliers by feature:\n", outlier\_summary)** |
| output | Number of outliers by feature:  tenure 0  MonthlyCharges 0  TotalCharges 0  dtype: int64 |

## Transformation

Use the make\_column\_transformer function on the the columns selected by selector. To these columns, apply the OneHotEncoder with drop = first. To the remainder columns, apply StandardScaler()

The dataset is splited to train data and test data using the tool, train\_test\_split

|  |  |
| --- | --- |
| **code** | **## split the data to train set and test set. set y=churn['Churn'] , Default test\_size = 0.25, random\_state = 42 or 442, both are fine.**  **X\_train, X\_test, y\_train, y\_test = train\_test\_split(churn.drop(['Churn'], axis = 1), churn['Churn'], random\_state = 442,**  **stratify = churn['Churn'])**  **selector = make\_column\_selector(dtype\_include=object)**  **transformer = make\_column\_transformer((OneHotEncoder(drop = 'first'), selector),**  **remainder = StandardScaler())**  **## select the logistic Regression as a base model with a penalty to avoid overfitting.**  **extractor = SelectFromModel(LogisticRegression(penalty='l1', solver = 'liblinear' ,random\_state = 42))** |

# Data Understanding and visualization

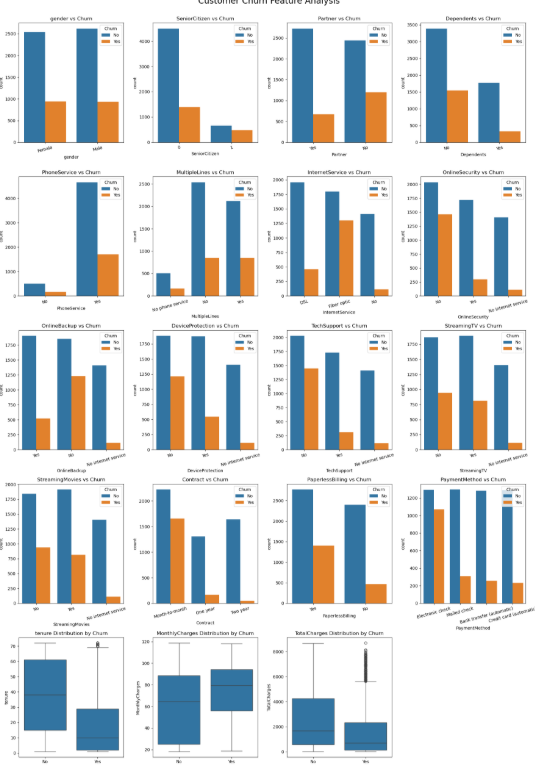
After considering the business understanding, we want to get familiar with our data.

Intuitive impression from the plots, we can see that features affect the churn score much include: SeniorCitizen, Dependents, InternetService,Contract,tenure,MonthlyCharges.

The features do not affect the churn score much include gender, Partner, StreamingMovies, StreamingTV, MultipleLines, MultipleLines, PaymentMethod, DeviceProtection,

The features affect the churn score somehow include: PhoneService, PaperlessBilling, PaymentMethod.

Please note this is an intuitive impression. A more accurate impact analysis is done in the correlation analysis.



# Correlation Analysis

This is to find out what features affect the churn score by doing a numerical correlation analysis.

|  |  |
| --- | --- |
| **code** | **# Calculate correlation matrix**  **correlation\_matrix = churn\_encoded.sample(1000).corr()**  **# Display the correlation of each feature with price**  **churn\_correlation = correlation\_matrix['Churn\_Yes'].sort\_values(ascending=False)**  **print(churn\_correlation)** |
| output | Churn\_Yes 1.000000  InternetService\_Fiber optic 0.288780  PaymentMethod\_Electronic check 0.281493  PaperlessBilling\_Yes 0.238920  MonthlyCharges 0.192502  SeniorCitizen 0.093910  StreamingTV\_Yes 0.069321  PhoneService\_Yes 0.053496  StreamingMovies\_Yes 0.031517  MultipleLines\_Yes 0.024235  gender\_Male 0.005107  DeviceProtection\_Yes -0.033535  OnlineBackup\_Yes -0.049992  MultipleLines\_No phone service -0.053496  PaymentMethod\_Mailed check -0.102890  PaymentMethod\_Credit card (automatic) -0.122459  Dependents\_Yes -0.143276  Contract\_One year -0.147222  TotalCharges -0.154829  OnlineSecurity\_Yes -0.155780  Partner\_Yes -0.157818  TechSupport\_Yes -0.183659  DeviceProtection\_No internet service -0.222510  StreamingTV\_No internet service -0.222510  OnlineBackup\_No internet service -0.222510  StreamingMovies\_No internet service -0.222510  OnlineSecurity\_No internet service -0.222510  InternetService\_No -0.222510  TechSupport\_No internet service -0.222510  Contract\_Two year -0.281193  tenure -0.319627 |

# Data analysis summary

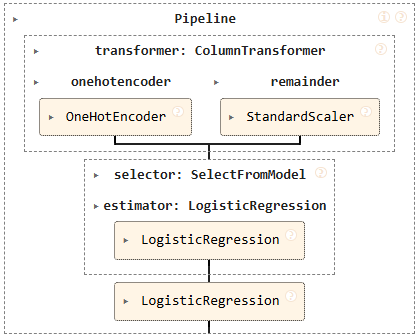
While the intuitive analysis on the plots can give us some rough ideas of each feature's importance, the correlation analysis can give us numerical evaluation of each feature influence, either positive correlation (toward churn score 1), or negative correlation (toward churn score 0).

* In general, long tenure customers, 2 year contracts will stay longer with the program.
* The customers without internet will stay longer, and on the opposite, the customers with fast fiber internet more likely to leave.
* when the monthly charges is below **$60** , churn score is low, and then increases from **$60** to **$100**. and then decrease after $100. It means the low income and high income will more likely stay.
* autopay will help keep the customers, and electronic check lead to high churn score. Surprisingly, the very traditional payment method, mailed check, lead to low churn score.
* paperbilling lead to higher churn score. The company should encourage electronic billing to keep the customers and also save mailing cost.

# Baseline Modeling:

A pipeline is created to do the data processing and modeling.

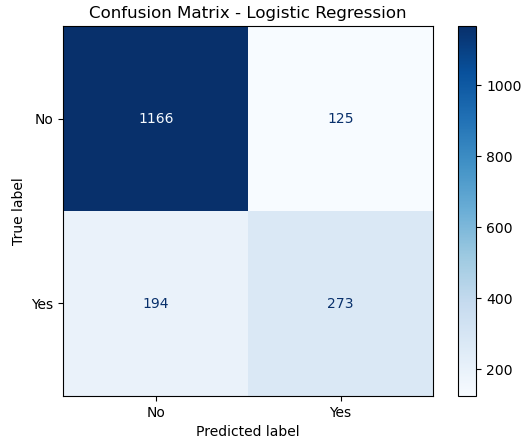
|  |  |
| --- | --- |
| **code** | **Pipeline(steps=[('transformer',**  **ColumnTransformer(remainder=StandardScaler(),**  **transformers=[('onehotencoder',**  **OneHotEncoder(drop='first'),**  **<sklearn.compose.\_column\_transformer.make\_column\_selector object at 0x000001FC7B7DFD10>)])),**  **('selector',**  **SelectFromModel(estimator=LogisticRegression(penalty='l1',**  **random\_state=42,**  **solver='liblinear'))),**  **('lgr', LogisticRegression(max\_iter=1000, random\_state=42))])** |

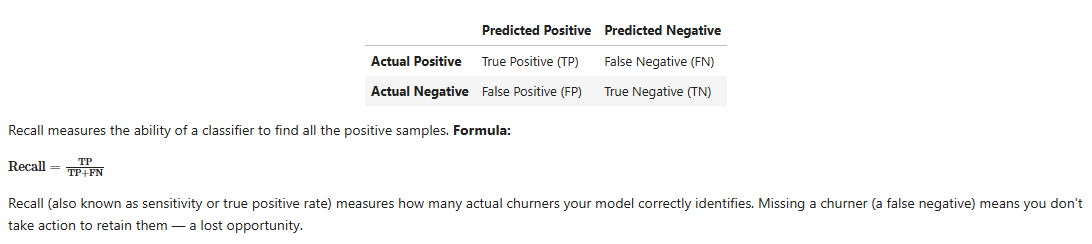


.

# Model Evaluation:

|  |  |
| --- | --- |
| **code** | **LogisticRegression\_pipe.fit(X\_train, y\_train)**  **# Accuracy scores**  **train\_acc = LogisticRegression\_pipe.score(X\_train, y\_train)**  **test\_acc = LogisticRegression\_pipe.score(X\_test, y\_test)**  **# Print accuracies**  **print(f"Train Accuracy: {train\_acc:.4f}")**  **print(f"Test Accuracy: {test\_acc:.4f}")**  **# Predict on test set**  **y\_pred = LogisticRegression\_pipe.predict(X\_test)**  **# Confusion matrix**  **cm = confusion\_matrix(y\_test, y\_pred)**  **disp = ConfusionMatrixDisplay(confusion\_matrix=cm, display\_labels=LogisticRegression\_pipe.classes\_)**  **# Plot**  **disp.plot(cmap="Blues")**  **plt.title("Confusion Matrix - Logistic Regression")**  **plt.show()** |
| output | Train Accuracy: 0.8015  Test Accuracy: 0.8185  Recall: 0.5846 |





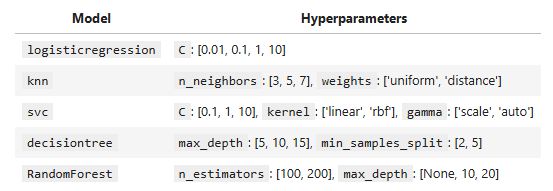
**Baseline modeling conclusion**

* The features of the churn data set are analyzed through the plots and correlation analysis. The tenure, contract length, internet service and monthly charges play important roles in the churn score.
* The data is cleaned and transformed for ML models. A Logistic Regression model is used for the intial training and prediction as the baseline. Its accuracy is about 80%.
* More models will be tuned with grid search and compared for the final report. We expect to develop a model with 95% and higher accuracy so that the telecommunication company can do some special promotions to keep the customers with high churn scores.

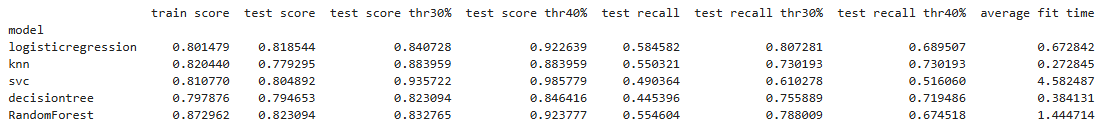
# Different modeling, tuning and evaluation

Since the baseline does not provide accurate enough prediction accuracy and recall score, different models are trained and the model parameters are searched for best accuracy.

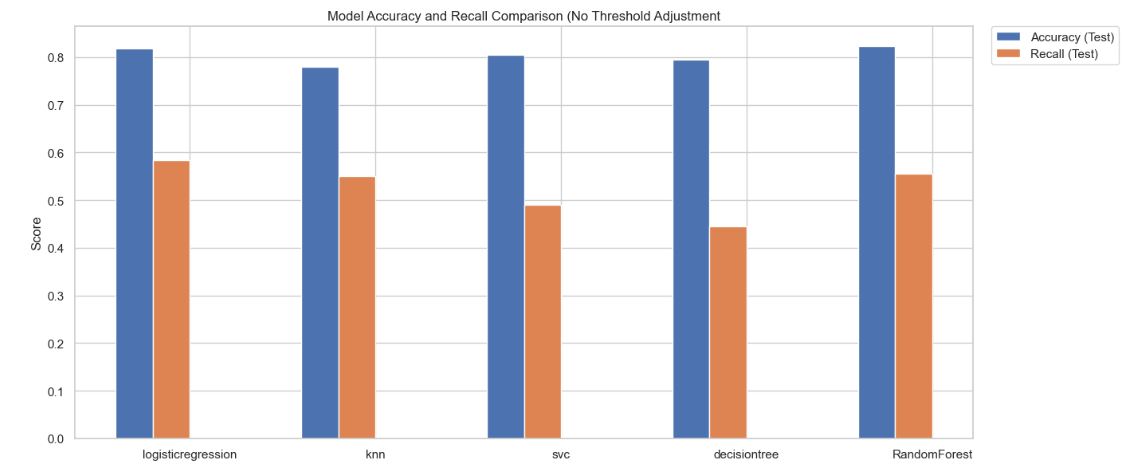
Total 5 models are chosen for the model prediction. Their parameters are tabulated below.



The model accuracy and recall results.

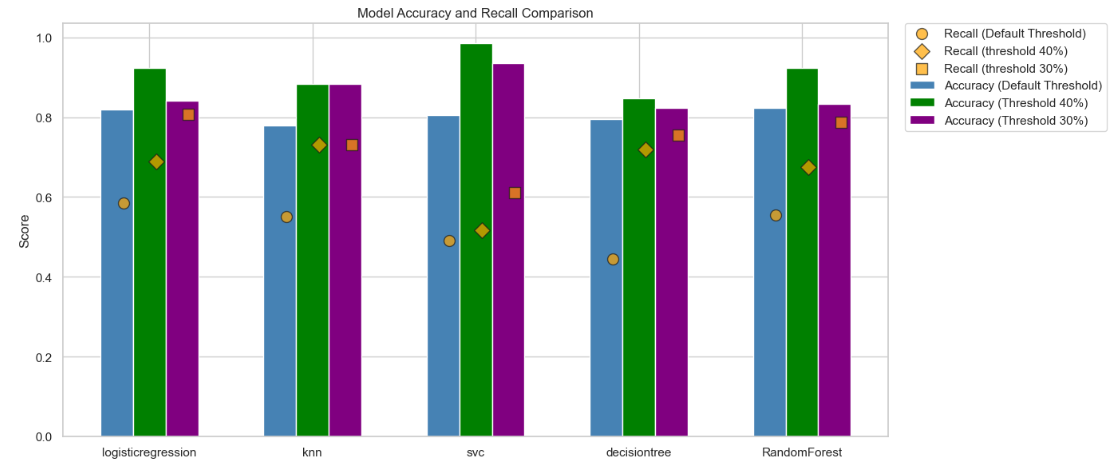


The models are optimized for their parameters. Surprisingly, the model accuracy is similar, around 80%, and the recall scores are all too low. The desired recall score is 75%.



Since the default probably threshold (Churn=’YES’ when probability >=50%) does not give good recall score. In order to get a higher recall score, the threshold is lowed. Two lower thresholds, 30% and 40% are compared.

The accuracy and recall scores for the test data are plot as below.



The computation time for each model is plot as below.



# Final Modeling findings

* The entire churn data sets are split to train data and test data.
* 5 models including logistic regression, knn, svc, decision tree, and random forest are trained based on the train data, and then to predict results for testing data. Combinations of parameters are searched for the best model parameters, by the tool, GridSearchCV. Model prediction accuracy and recall scores are compared.
* It is noted that even with optimized model parameters, there is no significant improvement from the baseline model. For example, the model accuracy is about 80%, and the recall scores range from 44% to 58%. Since the company want to keep as many customers as possible, a recall score higher than 75% is desired.
* Because the 5 models with optimized parameters trained with this small data set cannot give required recall score, lower thresholds for churn is used. By default, when the probabilities is higher than 50%, the churn score is considered as "YES". I lowered probabilities to 30% or 40% .
* Both lower thresholds give better accuracy and recall scores. While 40% threshold give best accuracy, the 30% threshold gives the highest recall score, which is desired. So it is recommended to use 30% threshold for future model prediction.
* Both logistic regression and random forest give the highest recall score (around 80%), while svc model gives the highest accuracy (98% for 40% threshold). Since high recall score is our goal, logistic regression and random forest are chosen for future model prediction.
* The svc model takes significantly longer computation time than the other 4 models. The logistic regression and random forest give desired recall score, but the logistic regression model take even less time than the random forest model.

# Next steps and recommendations

* The logistic regression model is recommended for the churn score prediction, based on its high recall score, and less computation time.
* The optimized model parameters for logistic regress model is {'logisticregression\_\_C': 1}.
* Lower threshold (30% probably to be consider churn='YES') is recommended.
* Next step is to try the model with large data set, and also more features. It is expected the model accuracy and recall scores will be better.