Survival Analysis on Telco Customer Churn

https://github.com/stevesong1121/Springboard-Course-Capstone-Project-1

Steve Song
May 11,2020

Executive Summary

Customer churn, defined as the percentage of customers that stop using a company's products or services, is one of the most import matrices for a business, as it usually costs more to acquire new customers than it does to retain existing ones.

The dataset of Telco customer churn was acquired from Kaggle competition website. The client is the management of the pertaining business. The objective of this project is to predict if customers are likely to stop doing business and when that event might happen. With the model built and data analysis performed, the business owners will be able to segment the customers based on the likelihood to churn and take appropriate actions to prevent that happening, such us launching customer retention program.

To perform survival analysis on such a time-event problem, we will treat the features of “tenure” and “churn” as target variable. During the process of data wrangling, we explored how each variable is distributed individually and how they are correlated with each other. Since the most of variables are of categorical type, we used Cramer'V to explore correlation for categorical variables, whereas Pearson correlation for numerical ones.

There are multiple algorithms built in Scikit-survival to perform survival analysis. First of all, we used the non-parametric method Kaplan–Meier estimator to find out how “tenure” affects the likelihood of customer churn by plotting survival function curve. However, semi-parametric method Cox Proportional Hazard is more powerful to predict whether a customer churn based on not only time factor but also other attributes. In order to achieve better performance, we applied feature selection techniques to figure out the importance of each variables in building the predictive model. L1-based feature selection CoxnetSurvivalAnalysis and univariate feature selection using SelectKBest were used in this project.

Furthermore, some other advanced algorithms such as survival tree, survival random forest, survival support vector machine, survival kernel support vector machine and gradient boosting survival analysis were also be explored in this project.

1 - Introduction

Survival Analysis is a set of statistical tools, which addresses questions such as 'how long would it be, before a particular event occurs'; in other words, we can also call it as a 'time to event' analysis. This technique is called survival analysis because this method was primarily developed by medical researchers and they were more interested in finding expected lifetime of patients in different cohorts. This analysis can be further applied to not just traditional death events, but to many different types of events of interest in different business domains. With the help of survival analysis, we can focus on churn prevention efforts of high-value customers with low survival time.

2 - Dataset

2.1 - Dataset acquisition

The dataset is for Telco Customer Churn, downloaded from Kaggle competition website. This dataset contains 7,043 unique observations in total, each of which represents a customer. Each column contains customer's 21 attributes. All entries have a column stating if the customer has churned or not.

The dataset includes the following information:

- 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

As for Total Charges, the fact is that we have no way to know how much it is at the time of prediction. To avoid the effect of data leakage, we will discard this variable in model building.

2.2 - Data wrangling strategy

Data wrangling is the terminology used to describe the process of transforming raw data to a clean and organized format ready for use. The most common data

structure used to “wrangle” data is the data frame. For this particular project, we imported the data from CSV file by using Pandas. We will take the following steps to perform data wrangling.

- Describing the data

The first step is to view some characteristics of a DataFrame. One of the easiest things we can do after loading the data is to view the first few rows using df.head(). We can also take a look at the number of rows and column using df.shape() attribute of the DataFrame. By using df.info attribute or df.dtypes attribute, we are able to view data type of each variables.

For numerical variables, we can use df.describe() method to get descriptive statistics. Pandas offers variance (var), standard deviation (std), kurtosis (kurt), skewness (skew), standard error of the mean (sem), mode (mode), median (median), and a number of others. Pandas also provides a large set of summary functions that operate on different kinds of Pandas objects (DataFrame columns, Series, GroupBy) and produce single values for each of the groups. When applied to a DataFrame, the result is returned as a pandas Series for each column. For categorical variables, both df.unique() and df.value_counts() are useful for manipulating and exploring the data.

- Dropping duplicate rows

The method df.drop_duplicate() can help perform this task.

Now we have determined the date type of each variables. For numerical features, the scaling matters for some machine learning algorithms. By checking the descriptive statistics, we can decide whether the data standardization is needed to build the model.

To detect missing data, the functions df.isnull() and df.notnull() return booleans indicating whether a value is missing. If missing data exist, we need to figure out its size and randomness. If the size is minimal and not highly dependent on the target variable, we can probably remove them. Otherwise, we should apply different imputation strategy such as mean or median. To detect outliers, a box-and-whisker or histogram plot helps in this regard. Numpy .percentile() command is also an effective tool to identify the extreme observations and define the upper and lower boundaries. The interquartile range (IQR) method can be used to fix outliers.

Once we have detected the missing values for categorical features, we can apply frequent category imputation strategy, replacing all occurrences of missing values with the most frequent value. Alternatively, we can treat missing data as an additional label or category of the variable so that the importance of 'missingness' can be captured. By using df.value_counts() and df.unique() methods, we can find out the cardinality of a categorical variable and rare labels if possible. Both high cardinality and rare labels may cause overfitting. Removing them can help improve machine learning performance.

3 - Exploratory Data Analysis (EDA)

The goal of Exploratory Data Analysis is to get comfortable with our data. We will do bivariate analysis and examine how each variable relates to the churn rate.

The first question is what is the distribution of those three numerical variables and how would you explain them? Probability density plots can be used to look at the distribution of the variable.

From *figure 3.1* below, we can see that the distribution of total charges is very positively skewed. The majority of customers are charged under $2000. As for the distribution of monthly charges, above $30 and beyond, it distributes nearly normal. However low-end customers with less than $30 accounts for a large portion.Even though the distribution of TotalCharges is positively skewed and the distribution of MonthlyCharges has a great number of observations in the low of monthly charges (not distributed normally), we wouldn't treat them as outliers according the business sense, since we will apply survival analysis in this case. The distribution of tenure shows a relatively stable trend between 10 months and 60 months. However, significant number of customers stay with business below 10 months (new customers), or above 60 months (loyal customers).

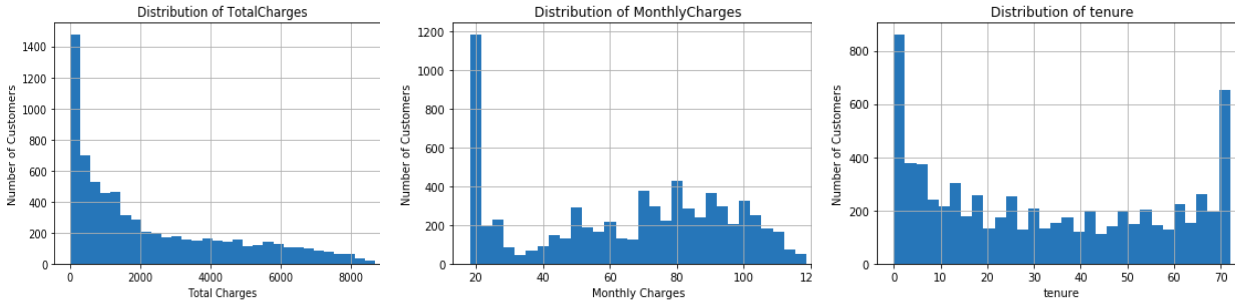

figure 3.1 Histogram of numerical variables

For categorical features, we can use frequency table or bar plots which will calculate the number of each category in a particular variable. From a business perspective, we ignore the first feature customerID since its value has nothing to do with customers churn. For all other categorical features, the largest number of different labels is 4. The cardinality is not high, so that it is not likely cause overfitting. As for the relatively rare labels, we found that 9% for 'No' values of PhoneService, and 9% for 'No phone service' values of MultipleLines. This is also unlikely to cause overfitting.

Since the majority of predictors is categorical, Pearson correlation is not applicable in this case. We use Cramer'V method to convert the categorical correlations (chi square estimates) within the range [0.1]. (see *figure 3.2*)

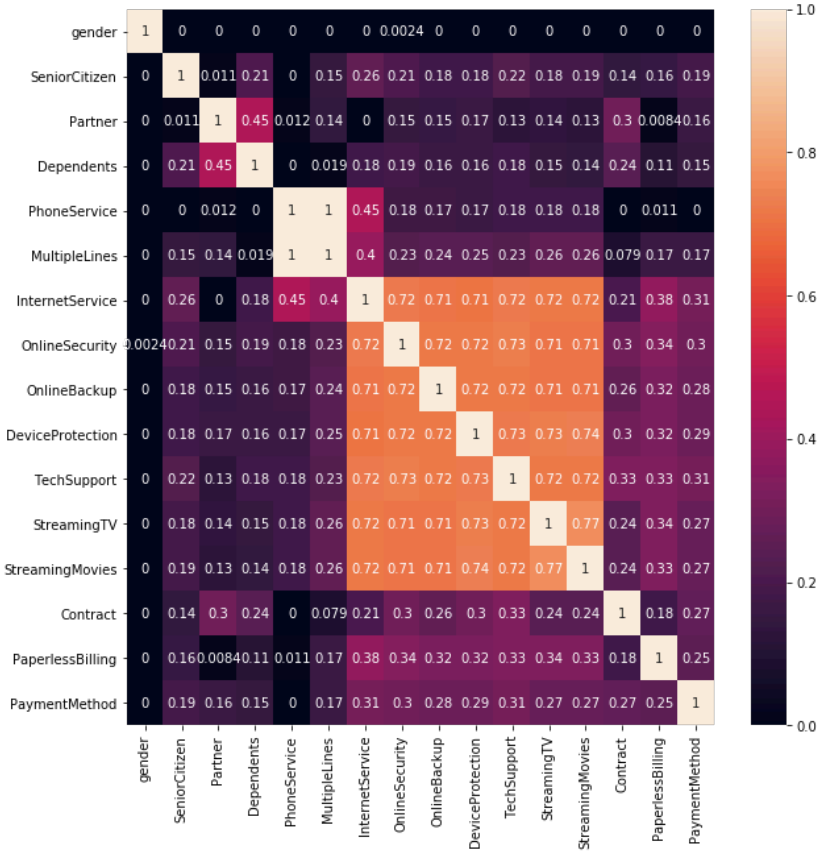

figure 3.2 Cramer'V correlation of categorical variables

From the diagram above, we can see InternetService is relatively highly related with other features: OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, Streaming and StreamingMovies.

4 - Statistical Data Analysis

After we completed data wrangling and data visualization, it is time to use statistical inference tool to explore the relationship between independent variables and dependent variable. For this particular project, we are going to perform survival analysis to predict telecom customers churn. In this context, “tenure” and “churn” will be treated as time-event variables. During the process of data wrangling, we found this dataset has 16 categorical variables and 3 numerical variables that can be used to build model.

First of all, we divide all observations into two groups using ‘churn’, i.e. churned and non-churned. We are interested to know how the numerical variables’ mean differ between these two subgroups. How would we explain them statistically? The result of t-test indicates that statistically significance does exist since p-values are all extremely low. We can confidently reject the null hypothesis and conclude that all these numerical variables differ between two groups.(t stats and p values are listed below)

```
Total Charges:
Ttest Result(statistic=-16.53673801593631, pvalue=2.706645606888261e-60)

Monthly Charges:
Ttest Result(statistic=16.978779727124437, pvalue=2.127211613240394e-63)

Tenure:
Ttest Result(statistic=31.57955051135377, pvalue=7.99905796059022e-205)
```

In survival analysis, we should treat variable 'tenure' as time factor that may bring censoring issue. Since the majority of the features are categorical, after we had an idea about cardinality, we are more interested in how each labeled group affect the performance of time-event in the context of survival analysis. Rather than using traditional chi-square contingence table, we use Log rank test together with Kaplan-Meier curve to figure out how those groups differ from each other. The analysis indicates that out the 16 categorical variables, "gender" ,”MultipleLines” and “Phone Service" have large value of p-values. We don't have sufficient evidence to reject null hypothesis for them. (see *figure* 4.1)

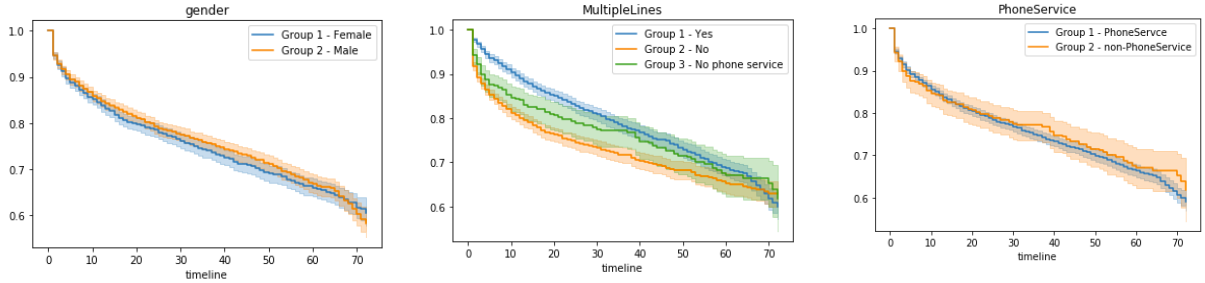

figure 4.1 Kaplan-Meier curves

The statics of other variables is statistically significant. They are more important to predict the customer churn.(see *figure* 4.2)

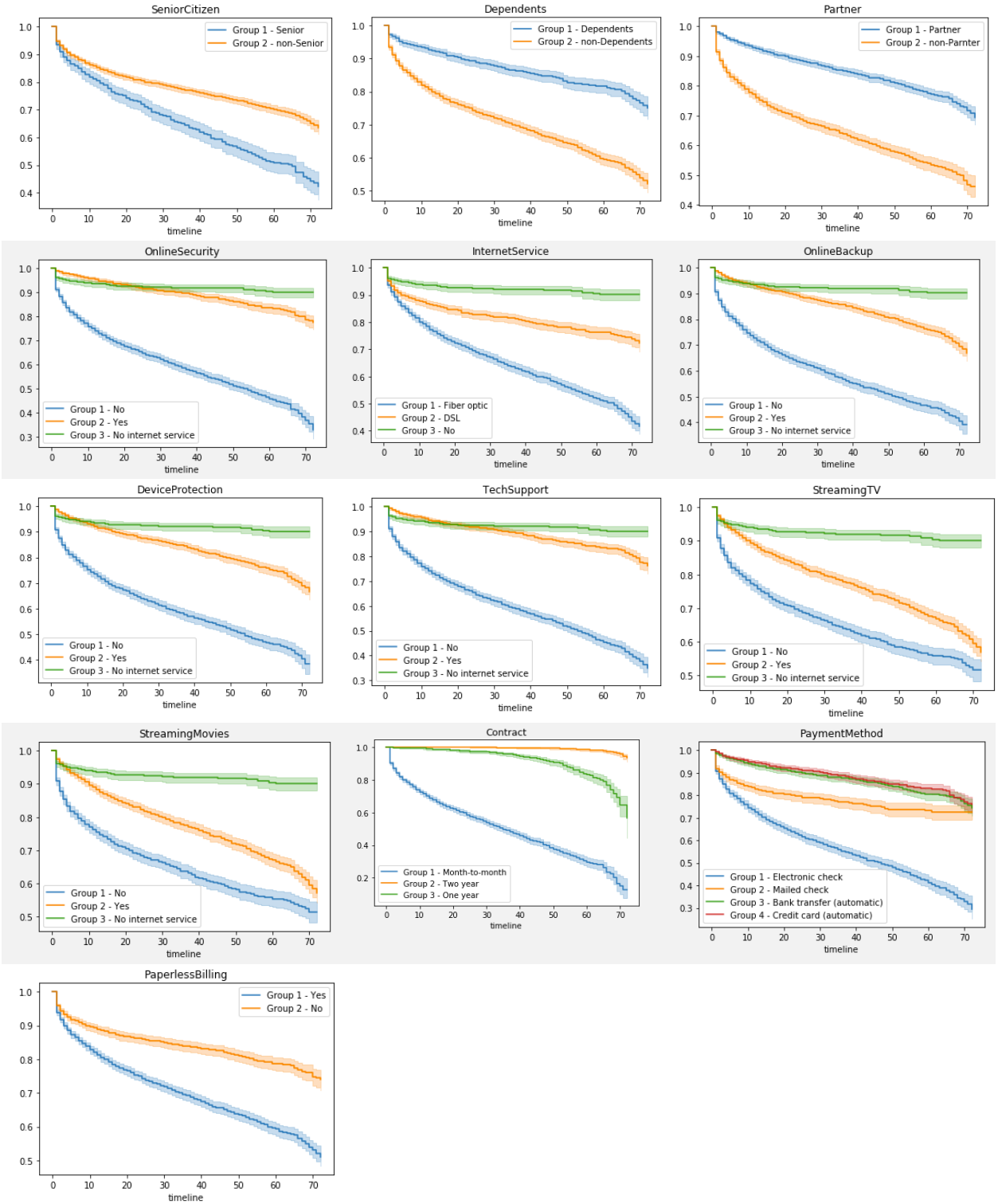

figure 4.2 Kaplan-Meier curves

5 - Modelling

5.1 - Kaplan-Meier estimator:

Kaplan-Meier estimator is a non-parametric statistic used to estimate the survival function from lifetime data. For Telco customer churn dataset, the Kaplan-Meier curve is plotted below. (see *figure* 5.1)

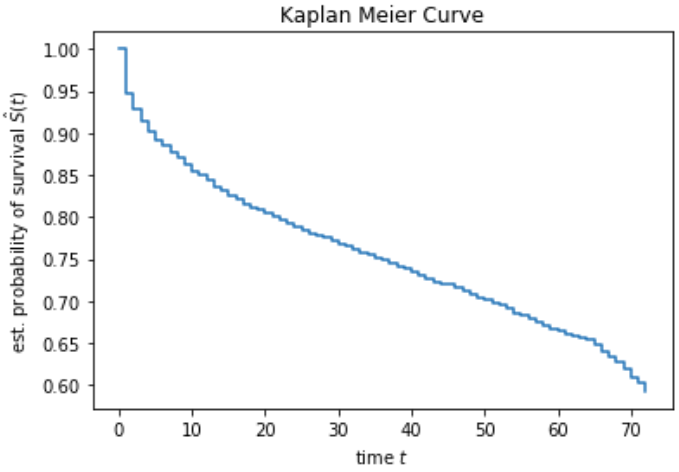

figure 5.1 Kaplan-Meier estimator

The trend of this survival function clearly shows that during the first three months of tenure, the event of churn occurred rapidly; Afterwards, the number of survivals declined gradually. In the late stage of month 65 and month 72, a significant number of customers got churned.

5.2 - Cox Proportional Hazards model

The formula for the Cox Proportional Hazards Regression Model is given as follows:

$$\overbrace{h(t|x)}^{\text{hazard}} = \overbrace{b_0(t)}^{\text{baseline hazard}} \overbrace{\exp\left(\sum_{i=1}^{n} b_i(x_i)\right)}^{\text{partial hazard}}$$

βo(t) is the baseline hazard function and it is defined as the probability of experiencing the event of interest when all other covariates equal zero. And It is the only time-dependent component in the model.

The model works such that the log-hazard of an individual subject is a linear function of their static covariates and a population-level baseline hazard function that changes over time.

After encoding on categorical variables, we applied the Cox Proportional Hazard regression model. The curve of baseline survival function is showed below. (see *figure* 5.2)

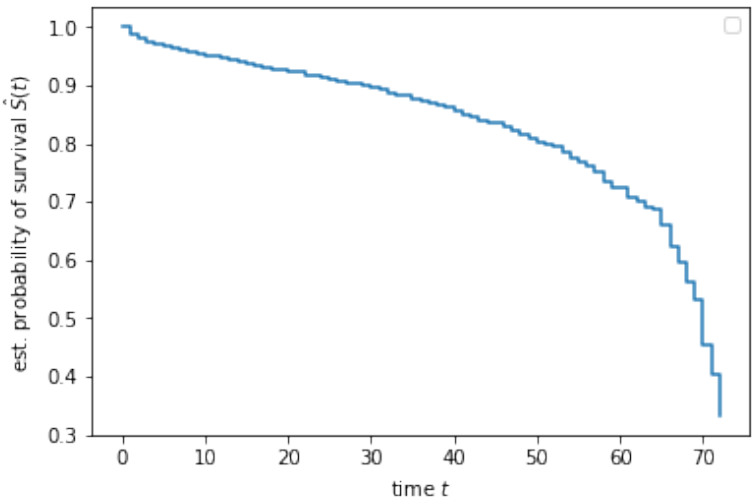

figure 5.2: baseline survival function

5.3 - Model Performance Evaluation

5.3.1 concordance index censored

C-index represents the model's ability to correctly provide a reliable ranking of the survival times based on the individual risk scores. In general, when the C-index is close to 1, the model has an almost perfect discriminatory power; but if it is close to 0.5, it has no ability to discriminate between low and high risk subjects.
The C-index on test dataset is 0.8581.

5.3.2 concordance index ipcw

Concordance index censored is too optimistic with increasing amount of censoring. Concordance index ipcw was invented to address this issue.
The concordance index ipcw on test dataset is 0.8593

5.3.3 time-dependent AUC

If specific time range is of primary interest, for example, we want to predict what is the churn rate within two years, time-dependent AUC is a good measurement to use. The time-dependent AUC on test dataset is 0.9192. The AUC curve is showed below. (see figure 5.3).

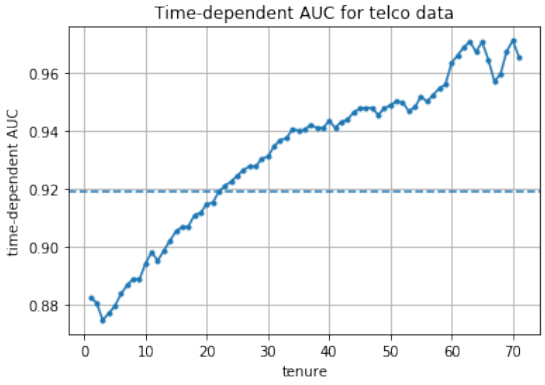

figure 5.3 time-dependent AUC

6 – Feature Selection

6.1 - L1-based feature selection using CoxnetSurvivalAnalysis

All the features don't contribute to the model equally. We need to apply feature selection techniques to find out which ones are more important. Elastic-net based model is built in the package. By setting the hyper parameter l1-ratio as 1, we can force the coefficients of some unimportant features to be zero. (see *table* 6.1). Along the regularization path of alpha values, the method CoxNetSurvivalAnalysis can generate multiple set of feature coefficients for each alpha. We tested each alpha individually and found alpha=0.000028 works best to achieve the largest concordance index score on test dataset. At the same time, the coefficients of features "MultipleLines=No phone service" and "MonthlyCharges" are trained to be zero. In other words, we can ignore these two features to re-fit the model. This methodology gave us the following performance measurements:
Concordance_index_censored: 0.8672;
Time-dependent AUC: 0.9194.

gender=Male	-4.206920
SeniorCitizen=1	-0.405074
Partner=Yes	-18.105150
Dependents=Yes	-0.671618
PhoneService=Yes	1.759813
MultipleLines=No phone service	0.000000
MultipleLines=Yes	-14.867965
InternetService=Fiber optic	12.030952
InternetService=No	-30.198872
OnlineSecurity=Yes	-20.501551
OnlineBackup=Yes	-19.841433
DeviceProtection=Yes	-11.475036
TechSupport=Yes	-13.025201

StreamingTV=Yes	-1.796792
StreamingMovies=Yes	-4.290482
Contract=One year	-47.873054
Contract=Two year	-93.754419
PaperlessBilling=Yes	6.425613
PaymentMethod=Credit card (automatic)	-5.250208
PaymentMethod=Electronic check	18.492247
PaymentMethod=Mailed check	10.447915
MonthlyCharges	0.000000

table 6.1 variable coefficients

6.2 - Univariate feature selection using SelectKBest

Another technique to select import features is univariate feature selections togeth er with SelectKBest method built in Scikit-learn. There are 22 predictors in training dataset. We need to figure out how many and which features perform best in term s of concordance index on test dataset. This approach suggests the feature 'Phone Service=Yes' and MultipleLines=No phone service' are the least important to build the model. Excluding these two features we rebuilt Cox's proportional hazard the model and got the performance:

Concordance_index_censored: 0.8581;

Time-dependent AUC: 0.9190.

7 – Survival Tree

We applied the algorithms of survival tree and random survival forest to build the model. To avoid the overfitting, we need to figure out the max_depth of the survival tree. Looping on the range from one to number of features, we found the optimal depth of survival tree is 6 since it gave us the best performance on test dataset. (see figure 7.1)

Figure 7.1: training score vs test score

This is the performance we achieved from survival tree:
concordance_index_censored: 0.8375; time-dependent AUC: 0.8951.

8 – Other Survival Analysis Algorithms Comparison

Scikit-Survival also provides some other parametric algorithms such as survival support vector machine, survival kernel support vector machine and gradient boosting survival analysis. We also applied them to this project. Below is the performance comparison.

model	concordance_index_censored	concordance_index_ipcw	time-dependent AUC
Cox PH	0.8581	0.8593	0.9192
Cox PH with l1 feature selection	0.8674	0.8334	0.9194
Cox PH with SelectKBest feature selection	0.8581	0.8540	0.9190
survival tree	0.8350	0.8436	0.9026
random survival forest	0.8375	0.9159	0.8951
Survival SVM	0.8623	0.7715	0.9273
Survival Kernel SVM	0.8440	0.8332	0.9117
Gradient boosting Survival analysis	0.8474	0.6837	0.9106

In conclusion, all of these models work well except Gradient Boosting Survival Analysis if evaluated by concordance_index_ipcw. If the churn rate over specific time range is of primary interest, Survival SVM outperforms other models with the highest time-dependent AUC score.

9 – Future Work

Cox's proportional hazards model is by far the most popular survival model, because once trained, it is easy to interpret. However, if prediction performance is the main objective, more sophisticated, non-linear or ensemble models might lead to better results. All the work we have done so far is based on supervised learning. In future, we can use any unsupervised pre-processing method available with scikit-learn, for instance, dimensionality reduction using Non-Negative Matrix Factorization (NMF), before training a Cox model.