Customer Survival Analysis

Importing Libraries

In [1]:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import norm
import statsmodels.api as st
from sklearn.preprocessing import LabelEncoder
labelencoder = LabelEncoder()

#Lifelines is a survival analysis package
from lifelines import KaplanMeierFitter
from lifelines.statistics import multivariate_logrank_test
from lifelines.statistics import logrank_test
from lifelines import CoxPHFitter
```

Data Preparation

In [2]:

```python
df = pd.read_csv("C:/Data/Telco-Customer-Churn.csv")
df.head()
```

Out[2]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	Mul
0	7590- VHVEG	Female	0	Yes	No	1	No	
1	5575-	Male	0	No	No	34	Yes	
2	3668- QPYBK	Male	0	No	No	2	Yes	
3	7795-	Male	0	No	No	45	No	
4	9237- HQITU	Female	0	No	No	2	Yes	

5 rows × 21 columns

In [3]:

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
customerID          7043 non-null object
gender              7043 non-null object
SeniorCitizen       7043 non-null int64
Partner             7043 non-null object
Dependents          7043 non-null object
tenure              7043 non-null int64
PhoneService        7043 non-null object
MultipleLines       7043 non-null object
InternetService     7043 non-null object
OnlineSecurity      7043 non-null object
OnlineBackup        7043 non-null object
DeviceProtection    7043 non-null object
TechSupport         7043 non-null object
StreamingTV         7043 non-null object
StreamingMovies     7043 non-null object
Contract            7043 non-null object
PaperlessBilling    7043 non-null object
PaymentMethod       7043 non-null object
MonthlyCharges      7043 non-null float64
TotalCharges        7043 non-null object
Churn               7043 non-null object
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

In [4]:

```
df.Churn = labelencoder.fit_transform(df.Churn)
df.Churn.value_counts()
```

Out[4]:

```
0    5174
1    1869
Name: Churn, dtype: int64
```

In [5]:

```
eventvar = df['Churn']
timevar = df['tenure']
```

In [6]:

```
categorical = ['gender', 'SeniorCitizen', 'Partner', 'Dependents', 'PhoneService
        'InternetService', 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
        'TechSupport', 'StreamingTV', 'StreamingMovies', 'Contract',
        'PaperlessBilling',  'PaymentMethod']

survivaldata = pd.get_dummies(df, columns = categorical, drop_first= True)
survivaldata.head()
```

Out[6]:

	customerID	tenure	MonthlyCharges	TotalCharges	Churn	gender_Male	SeniorCitiz
0	7590- VHVEG	1	29.85	29.85	0	0	
1	5575-	34	56.95	1889.5	0	1	
2	3668- QPYBK	2	53.85	108.15	1	1	
3	7795-	45	42.30	1840.75	0	1	
4	9237- HQITU	2	70.70	151.65	1	0	

5 rows × 32 columns

In [7]:

```python
survivaldata.drop(['customerID', 'tenure', 'Churn'], axis = 1, inplace= True)
survivaldata = st.add_constant(survivaldata, prepend=False)
survivaldata.head()
```

Out[7]:

	MonthlyCharges	TotalCharges	gender_Male	SeniorCitizen_1	Partner_Yes	Dependen
0	29.85	29.85	0	0	1	
1	56.95	1889.5	1	0	0	
2	53.85	108.15	1	0	0	
3	42.30	1840.75	1	0	0	
4	70.70	151.65	0	0	0	

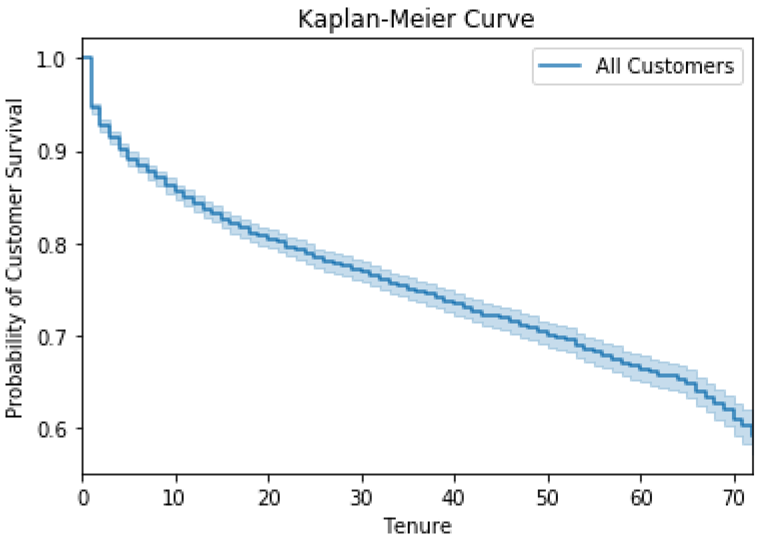
5 rows × 30 columns

Survival Analysis

Kaplan-Meier Curve

In [89]:

```python
#Create a KaplanMeier object, imported from lifelines
kmf = KaplanMeierFitter()
#Calculate the K-M curve for all groups
kmf.fit(timevar,event_observed = eventvar,label = "All Customers")
#Plot the curve and assign labels
kmf.plot()
plt.ylabel('Probability of Customer Survival')
plt.xlabel('Tenure')
plt.title('Kaplan-Meier Curve');
```

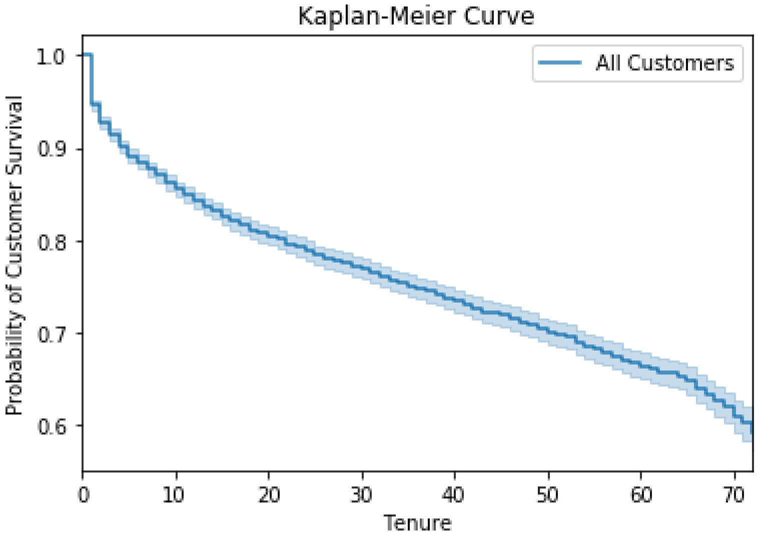


Log-Rank Test

In [90]:

```
male = (survivaldata['gender_Male'] == 1)
female = (survivaldata['gender_Male'] == 0)

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[male],event_observed = eventvar[male],label = "Male")
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[female],event_observed = eventvar[female],label = "Female")
plot2 = kmf.plot(ax = plot1)

plt.title('Survival of customers: Gender')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
groups = logrank_test(timevar[male], timevar[female], event_observed_A=eventvar[
groups.print_summary()
```

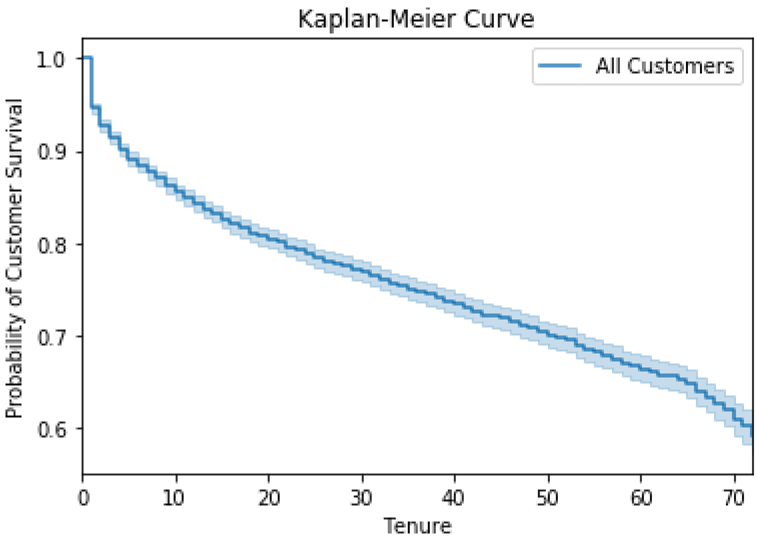
```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 1

---
 test_statistic    p  -log2(p)
           0.53 0.47      1.09
```



Senior Citizen

In [91]:
```
SeniorCitizen = (survivaldata['SeniorCitizen_1'] == 1)
no_SeniorCitizen = (survivaldata['SeniorCitizen_1'] == 0)

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[SeniorCitizen],event_observed = eventvar[SeniorCitizen],label =
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[no_SeniorCitizen],event_observed = eventvar[no_SeniorCitizen],la
plot2 = kmf.plot(ax = plot1)

plt.title('Survival of customers: Senior Citizen')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
groups = logrank_test(timevar[SeniorCitizen], timevar[no_SeniorCitizen], event_o
groups.print_summary()
```

```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 1

---
 test_statistic      p  -log2(p)
         109.49 <0.005     82.71
```

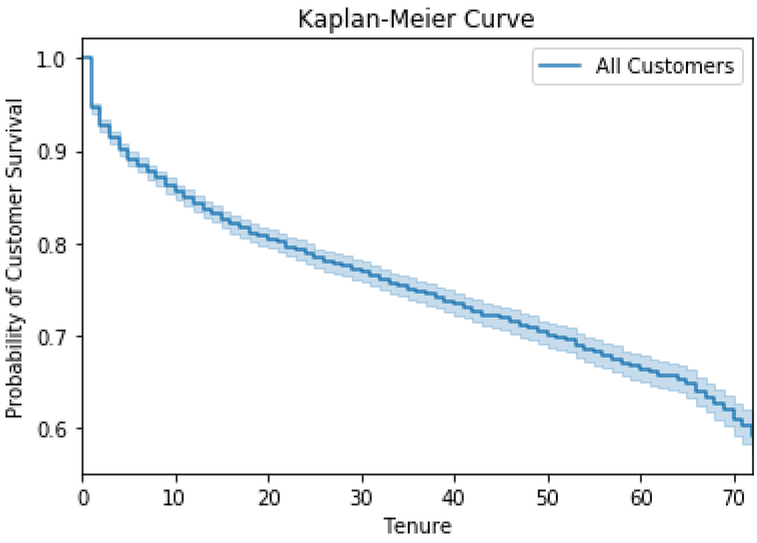


Partner

In [92]:

```
partner = (survivaldata['Partner_Yes'] == 1)
no_partner = (survivaldata['Partner_Yes'] == 0)

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[partner],event_observed = eventvar[partner],label = "Has partner
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[no_partner],event_observed = eventvar[no_partner],label = "Does
plot2 = kmf.plot(ax = plot1)

plt.title('Survival of customers: Partner')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
groups = logrank_test(timevar[partner], timevar[no_partner], event_observed_A=ev
groups.print_summary()
```

```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 1

---
 test_statistic      p  -log2(p)
         423.54 <0.005    310.21
```

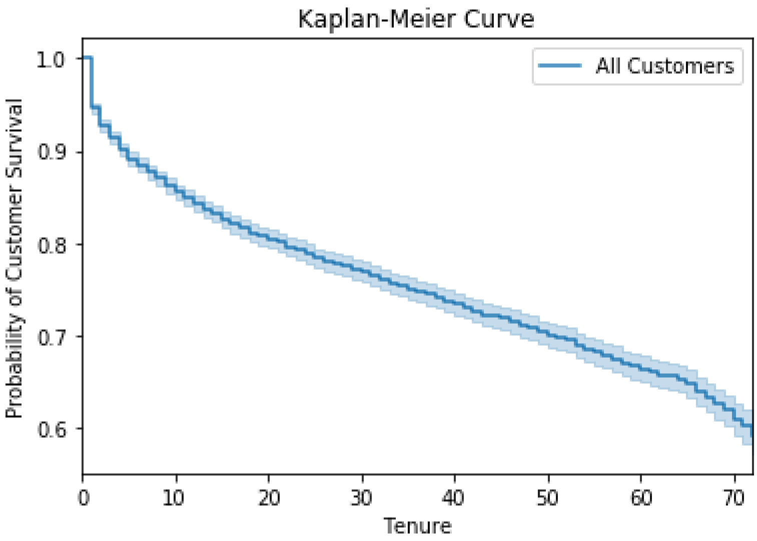


Dependents

In [93]:

```
Dependents = (survivaldata['Dependents_Yes'] == 1)
no_Dependents = (survivaldata['Dependents_Yes'] == 0)

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[Dependents],event_observed = eventvar[Dependents],label = "Has d
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[no_Dependents],event_observed  = eventvar[no_Dependents],label  =
plot2 = kmf.plot(ax = plot1)

plt.title('Survival of customers: Dependents')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
groups = logrank_test(timevar[Dependents], timevar[no_Dependents], event_observe
groups.print_summary()
```

```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 1

---
 test_statistic       p  -log2(p)
         232.70 <0.005    172.12
```

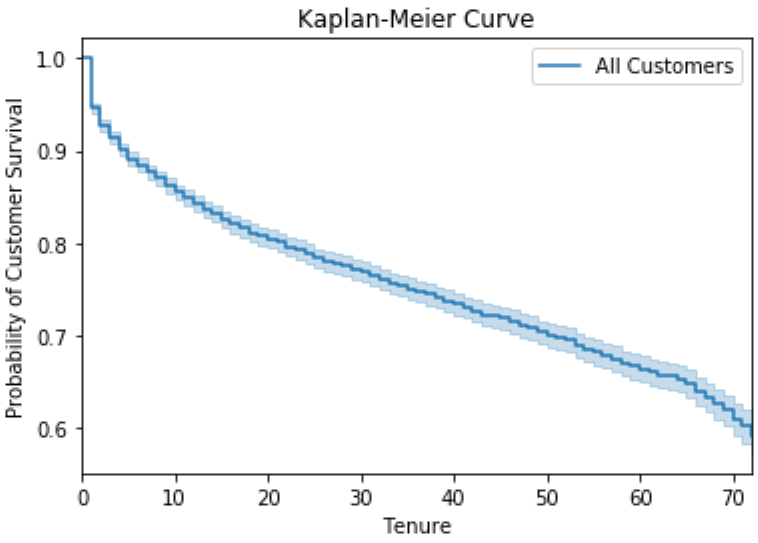


PhoneService

In [94]:

```
PhoneService = (survivaldata['PhoneService_Yes'] == 1)
no_PhoneService = (survivaldata['PhoneService_Yes'] == 0)

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[PhoneService],event_observed = eventvar[PhoneService],label = "H
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[no_PhoneService],event_observed = eventvar[no_PhoneService],labe
plot2 = kmf.plot(ax = plot1)

plt.title('Survival of customers: Phone Service')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
groups = logrank_test(timevar[PhoneService], timevar[no_PhoneService], event_obs
groups.print_summary()
```

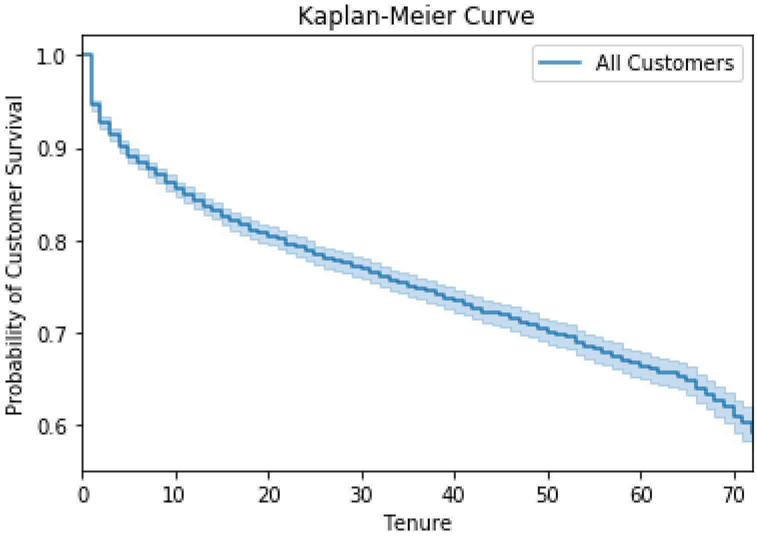
```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 1

---
 test_statistic    p  -log2(p)
           0.43 0.51      0.97
```



MultipleLines

In [95]:
```
no_phone = (survivaldata['MultipleLines_No phone service'] == 1)
multiLines = (survivaldata['MultipleLines_Yes'] == 1)
no_multiLines = ((survivaldata['MultipleLines_Yes'] == 0) & (survivaldata['Multi

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[no_phone],event_observed = eventvar[no_phone],label = "No Phone
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[multiLines],event_observed = eventvar[multiLines],label = "Multi
plot2 = kmf.plot(ax = plot1)

kmf.fit(timevar[no_multiLines],event_observed  =  eventvar[no_multiLines],label  =
plot3  =  kmf.plot(ax  =  plot2)

plt.title('Survival of customers: Mutliple Lines')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
twoplusgroups_logrank = multivariate_logrank_test(df['tenure'], df['MultipleLine
twoplusgroups_logrank.print_summary()
```

```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 2
             alpha = 0.95

---
 test_statistic      p  -log2(p)
          30.97 <0.005     22.34
```

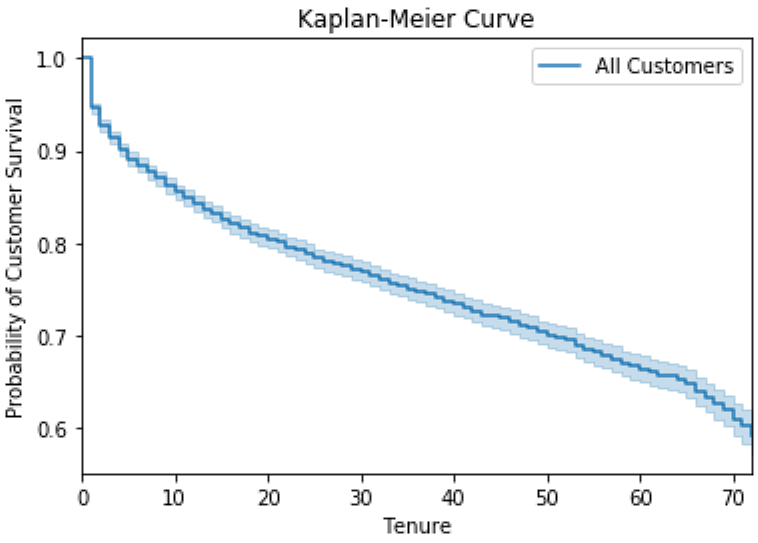


Internet Service

In [96]:

```
Fiber_optic = (survivaldata['InternetService_Fiber optic'] == 1)
No_Service = (survivaldata['InternetService_No'] == 1)
DSL  = ((survivaldata['InternetService_Fiber optic'] == 0) & (survivaldata['Inter

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[Fiber_optic],event_observed = eventvar[Fiber_optic],label = "Fib
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[No_Service],event_observed = eventvar[No_Service],label = "No Se
plot2 = kmf.plot(ax = plot1)

kmf.fit(timevar[DSL],event_observed = eventvar[DSL],label = "DSL")
plot3 = kmf.plot(ax = plot2)

plt.title('Survival of customers: Internet Service')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
twoplusgroups_logrank = multivariate_logrank_test(df['tenure'], df['InternetServ
twoplusgroups_logrank.print_summary()
```

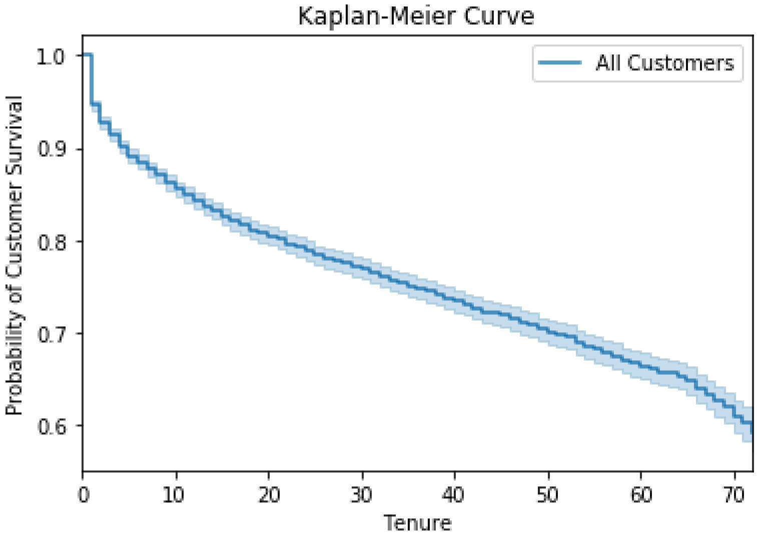
```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 2
             alpha = 0.95

---
 test_statistic       p  -log2(p)
         520.12 <0.005    375.19
```



Online Security

In [97]:
```
no_internetService = (survivaldata['OnlineSecurity_No internet service'] == 1)
onlineSecurity = (survivaldata['OnlineSecurity_Yes'] == 1)
no_onlineSecurity  = ((survivaldata['OnlineSecurity_No internet  service'] == 0) &

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[no_internetService],event_observed = eventvar[no_internetService
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[onlineSecurity],event_observed = eventvar[onlineSecurity],label
plot2 = kmf.plot(ax = plot1)

kmf.fit(timevar[no_onlineSecurity],event_observed = eventvar[no_onlineSecurity],
plot3 = kmf.plot(ax = plot2)

plt.title('Survival of customers: Online Security')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
twoplusgroups_logrank = multivariate_logrank_test(df['tenure'], df['OnlineSecuri
twoplusgroups_logrank.print_summary()
```

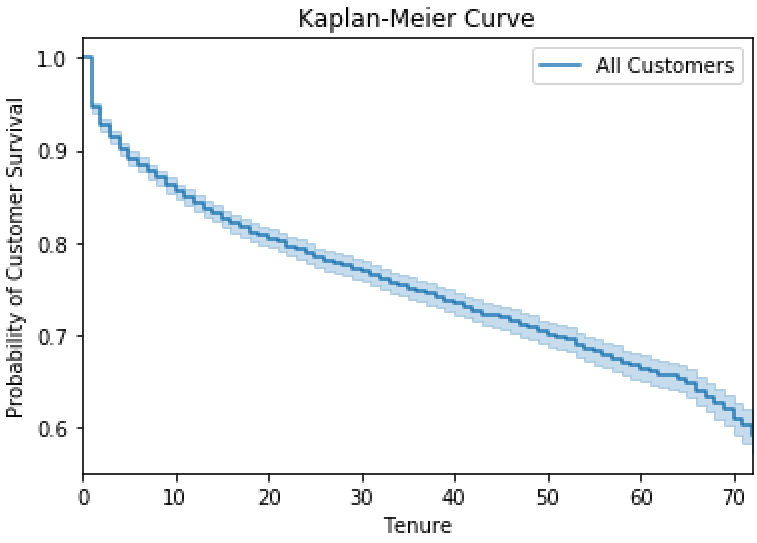
```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 2
             alpha = 0.95

---
 test_statistic      p  -log2(p)
        1013.86 <0.005    731.35
```



Online Backup

```
In [98]:  no_internetService = (survivaldata['OnlineBackup_No internet service'] == 1)
          onlineBackup = (survivaldata['OnlineBackup_Yes'] == 1)
          no_onlineBackup = ((survivaldata['OnlineBackup_No internet service'] == 0) & (su

          plt.figure()
          ax = plt.subplot(1,1,1)

          kmf.fit(timevar[no_internetService],event_observed = eventvar[no_internetService
          plot1 = kmf.plot(ax = ax)

          kmf.fit(timevar[onlineBackup],event_observed = eventvar[onlineBackup],label = "O
          plot2 = kmf.plot(ax = plot1)

          kmf.fit(timevar[no_onlineBackup],event_observed = eventvar[no_onlineBackup],labe
          plot3 = kmf.plot(ax = plot2)

          plt.title('Survival of customers: Online Backup')
          plt.xlabel('Tenure')
          plt.ylabel('Survival Probability')
          plt.yticks(np.linspace(0,1,11))
          twoplusgroups_logrank = multivariate_logrank_test(df['tenure'], df['OnlineBackup
          twoplusgroups_logrank.print_summary()
```

```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 2
             alpha = 0.95

---
 test_statistic      p  -log2(p)
         821.34 <0.005    592.47
```

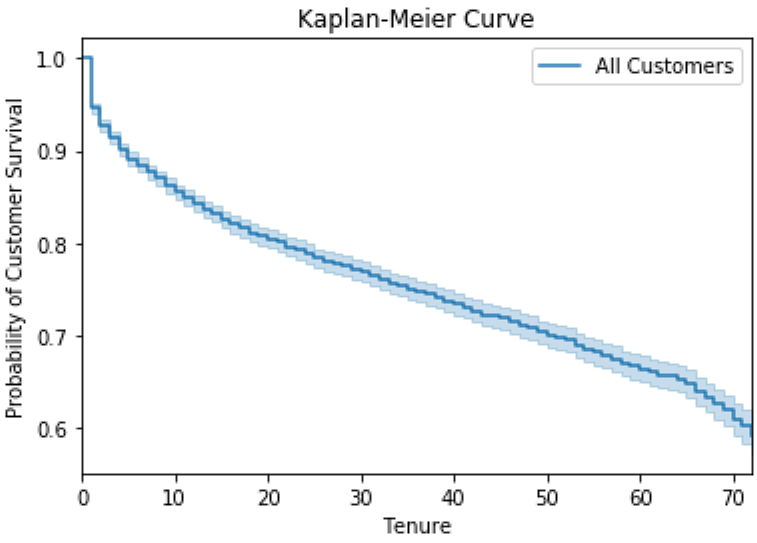


Device Protection

In [99]:

```
no_internetService = (survivaldata['DeviceProtection_No internet service'] == 1)
DeviceProtection = (survivaldata['DeviceProtection_Yes'] == 1)
no_DeviceProtection  = ((survivaldata['DeviceProtection_No  internet  service']  ==

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[no_internetService],event_observed = eventvar[no_internetService
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[DeviceProtection],event_observed = eventvar[DeviceProtection],la
plot2 = kmf.plot(ax = plot1)

kmf.fit(timevar[no_DeviceProtection],event_observed = eventvar[no_DeviceProtecti
plot3 = kmf.plot(ax = plot2)

plt.title('Survival of customers: Device Protection')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
twoplusgroups_logrank = multivariate_logrank_test(df['tenure'], df['DeviceProtec
twoplusgroups_logrank.print_summary()
```

```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 2
             alpha = 0.95

---
 test_statistic      p  -log2(p)
         763.51 <0.005    550.75
```

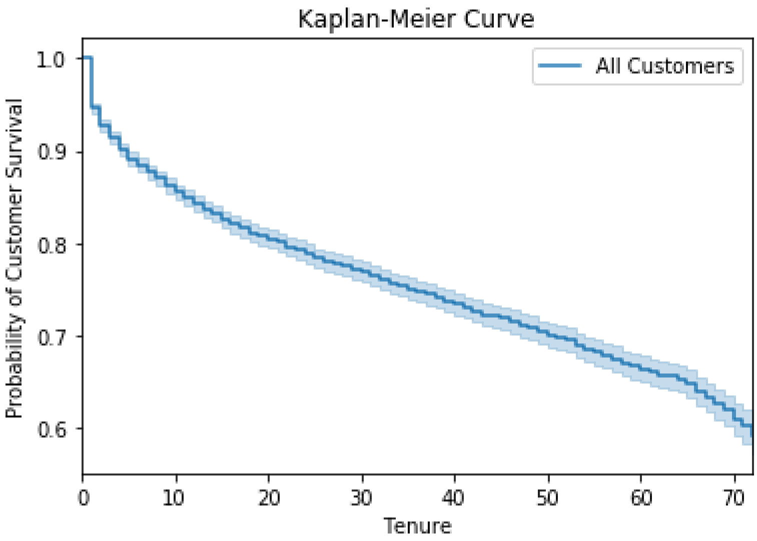


Tech Support

In [100…

```python
no_internetService = (survivaldata['TechSupport_No internet service'] == 1)
TechSupport = (survivaldata['TechSupport_Yes'] == 1)
no_TechSupport = ((survivaldata['TechSupport_No internet service'] == 0) & (surv

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[no_internetService],event_observed = eventvar[no_internetService
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[TechSupport],event_observed = eventvar[TechSupport],label = "Tec
plot2 = kmf.plot(ax = plot1)

kmf.fit(timevar[no_TechSupport],event_observed = eventvar[no_TechSupport],label
plot3 = kmf.plot(ax = plot2)

plt.title('Survival of customers: Tech Support')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
twoplusgroups_logrank = multivariate_logrank_test(df['tenure'], df['TechSupport'
twoplusgroups_logrank.print_summary()
```

```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 2
             alpha = 0.95

---
 test_statistic      p  -log2(p)
         989.56 <0.005     713.82
```

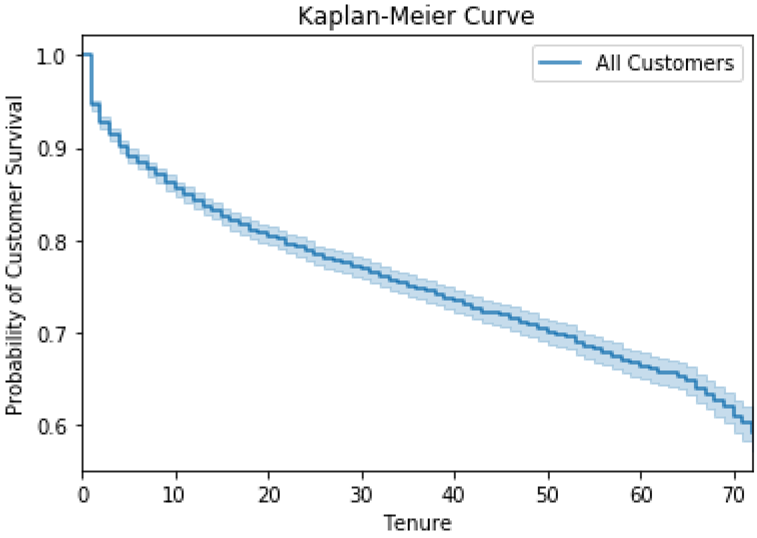


Streaming TV

In [101…

```
no_internetService = (survivaldata['StreamingTV_No internet service'] == 1)
StreamingTV = (survivaldata['StreamingTV_Yes'] == 1)
no_StreamingTV = ((survivaldata['StreamingTV_No internet service'] == 0) & (surv

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[no_internetService],event_observed = eventvar[no_internetService
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[StreamingTV],event_observed = eventvar[StreamingTV],label = "Str
plot2 = kmf.plot(ax = plot1)

kmf.fit(timevar[no_StreamingTV],event_observed = eventvar[no_StreamingTV],label
plot3 = kmf.plot(ax = plot2)

plt.title('Survival of customers: Streaming TV')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
twoplusgroups_logrank = multivariate_logrank_test(df['tenure'], df['StreamingTV'
twoplusgroups_logrank.print_summary()
```

```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 2
             alpha = 0.95

---
 test_statistic       p  -log2(p)
         368.31 <0.005    265.68
```

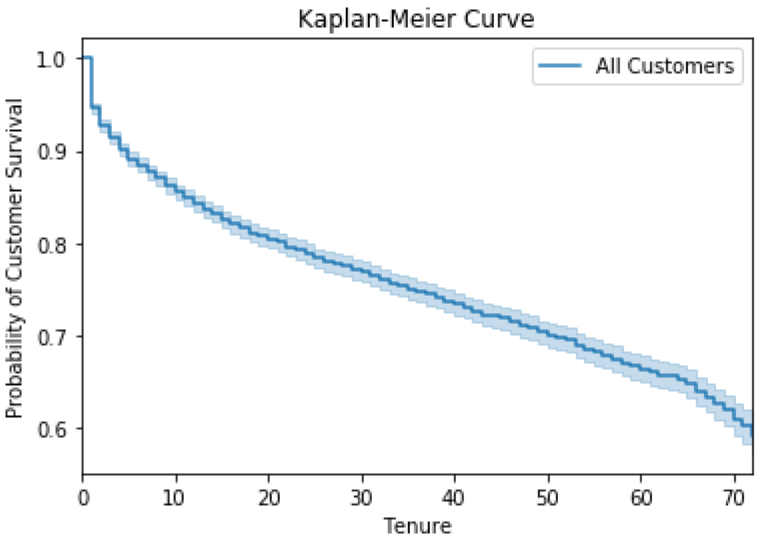


Streaming Movies

In [102…

```
no_internetService = (survivaldata['StreamingMovies_No internet service'] == 1)
StreamingMovies = (survivaldata['StreamingMovies_Yes'] == 1)
no_StreamingMovies = ((survivaldata['StreamingMovies_No internet service'] == 0)

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[no_internetService],event_observed = eventvar[no_internetService
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[StreamingMovies],event_observed = eventvar[StreamingMovies],labe
plot2 = kmf.plot(ax = plot1)

kmf.fit(timevar[no_StreamingMovies],event_observed = eventvar[no_StreamingMovies
plot3 = kmf.plot(ax = plot2)

plt.title('Survival of customers: Streaming Movies')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
twoplusgroups_logrank = multivariate_logrank_test(df['tenure'], df['StreamingMov
twoplusgroups_logrank.print_summary()
```

```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 2
             alpha = 0.95

---
 test_statistic      p  -log2(p)
         378.43 <0.005    272.98
```

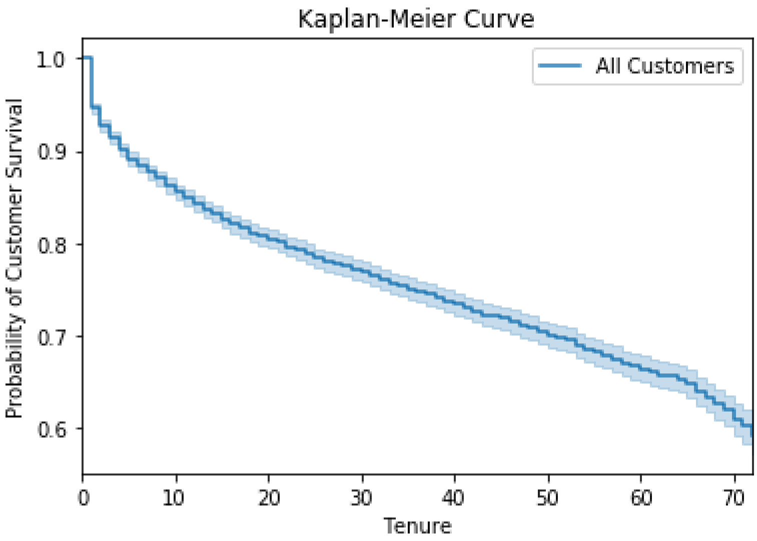


Contract

In [103…

```
Contract_One_year = (survivaldata['Contract_One year'] == 1)
Contract_Two_year  = (survivaldata['Contract_Two year'] == 1)
Contract_month_to_month = ((survivaldata['Contract_One year'] == 0) & (survivald

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[Contract_One_year],event_observed = eventvar[Contract_One_year],
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[Contract_Two_year],event_observed = eventvar[Contract_Two_year],
plot2 = kmf.plot(ax = plot1)

kmf.fit(timevar[Contract_month_to_month],event_observed = eventvar[Contract_mont
plot3 = kmf.plot(ax = plot2)

plt.title('Survival of customers: Contract')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
twoplusgroups_logrank = multivariate_logrank_test(df['tenure'], df['Contract'],
twoplusgroups_logrank.print_summary()
```

```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 2
             alpha = 0.95

---
 test_statistic      p  -log2(p)
        2352.87 <0.005       inf
```

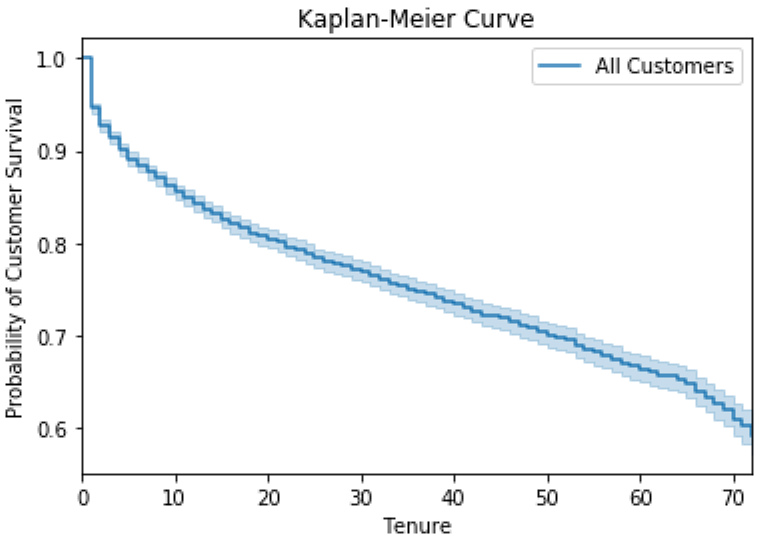


Payment Method

In [104…

```
automatic_Credit_Card  =  (survivaldata['PaymentMethod_Credit  card  (automatic)']  =
electronic_check = (survivaldata['PaymentMethod_Electronic check'] == 1)
mailed_check = (survivaldata['PaymentMethod_Mailed check'] == 1)
automatic_Bank_Transfer  =  ((survivaldata['PaymentMethod_Credit  card  (automatic)'

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[automatic_Credit_Card],event_observed = eventvar[automatic_Credi
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[electronic_check],event_observed = eventvar[electronic_check],la
plot2 = kmf.plot(ax = plot1)

kmf.fit(timevar[mailed_check],event_observed = eventvar[mailed_check],label = "M
plot3 = kmf.plot(ax = plot2)

kmf.fit(timevar[automatic_Bank_Transfer],event_observed = eventvar[automatic_Ban
plot4 = kmf.plot(ax = plot3)

plt.title('Survival of customers: PaymentMethod')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
twoplusgroups_logrank = multivariate_logrank_test(df['tenure'], df['PaymentMetho
twoplusgroups_logrank.print_summary()
```

```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 3
             alpha = 0.95

---
 test_statistic      p  -log2(p)
         865.24 <0.005    619.58
```

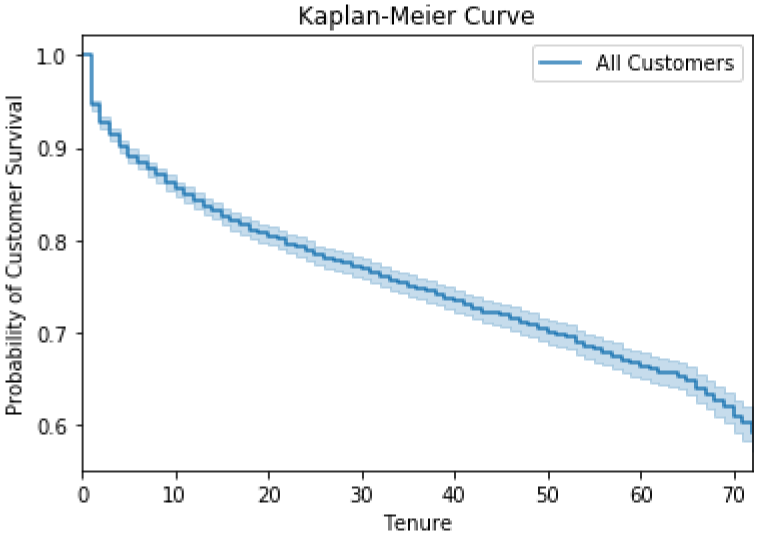


Paperless Billing

In [105…

```
PaperlessBilling = (survivaldata['PaperlessBilling_Yes'] == 1)
no_PaperlessBilling = (survivaldata['PaperlessBilling_Yes'] == 0)

plt.figure()
ax = plt.subplot(1,1,1)

kmf.fit(timevar[PaperlessBilling],event_observed = eventvar[PaperlessBilling],la
plot1 = kmf.plot(ax = ax)

kmf.fit(timevar[no_PhoneService],event_observed = eventvar[no_PhoneService],labe
plot2 = kmf.plot(ax = plot1)

plt.title('Survival of customers: Paperless Billing')
plt.xlabel('Tenure')
plt.ylabel('Survival Probability')
plt.yticks(np.linspace(0,1,11))
groups = logrank_test(timevar[PaperlessBilling], timevar[no_PaperlessBilling], e
groups.print_summary()
```

```
<lifelines.StatisticalResult>
               t_0 = -1
 null_distribution = chi squared
degrees_of_freedom = 1

---
 test_statistic      p  -log2(p)
         189.51 <0.005    140.82
```



Survival Regression

In [3]:

```
def datapreparation(filepath):

    df = pd.read_csv(filepath)
    df.drop(["customerID"], inplace = True, axis = 1)

    df.TotalCharges = df.TotalCharges.replace(" ",np.nan)
    df.TotalCharges.fillna(0, inplace = True)
    df.TotalCharges = df.TotalCharges.astype(float)

    cols1 = ['Partner', 'Dependents', 'PaperlessBilling', 'Churn', 'PhoneService
    for col in cols1:
        df[col] = df[col].apply(lambda x: 0 if x == "No" else 1)

    df.gender = df.gender.apply(lambda x: 0 if x == "Male" else 1)
    df.MultipleLines = df.MultipleLines.map({'No phone service': 0, 'No': 0, 'Ye

    cols2 = ['OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport'
    for col in cols2:
        df[col] = df[col].map({'No internet service': 0, 'No': 0, 'Yes': 1})

    df = pd.get_dummies(df, columns=['InternetService', 'Contract', 'PaymentMeth

    return df
```

In [4]:

```
regression_df = datapreparation("C:/Data/Telco-Customer-Churn.csv")
regression_df.head()
```

Out[4]:

	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	O
0	1	0	1	0	1	0	0	
1	0	0	0	0	34	1	0	
2	0	0	0	0	2	1	0	
3	0	0	0	0	45	0	0	
4	1	0	0	0	2	1	0	

5 rows × 24 columns

Survival Regression Ananlysis using Cox Proportional Hazard model

In [5]:

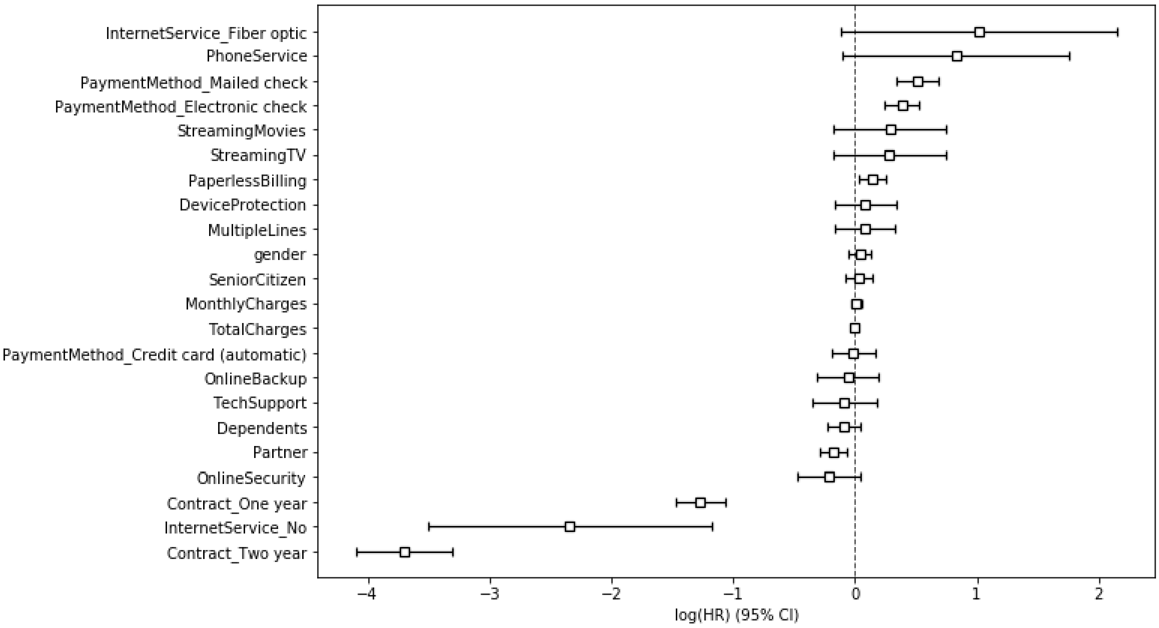
```
cph = CoxPHFitter()
cph.fit(regression_df, duration_col='tenure', event_col='Churn')

cph.print_summary()
```

```
<lifelines.CoxPHFitter: fitted with 7043 observations, 5174 censored>
      duration col = 'tenure'
         event col = 'Churn'
number of subjects = 7043
  number of events = 1869
partial log-likelihood = -12659.69
  time fit was run = 2020-09-22 14:53:48 UTC

---
                                        coef  exp(coef)   se(coef)      z      p  -l
og2(p)  lower 0.95  upper 0.95
gender                                  0.04       1.04       0.05   0.85   0.40
1.33       -0.05        0.13
SeniorCitizen                           0.03       1.04       0.06   0.61   0.54
0.88       -0.08        0.15
Partner                                -0.18       0.84       0.06  -3.23 <0.005
9.67       -0.29       -0.07
Dependents                             -0.09       0.91       0.07  -1.31   0.19
2.40       -0.23        0.05
PhoneService                            0.83       2.29       0.47   1.75   0.08
3.63       -0.10        1.76
MultipleLines                           0.09       1.09       0.13   0.69   0.49
1.03       -0.16        0.33
OnlineSecurity                         -0.21       0.81       0.13  -1.60   0.11
3.20       -0.47        0.05
OnlineBackup                           -0.06       0.95       0.13  -0.44   0.66
0.60       -0.31        0.19
DeviceProtection                        0.09       1.09       0.13   0.69   0.49
1.03       -0.16        0.34
TechSupport                            -0.08       0.92       0.13  -0.64   0.52
0.93       -0.34        0.17
StreamingTV                             0.28       1.32       0.24   1.19   0.23
2.10       -0.18        0.74
StreamingMovies                         0.29       1.33       0.24   1.22   0.22
2.16       -0.18        0.75
PaperlessBilling                        0.15       1.16       0.06   2.65   0.01
6.95        0.04        0.26
MonthlyCharges                          0.01       1.01       0.02   0.57   0.57
0.82       -0.03        0.06
TotalCharges                           -0.00       1.00       0.00 -39.16 <0.005
inf       -0.00       -0.00
InternetService_Fiber  optic            1.02       2.77       0.58   1.76   0.08
3.67       -0.12        2.15
InternetService_No                     -2.34       0.10       0.60  -3.93 <0.005
13.51       -3.51       -1.17
Contract_One  year                     -1.27       0.28       0.10 -12.55 <0.005
117.58       -1.46       -1.07
Contract_Two  year                     -3.70       0.02       0.20 -18.32 <0.005
246.60       -4.10       -3.31
PaymentMethod_Credit  card (automatic) -0.01       0.99       0.09  -0.13   0.90
0.16       -0.19        0.17
PaymentMethod_Electronic  check         0.39       1.47       0.07   5.31 <0.005
23.13        0.24        0.53
PaymentMethod_Mailed  check             0.51       1.67       0.09   5.87 <0.005
27.74        0.34        0.68
---
Concordance = 0.93
Log-likelihood ratio test = 5986.69 on 22 df, -log2(p)=inf
```

```
In [6]: cph.score_
```

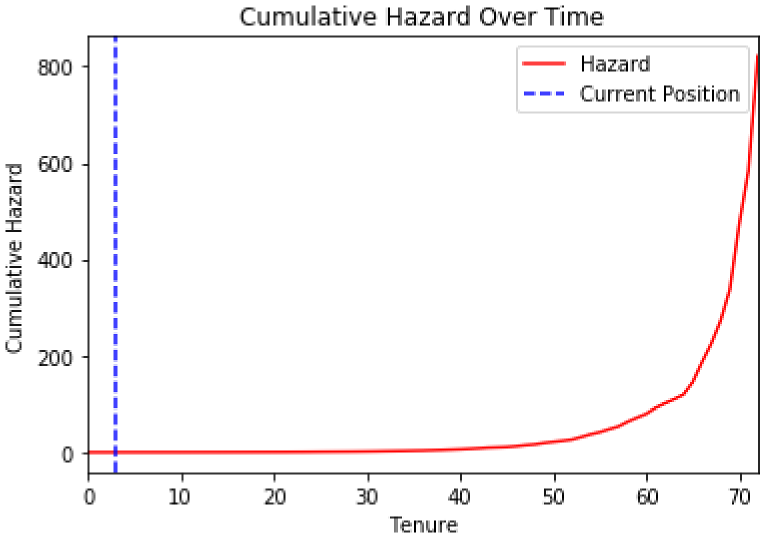
```
Out[6]:  0.9285636735265471
```

```
In [7]:  fig, ax = plt.subplots(figsize = (10,7))
         cph.plot(ax = ax);
```

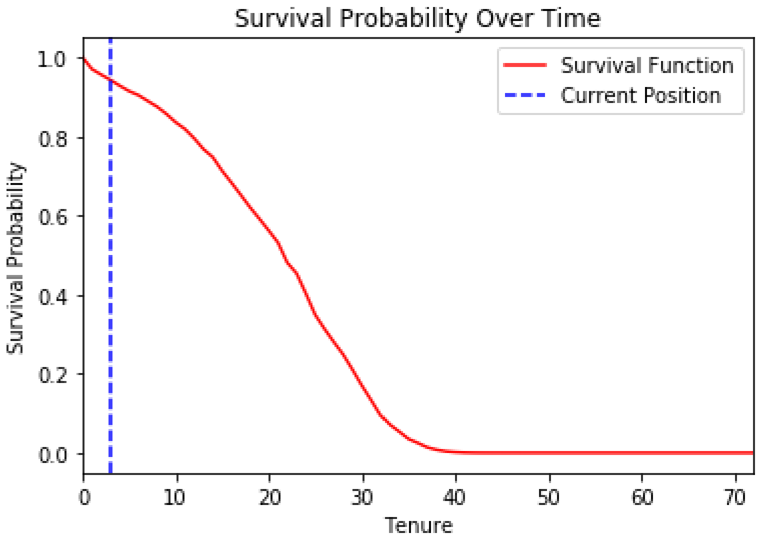


```
In [15]: test_id = regression_df.sample(1)
```

```
In [16]: fig, ax = plt.subplots()
         cph.predict_cumulative_hazard(test_id).plot(ax = ax, color = 'red')
         plt.axvline(x=test_id.tenure.values[0], color = 'blue', linestyle='--')
         plt.legend(labels=['Hazard','Current Position'])
         ax.set_xlabel('Tenure', size = 10)
         ax.set_ylabel('Cumulative Hazard', size = 10)
         ax.set_title('Cumulative Hazard Over Time');
```



```
In [17]: fig, ax = plt.subplots()
         cph.predict_survival_function(test_id).plot(ax = ax, color = 'red')
         plt.axvline(x=test_id.tenure.values[0], color = 'blue', linestyle='--')
         plt.legend(labels=['Survival Function','Current Position'])
         ax.set_xlabel('Tenure', size = 10)
         ax.set_ylabel('Survival Probability', size = 10)
         ax.set_title('Survival Probability Over Time');
```



Saving the model

In [8]:

```
import pickle
pickle.dump(cph, open('survivemodel.pkl','wb'))
```

Customer Lifetime Value

In [87]:

```
def LTV(info):
    life = cph.predict_survival_function(info).reset_index()
    life.columns = ['Tenure', 'Probability']
    max_life = life.Tenure[life.Probability > 0.1].max()

    LTV = max_life * info['MonthlyCharges'].values[0]
    return LTV
```

In [89]:

```
print('LTV of a testid is:', LTV(test_id), 'dollars.')
```

```
LTV of a testid is: 922.25 dollars.
```