data_challenge1

February 20, 2019

1 Employee Retention

```
In [54]: #Import data and necessary libraries
         import pandas as pd
         import numpy as np
         import datetime
         import os
         import seaborn as sns
         sns.set(style="whitegrid")
         import matplotlib.pyplot as plt
         %matplotlib inline

         working_dir = os.getcwd()
         employee_data = pd.read_csv(os.path.join(working_dir,"data/employee_retention_data.csv"

         #Check if each row is a unique employee id
         if len(employee_data) == len(employee_data.index.unique()):
             print("Each row represents a unique employee ID")
```

```
Each row represents a unique employee ID
```

```
In [55]: #Variable for how long employee has been at company
         #If still works there, assume tenure is from start date until 12/13/2015 (per data chal

         todays_date = pd.to_datetime(datetime.date(2015, 12, 13))

         def tenure_quit(row):
             if pd.isnull(row['quit_date']):
                 tenure = pd.to_datetime(todays_date) - pd.to_datetime(row['join_date'])
             else:
                 tenure = pd.to_datetime(row['quit_date']) - pd.to_datetime(row['join_date'])
             return(tenure)

         employee_data['tenure'] = employee_data.apply(tenure_quit, axis=1)

         #Binary flag for quitting and numerical tenure
```

```
employee_data['quit_binary'] = np.where(pd.isnull(employee_data.quit_date),0,1)
employee_data['tenure_int'] = round(pd.to_numeric(employee_data.tenure)/1e14)
```

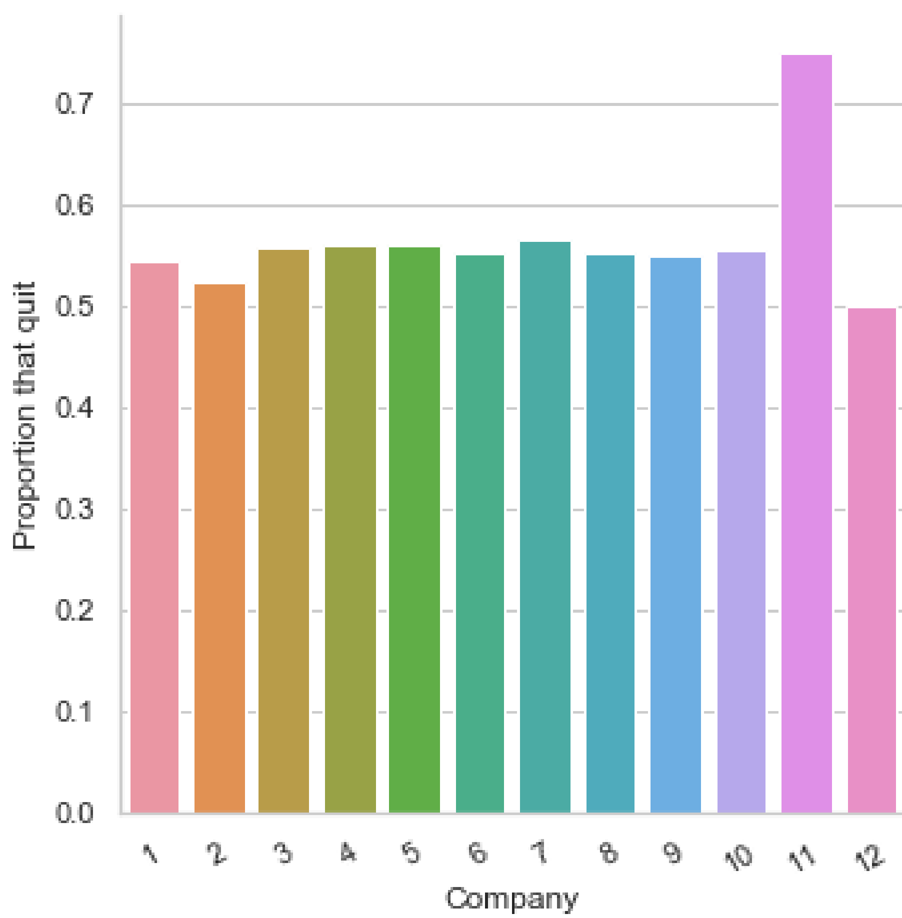
1.1 EDA

From the EDA below, the following becomes clear:

1. The proportion of employees that quit is similiar across companies (around 50%), except company 11 where it is higher (closer to 80%)
2. In aggregate, the proprtion of employees that quit is similiar across department, seniority, and salary level.
3. Looking at time until an employee quits (i.e., duration of employement), many employees seem to quit after 1 year, and then again after year 2, regardless of salary or seniority. Therefore, this problem is best modelled as a survival analysis (i.e., time to event), which I explore in the next section

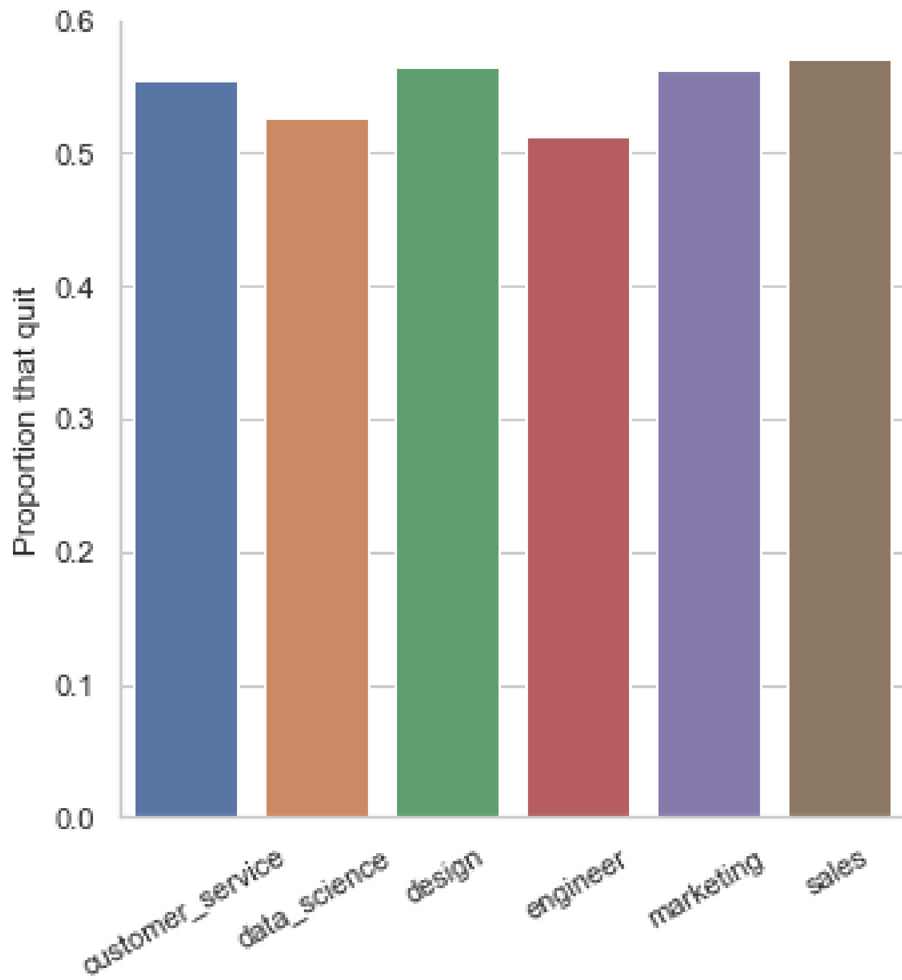
```
In [56]: #Look at quit rates by company
         ax = sns.catplot(x="company_id", y='quit_binary', data=employee_data.groupby('company_i
         ax.set(xlabel='Company', ylabel='Proportion that quit')
         ax.set_xticklabels(rotation=30)
```

```
Out[56]: <seaborn.axisgrid.FacetGrid at 0x11a4147f0>
```



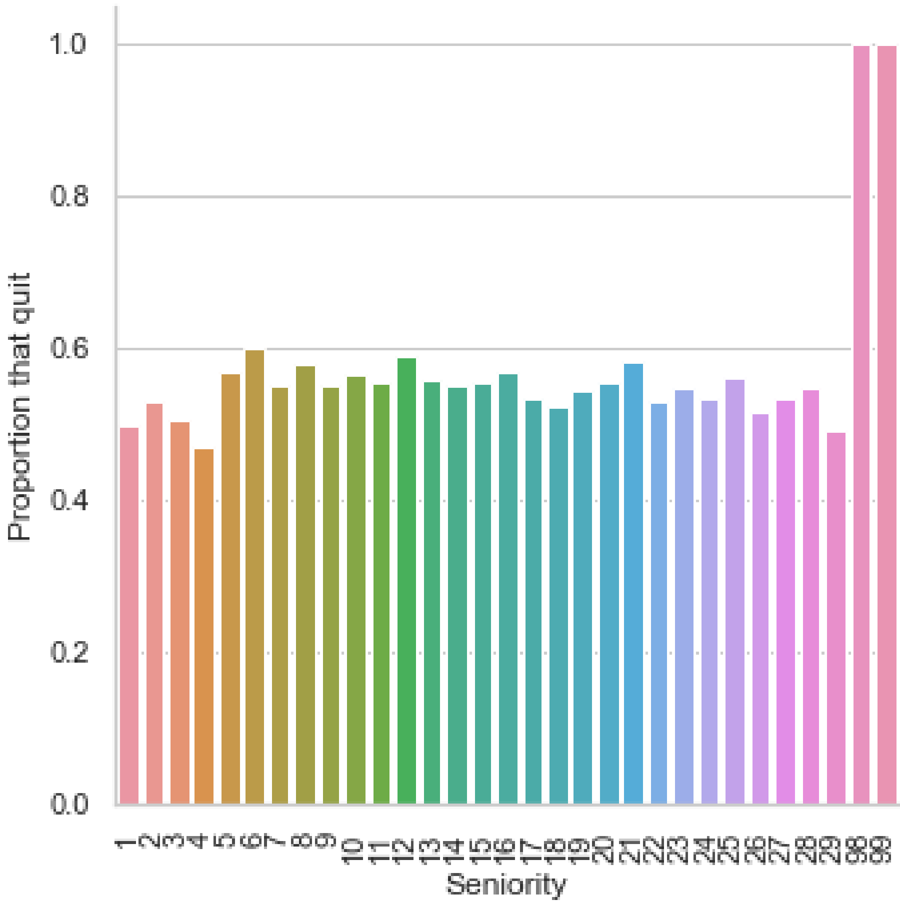
In [57]:
```
#Look at quit rates by department
ax = sns.catplot(x="dept", y='quit_binary', data=employee_data.groupby('dept').agg('mea
ax.set(xlabel='', ylabel='Proportion that quit')
ax.set_xticklabels(rotation=30)
```

Out[57]: <seaborn.axisgrid.FacetGrid at 0x11b103a20>

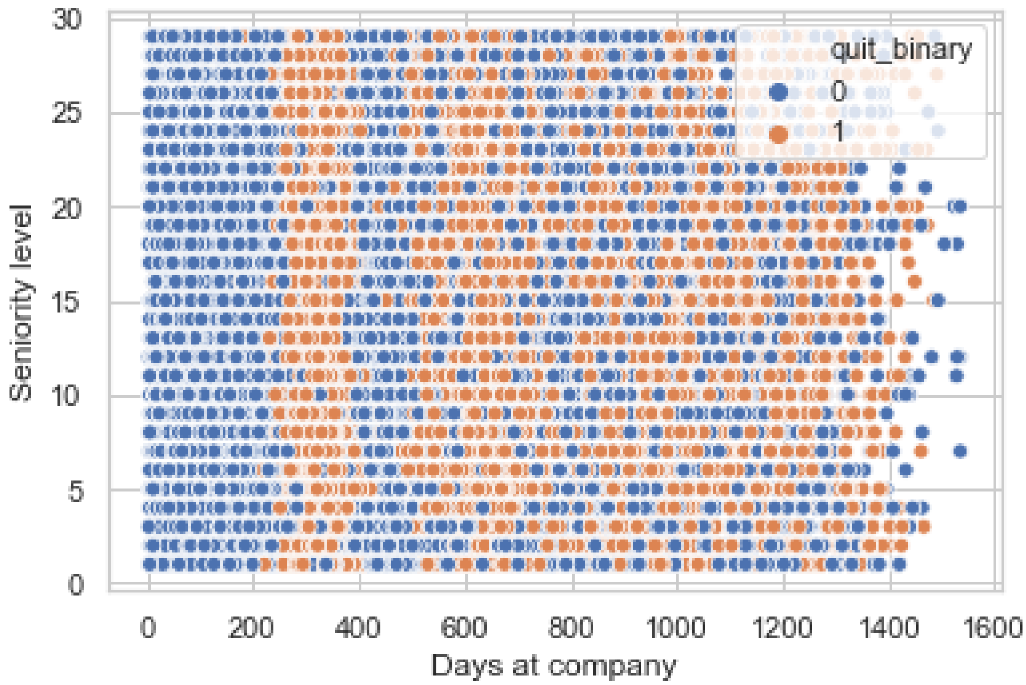


In [58]:
```
#Look at quit rates by seniority
ax = sns.catplot(x="seniority", y='quit_binary', data=employee_data.groupby('seniority'
ax.set(xlabel='Seniority', ylabel='Proportion that quit')
ax.set_xticklabels(rotation=90)
```

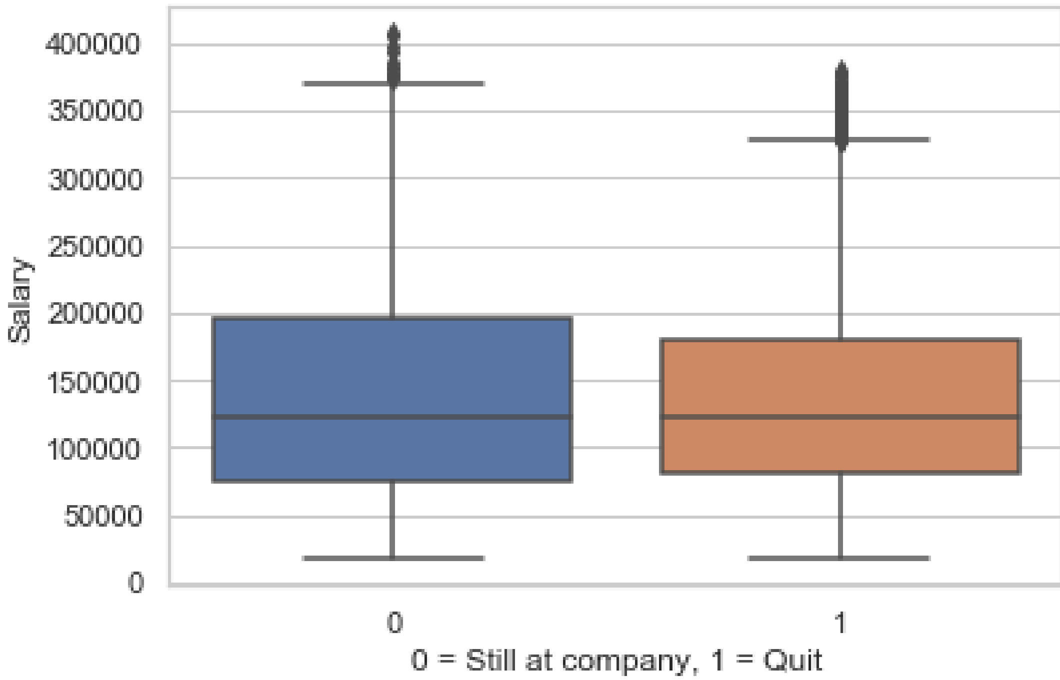
Out[58]: <seaborn.axisgrid.FacetGrid at 0x11b982a20>



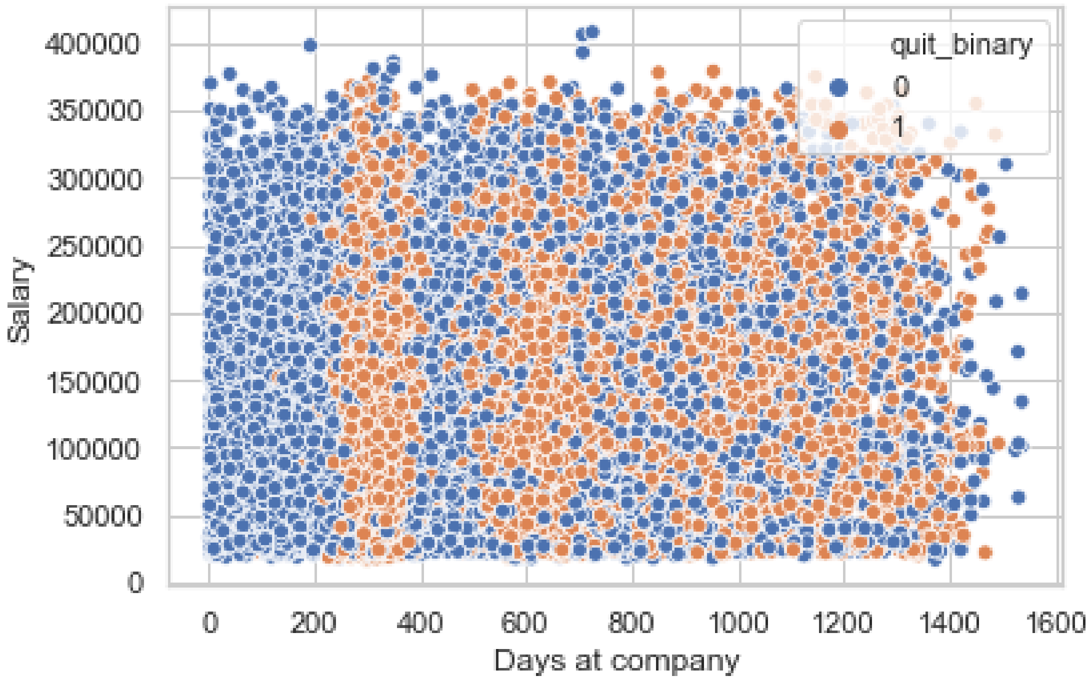

```
In [59]: #Seniority is missing for two records (set at 99 and 98 years, which cannot be accurate
         #Take these records out
         employee_data = employee_data[(employee_data.seniority != 98) & (employee_data.seniorit

In [60]: #Seniority and days at company scatterplot
         ax = sns.scatterplot(x =round(pd.to_numeric(employee_data.tenure)/1e14), y='seniority',
         ax.set(xlabel='Days at company', ylabel='Seniority level')

Out[60]: [<matplotlib.text.Text at 0x11b901da0>, <matplotlib.text.Text at 0x11b093860>]
```



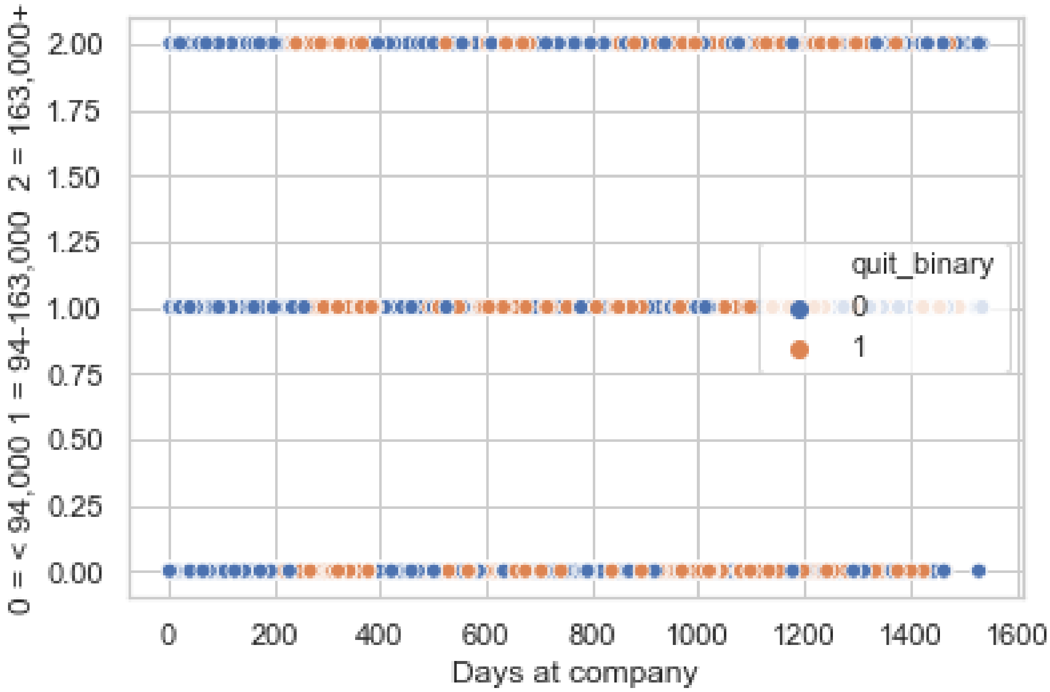

```
In [61]: #Look at quit rates by salary
         ax = sns.boxplot(x='quit_binary', y='salary', data=employee_data)
         ax.set(xlabel='0 = Still at company, 1 = Quit', ylabel='Salary')

Out[61]: [<matplotlib.text.Text at 0x11b778978>, <matplotlib.text.Text at 0x11b91b2e8>]
```




```
In [62]: #Look at quit rates by salary (scatterplot)
         ax = sns.scatterplot(x =round(pd.to_numeric(employee_data.tenure)/1e14), y='salary', hu
         ax.set(xlabel='Days at company', ylabel='Salary')

Out[62]: [<matplotlib.text.Text at 0x11b7bdb00>, <matplotlib.text.Text at 0x11ba5fc50>]
```




```
In [63]: #Look at quit rates by salary level
         #Low = < 94,000
         #Mid = 94,000 - 163,000
         #High = 163,000 +
         employee_data['salary_level'] = pd.qcut(employee_data.salary, 3, labels=(0,1,2))
         ax = sns.scatterplot(x =round(pd.to_numeric(employee_data.tenure)/1e14), y='salary_leve
         ax.set(xlabel='Days at company', ylabel='0 = < 94,000 1 = 94-163,000  2 = 163,000+')

Out[63]: [<matplotlib.text.Text at 0x11be0a908>, <matplotlib.text.Text at 0x11b06c128>]
```



1.2 Survival Analysis

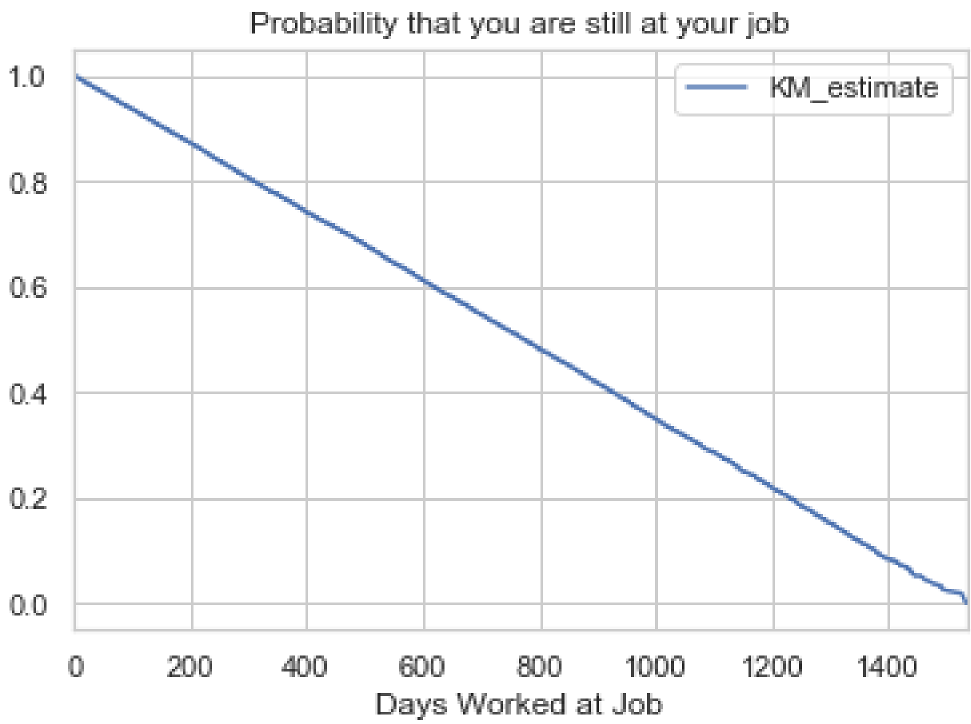
This section models the time for an employee to quit using a Kaplan Meier analysis. From it, we learn that:

1. Over all the data, the probability that an employee is still at their job descreases linearly with time at a company, with there being a 50% chance that an employee is at their job after 2 years, and almost a 0% chance after 4 years.
2. This relationship is similiar across departments and salary, however, differs again by company. At company 11, employees stay longer (with a 50% chance that an employee is at their job within 3 years) even though many end up quitting. At Company 12, employees leave at a faster rate than the average.
3. Looking at the company level, there is a difference in the employee retention rate by department. Therefore, I create a variable that is an interaction between the company ID and the department category.

In [27]:
```
from lifelines import KaplanMeierFitter
from lifelines.utils import datetimes_to_durations

T = employee_data['tenure_int']
E = employee_data['survival']

kmf = KaplanMeierFitter()
kmf.fit(T, event_observed=E)
```

```
kmf.survival_function_.plot()
plt.title('Probability that you are still at your job');
plt.xlabel('Days Worked at Job')
```

Out[27]: <matplotlib.text.Text at 0x1189db978>



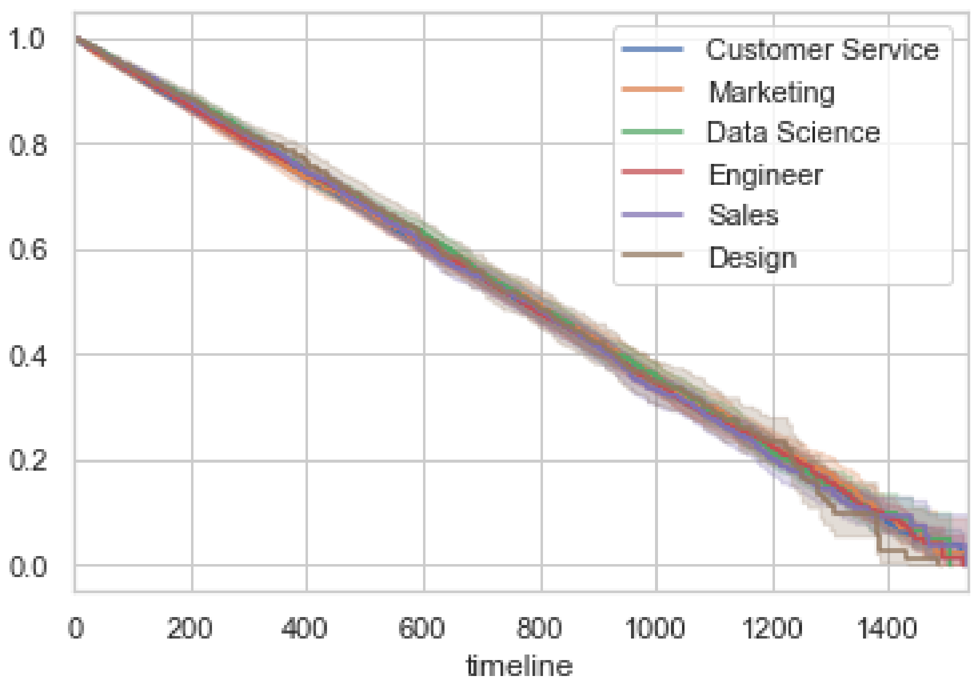
In [12]:
```
print("50% chance of employee quitting after " + str(kmf.median_) + ' days')
```

```
50% chance of employee quitting after 773.0 days
```

In [13]:
```
#Probability that you are still at your job by department
cust_service = (employee_data["dept"] == "customer_service")
marketing = (employee_data["dept"] == "marketing")
data_science = (employee_data["dept"] == "data_science")
engineer = (employee_data["dept"] == "engineer")
sales = (employee_data["dept"] == "sales")
design = (employee_data["dept"] == "design")

ax = plt.subplot(111)

kmf.fit(T[cust_service], event_observed=E[cust_service], label="Customer Service")
kmf.plot(ax=ax)
kmf.fit(T[marketing], event_observed=E[marketing], label="Marketing")
```

```
kmf.plot(ax=ax)
kmf.fit(T[data_science], event_observed=E[data_science], label="Data Science")
kmf.plot(ax=ax)
kmf.fit(T[engineer], event_observed=E[engineer], label="Engineer")
kmf.plot(ax=ax)
kmf.fit(T[sales], event_observed=E[sales], label="Sales")
kmf.plot(ax=ax)
kmf.fit(T[design], event_observed=E[design], label="Design")
kmf.plot(ax=ax)
```

```
Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x11780b128>
```

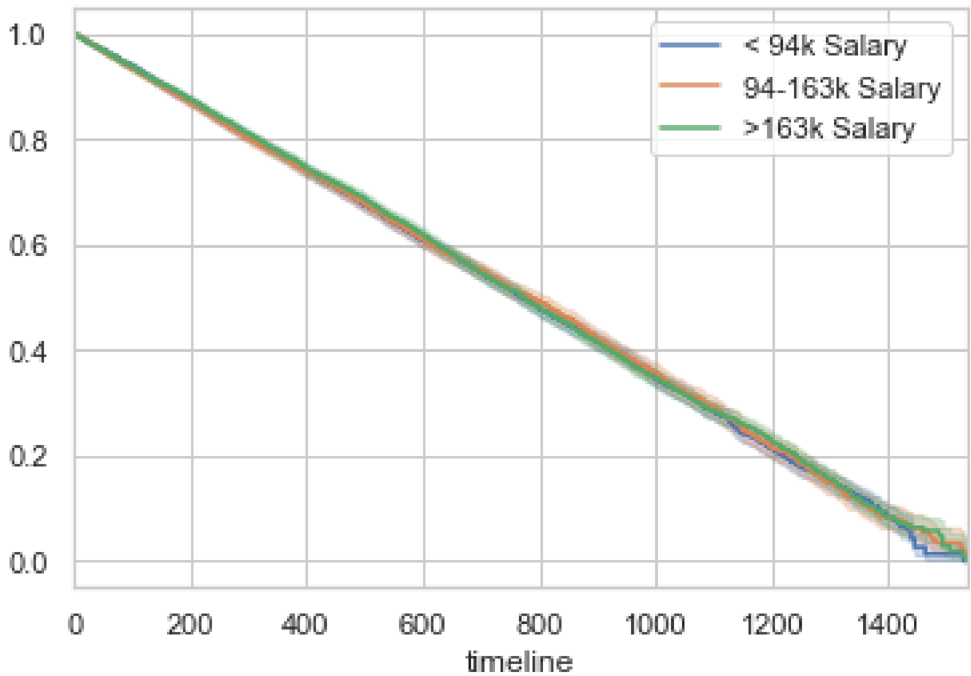



```
In [14]: #Probability that you are still at your job by salary level
         low_salary = (employee_data["salary_level"] == 0)
         mid_salary = (employee_data["salary_level"] == 1)
         high_salary = (employee_data["salary_level"] == 2)

         ax = plt.subplot(111)

         kmf.fit(T[low_salary], event_observed=E[low_salary], label="< 94k Salary ")
         print("Low Salary: 50% chance of employee quitting after " + str(kmf.median_) + ' days'
         kmf.plot(ax=ax)
         kmf.fit(T[mid_salary], event_observed=E[mid_salary], label="94-163k Salary")
         print("Mid Salary: 50% chance of employee quitting after " + str(kmf.median_) + ' days'
```

```
        kmf.plot(ax=ax)
        kmf.fit(T[high_salary], event_observed=E[high_salary], label=">163k Salary")
        print("High Salary: 50% chance of employee quitting after " + str(kmf.median_) + ' days
        kmf.plot(ax=ax)
```

```
Low Salary: 50% chance of employee quitting after 767.0 days
Mid Salary: 50% chance of employee quitting after 785.0 days
High Salary: 50% chance of employee quitting after 773.0 days
```

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x117d80128>



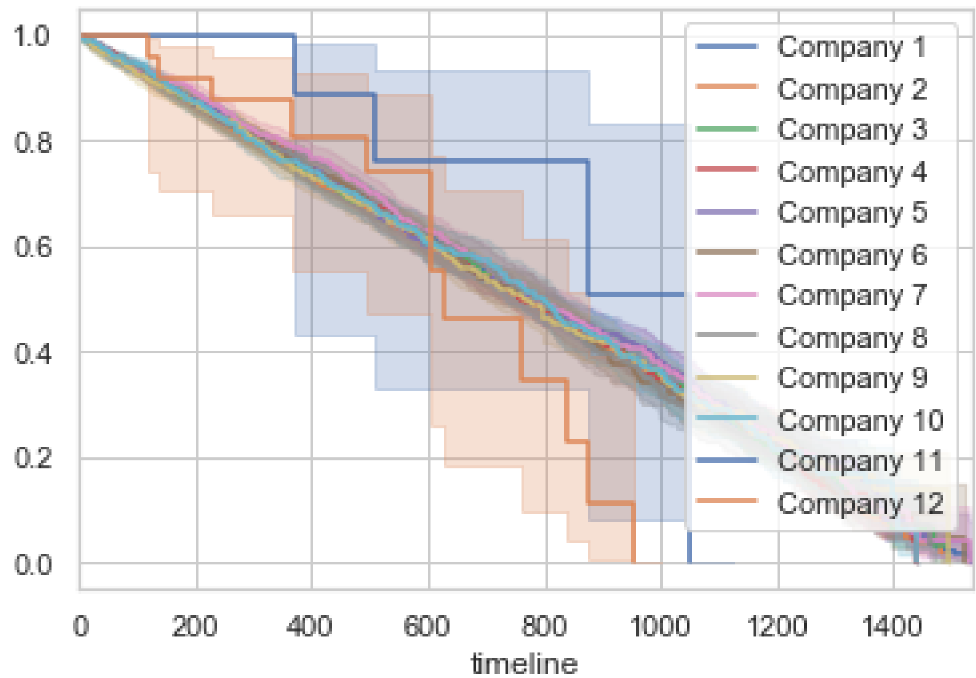
In [15]:
```
#Probability that you are still at your job by company
company_1 = (employee_data["company_id"] == 1)
company_2 = (employee_data["company_id"] == 2)
company_3 = (employee_data["company_id"] == 3)
company_4 = (employee_data["company_id"] == 4)
company_5 = (employee_data["company_id"] == 5)
company_6 = (employee_data["company_id"] == 6)
company_7 = (employee_data["company_id"] == 7)
company_8 = (employee_data["company_id"] == 8)
company_9 = (employee_data["company_id"] == 9)
company_10 = (employee_data["company_id"] == 10)
company_11 = (employee_data["company_id"] == 11)
```

```
company_12 = (employee_data["company_id"] == 12)

ax = plt.subplot(111)

kmf.fit(T[company_1], event_observed=E[company_1], label="Company 1")
print("Company 1: 50% chance of employee quitting after " + str(kmf.median_) + ' days')
kmf.plot(ax=ax)
kmf.fit(T[company_2], event_observed=E[company_2], label="Company 2")
print("Company 2: 50% chance of employee quitting after " + str(kmf.median_) + ' days')
kmf.plot(ax=ax)
kmf.fit(T[company_3], event_observed=E[company_3], label="Company 3")
print("Company 3: 50% chance of employee quitting after " + str(kmf.median_) + ' days')
kmf.plot(ax=ax)
kmf.fit(T[company_4], event_observed=E[company_4], label="Company 4")
print("Company 4: 50% chance of employee quitting after " + str(kmf.median_) + ' days')
kmf.plot(ax=ax)
kmf.fit(T[company_5], event_observed=E[company_5], label="Company 5")
print("Company 5: 50% chance of employee quitting after " + str(kmf.median_) + ' days')
kmf.plot(ax=ax)
kmf.fit(T[company_6], event_observed=E[company_6], label="Company 6")
print("Company 6: 50% chance of employee quitting after " + str(kmf.median_) + ' days')
kmf.plot(ax=ax)
kmf.fit(T[company_7], event_observed=E[company_7], label="Company 7")
print("Company 7: 50% chance of employee quitting after " + str(kmf.median_) + ' days')
kmf.plot(ax=ax)
kmf.fit(T[company_8], event_observed=E[company_8], label="Company 8")
print("Company 8: 50% chance of employee quitting after " + str(kmf.median_) + ' days')
kmf.plot(ax=ax)
kmf.fit(T[company_9], event_observed=E[company_9], label="Company 9")
print("Company 9: 50% chance of employee quitting after " + str(kmf.median_) + ' days')
kmf.plot(ax=ax)
kmf.fit(T[company_10], event_observed=E[company_10], label="Company 10")
print("Company 10: 50% chance of employee quitting after " + str(kmf.median_) + ' days'
kmf.plot(ax=ax)
kmf.fit(T[company_11], event_observed=E[company_11], label="Company 11")
print("Company 11: 50% chance of employee quitting after " + str(kmf.median_) + ' days'
kmf.plot(ax=ax)
kmf.fit(T[company_12], event_observed=E[company_12], label="Company 12")
print("Company 12: 50% chance of employee quitting after " + str(kmf.median_) + ' days'
kmf.plot(ax=ax)
```

```
Company 1: 50% chance of employee quitting after 767.0 days
Company 2: 50% chance of employee quitting after 773.0 days
Company 3: 50% chance of employee quitting after 785.0 days
Company 4: 50% chance of employee quitting after 761.0 days
Company 5: 50% chance of employee quitting after 785.0 days
Company 6: 50% chance of employee quitting after 785.0 days
Company 7: 50% chance of employee quitting after 785.0 days
```

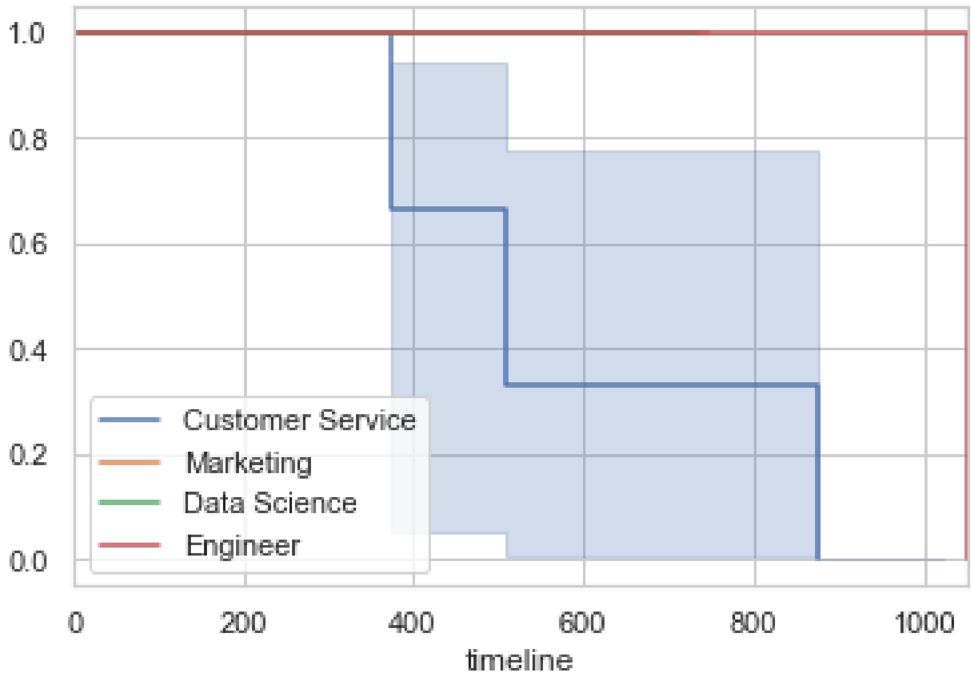
```
Company 8: 50% chance of employee quitting after 785.0 days
Company 9: 50% chance of employee quitting after 767.0 days
Company 10: 50% chance of employee quitting after 796.0 days
Company 11: 50% chance of employee quitting after 1051.0 days
Company 12: 50% chance of employee quitting after 627.0 days
```

Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x117ef60f0>



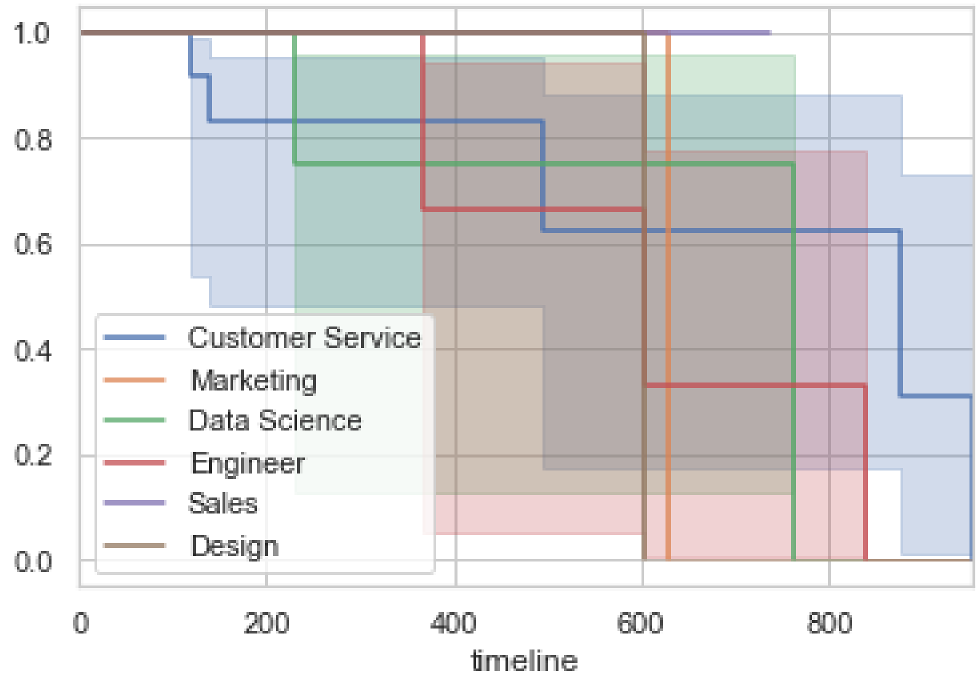
In [16]:
```
#Probability that you are still at your job by department at Company 11

ax = plt.subplot(111)

kmf.fit(T[(cust_service) & (company_11)], event_observed=E[(cust_service) & (company_11
kmf.plot(ax=ax)
kmf.fit(T[(marketing) & (company_11)], event_observed=E[(marketing) & (company_11)], la
kmf.plot(ax=ax)
kmf.fit(T[(data_science) & (company_11)], event_observed=E[(data_science) & (company_11
kmf.plot(ax=ax)
kmf.fit(T[(engineer) & (company_11)], event_observed=E[(engineer) & (company_11)], labe
kmf.plot(ax=ax)
```

Out[16]: <matplotlib.axes._subplots.AxesSubplot at 0x117ee2128>



In [17]:
```
#Probability that you are still at your job by department at Company 12

ax = plt.subplot(111)

kmf.fit(T[(cust_service) & (company_12)], event_observed=E[(cust_service) & (company_12
kmf.plot(ax=ax)
kmf.fit(T[(marketing) & (company_12)], event_observed=E[(marketing) & (company_12)], la
kmf.plot(ax=ax)
kmf.fit(T[(data_science) & (company_12)], event_observed=E[(data_science) & (company_12
kmf.plot(ax=ax)
kmf.fit(T[(engineer) & (company_12)], event_observed=E[(engineer) & (company_12)], labe
kmf.plot(ax=ax)
kmf.fit(T[(sales) & (company_12)], event_observed=E[(sales) & (company_12)], label="Sal
kmf.plot(ax=ax)
kmf.fit(T[(design) & (company_12)], event_observed=E[(design) & (company_12)], label="D
kmf.plot(ax=ax)
```

Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0x1180d9b70>



1.3 Model

The model below is a cox proportional hazards model that evaluates the likelihood of an employee still being at their job after X days of employement using the following features:

1. The log of their salary
2. The log of their years of seniority
3. The type of department they belong to
4. The company they work for
5. An interaction term between company and department
6. The number of employees in the company

It is trained on 80% of the data, and then predicts the likelihood that the other 20% of employees will still be working at the company in 1, 2, 3, and 4 years, given the characteristics above

The main assumption of the model is that time is the most important predictor of when an employee will leave, which makes sense. Other than that, the cox model shows that salary and seniority also matter. If I could have one other variable, it would be employee satisifcation (e.g., Glassdoor) or the number of times an employee has been promoted.

```
In [47]: #Some variation by company and department
         #Make interaction variable between department and company number
         employee_data['ln_salary'] = np.log(employee_data.salary)
         employee_data['ln_seniority'] = np.log(employee_data.seniority)
```

```
employee_data['dept_code'] = employee_data.dept.astype('category').cat.codes+1
employee_data['dept_company_interaction'] = employee_data.company_id*employee_data.dept
employee_data = employee_data.merge(pd.DataFrame(employee_data.groupby('company_id').ag
```

In [51]:
```
from lifelines import CoxPHFitter
from sklearn.model_selection import train_test_split
X_train, X_test = train_test_split(employee_data[['ln_salary', 'ln_seniority','dept_cod

cph = CoxPHFitter()
cph.fit(X_train, duration_col='tenure_int', event_col='quit_binary', show_progress=True

cph.print_summary()
```

```
Iteration 1: norm_delta = 0.14264, step_size = 0.9500, ll = -95100.23781, newton_decrement = 30.
Iteration 2: norm_delta = 0.00963, step_size = 0.9500, ll = -95069.70495, newton_decrement = 0.1
Iteration 3: norm_delta = 0.00049, step_size = 0.9500, ll = -95069.58055, newton_decrement = 0.0
Iteration 4: norm_delta = 0.00003, step_size = 1.0000, ll = -95069.58023, newton_decrement = 0.0
Iteration 5: norm_delta = 0.00000, step_size = 1.0000, ll = -95069.58023, newton_decrement = 0.0
Convergence completed after 5 iterations.
<lifelines.CoxPHFitter: fitted with 19760 observations, 8988 censored>
      duration col = 'tenure_int'
         event col = 'quit_binary'
number of subjects = 19760
  number of events = 10772
    log-likelihood = -95069.58
  time fit was run = 2019-02-20 20:01:13 UTC

---
                          coef  exp(coef)  se(coef)     z      p  -log2(p)  lower 0.95  upper 0.
ln_salary                -0.17       0.85      0.03 -6.67 <0.005     35.20       -0.22       -0.
ln_seniority              0.13       1.14      0.02  7.01 <0.005     38.61        0.09        0.
dept_code                 0.02       1.02      0.01  2.03   0.04      4.57        0.00        0.
dept_company_interaction  0.00       1.00      0.00  1.02   0.31      1.70       -0.00        0.
company_id               -0.00       1.00      0.01 -0.11   0.91      0.14       -0.02        0.
num_employees             0.00       1.00      0.00  0.11   0.91      0.13       -0.00        0.
---
Concordance = 0.53
Likelihood ratio test = 61.32 on 6 df, -log2(p)=35.26
```

In [64]:
```
pd.DataFrame(cph.predict_survival_function(X_test, times=[365, 730, 1095, 1460]).T).ren
```

Out[64]:

	1 year	2 years	3 years	4 years
16348	0.688507	0.436675	0.244203	0.031192
12311	0.650588	0.385072	0.197162	0.018428
13229	0.643370	0.375652	0.189026	0.016613
4421	0.652341	0.387379	0.199176	0.018894
1456	0.630502	0.359176	0.175138	0.013770
11024	0.660872	0.398715	0.209195	0.021318

9401	0.701807	0.455622	0.262504	0.037260
4258	0.637648	0.368274	0.182753	0.015290
6864	0.652222	0.387222	0.199038	0.018862
2564	0.635963	0.366118	0.180937	0.014919
14578	0.636829	0.367226	0.181869	0.015108
18001	0.648958	0.382933	0.195303	0.018003
4546	0.691329	0.440658	0.248004	0.032400
22066	0.645425	0.378321	0.191317	0.017113
11892	0.667228	0.407278	0.216896	0.023301
13169	0.698668	0.451110	0.258096	0.035740
22713	0.630665	0.359381	0.175308	0.013803
1201	0.645710	0.378692	0.191636	0.017183
23559	0.666409	0.406168	0.215892	0.023036
1684	0.635914	0.366056	0.180884	0.014908
20073	0.639949	0.371232	0.185258	0.015810
7131	0.664437	0.403506	0.213489	0.022411
9870	0.646757	0.380056	0.192813	0.017444
16264	0.636864	0.367270	0.181906	0.015116
13354	0.675992	0.419250	0.227855	0.026304
23634	0.634894	0.364754	0.179791	0.014687
501	0.637016	0.367465	0.182070	0.015150
22563	0.660856	0.398693	0.209175	0.021313
1972	0.633372	0.362814	0.178167	0.014363
20448	0.653998	0.389567	0.201094	0.019345
...	...	...	...	...
23478	0.655060	0.390972	0.202329	0.019638
115	0.643706	0.376087	0.189399	0.016694
13677	0.641237	0.372892	0.186670	0.016108
22767	0.659304	0.396619	0.207326	0.020853
18326	0.603967	0.326476	0.148884	0.009235
2498	0.654477	0.390200	0.201650	0.019477
21071	0.689174	0.437614	0.245097	0.031474
22086	0.635719	0.365806	0.180674	0.014865
8960	0.639098	0.370136	0.184328	0.015616
6173	0.630291	0.358908	0.174916	0.013727
13771	0.638682	0.369602	0.183876	0.015522
22921	0.625484	0.352860	0.169930	0.012785
1199	0.670784	0.412113	0.221295	0.024480
20572	0.677682	0.421579	0.230014	0.026921
22628	0.671947	0.413700	0.222747	0.024877
14305	0.653851	0.389372	0.200923	0.019304
1692	0.614104	0.338766	0.158544	0.010780
23699	0.648639	0.382516	0.194941	0.017921
14280	0.650660	0.385166	0.197244	0.018446
8551	0.649452	0.383581	0.195864	0.018131
24305	0.615533	0.340518	0.159942	0.011015
11448	0.649806	0.384045	0.196268	0.018223
19399	0.636465	0.366759	0.181476	0.015028

```
8361   0.665822  0.405376  0.215175  0.022849
21683  0.667721  0.407946  0.217501  0.023461
18521  0.677450  0.421259  0.229716  0.026836
11808  0.655870  0.392047  0.203277  0.019865
22858  0.682669  0.428497  0.236472  0.028819
1608   0.649480  0.383617  0.195897  0.018138
13626  0.681020  0.426203  0.234322  0.028179

[4940 rows x 4 columns]
```