CSML1000 - Group A - Course Project - REPORT

March 18, 2021

1 Human Resources Analysis - Predicting Employee Attrition

By: Katrina James, Christian Urday, Justin Kerry

1.1 Introductory Statement

Companies of all types have very specific processes for finding, screening, recruiting, and training job applicants. These processes are typically conducted by a division of the company known as Human Resources or "HR". This department of the company not only oversees the development of the the potential employees but it is also responsible for administering employee benefits programs, updating company knowledge of laws and bylaws, as well as firing employees when necessary.

Since Human Resources must be charged with the recruitment of employees, it would be very beneficial to the company to have a method of determining which employees will ultimately be lost through the natural process of attrition.

1.2 Background

For our Course Project, we will be studying Staff Attrition not Staff Turnover.

- Staff Attrition is the loss of employees through a natural process without the intention of filling the vacancy left by the former employee.
- Staff Turnover, on the other hand, is the voluntary or involuntary loss of employees with the intention of filling the vacancy left by the former employee.

Let us examine some of the reasons why employee attrition might occur in a given company: - Retirement - Resignation - Elimination of a Position - Personal Health - Other Personal Reasons

1.3 Objective

We will be using the unsupervised machine learning method of K-Means to segment the employees of the company into clusters.

Then we will use the supervised machine learning method of Random Forest to determine which employees will experience attrition within the company and which will not. The Random Forest method will also be used to give us our feature importance rankings.

Finally, we will use Survival Analysis to predict the rates of survival for each of the clusters that will be generated from employee segmentation. We will be using our feature importance rankings as the baseline for this type of analysis.

1.4 Dataset Analysis

First, we will import our libraries and provide the dataset as a viewable dataframe.

[1]:
```
import warnings
warnings.filterwarnings("ignore")
import imblearn
from imblearn.over_sampling import SMOTENC
import numpy as np
import os
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from jupyterthemes import jtplot
jtplot.style(theme='monokai',context='notebook',ticks=True, grid=False)
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.model_selection import GridSearchCV, train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.cluster import KMeans
import lifelines

pd.set_option("display.max_columns", None)
```

[2]:
```
hr_df = pd.read_csv(os.path.join('Human_resources.csv'))
hr_df.head()
```

[2]:
```
   Age Attrition     BusinessTravel  DailyRate              Department  \
0   41       Yes      Travel_Rarely       1102                   Sales
1   49        No  Travel_Frequently        279  Research & Development
2   37       Yes      Travel_Rarely       1373  Research & Development
3   33        No  Travel_Frequently       1392  Research & Development
4   27        No      Travel_Rarely        591  Research & Development

   DistanceFromHome  Education EducationField  EmployeeCount  EmployeeNumber  \
0                 1          2  Life Sciences              1               1
1                 8          1  Life Sciences              1               2
2                 2          2          Other              1               4
3                 3          4  Life Sciences              1               5
4                 2          1        Medical              1               7

   EnvironmentSatisfaction  Gender  HourlyRate  JobInvolvement  JobLevel  \
```

```
0                         2  Female         94               3         2
1                         3    Male         61               2         2
2                         4    Male         92               2         1
3                         4  Female         56               3         1
4                         1    Male         40               3         1

                  JobRole  JobSatisfaction MaritalStatus  MonthlyIncome  \
0         Sales Executive                4        Single           5993
1      Research Scientist                2       Married           5130
2   Laboratory Technician                3        Single           2090
3      Research Scientist                3       Married           2909
4   Laboratory Technician                2       Married           3468

   MonthlyRate  NumCompaniesWorked Over18 OverTime  PercentSalaryHike  \
0        19479                   8      Y      Yes                 11
1        24907                   1      Y       No                 23
2         2396                   6      Y      Yes                 15
3        23159                   1      Y      Yes                 11
4        16632                   9      Y       No                 12

   PerformanceRating  RelationshipSatisfaction  StandardHours  \
0                  3                         1             80
1                  4                         4             80
2                  3                         2             80
3                  3                         3             80
4                  3                         4             80

   StockOptionLevel  TotalWorkingYears  TrainingTimesLastYear  \
0                 0                  8                      0
1                 1                 10                      3
2                 0                  7                      3
3                 0                  8                      3
4                 1                  6                      3

   WorkLifeBalance  YearsAtCompany  YearsInCurrentRole  \
0                1               6                   4
1                3              10                   7
2                3               0                   0
3                3               8                   7
4                3               2                   2

   YearsSinceLastPromotion  YearsWithCurrManager
0                        0                     5
1                        1                     7
2                        0                     0
3                        3                     0
4                        2                     2
```

We can see our dataframe has numerical, categorical, and ranking-based features within. We will address these different data types during the Data Preprocessing stage.

Many of the categorical features contain different outputs. We will be assigning numerical values to differentiate the outputs from each other in these types of columns.

Now let us gather some statistical details regarding our dataframe.

[3]:
```
hr_df.describe()
```

[3]:
```
               Age    DailyRate  DistanceFromHome    Education  EmployeeCount  \
count  1470.000000  1470.000000       1470.000000  1470.000000         1470.0
mean     36.923810   802.485714          9.192517     2.912925            1.0
std       9.135373   403.509100          8.106864     1.024165            0.0
min      18.000000   102.000000          1.000000     1.000000            1.0
25%      30.000000   465.000000          2.000000     2.000000            1.0
50%      36.000000   802.000000          7.000000     3.000000            1.0
75%      43.000000  1157.000000         14.000000     4.000000            1.0
max      60.000000  1499.000000         29.000000     5.000000            1.0

       EmployeeNumber  EnvironmentSatisfaction   HourlyRate  JobInvolvement  \
count     1470.000000              1470.000000  1470.000000     1470.000000
mean      1024.865306                 2.721769    65.891156        2.729932
std        602.024335                 1.093082    20.329428        0.711561
min          1.000000                 1.000000    30.000000        1.000000
25%        491.250000                 2.000000    48.000000        2.000000
50%       1020.500000                 3.000000    66.000000        3.000000
75%       1555.750000                 4.000000    83.750000        3.000000
max       2068.000000                 4.000000   100.000000        4.000000

          JobLevel  JobSatisfaction  MonthlyIncome   MonthlyRate  \
count  1470.000000      1470.000000    1470.000000   1470.000000
mean      2.063946         2.728571    6502.931293  14313.103401
std       1.106940         1.102846    4707.956783   7117.786044
min       1.000000         1.000000    1009.000000   2094.000000
25%       1.000000         2.000000    2911.000000   8047.000000
50%       2.000000         3.000000    4919.000000  14235.500000
75%       3.000000         4.000000    8379.000000  20461.500000
max       5.000000         4.000000   19999.000000  26999.000000

       NumCompaniesWorked  PercentSalaryHike  PerformanceRating  \
count         1470.000000        1470.000000        1470.000000
mean             2.693197          15.209524           3.153741
std              2.498009           3.659938           0.360824
min              0.000000          11.000000           3.000000
25%              1.000000          12.000000           3.000000
50%              2.000000          14.000000           3.000000
75%              4.000000          18.000000           3.000000
```

```
max           9.000000       25.000000          4.000000

       RelationshipSatisfaction  StandardHours  StockOptionLevel  \
count               1470.000000         1470.0       1470.000000
mean                   2.712245           80.0          0.793878
std                    1.081209            0.0          0.852077
min                    1.000000           80.0          0.000000
25%                    2.000000           80.0          0.000000
50%                    3.000000           80.0          1.000000
75%                    4.000000           80.0          1.000000
max                    4.000000           80.0          3.000000

       TotalWorkingYears  TrainingTimesLastYear  WorkLifeBalance  \
count        1470.000000            1470.000000      1470.000000
mean           11.279592               2.799320         2.761224
std             7.780782               1.289271         0.706476
min             0.000000               0.000000         1.000000
25%             6.000000               2.000000         2.000000
50%            10.000000               3.000000         3.000000
75%            15.000000               3.000000         3.000000
max            40.000000               6.000000         4.000000

       YearsAtCompany  YearsInCurrentRole  YearsSinceLastPromotion  \
count     1470.000000         1470.000000              1470.000000
mean         7.008163            4.229252                 2.187755
std          6.126525            3.623137                 3.222430
min          0.000000            0.000000                 0.000000
25%          3.000000            2.000000                 0.000000
50%          5.000000            3.000000                 1.000000
75%          9.000000            7.000000                 3.000000
max         40.000000           18.000000                15.000000

       YearsWithCurrManager
count           1470.000000
mean               4.123129
std                3.568136
min                0.000000
25%                2.000000
50%                3.000000
75%                7.000000
max               17.000000
```

After conducting some analysis on this data, we can see that some of the features are creating unnecessary noise in our dataframe. These features include; Employee Count, Employee Number, and Standard Hours. We can drop these columns from our dataframe since they do not provide us with any valuable information regarding employee attrition and survival.

Another element for us to consider is Age. We will not drop Age since it is very relevant, however we can see from these statistics that the minimum Age is 18. This statistic renders the Over18 column rather useless since it will only contain 1 value. This is due to every employee in the dataframe being in the age of majority.

Keeping all of this in mind, we will be dropping Over18 as well.

[4]:
```
hr_df = hr_df.drop(columns=['EmployeeCount', 'StandardHours', 'EmployeeNumber',␣
 ↪'Over18'])
hr_df.head()
```

[4]:
```
   Age Attrition     BusinessTravel  DailyRate              Department  \
0   41       Yes      Travel_Rarely       1102                   Sales
1   49        No  Travel_Frequently        279  Research & Development
2   37       Yes      Travel_Rarely       1373  Research & Development
3   33        No  Travel_Frequently       1392  Research & Development
4   27        No      Travel_Rarely        591  Research & Development

   DistanceFromHome  Education EducationField  EnvironmentSatisfaction  \
0                 1          2  Life Sciences                        2
1                 8          1  Life Sciences                        3
2                 2          2          Other                        4
3                 3          4  Life Sciences                        4
4                 2          1        Medical                        1

   Gender  HourlyRate  JobInvolvement  JobLevel                JobRole  \
0  Female          94               3         2        Sales Executive
1    Male          61               2         2     Research Scientist
2    Male          92               2         1  Laboratory Technician
3  Female          56               3         1     Research Scientist
4    Male          40               3         1  Laboratory Technician

   JobSatisfaction MaritalStatus  MonthlyIncome  MonthlyRate  \
0                4        Single           5993        19479
1                2       Married           5130        24907
2                3        Single           2090         2396
3                3       Married           2909        23159
4                2       Married           3468        16632

   NumCompaniesWorked OverTime  PercentSalaryHike  PerformanceRating  \
0                   8      Yes                 11                  3
1                   1       No                 23                  4
2                   6      Yes                 15                  3
3                   1      Yes                 11                  3
4                   9       No                 12                  3

   RelationshipSatisfaction  StockOptionLevel  TotalWorkingYears  \
0                         1                 0                  8
1                         4                 1                 10
```

```
2                        2                  0                    7
3                        3                  0                    8
4                        4                  1                    6

   TrainingTimesLastYear  WorkLifeBalance  YearsAtCompany  YearsInCurrentRole  \
0                      0                1               6                   4
1                      3                3              10                   7
2                      3                3               0                   0
3                      3                3               8                   7
4                      3                3               2                   2

   YearsSinceLastPromotion  YearsWithCurrManager
0                        0                     5
1                        1                     7
2                        0                     0
3                        3                     0
4                        2                     2
```

_

We will now provide a histogram matrix to visualize all of the currently remaining features. This will show us how the data in the dataset is distributed.

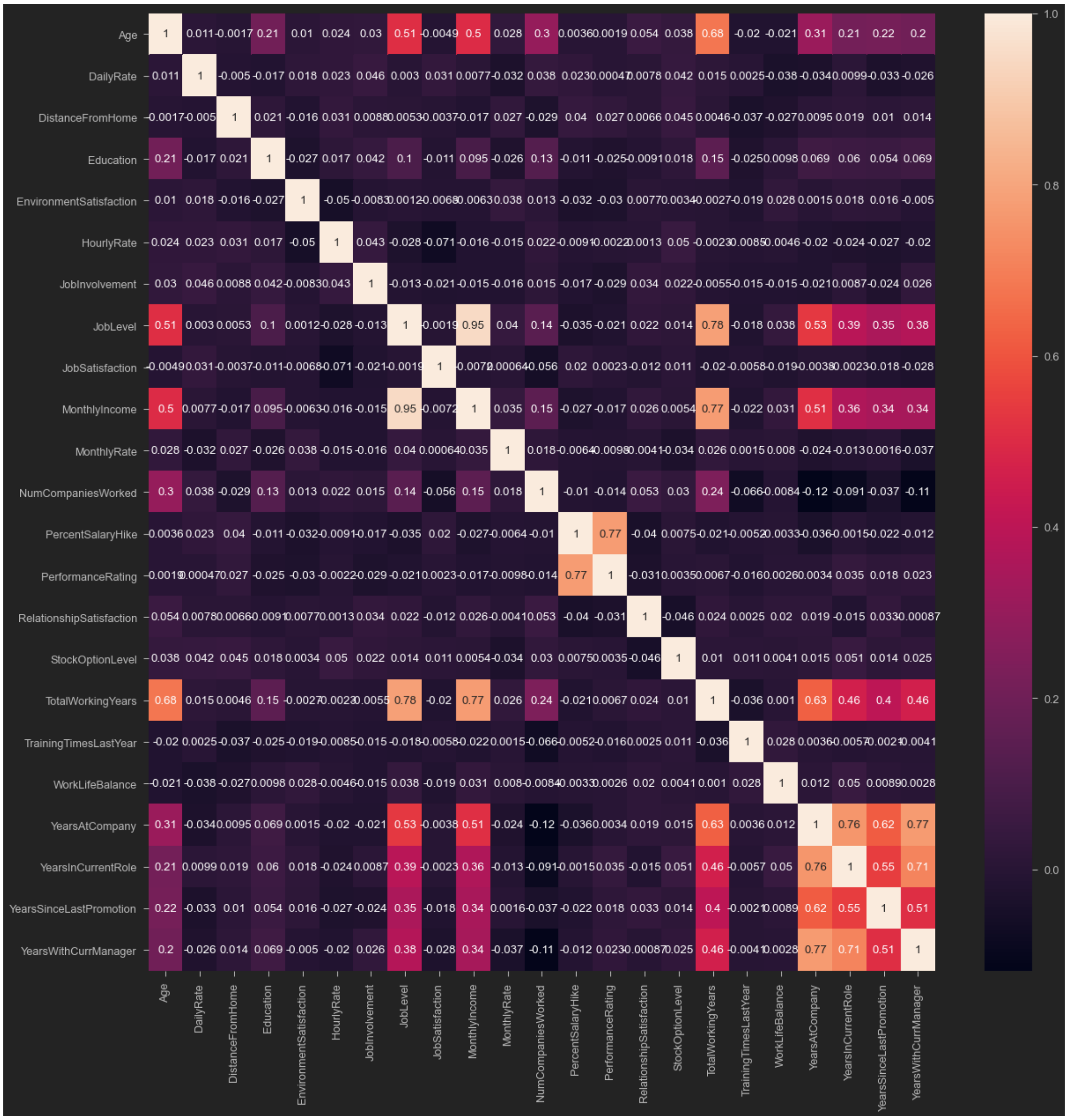
[5]:
```python
hr_df.hist(bins=30, figsize=(20,20), color='b');
```

1.5 Data Preprocessing

Before we begin our data preprocessing, we can use a correlation matrix to visualize the relationship between our features.

[6]:
```
corr_df = hr_df.corr()
f, ax = plt.subplots(figsize=(20,20))
sns.heatmap(corr_df,
            xticklabels=corr_df.columns,
            yticklabels=corr_df.columns,
           annot=True)
```

[6]: <AxesSubplot:>



After analyzing the correlation matrix, we can see that Monthly Income and Job Level are highly correlated (0.95).

We will need to eliminate one to reduce multicolinearity. Since Monthly Income provides greater nuance, we will drop Job Level from the dataframe.

---

[7]:
```
hr_df = hr_df.drop(columns=['JobLevel'])
hr_df.head()
```

[7]:
```
   Age Attrition     BusinessTravel  DailyRate              Department  \
0   41       Yes      Travel_Rarely       1102                   Sales
```

```
1   49         No  Travel_Frequently       279  Research & Development
2   37        Yes      Travel_Rarely      1373  Research & Development
3   33         No  Travel_Frequently      1392  Research & Development
4   27         No      Travel_Rarely       591  Research & Development

   DistanceFromHome  Education EducationField  EnvironmentSatisfaction  \
0                 1          2  Life Sciences                        2
1                 8          1  Life Sciences                        3
2                 2          2          Other                        4
3                 3          4  Life Sciences                        4
4                 2          1        Medical                        1

   Gender  HourlyRate  JobInvolvement                JobRole  JobSatisfaction  \
0  Female          94               3        Sales Executive                4
1    Male          61               2     Research Scientist                2
2    Male          92               2  Laboratory Technician                3
3  Female          56               3     Research Scientist                3
4    Male          40               3  Laboratory Technician                2

  MaritalStatus  MonthlyIncome  MonthlyRate  NumCompaniesWorked OverTime  \
0        Single           5993        19479                   8      Yes
1       Married           5130        24907                   1       No
2        Single           2090         2396                   6      Yes
3       Married           2909        23159                   1      Yes
4       Married           3468        16632                   9       No

   PercentSalaryHike  PerformanceRating  RelationshipSatisfaction  \
0                 11                  3                         1
1                 23                  4                         4
2                 15                  3                         2
3                 11                  3                         3
4                 12                  3                         4

   StockOptionLevel  TotalWorkingYears  TrainingTimesLastYear  \
0                 0                  8                      0
1                 1                 10                      3
2                 0                  7                      3
3                 0                  8                      3
4                 1                  6                      3

   WorkLifeBalance  YearsAtCompany  YearsInCurrentRole  \
0                1               6                   4
1                3              10                   7
2                3               0                   0
3                3               8                   7
4                3               2                   2
```

```
   YearsSinceLastPromotion  YearsWithCurrManager
0                        0                     5
1                        1                     7
2                        0                     0
3                        3                     0
4                        2                     2
```

We will now encode our dependent features, which will be Attrition and OverTime.

[8]:
```
hr_df['Attrition'] = np.where(hr_df['Attrition'] == 'Yes', 1, 0)
hr_df['OverTime'] = np.where(hr_df['OverTime'] == 'Yes', 1, 0)
```

We will need to define our categorical and numerical data type features separately.

Let us define the categorical valued features as X_categ.

[9]:
```
X_categ =hr_df[['BusinessTravel', 'Department', 'Education', 'EducationField',␣
 ↪'EnvironmentSatisfaction',
            'Gender', 'JobInvolvement', 'JobRole', 'JobSatisfaction',␣
 ↪'MaritalStatus', 'OverTime',
            'PerformanceRating', 'RelationshipSatisfaction', 'StockOptionLevel',␣
 ↪'WorkLifeBalance']]
```

These categorical and ranking-based dummies can be included with the original columns under the X_categ dataframe.

Next we will define our numeric features into a dataset which we will call X_numeric.

[10]:
```
X_numeric=hr_df[['Age', 'DailyRate', 'DistanceFromHome', 'HourlyRate',␣
 ↪'MonthlyIncome',
                        'MonthlyRate', 'NumCompaniesWorked', 'PercentSalaryHike',
                       'TotalWorkingYears', 'TrainingTimesLastYear',␣
 ↪'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',␣
 ↪'YearsWithCurrManager' ]]

numeric_cols=['Age', 'DailyRate', 'DistanceFromHome', 'HourlyRate',␣
 ↪'MonthlyIncome',
                        'MonthlyRate', 'NumCompaniesWorked', 'PercentSalaryHike',
                       'TotalWorkingYears', 'TrainingTimesLastYear',␣
 ↪'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',␣
 ↪'YearsWithCurrManager' ]
```

Since we have defined both our X_categ (including categorical and ranking-based features) and our X_numeric dataframes, we can now scale the X_numeric values separately without scaling the categorical features.

Next we can begin our Clustering, Prediction model, and Survival Analysis.

---

2 General Approach:

2.1 1. Identifying the Clusters

Using K-Means Clustering as an Unsupervised Model to segment the employees in our dataset.

2.2 2. Prediction Model

Using Random Forest as a Supervised Model to predict which employees will experience attrition within the company as well as which features in our dataset are most important.

2.3 3. Survival Analysis

Using Survival Analysis to determine which clusters of employees are most likely to survive and which clusters of employees are least likely to survive. We can use our top features along with a couple of other columns to determine this.

2.4 1. Identifying the Clusters:

2.4.1 Finding the optimal number of clusters using the "Elbow method"

In order to determine how many clusters we should segment the employees into, we can use the Elbow Method.

First we will need to scale our X_numeric dataframe in order to optimize the K-Means clustering algorithm.

2.4.2 Scaling and Transforming Data

[11]:
```
scaler=StandardScaler()
X_scaled=scaler.fit_transform(X_numeric)
X_scaled_df=pd.DataFrame(X_scaled, columns=numeric_cols)
X_scaled_df
```

[11]:
```
        Age  DailyRate  DistanceFromHome  HourlyRate  MonthlyIncome  \
0  0.446350   0.742527         -1.010909    1.383138      -0.108350
1  1.322365  -1.297775         -0.147150   -0.240677      -0.291719
2  0.008343   1.414363         -0.887515    1.284725      -0.937654
3 -0.429664   1.461466         -0.764121   -0.486709      -0.763634
```

```
4    -1.086676  -0.524295         -0.887515   -1.274014      -0.644858
…        …         …              …         …            …
1465 -0.101159   0.202082          1.703764   -1.224807      -0.835451
1466  0.227347  -0.469754         -0.393938   -1.175601       0.741140
1467 -1.086676  -1.605183         -0.640727    1.038693      -0.076690
1468  1.322365   0.546677         -0.887515   -0.142264      -0.236474
1469 -0.320163  -0.432568         -0.147150    0.792660      -0.445978

      MonthlyRate  NumCompaniesWorked  PercentSalaryHike  TotalWorkingYears  \
0        0.726020            2.125136          -1.150554          -0.421642
1        1.488876           -0.678049           2.129306          -0.164511
2       -1.674841            1.324226          -0.057267          -0.550208
3        1.243211           -0.678049          -1.150554          -0.421642
4        0.325900            2.525591          -0.877232          -0.678774
…           …                  …                  …                 …
1465    -0.284329            0.523316           0.489376           0.735447
1466     1.004010            0.523316          -0.057267          -0.293077
1467    -1.284418           -0.678049           1.309341          -0.678774
1468    -0.150393           -0.277594          -0.330589           0.735447
1469    -0.574124           -0.277594          -0.877232          -0.678774

      TrainingTimesLastYear  YearsAtCompany  YearsInCurrentRole  \
0                 -2.171982       -0.164613           -0.063296
1                  0.155707        0.488508            0.764998
2                  0.155707       -1.144294           -1.167687
3                  0.155707        0.161947            0.764998
4                  0.155707       -0.817734           -0.615492
…                     …              …                   …
1465               0.155707       -0.327893           -0.615492
1466               1.707500       -0.001333            0.764998
1467              -2.171982       -0.164613           -0.615492
1468               0.155707        0.325228            0.488900
1469               0.155707       -0.491174           -0.339394

      YearsSinceLastPromotion  YearsWithCurrManager
0                   -0.679146              0.245834
1                   -0.368715              0.806541
2                   -0.679146             -1.155935
3                    0.252146             -1.155935
4                   -0.058285             -0.595227
…                      …                    …
1465                -0.679146             -0.314873
1466                -0.368715              0.806541
1467                -0.679146             -0.314873
1468                -0.679146              1.086895
1469                -0.368715             -0.595227
```

```
[1470 rows x 14 columns]
```

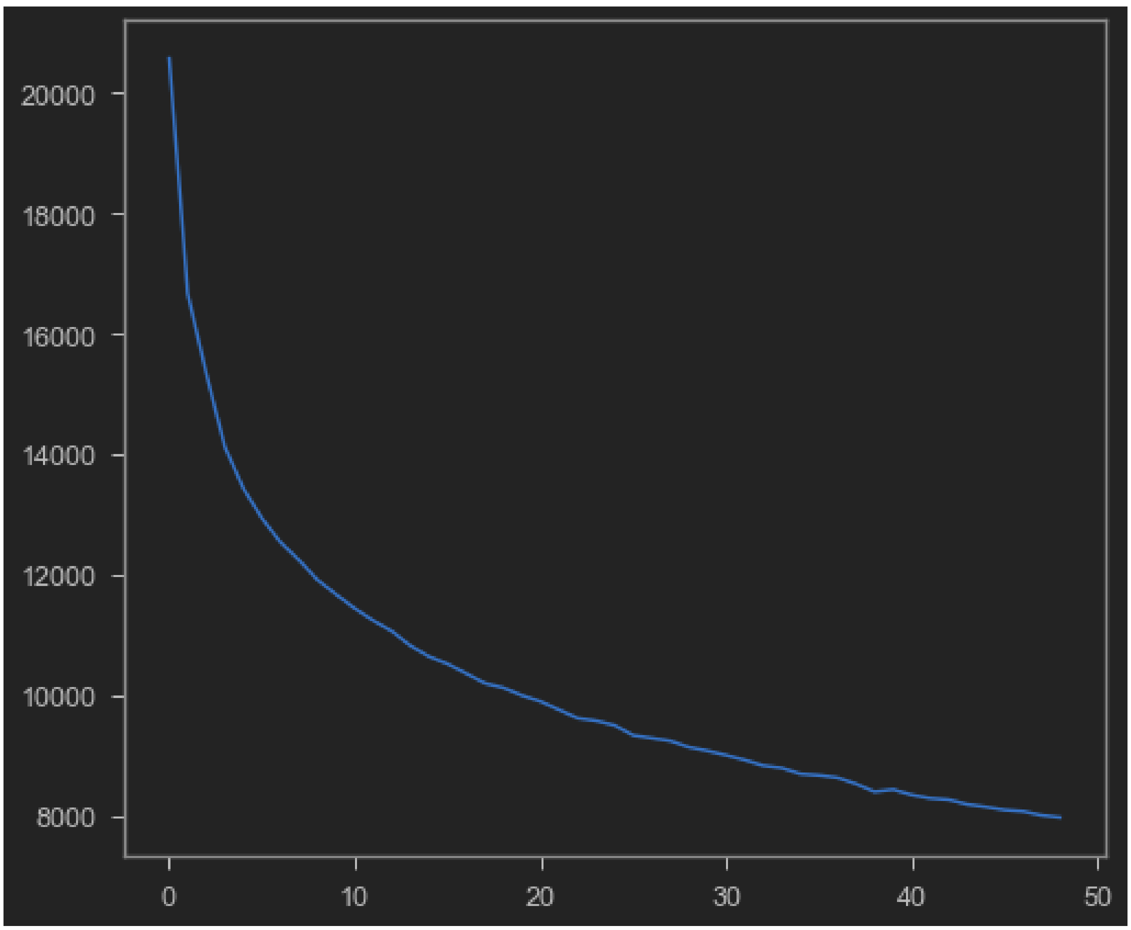
This scaled numeric data will be defined as X_scaled_df.

2.4.3 Using the Elbow Method to determine Clusters

We will use our X_scaled_df to develop an 'Elbow Method' K-Means plot.

[12]:
```python
scores=[]
range_val=range(1,50)
for i in range_val:
    kmeans=KMeans(n_clusters=i)
    kmeans.fit(X_scaled_df)
    scores.append(kmeans.inertia_)
plt.plot(scores,'bx-')
```

[12]:
```
[<matplotlib.lines.Line2D at 0x11bbdd880>]
```



Based on the inertia of this K-Means plot, we can see an elbow forming around 10 clusters. For the purpose of this study, we have rounded this value to the 10 cluster mark.

Now that we have the amount of clusters that we will use for the K-Means algorithm, we can proceed with the segmentation of the employees in our dataframe using clustering.

2.4.4 Applying K-Means Method

We will apply the K-Means algorithm to our X_scaled_df dataframe with the parameter of 10 clusters.

[13]:
```
kmeans=KMeans(10)
kmeans.fit(X_scaled_df)
labels=kmeans.labels_
```

[14]:
```
cluster_centers=pd.DataFrame(data=kmeans.cluster_centers_,columns=[X_numeric.
 ↪columns])
cluster_centers.head()
```

[14]:
```
        Age  DailyRate  DistanceFromHome  HourlyRate  MonthlyIncome  MonthlyRate  \
0 -0.733485   0.986682         -0.135024   -0.167009      -0.601853    -0.211136
1 -0.802477  -0.935553          0.151707   -0.606739      -0.651613     0.334728
2 -0.105920   0.044370         -0.365162   -0.372931      -0.098261    -0.119801
3  1.001589  -0.193437         -0.052789    0.001882       1.700984    -0.100774
4 -0.254883  -0.504303         -0.484902    0.991063      -0.400939    -0.435741

  NumCompaniesWorked PercentSalaryHike TotalWorkingYears  \
0          -0.597032         -0.027249         -0.800651
1          -0.409536          0.209735         -0.803623
2          -0.389975          0.012954          0.065179
3           0.141706          0.016690          1.680782
4          -0.400811         -0.430457         -0.443894

  TrainingTimesLastYear YearsAtCompany YearsInCurrentRole  \
0             -0.319698      -0.557241          -0.628259
1             -0.378002      -0.655398          -0.684117
2              0.164907       0.537556           0.981074
3             -0.045113       1.977239           1.463363
4             -0.296899      -0.336267          -0.396029

  YearsSinceLastPromotion YearsWithCurrManager
0               -0.474585            -0.613053
1               -0.402809            -0.689219
2                0.496318             0.933975
3                0.288667             1.584935
4               -0.382645            -0.307685
```

2.4.5 Applying Inverse Transformation to Understand the Numbers

[15]:
```
cluster_centers=scaler.inverse_transform(cluster_centers)
cluster_centers=pd.DataFrame(data=cluster_centers,columns=[X_numeric.columns])
cluster_centers.head()
```

[15]:
```
         Age    DailyRate DistanceFromHome HourlyRate MonthlyIncome  \
0  30.225434  1200.485549         8.098266  62.497110   3670.398844
1  29.595376   425.109827        10.421965  53.560694   3436.208092
2  35.956522   820.383399         6.233202  58.312253   6040.478261
3  46.070588   724.458824         8.764706  65.929412  14508.364706
4  34.596154   599.064103         5.262821  86.032051   4615.967949

   MonthlyRate NumCompaniesWorked PercentSalaryHike TotalWorkingYears  \
0  12810.791908           1.202312         15.109827          5.052023
1  16694.815029           1.670520         15.976879          5.028902
2  13460.675889           1.719368         15.256917         11.786561
3  13596.058824           3.047059         15.270588         24.352941
4  11212.647436           1.692308         13.634615          7.826923

  TrainingTimesLastYear YearsAtCompany YearsInCurrentRole  \
0              2.387283       3.595376           1.953757
1              2.312139       2.994220           1.751445
2              3.011858      10.300395           7.782609
3              2.741176      19.117647           9.529412
4              2.416667       4.948718           2.794872

  YearsSinceLastPromotion YearsWithCurrManager
0                0.658960             1.936416
1                0.890173             1.664740
2                3.786561             7.454545
3                3.117647             9.776471
4                0.955128             3.025641
```

2.4.6 Concatenating Cluster Labels to Dataset

First let us transform our X_categ array as X_categ_df.

We will want to concatenate the cluster labels with numerical and categorical data features to our new dataframe which we will now define as df_cluster. This will allow us to display which cluster each employee belongs. This new cluster labeled column can be found appended at the end of this dataframe.

[16]:
```
X_categ_df=pd.DataFrame(X_categ)
```

[17]:
```
df_cluster=pd.concat([X_numeric,X_categ_df,pd.DataFrame({'Cluster':labels})],␣
 ↪axis=1)
```

```python
df_cluster.head(10)
```

[17]:

```
   Age  DailyRate  DistanceFromHome  HourlyRate  MonthlyIncome  MonthlyRate  \
0   41       1102                 1          94           5993        19479
1   49        279                 8          61           5130        24907
2   37       1373                 2          92           2090         2396
3   33       1392                 3          56           2909        23159
4   27        591                 2          40           3468        16632
5   32       1005                 2          79           3068        11864
6   59       1324                 3          81           2670         9964
7   30       1358                24          67           2693        13335
8   38        216                23          44           9526         8787
9   36       1299                27          94           5237        16577

   NumCompaniesWorked  PercentSalaryHike  TotalWorkingYears  \
0                   8                 11                  8
1                   1                 23                 10
2                   6                 15                  7
3                   1                 11                  8
4                   9                 12                  6
5                   0                 13                  8
6                   4                 20                 12
7                   1                 22                  1
8                   0                 21                 10
9                   6                 13                 17

   TrainingTimesLastYear  YearsAtCompany  YearsInCurrentRole  \
0                      0               6                   4
1                      3              10                   7
2                      3               0                   0
3                      3               8                   7
4                      3               2                   2
5                      2               7                   7
6                      3               1                   0
7                      2               1                   0
8                      2               9                   7
9                      3               7                   7

   YearsSinceLastPromotion  YearsWithCurrManager     BusinessTravel  \
0                        0                     5      Travel_Rarely
1                        1                     7  Travel_Frequently
2                        0                     0      Travel_Rarely
3                        3                     0  Travel_Frequently
4                        2                     2      Travel_Rarely
5                        3                     6  Travel_Frequently
6                        0                     0      Travel_Rarely
7                        0                     0      Travel_Rarely
```

```
8                       1                     8  Travel_Frequently
9                       7                     7      Travel_Rarely

              Department  Education EducationField  EnvironmentSatisfaction  \
0                  Sales          2  Life Sciences                        2
1  Research & Development         1  Life Sciences                        3
2  Research & Development         2          Other                        4
3  Research & Development         4  Life Sciences                        4
4  Research & Development         1        Medical                        1
5  Research & Development         2  Life Sciences                        4
6  Research & Development         3        Medical                        3
7  Research & Development         1  Life Sciences                        4
8  Research & Development         3  Life Sciences                        4
9  Research & Development         3        Medical                        3

   Gender  JobInvolvement                    JobRole  JobSatisfaction  \
0  Female               3            Sales Executive                4
1    Male               2         Research Scientist                2
2    Male               2      Laboratory Technician                3
3  Female               3         Research Scientist                3
4    Male               3      Laboratory Technician                2
5    Male               3      Laboratory Technician                4
6  Female               4      Laboratory Technician                1
7    Male               3      Laboratory Technician                3
8    Male               2     Manufacturing Director                3
9    Male               3  Healthcare Representative                3

  MaritalStatus  OverTime  PerformanceRating  RelationshipSatisfaction  \
0        Single         1                  3                         1
1       Married         0                  4                         4
2        Single         1                  3                         2
3       Married         1                  3                         3
4       Married         0                  3                         4
5        Single         0                  3                         3
6       Married         1                  4                         1
7      Divorced         0                  4                         2
8        Single         0                  4                         2
9       Married         0                  3                         2

   StockOptionLevel  WorkLifeBalance  Cluster
0                 0                1        7
1                 1                3        2
2                 0                3        7
3                 0                3        0
4                 1                3        7
5                 0                2        2
6                 3                2        7
```

```
7                1                3       0
8                0                3       5
9                2                2       5
```

2.5 2. Random Forest Prediction Modeling:

2.5.1 Scaling the Data

Now that the clustering has been achieved, we can begin constructing the Random Forest machine learning model. We will scale only the continuous values in the dataframe and will not scale the categorical values.

Scaling will involve the continuous valued columns in our original dataset, hr_df, and will be concatenated with the Attrition column to be included with continuous data. This is important since the Attrition column is essentially our target variable.

[18]:
```
cont_cols = ['Age', 'DailyRate', 'DistanceFromHome', 'HourlyRate',␣
↪'MonthlyIncome', 'MonthlyRate',
            'NumCompaniesWorked', 'PercentSalaryHike', 'TotalWorkingYears',␣
↪'TrainingTimesLastYear',
            'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',␣
↪'YearsWithCurrManager']
cat_cols = ['BusinessTravel', 'Department', 'Education', 'EducationField',␣
↪'EnvironmentSatisfaction',
            'Gender', 'JobInvolvement', 'JobRole', 'JobSatisfaction',␣
↪'MaritalStatus', 'OverTime',
            'PerformanceRating', 'RelationshipSatisfaction', 'StockOptionLevel',␣
↪'WorkLifeBalance']
ss = StandardScaler()
hr_scaled_df = ss.fit_transform(hr_df[cont_cols].to_numpy())

hr_scaled_df = pd.concat([pd.DataFrame(hr_scaled_df, columns=cont_cols),
                          hr_df[cat_cols],
                          hr_df['Attrition']], axis=1)
hr_scaled_df.head()
```

[18]:
```
        Age  DailyRate  DistanceFromHome  HourlyRate  MonthlyIncome  \
0  0.446350   0.742527         -1.010909    1.383138      -0.108350
1  1.322365  -1.297775         -0.147150   -0.240677      -0.291719
2  0.008343   1.414363         -0.887515    1.284725      -0.937654
3 -0.429664   1.461466         -0.764121   -0.486709      -0.763634
4 -1.086676  -0.524295         -0.887515   -1.274014      -0.644858

   MonthlyRate  NumCompaniesWorked  PercentSalaryHike  TotalWorkingYears  \
0     0.726020            2.125136          -1.150554          -0.421642
1     1.488876           -0.678049           2.129306          -0.164511
```

```
2   -1.674841           1.324226         -0.057267          -0.550208
3    1.243211          -0.678049         -1.150554          -0.421642
4    0.325900           2.525591         -0.877232          -0.678774

   TrainingTimesLastYear  YearsAtCompany  YearsInCurrentRole  \
0              -2.171982       -0.164613           -0.063296
1               0.155707        0.488508            0.764998
2               0.155707       -1.144294           -1.167687
3               0.155707        0.161947            0.764998
4               0.155707       -0.817734           -0.615492

   YearsSinceLastPromotion  YearsWithCurrManager     BusinessTravel  \
0                -0.679146              0.245834      Travel_Rarely
1                -0.368715              0.806541  Travel_Frequently
2                -0.679146             -1.155935      Travel_Rarely
3                 0.252146             -1.155935  Travel_Frequently
4                -0.058285             -0.595227      Travel_Rarely

               Department  Education EducationField  EnvironmentSatisfaction  \
0                   Sales          2  Life Sciences                        2
1  Research & Development          1  Life Sciences                        3
2  Research & Development          2          Other                        4
3  Research & Development          4  Life Sciences                        4
4  Research & Development          1        Medical                        1

   Gender  JobInvolvement                JobRole  JobSatisfaction  \
0  Female               3        Sales Executive                4
1    Male               2     Research Scientist                2
2    Male               2  Laboratory Technician                3
3  Female               3     Research Scientist                3
4    Male               3  Laboratory Technician                2

  MaritalStatus  OverTime  PerformanceRating  RelationshipSatisfaction  \
0        Single         1                  3                         1
1       Married         0                  4                         4
2        Single         1                  3                         2
3       Married         1                  3                         3
4       Married         0                  3                         4

   StockOptionLevel  WorkLifeBalance  Attrition
0                 0                1          1
1                 1                3          0
2                 0                3          1
3                 0                3          0
4                 1                3          0
```

2.5.2 Balancing the Data

Since we are using the scaled original dataset, hr_scaled_df, we will now need to balance the dataset before we can finish the construction of the Random Forest model.

To achieve this, we will use SMOTE to balance the data.

[19]:
```
dependent_df = hr_scaled_df['Attrition']
independent_df = hr_scaled_df.drop(columns=['Attrition'])
sm = SMOTENC(
    categorical_features=[independent_df.columns.get_loc(col) for col in␣
 ↪cat_cols],
    sampling_strategy='auto',
    k_neighbors=8,
    random_state=42)
independent_resample, dependent_resample = sm.fit_resample(independent_df,␣
 ↪dependent_df)
smoted_df = pd.concat([pd.DataFrame(independent_resample), pd.
 ↪DataFrame(dependent_resample)], axis=1)
smoted_df
```

[19]:
```
           Age  DailyRate  DistanceFromHome  HourlyRate  MonthlyIncome  \
0     0.446350   0.742527         -1.010909    1.383138      -0.108350
1     1.322365  -1.297775         -0.147150   -0.240677      -0.291719
2     0.008343   1.414363         -0.887515    1.284725      -0.937654
3    -0.429664   1.461466         -0.764121   -0.486709      -0.763634
4    -1.086676  -0.524295         -0.887515   -1.274014      -0.644858
…          …          …                …           …            …
2461 -1.378159  -1.180086         -0.302229   -1.691602      -0.830399
2462  0.839179  -0.885440          2.039522   -0.944745       0.862164
2463 -0.994284   0.363404         -0.441497    0.951813      -0.714668
2464 -0.076276  -0.082383         -0.911821    0.927609      -0.454489
2465 -0.387778   0.841682         -0.441138    0.803774       0.115918

      MonthlyRate  NumCompaniesWorked  PercentSalaryHike  TotalWorkingYears  \
0        0.726020            2.125136          -1.150554          -0.421642
1        1.488876           -0.678049           2.129306          -0.164511
2       -1.674841            1.324226          -0.057267          -0.550208
3        1.243211           -0.678049          -1.150554          -0.421642
4        0.325900            2.525591          -0.877232          -0.678774
…             …                  …                  …                  …
2461    -1.092357           -0.780877          -0.393363          -1.255576
2462     0.572545            0.409237          -0.681773           0.828145
2463     1.009532           -0.678049           0.005367          -0.342629
2464    -0.938370            0.681074          -1.042880          -0.448912
2465     0.388720           -0.862595           1.266526          -0.223759

      TrainingTimesLastYear  YearsAtCompany  YearsInCurrentRole  \
```

```
0                  -2.171982       -0.164613           -0.063296
1                   0.155707        0.488508            0.764998
2                   0.155707       -1.144294           -1.167687
3                   0.155707        0.161947            0.764998
4                   0.155707       -0.817734           -0.615492
…                      …              …                  …
2461                0.155707       -0.939088           -1.025896
2462               -0.065325       -0.747482           -0.772797
2463               -0.381764        0.262296            0.510471
2464               -0.620189       -0.199245           -0.343571
2465               -0.201856        0.338016            0.318414

      YearsSinceLastPromotion  YearsWithCurrManager     BusinessTravel  \
0                   -0.679146              0.245834      Travel_Rarely
1                   -0.368715              0.806541  Travel_Frequently
2                   -0.679146             -1.155935      Travel_Rarely
3                    0.252146             -1.155935  Travel_Frequently
4                   -0.058285             -0.595227      Travel_Rarely
…                       …                     …                …
2461                -0.368715             -1.155935      Travel_Rarely
2462                -0.235151             -1.155935  Travel_Frequently
2463                -0.368715              0.289642      Travel_Rarely
2464                -0.429862              0.131148      Travel_Rarely
2465                 0.060460             -0.100394      Travel_Rarely

                  Department  Education EducationField  \
0                      Sales          2  Life Sciences
1     Research & Development          1  Life Sciences
2     Research & Development          2          Other
3     Research & Development          4  Life Sciences
4     Research & Development          1        Medical
…                      …        …            …
2461                   Sales          3      Marketing
2462                   Sales          2  Life Sciences
2463  Research & Development          1        Medical
2464  Research & Development          3  Life Sciences
2465                   Sales          3  Life Sciences

      EnvironmentSatisfaction  Gender  JobInvolvement                JobRole  \
0                           2  Female               3        Sales Executive
1                           3    Male               2     Research Scientist
2                           4    Male               2  Laboratory Technician
3                           4  Female               3     Research Scientist
4                           1    Male               3  Laboratory Technician
…                      …      …             …                   …
2461                        4  Female               3   Sales Representative
2462                        2    Male               3        Sales Executive
```

```
2463                          3    Male              3  Laboratory Technician
2464                          1    Male              3  Laboratory Technician
2465                          3    Male              3        Sales Executive

      JobSatisfaction MaritalStatus  OverTime  PerformanceRating  \
0                   4        Single         1                  3
1                   2       Married         0                  4
2                   3        Single         1                  3
3                   3       Married         1                  3
4                   2       Married         0                  3
…                 …           …       …                 …
2461                3        Single         0                  3
2462                3        Single         1                  3
2463                4        Single         0                  3
2464                1       Married         0                  3
2465                4        Single         0                  3

      RelationshipSatisfaction  StockOptionLevel  WorkLifeBalance  Attrition
0                            1                 0                1          1
1                            4                 1                3          0
2                            2                 0                3          1
3                            3                 0                3          0
4                            4                 1                3          0
…                          …                 …               …        …
2461                         2                 0                3          1
2462                         1                 0                3          1
2463                         1                 0                3          1
2464                         3                 0                3          1
2465                         1                 0                3          1

[2466 rows x 30 columns]
```

2.5.3 Dimension Reduction

Our new balanced dataframe for our Random Forest model has been defined as smoted_df. For our dimension reduction, we will use the Factor Analysis of Mixed Data also known as FAMD. This will be a useful method since we do indeed have mixed data in our smoted_df.

We will be looking at the first 10 components for our Random Forest model.

Using a random_state of 42 can be used as a constant for reproducability.

The Attrition column will be dropped since our Random Forest model will be trying to predict that column.

[20]:
```python
import prince

famd = prince.FAMD(
```

```
    n_components=10,
    n_iter=10,
    copy=True,
    check_input=True,
    engine='auto',
    random_state=42
)

famd_fit = famd.fit(smoted_df.drop(columns=['Attrition']))
principle_components_df = famd_fit.transform(smoted_df.
 ↪drop(columns=['Attrition']))
# smoted_df = pd.concat([smoted_df, principle_components_df], axis=1)

rename_cols = {}
#for col in principle_components_df:
#    rename_cols[col] = 'PC' + str(col + 1)

#smoted_df = smoted_df.rename(columns=rename_cols)
famd_fit.explained_inertia_
```

[20]:
```
array([0.10606221, 0.0639568 , 0.05878143, 0.0428383 , 0.04179812,
       0.03636339, 0.03390576, 0.03075983, 0.02979258, 0.02795983])
```

As before when we were creating our clusters, we will now need to address the categorical columns that contain multiple outputs. We will take our smoted_df, which is balanced and has been processed using dimension reduction, and this time we will assign numerical values for the multi-output categorical columns of this dataframe.

2.5.4 Assigning Numerical Values to Categorical Outputs

[21]:
```
smoted_df['BusinessTravel'] = np.select([
    smoted_df['BusinessTravel'] == 'Non-Travel',
    smoted_df['BusinessTravel'] == 'Travel_Rarely',
    smoted_df['BusinessTravel'] == 'Travel_Frequently'], [0,1,2])

smoted_df['Department'] = np.select([
    smoted_df['Department'] == 'Human Resources',
    smoted_df['Department'] == 'Research & Development',
    smoted_df['Department'] == 'Sales'], [0,1,2])

smoted_df['EducationField'] = np.select([
    smoted_df['EducationField'] == 'Human Resources',
    smoted_df['EducationField'] == 'Life Sciences',
    smoted_df['EducationField'] == 'Marketing',
    smoted_df['EducationField'] == 'Medical',
    smoted_df['EducationField'] == 'Other',
```

```
    smoted_df['EducationField'] == 'Technical Degree'], [0,1,2,3,4,5])

smoted_df['Gender'] = np.select([
    smoted_df['Gender'] == 'Male',
    smoted_df['Gender'] == 'Female'], [0,1])

smoted_df['JobRole'] = np.select([
    smoted_df['JobRole'] == 'Healthcare Representative',
    smoted_df['JobRole'] == 'Human Resources',
    smoted_df['JobRole'] == 'Laboratory Technician',
    smoted_df['JobRole'] == 'Manager',
    smoted_df['JobRole'] == 'Manufacturing Director',
    smoted_df['JobRole'] == 'Research Director',
    smoted_df['JobRole'] == 'Research Scientist',
    smoted_df['JobRole'] == 'Sales Executive',
    smoted_df['JobRole'] == 'Sales Representative'], [0,1,2,3,4,5,6,7,8])

smoted_df['MaritalStatus'] = np.select([
    smoted_df['MaritalStatus'] == 'Single',
    smoted_df['MaritalStatus'] == 'Married',
    smoted_df['MaritalStatus'] == 'Divorced'], [0,1,2])

smoted_df.head()
```

[21]:

```
        Age  DailyRate  DistanceFromHome  HourlyRate  MonthlyIncome  \
0  0.446350   0.742527         -1.010909    1.383138      -0.108350
1  1.322365  -1.297775         -0.147150   -0.240677      -0.291719
2  0.008343   1.414363         -0.887515    1.284725      -0.937654
3 -0.429664   1.461466         -0.764121   -0.486709      -0.763634
4 -1.086676  -0.524295         -0.887515   -1.274014      -0.644858

   MonthlyRate  NumCompaniesWorked  PercentSalaryHike  TotalWorkingYears  \
0     0.726020            2.125136          -1.150554          -0.421642
1     1.488876           -0.678049           2.129306          -0.164511
2    -1.674841            1.324226          -0.057267          -0.550208
3     1.243211           -0.678049          -1.150554          -0.421642
4     0.325900            2.525591          -0.877232          -0.678774

   TrainingTimesLastYear  YearsAtCompany  YearsInCurrentRole  \
0              -2.171982       -0.164613           -0.063296
1               0.155707        0.488508            0.764998
2               0.155707       -1.144294           -1.167687
3               0.155707        0.161947            0.764998
4               0.155707       -0.817734           -0.615492

   YearsSinceLastPromotion  YearsWithCurrManager  BusinessTravel  Department  \
0                -0.679146              0.245834               1           2
```

```
1           -0.368715          0.806541             2           1
2           -0.679146         -1.155935             1           1
3            0.252146         -1.155935             2           1
4           -0.058285         -0.595227             1           1

   Education  EducationField  EnvironmentSatisfaction  Gender  JobInvolvement  \
0          2               1                        2       1               3
1          1               1                        3       0               2
2          2               4                        4       0               2
3          4               1                        4       1               3
4          1               3                        1       0               3

   JobRole  JobSatisfaction  MaritalStatus  OverTime  PerformanceRating  \
0        7                4              0         1                  3
1        6                2              1         0                  4
2        2                3              0         1                  3
3        6                3              1         1                  3
4        2                2              1         0                  3

   RelationshipSatisfaction  StockOptionLevel  WorkLifeBalance  Attrition
0                         1                 0                1          1
1                         4                 1                3          0
2                         2                 0                3          1
3                         3                 0                3          0
4                         4                 1                3          0
```

2.5.5 Splitting Data Into Training Set and Testing Set

The Data Preprocessing for our Random Forest model has now been complete. We can now split the data into the training set and the testing set. We will apply this to the smoted_df since that dataframe has all of the preprocessing already applied to it.

[22]:
```python
x_train, x_test, y_train, y_test = train_test_split(smoted_df.
 ↪drop(columns=['Attrition']),
                                                    smoted_df['Attrition'],
                                                    train_size=0.7,
                                                    random_state=42)

rand_forest_fit = RandomForestClassifier(n_estimators=1000,
                                         criterion="gini",
                                         max_depth=100,
                                         min_samples_split=3,
                                         min_samples_leaf=2)


rand_forest_fit.fit(x_train, y_train)
```

```
print("Random Forest - Train Confusion Matrix\n", pd.crosstab(y_train,
                                                    rand_forest_fit.
 ↪predict(x_train),
                                                    rownames=["Actual"],
                                                    colnames=["Predicted"]))

print("Random Forest - Train accuracy", round(accuracy_score(y_train,␣
 ↪rand_forest_fit.predict(x_train)), 3))

print("Random Forest - Test Confusion Matrix", pd.crosstab(y_test,
                                                    rand_forest_fit.
 ↪predict(x_test),
                                                    rownames=["Actual"],
                                                    colnames=["Predicted"]))

print("Random Forest - Test accuracy", round(accuracy_score(y_test,␣
 ↪rand_forest_fit.predict(x_test)), 3))
```

```
Random Forest - Train Confusion Matrix
 Predicted    0    1
Actual
0           863    1
1             7  855
Random Forest - Train accuracy 0.995
Random Forest - Test Confusion Matrix Predicted    0    1
Actual
0           332   37
1            37  334
Random Forest - Test accuracy 0.9
```

–

Analyzing the confusion matrix of the model, we can see that our Random Forest model has: - Train accuracy of 0.995 - Test accuracy of 0.9

2.5.6 Using Pipeline to Automate Workflow

[23]:
```
pipeline = Pipeline([('clf', RandomForestClassifier(criterion='gini'))])
parameters = {
    'clf__n_estimators': (200, 300, 500),
    'clf__max_depth': (20, 30, 50),
    'clf__min_samples_split': (2, 3),
    'clf__min_samples_leaf': (1, 2)}
grid_search = GridSearchCV(pipeline, parameters, n_jobs=-1, cv=5, verbose=1,␣
 ↪scoring='accuracy')
```

```python
grid_search.fit(x_train, y_train)
print('Best Training score: ' + str(grid_search.best_score_))
print('Best parameters set:')
best_parameters = grid_search.best_estimator_.get_params()

for param_name in sorted(parameters.keys()):
    print(str(param_name) + ': ' + str(best_parameters[param_name]))

predictions = grid_search.predict(x_test)
print("Testing accuracy: " + str(accuracy_score(y_test, predictions)))
print("Complete report of Testing data", classification_report(y_test,␣
 ↪predictions))
print("Random Forest Grid Search - Test Confusion Matrix", pd.crosstab(y_test,␣
 ↪predictions, rownames=["Actual"], colnames=["Predicted"]))
```

```
Fitting 5 folds for each of 36 candidates, totalling 180 fits
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 4 concurrent workers.
[Parallel(n_jobs=-1)]: Done  42 tasks      | elapsed:   42.0s
[Parallel(n_jobs=-1)]: Done 180 out of 180 | elapsed:  2.0min finished
```

```
Best Training score: 0.895141157744827
Best parameters set:
clf__max_depth: 50
clf__min_samples_leaf: 1
clf__min_samples_split: 2
clf__n_estimators: 200
Testing accuracy: 0.9027027027027027
Complete report of Testing data               precision    recall  f1-score
support

           0       0.90      0.90      0.90       369
           1       0.90      0.90      0.90       371

    accuracy                           0.90       740
   macro avg       0.90      0.90      0.90       740
weighted avg       0.90      0.90      0.90       740

Random Forest Grid Search - Test Confusion Matrix Predicted    0    1
Actual
0          333   36
1           36  335
```

–

We can see that our precision, recall, and F1-scores are performing very well.

The weighted average score is 90%.

2.6 3. Survival Analysis

2.6.1 Feature Importance for Survival Analysis

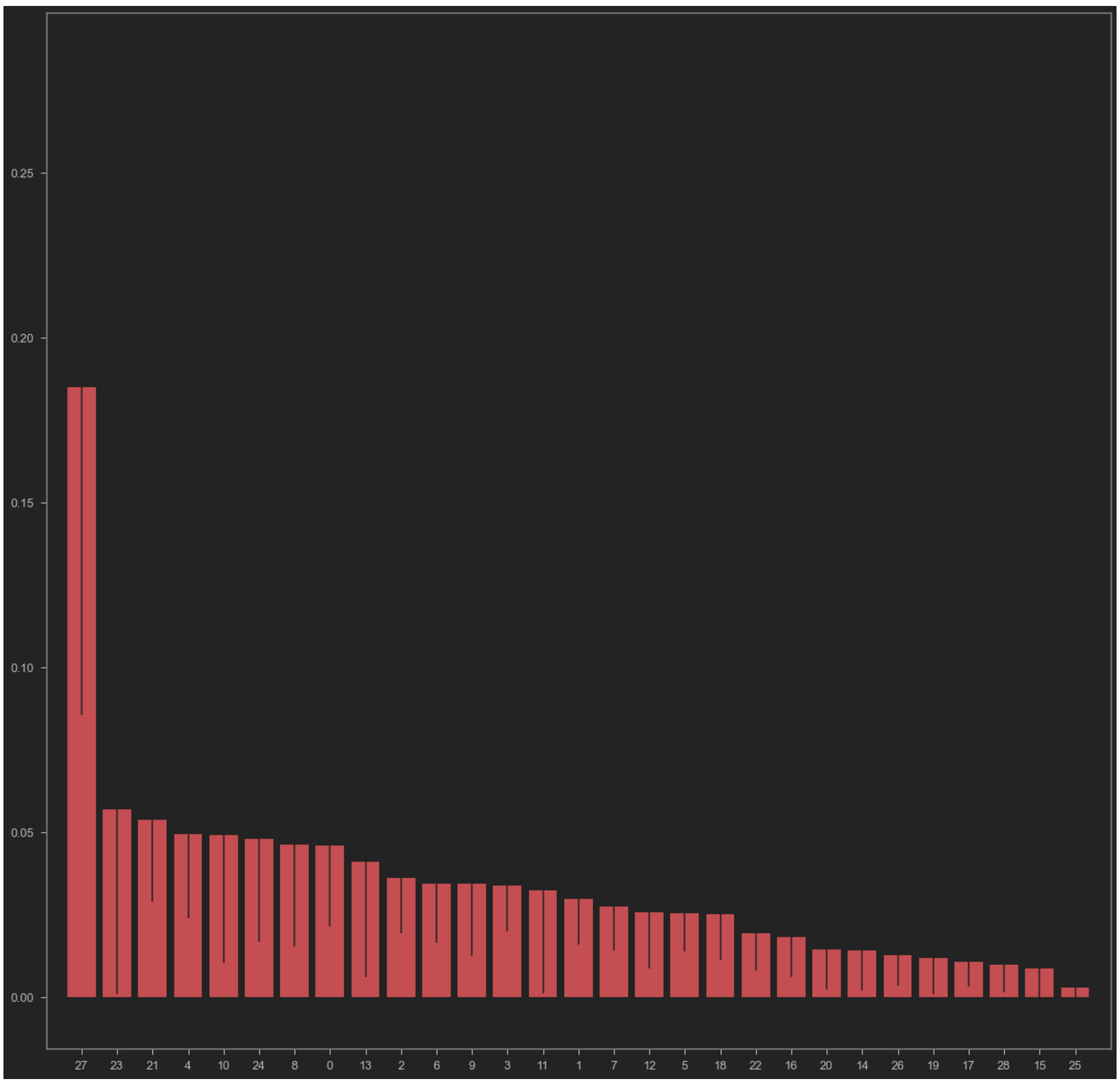
We will use the rand_forest_fit to print the feature ranking of our dataset. This feature ranking list will be derived from our Random Forest machine learning model.

This list will provide us with all of the features ranking in order of importance. We will use the top 10 most important features to provide us with a baseline for our Survival Analysis which will utilize our clusters.

[24]:
```
rand_forest_fit = RandomForestClassifier(n_estimators=500, criterion="gini",␣
↪max_depth=30, min_samples_split=2, min_samples_leaf=2)
rand_forest_fit.fit(x_train, y_train)
importances = rand_forest_fit.feature_importances_
standard_deviations = np.std([tree.feature_importances_ for tree in␣
↪rand_forest_fit.estimators_], axis=0)
indices = np.argsort(importances)[::-1]
column_names = list(x_train.columns)

print("Feature ranking:")
for feature in range(x_train.shape[1]):
    print ("Feature", indices[feature], ",", column_names[indices[feature]],␣
↪importances[indices[feature]])

plt.figure(figsize=(20,20))
plt.bar(range(x_train.shape[1]), importances[indices], color="r",␣
↪yerr=standard_deviations[indices], align="center")
plt.xticks(range(x_train.shape[1]), indices)
plt.xlim([-1, x_train.shape[1]])
plt.show()
```

```
Feature ranking:
Feature 27 , StockOptionLevel 0.1849876109834841
Feature 23 , MaritalStatus 0.056707383020280985
Feature 21 , JobRole 0.05375170087808784
Feature 4 , MonthlyIncome 0.049282408373176524
Feature 10 , YearsAtCompany 0.04907925812734965
Feature 24 , OverTime 0.04781780226598092
Feature 8 , TotalWorkingYears 0.04601871793477756
Feature 0 , Age 0.0458432807982385
Feature 13 , YearsWithCurrManager 0.04104578808825855
Feature 2 , DistanceFromHome 0.03614575104887354
Feature 6 , NumCompaniesWorked 0.034301537176991866
Feature 9 , TrainingTimesLastYear 0.03421518020933393
Feature 3 , HourlyRate 0.033761407852792025
Feature 11 , YearsInCurrentRole 0.03215452088087747
Feature 1 , DailyRate 0.02971423436426313
```

```
Feature 7 , PercentSalaryHike 0.02729605719421296
Feature 12 , YearsSinceLastPromotion 0.0257258766044947
Feature 5 , MonthlyRate 0.025343020175180128
Feature 18 , EnvironmentSatisfaction 0.025112548559102034
Feature 22 , JobSatisfaction 0.019161430028801014
Feature 16 , Education 0.01818078824742332
Feature 20 , JobInvolvement 0.0141839000343645
Feature 14 , BusinessTravel 0.01405216360617995
Feature 26 , RelationshipSatisfaction 0.012534609450910067
Feature 19 , Gender 0.01184540672076617
Feature 17 , EducationField 0.010664264891013398
Feature 28 , WorkLifeBalance 0.0097757712142504
Feature 15 , Department 0.008581462034037408
Feature 25 , PerformanceRating 0.0027161192364974236
```



2.6.2 Using top 10 features from Random Forest feature importance

We can see from the list that our top 10 most important features includes: - Stock Option Level - Marital Status - Job Role - Monthly Income - Years At Company - Over Time - Total Working Years - Age - Years With Current Manager - Distance From Home

[25]:
```
X_important=df_cluster[['StockOptionLevel', 'MaritalStatus', 'JobRole', 'Age',␣
 ↪'YearsAtCompany',
                           'MonthlyIncome', 'OverTime', 'TotalWorkingYears',␣
 ↪'YearsWithCurrManager', 'DistanceFromHome']]

X_important['JobRole'] = np.select([
    X_important['JobRole'] == 'Healthcare Representative',
    X_important['JobRole'] == 'Human Resources',
    X_important['JobRole'] == 'Laboratory Technician',
    X_important['JobRole'] == 'Manager',
    X_important['JobRole'] == 'Manufacturing Director',
    X_important['JobRole'] == 'Research Director',
    X_important['JobRole'] == 'Research Scientist',
    X_important['JobRole'] == 'Sales Executive',
    X_important['JobRole'] == 'Sales Representative'], [0,1,2,3,4,5,6,7,8])

X_important['MaritalStatus'] = np.select([
    X_important['MaritalStatus'] == 'Single',
    X_important['MaritalStatus'] == 'Married',
    X_important['MaritalStatus'] == 'Divorced'], [0,1,2])

X_important.head()
```

[25]:
```
   StockOptionLevel  MaritalStatus  JobRole  Age  YearsAtCompany  \
0                 0              0        7   41               6
1                 1              1        6   49              10
2                 0              0        2   37               0
3                 0              1        6   33               8
4                 1              1        2   27               2

   MonthlyIncome  OverTime  TotalWorkingYears  YearsWithCurrManager  \
0           5993         1                  8                     5
1           5130         0                 10                     7
2           2090         1                  7                     0
3           2909         1                  8                     0
4           3468         0                  6                     2

   DistanceFromHome
0                 1
1                 8
2                 2
```

```
3                  3
4                  2
```

Now that we have the dataframe with our top 10 most important features, we will need to concatenate the Attrition column and the Cluster column to the dataset.

2.6.3 Creating the DataFrame for Survival Analysis

[26]:
```
df_survival=pd.concat([X_important, hr_df['Attrition'], pd.DataFrame({'Cluster':
 ↪labels})],axis=1)
df_survival.head()
```

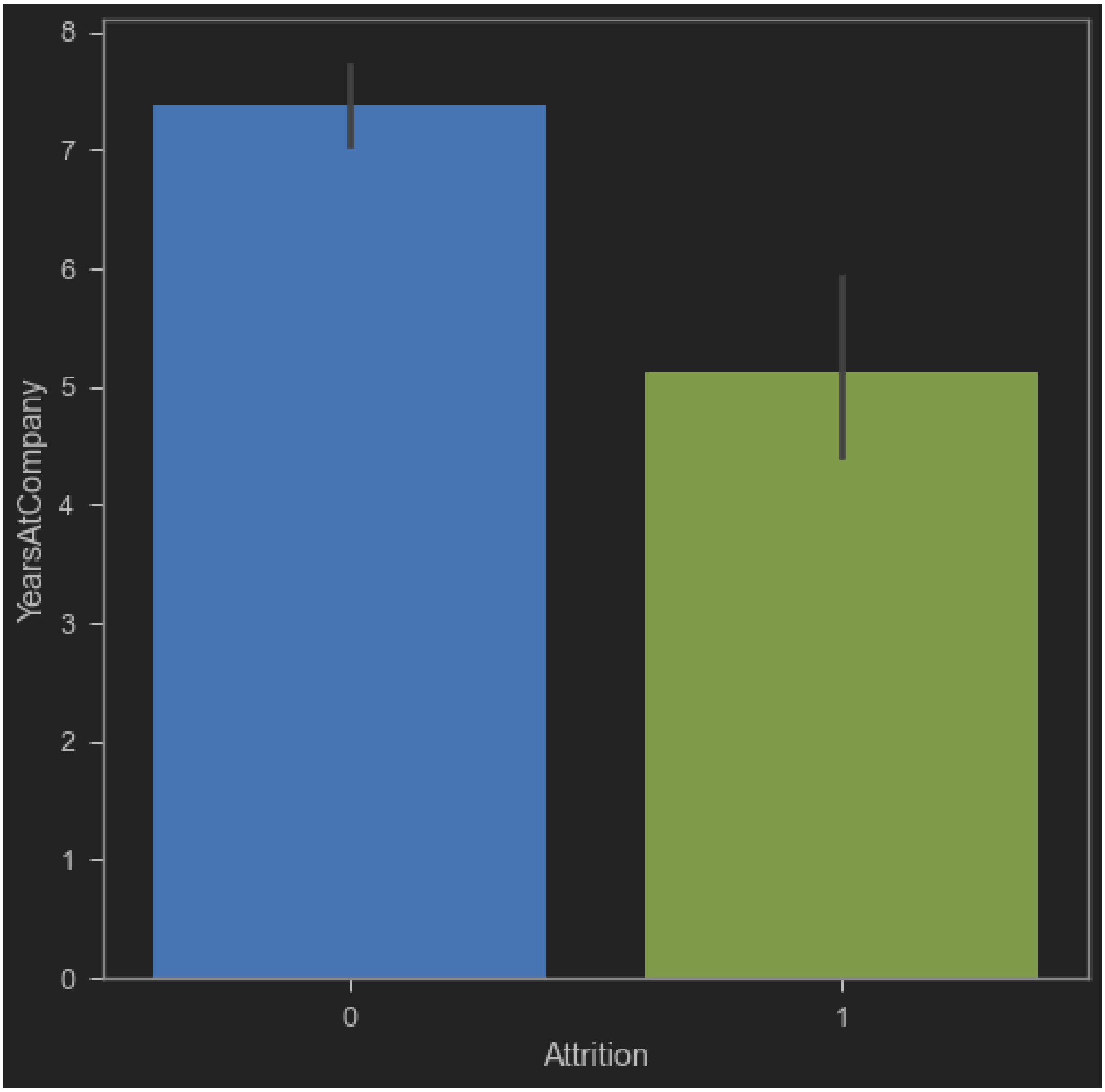
[26]:
```
   StockOptionLevel  MaritalStatus  JobRole  Age  YearsAtCompany  \
0                 0              0        7   41               6
1                 1              1        6   49              10
2                 0              0        2   37               0
3                 0              1        6   33               8
4                 1              1        2   27               2

   MonthlyIncome  OverTime  TotalWorkingYears  YearsWithCurrManager  \
0           5993         1                  8                     5
1           5130         0                 10                     7
2           2090         1                  7                     0
3           2909         1                  8                     0
4           3468         0                  6                     2

   DistanceFromHome  Attrition  Cluster
0                 1          1        7
1                 8          0        2
2                 2          1        7
3                 3          0        0
4                 2          0        7
```

We can now see our complete dataframe that we will be using for Survival Analysis which now includes the Attrition and Cluster columns.

Now we will provide a bar plot to show how our duration column relates to our event column. Keep in mind that our duration column will be Years At Company and our event column will be Attrition.

[27]:
```
plt.figure(figsize=(8,8))
sns.barplot(data=df_survival, x=df_survival['Attrition'],␣
 ↪y=df_survival['YearsAtCompany'])
```

[27]:
```
<AxesSubplot:xlabel='Attrition', ylabel='YearsAtCompany'>
```



In the Attrition columns, the value 0 means no and the value 1 means yes. In other words, employees that have 0 have not experienced attrition and employees that have 1 have experienced attrition.

We can see here that employees that have left the company through attrition have worked no longer than approximately 5 years. Employees who have worked longer than approximately 5 years tend to be still with the company.

[28]:
```
df_survival.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
```

```
 0   StockOptionLevel      1470 non-null   int64
 1   MaritalStatus         1470 non-null   int64
 2   JobRole               1470 non-null   int64
 3   Age                   1470 non-null   int64
 4   YearsAtCompany        1470 non-null   int64
 5   MonthlyIncome         1470 non-null   int64
 6   OverTime              1470 non-null   int64
 7   TotalWorkingYears     1470 non-null   int64
 8   YearsWithCurrManager  1470 non-null   int64
 9   DistanceFromHome      1470 non-null   int64
 10  Attrition             1470 non-null   int64
 11  Cluster               1470 non-null   int32
dtypes: int32(1), int64(11)
memory usage: 132.2 KB
```

2.6.4 Using the Kaplan-Meier Model

[29]:
```
from lifelines import KaplanMeierFitter
```

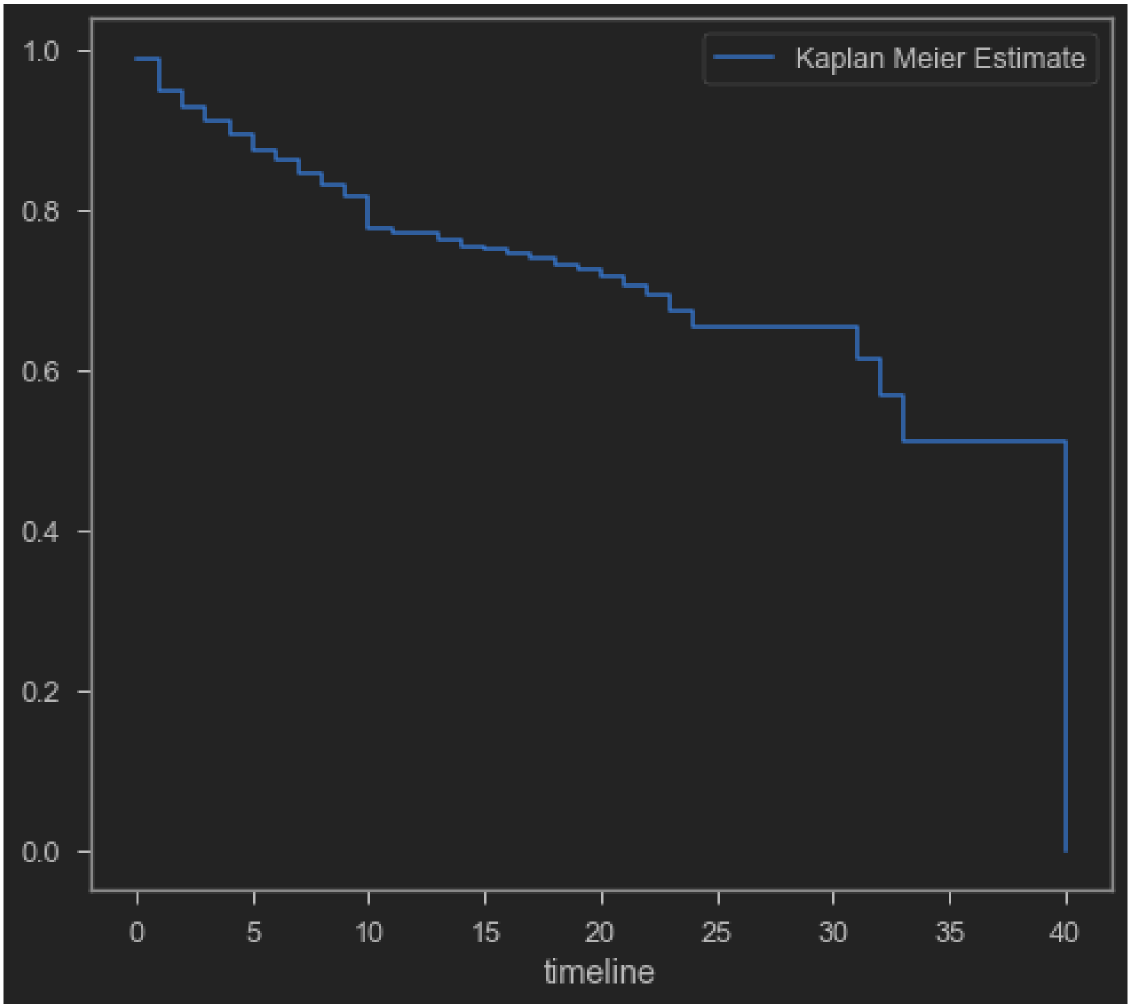
[30]:
```
kmf=KaplanMeierFitter()
kmf.fit(df_survival['YearsAtCompany'], df_survival['Attrition'], label='Kaplan␣
 ↪Meier Estimate')
```

[30]:
```
<lifelines.KaplanMeierFitter:"Kaplan Meier Estimate", fitted with 1470 total
observations, 1233 right-censored observations>
```

The Kaplan-Meier estimator is a non-parametric estimator that allows us to use observed data to estimate the survival distribution. The curve plots the cumulative probability of survival beyond each given time period.

[31]:
```
kmf.plot(ci_show=False)
```

[31]:
```
<AxesSubplot:xlabel='timeline'>
```



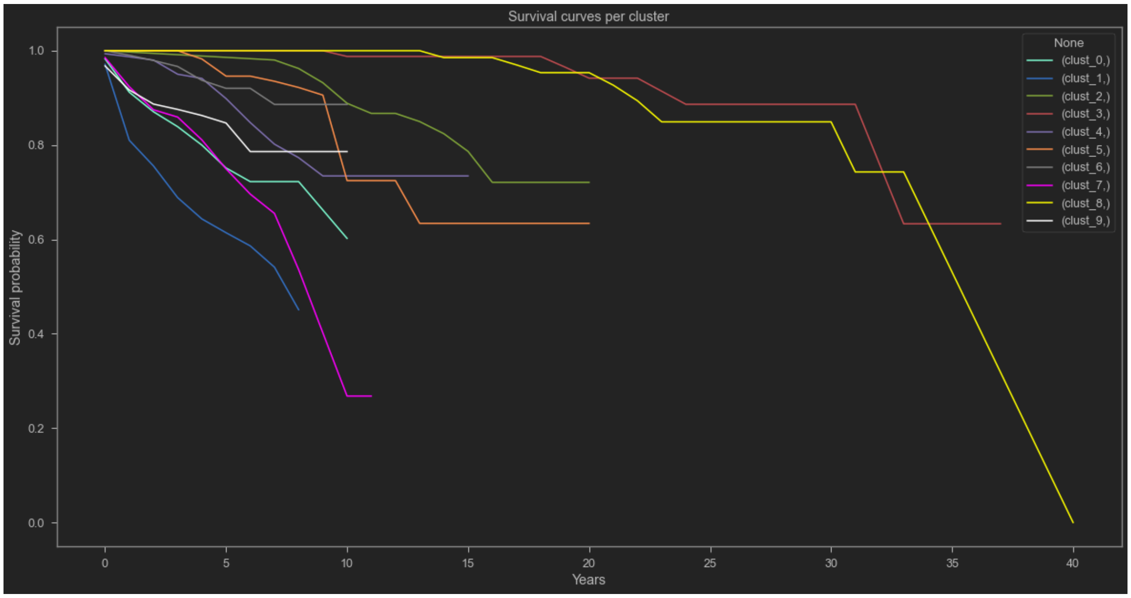
2.6.5 Plotting the Kaplan- Meier Curve per Cluster

[32]:
```
clust=pd.DataFrame({'Cluster':labels})
clust_0=df_survival[df_survival['Cluster']==0]
clust_1=df_survival[df_survival['Cluster']==1]
clust_2=df_survival[df_survival['Cluster']==2]
clust_3=df_survival[df_survival['Cluster']==3]
clust_4=df_survival[df_survival['Cluster']==4]
clust_5=df_survival[df_survival['Cluster']==5]
clust_6=df_survival[df_survival['Cluster']==6]
clust_7=df_survival[df_survival['Cluster']==7]
clust_8=df_survival[df_survival['Cluster']==8]
clust_9=df_survival[df_survival['Cluster']==9]
```

The survival function measures the probability that a cluster will survive past year "t". Using the Kaplan-Meier curve allows us to visually inspect differences in survival rates by cluster category.

[33]:
```
jtplot.style(theme='monokai',context='notebook',ticks=True, grid=False)
ax = plt.axes()
kmf.fit(clust_0['YearsAtCompany'], clust_0['Attrition'], label=['clust_0'])
kmf.survival_function_.plot(figsize=(20,10),ax=ax, color='aquamarine')
kmf.fit(clust_1['YearsAtCompany'], clust_1['Attrition'], label=['clust_1'])
kmf.survival_function_.plot(figsize=(20,10),ax=ax)
kmf.fit(clust_2['YearsAtCompany'], clust_2['Attrition'], label=['clust_2'])
kmf.survival_function_.plot(figsize=(20,10),ax=ax)
kmf.fit(clust_3['YearsAtCompany'], clust_3['Attrition'], label=['clust_3'])
kmf.survival_function_.plot(figsize=(20,10),ax=ax)
kmf.fit(clust_4['YearsAtCompany'], clust_4['Attrition'], label=['clust_4'])
kmf.survival_function_.plot(figsize=(20,10),ax=ax)
kmf.fit(clust_5['YearsAtCompany'], clust_5['Attrition'], label=['clust_5'])
kmf.survival_function_.plot(figsize=(20,10),ax=ax)
kmf.fit(clust_6['YearsAtCompany'], clust_6['Attrition'], label=['clust_6'])
kmf.survival_function_.plot(figsize=(20,10),ax=ax, color='grey')
kmf.fit(clust_7['YearsAtCompany'], clust_7['Attrition'], label=['clust_7'])
kmf.survival_function_.plot(figsize=(20,10),ax=ax,color='magenta')
kmf.fit(clust_8['YearsAtCompany'], clust_8['Attrition'], label=['clust_8'])
kmf.survival_function_.plot(figsize=(20,10),ax=ax, color='yellow')
kmf.fit(clust_9['YearsAtCompany'], clust_9['Attrition'], label=['clust_9'])
kmf.survival_function_.plot(figsize=(20,10),ax=ax, color='white')

plt.title('Survival curves per cluster')
plt.xlabel('Years')
plt.ylabel('Survival probability')
```

[33]:
```
Text(0, 0.5, 'Survival probability')
```



2.6.6 Using Cox Proportional Hazard Model

We can examine the confidence interval of different important features to assess its significance.

[34]:
```
from lifelines import CoxPHFitter
```

[35]:
```
cph = CoxPHFitter()
```

[36]:
```
cph.fit(df_survival, duration_col='YearsAtCompany', event_col='Attrition')
```

[36]:
```
<lifelines.CoxPHFitter: fitted with 1470 total observations, 1233 right-censored
observations>
```

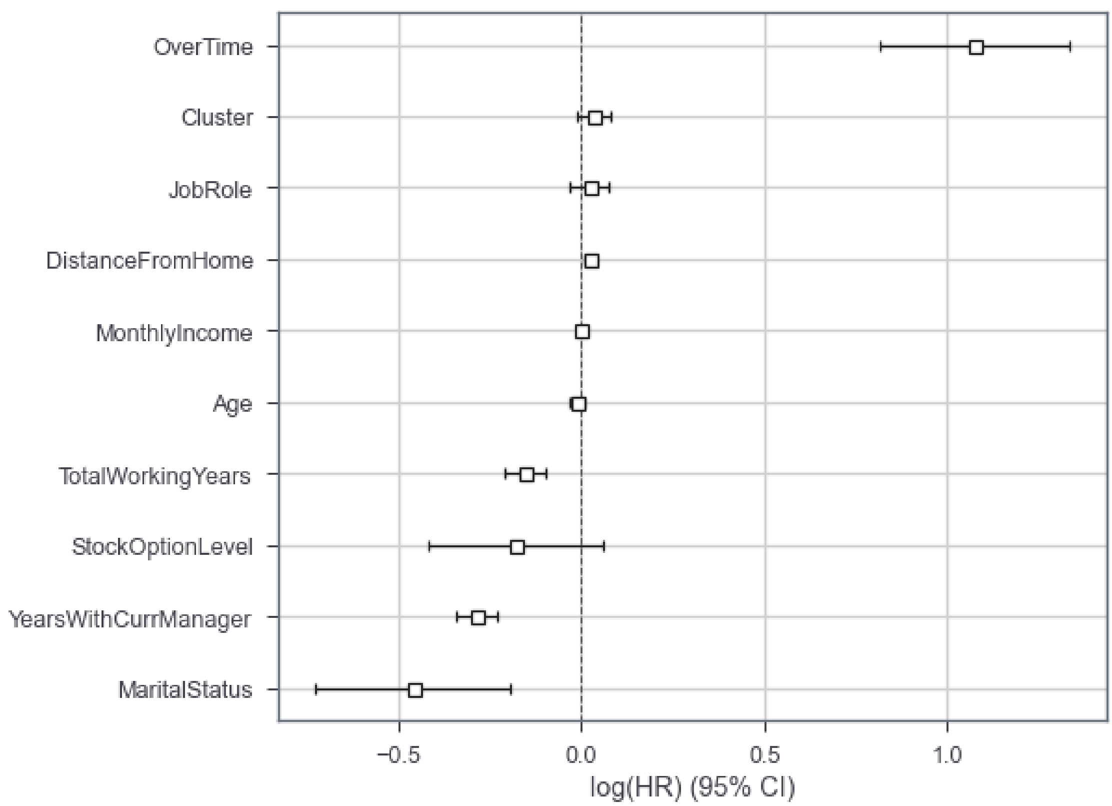
[37]:
```
cph.print_summary()
```

covariate	coef	exp(coef)	se(coef)	coef lower 95%	coef upper 95%	exp(coef) lower 95%
StockOptionLevel	-0.18	0.83	0.12	-0.42	0.06	0.66
MaritalStatus	-0.46	0.63	0.14	-0.73	-0.20	0.48
JobRole	0.02	1.02	0.03	-0.03	0.08	0.97
Age	-0.01	0.99	0.01	-0.03	0.01	0.97
MonthlyIncome	-0.00	1.00	0.00	-0.00	-0.00	1.00
OverTime	1.08	2.93	0.13	0.82	1.33	2.26
TotalWorkingYears	-0.16	0.86	0.03	-0.21	-0.10	0.81
YearsWithCurrManager	-0.29	0.75	0.03	-0.34	-0.23	0.71
DistanceFromHome	0.02	1.02	0.01	0.01	0.04	1.01
Cluster	0.03	1.04	0.02	-0.01	0.08	0.99

2.6.7 These features are the main drivers for employees to stay at the company.

[38]:
```
from jupyterthemes import jtplot
jtplot.style(theme='grade3',context='notebook',ticks=True, grid=True)
cph.plot()
```

[38]: <AxesSubplot:xlabel='log(HR) (95% CI)'>



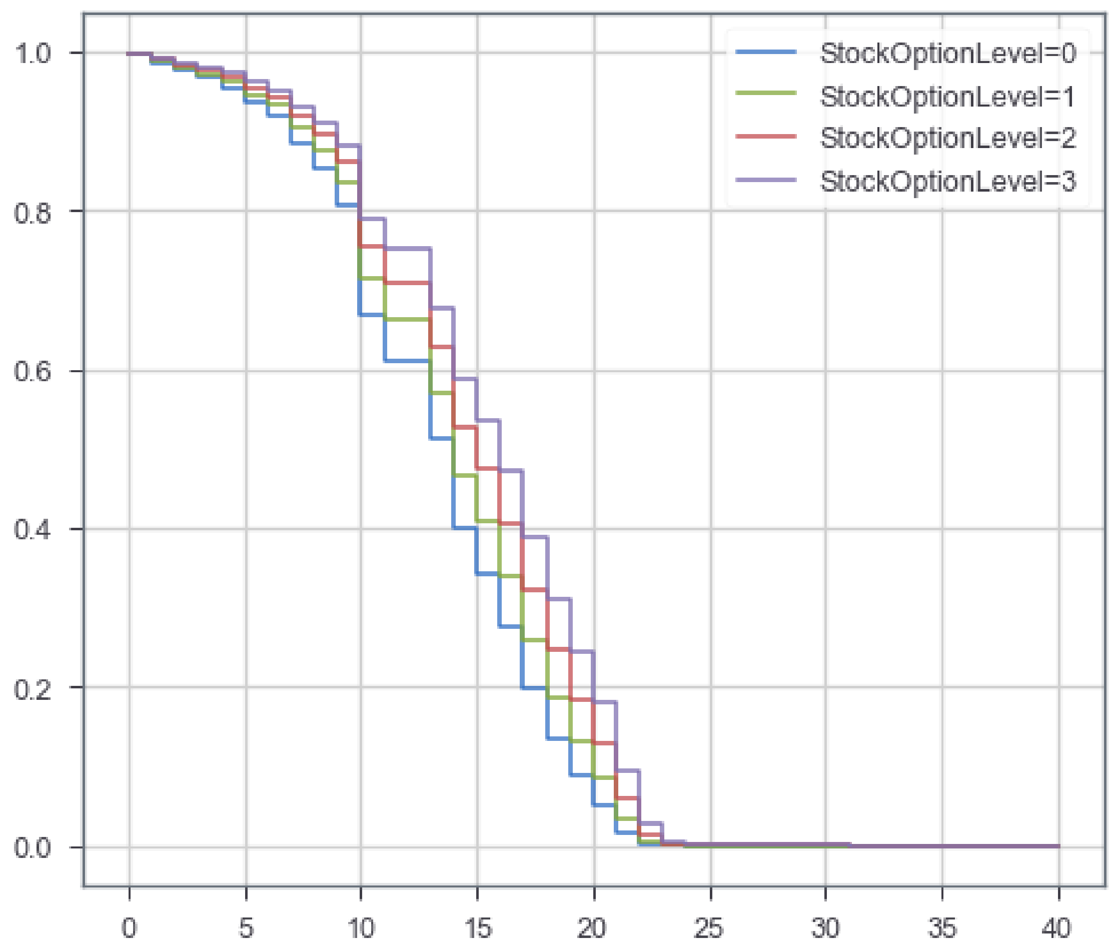
2.6.8 Based on the above plot, we can see that the main driving features includes:

- AGE
- MARITAL STATUS
- MONTHLY INCOME
- TOTAL WORKING YEARS
- STOCK OPTION LEVEL
- YEARS WITH CURRENT MANAGER

2.6.9 Impact of Stock Option Levels

[39]:
```
cph.plot_partial_effects_on_outcome('StockOptionLevel', [0,1,2,3],␣
↪plot_baseline=False)
```

[39]: <AxesSubplot:>



Stock Option Levels:

Level 1. Covered Call, Long Protective Puts

Level 2. Long call/put

Level 3. Spreads

Level 4. Uncovered or Naked

2.7 Deployment

Our models can be useful for any company that has an active Human Resources division. They can implement this to determine the likelihood of employees that will leave due to attrition. This will also allow these companies to perform their own survival analysis using their own data.

Companies can also use these models to determine factors of attrition. It may also help them to elaborate strategies to retain their top employees.

2.8 Performance Evaluation

What we have done well:

- Our representation of our analysis through the use of plots was effective.
- Using cluster segmentations together with the survival analysis was thorough.
- The App is very visually appealing and effective in determining the qualities of employees who will leave and who will stay.
- The inclusion of the survival analysis plot in the App was well placed since it gives us an idea of the employee clusters and the probability of each cluster's survival.

What we could have done better:

- Knowledge of deep learning would help us to improve our predictions.

2.9 Bibliography

- https://www.investopedia.com/terms/h/humanresources.asp
- https://jobzology.com/staff-attrition-vs-staff-turnover-whats-the-difference/
- https://stackoverflow.com/questions/40795141/pca-for-categorical-features
- https://nextjournal.com/pc-methods/calculate-pc-mixed-data?change-id=CWQNw1kVRgQMFFzobMC2bo&node-id=d4243af6-f940-41fc-8ffa-a235bc135601
- https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c
- https://towardsdatascience.com/one-hot-encoding-is-making-your-tree-based-ensembles-worse-heres-why-d64b282b5769

2.10 Storing the Object Data to Files for App Development

[40]:

```python
import os
from pathlib import Path
import pickle

pickle.dump(ss, open(Path().resolve().parent.joinpath('app').joinpath('ss.
↪sav'), 'wb'))
pickle.dump(grid_search, open(Path().resolve().parent.joinpath('app').
↪joinpath('grid_search.sav'), 'wb'))
pickle.dump(df_survival, open(Path().resolve().parent.joinpath('app').
↪joinpath('survival_df.sav'), 'wb'))
pickle.dump(kmf, open(Path().resolve().parent.joinpath('app').joinpath('kmf.
↪sav'), 'wb'))
```