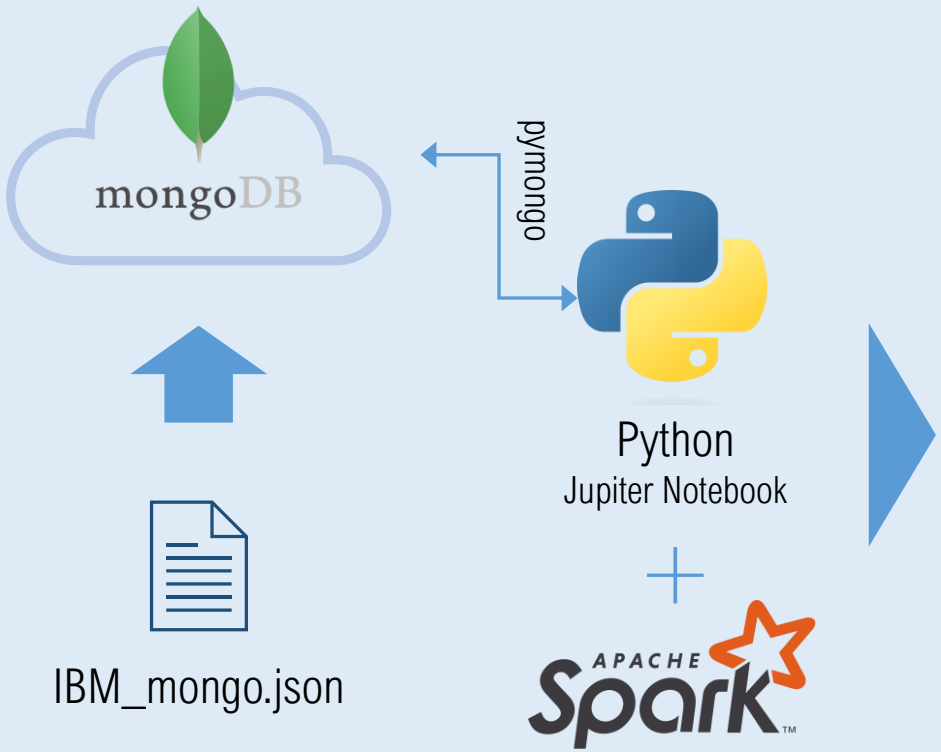
IBM Human Resource Management Proposal

Data Mining with Mysql, Nosql, Hadoop, Spark, and Hive

Team 7: Omar Abdelkhalek, Alvaro Chinguel, Yunsik Choung, Piyush Kumar

Contents

Database Connection & Manipulation with mongoDB




```python
import pandas as pd # Pandas DataFrame
import seaborn as sns # Data Visualization Tool
import numpy as np # Numpy
import matplotlib.pyplot as plt # Ploting Tool
import statsmodels.formula.api as smf # Statistic Analysis Tool
from pymongo.mongo_client import MongoClient # Mongo DB Client Connection Tool
import urllib.parse # Url String Parsing

# setting pandas print options
pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)

# Connecting String
username = urllib.parse.quote_plus('[redacted]')
password = urllib.parse.quote_plus('[redacted]')
uri = "mongodb+srv://" + username + ":" + password + "@hultyunsikchoung.wfvhiwp.mongodb.net/?retryWr

# Set client
client = MongoClient(uri)

# Send a ping to confirm a successful connection
try:
    client.admin.command('ping')
    print("Pinged your deployment. You successfully connected to MongoDB!")
except Exception as e:
    print(e)
```

```
Pinged your deployment. You successfully connected to MongoDB!
```

Data Manipulation
pymongo, pandas, Numpy

Data Visualization
matplotlib, seaborn

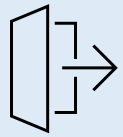
Data Analysis
Pandas, Statsmodels, PySpark

- Data Manipulation
- Descriptive Statistics
- Hypothesis & Research Design
- Data Analysis & Visualization
- Predictive Modeling
- Conclusion

HULT INTERNATIONAL BUSINESS SCHOOL

Descriptive Statistics Between Attrition: Numerical Variables

lmb_case
Total: 1470 obs
35 cols
(12 Categorical, 33 Numerical)



Attrition

Yes
237 obs

No
1233 obs

Attrition	Yes	No
Age	33.61	37.56
Distance From Home	10.63	8.92
Environment Satisfaction	2.46	2.77
Job Satisfaction	2.47	2.78
Monthly Income	4,787.09	6,832.74
Monthly Rate	14,559.31	14,265.78
# of Companies Worked	2.94	2.65
Percent Salary Hike	15.1	15.23
Performance Rating	3.16	3.15
Relationship Satisfaction	2.6	2.73
Stock Option Level	0.53	0.85
Total Working Years	8.24	11.86
Training Times Last Year	2.62	2.83
Work Life Balance	2.66	2.78
Years At Company	5.13	7.37
Years In Current Role	2.9	4.48
Years Since Last Promotion	1.95	2.23
Years With Current Manager	2.85	4.37

▼ Lower Variables
Age
Distance from Home
Environment Satisfaction
Job Satisfaction
Monthly Income
Monthly Rate
Relationship Satisfaction
Stock Option Level
Total Working Years
Training Times Last Year
Work Life Balance
Years At Company
Years In Current Role
Years Since Last Promotion
Years With Current Manager

▲ Higher Variables
# of Companies Worked

? Look Similar Variables
Percent Salary Hike

HULT INTERNATIONAL BUSINESS SCHOOL

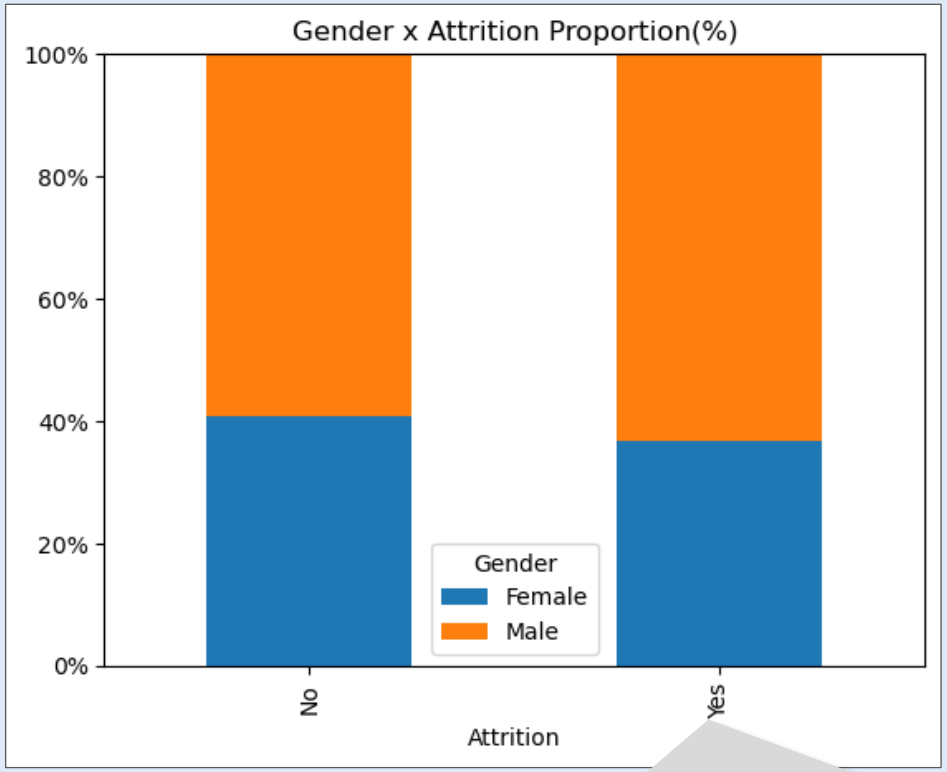
Descriptive Statistics Between Attrition: Categorical Variables

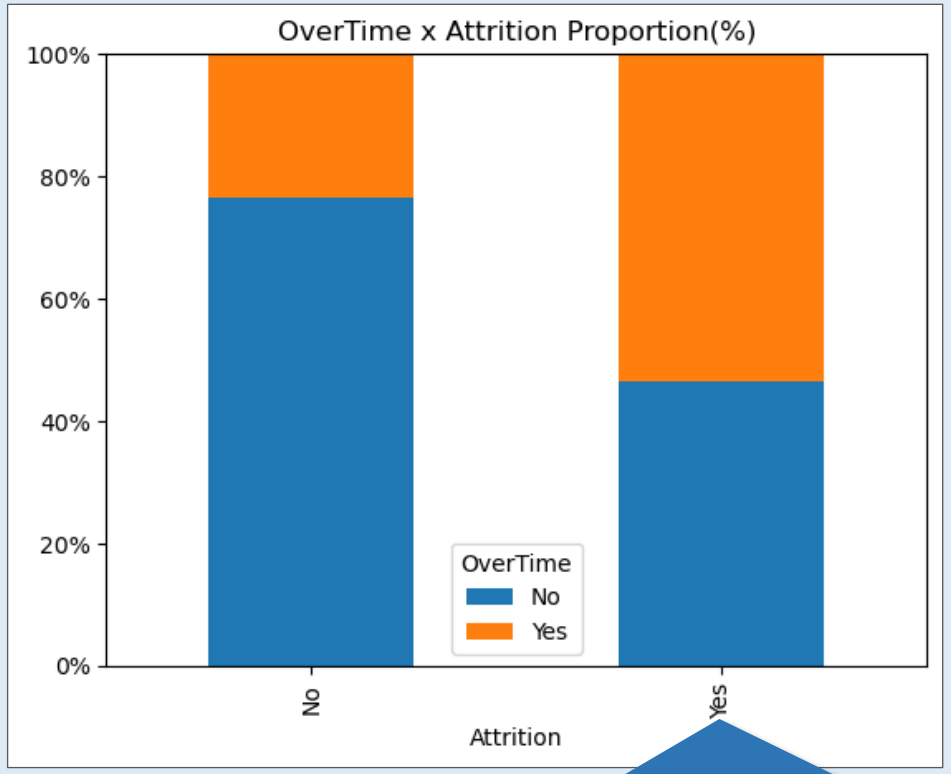
Yes

237 obs

No

1233 obs





▼ Less Differences within Attrition

Gender, Business Travel, Education, Job Involvement, Education Field, Over18

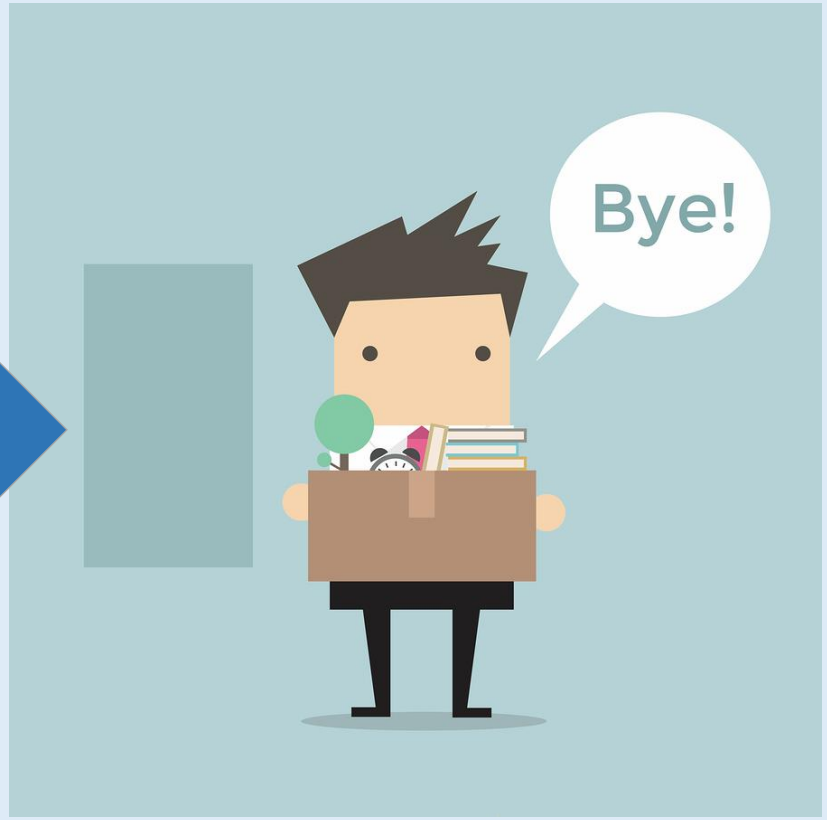
▲ More Differences within Attrition

Department, Job Role, Marital Status, Over Time

Hypothesis & Research Design

Hypothesis

H_0. Variables that have more differences between Attrition status cannot predict employee attrition.
H_1. Variables that have more differences between Attrition status can predict employee attrition.



Research Design

Step. 1
Data Massaging

Step. 2
Correlation Analysis and Basic Logit Model

Step. 3
Building Predictive Model

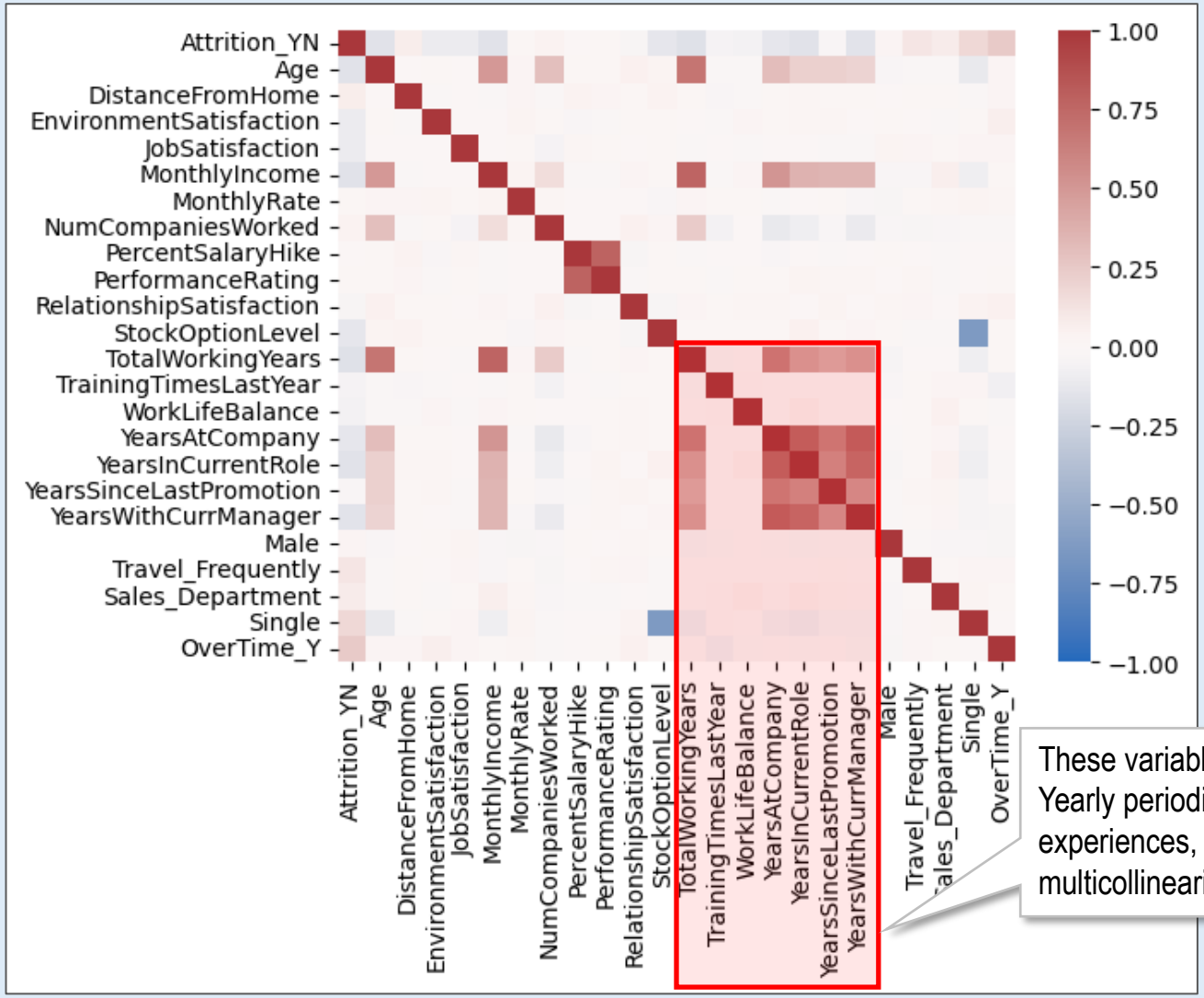
Step. 4
Hypothesis Test

Data Analysis & Visualization: Correlation Analysis

Three Variables has correlation coefficient more than 0.1.

Nine Variables has negative correlation coefficient less than -0.1.

Variables	Coefficient
Over Time Yes	**0.246**
Single (Marital)	**0.175**
Travel Frequently	**0.115**
Sales Department	0.081
Distance From Home	0.078
Num Companies Worked	0.043
Male	0.029
Years Since Last Promotion	-0.033
Relationship Satisfaction	-0.046
Training Times Last Year	-0.059
Work Life Balance	-0.064
Environment Satisfaction	**-0.103**
Job Satisfaction	**-0.103**
Years At Company	**-0.134**
Stock Option Level	**-0.137**
Years With Current Manager	**-0.156**
Age	**-0.159**
Monthly Income	**-0.160**
Years In Current Role	**-0.161**
Total Working Years	**-0.171**



These variables, related in Yearly periodic working experiences, seem to have multicollinearity.

HULT INTERNATIONAL BUSINESS SCHOOL

Data Analysis & Visualization: Logistic Regression Model

Full Model: 23 Variables

Attrition ~ Age + Distance From Home + Environment Satisfaction + Job Satisfaction + Monthly Income + Monthly Rate + Num Companies Worked + Percent Salary Hike + Performance Rating
+ Relationship Satisfaction + Stock Option Level + Total Working Years + Training Times Last Year + Work Life Balance + Years At Company + Years In Current Role
+ Years Since Last Promotion + Years With Current Manager + Male(Dummy) + Travel Frequently(Dummy) + Sales Department(Dummy) + Single(Dummy) + Over Time Yes(Dummy)

- Model Fitted and R-Squared is 0.281.
- Check Significancy of each variables. ($p < 0.1$)

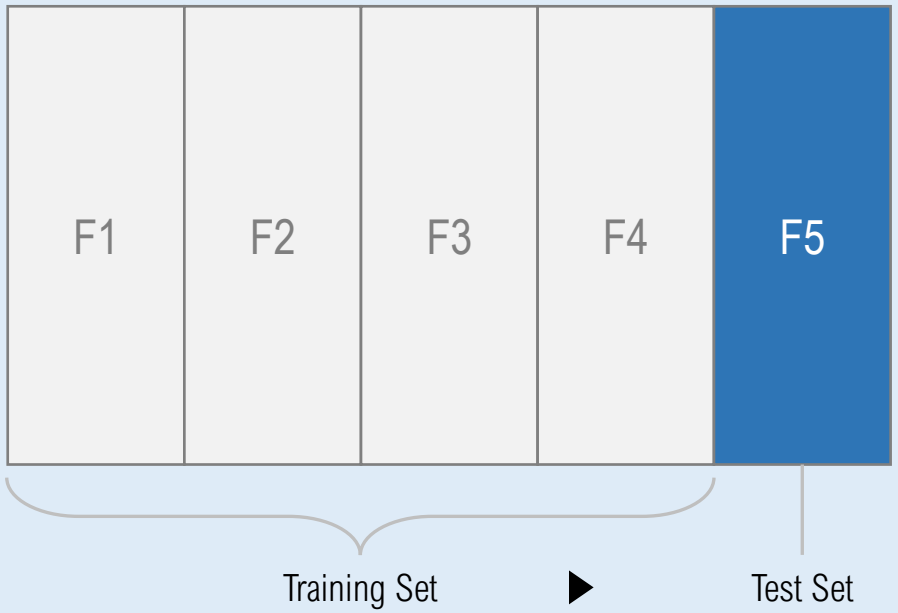
Model:	Logit	Pseudo R-squared:	0.281
Dependent Variable:	Attrition_YN	AIC:	981.434
#. Observations:	1470	BIC:	1,108.467
Df Model:	23	Log-Likelihood:	-466.720
Df Residuals:	1446	LL-Null:	-649.290
Converged:	1	LLR p-value:	0.000
No. Iterations:	8	Scale:	1.000

Final Model: 17 Variables

Attrition ~ Age + Distance From Home + Environment Satisfaction + Job Satisfaction + Monthly Income+ Num Companies Worked
+ Relationship Satisfaction + Stock Option Level + Total Working Years + Training Times Last Year + Work Life Balance
+ Years In Current Role + Male(Dummy) + Travel Frequently(Dummy) + Sales Department(Dummy) + Single(Dummy) + Over Time Yes(Dummy)

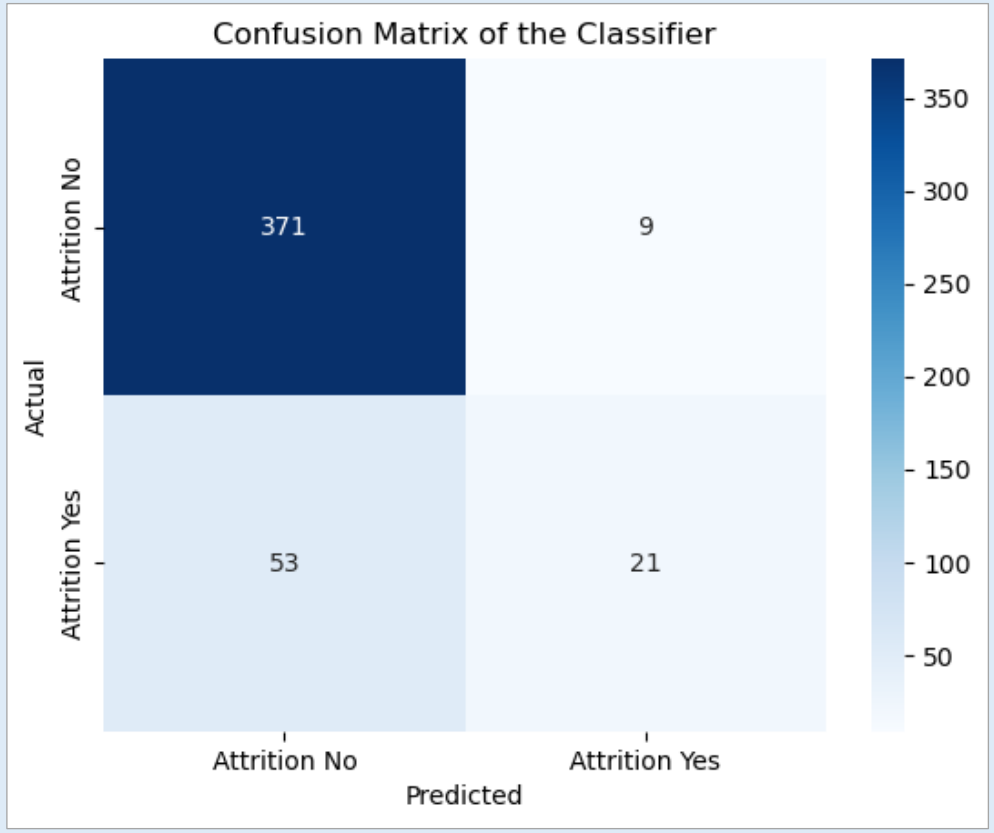
Predictive Modeling with PySpark ML

Cross Validating Logistic Classification ML Model



Model Accuracy

AUC(Area Under ROC): 0.781 – Able to predict 78.1% of employees



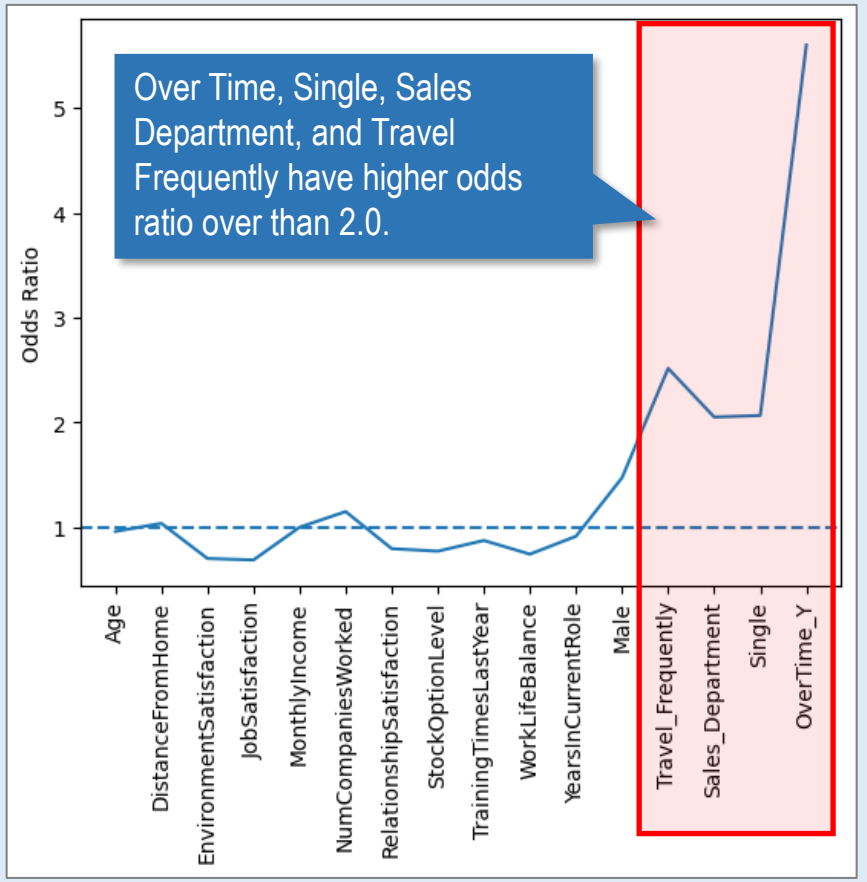
Logistic Regression Model Result

Final Model: 17 Variables

Attrition ~ Age + Distance From Home + Environment Satisfaction + Job Satisfaction + Monthly Income+ Num Companies Worked + Relationship Satisfaction + Stock Option Level + Total Working Years + Training Times Last Year + Work Life Balance + Years In Current Role + Male(Dummy) + Travel Frequently(Dummy) + Sales Department(Dummy) + Single(Dummy) + Over Time Yes(Dummy)

Model:	Logit	Pseudo R-squared:	0.250
Dependent Variable:	Attrition_YN	AIC:	1,007.770
#. Observations:	1470	BIC:	1,097.751
Df Model:	16	Log-Likelihood:	-486.890
Df Residuals:	1453	LL-Null:	-649.290
Converged:	1	LLR p-value:	0.000
No. Iterations:	7	Scale:	1.000

Variables	*Coef.*	*Std.Err.*	*P>|z|*
Age	-0.041	0.011	0.000
DistanceFromHome	0.036	0.010	0.000
EnvironmentSatisfaction	-0.355	0.075	0.000
JobSatisfaction	-0.374	0.074	0.000
MonthlyIncome	0.000	0.000	0.000
NumCompaniesWorked	0.139	0.034	0.000
RelationshipSatisfaction	-0.229	0.076	0.003
StockOptionLevel	-0.259	0.138	0.061
TrainingTimesLastYear	-0.135	0.067	0.045
WorkLifeBalance	-0.297	0.114	0.009
YearsInCurrentRole	-0.090	0.029	0.002
Male	0.387	0.172	0.024
Travel_Frequently	0.923	0.192	0.000
Sales_Department	0.719	0.176	0.000
Single	0.726	0.226	0.001
OverTime_Y	1.723	0.174	0.000

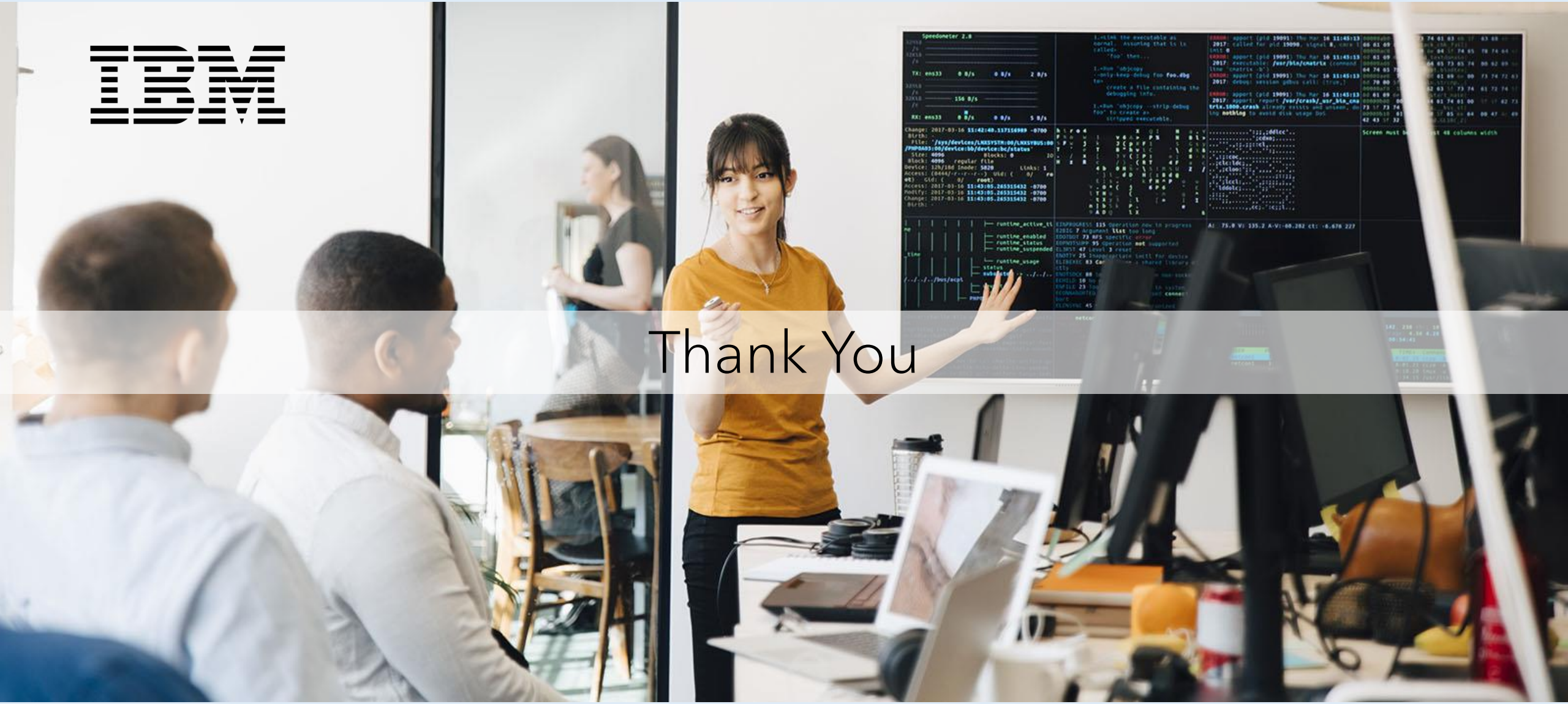


- Data Manipulation
- Descriptive Statistics
- Hypothesis & Research Design
- Data Analysis & Visualization
- **Predictive Modeling**
- Conclusion

HULT INTERNATIONAL BUSINESS SCHOOL

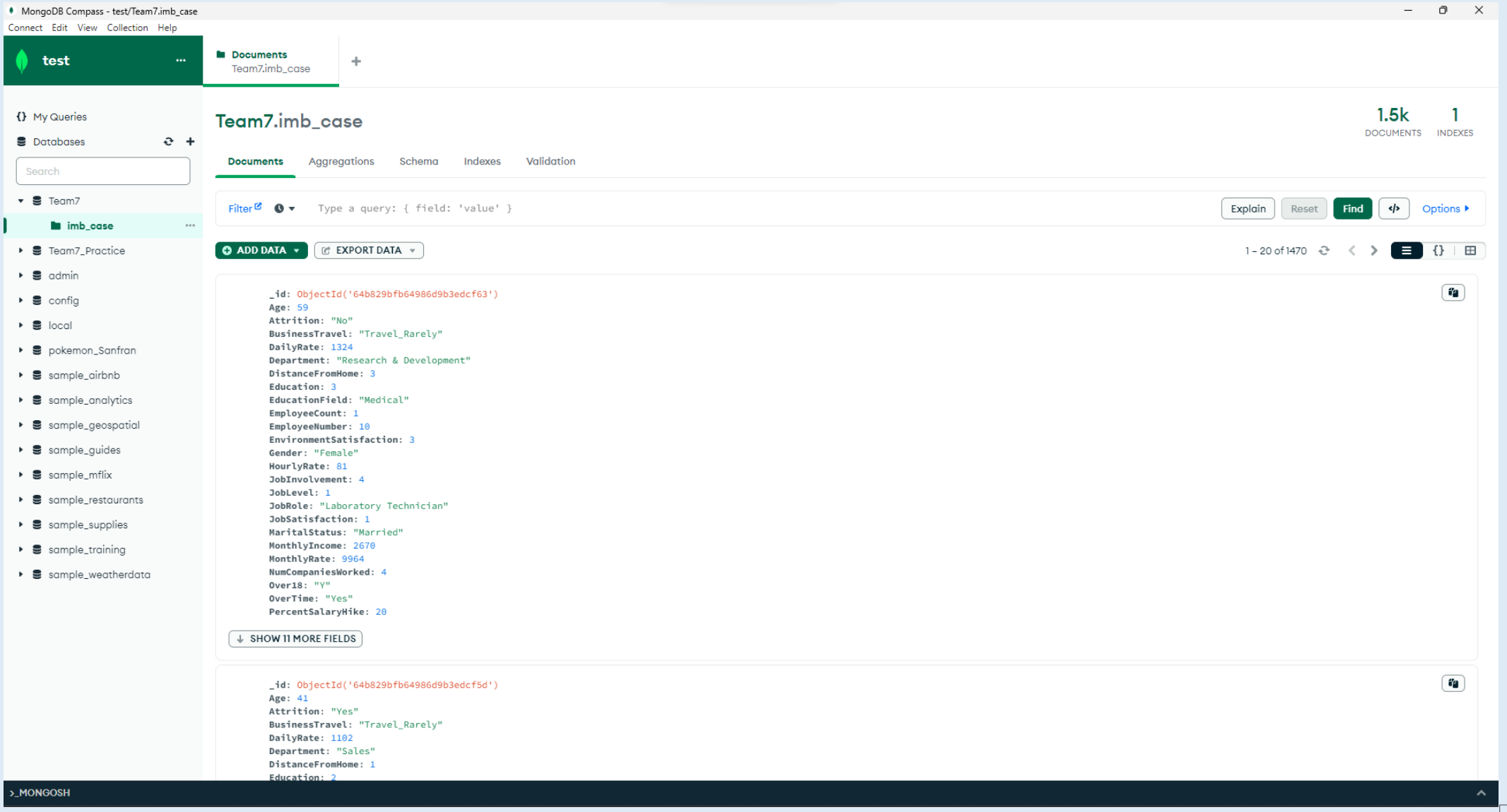
Conclusion: How to prevent employee Attrition at IBM?

- Monitoring Over Time Work
- Check Working Environments of Sales Department
- Single + Younger People Care
- Redefine Travel Policy
- Satisfaction Control – Job, Relationship, Environment Periodical Survey



HULT
INTERNATIONAL
BUSINESS SCHOOL

Mongo DB Dataset



Dataset Connection with Mongo DB in local Jupyter Notebook

In [1]:

```python
import pandas as pd # Pandas DataFrame
import seaborn as sns # Data Visualization Tool
import numpy as np # Numpy
import matplotlib.pyplot as plt # Ploting Tool
import statsmodels.formula.api as smf # Statistic Analysis Tool
from pymongo.mongo_client import MongoClient # Mongo DB Client Connection Tool
import urllib.parse # Url String Parsing

# setting pandas print options
pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)

# Connecting String for MongoDB Connector
username = urllib.parse.quote_plus('[redacted]')
password = urllib.parse.quote_plus('[redacted]')
uri = "mongodb+srv://" + username + ":" + password + "@hultyunsikchoung.wfvhiwp.mongodb.net/?retryWrites=true&w=majority"

# Set client
client = MongoClient(uri)

# Send a ping to confirm a successful connection
try:
    client.admin.command('ping')
    print("Pinged your deployment. You successfully connected to MongoDB!")
except Exception as e:
    print(e)
```

```
Pinged your deployment. You successfully connected to MongoDB!
```

In [2]:

```python
# Check Database list
print(client.list_database_names())
```

```
['Team7', 'Team7_Practice', 'pokemon_Sanfran', 'sample_airbnb', 'sample_analytics', 'sample_geospatial', 'sample_guides', 'sample_mflix', 'sample_restaurants', 'sample_supplies', 'sample_training', 'sample_weatherdata', 'admin', 'local']
```

HULT INTERNATIONAL BUSINESS SCHOOL

Descriptive Statistics using Mongo DB at Local Jupyter

In [6]:
```
# Descriptive Statistics (Numerical Variables) using aggregate Function with MongoDB QL
descriptive_statistics = mycol.aggregate([{
    "$group": {
        "_id": "$Attrition", # Attrition Status Yes / No
        "CNT": { # Count each Attrition Status
            "$count": {}
        }, "Avg_Age": { # Average Age between Yes and No
            "$avg": "$Age"
        }, "Avg_DistanceFromHome": { # Average Distance From Home
            "$avg": "$DistanceFromHome"
        }, "Avg_EnvironmentSatisfaction": { # Average Environment Satifaction Likert Scale
            "$avg": "$EnvironmentSatisfaction"
        }, "Avg_JobSatisfaction": { # Average Job Satisfaction Likert Scale
            "$avg": "$JobSatisfaction"
        }, "avg_MonthlyIncome":{ # Average Monthly Income in $
            "$avg": "$MonthlyIncome"
        }, "Avg_Monthly_rate": { # Average Monthly Rate
            "$avg": "$MonthlyRate"
        }, "Avg_NumCompaniesWorked": { # Average# of Companies worked
            "$avg": "$NumCompaniesWorked"
        }, "Avg_PercentSalaryHike": { # Average Percent Salary hike
            "$avg": "$PercentSalaryHike"
        }, "Avg_RelationshipSatisfaction": { # Average Relationship Satisfaction
            "$avg": "$RelationshipSatisfaction"
        }, "Avg_StockOptionLevel": { # Average Stock Option Level. Looks binary 1: Yes, 0: No Stock Option
            "$avg": "$StockOptionLevel"
        }, "Avg_TotalWorkingYears": { # Average Total working years
            "$avg": "$TotalWorkingYears"
        }, "Avg_TrainingTimesLastYear": { # Average Training Times Last year
            "$avg": "$TrainingTimesLastYear"
        }, "Avg_WorkLifeBalance": { # Average Work Life Balance. TIME SPENT BEWTWEEN WORK AND OUTSIDE
            "$avg": "$WorkLifeBalance"
        }, "Avg_YearsAtCompany": { # Average Years at company
            "$avg": "$YearsAtCompany"
        }, "Avg_YearsInCurrentRole": { # Average Years in current role
            "$avg": "$YearsInCurrentRole"
        }, "Avg_YearsSinceLastPromotion": { # Average Years Since last promotion
            "$avg": "$YearsSinceLastPromotion"
        }, "Avg_YearsWithCurrManager": { # Average Years with current manager
            "$avg": "$YearsWithCurrManager"
        }
    }
}])
list(descriptive_statistics)
# DistanceFromHome RelationshipSatisfaction PercentSalaryHike EnvironmentSatisfaction
# YearsAtCompany YearsInCurrentRole YearsSinceLastPromotion YearsWithCurrManager
```

In [7]:
```
# Categorical Variables Crosstabulation
# Gender pivoting
descriptive_gender = mycol.aggregate([{
    "$group": {
        "_id": ["$Attrition", "$Gender"],
        "CNT": {
            "$count": {}
        }
    }
}])
list(descriptive_gender)
```

Out[7]:
```
[{'_id': ['No', 'Male'], 'CNT': 732},
 {'_id': ['Yes', 'Male'], 'CNT': 150},
 {'_id': ['No', 'Female'], 'CNT': 501},
 {'_id': ['Yes', 'Female'], 'CNT': 87}]
```

In [8]:
```
# Categorical Variables Crosstabulation
# BusinessTravel pivoting
descriptive_BusinessTravel = mycol.aggregate([{
    "$group": {
        "_id": ["$Attrition", "$BusinessTravel"],
        "CNT": {
            "$count": {}
        }
    }
}])
list(descriptive_BusinessTravel)
```

Out[8]:
```
[{'_id': ['No', 'Travel_Frequently'], 'CNT': 208},
 {'_id': ['Yes', 'Travel_Rarely'], 'CNT': 156},
 {'_id': ['Yes', 'Travel_Frequently'], 'CNT': 69},
 {'_id': ['No', 'Non-Travel'], 'CNT': 138},
 {'_id': ['Yes', 'Non-Travel'], 'CNT': 12},
 {'_id': ['No', 'Travel_Rarely'], 'CNT': 887}]
```

In [ ]:
```

```

In [9]:
```
# Categorical Variables Crosstabulation
# Department pivoting
descriptive_Department = mycol.aggregate([{
    "$group": {
        "_id": ["$Attrition", "$Department"],
        "CNT": {
            "$count": {}
        }
    }
}])
list(descriptive_Department)
```

Out[9]:
```
[{'_id': ['Yes', 'Sales'], 'CNT': 92},
 {'_id': ['No', 'Sales'], 'CNT': 354},
 {'_id': ['No', 'Human Resources'], 'CNT': 51},
 {'_id': ['Yes', 'Human Resources'], 'CNT': 12},
 {'_id': ['No', 'Research & Development'], 'CNT': 828},
 {'_id': ['Yes', 'Research & Development'], 'CNT': 13
3}]
```

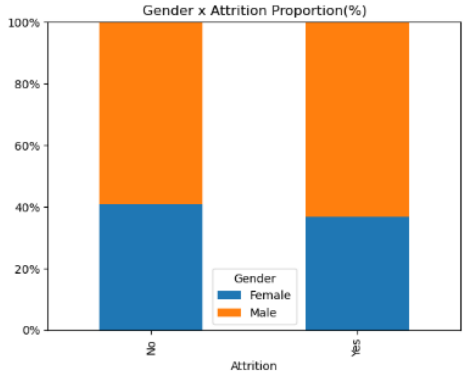
In [10]:
```
# Categorical Variables Crosstabulation
# Education pivoting
descriptive_Education = mycol.aggregate([{
    "$group": {
        "_id": ["$Attrition", "$Education"],
        "CNT": {
            "$count": {}
        }
    }
}])
list(descriptive_Education)
```

Out[10]:
```
[{'_id': ['No', 1], 'CNT': 139},
 {'_id': ['Yes', 2], 'CNT': 44},
 {'_id': ['No', 2], 'CNT': 238},
 {'_id': ['Yes', 3], 'CNT': 99},
 {'_id': ['No', 3], 'CNT': 473},
 {'_id': ['Yes', 4], 'CNT': 58},
 {'_id': ['No', 5], 'CNT': 43},
 {'_id': ['Yes', 1], 'CNT': 31},
 {'_id': ['No', 4], 'CNT': 340},
 {'_id': ['Yes', 5], 'CNT': 5}]
```

Data Analysis & Visualization using Pandas and PyPlot

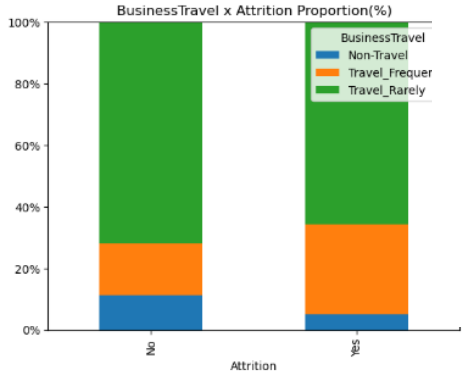
In [18]:
```
# Categorical Variables Crosstabulation in Pandas
# Visualize stacked percentage bar chart
col = 'Gender'
display(pd.crosstab(
    columns=df['Attrition'],
    index=df[col]
))
pd.crosstab(
    index=df['Attrition'],
    columns=df[col],
    normalize='index'
).plot(kind='bar', stacked=True,
       title=col + " x Attrition Proportion(%)")
plt.ylim(0,1)
plt.yticks(plt.yticks()[0], ['{:,.0%}'.format(x) f
plt.show()
```

Attrition	No	Yes
Gender		
Female	501	87
Male	732	150

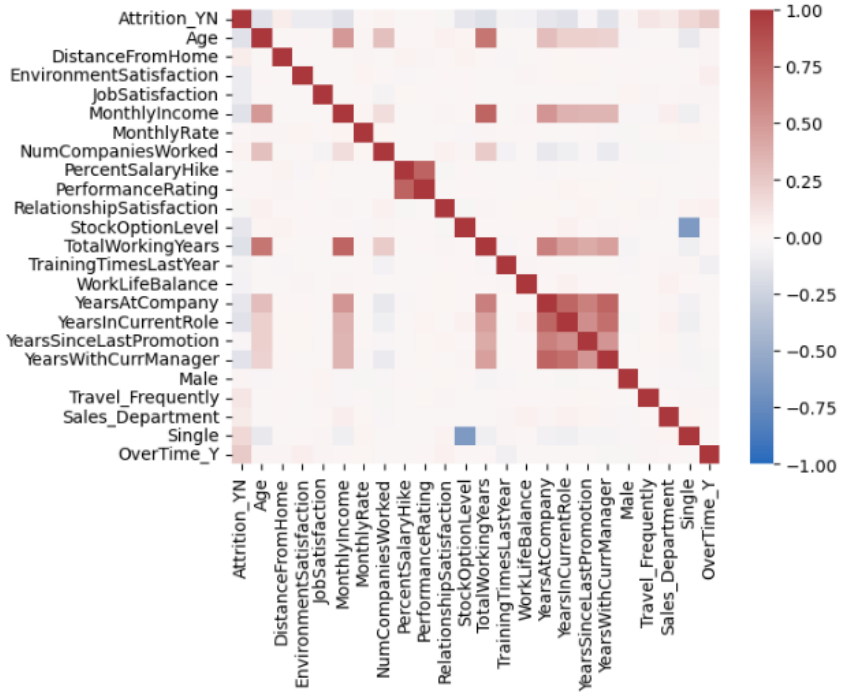


In [19]:
```
# Categorical Variables Crosstabulation in Pa
# Visualize stacked percentage bar chart
col = 'BusinessTravel'
display(pd.crosstab(
    columns=df['Attrition'],
    index=df[col]
))
pd.crosstab(
    index=df['Attrition'],
    columns=df[col],
    normalize='index'
).plot(kind='bar', stacked=True,
       title=col + " x Attrition Proportion(%)")
plt.ylim(0,1)
plt.yticks(plt.yticks()[0], ['{:,.0%}'.format
plt.show()
```

Attrition	No	Yes
BusinessTravel		
Non-Travel	138	12
Travel_Frequently	208	69
Travel_Rarely	887	156



In [42]:
```
# Check Heatmap
sns.heatmap(data=cor_test, annot=False, vmax=1, vmin=-1, cmap='vlag')
plt.show()
```



Data Massaging & Get Dummy Variables using Mongo DB

In [28]:

```
# Conditional Syntax: Building Dummy Variables from categorical variables
added_df = mycol.aggregate([{
    "$addFields": {
        "Attrition_YN": { # Dependent Variable
            "$switch": {
                "branches": [
                    {"case": {"$eq": ["$Attrition", "Yes"]}, "then": 1},
                    {"case": {"$eq": ["$Attrition", "No"]}, "then": 0}
                ], "default": 99
            }
        }, "Male": { # Gender Dummy Variable: Base = Female
            "$switch": {
                "branches":[
                    {"case": {"$eq": ["$Gender", "Male"]}, "then": 1},
                ], "default": 0
            }
        }, "Travel_Frequently": { # BusinessTravel Dummy Variable. Only focused on Travel Frequently
            "$switch": {
                "branches":[
                    {"case": {"$eq": ["$BusinessTravel", "Travel_Frequently"]}, "then": 1}
                ], "default": 0
            }
        }, "Sales_Department": { # Department Dummy Variable. Only Sales_Department
            "$switch": {
                "branches":[
                    {"case": {"$eq": ["$Department", "Sales"]}, "then": 1}
                ], "default": 0
            }
        }, "Single": { # MaritalStatus Dummy Variable. Only single
            "$switch": {
                "branches":[
                    {"case": {"$eq": ["$MaritalStatus", "Single"]}, "then": 1}
                ], "default": 0
            }
        }, "OverTime_Y": { # OverTime Binary Dummy. Over Time Yes.
            "$switch": {
                "branches":[
                    {"case": {"$eq": ["$OverTime", "Yes"]}, "then": 1}
                ], "default": 0
            }
        }
    }
}])
```

Logistic Regression with Statsmodels Library at local Jupyter

In [47]:
```
model2 = smf.logit(data=df, formula=fml2)
res = model2.fit()

# Final Model
res.summary2()
```

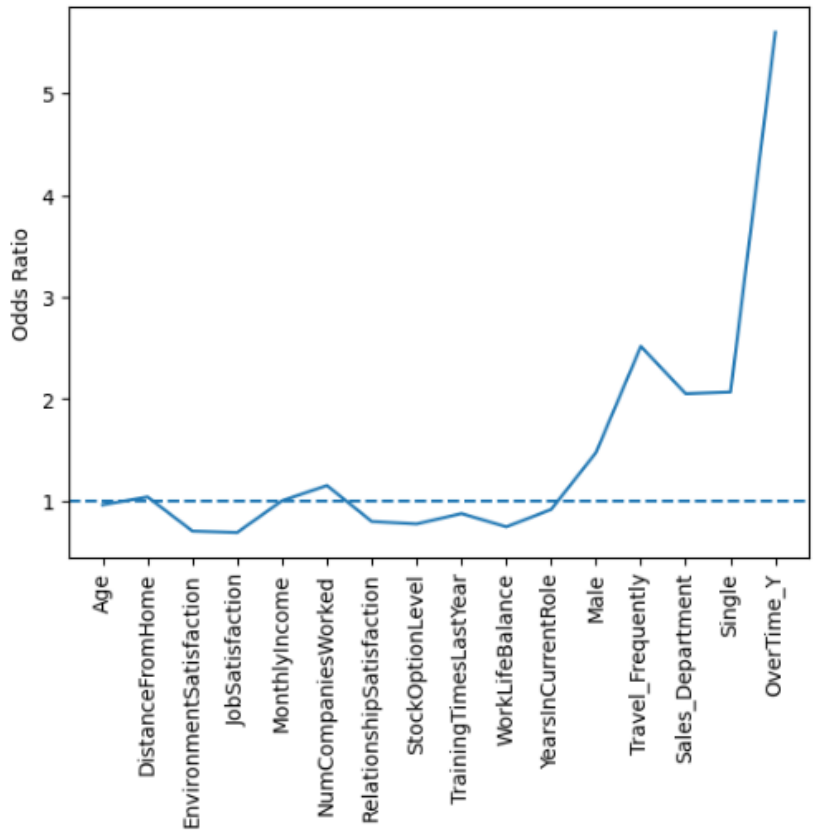
```
Optimization terminated successfully.
         Current function value: 0.331214
         Iterations 7
```

Out[47]:

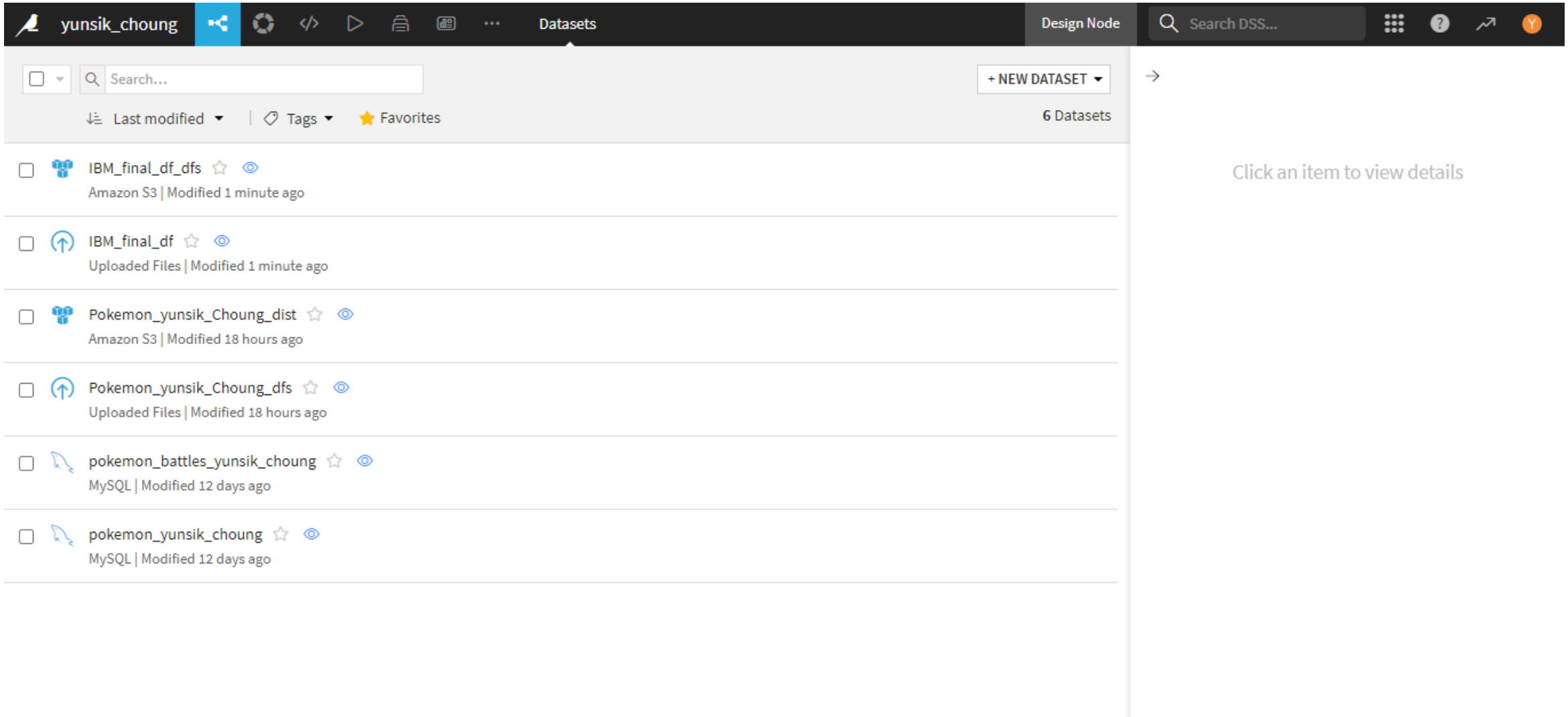
Model:	Logit	Pseudo R-squared:	0.250
Dependent Variable:	Attrition_YN	AIC:	1007.7700
Date:	2023-07-25 09:51	BIC:	1097.7513
No. Observations:	1470	Log-Likelihood:	-486.89
Df Model:	16	LL-Null:	-649.29
Df Residuals:	1453	LLR p-value:	1.8140e-59
Converged:	1.0000	Scale:	1.0000
No. Iterations:	7.0000		

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
Intercept	2.2186	0.6452	3.4388	0.0006	0.9541	3.4831
Age	-0.0409	0.0113	-3.6310	0.0003	-0.0630	-0.0188
DistanceFromHome	0.0363	0.0099	3.6564	0.0003	0.0168	0.0557
EnvironmentSatisfaction	-0.3545	0.0754	-4.7020	0.0000	-0.5023	-0.2067
JobSatisfaction	-0.3740	0.0744	-5.0284	0.0000	-0.5197	-0.2282
MonthlyIncome	-0.0001	0.0000	-3.6660	0.0002	-0.0002	-0.0000
NumCompaniesWorked	0.1391	0.0339	4.1027	0.0000	0.0726	0.2055
RelationshipSatisfaction	-0.2290	0.0761	-3.0104	0.0026	-0.3781	-0.0799
StockOptionLevel	-0.2593	0.1381	-1.8770	0.0605	-0.5300	0.0115
TrainingTimesLastYear	-0.1352	0.0673	-2.0076	0.0447	-0.2671	-0.0032
WorkLifeBalance	-0.2972	0.1137	-2.6136	0.0090	-0.5201	-0.0743
YearsInCurrentRole	-0.0901	0.0293	-3.0795	0.0021	-0.1474	-0.0327
Male	0.3872	0.1715	2.2582	0.0239	0.0511	0.7233
Travel_Frequently	0.9226	0.1918	4.8103	0.0000	0.5467	1.2986
Sales_Department	0.7185	0.1760	4.0813	0.0000	0.3734	1.0635
Single	0.7258	0.2263	3.2076	0.0013	0.2823	1.1694
OverTime_Y	1.7231	0.1742	9.8941	0.0000	1.3817	2.0644

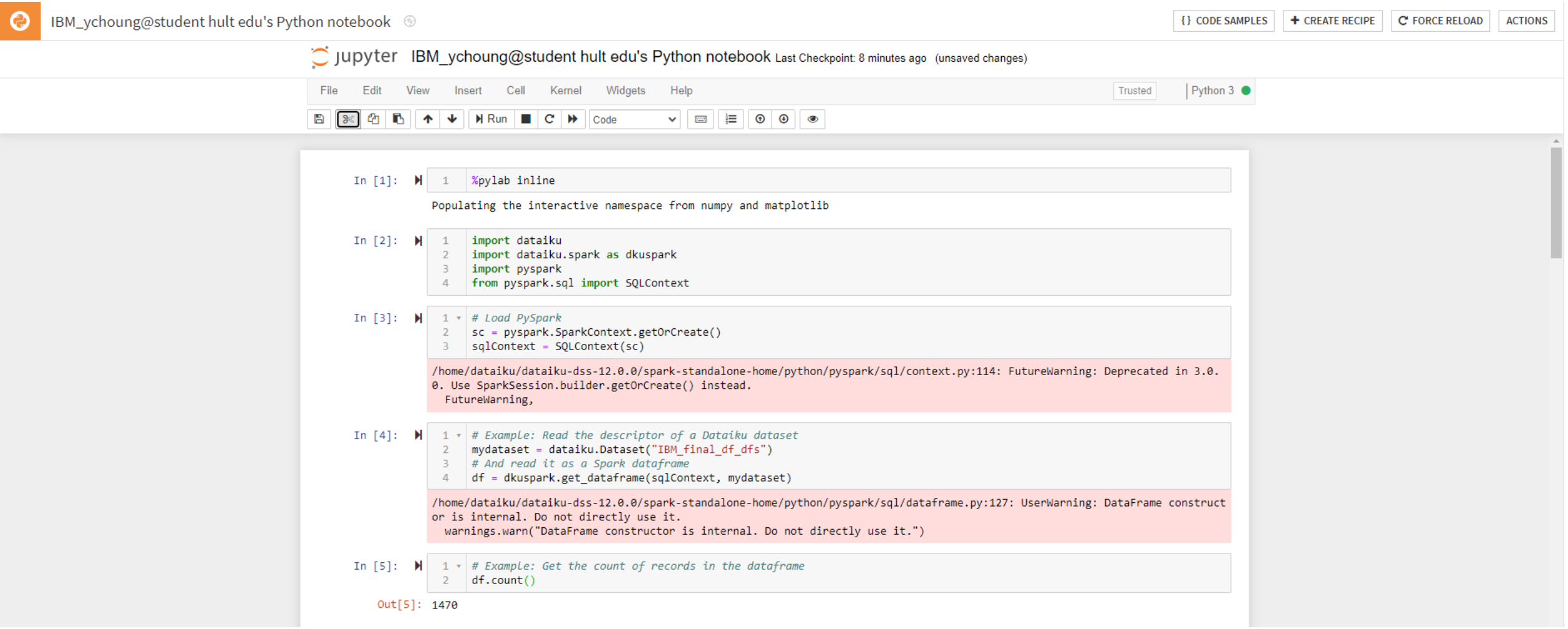
In [50]:
```
# Odds Ratio Visualization
g = sns.lineplot(data=or_tbl.iloc[1:]['Odds Ratio'], )
g.tick_params(axis='x', rotation=90)
g.axhline(y=1.0,linestyle='--')
plt.show()
```



Data Store at Dataiku DFS



Dataiku Notebook Dataset Connection



Building Logistic Classification ML Model at Dataiku PySpark

In [6]:
```
# Loading Libraries
from pyspark.ml import Pipeline
from pyspark.ml.classification import LogisticRegression
from pyspark.ml.feature import HashingTF, Tokenizer
from pyspark.sql import Row
from pyspark.sql.functions import UserDefinedFunction
from pyspa
from pyspa
from pyspa
from pyspa
```

In [14]:
```
# Define F
X = ['Age'
 'Distance
 'Environm
 'JobSatis
 'MonthlyI
 'NumCompa
 'Relation
 'StockOpt
 'Training
 'WorkLife
 'YearsInC
 'Male',
 'Travel_F
 'Sales_De
 'Single',
 'OverTime
```

In [15]:
```
#creating vectors with names of variables
vecAssembler = VectorAssembler(inputCols = X, outputCol
v_df = vecAssembler.transform(df)
vhouse_df = v_df.select(['features', 'Attrition_YN'])
vhouse_df = vhouse_df.withColumnRenamed("Attrition_YN",
```

In [16]:
```
#splitting the dataset
splits = vhouse_df.randomSplit([0.7, 0.3])
train_df = splits[0]
test_df = splits[1]
```

In [17]:
```
#creating an object with the logistic regression engine
lr = LogisticRegression(maxIter=20)
pipeline = Pipeline(stages=[lr])

#fitting the model
model = lr.fit(train_df)

#evaluating the model using testing data
result = model.transform(test_df)
result.prediction
result.show()

from pyspark.ml.evaluation import BinaryClassificationEvaluator
evaluator = BinaryClassificationEvaluator(rawPredictionCol="rawPrediction")
AUC_ROC = evaluator.evaluate(result,{evaluator.metricName: "areaUnderROC"})
print('AUC ROC:' + str(AUC_ROC))
```

```
+--------------------+-----+--------------------+--------------------+----------+
|            features|label|       rawPrediction|         probability|prediction|
+--------------------+-----+--------------------+--------------------+----------+
|(16,[0,1,2,3,4,5,...|    0|[2.38577825943823...|[0.91573637605174...|       0.0|
|(16,[0,1,2,3,4,5,...|    0|[3.09257914366593...|[0.95658560204141...|       0.0|
|(16,[0,1,2,3,4,5,...|    0|[4.51570331595847...|[0.98918238940619...|       0.0|
|(16,[0,1,2,3,4,5,...|    1|[2.93904835883414...|[0.94974332351530...|       0.0|
|(16,[0,1,2,3,4,5,...|    1|[6.80194334287698...|[0.99888962150833...|       0.0|
|(16,[0,1,2,3,4,6,...|    0|[3.21327987735179...|[0.96133097547455...|       0.0|
|(16,[0,1,2,3,4,6,...|    0|[1.69647147146948...|[0.84507332672845...|       0.0|
|(16,[0,1,2,3,4,6,...|    1|[2.29889215270991...|[0.90878524598189...|       0.0|
|(16,[0,1,2,3,4,6,...|    0|[2.88163541994249...|[0.94693110777703...|       0.0|
|[18.0,5.0,2.0,2.0...|    1|[-2.6484791853138...|[0.06608280597565...|       1.0|
|[18.0,5.0,2.0,4.0...|    0|[1.32389054697974...|[0.78982826391330...|       0.0|
|[18.0,10.0,4.0,3....|    0|[0.40444001655082...|[0.59975395282569...|       0.0|
|[19.0,21.0,4.0,2....|    1|[-1.6146692393790...|[0.16594135990624...|       1.0|
|[20.0,2.0,3.0,3.0...|    1|[1.39588010226257...|[0.80152930900454...|       0.0|
|[20.0,2.0,3.0,3.0...|    0|[1.05199964666936...|[0.74115870131374...|       0.0|
|[20.0,3.0,1.0,3.0...|    0|[0.04981006246190...|[0.51244994165256...|       0.0|
|[20.0,4.0,1.0,1.0...|    1|[-0.1686360448166...|[0.45794061572145...|       1.0|
|[20.0,9.0,4.0,1.0...|    0|[-0.1409727814251...|[0.46481505550586...|       1.0|
|[20.0,10.0,4.0,3....|    1|[0.19399076441069...|[0.54834617084121...|       0.0|
|[21.0,15.0,3.0,4....|    0|[1.88839258608584...|[0.86857214559722...|       0.0|
+--------------------+-----+--------------------+--------------------+----------+
only showing top 20 rows

AUC ROC:0.8098939357137055
```

HULT INTERNATIONAL BUSINESS SCHOOL

Cross-Validating Logistic Classification ML Model at Dataiku PySpark

In [18]:

```
from pyspark.ml.evaluation import BinaryClassificationEvaluator

# Evaluate model
evaluator = BinaryClassificationEvaluator(rawPredictionCol="rawPrediction")
evaluator.evaluate(result)

from pyspark.ml.tuning import ParamGridBuilder, CrossValidator

# Create ParamGrid for Cross Validation
paramGrid = (ParamGridBuilder()
             .addGrid(lr.regParam, [0.01, 0.5, 2.0])
             .addGrid(lr.elasticNetParam, [0.0, 0.5, 1.0])
             .addGrid(lr.maxIter, [1, 5, 10])
             .build())
# Create 5-fold CrossValidator
cv = CrossValidator(estimator=lr, estimatorParamMaps=paramGrid, evaluator=evaluator, nu

# Run cross validations
cvModel = cv.fit(train_df)
# this will likely take a fair amount of time because of the amount of models that we'r

# Use test set to measure the accuracy of our model on new data
predictions = cvModel.transform(test_df)

# cvModel uses the best model found from the Cross Validation
# Evaluate best model
evaluator.evaluate(predictions)

print('Model Intercept: ', cvModel.bestModel.intercept)
weights = cvModel.bestModel.coefficients
weights = [(float(w),) for w in weights]  # convert numpy type to float, and to tuple
weightsDF = sqlContext.createDataFrame(weights, ["Feature Weight"])
weightsDF.show()
# View best model's predictions and probabilities of each prediction class
selected = predictions.select("label", "prediction", "probability", "features")
selected.show()
```

```
Model Intercept:  1.9190516483834124
+--------------------+
|      Feature Weight|
+--------------------+
|-0.04369386980471...|
|0.035427510733245716|
| -0.2551387647715912|
|-0.27628731814289725|
|-9.06374172468278...|
| 0.12822677528119478|
|-0.23009935781559307|
|-0.14807598935330474|
|-0.10791676899760719|
|-0.35650727402803806|
|-0.10134574674295341|
|  0.4458859863296991|
|  0.9939280604547683|
|  0.6025395829110497|
|  0.7626955650098408|
|    1.45745781253733|
+--------------------+

+-----+----------+--------------------+--------------------+
|label|prediction|         probability|            features|
+-----+----------+--------------------+--------------------+
|    0|       0.0|[0.89928616669743...|(16,[0,1,2,3,4,5,...|
|    0|       0.0|[0.94272640413864...|(16,[0,1,2,3,4,5,...|
|    0|       0.0|[0.98182615401102...|(16,[0,1,2,3,4,5,...|
|    1|       0.0|[0.93367768167388...|(16,[0,1,2,3,4,5,...|
|    1|       0.0|[0.99737433867805...|(16,[0,1,2,3,4,5,...|
|    0|       0.0|[0.94581082995688...|(16,[0,1,2,3,4,6,...|
|    0|       0.0|[0.83367294845035...|(16,[0,1,2,3,4,6,...|
|    1|       0.0|[0.89169093077322...|(16,[0,1,2,3,4,6,...|
|    0|       0.0|[0.93043306011001...|(16,[0,1,2,3,4,6,...|
|    1|       1.0|[0.10398648737237...|[18.0,5.0,2.0,2.0...|
|    0|       0.0|[0.78073104674648...|[18.0,5.0,2.0,4.0...|
|    0|       0.0|[0.62089237210116...|[18.0,10.0,4.0,3....|
|    1|       1.0|[0.22195636616613...|[19.0,21.0,4.0,2....|
|    1|       0.0|[0.79082470724875...|[20.0,2.0,3.0,3.0...|
|    0|       0.0|[0.74328136181085...|[20.0,2.0,3.0,3.0...|
|    0|       0.0|[0.55780977389199...|[20.0,3.0,1.0,3.0...|
|    1|       0.0|[0.50568789919049...|[20.0,4.0,1.0,1.0...|
|    0|       0.0|[0.51669241271838...|[20.0,9.0,4.0,1.0...|
|    1|       0.0|[0.56872664126864...|[20.0,10.0,4.0,3....|
|    0|       0.0|[0.85194641494331...|[21.0,15.0,3.0,4....|
+-----+----------+--------------------+--------------------+
only showing top 20 rows
```