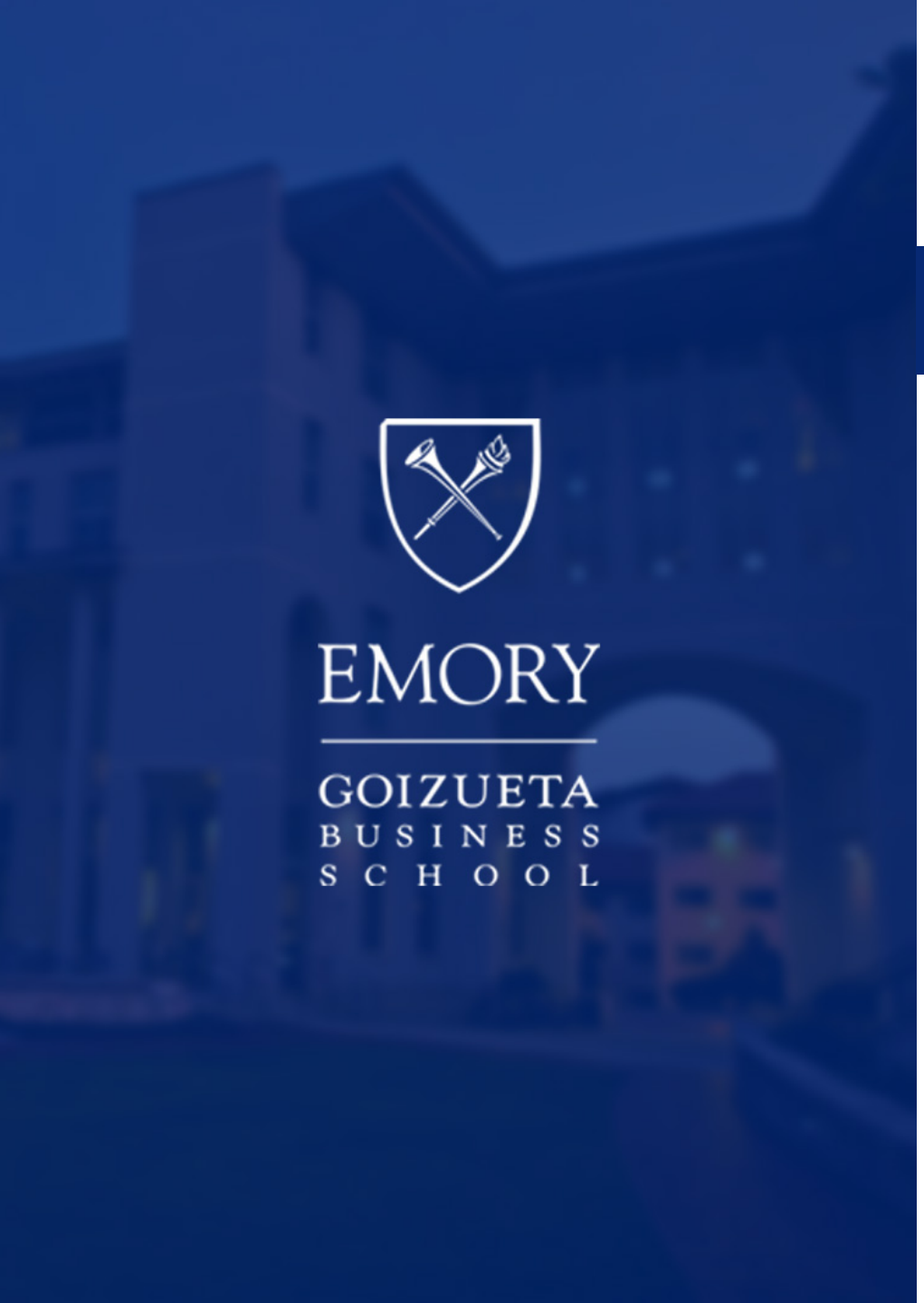
EMORY
GOIZUETA BUSINESS SCHOOL

Master of Science
in Business Analytics
MSBA

How to keep talents?

Group 11
"Stranger things"

Kwaku Danso-Manu
Yan Li
Miller Page
Kami Wu



- Business Understanding
- Data Understanding
- Data Preparation
- Modeling
- Evaluation & Model Selection
- Deployment

Business Understanding

Avoidable turnover is costing employers big.

EMORY GOIZUETA BUSINESS SCHOOL

Master of Science in Business Analytics MSBA

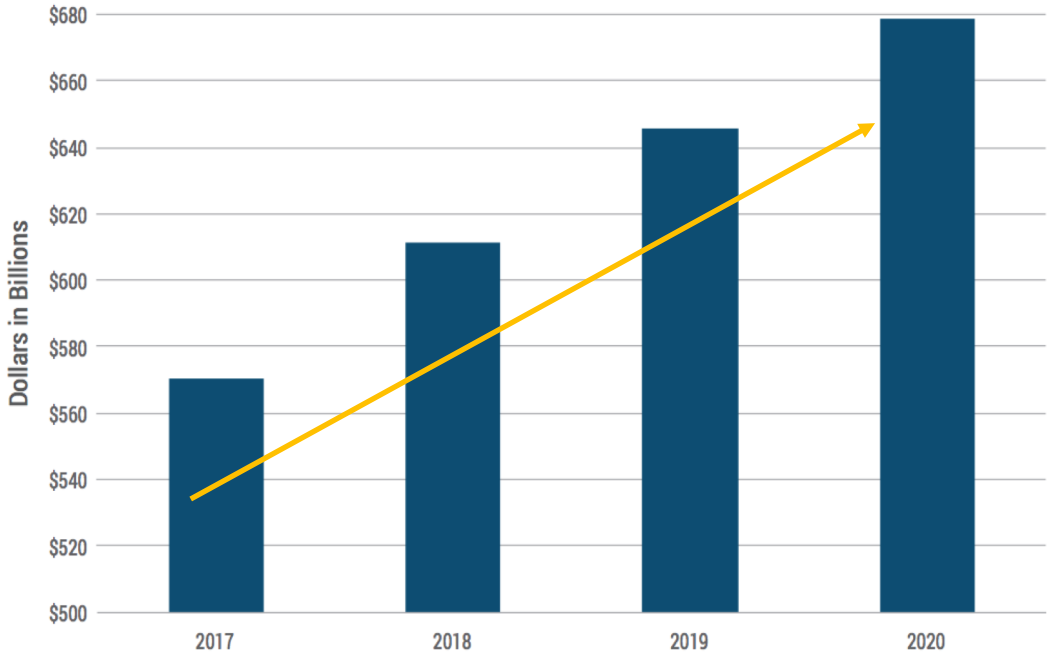
Total Cost of Voluntary Turnover in the U.S. is $536 Billion

In 2016, BLS reported 35,746,000 voluntary separations (Quits), which would amount to $536 billion in costs to employers. This follows a cost of turnover estimate of $15,000 per employee.

Estimating Average Turnover Cost

Meta-Analysis found conservative turnover cost for $8 per hour employee [8]	= $5,506
Annual Salary of $8 per hour employee	= $16,640
Turnover cost per employee: $5,506 / $16,640	= 33%
Median U.S. worker salary [9]	= $45,000
Average cost of turnover per employee: 33% of $45,000	**= $15,000**

Annual Cost of U.S. Turnover in Billions of Dollars



Source: Employee Benefit News, "2017 Retention Report"&"2018 Retention Report" from Work Institute

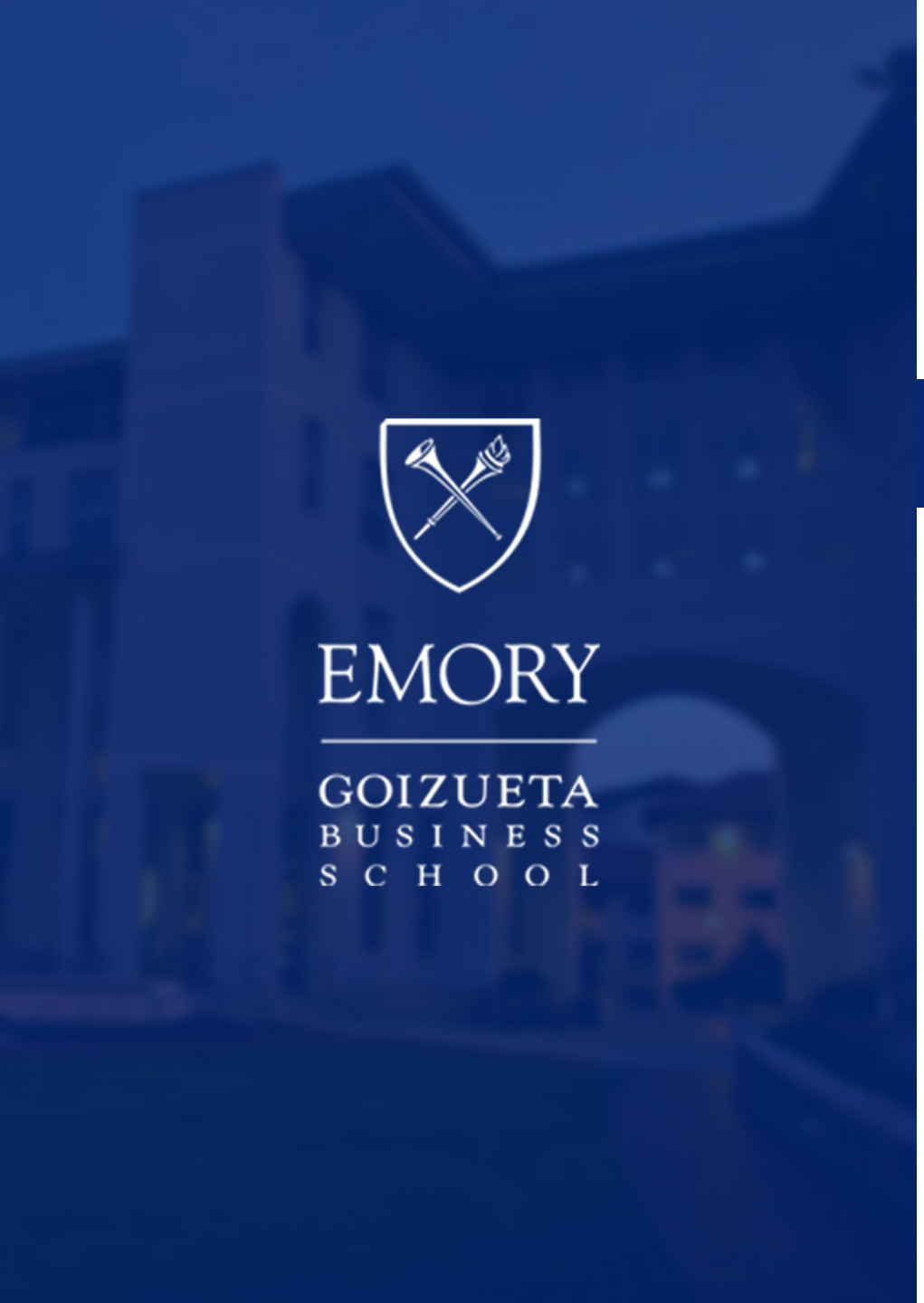
Business Understanding

The goal is to predict if an employee leaves a company within 6 months based on his/her behaviors

Cost/Benefit Matrix

	Actual Leave	Actual Stay
Predicted Leave	Costly, but worthwhile if they stay (+)	Relatively Expensive and unnecessary (-)
Predicted Stay	Most Expensive option (--)	No cost incurred (0)

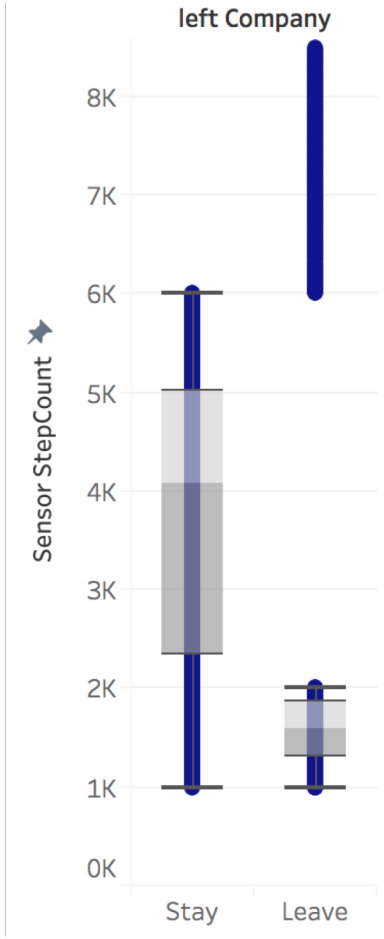
- **Replacing employees is expensive**
  - Hiring process, training cost, lost productivity
- **Knowing what makes employees leave will help prevent turnover**
- **More critical to know when someone leaves rather than vice versa**
  - i.e., we prefer false negatives vs. false positives

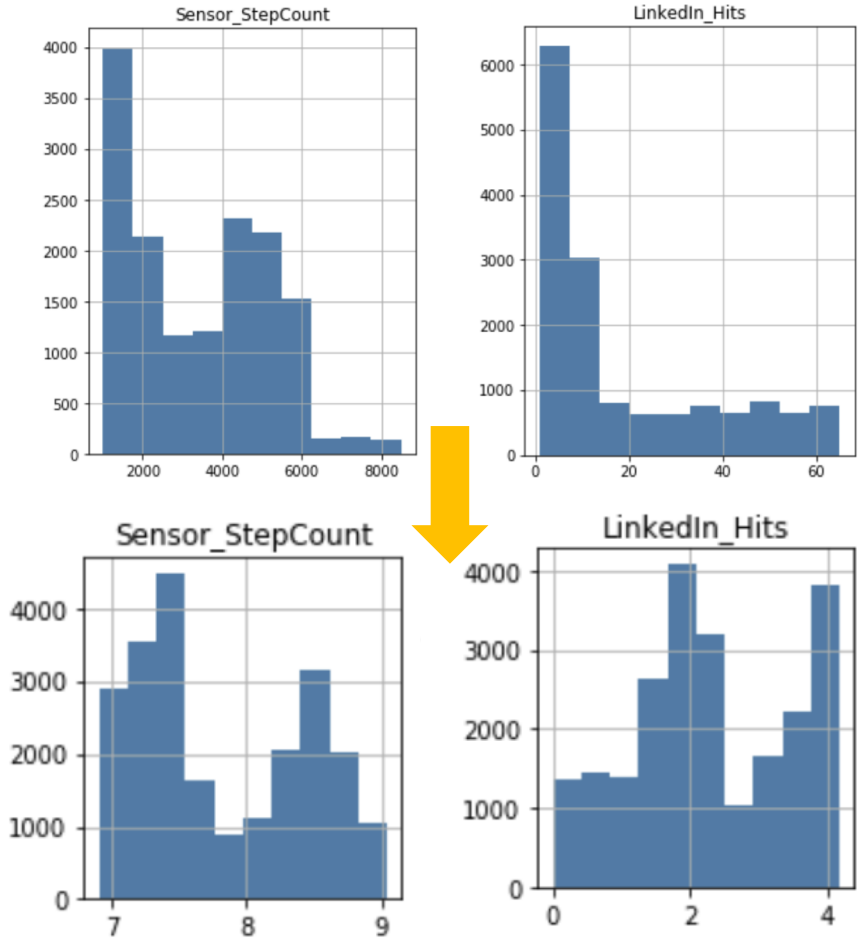


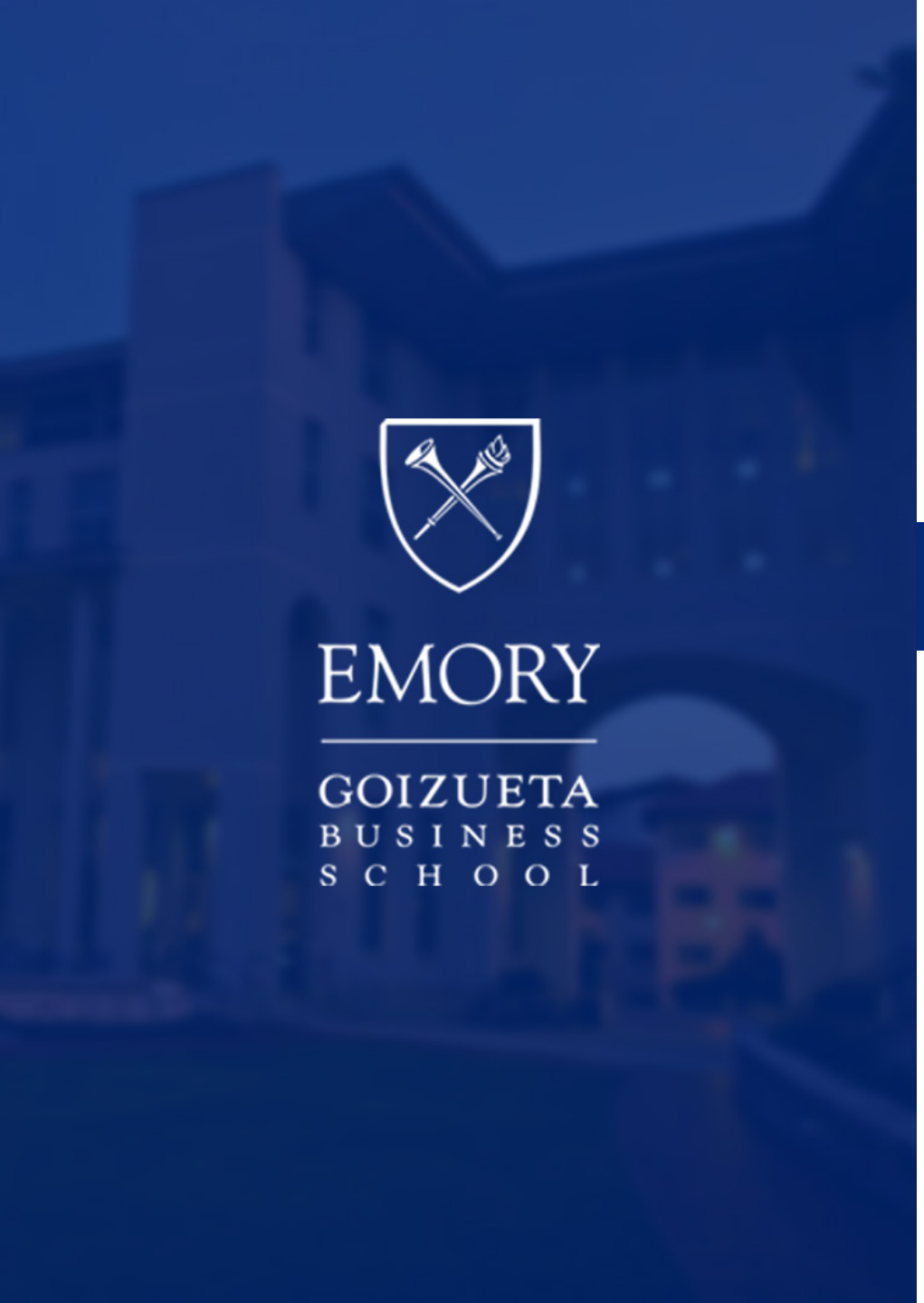
- Business Understanding
- Data Understanding
- Data Preparation
- Modeling
- Evaluation & Model Selection
- Deployment

Data Understanding

- **Data Collection**
  - measuring talent
  - predicting attrition
- **Data Dimensions**
  - 14,999 observations
  - 62 attributes
  - inferable missing values
- **Data Reliability**
  - ambiguous survey data
  - redundant attributes
- **Data Exploration**
  - Target Variable(0 - Stay; 1 – Leave)
  - Class Probability estimation based on business understanding

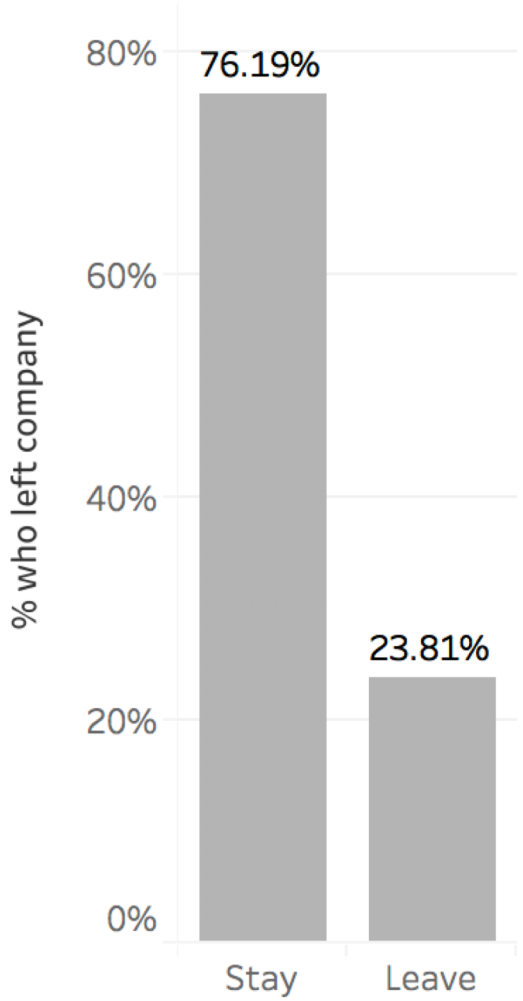






- Business Understanding
- Data Understanding
- Data Preparation
- Modeling
- Evaluation & Model Selection
- Deployment

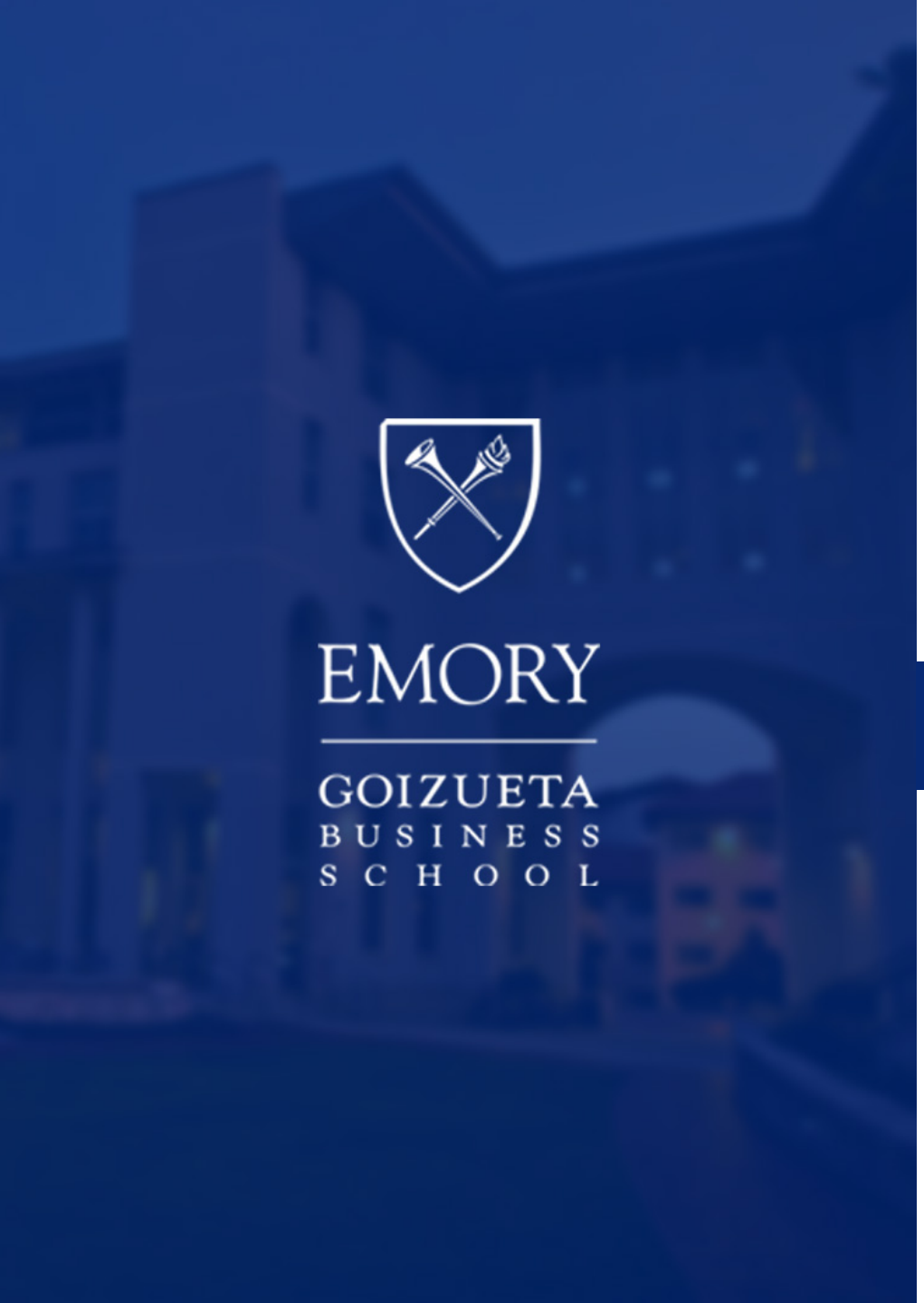
Data Preparation



Department
Finance
Operations
HR
Warehouse
Sales
IT

Finance	Operations	HR	Warehouse	Sales	IT
1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	0	0	0
0	0	0	1	0	0
0	0	0	0	1	0
0	0	0	0	0	1

- **Variable type conversions**
- **Categorical conversions**
  - Ordinal
  - Binary
  - Dummy
- **Deleting one dummy to reduce multicollinearity**
- **No need to resample**



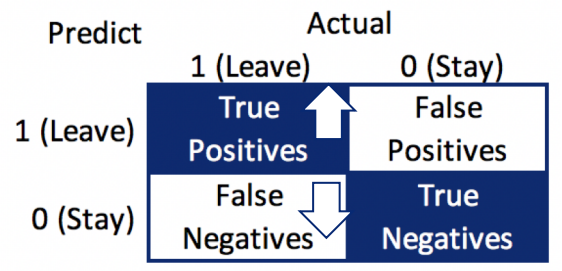
- Business Understanding
- Data Understanding
- Data Preparation
- Modeling
- Evaluation & Model Selection
- Deployment

Modeling

We chose "Recall (Positive)" as the main scorer and looked for the best para--meters for kNN, Decision Tree, Logistic Regression Classifiers using **grid search**

EMORY GOIZUETA BUSINESS SCHOOL | Master of Science in Business Analytics MSBA

Scoring:
Recall Rate (Positive)



Cost incurred when:

- Spending on keeping people who are predicted to leave and actually are about to leave ---- TP
- Spending on keeping people who are predicted to leave but actually aren't in need of any incentives ----FP
- Spending on finding substitutes for the people left and were predicted wrong ----FN

Random Seed: 11

Decision Tree

X_train, y_train

↓ Plot fitting curve
- Max Depth
- Min Impurity Decrease
- Min sample each node
- Min sample each split

X_train, y_train
Parameters Range

↓ Grid Search
- For the same parameters
- CV = 5

Best set of parameters
Score: recall
X_train, y_train

↓ Fit the model

Best Decision Tree

kNN

X_train, y_train

↓ Feature Selection
↓ Z-Score Scaling

Standardized X_train, y_train

↓ Plot fitting curve
- Number of K

Standardized X_train, y_train
Parameters Range

↓ Grid Search
- K, P(Distance)
- CV = 5

Best set of parameters
Score: recall
Standardized X_train, y_train

↓ Fit the model

Best k-NN Classifier

Logistic Regression

X_train, y_train

↓ Transformation

Transformed X_train, y_train

↓ Plot fitting curve
- Parameter C

Transformed X_train, y_train
Parameters Range

↓ Grid Search
- C, Penalty
- CV = 5

Best set of parameters
Score: recall
Transformed X_train, y_train

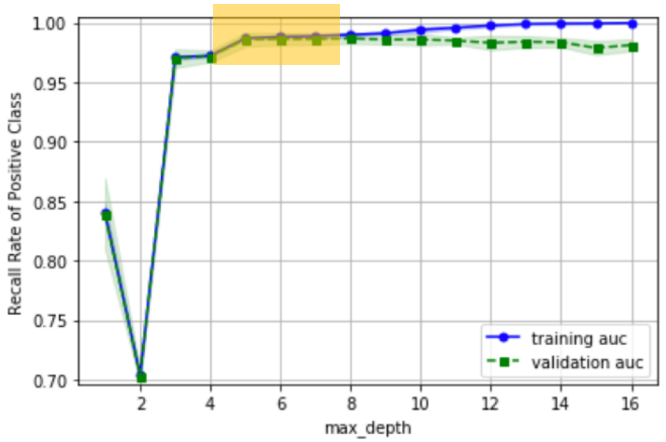
↓ Fit the model

Best Logistic Regression Classifier

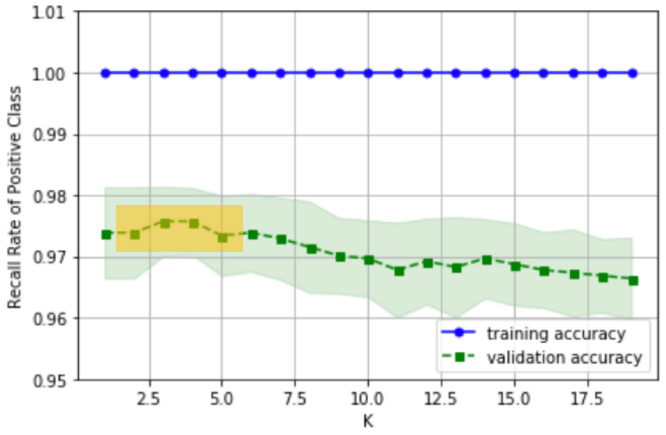
Modeling

We used **fitting curve** to set a range for the parameters to minimize the grid search running time.

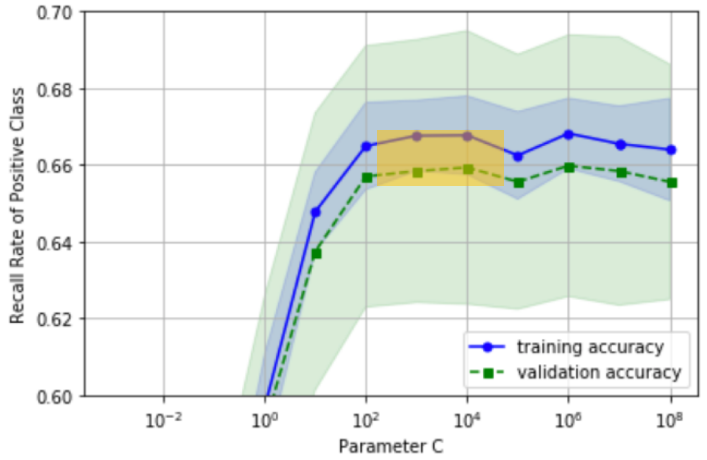
Decision Tree



kNN



Logistic Regression



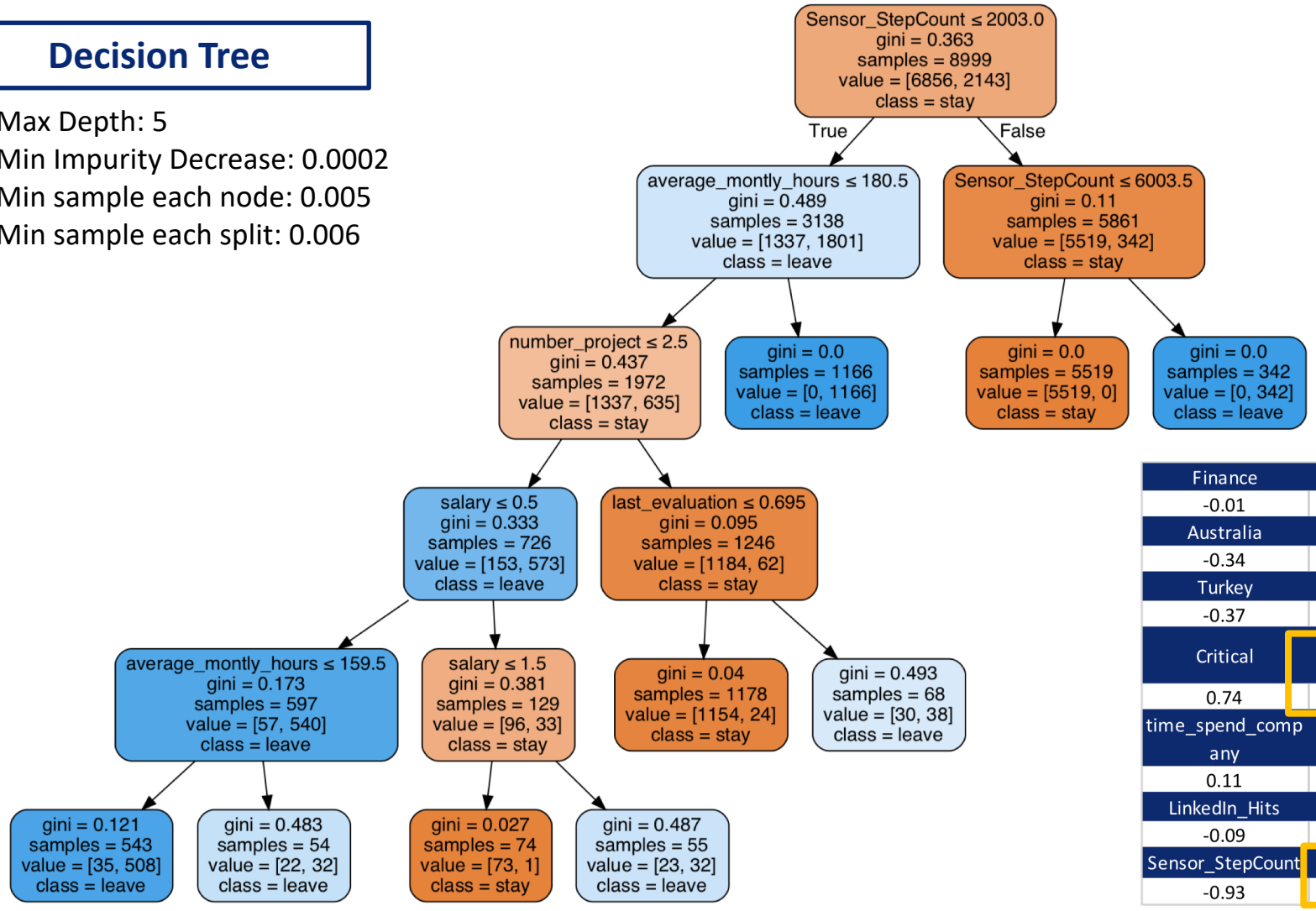
Modeling

The best DT has a depth of 5; the best kNN has a k of 4;
The best Logit has a C around 3×10^4, penalty = 'l2'.

Master of Science in Business Analytics MSBA

Decision Tree

- Max Depth: 5
- Min Impurity Decrease: 0.0002
- Min sample each node: 0.005
- Min sample each split: 0.006



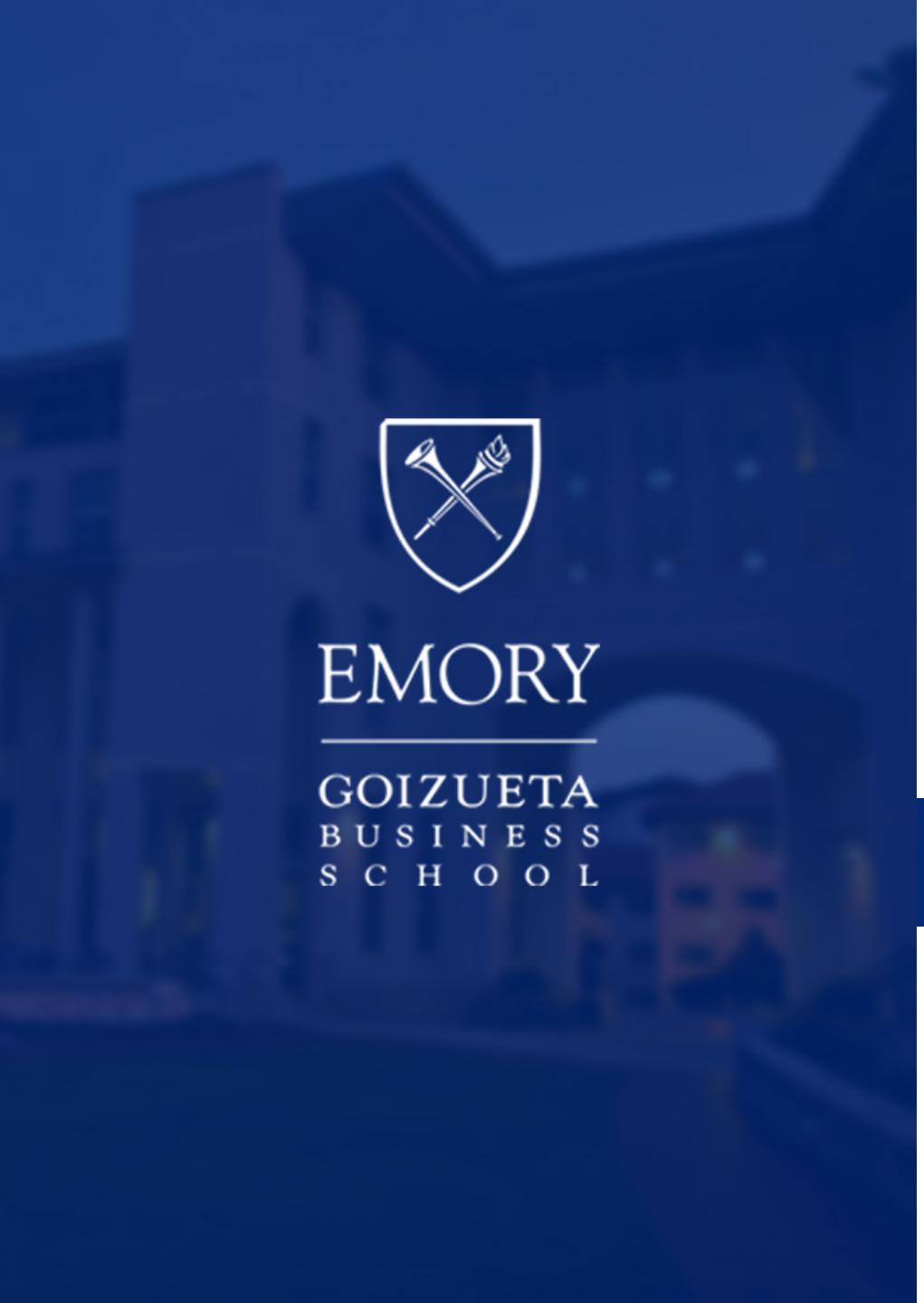
kNN

- Number of neighbors: 4
- Power parameter for the Minkowski metric: 2

Logistic Regression

- C = 31623
- P: l2

Finance	HR	IT	Operations	Sales
-0.01	0.01	0.07	-0.02	-0.04
Australia	China	France	Japan	Korea
-0.34	-0.06	-0.36	-0.18	-0.29
Turkey	UK	US	Role	Will_Relocate
-0.37	-0.14	-0.17	-0.05	-0.01
Critical	Percent_Remote	last_evaluation	number_project	average_montly_hours
0.74	14.47	-4.00	-0.56	0.00
time_spend_company	Work_accident	promotion_last_5years	salary	Gender
0.11	-1.44	2.51	-0.34	-0.11
LinkedIn_Hits	Emp_Identity	Emp_Role	Emp_Position	Emp_Title
-0.09	-1.17	-1.21	-1.10	-0.97
Sensor_StepCount	Sensor_Heartbeat(Average/Min)		Sensor_Proximity(1-highest/10-	
-0.93	-4.53		-0.04	



- Business Understanding
- Data Understanding
- Data Preparation
- Modeling
- Evaluation & Model Selection
- Deployment

Model Evaluation

We evaluated the generalization performance of each classifier using cross-validation on training data

- Decision Tree and kNN have very high scores
- Recall rate of class “Leave” is a more important metric in this problem

Cross-Validation Results

Score	Decision Tree	kNN	Logistic Regression
Accuracy	0.9848 +/- 0.0024	0.9858 +/- 0.0029	0.8751 +/- 0.0109
Precision(Leave)	0.9506 +/- 0.0078	0.9650 +/- 0.0089	0.7799 +/- 0.0194
F-1(Leave)	0.9687 +/- 0.0048	0.9703 +/- 0.0061	0.7157 +/- 0.0289
Recall(Leave)	0.9874 +/- 0.0038	0.9757 +/- 0.0055	0.6617 +/- 0.0364
AUC	0.9968 +/- 0.0011	0.9896 +/- 0.0030	0.9167 +/- 0.0115

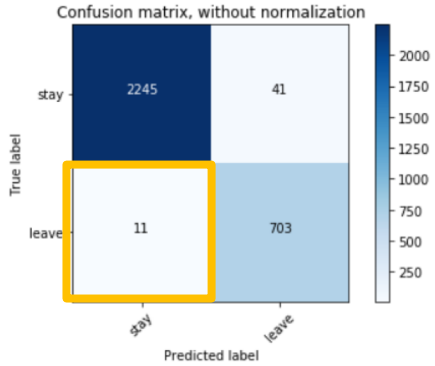
Model Selection

Our Decision Tree model performs the best, giving us a low number of false negatives and a high recall rate of positive class

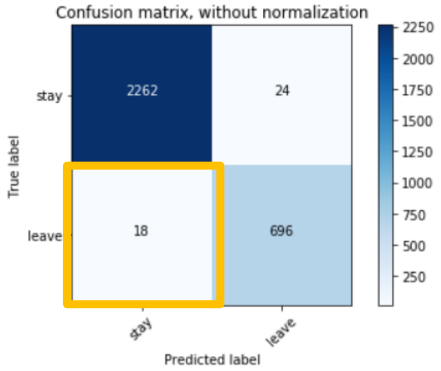
- We use the best models from the grid search and the validation data set to evaluate and compare the models
- Decision Tree performs slightly better than kNN

Confusion Matrix

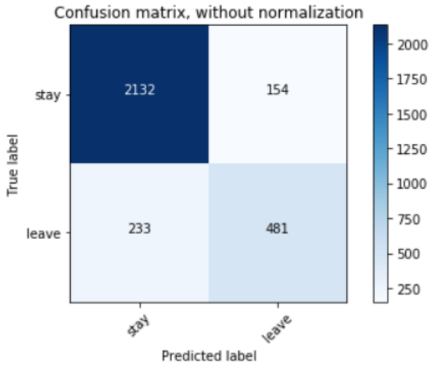
Decision Tree



kNN



Logistic Regression



Reports

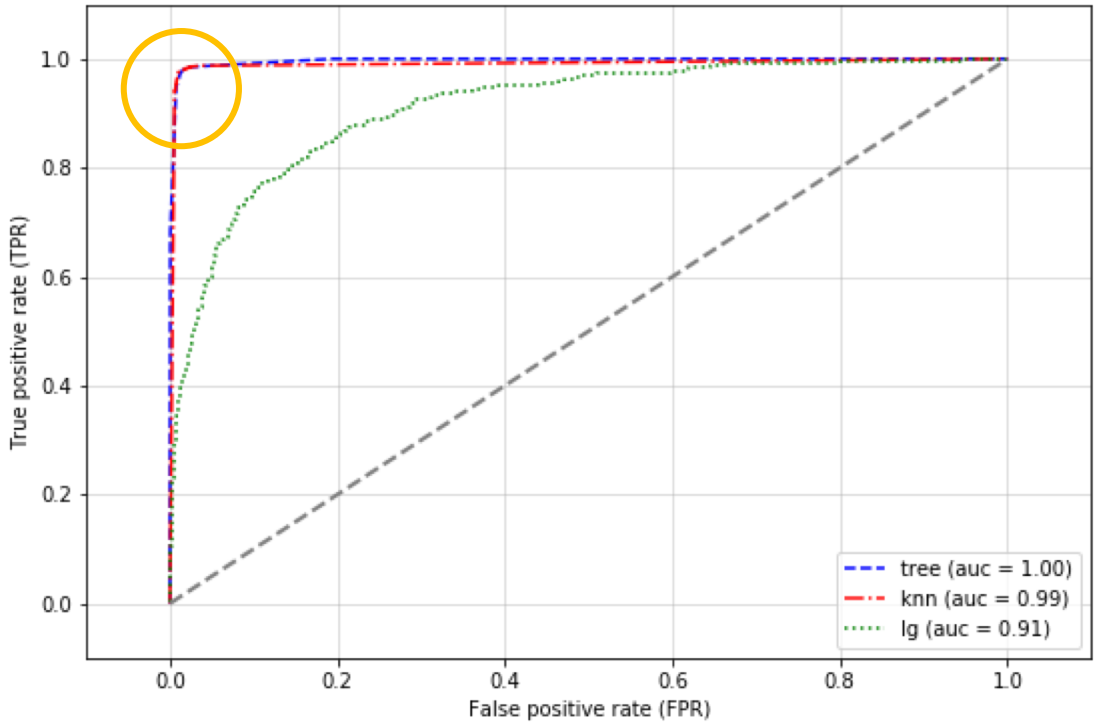
Score	Decision Tree	kNN	Logistic Regression
Precision (Leave)	0.94	0.97	0.76
F-1 (Leave)	0.96	0.97	0.71
Recall (Leave)	0.98	0.97	0.67

Cut-off Selection

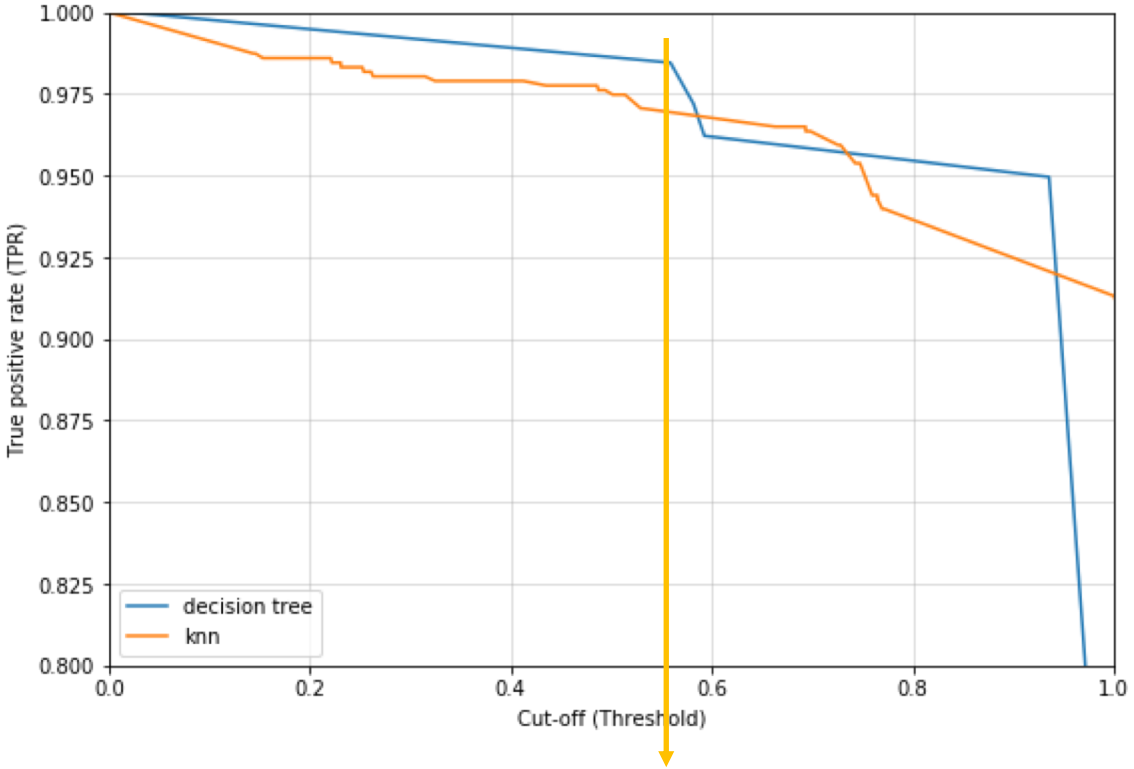
We chose the cut-off as 0.5588 based on ROC curve

- We suppose to choose the most north-western point on the ROC curve, to do that, we plot cut-off against TPR, and find the most ideal cut-off as 0.5588

ROC Curve



Cut-off selection



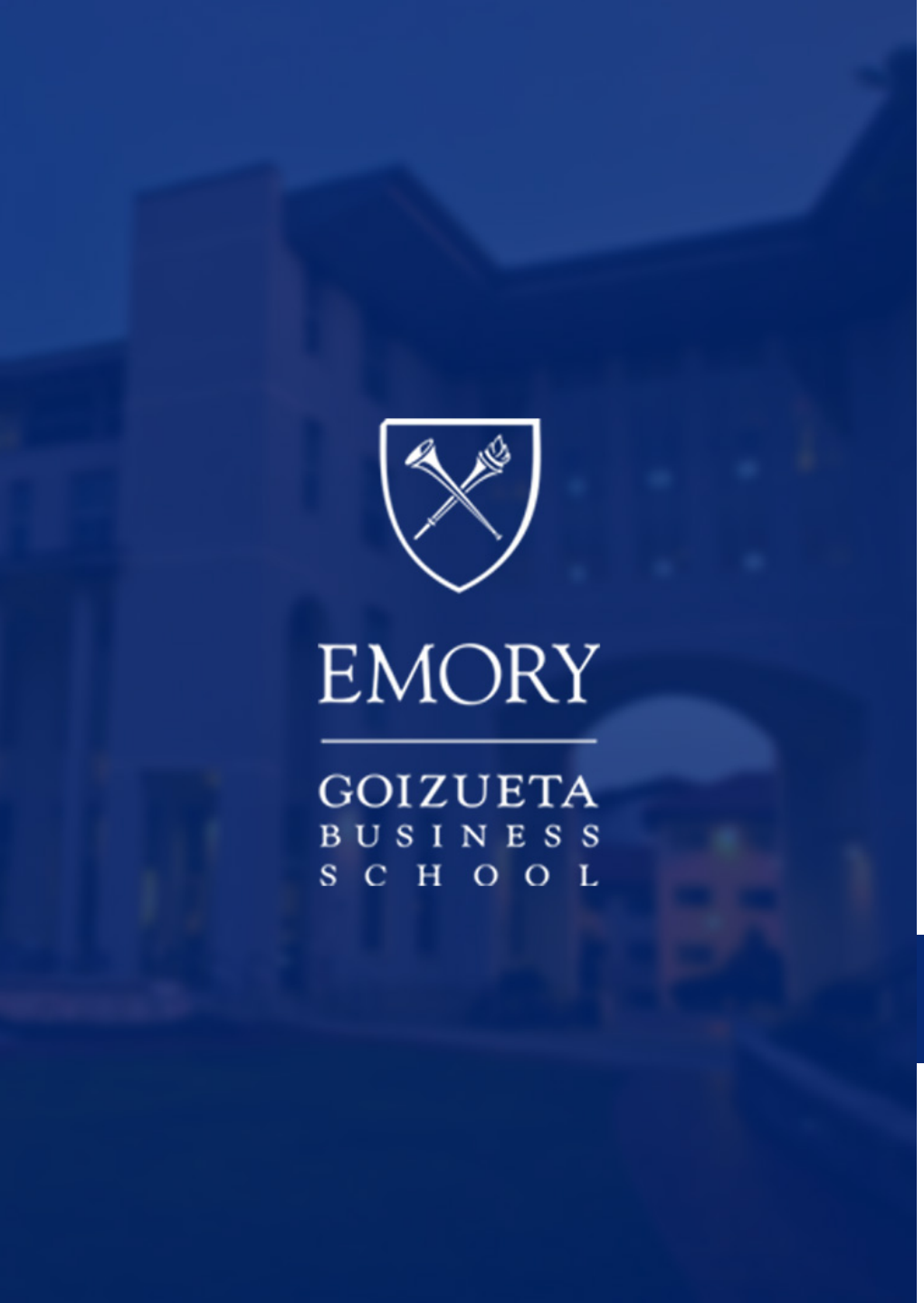
Cut-off: 0.5588

Learning Curve

According to learning curve, we've got enough data for training the model

- As our training data set has 9000 observations, we can see from learning curve that we have enough data to build a complete model

Learning Curve



- Business Understanding
- Data Understanding
- Data Preparation
- Modeling
- Evaluation & Model Selection
- Deployment

Deployment

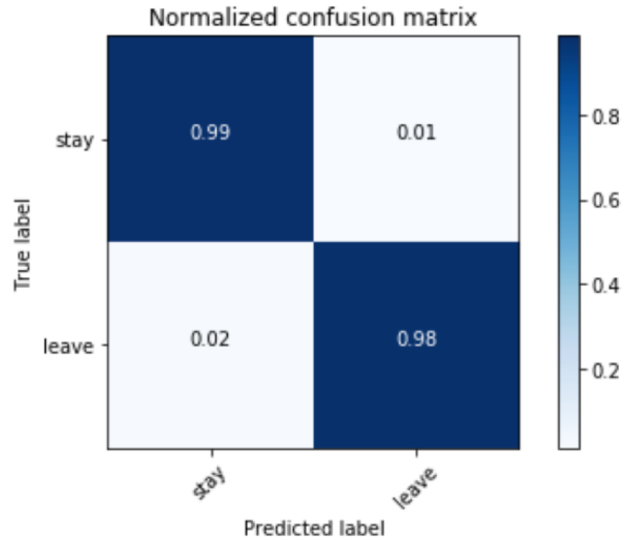
Decision Tree provided the best answer to our business problem.

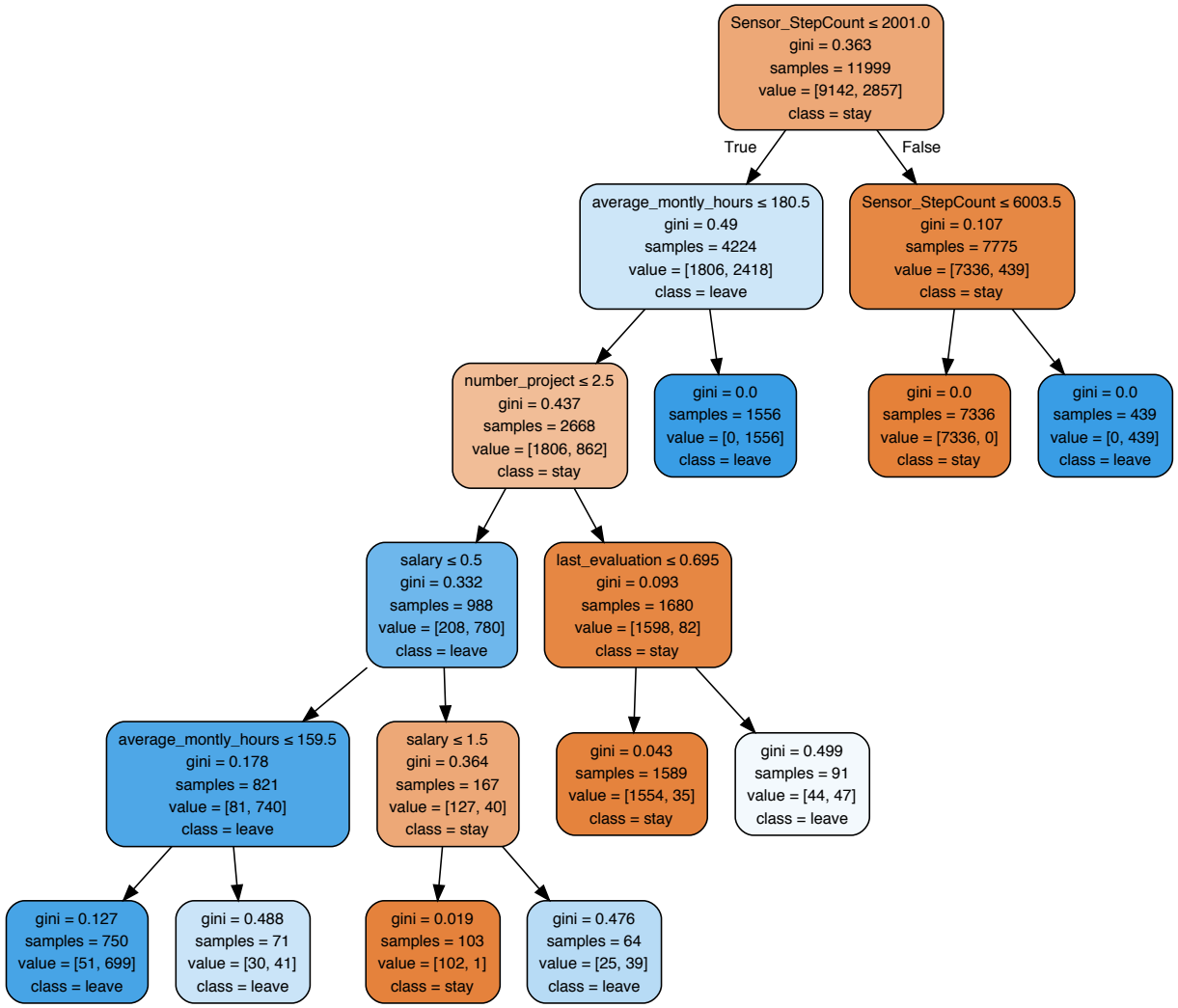
EMORY GOIZUETA BUSINESS SCHOOL | Master of Science in Business Analytics MSBA

Results from Test Data

Accuracy Report

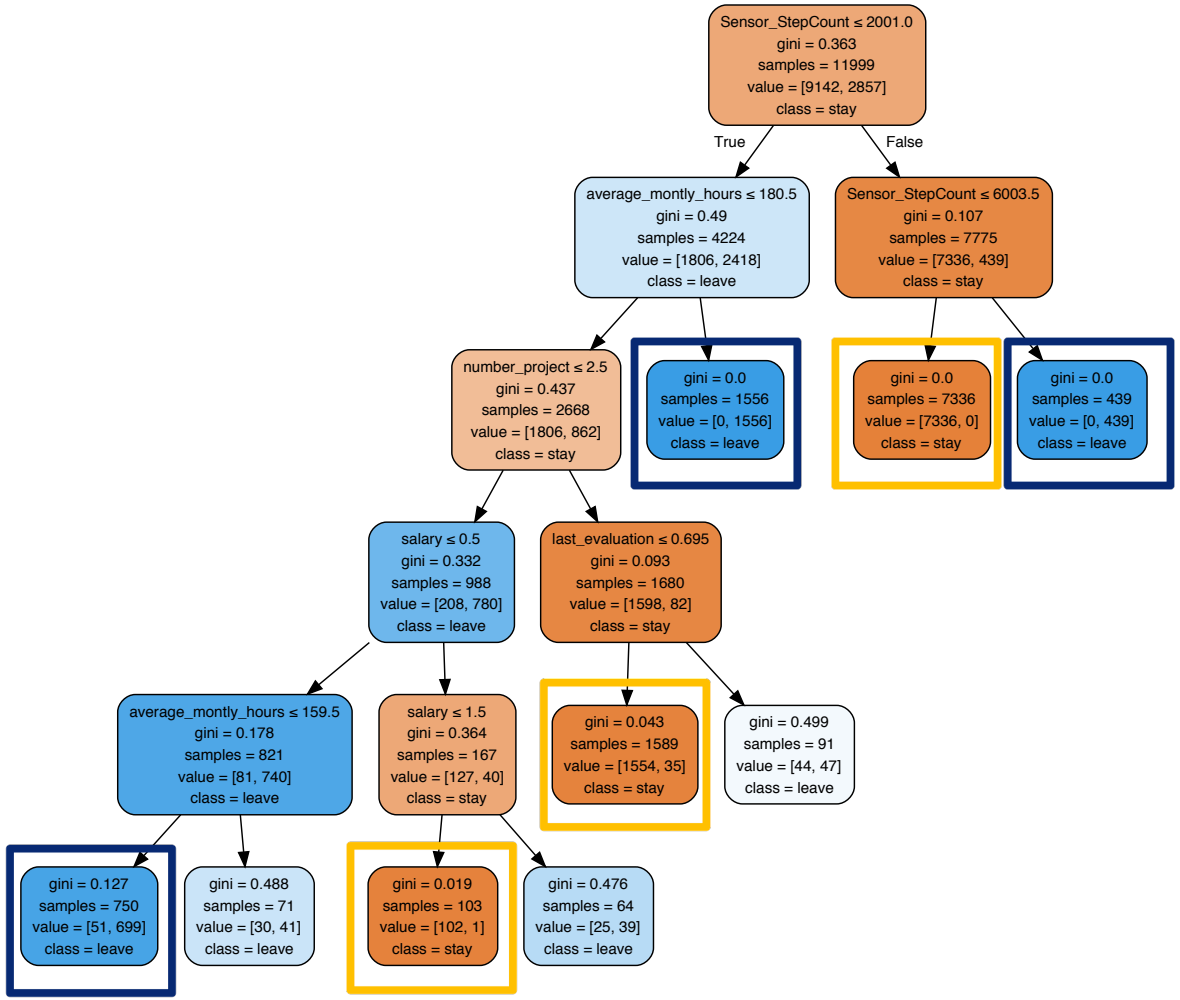
	precision	recall	f1-score	support
Stay	0.99	0.99	0.99	2286
Leave	0.95	0.98	0.97	714
avg / total	0.98	0.98	0.98	3000





Deployment

Converting the decision tree nodes into probability based estimations, and using the Laplace correction to account for uncertainty



Frequency Based Est.	Smoothed w/ Laplace
$\frac{439}{439} = 1$	$\frac{439+25}{439+50} = .949$
$\frac{0}{7336} = 0$	$\frac{0+25}{7336+50} = .003$
$\frac{1556}{1556} = 1$	$\frac{1556+25}{1556+50} = .984$
$\frac{47}{91} = .516$	$\frac{47+25}{91+50} = .510$
$\frac{35}{1589} = .022$	$\frac{35+25}{1589+50} = .036$
$\frac{39}{64} = .609$	$\frac{39+25}{64+50} = .561$
$\frac{1}{103} = .010$	$\frac{1+25}{103+50} = .170$
$\frac{41}{71} = .577$	$\frac{41+25}{71+50} = .545$
$\frac{699}{750} = .932$	$\frac{699+25}{750+50} = .905$

Deployment

Implementing our results will help prevent future employee turnover.

- **Plan to reduce turnover:**
  - Increase employee-to-employee interaction
  - Provide deserved bonus/raise
  - Reduce heavy monthly workload
- **Potential issue:**
  - Model predicts if people will leave, not if they can be persuaded to stay
- **Ethical concerns:**
  - The collection of personal data (screen proximity, heartrate, etc.)
  - Potential for discrimination
- **Potential risks:**
  - Isolating employees not selected
  - Cost of keeping an employee higher than replacing

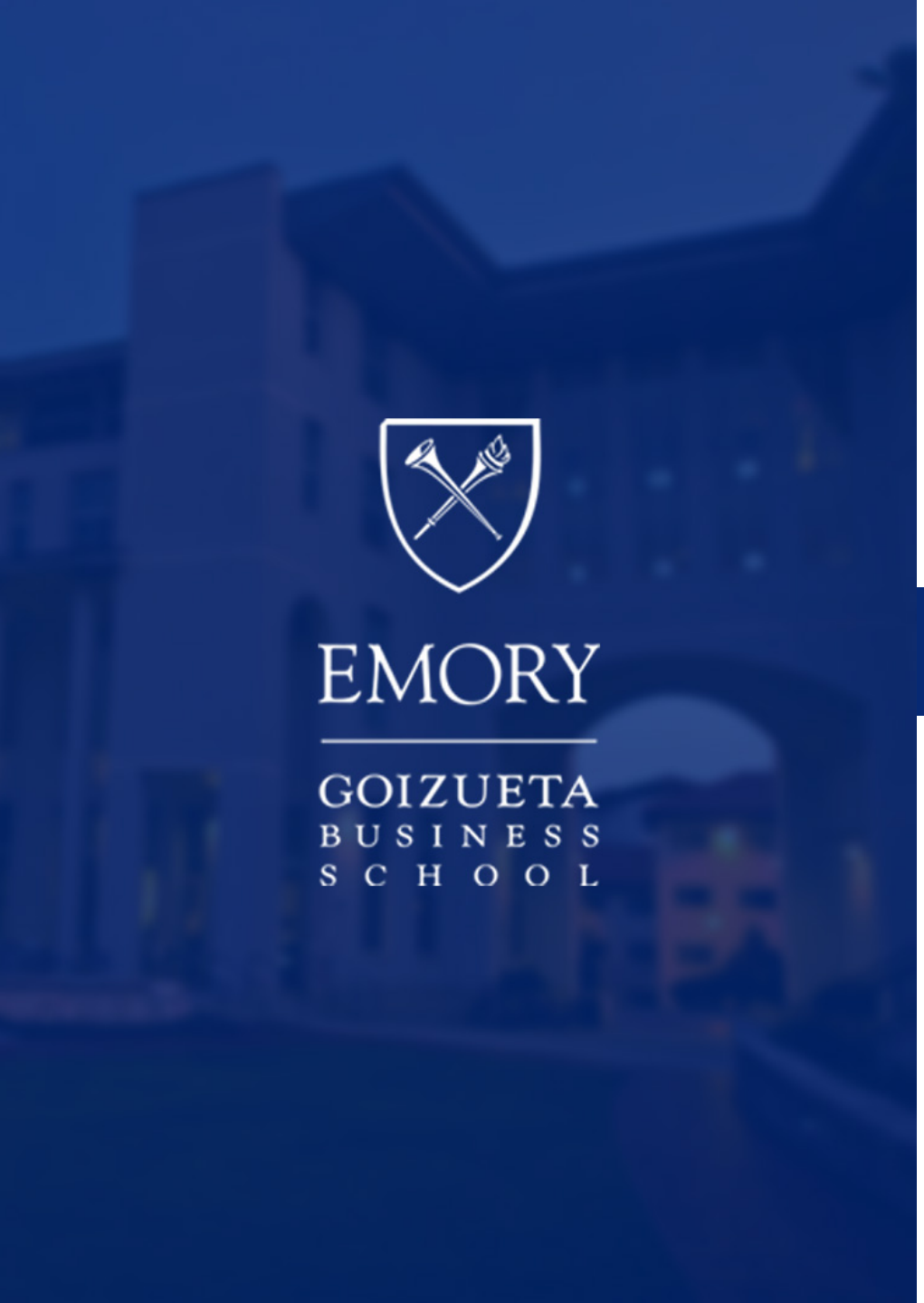
EMORY
GOIZUETA
BUSINESS
SCHOOL

Master of Science
in Business Analytics
MSBA

Thank you!
Questions?

Group 11
"Stranger things"

Kwaku Danso-Manu
Yan Li
Miller Page
Kami Wu



- Appendix

Data Cleaning

Step 1

- We selected 23 attributes based on our business understanding
- Replaced null value with 0 based on data understanding
- Changed the data type of some variables to category

```
# Selecting 23 attributes
df = df.loc[:,["Department", "GEO","Role","Will_Relocate","Critical","Percent_Remote","last_evaluation","number_project",
                    "average_montly_hours","time_spend_company","Work_accident","promotion_last_5years","salary","Gender",
                    "LinkedIn_Hits","Emp_Identity","Emp_Role","Emp_Position","Emp_Title","Sensor_StepCount",
                    "Sensor_Heartbeat(Average/Min)","Sensor_Proximity(1-highest/10-lowest)","left_Company"]]

# Missing Values
df['Critical'] = df['Critical'].replace(np.nan,0)

# Variable Type Cleaning

for column in ['left_Company', 'Department', 'GEO', 'Will_Relocate', 'Critical', 'promotion_last_5years', 'Gender',
               'Emp_Identity', 'Emp_Role', 'Emp_Position', 'Emp_Title']:
    df[column] = df[column].astype('category')
```

Data Cleaning

Step 2

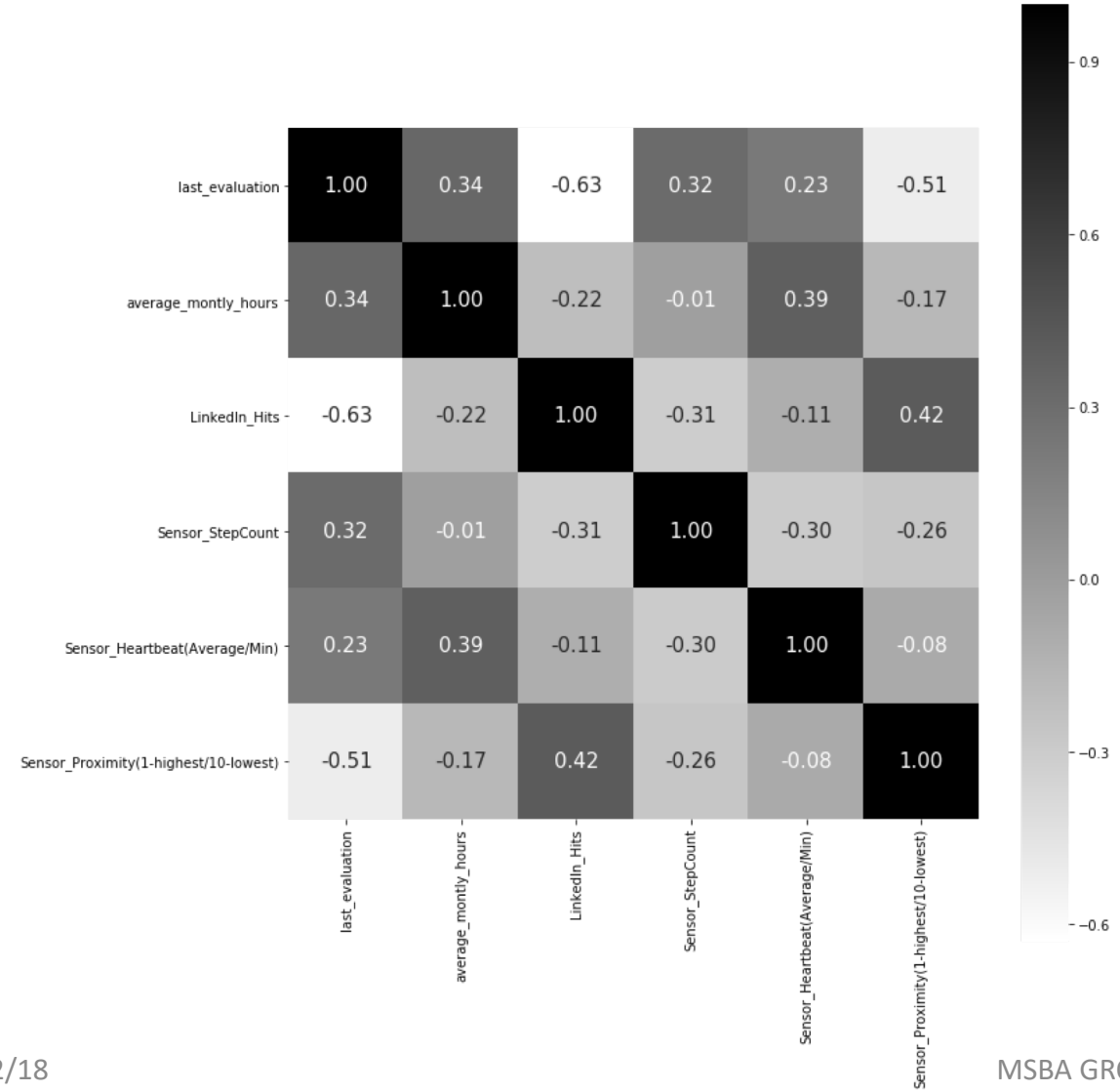
- We transferred some variables into binary, dummy and ordinal variables
- Finally we got 33 x variables

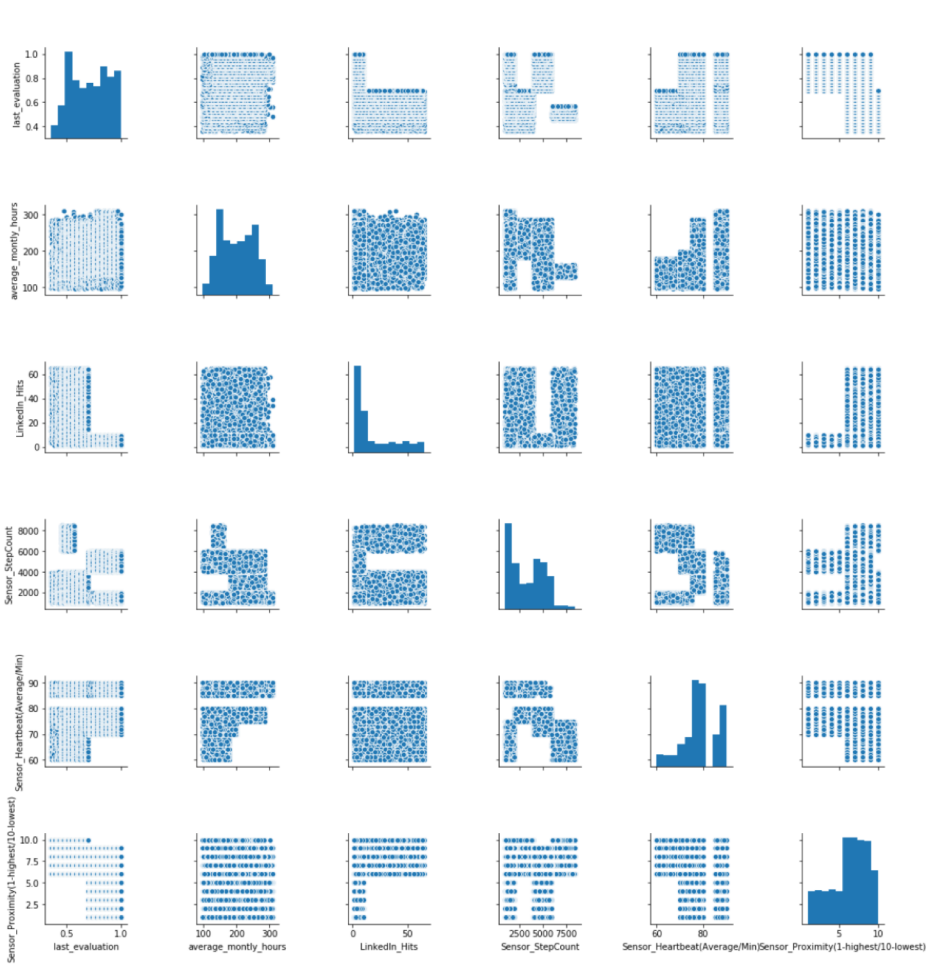
```
# Ordinal Conversion
salary_mapper = {'low':0, 'medium':1, 'high':2}
role_mapper = {'Level 1':0, 'Level 2-4':1, 'Manager':2, 'Senior Manager':3, 'Director':4, 'Senior Director':5, 'VP':6}

df.loc[:,'salary'] = df.loc[:,'salary'].replace(salary_mapper)
df['Role'] = df['Role'].replace(role_mapper)



# Binary Conversion with Label Encoder
df['Gender'] = le.fit_transform(df['Gender'])
df['Department'] = le.fit_transform(df['Department'])  # added
df['GEO'] = le.fit_transform(df['GEO'])  # added

# Categorical Conversion with OneHotEncoder
labels = ohe.fit_transform(df.iloc[:,:2]).toarray()

header=['Finance', 'HR', 'IT', 'Operations', 'Sales', 'Warehouse',
        'Australia', 'China', 'Colombia', 'France', 'Japan', 'Korea', 'Turkey', 'UK', 'US']

df_ohe = pd.DataFrame(labels, columns = header)
df = df.iloc[:,2:]
df = pd.concat([df_ohe,df],axis=1)


# Exluding Warehouse & Colombia to reduce multicollinearity (n-1)
df.drop(['Warehouse','Colombia'], axis=1, inplace=True)
```

```
Data columns (total 33 columns):
Finance                                14999
HR                                     14999
IT                                     14999
Operations                             14999
Sales                                  14999
Australia                              14999
China                                  14999
France                                 14999
Japan                                  14999
Korea                                  14999
Turkey                                 14999
UK                                     14999
US                                     14999
Role                                   14999
Will_Relocate                          14999
Critical                               14999
Percent_Remote                         14999
last_evaluation                        14999
number_project                         14999
average_montly_hours                   14999
time_spend_company                     14999
Work_accident                          14999
promotion_last_5years                  14999
salary                                 14999
Gender                                 14999
LinkedIn_Hits                          14999
Emp_Identity                           14999
Emp_Role                               14999
Emp_Position                           14999
Emp_Title                              14999
Sensor_StepCount                       14999
Sensor_Heartbeat(Average/Min)          14999
Sensor_Proximity(1-highest/10-lowest)  14999
```

Data Exploration

We found no high correlation among numerical variables





Feature Selection for kNN

Used feature_selection method from Scikitlearn, got 9 important features for kNN


```
# Feature Selection for kNN

etc= ExtraTreesClassifier()
etc = etc.fit(X_train, y_train) # changed clf to etc
#print(etc.feature_importances_)


model = SelectFromModel(etc, prefit=True)
X_new = model.transform(X_train)
X_train_new=pd.DataFrame(X_new)

X_valid_new=model.transform(X_valid)
X_valid_new=pd.DataFrame(X_valid_new)

X_test_new=model.transform(X_test)
X_test_new=pd.DataFrame(X_test_new)
```

	0	1	2	3	4	5	6	7	8
0	0.8	0.94	4.0	224.0	2.0	4.0	3.0	4653.0	78.0
1	0.8	0.88	6.0	263.0	4.0	1.0	2.0	1258.0	90.0
2	0.8	0.85	4.0	160.0	5.0	3.0	3.0	4018.0	76.0
3	0.8	0.93	3.0	271.0	5.0	2.0	3.0	4880.0	79.0
4	0.4	0.56	4.0	159.0	3.0	2.0	1.0	1442.0	90.0

Feature Transformation for Logistic Regression

Used logarithm to transfer 3 features


```
X_train_log['LinkedIn_Hits'] = np.log(X_train_log.loc[:,'LinkedIn_Hits'])
X_train_log['Sensor_StepCount'] = np.log(X_train_log.loc[:,'Sensor_StepCount'])
X_train_log['Sensor_Heartbeat(Average/Min)'] = np.log(X_train_log.loc[:,'Sensor_Heartbeat(Average/Min)'])
X_valid_log['LinkedIn_Hits'] = np.log(X_valid_log.loc[:,'LinkedIn_Hits'])
X_valid_log['Sensor_StepCount'] = np.log(X_valid_log.loc[:,'Sensor_StepCount'])
X_valid_log['Sensor_Heartbeat(Average/Min)'] = np.log(X_valid_log.loc[:,'Sensor_Heartbeat(Average/Min)'])
X_test_log['LinkedIn_Hits'] = np.log(X_test_log.loc[:,'LinkedIn_Hits'])
X_test_log['Sensor_StepCount'] = np.log(X_test_log.loc[:,'Sensor_StepCount'])
X_test_log['Sensor_Heartbeat(Average/Min)'] = np.log(X_test_log.loc[:,'Sensor_Heartbeat(Average/Min)'])
```

	LinkedIn_Hits	Sensor_StepCount	Sensor_Heartbeat(Average/Min)
13244	2.302585	8.445267	4.356709
11504	0.000000	7.137278	4.499810
10754	2.302585	8.298540	4.330733
12954	1.609438	8.492900	4.369448
3753	3.526361	7.273786	4.499810

Cut-off values and corresponding TPR values

We printed the FPR, TPR and cut-off value to choose the cut-off


```
FPRS of decision tree:
[0.00000000e+00 4.37445319e-04 7.43657043e-03 1.09361330e-02
 1.18110236e-02 1.79352581e-02 1.92913386e-01 2.05599300e-01
 1.00000000e+00]
TPRS of decision tree:
[0.         0.68207283 0.94957983 0.96218487 0.9719888  0.98459384
 1.         1.         1.        ]
ThresholdS of decision tree:
[2.         1.         0.93554328 0.59259259 0.58181818 0.55882353
 0.02037351 0.01351351 0.        ]

FPRS of kNN:
[0.         0.00393701 0.00393701 0.00437445 0.00437445 0.00524934
 0.00524934 0.00568679 0.00568679 0.00612423 0.00612423 0.00656168
 0.00656168 0.00743657 0.00743657 0.00787402 0.00787402 0.00918635
 0.00918635 0.01006124 0.01006124 0.01049869 0.01049869 0.01312336
 0.01312336 0.01487314 0.01487314 0.02055993 0.02055993 0.02974628
 1.        ]
TPRS of kNN:
[0.         0.93697479 0.94677871 0.94677871 0.94957983 0.94957983
 0.95798319 0.95798319 0.96358543 0.96358543 0.96918768 0.96918768
 0.97058824 0.97058824 0.9719888  0.9719888  0.97619048 0.97619048
 0.97759104 0.97759104 0.9789916  0.9789916  0.98039216 0.98039216
 0.98179272 0.98179272 0.98459384 0.98459384 0.9859944  0.9859944
 1.        ]
ThresholdS of kNN:
[2.         1.         0.79242753 0.77602191 0.76920584 0.76059119
 0.74857722 0.74675473 0.73333791 0.72062474 0.69794658 0.69540436
 0.69437123 0.64905469 0.59929219 0.59058483 0.53138595 0.4934344
 0.46568275 0.45952403 0.4416252  0.42015912 0.38851079 0.30017983
 0.29526803 0.26088629 0.2498412  0.22112413 0.22100241 0.11292119
 0.        ]
```

Cut-off application

We predicted the class based on the cut-off value we set, and built the confusion matrix


```
cutoff = thresholds[5] # choose the best cutoff from the list of thresholds
bestclf = dt2 # use the name of the best clasifier

y_pred_best = bestclf.predict_proba(X_test)[:, 1] # probability estimation
probdf = pd.DataFrame({"y_test": y_test,"prob":y_pred_best})
probdf = probdf.reset_index(drop=True) # reindex
probdf = probdf.sort_values('prob',ascending = 0) # sort the dataframe by probability descending
length = len(probdf[probdf.prob > cutoff]) # number of predicted positives

probdf["y_pred"]=1
probdf.loc[probdf["prob"] <= cutoff,"y_pred"]=0


num_tp = sum(probdf.y_test.iloc[:length])
num_fp = length - num_tp
num_fn = len(probdf[probdf.y_test == 1]) - num_tp
num_tn = len(probdf[probdf.y_test == 0]) - num_fp

# create an cnf matrix array
best_cnf_matrix = np.array([[num_tn,num_fp],[num_fn,num_tp]])
np.set_printoptions(precision=4)
```

Copy of our code

https://drive.google.com/open?id=1ScQmYyc2BHe84xStCl2qIQZuqLyFyIas