UT Data Analysis + Visualization Bootcamp: Final Presentation

Team 2: Claire, Megan, Sahih, Molly

September 2022

Table of Contents

Project Overview

Topic Overview

Topic Selected: Predicting employee attrition in the healthcare industry

The Data

Source data: Synthetic data based on the IBM Watson dataset for attrition, but tailored to the healthcare industry.

Questions we hope to answer with the data:

1. Which factors most heavily influence a person's likelihood of attritting?

2. What is the typical profile of a healthcare worker who is likely to attrit?

Technologies, languages, tools, + algorithms

Technologies, languages, tools, + algorithms employed throughout the project:

- **Technologies:** PostgreSQL, Quick Database Diagrams
- **Languages**: SQL, Python
- **Tools:** PgAdmin, Tableau, Jupyter Notebook
- **Algorithms:** Logistic Regression, Random Forest

Data Analysis

Overview - Exploring the Data

Used Jupyter Notebook + Pandas to show:

- DataFrame shape + structure
- Missing Values
- Data Types
- Integer Data
  - Descriptive summaries for numerical columns
- String Data
  - Possible responses for each column of object data

Machine Learning

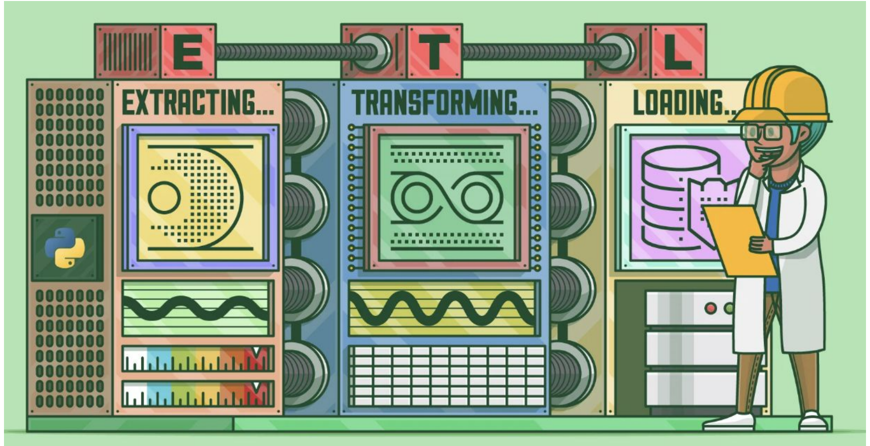
Overview

Model Choices	Selection	Explanation
Supervised (labels) vs Unsupervised (clustering):	Supervised (Logistic Regression + Random Forest)	Data already has labels
Classification (predict discrete outcomes) vs Regression (predict continuous variables):	Classification (binary)	Used to determine discrete outcomes (attriting or not attriting)
Features vs. Target:	Features = all columns except the "Attrition" column Target = "Attrition" column	Features = variables used to make a prediction Target = predicted outcome

Data Preprocessing + Engineering

Preliminary data preprocessing + engineering:

1. Review data distribution + types
2. Check for null values
3. Drop unnecessary columns
4. Binary encode variable columns
5. LabelEncoder on target column
6. Scale the data



Splitting the Data

```
In [6]: # Split our data into Training and Testing
        from sklearn.model_selection import train_test_split

        X_train, X_test, y_train, y_test = train_test_split(X,
                                                            y,
                                                            random_state=1,
                                                            stratify=y)
        X_train.shape

Out[6]: (1257, 34)
```

Model (Draft 2) Results

Description of current accuracy score:

- Improvement over LR model (88% accuracy) by using a RFC model (92% accuracy)

LR Model had decent accuracy (88%) but did not successfully predict any attrition

	precision	recall	f1-score	support
0	0.88	1.00	0.94	369
1	0.00	0.00	0.00	50
accuracy			0.88	419
macro avg	0.44	0.50	0.47	419
weighted avg	0.78	0.88	0.82	419

RFC Model improved accuracy + a 46% recall rate

Confusion Matrix

	Predicted 0	Predicted 1
Actual 0	364	5
Actual 1	27	23

Accuracy Score : 0.9236276849642004

Classification Report

	precision	recall	f1-score	support
0	0.93	0.99	0.96	369
1	0.82	0.46	0.59	50
accuracy			0.92	419
macro avg	0.88	0.72	0.77	419
weighted avg	0.92	0.92	0.91	419

Model (Draft 3) Results

Description of latest accuracy score:

- Improvement in LR model (91% accuracy) + RFC model (92% accuracy) from Draft 2 results

LR Model accuracy increased w/ oversampling and increase in # of estimators (100>1000)

	precision	recall	f1-score	support
0	0.99	0.91	0.95	369
1	0.58	0.92	0.71	50
accuracy			0.91	419
macro avg	0.79	0.92	0.83	419
weighted avg	0.94	0.91	0.92	419

RFC Model improved accuracy + recall rate

	Predicted 0	Predicted 1
Actual 0	358	11
Actual 1	21	29

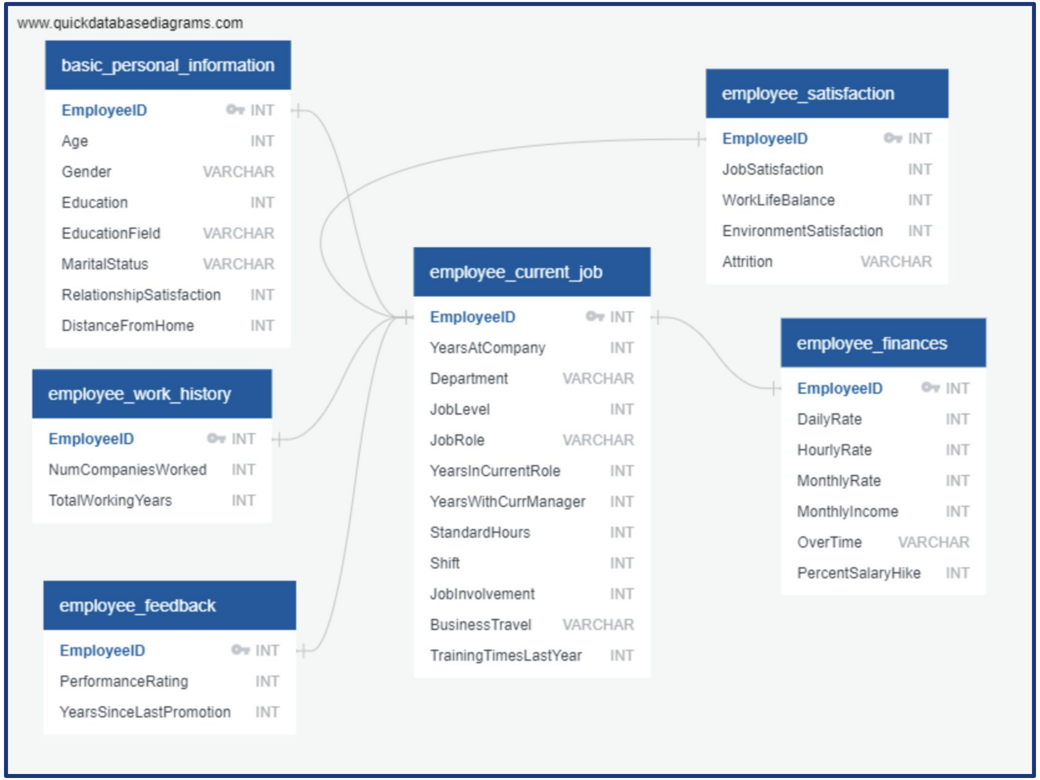
Accuracy Score : 0.9236276849642004

Classification Report

	precision	recall	f1-score	support
0	0.94	0.97	0.96	369
1	0.72	0.58	0.64	50
accuracy			0.92	419
macro avg	0.83	0.78	0.80	419
weighted avg	0.92	0.92	0.92	419

Database

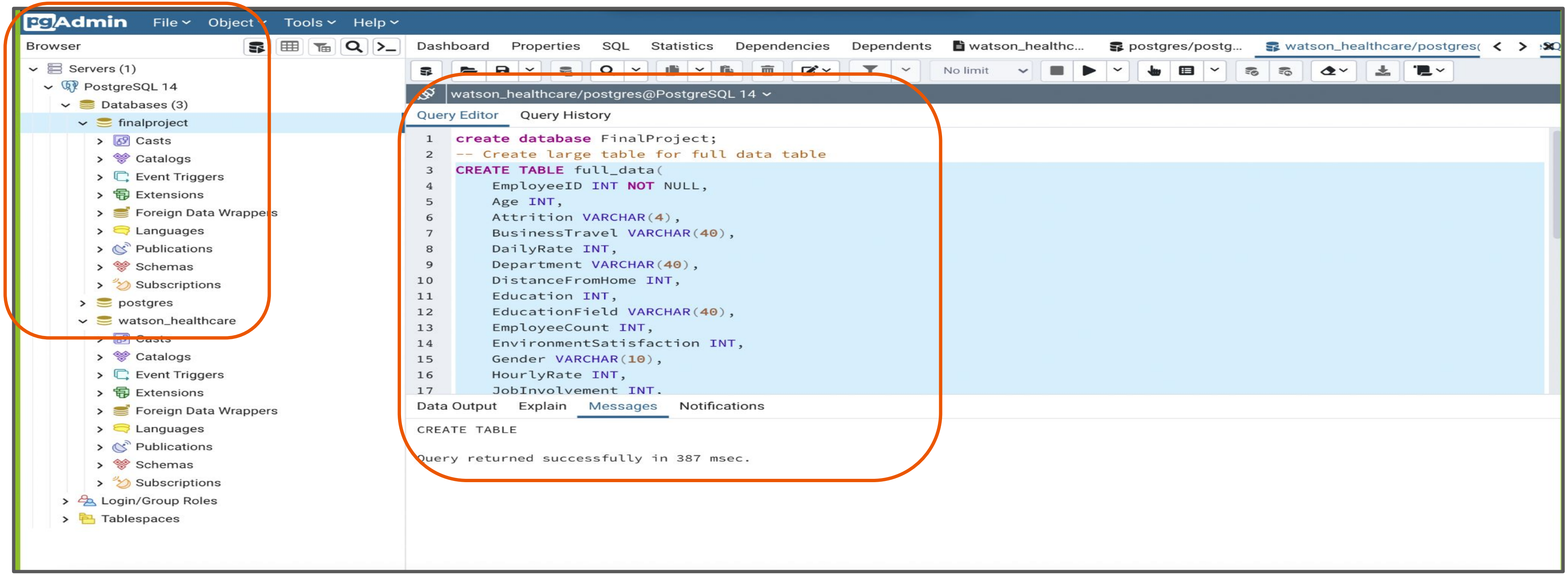
ERD Visual



www.quickdatabasediagrams.com
basic_personal_information
EmployeeID INT
Age INT
Gender VARCHAR
Education INT
EducationField VARCHAR
MaritalStatus VARCHAR
RelationshipSatisfaction INT
DistanceFromHome INT
employee_work_history
EmployeeID INT
NumCompaniesWorked INT
TotalWorkingYears INT
employee_feedback
EmployeeID INT
PerformanceRating INT
YearsSinceLastPromotion INT
employee_current_job
EmployeeID INT
YearsAtCompany INT
Department VARCHAR
JobLevel INT
JobRole VARCHAR
YearsInCurrentRole INT
YearsWithCurrManager INT
StandardHours INT
Shift INT
JobInvolvement INT
BusinessTravel VARCHAR
TrainingTimesLastYear INT
employee_satisfaction
EmployeeID INT
JobSatisfaction INT
WorkLifeBalance INT
EnvironmentSatisfaction INT
Attrition VARCHAR
employee_finances
EmployeeID INT
DailyRate INT
HourlyRate INT
MonthlyRate INT
MonthlyIncome INT
OverTime VARCHAR
PercentSalaryHike INT

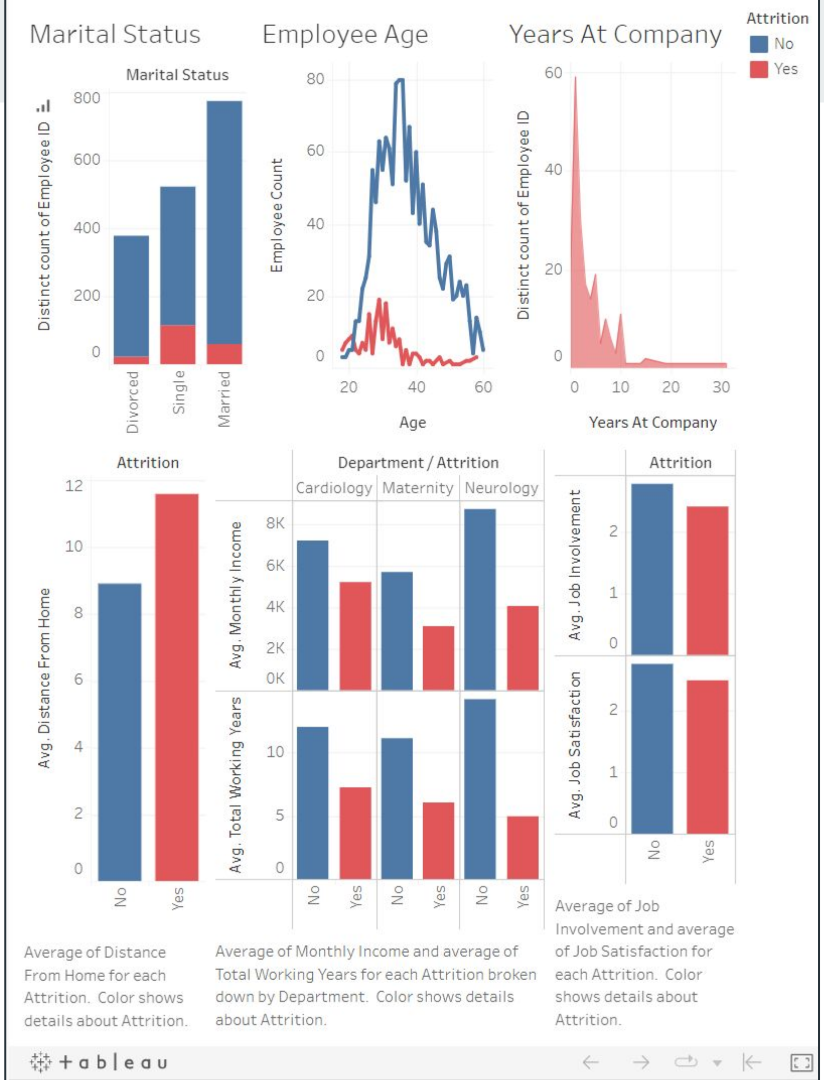
Database Visual:

PostgreSQL DB Creation in PgAdmin



Dashboard

Storyboard: Attrition in Health Care



Project Conclusion

Results, Recommendations, + Learnings

- **Analysis Results:** Age, if someone works overtime or not, total working years and monthly income were the variables that contributed most to the model
- **Recommendations for future analysis:**
  - We could bucket out the monthly income into chunks of 500-1000, drop additional unnecessary columns, try undersampling our variables, etc., in order to increase the model's accuracy score
- **What we would have done differently:**
  - Explore other employee/attrition data sets and test them against the model
  - Conduct more statistical analysis on the data in order to further understand the distribution + the most relevant/dependable variables for the ML model

Questions?

Appendix

Data Analysis

Exploring the Data

DataFrame Structure:

Data divided into basic personal information, employee work history, employee current job, employee finances, employee satisfaction, and employee feedback

Shape

```
The data set has 1676 rows and 35 columns
```

Column names & null values

```
EmployeeID                  0
Age                         0
Attrition                   0
BusinessTravel              0
DailyRate                   0
Department                  0
DistanceFromHome            0
Education                   0
EducationField              0
EmployeeCount               0
EnvironmentSatisfaction     0
Gender                      0
HourlyRate                  0
JobInvolvement              0
JobLevel                    0
JobRole                     0
JobSatisfaction             0
MaritalStatus               0
MonthlyIncome               0
```

```
MonthlyRate                 0
NumCompaniesWorked          0
Over18                      0
OverTime                    0
PercentSalaryHike           0
PerformanceRating           0
RelationshipSatisfaction    0
StandardHours               0
Shift                       0
TotalWorkingYears           0
TrainingTimesLastYear       0
WorkLifeBalance             0
YearsAtCompany              0
YearsInCurrentRole          0
YearsSinceLastPromotion     0
YearsWithCurrManager        0
```

Exploring the Data

Data Types:

Primarily int, some strings

Data Type Counts

```
int64      26
object      9
```

Integer Columns

```
Index(['EmployeeID', 'Age', 'DailyRate', 'DistanceFromHome', 'Education',
       'EmployeeCount', 'EnvironmentSatisfaction', 'HourlyRate',
       'JobInvolvement', 'JobLevel', 'JobSatisfaction', 'MonthlyIncome',
       'MonthlyRate', 'NumCompaniesWorked', 'PercentSalaryHike',
       'PerformanceRating', 'RelationshipSatisfaction', 'StandardHours',
       'Shift', 'TotalWorkingYears', 'TrainingTimesLastYear',
       'WorkLifeBalance', 'YearsAtCompany', 'YearsInCurrentRole',
       'YearsSinceLastPromotion', 'YearsWithCurrManager'],
      dtype='object')
```

Object Columns

```
Index(['Attrition', 'BusinessTravel', 'Department', 'EducationField', 'Gender',
       'JobRole', 'MaritalStatus', 'Over18', 'OverTime'],
      dtype='object')
```

Exploring the Data

Integer Data

Descriptive data for the numerical columns

Takeaways:

- EmployeeCount = 1 for all values
- StandardHours = 80 for all values
- DailyRate, HourlyRate, MonthlyIncome, and MonthlyRate all contain equivalent data

Example:

	Age	DistanceFromHome
count	1676.000000	1676.000000
mean	36.866348	9.221957
std	9.129126	8.158118
min	18.000000	1.000000
25%	30.000000	2.000000
50%	36.000000	7.000000
75%	43.000000	14.000000
max	60.000000	29.000000

Exploring the Data

Data Scales:

- Some of the columns = numerical keys used to classify categorical data
- Mean + standard deviation help identify the most common responses + the spread of responses

	Education	EnvironmentSatisfaction	JobInvolvement	JobSatisfaction	PerformanceRating	RelationshipSatisfaction	WorkLifeBalance
count	1676.000000	1676.000000	1676.000000	1676.000000	1676.000000	1676.000000	1676.000000
mean	2.907518	2.714797	2.724940	2.738663	3.150358	2.718377	2.766110
std	1.025835	1.097534	0.714121	1.104005	0.357529	1.078162	0.702369
min	1.000000	1.000000	1.000000	1.000000	3.000000	1.000000	1.000000
25%	2.000000	2.000000	2.000000	2.000000	3.000000	2.000000	2.000000
50%	3.000000	3.000000	3.000000	3.000000	3.000000	3.000000	3.000000
75%	4.000000	4.000000	3.000000	4.000000	3.000000	4.000000	3.000000
max	5.000000	4.000000	4.000000	4.000000	4.000000	4.000000	4.000000

Education	EnvironmentSatisfaction	JobInvolvement	JobSatisfaction	PerformanceRating	RelationshipSatisfaction	WorkLifeBalance
1 'Below College'	1 'Low'	1 'Low'	1 'Low'	1 'Low'	1 'Low'	1 'Bad'
2 'College'	2 'Medium'	2 'Medium'	2 'Medium'	2 'Good'	2 'Medium'	2 'Good'
3 'Bachelor'	3 'High'	3 'High'	3 'High'	3 'Excellent'	3 'High'	3 'Better'
4 'Master'	4 'Very High'	4 'Very High'	4 'Very High'	4 'Outstanding'	4 'Very High'	4 'Best'
5 'Doctor'						

Exploring the Data

Object Type:

Columns with object data types had at most 6 different possible responses; still fairly limited response options

```
raw_data_df["Attrition"].value_counts()
```

```
No     1477
Yes     199
Name: Attrition, dtype: int64
```

```
raw_data_df["BusinessTravel"].value_counts()
```

```
Travel_Rarely        1184
Travel_Frequently     320
Non-Travel            172
Name: BusinessTravel, dtype: int64
```

```
raw_data_df["Department"].value_counts()
```

```
Maternity     796
Cardiology    531
Neurology     349
Name: Department, dtype: int64
```

```
raw_data_df["EducationField"].value_counts()
```

```
Life Sciences       697
Medical             524
Marketing           189
Technical Degree    149
Other                88
Human Resources      29
Name: EducationField, dtype: int64
```

```
raw_data_df["Gender"].value_counts()
```

```
Male      998
Female    678
Name: Gender, dtype: int64
```

```
raw_data_df["JobRole"].value_counts()
```

```
Nurse             822
Other             534
Therapist         189
Administrative    115
Admin              16
Name: JobRole, dtype: int64
```

```
raw_data_df["MaritalStatus"].value_counts()
```

```
Married     777
Single      522
Divorced    377
Name: MaritalStatus, dtype: int64
```

```
raw_data_df["Over18"].value_counts()
```

```
Y    1676
Name: Over18, dtype: int64
```

```
raw_data_df["OverTime"].value_counts()
```

```
No     1200
Yes     476
Name: OverTime, dtype: int64
```

Machine Learning

Code, v1

- The Code for Section 1 lives in the “ML_Model_Draft1.ipynb” Jupyter Notebook file
  - Located on our repo: https://github.com/meganmcmahon/Final_Project_Team2
- The current code:
  - Imports the dependencies
  - Connects to a Postgres SQL database
  - Loads the data as a dataframe into JN
  - Defines our feature + target variables
  - Splits the data into training + testing variables
  - Creates + trains our model
  - Prints our prediction results in a decision matrix

Code, v2

- **The Code for Section 2 lives in the "ML_Model_Draft2.ipynb" Jupyter Notebook file**
  - Located on our repo: https://github.com/meganmcmahon/Final_Project_Team2
- **The second batch of code does everything from v1 and also:**
  - Performs initial data analysis
  - Performs data preprocessing to support the ML model
  - Performs feature engineering
  - Performs feature selection
  - Adds a new model (RandomForestClassification), which outperforms the Logistic Regression model

Limitations + Benefits of Logistic Regression

Explanation of model choice: We chose a supervised logistic regression model because our data already has labels (column names) and we are simply looking to perform a binary classification in order to help us predict if a healthcare employee will attrit or not.

Limitations:

1. Only able to predict discrete functions
2. Not as capable of determining complex relationships as Neural Networks

Benefits:

1. Relatively easy/efficient to implement, interpret + train
2. Not as prone to overfitting as other types of models

Data Preprocessing

Description of preliminary data preprocessing after loading in the data:

1. Ran df.describe() to review the distribution of the data in each column
2. Ran df.isnull().sum() to double check for any null values
3. Ran df.dtypes to see what types of data are in each column
4. Ran df.drop() to drop any unnecessary columns: [EmployeeID, Over18, EmployeeCount, StandardHours, DailyRate, HourlyRate, MonthlyRate]

Feature Engineering

Description of preliminary feature engineering:

1. Binary encoding using Pandas
    a. Columns = ["BusinessTravel", "Department"," EducationField", "Gender", "JobRole", "MaritalStatus", "OverTime"
2. From sklearn.preprocessing import LabelEncoder
    a. Encode the “Attrition” column
3. Scale continuous data using StandardScaler

Feature Selection

Description of preliminary feature selection, including the decision-making process:

After preprocessing the data by dropping unnecessary columns and completing feature engineering, we narrowed down the features to use for our ML model.

To refine our feature selection we will:

1. Run the model
2. Check the accuracy score
3. Rank the feature importance
4. Remove the least important features to improve the efficiency of the model

Model Development

Explanation of changes in model choice between Segment 2 + 3 deliverables:

- Moved from using a Logistic Regression model to a RandomForestClassifier model
  - Based on results outlined on slide 23

Description of how the model has been trained so far + any additional training that will occur:

- Achieved first round results after initial data preprocessing, feature engineering +selection outlined on slides 18-20
- Next, in ***ML_Model_Draft3***, we will:
  - Update the Logistic Regression model by increasing the # of estimators (100 >1000)
  - Use Oversampling on our X_train and y_train variables

Model Results cont.

How the model addresses the problem we're trying to solve:

- The model does an good job of predicting that someone will not attrit, but it is not as accurate at predicting with precision that someone will attrit.
    - The stakes are not as high as with needing precision for predicting, for example, if someone has cancer or not.
    - We still have options to fine tune the data more in order to achieve an even higher accuracy score:
        - We could bucket out the monthly income into chunks of 500-1000, drop additional unnecessary columns, try undersampling our variables, etc.

Database

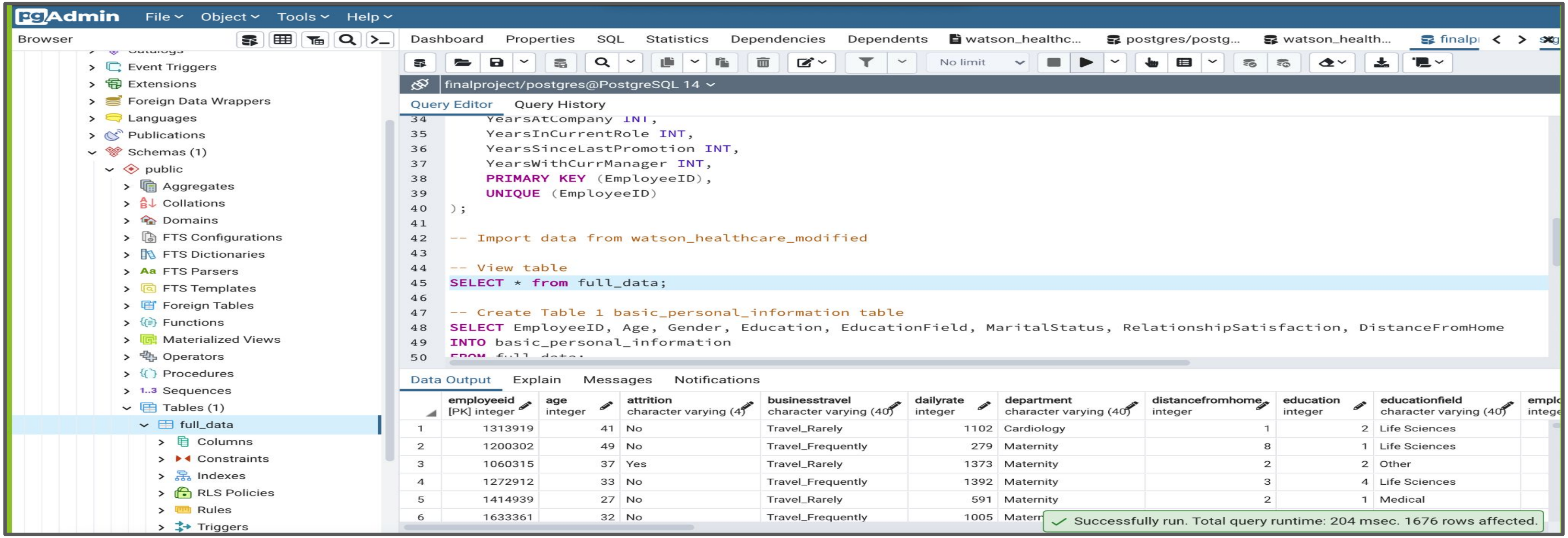
Database Requirements

We've created a fully integrated PostgreSQL database + have met the following requirements:

- ❏ *Database stores static data for use during the project:* full_data table
- ❏ *Database interfaces with the project:* data is separated + joined into tables and then connected to the ML model with a SQLAlchemy connection string
- ❏ *Database includes at least two tables:* basic_personal_information, employee_current_job, employee_feedback, employee_finances, employee_satisfaction, employee_work_history, full_data
- ❏ *Database includes at least one join using SQL: See* Database splits joins .sql file
- ❏ *Database includes at least one connection string: using SQLAlchemy:* ML_Model_Draft2.ipynb file
- ❏ *Presentation includes an ERD with relationships:* see QuickDBD-export.png file

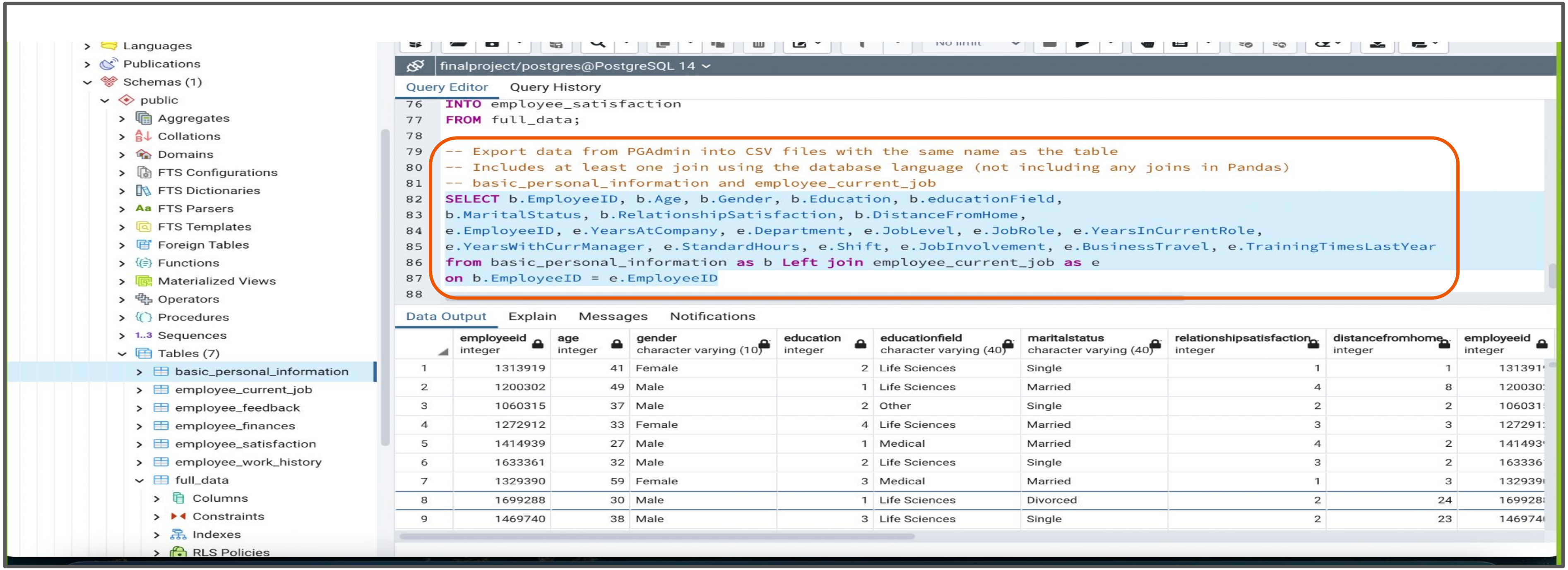
Database Visual:

Create main table with all data



Database Visual:

Create 1 join using SQL to combine 2 tables



Dashboard

Dashboard Outline

Storyboard on a Google Slide(s):

The storyboard, displayed in the next slide, outlines our Tableau dashboard, which will visualize the answers to 1. who is most likely to attrit, 2. when are they most likely to attrit, and 3. why they are most likely to attrit.

Description of the tool(s) that will be used to create the final dashboard:

The final dashboard will be created using multiple graph types in Tableau. These graphs will help visualize which factors most heavily influence a person's likelihood of attritting, and the typical profile of a healthcare worker who is likely to attrit.

Description of interactive element(s):

We will incorporate interactive elements into the graphs by toggling between male and female filters to show the difference in attrition data by gender.

Dashboard (30 pts)

The dashboard presents a data story that is logical and easy to follow for someone unfamiliar with the topic. It includes the following:

Images from the initial analysis: See slides 9-13

Data (images or report) from the machine learning task: See slides 22, 24-25

At least one interactive element: See Tableau Public Link: https://public.tableau.com/app/profile/megan.mcmahon/viz/Final_Project_Dashboard_16626658098950/ProjectDashboard?publish=yes