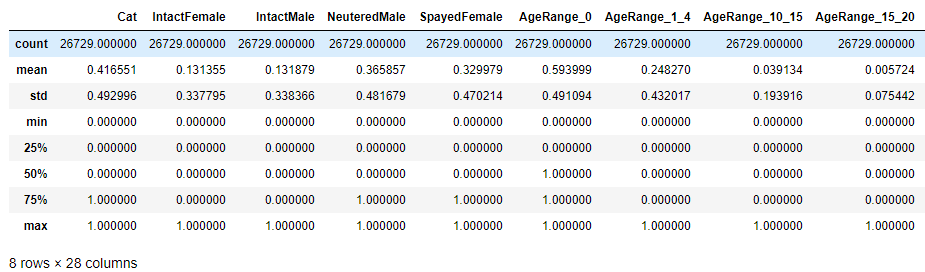
# ITSM Semester Project Summary

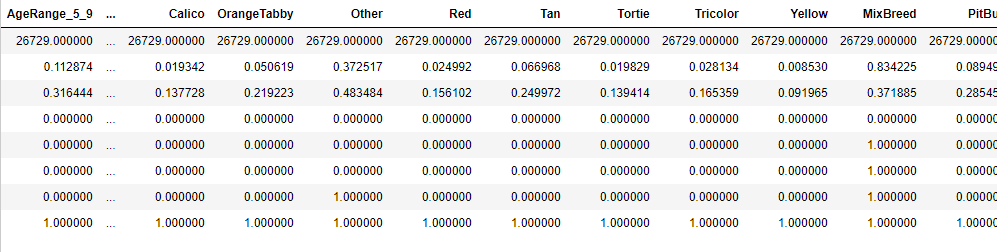
## Jupyter Notebook Steps

1. Set initial global constants
2. Import data science and other libraries
3. Import clean and transformed data from a CSV
4. Estimate predictor coefficients
5. Variable selection with Lasso; drop insignificant variables from data model
6. Run Linear Discriminant Analysis (LDA) with 5-fold cross validation and best probability threshold according to Balanced Accuracy formula
   1. Output ROC Curve (5) with mean TPR and FPR
   2. Output Confusion Matrix from mean TPR and FPR
7. Run Logistic Regression with 5-fold cross validation and best probability threshold according to Balanced Accuracy formula
   1. Output ROC Curve (5) with mean TPR and FPR
   2. Output Confusion Matrix from mean TPR and FPR
8. Compare models, choosing best one.

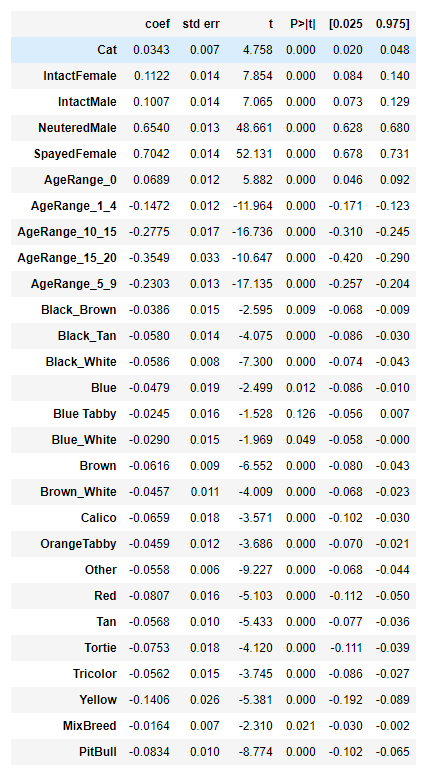
## Model Summary



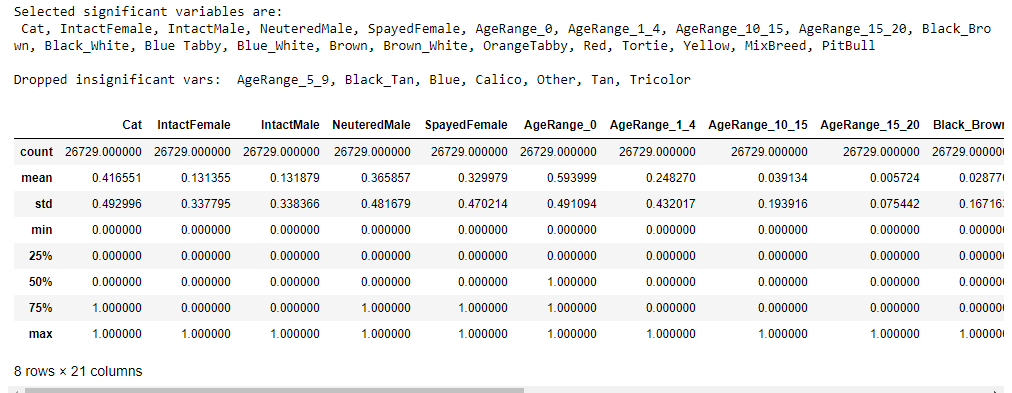
Cont’d



## OLS Parameter Estimates



## Lasso Variable Selection



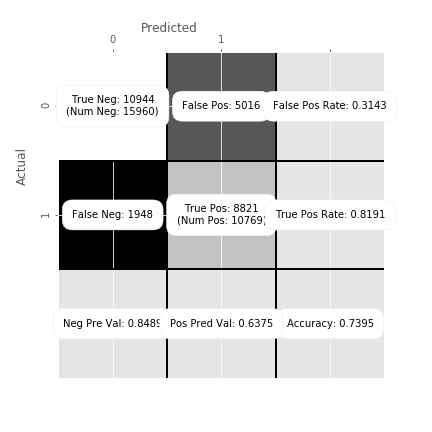
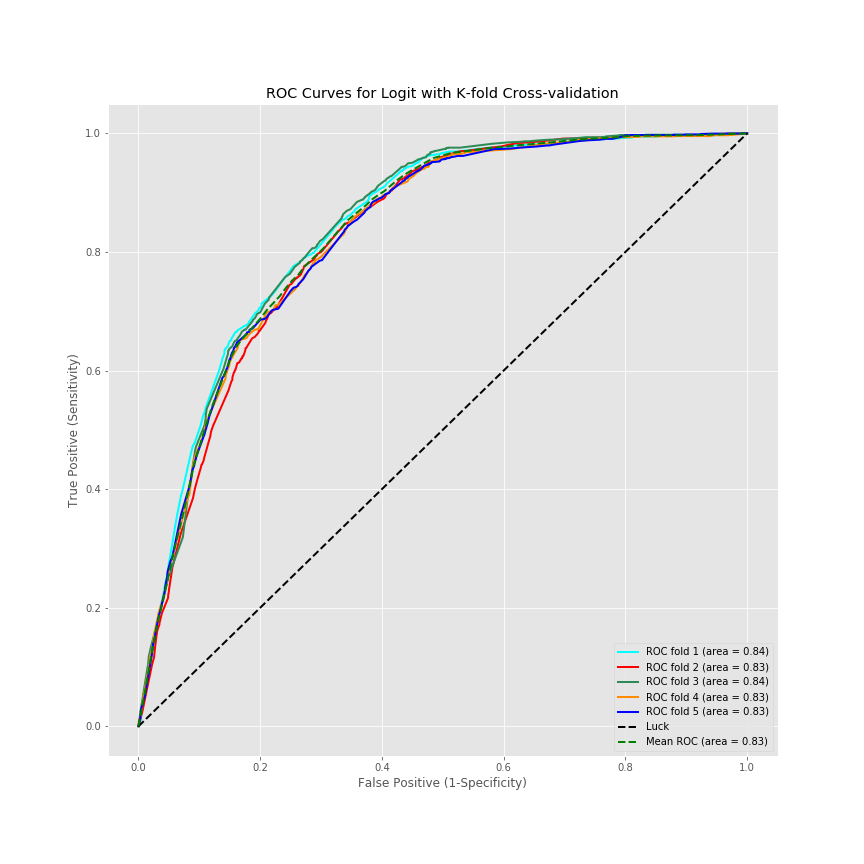
## LDA Output

## C:\Users\bonfardeci-j\AppData\Local\Microsoft\Windows\INetCache\Content.Word\lda-roc.pngC:\Users\bonfardeci-j\AppData\Local\Microsoft\Windows\INetCache\Content.Word\lda-best-tp-cm.png

LDA TPR = 81.65%

LDA FPR = 68.77%

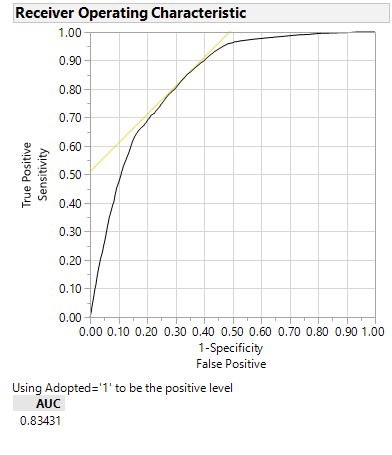
## Logit Output

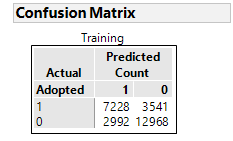


Logit TPR = 81.91%

Logit FPR = 68.57%

## JMP Model





JMP TPR = 7228 / (7228 + 3541) = 7228 / 10769 = 67.12%

JPM FPR = 12968 / (12968 + 2992) = 12968 / 15960 = 81.25%

## Comparison of Models

Our logit model outperformed our LDA model in Python as well as the JMP logit model. Since our goal was to predict adoptions where 1 = success, the logit true-positive rate was the highest out of all the models – 81.91% vs. 81.65% (LDA) and 67.12% (JMP).

Although, the JMP FPR was higher than the other 2 models: 81.25% vs. 68.77% and 67.12%. While one could also predict non-adoptions at a reasonable rate with the JMP model, the TPR for the Python models are still higher than the FPR rate obtained with the JMP model.

The advantages with the Python models is that we could utilize 5-fold cross validation to split the training data into 80/20 train/validation sets to obtain a more realistic mean TPR and FPR, and choose the optimal cutoff with custom Python functions that automatically select the most optimal probability threshold/cutoff.

While it’s much less work and more intuitive to build a model in JMP, overall one has more programmatic control utilizing a language such as Python with good data science libraries.