**IN498 – Unit 3 Assignment – Laurence Burden**

**Question 8**

**Compare Results**

The logistic regression results for the initial data set outputs the probability of 36 different classes. This is due to the wide variety of numbers in the Installers\_retained\_for\_30\_days column of the data frame. The second logistic regression test becomes much more readable by narrowing the classes down to two, 0 or 1. This lets us have a better insight into whether we will keep a user based on the number of installs.

**Intercept Value Meaning**

The intercept values represent the Y-intercept. This is the value of y when X = 0. The linear regression model has a y value of 0.0473 when X = 0. The logistic regression model has a y value of -2.866 when X = 0.

**Coefficient Value Meaning**

The coefficient value represents the number that is multiplied by a given X value for a regression line. The output of the linear regression in this code gives a coefficient of 0.324. Adding the y-intercept will give the following equation: y = 0.324X + 0.0473. This allows us to predict the value of y with a given value of X.

**Results Summary**

The results of both these models tells us that the probability of holding onto at least one customer rises with the number of installs retained for 30 days. The number of installs retained for 30 days is around 3. 2 installs kept for 30 days only has a 43% of being retained, whereas 4 installs has a 91% chance of retention.

**Code Output**

### First 10 Rows ###

Date ... Installer-to-30\_days\_retention\_rate

0 4/1/19 ... 0.0

1 4/1/19 ... 0.5

2 4/1/19 ... 0.0

3 4/1/19 ... 0.0

4 4/1/19 ... 0.5

5 4/1/19 ... 1.0

6 4/1/19 ... 0.0

7 4/1/19 ... 0.0

8 4/1/19 ... 1.0

9 4/1/19 ... 0.0

[10 rows x 14 columns]

### Data Shape ###

(6409, 14)

### Data Description ###

Store\_Listing\_Visitors ... Installer-to-30\_days\_retention\_rate

count 6401.000000 ... 6393.000000

mean 8.562100 ... 0.122888

std 31.625562 ... 0.298208

min 0.000000 ... 0.000000

25% 1.000000 ... 0.000000

50% 1.000000 ... 0.000000

75% 2.000000 ... 0.000000

max 354.000000 ... 1.000000

[8 rows x 11 columns]

### Shape of X ###

(6409, 1)

### Description of X ###

<bound method NDFrame.describe of Installers

0 1.0

1 2.0

2 0.0

3 0.0

4 2.0

... ...

6404 1.0

6405 0.0

6406 0.0

6407 34.0

6408 1.0

[6409 rows x 1 columns]>

### Shape of y ###

(6409,)

### Description of y ###

count 6409.000000

mean 0.855360

std 3.521889

min 0.000000

25% 0.000000

50% 0.000000

75% 0.000000

max 36.000000

Name: Installers\_retained\_for\_30\_days, dtype: float64

### Prediction for number retained for 30 days with 1 install ###

[0.37117343]

### Prediction for number retained for 30 days with 2 installs ###

[0.69505919]

### Prediction for number retained for 30 days with 3 installs ###

[1.34283072]

### Y Intercept ###

0.047287665244239996

### Coefficient ###

[0.32388576]

### Logistic Regression Predictions ###

[ 0. 0. 0. ... 0. 17. 0.]

### Log Regression Probabilities ###

[1.61139923e-01 3.53521465e-01 5.91248676e-02 ... 5.91248676e-02

3.34563725e-07 1.61139923e-01]

### Prediction for number retained for 30 days with 1 install ###

[[8.16736819e-01 1.61139923e-01 1.79506012e-02 1.29566322e-03

1.41319960e-04 1.17967771e-04 1.09930274e-04 1.09501369e-04

1.18168706e-04 1.12502658e-04 1.35243577e-04 1.05697553e-04

1.11571961e-04 1.28373602e-04 1.00462332e-04 1.00477613e-04

9.93180491e-05 1.11977312e-04 7.42719859e-05 9.32773437e-05

8.48154274e-05 7.58948032e-05 8.28961863e-05 8.04663816e-05

7.94082111e-05 7.79010800e-05 6.39546985e-05 7.66965357e-05

7.44938445e-05 8.11830761e-05 5.64802011e-05 9.15496752e-05

9.09648071e-05 9.02270931e-05]]

### Prediction for number retained for 30 days with 2 installs ###

[[5.74040964e-01 3.53521465e-01 5.66841749e-02 4.65966636e-03

4.77015123e-04 4.38046316e-04 4.22390847e-04 4.18294454e-04

4.46801774e-04 4.34425729e-04 5.30886864e-04 4.08867338e-04

4.35481363e-04 5.04333530e-04 3.89669151e-04 3.97659029e-04

3.91618306e-04 4.42393352e-04 2.92079046e-04 3.64018167e-04

3.35235516e-04 3.01919921e-04 3.25449699e-04 3.16542672e-04

3.10016240e-04 3.01596680e-04 2.51907205e-04 2.96149470e-04

2.88726233e-04 3.11377544e-04 2.21159014e-04 3.45875834e-04

3.45575071e-04 3.48218709e-04]]

### Prediction for number retained for 30 days with 4 installs ###

[[0.10214894 0.61292953 0.203609 0.02170949 0.00195776 0.00217572

0.00224635 0.00219875 0.00230096 0.00233341 0.00294674 0.00220387

0.00238984 0.00280396 0.00211179 0.00224369 0.00219332 0.00248734

0.00162712 0.00199704 0.00188656 0.00172116 0.00180697 0.00176456

0.00170213 0.0016284 0.00140782 0.00159056 0.00156238 0.00165006

0.00122149 0.00177835 0.00179659 0.00186832]]

### Top 10 of Install\_30 Column ###

0 0

1 1

2 0

3 0

4 1

5 1

6 0

7 0

8 1

9 0

Name: Install\_30, dtype: int32

### Install\_30 Shape ###

(6409,)

### Install\_30 Description ###

count 6409.000000

mean 0.173350

std 0.378579

min 0.000000

25% 0.000000

50% 0.000000

75% 0.000000

max 1.000000

Name: Install\_30, dtype: float64

### Top 10 Rows of X ###

Installers

0 1.0

1 2.0

2 0.0

3 0.0

4 2.0

5 1.0

6 0.0

7 0.0

8 1.0

9 0.0

### Shape of X ###

(6409, 1)

### Description of X ###

Installers

count 6409.000000

mean 2.494929

std 10.202951

min 0.000000

25% 0.000000

50% 0.000000

75% 1.000000

max 238.000000

### Top 10 Rows of y ###

0 0

1 1

2 0

3 0

4 1

5 1

6 0

7 0

8 1

9 0

Name: Install\_30, dtype: int32

### Shape of y ###

(6409,)

### Description of y ###

count 6409.000000

mean 0.173350

std 0.378579

min 0.000000

25% 0.000000

50% 0.000000

75% 0.000000

max 1.000000

Name: Install\_30, dtype: float64

### Predictions Based on X ###

[0 0 0 ... 0 1 0]

### Predictions for Class 1 ###

[[0.82848498 0.17151502]]

### Prediction for number retained for 30 days with 1 install ###

[[0.82848498 0.17151502]]

### Prediction for number retained for 30 days with 2 installs ###

[[0.5704965 0.4295035]]

### Prediction for number retained for 30 days with 4 installs ###

[[0.09127014 0.90872986]]

### Log Regression Y Intercept ###

[-2.86597854]

### Log Regression Coefficient ###

[[1.29105069]]

**Python Code**

import pandas as pd  
import numpy as np  
from sklearn.linear\_model import LinearRegression  
from sklearn.linear\_model import LogisticRegression  
import sys  
  
# Ignoring warnings  
if not sys.warnoptions:  
 import warnings  
 warnings.simplefilter("ignore")  
  
# Widen the column display  
pd.set\_option('max\_colwidth',500)  
  
# Read data into a DataFrame using these columns  
# "Date","Package\_Name","Country","Store\_Listing\_Visitors",  
# "Installers","Visitor-to-Installer\_conversion\_rate",  
# "Installers\_retained\_for\_1\_day","Installer-to-1\_day\_retention\_rate",  
# "Installers\_retained\_for\_7\_days","Installer-to-7\_days\_retention\_rate",  
# "Installers\_retained\_for\_15\_days","Installer-to-15\_days\_retention\_rate",  
# "Installers\_retained\_for\_30\_days","Installer-to-30\_days\_retention\_rate"  
col\_names = ["Date", "Package\_Name", "Country", "Store\_Listing\_Visitors", "Installers",  
 "Visitor-to-Installer\_conversion\_rate", "Installers\_retained\_for\_1\_day",  
 "Installer-to-1\_day\_retention\_rate", "Installers\_retained\_for\_7\_days",  
 "Installer-to-7\_days\_retention\_rate", "Installers\_retained\_for\_15\_days",  
 "Installer-to-15\_days\_retention\_rate", "Installers\_retained\_for\_30\_days",  
 "Installer-to-30\_days\_retention\_rate"]  
data = pd.read\_csv('final\_retentions\_parsed.csv', names=col\_names)  
  
########### EXPLORE THE DATA SET #######################  
# Print the top 10 rows  
print('### First 10 Rows ###')  
print(data.head(10), '\n')  
  
# Print the shape  
print('### Data Shape ###')  
print(data.shape, '\n')  
  
# Print the description of the data  
print('### Data Description ###')  
print(data.describe(), '\n')  
  
############ FIX MISSING DATA #######################  
# Replace NaN with 0 for Installers\_retained\_for\_30\_days  
data[np.isnan(data.Installers\_retained\_for\_30\_days)] = 0  
  
########### LINEAR REGRESSION #######################  
# Create a linear regression model  
linreg = LinearRegression()  
  
# Set feature\_cols to Installers column data only  
feature\_cols = ['Installers']  
  
# Set X to feature\_cols  
X = data[feature\_cols]  
  
# Set y to installer retained for 30 days  
#y = data['Installers\_retained\_for\_30\_days']  
y = data.Installers\_retained\_for\_30\_days  
  
# Print the shape of X  
print('### Shape of X ###')  
print(X.shape, '\n')  
  
# Print the description of X  
print('### Description of X ###')  
print(X.describe, '\n')  
  
# Print the shape of y  
print('### Shape of y ###')  
print(y.shape, '\n')  
  
# Print the description of y  
print('### Description of y ###')  
print(y.describe(), '\n')  
  
# Fit the linear regression model with X and y  
linreg.fit(X[:], y[:])  
  
# Get predictions for retained for 30 days with 1 install  
pred\_30\_days\_1\_install = linreg.predict([[1]])  
print('### Prediction for number retained for 30 days with 1 install ###')  
print(pred\_30\_days\_1\_install, '\n')  
  
# Get predictions for retained for 30 days with 2 installs  
pred\_30\_days\_2\_installs = linreg.predict([[2]])  
print('### Prediction for number retained for 30 days with 2 installs ###')  
print(pred\_30\_days\_2\_installs, '\n')  
  
# Get predictions for retained for 30 days with 4 installs  
pred\_30\_days\_4\_installs = linreg.predict([[4]])  
print('### Prediction for number retained for 30 days with 3 installs ###')  
print(pred\_30\_days\_4\_installs, '\n')  
  
# Print the intercept  
print('### Y Intercept ###')  
print(linreg.intercept\_, '\n')  
  
# Print the coefficient  
print('### Coefficient ###')  
print(linreg.coef\_, '\n')  
  
########### LOGISTIC REGRESSION #######################  
# Get logistic regression model  
logreg = LogisticRegression()  
  
# Fit X and y to logistic regression model  
logreg.fit(X, y)  
  
# Predict classes using X  
predictions = logreg.predict(X)  
  
# Print the predictions using X  
print('### Logistic Regression Predictions ###')  
print(predictions)  
  
  
# Get the predicted probabilities of class 1  
assorted\_pred\_prob = logreg.predict\_proba(X)[:, 1]  
  
# Print the probabilities using X  
print('### Log Regression Probabilities ###')  
print(assorted\_pred\_prob, '\n')  
  
# Predict the probablity of maintaining  
# number of users for 30 days (0, 1, 2 users)  
  
# Get probability for retained users for 30 days with 1 install  
print('### Prediction for number retained for 30 days with 1 install ###')  
print(logreg.predict\_proba([[1]]), '\n')  
  
# Get probability for retained users for 30 days with 2 installs  
print('### Prediction for number retained for 30 days with 2 installs ###')  
print(logreg.predict\_proba([[2]]), '\n')  
  
# Get probability for retained users for 30 days with 4 installs  
print('### Prediction for number retained for 30 days with 4 installs ###')  
print(logreg.predict\_proba([[4]]), '\n')  
  
  
#################################### INSTALL\_30 ##########################################  
# Add a new column for installers retained for 30 days. Call it Install\_30.  
# If greater than 0, put 1, if 0, put 0  
data['Install\_30'] = np.where(data['Installers\_retained\_for\_30\_days'] > 0, 1, 0)  
  
# Print the top 10 rows of the new data set  
print('### Top 10 of Install\_30 Column ###')  
print(data['Install\_30'].head(10))  
  
# Print the shape of the new data set  
print('### Install\_30 Shape ###')  
print(data['Install\_30'].shape)  
  
# Print the description of the new data set  
print('### Install\_30 Description ###')  
print(data['Install\_30'].describe())  
  
  
  
########### LOGISTIC REGRESSION #######################  
  
# Perform logistic regression using Install\_30 column for y  
# X = Installers column  
# Set y to numpy array with the right shape  
y = data['Install\_30']  
  
# Fit X and y for logistic regression  
logreg.fit(X, y)  
  
# Print the top 10 rows of X  
print('### Top 10 Rows of X ###')  
print(X.head(10), '\n')  
  
# Print the shape rows of X  
print('### Shape of X ###')  
print(X.shape, '\n')  
  
# Print the description of X  
print('### Description of X ###')  
print(X.describe(), '\n')  
  
# Print the top 10 rows of y  
print('### Top 10 Rows of y ###')  
print(y.head(10), '\n')  
  
# Print the shape of y  
print('### Shape of y ###')  
print(y.shape, '\n')  
  
# Print the description of y  
print('### Description of y ###')  
print(y.describe(), '\n')  
  
# Predict on X and capture the result to assorted\_pred\_class  
assorted\_pred\_class = logreg.predict(X)  
  
# Print the predictions using assorted \_pred\_class (X predictions)  
print('### Predictions Based on X ###')  
print(assorted\_pred\_class, '\n')  
  
# Get the predicted probabilities of class 1 and save to assorted\_pred\_prob  
assorted\_pred\_prob = logreg.predict\_proba([[1]])  
  
# Print the probabilities using assorted\_pred\_prob  
print('### Predictions for Class 1 ###')  
print(assorted\_pred\_prob, '\n')  
  
# Predict the probability of maintaining  
# a user for 30 days (0, 1)  
  
# Get probability for retained users for 30 days with 1 install  
print('### Prediction for number retained for 30 days with 1 install ###')  
print(logreg.predict\_proba([[1]]), '\n')  
  
# Get probability for retained users for 30 days with 2 installs  
print('### Prediction for number retained for 30 days with 2 installs ###')  
print(logreg.predict\_proba([[2]]), '\n')  
  
# Get probability for retained users for 30 days with 4 installs  
print('### Prediction for number retained for 30 days with 4 installs ###')  
print(logreg.predict\_proba([[4]]), '\n')  
  
  
# Print the intercept  
print('### Log Regression Y Intercept ###')  
print(logreg.intercept\_, '\n')  
  
# Print the coefficient  
print('### Log Regression Coefficient ###')  
print(logreg.coef\_)