This assessment will consist of two parts. The first will require Python or R, while the second will require SQL.

Upon completion, please either attach relevant file(s) or send a link to your own GitHub repository replying to the email from which you have received this exercise.

You are encouraged to think of this as an incremental exercise (i.e., reply when you have implemented the basic acceptance criteria, then again if you make further improvements or implement one or more of the bonuses).

Python or R:

1. Perform the following with the attached “top50.csv” file:

- Load the dataset and perform EDA and determine if any cleanup (data conversions, name changes, scaling, imputation of missing values, etc.) is needed.

- Determine if any correlation exists between features.

- Visualize and examine the distribution of popularity scores and show if any outliers or skewing exists.

- Perform any transformations as needed and create a model to predict the popularity of a song, while avoiding overfitting and showing relevant scores.

\*Bonus for reducing dimensionality or clustering

2. Write the code to calculate and plot the Precision-Recall curve for any existing ML model you like. Could be a predictor, classifier, or detector. Use any simple ground truth dataset you like. For example, if you use an object classifier or detector, it could be of dogs and cats, or bicycles and cars, or anything like that. However, the dataset choice is yours. Turn in the code sample, GT dataset, and PR curve plot.

\*Bonus for adding the P-R results of any second corresponding predictor, classifier, or detector to the same chart for comparison.

SQL:

\*/

CREATE TABLE users

id INTEGER PRIMARY KEY,

firstName VARCHAR(30) NOT NULL,

lastName VARCHAR(30) NOT NULL;

CREATE TABLE sessions

id INTEGER PRIMARY KEY,

userId INTEGER NOT NULL FOREIGN KEY REFERENCES users(id),

duration DECIMAL NOT NULL;

CREATE TABLE transactions

id INTEGER PRIMARY KEY,

userId INTEGER NOT NULL FOREIGN KEY REFERENCES users(id)

productId INTEGER,

originDate TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

description VARCHAR(30),

qty DECIMAL;

/\*

1. User data and their app usage data are kept in the tables above. Each

user can have multiple sessions. Write a SQL query that:

- selects the ‘userIds’ with the top 50 average session durations

- selects their full name from the user table as a single new column

\*Bonus for adding in a new column for each user and flagging their 3rd longest duration with 1,

otherwise put 0 (put 0 if they have fewer than 3 sessions)

-- Your query or pseudo query here.