This is not mandatory reading, but here's the code we'll run in the "Snowpark ML Modeling" videos.

# Note: This is not code you can run in a SQL worksheet. We ran this in a Jupyter notebook

# install these two libraries

!pip install snowflake-ml-python

!pip install snowflake-snowpark-python

# don’t worry too much about this – I created credential.py to hold my login credentials

from credential import params

# if you want guidance on connecting to Snowflake from your IDE, see here:

# https://docs.snowflake.com/en/developer-guide/snowpark/python/creating-session#creating-a-session

# import the libraries you’ll need

from snowflake.snowpark import Session

from snowflake.ml.modeling.xgboost import XGBClassifier

from snowflake.snowpark.functions import col

from snowflake.ml.modeling import preprocessing

# Here’s the neighborhood visiting pattern the truck follows:

# In January, the truck goes to N1 on the 1st, 8th, 15th, 22nd, and 29th, and N2 the other days.

# From February through November, it goes to:

# N1 on the 1st

# N2 on the 2nd

# N3 on the 3rd

# N4 on the 4th

# N5 on the 5th

# N6 on the 6th

# N7 on the 7th

# N1 on the 8th

# N2 on the 9th

# etc.

# Every December, it only goes to N8.

month\_days = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]

pre = {}

for i,month\_length in enumerate(month\_days):

month = i + 1

for day in range(1,month\_length+1):

# In January, it goes to neighborhood 1 on Mondays, and neighborhood 2 the other days.

if ((month) == 1):

if (day) % 7 == 1:

pre[(month,day)] = 1

else:

pre[(month,day)] = 2

# From February through November, it goes to neighborhood 1 on the 1st, 2 on the 2nd, 3 on the 3rd,

# 4 on the 4th, 5 on the 5th, 6 on the 6th, and 7 on the 7th, 1 on the 8th, 2 on the 9th, etc.

elif ((month) <= 11):

pre[(month,day)] = ((day-1) % 7) + 1

# Every December, it only goes to neighborhood 8.

elif ((month) == 12):

pre[(month,day)] = 8

# see what the pre dictionary looks like

pre

# Note: Here, I skipped the step of uploading the final “df\_clean” dataset to Snowflake

# create a Session with the necessary connection info

session = Session.builder.configs(params).create()

# create a dataframe (though note that this doesn’t pull data into your local machine)

snowpark\_df = session.table("test\_database.test\_schema.df\_clean")

# show the first forty rows of the dataframe

snowpark\_df.show(n=40)

# count the rows in the dataframe

snowpark\_df.count()

# describe the dataframe

snowpark\_df.describe().show()

# groupby neighborhood, and show the counts

snowpark\_df.group\_by("Neighborhood").count().show()

# one way to scale your target (neighborhood) so you can use it in the XGBClassifier model

test = snowpark\_df.withColumn('NEIGHBORHOOD2', snowpark\_df.neighborhood - 1).drop("Neighborhood")

test.show()

# now use scikit-learn's LabelEncoder -- a more general solution -- through Snowpark ML

le = LabelEncoder(input\_cols=['NEIGHBORHOOD'], output\_cols= ['NEIGHBORHOOD2'], drop\_input\_cols=True)

# apply the LabelEncoder

fitted = le.fit(snowpark\_df.select("NEIGHBORHOOD"))

snowpark\_df\_prepared = fitted.transform(snowpark\_df)

snowpark\_df\_prepared.show()

# split the data into a training set and a test set

train\_snowpark\_df, test\_snowpark\_df = snowpark\_df\_prepared.randomSplit([0.9, 0.1])

# save training data

train\_snowpark\_df.write.mode("overwrite").save\_as\_table("df\_clean\_train")

# save test data

test\_snowpark\_df.write.mode("overwrite").save\_as\_table("df\_clean\_test")

# create and train the XGBClassifier model

FEATURE\_COLS = ["MONTH", "DAY"]

LABEL\_COLS = ["NEIGHBORHOOD2"]

# Train an XGBoost model on snowflake.

xgboost\_model = XGBClassifier(

input\_cols=FEATURE\_COLS,

label\_cols=LABEL\_COLS

)

xgboost\_model.fit(train\_snowpark\_df)

# check the accuracy using scikit-learn's score functionality through Snowpark ML

accuracy = xgboost\_model.score(test\_snowpark\_df)

print("Accuracy: %.2f%%" % (accuracy \* 100.0))