# Import libraries and methods/functions

import pandas as pd

from sklearn.preprocessing import StandardScaler, OneHotEncoder

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import RidgeClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification\_report, confusion\_matrix

# Start your code here!

# Load data

telco\_demog = pd.read\_csv('telecom\_demographics.csv')

telco\_usage = pd.read\_csv('telecom\_usage.csv')

# Join data

churn\_df = telco\_demog.merge(telco\_usage, on='customer\_id')

# Identify churn rate

churn\_rate = churn\_df['churn'].value\_counts() / len(churn\_df)

print(churn\_rate)

# Identify categorical variables

print(churn\_df.info())

# One Hot Encoding for categorical variables

churn\_df = pd.get\_dummies(churn\_df, columns=['telecom\_partner', 'gender', 'state', 'city', 'registration\_event'])

# Feature Scaling

scaler = StandardScaler()

# 'customer\_id' is not a feature

features = churn\_df.drop(['customer\_id', 'churn'], axis=1)

features\_scaled = scaler.fit\_transform(features)

# Target variable

target = churn\_df['churn']

# Splitting the dataset

X\_train, X\_test, y\_train, y\_test = train\_test\_split(features\_scaled, target, test\_size=0.2, random\_state=42)

# Instantiate the Logistic Regression

logreg = LogisticRegression(random\_state=42)

logreg.fit(X\_train, y\_train)

# Logistic Regression predictions

logreg\_pred = logreg.predict(X\_test)

# Logistic Regression evaluation

print(confusion\_matrix(y\_test, logreg\_pred))

print(classification\_report(y\_test, logreg\_pred))

# Instantiate the Random Forest model

rf = RandomForestClassifier(random\_state=42)

rf.fit(X\_train, y\_train)

# Random Forest predictions

rf\_pred = rf.predict(X\_test)

# Random Forest evaluation

print(confusion\_matrix(y\_test, rf\_pred))

print(classification\_report(y\_test, rf\_pred))

# Which accuracy score is higher? Ridge or RandomForest

higher\_accuracy = "RandomForest"