# Import necessary libraries from sklearn.datasets import fetch\_openml from sklearn.model\_selection import train\_test\_split from sklearn.svm import SVC from sklearn.metrics import accuracy\_score from imblearn.under\_sampling import RandomUnderSampler from imblearn.over\_sampling import RandomOverSampler # Load MNIST dataset mnist = fetch\_openml('mnist\_784') X, y = mnist['data'], mnist['target'] # Split into training and test sets X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, random\_state=42) # Undersample majority class in training set undersampler = RandomUnderSampler(random\_state=42) X\_train\_under, y\_train\_under = undersampler.fit\_resample(X\_train, y\_train) # Train SVM classifier on undersampled dataset clf\_under = SVC(gamma='auto', random\_state=42) clf\_under.fit(X\_train\_under, y\_train\_under) # Evaluate classifier on test set y\_pred\_under = clf\_under.predict(X\_test) accuracy\_under = accuracy\_score(y\_test, y\_pred\_under) # Oversample minority class in training set oversampler = RandomOverSampler(random\_state=42) X\_train\_over, y\_train\_over = oversampler.fit\_resample(X\_train, y\_train) # Train SVM classifier on oversampled dataset clf\_over = SVC(gamma='auto', random\_state=42) clf\_over.fit(X\_train\_over, y\_train\_over) # Evaluate classifier on test set y\_pred\_over = clf\_over.predict(X\_test) accuracy\_over = accuracy\_score(y\_test, y\_pred\_over) # Compare results print(f"Accuracy of classifier trained on undersampled dataset: {accuracy\_under:.3f}") print(f"Accuracy of classifier trained on oversampled dataset: {accuracy\_over:.3f}")