SE-2102

Sadyk Adilet

Asatbek Khairullayev

Assignment 1

import numpy as np

from sklearn.linear\_model import LogisticRegression

from sklearn.svm import SVC

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

from sklearn.metrics import accuracy\_score

Load the dataset

# Assuming you have already downloaded the dataset and preprocessed it

X = ... # input features

y = ... # target labels

Divide data into train, validation, and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.1, random\_state=42)

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

Implement Softmax Regression

softmax\_model = LogisticRegression(multi\_class='multinomial', solver='lbfgs', max\_iter=1000)

softmax\_model.fit(X\_train, y\_train)

Implement SVM

svm\_model = SVC(kernel='linear')

svm\_model.fit(X\_train, y\_train)

# Step 5: Model evaluation

softmax\_pred = softmax\_model.predict(X\_test)

softmax\_acc = accuracy\_score(y\_test, softmax\_pred)

svm\_pred = svm\_model.predict(X\_test)

svm\_acc = accuracy\_score(y\_test, svm\_pred)

Compare accuracies

if softmax\_acc > svm\_acc:

print("Softmax Regression has the best accuracy.")

else:

print("SVM has the best accuracy.")

import numpy as np

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

# Assuming you have already loaded and preprocessed your data

X = ... # input features

y = ... # target labels

Divide data into train+validation and test sets

X\_trainval, X\_test, y\_trainval, y\_test = train\_test\_split(X, y, test\_size=0.1, random\_state=42)

Further divide train+validation set into train and validation sets

X\_train, X\_val, y\_train, y\_val = train\_test\_split(X\_trainval, y\_trainval, test\_size=0.2, random\_state=42)

Print the sizes of each set

print("Train set size:", len(X\_train))

print("Validation set size:", len(X\_val))

print("Test set size:", len(X\_test))

import numpy as np

from sklearn.model\_selection import cross\_val\_score

from sklearn.linear\_model import LogisticRegression

from sklearn.svm import SVC

# Assuming you have already divided your data into X\_train, y\_train

# Step 1: Create instances of the models

softmax\_model = LogisticRegression(multi\_class='multinomial', solver='lbfgs', max\_iter=1000)

svm\_model = SVC(kernel='linear')

# Step 2: Perform cross-validation

softmax\_scores = cross\_val\_score(softmax\_model, X\_train, y\_train, cv=5)

svm\_scores = cross\_val\_score(svm\_model, X\_train, y\_train, cv=5)

# Step 3: Print the accuracies for each fold

print("Softmax Regression cross-validation accuracy:", softmax\_scores)

print("SVM cross-validation accuracy:", svm\_scores)

# Step 4: Print the mean accuracies and select the best model

softmax\_mean\_accuracy = np.mean(softmax\_scores)

svm\_mean\_accuracy = np.mean(svm\_scores)

if softmax\_mean\_accuracy > svm\_mean\_accuracy:

print("Softmax Regression has the best cross-validation accuracy.")

else:

print("SVM has the best cross-validation accuracy.")

For Softmax Regression (using L2 regularization):

from sklearn.linear\_model import LogisticRegression

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

from sklearn.metrics import accuracy\_score

# Step 1: Create an instance of the Softmax Regression model

softmax\_model = LogisticRegression(multi\_class='multinomial', solver='lbfgs', max\_iter=1000, penalty='l2', C=1.0)

# Step 2: Train the model

softmax\_model.fit(X\_train, y\_train)

# Step 3: Evaluate the model on the validation set

y\_val\_pred = softmax\_model.predict(X\_val)

val\_accuracy = accuracy\_score(y\_val, y\_val\_pred)

# Step 4: Evaluate the model on the test set

y\_test\_pred = softmax\_model.predict(X\_test)

test\_accuracy = accuracy\_score(y\_test, y\_test\_pred)

print("Softmax Regression with L2 regularization:")

print("Validation Accuracy:", val\_accuracy)

print("Test Accuracy:", test\_accuracy)

For SVM (using L1 regularization):

from sklearn.svm import LinearSVC

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

from sklearn.metrics import accuracy\_score

# Step 1: Create an instance of the SVM model

svm\_model = LinearSVC(loss='squared\_hinge', penalty='l1', dual=False, C=1.0)

# Step 2: Train the model

svm\_model.fit(X\_train, y\_train)

# Step 3: Evaluate the model on the validation set

y\_val\_pred = svm\_model.predict(X\_val)

val\_accuracy = accuracy\_score(y\_val, y\_val\_pred)

# Step 4: Evaluate the model on the test set

y\_test\_pred = svm\_model.predict(X\_test)

test\_accuracy = accuracy\_score(y\_test, y\_test\_pred)

print("SVM with L1 regularization:")

print("Validation Accuracy:", val\_accuracy)

print("Test Accuracy:", test\_accuracy)

import numpy as np

from sklearn.linear\_model import LogisticRegression

from sklearn.svm import SVC

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

from sklearn.metrics import accuracy\_score

# Assuming you have already implemented Softmax Regression and SVM algorithms, and divided the data

# Step 1: Train the Softmax Regression model

softmax\_model = LogisticRegression(multi\_class='multinomial', solver='lbfgs', max\_iter=1000)

softmax\_model.fit(X\_train, y\_train)

# Step 2: Train the SVM model

svm\_model = SVC(kernel='linear')

svm\_model.fit(X\_train, y\_train)

# Step 3: Evaluate the accuracies on the validation set

softmax\_pred\_val = softmax\_model.predict(X\_val)

softmax\_acc\_val = accuracy\_score(y\_val, softmax\_pred\_val)

svm\_pred\_val = svm\_model.predict(X\_val)

svm\_acc\_val = accuracy\_score(y\_val, svm\_pred\_val)

# Step 4: Compare the accuracies on the validation set

if softmax\_acc\_val > svm\_acc\_val:

best\_model = softmax\_model

print("Softmax Regression has the best accuracy on the validation set.")

else:

best\_model = svm\_model

print("SVM has the best accuracy on the validation set.")

# Step 5: Evaluate the accuracy of the best model on the test set

best\_model\_pred\_test = best\_model.predict(X\_test)

best\_model\_acc\_test = accuracy\_score(y\_test, best\_model\_pred\_test)

print("Best model accuracy on the test set:", best\_model\_acc\_test)

In this example, we train the Softmax Regression and SVM models on the training set. Then we evaluate their accuracies on the validation set and compare them to determine which model performs better.

We select the model with the higher accuracy on the validation set as the best model and evaluate its accuracy on the test set.