lecture_7_743

October 11, 2019

1 Lecture 7 SVM

In this assignement we used SVM to perform classification on the mnist dataset. In order to do this, sci-kit learns SVM classifier SVC was used, and all different supported kernel were tested As the data isn't scaled, the gamma value "scaled" was used. Because SVM is slow, the training was initially only done on 10% of the data, however since the code was run on a server, we also ran the training on the entire train set. The results are compared in the below tables

---

Kernel	Accuracy
Linear	91.42%
Poly	94.65%
RBF	95.60%
Sigmoid	83.96%

10% training set

Kernel	Accuracy
Linear	95.82%
Poly	97.71%
RBF	97.92%
Sigmoid	77.62%

100% training set

2 Code

In [8]:
```
# ###################################
# Group ID : 743
# Members : Mathias Stougaard Lynge, Moaaz allahham, Kasper Schøn Henriksen, Mikkel D.B.
# Date : 09/10-2019
# Lecture: 7  Support Vector Machines
# Dependencies  : Are all described in requirements.txt
# Python version: >3.5
# Functionality :
# ###################################

from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
import numpy as np
import pandas as pd
```

2.1 Config

In [2]:
```
train_size = 1.0                                # Percentage of the train set to train on
kernels = ('linear', 'poly', 'rbf', 'sigmoid')   # The kernels to use for SVM
```

2.2 Data

In [3]:
```
train_data = pd.read_csv("mnist_train.csv")
test_data = pd.read_csv("mnist_test.csv")

train_labels = train_data['label']
train_data.drop(columns=['label']);

test_labels = test_data['label']
test_data.drop(columns=['label']);
```

In [4]:
```
if train_size == 1:
    train_data_reduced = train_data
    train_labels_reduced = train_labels
else:
    train_data_reduced, waste_1, train_labels_reduced, waste_2 = train_test_split(train_
    del waste_1
    del waste_2
```

3 Implementation

```
In [6]: models = {}
        for idx, kernel in enumerate(kernels):
            print(f'Training kernel {idx+1}/{len(kernels)}: {kernel}')
            models[kernel] = {}
            models[kernel]['model'] = SVC(kernel = kernel, gamma = 'scale')
            models[kernel]['model'].fit(train_data_reduced, train_labels_reduced)
            models[kernel]['accuracy'] = models[kernel]['model'].score(test_data, test_labels)
            print(f'  Accuracy: {models[kernel]["accuracy"]*100:.2f}%')
```

```
Training kernel 1/4: linear
  Accuracy: 95.82%
Training kernel 2/4: poly
  Accuracy: 97.71%
Training kernel 3/4: rbf
  Accuracy: 97.92%
Training kernel 4/4: sigmoid
  Accuracy: 77.62%
```

```
In [7]: out_string = ["Accuracy:\n"]

        model_list = [m for m in models]
        longest_word = len(max(model_list, key=len))

        for m in models:
            base_pad = 8
            padding = longest_word+base_pad - len(m)
            out_string.append(f'  {m}:{models[m]["accuracy"] * 100:>{padding}.2f}%\n')
        print(''.join(out_string))
```

```
Accuracy:
  linear:     95.82%
  poly:       97.71%
  rbf:        97.92%
  sigmoid:    77.62%
```