Batuhan Yalçın

64274

10.05.2022

# COMP 448/548 – Medical Image Analysis

# Homework #2

## PART I

All functions implemented in python and jupyternotebook file.

## PART II

Since calculating features takes too much time, I added my result file to uploaded zip you can use directly that files.

CLASS-IMBALANCE:

In the given train set class 2 has more data than the class 1 and it also has more data than the class 3. This is also reason that why we have better accuracy in class 2. To solve this issue, I used the “” library in the library the input function for the SVM algorithm takes class weights as an input. To solve class-imbalance problem I give the input as a balanced here the implementation example and source page:

Source page:

https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

Example:

https://stackoverflow.com/questions/52896387/svc-with-class-weight-in-scikit-learn

NORMALIZATION:

For the normalization I used “sklearn.preprocessing” tool of “StandartScaler()” which is commonly used for the machine learning algorithms. Standardization of a dataset is a common requirement for many machine learning estimators: they might behave badly if the individual features do not more or less look like standard normally distributed data. To achieve standardization function first centers and scale each features independently by computing the relevant statistics on the samples in the training set. Then using the parameters of mean and standard deviation after process I transformed data to use standardized (Normalized) data.

Source page:

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

LIBRARY FOR SVM:

Library used for SVM (Support Vector Machines) is that scikit-learn's sklearn.svm liblary. Which has lots of the SVM methods. From that library I import the SVC and I use SVC for linear kernel and RBF kernel.

The implementation link of the library:

https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

Additional comments:

Since the class 2 has more data than the others although normalization and class-balance implementation the accuracy on the class 2 has more than the other classes.

I also normalize to confusion matrix to only take count results between 0 and 1 for the accuracy.

Since the limit is 1 page, I did not go over for loop results on chosen C and Gamma values. For more details on the choosing appropriate C and gamma values look the description in the code.

| | Selected parameters | Training set accuracies | | | | Test set accuracies | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Class1 | Class2 | Class3 | Overall | Class1 | Class2 | Class3 | Overall |
| Linear Kernel | C = 5.2 (optimal) | %36 | %78 | %84 | %66 | %27 | %40 | %64 | %42.4 |
| Linear Kernel | C = 0.196 (Best test) | %30 | %83 | %79 | %65 | %27 | %53 | %64 | &47.2 |
| RBF kernel | C = 50 gamma=3.4 | %100 | %100 | %100 | %100 | %29 | %67 | %41 | %47.2 |
| Statistically different? | | Yes | No | Yes | Yes | Yes | Yes | No | Yes |

## PART III

For the class-imbalance problem and normalization see the part 2.

For the Grid-Based approach the result accuracies get higher than the entire image especially in the test results. For the statistical differences in linear kernel there were some differences and in RBF kernel all statistics where different. The main reason of that is when choosing the RBF and Linear method I choose the C and gamma values where the test accuracy gets peaks. If I choose the C and gamma values with considering train set accuracy, and similar to the entire image if I were full overall on train and test may less or not statically difference occur. The reason of this possible similarity and seen similarity on statistic is that same data and image used with the cropped size.

| | Selected | Training set accuracies | | | | Test set accuracies | | | |
|---|---|---|---|---|---|---|---|---|---|
| | parameters | Class1 | Class2 | Class3 | Overall | Class1 | Class2 | Class3 | Overall |
| Linear Kernel (grid-based) | C = 5.1 | %30 | %77 | %84 | %63 | %21 | %53 | %67 | %46 |
| Linear Kernel (entire image) | C = 5.2 | %36 | %78 | %84 | %66 | %27 | %40 | %64 | %42.4 |
| Statistically different? | | Yes | No | Yes | Yes | Yes | Yes | No | Yes |

| | Selected | Training set accuracies | | | | Test set accuracies | | | |
|---|---|---|---|---|---|---|---|---|---|
| | parameters | Class1 | Class2 | Class3 | Overall | Class1 | Class2 | Class3 | Overall |
| RBF kernel (grid-based) | C = 10 gamma = 0.4 | %52 | %92 | %92 | %79 | %29 | %63 | %59 | %51 |
| RBF kernel (entire image) | C = 50 gamma = 3.4 | %100 | %100 | %100 | %100 | %29 | %67 | %41 | %47.2 |
| Statistically different? | | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

## PART IV

When I add the features recognition with the filters. I used Python cv2.xfeatures2d.SIFT_create() and “keypoints_1, descriptors_1 = shift.detectAndCompute(img_initial,None)” methods. I get the key points of the image and using them I create new image with this method I get best accuracy results for overall. However, class 3 due to library of SHIFT get wrong results. I guess the reason of that while the calculation the class 3 property it uses libraries key points information and try to compare according to them. Despite this the key point approach gives the best result comparing the grid-based and entire image. It can be concluded that using key points idea better than the cropping the image with grid for this case.  Statistical differences have similar explanation to the part 3.

Result table of the part 4 is in below:

| | Selected parameters | Training set accuracies | | | | Test set accuracies | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Class1 | Class2 | Class3 | Overall | Class1 | Class2 | Class3 | Overall |
| Linear Kernel (Key_points) | C = 5.1 | %98 | %100 | %100 | %99 | %19 | %88 | %13 | %44 |
| Linear Kernel (grid-based) | C = 5.1 | %30 | %77 | %84 | %63 | %21 | %53 | %67 | %46 |
| Linear Kernel (entire image) | C = 5.2 | %36 | %78 | %84 | %66 | %27 | %40 | %64 | %42.4 |
| Statistically different? Between grid-based and Key_points | | Yes | Yes | No | Yes | Yes | Yes | Yes | Yes |

error due to liblarv used at class 3

| | Selected parameters | Training set accuracies | | | | Test set accuracies | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Class1 | Class2 | Class3 | Overall | Class1 | Class2 | Class3 | Overall |
| RBF kernel (Key_points) | C = 5000 gamma = 0.3 | %100 | %100 | %100 | %100 | %73 | %85 | 0 | %56.25 |
| RBF kernel (grid-based) | C = 10 gamma = 0.4 | %52 | %92 | %92 | %79 | %29 | %63 | %59 | %51 |
| RBF kernel (entire image) | C = 50 gamma = 3.4 | %100 | %100 | %100 | %100 | %29 | %67 | %41 | %47.2 |
| Statistically different? | | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

Thank you.