2019/05/20  
0780828  
Alisher Mukashev

Homework #5

# **SVM on MNIST dataset**

## 1. Use different kernel functions (linear, polynomial, and RBF kernels) and have comparison between their performance.

## **Explanation of code**.

## I used libsvm library. Firstly, I just read csv files. Then, using functions *svm\_train* and *svm\_predict* I got metrics, which I used for comparison *linear*, *polynomial* and *RBF* kernels performance.

## *Linear* - 𝐾(𝑢,𝑣)=𝑢𝑇𝑣 (1.1)

## *Polynomial* - 𝐾(𝑢,𝑣,𝑐,𝑑)=(𝑢𝑇𝑣+𝑐)𝑑 (1.2)

## *RBF* - 𝐾(𝑢,𝑣,𝛾)=𝑒𝑥𝑝(−𝛾|𝑢−𝑣|2) (1.3)

## Where *u, v* – datasets, *c* – coef0, *d* – degree, 𝛾 – gamma.

## **Results.**

|  |  |  |  |
| --- | --- | --- | --- |
| (Kernel) | *Linear* | *Polynomial* | *RBF* |
| ***Testing accuracy*** | 95.08% (2377/2500) | 34.68% (867/2500) | 95.32% (2383/2500) |

## Note: - *polynomial* (default: gamma=1/num\_features, coef=0, degree=3) - *RBF* (default: gamma=1/num\_features)

## 2. Please use C-SVC (you can choose by setting parameters in the function input, C-SVC is soft-margin SVM). Since there are some parameters you need to tune for, please do the grid search for finding parameters of best performing model. For instance, in C-SVC you have a parameter C, and if you use RBF kernel you have another parameter γ, you can search for a set of (C, gamma) which gives you best performance in cross-validation.

## **Explanation of code**.

## Firstly, I read csv files. Then, I selected hyperparameters which I changed to find the best solution (this is the grid search method). These hyperparameters are cost (the parameter C of C-SVC), gamma, degree, coef0, kernel type.

## **Results.**

## I got the best solution (accuracy = 98.2). The best Kernel parameters are ***polynomial*** kernel, ***cost*** = 1, ***gamma*** = 0.2, ***degree*** = 2, ***coef0*** = 2.

## 3. Use *linear* kernel+*RBF* kernel together (therefore a new kernel function) and compare its performance with respect to others. You would need to find out how to use a user-defined kernel in libsvm.

## **Explanation of code**.

## Here I had to use my own kernel function (without libsvm library). Using equations 1.1 and 1.3, I created new kernel function (it is not model). Then, using library function, I trained getting model.

## **Results.**

## ***Testing accuracy*** = 95.32% (2383/2500).

# **find out support vectors**

## 1. Train SVM model with different kernel functions (linear, polynomial, RBF and linear+RBF kernels) and visualize the result.

## **Explanation of code**.

## I had two datasets which I tried to cluster. I also showed support vectors (red square dots). Firstly, as usually, I read csv files. Then, using functions *svm\_train* and *svm\_predict* I got 4 figures, which I used for comparison *linear*, *polynomial*, *RBF* and *linear*+*RBF* kernels.

## **Results.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| (Kernel) | *Linear* | *Polynomial* | *RBF* | *Linear+RBF* |
| ***Testing accuracy*** | 99.5667% (2987/2500) | 99.3333% (2980/2500) | 99.4667% (2984/2500) | 99.4333% (2983/2500) |

## 