标注一览

陈闽在2页标注了10处

x10

1. Overview

In this project, you should use SVM to deal with data. You can use any related methods to train your model, for example, SMO or Gradient Descent Algorithm. The given data contains 2000 ten dimensional samples. The training time of the model on the testing dataset is up to 60s. The SVM model you built should obtain the good generalization performance on dataset that used to evaluate your algorithm. The scores you get in this project will be given according to the performance of

#1 p.1

efficiency issues of different programming languages, your algorithm must be implemented using Python 3.6 and the only allowed library is **numpy.**

ii. We will check your code to make sure that you do not use other packages.

iii. You can use **train_data.txt** to train your model.

iv. The SVM program must be named by **SVM.py**

v. **Input:** the format of the estimator call should be as follows:

python SVM.py <test data > -t <time budget>

<test data> is the absolute path of the test data

<time budget> is a positive number which indicates how many seconds

#2 p.1

You can output one label per line for each test example. We will redirect your print information to a file.

Output format:

1

-1

.....

#3 p.2

Project 4 – Support Vector Machine(SVM)

1. Overview

In this project, you should use SVM to deal with data. You can use any related methods to train your model, for example, SMO or Gradient Descent Algorithm. The given data contains 2000 ten dimensional samples. The training time of the model on the testing dataset is up to 60s. The SVM model you built should obtain the good generalization performance on dataset that used to evaluate your algorithm. The scores you get in this project will be given according to the performance of your algorithms in our test.

2. General rules

After your submission, we will test your SVM program. To make this process as smooth as possible, the package you submit must satisfy the following requirements.

a) Report

i. The report should be submitted in pdf format, in which you should describe the core idea of your design and give the pseudo code. Please see the report template and evaluation standard (Released on sakai).

b) Programming aspects

i. To get rid of the operating system related issues and the execution efficiency issues of different programming languages, your algorithm must be implemented using Python 3.6 and the only allowed library is **numpy.**

ii. We will check your code to make sure that you do not use other packages.

iii. You can use **train_data.txt** to train your model.

iv. The SVM program must be named by **SVM.py**

v. **Input:** the format of the estimator call should be as follows:

python SVM.py <test data > -t <time budget>

<test data> is the absolute path of the test data

<time budget> is a positive number which indicates how many seconds (in Wall clock time, range: [10s, 60s]) your algorithm can spend on this instance.

vi. **Output**: the predicted value of your program according the test data.

You can output one label per line for each test example. We will redirect your print information to a file.

Output format:

1

-1

.....

1

3. Evaluation

After your submission, we will test your SVM program. The scores you get in this test depend on your predict accuracy.

4. Test Environment

1) Operation System: Debian 10
2) Server CPU： 2.2GHz*2, 20-core total
3) Python version: 3.6.6