0880803 – 馮清海

Thanh – Hai, Phung

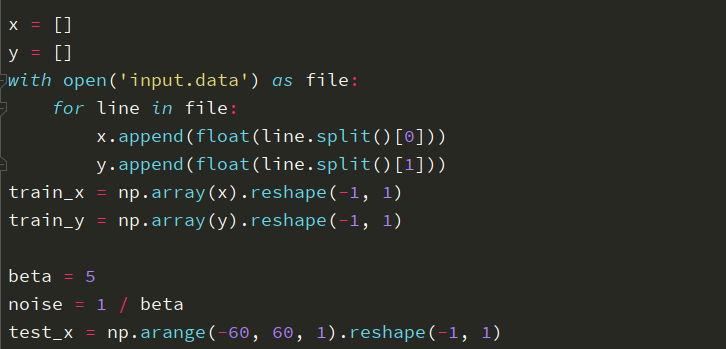
**HOMEWORK 5**

**MACHINE LEARNING**

1. **Gaussian Process**

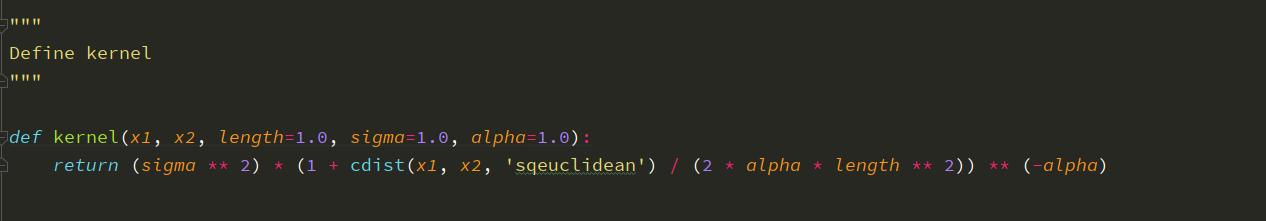
Reading input

<code>



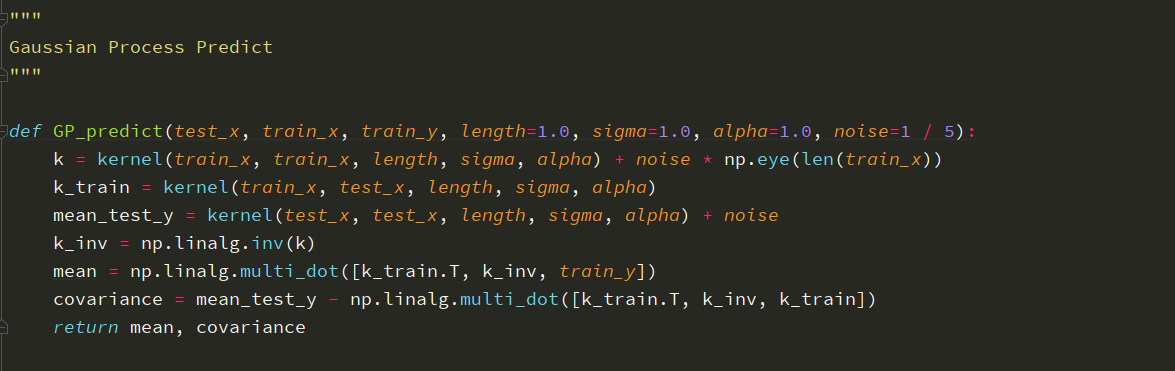
Define quadratic kernel

<code>



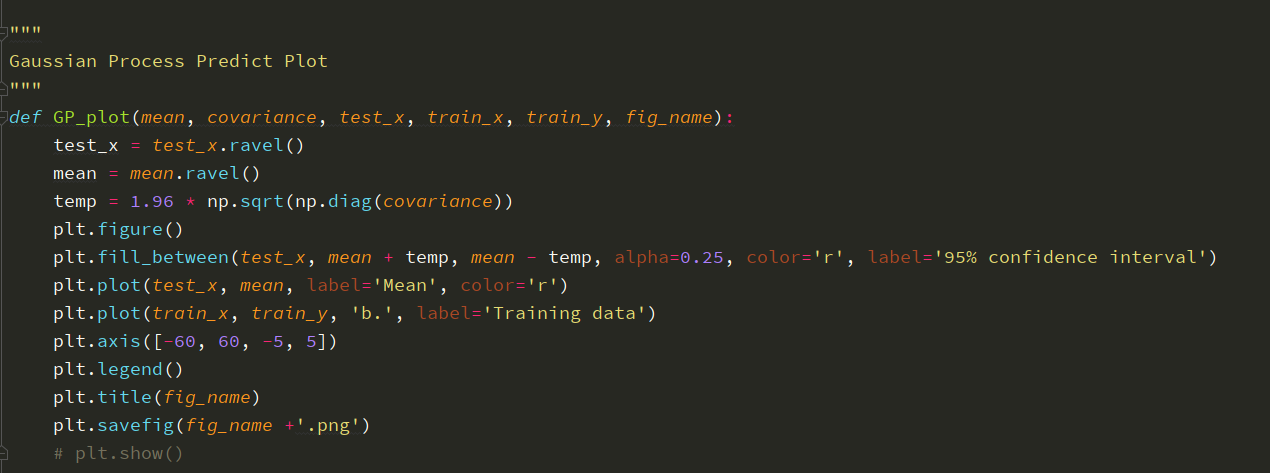
Define Posterior Gaussian Process

<code>

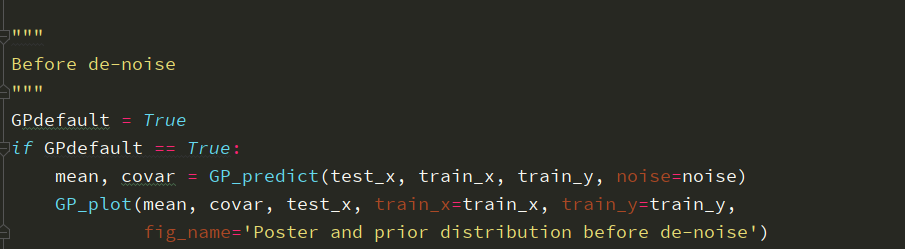


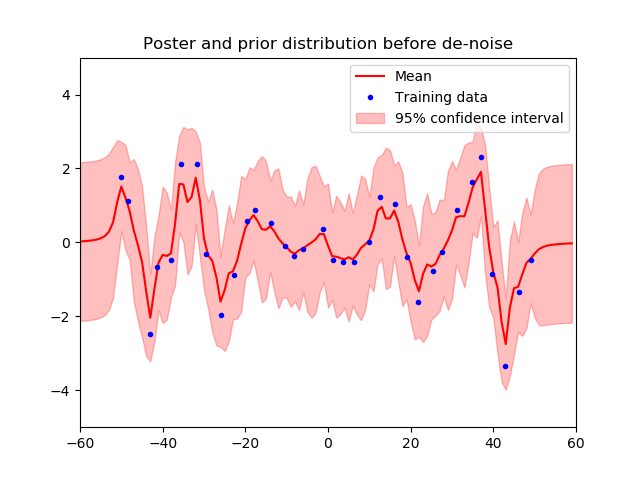
Define Plot function

<code>



Before de-noise

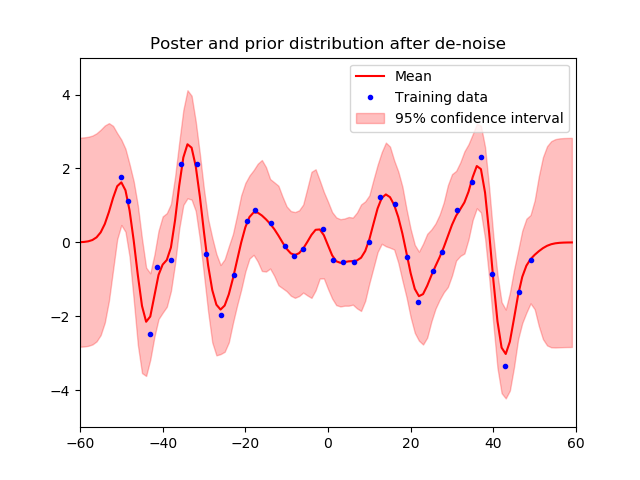




After de-noise

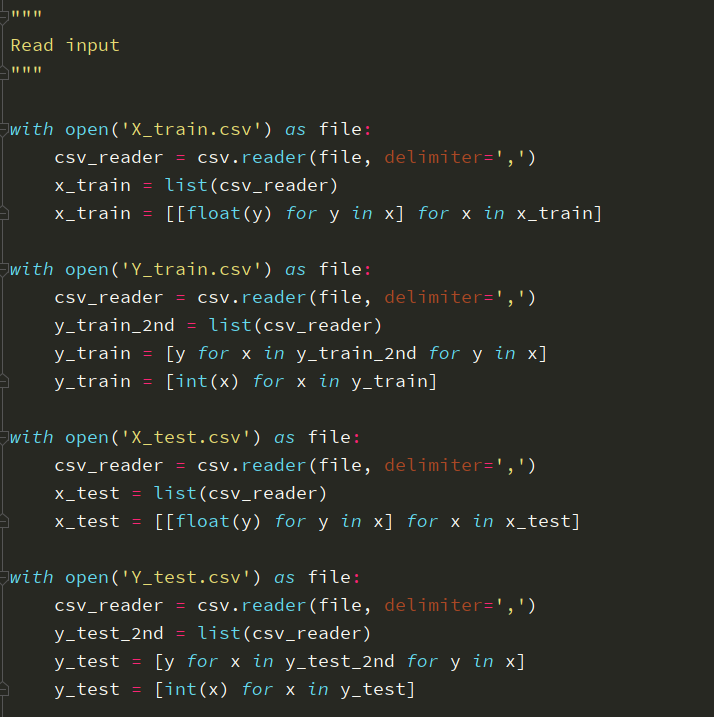
<code>





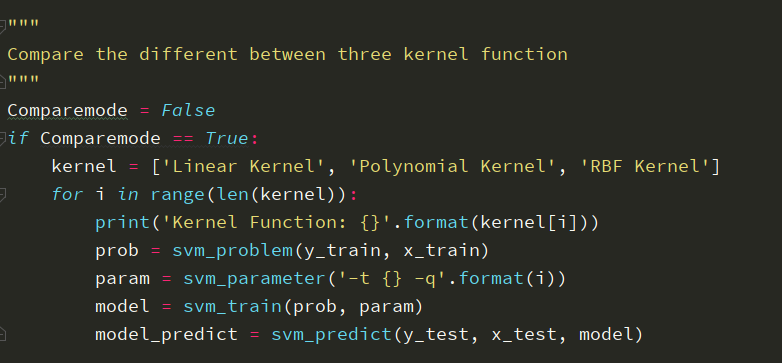
1. **Support Vector Machine (SVM)**

Reading input

<code>

1. **Use difference kernel function (by build-in kernels) (No Cross – Validation)**

<code>



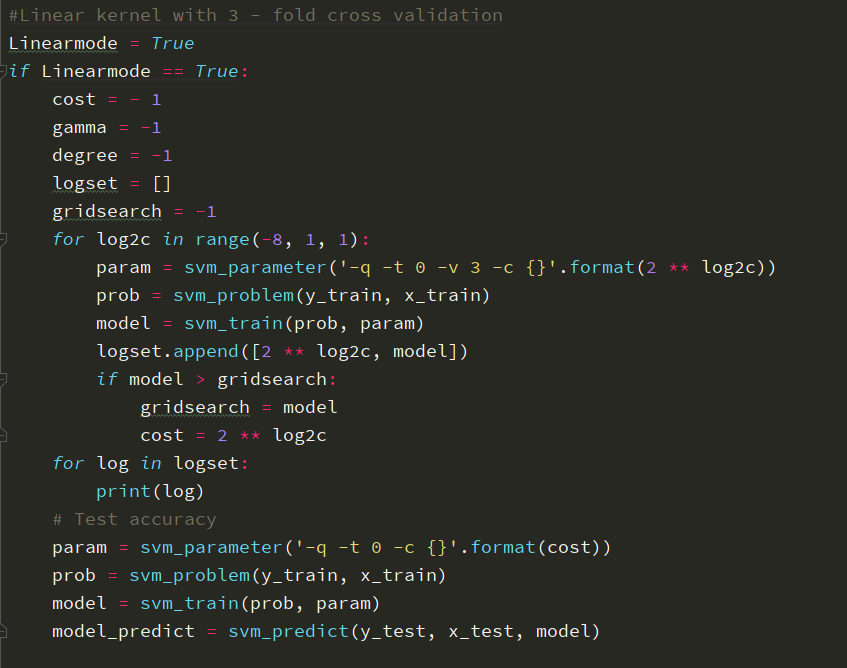
The setting default built in run thru three kernel functions and run in test set.

Table 1: Comparison of 3 different default kernels

|  |  |  |
| --- | --- | --- |
| Kernel | Setting | Accuracy |
| Linear | c = 1 | **95.08%** (2377/2500) |
| Polynomial | c = 1, d = 3, gamma = 0, r = 0 | **34.68%** (867/2500) |
| Radial Basis Function | gamma = 0 | **95.32%** (2382/2500) |

1. **C-SVC, Grid – Search, Cross – Validation**
2. **Linear, 3 – folds cross – validation, searching for variable c**

<code>



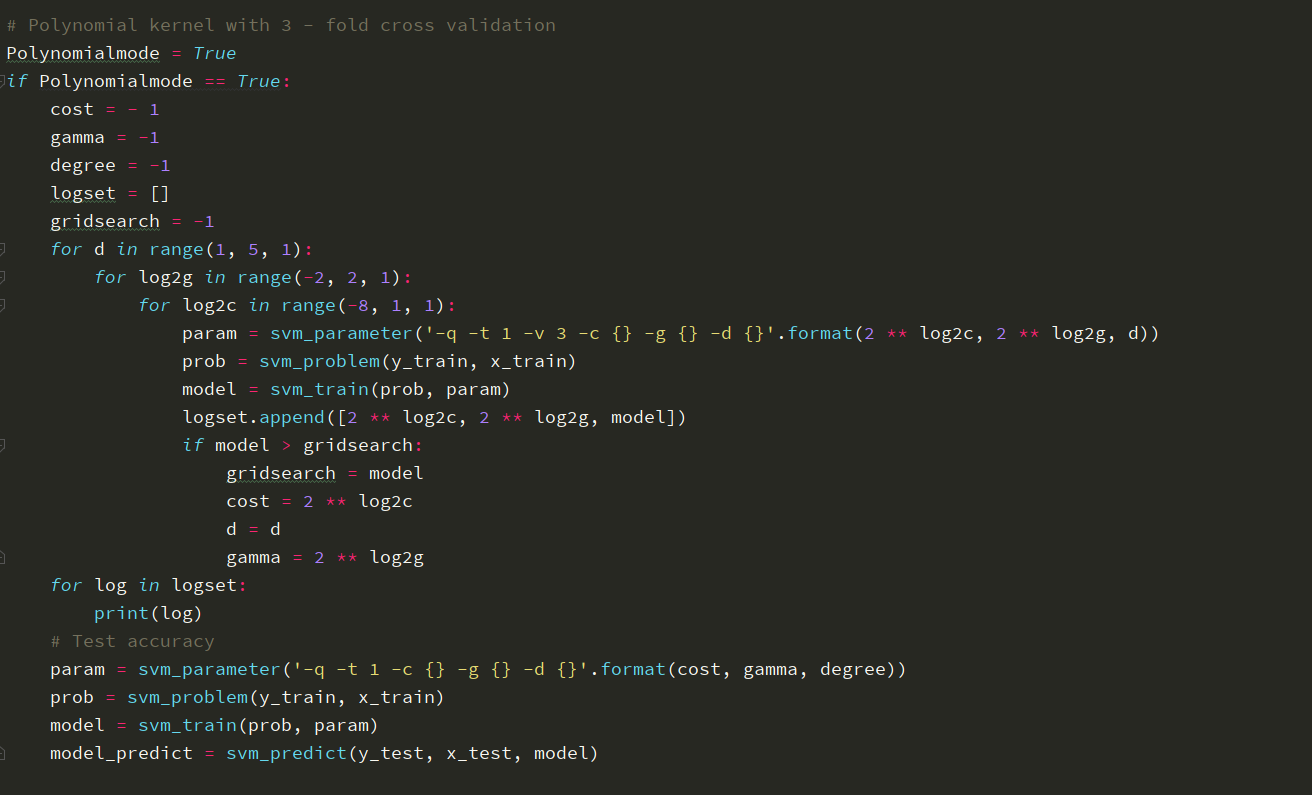
Result of finding parameter c, performance on cross – validation

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0.00390625 | 0.0078125 | 0.015625 | **0.03125** | 0.0625 | 0.125 | 0.25 | 0.5 | 1 |
| 96.3 | 96.82 | 96.899 | **97.1** | 97.08 | 96.76 | 96.32 | 96.06 | 96.24 |

* **c = 0.03125** gave the best performance on cross – validation
* Accuracy = **96%** (2400/2500) (classification)
* The accuracy is slightly higher than default setting
* The range of c is based on the suggestion of **libsvm** library. We firstly do the spare search to find a range of c, then make a fine-turn to get the better result

1. **Polynomial, 3 – folds cross – validation, searching for variable c, gamma and degree**

<code>



Result of finding parameter c, gamma and degree performance on cross – validation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| c g | 0.25 | 0.5 | 1 | 2 |
| 0.00390625 | 97.74 | 97.78 | 97.88 | 97.84 |
| 0.0078125 | 97.72 | 97.8 | 97.86 | 98.08 |
| 0.015625 | 98.04 | 97.98 | 97.88 | 97.88 |
| 0.03125 | 97.82 | 97.82 | 97.94 | 97.78 |
| 0.0625 | **98.06** | 97.619 | 97.96 | 97.619 |
| 0.125 | 97.82 | 97.92 | 97.89 | 98.0 |
| 0.25 | 97.92 | 97.94 | 97.88 | 97.98 |
| 0.5 | 97.76 | 97.84 | 98.02 | 97.8 |
| 1 | 97.86 | 97.72 | 97.89 | 97.94 |

* **c = 0.0125 and g = 0.25** gave the best performance on cross – validation
* Accuracy = **97.68%** (2442/2500)(classification)
* The accuracy is slightly higher than linear kernel
* The polynomial kernel required more time and memory for training compare to the linear kernel.

1. **RBF kernel with 3 – folds cross validation, searching for variable c and gamma**

<code>



Result of finding parameter c and gamma, performance on cross – validation

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| c g | 0.03125 | 0.0625 | 0.125 | 0.25 | 0.5 | 1 | 2 |
| 0.125 | 97.04 | 84.6 | 48.19 | 27.6 | 21.64 | 20.56 | 20.28 |
| 0.25 | 97.58 | 92.9 | 48.96 | 35.74 | 21.68 | 20.84 | 20.26 |
| 0.5 | 97.94 | 96.8 | 55.059 | 39.04 | 25.28 | 20.76 | 20.38 |
| 1 | 98.36 | 97.84 | 84.119 | 62.539 | 44.879 | 30.38 | 24.14 |
| 2 | 98.48 | 97.98 | 84.96 | 65.539 | 45.879 | 32.58 | 25.28 |
| 4 | 98.5 | 97.84 | 85.119 | 65.48 | 45.18 | 32.36 | 25.2 |
| 8 | 98.52 | 97.86 | 85.02 | 65.38 | 45.6 | 31.319 | 25.18 |
| 16 | 98.42 | 97.8 | 85.42 | 65.259 | 45.94 | 31.9 | 25.58 |
| 32 | 98.52 | 97.86 | 85.36 | 65.039 | 44.94 | 31.879 | 25.7 |
| 64 | **98.56** | 97.78 | 85.04 | 65.96 | 45.1 | 32.42 | 25.52 |
| 128 | 98.34 | 97.78 | 85.26 | 65.8 | 44.26 | 31.259 | 25.6 |
| 256 | 98.44 | 97.84 | 85.3 | 65.94 | 44.48 | 32.1 | 24.98 |

* **c = 64 and g = 0.03125** gave the best performance on cross – validation
* Accuracy = **98.52%** (2463/2500)(classification)
* The accuracy is slightly higher than linear kernel and polynomial kernel
* The polynomial kernel required more time and memory for training compare to the linear kernel and polynomial kernel.
* Since we may do pre-compute kernel, training time may be reduced.

1. **Combine linear and RBF kernel**



* Accuracy = **95.08%** (2377/2500)(classification)
* Compare with the previous kernel method

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Method | Linear | Polynomial | RBF | Linear + RBF |
| Accuracy | 96% | 97.68% | 98.52% | 95.08% |

* Using pre-compute kernel give the result faster than using build – in kernel functions when we do the grid search
* The RBF kernel has better accuracy than other method but slower in compute than other method, problem is also going with Polynomial compare to Linear Kernel. This can be trade – off for computation efficiency and accuracy.

If you have any concern about my code or other question don’t hesitate to contact me!

My email is: *haiphung106@gmail.com*