THE UNIVERSITY OF SYDNEY

School of Information Technologies
Faculty of Engineering & IT

ASSIGNMENT/PROJECT COVERSHEET - GROUP ASSESSMENT

Unit of Study: COMP5318 Machine Learning and Data Minning

Assignment name: Assignment 1 Classifier biulding

Tutorial time: 6pm to 9pm every Monday **Tutor name:** Prof Fabio Ramos

DECLARATION

We the undersigned declare that we have read and understood the *University of Sydney Academic Dishonesty and Plagiarism in Coursework Policy*, an, and except where specifically acknowledged, the work contained in this assignment/project is our own work, and has not been copied from other sources or been previously submitted for award or assessment.

We understand that failure to comply with the *Academic Dishonesty and Plagiarism in Coursework Policy* can lead to severe penalties as outlined under Chapter 8 of the *University of Sydney By-Law 1999* (as amended). These penalties may be imposed in cases where any significant portion of my submitted work has been copied without proper acknowledgement from other sources, including published works, the internet, existing programs, the work of other students, or work previously submitted for other awards or assessments.

We realise that we may be asked to identify those portions of the work contributed by each of us and required to demonstrate our individual knowledge of the relevant material by answering oral questions or by undertaking supplementary work, either written or in the laboratory, in order to arrive at the final assessment mark.

Project team members				
Student name	**Student ID**	**Participated**	**Agree to share**	**Signature**
1. Shaowei Zhang	470144491	Yes / No ✓	Yes/No ✓	Shaowei Zhang
2. Dongdong Zhang	470161133	Yes / No ✓	Yes / No ✓	Dongdong Zhang
3. Jiayu Ma	450056189	Yes / No ✓	Yes / No ✓	Jiayu Ma
4.		Yes / No	Yes / No	
5.		Yes / No	Yes / No	
6.		Yes / No	Yes / No	
7.		Yes / No	Yes / No	
8.		Yes / No	Yes / No	
9.		Yes / No	Yes / No	
10.		Yes / No	Yes / No	

SIT Building, J12
The University of Sydney
NSW 2006 Australia

T +61 2 9351 3423
F +61 2 9351 3838
E sit.info@sydney.edu.au
sydney.edu.au/it

ABN 15 211 513 464
CRICOS 00026A

1. Introduction

1.1 The aim of the study

There are three aims of this study.

The first one is to realize the mainstream methods of classifier algorithm. Basically, Support Vector Machine (SVM), Bayesian algorithm, K-nearest Neighbors (KNN) and Logistic Regression are four powerful classifiers supporting classification. These classifiers have both positive and negative features. For instance, SVM is able to provide a high accuracy rate. Bayesian algorithm is high-efficacy and easy-training while accuracy is lower. KNN and Logistic Regression are simple and easy to use with a high-cost computation and memory. Moreover, KNN cannot afford to compute high dimensional feature space. Considering the theoretical high accuracy of SVM, we are strongly interested in choosing SVM as our main algorithm.

Another aim of this study is to learn about the working principle of SVM. SVM is a supervised learning model which is widely used to classification and regression analysis. It works like a training machine which provides two options (positive or negative) for data set. Input data could be determined automatically if it belongs to a certain category or not. After detecting the first category, input record will be test if it belongs to the second one. In short, SVM is a binary classifier which could decide whether one data pertains to a category.

Additionally, it is a good opportunity to increase our practical ability. In this study, we have access to solve problems by learned theories. For example, the PCA method to reduce the size of data-set, the SVM classifier and so on.

This assignment aims to build a reasonable and effective classifier for the provided data set.

1.2 The importance of this study

Automatically classifier is one of the most significant and wide-used application in machine learning field. It is useful to understand and apply Classifier well. For example, it is widely used for multimedia recognition. Specifically, the concept of term frequency is applied to describe text values in this assignment. Similarly, the pixel frequency could be used to define images.

Additionally, Classifier is not only used in IT industry, but also could be applied in biology field. For instance, Ben-Hur and his colleagues (2008) shows that SVM classifier could be used to predict the gene structure and its function, interactions, and role in disease. Moreover, it could figure out the organisms DNA/RNA, Adaptations and Embryonic development as well.

2. Methods

2.1 Pre-processing for data

2.1.1 Standard deviation sorting

Firstly，it needs to acquire index of standard deviation by rows. Standard deviation is computed by rows, then reversely orders values and select top t rows. “t” means the number of rows with the highest standard deviation:

$$\sigma(X_j) = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{n-1}} \qquad (1)$$

2.1.2 Using PCA to reduce data dimensions

The way to reduce running time is to select less and meaningful rows in data matrix. For example, Bro and Smilde (2014) used this method to reduce the samples.

The original data source seems messy and hard to manage. Normalization is used to achieve a more optimum dataset. For further detail, the formula is showed below:

$$\mathrm{a}_i = a_{ij} - \mu \qquad (2)$$

PCA could reduce the dimension of a data set by computing the principal components of a dataset.

The second step is to create data matrix. In order to be convenient to calculate, the first step is creating data matrix (***X***) by combining training dataset and test dataset:

$$\boldsymbol{X} = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mn} \end{pmatrix}, m, n\ is\ row, column \qquad (3)$$

The third step is computing covariance (∑) matrix that base on column of tf-idf data matrix:

$$cov(X_i, X_j) = \frac{\sum_{i=1}^{n}(X_i - \bar{X}_i)(X_j - \bar{X}_j)}{(n-1)}, i, j \le n$$

$$\sum = \begin{pmatrix} cov(X_1, X_1) & \cdots & cov(X_1, X_n) \\ \vdots & \ddots & \vdots \\ cov(X_n, X_1) & \cdots & cov(X_n, X_n) \end{pmatrix} \qquad (4)$$

Next step is to obtain eigenvalues and eigenvectors. First, ecompose the covariance matrix’s to get n eigenvalues (λ) and n eigenvectors (**v**). Second, sort eigenvalues in a descending order. Third, select eigenvectors of top p eigenvalues:

λ_i	V_i	
max	$v_{\lambda max}$	
.	.	P
.	.	
.	.	
min	$v_{\lambda min}$	

Table 1. Eigenvalues and eigenvectors

After selecting eigenvalues, eigenvectors matrix (***P***) are:

$$\boldsymbol{P} = (\boldsymbol{v}_{\lambda max1} \quad \cdots \quad \boldsymbol{v}_{\lambda maxp}) \qquad (5)$$

Then we can get reduced data matrix (***Q***) after multiplying data matrix (***X***) with eigenvectors matrix (***P***):

$$\boldsymbol{Q} = \boldsymbol{X} \cdot \boldsymbol{P} \qquad (6)$$

2.2 Classifier

2.2.1 SVM theory

Theory of Support Vector Machine (SVM)

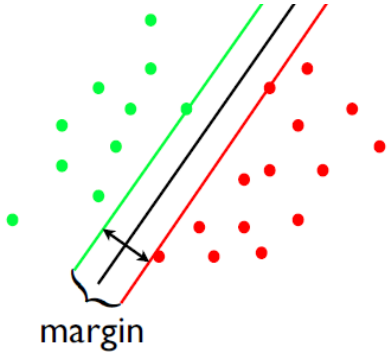

Figure 1. The line used to separate

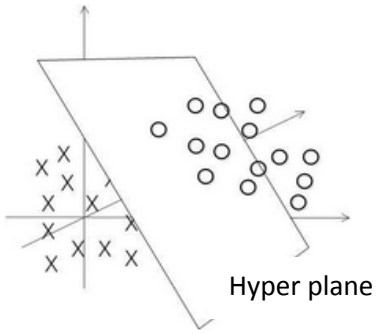

Figure 2. The hyper plane used to separate

SVM is a classifier used to class data points into 2 classes, either 1 or -1. If data points X only have one dimension, SVM can find a line to separate points in a 2D plane (figure 1). If the data points X have more than one dimension, SVM can find a hyper plane to separate the points (figure 2). The line of the hyper plane can be represented by:

$$f(x) = y = \omega^T x + b \qquad (7)$$

If two classes are represented by 1 and -1, then points x_i will be Class 1 if their $x_i > 0$, points x_j will be Class 2 if their $x_j < 0$.

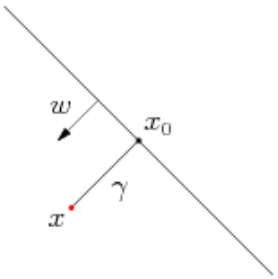

Figure 3. The distance γ

To separate the points, we need to know the distance γ between points and line/hyper plane. We can get the distance by multiplying ω^T on both sides of formula 8:

$$\mathrm{x} = x_0 + \gamma \frac{\omega}{\|\omega\|} \qquad (8)$$

Due to $y = \omega^T x_0 + b = 0$, we can get the distance $\gamma = \frac{\omega^T x + b}{\|\omega\|} = \frac{f(x)}{\|\omega\|}$. To get the absolute value, we can multiply its y value and call the result as Geometrical Distance:

$$\tilde{\gamma} = \mathrm{y}\frac{\omega^T x + b}{\|\omega\|} = \frac{y f(x)}{\|\omega\|} \qquad (9)$$

To have more confidence with the classification, we need to make sure the margin can be as large as possible, which means we need to get $\max\tilde{y}$. Define the half of the $\max\tilde{y}$ as 1, we find that we need to find the answer of $\max\frac{1}{\|\omega\|}$ and meanwhile, every point should obey the condition:

$$y_i(\omega^T x_i + b) \geq 1, i = 1, \cdots, n \qquad (10)$$

The problem can also become:

$$min\frac{1}{2}\|\omega\|^2 \quad s.t., y_i(\omega^T x_i + b) \geq 1, i = 1, \cdots, n \qquad (11)$$

To get the best solution of formula 11, we need to construct a Lagrange Function using α_i as parameter and try to get its maximum value $\theta(\omega)$:

$$L(\omega, b, a) = \frac{1}{2}\|\omega\|^2 - \sum_{i=1}^{n} \alpha_i (y_i(\omega^T x_i + b) - 1) \qquad (12)$$

$$\theta(\omega) = \max_{\alpha_i \geq 0} L(\omega, b, a) \qquad (13)$$

Now that formula 11 becomes:

$$\min_{\omega,b}\theta(\omega) = \min_{\omega,b}\max_{\alpha_i \geq 0} L(\omega, b, a) \qquad (14)$$

According to the duality of Lagrange Function, formula 14 can also be written as formula 15 on the condition called KTT:

$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \qquad (15)$$

KTT is $0 \leq \alpha_i \leq C, i = 1, \ldots, n$ and $\sum_{i=1}^{n} a_i y_i = 0$

Now the problem of SVM finally becomes:

$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

$$\text{s.t.}, 0 \leq \alpha_i \leq C, i = 1, \ldots, n$$

$$\sum_{i=1}^{n} a_i y_i = 0 \qquad (16)$$

We can use an algorithm called SMO to solve this. The main idea of SMO is that alphas become the main vector which has several options from α_i to α_j. Randomly select α_i and α_j and set the rest of α as constant. Under the condition that $\sum_{i=1}^{n} a_i y_i = 0$ and $0 \leq \alpha_i \leq C$, modify the value of α_i and α_j in order to achieve a maximum value in formula 16. Iteratively change another pair of (α_i, α_j) and make the formula 16 maximum when alpha still meets the condition. Until all alpha has been done, this step finish.

2.2.2 Optimizations

2.2.2.1 Full Platt SMO

Simplified SMO selects alphas randomly which results in a repetitive and time consuming calculation. Therefore, the Full Platt SMO, a more effective approach, is utilized to deal with the data. This algorithm is described by Tong and Koller (2001).

Full Platt SMO has two kinds of loops which are an outer loop for choosing the first alpha and an inner loop for choosing the second one. Since the second alpha is chosen in a way that will maximize the step size during optimization, the computation speed is effectively increased.

2.2.2.2 Using kernels for complex data

Another way used to optimize the algorithm is using kernels to mapping data to higher dimensions. Kernels work as a wrapper which aims to achieve a simpler formatting. Specifically, Gaussian version is:

$$\text{k(x, y)} = \exp\left(\frac{-\|x-y\|^2}{2\sigma^2}\right) \qquad (17)$$

Sigma is a parameter which has an effect on accuracy. In this assignment, we choose sigma as 5 and it works comparatively well. Gaussian version is a highly flexible algorithm in which Sigma could be defined by users. That is the reason we choose Gaussian version to optimize the processing.

2.2.2.3 Using PCA to reduce data dimensionality

Data dimensionality reduction is another effective way to reduce computations. Assuming that we have an n*n matrix, the complexity is $O(n^2)$. If the size of matrix could be reduced, computation time will be greatly decreased.

2.2.3 Flow chart of computation algorithm

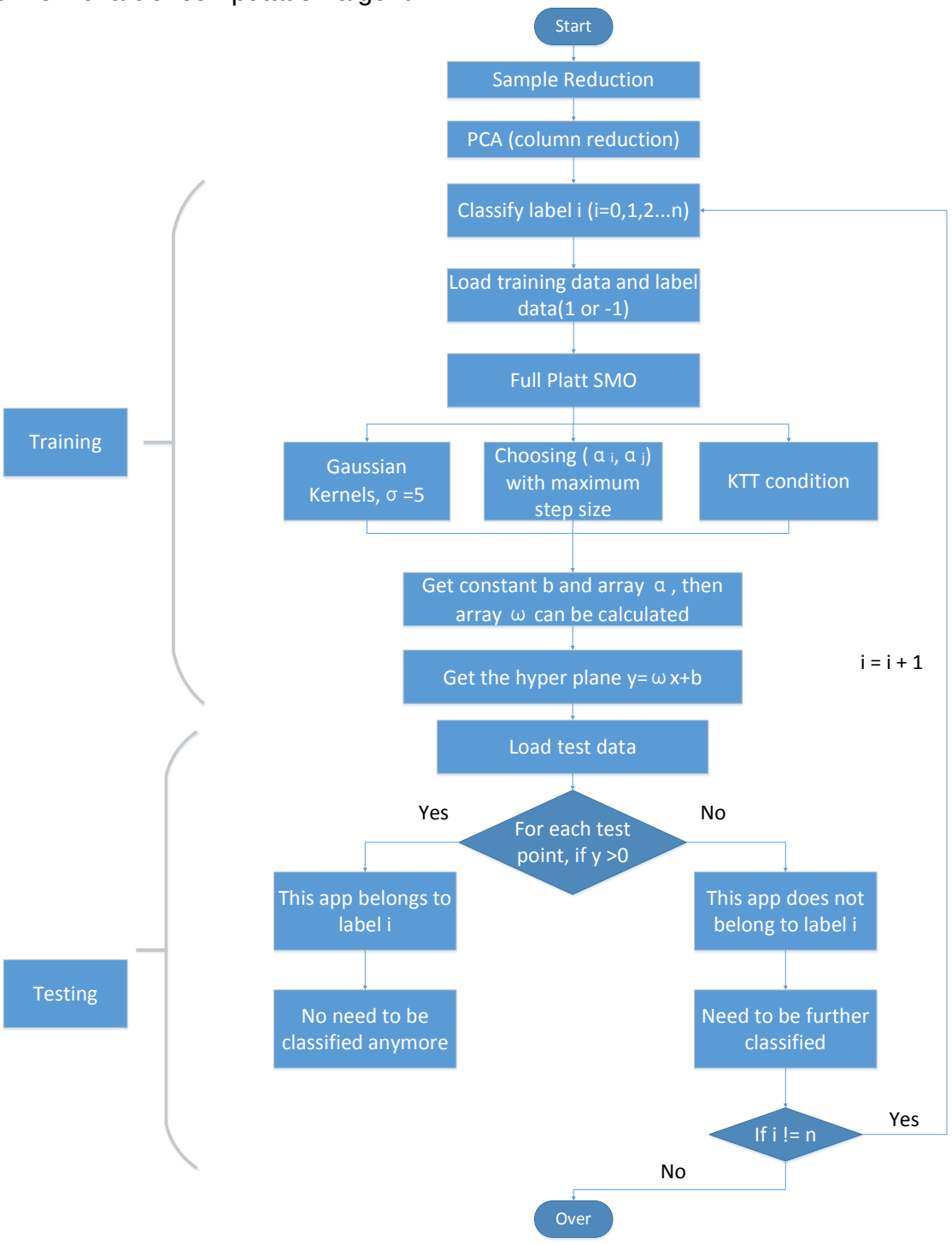

Figure 4 Flow chart of computation algorithm

3. Experiments and results

3.1 Accuracy

Hardware and software specifications of the computer that we used for performance evaluations are in Appendix I.

We use formula 18 to measure the whole accuracy of our classifier.

$$\text{Precession } (p) = \frac{TP}{TP+FP} \qquad (18)$$

TP represents that number of correct prediction. FP represents number of false prediction. For example, if we are going to predict 100 apps. If only 30 of them are predicted correctly, then TP equals 30 and FP equals 70. The accuracy will come to 30%.

3.2 Extensive analysis

As one of the disadvantage of SVM, the more training samples are, the more running time will be. According to our experience, if the number of training samples are over 3000, the running time will be unacceptable. Also, we find that the number of columns in our training dataset only have limit effect on the accuracy.

Therefore, we only change the number of samples in training dataset in every round. Unfortunately, after three rounds of 500 samples, 1000 samples and 1500 samples, accuracy rate still could not reach a satisfying high level. However, we can find that we can have a higher accuracy with more training samples. So theoretically, SVM perhaps can indeed get a high accuracy when dealing with a large dataset, but it will be very struggling with a very long running time. The details are showed as below:

Test No.	Accuracy rate		
	500 samples	1000 samples	1500 samples
1	8.10	11.95	21.13
2	3.75	19.25	15.60
3	9.20	7.71	15.44
4	11.87	22.67	22.34
5	7.46	14.51	28.12
6	20.91	8.46	17.92
7	8.75	22.83	18.46
8	3.30	10.38	28.82
9	7.52	7.78	23.87
10	19.15	19.31	24.71
avg	10.00	14.49	21.64

Table 2. First three test results

The highest average accuracy rate is approximately 20 which is still not a satisfied result. To improve the accuracy rate, increasing the number of samples is necessary, so we raised the sample number to 2500 which seem to be the limit of SVM to predict the

labels of original “test_data.csv”. Furthermore, the increasing number of PCA columns will help to achieve a more identified tf-idf value which will lead to a more accurate results as well, so we raised the column number to 5000 to construct the classifier.

However, larger data source means more running time and we spent approximately 3 hours to predict the labels. Considering the limited time provided, one feasible solution is to look for hardware support. That means if a distributed system is available and accessible, our scheme will works more efficiently with high accuracy rate.

4. Discussion

Meaningful and relevant personal reflection

4.1 Solution to inefficiency

When dealing with a comparatively large dataset, it is really time consuming. Therefore, it is in great demand to reduce the run time and increase the efficiency. After deep consideration and discussion, in spite of our comparatively low accuracy, the ways we applied to decrease computing time still should be stick to.

Firstly, PCA still should be used to reduce data dimensionality. It is a mature and effective approach which help us to reduce 13627 columns to 500. It really increase the computing speed and save time. The key point should be finding the balance between accuracy and running time. For instance, 500 columns may be really fast but at the meantime we lose many important features of our samples. On the other hand, if we use 13000 columns, the features will be enough but the running time will increase rapidly.

Further method is to optimize the algorithm to solve SVM. Full Platt SMO algorithm may not be a best way because it has too many loops. Maybe we can choose other ways with less loops. We need to try our best to avoid repeated computation.

The last method to improve run speed is utilization of kernels. Kernels plays an really important role in Full Platt SMO and we believe it will still be necessary for other algorithms because Kernels maps data into a higher dimensions which could improve efficiency to a large extent.

4.2 Efforts to increase accuracy

Apart from the solutions mentioned above, we have also do a lot of detailed jobs. For instance,in order to achieve a higher accurate rate, we have tried several times to determine the value of sigma used in Gaussian Kernels. After repeatedly test, it is finally fixed as 5. Because the result is more accurate when sigma equals to 5. We believe there are many other jobs can be done. For example, the high boundary of alphas C and the tolerance of errors can also be optimized to achieve a high accuracy.

5. Conclusions and future work

5.1 Meaningful conclusions based on results

SVM contributes to accurate results while it spends too much time. It is an appreciate method for assignment, but is not a good way for processing large-scaled data in practical works. Comparing to SVM, Bayesian Classifier or Logistic Regression is more applicable and feasible approach for big data classification.

5.2 Meaningful future work suggested

Although SVM is a time-consuming way to classify a large scaled data source, it still powerful if it could be used with a distributed system.

6. Other

In the large data files, there are plenty of redundant data dimensions which have negative influence on computing running time. Therefore, PCA is a good way to solve this problem, we highly reduce the running time by applying PCA.

According to methodology of SVM, it can only directly classify two classes rather than other methods, so SVM is theoretically more accuracy than others. Due to the slides, SVM are not suitable to be applied to large dataset, but before this assignment we do not know exactly how large is large. After test, we think 3000*2000 is large when the computer is normal laptop. However, if SVM runs in the workstations, it will computes more accuracy and quickly.

Reference

Ben-Hur, A., Ong, C.S., Sonnenburg, S., Schölkopf, B. and Rätsch, G., 2008. Support vector machines and kernels for computational biology. *PLoS Comput Biol*, *4*(10), p.e1000173.

Bro, R. and Smilde, A.K., 2014. Principal component analysis. *Analytical Methods*, *6*(9), pp.2812-2831.

Harrington, P., 2012. *Machine learning in action* (Vol. 5). Greenwich, CT: Manning.

Tong, S. and Koller, D., 2001. Support vector machine active learning with applications to text classification. *Journal of machine learning research*, *2*(Nov), pp.45-66.

Appendix

Hardware and Software Specifications:

Software:

Python3

Hardware:

a) For “finalsvm.py” (Must use MacBook, or there will be memory error.)
Computer: MacBook Pro

Processor: 2GHz Intel Core i5

Memory: 8 GB

b) For other python files:
Computer: Thinkpad E450

Processor: 2.2GHz Intel Core i5

Memory: 8 GB

Appendix II

How to run the code:

There’re three python files.

The main file is “finalsvm.py”. It can do all the jobs we described in our report. The Full Platt SMO code part is referenced from *Machine Learning in Action* (Harrington, P., 2012)

If you want to classify your test dataset, code in:

“classify(datafilename,lablefilename,testfilename,C,toler,maxIter,rows,columns,kTup)”

“datafilename” represents your training dataset (csv file); “labelfilename” represents your training lables (csv file); “testfilename” represents your test dataset (csv file); “C” represents the high boundary of alphas and in this assignment, 0.6 will be OK; “toler” represents the tolerance of errors during iteration and in this assignment, 0.001 will be OK; “maxIter” represents the maximum iteration times you can accept and in this assignment, 50 will be OK; “rows” represents how many samples you want to keep after reduction; “columns” represents how many columns you want to keep after PCA; “kTup” represents what kind of kernel you want to use and in this assignment we only defined Gaussian version. You should code in for example “kTup = (‘rbf’, 5)” here. ‘rbf’ represents Gaussian version and 5 mean the value of σ you want. Here σ=5 will have a comparatively good result. You can code in as Figure 1 below:

```
#####Main#############################################################################################
classify('training_data.csv','training_labels.csv','test_data.csv',0.6,0.001,50,3000,3000,kTup=('rbf',5))
```

Figure 1 An example of code

After “finalsvm.py” has been run, there will be three outputs, “predicted_labels.csv”, “trainingdataMat.csv” and “trainingappnameMat.csv”. “predicted_labels.csv” contains the predicted labels of test dataset and their corresponding application names. “trainingdataMat.csv” contains the training dataset after sample and PCA reduction and “trainingappnameMat.csv” contains its corresponding application names.

There’re another two python files, “makesmall.py” and “evaluation.py”. They can be used for evaluation.

“makesmall.py” is used to cut out the training dataset and test dataset for our N-fold cross evaluation. After “finalsvm.py” has been run, “trainingdataMat.csv” and “trainingappnameMat.csv” will be written. For example, the samples in “trainingdataMat.csv” are 20000, we can use “makesmall.py” to cut out a dataset with 18000 samples for training and to cut out a dataset with 2000 samples for testing. Repeat it for N times and we can perform our N-fold cross evaluation.

“evaluation.py” is used to evaluate the precise of prediction. It’s pretty much like “finialsvm.py”, so you need to code in the similar codes as showed in Figure 2:

```
######Main############################################################################################
classifiedlabel=classify('smalldata.csv','training_labels.csv','smalltestdata.csv',0.6,0.001,50,kTup=('rbf',5))
calcaccu('training_labels.csv',classifiedlabel)
```

Figure 2 An example of code

‘smalldata.csv’ still represents the training dataset and ‘smalltestdata’ still represents the test dataset. We need “Classify” Function to return the prediction matrix this time (“classifiedlabel” in Figure2) and we can use this prediction matrix to calculate precise. The code is:

“calcaccu(lablefilename,classifiedlabel)”

“labelname” represents the training labels (csv file); “classifiedlabel” represent the prediction matrix (matrix).