CS412 HW2

Wangfei Wang

1 Feature Extraction

Python package "sklearn" was used. **Figure 2.1** shows the two features extracted by Kernel PCA with polynomial kernel with degree 3.

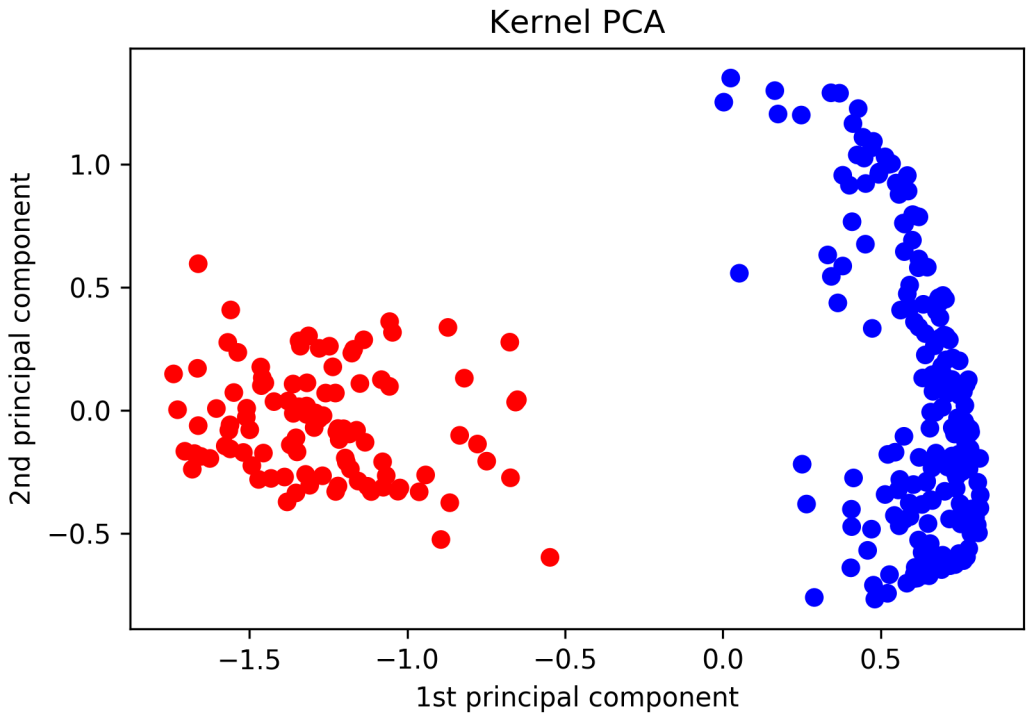


Figure 2.1. Kernel PCA using features mean intensity and intensity standard deviation (polynomial kernel with degree 3).

a) Compare kernel PCA features with the features you selected from HW1. Do these features seem to better separate the data?

Yes. The features extracted by kernel PCA seems to be separating the data better. The features I used for HW1 are mean intensity and intensity standard deviation. Although these two features separate data pretty well, the gap between two groups are pretty narrow (in HW1, not shown here). But in this kPCA case, the two groups are widely separated and are thus better (see **Figure 2.1**).

b) Give the explained variance ratio for each of the two feature extractions given above. Is there one of the methods which explains more variance than the other? Is this what you expect? Explain your answer.

The variance explained by the two features extracted by model is calculated as:

$$\frac{\text{variance of feature 1} + \text{variance of feature 2}}{\sum_{i=1}^{256} \text{variance of feature i}}$$

Total variance of the original 256 features is 71.521. For kernel PCA case, variances explained by the two features are 0.835 and 0.225. The variance explained by two features is

$$\frac{(0.835 + 0.225)}{71.521} = 0.0148$$

Similarly, the variances explained by the two features in HW1 (mean intensity and intensity standard deviation) is calculated as

$$\frac{(0.127 + 0.175)}{71.521} = 0.00422$$

The variance explained by features in kernel PCA is much bigger than that explained by features used in HW1. This is as expected because kernel PCA extracted two features that better separated the two groups, meaning the two features extracted by kernel PCA better represent the original data.

2 Logistic Regression

a) Plot the decision region for your 2D space with a logistic regression solver where c is 0.01. You should reuse the region plotting code from HW1. Label this figure 2.2. Unless specified, use L2 regularization.
See **Figure 2.2**. See discussion in c).

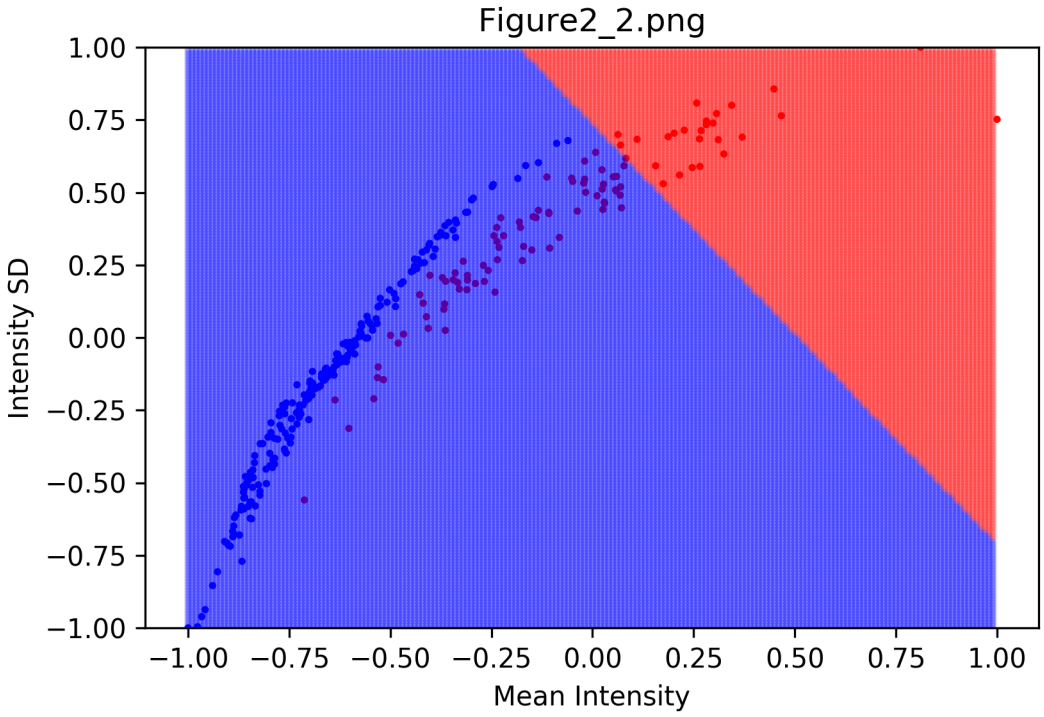


Figure 2.2. Decision region for 2D space with logistic regression (L2 penalty, c = 0.01).

b) Plot the decision region for your 2D space as above but with C = 2.0. Label this figure 2.3.
See **Figure 2.3**. See discussion in c).

c) Graduate student question: Repeat the experiment for Figure 2.2 and 2.3 using L1 regularization. Does this regularization method make it more or less likely for the model to overfit the data. If you don't think there is any overfitting, defend your answer.

See **Figures (a) and (b)** below. Two features used in HW1 are mean intensity and intensity standard deviation. The parameter C in logistic regression model is the inverse of the strength of the regularization. So the smaller C is, the bigger the model penalize the coefficient to prevent overfitting.

I don't think any of these four models is overfitting. Because an overfit model will have extremely low training error but a high testing error. But none of these models has this pattern. With a simple linear separater like logistic regression, I don't actually think overfitting

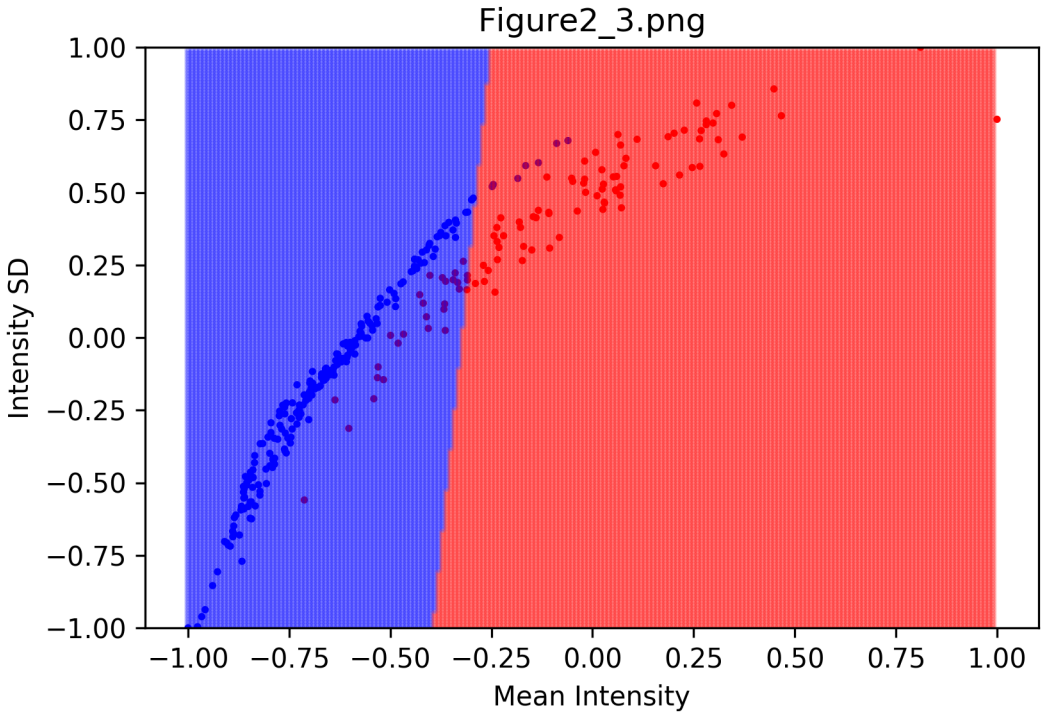
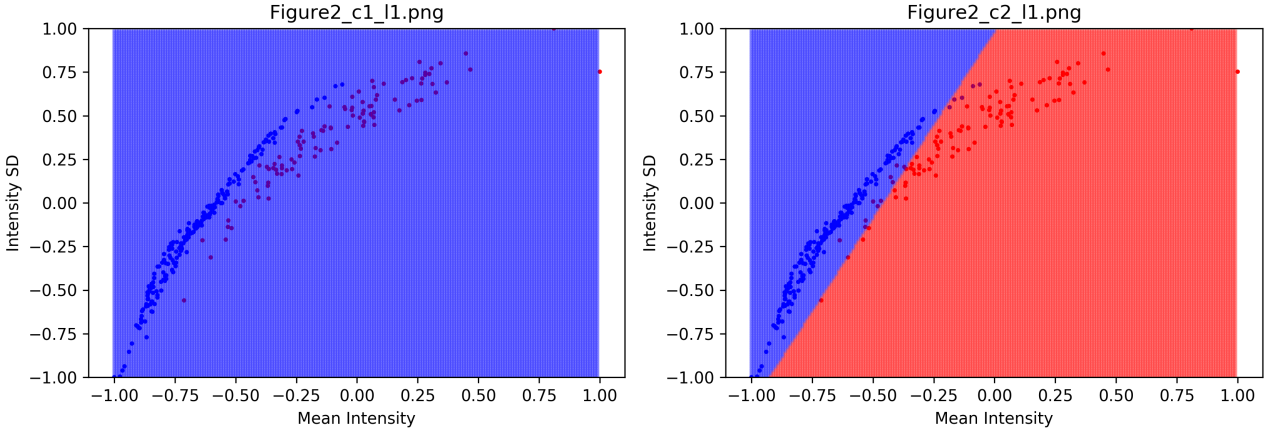


Figure 2.3. Decision region for 2D space with logistic regression (L2 penalty, c = 2.0).



(a) Decision region for 2D space with logistic regression (L1 penalty, c = 0.01)

(b) Decision region for 2D space with logistic regression (L1 penalty, c = 2.0)

is a concern.

Comparison of the four models:

Overall, L1 penalty with C = 2 performs the best among all these four logistic regression models. Lasso shrinkage (L1) tends to shrink the less important features' coefficients to zero together. In this homework, the feature number is not big and when we penalize too much

(C = 0.01 case), the model is predicting all data into one single group.

Logistic Regression Model	C = 0.01, L2	C = 2, L2	0.01, L1	C = 2, L1
Training Error	0.247	0.109	0.330	0.048
Testing Error	0.214	0.101	0.363b	0.040
10-fold CV Error (Mean)	0.250	0.119	0.33	0.055

Table 1: Performance of four logistic regression models with varying C and type of penalty.

3 Support Vector Machine

a) On your 2D data from HW1, find the cross-validation errors for all values of c from 0.01 to 100. Your x axis should have logarithmic scale and you should have at least 20 data points. Label this figure 2.4.
See **Figure 2.4**. Cross validation errors decrease with the increase of C between 0 and 10 and then stablize.

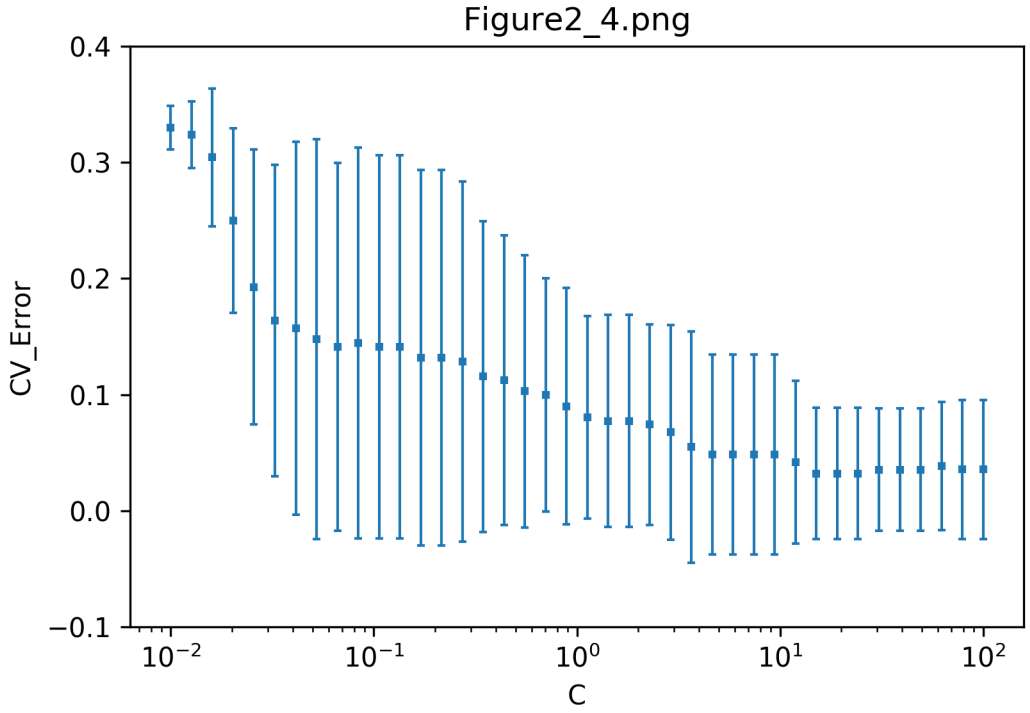


Figure 2.4. Cross validation error for different c values in SVM on 2D data (linear, soft margin SVM; c from 0.01 to 100, log scale).

b) Repeat the above experiment for the 256D degree data. Label this figure 2.5.
See **Figure 2.5**. Cross validation errors are all 0 across all C's.

c) Find the value of c from Figure 2.4 which has the lowest cross validation error and plot the decision region in your 2D space. Label this 2.6 Your x and y variables should be the same ones you chose in HW1.
See **Figure 2.6**.

d) Now, repeat the experiment for the polynomial kernel model for your 2D data. Let the degree of the kernel be each of 2,5,10,20. Give the value of c for each of these kernel degrees which gives the lowest cross validation error. Explain the tradeoff between the 'degree' and 'c' as far as overfitting is concerned.

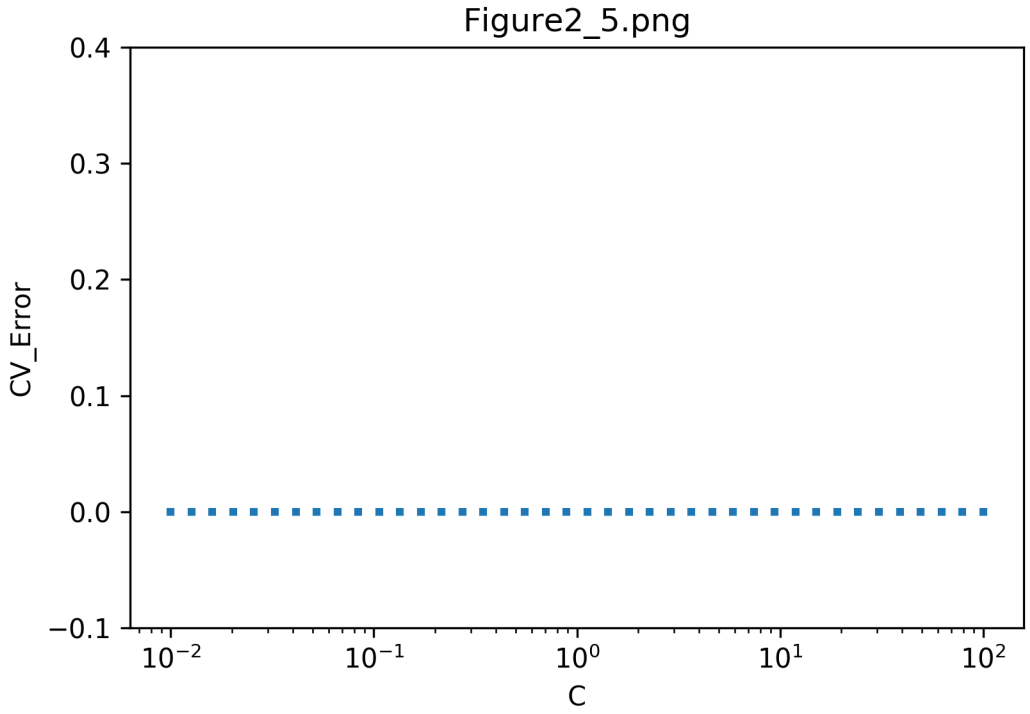


Figure 2.5. Cross validation error for different c values in SVM on 256D data (linear, soft margin SVM; c from 0.01 to 100, log scale)

The cross validation errors for different c values in polynomial kernel SVM on 2D data were plotted in **Figure 2.7**. Degree for polynomial kernel was chosen as 2, 5, 10 and 20. c ranges from 0.01 to 100. The optimal c was chosen at the smallest upper bound of the 95% confidence interval of the cross validation errors across all c's. The decision region for each degree and their corresponding optimal c was plotted in **Figure 2.8**.

The optimal c values for different polynomial kernel SVMs are:

Degree of polynomial	2	5	10	20
Optimal c	38.882	100.0	-	-

Table 2: Optimal c's for different degrees in polynomial SVM models. When degrees =10 and 20, the CV errors are immune to the change of c.

This 'degree' and 'c' tradeoff can be explained as:

Generally, when polynomial degree increases, the model becomes more complicated. Then c needs to kick in to control the potential overfit of the model. The bigger c is, the smaller the margin is considered, and the fewer support vectors are. Then a relatively big c would ameliorate the overfit of a model. Therefore, in **Table 2**, we can see, degree of 5 has a bigger c than degree of 2. However, when the model has big degree of polynomial, such as 10 and 20, the models become very complicated. In this case, no c would actually work to control the complexity of the model.

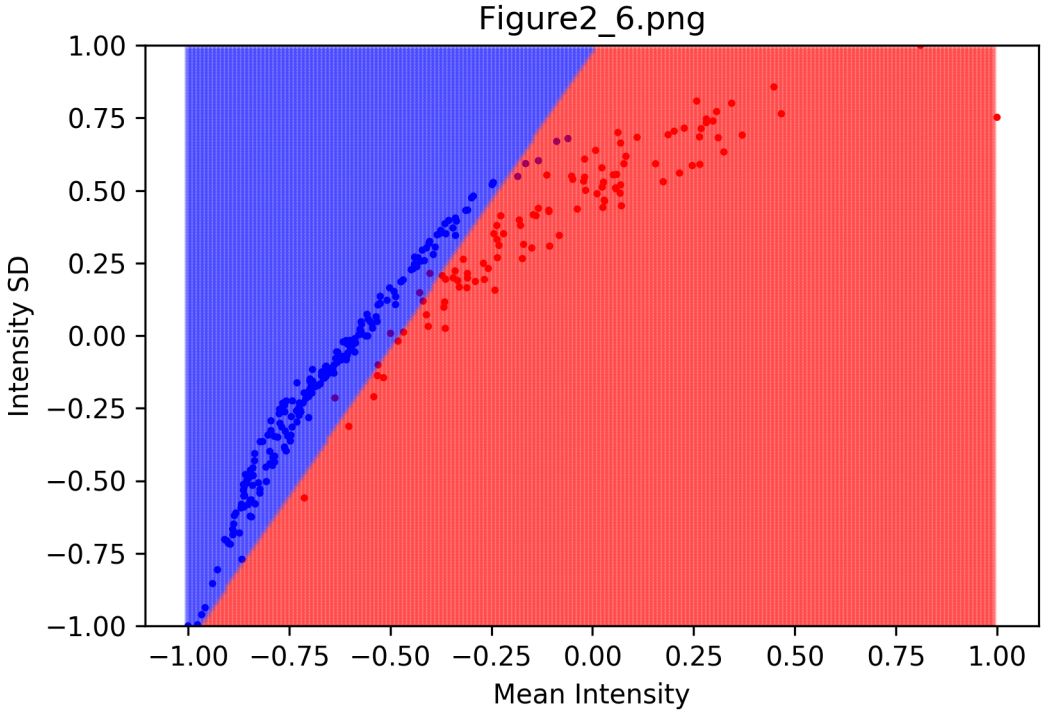


Figure 2.6. Decision region for 2D space with linear SVM with optimal c = 30.703.

e) Using your 2D data from HW1, find the values for 'degree' and 'c' that you believe finds the best fit for the model. Plot the decision region for this model (From figure 2.6) and explain why you think this model is best. Support your conclusion with data.

The upper bound of the 95% confidence interval of the cross validation error reaches its minimum at degree = 2 and c = 38.882 by comparing all the cross validation errors achieved by the different parameters listed in **Table 2**. The decision region on 2D data is shown in **Figure 2.8 top left subplot**. We can also pick the best 'degree' and 'c' by comparing the four plots in **Figure 2.8**. Since c's are already optimal for the corresponding degree, we can just compare the different degrees. For degree 10 and 20, the polynomial SVM models classify all the data set into one single group, causing big cv error (**Figure 2.7**). Polynomial with degree = 5 and its optimal c also mis-classified many training data. Therefore, in comparison, although degree = 2 and c = 38.882 is overfitting, it still outperforms the other three model in this experiment.

Graduate student question: Provide two graphs of decision regions for SVC models in the 2D space. One should have evidence of overfitting, and the other should have evidence of underfitting. Explain the parameters that lead to each of the graphs and their cross-validation errors.

An example of underfitting and overfitting model is shown as below. Two models were fit using 'rbf' as kernels and c was set to 1, gamma was set to 1 and 200 respectively. The training error and test errors are shown below in **Table 3**. In the underfitting model, the training error is relatively high and is higher than test error. In the overfitting model, the

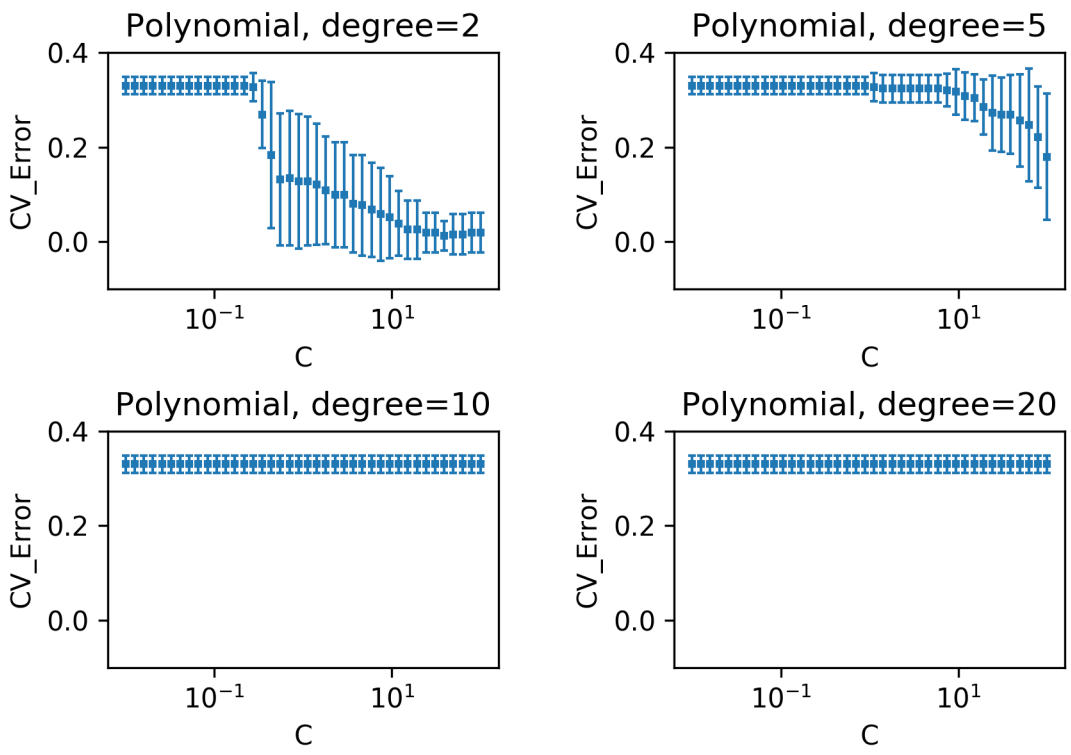


Figure 2.7. Cross validation error for different c values in polynomial kernel SVM on 2D data (polynomial kernel degree = 2, 5, 10, 20;c from 0.01 to 100, log scale).

training error is 0 while the test error is bigger than 0.

Model	Underfitting	Overfitting
Training Error	0.0577	0.000
Test Error	0.0376	0.0120
10-fold CV Error (Mean)	0.0645	0.00645

Table 3: Performance of two models. Left: underfitting and right: overfitting.

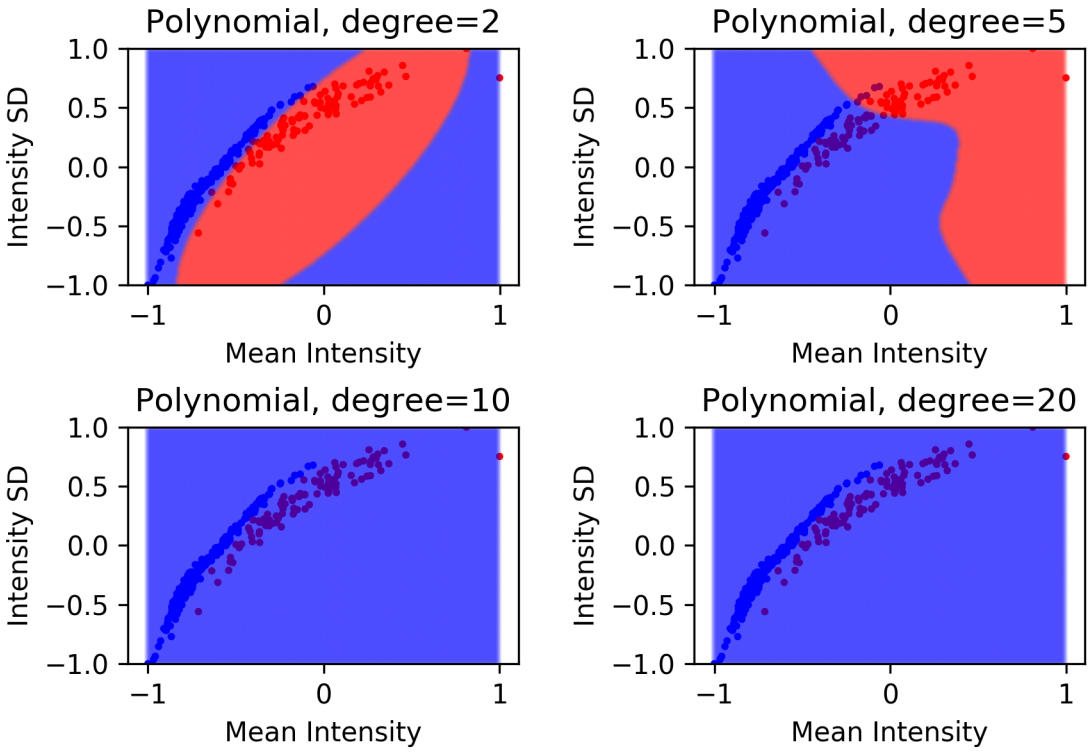


Figure 2.8. Decision regions for polynomial kernel SMV with different degrees and their corresponding optimal c's (shown in Table 2).

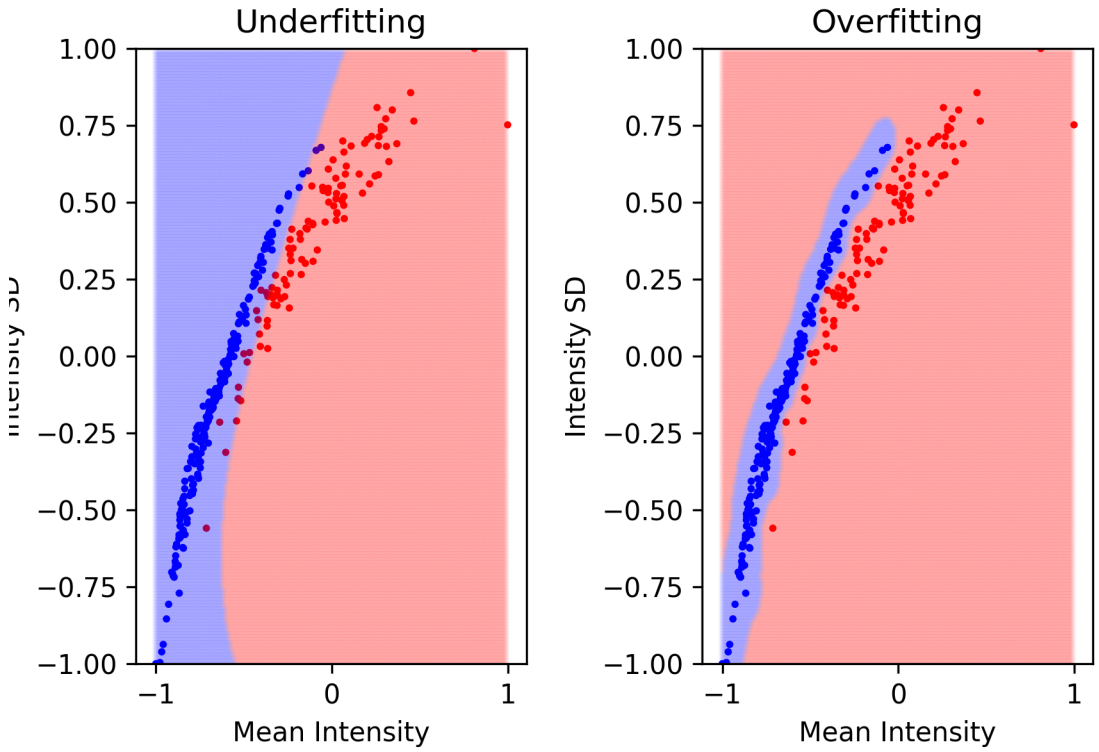


Figure 2.9. Radial as kernel, C = 1. Left: underfitting ($\gamma = 1$); Right: overfitting ($\gamma = 200$).

4 Extra Credit

For the kernel support vector classifiers (both radial and polynomial), examine the parameter gamma. Conduct experiments for different values of gamma, c and degree and try to determine the relationship between gamma and over/under fitting for the model. Support your explanation with figures.

4.1 SVM with 'radial' kernel

Parameter gamma (γ) in 'radial' kernel plays a role of controlling how far the influence of a single training data reaches. It is proportional to the inverse of variance in gaussian model and therefore, the low value of gamma means 'far' and big value of gamma means 'close'.

$$K(\boldsymbol{x}, \boldsymbol{x'}) = exp(-\gamma \left\| \boldsymbol{x} - \boldsymbol{x'} \right\|^2)$$

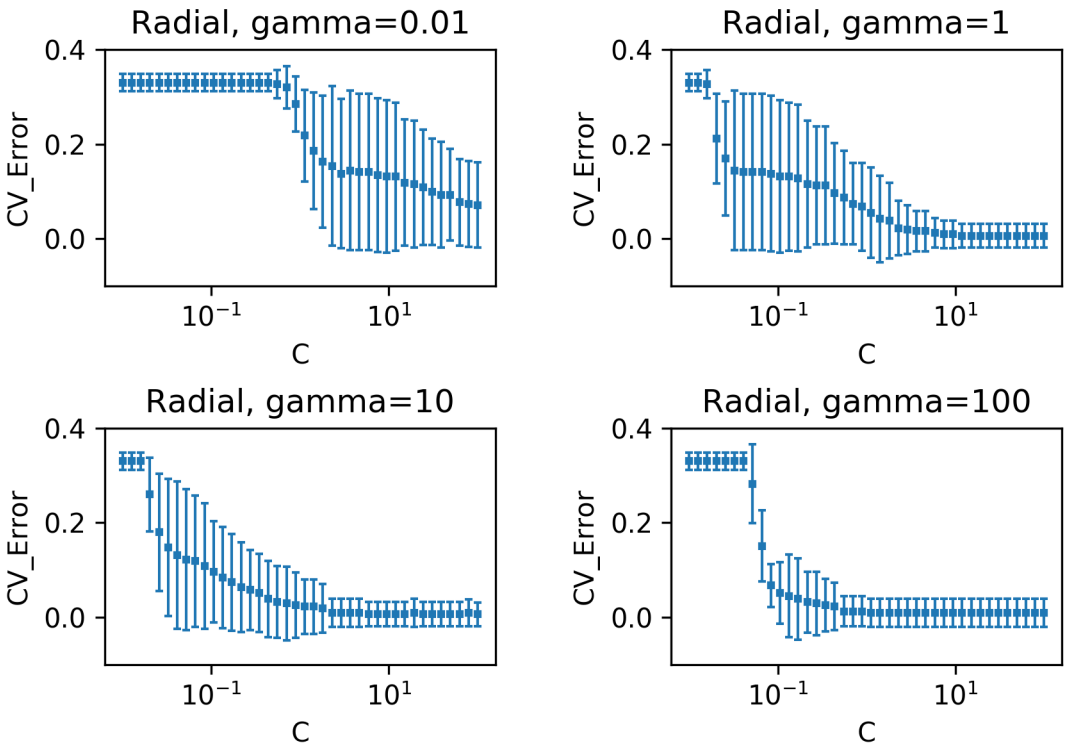


Figure 2.10. Cross validation error for different gamma in radial kernel SVM on 2D data ($\gamma = 0.01, 1, 10, 100$).

Optimal C's were chosen based on the method described before. As shown in **Figure 2.11**, the data is pretty sensitive to the choice of γ. When γ is big, the radius of area of influence only includes the support vectors itself (**Figure 2.11 bottom**, overfitting). Sometimes when γ is too big, no C will be able to prevent overfitting. While when γ is too small, the radius cannot capture all the shape of the training data (**Figure 2.11 top left**, underfitting). **Figure 2.11 top right** seems to be a pretty good model.

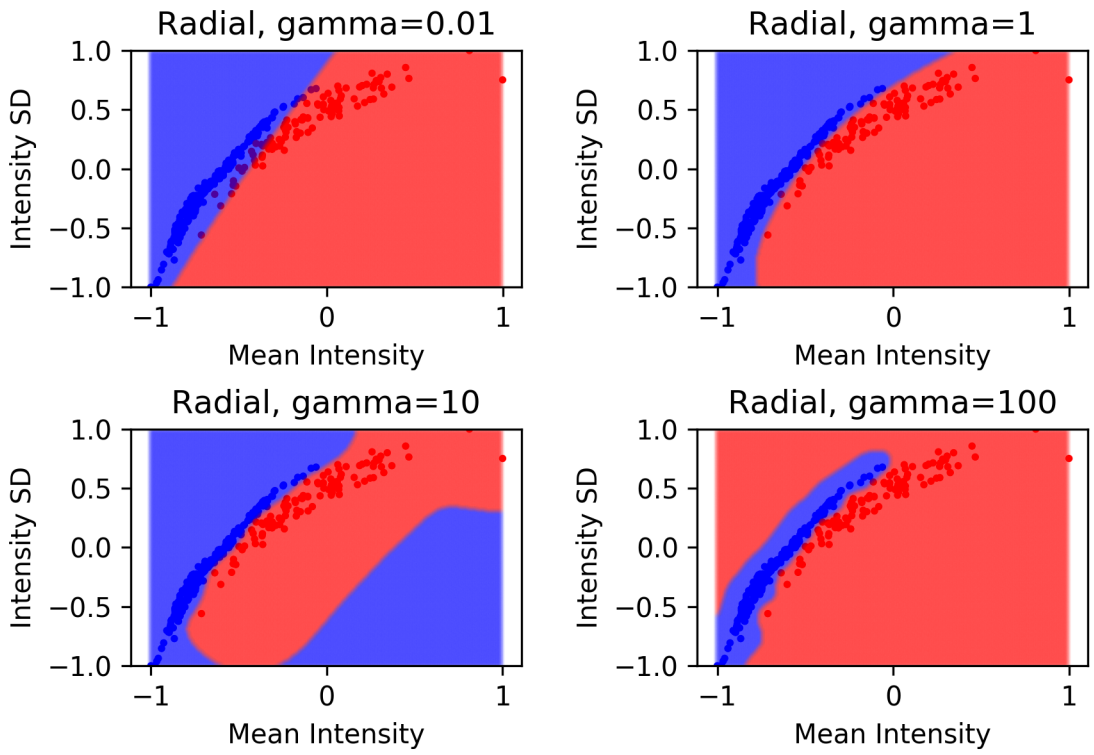


Figure 2.11. Decision regions for radial kernel SVM on 2D data (optimal C for corresponding γ, $\gamma = 0.01, 1, 10, 100$).

4.2 SVM with 'polynomial' kernel

Different degrees in 'polynomial' kernel with their corresponding optimal C's were compared in **Figure 2.8**. All degrees 2, 5, 10 and 20 seem to be causing overfitting.

Next, different C's were compared for degree = 2. C controls the margin width. When C is big, a small margin will be considered, and therefore leads to a more complicated decision function. In contrast, small C leads to a less complicated decision function. C in SVM model behaves as a regularization parameter. A special case when C = 0.01, a lot of data that are not very close to the decision boundary were considered to build a SVM model, and thus the model caused a big mis-classification. All the training data were classified into one single group. This case, the model is underfitted. Models with degree = 2 and C = 10 and 100 are overfitted.

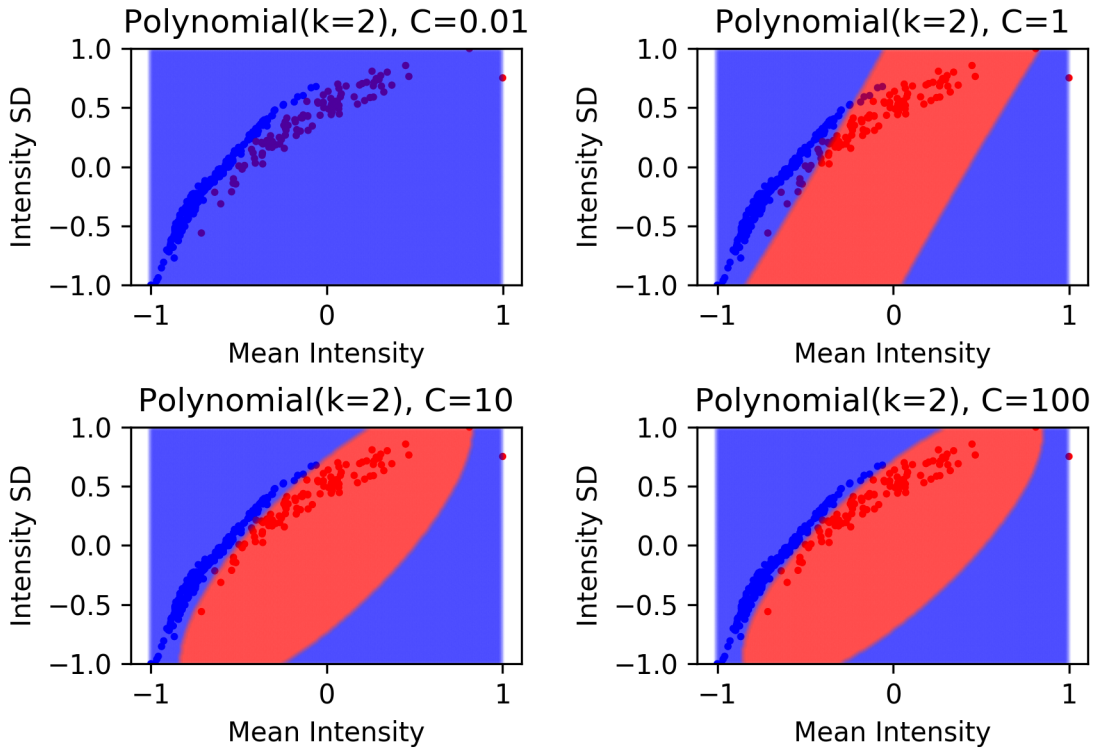


Figure 2.12. Decision regions for 'polynomial' kernel SVM on 2D data (degree =2, C = 0.01, 1, 10, 100).