Exercise Session3: FUnsupervised Learning and Large Scale Problems Prediction

Support Vector Machines

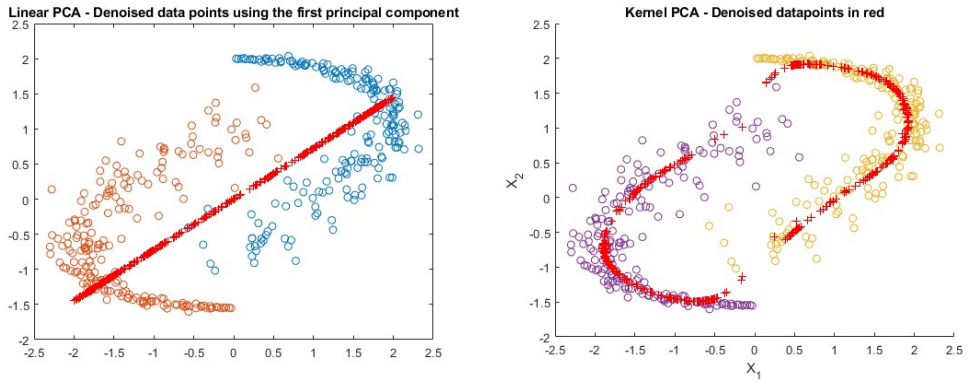
Zirui Yan r0916941

1.1 Kernel principal component analysis

(a) Principal component analysis (PCA) is a technique which is widely used for analyzing large datasets containing a high number of dimensions/features per observation. The idea of it is to reduce the dimension of data by mapping each data point onto few principle component.

PCA can reduce the noise, but cannot eliminate the noise. Noise distributes in every component. But PCA can capture the components with the most important information. Thus, noise in principle components is relatively small. So we can use PCA for denoising high-dimensional data.

If we increase the number of principal components, components with less information will be added to the data. So signal-to-noise ratio decrease. But if the number of principal components is not too much, more features remain in the data. So it is important to choose a good umber of principal components. (b) Linear PCA uses linear transformation to



(a) linear PCA with $n_c = 1$

(b) kernel PCA with $n_c = 6$

Figure 1: Comparison of linear PCA and kernel PCA. Kernel PCA is able to capture higher-dimension feature while linear PCA can only map the data points on a line.

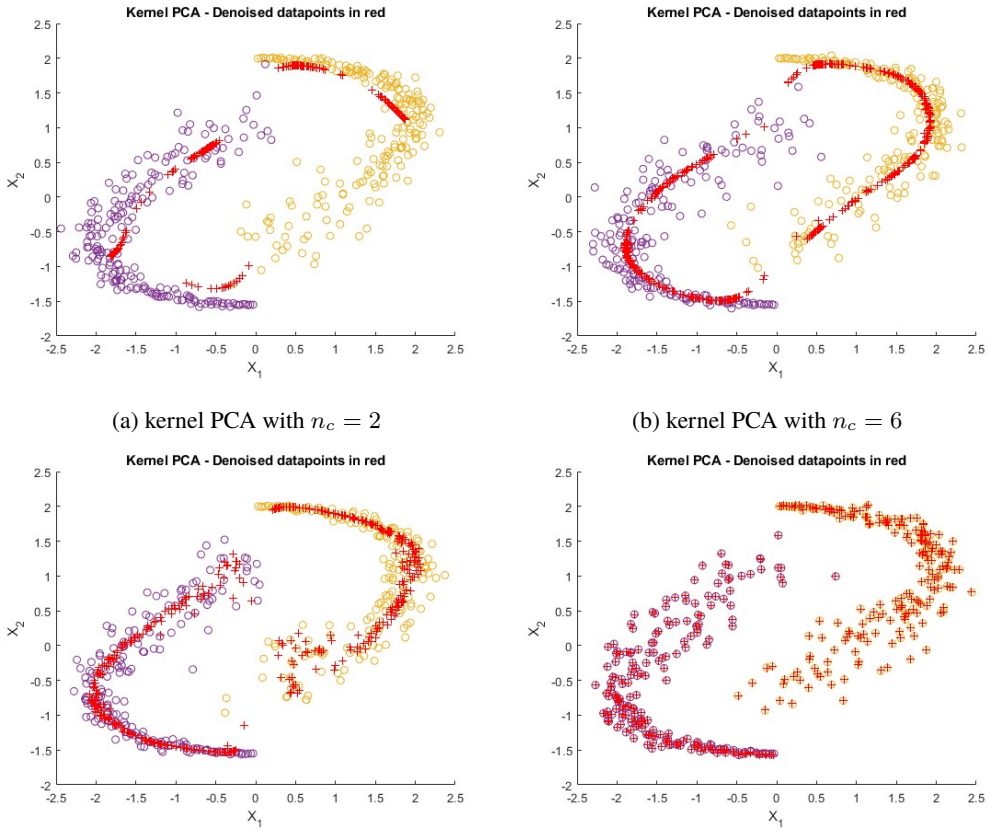
map the input space into target space. The main difference between it and kernel PCA is that kernel PCA first maps the input space into high-dimensional feature space, then uses linear transformation to map the input space into target space. So kernel PCA can capture features in higher-dimensional space, while linear PCA can only deal with linear features.

Another difference is that number of principal components of linear PCA is limited by input space while kernel PCA can use kernel method to get more components. So kernel PCA can get components even more than its input space. Score

variables of kernel PCA

$$z(x) = \sum_{l=1}^{N} \alpha_l \left(K(x_l, x) - \frac{1}{N} \sum_{r=1}^{N} K(x_r, x) - \frac{1}{N} \sum_{r=1}^{N} K(x_r, x_l) + \frac{1}{N^2} \sum_{r=1}^{N} \sum_{s=1}^{N} K(x_r, x_s) \right). \quad (1)$$

To make $K(x, x_i)$ distinct from $K(x, x_j))$ for $i \neq j$, we have N cannot be higher than the number of data points. So number of components of kernel PCA is limited by the number of data points.



(a) kernel PCA with $n_c = 2$

(b) kernel PCA with $n_c = 6$

(c) kernel PCA with $n_c = 6$

(d) kernel PCA with $n_c = 40$

Figure 2: kernel PCA with different n_c

(c) Figure 2 shows that kernel PCA cannot well reconstruct when n_C small. It capture high-dimensional noise when n_C too large, even it can reconstruct every data point, which also means it does not get rid of noise. So it is important to choose a good n_c. For choosing a better number of n_C, we can apply crossvalidation to this. Apply kernel PCA on training set and use validation set to get misclassification rate. Then, use the misclassification rate for every n_c to decide the optimal n_c.

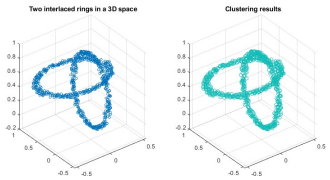
1.2 Spectral clustering

(a) The basic idea of spectral clustering is to make the intra-cluster weights(similarities) are high and the inter-cluster weights are low.

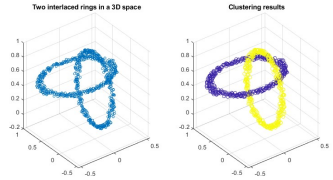
We first construct similarity matrix and degree matrix. Then use them to construct the Laplacian. Each eigenvector of Laplacian contains a feature of all points. So we can some eigenvectors as a reduced representation of each points. Then, use other clustering method such as K-mean algorithm to cluster the points.

(b) Classification is a supervised technique that aims to predict the labels of given data, while clustering is unsupervised. But for clustering, we don't know how many clusters and which cluster each point belongs to.

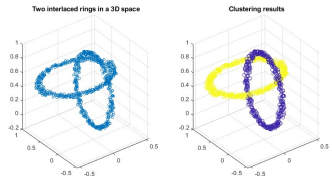
(c) As shown in Figure 3, small sig2 tends to cluster more points together, because small sig2 is better at find non-linear relation of the data points. When sig2=0.001, the Laplacian doe not divide into blocks and the eigenvector which is the feature is consider to be one cluster.



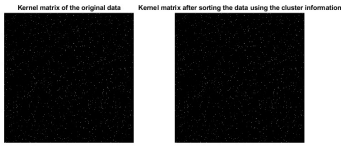
(a) sig2=0.001



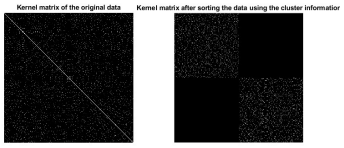
(b) sig2=0.005



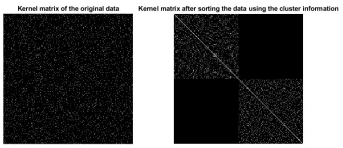
(c) sig2=0.01



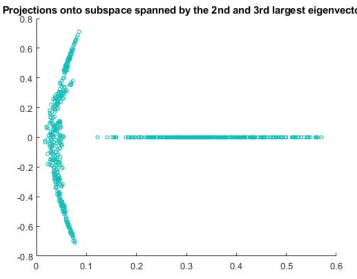
(d) sig2=10



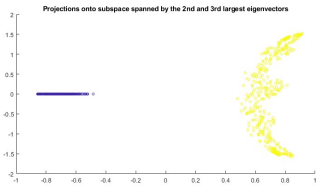
(e) sig2=0.01



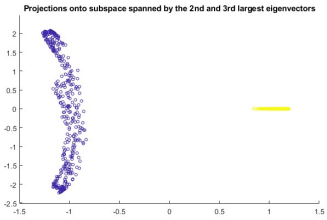
(f) sig2=0.1



(g) sig2=0.001



(h) sig2=0.005



(i) sig2=0.01

Figure 3: spectral clustering with different sig2

1.3 Fixed-size LS-SVM

(a) In primal space,

$$y(x) = w^T \phi(x) + b \tag{2}$$

where $\phi(\cdot)$ is a mapping to a feature space $R^D \rightarrow R^{Dh}$. If we want to solve the problem in dual space space, we have

$$y(x) = \sum_{k=1}^{N} a_k K(x, x_k) + b \tag{3}$$

where $K(x, x_k) = \phi(x_k)(x)$ is a positive definite kernel function and N refers to the number of the training points.

But when having a huge N, the kernel matrix is N-by-N, which is unacceptable. So when N»D, it is better to solve a model in the primal.

But when D is large, $\phi(\cdot)$ is complex. So when D»N, it is better solve a model in the dual.

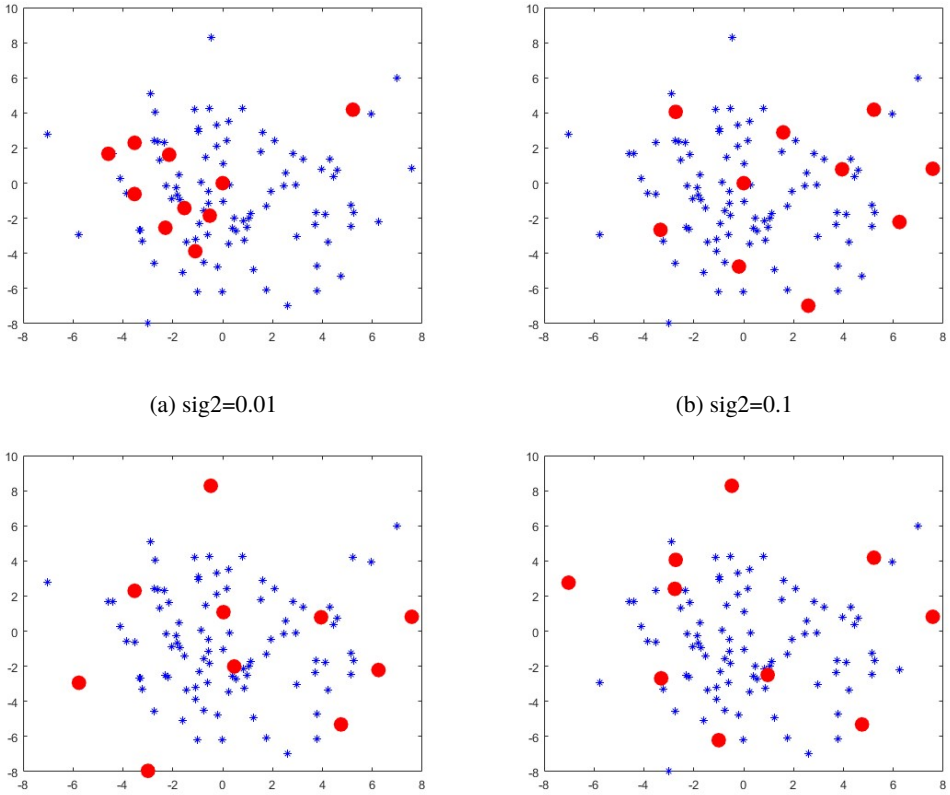
(b) **fixedsize_script1.m** randomly generates 100 points in 2-D plane and apply fixed-size LS-SVM to select support vectors. The criteria for selecting is the quadratic Renyi entropy

$$H_R = -\log \int p(x)^2 dx \tag{4}$$

Density estimation by kernel PCA gives

$$\int \hat{p}(x)^2 dx = \sum_{i=1}^{N} \tilde{\lambda}_i \left(\frac{1}{N} 1_v^T u_i \right)^2 \tag{5}$$

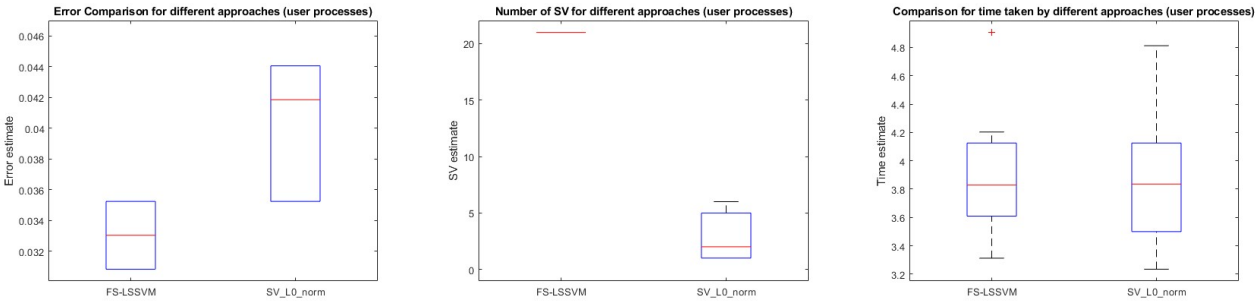
Figure 3 shows the effect of sig2 to selection of support vectors. Smaller sig2, the more selected support vectors of uniformly distribute. When sig2 becomes larger, the support vectors are tend to be closer to each other.



(a) sig2=0.01 (b) sig2=0.1

(c) sig2=1 (d) sig2=10

Figure 4: The effect of sig2 to selection of support vectors for Fixed Size LS-SVM



(a) error estimation (b) number of support vectors (c) computation time

Figure 5: Comparison of FS-LSSVM and l_0-approxiamtion for breast cancer wisconsin datasets

Large contributions to the entropy come from components that have small values of $\tilde{\lambda}_i \left(\frac{1}{N} 1_v^T u_i \right)^2$ and are related to elements with little or no structure, caused by observation noise or diffuse regions in the data. So fixed-size SVM tries to get out of the data caused by noise or diffuse regions and keep the structure most related to kernel.

As discussed in other exercise sessions, larger sig2 make RBF kernel more like linear kernel and smaller sig2 gives more curvature. So when sig2 is small, the selected support vectors tend to gather in a small circle. And when sig2 is

large, the selected support vectors are more likely to uniformly distribute in large circle.

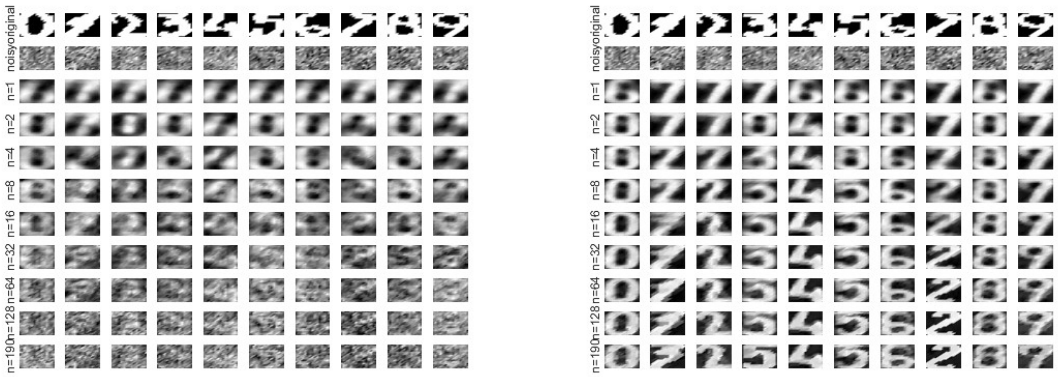
(c) I run **fslssvm_script.m** on different data sets multiple times. The plot of every run is different, which comes from different results given by optimization.

But there are something in common in every run. Since l_0-penalty add punishment for the number of non-zero term, the number of support vectors for l_0-approxiamtion is not a fixed number and smaller the number of support vectors of fixed-size LS-SVM. On the other hand, the estimated error of l_0-approxiamtion is higher than fixed-size LS-SVM in every run. The reason for this is that l_0-approxiamtion sacrifice some performance for sparsity.

And the behaviour of computation time changes with datasets.

2.1 Homework: Kernel principal component analysis

Learning handwriting number is a classic problem. In this exercise, we use a dataset which contains 198 data points. Each data point represents a digit and has 240 pixels. The goal of this exercise is to use the pixels of digit to learn the feature of digit.



(a) linear PCA

(b) kernel PCA

Figure 6: Comparing reconstructions of linear PCA and kernel PCA

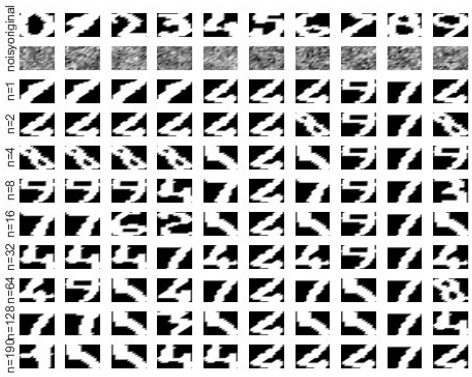
(a) Both linear PCA and kernel PCA use 190 principle components for reconstruction. The reconstruction of linear is meaningless for human, but PCA kernel PCA can generate readable results. kernel PCA is good at capturing high-dimension information. And another reason for better performance is that this dataset has at most 198 components for kernel PCA.

Figure 6 also shows the reconstructed digit looks like other digits. For example, 1, 2, 9 looks like 7 and 3, 4 looks like 5, which caused by different patterns giving confusing information. When n_c increase, reconstructing process sometimes adds misleading components. And for some smaller n_c even have better result.

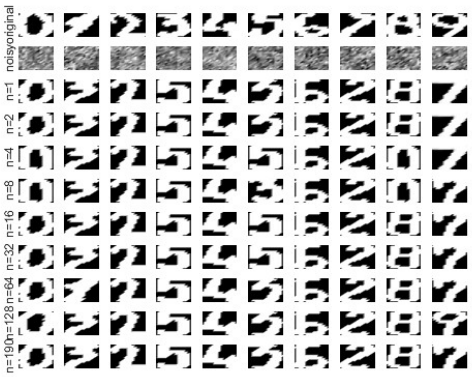
(b) sig2 is suggested to be around the mean of the variances of each dimension times the dimension (number of features) of the training data.

When sig2 grows, the reconstructed image becomes more blurry since RBF kernel behaves more like linear kernel. The larger sig2 is, the image is more sharpen since RBF kernel is more easy to capture the non-linear feature such as the sudden change of edges of digits. But too small sig2 makes the image meaningless and likes sloping line.

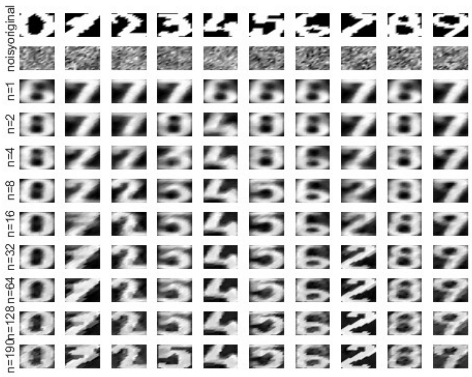
(c)



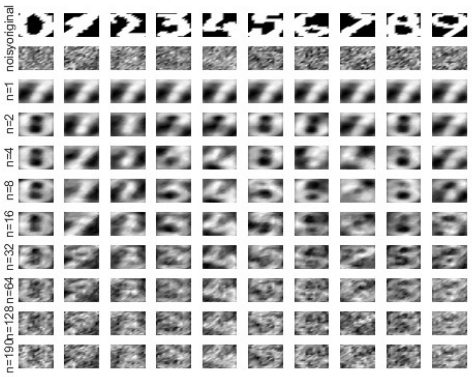
(a) sigmafactor=0.01



(b) sigmafactor=0.1



(c) sigmafactor=1



(d) sigmafactor=10

Figure 7: Reconstruct by kernel PCA with different sigmafactor

2.2.1 Homework: Shuttle (statlog)

(a) The shuttle dataset contains the time and condition of a shuttle. There are 9 attributes all of which are numerical. And the first attributes is the time. The number of data points is 58000. But in this exercise, we only use the first 700 data points.

There are 7 classes. Approximately 80% of the data belongs to class 1. Therefore the default accuracy is about 80%, since 80% can be easily achieved by classifying all data points to be in class 1. In this exercise, we set class 2-7 to be the negative class and class 1 to be the possitive class. The visualization shows the comparison of 9 attributes for class 1 and other classes.

(b) Both FS-LSSVM and l_0-approxiamtion perform very well on this datasets. The outliers for l_0-approxiamtion come from that l_0-approxiamtion hugely reduces the number of support vectors into 2. But if we want to classify class 2-7, we need multiclass method.

2.2.2 Homewrok: California

(a) the California dataset consists of 20640 data points. Specifically, it contains ln(median house value), ln(median income), ln(housing median age), ln(total rooms), ln(total bedrooms), ln(population), ln(households), latitude, and longi-

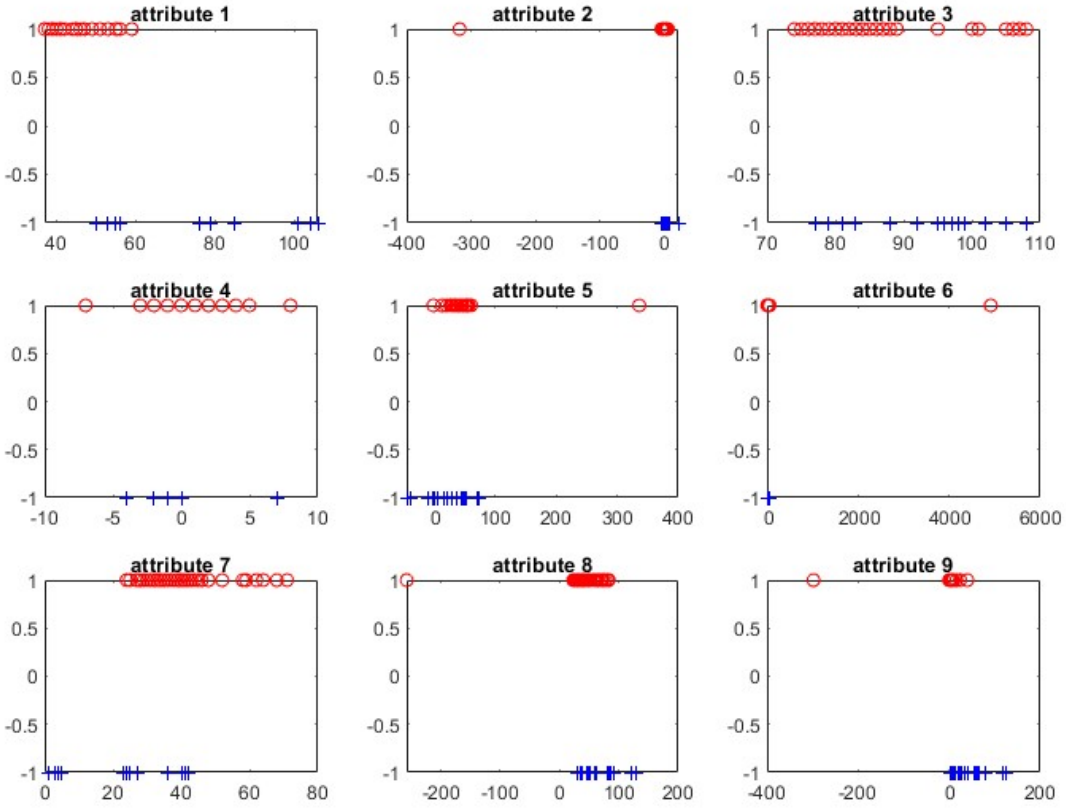
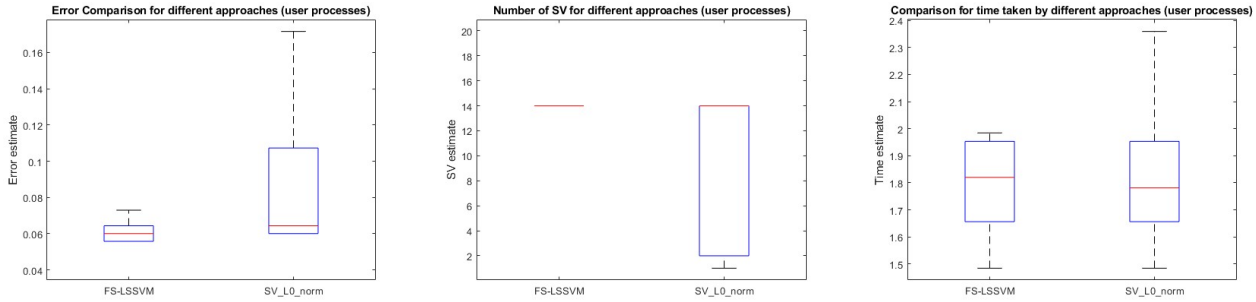


Figure 8: Comparison of 9 attributes. x-axis being the value of the attribute, y-axis being the class. (red) class 1 (blue) other classes

tude. Our goal is to do regression and predict the ln(median house value)

(b) As shown in Figure 11, both S-LSSVM and l_0-approxiamtion use quite long time for computing because the the dataset consists 20640 data points. l_0-approxiamtion can effectively reduce the number of support vectors, but also cause huge error sometimes, see the outlier of Figure 11 (a).



(a) error estimation (b) number of support vectors (c) computation time

Figure 9: Comparison of FS-LSSVM and l_0-approxiamtion for shuttle dataset

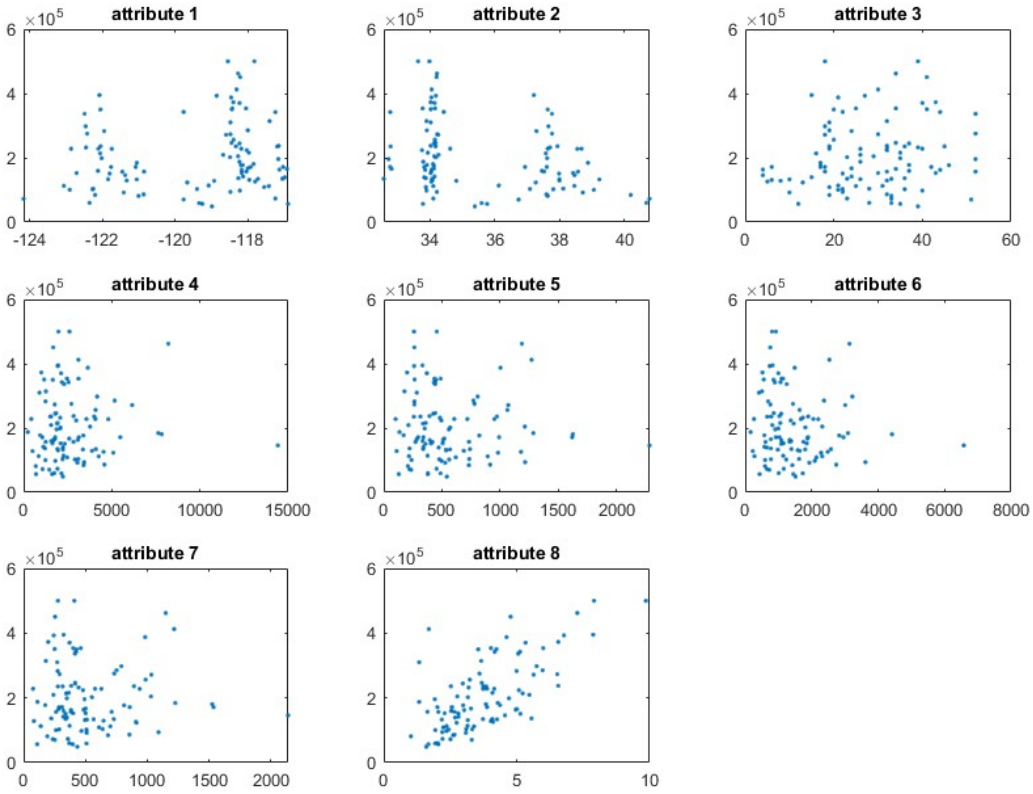
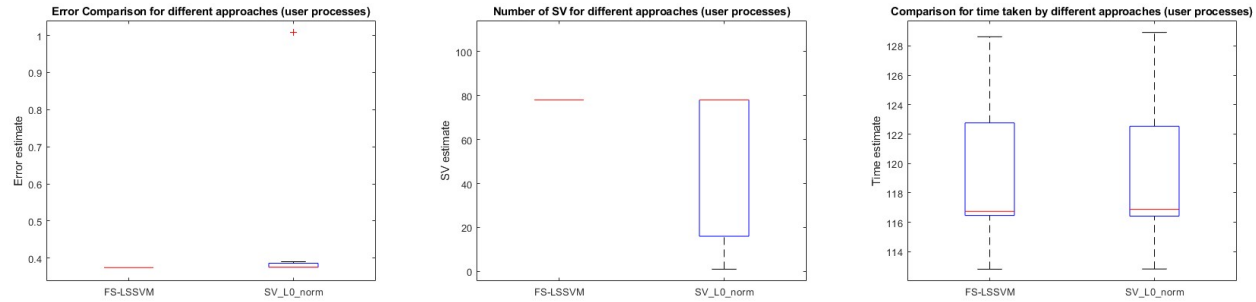


Figure 10: Relation between ln(median house value) and other attribute



(a) error estimation (b) number of support vectors (c) computation time

Figure 11: Comparison of FS-LSSVM and l_0-approxiamtion for California housing dataset