PCA / KPAC report

Introduction

In machine learning field, especially resolving classification questions, the feature selection is one of the biggest problems for data pre-processing, since many features may not helpful or have a lower effect for labels. For example, if you want to predict the house price of next year, it is hard to contact what kind of food you eat with predicted house prices. For solving this problem, the PCA and KPCA will be used to remove unnecessary features to guarantee a high relationship between features and labels. The PCA and KPCA are two important algorithms, which applied in reducing noise, decreasing feature size, increasing the training speed, etc. In this report, I will introduce the algorithm principle of PCA and KPCA and what i think about PCA and KPCA, then some implementation and analysis of PCA/KPCA will be presented, finally, the data with noise also will be analysed.

Algorithm

First, we have a data set, which have m samples and n features:

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ x_{m1} & \cdots & \cdots & x_{mn} \end{bmatrix}$$

Some features we do not need, so we use PCA / KPCA remove part of feature (dimension), and then keep k features. The goal of PCA is to find a set of k vectors (k < n), in space Rd, where space Rd include the maximum of variance values in data.

Suppose a samples only have 2 features, the figure about feature1 and feature2 like blow, if we want to reduce 1 feature from 2 features, we also need to reduce 2 dimensions to 1 dimension, from a face to a link, since in face we have infinite line can be chosen, so which line should be chosen? The answer is the link that make sample has biggest variance value.

If we project each sample xj on a vector v, the projection of each sample on that vector are:

$$\text{the coefficient is } < x_j, v >= x_j^T v = v^T x_j$$

we regulate the mean of each feature vector is 0, so the variance is :

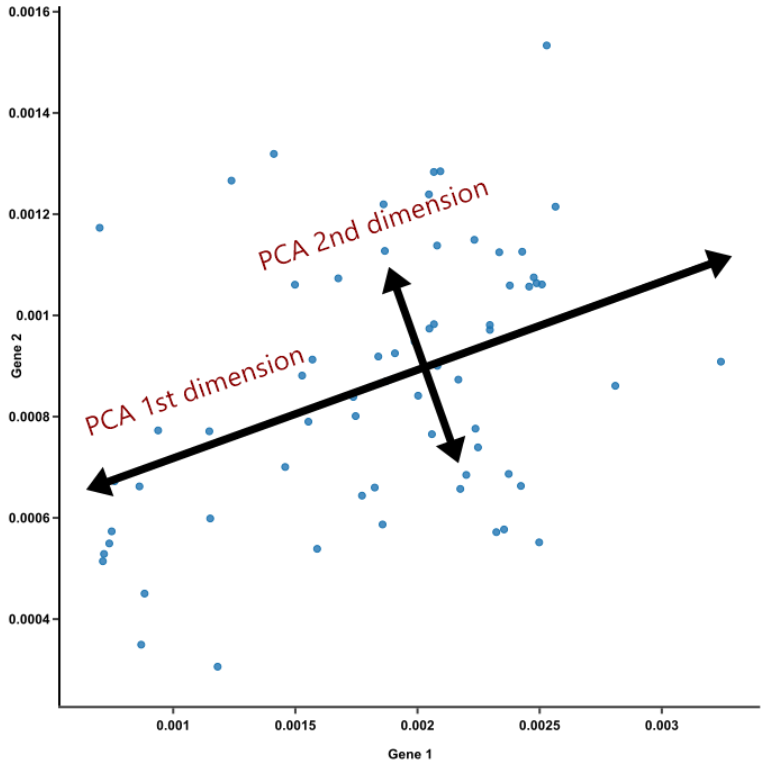
$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}\left(v^T x_i - 0\right)^2 = \frac{1}{N}\sum_{i=1}^{N}\left(v^T x_i\right)\left(v^T x_i\right) = \frac{1}{N}\sum_{i=1}^{N}\left(v^T x_i\right)\left(v^T x_i\right)^T$$

$$= \frac{1}{N}\sum_{i=1}^{N} v^T x_i x_i^T v = v^T\left(\frac{1}{N}\sum_{i=1}^{N} x_i x_i^T\right)v = v^T C v$$

And the C is convariance matrix:

$$C = \frac{1}{N}\sum_{i=1}^{N} x_i x_i^T$$

Then, we can find if a vector have greatest variance value (the longest distance for data projection on current vector), which vector become best vector for dimensionality reduction. (v called principle vector and the ||v|| = 1)



Thus, the question transfer from finding the best vector to reduce dimension to find the greatest variance value for data.

$$v = \underset{v \in R^d, \|v\|=1}{\arg\max}\, v^T C v$$

Because we can use lagrange function to optimize the best variance, where restrict condition is ||v|| = dot(v^T, v) = 1 and optimizing function is vTCv:

$$\text{Lagrangian: } f(v,\lambda) = v^T C v - \lambda\left(v^T v - 1\right)$$

$$\frac{\partial f}{\partial v} = 2Cv - 2\lambda v = 0 \Rightarrow Cv = \lambda v$$

$$\frac{\partial f}{\partial \lambda} = v^T v - 1 = 0 \Rightarrow v^T v = 1$$

We find Partial derivation of vector v for the equation, then we find the standard function of computing eigenvalue and eigenvector:

$$Cv = \lambda v$$

So the best vector can be computed, and if we add the restrict condition and we can find that the best eigenvalue can replace the variance, which if we know suitable eigenvalue we can know the best space to compress feature dimension

$$\overline{v^T v = 1}$$

$$v^T C v = v^T \lambda v = \lambda v^T v = \lambda$$

KPCA

In the previous about the PCA dimensionality reduction algorithm, PCA can only deal with the dimensionality reduction of linear data, which is essentially linear transformation, and it is only the feature that filters the largest variance, and removes the linear correlation between features. It is often ineffective for linearly inseparable data.

Suppose the data we have is not linearly separable in low-dimensional space, but can be linearly separable in high-dimensional space. Using this feature, KPCA can increase the original data to the high-dimensional space through the kernel function, and then uses the PCA algorithm to reduce the dimension, so it is called K PCA dimension reduction. So the key to the KPCA algorithm is this kernel function.

Similar we now have data, but this time the data is not linearly sepearable in low dimension:

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ x_{m1} & \cdots & \cdots & x_{mn} \end{bmatrix}$$

So we should map data into higher dimensional space, we use φ represent high dimensional space, so we get kernel matrix:

$$K = XX^T \begin{bmatrix} \phi(x_1)^T \phi(x_1) & \cdots & \phi(x_1)^T \phi(x_n) \\ \vdots & \ddots & \vdots \\ \phi(x_m)^T \phi(x_1) & \cdots & \phi(x_m)^T \phi(x_n) \end{bmatrix}$$

And the convariance matrix is :

$$C = \frac{1}{N} X^T X$$

$$\text{Note that } X^T X \neq XX^T = K$$

Then, we try to find some relationship between K and C by computing the eigenvalue of K, and the u is eigenvector on the higher dimensions and we can not know that:

$$(XX^T)u = \lambda u$$

$$X^T(XX^T)u = \lambda X^T u \Rightarrow (X^T X)(X^T u) = \lambda(X^T u)$$

That mean the X^Tu is eigenvector of X^TX, and the v can be computed by X^T u, so we transfer the higher dimension vector u to the vector that we want:

$$v = \frac{1}{\|X^T u\|} X^T u = \frac{1}{\sqrt{u^T X X^T u}} X^T u$$
$$= \frac{1}{\sqrt{u^T(\lambda u)}} X^T u = \frac{1}{\sqrt{\lambda}} X^T u$$

Then we project the data to our new v:

$$v^T \phi(x') = \left(\frac{1}{\sqrt{\lambda}} X^T u\right)^T \phi(x') = \frac{1}{\sqrt{\lambda}} u^T X \phi(x')$$

For my understanding, I think that the goal of PCA and KPCA is to find the maximum value of variance in one dimension, if the data is linear separable, we process it by PCA or KPCA if data is not linear separable.

Implementation

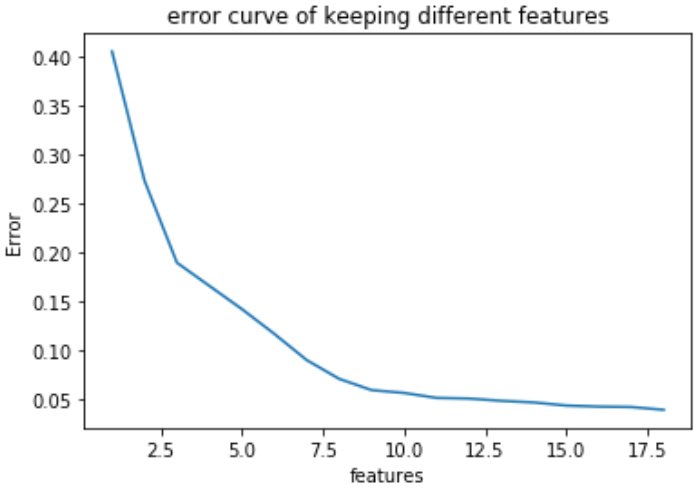
I implement PCA and KPCA by python, I use 3rd part tool box sklearn as the control group and print out the error rate in each epoch. The classifiers are SVC and KNN classifier in different group with visualized figure.

Before PCA, we also do normalization for data, setting feature means to 0, so that data will not offset and easy to compute, the life side do not normalized and the right side has been normalized:

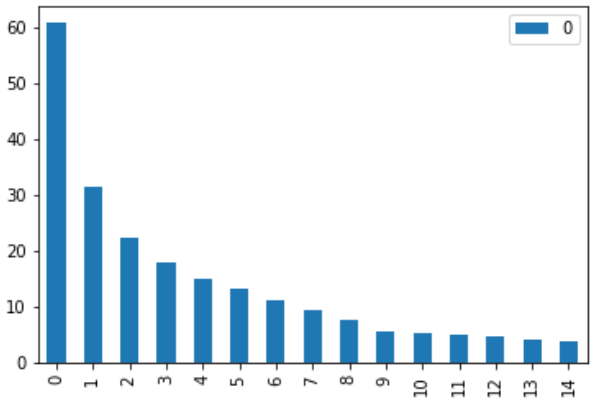
count	7291.000000	7291.00000	7291.000000
mean	-0.991950	-0.97257	-0.930964
std	0.050670	0.11769	0.195285
min	-1.000000	-1.00000	-1.000000
25%	-1.000000	-1.00000	-1.000000
50%	-1.000000	-0.99999	-0.999620
75%	-0.999970	-0.99848	-0.980065
max	0.000308	0.33293	0.450490

count	2.007000e+03	2.007000e+03	2.007000e+03	2.007000e+03
mean	4.123653e-14	2.262505e-14	1.046030e-14	7.582099e-15
std	1.000249e+00	1.000249e+00	1.000249e+00	1.000249e+00
min	-1.703537e-01	-2.409172e-01	-3.664228e-01	-5.252631e-01
25%	-1.703537e-01	-2.409172e-01	-3.664228e-01	-5.249315e-01
50%	-1.703537e-01	-2.408342e-01	-3.641696e-01	-4.927366e-01
75%	-1.697692e-01	-2.274689e-01	-2.539399e-01	-1.841623e-02
max	1.237032e+01	9.592452e+00	7.209799e+00	4.751800e+00

The we use PCA process data in different dimension and calculate error rate, where the error rate is cross_val_score from sklearn, and the figure below:

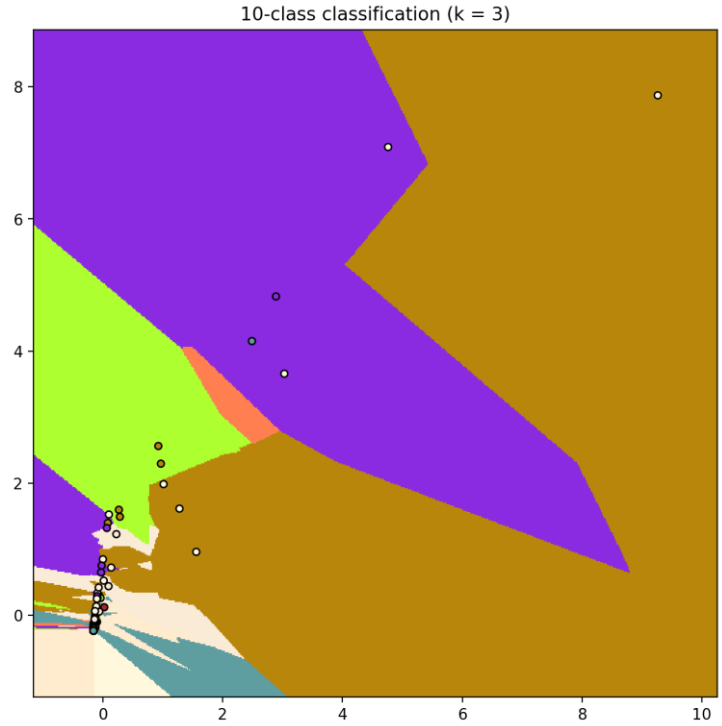


We find the error rate is greatly reduced until the feature number is 10, for evaluate if the 10 features is good size of feature, we also sort the eigenvalues and print the figure about eigenvalue of different eigenvectors.

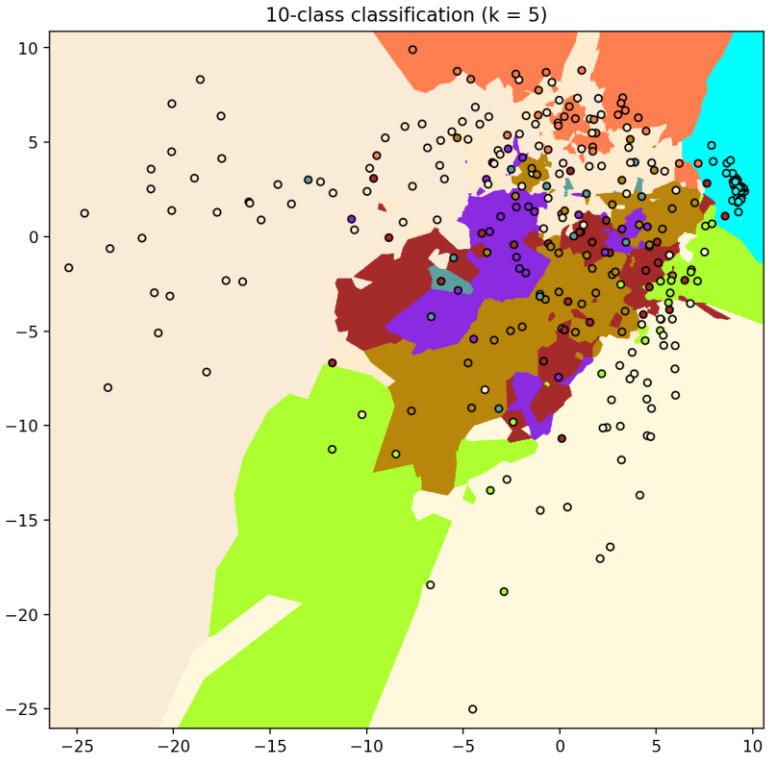


So you can find the top 10 features may occupy 95% eigenvalues or above, and that mean the all features can be reduced to top 10 features, so we evaluate our hypothesis.

For KPCA, we use different classifier to implement visualization, the figures below is the classification in KNN classifier by using test set with 256 features:

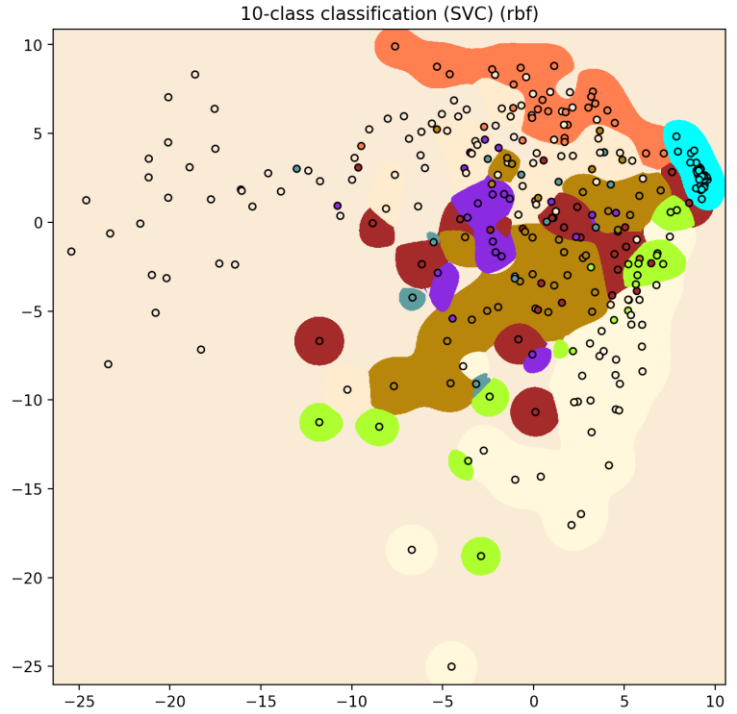


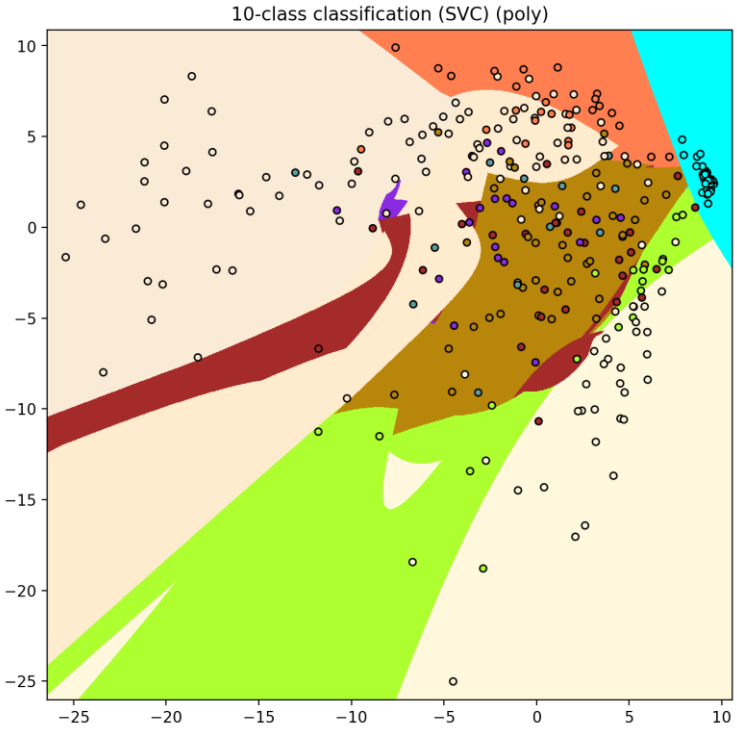
After PCA, the feature size change to 10, then the figure is like:

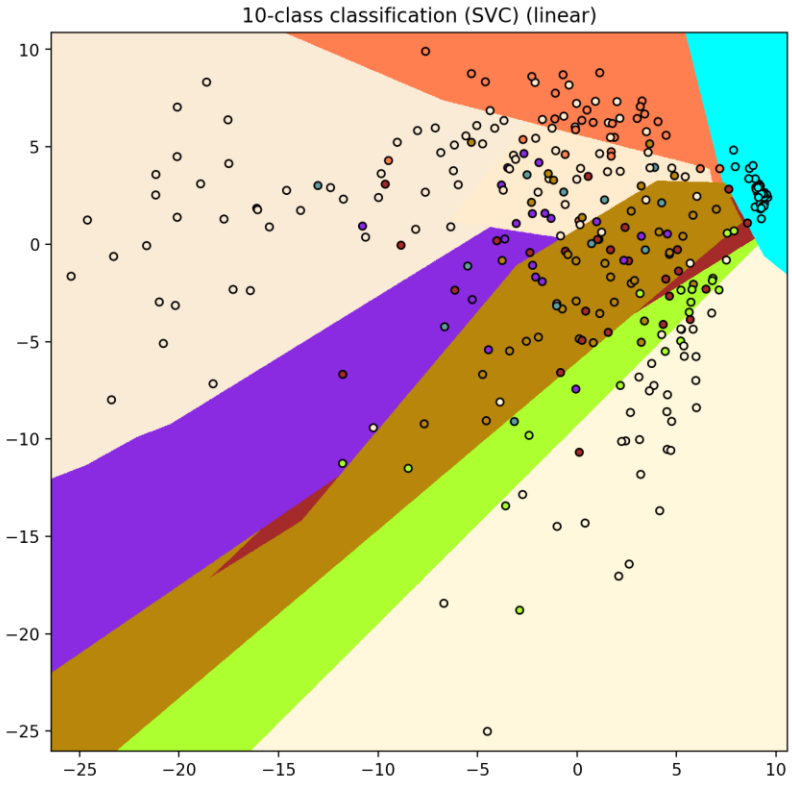


Compare with 256 feature, the reduced features have better classification, it is obviously present in figure that the most data (point, same color with background means correct classification).

Then we try to use KPCA process the data by using SVC, we prepare 3 kernel functions, linear kernel, polynomial kernel, and gaussian kernel, and plot them:

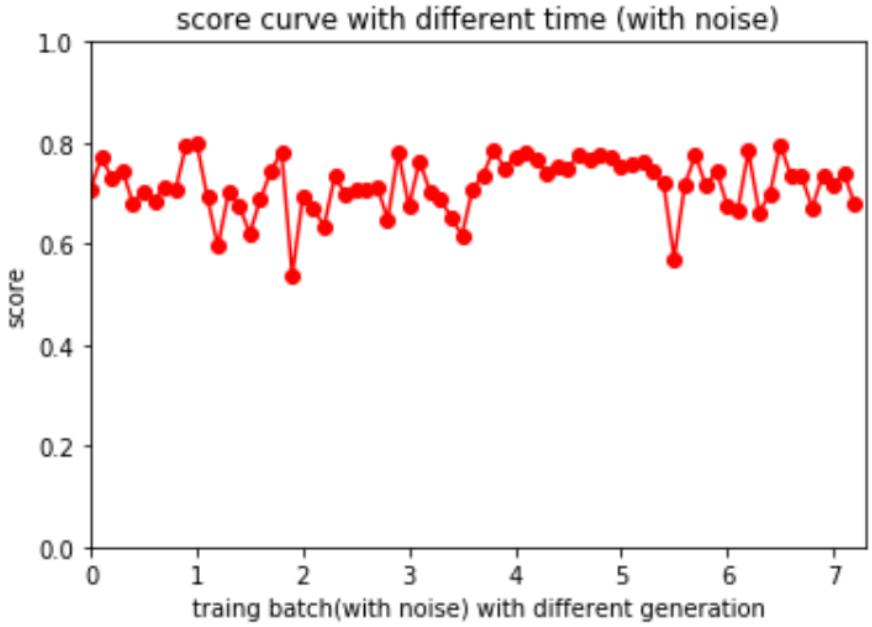

10-class classification (SVC) (rbf)
10
5
0
−5
−10
−15
−20
−25
−25
−20
−15
−10
−5
0
5
10


10-class classification (SVC) (poly)
10
5
0
−5
−10
−15
−20
−25
−25
−20
−15
−10
−5
0
5
10

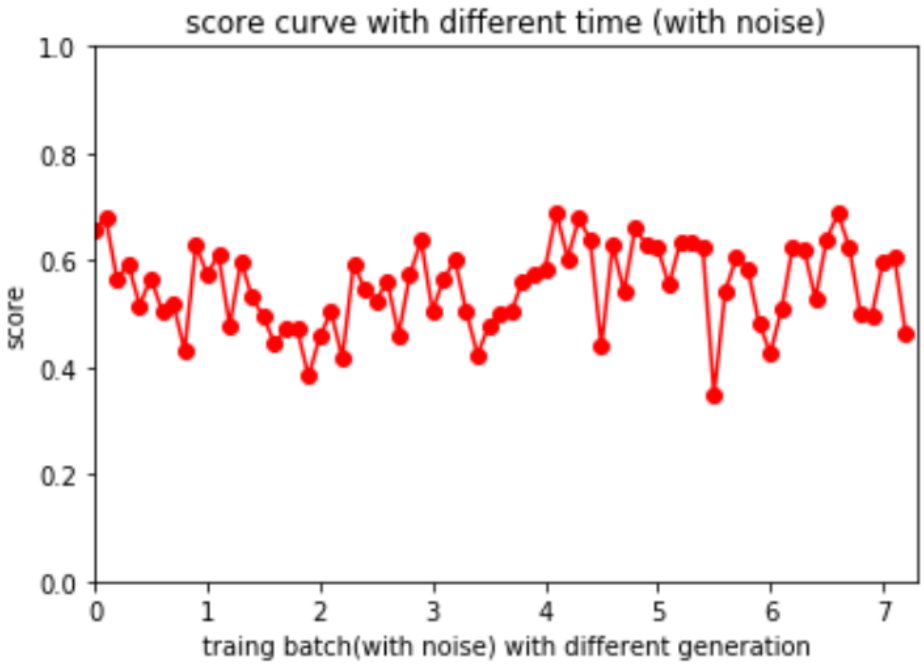


We find the KCPCA with gaussian kernel has a good performance. Comparing to linear kernel, the rbf kernel can curve the edge of classification field, which make gaussian PCA have better performance to processing non linear separable data set.

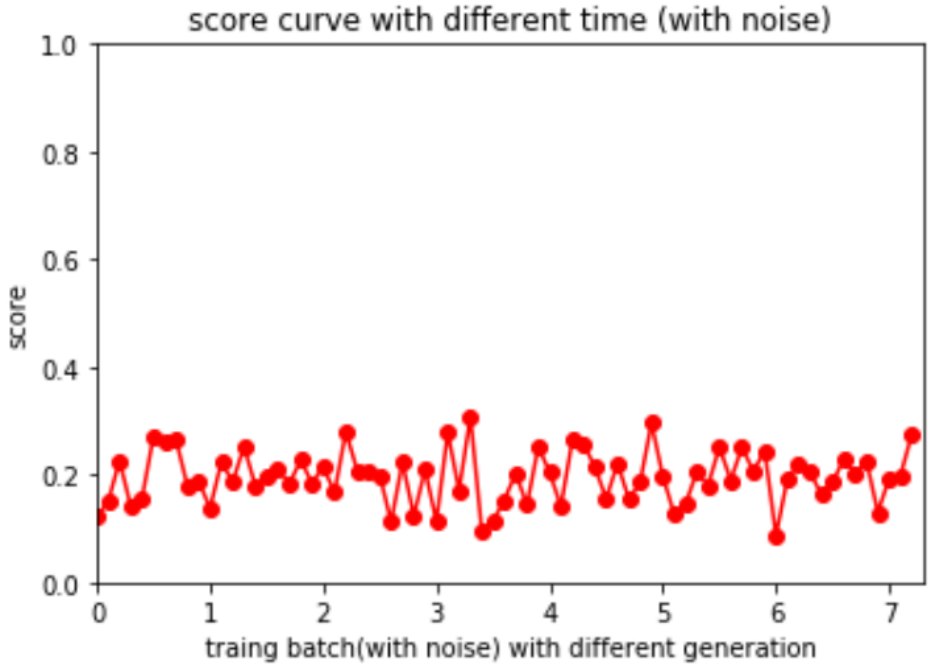
When we add another 512 dimension noise into original dataset, and use svm with different kernel function
Linear kernel SVM:



Gaussian kernel SVM:



Polynomial kernel:



Based on the figures, we find that the scores of SVM with linear kernel are stable between 0.6 and 0.8, and has the lowest vibration. Comparing to linear, the gaussian kernel function show a higher vibration and lower score. In the last figure, it is score curve with Polynomial kernel SVM, the score of PKSVM may not rational, which is the lowest between SVM with other kernel function and it also has the higher vibeation.

Conclusion

Based on what i done in this assignment, some point can be concluded in below

1. The goal of PCA and KPCA is to reduce the size of features, which make model have a good performance in accuracy and decreasing train time.
2. KPCA can solve the problem that the data is not linear separable.
3. Removing mean of feature can prevent the offset of data.
4. The number of variance can be computed by eigenvalue, which is the important fact to decide how many dimensions should be kept.
5. Based on the dataset we have, we add some noise in dataset , and the SVM with linear kernel show a better performance than other kernel SVM.