STATS 201B Final Paper
Application of Kernel on German Credit

Kun Zhou 204688165, Xin Kang 604589390

March 17, 2016

Abstract

Nowadays, credit risk evaluation becomes increasingly important in finance analysis area. And we need more reliable methodologies to help lenders in all credit segments better assess the credit risk of potential borrowers. Therefore, advanced techniques based on big data and machine learning come into public attention and play a big role in credit risk evaluation domain. In this paper, we mainly compared the credit classification performance between Kernel Logistic Regression and Support Vector Machine, and compared the credit-amount prediction performance between Kernel Ridge Regression and Robust Regularized Kernel Regression, and then made corresponding conclusions.

1 Introduction

Credit risk modeling aims to measure the amount of credit risk which banks or other related organizations are exposed to. The second section provides description of the data set we used and related data preprocessing procedures. In the third section, we researched in Kernel Logistic Regression (KLR) for classifying good or bad credit, referring to paper "Applying KLR in Data Mining to Classify Credit Risk"[1]. Then we compared the credit classification performance of KLR with Support Vector Machine (SVM) which is another well-known kernel machine, and made our conclusion. In the fourth section, we applied Kernel Ridge Regression (KRR) which we have learnt in the class to predict credit amount. Then we changed loss function to Huber Loss in order to improve prediction performance, referring to paper "Robust Regularized Kernel Regression"[2] (RRKR). We also compared the credit-amount prediction performance between KRR and RRKR and made conclusion.

2 Data

In this paper, we used the German credit data set which is available in the UCI Repository[3]. There are totally 1000 observations (700 good credits, 300 bad credits). The data set consists of 20 attributes (7 numerical, 13 categorical).

For 20-attribute case, we used all 20 attributes.

For 15-attribute case, we removed 5 qualitative attributes and used following 15 attributes:

- Status of existing checking account (qualitative)

- Duration in month (numerical)
- Credit history (qualitative)
- Credit amount (numerical)
- Savings account/bonds (qualitative)
- Present employment since (qualitative)
- Installment rate in percentage of disposable income (qualitative)
- Present residence since (numerical)
- Property (qualitative)
- Age in years (numerical)
- Other installment plans (qualitative)
- Number of existing credits at this bank (numerical)
- Number of people being liable to provide maintenance for (numerical)
- Telephone (qualitative)
- Foreign worker (qualitative)

For 10-attribute case, we removed 10 qualitative attributes and used following 10 attributes:

- Duration in month (numerical)
- Credit amount (numerical)
- Installment rate in percentage of disposable income (qualitative)
- Present residence since (numerical)
- Property (qualitative)
- Age in years (numerical)
- Number of existing credits at this bank (numerical)
- Number of people being liable to provide maintenance for (numerical)
- Telephone (qualitative)
- Foreign worker (qualitative)

The data set was divided into two subsets : a training set and a testing set, with 70% - 30% of training-testing. We converted qualitative attributes to dummy variables and pre-processed the data set so that the mean is 0 and standard deviation is 1. We used 1 to indicate good credit and 0 to indicate bad credit.

3 Kernel Logistic Regression

Logistic Regression is a linear classifier and has been applied in credit risk evaluation. But what if the data set we want to classify has non-linear boundary? Hence we need a non-linear form of logistic regression called Kernel Logisitic Regression. Basically, Kernel Logistic Regression is a kernelized version of traditional logistic regression, which means applying "kernel trick "to logistic regression, and we used Gaussian kernel. In this paper, our goal is to study the performance of Kernel Logistic Regression in credit risk evaluation. So we wrote the function of Kernel Logistic Regression and the function of Gaussian kernel in R by ourselves.

3.1 Formula Deduction

Based on the knowledge of Logistic Regression and kernel trick that we have learnt in the class, first, let's assume $logit(p) = \phi(X)^\top \theta$ instead of $logit(p) = X^\top \beta$, mapping original data X

into a high-dimensional feature space. Then we can get:

$$p = \frac{1}{1 + e^{-\phi(X)^\top \theta}} \tag{1}$$

The loss function of kernel logistic regression equals to negative log likelihood plus L2 norm:

$$Loss(\theta) = -logL(\theta) + \frac{\lambda}{2}\|\theta\|^2 \tag{2}$$

where

$$L(\theta) = \prod_{i=1}^{N} p_i^{y_i}(1 - p_i)^{(1-y_i)} \tag{3}$$

$$l(\theta) = logL(\theta) = \sum_{i=1}^{N}[y_i log(p_i) + (1 - y_i)log(1 - p_i)] \tag{4}$$

Then we can rewrite our loss function as:

$$Loss(\theta) = \sum_{i=1}^{N}[-y_i\phi(X_i)^\top\theta + log(1 + e^{\phi(X_i)^\top\theta})] + \frac{\lambda}{2}\|\theta\|^2 \tag{5}$$

$$\begin{aligned} \frac{\partial Loss(\theta)}{\partial \theta} &= \sum_{i=1}^{N}[-y_i\phi(X_i) + \frac{e^{\phi(X_i)^\top\theta}}{1 + e^{\phi(X_i)^\top\theta}}\phi(X_i)] + \lambda\theta \\ &= \sum_{i=1}^{N}(p_i - y_i)\phi(X_i) + \lambda\theta \end{aligned} \tag{6}$$

Let $\frac{\partial Loss(\theta)}{\partial \theta} = 0$ and with Representer Theorem[4], we get $\theta = \sum_{i=1}^{N} c_i\phi(X_i)$, where $c_i = \frac{1}{\lambda}(y_i - p_i)$ is a constant. Since $K(X_i, X_j) = \langle\phi(X_i), \phi(X_j)\rangle = \phi(X_i)^\top\phi(X_j)$, we can get $\langle\theta, \theta\rangle = c^\top Kc$ and $\phi(X)^\top\theta = Kc$. We can also rewrite our equation (1) as:

$$p = \frac{1}{1 + e^{-Kc}} \tag{7}$$

The regularized loss function can be rewritten with respect to c as:

$$Loss(c) = \sum_{i=1}^{N}[-y_i k_i c + log(1 + e^{k_i c})] + \frac{\lambda}{2}c^\top Kc \tag{8}$$

where k_i is the ith row in the kernel matrix K.

$$\frac{\partial Loss(c)}{\partial c} = K(p - y) + \lambda Kc \tag{9}$$

$$\frac{\partial^2 Loss(c)}{\partial c \partial c^\top} = KWK + \lambda K \tag{10}$$

where W is a diagonal matrix with diagonal elements $p_i(1 - p_i)$ for $i = 1, ..., N$.

According to Newton-Raphson's method:

$$c^{(t+1)} = c^{(t)} - \left(\frac{\partial^2 Loss(c)}{\partial c \partial c^\top}\right)^{-1} \frac{\partial Loss(c)}{\partial c} \tag{11}$$

The Newton-Raphson's method updates with respect to c on the $(t+1)$th iteration is

$$c^{(t+1)} = c^{(t)} + (KW^{(t)}K + \lambda K)^{-1}(K(y - p^{(t)}) - \lambda K c^{(t)}) \tag{12}$$

$$(KW^{(t)}K + \lambda K)c^{(t+1)} = KW^{(t)}(Kc^{(t)} + (W^{(t)})^{-1}(y - p^{(t)}) \tag{13}$$

Finally, we get:

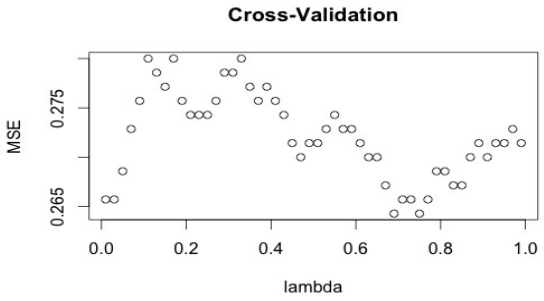
$$c^{(t+1)} = \left(K + \lambda(W^{(t)})^{-1}\right)^{-1}\left(Kc^{(t)} + (W^{(t)})^{-1}(y - p^{(t)})\right) \tag{14}$$

where λ can be chosen by cross-validation.

3.2 Results

We aimed to study the credit classification performance of Kernel Logistic Regression and compare Kernel Logistic Regression and Support Vector Machine in terms of their classification performance. The RBF kernel was used for the SVM model. We used confusion matrix to evaluate their results.

In order to find our best λ, we wrote the function of KLR.cv to find best λ through cross-validation. We set $\sigma = \sqrt{P}$ in Gaussian Kernel where P is the number of predictors, and set 30 iterations which is enough to converge.



(a) Cross-Validation

```
> klrresult_20    # with best lambda=0.7
Confusion Matrix and Statistics
          Reference
Prediction   0   1
         0  22  10
         1  73 195

               Accuracy : 0.7233
               Sensitivity : 0.23158
               Specificity : 0.95122
```

(b) KLR with best $\lambda = 0.7$

```
> svmresult_20
Confusion Matrix and Statistics
          Reference
Prediction   0   1
         0   8   2
         1  87 203

               Accuracy : 0.7033
               Sensitivity : 0.08421
               Specificity : 0.99024
```

(c) SVM

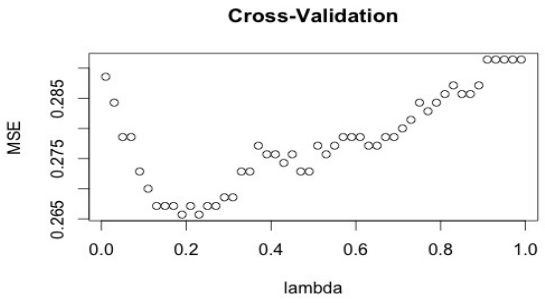
```
> klrresult_20      #with lambda=2.0
Confusion Matrix and Statistics
          Reference
Prediction   0   1
         0   7   4
         1  88 201

               Accuracy : 0.6933
               Sensitivity : 0.07368
               Specificity : 0.98049
```

(d) KLR with $\lambda = 2.0$

Figure 1

The results of 20-attribute case is shown in Figure 1. Since the German credit data set is imbalanced which means it has more good credits than bad credits, it is more suitable to look at sensitivity and specificity rather than accuracy. For 20-attribute case, we can get best $\lambda = 0.7$ from Figure 1a . Kernel Logistic Regression with best λ gave better sensitivity than SVM on German Credit data set, and their specificity and accuracy are similar. When we changed λ to 2.0, sensitivity decreased but specificity increased. So there is a tradeoff between sensitivity and specificity balanced by λ.

The 15-attribute case and 10-attribute case are respectively shown in Figure 2 and Figure 3. The analyses of 15-attribute case and 10-attribute case are similar to preceding analysis. Between cases with different number of attributes, as we removed 5 qualitative attributes, there is only slight fluctuation on the performance.



(a) Cross-Validation

```
> klrresult_15        #with best lambda=0.2
Confusion Matrix and Statistics
          Reference
Prediction   0   1
         0  29  22
         1  61 188

               Accuracy : 0.7233
            Sensitivity : 0.32222
            Specificity : 0.89524
```

(b) KLR with best $\lambda = 0.2$

```
> svmresult_15
Confusion Matrix and Statistics
          Reference
Prediction   0   1
         0  15   6
         1  75 204

               Accuracy : 0.73
            Sensitivity : 0.1667
            Specificity : 0.9714
```

(c) SVM

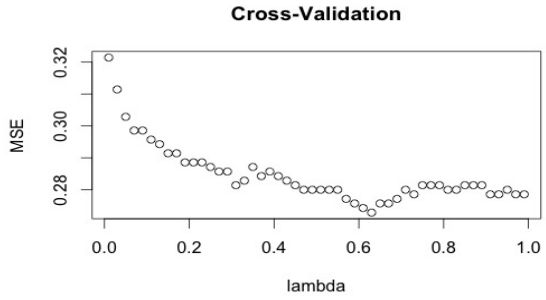
```
> klrresult_15        #lambda=2.0
Confusion Matrix and Statistics
          Reference
Prediction   0   1
         0  11   5
         1  79 205

               Accuracy : 0.72
            Sensitivity : 0.12222
            Specificity : 0.97619
```

(d) KLR with $\lambda = 2.0$

Figure 2

3.3 Comments

The credit classification performance of Kernel Logistic Regression is comparable with SVM and can be recommended as a method to evaluate credit approval. The advantages of Kernel Logistic Regression compared to SVM is that it provides not only the predicted class, but also the estimated posterior probability of the predicted class which can be used as a confidence measure and sometimes are more useful than the classifications like in credit risk scoring area. And Kernel Logistic Regression can be generalized naturally to M-class classification through kernel multi-logit regression. But Kernel Logistic Regression is computationally more expensive because of



(a) Cross-Validation

```
> klrresult_10      #with best lambda=0.6
Confusion Matrix and Statistics
          Reference
Prediction   0   1
         0  13  10
         1  81 196

               Accuracy : 0.6967
            Sensitivity : 0.13830
            Specificity : 0.95146
```

(b) KLR with best $\lambda = 0.6$

```
> svmresult_10
Confusion Matrix and Statistics
          Reference
Prediction   0   1
         0   7   6
         1  87 200

               Accuracy : 0.69
            Sensitivity : 0.07447
            Specificity : 0.97087
```

(c) SVM

```
> klrresult_10      #with lambda=2.0
Confusion Matrix and Statistics
          Reference
Prediction   0   1
         0  10   8
         1  84 198

               Accuracy : 0.6933
            Sensitivity : 0.10638
            Specificity : 0.96117
```

(d) KLR with $\lambda = 2.0$

Figure 3

the time complexity $O(N^3)$, and it is hard to specify parameter c.

If the credit data is extremely imbalanced, like the large credit card fraud data set, we should use specific method improvement to deal with this problem. For German Credit data set, we still did not get a good sensitivity in above three cases and original paper[1]. We tried considering class weight into SVM on 20-attribute case ($w_0/w_1 = N_1/N_0$, w_i is class weight and N_i is the number of class observations), and we got following better result.

```
> svmresult_20_weight
Confusion Matrix and Statistics
          Reference
Prediction   0   1
         0  66  58
         1  29 147

               Accuracy : 0.71
            Sensitivity : 0.6947
            Specificity : 0.7171
```

Figure 4: SVM with class weight

The accuracy is similar to preceding result, but sensitivity is largely improved, and the tradeoff between sensitivity and specificity is more balanced.

4 Robust Regularized Kernel Regression

In Kernel Ridge Regression, squared loss works quite well for observations in which error term is normal distribution and L2 norm constrains coefficients in order to avoid overfitting. But in practice, we should pay more attention to the data set, because Y may be skewed and cannot be assumed as normal distribution. In this situation, more frequent outliers will distort prediction because KRR will fit outliers so that the squared loss could decrease significantly. Thus, we need a loss function which can tolerate more outliers than squared loss. Besides, we also wish the loss function works better for error with mixture distribution, aggregate distribution and so on. The following parts will introduce a robust model with Huber loss called RRKR and then compare its performance with KRR.

4.1 Formula Deduction

A Kernel Ridge Regression model is the sum of squared loss and L2 norm, shown as follows:

$$\underset{\theta\in\mathbb{R}^{p'}}{argmin}\sum_{i=1}^{n}(Y_i-\phi(X_i)^\top\theta)^2+\lambda\langle\theta,\theta\rangle \tag{15}$$

We rewrite the whole function in the following form:

$$\underset{\theta\in\mathbb{R}^{p'}}{argmin}\sum_{i=1}^{n}V(Y_i,\phi(X_i)^\top\theta)^2+\lambda\langle\theta,\theta\rangle \tag{16}$$

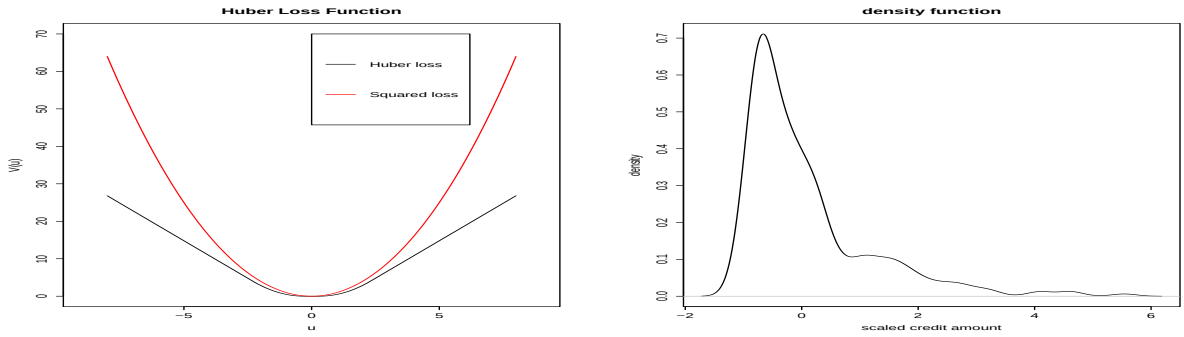
$V(Y_i,\phi(X_i)^\top\theta)$ is loss function. Usually we calculate loss with $u_i=Y_i-\phi(X_i)^\top\theta$, so loss function changes to $V(u_i)$. For KRR, $V(u_i)=u_i^2$.

In general, outliers are overemphasized using the squared loss in KRR. Although the absolute loss can avoid that problem, it overemphasizes loss on points close to the predicted points. Therefore Huber loss is applied to provide a compound solution, which takes the advantage of both loss functions. It is defined as follows:

$$V(Y_i,\phi(X_i)^\top)=V(u_i)=\begin{cases}m(2(u_i-\epsilon)-m) & u_i-\epsilon>m\\ u_i^2 & 0<u_i-\epsilon\le m\\ 0 & |u_i|\le\epsilon\\ u_i^2 & -m\le u+\epsilon<0\\ -m(2(u_i+\epsilon)+m) & u_i+\epsilon<-m\end{cases} \tag{17}$$

Figure 5a shows the difference between squared loss and Huber loss. Both lines are overlapped when u is small, but Huber loss function doesn't weight too much on large loss.

According to the Representer Theorem[4], there exists an optimum $f(x)$ among the set $\{f(x)=\phi(X_i)^\top\theta|\theta=\sum_i^n c_i\phi(X_i)\}$ for equation (16) with Huber loss. In other words, $f(x)=\sum_j^n K(x,x_j)c_j$. That's what we want and Newton iteration is applied to calculate the numeric



(a) loss function (b) density function for scaled credit amount

Figure 5

result. First, we substitute θ with $\sum_i^n c_i\phi(X_i)$ in equation (16).

$$\begin{aligned} E &= \sum_i^n V(y_i - \sum_j^n K(x_i, x_j)c_j) + \lambda \sum_{i,j}^n c_i c_j K(x_i, x_j) \\ &= \sum_i^n V(y_i - \sum_j^n K(x_i, x_j)c_j) + \lambda c^\top K c \end{aligned} \tag{18}$$

We rewrite $V(y - \sum_j^n K(x, x_j)c_j)$.

$$V(y - \sum_j^n K(x, x_j)c_j) = V(u) = \begin{cases} m(2(u-\epsilon) - m) & S_1 = \{x | u - \epsilon > m\} \\ (u-\epsilon)^2 & S_2 = \{x | 0 < u - \epsilon \leq m\} \\ 0 & S_3 = \{x | \quad |u| \leq \epsilon\} \\ (u+\epsilon)^2 & S_4 = \{-m \leq u + \epsilon < 0\} \\ -m(2(u+\epsilon) + m) & S_5 = \{u + \epsilon < -m\} \end{cases} \tag{19}$$

Then calculate the derivative.

$$\frac{\partial E}{\partial c} = 2(\lambda K c + K I^0 K c + K q) \tag{20}$$

where $q = -I^0 y + e$, I^0 is a n*n diagonal matrix with diagonal elements, that belong to S2 and S4, being 1 and other elements being 0. e is a vector defined as:

$$e_i = \begin{cases} m & x_i \in S_1 \\ \epsilon & x_i \in S_2 \\ 0 & x_i \in S_3 \\ -\epsilon & x_i \in S_4 \\ -m & x_i \in S_5 \end{cases} \tag{21}$$

The second order derivative is

$$\frac{\partial^2 E}{\partial c c^\top} = 2(\lambda K + K I^0 K) \tag{22}$$

Finally we calculate c by iterations with initial value c_0.

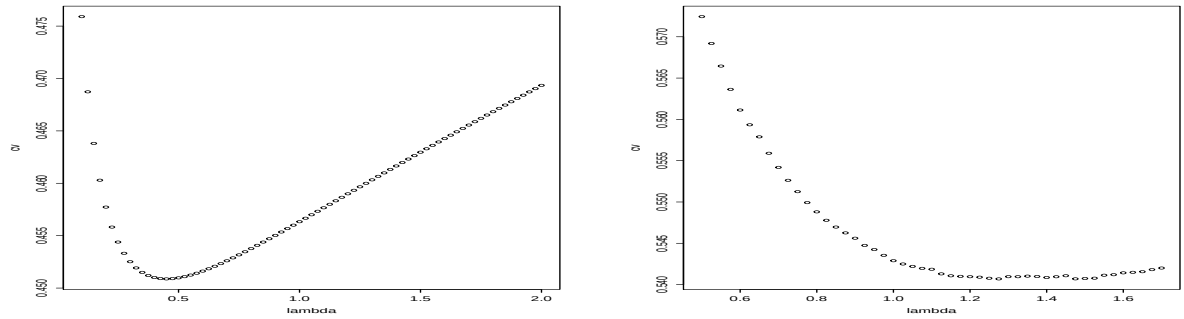
$$c^{(t+1)} = c^{(t)} - \left(\frac{\partial^2 E}{\partial c c^\top}\right)^{-1} \frac{\partial E}{\partial c} = \left(\lambda K + K I^0 K\right)^{-1} \left(\lambda K c^{(t)} + K I^0 K c^{(t)} + K q\right) \tag{23}$$

For each iteration, we partition data into $S_1, \ldots, S_5$ by calculating u through $c^{(t)}$. Actually, the above $f(x)$ can be solved by the first derivative directly. However, for the following prediction, we actually added an offset to $f(x)$ so that $f(x) = \phi(X_i)^\top + c_0$. Here Newton method is applied. For simplicity, we only display the formula without the offset.

4.2 Results

We aimed to predict credit amount given the known data. In reality, credit amount is calculated by bank but the formula is not simple linear combination, so predicting it with given predictors can test the performance of models. Here we mainly compared the performance of Kernel Ridge Regression and Robust Regularized Kernel Regression.

We scaled all continuous data so that $(-3.5\sigma, 3.5\sigma)$ should cover most credit amount. Thus we set $m = 3.5$. we set $\epsilon = 0.05$, which means two loss functions are similar with the difference less than 0.05. Figure 5b shows that the data is skewed so our former assumption may not work. First we calculated optimum λ by cross validation for 15-attribute case. Figure 6a shows MSE



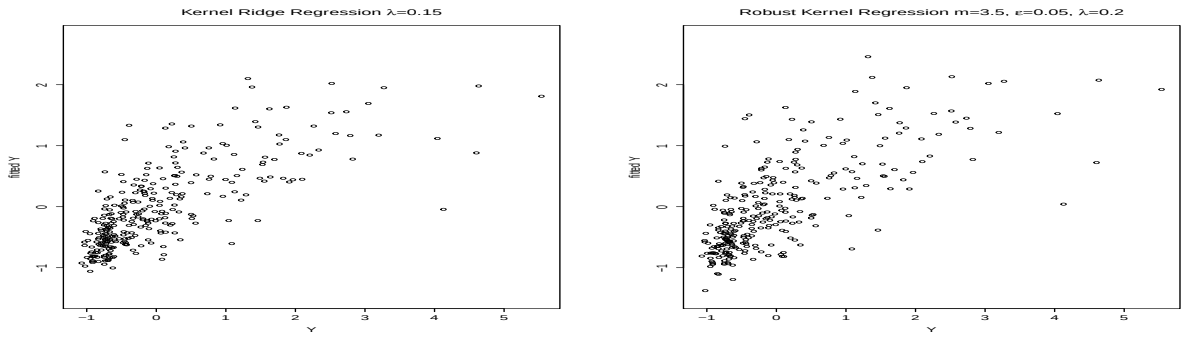
(a) CV for KRR (b) CV for RRKR

Figure 6

of Kernel Ridge Regression against λ while Figure 6b shows MSE of Robust Regularized Kernel Regression. So we took λ as 0.5, 1.2 respectively. It demonstrates that RRKR puts more weight on penalty for the optimum so that the model could be less overfitting than KRR.

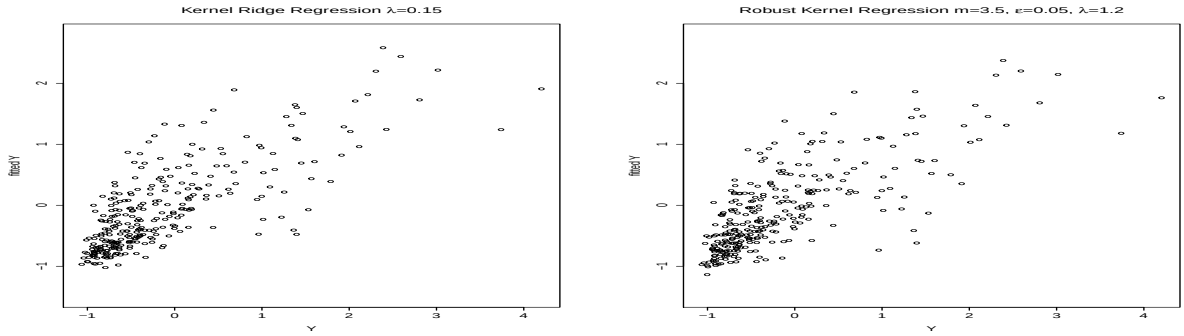
Figure 7a and Figure 7b show predicted credit amounts against real values. From figures, they seem to have similar performance. However, if we apply wilcoxon signed rank test[1] to check the distribution of real values with predicted values, p-values are 0.3681 and 0.0651 with respect to KRR and RRKR. So RRKR performs much better than KRR.

We also tested KRR and RRKR on the 10-attribute case. The result is similar. λs are still 0.5 and 1.2. However, for the prediction, both method improve a lot. Figure 8a and Figure 8b show the scatter plots. For wilcoxon signed rank test, p-values of KRR and RRKR are 0.0005584 and 0.0001922 respectively. It suggests that KRR and RRKR have similar performance, given a



(a) (b)

Figure 7



(a) (b)

Figure 8

proper number of variables. But in reality, it's hard to decide a proper number of variables so it is robust to apply RRKR. It performs at least as well as KRR does and prediction is not largely changed by larger number of variables.

4.3 Comments

In the last subsection, we see the advantage of Robust Regularized Kernel Regression. Here we discuss some problems when applying it. The first is time-consuming. When data grows, although its time complexity is polynomial, it takes much more time than KRR. Choosing m and ϵ is another problem. Maybe we can use cross validation. But in that case, we have three variables including λ so computation is large. A proper initial value is necessary, otherwise the final result may not converge. It is hard to guess the set of proper initial values when it is small. But overall RRKR is a good choice with high accuracy.

5 Conclusions

In this paper, we compared Support Vector Machine with Kernel Logistic Regression for discrete output like credit risk classification and compared Kernel Ridge Regression with Robust Regularized Kernel Regression for continuous output like credit amount prediction. For discrete case, we show that SVM and KLR have similar performance. For continuous case, we show that

the performance of RRKR is at least as good as the performance of KRR. When data is skewed, RRKR can still work well. The preceding sections include general theory frameworks, results and comments we concluded in practical problems.

Reference

[1] S. P. Rahayu, S. W. Purnami, A. Embong, "Applying Kernel Logistic Regression in Data Mining to Classify Credit Risk", Information Technology, 2008. ITSim 2008. International Symposium on (Volume:2), 26-28 Aug. 2008.

[2] Jianke Zhu, Steven C. H. Hoi, Michael Rung-Tsong Lyu, "Robust Regularized Kernel Regression", IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics: a publication of the IEEE Systems, Man, and Cybernetics Society (Impact Factor: 6.22), Dec. 2008.

[3] UCI Repository of Machine Learning Databases: http://archive.ics.uci.edu/ml/

[4] Bernhard Scholkopf, Ralf Herbrich, Alex J. Smola, "A Generalized Representer Theorem", Computational Learning Theory, Springer Berlin Heidelberg, 2001, pp. 416-426.

[5] Wilcoxon, Frank. "Individual comparisons by ranking methods."Biometrics bulletin 1.6 (1945): 80-83.