Project 3

Willie Langenberg

2021-12-13

Task 1

a)

Equation Eq.12.8 is

$$\min_{\beta,\beta_0} \frac{1}{2}||\beta||^2 + C\sum_{i=1}^{N} \xi_i,$$

with subject to $\xi_i \geq 0, \quad y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i \quad \forall i$ and equation Eq.12.25 is

$$\min_{\beta,\beta_0} \sum_{i=1}^{N} [1 - y_i f(x_i)]_+ + \frac{\lambda}{2}||\beta||^2,$$

with $f(x) = h(x)^T\beta + \beta_0$ according to the textbook. Using that $\lambda = 1/C$ we see that minimizing Eq.12.25 is equal to minimizing

$$\min_{\beta,\beta_0} \sum_{i=1}^{N} [1 - y_i f(x))]_+ + \frac{1}{2C}||\beta||^2 = \min_{\beta,\beta_0} C\sum_{i=1}^{N} [1 - y_i f(x)]_+ + \frac{1}{2}||\beta||^2$$

by replacing λ with $1/C$, and also realizing that we can multiply the equation with C since it would not change the minimization. We also know that $[1 - y_i f(x_i)]_+ = 1 - y_i f(x_i) \quad$ if $\quad y_i f(x_i) < 1$, else 0. We now set $\xi_i \geq 1 - y_i f(x_i)$ and $\xi > 0$ for all i. Now we could replace $[1 - y_i f(x_i)]_+$ with ξ_i and the equation is now equal to

$$\min_{\beta,\beta_0} C\sum_{i=1}^{N} \xi_i + \frac{1}{2}||\beta||^2$$

with subject to

$$\xi_i \geq 0, \quad \xi_i \geq 1 - y_i(h(x_i)^T\beta + \beta_0) \quad \forall i.$$

The only difference from Eq.12.8 is that instead of using x_i we now use $h(x_i)$ which doesn't matter when optimizing the function with respect to β and β_0.

b)

Binomial Deviance

We start to derive the minimizing function for the binomial deviance. The aim is to find the function $f(x)$ that minimizes $E[L[y, f(x)]]$. The loss function for binomial deviance is given by

$$L[y, f(x)] = log[1 + e^{-yf(x)}].$$

Since we know that the outcome of y is either $+1$ or -1 we can write the expected value as

$$E[L[y, f(x)]] = P(Y = +1|x)\ log[1 + e^{-f(x)}] + P(Y = -1|x)\ log[1 + e^{f(x)}].$$

To find the minimal value of this we derives $E[L[y, f(x)]]$ with respect on $f(x)$

$$\frac{d}{df(x)}E[L[y, f(x)]] = \frac{P(Y = +1|x)}{1 + e^{-f(x)}}(-e^{-f(x)}) + \frac{P(Y = -1|x)}{1 + e^{f(x)}}e^{f(x)},$$

which after some work simplifies to

$$\frac{d}{df(x)}E[L[y, f(x)]] = \frac{P(Y = -1|x)e^{f(x)} - P(Y = +1|x)}{1 + e^{f(x)}}.$$

It is easy to see that to solve for the derivative equal to 0 we get the solution

$$f(x) = log\frac{P(Y = +1|z)}{P(Y = -1|z)}.$$

SVM Hinge Loss

The SVM Hinge Loss is given by the function

$$L[y, f(x)] = [1 - yf(x)]_+.$$

Further, the expected value of the loss function is given by

$$E[L[y, f(x)]] = P(Y = +1|x)[1 - f(x)]_+ + P(Y = -1|x)[1 + f(x)]_+.$$

We start by analyzing the expected value when $P(Y = +1|x) > P(Y = -1|x)$. First of all we can see that the expected value is having a negative slope in $f(x)$ when $f(x) \leq -1$ since $E[L[y, f(x)]] = P(Y = +1|x)[1 - yf(x)]$ and would be minimized by $f(x) = -1$. It is the same for $f(x) \geq 1$ but the slope is positive in $f(x)$ and thus minimized by $f(x) = 1$. For the interval $-1 \leq f(x) \leq 1$ the expected value is minimized by 1 since the slope would be negative in $f(x)$ between the values -1 and 1. This is illustrated by

$$P(Y = +1|x)[1 - f(x)] + P(Y = -1|x)[1 + f(x)] = 1 + (P(Y = -1|x) - P(Y = +1|x))f(x)$$

when $-1 \leq f(x) \leq 1$. We then know that the minimizing function of the expected value is $f(x) = 1$ if $P(Y = +1|x) > P(Y = -1|x)$ and that $f(x) = -1$ if $P(Y = +1|x) < P(Y = -1|x)$ using the fact that $P(Y = +1|x) + P(Y = -1|x) = 1$. When $P(Y = +1|x) = P(Y = -1|x)$ we know that the minimizing function is anything between -1 and 1. It follows that the minimizing function can be simplified to

$$f(x) = \left\{ \begin{array}{ll} 1, & \text{for } P(Y = +1|x) > 0.5 \\ -1, & \text{for } P(Y = +1|x) < 0.5 \\ 0, & \text{for } P(Y = +1|x) = 0.5 \end{array} \right\} = \text{sign}[P(Y = +1|x) - \frac{1}{2}]$$

c)

When changing γ we are essentially ignoring or taking into consideration of observations near or far away from the decision boundary. So when γ increases, we are taking the closest observations more into consideration. In the opposite way, if γ shrinks, we would consider points that are farther away as well. In terms of bias-variance tradeoff, I would suggest that the bias is small when γ is big since we are more prone to fit the decision to a few points near the boundary, to make them correct. For another set those points might look very different thus a larger variance between datasets. In the same way we would get larger bias but lower variance for smaller values of γ, since we are sort of looking at the data from a bigger picture.

I.e if an test observation is far away from a training observation, the exponent becomes a large negative, and then the $K(x, x')$ is close to zero. Depending on the γ we can control this diminishing relevance of observations far away.

Task 2

a)

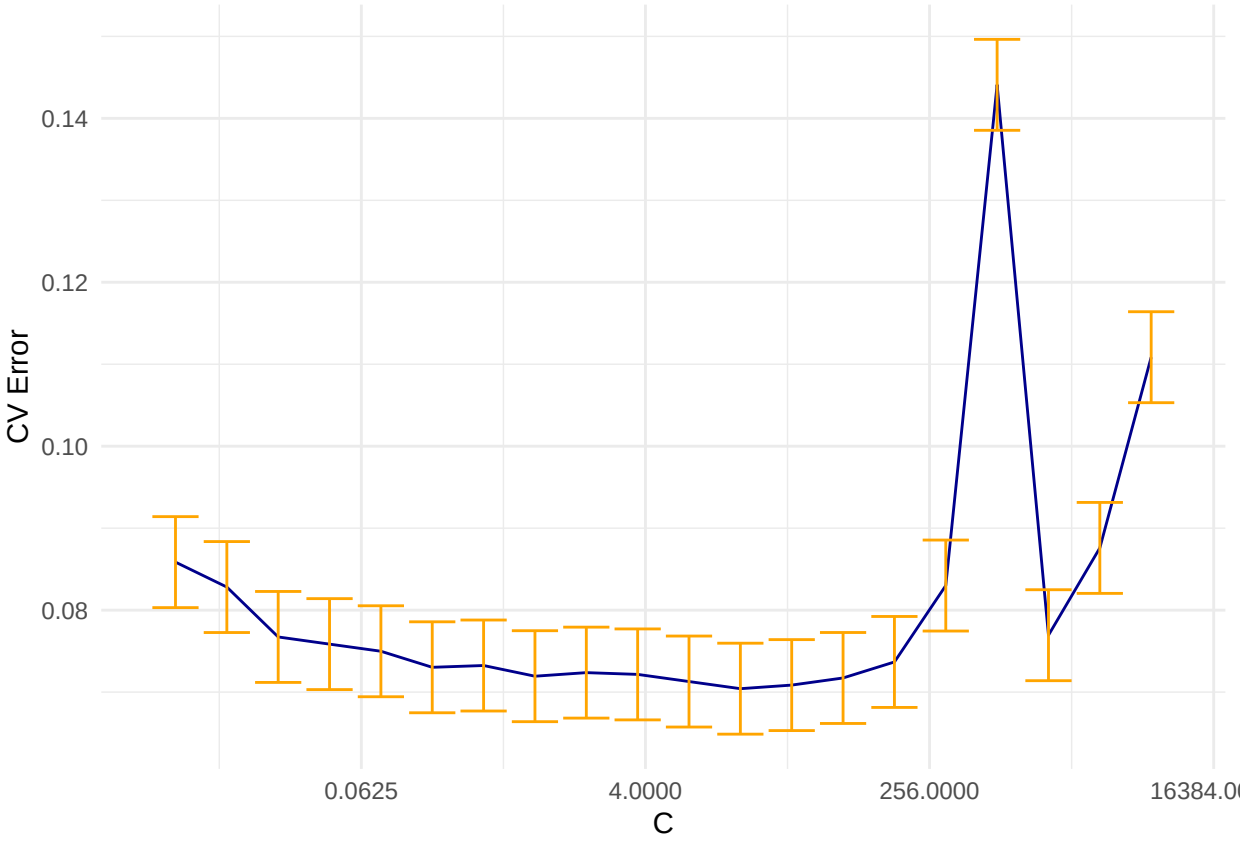
I chose the package **caret** and used the function **svmLinear**. This function is using the linear kernel $K(x, x') =< x, x' >$. I also tried the radial basis kernel but recieved significantly higher cross-validation error. The only parameter of choice is C, which I decided by a 10-fold cross-validation. I chose this kernel because I thought it to be a good kernel to begin with, and also that it might reduce the training time.

b)

The code is seen in the appendix.

c)

I constructed a 10-fold cross-validation plot to compare models using different values of C. I tried values of C from around 0.01 - 1000, with 20 steps. In the plot below we can see the cross-validation error and it's standard error for all models. Using the one-standard-error rule one would pick the most parsimonious model within one standard error from the best performing model. The best perfoming model uses $C = 16$, and moving to the left within the standard error bar we would use $C = 0.039$ as our model of choice. This model has CV error at 0.0759.



d)

The CV error is somewhat higher for the support vector machine in comparison to the boosted tree model in project 2. The SVM model are dependent on what kernel to use, so I think that we could gain some accuracy by testing other kernels. Probably the radial basis kernel would be better with some more tuning, since I only tried with $.\sigma = 1$ and solely tuned C by cross-validation.

Appendix

```
## Reading in the data
df <- read.table('../Data/data.txt')
df$V58 = as.factor(df$V58)

## Shuffling the data
set.seed(9907)
shuffled_df <- df[sample(nrow(df)), ]

## Using Caret To Find C Using 10-fold CV
control <- trainControl(method = "cv", number = 10, search = 'grid')
tunegrid <- expand.grid(.C = 3**seq(-5.5,8,length.out = 20), .sigma = 1)
svm_gridsearch5 <- train(V58 ~.,
                         data = shuffled_df,
                         method = 'svmLinear',
                         metric = 'Accuracy',
                         tuneGrid = tunegrid,
                         trControl = control)

save(svm_gridsearch5, file="svm_gridsearch5.RData")
#load("svm_gridsearch5.RData")

## Code to  add cverror and its standard error.
## Also plot the Cverror against C plot with SE bars.
svm_gridsearch5$results %>%
  mutate(cverror = 1-Accuracy, se = sd(cverror)/sqrt(10)) %>%
  ggplot(aes(x=C, y= cverror)) + theme_minimal() +
  geom_line(color="darkblue") +
  geom_errorbar(aes(ymin=cverror-se, ymax=cverror+se), color="orange") +
  xlab("C") + ylab("CV Error") + scale_x_continuous(trans = log2_trans())
```