Assessment_2

2023-03-15

Kernel Methods For Regression

Part 1

Question 1

The Gaussian kernel is an example of an kernel for which the model is identifiable. The model is unidentifiable for the kernel: $k(x, y) = 1$ for $x, y \in \mathbb{R}^p$. This kernel is positive semi-definite and by the Moore-Aronszajn theorem there exists a unique RKHS for which k is the reproducing kernel. In this RKHS all the functions are constant, let's pick two functions $f_1, f_2 \in H_k$ where $f_1(x) = c$, $f_2(x) = d$ for all $x \in \mathcal{X}$. Then if we let the α we use with f_2, $\alpha_2 = \alpha_1 - d + c$, where α_1 is the α we use with f_1, then these are the same model. Hence our model is unidentifiable.

Question 2

We have for some $f \in H_k$: $f = f_1 + f_2$, for some $f_1 \in \tilde{H}, f_2 \in \tilde{H}^{\perp}$. By orthogonality:

$$||f_1 + f_2||^2 = ||f_1||^2 + ||f_2||^2$$

And by the reproducing property:

$$\frac{1}{2n}\sum_{i=1}^{n} \log f(y_i; g^{-1}(\alpha + (f_1 + f_2)(x_i^0)), \phi)$$

$$= \frac{1}{2n}\sum_{i=1}^{n} \log f(y_i; g^{-1}(\alpha + f_1(x_i^0)), \phi)$$

Combining the above,

$$\frac{1}{2n}\sum_{i=1}^{n} \log f(y_i; g^{-1}(\alpha + (f_1 + f_2)(x_i^0)), \phi) - \lambda||f_1 + f_2||^2$$

$$= \frac{1}{2n}\sum_{i=1}^{n} \log f(y_i; g^{-1}(\alpha + f_1(x_i^0)), \phi) - \lambda||f_1||^2 + \lambda||f_2||^2$$

$$\geq \frac{1}{2n}\sum_{i=1}^{n} \log f(y_i; g^{-1}(\alpha + f_1(x_i^0)), \phi) - \lambda||f_1||^2$$

Hence, we have that $\hat{f}_\lambda \in \tilde{H}$ and therefore can write $\hat{f}_\lambda$ as a linear combination of $k(x_1^0, \cdot), ..., k(x_n^0, \cdot)$ therefore we can write:

$$\hat{f}_\lambda = \sum_{i=1}^{n} \hat{\beta}_{\lambda,i} k(x_i^0, \cdot)$$

Question 3

We have that,

$$\begin{aligned}
-\lambda||f||^2_{H_k} &= -\lambda||\sum_{i=1}^{n}\beta_{\lambda,i}k(x_i^0,\cdot)||^2_{H_k} \\
&= -\lambda < \sum_{i-1}^{n}\beta_{\lambda,i}k(x_i^0,\cdot), \sum_{j=1}^{n}\beta_{\lambda,j}k(x_j^0,\cdot) >_k \\
&= -\lambda \sum_{i-1}^{n}\beta_{\lambda,i} < k(x_i^0,\cdot), \sum_{j=1}^{n}\beta_{\lambda,j}k(x_j^0,\cdot) >_k \\
&= -\lambda \sum_{i-1}^{n}\sum_{j=1}^{n}\beta_{\lambda,i}\beta_{\lambda,j} < k(x_i^0,\cdot), k(x_j^0,\cdot) >_k \\
&= -\lambda \sum_{i-1}^{n}\sum_{j=1}^{n}\beta_{\lambda,i}\beta_{\lambda,j} k(x_i^0, x_j^0) \\
&= -\lambda\beta_\lambda^T K\beta_\lambda
\end{aligned}$$

Where $K = [k(x_i^0, x_j^0)]_{i,j}$. Hence we can rewrite (2) as:

$$\frac{1}{2n}\sum_{i=1}^{n}\log f(y_i; g^{-1}(\alpha + \beta_\lambda^T \cdot K^{(i)}), \phi) - \lambda\beta_\lambda^T K\beta_\lambda$$

Question 4

Let $m < n+2$, consider the optimization problem posed in the previous question, the Nyonstrom method involves reducing the dimension of the gram matrix, K, hence reducing the dimensionality of the optimization problem. We will approximate the kernel, k, by $\tilde{k}^{(m)}$ such that the matrix $\tilde{K}^{(m)}$ obtained by replacing k with $\tilde{k}^{(m)}$ has rank $\leq m$. We will let,

$$\tilde{k}^{(m)}(x, x') = k_m(x)^T(K_m)^{-1}k_m(x')$$

where K_m is the first m rows and columns of K and

$$k_m(x) = (k(x_1^0, x), ..., k(x_m^0, x))$$

Let's now rewrite f using our new kernel:

$$\begin{aligned}
f_\lambda(x) &\approx \beta_\lambda^T\tilde{k}^{(m)}(x) \\
&= \beta_\lambda^T K(X_{1:m}, X)^T(K_m^0)^{-1}K(X_{1:m}, x) \\
&= \gamma(K_m^0)^{-1}K(X_{1:m}, x)
\end{aligned}$$

where $K(A,B) = [k(a_i, b_j)]_{i,j}$, X is our data matrix where $X_{1:m}$ means the data matrix including only the first m datapoints and we let $\gamma = \beta_\lambda^T K(X_{1:m}, X)^T$. Let's now rewrite the penalty:

$$\begin{aligned}
-\lambda\beta_\lambda^T K\beta_\lambda &\approx -\lambda\beta_\lambda^T\tilde{K}^{(m)}\beta_\lambda \\
&= -\lambda\beta_\lambda^T K(X_{1:m}, X)^T(K_m^0)^{-1}K(X_{1:m}, X)\beta_\lambda \\
&= -\lambda\gamma(K_m^0)^{-1}\gamma^T
\end{aligned}$$

This leaves us with the following optimization problem,

$$\arg\max_{\alpha\in\mathbb{R},\phi\in(0,\text{inf}),\gamma\in\mathbb{R}^m}\frac{1}{2n}\sum_{i=1}^{n}\log f(y_i; g^{-1}(\alpha + \gamma(K_m^0)^{-1}K(X_{1:m}, x_i^0)), \phi) - \lambda\gamma(K_m^0)^{-1}\gamma^T$$

Question 5

#TODO: change Km0 to inverse? Let's first find the spectral decomposition of K_m^0,

$$K_m^0 = S\Lambda S^{-1}$$

Then we can write,

$$\gamma^T K_m^0 \gamma = \gamma^T S^{-T} \Lambda S^{-1} \gamma \quad = ||\Lambda^{\frac{1}{2}} S^{-1} \gamma||_2^2$$

Glmnet estimates parameters that minimize the following (if using the ridge penalty):

$$-\frac{1}{n}\sum_{i=1}^{n} \log f(y_i; g^{-1}(\alpha + \gamma X_i), \phi) + \frac{\lambda}{2}||\gamma_{glm}||_2^2$$

which is the same as maximizing optimization problem as ours, except for if we instead try to find the minimum of the negative of our optimization problem, a factor of two and if we substitute for γ_{glm}. We have that,

$$\gamma_{glm} = \Lambda^{\frac{1}{2}} S \gamma$$

$$\Rightarrow \gamma = S^{-1} \Lambda^{-\frac{1}{2}} \gamma_{glm}$$

Therefore we can find our value for γ using the estimate we get from glmnet where for X_i we use $(K_m^0)^{-1} K(X_{1:m}, x_i^0)$.

Part 2

We are going to use the wesdr dataset:

```
library(gss)
data(wesdr)
head(wesdr)
```

```
##    dur  gly  bmi ret
## 1 10.3 13.7 23.8   0
## 2  9.9 13.5 23.5   0
## 3 15.6 13.8 24.8   0
## 4 26.0 13.0 21.6   1
## 5 13.8 11.1 24.6   1
## 6 31.1 11.3 24.6   1
```

Let's now split it into a testing and training set:

```
n.test <- round(0.15 * nrow(wesdr))
test_ind <- sample(seq_len(nrow(wesdr)), size = n.test)

train <- wesdr[-test_ind, ]
test <- wesdr[test_ind, ]
```

Question 6

We are going to use a binomial distribution as we are modelling a response variable that is either 0 or 1, we are going to set $\alpha = 0$ so that we are using the ridge penalty and we will use the radial basis kernel function for which the model is identifiable.

```
library(glmnet)
library(kernlab)

gaussian_kernel <- function(x, y, sigma) {
  exp(-sum((x - y)^2) / (2*sigma^2))
```

```
}

fit_model <- function(X, y, lambda, sigma, m){
  n <- nrow(X)
  rbf <- rbfdot(sigma = sigma)
  K <- kernelMatrix(rbf, X)
  K_m_inverse <- solve(K[1:m, 1:m])
  K_mn <- matrix(0, m, n)
  X_m <- X[1:m,]
  for (i in 1:m) {
    for (j in 1:n) {
    K_mn[i,j] <- gaussian_kernel(X_m[i,], X[j,], sigma)
    }
  }
  input <- K_m_inverse %*% K_mn
  print(dim(input))
  results <- glmnet(input, y, family = "binomial" , alpha = 0, lambda = lambda)
  gamma_glm <- results$beta

  eig <- eigen(K_m_inverse)
  S <- eig$vectors
  L <- diag(eig$values)

  gamma <- solve(S) %*% sqrt(solve(L)) %*% gamma_glm
}

X <- as.matrix(train[,-4])
y <- train[,4]
fit_model(X, y, 0.1, 1, nrow(train)-5)
```