STATS790HW1

Hanwen Ju(400180920)

2023-01-21

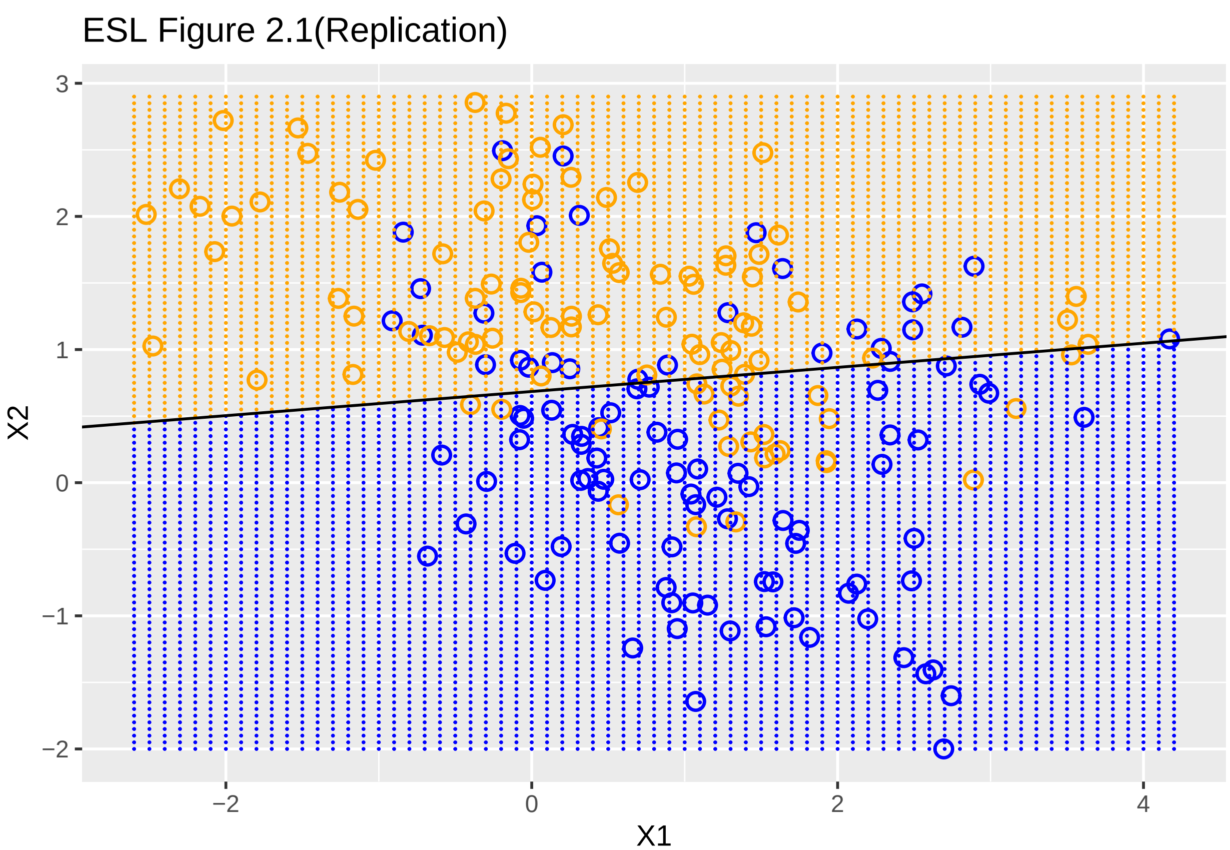
Question 1

In the last two sections of the article, Professor Breiman pointed out that algorithmic models usually give better accuracy than data models and are able to discover some relation behind data that data models may not discover. Also, higher accuracy usually means a more reliable relation behind data is discovered. Do algorithmic models or focusing on accuracy always be a good choice? I think data models can also give some useful information that algorithmic can not give. For instance, Model-based clustering can give a mixture distribution underlining data and soft assignment where each data observation has a probability of belonging to each cluster. Latent variable analysis can discover hidden components which most other algorithmic models can not discover.

Question 2

```
library("ggplot2")
library(dplyr)

#read and process data
load(file = "ESL.mixture.rda")
df.q2.x <- as.matrix(ESL.mixture$x)
df.q2.y <- as.matrix(ESL.mixture$y)
df.q2 <- as.data.frame(matrix(nrow = length(df.q2.y), ncol = 3))
df.q2[,1:2] <- df.q2.x
df.q2[,3] <- df.q2.y
df.q2 <- df.q2 %>%
  mutate(Color = ifelse(V3 == 0, "blue",
                        ifelse(V3 == 1, "orange", "none")))


xnew <- as.matrix(ESL.mixture$xnew)
#df.xnew <- as.data.frame(xnew)




#Linear regression

#Add column of 1 for beta 0
df.q2.x <- cbind(rep(1,nrow(df.q2.x)), df.q2.x)


#Train
#Coefficients
```

```r
beta.q2 <- (solve(t(df.q2.x) %*% df.q2.x) %*% t(df.q2.x)) %*% df.q2.y


#Test
xnew <- cbind(rep(1,nrow(xnew)), xnew)
y.xnew <- xnew %*% beta.q2

#Change color
y.xnew <- ifelse(y.xnew < 0.5, 0, 1)
xynew <- as.data.frame(cbind(xnew,y.xnew))
xynew <- xynew %>%
  mutate(Color = ifelse(xynew[,4] == 0, "blue",
                        ifelse(xynew[,4] == 1, "orange", "none")))



#Create Figure
ggplot() + geom_point(data = df.q2,aes(x=V1,y=V2,col=Color),shape = 1, size = 2.5, stroke=0.9) +
  geom_point(data = xynew, aes(x=x1,y=x2,col=Color),size = 0.01) +

  geom_abline(slope = -beta.q2[2]/beta.q2[3],
              intercept = (0.5 - beta.q2[1])/beta.q2[3]) +
  scale_color_identity() +

  ggtitle("ESL Figure 2.1(Replication)") +
  xlab("X1") + ylab("X2")
```



Question 6

```
#ESL 2.8
df.train <- read.table("zip.train")
df.train <- as.data.frame(df.train)
df.train <- df.train[df.train$V1==2 | df.train$V1==3,]

df.test <- read.table("zip.test")
df.test <- as.data.frame(df.test)
df.test <- df.test[df.test$V1==2 | df.test$V1==3,]

#Create a table to store acurracy:
acc.table <- as.data.frame(matrix(nrow = 6,ncol = 2))


#linear regression:

y.zip.train <- as.matrix(df.train[,1])
y.zip.train01 <- ifelse(y.zip.train==2, 0, 1)
x.zip.train <- as.matrix(cbind(rep(1,nrow(df.train)), df.train[,-1]))

y.zip.test <- as.matrix(df.test[,1])
x.zip.test <- as.matrix(cbind(rep(1,nrow(df.test)), df.test[,-1]))


#Coefficients
beta <- (solve(t(x.zip.train) %*% x.zip.train) %*% t(x.zip.train)) %*% y.zip.train01

#training error
pred.zip.train <- x.zip.train %*% beta
pred.zip.train <- ifelse(pred.zip.train < 0.5,2,3)
acc.table[1,1] <- 1 - mean(pred.zip.train == y.zip.train)

#test error
pred.zip.test <- x.zip.test %*% beta
pred.zip.test <- ifelse(pred.zip.test < 0.5,2,3)
acc.table[1,2] <- 1 - mean(pred.zip.test == y.zip.test)









#k-nearest neighbour
df.train.zip <- df.train[,-1]
v1.train <- df.train[,1]

df.test.zip <- df.test[,-1]
v1.test <- df.test[,1]

library(FNN)
#k = 1
knn.zip1.train <- knn(
```

```
  train = df.train.zip,
  test = df.train.zip,
  v1.train,
  k = 1
)

#train error
acc.table[2,1] <- 1 - mean(knn.zip1.train == v1.train)


knn.zip1.test <- knn(
  train = df.train.zip,
  test = df.test.zip,
  v1.train,
  k = 1
)

#test error
acc.table[2,2] <- 1 - mean(knn.zip1.test == v1.test)
colnames(acc.table) <- c("Training Error", "Test Error")
rownames(acc.table) <- c("Linear Regression","KNN k=1","KNN k=3","KNN k=5","KNN k=7","KNN k=15")



#k = 3
knn.zip3.train <- knn(
  train = df.train.zip,
  test = df.train.zip,
  v1.train,
  k = 3
)

#train error
acc.table[3,1] <- 1 - mean(knn.zip3.train == v1.train)


knn.zip3.test <- knn(
  train = df.train.zip,
  test = df.test.zip,
  v1.train,
  k = 3
)

#test error
acc.table[3,2] <- 1 - mean(knn.zip3.test == v1.test)






#k = 5
knn.zip5.train <- knn(
  train = df.train.zip,
```

```
  test = df.train.zip,
  v1.train,
  k = 5
)

#train error
acc.table[4,1] <- 1 - mean(knn.zip5.train == v1.train)


knn.zip5.test <- knn(
  train = df.train.zip,
  test = df.test.zip,
  v1.train,
  k = 5
)

#test error
acc.table[4,2] <- 1 - mean(knn.zip5.test == v1.test)




#k = 7
knn.zip7.train <- knn(
  train = df.train.zip,
  test = df.train.zip,
  v1.train,
  k = 7
)

#train error
acc.table[5,1] <- 1 - mean(knn.zip7.train == v1.train)


knn.zip7.test <- knn(
  train = df.train.zip,
  test = df.test.zip,
  v1.train,
  k = 7
)

#test error
acc.table[5,2] <- 1 - mean(knn.zip7.test == v1.test)





#k = 15
knn.zip15.train <- knn(
  train = df.train.zip,
  test = df.train.zip,
  v1.train,
```

```r
  k = 15
)

#train error
acc.table[6,1] <- 1 - mean(knn.zip15.train == v1.train)


knn.zip15.test <- knn(
  train = df.train.zip,
  test = df.test.zip,
  v1.train,
  k = 15
)

#test error
acc.table[6,2] <- 1 - mean(knn.zip15.test == v1.test)
```

```r
library(knitr)

kable(acc.table,caption = "Accuracy table")
```

Table 1: Accuracy table

	Training Error	Test Error
Linear Regression	0.0057595	0.0412088
KNN k=1	0.0000000	0.0247253
KNN k=3	0.0050396	0.0302198
KNN k=5	0.0057595	0.0302198
KNN k=7	0.0064795	0.0329670
KNN k=15	0.0093593	0.0384615

3. ADA problem 1.2

- proof 1:

$MAE(m) = E[|Y-m|]$

$\because |Y-m| = \begin{cases} Y-m & \text{if } Y>m \\ m-Y & \text{if } m>Y \end{cases}$

$\frac{\partial}{\partial m}|Y-m| = \begin{cases} -1 & \text{if } Y>m \\ 1 & \text{if } m>Y \end{cases}$

we can create 2 indicator function $I\{m>Y\}$, $I\{Y>m\}$

$I\{m>Y\} = \begin{cases} 1 & \text{if } m>Y \\ 0 & \text{otherwise} \end{cases}$ $\quad I\{Y>m\} = \begin{cases} 1 & \text{if } Y>m \\ 0 & \text{otherwise.} \end{cases}$

$\therefore \frac{\partial}{\partial m}|Y-m| = 1\cdot I\{m>Y\} - 1\cdot I\{Y>m\}$

we want to minimize $MAE(m)$:

$$\frac{\partial}{\partial m}E[|Y-m|] = E[\frac{\partial}{\partial m}|Y-m|] = 0$$

$$= E[1\cdot I\{m>Y\} - 1\cdot I\{Y>m\}]$$

$$= E[I\{m>Y\}] - E[I\{Y>m\}]$$

$$= P(Y<m) - P(Y>m) = 0$$

$\because E[I\{m>Y\}]$
$= 1\cdot P(m>Y) + 0\cdot P(m\le Y)$
$= P(m>Y)$

$\therefore P(Y<m) = P(Y>m)$

$\because P(Y<m) + P(Y>m) = 1$

$\therefore P(Y<m) = P(Y>m) = 0.5$ Hence, m must be median

- proof 2:

Assume we have n samples, and n large enough.

Order sample as $y_1 \le y_2 \le y_3 \le \cdots \le y_n$, and we want to minimize $MAE(m) = E[|Y-m|]$.

$$
\begin{aligned}
MAE(m) &= E[|Y-m|] \\
&= \frac{1}{n}\sum_{i}^{n}|y_i - m| \qquad (\text{Assuming } y_i \le m \le y_{i+1}) \\
&= \frac{1}{n}\left[\sum_{k=1}^{i}(m-y_k) + \sum_{k=i+1}^{n}(y_k - m)\right] \\
&= \frac{1}{n}\left[\sum_{k=1}^{i} m - \sum_{k=1}^{i} y_k + \sum_{k=i+1}^{n} y_k - \sum_{k=i+1}^{n} m\right] \\
&= \frac{1}{n}\sum_{k=1}^{i} m - \frac{1}{n}\sum_{k=1}^{i} y_k + \frac{1}{n}\sum_{k=i+1}^{n} y_k - \frac{1}{n}\sum_{k=i+1}^{n} m
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial}{\partial m} MAE(m) &= \frac{\partial}{\partial m}\left(\frac{1}{n}\sum_{k=1}^{i} m - \frac{1}{n}\sum_{k=1}^{i} y_k + \frac{1}{n}\sum_{k=i+1}^{n} y_k - \frac{1}{n}\sum_{k=i+1}^{n} m\right) = 0 \\
&= \frac{\partial}{\partial m}\left(\frac{1}{n} im - \frac{1}{n}(n-i)m\right) \\
&= \frac{1}{n}\frac{\partial}{\partial m}(im - (nm - im)) \\
&= \frac{1}{n}\frac{\partial}{\partial m}(2im - nm) \\
&= \frac{1}{n}\frac{\partial}{\partial m}(2i-n)m \\
&= \frac{1}{n}(2i-n)
\end{aligned}
$$

$$
\begin{aligned}
\therefore \quad \frac{1}{n}(2i-n) &= 0 \\
2i - n &= 0 \\
2i &= n \\
i &= \frac{n}{2}
\end{aligned}
$$

$\therefore$ any m satisfy $y_{\frac{n}{2}} \le m \le y_{\frac{n}{2}+1}$ will give the minimum.

so pick m = median will always satisfy the above condition. Thus, median minimize MAE.

MSE is more sensitive to the outliers, because squaring will give a higher importane on outliers. If you don't want to focus the outliers. MAE will be a more robust choice.

4. ADA question 1.7:

Consider global mean as linear smoother. Assuming there are n training samples. $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$

$$\therefore \hat{\mu}(x_j) = \frac{1}{n}(y_1 + y_2 + \cdots + y_n)$$
$$= \frac{1}{n}y_1 + \frac{1}{n}y_2 + \cdots + \frac{1}{n}y_n$$
$$= \sum_i \frac{1}{n} y_i$$
$$= \sum_i y_i \, \hat{w}(x_i, x_j) \quad \text{where } \hat{w}(x_i, x_j) = \frac{1}{n} \text{ for any } x_i, x_j$$

$\therefore$ Consider the matrix form $\hat{\mu}$:

$$\hat{\mu} = \begin{bmatrix} \sum_i y_i \hat{w}(x_i, x_1) \\ \sum_i y_i \hat{w}(x_i, x_2) \\ \vdots \\ \sum_i y_i \hat{w}(x_i, x_n) \end{bmatrix} = \begin{bmatrix} \hat{w}(x_1, x_1) & \hat{w}(x_2, x_1) & \cdots & \hat{w}(x_n, x_1) \\ \hat{w}(x_1, x_2) & \hat{w}(x_2, x_2) & \cdots & \hat{w}(x_n, x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{w}(x_1, x_n) & \hat{w}(x_2, x_n) & \cdots & \hat{w}(x_n, x_n) \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$$= wy$$

$$w = \begin{bmatrix} \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} \\ \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} \end{bmatrix} \quad (w \text{ is a } n \times n \text{ matrix})$$

By the definition of (1.70)

$$df(\hat{\mu}) = tr\, w$$
$$= \sum_i^n \frac{1}{n}$$
$$= \frac{1}{n} + \frac{1}{n} + \cdots + \frac{1}{n}$$
$$= 1$$

Hence $\hat{\mu}$ has one degree of freedom.

5. ADA question 1.8

Consider knn regression as a linear smoother: Assuming there are n training samples. $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$

$$\hat{w}(x_i, x_j) = \begin{cases} \frac{1}{k} & x_i \text{ one of the } k \text{ nearest neighbors of } x_j \\ 0 & \text{otherwise} \end{cases}$$

$$\hat{\mu} = \begin{bmatrix} \sum_i y_i \hat{w}(x_i, x_1) \\ \sum_i y_i \hat{w}(x_i, x_2) \\ \vdots \\ \sum_i y_i \hat{w}(x_i, x_n) \end{bmatrix} = \begin{bmatrix} \hat{w}(x_1, x_1) & \hat{w}(x_2, x_1) & \cdots & \hat{w}(x_n, x_1) \\ \hat{w}(x_1, x_2) & \hat{w}(x_2, x_2) & \cdots & \hat{w}(x_n, x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{w}(x_1, x_n) & \hat{w}(x_2, x_n) & \cdots & \hat{w}(x_n, x_n) \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$$= wy$$

$\therefore$ for $k \geq 1$, we must have the diagnal $\hat{w}(x_i, x_i) = \frac{1}{k}$

$$\therefore \quad W = \begin{bmatrix} \frac{1}{k} & \hat{w}(x_2, x_1) & \cdots & \hat{w}(x_n, x_1) \\ \hat{w}(x_1, x_2) & \frac{1}{k} & \cdots & \hat{w}(x_n, x_2) \\ & \vdots & \ddots & \vdots \\ \hat{w}(x_1, x_n) & \hat{w}(x_2, x_n) & \cdots & \frac{1}{k} \end{bmatrix}$$

By definition of (1.70)

$$\boxed{df(\hat{\mu}) = tr(w) = \sum_i^n \frac{1}{k} = \frac{n}{k}}$$

If $k=n$, the knn regression smoother will become global mean smoother. $df(\hat{\mu}) = \frac{n}{n} = 1$

$\hat{w}(x_i, x_j) = \frac{1}{n}$ for all i, j

Hence, the influence matrix w will be:

$$w = \begin{bmatrix} \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} \\ \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} \end{bmatrix}$$

$$\boxed{df(\hat{\mu}) = tr(w) = \sum_i^n \frac{1}{n} = 1}$$