STOCKHOLMS UNIVERSITET
MATEMATISKA INSTITUTIONEN
Avd. Beräkningsmatematik

LÖSNINGAR
Introduktion till maskininlärning
17 mars 2021

Lösningar
Introduktion till maskininlärning
17 mars 2021 8–14

Uppgift 1

a.) Låg bias, hög varians.

b.) Låg bias, låg varians.

c.) Låg bias, hög varians.

d.) Hög bias, låg varians.

e.) Låg bias, hög varians.

f.) Hög bias, låg varians.

Uppgift 2

a.)

$$\text{MAE} = L(\theta, \{(x_i, y_i),\ i = 1, \ldots, n\}) = \frac{1}{n}\sum_{i=1}^{n} |y_i - f_\theta(x_i)|$$

$$\text{MAPE} = L(\theta, \{(x_i, y_i),\ i = 1, \ldots, n\}) = \frac{1}{n}\sum_{i=1}^{n} \frac{|y_i - f_\theta(x_i)|}{y_i}.$$

MAE är bättre än MAPE om data innehåller outputs med värdet 0 (och det är olämpligt att ta bort dessa outputs).

b.) Det primära syftet med bagging är att minska variansen i en modell, och beslutsträd är lämpliga då de typiskt har låg bias, hög varians, och är snabba att träna.

c.) Algoritmen använder en avståndsmetrik, och om inputs har olika skalor påverkar de avståndet olika mycket. Till exempel om $x = (x_1, x_2)$ och x_2 har mycket större skala så kommer punkter som ligger ”nära” (x_1, x_2) i princip bara bestämmas av x_2.

d.) Enkel logistisk regression ges av

$$\mathbb{P}(y = 1; x) = \sigma(\alpha + \beta x) = \frac{1}{1 + e^{-(\alpha+\beta x)}}.$$

Funktionen $\sigma(\alpha + \beta x)$ är monoton och kan således inte användas till att modellera ett icke-monotont samband.

e.) En policy π^* kallas optimal om

$$V^{\pi^*}(s) \geq V^{\pi}(s) \qquad \text{för alla } \pi,\ s.$$

f.) Låt $T^*(v)(s) = \max_{a \in A(s)}(r(s,a) + \sum_{s'} v(s')p(s'|s,a))$. Låt $v_0 \in \mathbb{R}^{|S|}$ vara en godtycklig vektor med $|S|$ komponenter, och definiera $v_{k+1} = T^*(v_k)$. Då gäller det att $v_k \to V^*$ då $k \to \infty$ (alltså $v_k(s) \to V^*(s)$ för alla s) och detta kallas *value iteration*. Givet V^* definieras en optimal policy genom

$$\pi^*(s) = \arg\max_{a \in A(s)} (r(s,a) + \sum_{s'} v(s')p(s'|s,a)).$$

Uppgift 3

a.) Då y verkar ha ett kvadratiskt beroende på x ges en lämplig transformation av $\phi(x) = (1, x, x^2)^t$. Alltså $f_\theta(x) = \alpha + \beta_1 x + \beta_2 x^2$.

b.) $\hat{\alpha} = 7.0889$, $\hat{\beta} = 1.0341$, MSE $= 34.16$.

```
tbl <- read_csv("data_uppgift3.csv")
m <- lm(y ~ x, data = tbl)
summary(m)
MSE <- mean(abs(m$fitted.values-tbl$y)^2)
```

Det är också okej om man delat in bifogad data i ett träning- och testset och beräknat MSE.

c.) Syftet med korsvalidering är främst för att flagga för overfitting, och även används även för att välja lämplig modellfamilj.

För att utföra korsvalidering för en modell f_θ. Låt data ges av $D = \{(x_i, y_i), i = 1, \ldots, n\}$. Dela upp data i k stycken folds $D_1, \ldots, D_k$.

1. För $i = 1, \ldots, k$: Träna modellen på $D \backslash D_i$ och beräkna ett prediktionsfel L_i på valideringsdata D_i
2. Vikta ihop felen $L_1, \ldots, L_k$, eller presentera felen per fold i en tabell.

d.) Två sätt att mitigera denna effekt är att använda Ridge- eller Lassoregreesion. Antag att modellens parametrar ges av $\theta = (\alpha, \beta_1, \ldots, \beta_k)$. Vanligtvis väljs θ genom

$$\theta^* = \arg\min_\theta \sum_{i=1}^{n} (y_i - f_\theta(x_i))^2.$$

Båda Ridge och Lassoregression straffar höga parametervärden genom

$$\text{Ridge}: \qquad \arg\min_\theta \sum_{i=1}^{n} (y_i - f_\theta(x_i))^2 + \lambda \sum_{i=1}^{k} \beta_i^2$$

$$\text{Lasso}: \qquad \arg\min_\theta \sum_{i=1}^{n} (y_i - f_\theta(x_i))^2 + \lambda \sum_{i=1}^{k} |\beta_i|.$$

Uppgift 4

a.) Då underliggande beroende mellan y på x är icke-linjärt och vi har mycket träningsdata.
b.) Då vi har lite träningsdata.

c.) Antal parametrar i modellen ges av

$$(12 \cdot 5 + 12) + (13 \cdot 12 + 13) + (14 \cdot 13 + 14) + (1 \cdot 14 + 1) = 452.$$

d.) Låt $h^{(i)} = x$. Låt vidare $\theta = \{(W^{(i)}, b^{(i)}), i = 1, 2, 3, 4\}$ vara modellens parametrar. Då gäller det att $h^i = \text{ReLu}(W^i h^{i-1} + b^i)$, $i = 1, 2, 3$, där ReLu-funktionen appliceras komponentvis på vektorn $W^{(i)}h^{(i-1)} + b^{(i)}$. Vidare gäller det att $f_\theta(x) = W^{(4)}h^{(3)} + b^{(4)}$

Uppgift 5

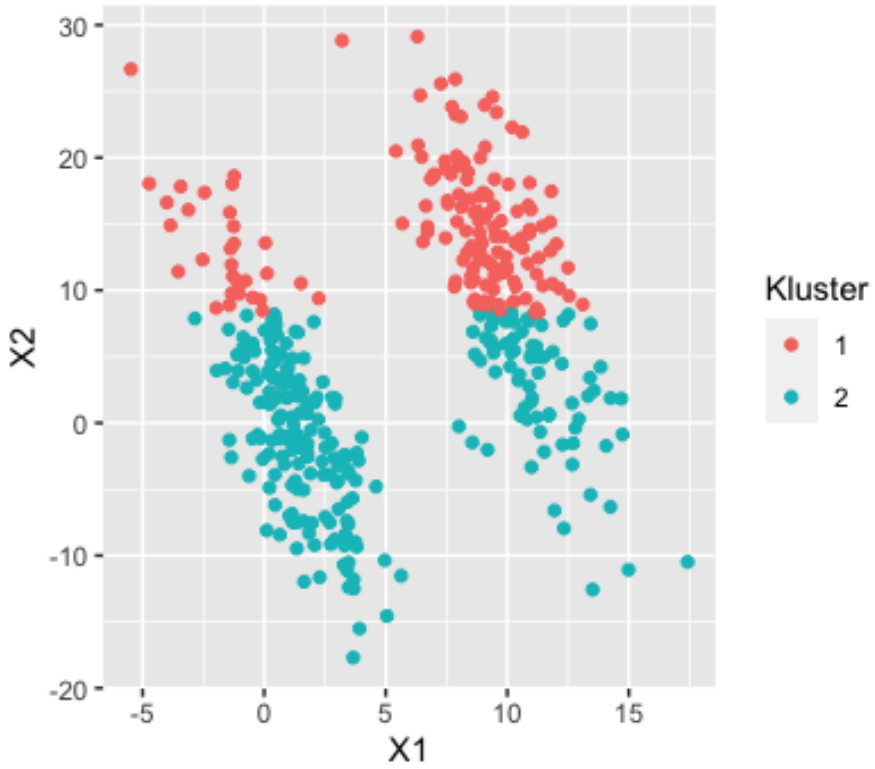
a) Se Bishop 9.1.

b)

```
## Implementation av k-means klustring (2 dim)
k_means <- function(X, centers) {

  # Initiala värden
  k <- length(centers)
  n <- dim(X)[1]
  d <- dim(X)[2]

  # Namnge kolumner
  names(X) <- paste0("X", 1:d)
  names(centers) <- paste0("X", 1:d)

  # Spara datapunkter och klustertillhörighet
  df <- X %>% mutate(c = NA)
  clus <- centers %>% mutate(name = 1:d)

  finish <- FALSE
  while(finish == FALSE) {

    # Tilldela varje data punkt ett kluster
    for(i in 1:n){
      dist <- sqrt((X$X1[1]-clus$X1)^2 + (X$X2[i]-clus$X2)^2)
      df$c[i] <- which.min(dist)
    }

    X1cen_old <- clus$X1
    X2cen_old <- clus$X2


    # Nya klustercenter
    for(i in 1:k) {
      clus[i,1] <- mean(subset(df$X1, df$c == i))
      clus[i,2] <- mean(subset(df$X2, df$c == i))
    }

    # Stanna loopen när inga data punkter ändras i klustret
    if(identical(X1cen_old, clus$X1) & identical(X2cen_old, clus$X2)){
      finish <- TRUE
    }
  }
  df$c
}
```

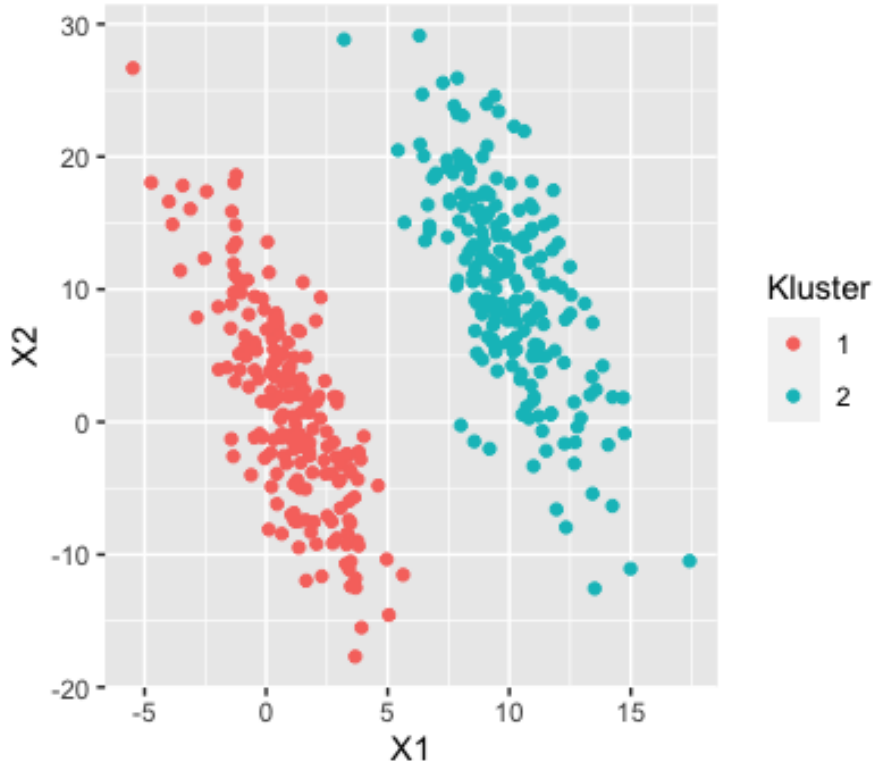
c)

```
# Initiala klustercenters
X1_cen <- runif(n = k, min = min(X[,1]), max = max(X[,1]))
X2_cen <- runif(n = k, min = min(X[,2]), max = max(X[,2]))
centers <-  bind_cols(X1_cen, X2_cen)

# Spara värden
knn_res <- k_means(X, centers)

# Plotta resultat
ggplot(X, aes(X1,X2, color= knn_res)) + geom_point()
```



d)

Klustringen misslyckas att identifiera de två grupperna av datapunkter, istället görs en indelning av övre och undre delen av datasetes. Eftersom data ser ut att vara genererat från två 2-dimensionella normalfördelningar skulle GMM vara en mer lämplig klustringsmetod. Det här är resultatet av GMM klustring:


30
20
10
0
-10
-20
X2
-5
0
5
10
15
X1
Kluster
1
2