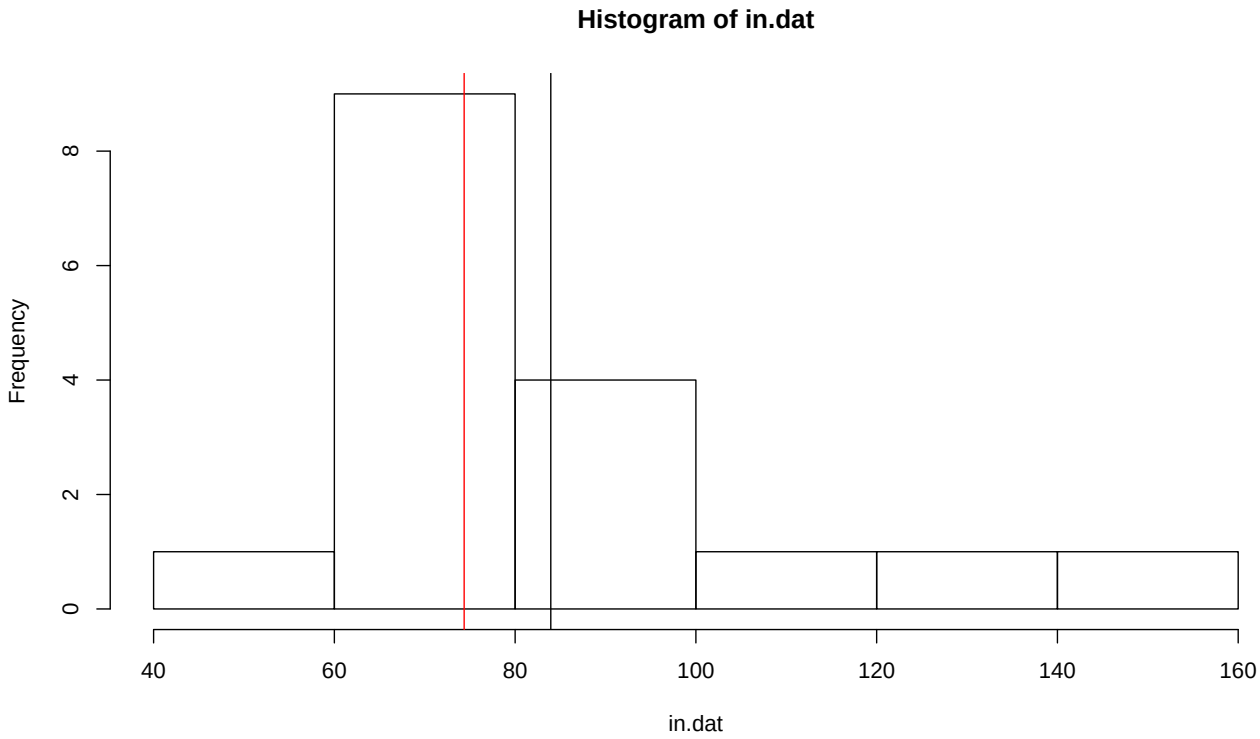
HW 4

Adon Rosen

Date: 2020-11-30

Problem 1A



These data appear to be skewed to the right which can be observed by the minor difference between the mean (black line) and the median (red line) of the data. So they do not appear to be normally distributed at first glance.

Problem 1B

```
parametric.method <- function(x, threshold=3){
  ## First calculate the mean of the data
  mean.val <- sum(x)/length(x)
  ## Now calc the standard deviation
  stand.val <- sqrt(sum((x - mean.val)^2)/length(x))
  ## Now calculate the z-scores
  z.scores <- (x - mean.val) / stand.val
  ## Now identify p-values
  p.values <- dnorm(z.scores)
  ## Now identify outliers
  index <- which(abs(z.scores)>threshold)
```

```
  ## Return these values
  return(list(z.scores <- z.scores, p.values <- p.values, index <- index))
}
```

Problem 1C

```
nonparametric.method <- function(x, p=.05){
  ## In a loop calculate the z scores;
  ## Then calculate the p values from a t distribution
  ## then flag and remove and observations below the alpha level
  ## repeat this until no more flags
  flag <- TRUE
  orig.vals <- x
  out.index <- NULL
  while(flag == TRUE){
    z.scores <- parametric.method(abs(x))[[1]]
    max.val <- z.scores[which.max(z.scores)]
    ## Now calculate the threshold
    n <- length(z.scores)
    threshold <- ((n-1)/sqrt(n))*sqrt(qt((1-p)/n,n-2)^2/(n-2+qt((1-p)/n,n-2)^2))
    ## check if the z score is larger than the threshold
    if(max.val > threshold){
      out.index <- c(out.index, which.max(z.scores))
      x <- x[-which.max(z.scores)]
    }else{
      flag <- FALSE
    }
  }
  return(list(out.vals = x, out.index = out.index))
}
```

Problem 1D

```
euclid <- function(points1, points2) {
  if(is.null(dim(points1))){
    points1 <- cbind(points1,0)
  }
  if(is.null(dim(points2))){
    points2 <- cbind(points2,0)
  }
  distanceMatrix <- matrix(NA, nrow=dim(points1)[1], ncol=dim(points2)[1])
  for(i in 1:nrow(points2)) {
    distanceMatrix[,i] <- sqrt(rowSums(t(t(points1)-points2[i,])^2))
  }
  distanceMatrix
}

neighbor.method <- function(x, k=3){
  ## This function will calculate the distance for each object from it's kth nearest neighbor'
  ## First identify distance from all other points
  dist.mat <- euclid(x, x)
  diag(dist.mat) <- NA
```

```
  ## Now go through each column and find the k'th value
  dist.vals <- apply(dist.mat, 2, function(x) x[order(x, decreasing = F)][k])
  ## Now prepare the output
  out.vals <- cbind(x, dist.vals)
  ## Now return these
  return(out.vals)
}
```

Problem 1E

Below are the results from the parametric method

```
# Run with w = 1
parametric.method(in.dat, threshold = 1)
```

```
[[1]]
 [1]  2.83054425  1.92114280  0.72791543  0.50946048  0.16895212  0.07089564
 [7] -0.01350741 -0.30684937 -0.39663104 -0.41690432 -0.43800508 -0.43841882
[13] -0.64901270 -0.77354856 -0.83850581 -0.86457145 -1.09295617

[[2]]
 [1] 0.007263243 0.063018095 0.306092184 0.350388225 0.393288853 0.397940959
 [7] 0.398905889 0.380596013 0.368764657 0.365736138 0.362452170 0.362386461
[13] 0.323179536 0.295783585 0.280695584 0.274533926 0.219541253

[[3]]
[1]  1  2 17
```

```
# Run with w = 2
parametric.method(in.dat, threshold = 2)
```

```
[[1]]
 [1]  2.83054425  1.92114280  0.72791543  0.50946048  0.16895212  0.07089564
 [7] -0.01350741 -0.30684937 -0.39663104 -0.41690432 -0.43800508 -0.43841882
[13] -0.64901270 -0.77354856 -0.83850581 -0.86457145 -1.09295617

[[2]]
 [1] 0.007263243 0.063018095 0.306092184 0.350388225 0.393288853 0.397940959
 [7] 0.398905889 0.380596013 0.368764657 0.365736138 0.362452170 0.362386461
[13] 0.323179536 0.295783585 0.280695584 0.274533926 0.219541253

[[3]]
[1] 1
```

```
# Run with w = 3
parametric.method(in.dat, threshold = 3)
```

```
[[1]]
 [1]  2.83054425  1.92114280  0.72791543  0.50946048  0.16895212  0.07089564
 [7] -0.01350741 -0.30684937 -0.39663104 -0.41690432 -0.43800508 -0.43841882
[13] -0.64901270 -0.77354856 -0.83850581 -0.86457145 -1.09295617

[[2]]
 [1] 0.007263243 0.063018095 0.306092184 0.350388225 0.393288853 0.397940959
 [7] 0.398905889 0.380596013 0.368764657 0.365736138 0.362452170 0.362386461
[13] 0.323179536 0.295783585 0.280695584 0.274533926 0.219541253
```

```
[[3]]
integer(0)
```

Below are the results from the nonparametric method

```
# Nonparametric with alpha = .05
nonparametric.method(in.dat, p=.05)
```

```
$out.vals
 [1] 76.53 74.36 73.87 73.36 73.35 68.26 65.25 63.68 63.05 57.53

$out.index
[1] 1 1 1 1 1 1 1
```

Below are the results from the nearest neighbor method

```
# neighbor method w/ k = 2
neighbor.method(in.dat, k = 2)
```

```
           x dist.vals
 [1,] 152.36     50.82
 [2,] 130.38     28.84
 [3,] 101.54     13.51
 [4,]  96.26      8.23
 [5,]  88.03      4.41
 [6,]  85.66      2.37
 [7,]  83.62      4.41
 [8,]  76.53      2.66
 [9,]  74.36      1.00
[10,]  73.87      0.51
[11,]  73.36      0.51
[12,]  73.35      0.52
[13,]  68.26      4.58
[14,]  65.25      2.20
[15,]  63.68      1.57
[16,]  63.05      2.20
[17,]  57.53      6.15
```

```
# neighbor method w/ k = 3
neighbor.method(in.dat, k = 3)
```

```
           x dist.vals
 [1,] 152.36     56.10
 [2,] 130.38     34.12
 [3,] 101.54     15.88
 [4,]  96.26     10.60
 [5,]  88.03      8.23
 [6,]  85.66      9.13
 [7,]  83.62      7.09
 [8,]  76.53      3.17
 [9,]  74.36      1.01
[10,]  73.87      0.52
[11,]  73.36      1.00
[12,]  73.35      1.01
[13,]  68.26      5.09
[14,]  65.25      3.01
```

```
[15,]  63.68      4.58
[16,]  63.05      5.21
[17,]  57.53      7.72
```

Here it appears the most sensitive metric is the nonparametric method which identifies 7 outliers, whereas the parametric method, with the most strict rules (w=1) only identified 3 outliers. The nonparamteric method agrees with my initial assumption of a largely skewed dataset, however the nearest neighbor method would most likely identify 3~4 outliers with distances > 8; this largely argues that these data need to be carefully used as there is a large amount of disagreement across these methods. I would argue that the most appropriate method to use with these data is the nonparametric method given the shape of the distribution of the data, the mean should not be used to identify outliers as is done with the parametric method, the nearest neighbor method does correctly identity the largest outlier, although there is some considerable variance across the distance as k changes from 2 and 3.

Problem 2

Problem 2A

1

$P(accident = yes|+) = P(accident = yes \cap congestion = yes)/P(congestion = yes) = 25/40 = 0.625$

2

$P(accident = yes|-) = P(accident = yes \cap congestion = no)/P(congestion = no) = 10/60 = 0.1666667$

3

$P(weather = good|+) = P(weather = good \cap congestion = yes)/P(congestion = yes) = 25/40 = 0.625$

4

$P(weather = good|-) = P(weather = good \cap congestion = no)/P(congestion = no) = 55/60 = 0.9166667$

5

$P(consturction = yes|+) = P(construction = yes \cap congestion = yes)/P(congestion = yes) = 15/40 = 0.375$

6

$P(construction = yes|-) = P(construction = yes \cap congestion = no)/P(congestion = no) = 20/60 = 0.3333333$

Problem 2B

Probability of positive

$X = (Accident = no, weather = bad, construction = yes)$

$P(X|+) = P(accident = no|congestion = yes) \times P(weather = bad|congestion = yes) \times P(construction = yes|congestion = yes)$

$P(accident = no|congestion = yes) = 1 - P(accident = yes|congestion = yes) = 0.375$

$P(weather = bad|congestion = yes) = 1 - P(weather = good|+)$=0.375

$P(construction = yes|+) = 0.375$

$P(X|+) = 0.0527344$

Probability of negative

$P(X|-) = P(accident = no|congestion = no) \times P(weather = bad|congestion = no) \times P(construction = yes|congestion = no)$

$P(accident = no|congestion = no) = 0.1666667$

$P(weather = bad|congestion = no) = 1 - P(weather = good|-) = 0.0833333$

$P(construction = yes|congestion = no) = 0.3333333$

$P(X|+) = 0.0046296$

Given these results the naive bayes classifier would predict a positive case

Problem 3

Clustering

Clustering algorithms can be used to place individuals into a group of individuals who have similar behavior patterns as the others. Once an individual is allocated to a specific cluster, this can be used to better understand their internet search behavior. For instance, searches can be clustered on language used in the searches to better understand geographic and language use patterns.

Classification

Classification can be used to distinguish individuals into typically one of two groups. This can be useful when trying to distinguish between demographic groups of the search provider. For instance, if the company was interested in distinguishing between language differences in young versus older individuals, they can use a classification algorithm to perform such a task.

Association rule mining

Association rule mining can be used to identify phrases and searches that are typically paired together. For instance if the company is interested in identifying searches that are associated with planning a party they can identify phrases that are commonly used with words like “party” and “plan”.

Anomaly detection

Anomaly detection can be used to identify instances that deviate for normative states, as well as identifying observations to deviate from normative rules. For instance, if the company is attempting to identify influential cases that may be contributing to skewed results, they can identify individuals who are making a large number of searches in a day. By exploring the distribution of the number of searches per day the company can remove individuals who make far more searches than would be feasible.