Hw 7

Jacob Thoma

4/17/2024

Recall that in class we showed that for randomized response differential privacy based on a fair coin (that is a coin that lands heads up with probability 0.5), the estimated proportion of incriminating observations $\hat{P}$[1] was given by $\hat{P} = 2\pi - \frac{1}{2}$ where π is the proportion of people answering affirmative to the incriminating question.

I want you to generalize this result for a potentially biased coin. That is, for a differentially private mechanism that uses a coin landing heads up with probability $0 \leq \theta \leq 1$, find an estimate $\hat{P}$ for the proportion of incriminating observations. This expression should be in terms of θ and π.

From class, the expected value of answering positively Y=1 (pi) in the fair coin case was pi = P(1/2) + 1/4 where $\hat{P}$ is the probability of a positive answer given a truthful (non-coerced) answer. As such, isolating $\hat{P}$ for this case yeilds $\hat{P}$/2 = pi - 1/4 which reduces to $\hat{P}$ = 2pi -1/2.

The general case for an unfair coin with heads probability θ is pi = $\hat{P}(\theta)$ + (1/2)(1-θ). Isolating $\hat{P}(\theta)$ yields $\hat{P}(\theta)$ = π - (1/2)(1-θ) which then reduces to $\hat{P}$ = (π - (1/2)(1-θ))/θ .

Next, show that this expression reduces to our result from class in the special case where $\theta = \frac{1}{2}$.

Plugging in $\theta = 1/2$ into $\hat{P}$ = (π - (1/2)(1-θ))/θ yeilds (π - (1/2)(1/2))/(1/2). Multiplying by 1/2 yeilds (1/2)$\hat{P}$ = π - (1/2)(1/2). Multiplying each side by 2 yields $\hat{P}$ = 2π - (1/2)

[1] in class this was the estimated proportion of students having actually cheated

Consider the additive feature attribution model: $g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x_i{}'$ where we are aiming to explain prediction f with model g around input x with simplified input x'. Moreover, M is the number of input features.

Give an expression for the explanation model g in the case where all attributes are meaningless, and interpret this expression. Secondly, give an expression for the relative contribution of feature i to the explanation model.

If all attributes are meaningless, the coefficient weights ϕ_i 's are all zero because they should have no effect on the prediction. In this case g(x') = ϕ_0 . This means the predictive model g for some x' is the exact same as the base prediction f(x) for some x.

The relative contribution for feature i is based on the relative weight of coefficient ϕ_i. This can be represented as ϕ_i / ($\sum_{i=1}^{M} \phi_i x_i{}'$) .

Part of having an explainable model is being able to implement the algorithm from scratch. Let's try and do this with `KNN`. Write a function entitled `chebychev` that takes in two vectors and outputs the Chebychev or L^∞ distance between said vectors. I will test your function on two vectors below. Then, write a `nearest_neighbors` function that finds the user specified k nearest neighbors according to a user specified distance function (in this case L^∞) to a user specified data point observation.

```r
#chebychev function
chebychev <- function(vector1,vector2){
  abs_coord_dists <- abs(vector1-vector2)
  max_abs_coord_dist <- max(abs_coord_dists)
  return(max_abs_coord_dist)}


#nearest_neighbors function
nearest_neighbors <- function(x, obs, k, class_labels, dist_func){
  dist = apply(x, 1, dist_func, obs)
  distances = sort(dist)[1:k]
  neighbor_list = which(dist %in% sort(dist)[1:k])
  return(list(neighbor_list, distances))
}


x<- c(3,4,5)
y<-c(7,10,1)
chebychev(x,y)

## [1] 6
```

Finally create a `knn_classifier` function that takes the nearest neighbors specified from the above functions and assigns a class label based on the mode class label within these nearest neighbors. I will then test your functions by finding the five nearest neighbors to the very last observation in the `iris` dataset according to the `chebychev` distance and classifying this function accordingly.

```
library(class)
df <- data(iris)

knn_classifier = function(x,y){

  groups = table(x[,y])
  pred = groups[groups == max(groups)]
  return(pred)
}

#data less last observation
x <- iris[1:(nrow(iris)-1),]
#observation to be classified
obs <- iris[nrow(iris),]

#find nearest neighbors
ind <- nearest_neighbors(x[,1:4], obs[,1:4], 5, x$Species,chebychev)[[1]]
as.matrix(x[ind,1:4])

##     Sepal.Length Sepal.Width Petal.Length Petal.Width
## 71           5.9         3.2          4.8         1.8
## 84           6.0         2.7          5.1         1.6
## 102          5.8         2.7          5.1         1.9
## 127          6.2         2.8          4.8         1.8
## 128          6.1         3.0          4.9         1.8
## 139          6.0         3.0          4.8         1.8
## 143          5.8         2.7          5.1         1.9

obs[,1:4]

##     Sepal.Length Sepal.Width Petal.Length Petal.Width
## 150          5.9           3          5.1         1.8

knn_classifier(x[ind, ], 'Species') # Call your knn_classifier function

## virginica
##         5

obs[,'Species']

## [1] virginica
## Levels: setosa versicolor virginica
```

Interpret this output. Did you get the correct classification? Also, if you specified $K = 5$, why do you have 7 observations included in the output dataframe?

Based on the nearest k=5 observations to point 150 by chebychev distance, the majority class for species is "virginica". This lines up with the actual species of point 150. The reason there are 7 observations despite k=5 is likely because of ties in the chebychev distances. This would result in 7 observations if there are ties in distance across the closest indices. In that case, all indices with a tied chebychev distance are returned until the total number of returned observations is 5 or greater.

Earlier in this unit we learned about Google's DeepMind assisting in the management of acute kidney injury. Assistance in the health care sector is always welcome, particularly if it benefits the well-being of the patient. Even so, algorithmic assistance necessitates the acquisition and retention of sensitive health care data. With this in mind, who should be privy to this sensitive information? In particular, is data transfer allowed if the company managing the software is subsumed? Should the data be made available to insurance companies who could use this to better calibrate their actuarial risk but also deny care? Stake a position and defend it using principles discussed from the class.

Data transfer of sensitive health care information should not be made available to insurance companies. The primary intentions of an insurance company are to make money. Exposing sensitive data to them allows for better trained models that help them minimize risk and maximize profit at the expense of the insured, opening opportunities for insurance companies to abuse their power. This abuse of power could happen by denying care/charging exorbitant prices for care based on sensitive health care information that they would not otherwise have. In this case, patients would be used for the insurance companies' gain instead of being treated as moral agents, directly violating the categorical imperative.

An alternative solution involves protecting health care data by keeping it within health care facilities. Even in a privatized health care market, the best hospitals could pay top dollar to create their own algorithms that can assist patients better than they would be assisted otherwise without leaking sensitive information to insurance companies.