Hw 7

Jiangyuan Yuan

11/28/2024

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 $\hat{P} = 2\hat{\pi} - \frac{1}{2}$ where $\hat{\pi}$ 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 $\hat{\pi}$.

Originally, with a fair coin (probability $\theta = 0.5$ of landing heads), the estimated proportion of incriminating observations was given by the formula in the question above. To generalize this for a biased coin, where the probability of answering truthfully is θ , we can make the following adjustment:

When using a biased coin, the expected proportion of "Yes" responses $\hat{\pi}$ can be expressed as the sum of the probability of answering truthfully times the true proportion P, plus the probability of answering randomly times the chance of saying "Yes" during a random response. This is $\hat{\pi} = \theta P + (1-\theta) \times \frac{1}{2}$. Solving for the true proportion P, we get the generalized estimate:

$$\hat{P} = \frac{\hat{\pi} - \frac{1}{2}(1-\theta)}{\theta}$$

.

This formula allows us to accurately estimate the true proportion of incriminating observations P based on the observed proportion $\hat{\pi}$ and the bias θ of the coin used in the randomized response mechanism.

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

To verify that the generalized estimate , substitute $\theta = \frac{1}{2}$ into the generalized formula:

$$\hat{P} = \frac{\hat{\pi} - \frac{1}{2}\left(1 - \frac{1}{2}\right)}{\frac{1}{2}}$$

Simplifying the numerator:

$$\hat{\pi} - \frac{1}{2}\left(\frac{1}{2}\right) = \hat{\pi} - \frac{1}{4}$$

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

Now, divide by 1/2:

$$\hat{P} = \frac{\hat{\pi} - \frac{1}{4}}{\frac{1}{2}} = 2\left(\hat{\pi} - \frac{1}{4}\right) = 2\hat{\pi} - \frac{1}{2}$$

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
#student input
#chebychev function
chebychev <- function(vec1, vec2) {
  max(abs(vec1 - vec2))
}
# nearest_neighbors function
nearest_neighbors <- function(data, query, k, distance_func) {
  distances <- apply(data, 1, function(row) distance_func(row, query))
    neighbor_indices <- order(distances)[1:k]
    data[neighbor_indices, ]
}


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.

```r
library(class)
df <- data(iris)
#student input
knn_classifier <- function(neighbors, class_col) {
  classes <- neighbors[[class_col]]
  mode_class <- names(sort(table(classes), decreasing = TRUE))[1]
  return(mode_class)
}

#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, chebychev)[[1]]
as.matrix(x[ind,1:4])
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width
## 6            5.4         3.9          1.7         0.4
## 5            5.0         3.6          1.4         0.2
## 6.1          5.4         3.9          1.7         0.4
## 5.1          5.0         3.6          1.4         0.2
## 6.2          5.4         3.9          1.7         0.4
```

```
obs[,1:4]
```

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

```
knn_classifier(x[ind,], 'Species')
```

```
## [1] "setosa"
```

```
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?

Student Answer

The output indicates that the KNN classifier function predicted the class label "setosa" for the last observation in the iris dataset, whereas the actual class label is "virginica", meaning the classification was incorrect. The discrepancy in the K value provided and shown is likely caused by duplicate data points or ties in the chebychev distance calculations, causing the nearest neighbors function to return more neighbors than intended.

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.

Student Answer

Sensitive healthcare data, such as that managed by Google's DeepMind for AKIs, should be strictly accessible only to authorized healthcare professionals directly involved in patient care, to promote patient autonomy and privacy. Data transfer to third parties, including insurance companies, should be prohibited unless explicit, informed consent is obtained from the patient, aligning with the harm principle by preventing potential misuse. Allowing insurance companies access solely for actuarial risk assessment poses significant risks of infringing on patients' rights and well-being, as it could result in denial of necessary medical treatments. Put simply, giving this information to an insurance company will not have any positive effects on a healthcare professionals ability to care for a patient. By prioritizing informed consent and limiting data access to essential medical personnel, we follow the guidelines set by the harm principle, and more generally, virtue ethics.

I have described our responsibility to proper interpretation as an *obligation* or *duty*. How might a Kantian Deontologist defend such a claim?

Student Answer

A Kantian Deontologist would defend the responsibility to proper interpretation as a moral obligation by invoking the categorical imperative. To recap, this mandates that one should act only according to maxims that can be universally applied without contradiction. Correctly interpreting data ensures honesty and integrity, treating all involved parties as ends in themselves rather than merely as means to an end. Misinterpretation or negligence in data analysis would violate the duty to uphold truthfulness and respect, essential for maintaining trust and ethical relationships. Thus, from a Kantian Deonotologist, the obligation to accurately and ethically interpret data is required, as it aligns with the fundamental principles of duty and respect.