HW7

Abigail Mabe

4/16/2024

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.

For a potentially biased coin, the generalized expression for the proportion of incriminating observations is as follows: $\hat{P} = \frac{\hat{\pi}-(1-\theta)\theta}{\theta}$. This stems from the proportion of people answering affirmative to the incriminating question ($\hat{\pi}$) being equivalent to $\hat{P}(\theta) + (1-\theta)(\theta)$. We can then isolate $\hat{P}$ to achieve $\hat{P} = \frac{\hat{\pi}-(1-\theta)\theta}{\theta}$, which is our generalized expression displayed above.

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

When $\theta = \frac{1}{2}$, and utilizing the expression above, we see $\hat{P} = \frac{\hat{\pi}-(1-\theta)\theta}{\theta}$ becomes $\hat{P} = \frac{\hat{\pi}-(\frac{1}{2})(\frac{1}{2})}{\frac{1}{2}}$ or $2\hat{P} = \hat{\pi} - (\frac{1}{4})$. Then, we can observe this is equivalent to the expression $\hat{P} = 2\hat{\pi} - (\frac{1}{2})$ as deduced in class.

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, ϕ_i will approach 0, giving an expression of g as follows: $g(x') = \phi_0$. This means no variables individually impact the prediction f from model g, and f depends arbitrarily on ϕ_0.

The relative contribution of feature i to the explanation model can be written as: $\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!(|F|-|S|-1)!}{|F|!}[f_{S\cup\{i\}}(x_{S\cup\{i\}}) - f_S(x_S)]$, where S is a subset of F, and F is our full set of predictors.

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

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.

```
#student input
#chebychev function
#nearest_neighbors function
chebychev <- function(vec1, vec2){
  vec1<-as.numeric(vec1)
  vec2<-as.numeric(vec2)
  vec3 <- numeric()
  for (i in 1:length(vec1)){
    vec3[i]<-abs(vec1[i]-vec2[i])
  }
  return(max(vec3))
}

nearest_neighbors <- function(x, obs, k, 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)
#student input
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, 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')
```

```
## 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?

As viewed above, we see *ind* indicates the indices of the "nearest neighbors" to the user-specified data point observation. Then, it prints out a matrix that displays those rows of data and corresponding columns for Sepal.Length, Sepal.Width, Petal.Length, and Petal.Width. The knn classifier determines what species the flower belongs to based on the information provided by the nearest neighbor information. Finally, the code prints the true species observation, which is virginica, indicating the classification was correct (since the knn classifier also predicted this species for the given nearest neighbors). Although I set $K = 5$, there are 7 observations included in the output data frame due to ties between nearest neighbors. This means the knn algorithm correctly classified the 5 nearest neighbors. However, some of these neighbors may have the exact same chebychev distance as another point in the data, so both are included in the output because knn does not inherently discriminate between two such neighbors of equal distance. Then, in this case, it can likely be deduced that two of the above neighbors have ties to other points, resulting in an output of 7 rather than the expected 5 nearest neighbors (although the knn classifier still correctly defines the 5 nearest neighbors).

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.

The transfer of data to a company that is subsumed by the original managers of sensitive information, including personal health data, is likely driven by the concept of paternalism. This is the concept that, in context, even if the original purpose/management of sensitive data changes, it is for the user's "own good", as in the creation of a new healthcare app that provides informed medical assistance. However, this claim of paternalism is not justified when considering the harm principle, which states autonomy should extend up until its use results in the objective harm of another agent. Then, the harm principle supports the claim that lucrative practices are not sufficient justification for data acquisition (even considering what is in the "best interest" for others) if the potential to cause harm is too great. In the context of Google's DeepMind app, the amount of sensitive data is very large, and therefore the potential for harm, and clear violation of the harm principle, is consequently large (if that information were to be acquired by another company without informed consent). This "harm" includes the potential for information to be leaked to other companies/managers such as insurance agencies, who may deny insurance to those who previously kept their information private. That is just one example of a very negative outcome that could arise from such lucrative data acquisitions, and is a great example why companies such as insurance agencies should not have full access to sensitive medical data without fully informed consent. In addition, in terms of moral frameworks, and the categorical imperative as defined by Kant, the use of sensitive data outside the scope of original intentions and without informed consent treats patients as means to an end rather than the end. Therefore, although the transfer of sensitive data comes from a paternalistic mindset, overall it can cause more detriment than good, violating both the harm principle and Kant's categorical imperative. Then, the owners of sensitive information must not share that information without informed consent, even in the instance where a company is subsumed by another.