Group Assignment 9

group 5

2022-11-19

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com (http://rmarkdown.rstudio.com).

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
library(neuralnet)
```

```
## Warning: 程辑包'neuralnet'是用R版本4.2.2 来建造的
```

```
library(nnet)
library(ggplot2)
```

```
## Warning: 程辑包'ggplot2'是用R版本4.2.2 来建造的
```

```
library(caret)
```

```
## Warning: 程辑包'caret'是用R版本4.2.2 来建造的
```

```
## 载入需要的程辑包：lattice
```

```
df=read.csv("C:/Users/wuzix/Desktop/ToyotaCorolla.csv")
attach(df)
df = na.omit(df)
dim(df)
```

```
## [1] 1436   39
```

```
var=c('Age_08_04', 'KM', 'Fuel_Type', 'HP', 'Automatic', 'Doors', 'Quarterly_Tax', 'Mfr_Guarant
ee', 'Guarantee_Period', 'Airco', 'Automatic_airco', 'CD_Player', 'Powered_Windows', 'Sport_Mod
el', 'Tow_Bar')
df=df[,c('Price',var)]
```

```
price=df[,'Price']
max_price=range(df['Price'])[2]
min_price=range(df['Price'])[1]
numerical=c('Price','Age_08_04','KM','HP','Quarterly_Tax','Guarantee_Period','Doors')
norm.values=preProcess(df[,numerical],method='range')
df[,numerical]=predict(norm.values,df[,numerical])
```

```
#convert categorial perdictor to dummies
#get class names
fuel_types=colnames(class.ind(df$Fuel_Type))
#add dummies to dataframe
df=cbind(df,class.ind(df$Fuel_Type))
#rename columns
names(df)=c('Price',var,paste('Fuel_Type_',fuel_types,sep=""))
#drop original columns
df=subset(df,select=-c(Fuel_Type))
```

```
set.seed(18)
train=sample(nrow(df),nrow(df)*0.7)
#fit a neural network using a single hidden layer with 2 nodes
f=as.formula(paste('Price~',paste(names(df)[!names(df) %in% c('Price')],collapse='+')))
nn=neuralnet(f,data=df[train,],hidden=2)
```

```
#get RMSE
rmsef = function(nn,df,train,price){
  pred.train = neuralnet::compute(nn,subset(df[train,],select = -c(Price)))
  pred.train.orig = pred.train$net.result*(max_price-min_price)+min_price
  train.rmse = sqrt(mean((price[train]-pred.train.orig)^2))
  pred.test = neuralnet::compute(nn,subset(df[-train,],select = -c(Price)))
  pred.test.orig = pred.test$net.result*(max_price-min_price)+min_price
  test.rmse = sqrt(mean((price[-train]-pred.test.orig)^2))
  #return rmse
  rmse = as.data.frame(rbind(train.rmse,test.rmse))
  return(rmse)
}
```

```
rmse = rmsef(nn,df,train,price)
rmse
```

```
##                  V1
## train.rmse 1046.665
## test.rmse  1081.820
```

```
nn1 = neuralnet(f,data = df[train,], hidden = 5)
rmse1 = rmsef(nn1,df,train,price)

nn2 = neuralnet(f,data = df[train,], hidden = c(5,5))
rmse2 = rmsef(nn2,df,train,price)

rmse_df = cbind(rmse,rmse1,rmse2)
names(rmse_df) = c('1 layer 2nodes','1 layer 5 nodes','2 layer 5nodes')
rmse_df
```

```
##            1 layer 2nodes 1 layer 5 nodes 2 layer 5nodes
## train.rmse       1046.665        942.5672       934.7064
## test.rmse        1081.820       1138.3768      1212.9268
```

1. We can find that the RMS error for the training data decreases as the number of layers and nodes increases.
2. But the test RMSE has the highest number at 1 layer 5 nodes model.
3. So we can assume that 2 layer 5 nodes model is the best.