Predicting Prices of Used Cars Linear Regression

Lucky

```
### Predicting Prices of Used Cars Linear Regression
## Evaluate performance before building the model
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
# load file
toyota.corolla.df <- read.csv("ToyotaCorolla.csv")

# randomly generate training and validation sets
training <- sample(toyota.corolla.df$Id, 600)
validation <- sample(setdiff(toyota.corolla.df$Id, training), 400)

# run linear regression model
reg <- lm(Price~., data=toyota.corolla.df[,-c(1,2,8)], subset=training,
          na.action=na.exclude)

pred_t <- predict(reg, na.action=na.pass)
pred_v <- predict(reg, newdata=toyota.corolla.df[validation,-c(1,2,8)],
                  na.action=na.pass)
```

```
## Warning in predict.lm(reg, newdata = toyota.corolla.df[validation, -c(1, :
## prediction from a rank-deficient fit may be misleading
```

```
# evaluate performance
# training
accuracy(pred_t, toyota.corolla.df[training,]$Price)
```

```
##                      ME     RMSE      MAE       MPE     MAPE
## Test set 8.367216e-12 1073.536 806.0069 -0.853454 8.059926
```

```
# validation
accuracy(pred_v, toyota.corolla.df[validation,]$Price)
```

```
##                 ME     RMSE      MAE         MPE     MAPE
## Test set 66.91117 1119.215 855.4607 -0.08857635 8.847048
```

```
# Based on the results, performance of training is better than validation as RMSE and MAE are lower

## Build model
# remove missing data in price variable
toyota.corolla.df <- toyota.corolla.df[!is.na(toyota.corolla.df[validation,]$Price),]

# generate random training and validation sets
training <- sample(toyota.corolla.df$Id, 600)
validation <- sample(toyota.corolla.df$Id, 400)

# regression model based on all numerical predictors
reg <- lm(Price~., data = toyota.corolla.df[,-c(1,2,8)], subset = training)

# predictions
pred_v <- predict(reg, newdata = toyota.corolla.df[validation,-c(1,2,8)])
```

```
## Warning in predict.lm(reg, newdata = toyota.corolla.df[validation, -c(1, :
## prediction from a rank-deficient fit may be misleading
```

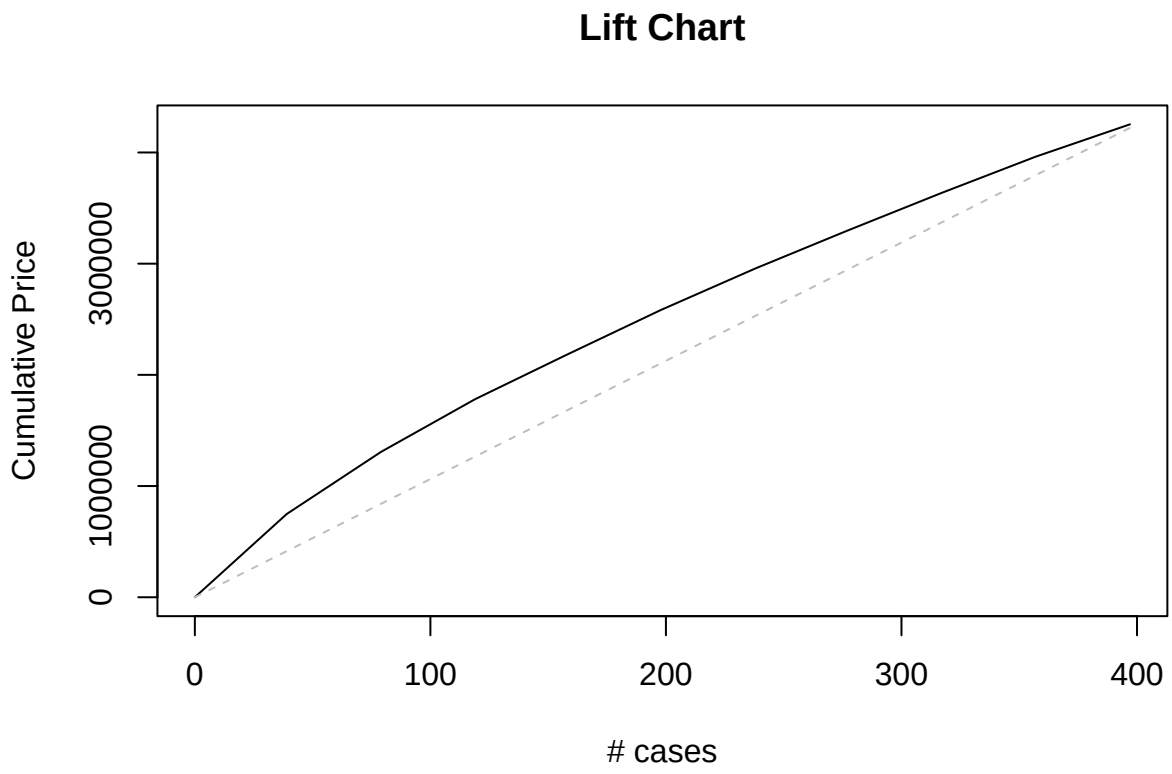
```
head(pred_v)
```

```
##      1355      1098       932      1337       157       850
##  9294.810  7402.638  9495.794  8669.015 19794.087  9948.916
```

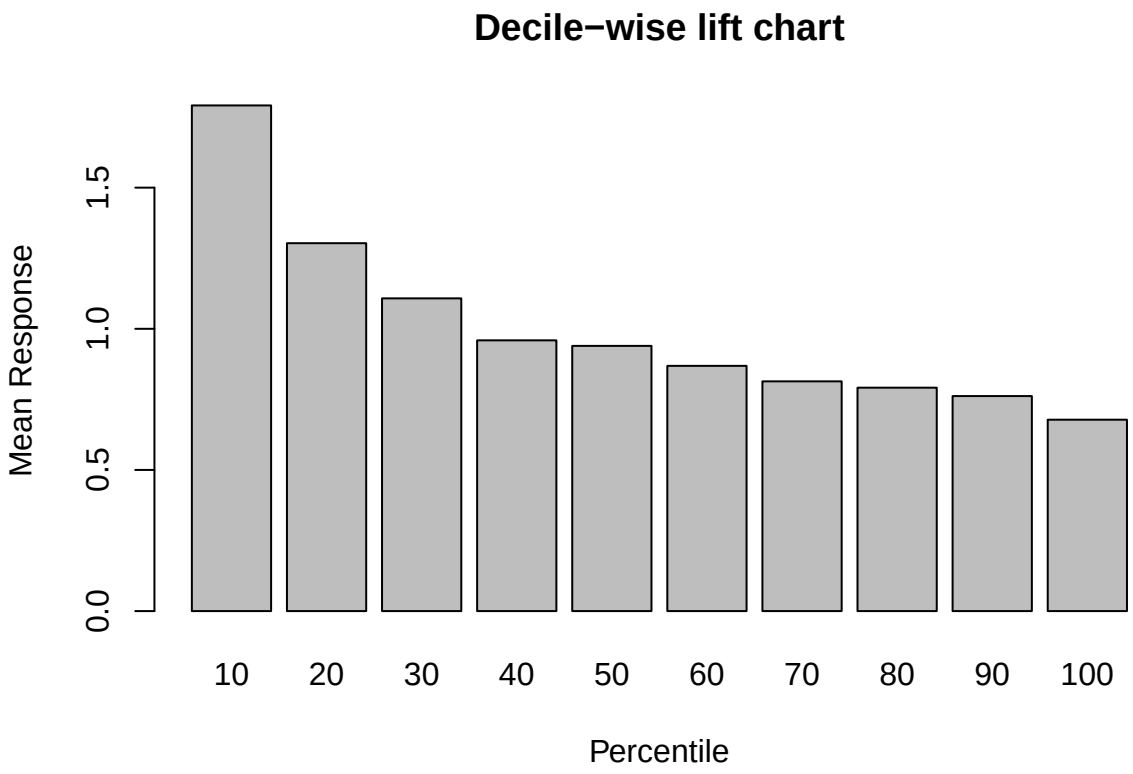
```
# load package gains
library(gains)
## Compute gains
gain <- gains(toyota.corolla.df[validation,]$Price[!is.na(pred_v)], pred_v[!is.na(pred_v)])

# cumulative lift chart
options(scipen=999) # avoid scientific notation
# compute the gain relative to price
price <- toyota.corolla.df[validation,]$Price[!is.na(toyota.corolla.df[validation,]$Price)]
plot(c(0,gain$cume.pct.of.total*sum(price))~c(0,gain$cume.obs),
     xlab="# cases", ylab="Cumulative Price", main="Lift Chart", type="l")

# baseline
lines(c(0,sum(price))~c(0,dim(toyota.corolla.df[validation,])[1]), col="gray", lty=2)
```




```
# decile-wise lift chart
barplot(gain$mean.resp/mean(price), names.arg = gain$depth,
        xlab = "Percentile", ylab = "Mean Response", main = "Decile-wise lift chart")
```


Decile−wise lift chart
Mean Response
0.0
0.5
1.0
1.5
10
20
30
40
50
60
70
80
90
100
Percentile