Trabalho 1

Aprendizado de Máquina - SME0829

Beatriz Carvalho - 10408302 Brunna Quatrochi - 10872833
Gustavo Terra Brandão - 11274935 Lua Nardi Quito - 11371270

26/05/2022

Sumário

Introdução

Nesse trabalho iremos Utilizar diferentes métodos paramétricos para regressão linear e comparar a eficiência de cada um deles para concluir qual a melhor técnica a ser usada para predição da variável de interesse.

Descrição do banco

O banco de dados que será utilizado é o "Auto", que pertence ao pacote de Introdução ao Aprendizado Estatístico no R com Aplicações (ISLR), do próprio R.

O banco possui 392 observações e 9 variáveis, sendo elas:

Nome da variável	**Descrição**
mpg	Milhas por galão
cylinders	Número de cilindros (entre 4 e 8)
displacement	Cilindradas (em polegadas cúbicas)
horsepower	Cavalos-força
weight	Peso do veículo (em libras)
acceleration	Tempo de aceleração de 0 a 60 milhas por hora (em segundos)
year	Ano do Modelo (mod 100)
origin	País de origem do veículo (1. Americano, 2. Europeu e 3. Japonês)
name	Nome do veículo

Iremos tratar "acceleration", a aceleração do veiculo como nossa variavel de interesse, enquanto o resto serão nossas covariaveis. Pode-se observar que temos tanto variáveis categóricas quanto variáveis numéricas discretas e contínuas.

Análise dos dados

Importando o banco

Vamos utilizar o banco de dados "Auto" do pacote do R "ISLR"

```
library(ISLR)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-4
```

```
library(plm)
library(Metrics)
library(ggplot2)
library(knitr)
a <- ISLR::Auto
```

Ajustes iniciais

Para a nossa análise, as variáveis como nome do modelo, país de origem e ano de lançamento são desnecessárias, portanto, vamos retira-las do nosso banco.

```
df = subset(a, select = -c(name,origin, year))
```

Análise Descritiva

Medidas principais dos dados:

```
summary(df)
```

```
##       mpg          cylinders      displacement     horsepower        weight
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.   :1613
##  1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2225
##  Median :22.75   Median :4.000   Median :151.0   Median : 93.5   Median :2804
##  Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5   Mean   :2978
##  3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0   3rd Qu.:3615
##  Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0   Max.   :5140
##   acceleration
##  Min.   : 8.00
##  1st Qu.:13.78
##  Median :15.50
##  Mean   :15.54
##  3rd Qu.:17.02
##  Max.   :24.80
```

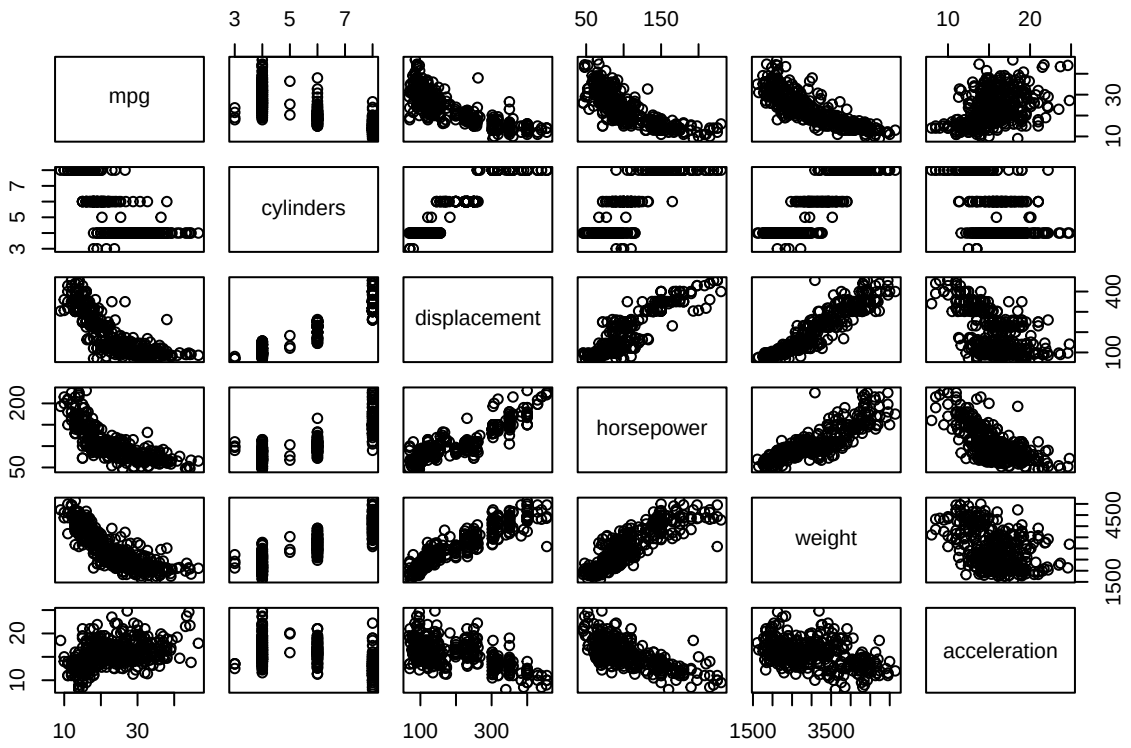
```
unique(df$cylinders)
```

```
## [1] 8 4 6 3 5
```

obs: a variável "cilindros" é discreta (só assume os valores (8,4,6,3,5)) enquanto o resto é contínua.

Pairplot para ajudar na vizualizacao dos dados.

```
pairs(df)
```



Aplicação dos modelos

Para ajustar os modelos, iremos utilizar os métodos de regressão linear de mínimos quadrados, regressão lasso e regressão ridge.

Os métodos de Ridge e Lasso aplicam penalizações nos seus coeficientes de regressão, para diminuir a complexidade do modelo. Enquanto o método de MQ faz só a regressão linear normalmente.

A penalização do Ridge utiliza o valor dos coeficientes ao quadrado, enquanto o Lasso utiliza seu módulo.

Agora separamos os nossos dados nos conjuntos de treinamento e de validação. Decidimos utilizar 30% dos dados para teste e 70% dos dados para o conjunto treinamento.

```
set.seed(100) #setar a semente pra que os resultados deem sempre iguais
split1<- sample(c(rep(0, 0.7 * nrow(df)), rep(1, 0.3 * nrow(df))))

train <- df[split1 == 0, ]
test <- df[split1== 1, ]
```

Para nossa análise, a variável de interesse é a aceleração - Queremos prever a aceleração através dos valores das covariáveis: weight, horsepower, displacemente, cylinders e mpg.

No nosso banco a aceleração esta na sexta coluna, e as covariáveis ocupam da primeira a quinta coluna.

```
trainx <- subset(train, select = -c(acceleration)) #covariaveis treinamento
trainx <- data.matrix(trainx)
trainy <- train[["acceleration"]]     #variavel de interesse treinamento
trainy <- data.matrix(trainy)
testx <- subset(test, select = -c(acceleration)) #covariaveis teste
testx <- data.matrix(testx)
testy <- test[["acceleration"]]   #variavel de interesse teste
testy <- data.matrix(testy)

#estimador de mínimos quadrados (MQ)
ajuste1 <- glmnet(trainx, trainy, alpha = 0,lambda = 0)

#Regressão de Lasso
ajuste2 <- cv.glmnet(trainx, trainy, alpha = 1)
best_lambda <- ajuste2$lambda.min

#Regressão de Ridge
ajuste3 <- cv.glmnet(trainx, trainy, alpha = 0)
bestlambda <- ajuste3$lambda.min
```

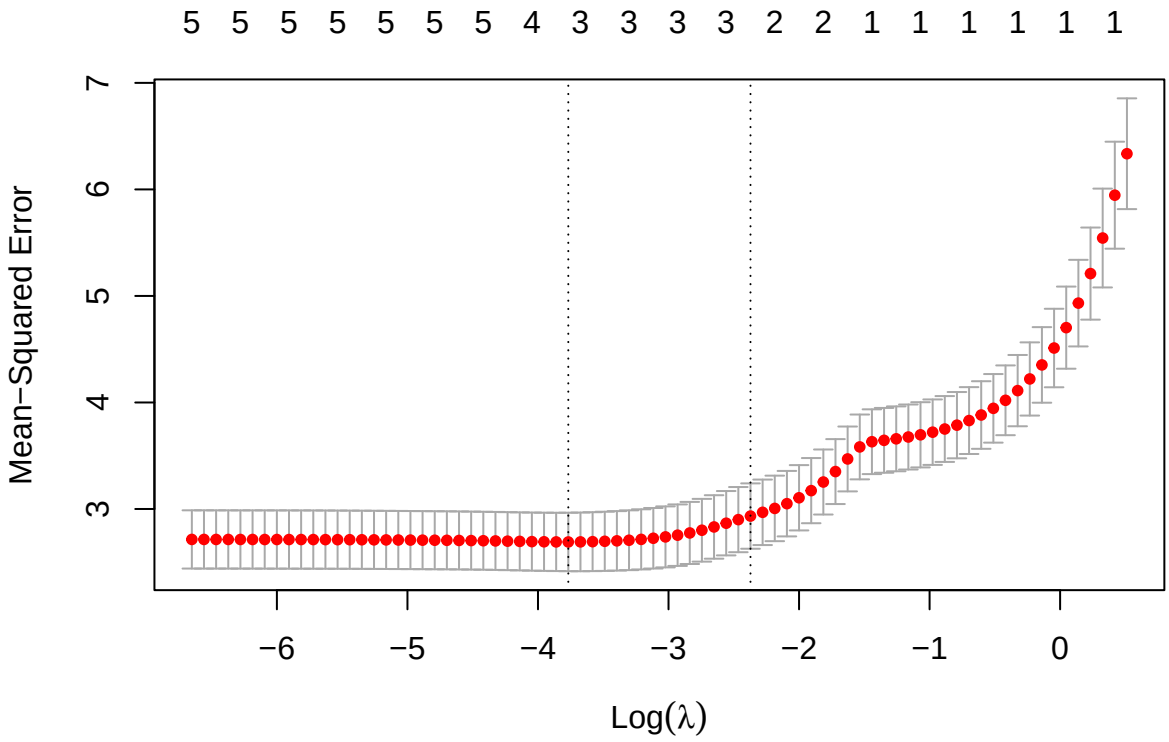
```
#tabela com os valores ideais de \lambda
table <- data.frame(
    Modelo = c("Lasso", "Ridge"),
    Lambda = c(best_lambda, bestlambda))
kable(table)
```

Modelo	Lambda
Lasso	0.0231082
Ridge	0.1668596

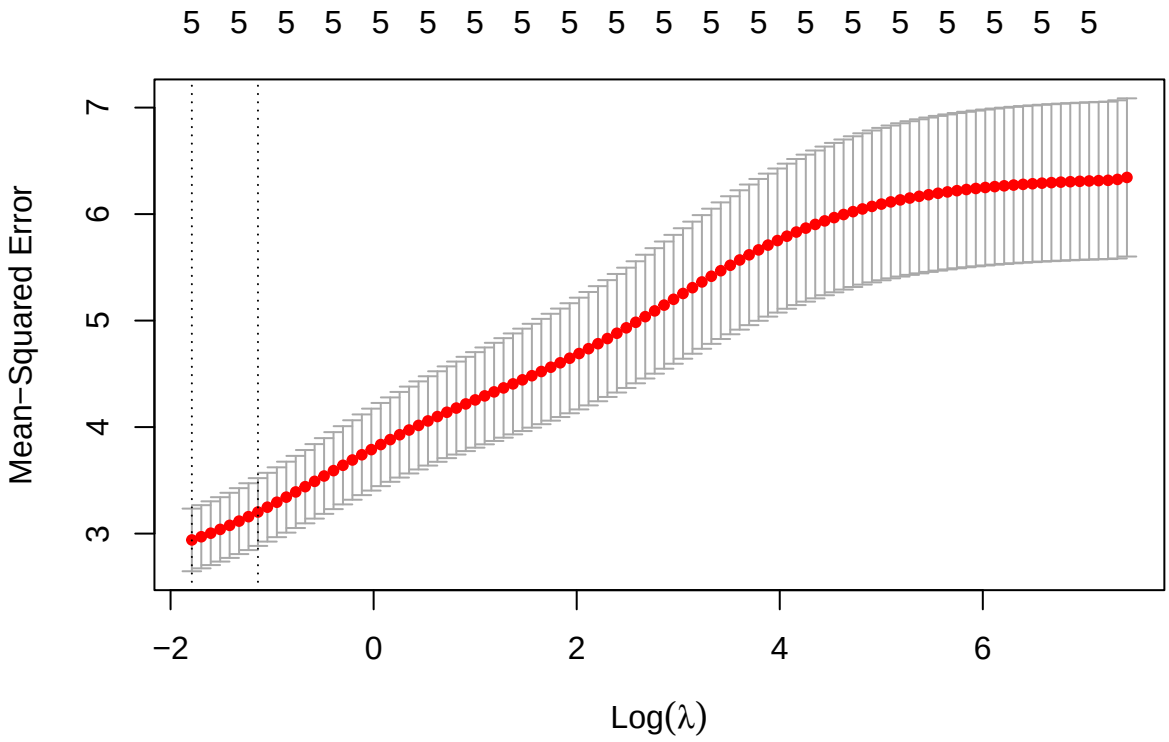
Esses são os valores de lambda que minimizam o erro quadrático para o lasso e ridge respectivamente

Plots do erro quadratico em função do “tuning parameter”, lambda.

```
#lasso
plot(ajuste2)
```




```
#ridge
plot(ajuste3)
```

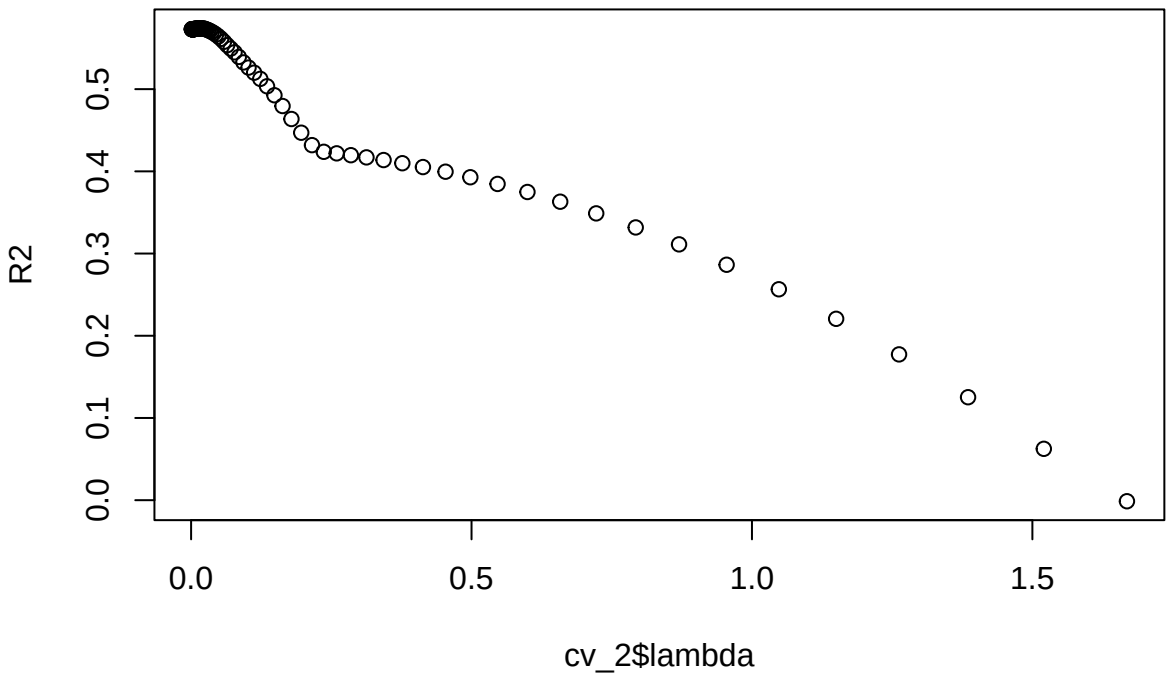


Plots da função R^2 em função do “tuning parameter”, lambda.

```
#no Lasso
cv_2<-cv.glmnet(trainx,trainy, alpha=1)
R2 = 1 - cv_2$cvm/var(trainy)
```

```
## Warning in cv_2$cvm/var(trainy): Recycling array of length 1 in vector-array arithmetic is deprecated
##   Use c() or as.vector() instead.
```

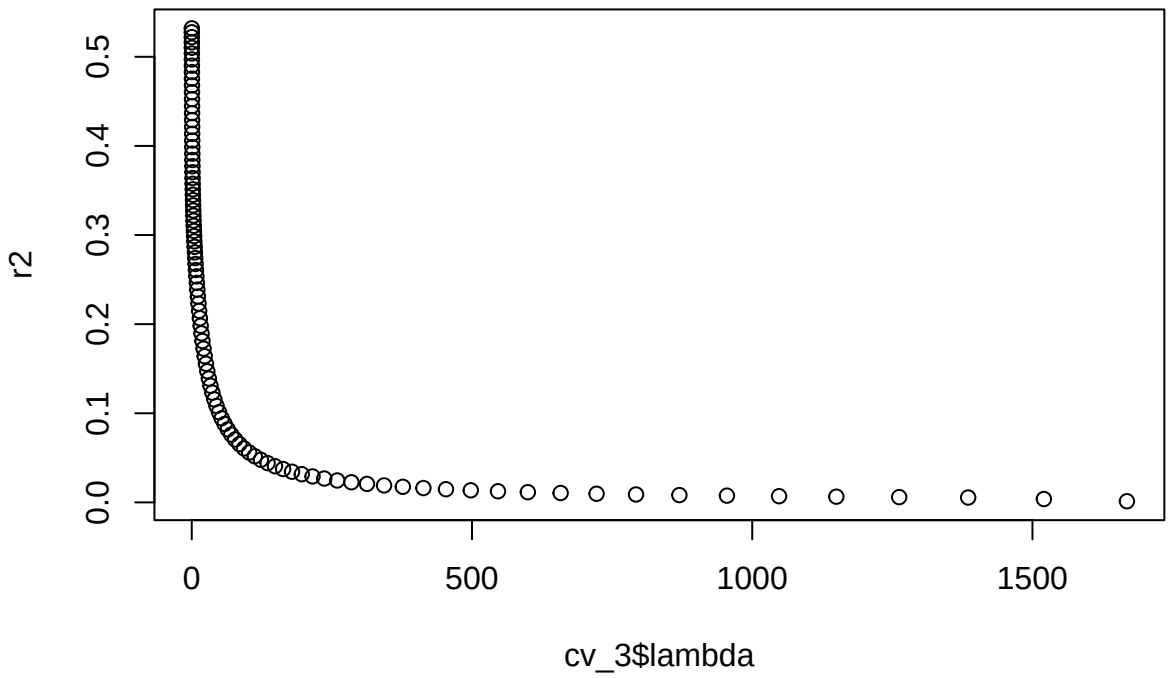
```
plot(cv_2$lambda,R2)
```




```
#no Ridge
cv_3<-cv.glmnet(trainx,trainy, alpha=0)
r2 = 1 - cv_3$cvm/var(trainy)
```

```
## Warning in cv_3$cvm/var(trainy): Recycling array of length 1 in vector-array arithmetic is deprecate
##   Use c() or as.vector() instead.
```

```
plot(cv_3$lambda,r2)
```



Podemos observar que quando os valores de lambda são ideias, nosso modelo consegue explicar até 50% da variação da variável de interesse (o que é bastante), o que significa que os modelos lineares são, de fato, adequados para esses dados.

obs: no método de mínimos quadrados, o valor de lambda é sempre zero.

```
#atualizando nosso ajustes para usar o melhor valor de lambda
ajuste2 <- glmnet(trainx, trainy, alpha = 1, lambda = best_lambda )
ajuste3 <- glmnet(trainx, trainy, alpha = 0, lambda = bestlambda)
```

```
#MQ
coef(ajuste1)
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                        s0
## (Intercept)  15.842748685
## mpg           0.014010684
## cylinders     0.155694570
## displacement -0.016241555
## horsepower   -0.084335736
## weight        0.003484488
```

```
#lasso
coef(ajuste2)
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
```

```
##                          s0
## (Intercept)  17.3604323069
## mpg           0.0007553691
## cylinders     .
## displacement -0.0106619388
## horsepower   -0.0825902010
## weight        0.0029455968
```

```
#ridge
coef(ajuste3)
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
##                          s0
## (Intercept)  18.010742241
## mpg           0.006526461
## cylinders     0.006296704
## displacement -0.006982080
## horsepower   -0.060806460
## weight        0.001679816
```

```
tabel <- data.frame(
    Beta = c("Intercepto", "mpg", "cylinders", "displacement", "horsepower",
             "weight"),
    MQ = c(15.842748685, 0.014010684, 0.155694570,-0.016241555, -0.084335736,
           0.003484488),
    Lasso = c(17.3604323069, 0.0007553691, 0, -0.0106619388,-0.0825902010,
              0.0029455968),
    Ridge = c(18.01074224, 0.00652646, 0.00629670, -0.00698208, -0.06080646,
              0.001679816))
kable(tabel)
```

Beta	MQ	Lasso	Ridge
Intercepto	15.8427487	17.3604323	18.0107422
mpg	0.0140107	0.0007554	0.0065265
cylinders	0.1556946	0.0000000	0.0062967
displacement	-0.0162416	-0.0106619	-0.0069821
horsepower	-0.0843357	-0.0825902	-0.0608065
weight	0.0034845	0.0029456	0.0016798

Fazendo a predição com os modelos ajustados:

```
#MQ
predito1<-predict(ajuste1,newx=testx)

#lasso
predito2<- predict(ajuste2,s=cv_2$lambda.1se,newx = testx)

#ridge
predito3<- predict(ajuste3,s=cv_3$lambda.1se,newx = testx)
```

Analise das predições:

```r
mse(testy, predito1) #erro quadratico medio MQ
```

```
## [1] 4.148282
```

```r
mse(testy, predito2) #erro quadratico medio LASSO
```

```
## [1] 3.896637
```

```r
mse(testy, predito3) #erro quadratico medio RIDGE
```

```
## [1] 4.101802
```

```r
tabela <- data.frame(
    Modelo = c("MQ", "Lasso", "Ridge"),
    EQM = c(4.148282, 3.896637, 4.101802))
kable(tabela)
```

Modelo	EQM
MQ	4.148282
Lasso	3.896637
Ridge	4.101802

Conclusão

Após a análise, concluímos que os modelos lineares são de fato adequados para a análise do conjunto de dados escolhido, já que os valores dos EQM deram relativamente baixos e nossa regressão consegue explicar por volta de 50% de toda a variação na variável "acceleration".

Pudemos observar que o método Lasso apresentou um menor erro quadrático, porém a diferença entre os 3 métodos foi relativamente bem pequena.

Também pudemos observar que o método Lasso, por ser mais punitivo com os seus coeficientes, zerou o coeficiente da variável cylinders, por não achar que essa variável tinha um impacto relevante em prever o valor da aceleração.

Bibliografia

- Conjunto de dados disponível na biblioteca ISLR.
- James, G., et al., An Introduction to Statistical Learning with Applications in R.