checkpoint3

Leonardo Alves dos Santos

04-04-2016

```
#Libraries
library(plyr)
library(dplyr)
library(ggplot2)
library(caret)
```

Introdução

```
#http://statweb.stanford.edu/~tibs/ElemStatLearn/datasets/prostate.data
prostate.data <- read.delim("./data/prostate.data.txt")
```

Os dados utilizados representam resultados de exames realizados em pacientes do sexo Masculino, com o objetivo de diagnosticar pacientes com Câncer de Prostata. O dataset possui os seguintes dados.

- vol: volume do câncer
- weight: peso do paciente
- age: idade do paciente
- bph: hiperplasia prostática benigna
- svi: invasão das vesículas seminais
- cp: penetração capsular
- gleason: escore Gleason
- pgg45: percentagem escore Gleason 4 ou 5
- psa: antígeno específico da próstata.

O objetivo é criar um modelo melhorado do encontrado no Checkpoint 2.

O Modelo Inicial

O modelo utilizado era composto por: lcavol, lweight, age, lbph, svi, lcp, gleason, pgg45. Este modelo possuia um RMSE de 0.7219931. A baixo está o detalhamento do modelo que encontramos.

```
summary(reg_multipla)
```

```
##
## Call:
## lm(formula = lpsa ~ ., data = dados_treino)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.64870 -0.34147 -0.05424  0.44941  1.48675
##
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.429170   1.553588   0.276  0.78334
## lcavol       0.576543   0.107438   5.366 1.47e-06 ***
## lweight      0.614020   0.223216   2.751  0.00792 **
## age         -0.019001   0.013612  -1.396  0.16806
## lbph         0.144848   0.070457   2.056  0.04431 *
## svi          0.737209   0.298555   2.469  0.01651 *
## lcp         -0.206324   0.110516  -1.867  0.06697 .
## gleason     -0.029503   0.201136  -0.147  0.88389
## pgg45        0.009465   0.005447   1.738  0.08755 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7123 on 58 degrees of freedom
## Multiple R-squared:  0.6944, Adjusted R-squared:  0.6522
## F-statistic: 16.47 on 8 and 58 DF,  p-value: 2.042e-12
```

Modelo Usando LASSO

O LASSO é um método de Encolhimento, ou de Shrinkage, utilizado para reduzir o erro do modelo e evitar o overfitting dos dados.Segundo o blog Mineração de Dados para fazer isto, ele utiliza de "mecanismo de penalização dos coeficientes com um alto grau de correlação entre si, mas que usa o mecanismo de penalizar os coeficientes de acordo com o seu valor absoluto (soma dos valores dos estimadores) usando o mecanismo de minimizar o erro quadrático."

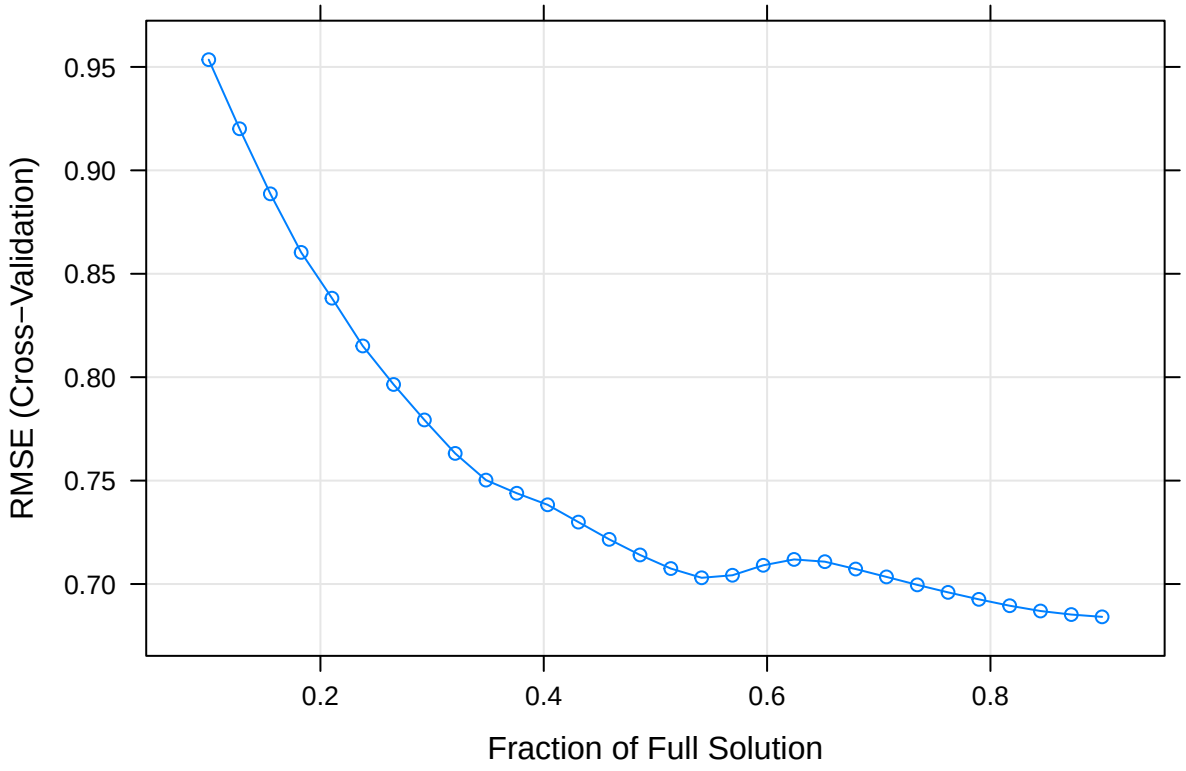
Para calcular o LASSO, iremos utilizar funções do pacote Caret. Primeiro iremos criar o nosso modelo usando a função train.

```
fitControl <- trainControl(method='cv', number = 30)

lasso.fit <- train(lpsa ~ ., dados_treino ,
                   method='lasso',
                   metric="RMSE",
                   tuneLength = 30,
                   trControl=fitControl)
```

A baixo é possivel observar os valores encontrados do RMSE, durante a criação do modelo. O modelo gerado é aquele que possuir o menor RMSE.

```
plot(lasso.fit)
```



Abaixo podemos ver o detalhamento do LASSO.

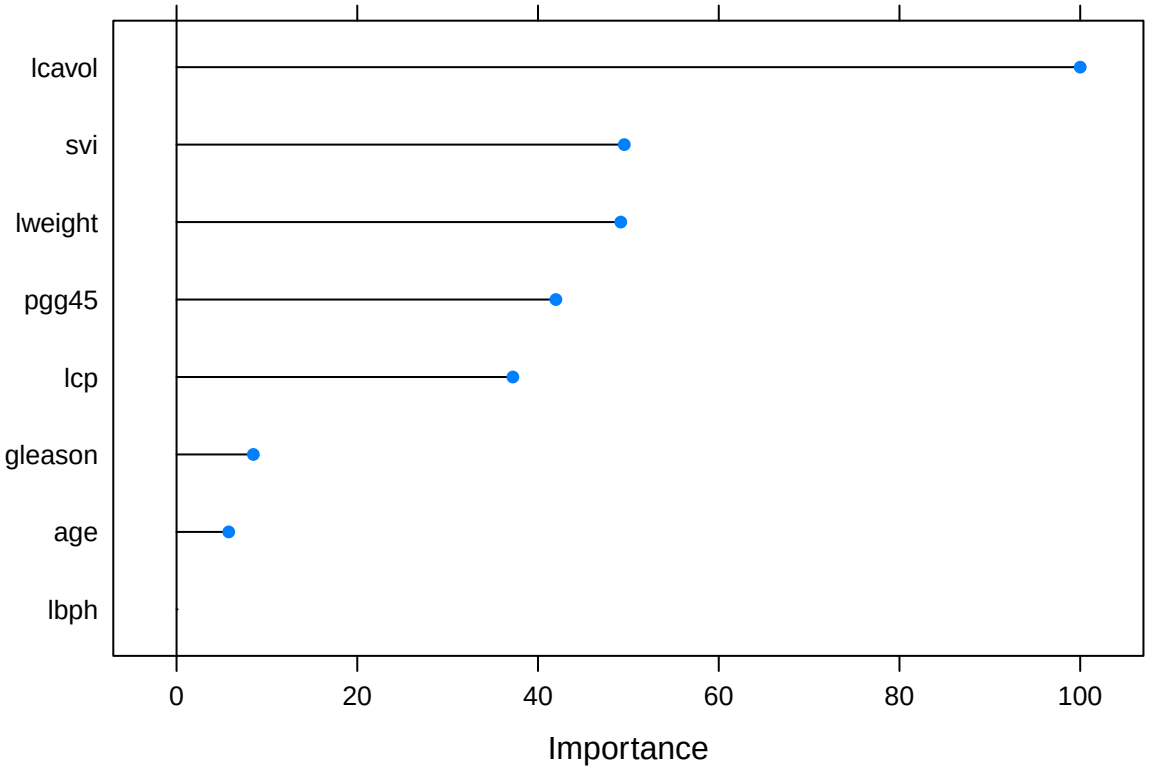
```
lasso.fit
```

```
## The lasso
##
## 67 samples
##  8 predictors
##
## No pre-processing
## Resampling: Cross-Validated (30 fold)
## Summary of sample sizes: 65, 65, 65, 65, 65, 65, ...
## Resampling results across tuning parameters:
##
##   fraction   RMSE       Rsquared   RMSE SD    Rsquared SD
##   0.1000000  0.9535025  0.9298169  0.4920781  0.19405730
##   0.1275862  0.9201083  0.9298169  0.4767597  0.19405730
##   0.1551724  0.8886475  0.9298169  0.4635171  0.19405730
##   0.1827586  0.8603307  0.9325642  0.4533773  0.18098485
##   0.2103448  0.8382021  0.9377525  0.4450720  0.15618185
##   0.2379310  0.8151005  0.9421778  0.4362307  0.13715432
##   0.2655172  0.7964818  0.9453788  0.4266859  0.12531986
##   0.2931034  0.7793281  0.9476265  0.4163917  0.11733909
##   0.3206897  0.7631488  0.9501768  0.4064741  0.10989888
##   0.3482759  0.7502457  0.9523371  0.3972161  0.10388812
##   0.3758621  0.7438955  0.9514008  0.3871598  0.10353724
##   0.4034483  0.7383024  0.9518896  0.3778244  0.10171494
##   0.4310345  0.7299405  0.9528873  0.3692090  0.09885216
##   0.4586207  0.7215943  0.9536808  0.3610939  0.09711005
```

```
##   0.4862069  0.7140625  0.9542594  0.3541646  0.09636462
##   0.5137931  0.7074863  0.9546637  0.3485793  0.09636905
##   0.5413793  0.7030279  0.9547565  0.3469717  0.09725346
##   0.5689655  0.7042414  0.9540222  0.3485142  0.09974738
##   0.5965517  0.7090593  0.9521288  0.3536171  0.10520307
##   0.6241379  0.7118937  0.9504357  0.3586643  0.11040137
##   0.6517241  0.7108066  0.9492224  0.3633451  0.11483422
##   0.6793103  0.7072527  0.9480522  0.3644694  0.11938670
##   0.7068966  0.7034481  0.9477011  0.3659513  0.12090276
##   0.7344828  0.6995897  0.9478406  0.3677249  0.12048248
##   0.7620690  0.6959702  0.9479393  0.3697068  0.12017331
##   0.7896552  0.6925978  0.9479977  0.3718852  0.11996630
##   0.8172414  0.6895101  0.9480164  0.3742280  0.11985307
##   0.8448276  0.6869619  0.9479718  0.3766117  0.11987488
##   0.8724138  0.6852489  0.9476834  0.3792776  0.12042771
##   0.9000000  0.6841345  0.9472574  0.3819344  0.12126968
##
## RMSE was used to select the optimal model using  the smallest value.
## The final value used for the model was fraction = 0.9.
```

É possivel ver também o nivel de importância de cada uma das variaveis utilizadas.

```
plot(varImp(lasso.fit))
```

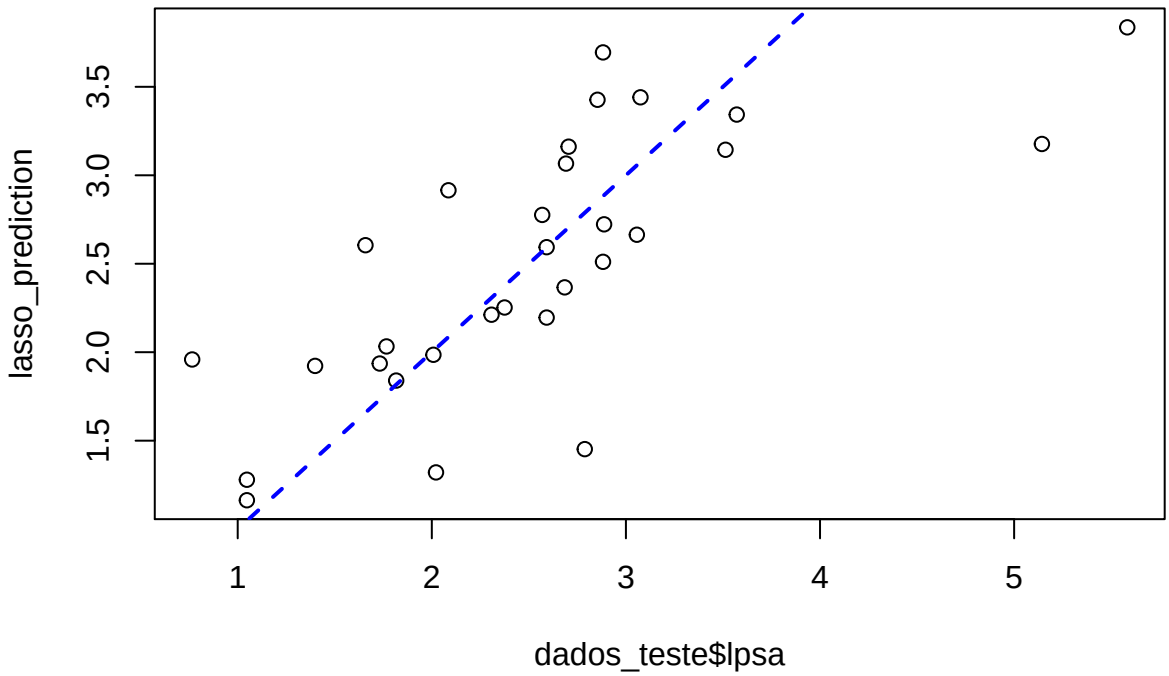


Após calcular o modelo, iremos fazer a predição dos dados

```
lasso_prediction <- predict(lasso.fit, dados_teste)
lasso_residuos = dados_teste$lpsa - predicoes
```

Abaixo podemos ver os dados reais versos os preditos

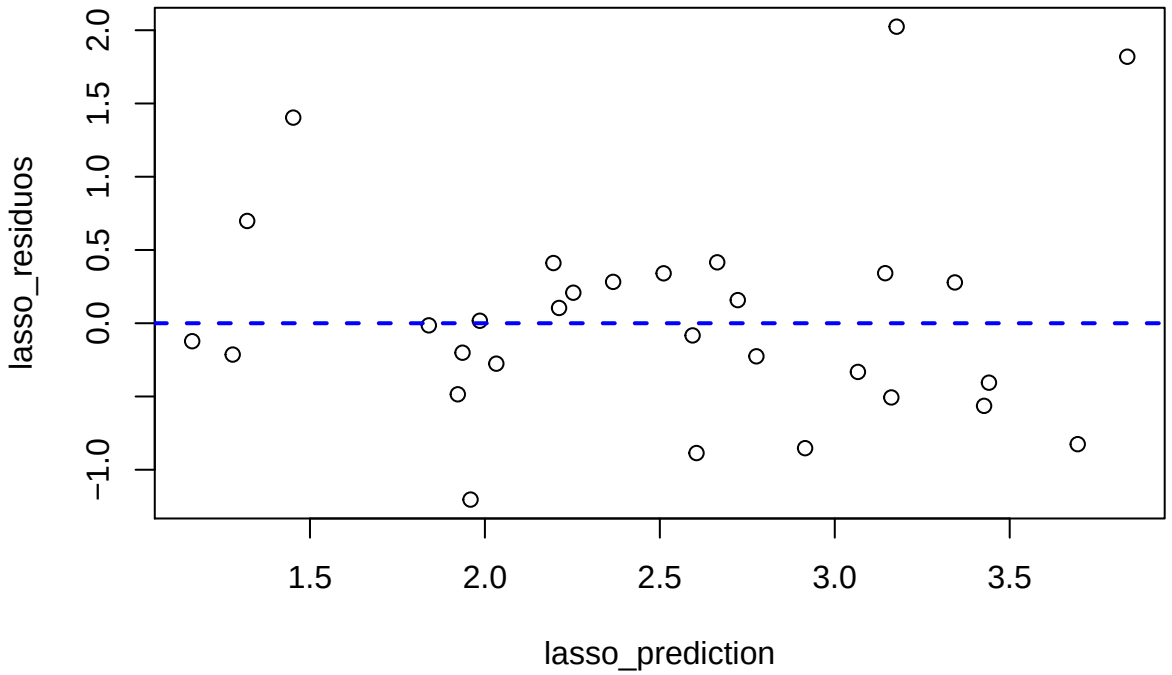
```
axisRange = extendrange(c(dados_teste$lpsa,lasso_prediction))
plot(dados_teste$lpsa,lasso_prediction)
abline(0,1,col="blue",lty=2,lwd=2)
```



Com excessão de alguns outliers, os valores preditos, se aproximam dos valores reais, e são aparentemente próximos dos encontrados no modelo do Checkpoint 2.

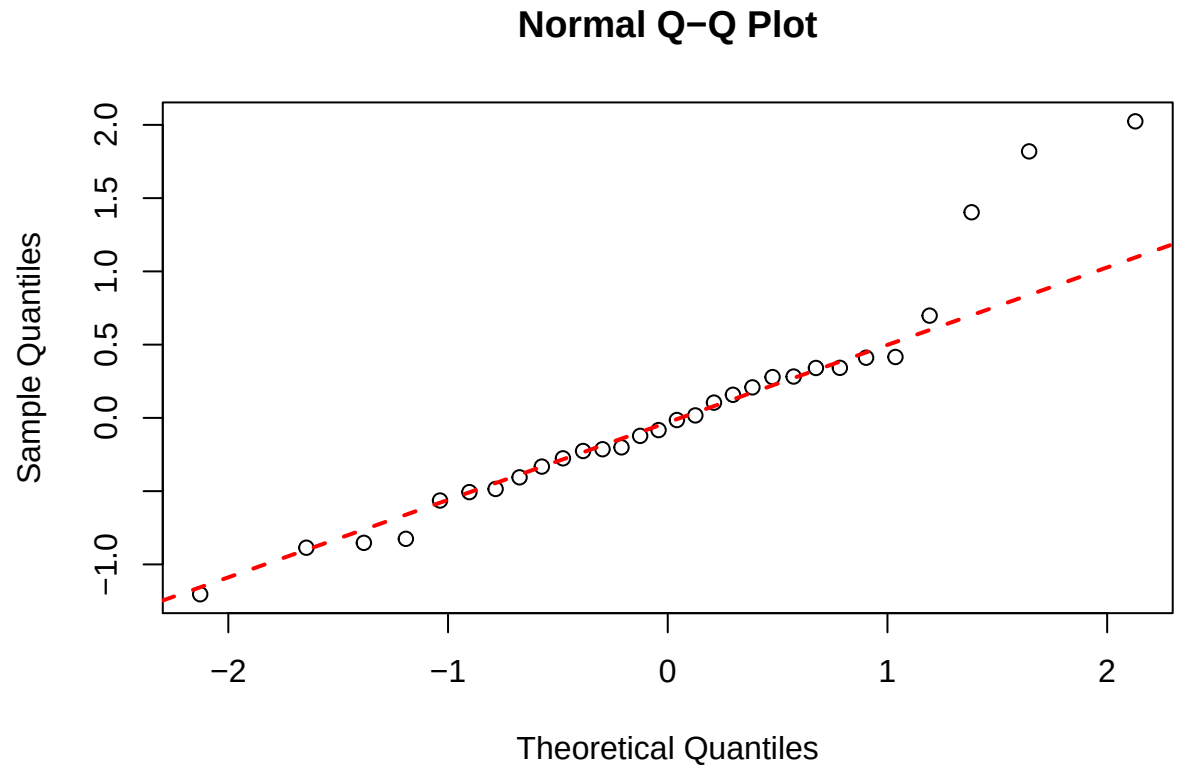
Outra forma de ver a relação acertos x erros é olhando o valor predito versus os residuos.

```
plot(lasso_prediction,lasso_residuos)
abline(h=0,col="blue",lty=2,lwd=2)
```



Um forma de testar se temos um bom modelo, é olhando se os residuos seguem um distribuição normal com média 0. Uma forma, visual, de verificar isso é usando o gráfico do qq-plot. quanto mais próximos da linha da normal estiverem os dados, mais eles se apróximam de uma distribuição normal. Isso é o que acontece com a distribuição dos residuos, pode ser visto a baixo. Sendo assim temos um forte indicio de que temos um bom modelo.

```
qqnorm(lasso_residuos)
qqline(lasso_residuos, col = 2,lwd=2,lty=2)
```



Comparando Resultados

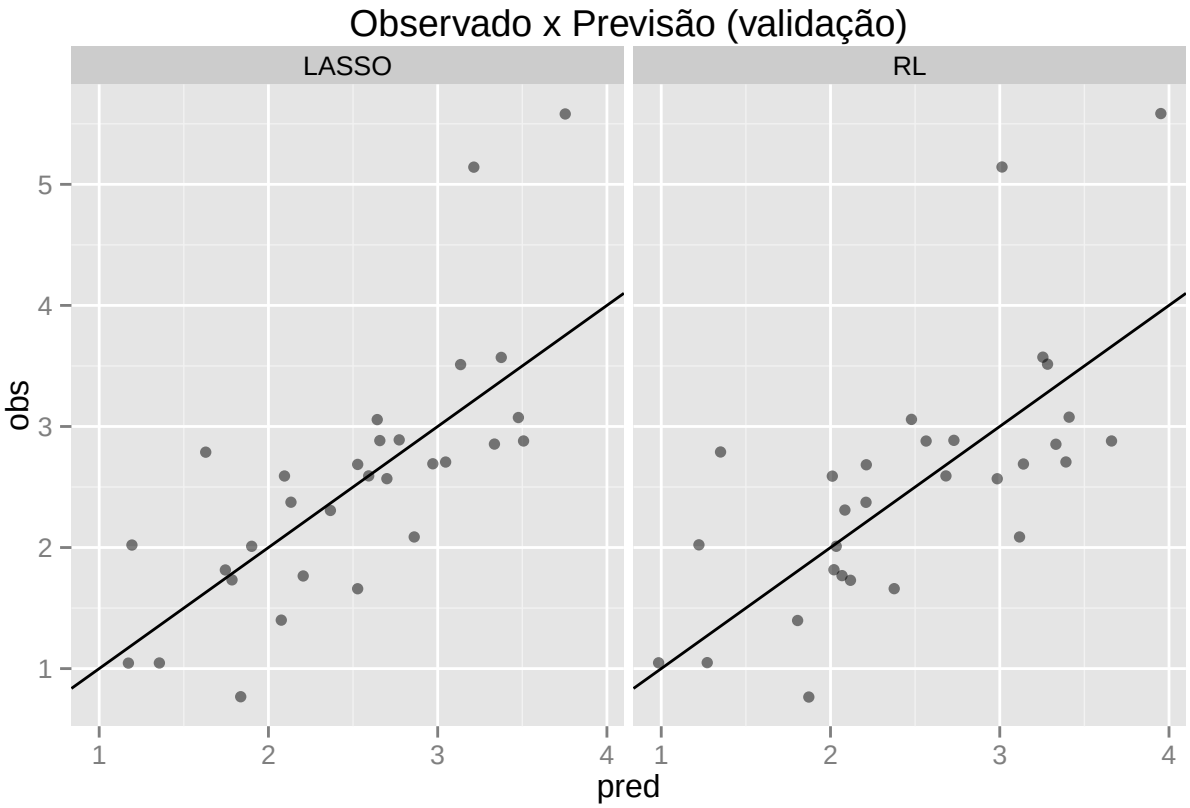
Depois de ter os dois modelos calculados podemos observar qual dos dois tem o melhor resultado. Uma forma inicial de fazer isso é olharmos a comparação dos valores preditos com os reais de forma gráfica.

```
lm_prediction <- data.frame(pred = predicoes, obs = dados_teste$lpsa)
compare <- data.frame(pred = predicoes, obs = dados_teste$lpsa)
compare$model <- "RL"

lasso_prediction <- data.frame(pred = lasso_prediction, obs = dados_teste$lpsa)
lasso_prediction$model <- "LASSO"

compare <- rbind(compare, lasso_prediction)

ggplot(compare, aes(x = pred, y = obs)) +
  geom_point(alpha = 0.5, position = position_jitter(width=0.2)) +
  facet_grid(. ~ model) +
  geom_abline() +
  ggtitle("Observado x Previsão (validação)")
```



Observamos pouca variação entre os resultados. Porém o resultado do LASSO se mostra minimamente mais próximos do real. Comparando os RMSE temos:

RMSELM = 0.7219931,

RMSELASSO = 0.7061778

Como estamos atrás do menor valor do RMSE, podemos dizer que o modelo do laço encontrou sim o melhor modelo. Porém com valores muito próximos dos encontrados com regressão linear multipla, utilizando todas as variáveis.