Lista 2

Lyncoln Sousa

18/07/2020

```
library(caret)
library(ggplot2)
```

1 - Carregue o banco de dados Real_Estate.csv.

```
data = read.csv("real_estate.csv")
```

Ele contém diversas informações sobre imóveis. O objetivo é predizer o pricem2 (preço do metro quadrado do imóvel) em função das variáveis age (idade do imóvel), distance (distância do quartel de bombeiros mais próximo), stores (número de lojas de conveniência próximas).

Para realizar tal tarefa, precisamos decidir qual o melhor método de treinamento do regressor: lm (modelo linear), knn (k-vizinhos próximos) ou rf (floresta aleatória).

- Utilize a mesma metodologia que utilizamos para avaliar classicadores.
- Utilize o método k-fold repetido, com k=10 e 3 repetições.
- Lembre de xar o set.seed(100) antes de cada treinamento.
- Ao nal, redija um texto justicando qual método elegeu como melhor para esse pro- blema.

lm

```
set.seed(100)
modelo_lm = train(pricem2 ~ age + distance + stores,
                  data = data,
                  method = "lm",
                  trControl = trainControl(method = "repeatedcv",
                                           number = 10,
                                           repeats = 3)
                  )
```

knn

```
set.seed(100)
modelo_knn = train(pricem2 ~ age + distance + stores,
                  data = data,
                  method = "knn",
                  trControl = trainControl(method = "repeatedcv",
                                           number = 10,
                                           repeats = 3)
                  )
```

rf

```
set.seed(100)
modelo_rf = train(pricem2 ~ age + distance + stores,
                  data = data,
                  method = "rf",
                  trControl = trainControl(method = "repeatedcv",
                                           number = 10,
                                           repeats = 3)
                  )
```

```
## note: only 2 unique complexity parameters in default grid. Truncating the grid to 2 .
```

Comparando os modelos

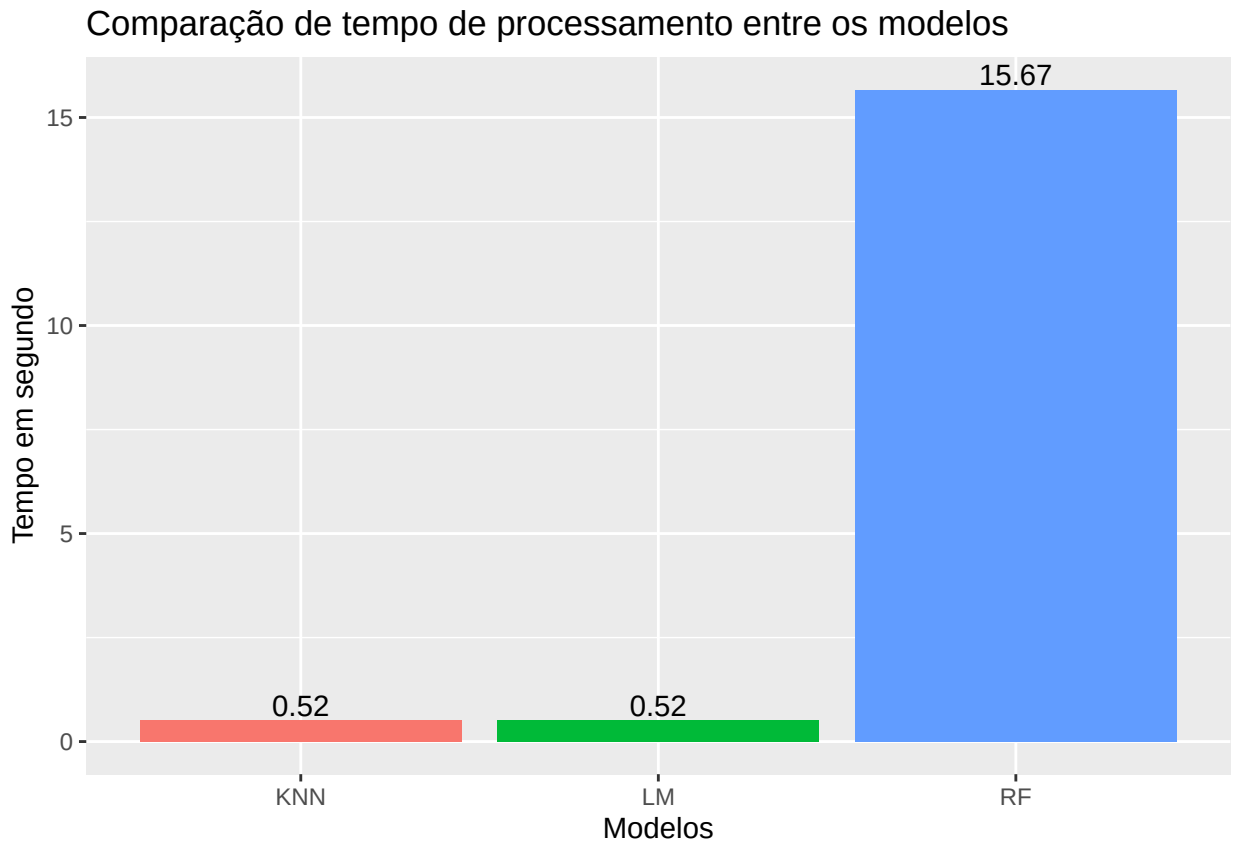
```
resultados = resamples(list(LM = modelo_lm,
                            KNN = modelo_knn,
                            RF = modelo_rf))
summary(resultados)
```

```
##
## Call:
## summary.resamples(object = resultados)
##
## Models: LM, KNN, RF
## Number of resamples: 30
##
## MAE
##         Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## LM  4.200198 5.892169 6.466183 6.558289 7.254442 9.252147    0
## KNN 3.904492 4.975541 5.548330 5.626460 6.092533 8.278500    0
## RF  3.056252 4.318769 4.964286 4.901926 5.484332 6.785243    0
##
## RMSE
##         Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## LM  5.512714 7.362158 8.581350 9.017692 9.683342 15.95866    0
## KNN 5.134157 6.594747 7.794386 8.057142 8.728222 14.03805    0
## RF  3.794911 5.686386 6.957627 7.310426 7.948350 13.91973    0
##
## Rsquared
##          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## LM  0.2368032 0.4838605 0.5867511 0.5646433 0.6739797 0.7817629    0
## KNN 0.3671166 0.6065154 0.6563854 0.6528387 0.7503466 0.8658804    0
## RF  0.3450696 0.6567264 0.7215267 0.7086823 0.8261777 0.9266953    0
```

Observa-se que os valores descritívos das métricas foram mais favoráveis para o modelo de random forest (RF) comparado com os de LM e KNN. Obteve-se um R^2 maior e erros de RMSE E MAE menores. Comparando as métricas de KNN e LM é possível observar que foram mais parecidas, tendo uma vantagem para KNN, que em relação a LM registrou valores descritívos maiores para R^2 e menores para os erros de RMSE e MAE.

```
#Gráfico de barras para comparação de tempo
ggplot(resultados$timings, aes(x = row.names(resultados$timings),
```

```
                              y = Everything,
                              fill = row.names(resultados$timings))) +
  geom_bar(stat = "identity") +
  labs(title = "Comparação de tempo de processamento entre os modelos",
       y = "Tempo em segundo",
       x = "Modelos") +
  geom_text(aes(label = sprintf("%.2f", resultados$timings$Everything),
                y= resultados$timings$Everything),vjust = -0.2)+
  theme(legend.position = "none")
```



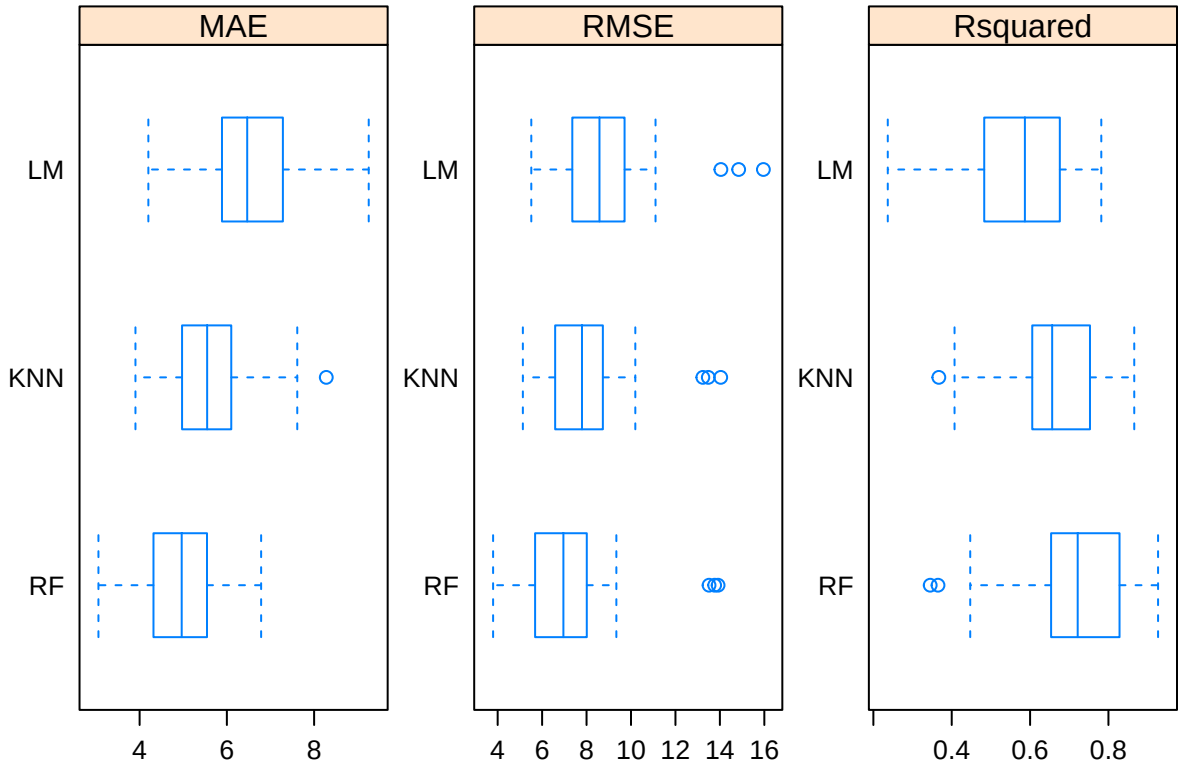
Em relação ao tempo de processamento, os modelos LM e KNN tiveram um desempenho bastante parecido. Comparando o modelo de RF com os demais, é possível observar que o mesmo demorou significativamente mais para ser executado. Então surge um questionamento, mesmo com valores descritivos melhores para as métricas do modelo RF, será que compensa pela sua demora de execução? Tendo em vista que para computadores com processadores mais "fracos" e/ou base de dados maiores esse valor pode aumentar ainda mais em relação aos outros dois modelos, possivelmente, podendo até não ser viável a execução para alguns computadores. Então, é interessante avaliar dois possíveis cenários:

- Cenário 1: O mais ideal, onde é possível ter acesso um computador que torne viável a execução do modelo de RF;

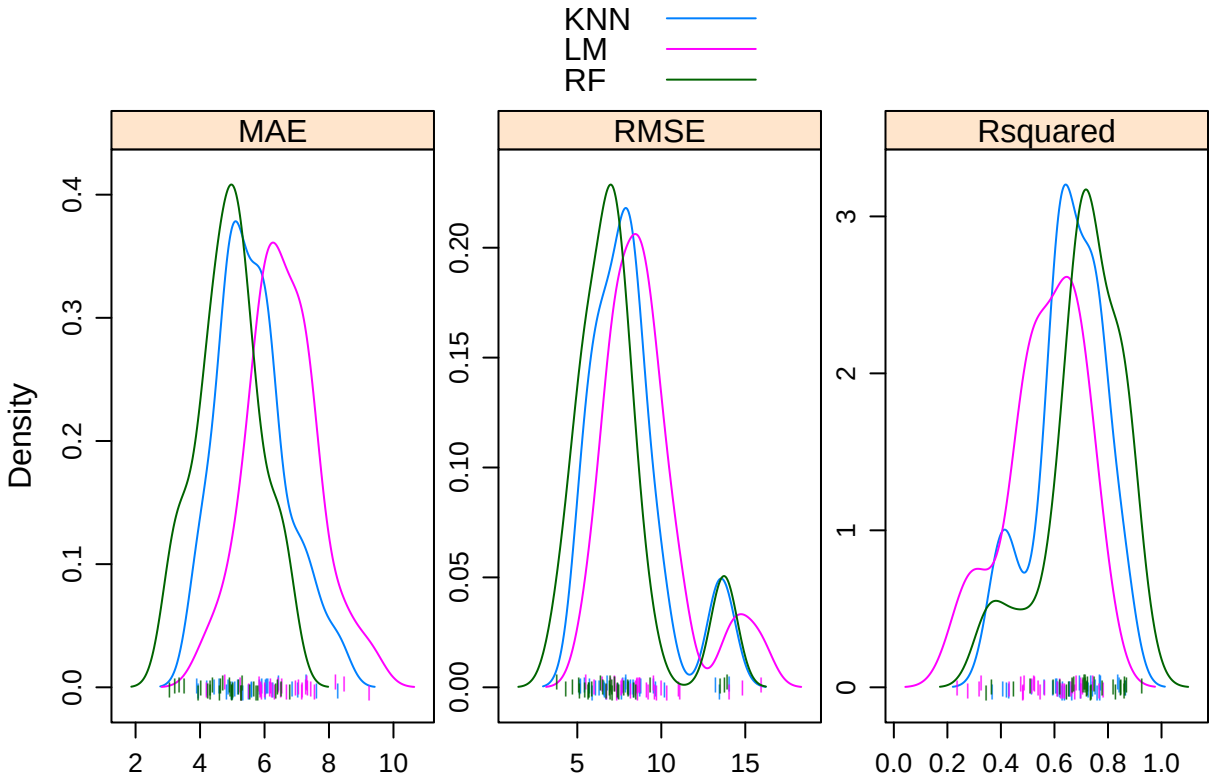
- Cenário 2: Onde só pode ter acesso a um computador "fraco" e/ou a base de dados seja muito grande para realizar a modelagem por RF, por conta disso, é interessante avaliar o desempenho dos modelos de LM e KNN.

```
#Boxplot
escalas = list(x=list(relation="free" ),
```

```
              y=list(relation="free" ))
bwplot(resultados,
       scales = escalas,
       pch = "|")
```




```
#Density
densityplot(resultados,
       scales = escalas,
       pch = "|",
       auto.key = TRUE)
```

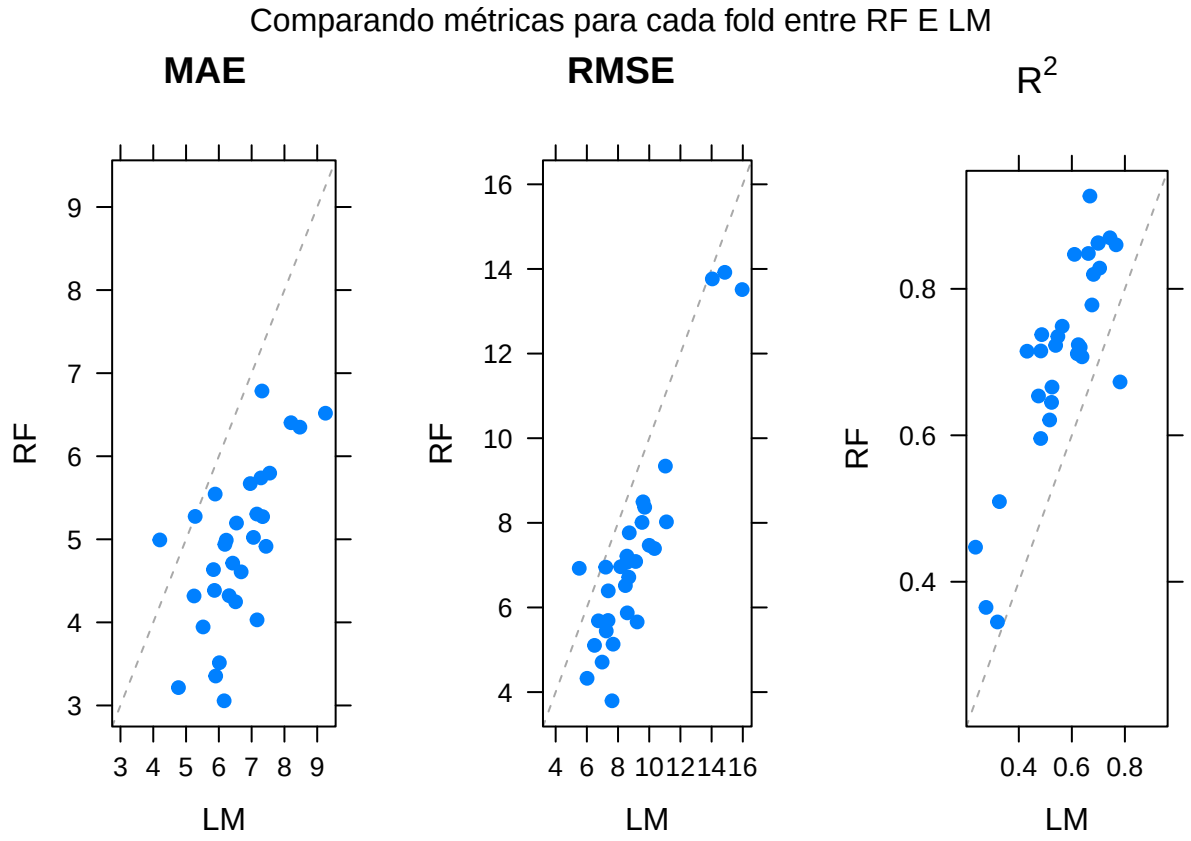


Cenário 1: Analisando o boxcomplot e o gráfico de densidade, confirma-se de maneira visual a superioridade do modelo de RF. Nota-se que o modelo de RF possui os erros MAE E RMSE mais baixos e sua variação não é muito grande, tendo alguns out-liers para RMSE. Enquanto seu R^2 se concentra mais para valores maiores em comparação aos 2 outros modelos, vale a pena observar que há 2 out-liers inferiores que fazem o R^2 do modelo de RF ser menor que valores obtidos na modelagem KNN.

Cenário 2: Comparando KNN contra LM, nota-se que há uma maior concentração para valores menores dos erros de MAE e RMSE para a modelagem KNN. Enquanto para R^2 O modelo LM possui mais variação nos valores obtidos e se concentra para valores menores comparado ao KNN. Comparando os dois modelos, percebe-se que o modelo KNN é superior ao LM.

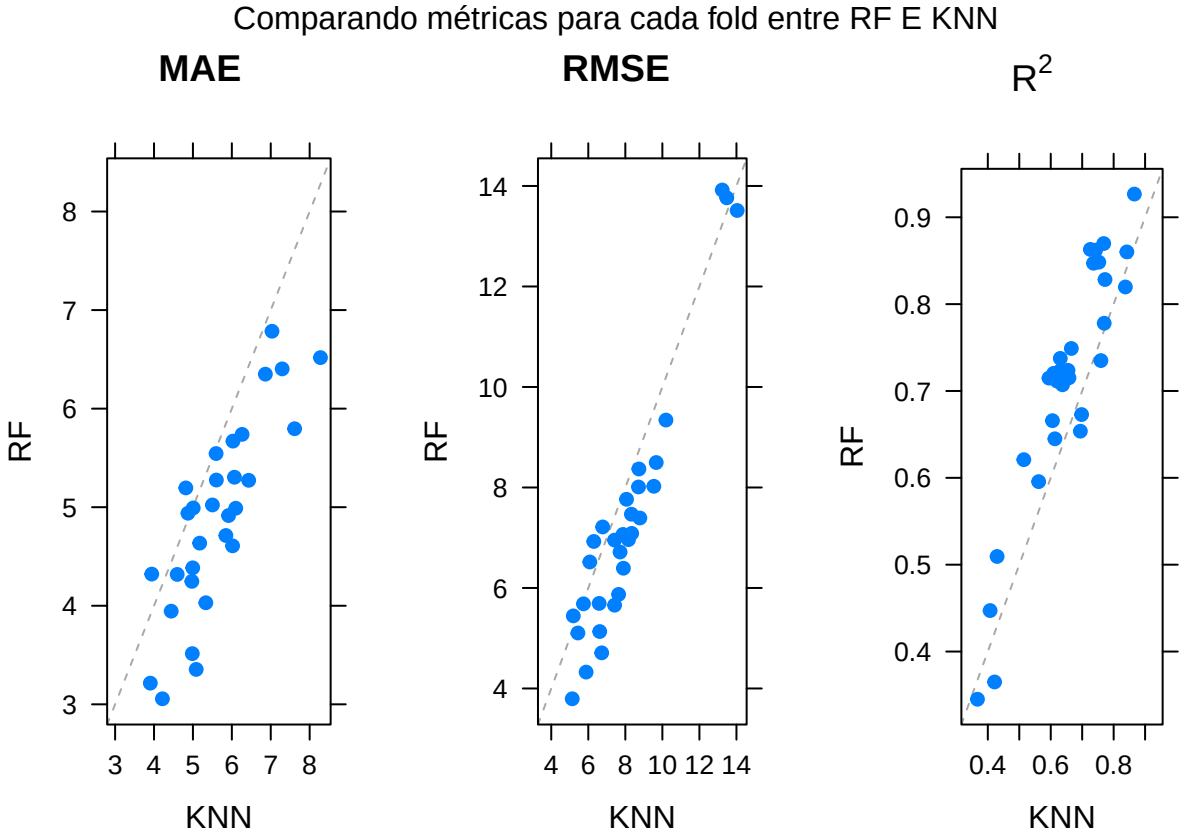
Agora será comparado as métricas obtidas para cada reamostragem realizada pelo método de k-fold repetido, com k=10 e 3 repetições, comparando os modelos em 2 a 2.

```
gridExtra::grid.arrange(
  xyplot(resultados, models = c("RF","LM"), metric = "MAE", pch = 19),
  xyplot(resultados, models = c("RF","LM"), metric = "RMSE", pch = 19),
  xyplot(resultados, models = c("RF","LM"), metric = "Rsquared", pch = 19),
  ncol = 3,
  top = "Comparando métricas para cada fold entre RF E LM"
)
```



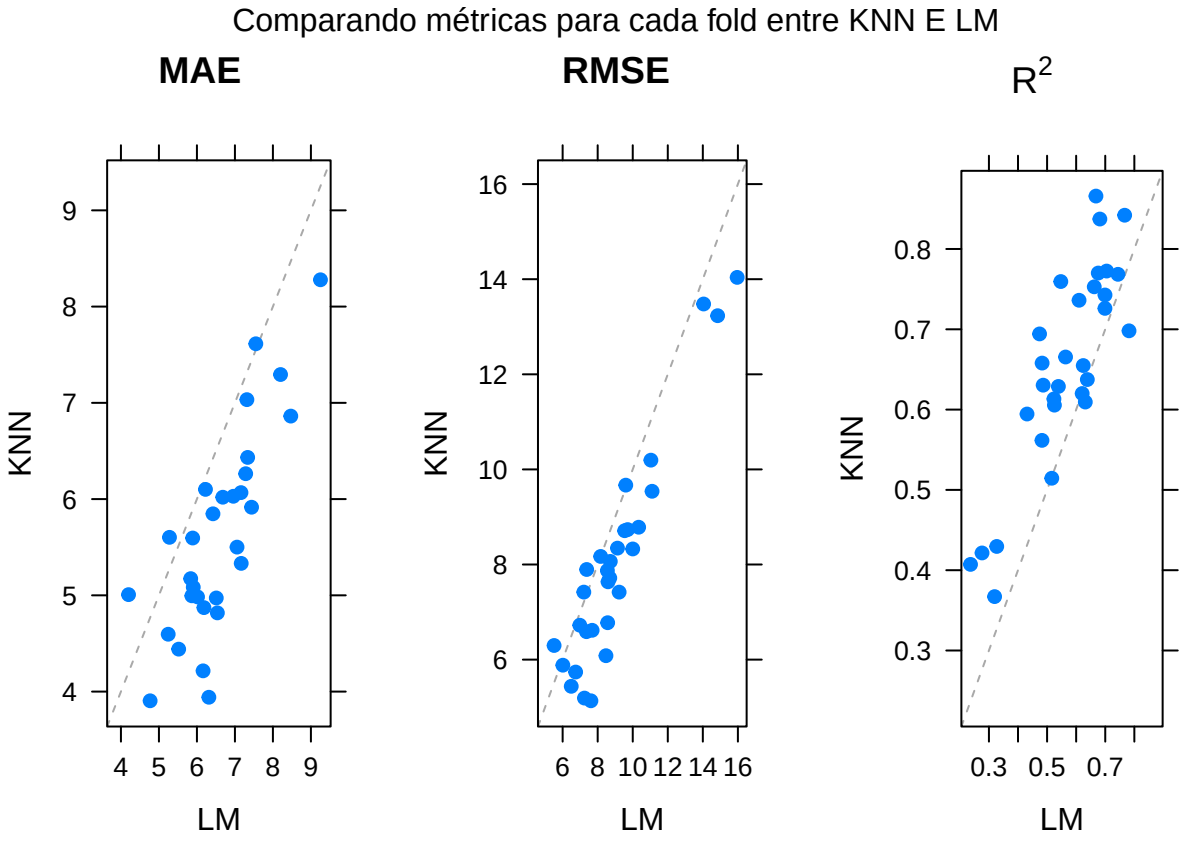
Observa-se que para os erros MAE e RMSE, a grande maioria dos folds apresentaram maiores valores para o modelo de LM. Em relação ao R^2, houve uma grande concentração para maiores valores para o modelo RF. Ou seja, os gráficos mostram a melhor eficiência do modelo RF em relação ao LM

```
gridExtra::grid.arrange(
  xyplot(resultados, models = c("RF","KNN"), metric = "MAE", pch = 19),
  xyplot(resultados, models = c("RF","KNN"), metric = "RMSE", pch = 19),
  xyplot(resultados, models = c("RF","KNN"), metric = "Rsquared", pch = 19),
  ncol = 3,
  top = "Comparando métricas para cada fold entre RF E KNN"
)
```



Os gráficos de comparação dos folds entre RF e KNN se assemelham aos gráficos da comparação anterior, mostrando a superioridade do modelo RF em relação ao KNN, assim mais uma vez foi apresentado a superioridade do cenário 1 em relação ao cenário 2.

```
gridExtra::grid.arrange(
  xyplot(resultados, models = c("KNN","LM"), metric = "MAE", pch = 19),
  xyplot(resultados, models = c("KNN","LM"), metric = "RMSE", pch = 19),
  xyplot(resultados, models = c("KNN","LM"), metric = "Rsquared", pch = 19),
  ncol = 3,
  top = "Comparando métricas para cada fold entre KNN E LM"
)
```



Observa-se agora a comparação das métricas dos modelos KNN e LM, ilustrando o cenário 2. O modelo LM apresenta maiores valores para os erros MAE E RMSE na maioria dos folds, enquanto a maioria das suas métricas de R^2 são menores do que a do modelo KNN. Então, prefere-se a utilização do modelo de KNN em relação ao de LM.

Agora irá ser realizado testes de hipótese para a diferença entre as métricas para os modelos, comparando 2 a 2. Irá ser adotado um nível de significância $\alpha = 0.05$.

```
difs = diff(resultados)
summary(difs)
```

```
##
## Call:
## summary.diff.resamples(object = difs)
##
## p-value adjustment: bonferroni
## Upper diagonal: estimates of the difference
## Lower diagonal: p-value for H0: difference = 0
##
## MAE
##     LM         KNN       RF
## LM             0.9318    1.6564
## KNN 1.098e-07            0.7245
## RF  8.048e-11 8.860e-07
##
## RMSE
```

```
##     LM        KNN       RF
## LM            0.9605    1.7073
## KNN 1.713e-06           0.7467
## RF  2.518e-09 3.657e-05
##
## Rsquared
##     LM        KNN       RF
## LM            -0.08820  -0.14404
## KNN 1.123e-06           -0.05584
## RF  1.708e-10 1.320e-05
```

Para todos os testes realizados foram obtidos $p - valor < \alpha$, ou seja, rejeitamos a hipótese nula H_0. Então pode-se concluir que para a comparação 2 a 2 dos modelos, há diferença entre todas as métricas.

2 - Uma vez escolhido o melhor método regressor, vamos criar o regressor de fato. Fixe a semente no valor 100. Crie amostras treino (75%) e teste (25%). Utilize o método de re-amostragem bootstrap, com 20 re-amostragens.

Como é possível realizar a modelagem por RF neste computador, e tendo em vista a superioridade do modelo em relação aos demais obtidas nas análises anteriores, irá ser utilizado o modelo de RF para treino e teste dos dados.

```
set.seed(100)

amostra_treino = createDataPartition(y = data$pricem2,
                                     p = 0.75,
                                     list = F)
treino = data[amostra_treino,]
teste = data[-amostra_treino,]

set.seed(100)
modelo_boot_rf = train(pricem2 ~ age + distance + stores,
                       data = treino,
                       method = "rf",
                       trControl = trainControl(method = "boot",
                                                number = 20)
                       )
```

```
## note: only 2 unique complexity parameters in default grid. Truncating the grid to 2 .
```

3 - Treine o regressor pelo método escolhido e calcule uma estimativa para o RMSE, R2 e MAE fora da amostra (Out-of-Sample).

```
predicao_boot_rf = predict(modelo_boot_rf,teste)
resultado = postResample(predicao_boot_rf, teste$pricem2)
resultado
```

```
##       RMSE   Rsquared        MAE
## 10.0492257  0.6045582  5.5109270
```