Titanic: Machine Learning from Disaster - Naives Bayes

Vanderlei Kleinschmidt

07/12/2020

Pré processamento

Como eu já fiz todo o pré processamento dos dados anteriormente, no script disponível no repositório "titanic_knn", não faz sentido eu repetir todos os passos. Por isso, eu começo rodando o pré processamento em um outro arquivo R.

```
source("prepro_titanic.R")
```

packages

Vou usar o pacote 'naivebayes' para treinar o modelo de machine learning. Poderia usar o pacote 'e1071', mas dei preferência pelo 'naivebayes' por causa da documentação que é bem interessante, pois apresenta todas as equações do algorítmo, o que facilita muito o meu entendimento do que extamente estou fazendo.

```
library(naivebayes)

library(caret)
```

Para treinar o Naive Bayes precisamos nos certificar que os dados estão com os tipos corretos. O algorítmo requer que a variável dependente seja do tipo factor ou character.

```
sapply(train, class)
```

```
##  Survived       Age     SibSp     Parch      Fare    Female   FClasse   SClas
se
##  "factor" "numeric" "numeric" "numeric" "numeric"  "factor"  "factor"  "facto
r"
## CEmbarked QEmbarked
##  "factor"  "factor"
```

```
head(train, 10)
```

```
##    Survived        Age SibSp     Parch       Fare Female FClasse SClasse
## 1         0 0.27117366 0.125 0.0000000 0.01415106      0       0       0
## 2         1 0.47222920 0.125 0.0000000 0.13913574      1       1       0
## 3         1 0.32143755 0.000 0.0000000 0.01546857      1       0       0
## 4         1 0.43453129 0.125 0.0000000 0.10364430      1       1       0
## 5         0 0.43453129 0.000 0.0000000 0.01571255      0       0       0
## 6         0 0.29630560 0.000 0.0000000 0.01650950      0       0       0
## 7         0 0.67328474 0.000 0.0000000 0.10122886      0       1       0
## 8         0 0.01985423 0.375 0.1666667 0.04113566      0       0       0
## 9         1 0.33400352 0.000 0.3333333 0.02173075      1       0       0
## 10        1 0.17064589 0.125 0.0000000 0.05869429      1       0       1
```

```
##    CEmbarked QEmbarked
## 1          0         0
## 2          1         0
## 3          0         0
## 4          0         0
## 5          0         0
## 6          0         1
## 7          0         0
## 8          0         0
## 9          0         0
## 10         1         0
```

A função naivebayes() usa uma distribuição de probabilidade diferente para modelagem da probabilidade condicional de classe. Testei pelo menos três distribuições diferentes e os resultados não foram significativamente distintos, o que me levou a considerar neste script apenas a distribuição de Poisson.

```
nb_poisson <- naive_bayes(train$Survived ~ ., train, laplace = 0, usekern
el = FALSE, usepoisson = TRUE)
```

Uma vez que o modelo foi treinado, preciso verificar qual a capacidade preditiva dele. Para isso eu crio um objeto chamado 'previsao_p', uso a função predict, passando como parâmetro o modelo treinado e os dados de treino.

```
previsao_p = predict(nb_poisson, newdata = train[-1])
```

Crio a matriz de confução para avaliar a capacidade preditiva do modelo:

```
matriz = table(train$Survived,previsao_p)
```

A matriz de confusão é criada usando a função 'confusionMatrix' do pacote 'caret'.

```
confusionMatrix(matriz)
```

```
## Confusion Matrix and Statistics
##
##    previsao_p
##       0   1
##   0 486  63
##   1 116 226
##
##                Accuracy : 0.7991
##                  95% CI : (0.7713, 0.8249)
##     No Information Rate : 0.6756
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5625
##
##  Mcnemar's Test P-Value : 0.0001016
##
##             Sensitivity : 0.8073
##             Specificity : 0.7820
```

```
##          Pos Pred Value : 0.8852
##          Neg Pred Value : 0.6608
##              Prevalence : 0.6756
##          Detection Rate : 0.5455
##    Detection Prevalence : 0.6162
##       Balanced Accuracy : 0.7947
##
##        'Positive' Class : 0
##
```

Dá para perceber que o modelo não tem uma acurácia melhor do que a do KNN, que empreguei anteriormente. Isso pode ser observado no site do Kagle também!

Sumarizando o resultado do modelo treinado:

```
summary(nb_poisson)
```

```
##
## ================================== Naive Bayes ======================
===========
##
## - Call: naive_bayes.formula(formula = train$Survived ~ ., data = train
,      laplace = 0, usekernel = FALSE, usepoisson = TRUE)
## - Laplace: 0
## - Classes: 2
## - Samples: 891
## - Features: 9
## - Conditional distributions:
##     - Bernoulli: 5
##     - Gaussian: 4
## - Prior probabilities:
##     - 0: 0.6162
##     - 1: 0.3838
##
## ------------------------------------------------------------------------
-----------
```

Agora que o modelo foi treinado, posso usá-lo nos dados de teste para depois gerar o arquivo e enviar ao Kagle

```
modelo_p <- predict(nb_poisson, test, type = "class")
```

Gerando o arquivo e enviado ao Kagle:

```
submission_p <- data.frame(PassengerId = passengers, Survived = modelo_p)
write.csv(submission_p,'titanic_nb_p.csv', row.names = FALSE)
```