U.PORTO

FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO

Professor Joaquim Costa

Métodos Estatísticos em *Data Mining*

Segundo Projeto: Aprendizagem Supervisionada Classificatória

Maio de 2023

Trabalho realizado por:

Pedro Leite - 201906697

Pedro Carvalho - 201906291

Índice:

1. Introdução

O objetivo deste projeto é fazer uma análise dos atributos do *dataset* (2.) "Telco Customer Churn", para perceber a sua influência nos clientes que mudaram de operadora. Para fazer esta análise vamos utilizar vários métodos estatísticos de *data mining*, nomeadamente: (4.1.) *Naive Bayes*, (4.2.) LDA e QDA, (4.3.) Regressão Logística, (4.4.) Árvore de Decisão, (4.5.) Método do Núcleo, (4.6.) Método dos K-Vizinhos mais Próximos, (4.7.) SVM e (4.8.) Redes Neuronais. Estes são métodos de aprendizagem supervisionada classificatória. No final, iremos apresentar uma discussão e conclusão dos resultados.

2. Background

O *dataset* que escolhemos para este projeto foi o [1] "Telco Customer Churn". Este *dataset* contém 7043 clientes e 21 atributos, 11 dos clientes não foram utilizados porque têm atributos com valores não atribuídos. Os atributos são:

- "CustomerID" é uma *string* única que identifica o cliente.
- "Gender" é uma *string* que identifica o cliente como "Female" ou "Male".
- "SeniorCitizen" é um número binário se o cliente é ou não idoso.
- "Partner" é uma *string* "Yes" ou "No" que diz se o cliente está casado.
- "Dependents" é uma *string* "Yes" ou "No" que diz se o cliente tem filhos.
- "Tenure" é o número de meses que o cliente está na operadora atual.
- "PhoneService" é uma *string* "Yes" ou "No" que diz se o cliente tem serviço de telefone.
- "MultipleLines" é uma *string* "Yes", "No" ou "No phone service" que diz se o cliente tem múltiplas linhas de telemóvel.
- "InternetService" é uma *string* "DSL", "Fiber optic" ou "No" que diz que tipo de serviço de internet o cliente possui.
- "OnlineSecurity" é uma *string* "Yes", "No" ou "No internet service" que diz se o cliente tem segurança online.
- "OnlineBackup" é uma *string* "Yes", "No" ou "No internet service" que diz se o cliente tem *backup* online.

- "DeviceProtection" é uma *string* "Yes", "No" ou "No internet service" que diz se o cliente tem proteção do dispositivo .
- "TechSupport" é uma *string* "Yes", "No" ou "No internet service" que diz se o cliente tem suporte técnico.
- "StreamingTV" é uma *string* "Yes", "No" ou "No internet service" que diz se o cliente tem *streaming* de canais televisivos.
- "StreamingMovies" é uma *string* "Yes", "No" ou "No internet service" que diz se o cliente tem *streaming* de filmes.
- "Contract" é uma *string* "Month-to-month", "One year" ou "Two year" que diz que tipo de contrato o cliente tem.
- "PaperlessBilling" é uma *string* "Yes" ou "No" que diz se o o cliente tem faturamento sem papel.
- "PaymentMethod" é uma *string* "Electronic check", "Mailed check", "Bank transfer (automatic)" ou "Credit card (automatic)" que diz o tipo de pagamento que o cliente utiliza.
- "MonthlyCharges" é o montante que o cliente paga mensalmente.
- "Total Charges" é o montante que o cliente pagou até agora.
- "Churn" é o atributo objetivo que quer dizer se um cliente mudou de operadora, no último mês.

Aplicamos *One Hot Encoding* ao nosso *dataset*, que consiste em substituir todos os valores das variáveis categóricas, por números, por exemplo, no "Multiple Lines" substituímos o "Yes", "No" e "No phone service" por "1", "2" e "3", respetivamente. Mas após aplicar os métodos os resultados que obtemos foram muito confusos e difíceis de interpretar. Logo, ao longo do trabalho apenas utilizamos as variáveis binárias e contínuas, ou seja: "Gender", "SeniorCitizen", "Partner", "Dependents", "Tenure", "PhoneService", "PaperlessBilling", "MonthlyCharges", "Total Charges" e "Churn".

Separamos os dados onde 70% são usados para treino e 30% para teste.

3. Materiais

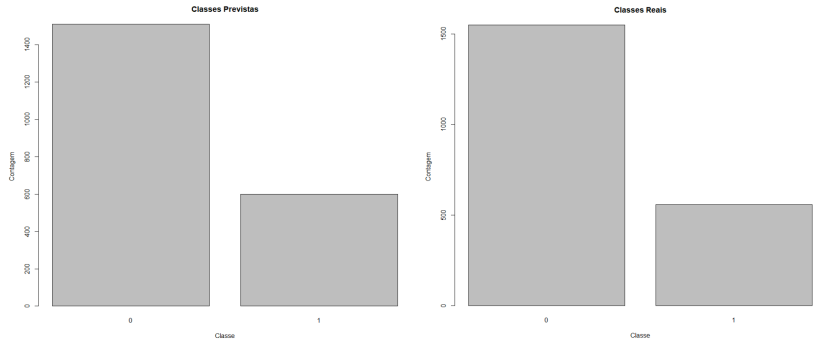
O projeto foi feito utilizando a linguagem R, com as bibliotecas: "MASS", "e1071", "ggplot2", "tidyverse", "caret", "naivebayes", "stats", "ggplot2",

"rpart", "rpart.plot", "class", "keras" e "reticulate". Os resultados foram corridos num computador com as seguintes especificações: AMD Rysen 5 2600X SixCore Processor 3.6 GHz, 16GB RAM, Asus GeForce GTX 1660 Ti.

4. Estudo e Aplicação

4.1. *Naive Bayes*

Naive Bayes é um algoritmo que se baseia no Teorema de Bayes, para fazer classificação de dados. Começamos por calcular as probabilidades *a priori*. Construímos um modelo de *Naive Bayes* usando as variáveis predicativas e a variável objetivo. Calculamos as probabilidades *a posteriori* das variáveis para uma determinada instância. Para cada instância nos dados de teste, calculamos as probabilidades *a posteriori*. Atribuímos o rótulo da variável objetivo à instância com base na variável com maior probabilidade *a posteriori*. Podemos analisar os resultados através destes gráficos de barra, que comparam as classes previstas e as classes reais:



Como podemos verificar os resultados estão semelhantes como pretendido.
Obtemos a seguinte taxa de erro:

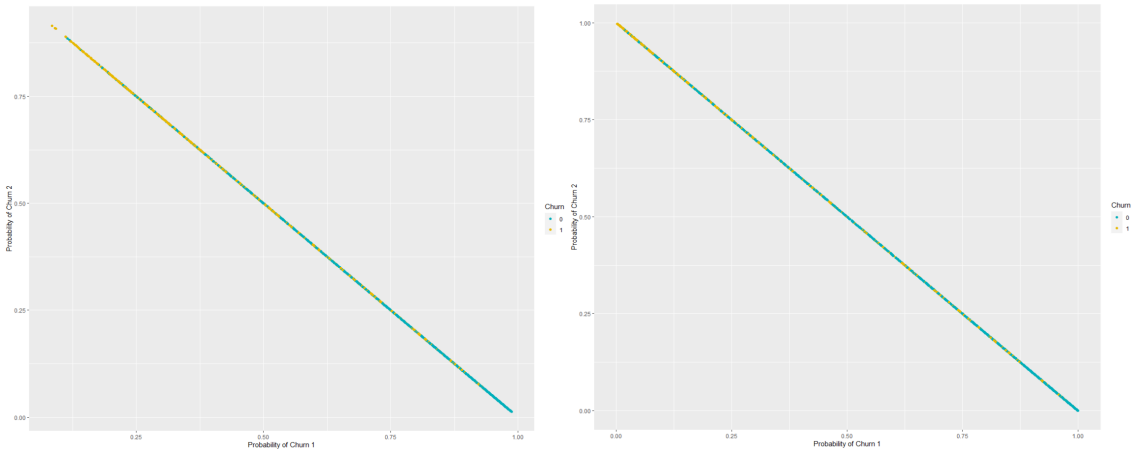
```
> error_rate <- mean(test_data$Churn != test_data$predicted)
> print(error_rate)
[1] 0.2388626
```

4.2. LDA e QDA

O modelo LDA assume que as variáveis predicativas têm distribuição normal e que a variância das variáveis é igual em todas. O LDA cria uma função linear que maximiza a separação entre as variáveis. E depois calcula as probabilidades

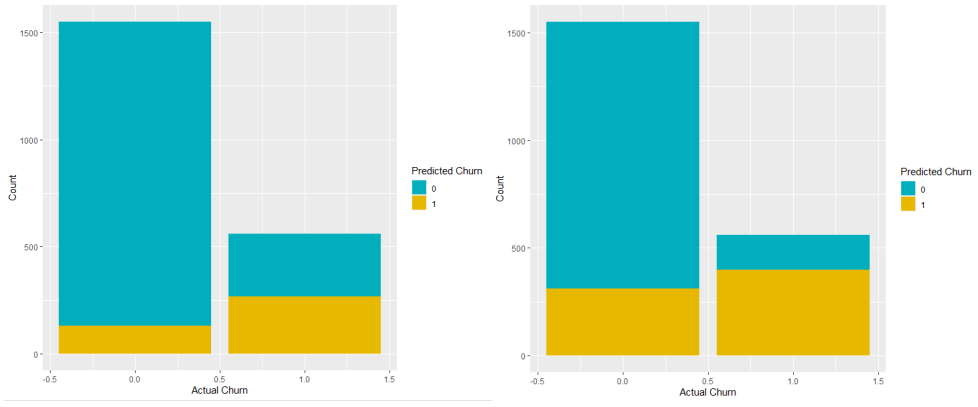
a posteriori das variáveis para cada observação e atribui a observação à variável com a maior probabilidade. Por outro lado, o modelo QDA não assume igualdade da variância entre as variáveis. Em vez disso, o QDA assume que as variáveis predicativas têm distribuições normais, mas permite que a variância mude entre as variáveis. O QDA cria uma função quadrática que descreve a separação entre as variáveis e tal como o LDA, calcula as probabilidades *a posteriori* das variáveis para cada observação e atribui a observação à variável com a maior probabilidade.

Gráficos com a separação das variáveis do LDA, QDA, respetivamente:



Como os gráficos são linhas diagonais, significa que o "Churn1" e o "Churn2" são iguais para todos os pontos dos dados. E como é previsível quão maior for a probabilidade dos modelos em dar "Churn", menor vai ser a probabilidade de não dar, e vice-versa. Também é possível observar que há uma falta de dispersão de dados, o que significa que os modelos não conseguem distinguir entre as variáveis e separá-las.

Gráficos com as previsões do LDA e QDA, respetivamente:



Neste gráfico o “x” representa a churn real e o “y” a contagem de clientes em cada categoria. As barras mostram a distribuição das variáveis de “Churn” previstas com base nas variáveis de “Churn" real. Ambos os modelos têm tendência em classificar mais observações como não “Churn”.

4.3. Regressão Logística

Aplicamos o método da Regressão Logística, para modelar a relação entre a variável objetivo e um conjunto de variáveis independentes. Com o objetivo de estimar a probabilidade de ocorrência de um evento ou resultado em função das variáveis independentes. Depois de aplicar o modelo, fazemos previsões com base no modelo ajustado.

```
Call:
glm(formula = Churn ~ gender + Partner + SeniorCitizen + Dependents +
    tenure + PhoneService + PaperlessBilling + TotalCharges +
    MonthlyCharges, family = binomial, data = df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9420  -0.6895  -0.3716   0.7355   3.2678

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)     -1.055e+00  1.500e-01  -7.033 2.02e-12 ***
gender           3.406e-03  6.294e-02   0.054 0.956850
Partner          2.349e-02  7.541e-02   0.311 0.755440
SeniorCitizen    4.879e-01  8.177e-02   5.967 2.41e-09 ***
Dependents      -3.155e-01  8.636e-02  -3.653 0.000259 ***
tenure          -7.038e-02  5.631e-03 -12.499  < 2e-16 ***
PhoneService    -8.803e-01  1.173e-01  -7.505 6.14e-14 ***
PaperlessBilling 5.321e-01  7.121e-02   7.473 7.86e-14 ***
TotalCharges     1.811e-04  6.277e-05   2.885 0.003911 **
MonthlyCharges   2.845e-02  1.911e-03  14.886  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 8143.4  on 7031  degrees of freedom
Residual deviance: 6174.5  on 7022  degrees of freedom
AIC: 6194.5

Number of Fisher Scoring iterations: 6
```

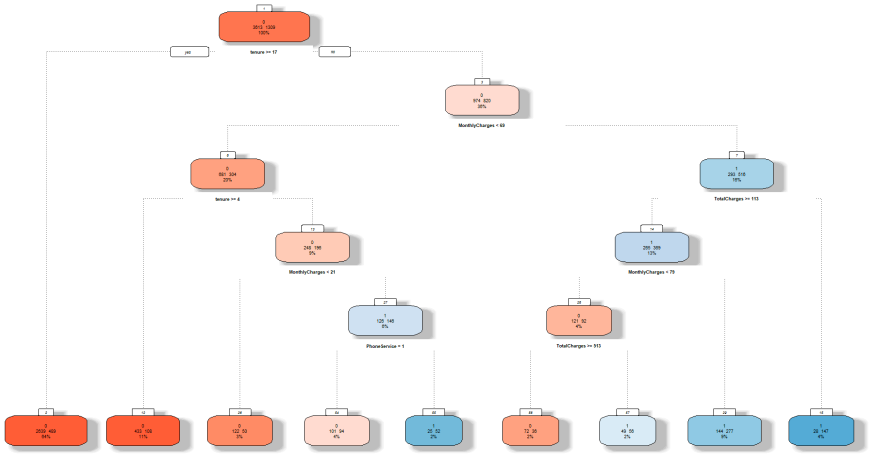
- Nos coeficientes verificamos que temos estimativas positivas em: “gender”, “Partner”, “SeniorCitizien”, “PaperlessBilling”, “TotalCharges” e “MonthlyCharges”, o que indica que que estas variáveis têm um efeito positivo na probabilidade de ocorrência de “Churn”.
- Temos também o erro padrão e o valor z para cada uma das variáveis.
- Verificamos também que “SeniorCitizien”, “Dependents”, “tenure”, ” PhoneService”, “PaperlesseBilling”, “TotalCharges” e ”MonthlyCharges”

apresentam valores-p abaixo de 0.5, o que representa que têm uma significância estatística positiva em relação ao "Churn".

- Depois de analisar a "Deviance Residuals", "Null Deviance", a "Residual Deviance" e o "AIC", conseguimos interpretar que o modelo ajustado comas variáveis independentes é melhor em relação ao modelo inicial.
- O número de iterações para calcular o coeficiente do modelo foi 6.

4.4. Árvores de Decisão

Árvores de Decisão dividem o conjunto e dados em subconjuntos menores com base nas características selecionadas e criam uma estrutura de árvore para tomar decisões. Cada nó representa uma condição ou uma característica, e cada ramo representa uma escolha possível com base na condição. As folhas representam a variável objetivo. Calculamos o modelo que simula a Árvore de Decisão do nosso *dataset:*

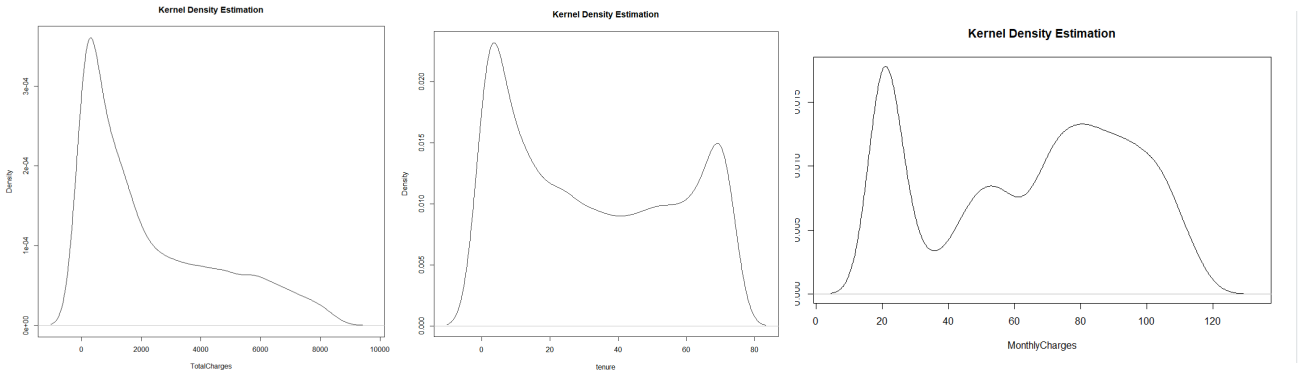


previsoes	0	1
0	1460	339
1	90	221

A precisão que obtemos foi 79.5%. Calculamos também o *Gini Index*, que é uma media de desigualdade que quantifica a divergência entre a distribuição de frequência observada no conjunto de dados e uma distribuição igualitária. O valor que obtivemos foi 0.02368064, o que significa que há uma baixa desigualdade nos dados.

4.5. Método do Núcleo

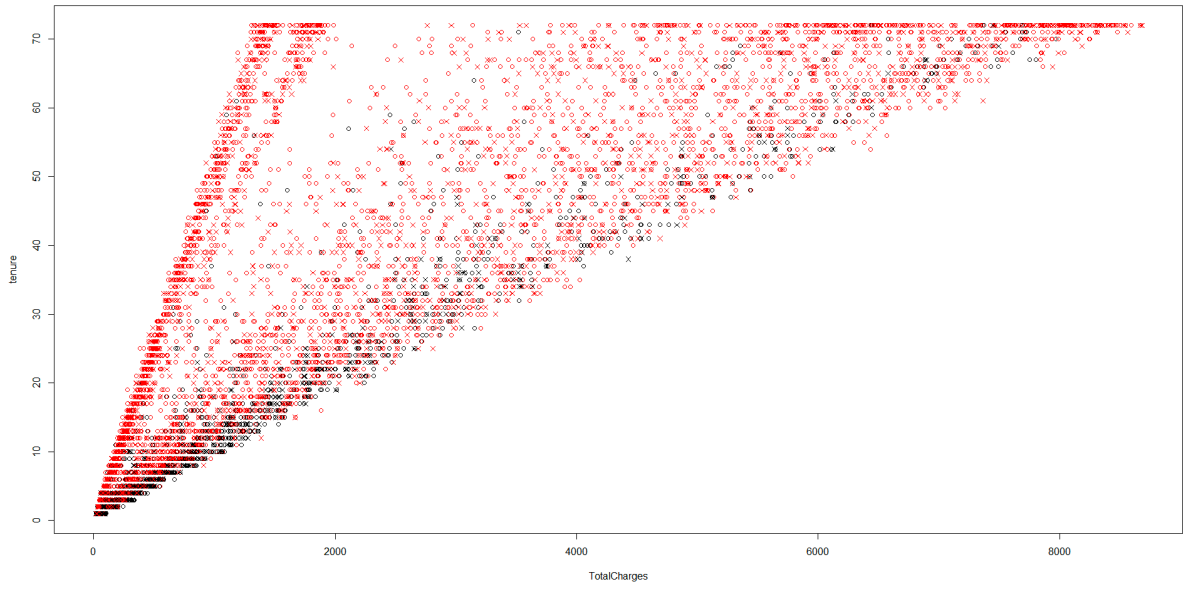
O Método do Núcleo é o usado para calcular a função de densidade de probabilidade de uma variável contínua. Aplicamos este método às variáveis contínuas "Tenure", "TotalCharges" e "MonthlyCharges":



Os picos de densidade são como esperado e representam o cliente comum.

4.6. Método dos K-Vizinhos mais Próximos

O Método dos K-Vizinho mais Próximos classifica o valor de uma instância de teste com base nos valores das K instâncias de treino mais próximas no espaço de características. Aplicamos este método às variáveis contínuas e aqui está um exemplo de uma relação que obtivemos:

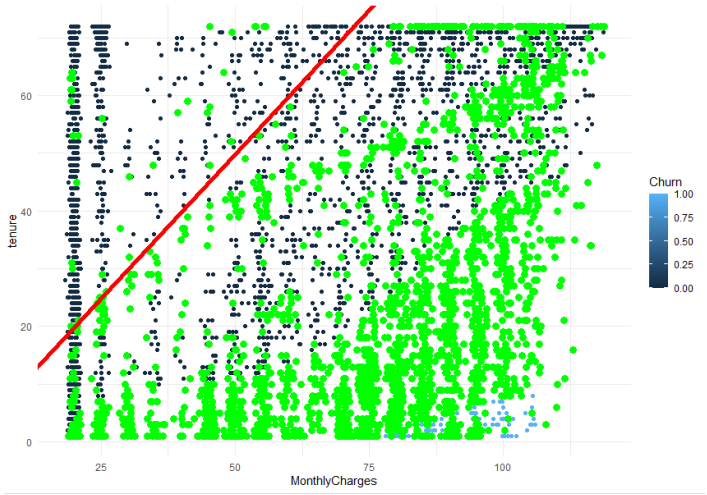


A precisão do modelo que obtivemos foi 76%. Os círculos são do *dataset* de treino, as cruzes são as de teste. A preto temos as observações que deram

“Churn" e a vermelho as que não deram “Churn". Ao analisar o gráfico observamos que as observações que deram “Churn” estão concentradas entre si, e o mesmo com as que não deram, tal como pretendido, justificando a precisão do modelo.

4.7. SVM

No SVM, o *dataset*, é usado para separar os dados em diferentes categorias. Encontrando um hiper-plano de separação que maximiza a margem entre as variáveis. O modelo SVM é treinado e as previsões são então feitas nos dados de teste. Aplicamos o método a todas as variáveis contínuas, aqui está o resultado da “Tenure" e “MonthlyCharges”:

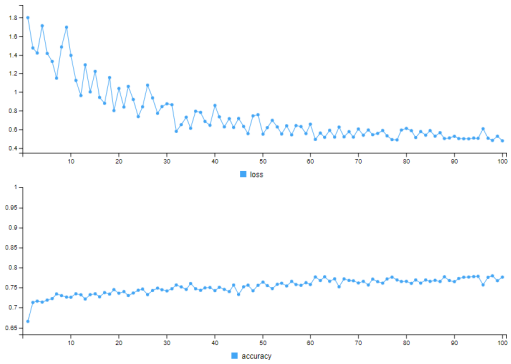


O hiper-plano está a tentar separar as observações que deram “Churn” das que não deram, baseado no “Tenure” e “MonthlyCharges”, neste caso. A verde temos os vetores de suporte que deveriam acompanhar o hiper-plano. Como as variáveis estão muito correlacionadas, não conseguimos aplicar o algoritmo com sucesso.

4.8. Redes Neuronais

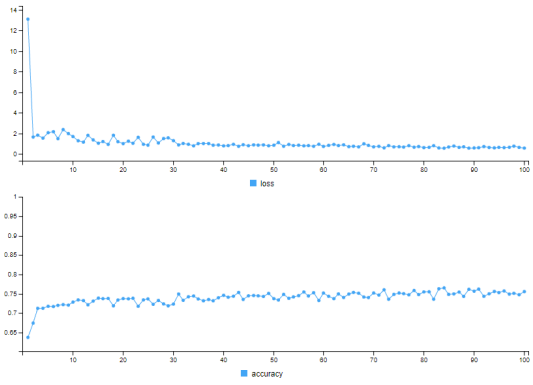
Redes Neuronais possuem camadas de neurónios organizadas de uma maneira específica. A camada de entrada recebe os dados , que são propagados através de camadas ocultas, onde ocorre o processamento e extração de características

relevantes, até a camada de saída, onde são geradas as previsões. Utilizamos uma rede com 3 camadas: a primeira tem 64 neurónios e a função de ativação *ReLu*, a segunda tem 32 neurónios e a função de ativação *ReLu* e a terceira tem apenas 1 neurónio e a função de ativação *sigmoid*. Obtendo o seguinte resultado:




```
> metrics <- model %>% evaluate(x_test, y_test)
66/66 [==============================] - 0s 909us/step - loss: 0.4230 - accuracy: 0.8033
> print(metrics)
     loss  accuracy
0.4230382 0.8033175
```

Inicialmente corremos com 10 *epochs*, mas o gráfico oscilava muito e por isso decidimos utilizar 100 *epochs*. Aplicamos ao modelo anterior *weight decay,* para regularizar as camadas mais densas e evitar *overfitting*. Obtendo o seguinte resultado:




```
> metrics <- model %>% evaluate(x_test, y_test)
66/66 [==============================] - 0s 939us/step - loss: 0.5089 - accuracy: 0.7668
> print(metrics)
     loss  accuracy
0.5089459 0.7668247
```

Ao aplicar *weight decay*, o modelo menos preciso, mas estabilizou mais rapidamente como se pode ver pelo gráfico.

5. Discussão e Conclusão

Neste projeto, realizamos uma análise dos atributos do conjunto de dados "Telco Customer Churn" para entender a sua influência nos clientes que mudaram de operadora, utilizando diversos métodos estatísticos de *data mining* relacionados com aprendizagem supervisionada classificatória. Concluímos, que:

- O Naive Bayes é um bom modelo para prever a nossa variável objetivo, demonstrando uma taxa de erro de 23%.
- As variáveis de "Churn" não estão bem separadas, dificultando a distinção entre elas.
- Os coeficientes das variáveis indicaram as suas contribuições para a probabilidade de "Churn", algumas delas mostraram-se estatisticamente significativas.
- A precisão da Árvore de Decisão foi de 79.5%, indicando uma boa capacidade de classificação.
- No modelo do K-Vizinhos mais Próximos, a precisão foi de 76%, com as observações de "Churn" e não "Churn" agrupadas de acordo como esperado.
- Como as variáveis contínuas estão muito correlacionadas, não conseguimos aplicar o algoritmo SVM com sucesso.
- A Rede Neural apresentou os melhores resultados com precisão de 80%, onde a adição de weight decay piorou o desempenho do modelo tendo precisão de 76%.

6. Bibliografia

[1] https://www.kaggle.com/datasets/blastchar/telco-customer-churn