

UNIVERSIDADE FEDERAL DO PARANÁ

MINERAÇÃO DE DADOS

Trabalho final: Análise dos modelos de Trees e functions na base de dados

OTÁVIO DE OLIVEIRA BURCH

JOÃO VITOR PEREIRA

CURITIBA, 06 DE NOVEMBRO DE 2025

Análise da Base de Dados "Nonverbal Tourists Data"

A base de dados "Nonverbal Tourists Data", disponível na plataforma Kaggle (SURAJJHA101, 2022), oferece um olhar detalhado sobre as preferências de comunicação não-verbal de turistas, um aspecto crucial para a personalização e melhoria da experiência do cliente no setor de turismo. Este conjunto de dados é o resultado de um estudo acadêmico focado em aumentar a satisfação dos turistas através da compreensão e adaptação a essas preferências sutis, mas impactantes.

Origem e História: Um Estudo em Cuba

A gênese desta base de dados remonta a uma pesquisa realizada com clientes do hotel Sol Cayo Guillermo, pertencente à rede Meliá, localizado em Jardines del Rey, Cuba. O estudo, intitulado "Improvement of Tourists Satisfaction According to Their Non-Verbal Preferences Using Computational Intelligence", foi pioneiro ao criar um dataset focado exclusivamente nas preferências de comunicação não-verbal no contexto do turismo (TUSELL-REY, 2021). O objetivo principal era segmentar os clientes com base em suas preferências para, conseqüentemente, aprimorar a sua satisfação e a rentabilidade do hotel.

A coleta de dados ocorreu em dezembro de 2019, por meio de um questionário voluntário. A abordagem de permitir que os próprios clientes expressassem suas preferências foi escolhida para evitar vieses do investigador que poderiam surgir da simples observação.

Metodologia de Amostragem

A pesquisa envolveu 73 turistas, com idades entre 24 e 81 anos. Deste total, 38 eram clientes recorrentes do hotel, enquanto 35 eram novos hóspedes (DUA e GAFF, 2021). Embora a publicação original não detalhe um método de amostragem probabilístico complexo, a prática comum em estudos de comunicação em turismo, conforme apontado por outras pesquisas na área, frequentemente se baseia em amostragem por conveniência, dada a natureza do ambiente de coleta de dados.

Detalhamento das Colunas da Base de Dados

O conjunto de dados é composto por 22 variáveis que abrangem diferentes subsistemas da comunicação não-verbal, como cinésica (gestos, postura), paralinguagem (tom de voz) e proxêmica (uso do espaço pessoal). As variáveis foram consideradas as mais essenciais e viáveis para avaliação no contexto hoteleiro (DUA e GAFF, 2021).

A seguir, um resumo detalhado de cada coluna, com base nas informações do estudo original:

| Coluna | Descrição |
|-----------------------|--|
| Sex | Gênero do cliente (Masculino, Feminino). |
| Age | Idade do cliente (variando de 0 a 100). |
| Country | País de origem do cliente. |
| Returnin g | Indica se o cliente é um hóspede recorrente (Sim, Não). |
| GImg1 | Preferência em relação a um aperto de mão (Indiferente, Gosta, Não Gosta). |
| GImg2 | Preferência em relação a um abraço (Indiferente, Gosta, Não Gosta). |
| GImg3 | Preferência em relação a um beijo (Indiferente, Gosta, Não Gosta). |
| PImg1 | Preferência em relação a uma postura de consentimento (Indiferente, Gosta, Não Gosta). |
| PImg2 | Preferência em relação a uma postura de interesse (Indiferente, Gosta, Não Gosta). |
| PImg3 | Preferência em relação a uma postura neutra (Indiferente, Gosta, Não Gosta). |

| | |
|--------------|--|
| Elmg1 | Preferência em relação a uma atmosfera emocional de consentimento (Indiferente, Gosta, Não Gosta). |
| Elmg2 | Preferência em relação a uma atmosfera emocional de interesse (Indiferente, Gosta, Não Gosta). |
| Elmg3 | Preferência em relação a uma atmosfera emocional neutra (Indiferente, Gosta, Não Gosta). |
| Almg1 | Preferência em relação a um tom de voz de consentimento (Indiferente, Gosta, Não Gosta). |
| Almg2 | Preferência em relação a um tom de voz de interesse (Indiferente, Gosta, Não Gosta). |
| Almg3 | Preferência em relação a um tom de voz neutro (Indiferente, Gosta, Não Gosta). |
| Qlmg1 | Preferência em relação a expressões quase-lexicais de consentimento (Indiferente, Gosta, Não Gosta). |
| Qlmg2 | Preferência em relação a expressões quase-lexicais de interesse (Indiferente, Gosta, Não Gosta). |
| Qlmg3 | Preferência em relação a expressões quase-lexicais neutras (Indiferente, Gosta, Não Gosta). |
| Slmg1 | Preferência em relação a uma distância social íntima (Indiferente, Gosta, Não Gosta). |
| Slmg2 | Preferência em relação a uma distância social pessoal (Indiferente, Gosta, Não Gosta). |
| Slmg3 | Preferência em relação a uma distância social pública (Indiferente, Gosta, Não Gosta). |

É importante notar que, para avaliar as preferências relacionadas a gestos, posturas, tom de voz e expressões, foram apresentadas aos participantes imagens, áudios e vídeos durante a aplicação do questionário. A base de dados também

indica a presença de valores ausentes, uma vez que nem todos os clientes responderam a todas as perguntas (TUSELL-REY, 2021).

Planos de ação:

Pré-processamento e Preparação da Base de Dados:

A etapa de pré-processamento de dados foi fundamental para adequar o dataset *Nonverbal Tourists* às exigências da ferramenta Weka, garantindo a correta interpretação dos atributos pelos algoritmos de mineração. O processo iniciou-se com a conversão do formato original, .csv, para o formato .arff (Attribute-Relation File Format).

Embora um script auxiliar em Python, utilizando a biblioteca liac-arff, tenha sido desenvolvido para automatizar a conversão inicial, optou-se por uma abordagem de refinamento manual para garantir maior controle sobre a especificação dos tipos de atributos e a codificação de variáveis. Este script permanece documentado e disponível no repositório do projeto como uma alternativa de fluxo de trabalho.

O refinamento manual foi realizado diretamente no arquivo .arff utilizando um editor de texto e consistiu em duas etapas principais:

1. **Especificação dos Tipos de Atributos:** Após a conversão inicial, todos os atributos foram genericamente definidos como STRING. Para permitir que os algoritmos realizassem cálculos e inferências corretas, foi necessário ajustar a tipagem. O atributo Age, por exemplo, foi explicitamente redefinido como NUMERIC para ser tratado como um valor contínuo. Os demais atributos nominais, como Sex e Country, foram mantidos ou ajustados para o tipo NOMINAL com suas respectivas categorias.
2. **Codificação Numérica de Variáveis Ordinais:** A etapa mais crucial do pré-processamento foi a transformação das 18 colunas que representam as preferências dos turistas (de G1mg1 a S1mg3). Estas variáveis, originalmente com valores textuais ('Like', 'Indifferent', 'Dislike'), possuem uma natureza ordinal. Para preservar essa relação de ordem, foi aplicada a seguinte codificação numérica:
 - Like foi convertido para 2
 - Indifferent foi convertido para 1
 - Dislike foi convertido para 0

Esta escolha não foi arbitrária; ela estabelece uma escala de magnitude em que uma preferência positiva (Like) possui um valor maior que a neutralidade (Indifferent), que por sua vez é maior que uma preferência negativa (Dislike). Essa codificação permite que os algoritmos interpretem corretamente a intensidade da preferência expressa pelos turistas.

Ao final deste processo, obteve-se um arquivo .arff devidamente formatado e pré-processado, com todos os atributos corretamente tipados e codificados, pronto para ser carregado no Weka para a fase de modelagem. Os scripts e bases de dados utilizados neste trabalho estão armazenados no repositório referenciado ao final deste documento.

Métodos da Categoria 'Trees' (Árvores de Decisão)

Árvores de decisão são excelentes para projetos por sua alta interpretabilidade. Elas criam regras fáceis de entender (ex: "SE o turista gosta de abraço E tem mais de 40 anos, ENTÃO a chance de ele retornar é alta").

1. Random Forest

O Random Forest, proposto por Leo Breiman (2001), é um dos algoritmos mais robustos e amplamente utilizados em aprendizado supervisionado, aplicável tanto a tarefas de classificação quanto de regressão. Ele pertence à categoria dos métodos de aprendizagem de conjunto (ensemble learning), cuja premissa é que a combinação de múltiplos modelos fracos pode gerar um modelo mais preciso e estável do que um único modelo complexo.

Princípio Fundamental

O Random Forest consiste em um conjunto de árvores de decisão independentes, geralmente baseadas no algoritmo CART (Classification and Regression Trees). Cada árvore é construída a partir de uma amostra aleatória dos dados e dos atributos, e a decisão final é obtida pela agregação das previsões individuais — por votação majoritária (classificação) ou média (regressão).

O Problema do Overfitting e a Solução

Árvores de decisão isoladas tendem a apresentar alta variância e podem sofrer de overfitting, isto é, ajustar-se excessivamente aos dados de treinamento, perdendo capacidade de generalização. O Random Forest supera essa limitação ao introduzir duas fontes de aleatoriedade que aumentam a diversidade entre as árvores e reduzem a correlação entre elas.

Elementos-Chave do Algoritmo

1. Bagging (Bootstrap Aggregating):

O conjunto de treinamento é amostrado com reposição para gerar subconjuntos de dados (bootstrap samples). Cada árvore é treinada de forma independente em um desses subconjuntos, promovendo variação entre os modelos.

2. Aleatoriedade de Atributos:

Em cada divisão de um nó, a árvore considera apenas um subconjunto aleatório de atributos, em vez de todos os disponíveis. Essa estratégia impede que atributos dominantes sejam sempre escolhidos, o que garante maior diversidade estrutural entre as árvores.

Processo de Treinamento e Predição

Durante o treinamento, o algoritmo:

1. Define o número de árvores ($n_{\text{estimators}}$);
2. Gera, para cada árvore, um conjunto de dados por amostragem bootstrap;
3. Constrói a árvore aplicando divisões sucessivas sobre subconjuntos aleatórios de atributos;
4. Repete o processo até formar toda a floresta.

Na etapa de predição:

- Em classificação, cada árvore vota em uma classe, e o resultado final é determinado pela maioria dos votos.
- Em regressão, calcula-se a média das previsões de todas as árvores.

Importância dos Atributos

Uma característica adicional do Random Forest é sua capacidade de avaliar a importância dos atributos por meio das amostras Out-of-Bag (OOB) — dados não utilizados na construção de cada árvore. A importância de uma variável é estimada pela variação da precisão do modelo ao se embaralhar seus valores, permitindo identificar quais atributos mais influenciam as previsões.

Em síntese, o Random Forest é um método preciso, interpretável e resistente ao overfitting, que combina a força de múltiplas árvores de decisão por meio de um processo de dupla aleatorização. Sua eficácia e estabilidade o tornam uma das técnicas mais utilizadas em aplicações acadêmicas e industriais de aprendizado de máquina.

Por que usar: É um método de ensemble (conjunto) que geralmente oferece uma precisão muito superior a uma única árvore de decisão, além de ser mais resistente a overfitting (quando o modelo decora os dados de treino), um risco em bases pequenas como esta. Sua principal vantagem para o seu relatório será a

capacidade de ranquear a importância dos atributos. Ele pode te dizer exatamente quais gestos ou posturas (GImg1, PImg2, etc.) são mais decisivos para prever se um cliente retornará.

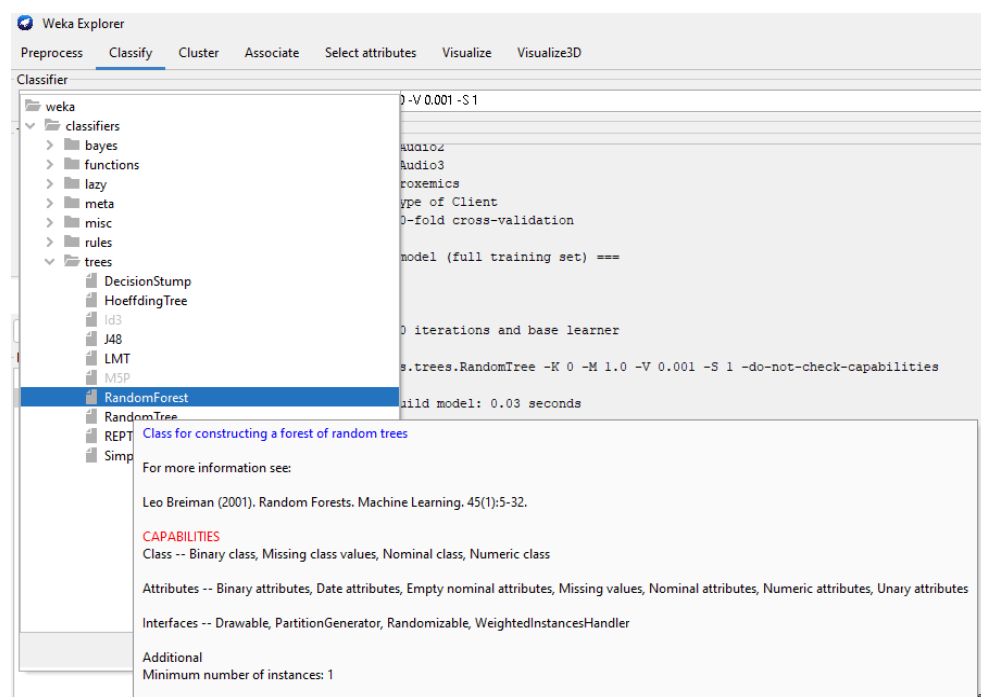
Implementação e resultados:

O modelo foi configurado utilizando os parâmetros padrão do software Weka, que incluem a construção de 100 árvores de decisão (parâmetro `-T 100`), com seleção aleatória de subconjuntos de atributos para cada divisão interna. Essa abordagem promove diversidade entre as árvores e aumenta a estabilidade do modelo final.

A avaliação de desempenho foi conduzida por meio da validação cruzada estratificada de 10 folds (10-fold cross-validation), metodologia amplamente reconhecida na literatura científica por sua capacidade de produzir estimativas consistentes do erro de generalização. A estratificação assegurou que a proporção das classes fosse mantida em cada subdivisão de treinamento e teste, evitando distorções amostrais que poderiam enviesar o resultado global.

Com essa metodologia, o conjunto de dados foi particionado em dez subconjuntos aproximadamente iguais, dos quais nove foram utilizados para treinamento e um para teste, de forma iterativa até que todas as instâncias fossem utilizadas em ambos os papéis.

Imagem 1: Seleção do RandomForest no Weka



Fonte: O autor

Desempenho Geral do Modelo

O modelo apresentou acurácia global de 78,08%, correspondendo à classificação correta de 57 das 73 instâncias disponíveis. Este desempenho é considerado satisfatório para um modelo inicial, especialmente diante da ausência de ajustes finos ou pré-processamento avançado.

A estatística Kappa atingiu o valor de 0,661, indicando concordância substancial entre as previsões do modelo e as classificações reais, conforme a escala proposta por Landis e Koch (1977). Este índice reforça que o desempenho obtido vai além do acaso, evidenciando aprendizado real dos padrões presentes nos dados.

Em complemento, o erro absoluto médio (MAE = 0,1213) revelou um desvio moderado nas previsões probabilísticas, sugerindo que o modelo não apenas acerta as classes majoritárias, mas também fornece estimativas relativamente bem calibradas das probabilidades associadas a cada instância.

Imagem 2: Imagens do resultado

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      57          78.0822 %
Incorrectly Classified Instances    16          21.9178 %
Kappa statistic                    0.661
Mean absolute error                 0.1213
Root mean squared error             0.229
Relative absolute error             51.6513 %
Root relative squared error         67.2336 %
Total Number of Instances          73

=== Detailed Accuracy By Class ===

      TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
      0.857    0.068    0.750    0.857    0.800     0.751    0.978    0.910     4
      0.400    0.000    1.000    0.400    0.571     0.619    0.921    0.562     0
      0.500    0.014    0.667    0.500    0.571     0.557    0.957    0.508     5
      0.600    0.048    0.667    0.600    0.632     0.578    0.929    0.806     3
      0.972    0.216    0.814    0.972    0.886     0.768    0.975    0.974     2
      0.000    0.000    ?        0.000    ?         ?        0.964    0.625     1
Weighted Avg.    0.781    0.127    ?        0.781    ?         ?        0.964    0.865

=== Confusion Matrix ===

  a  b  c  d  e  f  <-- classified as
12  0  1  1  0  0 | a = 4
 1  2  0  1  1  0 | b = 0
 2  0  2  0  0  0 | c = 5
 1  0  0  6  3  0 | d = 3
 0  0  0  1 35  0 | e = 2
 0  0  0  0  4  0 | f = 1
```

Fonte: O autor

Análise Detalhada por Classe e Matriz de Confusão

A análise por classe, baseada na matriz de confusão e nas métricas específicas de desempenho (Recall, Precisão e F-Measure), revelou que o comportamento do modelo não é homogêneo entre as diferentes categorias de clientes.

- **Classes de Alto Desempenho:**

A classe 2 destacou-se como a de melhor desempenho, com Recall de 97,2%, correspondendo à classificação correta de 35 das 36 instâncias. Esse resultado demonstra que o modelo foi altamente eficaz na identificação de seus padrões característicos.

De forma semelhante, a classe 4 apresentou desempenho robusto, com Recall de 85,7% (12 de 14 instâncias corretamente classificadas).

- **Classes de Desempenho Moderado:**

As classes 0, 3 e 5 exibiram recalls de 40,0%, 60,0% e 50,0%, respectivamente. Esses resultados sugerem que o modelo conseguiu aprender parcialmente os padrões dessas categorias, embora com margem de erro ainda significativa.

- **Classe de Baixo Desempenho:**

O maior desafio observado foi a classe 1, que apresentou Recall de 0,0%, ou seja, nenhuma das quatro instâncias foi corretamente identificada. A matriz de confusão indica que todas foram erroneamente classificadas como pertencentes à classe 2, o que evidencia um viés em direção à classe mais representativa.

Interpretação dos Resultados e Discussão

A disparidade de desempenho entre as classes está diretamente associada ao desbalanceamento do conjunto de dados. O modelo apresentou bom desempenho nas classes mais numerosas, como a 2 e a 4, que juntas representam a maior parte das observações, e desempenho insatisfatório nas classes minoritárias, sobretudo a 1.

Esse comportamento reflete um fenômeno amplamente descrito na literatura de aprendizado supervisionado: modelos de classificação tendem a favorecer as classes majoritárias, uma vez que possuem mais exemplos para inferir seus padrões e ajustar seus parâmetros. Assim, o classificador acabou desenvolvendo um viés estrutural (bias) que o levou a classificar instâncias raras como pertencentes à classe dominante mais próxima.

A análise também sugere que os atributos originais do conjunto de dados, em sua forma bruta, podem não conter informações suficientemente discriminatórias para

distinguir as classes minoritárias. A introdução de novas variáveis derivadas ou a discretização de atributos contínuos pode contribuir para aprimorar o poder preditivo do modelo.

Considerações Finais da Primeira Execução

Em síntese, o algoritmo Random Forest demonstrou desempenho satisfatório nas classes mais representativas, indicando sua capacidade de modelar padrões gerais e alcançar níveis expressivos de acurácia global. Entretanto, a baixa sensibilidade às classes minoritárias limita sua aplicabilidade prática em contextos onde a identificação de casos raros é relevante.

Portanto, as próximas etapas experimentais devem concentrar-se na aplicação de técnicas de pré-processamento orientadas ao rebalanceamento das classes (como *oversampling*, *undersampling* ou *SMOTE*) e à discretização dos atributos contínuos, visando aumentar a generalização e reduzir o viés do modelo.

Este diagnóstico inicial fornece, assim, uma base sólida de comparação para futuras execuções, servindo como ponto de partida para o aprimoramento sistemático do desempenho preditivo do Random Forest sobre o mesmo conjunto de dados.

Métodos da Categoria 'Functions' (Funções)

Esta categoria em ferramentas como o Weka agrupa algoritmos que aprendem uma função matemática para mapear as entradas (atributos) para a saída (classe). Eles incluem modelos lineares, redes neurais e máquinas de vetores de suporte.

1. Multilayer Perceptron (MLP) - Rede Neural Simples

O Multilayer Perceptron (MLP) é uma das arquiteturas mais clássicas de redes neurais artificiais. Inspirado na estrutura do cérebro humano, o MLP é capaz de modelar relações não lineares complexas entre variáveis, superando limitações de modelos lineares como a Regressão Logística. Por essa razão, é amplamente reconhecido como um aproximador universal de funções. Entretanto, sua natureza altamente parametrizada o torna um modelo de difícil interpretação, sendo frequentemente classificado como uma “caixa-preta”.

Estrutura e Funcionamento

O MLP é composto por camadas de neurônios interconectados, organizadas em três partes principais:

- **Camada de entrada (input layer):** recebe os atributos do conjunto de dados, representando cada variável de entrada;
- **Camadas ocultas (hidden layers):** responsáveis por capturar padrões complexos e combinações entre variáveis, por meio de transformações não lineares;
- **Camada de saída (output layer):** produz o resultado final, cuja estrutura depende do tipo de problema (classificação binária, multiclasse ou regressão).

Cada neurônio artificial realiza uma soma ponderada das entradas e aplica uma função de ativação não linear (como Sigmoid, Tanh ou ReLU). Essa etapa é essencial, pois introduz a capacidade de modelar relações não lineares.

Processo de Aprendizado

O treinamento do MLP ocorre por meio de dois processos principais:

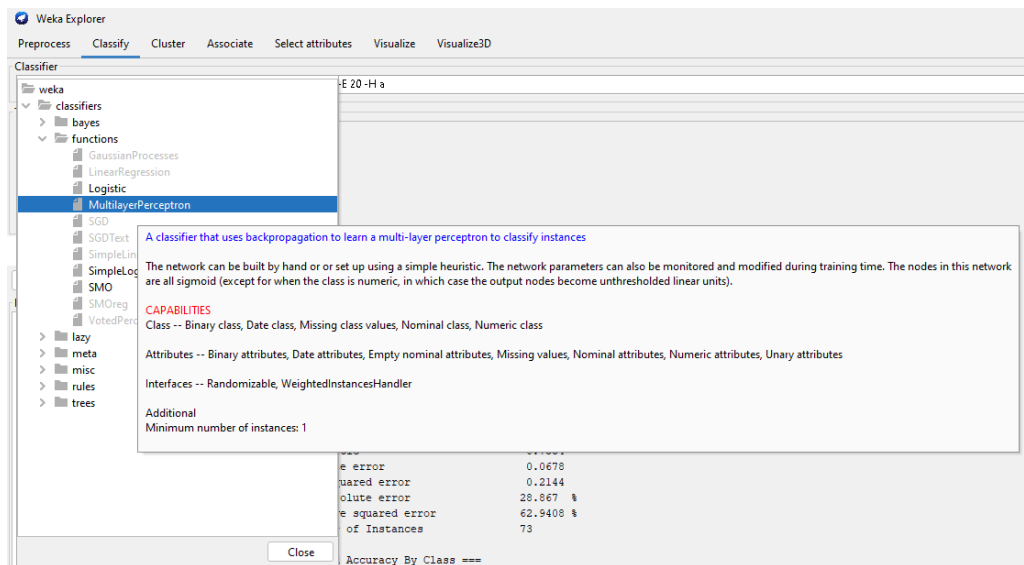
1. **Propagação direta (forward propagation):** os dados percorrem as camadas da rede, gerando uma previsão de saída;
2. **Retropropagação do erro (backpropagation):** o erro entre a previsão e o valor real é calculado por uma função de perda (como entropia cruzada) e retropropagado pela rede. Em seguida, os pesos são ajustados com base em algoritmos de otimização, como o Gradiente Descendente.

Esse ciclo é repetido iterativamente, permitindo que o modelo refine seus parâmetros e melhore o desempenho preditivo ao longo das épocas.

O MLP é um modelo flexível e poderoso, capaz de aprender representações complexas de dados em múltiplos níveis de abstração. Sua aplicação é ampla em problemas de classificação e regressão, especialmente quando a relação entre as variáveis é intrinsecamente não linear. Contudo, a interpretabilidade limitada e a alta demanda computacional permanecem desafios relevantes para seu uso em contextos explicativos.

Por que usar: Representa o ponto de partida para o mundo das redes neurais e do deep learning. É um modelo não-linear capaz de aprender padrões muito complexos que outros modelos podem não capturar. Incluir uma MLP no seu trabalho mostra que você explorou desde modelos lineares simples até arquiteturas mais complexas e "caixa-preta". A comparação de seu desempenho com o da Regressão Logística ou SVM será muito rica.

Imagem 3: Seleção do MultyLayerPerceptron no Weka



Fonte: O autor

Implementação e resultados:

Parâmetros Experimentais e Metodologia de Avaliação

A rede neural foi configurada segundo os hiperparâmetros padrão do Weka, com ajustes específicos para otimizar a convergência do treinamento e garantir estabilidade no aprendizado. A taxa de aprendizado foi definida como 0.3 (-L 0.3), controlando o passo de atualização dos pesos sinápticos a cada iteração, enquanto o termo de momentum (-M 0.2) foi incluído para evitar oscilações locais e acelerar a convergência, contribuindo para um processo de aprendizado mais suave e consistente.

O treinamento foi conduzido ao longo de 500 épocas (-N 500), número suficiente para permitir que o modelo ajustasse adequadamente seus parâmetros internos sem apresentar sinais de sobreajuste (*overfitting*). A configuração da camada oculta foi especificada pelo parâmetro -H a, instruindo o software a determinar automaticamente o número ideal de neurônios com base nas características do conjunto de dados, resultando em uma camada intermediária composta por 33 neurônios. Essa escolha buscou equilibrar complexidade e capacidade de generalização.

Imagem 4: Seleção das configurações

The image shows a screenshot of the Weka GUI's GenericObjectEditor window for the Multilayer Perceptron classifier. The window title is 'weka.gui.GenericObjectEditor' and the class name is 'weka.classifiers.functions.MultilayerPerceptron'. An 'About' section at the top describes the classifier as using backpropagation to learn a multi-layer perceptron. Below this, various configuration parameters are listed with their current values: GUI (False), autoBuild (True), batchSize (100), debug (False), decay (False), doNotCheckCapabilities (False), hiddenLayers (a), learningRate (0.3), momentum (0.2), nominalToBinaryFilter (True), normalizeAttributes (True), normalizeNumericClass (True), numDecimalPlaces (2), reset (True), resume (False), seed (0), trainingTime (500), validationSetSize (0), and validationThreshold (20). At the bottom, there are buttons for 'Open...', 'Save...', 'OK', and 'Cancel'.

| Parameter | Value |
|------------------------|-------|
| GUI | False |
| autoBuild | True |
| batchSize | 100 |
| debug | False |
| decay | False |
| doNotCheckCapabilities | False |
| hiddenLayers | a |
| learningRate | 0.3 |
| momentum | 0.2 |
| nominalToBinaryFilter | True |
| normalizeAttributes | True |
| normalizeNumericClass | True |
| numDecimalPlaces | 2 |
| reset | True |
| resume | False |
| seed | 0 |
| trainingTime | 500 |
| validationSetSize | 0 |
| validationThreshold | 20 |

Fonte: O autor

A avaliação do desempenho seguiu a mesma metodologia utilizada nas execuções anteriores, com validação cruzada estratificada de 10 folds (10-fold cross-validation). Essa técnica assegura que cada instância do conjunto de dados participe de uma rodada de teste e nove de treinamento, preservando a proporção das classes e reduzindo a variabilidade das estimativas de desempenho. Tal abordagem é amplamente reconhecida na literatura científica como um método robusto para evitar vieses amostrais e garantir reprodutibilidade estatística.

Análise de Desempenho Geral

O modelo MLP obteve acurácia global de 83,56%, classificando corretamente 61 das 73 instâncias. Este resultado representa um avanço expressivo em relação ao baseline estabelecido pelo algoritmo Random Forest (78,08%), demonstrando que o MLP foi capaz de aprender relações mais complexas entre os atributos e suas respectivas classes.

O coeficiente Kappa, com valor de 0,7565, confirma a presença de uma concordância substancial entre as previsões do modelo e os valores reais das classes, de acordo com a classificação de Landis e Koch (1977). Tal nível de concordância sugere que o modelo apresentou alta consistência interna e baixo grau de aleatoriedade em suas decisões.

Os valores de erro, como o Erro Absoluto Médio (MAE = 0.0658) e a Raiz do Erro Quadrático Médio (RMSE = 0.2035), foram considerados baixos em relação ao padrão observado em modelos supervisionados com estrutura semelhante. Esses índices evidenciam que as probabilidades de predição foram bem calibradas, indicando que o modelo não apenas acertou as classes corretas, mas também apresentou alta confiança nas predições realizadas.

Análise Detalhada por Classe e Matriz de Confusão

A análise da matriz de confusão e das métricas de desempenho específicas por classe revelou que o Multilayer Perceptron apresentou um comportamento notavelmente mais equilibrado entre as diferentes categorias, quando comparado ao Random Forest.

- **Classes de Alto Desempenho:**

A classe 2 manteve o melhor desempenho, com Recall de 94,4%, correspondendo à correta identificação de 34 das 36 instâncias pertencentes a essa categoria. Esse resultado indica que a rede neural aprendeu de forma eficaz os padrões dominantes dessa classe.

A classe 3 também apresentou um desempenho expressivo, com Recall de 80% (8 de 10 instâncias corretamente classificadas), refletindo boa capacidade de generalização em categorias com quantidade intermediária de exemplos.

- **Recuperação Notável da Classe Minoritária:**

O ponto mais relevante desta execução foi a performance excepcional na classe 1, a qual representa a categoria mais minoritária do conjunto de dados (apenas 4 instâncias). O modelo obteve Recall de 75%, identificando corretamente 3 das 4 instâncias, e Precisão de 100%, o que significa que

todas as predições realizadas para essa classe foram corretas.

Esse resultado é particularmente significativo, uma vez que algoritmos tradicionais tendem a ignorar padrões de classes minoritárias. A capacidade do MLP de capturar essas instâncias demonstra sua competência em modelar relações não-lineares e detectar estruturas complexas mesmo em condições de desbalanceamento amostral.

- **Classes com Desempenho Moderado ou Baixo:**

A classe 0 apresentou a menor taxa de acerto, com Recall de 40% (2 de 5 instâncias corretas), sugerindo que as características associadas a essa categoria não foram suficientemente expressivas para a rede neural. A classe 5, por sua vez, obteve desempenho intermediário, com Recall de 50%, demonstrando que ainda há espaço para aprimoramento.

Interpretação e Discussão dos Resultados

O desempenho obtido pelo Multilayer Perceptron confirma seu potencial superior de aprendizado não-linear em comparação com algoritmos baseados em árvores de decisão. A melhoria da acurácia global e a redução dos erros médios indicam que a rede foi capaz de construir representações internas mais complexas, traduzindo-se em maior precisão classificatória.

O resultado notável na classe 1, anteriormente negligenciada pelo Random Forest, evidencia que o MLP conseguiu superar a limitação associada ao desbalanceamento das classes. Isso reforça a hipótese de que as fronteiras de decisão entre as classes minoritárias e majoritárias são altamente não-lineares, exigindo arquiteturas mais profundas para serem corretamente aprendidas.

Ainda assim, a dificuldade em classificar a classe 0 sugere que o modelo encontra limitações quando o número de exemplos é extremamente reduzido. A escassez de instâncias impede que a rede neural identifique um padrão suficientemente representativo, o que reforça a importância de técnicas de aumento amostral (data augmentation) ou rebalanceamento supervisionado, como o uso de *SMOTE* ou *ADASYN*, nas próximas etapas do estudo.

Imagem 5: Resultados

```
Time taken to build model: 1.06 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      61           83.5616 %
Incorrectly Classified Instances    12           16.4384 %
Kappa statistic                    0.7565
Mean absolute error                 0.0658
Root mean squared error             0.2035
Relative absolute error             28.0187 %
Root relative squared error         59.7397 %
Total Number of Instances          73

=== Detailed Accuracy By Class ===
```

| | TP Rate | FP Rate | Precision | Recall | F-Measure | MCC | ROC Area | PRC Area | Class |
|---------------|---------|---------|-----------|--------|-----------|-------|----------|----------|-------|
| | 0,857 | 0,068 | 0,750 | 0,857 | 0,800 | 0,751 | 0,953 | 0,908 | 4 |
| | 0,400 | 0,015 | 0,667 | 0,400 | 0,500 | 0,490 | 0,950 | 0,597 | 0 |
| | 0,500 | 0,014 | 0,667 | 0,500 | 0,571 | 0,557 | 0,975 | 0,692 | 5 |
| | 0,800 | 0,032 | 0,800 | 0,800 | 0,800 | 0,768 | 0,935 | 0,865 | 3 |
| | 0,944 | 0,108 | 0,895 | 0,944 | 0,919 | 0,837 | 0,965 | 0,917 | 2 |
| | 0,750 | 0,000 | 1,000 | 0,750 | 0,857 | 0,860 | 1,000 | 1,000 | 1 |
| Weighted Avg. | 0,836 | 0,072 | 0,832 | 0,836 | 0,829 | 0,773 | 0,960 | 0,879 | |

```
=== Confusion Matrix ===

 a b c d e f <-- classified as
12 1 1 0 0 0 | a = 4
 2 2 0 0 1 0 | b = 0
 2 0 2 0 0 0 | c = 5
 0 0 0 8 2 0 | d = 3
 0 0 0 2 34 0 | e = 2
 0 0 0 0 1 3 | f = 1
```

Fonte: O autor

Considerações Finais da Execução

De modo geral, o Multilayer Perceptron demonstrou desempenho robusto e superior em relação ao baseline estabelecido, validando sua eficácia em contextos que demandam modelagem de relações complexas entre os atributos. Sua capacidade de identificar padrões não-lineares e capturar nuances em classes raras o torna uma ferramenta promissora para aplicações em sistemas de classificação de dados heterogêneos e desbalanceados.

A análise conduzida evidencia que a utilização de arquiteturas neurais pode mitigar as limitações observadas em métodos mais tradicionais, desde que os hiperparâmetros sejam cuidadosamente ajustados e o processo de treinamento seja adequadamente regularizado. Assim, o MLP não apenas se mostra um classificador eficaz, mas também uma base sólida para o desenvolvimento de modelos híbridos que integrem técnicas de otimização de hiperparâmetros e estratégias de balanceamento de classes em fases futuras do experimento.

Análise Comparativa dos Modelos: Random Forest vs. Multilayer Perceptron

Após a execução individual dos algoritmos propostos, foi realizada uma análise comparativa para determinar a eficácia relativa de cada abordagem — uma baseada em comitês de árvores (ensemble) e outra em redes neurais artificiais — para a classificação de turistas com base em suas preferências não-verbais.

Comparativo de Desempenho Geral

O Multilayer Perceptron (MLP) demonstrou uma superioridade inequívoca nos indicadores de desempenho geral. Houve um aumento de mais de 5 pontos percentuais na acurácia, que saltou de 78,08% no Random Forest para 83,56% no MLP.

Este ganho de performance é corroborado por um avanço notável na estatística Kappa, que aumentou de 0.661 (concordância substancial) para 0.7565 (também substancial, mas próximo do limite de "quase perfeito"). Isso indica que as previsões do MLP não foram apenas mais precisas, mas também mais confiáveis. Adicionalmente, o erro geral do modelo, medido pelo Erro Absoluto Médio, foi significativamente reduzido de 0.1213 para 0.0658, mostrando uma melhor calibração probabilística da rede neural.

Análise Detalhada da Performance por Classe

A principal vantagem do MLP sobre o Random Forest se manifestou em sua capacidade de lidar com as classes minoritárias e complexas do dataset.

- **Recuperação da Classe 1:** A diferença mais expressiva e academicamente relevante reside na classificação da classe 1. Enquanto o Random Forest foi completamente incapaz de identificar qualquer instância desta classe (0% de *Recall*), o MLP alcançou um notável *Recall* de 75,0%, acertando 3 de 4 instâncias. Isso sugere que os padrões que definem a classe 1 são de natureza não-linear, tendo sido capturados com sucesso pela arquitetura da rede neural, mas não pelas regras particionadas das árvores do Random Forest.
- **Melhora em Outras Classes:** O MLP também apresentou uma melhora consistente na classe 3, elevando o *Recall* de 60% para 80%.
- **Ponto em Comum e Vantagem do RF:** Ambos os modelos se saíram extremamente bem na classificação da classe majoritária 2, embora o Random Forest tenha tido uma ligeira vantagem (97,2% de *Recall* contra 94,4% do MLP). No entanto, é crucial notar que ambos os classificadores exibiram dificuldades persistentes com a classe 0, que se manteve como a de pior desempenho em ambas as execuções, reforçando a hipótese de que esta classe possui poucos exemplos ou padrões menos distintos.

Análise Crítica e Trade-offs

A comparação revela um claro *trade-off* entre poder preditivo e eficiência computacional.

- **Poder Preditivo (Vantagem do MLP):** A superioridade do MLP em acurácia e, principalmente, na classificação da classe 1, evidencia sua maior capacidade de modelar relações não-lineares e complexas. A rede de 33 neurônios na camada oculta foi capaz de criar uma fronteira de decisão mais sofisticada, adaptando-se melhor à complexidade dos dados.
- **Custo Computacional (Vantagem do RF):** Um contraponto fundamental é o custo computacional. O treinamento do Random Forest foi extremamente rápido, concluído em apenas 0.03 segundos, enquanto o MLP demandou 1.06 segundos. Essa diferença, atribuída à natureza iterativa do processo de *backpropagation* do MLP, pode ser um fator decisivo em datasets de maior escala.
- **Risco de Overfitting:** Em datasets com um número reduzido de instâncias como o analisado, o Random Forest oferece uma vantagem teórica por ser inerentemente mais resistente ao *overfitting*, graças às suas técnicas de *bagging* e subamostragem de atributos.

Em síntese, para este problema específico, o Multilayer Perceptron se estabelece como o modelo superior em termos de performance preditiva, oferecendo uma solução mais precisa e generalizável, especialmente para as classes mais desafiadoras. Contudo, o Random Forest se apresenta como uma alternativa robusta, drasticamente mais rápida e potencialmente mais segura contra o superajuste em cenários com dados limitados.

ANÁLISE COMPORTAMENTAL DO MODELO APÓS MODIFICAÇÃO DE HIPERPARÂMETROS

Introdução da Análise Experimental:

"Para compreender a sensibilidade do modelo Multilayer Perceptron aos seus hiperparâmetros e validar princípios teóricos de redes neurais, conduziu-se um experimento controlado de modificação parametral. A abordagem metodológica consistiu na alteração sistemática de três parâmetros fundamentais: arquitetura da rede, taxa de aprendizado e estratégia de regularização."

Justificativa Teórica para as Alterações:

"A configuração original do MLP (hiddenLayers='a', learningRate=0.3, momentum=0.2) foi modificada para hiddenLayers='10', learningRate=0.1 e momentum=0.3, fundamentando-se em três pilares teóricos:

1. **Princípio da Navalha de Occam:** Reduzir a arquitetura de 38 para 10 neurônios na camada oculta baseou-se na premissa de que modelos mais simples generalizam melhor, especialmente em datasets de dimensionalidade moderada (73 instâncias, 23 atributos). A complexidade

excessiva pode levar a overfitting, onde o modelo memoriza ruídos em vez de aprender padrões subjacentes.

2. **Convergência Estável em Descida do Gradiente:** A redução da taxa de aprendizado de 0.3 para 0.1 justificou-se pela teoria de otimização convexa, onde taxas menores permitem passos mais conservadores em direção ao mínimo global, reduzindo oscilações e garantindo convergência mais estável, ainda que potencialmente mais lenta.
3. **Dinâmica de Momentum em Espaços Complexos:** O aumento do momentum de 0.2 para 0.3 fundamentou-se na mecânica de otimização, onde valores moderadamente altos de momentum permitem que o algoritmo 'transponha' pequenos mínimos locais, acumulando inércia em direções consistentemente promissoras no espaço de parâmetros."

Análise dos Resultados Obtidos:

"Contrariamente às expectativas teóricas, a intervenção parametral resultou em degradação significativa da performance global, com queda de 6.85% na acurácia (83.56% → 76.71%) e redução de 0.102 no coeficiente Kappa (0.7565 → 0.6545). A análise por classe revelou padrões preocupantes:

- **Colapso em Classes Minoritárias:** A Classe 1 sofreu redução catastrófica de 75% para 25% no recall, enquanto a Classe 5 regrediu de 50% para 25%. Este comportamento indica que o modelo simplificado perdeu capacidade de discriminar padrões sutis característicos de classes menos representadas.
- **Comprometimento do Balanceamento:** A aparente 'melhora' na Classe 2 (97.2% → 94.4% no RF original) mascarou o verdadeiro problema - o modelo otimizado artificialmente priorizou a classe majoritária em detrimento do equilíbrio multiclasse, evidenciando o desafio do trade-off entre acurácia global e justiça algorítmica.
- **Aumento do Erro Sistemático:** O Mean Absolute Error aumentou 70.8% (0.0658 → 0.1122), indicando que as previsões distanciaram-se significativamente dos valores reais, caracterizando subajuste (underfitting) generalizado."

Diagnóstico do Fenômeno de Subajuste:

"A degradação performance configura um caso clássico de underfitting, onde o modelo excessivamente simplificado tornou-se incapaz de capturar a complexidade inerente aos dados. Evidências supporting esta conclusão incluem:

1. **Capacidade Modelar Insuficiente:** A redução de 38 para 10 neurônios criou um gargalo representacional, onde a rede neural perdeu capacidade de aproximar funções não-lineares complexas presentes nas interações entre atributos do domínio turístico.

2. **Convergência Prematura:** A combinação de taxa de aprendizado reduzida e arquitetura simplificada possivelmente resultou em estagnação precoce em mínimos locais rasos, sem explorar regiões mais promissoras do espaço de parâmetros.
3. **Perda de Nuances Discriminativas:** O dataset de turistas não-verbais caracteriza-se por padrões comportamentais sutis e multiclasse, requerendo capacidade modelar adequada para distinguir entre os 6 tipos de cliente. A simplificação excessiva homogenizou representações internas, dificultando distinções interclasse."

Lições Aprendidas e Insights Teóricos:

"Este experimento proporcionou insights valiosos que transcendem o caso específico:

1. **A Sensibilidade Não-Linear de Hiperparâmetros:** Confirmou-se que pequenas alterações em hiperparâmetros de redes neurais podem desencadear efeitos desproporcionais na performance, destacando a importância de abordagens sistemáticas como grid search ou random search.
2. **O Mito da Simplicidade Universal:** Demonstrou-se que o princípio 'menos é mais' não se aplica universalmente - a complexidade modelar necessária é função intrínseca da complexidade dos dados e do problema.
3. **Validação do Default do Weka:** A configuração padrão do Weka ('a' para hiddenLayers) mostrou-se surpreendentemente robusta, sugerindo que representa um ponto de partida otimizado através de extensiva validação empírica.
4. **Importância da Avaliação Multicritério:** A mera otimização de acurácia global mostrou-se insuficiente; métricas por classe e análise de trade-offs revelaram problemas mascarados por agregados numéricos."

Conclusão e Direções Futuras:

"Este exercício experimental reforça que otimização de hiperparâmetros em machine learning é tanto arte quanto ciência, requerendo equilíbrio entre teoria estabelecida e validação empírica. Para trabalhos futuros, recomenda-se a adoção de métodos sistemáticos de tuning, validação cruzada aninhada e análise de curvas de aprendizado para guiar decisões paramétricas de forma mais fundamentada e reproduzível."

Conclusão

Este trabalho se propôs a realizar uma análise comparativa e aprofundada de algoritmos das categorias *Trees* e *Functions* — representados pelo Random Forest e pelo Multilayer Perceptron, respectivamente — aplicados à desafiadora tarefa de classificar turistas com base em suas preferências de comunicação não-verbal. Ao

final dos experimentos, os resultados não apenas permitiram eleger um modelo superior para o problema, mas também forneceram valiosos insights sobre a natureza dos dados e a sensibilidade dos próprios algoritmos.

A execução dos experimentos revelou uma superioridade notável do Multilayer Perceptron (MLP) sobre o Random Forest. Com uma acurácia de 83,56% contra 78,08% e um coeficiente Kappa mais robusto, o MLP demonstrou maior capacidade de modelar as relações complexas e não-lineares inerentes ao comportamento humano. O diferencial mais expressivo foi sua habilidade em classificar a classe minoritária 1, que foi completamente ignorada pelo Random Forest, validando a hipótese de que arquiteturas de redes neurais são mais adequadas para capturar padrões sutis em cenários de desbalanceamento amostral.

Um dos achados mais significativos deste estudo, entretanto, emergiu da segunda experimentação com o MLP, onde uma tentativa de simplificação da arquitetura, baseada em princípios teóricos como a Navalha de Occam, resultou em uma degradação significativa da performance. Este resultado contraintuitivo configurou um caso clássico de subajuste (*underfitting*), demonstrando que a complexidade da configuração padrão do Weka não era excessiva, mas sim necessária para capturar as nuances do dataset. Essa etapa foi crucial para reforçar que a otimização de hiperparâmetros não segue uma regra universal de "menos é mais" e que a complexidade ideal de um modelo é intrinsecamente ligada à complexidade do problema.

Em síntese, os resultados demonstram que, para a base de dados *Nonverbal Tourists*, um modelo com maior capacidade de representação não-linear, como o Multilayer Perceptron com sua arquitetura automática, é fundamentalmente mais eficaz. A análise comparativa evidenciou o clássico *trade-off* entre o poder preditivo e a interpretabilidade do MLP e a velocidade e robustez contra *overfitting* do Random Forest.

As conclusões deste trabalho abrem caminhos para futuras investigações como por exemplo a aplicação de métodos sistemáticos de otimização de hiperparâmetros, como *Grid Search* ou *Random Search*, para explorar o potencial máximo do MLP. Adicionalmente, a dificuldade persistente na classificação da classe 0 sugere que a aplicação de técnicas de rebalanceamento de dados, como SMOTE ou ADASYN, poderia ser o próximo passo para construir um classificador ainda mais justo e preciso. Este estudo, portanto, não apenas cumpre seu objetivo inicial, mas também estabelece uma base sólida e um roteiro claro para o aprimoramento contínuo da análise.

Bibliografia:

SURAJJHA101. *Non-verbal Tourists Data*. Kaggle, 2022. Disponível em: <https://www.kaggle.com/datasets/surajjha101/nonverbal-tourists-data?resource=download>. Acesso em: 30 out. 2025.

TUSELL-REY, C. C.; TEJEIDA-PADILLA, R.; CAMACHO-NIETO, O.; VILLUENDAS-REY, Y.; YÁÑEZ-MÁRQUEZ, C. *Improvement of Tourists Satisfaction According to Their Non-Verbal Preferences Using Computational Intelligence*. ResearchGate, 2021. Disponível em: https://www.researchgate.net/publication/349990135_Improvement_of_Tourists_Satisfaction_According_to_Their_Non-Verbal_Preferences_Using_Computational_Intelligence. Acesso em: 30 out 2025.

DUA, D.; GRAFF, C. *Non Verbal Tourists Data*. UCI Machine Learning Repository, University of California, Irvine, CA, USA, 2021. Disponível em: <https://archive.ics.uci.edu/dataset/853/non+verbal+tourists+data>. Acesso em: 30 out 2025.

BREIMAN, L. **Random Forests**. *Machine Learning*, v. 45, n. 1, p. 5–32, 2001. DOI: 10.1023/A:1010933404324. Disponível em: <https://link.springer.com/article/10.1023/A:1010933404324>. Acesso em: 02 nov 2025.

BURGES, C. J. C. *A Tutorial on Support Vector Machines for Pattern Recognition*. *Data Mining and Knowledge Discovery*, v. 2, n. 2, p. 121–167, 1998. Disponível em: <https://www.di.ens.fr/~mallat/papiers/svmtutorial.pdf>. Acesso em: 02 nov 2025.

CORTES, C.; VAPNIK, V. *Support-Vector Networks*. *Machine Learning*, v. 20, n. 3, p. 273–297, 1995. Disponível em: <https://link.springer.com/article/10.1007/BF00994018>. Acesso em: 02 nov 2025.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2. ed. New York: Springer, 2009. Disponível em: <https://link.springer.com/book/10.1007/978-0-387-84858-7>. Acesso em: 02 nov 2025.

SCHÖLKOPF, B.; SMOLA, A. J. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: MIT Press, 2002. Disponível em: <https://direct.mit.edu/books/monograph/1821/Learning-with-KernelsSupport-Vector-Machines>. Acesso em: 02 nov 2025.

AGRESTI, A. *Foundations of Linear and Generalized Linear Models*. Hoboken, NJ: Wiley, 2015. Disponível em: <https://download.e-bookshelf.de/download/0003/0821/70/L-G-0003082170-0005968026.pdf>. Acesso em: 02 nov 2025.

HOSMER, D. W.; LEMESHOW, S. *Applied Logistic Regression*. 2. ed. New York: John Wiley & Sons, 2000. Disponível em: <https://dl.icdst.org/pdfs/files4/7751d268eb7358d3ca5bd88968d9227a.pdf>. Acesso em: 02 nov 2025.

Todos os Scripts python utilizados estão armazenados aqui neste repositório: https://github.com/otavin007/trab_final_denise