EFC 02 - Classificação

Luís Antônio Almeida Lima Vieira (221045) | Nathan Shen Baldon (242448)

13 de outubro de 2022

1 Introdução

O objetivo desta atividade é utilizar dois modelos de classificação (Regressão Logística e k vizinhos mais próximos) para solucionar um problema de classificação binária de imagens de ultrassom mamário, provenientes da base de dados *breastmnist* (YANG et al., 2021). Esta é uma base de dados composta por 780 imagens em níveis de cinza com dimensão 28x28 pixels. A classificação foi feita em duas categorias: câncer (classe negativa, com rótulo 0) e normal (classe positiva, com rótulo 1).

2 Análise do conjunto de dados

Inicialmente, importou-se a base de dados *breastmnist* (YANG et al., 2021) e foram separados os dados e seus respectivos rótulos (com *.imgs* para as imagens e *.labels* para os rótulos). A separação utilizada para os conjuntos de treino, teste e validação, foi a separação padrão já feita na base de dados, com as 546 primeiras imagens e rótulos separadas para treinamento, as 78 imagens e rótulos seguintes separadas para validação e as últimas 156 imagens e rótulos separadas para teste. Na Figura 1 estão algumas amostras de imagens do conjunto de treinamento.

(a) (b)

Figura 1: Em (a), a primeira imagem do conjunto de treinamento, com rótulo 1 (normal) e, em (b), a quinta imagem do conjunto de treinamento, com rótulo 0 (câncer).

Foi feita uma análise simples do balanceamento das classes nos conjuntos disponíveis. Com isso, foi possível observar que no conjunto de treinamento há 147 classes negativas e 399 classes positivas, portanto aproximadamente 26,923% do conjunto de treinamento é composto por classes negativas (rótulo 0) e aproximadamente 73,077% é composto por classes positivas (rótulo 1). Para o conjunto de validação há 21 classes negativas e 57 classes positivas, portanto aproximadamente 26,923% do conjunto de validação é composto por classes negativas (rótulo 0) e aproximadamente 73,077% é composto por classes positivas (rótulo 1). Para o conjunto de teste há 42 classes negativas e 114 classes positivas, portanto aproximadamente 26,923% do

conjunto de teste é composto por classes negativas (rótulo 0) e aproximadamente 73,077% é composto por classes positivas (rótulo 1).

Com isso, pode-se concluir que os três conjuntos estão desbalanceados, contendo mais dados referentes à classe positiva (classe 1, normal) do que dados referentes à classe negativa (classe 0, câncer). Os três conjuntos foram feitos de forma a possuírem o mesmo desbalanceamento entre as classes positiva e negativa. Para que esse desbalanceamento não prejudique o modelo de classificação, é necessário atribuir pesos diferentes entre as classes, durante o treinamento, de forma a não ocorrer muitos resultados falsos na predição da classe minoritária.

3 Regressão Logística

Para regressão logística, utilizou-se a função *sklearn.linear_model.LogisticRegression()* da biblioteca *Scikit-learn*. Os parâmetros do modelo utilizados foram:

- *penalty*: escolhido através da validação cruzada.
- *C*: valor inverso à força da regularização (valores menores representam regularizações maiores).
- *class_weight*: tipo de pesos das classes, podendo ser *balanced* para pesar as classes pelo balanceamento ou *None* para atribuir pesos iguais às classes.
- *random_state*: para estabilizar resultados entre cada treinamento.
- *solver*: escolhido 'liblinear', pois a documentação recomenda essa opção para conjuntos de dados pequenos (SCIKIT-LEARN, s.d.).
- *max_iter*: valor escolhido de 3000, uma vez que um valor maior não implicava em melhoria significativa no desempenho do modelo.

Esses parâmetros foram escolhidos por serem parâmetros já estudados na matéria e por serem coerentes com a classificação abordada.

Para uso da função, ainda foi necessário ajustar o formato das imagens, de forma a ficarem no formato de 1 linha e 784 colunas cada.

3.1 Validação Cruzada

Para validação cruzada, foi utilizada a função *sklearn.model_selection.GridSearchCV()*, que testa exaustivamente os parâmetros de um modelo, a fim de escolher a combinação que maximize uma métrica de desempenho escolhida. Para essa validação, escolheu-se fazer a validação com 5 *folds* (parâmetro *cv*) e buscou-se maximizar a área sob a curva ROC (parâmetro *scoring*), uma vez que era o que melhorava o desempenho geral e diminuía o número de falso positivos. Fez-se essa escolha pensando no contexto do classificador, o qual detecta presença de câncer mamário. Nesse caso, acreditou-se que é pior receber um diagnóstico normal quando, na realidade, há presença de câncer do que o contrário.

Os parâmetros ajustados podiam ser escolhidos entre os seguintes valores:

- *penalty*: 'l1', 'l2'.
- *class_weight*: 'balanced', None.

- C: 1000, 100, 10, 1, 1e-1, 1e-2, 1e-3, 1e-4, 1e-5.

Assim, foi feita a validação cruzada, com o conjunto de treinamento e de validação, e obteve-se o seguinte conjunto de parâmetros ótimos: **C=10, class_weight='balanced', penalty='l2'**.

3.2 Análise de Desempenho

Com os parâmetros ótimos escolhidos, foi feito o treinamento utilizando tanto o conjunto de treinamento como o de validação. Depois, foi realizada a classificação do conjunto de teste, sendo possível analisar o desempenho do classificador. Para essa análise, utilizou-se: matriz de confusão, acurácia, acurácia balanceada, $F_1 - medida$ e curva ROC.

3.2.1 Matriz de Confusão

A matriz de confusão tem suas linhas representando a quantidade de dados originais, e as suas colunas representando a quantidade de dados previstos, para cada classe. Um bom modelo apresenta a maioria dos dados na diagonal principal desta matriz, que representa os dados corretamente previstos. A matriz de confusão obtida está apresentada na Figura 2.

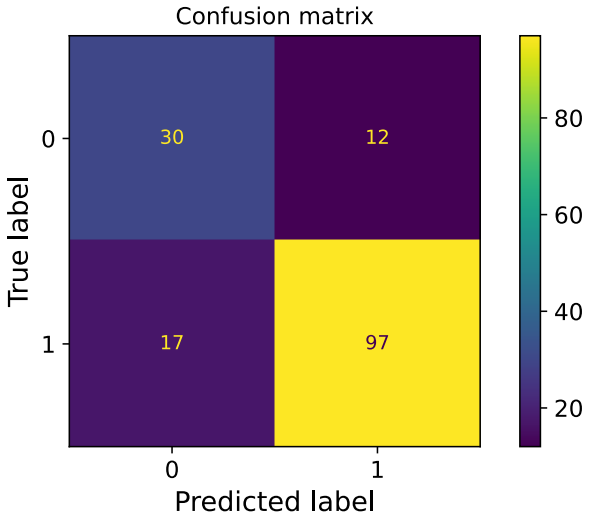

Figura 2: Matriz de confusão do classificador com regressão logística e parâmetros ótimos.

Note que, dentre 42 imagens com rótulo 0 (câncer), 30 foram corretamente classificadas, o que foi um ótimo resultado considerando que a classe negativa tem muito menor representatividade em todos os conjuntos. Ainda assim, seria um desempenho não satisfatório dada a seriedade da aplicação.

3.2.2 Acurácia

A acurácia é calculada pela equação 1.

$$Acuracia = \frac{TP + TN}{N} \tag{1}$$

Sendo TP o número de verdadeiros positivos, TN o número de verdadeiros negativos, e N o número total dos dados. Obteve-se uma acurácia de **0.8141025641025641**, de forma que

o classificador teve boa taxa de acerto, com a maioria das classes corretamente classificadas. Apesar disso, não seria suficiente para a aplicação real, conforme comentado anteriormente.

3.2.3 F_1-medida

O valor da sensibilidade (*recall*) é calculado pela equação 2.

$$recall = \frac{TP}{TP + FN} \tag{2}$$

O valor da precisão é calculado pela equação 3.

$$precisao = \frac{TP}{TP + FP} \tag{3}$$

A F_1-medida é calculada pela equação 4.

$$F1 - medida = \frac{2.precisao.recall}{recall + precisao} \tag{4}$$

Sendo TP o número de verdadeiros positivos, FN o número de falsos negativos, e FP o número de falsos positivos. O valor da F1-Medida foi de **0.8699551569506726**, o que indica que o classificador obteve bons resultados tanto na precisão quanto no *recall*. Isso indica que os casos verdadeiros positivos são maioria em relação aos casos falso positivo e falso negativo.

3.2.4 Acurácia Balanceada

O valor da especificidade é calculado pela equação 5.

$$especificidade = \frac{TN}{TN + FP} \tag{5}$$

O valor da sensibilidade é obtido pela equação 2. O valor da Acurácia Balanceada é obtido pela equação 6.

$$acurcia\_balanceada = \frac{especificidade + sensibilidade}{2} \tag{6}$$

Sendo TN o número de verdadeiros negativos, FP o número de falsos positivos. O valor da acurácia balanceada é **0.7825814536340852**. Note que o valor é menor que o das outras métrica, pois os valores de especificidade não foram tão bons e valores de *Recall* foram muito bons. Isso pode ser explicado porque, como visto na análise do conjunto de dados, há maior representatividade da classe positiva do que da classe negativa em cada conjunto. Isso significa que na classe positiva, por possuir mais dados, o classificador tem um melhor desempenho do que na classe negativa, que possui menos dados.

3.2.5 Curva ROC (sensibilidade X especificidade)

A curva ROC obtida foi apresentada na Figura 3. No canto inferior direito do gráfico, pode-se ver o valor da área sob a curva ROC: AUC = **0.81**.

Observando-a, pode-se perceber que o classificador tem certa utilidade, tal que se distancia de um classificador aleatório (pior caso, quando AUC=0.5). Ainda sim, percebe-se também que pode ser muito melhorado, de forma que a curva poderia ser muito mais próxima do canto superior esquerdo (caso ideal, quando AUC = 1) (GÉRON, 2019).

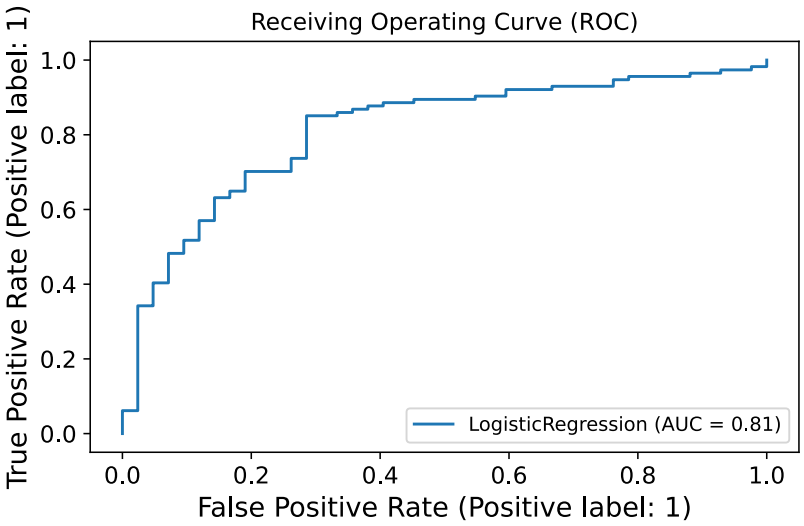

Figura 3: Curva ROC do classificador com regressão logística e parâmetros ótimos.

4 K vizinhos próximos (KNN)

Para o modelo de k vizinhos mais próximos, utilizou-se a função *KNeighborsClassifier()* da biblioteca *sklearn.neighbors* do *Scikit-learn*. Os parâmetros ajustados do modelo foram:

- Número de vizinhos próximos (*n_neighbors*).
- Pesos de cada ponto (*weights*). Este parâmetro pode ser do tipo *uniform*, que considera todos os pontos com o mesmo peso, ou ser do tipo *distance*, em que os pesos de cada ponto são proporcionais ao inverso da distância entre os pontos.
- Parâmetro "p"da métrica de distância de *Minkowski*, sendo p=1 definindo a métrica de distância de *Manhattan* e p=2 definindo a métrica de distância Euclidiana.
- O parâmetro *metric* define a métrica de distância utilizada. A métrica de distância foi fixada como distância de *Minkowski*.

Eles foram escolhidos por serem parâmetros já estudados na matéria e por serem coerentes com a classificação abordada.

Para uso da função, ainda foi necessário ajustar o formato das imagens, de forma a ficarem no formato de 1 linha e 784 colunas cada.

4.1 Validação Cruzada

Para a validação cruzada, foi utilizada a função *sklearn.linear_model.GridSearchCV()* da biblioteca *Scikit-learn* para testar diversas combinações de parâmetros para o classificador KNN, de forma a encontrar o melhor classificador. Para a validação cruzada, foi utilizado o método de *k_fold* para 5 *folds*, feitos a partir da junção dos conjuntos de treinamento e validação originais. Para a função de validação cruzada, foram ajustados os seguintes parâmetros:

- O estimador utilizado (*estimator*). Neste caso, foi utilizado como estimador a função *KNeighborsClassifier*.

- O dicionário de parâmetros a serem testados (*param_grid*).
- Parâmetro "cv"é referente ao número de *folds* utilizados. Foi fixado cv em 5 pastas.
- O parâmetro *verbose* define a verbalização do processo de treinamento.
- O parâmetro *scoring* define a função utilizada para avaliar o desempenho das combinações de parâmetros. Para este caso, foi utilizada a métrica de acurácia balanceada para encontrar os parâmetros que maximizam esta métrica.

Os parâmetros de KNN variados na validação foram: o número de vizinhos próximos (*n_neighbors*) para valores inteiros de 1 a 51, os pesos de cada ponto (*weights*) para o tipo *uniform* e para o tipo *distance*, o valor de "p"sendo 1 para distância de *Manhattan* e 2 para a distância Euclidiana. Após o treinamento do modelo com a validação cruzada, a função *best_params_* mostrou os parâmetros ótimos obtidos, que foram ***n_neighbors* igual a 1, p=2 e *weights*= *uniform***.

Com esses parâmetros, foi feito um gráfico da acurácia balanceada em função do número de vizinhos próximos. Este gráfico está na Figura 4. Analisou-se a acurácia balanceada, pois esta foi a métrica que buscou-se maximizar na função *GridSearchCV()*.

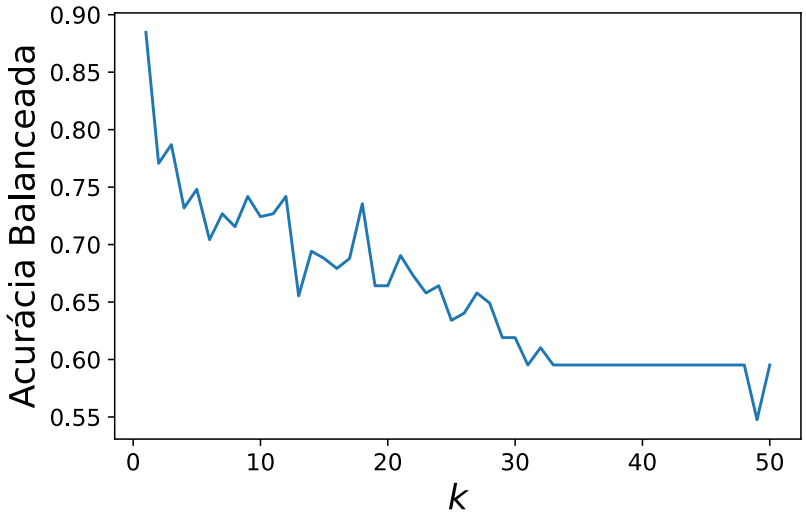

Figura 4: Acurácia balanceada para K vizinhos próximos.

Para os parâmetros ótimos, o modelo foi retreinado juntando os dados de validação e treinamento, e foi realizada a previsão para os dados de teste, obtendo uma acurácia de aproximadamente **0,769**.

4.2 Análise de Desempenho

Com o modelo pronto em sua melhor versão, foram calculadas as métricas de desempenho para avaliar a qualidade das previsões. Foi feita a Matriz de Confusão e as medidas de Acurácia, F_1-medida, e Acurácia Balanceada.

4.2.1 Matriz de Confusão

A matriz de confusão têm suas linhas representando a quantidade de dados originais, e as suas colunas representando a quantidade de dados previstos, para cada classe. Um bom modelo apresenta a maioria dos dados na diagonal principal desta matriz, que representa os dados corretamente previstos. A matriz de confusão obtida está na Figura 5.

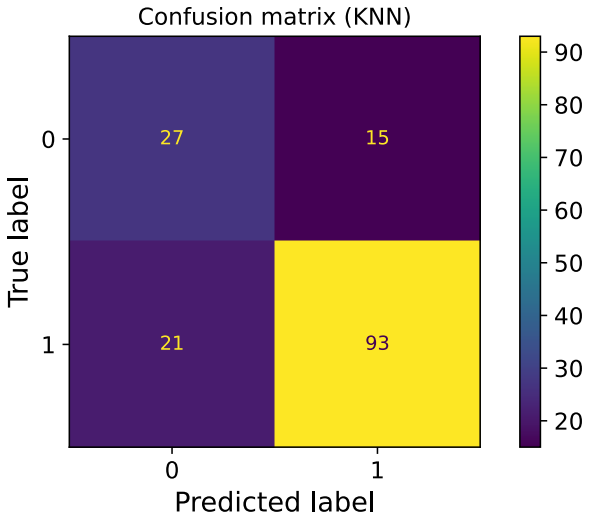

Figura 5: Matriz de confusão para o modelo KNN.

É possível perceber que a quantidade de verdadeiros negativos (câncer) é maior que a quantidade de falsos negativos e a quantidade de verdadeiros positivos (não câncer) é maior que a quantidade de falsos negativos. Portanto, há mais acertos do que erros nos dados de teste para o modelo escolhido. O KNN apresentou um desempenho ligeiramente inferior à regressão logística.

4.2.2 Acurácia

A acurácia é calculada pela equação 1. O valor da acurácia obtido foi **0,7692307692307693**, isso indica que o classificador obteve boa taxa de acerto com a maioria das classes corretamente classificadas. O KNN apresentou uma Acurácia ligeiramente inferior à regressão logística.

4.2.3 F_1-medida

A F_1-medida é calculada pela equação 4. O valor da F1-Medida obtido foi **0,8378378378378377**, isso indica que o cl obteve bons resultados tanto na precisão quanto no *recall*, ou seja, indica que os casos verdadeiros positivos são maioria com relação aos casos falso positivo e falso negativo. O KNN apresentou uma F1-medida ligeiramente inferior à regressão logística.

4.2.4 Acurácia Balanceada

O valor da Acurácia Balanceada é obtido pela equação 6. O valor da acurácia balanceada obtido foi **0,7293233082706767**, isso indica que o classificador obteve valores de especificidade não tão bons quanto os valores de *recall*. Isso significa que na classe positiva, por possuir mais dados, o classificador tem um melhor desempenho do que na classe negativa, que possui menos dados. O KNN apresentou uma Acurácia Balanceada ligeiramente inferior à regressão logística.

Referências

GÉRON, A. **Mãos à Obra: Aprendizado de Máquina com Scikit-Learn & TensorFlow**. [S.l.]: Alta Books, 2019. ISBN 9788550809021. Disponível em: <https://books.google.com.br/books?id=Z0mvDwAAQBAJ>.

SCIKIT-LEARN. ***sklearn.linear_model.LogisticRegression***. [S.l.: s.n.]. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html [Accessed: 12.10.2022].

YANG, Jiancheng et al. MedMNIST v2: A Large-Scale Lightweight Benchmark for 2D and 3D Biomedical Image Classification. **arXiv preprint arXiv:2110.14795**, 2021.