



Universidade Estadual de Campinas

Faculdade de Engenharia Elétrica e de Computação

IA048 – Aprendizado de Máquina

13 de abril de 2024

Docentes: Levy Boccato & Romis Attux

Discente:

– Gabriel Toffanetto França da Rocha – 289320

## Atividade 2 – Classificação

---

### Sumário

<b>1</b>	<b>Apresentação dos dados</b>	<b>2</b>
1.1	Dados tratados . . . . .	2
<b>2</b>	<b>Classificação via Regressão Logística</b>	<b>3</b>
2.1	Dados tratados . . . . .	4
2.2	Dados brutos . . . . .	8
<b>3</b>	<b>Classificação via <i>k</i> nearest neighbours</b>	<b>9</b>
3.1	Dados tratados . . . . .	9
3.2	Dados brutos . . . . .	12
<b>4</b>	<b>Comparação entre os classificadores</b>	<b>12</b>
4.1	Dados tratados . . . . .	12
4.2	Dados brutos . . . . .	12
	<b>Anexos</b>	<b>13</b>

---

# 1 Apresentação dos dados

O problema de classificação com o reconhecimento de atividades humanas utiliza como base de dados amostras tomadas do acelerômetro e do giroscópio do *smartphone* preso à cintura do candidato. Dessa forma, com base na leitura desses sensores, pode-se identificar se a pessoa está caminhando, subindo escadas, descendo escadas, sentada, de pé ou deitada, que representam as seis classes do problema.

Além dos dados brutos, é fornecido também os dados processados, com extração sobre os dados no tempo, na frequência, e também características estatísticas dos mesmos.

## 1.1 Dados tratados

Os dados tratados são formados por amostras de 561 atributos derivados da análise no tempo e na frequência dos dados provenientes do acelerômetro e do giroscópio do *smartphone*. São um total de 7352 amostras para treinamento e validação, e 2947 amostras para teste.

O balanceamento das classes nos conjuntos de dados foi realizado por meio do cálculo da taxa de ocorrência dos mesmos, dada de acordo com (1). A Figura 1 mostra a distribuição das classes, e pode-se ver que não existe um balanceamento homogêneo, onde a classe 3 é a que menos ocorre, enquanto a classe 6 é a que mais ocorre.

Devido a esse desbalanceamento, a métrica que será utilizada para a avaliação do desempenho de cada classificador será a acurácia balanceada, dada por (2).

$$Rate_i = \frac{N_i}{N} \quad (1)$$

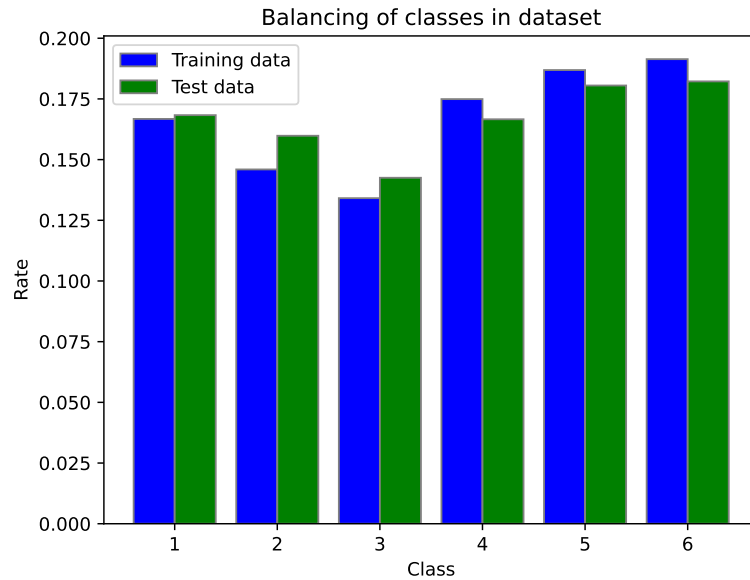


Figura 1: Gráfico da ocorrência das classes nos conjuntos de dados de treinamento e teste.

$$BA = \frac{\sum_{i=1}^Q Recall_i}{Q} = \frac{\sum_{i=1}^Q \frac{TP_i}{N_i}}{Q} = \frac{\sum_{i=1}^Q \frac{TP_i}{N \cdot Rate_i}}{Q} \quad (2)$$

## 2 Classificação via Regressão Logística

A classificação multi-classe é feita de forma elegante ao ter um modelo, que dadas  $Q$  classes à serem reconhecidas, apresente  $Q$  saídas, onde cada saída é a probabilidade da amostra pertencer à classe em questão. Tal implementação se dá por meio da função *softmax*, enunciada em (3), e a saída é dada pela notação *one-hot encoding*.

$$\hat{y}_k(\mathbf{x}(i)) = \frac{e^{(\Phi(\mathbf{x}(i))^T \mathbf{w}_k)}}{\sum_j e^{(\Phi(\mathbf{x}(i))^T \mathbf{w}_j)}} \quad (3)$$

Onde o  $\mathbf{w}_k$  é o vetor de pesos para a classe  $k$ , e a matriz de pesos  $\mathbf{W}$  é dada por (4).

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_Q \end{bmatrix} \quad (4)$$

Não existe forma fechada para a obtenção dos pesos de  $\mathbf{W}$ , logo, o mesmo precisa ser feito de forma iterativa. A métrica utilizada como função de custo para o problema é a entropia cruzada, dada por (5). O otimização dos pesos se dá pela técnica do gradiente descendente, dado por (6).

$$J_{CE}(\mathbf{W}) = - \sum_{i=0}^{N-1} \sum_{k=1}^Q y_{i,k} \log [\hat{y}_k(\mathbf{x}(i))] \quad (5)$$

$$\frac{\partial J_{CE}(\mathbf{W})}{\partial \mathbf{w}_k} = \sum_{i=0}^{N-1} (y_{i,k} - \hat{y}_k(\mathbf{x}(i))) \Phi(\mathbf{x}(i))^T \quad (6)$$

A atualização dos pesos é dada por (7), onde  $l$  é a iteração dos pesos.

$$\mathbf{W}[l+1] = \mathbf{W}[l] - \eta \nabla \mathbf{W} \quad (7)$$

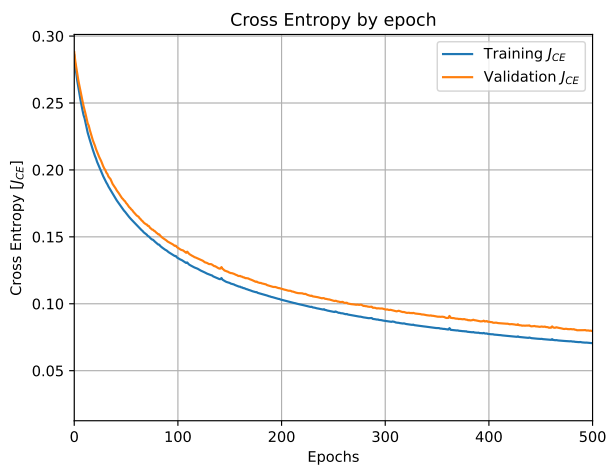
Devido ao tamanho do conjunto de dados, foi proposto o treinamento por *mini-batch*, testadas com tamanhos de 500, 1000 e 2000 amostras, e um passo ( $\eta$ ) de 0,01, que apresentou boa convergência nos testes realizados *a priori*. O processo de validação cruzada para o treinamento do modelo foi feito na forma *holdout*, considerando o conjunto de validação 30% do *dataset* de treinamento. A inicialização dos pesos foi feita de forma aleatória, seguindo uma distribuição gaussiana, nos entornos de 0.

## 2.1 Dados tratados

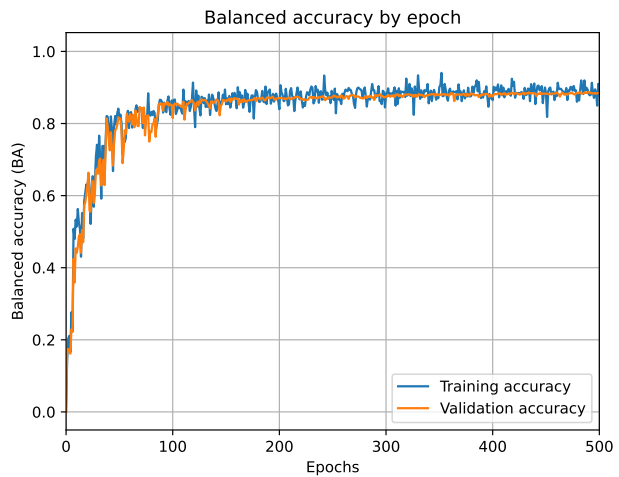
### *Mini-batch* 500 amostras

Para um primeiro caso, treinou-se o modelo de regressão logística utilizando o *mini-batch* de 500 amostras. A cada 500 amostras de treinamento, entregues em ordem aleatória a cada época, realiza-se a evolução da matriz de pesos  $\mathbf{W}$ , calculando a nova função de custo (entropia cruzada), e o valor da acurácia balanceada, para todo conjunto de treinamento e validação.

A Figura 2a mostra o decaimento da entropia cruzada de treinamento e validação durante as épocas de treinamento. Para 500 épocas, observa-se que não há o estado de regime das métricas, ou a inversão do crescimento da entropia cruzada de validação, que indicaria *overfitting*. Porém, analisando a acurácia balanceada do modelo, exibida na Figura 2b, percebe-se que a medida de desempenho apresenta saturação, após aproximadamente 300 épocas de treinamento, sendo assim utilizada para a parada antecipada (*early stopping*) do treinamento do modelo.



(a) Evolução da entropia cruzada.



(b) Evolução da acurácia balanceada.

Figura 2: Evolução da entropia cruzada e da acurácia balanceada durante o treinamento para *mini-batch* de 500 amostras e  $\eta = 0,01$ .

Com o treinamento, obteve-se um classificador que ao aplicar os dados de teste, obteve uma acurácia balanceada média de 0,9162, com matriz de confusão mostrada na Tabela 1 e métricas de precisão e *recall* por classe listadas na Tabela 2. Observa-se facilmente o grande desempenho do classificador para a classe 6, possuindo total assertividade em identificar a classe, sem haver nenhum engano da classe com as outras, e vice-versa. Pela matriz, observa-se grande confusão da classe 1 com a classe 2 e 3, justificando a mesma possuir o menor *recall*. Já a classe 3 é a que possui menor precisão, sendo classificada erroneamente como a classe 1 e 2.

$$BA = 0,9162 \quad (8)$$

	1	2	3	4	5	6
1	486	0	10	0	0	0
2	30	440	1	0	0	0
3	39	52	329	0	0	0
4	0	3	0	424	64	0
5	1	0	0	33	498	0
6	0	0	0	0	0	537

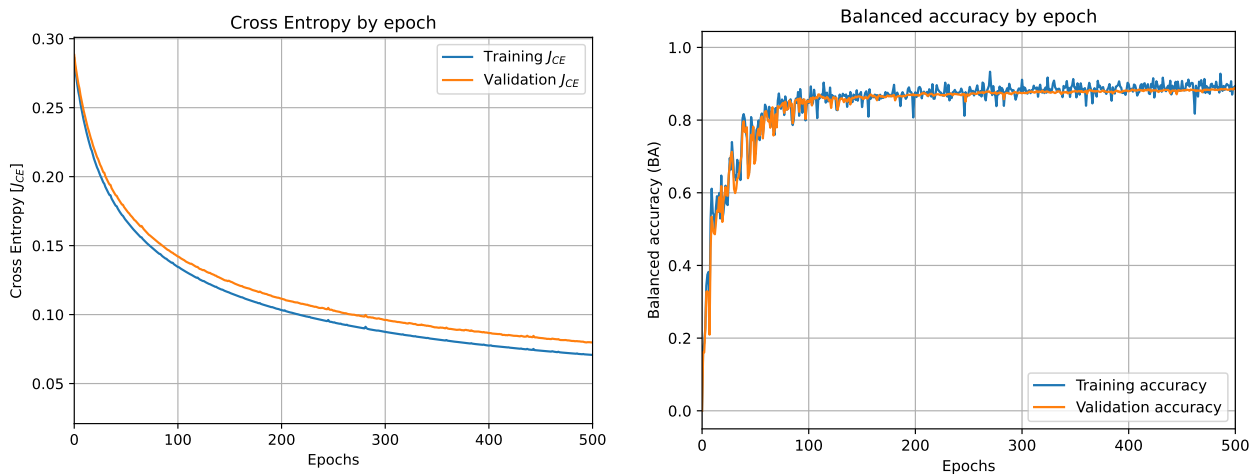
Tabela 1: Matriz de confusão do classificador com *mini-batch* de 500 amostras.

Classe	Precisão	Recall
1	0.9798	0.8741
2	0.9342	0.8889
3	0.7833	0.9676
4	0.8635	0.9278
5	0.9361	0.8861
6	1.0000	1.0000

Tabela 2: Precisão e *Recall* do classificador com *mini-batch* de 500 amostras por classe.

### Mini-batch 1000 amostras

Ao aumentar o tamanho de amostras por atualização da matriz de pesos, observa-se uma convergência mais suave e mais assertiva, apresentando uma curva de acurácia balanceada (Figura 3b) mais bem comportada, devido à apresentar uma maior média de amostras para formar uma iteração de  $\mathbf{W}$  pelo gradiente negativo da entropia cruzada (Figura 3a).



(a) Evolução da entropia cruzada.

(b) Evolução da acurácia balanceada.

Figura 3: Evolução da entropia cruzada e da acurácia balanceada durante o treinamento para *mini-batch* de 1000 amostras e  $\eta = 0,01$ .

Com a alteração do tamanho da batelada utilizada, o modelo obteve um decréscimo no seu desempenho, atingindo uma acurácia balanceada de 0,9042. O comportamento da matriz de confusão (Tabela 3) e das métricas de precisão e *recall* (Tabela 4) são similares, porém com menor desempenho comparados ao modelo anterior.

$$BA = 0,9042 \quad (9)$$

	1	2	3	4	5	6
1	486	0	10	0	0	0
2	28	443	0	0	0	0
3	56	53	311	0	0	0
4	0	3	0	426	61	1
5	0	1	0	54	477	0
6	0	0	0	0	0	537

Tabela 3: Matriz de confusão do classificador com *mini-batch* de 1000 amostras.

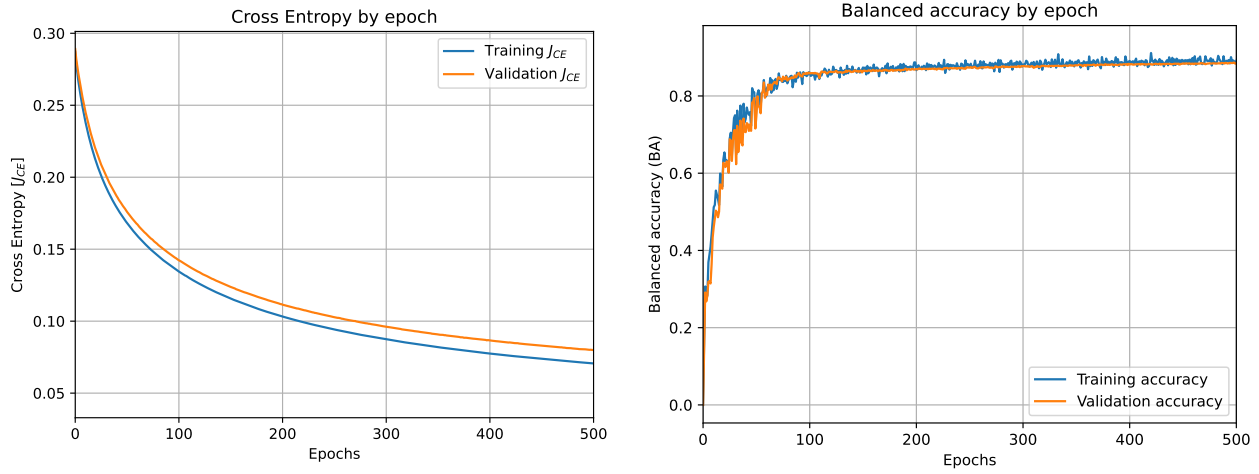
Classe	Precisão	<i>Recall</i>
1	0.9798	0.8526
2	0.9406	0.8860
3	0.7405	0.9688
4	0.8676	0.8875
5	0.8966	0.8866
6	1.0000	0.9981

Tabela 4: Precisão e *Recall* do classificador com *mini-batch* de 1000 amostras por classe.

### ***Mini-batch* 2000 amostras**

Aumentando ainda mais o número de amostras por *mini-batch*, observa-se uma envolução ainda mais comportada da acurácia balanceada, apresentando oscilações muito menos perceptíveis, como visto na Figura 4b. O decaimento da função de custo, Figura 4a se dá da mesma forma, uma vez que cada evolução do vetor de peso carrega a média da entropia cruzada para cada amostra da batelada.

O desempenho obtido ao utilizar 2000 amostras por *mini-batch* é ainda menor, chegando a uma acurácia balanceada de 0,8898. A matriz de confusão, como pode ser observado na Tabela 5 e precisão e *recall* das classes (Tabela 6), se apresentam da mesma forma, onde mesmo com a queda de *performance*, se observa o mesmo perfil de comportamento para as classes.



(a) Evolução da entropia cruzada.

(b) Evolução da acurácia balanceada.

Figura 4: Evolução da entropia cruzada e da acurácia balanceada durante o treinamento para *mini-batch* de 2000 amostras e  $\eta = 0,01$ .

$$BA = 0,8898 \quad (10)$$

	1	2	3	4	5	6
1	489	0	7	0	0	0
2	27	444	0	0	0	0
3	70	49	301	0	0	0
4	0	3	0	396	91	1
5	1	1	0	58	472	0
6	0	0	0	0	0	537

Tabela 5: Matriz de confusão do classificador com *mini-batch* de 2000 amostras.

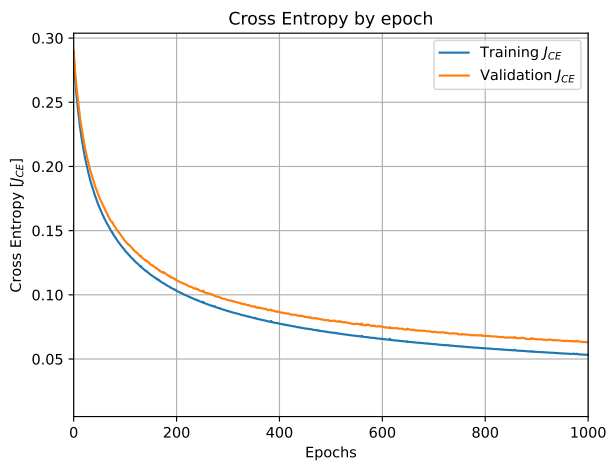
Classe	Precisão	Recall
1	0.9859	0.8330
2	0.9427	0.8934
3	0.7167	0.9773
4	0.8065	0.8722
5	0.8872	0.8384
6	1.0000	0.9981

Tabela 6: Precisão e *Recall* do classificador com *mini-batch* de 2000 amostras por classe.

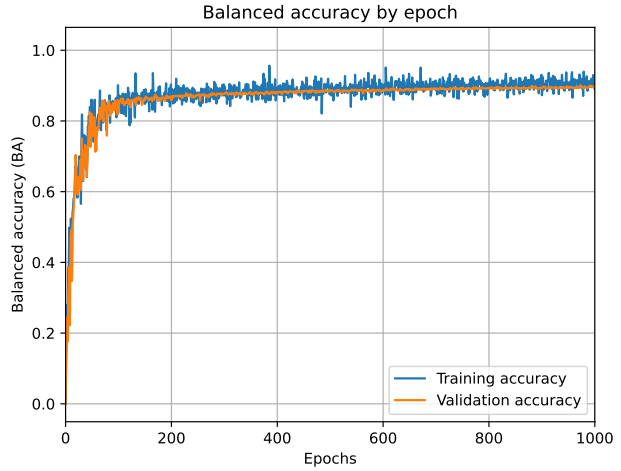
## Análise

Foi realizado o treinamento do modelo para a *mini-batch* de 500 amostras em um *range* de 1000 épocas, para observar a acomodação da função de custo. Observa-se na Figura 5a que

a taxa de decrescimento da entropia cruzada vai reduzindo, porém não chega a um estado de regime, também não obtendo o estado de *overfitting* com o aumento do custo para a validação. É possível aferir também que não há grande aumento da acurácia balanceada, se fazendo então, o treinamento de 500 épocas feito anteriormente, suficiente para a obtenção de um modelo que explora de forma suficiente o espaço de hipóteses ( $\mathcal{H}$ ).



(a) Evolução da entropia cruzada.



(b) Evolução da acurácia balanceada.

Figura 5: Entropia cruzada e acurácia balanceada para o treinamento do modelo com *mini-batch* de 500 amostras por 1000 épocas.

## 2.2 Dados brutos



### 3 Classificação via *k nearest neighbours*

A classificação pelo método *k nearest neighbours* é baseada em inferir a classe do dado a ser classificado com base nos  $k$  dados mais próximos à ele. Como hiper-parâmetros para esse problema, têm-se principalmente o valor de  $k$ , a ordem  $p$  da distância de Minkowski entre os dados e o critério de classificação.

O critério de classificação pode se basear puramente na classe majoritária entre os  $k$  vizinhos, ou levar em consideração a distância como um peso, que normalmente é inversamente proporcional a distância, evidenciando o rótulo dos pontos mais próximos do dado teste.

#### 3.1 Dados tratados

Para implementação do algoritmo de  $k$ -NN, foi escolhido a utilização da distância euclidiana no espaço dos atributos, e a decisão do rótulo vencedor por meio do voto majoritário dos  $k$  vizinhos mais próximos.

Para obtenção do valor de  $k$ , foi executada uma busca em *grid* do hiper-parâmetro, variando seu valor entre 1 e 29. Utilizando da técnica de validação cruzada *k-fold*, com quatro pastas, foi realizada a inferência das classes dos dados da pasta de validação com base nos vizinhos mais próximos encontrados nas pastas de treinamento, para cada valor de  $k$  testado. A Figura 6 exibe a evolução da acurácia balanceada para os valores de  $k$ , obtendo um conjunto de valores ótimos em (11).

$$k = [17 \ 28 \ 12 \ 18] \quad (11)$$

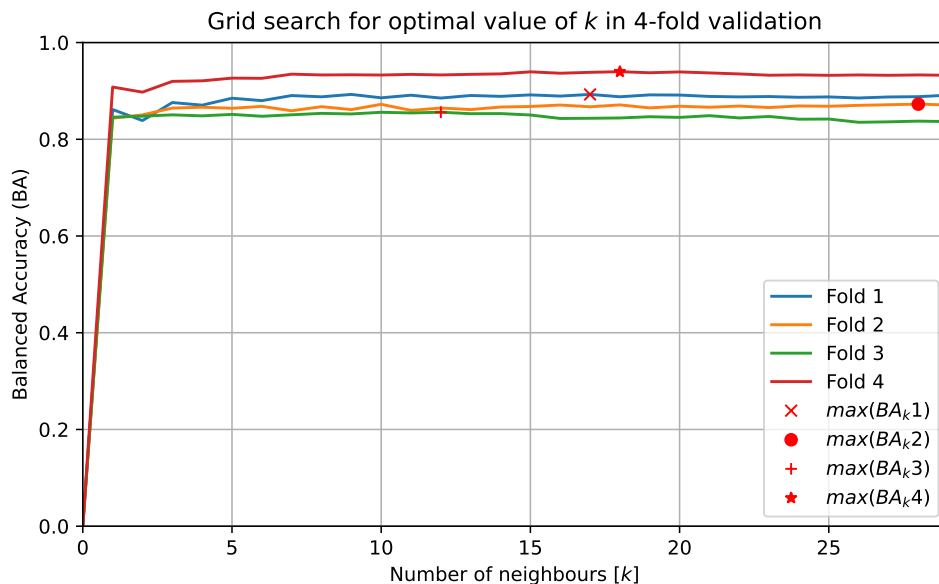


Figura 6: Busca em *grid* do valor de  $k$  ótimo utilizando *4-fold validation*.

A heurística escolhida para avaliar o melhor valor de  $k$  com base no conjunto obtido por meio da busca em *grid* com validação cruzada se dá em obter a acurácia balanceada média das

pastas e obter o número de vizinhos que maximiza essa combinação das pastas. A Figura 7 mostra a progressão da acurácia balanceada média de acordo com  $k$ , e assim se obtém o valor de  $k$  ótimo em  $k = 15$ .

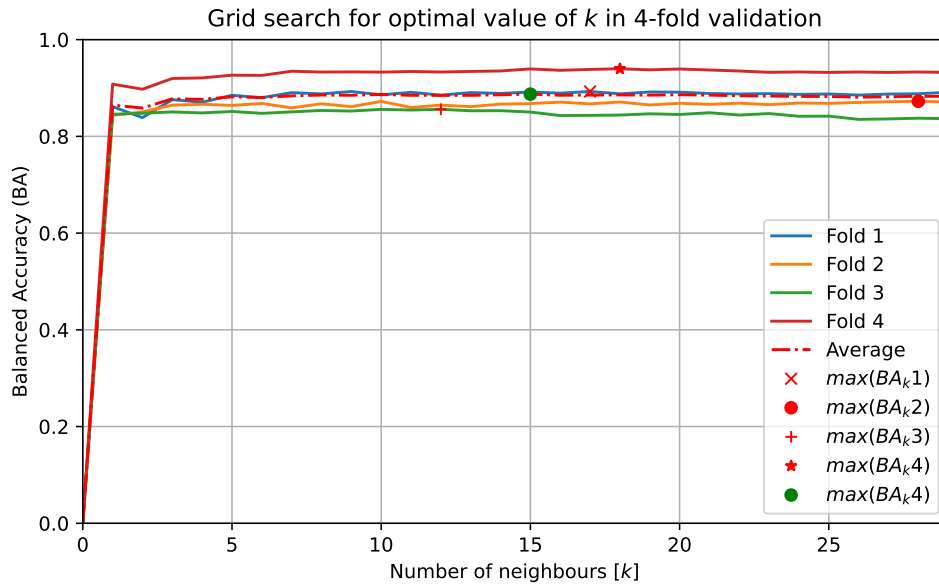


Figura 7: Busca em *grid* do valor de  $k$  ótimo utilizando *4-fold validation*.

Uma vez definido o classificador ótimo, obtém-se os indicadores de performance do classificador com base nos dados de teste. A acurácia balanceada encontrada foi de 0,8991 e a matriz de confusão do classificador por ser vista na Tabela 7.

$$BA = 0,8991 \quad (12)$$

	1	2	3	4	5	6
1	488	0	8	0	0	0
2	39	427	5	0	0	0
3	51	44	325	0	0	0
4	0	4	0	389	98	0
5	0	0	0	31	501	0
6	0	0	0	1	1	535

Tabela 7: Matriz de confusão do classificador  $k$ -NN com  $k = 15$ .

Extraíndo da matriz de confusão as métricas de precisão e *recall*, obtém-se a Tabela 8. Pode-se observar que a classe 3 foi a que apresentou menor precisão, sendo muito confundida com a classe 1 e 2. Já a classe 1 possui o pior *recall*, uma vez que as classes 2 e 3 se confundem com a 1. A classe 6 foi a que apresentou o melhor desempenho, apresentando *recall* unitário, logo, nenhuma classe se confunde com ela, e a maior precisão, muito próxima de 1.

Classe	Precisão	Recall
1	0.9839	0.8443
2	0.9066	0.8989
3	0.7738	0.9615
4	0.7923	0.9240
5	0.9417	0.8350
6	0.9963	1.0000

Tabela 8: Precisão e *Recall* do classificador por classe.

### Comparação com a regressão logística

Ao comparar os classificadores utilizando a regressão logística e o  $k$ -NN, observa-se que a acurácia balanceada para a regressão logística de melhor desempenho foi superior, mas para ter uma comparação mais minuciosa entre os dois classificadores, deve-se analisar também suas matrizes de confusão. Comparando a precisão e o *recall*, o desempenho das classes em *ranking* se apresenta da mesma forma, porém, para a regressão logística a classe 6 conseguiu obter tanto precisão quanto *recall* máximo.

A regressão logística apresentou um desempenho relativo 1,87% maior que o  $k$ -NN para a acurácia balanceada, porém possui uma complexidade muito maior. Para obter o modelo de regressão logística multi-classe com *softmax*, é necessário realizar o treinamento do modelo, buscando variar os hiper-parâmetros para se conseguir o melhor desempenho. Essa etapa demanda de muito processamento, porém garante uma utilização do classificador mais leve, uma vez que é necessária apenas o vetor de pesos. Já o  $k$ -NN, não demanda treinamento, sendo necessária apenas a inferência da classe com base nos vizinhos mais próximos, o que apesar de consumir armazenamento para manter o *dataset*, não precisa de tanto processamento prévio, sendo uma solução simples e rápida. Porém, devido a busca em *grid* utilizando a validação *k-fold* para obter o melhor  $k$  para os dados de treinamento, tal etapa demandou bastante processamento, o que elevou consideravelmente a complexidade do  $k$ -NN.

### Comparação com o $k$ ótimo

Somente a título de comparação, foi realizada a busca em *grid* do melhor valor de  $k$ , utilizando os próprios dados de teste do problema. A Figura 8 mostra a evolução da acurácia balanceada dos dados de teste para o aumento do número de vizinhos, chegando ao valor ótimo em  $k = 8$ , com  $BA = 0,9028$ .

Em comparação com o número de vizinhos obtidos via validação em  $k$ -pastas, observa-se que o número de vizinhos é perto da metade do obtido anteriormente, e apresentando uma performance melhor. Porém, ao se realizar uma busca de um hiper-parâmetro com métodos de busca em *grid* utilizando validação cruzada, busca-se obter o melhor valor com base nos dados de treinamento, que são os únicos conhecidos na etapa de treinamento/modelagem do classificador. Espera-se que os dados de treinamento sejam suficiente para generalizar os dados

como um todo, fazendo com que a solução ótima para o *dataset* de treinamento tenda a ser a solução ótima para todos os dados inéditos que o modelo irá receber.

Dessa forma, mesmo o valor de  $k$  tendo mudado consideravelmente, a acurácia não apresentou grandes mudanças, sendo ainda sim um bom modelo para conseguir fazer a classificação dos dados de teste.

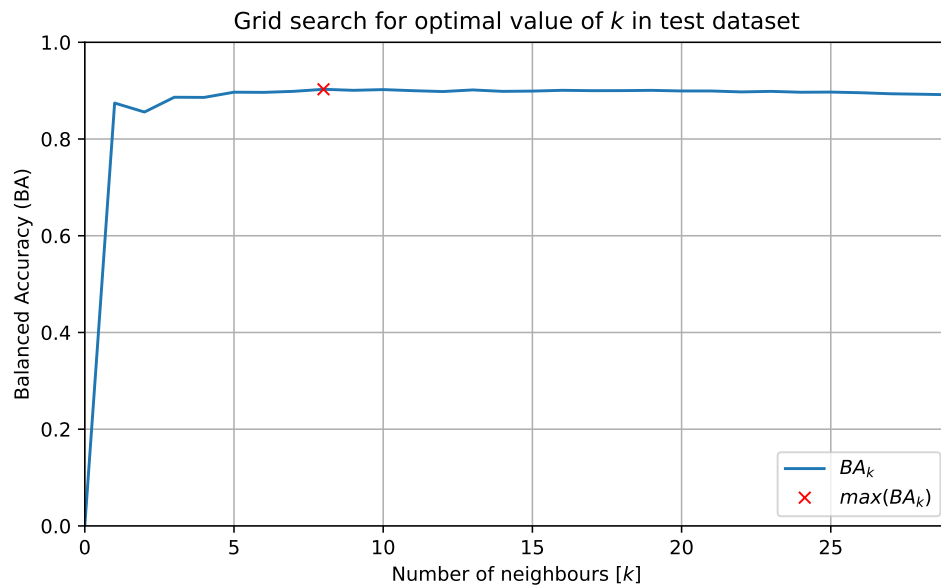


Figura 8: Busca em *grid* do valor de  $k$  ótimo para o conjunto de testes.

### 3.2 Dados brutos

## 4 Comparação entre os classificadores

### 4.1 Dados tratados

### 4.2 Dados brutos

# Anexos

## Códigos fonte

Todos os códigos fonte e arquivos de dados utilizados para a elaboração deste documento podem ser encontrados no repositório do GitHub no link: [github.com/toffanetto/ia048](https://github.com/toffanetto/ia048).