Programa de Pós-graduação em Modelagem Computacional
GB-500 – Redes Neurais Profundas Para Análise de Síntese de Dados (2025.2)
Discente: Gilson A. Giraldi

Gabriel Rezende da Silva
rezgab@posgrad.lncc.br

Exercise 1

Implementation and test of perceptron model:

(a) Generate and visualize a database S such that:

$$S \subset \mathrm{R}^2 \times \{+1, -1\}, \tag{1}$$

with $|S| = 100$ samples, composed of two classes C_1 and C_2.

O conjunto de dados foi gerado a partir de duas distribuições gaussianas (`np.random.randn`), um para cada clsse, multiplicadas por um mesmo ruído `noise = 0.5` (para tornar os dados mais realistas) e somadas por médias específicas para cada classe, `mean0 = np.array([1, 1])` e `mean1 = np.array([3, 3])`. A Figura 1 mostra a distribuição do conjunto de dados linearmente separável.

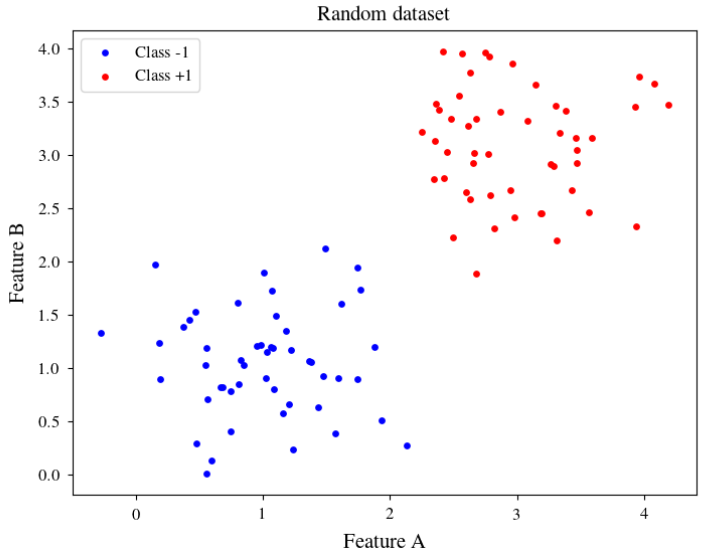

Figura 1: Conjunto de dados linearmente separável.

(b) Shuffle the dataset S and randomly split it into disjoint sets D_{tr} (training) and D_{te} (testing). OBS: Verify if the classes C_1 and C_2 are **balanced** in D_{tr} and D_{te}.

O conjunto de dados foi dividido em treinamento (80%) e teste (20%) através da função `train_test_split` da biblioteca scikit-learn, respeitando a proporção entre as classes nos subconjuntos. A Figura 2 mostra a distribuição do conjunto de dados linearmente separável divido em treinamento e teste.

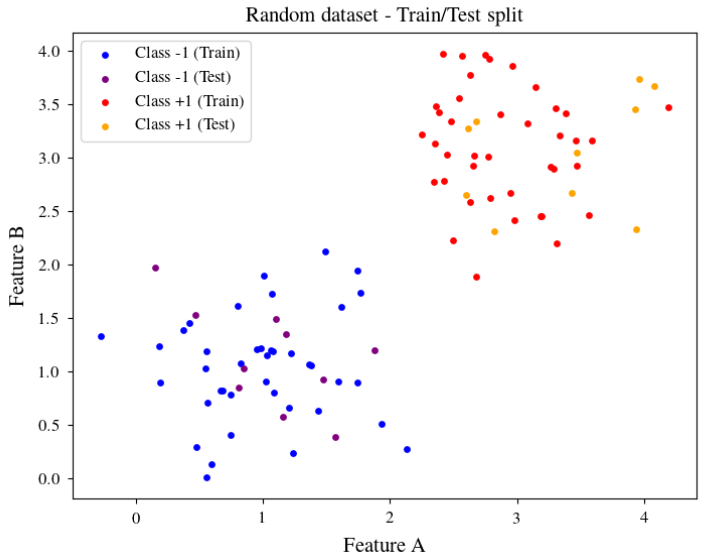

Figura 2: Conjunto de dados dividido em treinamento e teste.

(c) Implement the perceptron model for classification in $\mathrm{R}^2 \times \{+1, -1\}$ and perform training using the training set D_{tr}.
Show some graphical configurations of the line that partitions the pattern space together with the final solution.

O Perceptron foi implementando a partir da classe `CustomPerceptron`, sem a utilização de bibliotecas encapsuladas. Dado um número máximo de épocas e uma taxa de aprendizagem, a classe segue os passos do algoritmo de referência. A classe implementa duas funções principais: `fit`, que realiza o treinamento até o erro ser zero (todas as respostas desejadas corretas) ou exceder o número máximo de épocas; e `predict`, que utiliza os melhores pesos encontrados para calcular a reposta para um dado subconjunto, normalmente de teste. Além disso, pode-se utilizar o atributo `weights` para retornar os pesos e o bias calculados, a fim de estabelecer a reta de decisão do Perceptron. A Figura 3 mostra a distribuição do conjunto de dados linearmente separável, divido em treinamento e teste, e a reta de decisão do Percpetron. A convergência para os parâmetros estabelcidos (`epochs = 10` e `lr = 1`) foi atingida em 3 épocas.

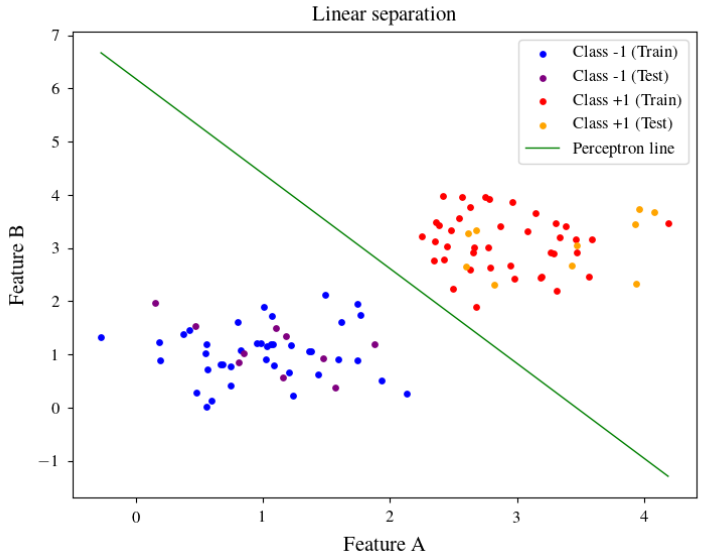

Figura 3: Reta de decisão encontrada pelo Perceptron.

(d) Evaluate the model using the accuracy measure computed over D_{te}.

Analisando a Figura 3, percebe-se que o Percetron encontrou uma reta de decisão que separa as classes corretamente. Portanto, a acurácia obtida foi de 100%.

Exercise 2

Consider the FEI face image database and the gender classification problem. Choose a feature space and apply leave-one-out multi-fold cross-validation, with $K = 4$, for perceptron model. In this case, use the perceptron available in libraries for neural network implementation, like Keras, Tensor flow or scikit-learn.

Neste trabalho, foram utilizadas apenas imagens frontais, totalizando 198 homens e 202 mulheres. A resolução original das imagens é de $360px \times 320px$. Primeiro, as imagens foram redimensionadas por um fator de $\frac{1}{5}$, obtendo uma resolução de $90px \times 65px$ e preservando a razão de aspecto. Em seguida, as imagens foram convertidas em escala de cinza, vetorizadas e normalidas pela técnica Min-Max, alterando os valores do pixels para o intervalo $[0, 1]$. Dessa forma, a dimensionalidade do conjunto de dados após o pré-processamento é de 400×5850.

Por fim, utilizou-se o algoritmo PCA da biblioteca scikit-learn a fim de analisar se é possível trabalhar em um espaço de dimensão ainda mais reduzida. As Figuras 4 e 5 mostram o espaço de dimensão reduzida para os dois maiores autovalores e a soma acumulada dos autovalores, respectivamente. Note que, além de uma separabilidade aparente dos dados, é possível truncar os dados a partir de um treshold de referência que explique uma alta variabilidade do conjunto de dados. Portanto, foi utilizado um treshold de 95%, fazendo com que a dimensionalidade dos dados seja reduzida para 400×110.

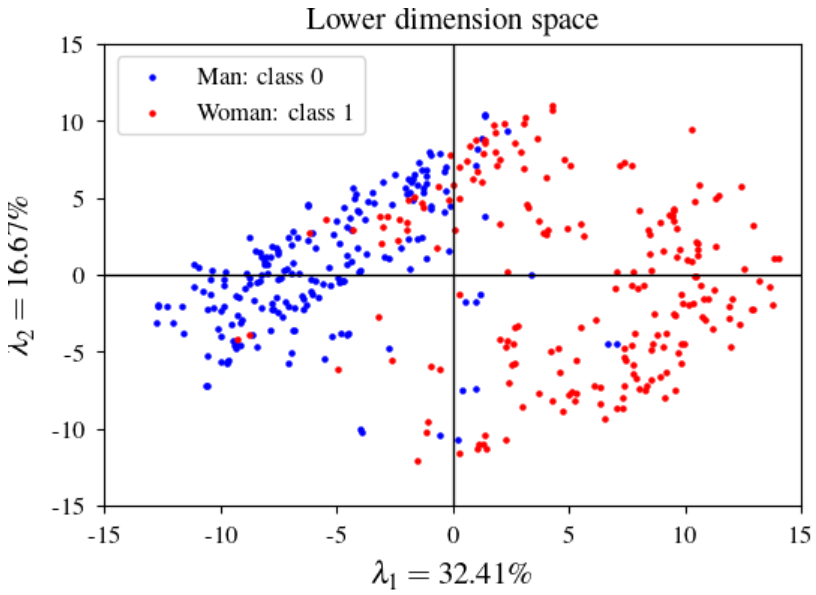

Figura 4: Espaço de dimensão reduzida para os dois maiores autovalores.

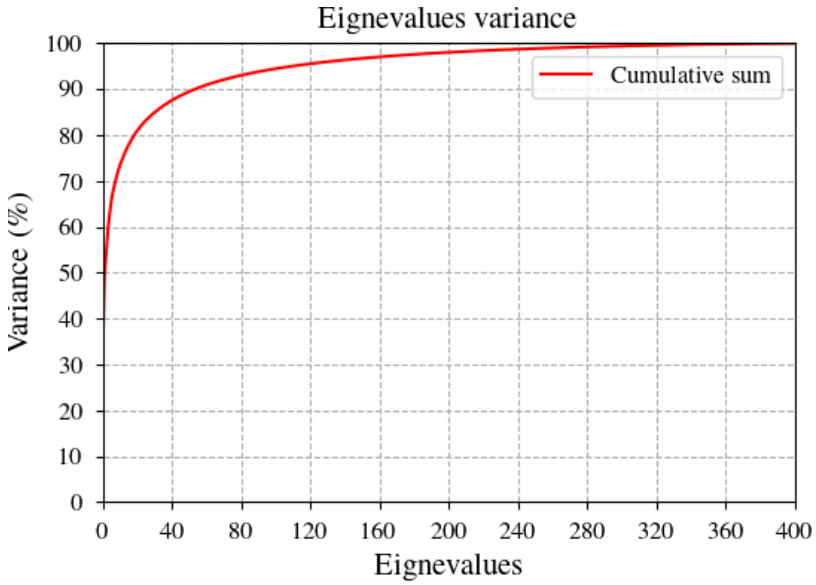

Figura 5: Soma acumulada dos autovalores.

A validação cruzada foi implementada pelas bibliotecas do scikit-learn `train_test_split` e [StratifiedKFold] (https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html). Cada k-fold tem a seguinte configuração: `Train: [man: 111, woman: 114] | Val.: [man: 37, woman: 38] | Test: [man: 50, woman: 50]`. Seguindo a recomendação do exercício, o subconjunto de teste é fixo, enquanto que o de validação é variável, fazendo com que cada amostra participe da validação uma única vez.

O Percetron foi implementado pela biblioteca do scikit-learn `Perceptron`. Para avaliar o histórico do treinamento, foi utilizada a função [`partial_fit`], com o loop em torno das épocas sendo implementando manualmente. Além do histórico da acurácia, foi salvo o histórico da loss, que segue a seguinte equação:

$$\text{Loss} = \sum_i \max(0, -y_i \cdot f(x_i)),$$

em que:

- $y_i \in \{-1, 1\}$ são os rótulos verdadeiros.
- $f(x_i) = w \cdot x_i + b$ é a saída do modelo (antes da função de ativação).

Os rótulos foram ajustados para $\{-1, +1\}$ apenas para o cálculo da função de perda. Os parâmetros utilizados foram `epochs = 100`, `lr = 1e-3` (taxa de aprendizagem), `tol = 1e-4` (tolerância da loss de validação) e `patience = 10` (paciência de epocas sem melhora da loss de validação).

(a) Show the graphical representation of the evolution of training and validation stages.

A seguir são mostrados os gráficos dos históricos de acurácia e perda para o Perceptron. Note que, aparentemente, há uma situação de overfitting dos modelos devido à distância de desempenho dos subconjuntos de treinamento e validação (alguns folds mais outro menos). Apesar disso, os modelos convergem para patamares de acurácia de validação satisfatórios, indicando também que houve aprendizado.

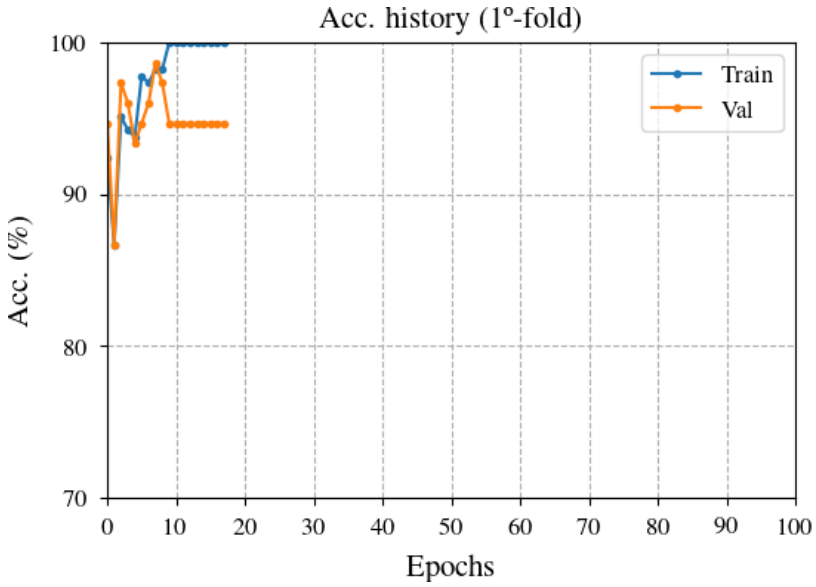

Figura 6: Histórico da acurácia para o Perceptron (1º-fold).

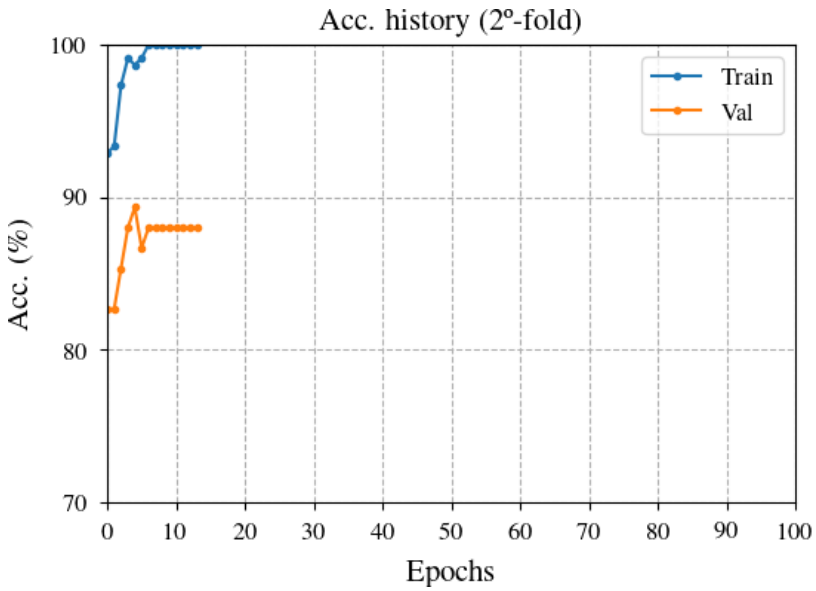

Figura 7: Histórico da acurácia para o Perceptron (2º-fold).

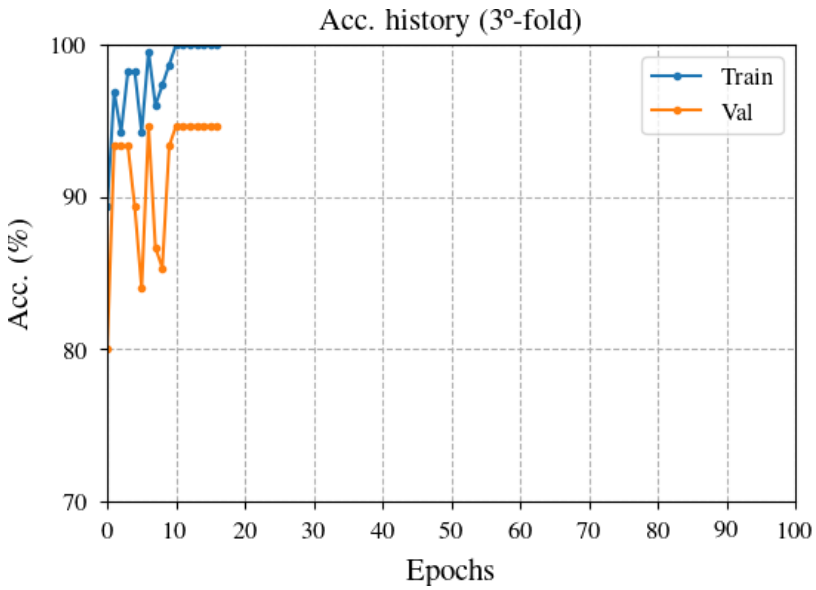

Figura 8: Histórico da acurácia para o Perceptron (3º-fold).

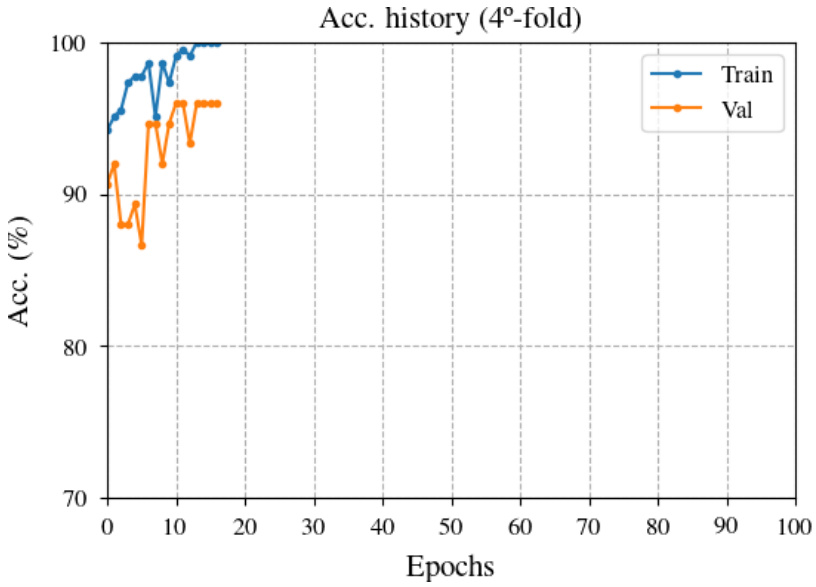

Figura 9: Histórico da acurácia para o Perceptron (4º-fold).

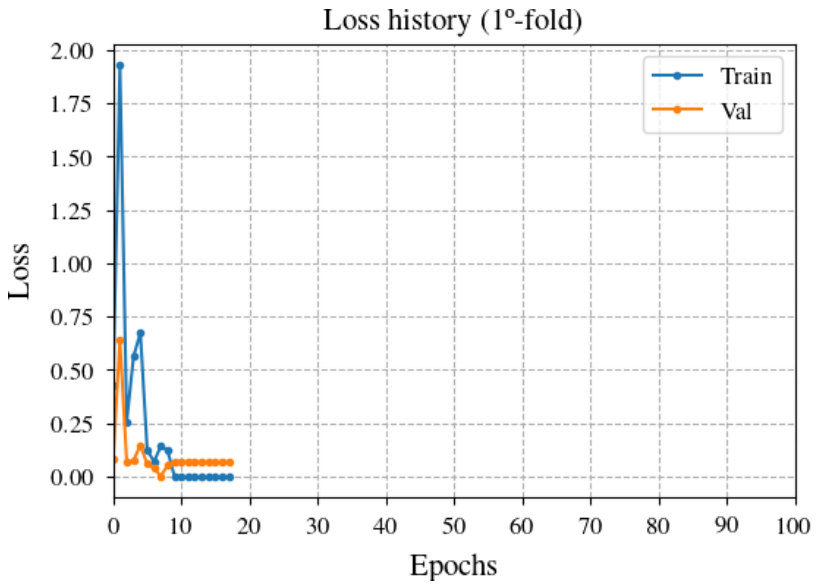

Figura 10: Histórico da loss para o Perceptron (1º-fold).

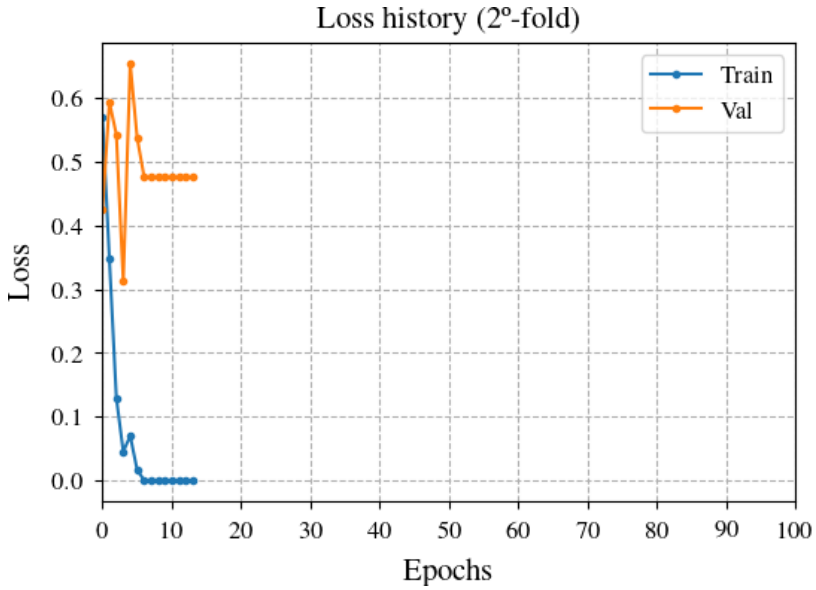

Figura 11: Histórico da loss para o Perceptron (2º-fold).

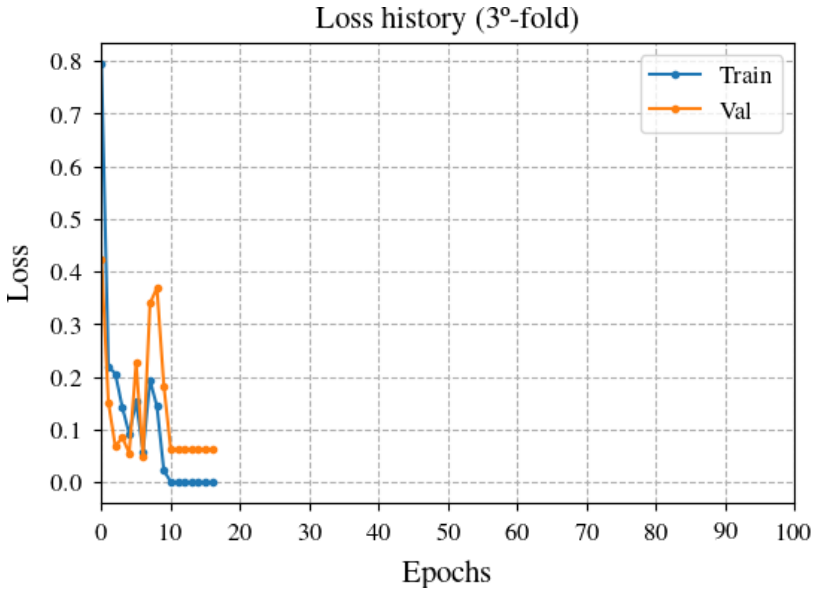

Figura 6: Histórico da loss para o Perceptron (3º-fold).

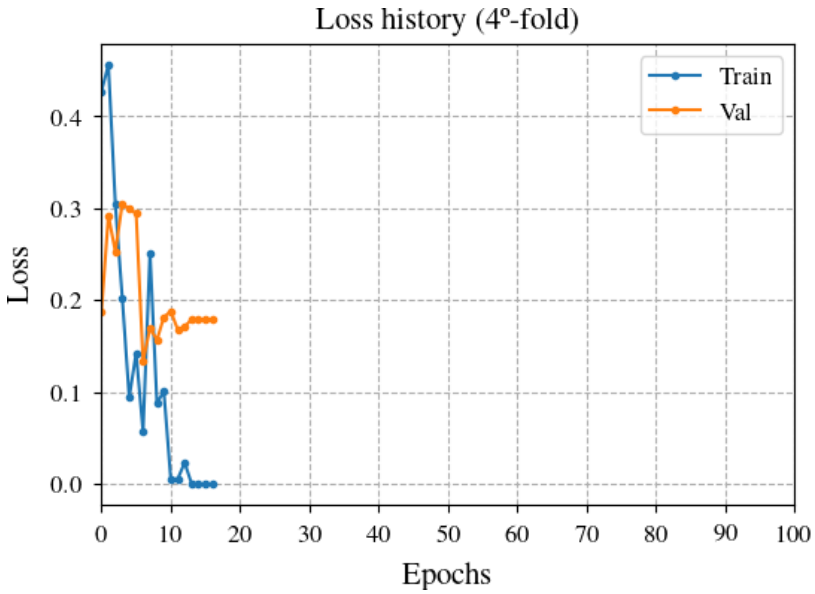

Figura 6: Histórico da loss para o Perceptron (4º-fold).

(b) Perform a statistical analysis of the performance of the four models applied over the test set D_{te}.

Utilizou-se a acurácia para analisar o desempenho dos modelos no subconjunto de teste.

- Modelo 1: `Overall acc.: 94.00% | Man acc.: 92.00% | Woman acc.: 96.00%`
- Modelo 2: `Overall acc.: 95.00% | Man acc.: 92.00% | Woman acc.: 98.00%`
- Modelo 3: `Overall acc.: 97.00% | Man acc.: 96.00% | Woman acc.: 98.00%`
- Modelo 4: `Overall acc.: 96.00% | Man acc.: 98.00% | Woman acc.: 94.00%`

Os resultados se demonstraram satisfatórios, com o Modelo 3 obtendo e melhor acurácia geral, o Modelo 4 a melhor acurácia para a classe homem, e os Modelos 2 e 3 para a classe mulher. É importante ressaltar que isso pode ser explicado por uma divisão do conjunto de dados que favoreça esses resultados, como um subconjunto de teste com amostras fáceis de serem classificadas, ou seja, distantes da reta de decisão. Uma análise mais aprofundada pode ser realizada, variando o subconjunto de teste.

Exercise 3

Repeat exercise 2 but now using a MLP network.

O MLP foi implementado pela biblioteca do scikit-learn `MLP`. Para avaliar o histórico do treinamento, também foi utilizada a função [`partial_fit`], com o loop em torno das épocas sendo implementando manualmente. Além do histórico da acurácia, foi salvo o histórico da loss, que segue a equação da entropia cruzada, implementada pela biblioteca log_loss. Foram utilizados os mesmos parâmetros do experimento com o modelo Perceptron, além da configuração da rede `mlpConfig = (10,)` (uma camada escondida com 10 neurônios) e `batchSize = 4`.

Exercise 4

Compare Sigmoid and ReLU activation functions.

A seguir são mostrados os gráficos dos históricos de acurácia e perda para o o MLP com função de ativação Sigmoid. Note que, aparentemente, também há uma situação de overfitting dos modelos devido à distância de desempenho dos subconjuntos de treinamento e validação (alguns folds mais outro menos), tal qual ocorreu com o modelo Perceptron. Porém, as curvas estão mais suaves, e o treinamento se extendeu por mais épocas. Os modelos convergem para patamares de acurácia de validação satisfatórios, indicando também que houve aprendizado.

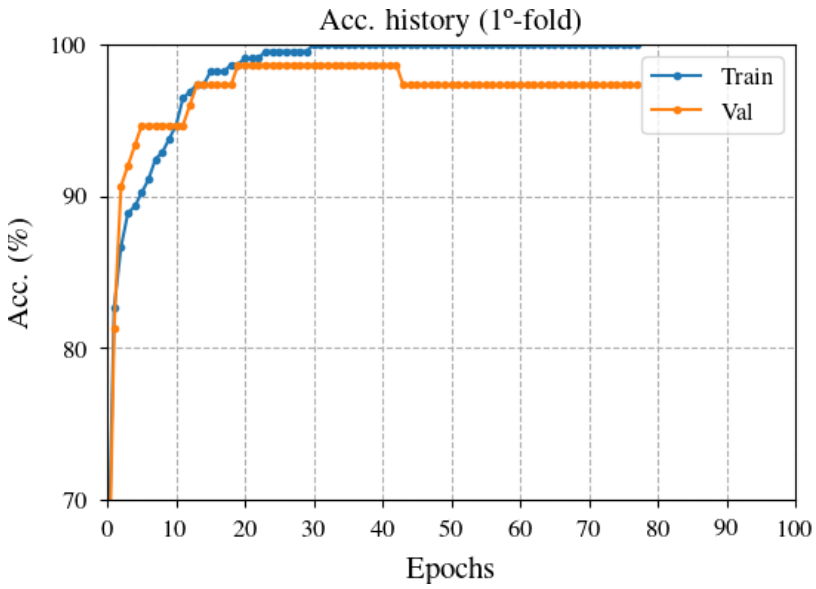

Figura 14: Histórico da acurácia para o MLP com função de ativação Sigmoid (1º-fold).

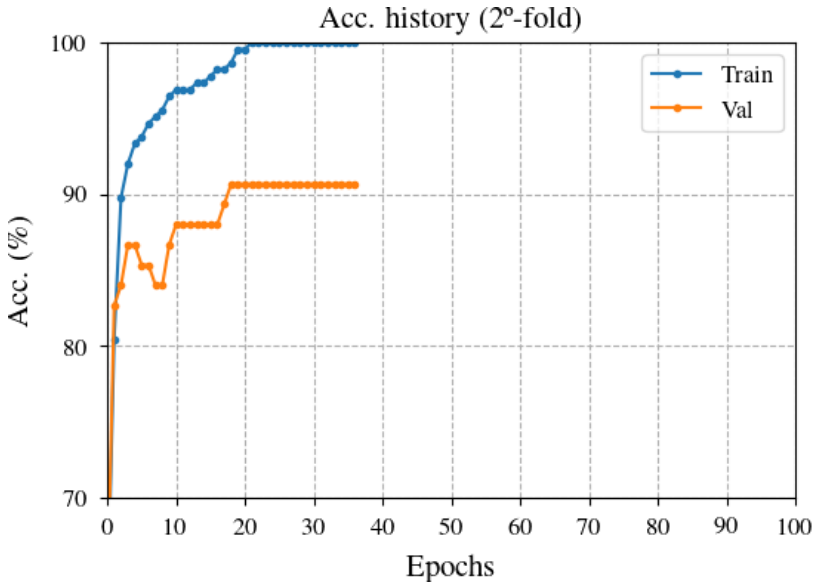

Figura 15: Histórico da acurácia para o MLP com função de ativação Sigmoid (2º-fold).

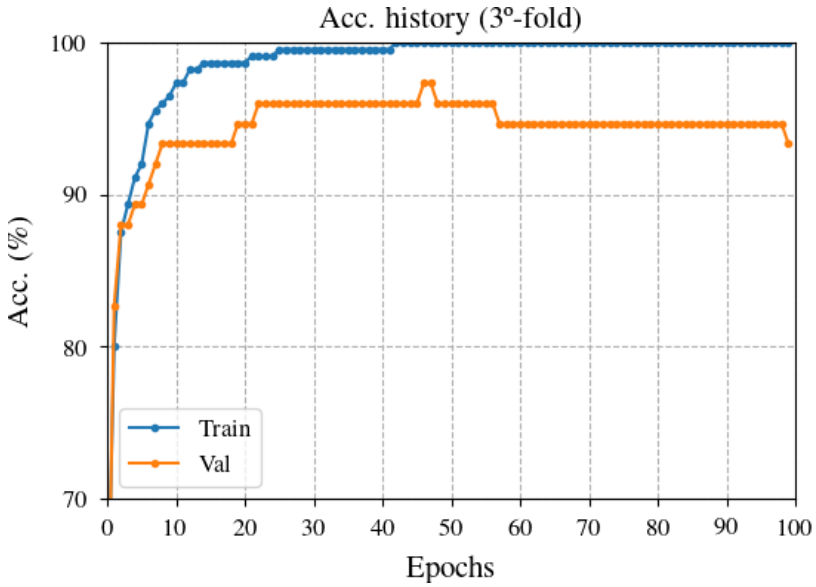

Figura 16: Histórico da acurácia para o MLP com função de ativação Sigmoid (3º-fold).

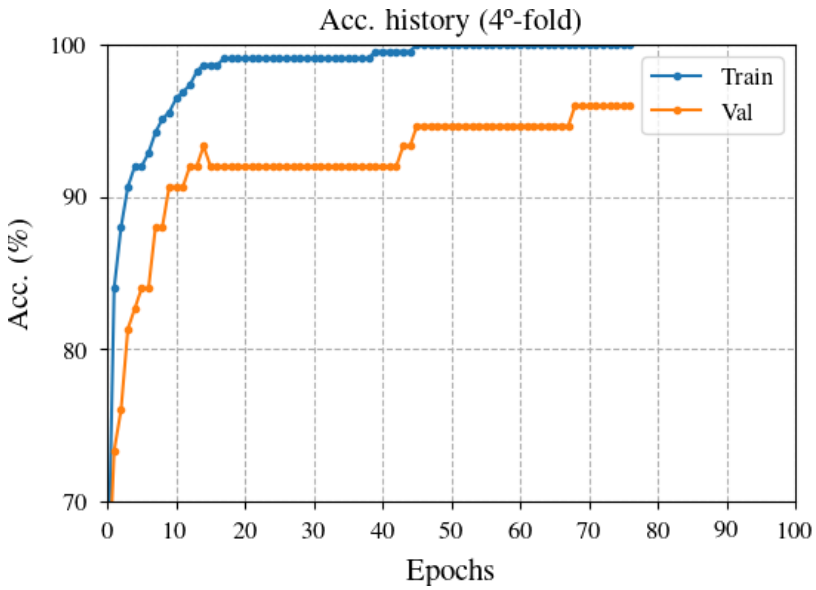

Figura 17: Histórico da acurácia para o MLP com função de ativação Sigmoid (4º-fold).

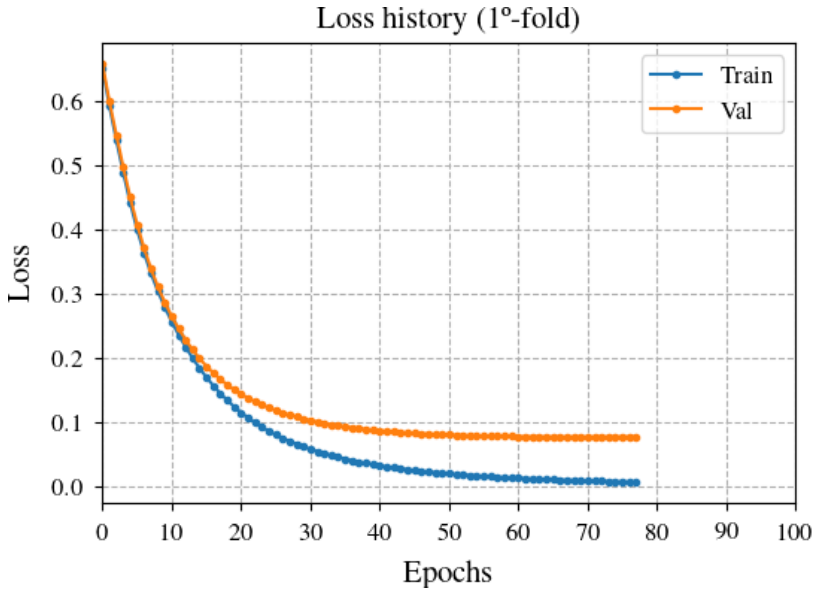

Figura 18: Histórico da loss para o MLP com função de ativação Sigmoid (1º-fold).

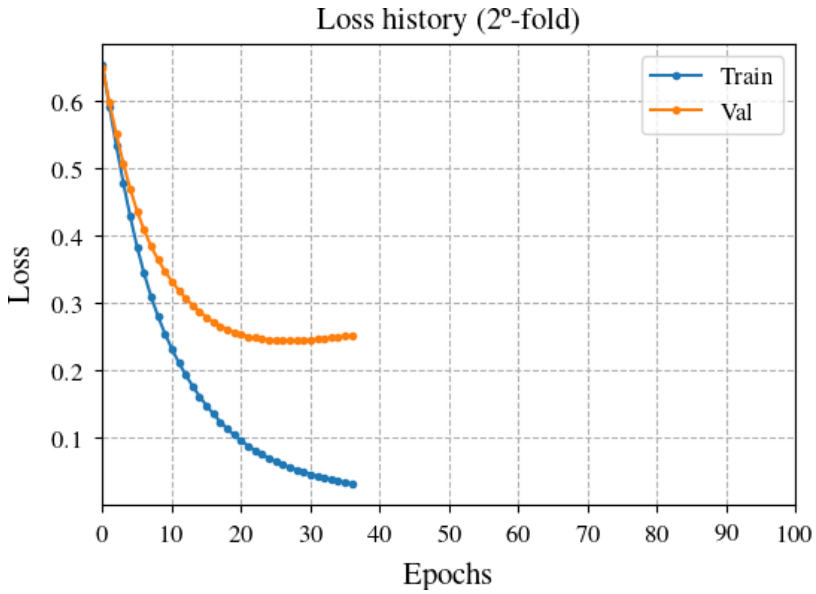

Figura 19: Histórico da loss para o MLP com função de ativação Sigmoid (2º-fold).

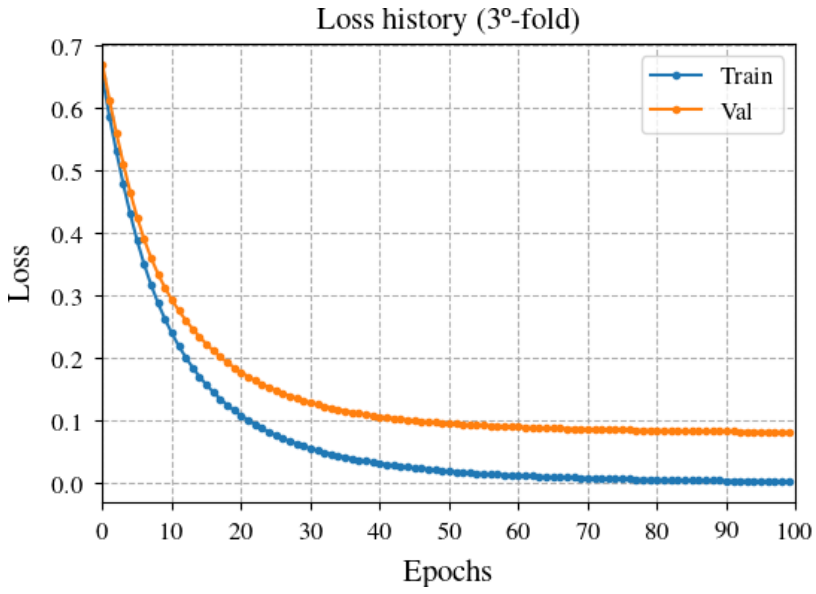

Figura 20: Histórico da loss para o MLP com função de ativação Sigmoid (3º-fold).

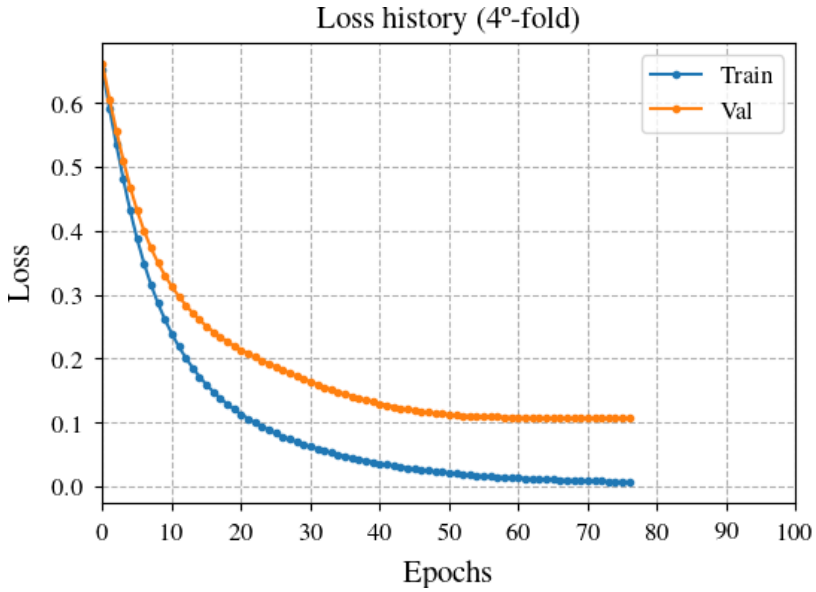

Figura 21: Histórico da loss para o MLP com função de ativação Sigmoid (4º-fold).

A seguir são mostrados os gráficos dos históricos de acurácia e perda para o o MLP com função de ativação ReLU. O comportamento não difere muito para a função de ativação Sigmoid.

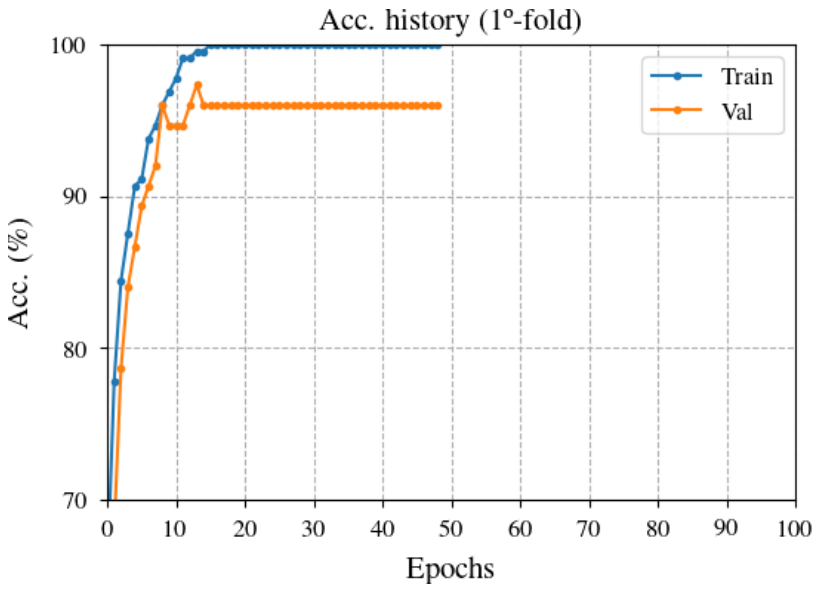

Figura 14: Histórico da acurácia para o MLP com função de ativação ReLU (1º-fold).

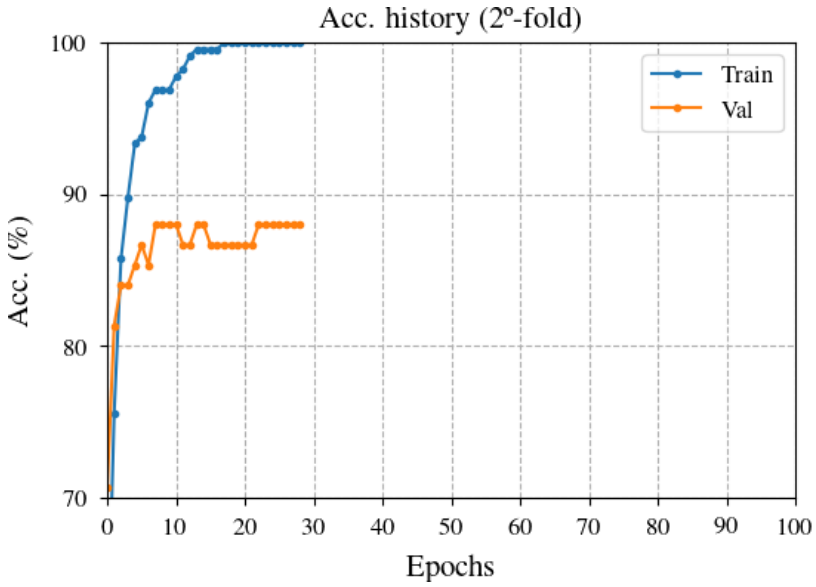

Figura 15: Histórico da acurácia para o MLP com função de ativação ReLU (2º-fold).

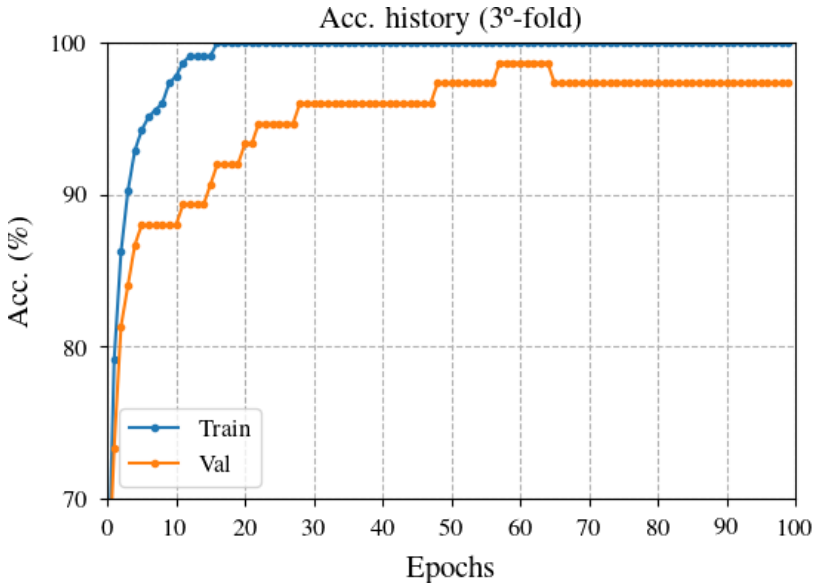

Figura 16: Histórico da acurácia para o MLP com função de ativação ReLU (3º-fold).

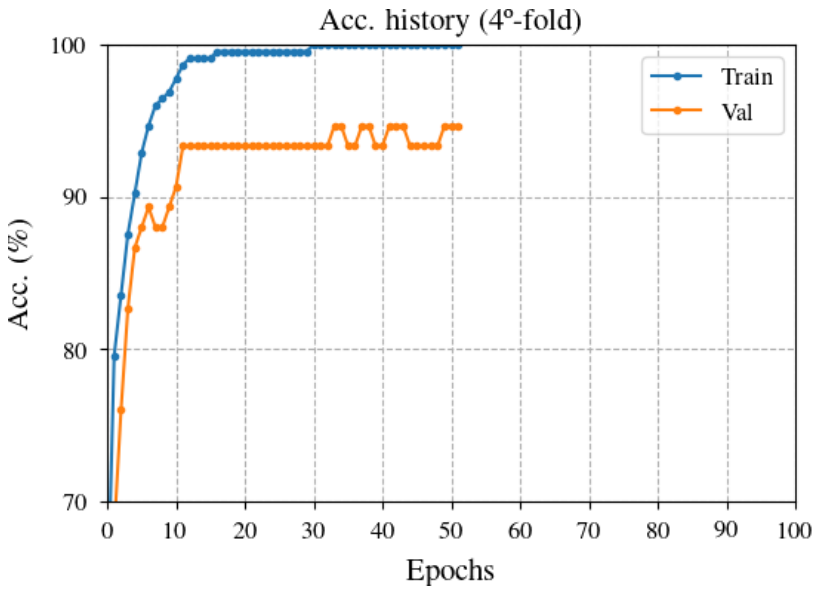

Figura 17: Histórico da acurácia para o MLP com função de ativação ReLU (4º-fold).

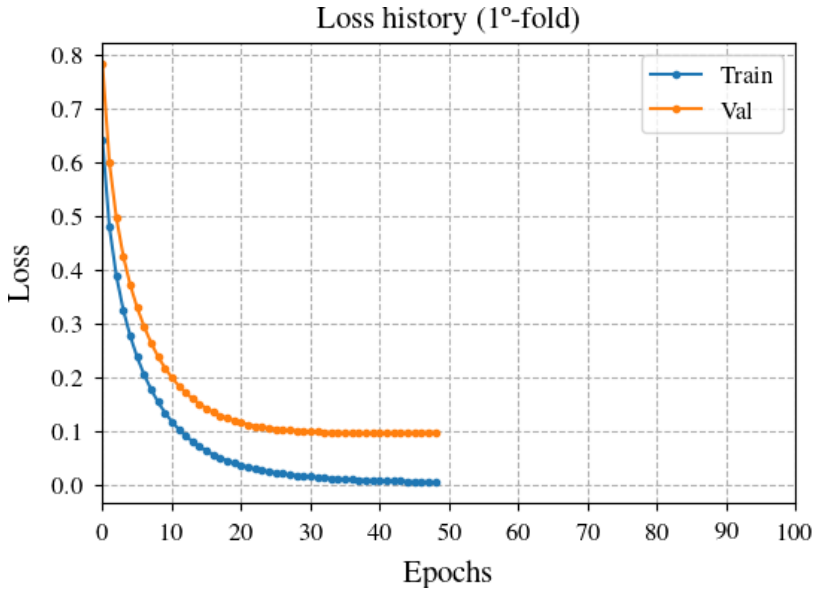

Figura 18: Histórico da loss para o MLP com função de ativação ReLU (1º-fold).

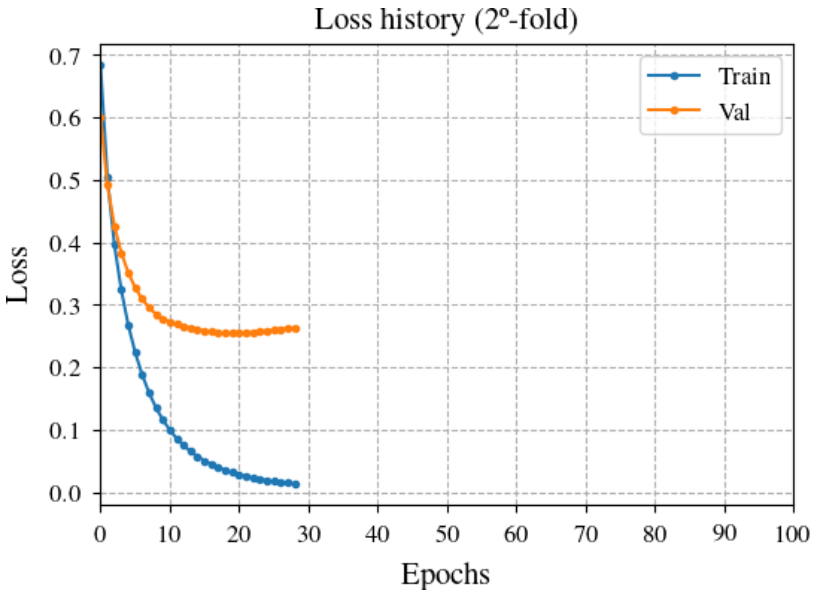

Figura 19: Histórico da loss para o MLP com função de ativação ReLU (2º-fold).

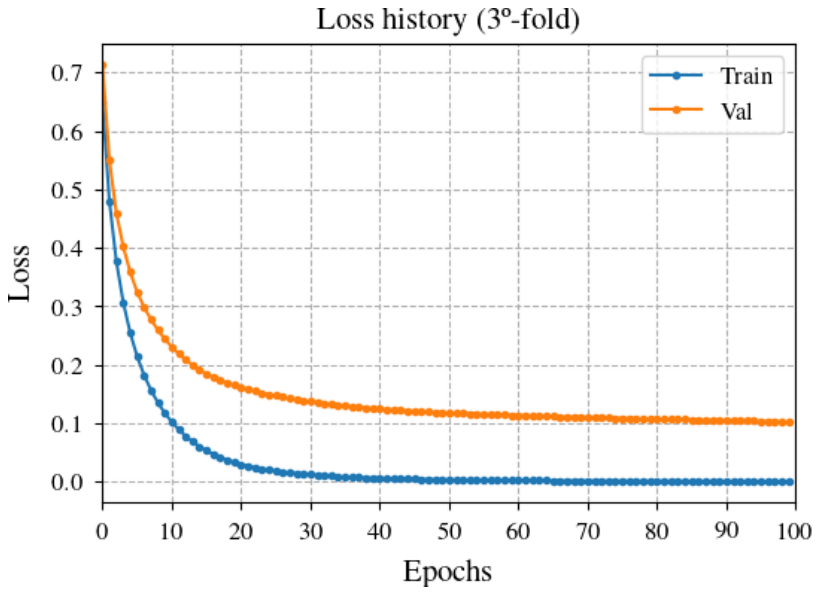

Figura 20: Histórico da loss para o MLP com função de ativação ReLU (3º-fold).

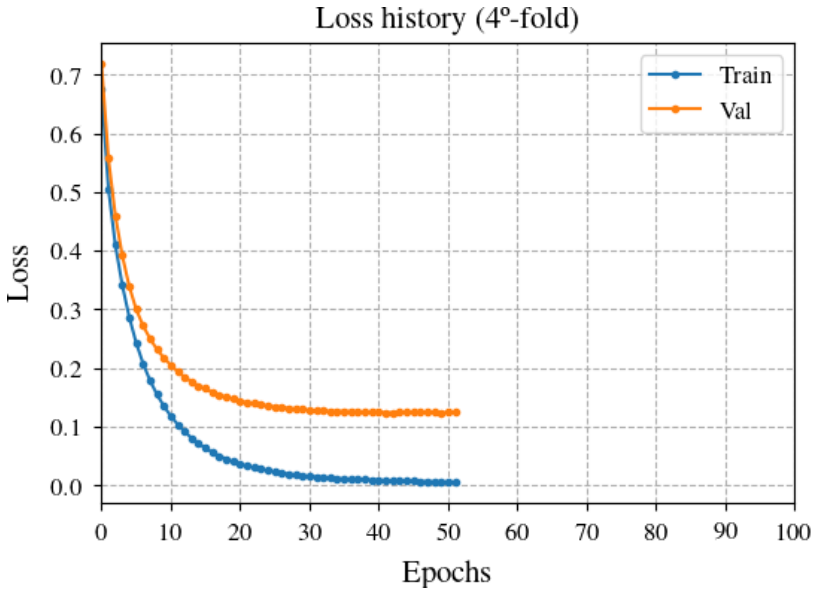

Figura 21: Histórico da loss para o MLP com função de ativação ReLU (4º-fold).

Utilizou-se a acurácia para analisar o desempenho dos modelos no subconjunto de teste.

- Modelo 1 (MLP Sigmoid): `Overall acc.: 100.00% | Man acc.: 100.00% | Woman acc.: 100.00%`
- Modelo 2 (MLP Sigmoid): `Overall acc.: 97.00% | Man acc.: 96.00% | Woman acc.: 98.00%`
- Modelo 3 (MLP Sigmoid): `Overall acc.: 98.00% | Man acc.: 98.00% | Woman acc.: 98.00%`
- Modelo 4 (MLP Sigmoid): `Overall acc.: 97.00% | Man acc.: 98.00% | Woman acc.: 96.00%`
- Modelo 1 (MLP ReLU): `Overall acc.: 97.00% | Man acc.: 98.00% | Woman acc.: 96.00%`
- Modelo 2 (MLP ReLU): `Overall acc.: 95.00% | Man acc.: 94.00% | Woman acc.: 96.00%`
- Modelo 3 (MLP ReLU): `Overall acc.: 98.00% | Man acc.: 98.00% | Woman acc.: 98.00%`
- Modelo 4 (MLP ReLU): `Overall acc.: 97.00% | Man acc.: 98.00% | Woman acc.: 96.00%`

Os resultados se demonstraram satisfatórios, com destaque para Modelo 1 obtendo 100% de acurácia. No geral, os resultados do MLP Sigmoid são iguais ou melhoes que o MLP ReLU, sendo ambos iguais ou superiores ao Perceptron.