

Universidade Federal do Rio de Janeiro  
Instituto Alberto Luiz Coimbra de  
Pós-Graduação e Pesquisa de Engenharia



Programa de Engenharia de Sistemas e  
Computação

CPS769 - Introdução à Inteligência Artificial e Aprendizagem Generativa

Prof. Dr. Edmundo de Souza e Silva (PESC/COPPE/UFRJ)

Profa. Dra. Rosa M. Leão (PESC/COPPE/UFRJ)

Participação Especial: Gaspare Bruno (Diretor Inovação, ANLIX)

*Lista de Exercícios 2*

Luiz Henrique Souza Caldas

email: lhscaldas@cos.ufrj.br

25 de julho de 2024

## Questão 1

O objetivo deste trabalho é entender como construir um modelo preditivo simples (de Redes Neurais) usando uma ou mais camadas ocultas, e se familiarizar com os códigos em Python. Nesta tarefa, você deverá construir um modelo de rede neural simples para prever a nota de cada aluno em uma turma com base em duas features:

- Fração de palestras assistidas
- Número de horas estudadas por semana (até o máximo de 8 horas em uma semana).

Sobre a Rede Neural e programa:

- A sua rede neural deverá ter uma camada oculta com 3 neurônios. Portanto teremos 2 entradas (as 2 features), uma camada oculta e, na camada de saída, um neurônio.
- Os dados de entrada são fornecidos em uma planilha .ods (libreoffice), com os dados de 500 estudantes.
- Uma vez lido, o seu dataset deve ser aleatoriamente dividido de forma a que 80% seja para treino do modelo e os restantes 20% para teste.
- No seu programa:

- Use um Scaler padrão do sklearn para dimensionar os dados de treinamento. Explique o motivo de escalonar os dados de entrada.

**Resposta:**

Escalonar os dados de entrada normaliza suas distribuições, permitindo uma convergência mais rápida e estável do modelo de machine learning.

- Para a camada oculta, use a função de ativação ReLU (Rectified Linear Unit).
- A camada de saída usa a ativação linear padrão (explique o que é).

**Resposta:**

A ativação linear padrão retorna a entrada sem modificações ( $f(x) = x$ ), usada em problemas de regressão para prever valores contínuos.

- Use o algoritmo de otimização Adam (Adaptive Moment Estimation). Explique bem resumidamente as vantagens em relação ao Stochastic Gradient Descent padrão.

**Resposta:**

Adam ajusta a taxa de aprendizagem para cada parâmetro individualmente e utiliza momentos dos gradientes, permitindo convergência mais rápida e estável.

- Use mean square error para a função de perda. Explique bem resumidamente o objetivo da função de perda.

**Resposta:**

A função de perda mede a diferença entre as previsões do modelo e os valores reais. Durante o aprendizado, o ajuste dos pesos é feito de forma a minimizar o valor da função de perda, melhorando a acurácia do modelo.

Responda:

1. Como o modelo de rede neural está estruturado? Explique a arquitetura.

**Resposta:**

O modelo possui duas entradas (as duas features), uma camada oculta com 3 neurônios usando a função de ativação ReLU, e uma camada de saída com um neurônio e ativação linear.

2. Explique o papel da função de ativação usada na camada oculta.

**Resposta:**

A função ReLU (Rectified Linear Unit) ajuda a introduzir não-linearidade ao modelo, permitindo que ele aprenda relações complexas entre as entradas e as saídas.

3. Treine o modelo com o conjunto de dados fornecido. Qual o erro quadrático médio nos dados de teste?

**Resposta:**

\*Exibir valor final\*

4. Trace o erro quadrático médio em função das “épocas” (dos passos para a convergência). Descreva a tendência que você observa.

**Resposta:**

\*Exibir gráfico final\*

5. Mostre os pesos e bias de cada camada após o treinamento, isto é, mostre os parâmetros aprendidos do modelo.

**Resposta:**

\*tabela com os pesos\*

6. Use o modelo treinado para prever as notas a partir dos dados de novos alunos (com um segundo dataset fornecido sem as notas). Mostre as previsões feitas pelo modelo e explique os resultados.

**Resposta:**

\*tabela com as previsões\*

7. Modifique o programa para adicionar mais camadas e/ou maior número de neurônios ocultos. Como suas modificações afetam o desempenho do modelo?

**Resposta:**

8. Quais seriam algumas melhorias potenciais ou recursos adicionais que poderiam ser adicionados ao modelo para melhorar sua precisão preditiva?

**Resposta:**

Para melhorar a precisão preditiva, poderia-se adicionar regularização (como Dropout), usar técnicas de aumento de dados, ou experimentar diferentes arquiteturas de rede.

9. Faria sentido usar a função de ativação sigmoid no modelo? Explique em poucas palavras.

**Resposta:**

A função sigmoid é mais adequada para tarefas de classificação binária. Para problemas de regressão, como o apresentado, a ativação linear na saída é mais apropriada.

## Código

O código abaixo encontra-se no repositório <https://github.com/lhscaldas/cps769-ai-gen>, bem como o arquivo LaTeX com o relatório.

**Código 1:** código fornecido completo com algumas modificações

```
1 import pandas as pd
2 import numpy as np
3 import tensorflow as tf
4 from sklearn.model_selection import train_test_split
5 from sklearn.preprocessing import StandardScaler
6 import matplotlib.pyplot as plt
7
8 class NeuralNetworkModel:
9     def __init__(self, data_path):
10         self.data_path = data_path
11         self.model = None
12         self.history = None
13         self.scaler = StandardScaler()
14
15     def load_data(self):
16         self.data = pd.read_csv(self.data_path, decimal=',', dtype=float)
17         self.X = self.data[['Presença', 'HorasEstudo']]
18         self.y = self.data['Nota']
19
20     def preprocess_data(self):
21         self.X_train, self.X_test, self.y_train, self.y_test = train_test_split(
22             self.X, self.y, test_size=0.2, random_state=42)
23         self.X_train_scaled = self.scaler.fit_transform(self.X_train)
24         self.X_test_scaled = self.scaler.transform(self.X_test)
25
26     def build_model(self):
27         self.model = tf.keras.Sequential([
28             tf.keras.layers.InputLayer(input_shape=(2,)),
29             tf.keras.layers.Dense(3, activation='relu'),
30             tf.keras.layers.Dense(1)
31         ])
32         self.model.compile(optimizer='adam', loss='mean_squared_error')
33
34     def train_model(self, epochs=600):
35         self.history = self.model.fit(self.X_train_scaled, self.y_train, epochs
36             =epochs, validation_split=0.2, verbose=0)
37
38     def evaluate_model(self):
39         test_loss = self.model.evaluate(self.X_test_scaled, self.y_test)
40         print(f'Mean Squared Error on Test Data: {test_loss}')
```

```

39
40 def plot_loss(self):
41     plt.plot(self.history.history['loss'], label='Train Loss')
42     plt.plot(self.history.history['val_loss'], label='Validation Loss')
43     plt.xlabel('Epochs')
44     plt.ylabel('Mean Squared Error')
45     plt.legend()
46     plt.show()
47
48 def show_weights_and_biases(self):
49     for layer in self.model.layers:
50         weights, biases = layer.get_weights()
51         print(f'Pesos da camada: {weights}')
52         print(f'Bias da camada: {biases}')
53
54 def predict_new_data(self, new_data_path):
55     new_data = pd.read_csv(new_data_path, decimal='.', dtype=float)
56     new_X = new_data[['Presença', 'HorasEstudo']]
57     new_X_scaled = self.scaler.transform(new_X)
58     predictions = self.model.predict(new_X_scaled, verbose=0)
59     new_data['Nota'] = predictions
60     new_data.to_csv('lista_2/lista_2-students_data_new.csv', index=False)
61     print(f'Previsões salvas no arquivo lista_2-students_data_new.csv')
62
63 def modify_and_train_new_model(self, new_layers, epochs=500):
64     modified_model = tf.keras.Sequential()
65     modified_model.add(tf.keras.layers.InputLayer(input_shape=(2,)))
66
67     for units in new_layers:
68         modified_model.add(tf.keras.layers.Dense(units, activation='relu'))
69
70     modified_model.add(tf.keras.layers.Dense(1))
71     modified_model.compile(optimizer='adam', loss='mean_squared_error')
72
73     modified_history = modified_model.fit(self.X_train_scaled, self.y_train
74         , epochs=epochs, validation_split=0.2, verbose=0)
75     modified_test_loss = modified_model.evaluate(self.X_test_scaled, self.
76         y_test)
77     print(f'Mean Squared Error on Test Data with Modified Model: {
78         modified_test_loss}')
79
80 if __name__ == "__main__":
81     data_path = 'lista_2/lista_2-students_data.csv'
82     new_data_path = 'lista_2/lista_2-students_data_new.csv'
83
84     nn_model = NeuralNetworkModel(data_path)
85     nn_model.load_data()
86     nn_model.preprocess_data()
87     nn_model.build_model()
88     nn_model.train_model()
89     nn_model.evaluate_model()
90     nn_model.plot_loss()
91     nn_model.show_weights_and_biases()
92     nn_model.predict_new_data(new_data_path)

```

90

```
nn_model.modify_and_train_new_model([5, 3])
```