

I. Pen-and-paper

1)

1

$$\phi(y_1, y_2) = y_1 \times y_2$$

$$x_1 \rightarrow \phi(y_1, y_2) = 1 \times 1 = 1$$

$$x_2 \rightarrow \phi(y_1, y_2) = 1 \times 3 = 3$$

$$x_3 \rightarrow \phi(y_1, y_2) = 3 \times 2 = 6$$

$$x_4 \rightarrow \phi(y_1, y_2) = 3 \times 3 = 9$$

$$x_5 \rightarrow \phi(y_1, y_2) = 2 \times 4 = 8$$

$$X = \begin{bmatrix} 1 & 1 \\ 1 & 3 \\ 1 & 6 \\ 1 & 9 \\ 1 & 8 \end{bmatrix} \quad X^T = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 3 & 6 & 9 & 8 \end{bmatrix} \quad X^T X = \begin{bmatrix} 5 & 27 \\ 27 & 191 \end{bmatrix}$$

$$(X^T X)^{-1} = \begin{bmatrix} 0,84513 & -0,11947 \\ -0,11947 & 0,02212 \end{bmatrix}$$

$$(X^T X)^{-1} X^T = \begin{bmatrix} 0,72566 & 0,48673 & 0,12832 & -0,23009 & -0,11062 \\ -0,09735 & -0,05310 & 0,01372 & 0,07965 & 0,05752 \end{bmatrix}$$

$$y_{nm} = \begin{bmatrix} 1,25 \\ 7 \\ 2,7 \\ 3,2 \\ 5,5 \end{bmatrix}$$

OLS:

$$w = (X^T X)^{-1} X^T y_{nm} = (X^T X)^{-1} X^T \begin{bmatrix} 1,25 \\ 7 \\ 2,7 \\ 3,2 \\ 5,5 \end{bmatrix} = \begin{bmatrix} 3,31593 \\ 0,11372 \end{bmatrix}$$

Logo a equação do modelo de regressão para ter y_{nm} a partir de $\phi(y_1, y_2)$:

$$\phi(y_1, y_2) :$$

$$\hat{y}_{nm} = 3,31593 + 0,11372 \phi(y_1, y_2)$$

2)

2

$$w_{\text{ridge}} = (X^T X + \lambda I)^{-1} X^T y_{\text{num}}$$

$$(X^T X + \lambda I) = \begin{bmatrix} 6 & 27 \\ 27 & 192 \end{bmatrix} \quad (X^T X + \lambda I)^{-1} = \begin{bmatrix} 0,45390 & -0,06383 \\ -0,06383 & 0,01918 \end{bmatrix}$$

$$(X^T X + \lambda I)^{-1} X^T = \begin{bmatrix} 0,39007 & 0,26241 & 0,07092 & -0,12067 & -0,05674 \\ -0,09965 & -0,0218 & 0,0213 & 0,06383 & 0,04965 \end{bmatrix}$$

$$w_{\text{ridge}} = (X^T X + \lambda I)^{-1} X^T \begin{bmatrix} 1,28 \\ 7 \\ -2,7 \\ 3,2 \\ 5,5 \end{bmatrix} = \begin{bmatrix} 1,81809 \\ 0,32376 \end{bmatrix}$$

com a regressão de Ridge com $\lambda=1$, a equação não teria termo $\phi(y_1, y_2)$.
 Continua de $\phi(y_1, y_2)$:

$$\hat{y}_{\text{num}, 2} = 1,81809 + 0,32376 \phi(y_1, y_2)$$

Na OLS, o bias era 3,31593, enquanto na regressão Ridge, ele reduziu significativamente para 1,81809. Relativamente ao peso, na regressão OLS é 0,11372 e na regressão Ridge é 0,32376, existindo um ligeiro aumento. A regularização Ridge evita que os coeficientes assumam valores muito altos, para reduzir o overfitting. Neste caso, a regularização fez com que o modelo ajustasse os coeficientes de forma mais moderada para obter uma melhor generalização, resultando em coeficientes com menor magnitude para o bias e maior ênfase no peso.

3)

3

$$x_6 \rightarrow \phi(y_1, y_2) = 2 \times 2 = 4$$

$$x_7 \rightarrow \phi(y_1, y_2) = 1 \times 2 = 2$$

$$x_8 \rightarrow \phi(y_1, y_2) = 5 \times 1 = 5$$

$$X_{\text{train}} = \begin{bmatrix} 1 & 1 \\ 1 & 3 \\ 1 & 6 \\ 1 & 9 \\ 1 & 8 \end{bmatrix}$$

$$X_{\text{test}} = \begin{bmatrix} 1 & 4 \\ 1 & 2 \\ 1 & 5 \end{bmatrix}$$

~~test~~ observações teste: x_6, x_7, x_8 1 → OLS

observações train: x_1, x_2, x_3, x_4, x_5 2 → Ridge

$$\hat{y}_{\text{num}, 1} = X_{\text{train}} w = \begin{bmatrix} 1 & 1 \\ 1 & 3 \\ 1 & 6 \\ 1 & 9 \\ 1 & 8 \end{bmatrix} \begin{bmatrix} 3,31593 \\ 0,11372 \end{bmatrix} = \begin{bmatrix} 3,92965 \\ 3,65709 \\ 3,99825 \\ 4,33941 \\ 4,22569 \end{bmatrix}$$

$$\hat{y}_{\text{num}, 2} = X_{\text{train}} w_{\text{ridge}} = \begin{bmatrix} 1 & 1 \\ 1 & 3 \\ 1 & 6 \\ 1 & 9 \\ 1 & 8 \end{bmatrix} \begin{bmatrix} 1,81909 \\ 0,32376 \end{bmatrix} = \begin{bmatrix} 2,14185 \\ 2,78937 \\ 3,76065 \\ 4,73193 \\ 4,90817 \end{bmatrix}$$

$$\hat{Y}_{\text{num},1} = X_{\text{test}} W = \begin{bmatrix} 1 & 4 \\ 1 & 2 \\ 1 & 5 \end{bmatrix} \begin{bmatrix} 3,31593 \\ 0,11372 \end{bmatrix} = \begin{bmatrix} 3,77081 \\ 3,54337 \\ 3,88453 \end{bmatrix}$$

$$\hat{Y}_{\text{num},2} = X_{\text{test}} W_{\text{ridge}} = \begin{bmatrix} 1 & 4 \\ 1 & 2 \\ 1 & 5 \end{bmatrix} \begin{bmatrix} 1,81809 \\ 0,32376 \end{bmatrix} = \begin{bmatrix} 3,11313 \\ 2,46561 \\ 3,43689 \end{bmatrix}$$

$$Y_{\text{test}} = Y_{\text{num}} = \begin{bmatrix} 1,25 \\ 7 \\ 2,7 \\ 3,2 \\ 5,5 \end{bmatrix} \quad Y_{\text{num}} = \begin{bmatrix} 0,7 \\ 1,1 \\ 2,2 \end{bmatrix}$$

$$\text{RMSE}_{\text{ols}} = \sqrt{\frac{(1,25 - 3,42965)^2 + (7 - 3,65709)^2 + (2,7 - 3,99825)^2 + *}{5}} \\ * (3,2 - 9,33991)^2 + (5,5 - 4,22569)^2 = 2,02650$$

Usando o mesmo raciocínio para o teste:

$$\text{RMSE}_{\text{OLS}} = 2,46560$$

$$\text{RMSE}_{\text{Ridge}} = 2,15354$$

$$\text{RMSE}_{\text{ridge}} = 1,75290$$

Os resultados indicam que, quanto ao conjunto de treino, o RMSE da regressão de Ridge é pior em comparação com o OLS (que já era esperado), pois a regularização reduz o ajuste aos dados de treino para evitar o overfitting. No entanto, para o conjunto de teste, o RMSE é menor na regressão de Ridge do que para OLS, ou seja, a regularização melhorou a capacidade de generalização do modelo, resultando em previsões mais precisas.

4)

$$\begin{aligned}
 & W^{[2]} = \begin{bmatrix} 0,1 & 0,7 \\ 0,7 & 0,2 \\ 0,2 & 0,1 \end{bmatrix} \quad b^{[2]} = \begin{bmatrix} 0,1 \\ 0 \\ 0,1 \end{bmatrix} \quad W^{[1]} = \begin{bmatrix} 1 & 2 & 2 \\ 1 & 2 & 2 \\ 1 & 1 & 1 \end{bmatrix} \quad b^{[1]} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \\
 & \text{Forward Propagation} \quad X^{[0]} \rightarrow Z^{[1]} \rightarrow X^{[1]} \rightarrow Z^{[2]} \rightarrow \\
 & \qquad \qquad \qquad \rightarrow \cancel{X^{[2]}} \rightarrow Z^{[\text{out}]} \rightarrow \cancel{Z^{[\text{out}]}} \\
 & Z^{[1]} = W^{[1]} \times X^{[0]} + b^{[1]} = \begin{bmatrix} 0,1 & 0,1 \\ 0,2 & 0,2 \\ 0,2 & 0,1 \end{bmatrix}_{3 \times 2} \begin{bmatrix} 1 \\ 1 \end{bmatrix}_{2 \times 1} + \begin{bmatrix} 0,1 \\ 0 \\ 0,1 \end{bmatrix} = \begin{bmatrix} 0,3 \\ 0,3 \\ 0,4 \end{bmatrix} \\
 & X^{[1]} = Z^{[1]} \rightarrow \text{Sem função de ativação} \\
 & Z^{[2]} = W^{[2]} X^{[1]} + b^{[2]} = \begin{bmatrix} 1 & 2 & 2 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0,3 \\ 0,3 \\ 0,4 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2,7 \\ 2,3 \\ 2 \end{bmatrix}
 \end{aligned}$$

$$X^{[2]} = \text{softmax}(Z^{[2]}) = \begin{bmatrix} 0,46749 \\ 0,30934 \\ 0,22977 \end{bmatrix} \rightarrow A \\ \rightarrow B \\ \rightarrow C$$

Backward Propagation

$$W^{[2] \text{ new}} = W^{[2] \text{ old}} - \eta \times \frac{\partial E}{\partial W^{[2]}} \quad b^{[2] \text{ new}} = b^{[2] \text{ old}} - \eta \times \frac{\partial E}{\partial b^{[2]}}$$

$$\frac{\partial E}{\partial W^{[2]}} = \underbrace{\frac{\partial E}{\partial X^{[2]}}}_{\delta^2} \circ \underbrace{\frac{\partial X^{[2]}}{\partial Z^{[2]}}} \times \underbrace{\left(\frac{\partial Z^{[2]}}{\partial W^{[2]}} \right)^T}_{(X^{[2]})^T}$$

$$\frac{\partial E}{\partial b^{[2]}} = \delta^2$$

$$\delta^2 = X^{[2]} - t \quad (\text{Último layer e Classificação})$$

$$= \begin{bmatrix} 0,46749 \\ 0,30934 \\ 0,22977 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0,46749 \\ -0,69066 \\ 0,22977 \end{bmatrix}$$

$$\frac{\partial E}{\partial W^{[1]}} = \begin{bmatrix} 0,46749 \\ -0,69066 \\ 0,22977 \end{bmatrix} \times \begin{bmatrix} 0,3 \\ 0,3 \\ 0,4 \end{bmatrix}^T =$$

$$= \begin{bmatrix} 0,73845 & 0,73845 & 0,784996 \\ -0,207798 & -0,207720 & -0,27626 \\ 0,06875 & 0,06875 & 0,09767 \end{bmatrix}$$

$$W^{[2] \text{ new}} = \begin{bmatrix} 1 & 2 & 2 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} - 0,7 \begin{bmatrix} 0,73845 & 0,73845 & 0,784996 \\ -0,207720 & -0,207720 & -0,27626 \\ 0,06875 & 0,06875 & 0,09767 \end{bmatrix}$$

$$= \begin{bmatrix} 0,98676 & 1,98676 & 1,98754 \\ 1,02072 & 2,02072 & 2,02763 \\ 0,99373 & 0,99373 & 0,99083 \end{bmatrix}$$

$$b^{[2] \text{ new}} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - 0,7 \begin{bmatrix} 0,46749 \\ -0,69066 \\ 0,22977 \end{bmatrix} = \begin{bmatrix} 0,95385 \\ 0,06907 \\ 0,97708 \end{bmatrix}$$

$$\begin{aligned}
 w^{[1]_{\text{new}}} &= w^{[1]_{\text{old}}} - h \times \frac{\partial E}{\partial w^{[1]}} \quad / \quad b^{[1]_{\text{new}}} = b^{[1]_{\text{old}}} - h \times \frac{\partial E}{\partial b^{[1]}} \\
 \frac{\partial E}{\partial w^{[1]}} &= S^{[1]} \times (x^{[0]})^T \\
 S^{[1]} &= (w^{[2]})^T \times S^{[2]} \circ \frac{\partial x^{[1]}}{\partial z^{[1]}} \\
 x^{[1]} &= z^{[0]} \Leftrightarrow \frac{\partial x^{[1]}}{\partial z^{[1]}} = \frac{\partial z^{[1]}}{\partial z^{[0]}} = 1 \\
 S^{[2]} &= \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 1 \\ 2 & 1 & 1 \end{bmatrix}_{3 \times 3} \times \begin{bmatrix} 0,46749 \\ -0,62066 \\ 0,22977 \end{bmatrix}_{3 \times 1} = \begin{bmatrix} -0,22977 \\ 7,07402 \\ 0,46749 \end{bmatrix}_{3 \times 1} \\
 w^{[1]_{\text{new}}} &= \begin{bmatrix} 0,1 & 0,1 \\ 0,1 & 0,2 \\ 0,2 & 0,1 \end{bmatrix}_{3 \times 2} - 0,7 \left(\begin{bmatrix} -0,22977 \\ 7,07402 \\ 0,46749 \end{bmatrix}_{3 \times 1} \times \begin{bmatrix} 1 & 1 \end{bmatrix}_{1 \times 2} \right)_{3 \times 2} \\
 &= \begin{bmatrix} 0,1 & 0,1 \\ 0,1 & 0,2 \\ 0,2 & 0,1 \end{bmatrix}_{3 \times 2} - \begin{bmatrix} -0,02292 & -0,02292 \\ 0,70740 & 0,70740 \\ 0,04675 & 0,04675 \end{bmatrix}_{3 \times 2} \\
 &= \begin{bmatrix} 0,12292 & 0,12292 \\ -0,0074 & 0,0986 \\ 0,75385 & 0,05385 \end{bmatrix}_{3 \times 2} \\
 b^{[1]_{\text{new}}} &= \begin{bmatrix} 0,1 \\ 0 \\ 0,1 \end{bmatrix} - 0,7 \begin{bmatrix} -0,22977 \\ 7,07402 \\ 0,46749 \end{bmatrix} = \begin{bmatrix} 0,12292 \\ -0,70740 \\ 0,05385 \end{bmatrix}
 \end{aligned}$$

II. Programming and critical analysis

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

data = pd.read_csv('parkinsons.csv')
x = data.drop('target', axis=1)
y = data['target']
```

5)

```
import warnings
from sklearn.neural_network import MLPRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error
from sklearn.linear_model import LinearRegression
from sklearn.exceptions import ConvergenceWarning
warnings.filterwarnings("ignore", category=ConvergenceWarning)

mlp1_mae_all = []
mlp2_mae_all = []
mae_linear = []

for i in range(1, 11):
    x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=i)

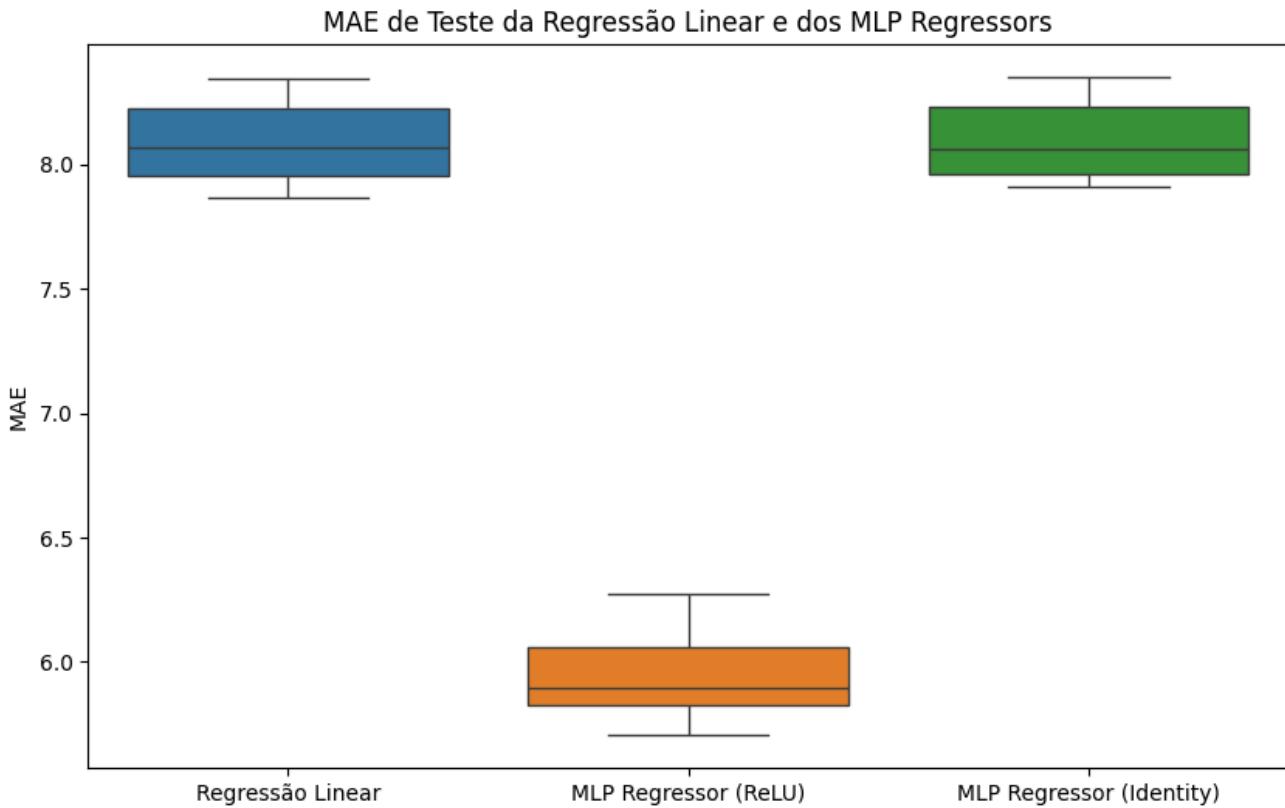
    lin_reg = LinearRegression()
    lin_reg.fit(x_train, y_train)
    y_pred = lin_reg.predict(x_test)
    mae_linear.append(mean_absolute_error(y_test, y_pred))

    mlp1 = MLPRegressor(hidden_layer_sizes=(10, 10), activation= 'relu', random_state=0)
    mlp1.fit(x_train, y_train)
    y_pred = mlp1.predict(x_test)
    mlp1_mae_all.append(mean_absolute_error(y_test, y_pred))

    mlp2 = MLPRegressor(hidden_layer_sizes=(10, 10), activation= 'identity', random_state=0)
    mlp2.fit(x_train,y_train)
    y_pred = mlp2.predict(x_test)
    mlp2_mae_all.append(mean_absolute_error(y_test, y_pred))

mae_data = pd.DataFrame({
    'Regressão Linear': mae_linear,
    'MLP Regressor (ReLU)': mlp1_mae_all,
    'MLP Regressor (Identity)': mlp2_mae_all
})

plt.figure(figsize=(10, 6))
sns.boxplot(data=mae_data)
plt.title('MAE de Teste da Regressão Linear e dos MLP Regressors')
plt.ylabel('MAE')
plt.show()
```



6)

Observando os resultados do exercício anterior, vemos que o MLP com ativação ReLU tem um MAE de ~6 enquanto a Regressão linear e o MLP sem ativação têm MAE semelhantes, próximas de ~8. A regressão linear e o MLP sem ativações têm erros muito próximos, pois sem funções de ativação, o comportamento da MLP torna-se muito semelhante ao de uma regressão linear. A ausência de não linearidade impede o MLP de encontrar padrões de informação mais complexos.

O MAE diminui bastante quando introduzimos uma função de ativação ReLU. O uso de funções de ativação é importante, pois introduz não linearidade no modelo, permitindo ao mesmo aprender padrões de informação mais complexos e adaptar-se melhor aos dados, diminuindo o erro.

7)

```

from sklearn.model_selection import GridSearchCV
import numpy as np

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)

param_grid = {
    'alpha': [0.0001, 0.001, 0.01],
    'learning_rate_init': [0.001, 0.01, 0.1],
    'batch_size': [32, 64, 128]
}

mlp = MLPRegressor(hidden_layer_sizes=(10, 10), random_state=0)
grid_search = GridSearchCV(estimator=mlp, param_grid=param_grid, scoring='neg_mean_absolute_error', cv=5)
grid_search.fit(x_train, y_train)

best_model = grid_search.best_estimator_

y_pred = best_model.predict(x_test)
mae = mean_absolute_error(y_test, y_pred)

results = grid_search.cv_results_

param_combinations = [f'L2: {params["alpha"]}, LR: {params["learning_rate_init"]}, Batch: {params["batch_size"]}' for params in results['params']]

mae_scores = -results['mean_test_score']

mae_data = pd.DataFrame({
    'Parameters': param_combinations,
    'Mean Test Score': mae_scores
})

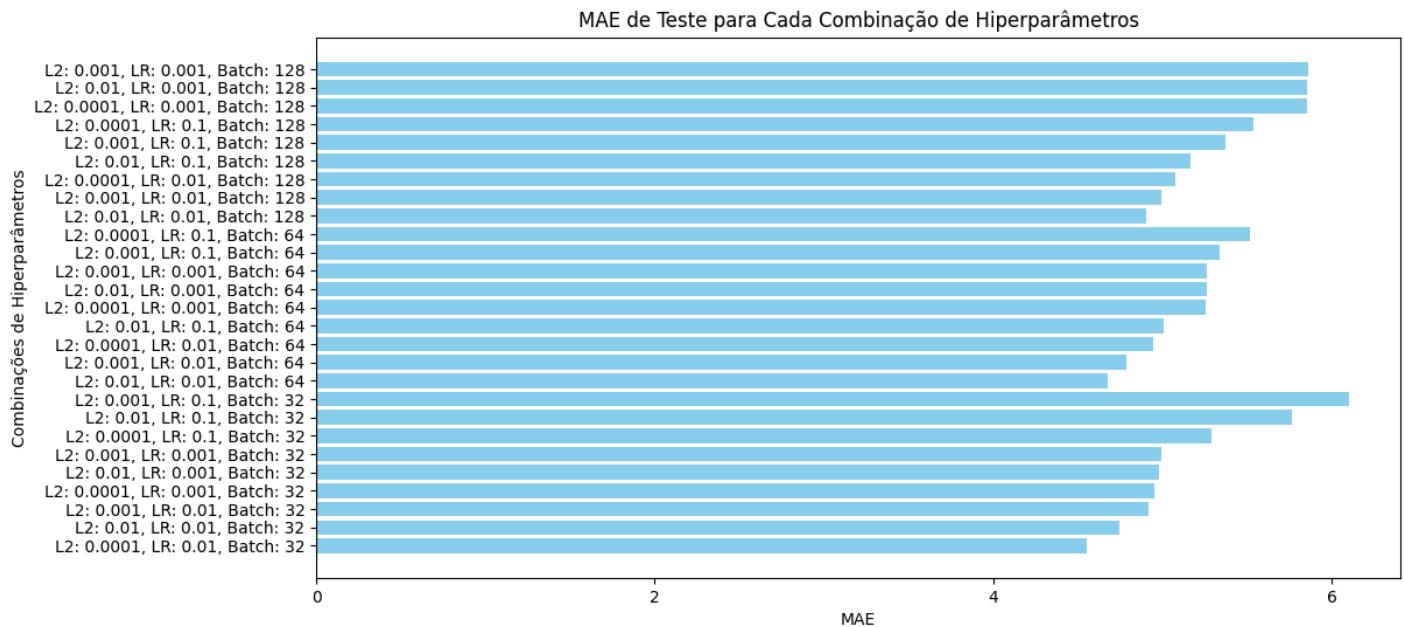
mae_data['Batch Size'] = [int(params.split(', ')[2].split(':')[1]) for params in mae_data['Parameters']]

mae_data_sorted = mae_data.sort_values(by=['Batch Size', 'Mean Test Score'])

plt.figure(figsize=(12, 6))
plt.barchart(mae_data_sorted['Parameters'], mae_data_sorted['Mean Test Score'], color='skyblue')
plt.xlabel('MAE')
plt.ylabel('Combinações de Hiperparâmetros')
plt.title('MAE de Teste para Cada Combinação de Hiperparâmetros')
plt.xticks(np.arange(0, 7, 2))
plt.show()

print(f"Melhor combinação: {grid_search.best_params_} MAE: {-grid_search.best_score_}")

```



Melhor combinação: {'alpha': 0.0001, 'batch_size': 32, 'learning_rate_init': 0.01} MAE: 4.555817948029912

Trade offs

L2 - penalty: Para valores baixos (0.0001) a regularização dos pesos é baixa e por isso o modelo corre o risco de ficar *overfitted* aos dados de treino. Aumentando a regularização os pesos mais altos deixam de ter tanto impacto e o modelo torna-se mais flexível, embora corra o risco de *underfitting*.

Batch size: Para Batchs maiores, as atualizações tornam-se mais estáveis e confiáveis, já que o treino é mais rápido, mas também pode levar a convergência em mínimos locais, o que reduz a capacidade de generalização do modelo. Em Batchs menores as atualizações são mais frequentes, o que ajuda a escapar aos mínimos locais e melhora a generalização do modelo, mas o treino pode ser mais instável, devido uma maior variância dos dados.

Learning Rate: Para valores mais baixos de LR a convergência dos pesos será mais suave, no entanto também será mais lenta, podendo nunca convergir. Aumentando a LR a convergência será mais errática, com os valores dos pesos a oscilar mais, correndo o risco de ultrapassar os pesos ideias.

A nossa combinação ideal usa o *L2 penalty* e *batch size* mais baixos e o learning rate intermédio.

END