Grupo 2 - Classificador logístico multiclasse: OvO vs ECOC

Otimização em Machine Learning

Guilherme Martins, PG52214
Faustino Sachimuco, PG50944
José Virgílio Loureiro, PG52252
João Fão Meira, PG52251

Mestrado em Matemática e Computação
Orientador: Gaspar Machado

Conteúdo

Introdução

- Análise e comparação das técnicas OvO e ECOC (Primal e Dual com ou sem Kernel) aplicadas ao classificador logístico multi-classe.
- Objetivo: Determinar a eficácia de cada técnica em diferentes contextos.
- Descrição de cada técnica e das metodologias aplicadas, seguida de validação e avaliação.

Classificadores Logísticos Binários

- Classificador binário: Algoritmo de machine learning para categorizar dados em duas classes (+1 e 0).
- Função Sigmóide:

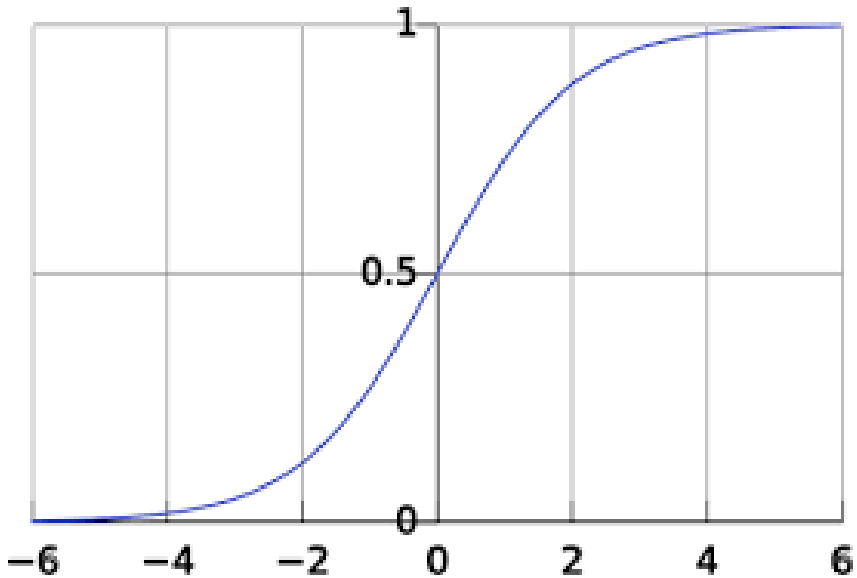


Figure: Representação gráfica da função sigmóide

Algoritmo CLogP-MGmB (Primal Mini-Batch)

Input: $(E(x) = \frac{1}{N}\sum_{n=1}^{N} E_n(x), E_n(x) : \mathbb{R}^M \rightarrow \mathbb{R}), rE_1, \ldots, rE_N, x(0) \in \mathbb{R}^M, \eta \in \mathbb{R}^+, B \in \mathbb{N}, CP$
Output: $x^* \in \mathbb{R}^M$

```
1   t = 0;
2   while V do
3   Gerar subconjunto índices D(t) ⊆ {1, ..., N};
4       s(t) = (1/B) Σ_{n∈D(t)} rE_n(x(t));
5       w̃(t+1) = w̃(t) − ηs(t);
6       if CP = V then
7           w̃* = w̃(t+1); return w̃*;
8       else
9           t = t + 1;
```

- Dependendo do tamanho da batch, pode ser **Estocástico** (#batch = 1) ou **Batch** (#batch = #dataset). Se a escolha dos elementos da base de dados for sequencial, chamamos de **Estocástico Ordenado**.

Algoritmo CLogDKPd-MGmB (Dual Mini-Batch Kernel)

Input: $D = \{(x^n, y^n)\}_{n=1}^{N}$, $x^n \in \mathbb{R}^I$, $y^n \in \{0, 1\}$, $\eta \in \mathbb{R}^+$, $d \in \mathbb{N}$
Output: $\boldsymbol{\alpha}^* \in \mathbb{R}^N$

$t = 0$;
$\boldsymbol{\alpha}(0) = (0, \ldots, 0) >\in \mathbb{R}^N$;
while V do
    Selecionar aleatoriamente M elementos distintos
    $\hat{p}^n = \sigma\left(\sum_{l=1}^{N} \alpha_{(t),l} (\tilde{x}^l \cdot \tilde{x}^n)^d\right)$, $n \in M$;
    $s_{(t)} = \frac{1}{M} \sum_{n=1}^{M} (\hat{p}^n - y^n) \left((\tilde{x}^l \cdot \tilde{x}^n)^d\right)$, $l = 1; \ldots; N$;
    $\boldsymbol{\alpha}_{(t+1)} = \boldsymbol{\alpha}_{(t)} - \eta s_{(t)}$;
    if $CP = V$ then
        $\boldsymbol{\alpha}^* = \boldsymbol{\alpha}_{(t+1)}$; **return** $\boldsymbol{\alpha}^*$;
    else
        $t = t + 1$;

- Dependendo do tamanho da batch, pode ser **Estocástico** (#batch = 1) ou **Batch** (#batch = #dataset). Se a escolha dos elementos for sequencial: **Estocástico Ordenado**.

Aplicação nas bases de dados 'AND' e 'XOR'

- Testes em bases de dados simples para verificar a eficácia dos algoritmos.
- Observações: AND é linearmente separável, XOR não é.

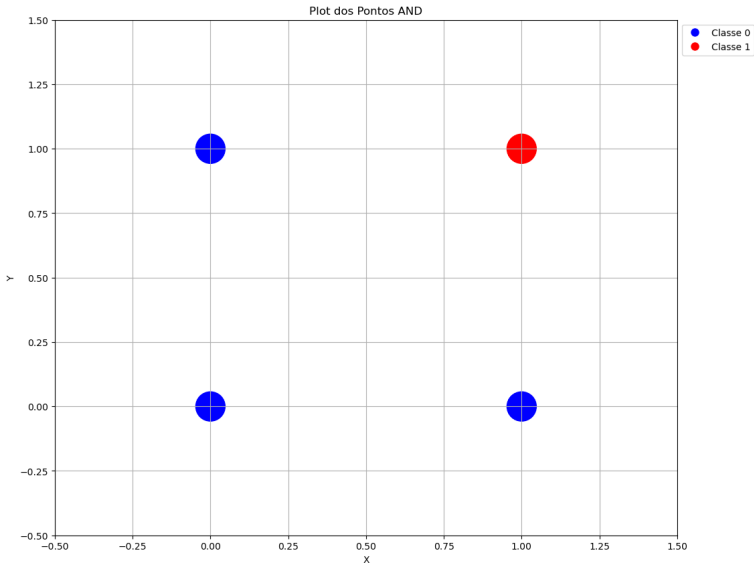


Figure: Base Dados AND

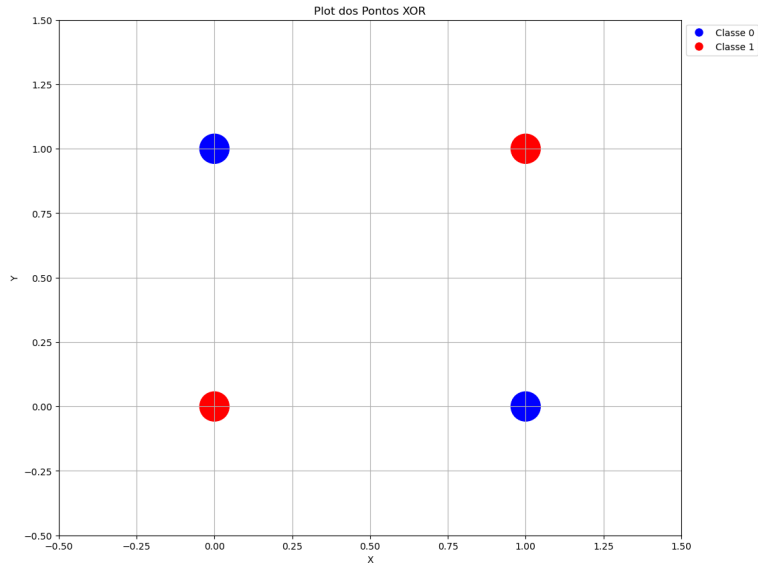


Figure: Base Dados XOR

Aplicação nas bases de dados 'AND' e 'XOR'
Gráficos de Erro

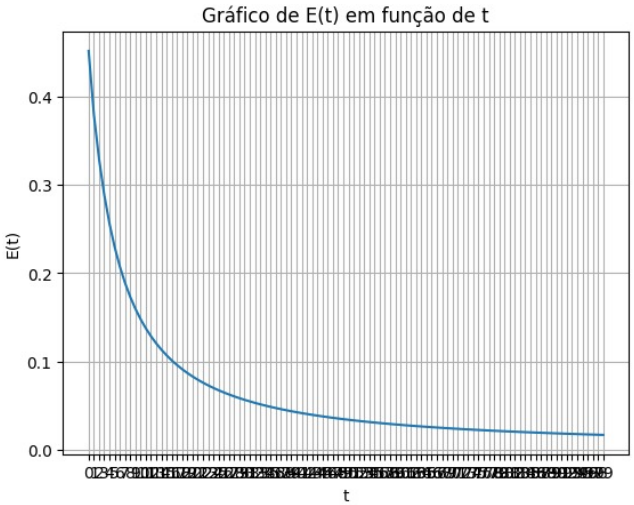


Figure: Gráfico da evolução do Erro ao longo do treino dual da base de dados AND

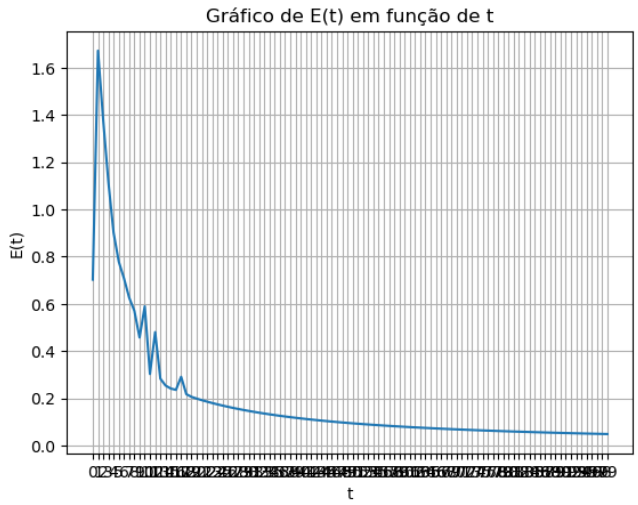


Figure: Gráfico da evolução do Erro ao longo do treino dual da base de dados XOR

OvO (One versus One)

- Divide o problema multi-classe em múltiplos problemas binários.
- Treina N(N-1)/2 classificadores binários para N classes.

Passo 1: Criar todas as combinações pares possíveis por classe e construir os classificadores de acordo.
Passo 2: Aplicar o classificador aos respetivos dados de treino do respetivo par e então desenvolver um classificador binário para cada par.
Passo 3: Uma vez que esses classificadores são aplicados a uma amostra, a classe que obteve o maior número de previsões '+1' (após o argmax da soma dos *scores*) ou a classe com o maior *score* é selecionada como rótulo da classe.

ECOC (Error-Correcting Output Codes)

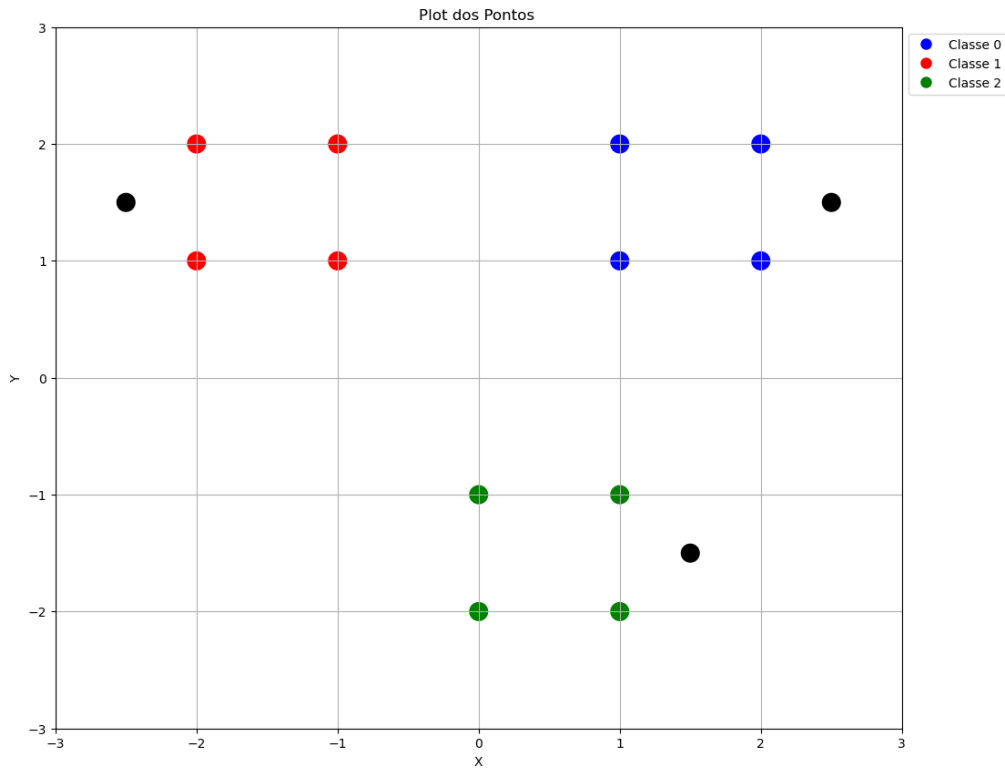
- Utiliza uma matriz de códigos de saída para representar cada classe.
- Flexibilidade na escolha dos classificadores binários e na construção da matriz.
- Distâncias de Hamming - afastamento das previsões com as linhas/classes da matriz
- 4 classes: podemos usar geração de códigos **exaustivos**:

	f_1	f_2	f_3	f_4	f_5	f_6	f_7
0	1	1	1	1	1	1	1
1	1	0	0	1	0	0	1
2	0	1	0	1	1	0	1
3	1	1	1	1	0	1	0

Table: Matriz de códigos de output com 7 bits para um problema com 4 classes

Dataset Sintético

- Dataset de treino (a cores) com 3 classes e 4 pontos cada.
- Teste de 1 ponto perto de cad nuvem de pontos (a preto)
- Todos os algoritmos classificaram corretamente os dados de teste.



Dataset MNIST

O dataset real utilizado: MNIST (Modified National Institute of Standards and Technology).

- formado por imagens de dígitos manuscritos de 0 a 9 (10 classes no total), totalizando 70.000 exemplos, dos quais 60.000 são utilizados para treino e 10.000 para teste.
- monocromáticas (Cada pixel da imagem é representado por um valor de intensidade que varia entre 0 (preto) e 255 (branco))
- resolução de 28x28 pixels
- 2 versões: 3 classes (0, 1 e 2) e com 4 classes (0, 1, 2 e 3)

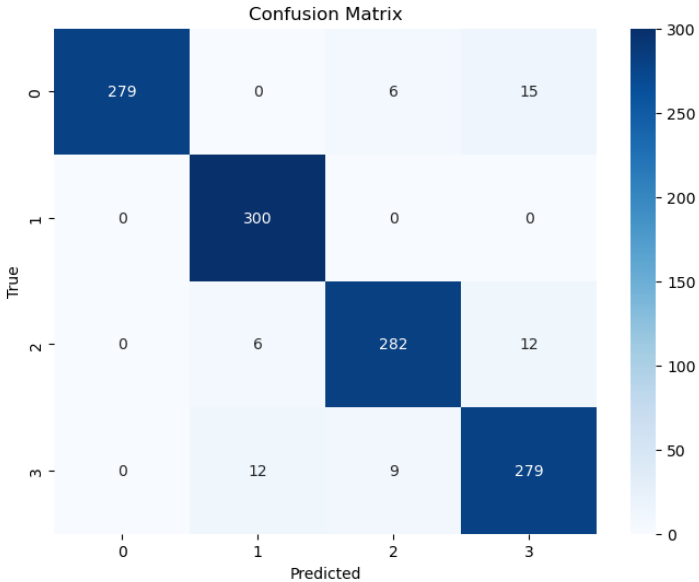
Figure: Dataset MNIST.

Resultados com MNIST (3 classes)

- 3 classes: 0, 1 e 3
  - Número de iterações: 1000
  - $\eta = 0.5$
  - (só para as versões com Kernel) d $= 2$
- Todos os algoritmos executaram **corretamente** (apesar dos resultados variados).
- Confirmação para avançar para para a versão com 4 classes.

Apresentamos de seguida os resultados pormenorizados para o MNIST com 4 classes.

Clog - MGmB (Primal - OvO)



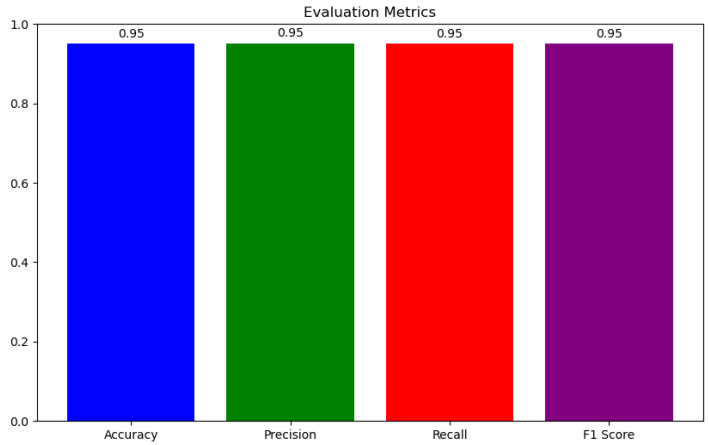


Figure: Matriz de confusão do Classificador Logístico Primal com Mini-Batch com o dataset MNIST (esquerda) e métricas de performance (direita).

Clog - MGE (Primal - OvO)

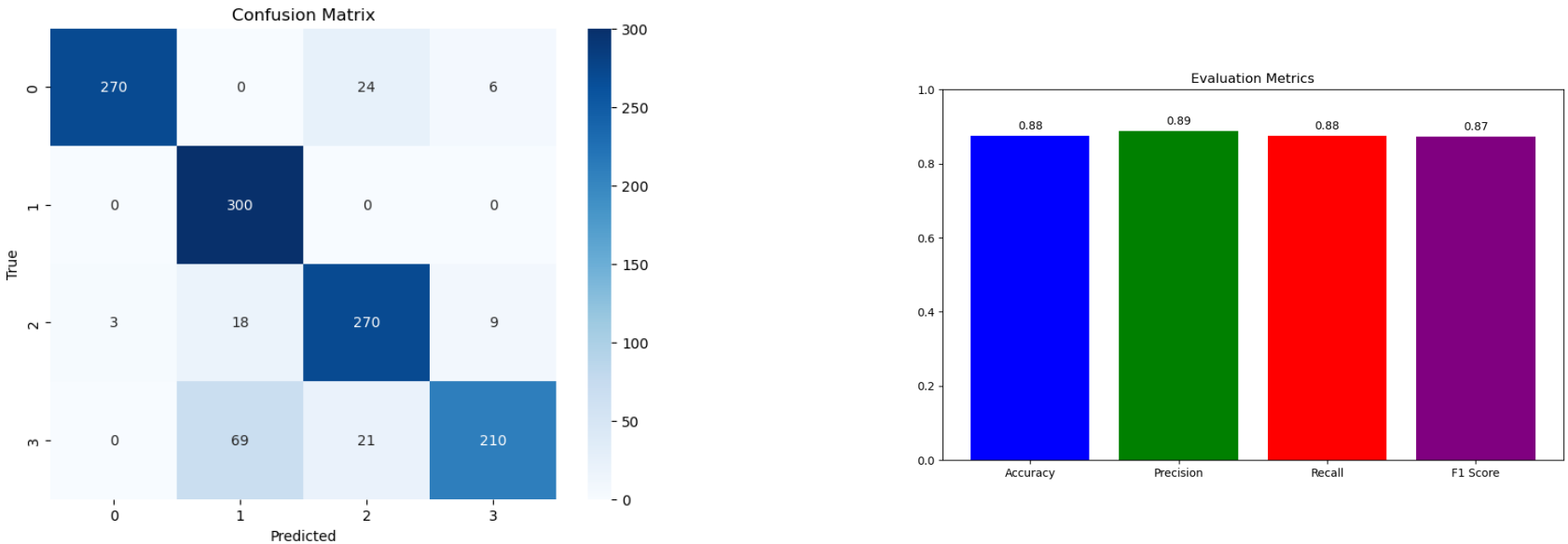
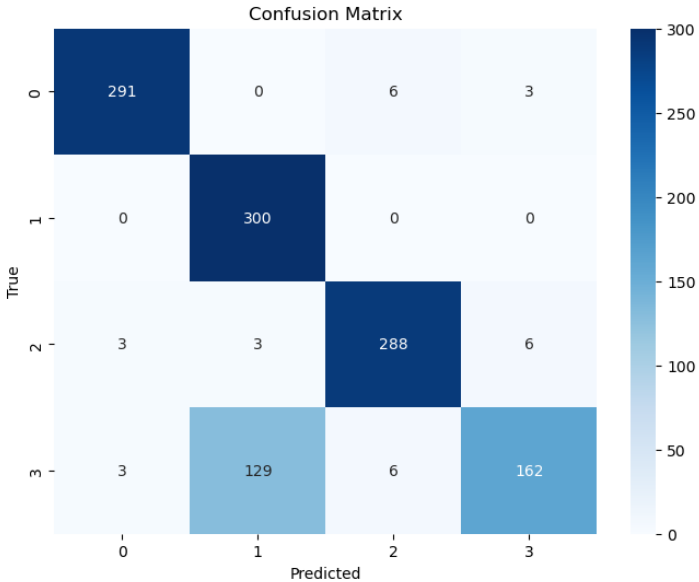


Figure: Matriz de confusão do Classificador Logístico Primal Estocástico com o dataset MNIST (esquerda) e métricas de performance (direita).

Clog - MGE-Ordenado (Primal - OvO)



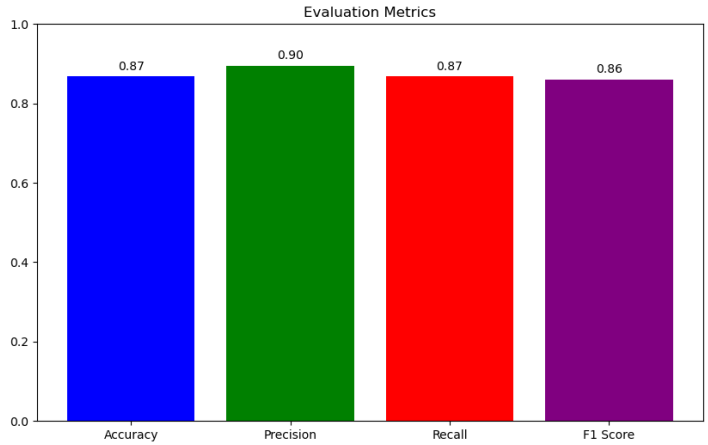


Figure: Matriz de confusão do Classificador Logístico Primal com Estocástico Ordenado com o dataset MNIST (esquerda) e métricas de performance (direita).

ClogD - MGmB (Dual sem kernel - OvO)

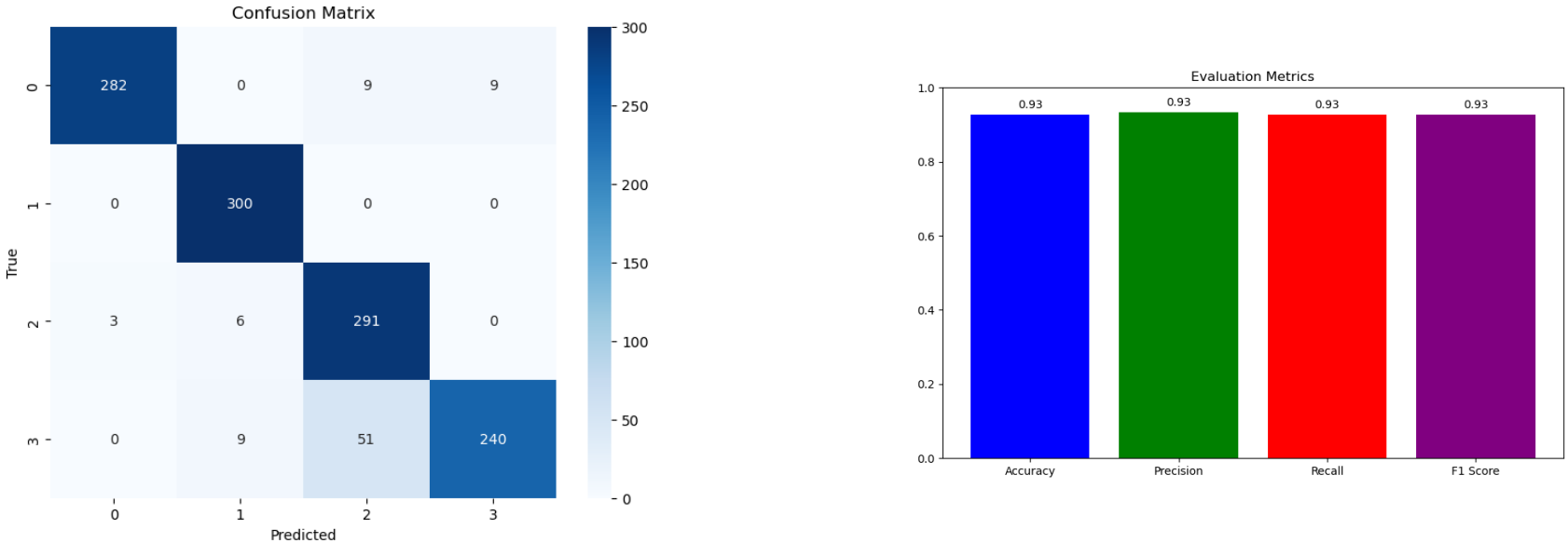


Figure: Matriz de confusão do Classificador Logístico Dual (sem kernel) com Mini-Batch com o dataset MNIST (esquerda) e métricas de performance (direita).

ClogD - MGE (Dual sem kernel - OvO)

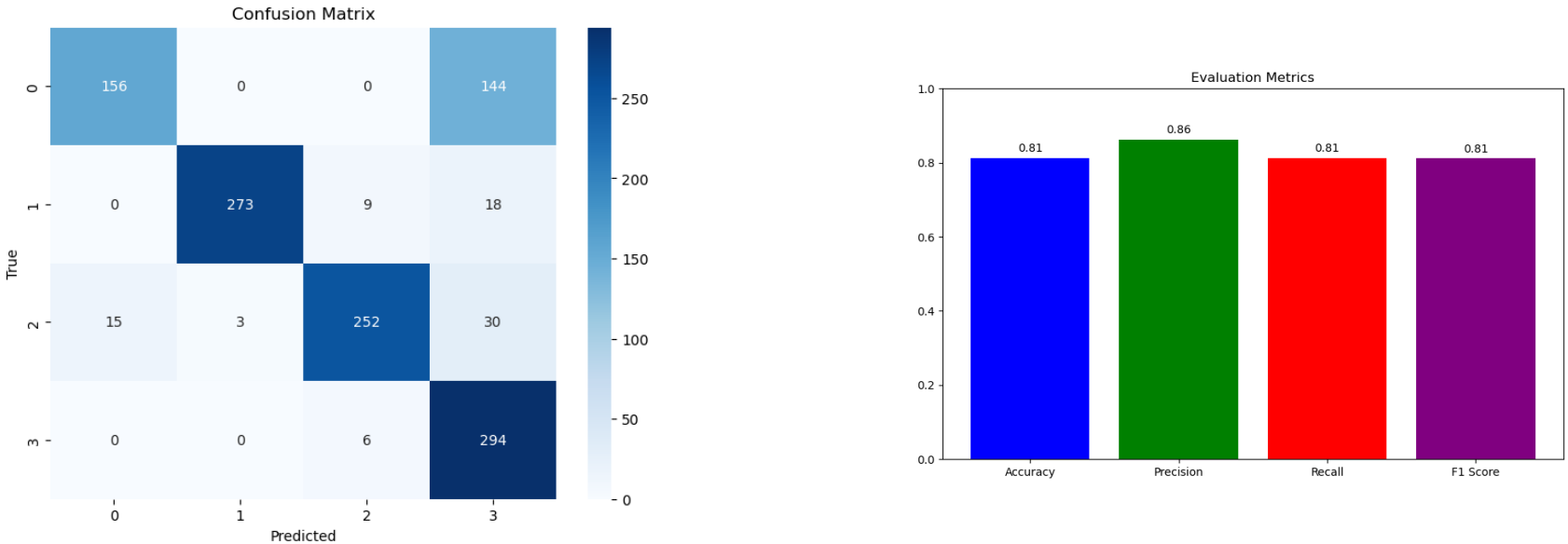
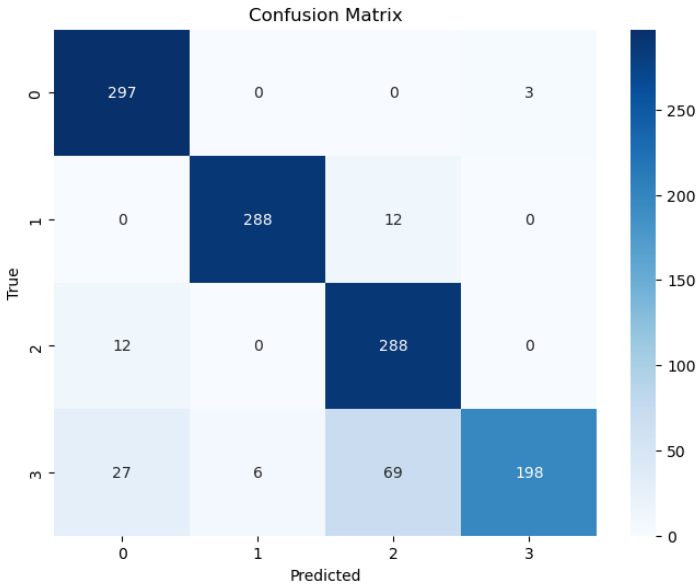


Figure: Matriz de confusão do Classificador Logístico Dual (sem kernel) Estocástico com o dataset MNIST (esquerda) e métricas de performance (direita).

ClogD - MGE_Ordenado (Dual sem kernel - OvO)



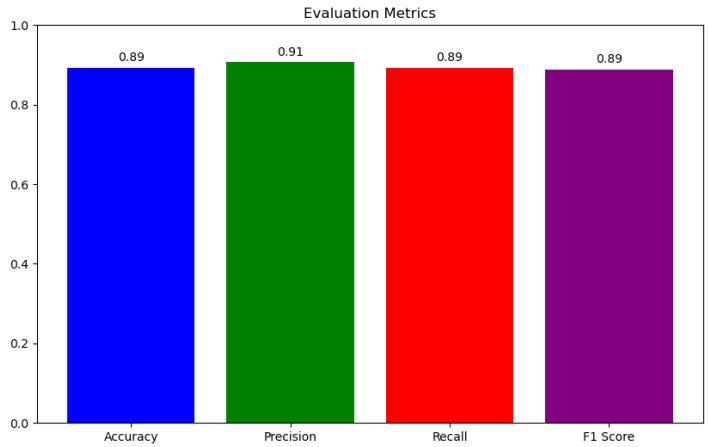
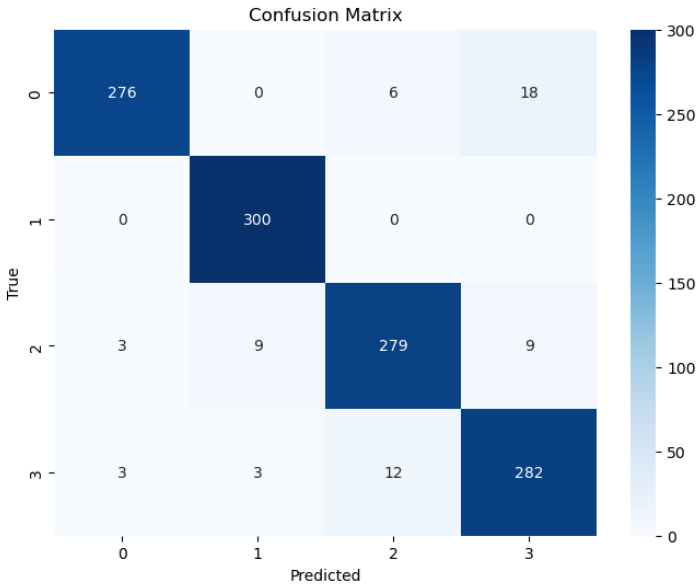


Figure: Matriz de confusão do Classificador Logístico Dual (sem kernel) Estocástico Ordenado com o dataset MNIST (esquerda) e métricas de performance (direita).

ClogDKP2 - MGmB (Dual com kernel - OvO)



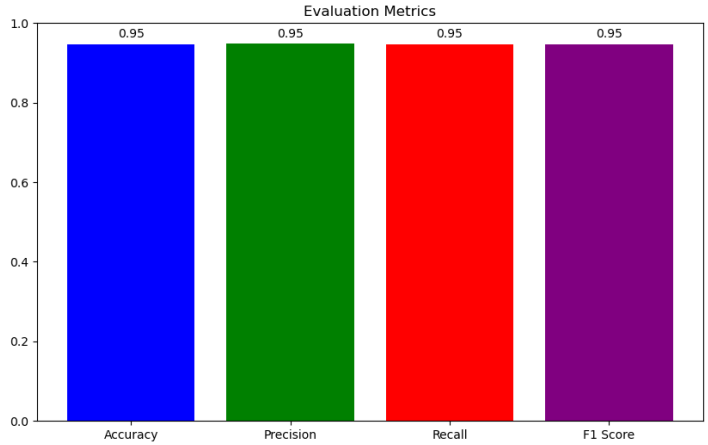


Figure: Matriz de confusão do Classificador Logístico Dual (com kernel) com Mini-Batch com o dataset MNIST (esquerda) e métricas de performance (direita).

ClogDKP2 - MGE (Dual com kernel - OvO)

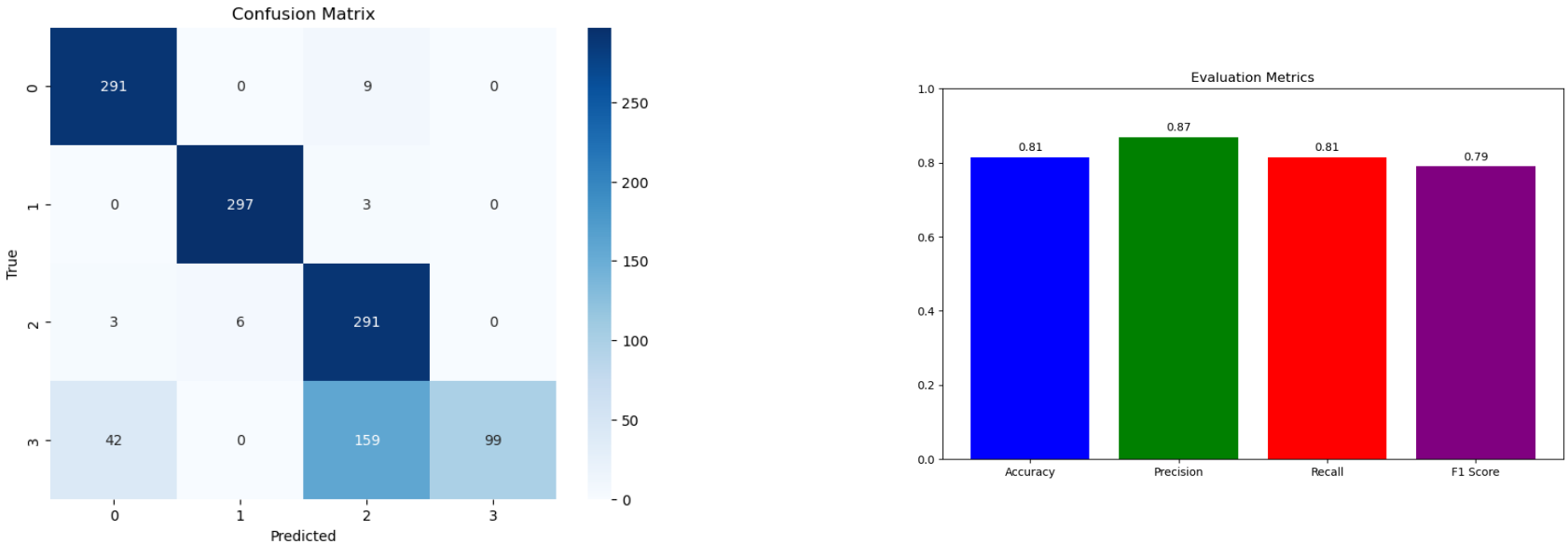
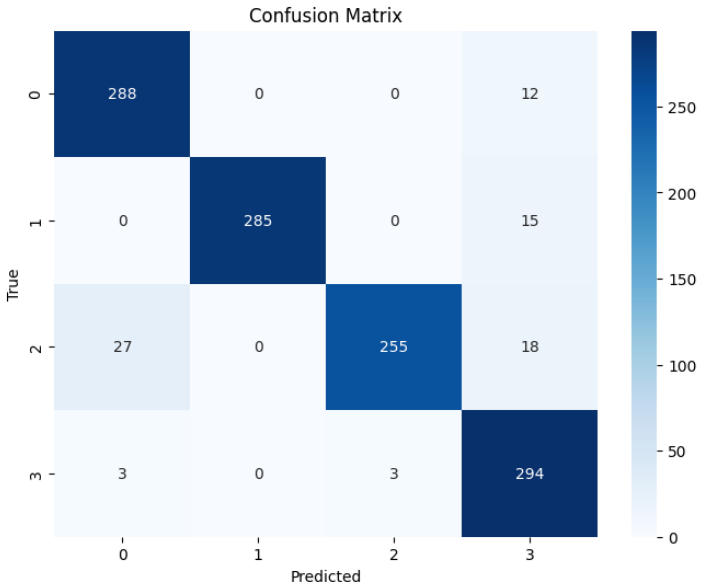


Figure: Matriz de confusão do Classificador Logístico Dual (com kernel) Estocástico com o dataset MNIST (esquerda) e métricas de performance (direita).

ClogDKP2 - MGE_Ordenado (Dual com kernel - OvO)



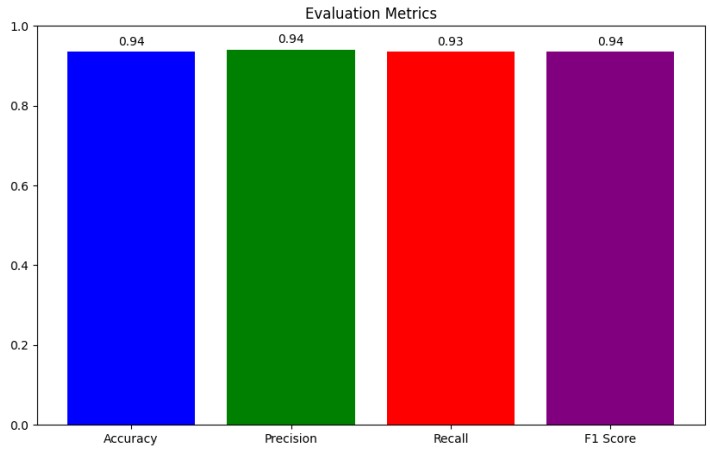


Figure: Matriz de confusão do Classificador Logístico Dual (com kernel) Estocástico Ordenado com o dataset MNIST (esquerda) e métricas de performance (direita).

Clog - MGmB (Primal - ECOC)

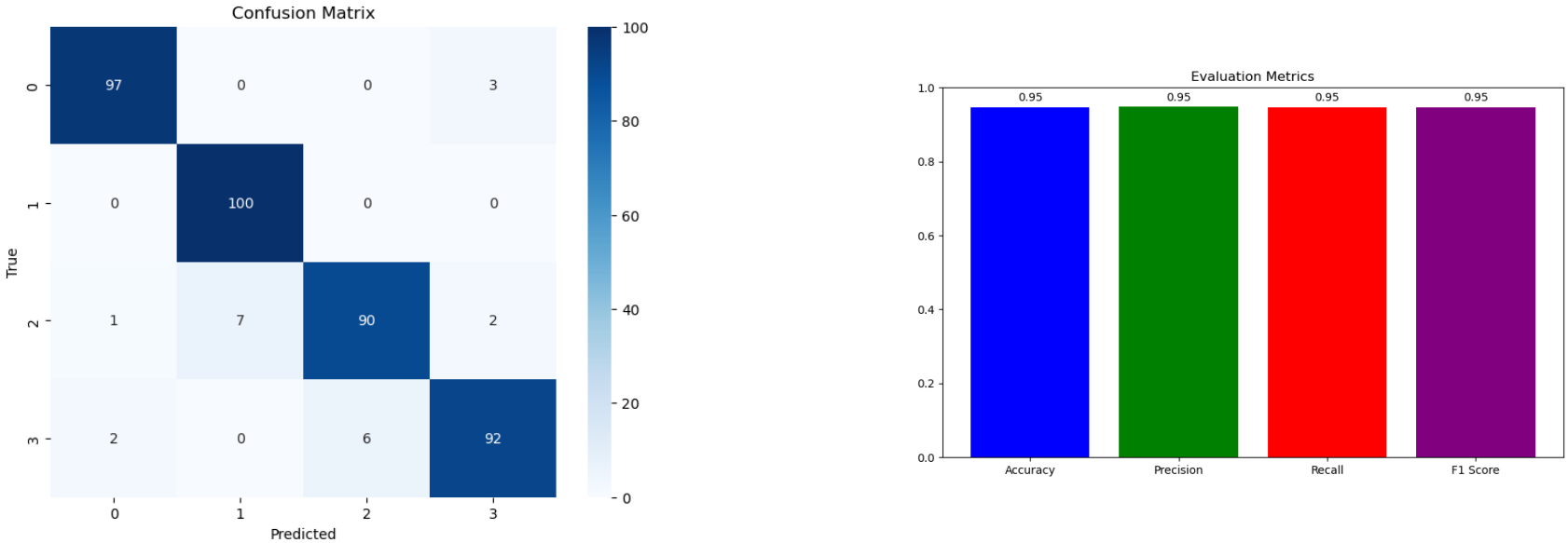


Figure: Matriz de confusão do Classificador Logístico Primal com Mini-Batch com o dataset MNIST (esquerda) e métricas de performance (direita).

Clog - MGE (Primal - ECOC)

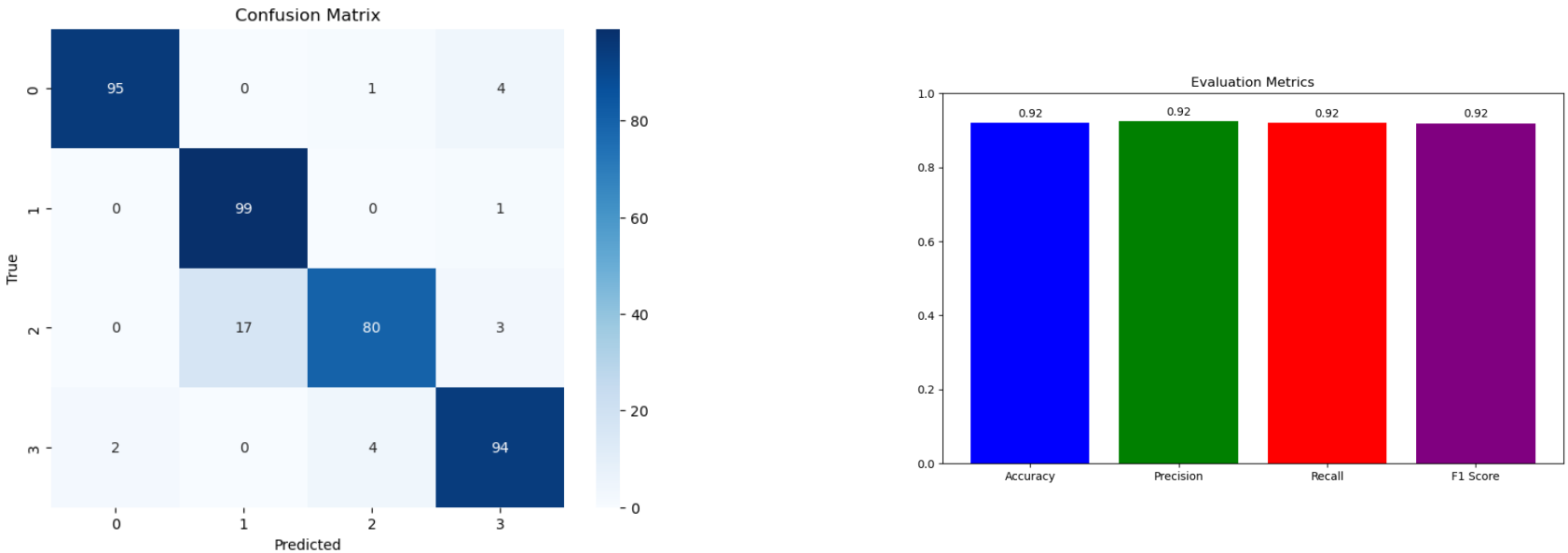


Figure: Matriz de confusão do Classificador Logístico Primal Estocástico com o dataset MNIST (esquerda) e métricas de performance (direita).

Clog - MGE_Ordenado (Primal - ECOC)

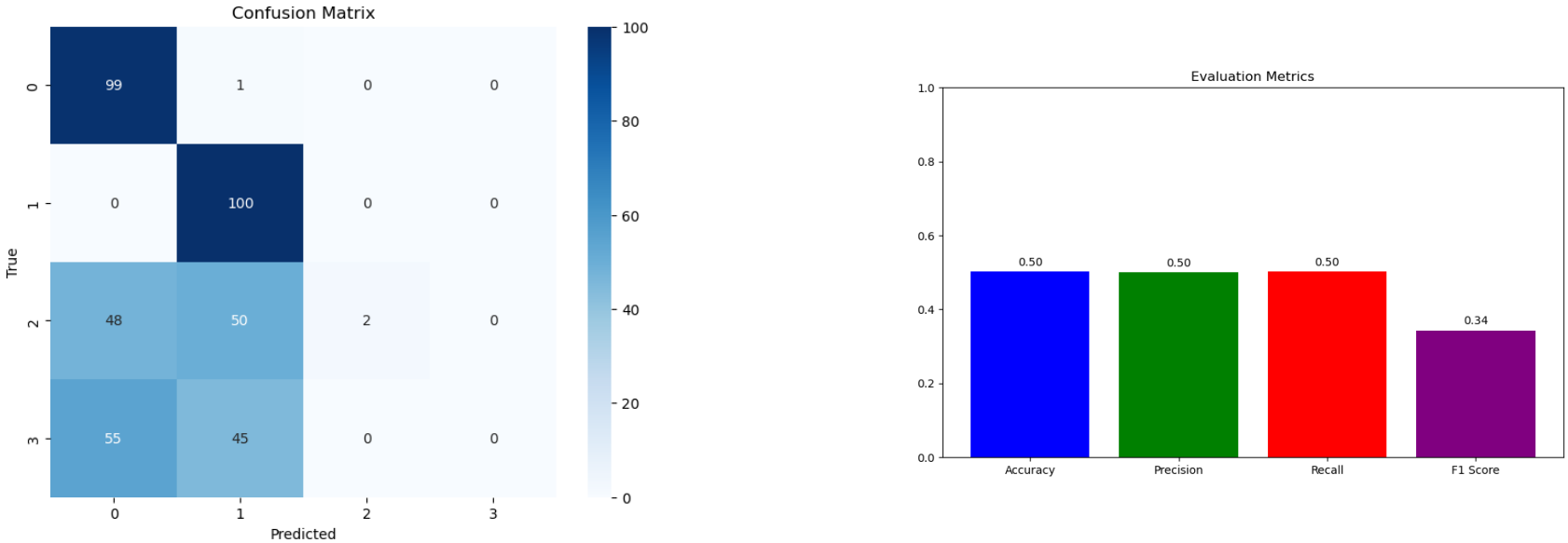


Figure: Matriz de confusão do Classificador Logístico Primal com Estocástico Ordenado com o dataset MNIST (esquerda) e métricas de performance (direita).

ClogD - MGmB (Dual sem kernel - ECOC)

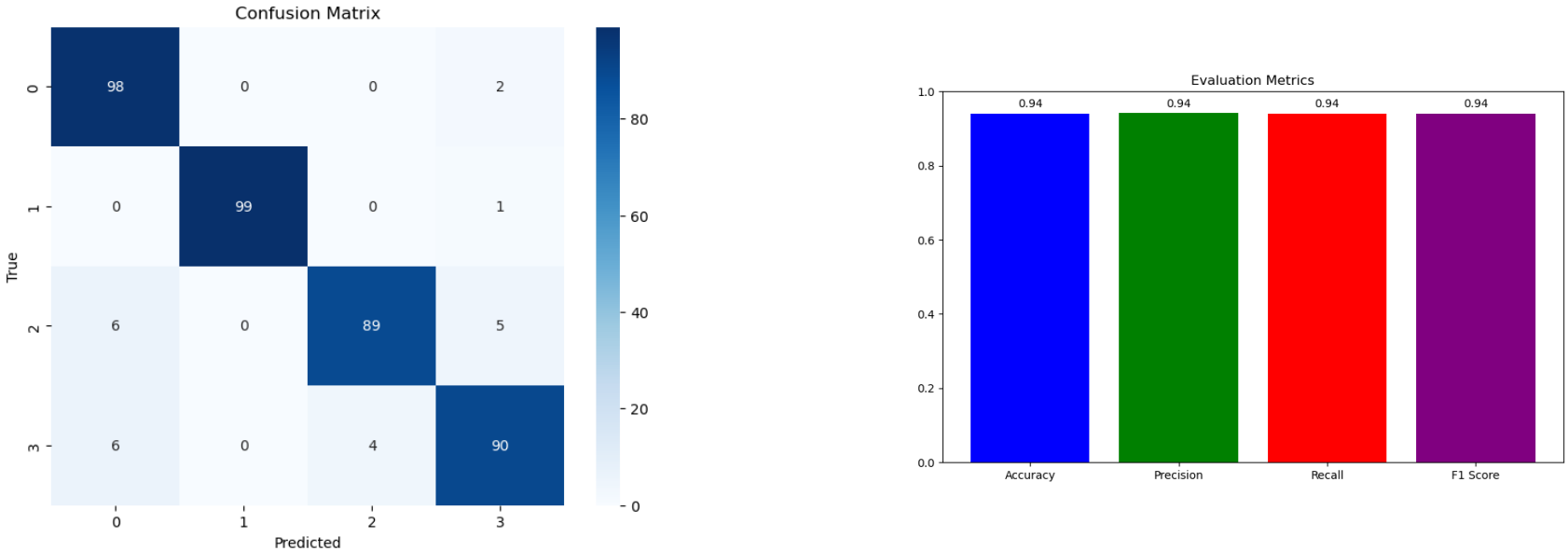


Figure: Matriz de confusão do Classificador Logístico Dual (sem kernel) com Mini-Batch com o dataset MNIST (esquerda) e métricas de performance (direita).

ClogD - MGE (Dual sem kernel - ECOC)

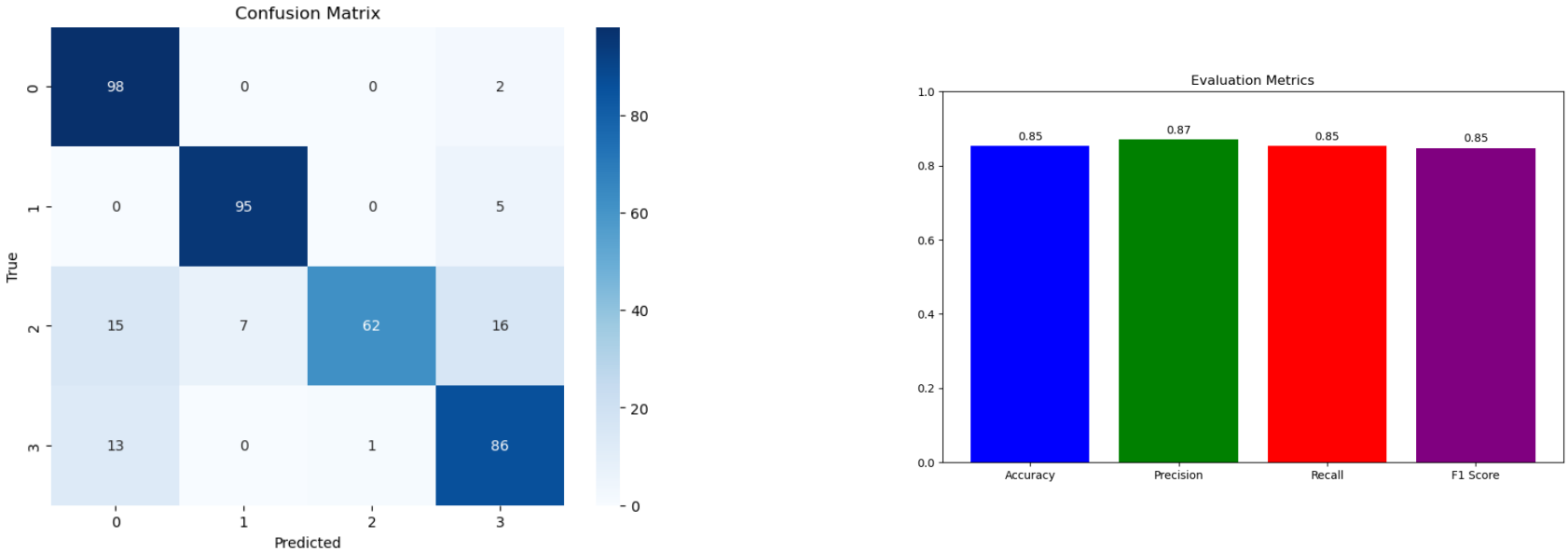
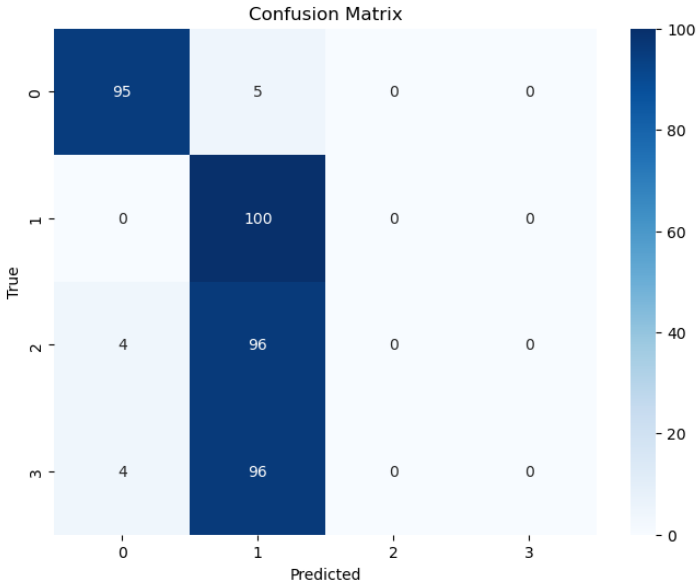


Figure: Matriz de confusão do Classificador Logístico Dual (sem kernel) Estocástico com o dataset MNIST (esquerda) e métricas de performance (direita).

ClogD - MGE_Ordenado (Dual sem kernel - ECOC)



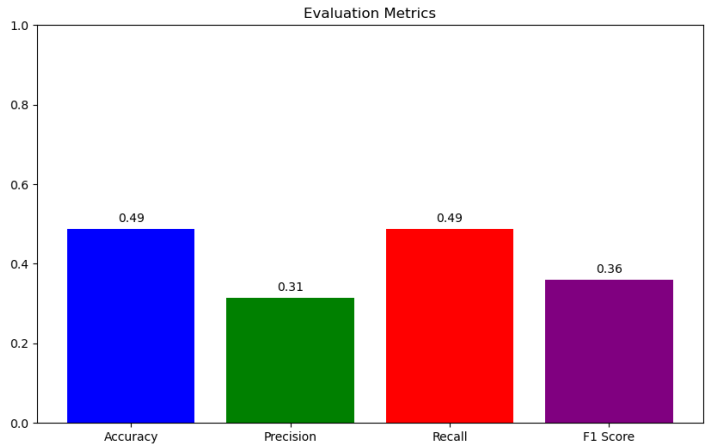


Figure: Matriz de confusão do Classificador Logístico Dual (sem kernel) Estocástico Ordenado com o dataset MNIST (esquerda) e métricas de performance (direita).

ClogDKP2 - MGmB (Dual Com kernel - ECOC)

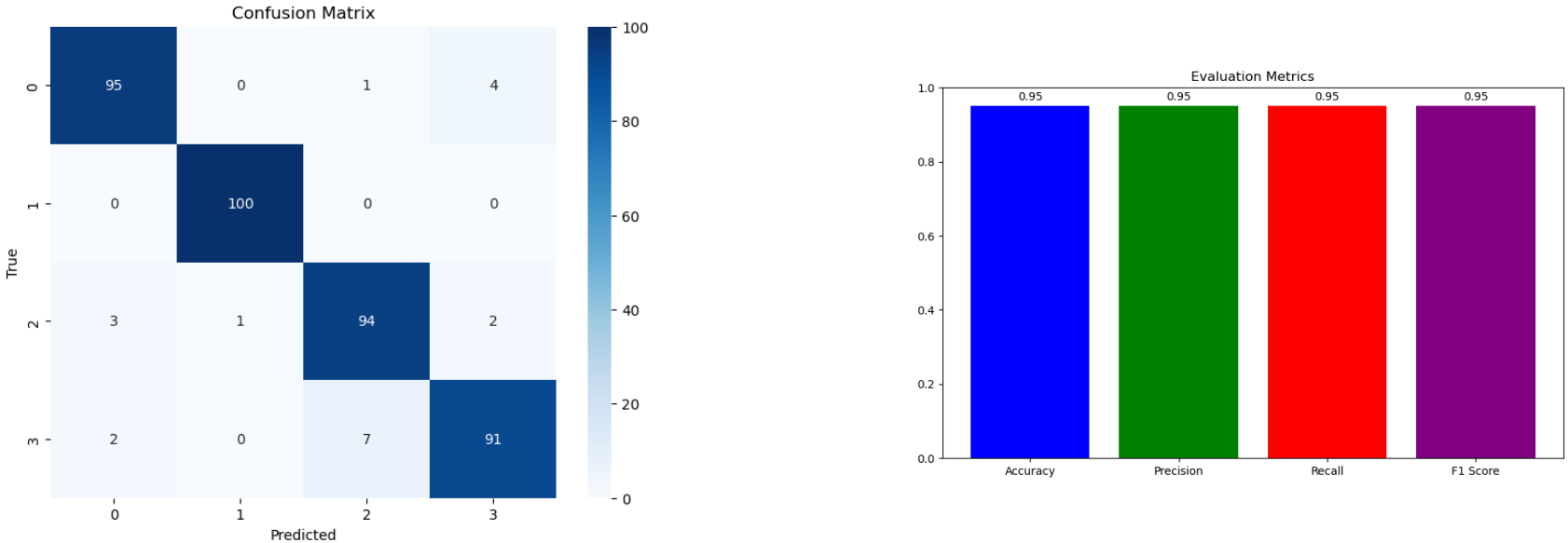
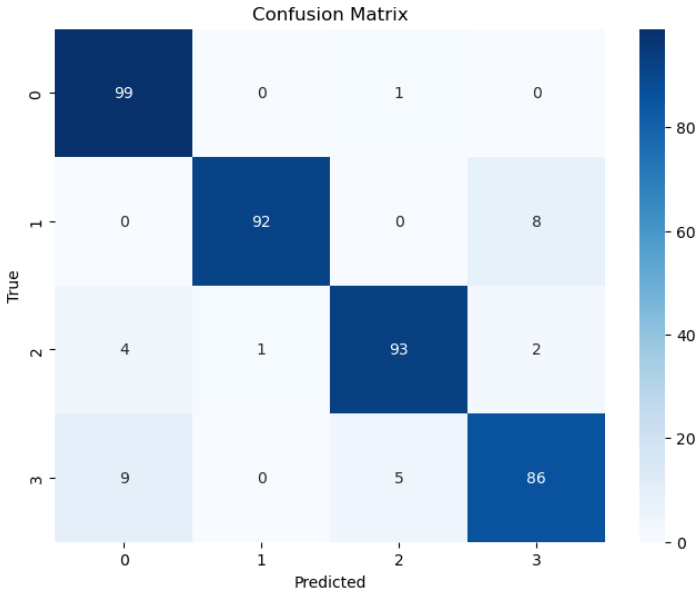


Figure: Matriz de confusão do Classificador Logístico Dual (com kernel) com Mini-Batch com o dataset MNIST (esquerda) e métricas de performance (direita).

ClogDKP2 - MGE (Dual Com kernel - ECOC)



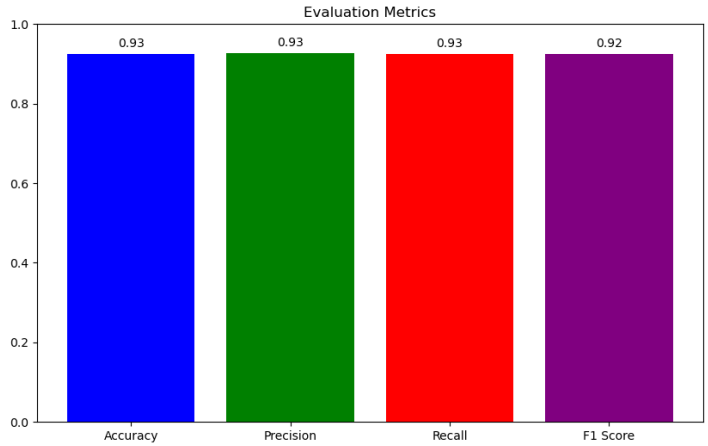


Figure: Matriz de confusão do Classificador Logístico Dual (com kernel) Estocástico com o dataset MNIST (esquerda) e métricas de performance (direita).

ClogDKP2 - MGE_Ordenado (Dual Com kernel - ECOC)

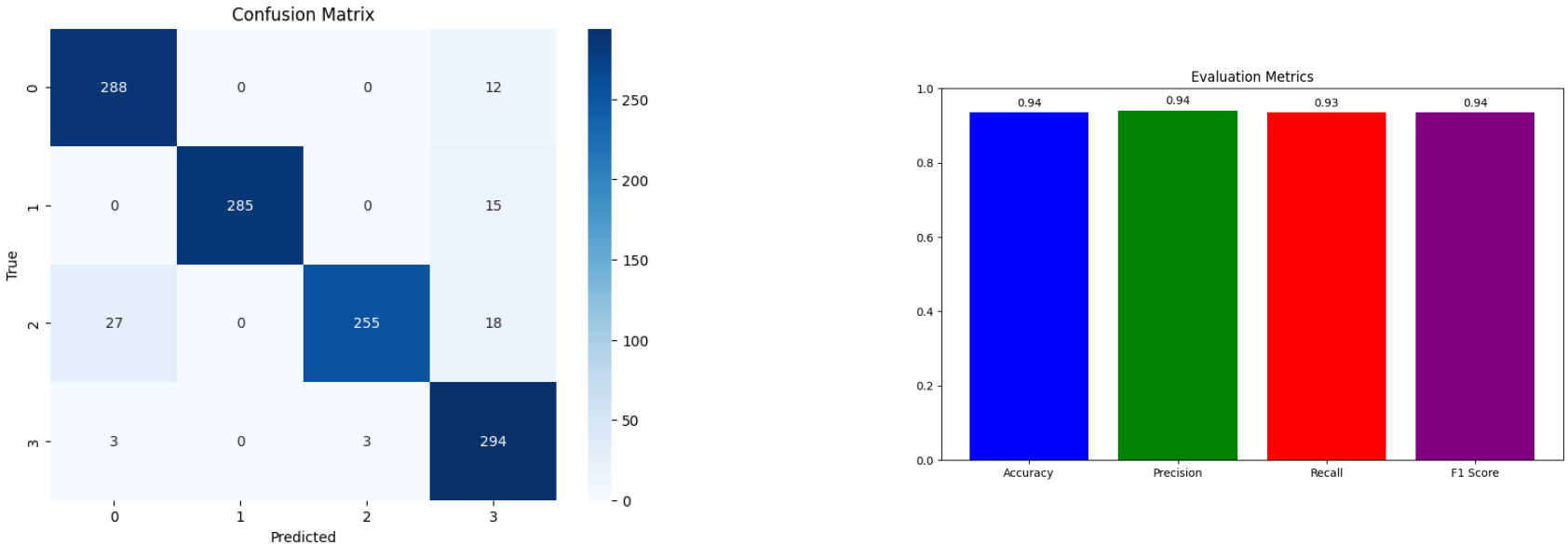


Figure: Matriz de confusão do Classificador Logístico Dual (com kernel) Estocástico Ordenado com o dataset MNIST (esquerda) e métricas de performance (direita).

Análise dos Resultados Obtidos

- Comparação detalhada dos resultados obtidos com OvO e ECOC;
- Identificação das situações em que cada técnica é mais vantajosa.
- De notar também que de modo geral a metodologia OvO foi computacionalmente mais rápida que a do ECOC

Conclusão

- Ao longo deste trabalho verificou-se que a utilização do CLassificador Logístico Binário com as técnicas (OvO e ECOC) mostrou resultados satisfatórios, cumprindo os objetivos estabelecidos no início da pesquisa;
- A inclusão de *shuffle* dos dados, aumenta perfomance dos classificadores, pois a ordem dos dados na alimentação do algoritmo é bastante relevante;
- A aleatoriedade na escolha de elementos durante o treinamento de um classificador logístico estocástico é crucial para evitar mínimos locais e ciclos indesejados e rápida aproximação da solução desejada.

”MUITO OBRIGADO
*
pela vossa atenção!”