Breast Cancer Wisconsin (Diagnostic)

Luís Gustavo
Aramys Almeida Matos

Inteligência Computacional

7 de dezembro de 2016

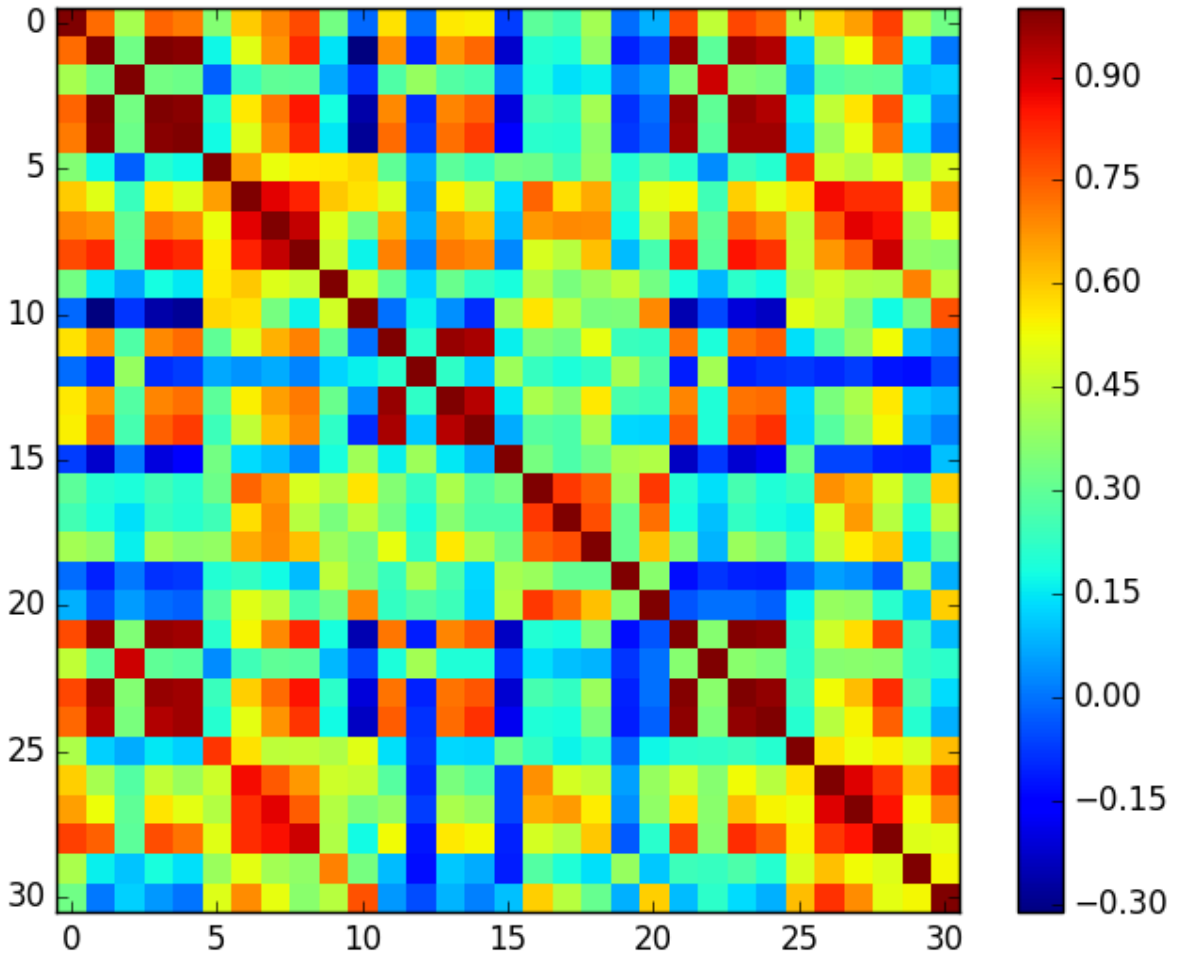
Dataset

- Exames para câncer de mama.
- Características do núcleo de células computadas a partir de imagens digitalizadas
- 30 variáveis de entrada
- 1 variável de saída
- Raio, textura, perímetro, área, suavidade, compacidade, concavidade, pontos côncavos, simetria, dimensão fractal

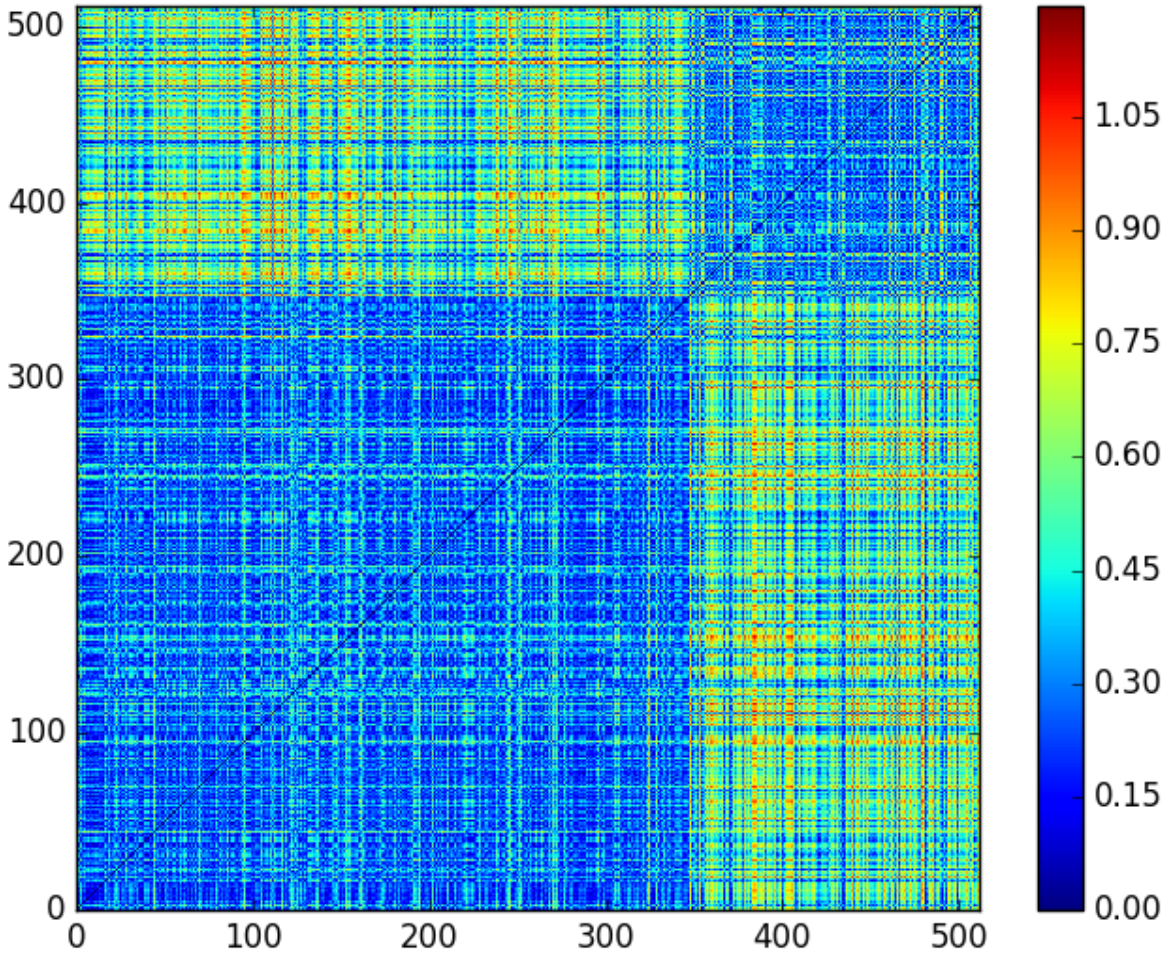
Problema

- Problema de classificação
- Identificar se o câncer é benigno ou maligno.
- Aplicar e avaliar os modelos de classificação:
  - Classificador Bayesiano Simples
  - Classificador Bayesiano Quadrático
  - Regressão Logística
  - Perceptron
  - Perceptron Múltiplas Camadas
  - SVM

Matriz de Correlação

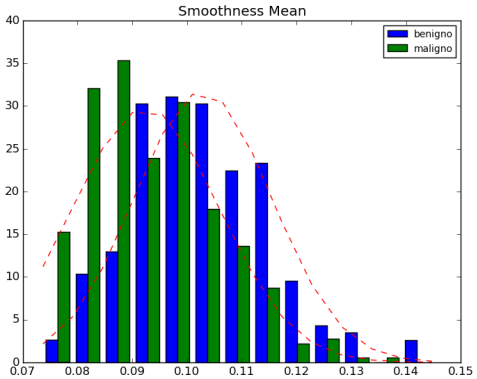


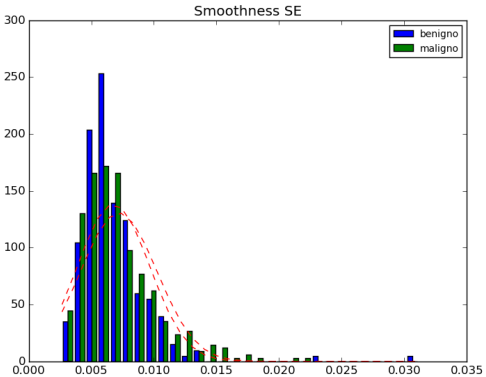
Distância Euclidiana

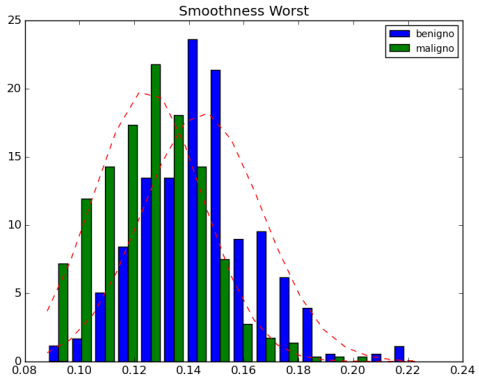


Histogramas

Smoothness

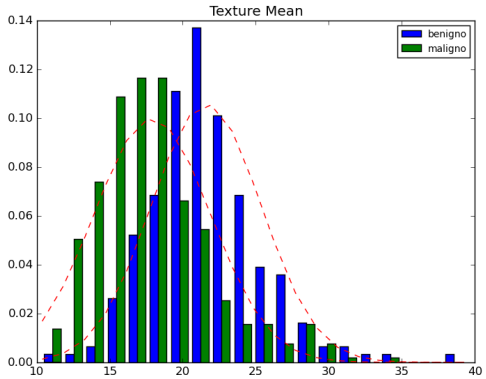


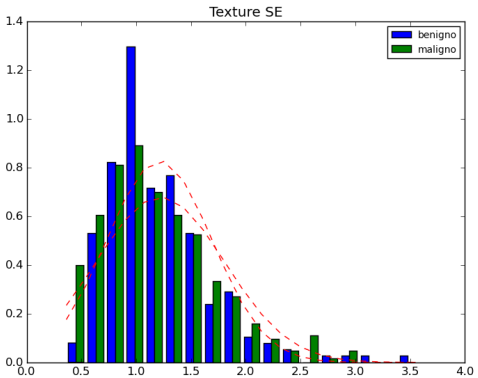


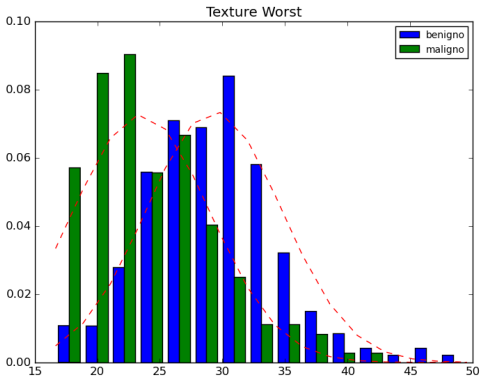


Histogramas

Texture







Tecnologia

Python

- SciPy
  - *NumPy*
  - *matplotlib*
  - *pandas*: Python Data Analysis Library
- *scikit-learn*: Machine Learning in Python

Matlab

- Statistics and Machine Learning Toolbox

Metodologia

Para cada método de classificação foi feito:

- Validação cruzada de 10 ciclos
- Matriz de confusão (média de cada ciclo)
- Métricas de acurácia, precisão, recuperação.

Classificador Bayesiano Simples (Naive Bayes)

- Aplicação do teorema de Bayes
- Supõe que cada par de variáveis é independente

Theorem (Teorema de Bayes)

$$P(y|x_1 \dots x_n) = \frac{P(y)P(x_1, \dots, x_n|y)}{P(x_1, \dots, x_n)}$$

Onde:
y é a variável de saída que identifica a classe
$x = [x_1, \dots, x_n]$ é o vetor de entrada

Classificador Bayesiano (Naive Bayes) - Resultados

Avaliação dos resultados

- $ACC = 93.67\%$
- $AUC = 0.9266$
- $PRE(C1) = 0.94$
- $REC(C1) = 0.8868$
- $PRE(C2) = 0.9350$
- $REC(C2) = 0.9664$

	$\hat{C}_1$ **(Predita)**	$\hat{C}_2$**(Predita)**
C_1	18.80	2.40
C_2	1.20	34.50

Tabela: Matriz de confusão

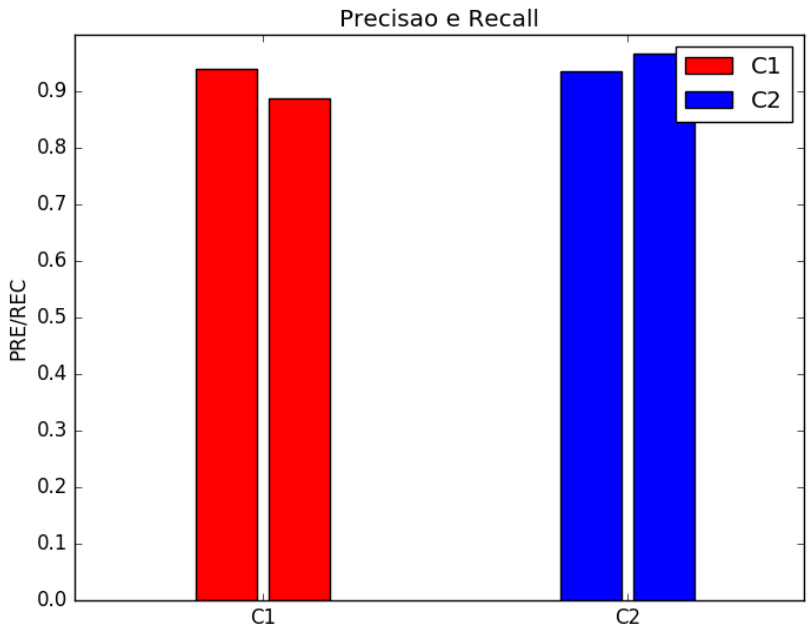


Figura: Precisão e Recall

Classificador Bayesiano Quadrático

- Um classificador com uma fronteira de decisão quadrática, gerado pela densidades condicionais dos dados e utilizando a regra de Bayes.
- A decisão é calculada pela função discriminante:

Theorem (Função Discriminante)

$$g_i(x(t)) = lnP(C_i|x(t)) = lnP(x(t)|C_i) + lnP(C_i)$$

Classificador Bayesiano Quadrático - Resultados

Avaliação dos resultados

- $ACC = 95.78\%$
- $AUC = 0.9549$
- $PRE(C1) = 0.9434$
- $REC(C1) = 0.9434$
- $PRE(C2) = 0.9664$
- $REC(C2) = 0.9664$

	$\hat{C}_1$ **(Predita)**	$\hat{C}_2$**(Predita)**
C_1	20.00	1.20
C_2	1.20	34.50

Tabela: Matriz de confusão

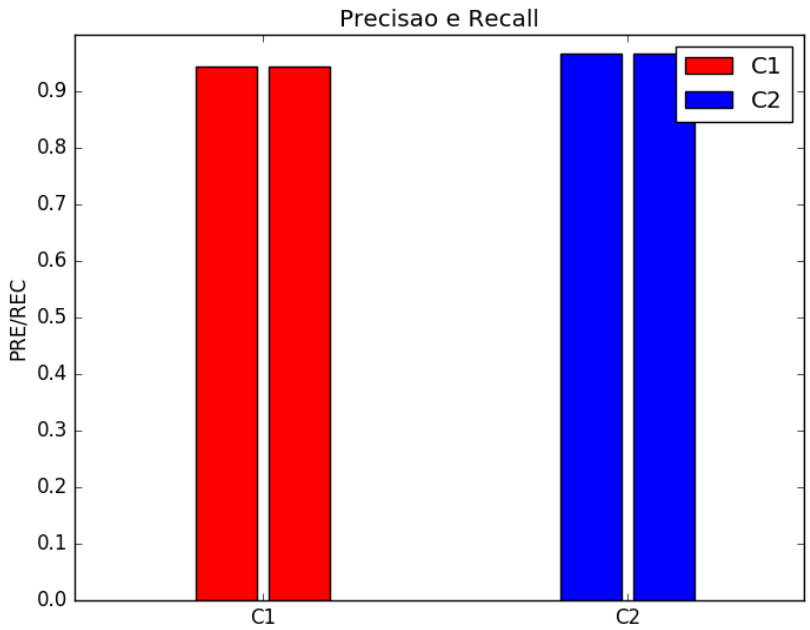


Figura: Precisão e Recall

Regressão Logística

- Na regressão logística, a saída do modelo é uma aproximação da probabilidade a posteriori.
- A função discriminante é calculada pela função sigmoide, ou função logística ou logit

Theorem (Função Logística)

$$g_i(x(t)|\theta_i)) = \frac{1}{1 + exp(\hat{x}(t)\theta_i)}$$

Onde:

$$\hat{x}(t) = [1, x(t)] e \theta_i = [\theta_{i0}, \theta_{i1}]^T$$

Regressão Logística - Resultados

Avaliação dos resultados

- $ACC = 95.96\%$
- $AUC = 0.9563$
- $PRE(C1) = 0.9479$
- $REC(C1) = 0.9434$
- $PRE(C2) = 0.9665$
- $REC(C2) = 0.9692$

	$\hat{C}_1$ (Predita)	$\hat{C}_2$(Predita)
C_1	20.00	1.20
C_2	1.10	34.60

Tabela: Matriz de confusão

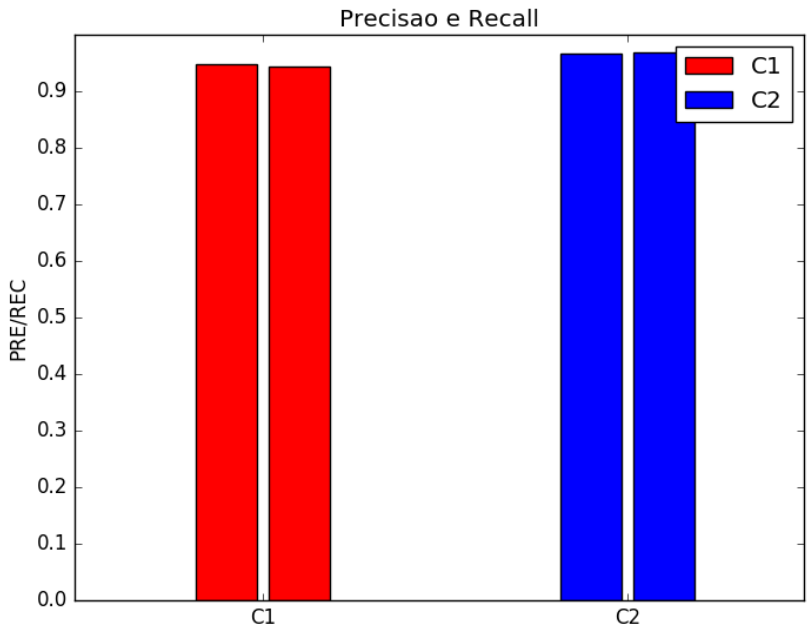


Figura: Precisão e Recall

Perceptron

O Perceptron utiliza o modelo McCulloch-Pitts para o neurônio artificial. O processamento de cada unidade é dado por:

McCulloch-Pitts

$$u(t) = h(z(t)) = h\left(\theta_0 + \sum_{i=1}^{n} x_i(t)\theta_i\right)$$

Onde:
$u(t)$: valor de ativação
$z(t)$: potencial de ativações
h: função de ativação
$x_i(t)$: entradas do neurônio

Perceptron - Implementação

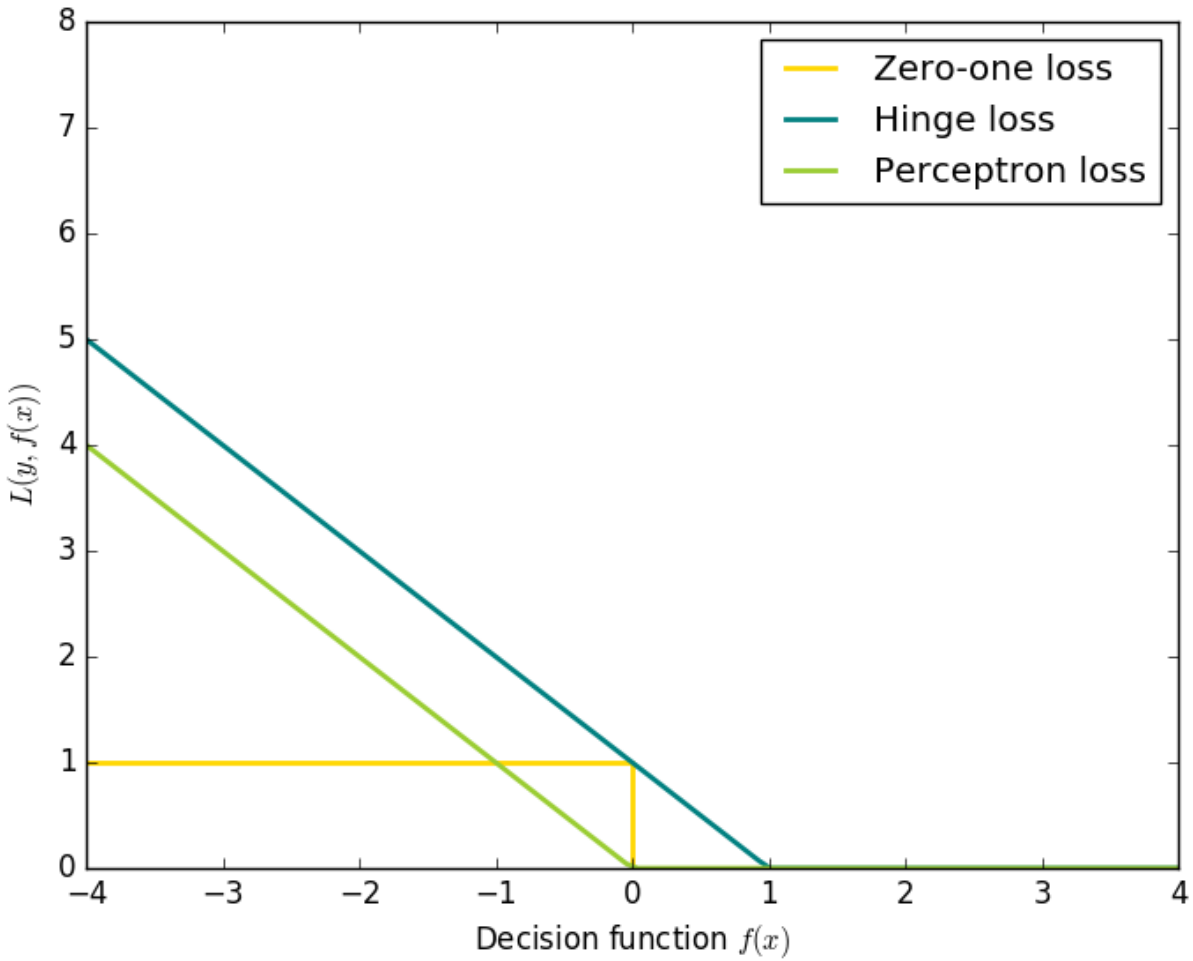
Scikit Learn

```
class sklearn.linear_model.Perceptron(penalty=None,
  alpha=0.0001, fit_intercept=True, n_iter=5,
  shuffle=True, verbose=0, eta0=1.0, n_jobs=1,
  random_state=0, class_weight=None, warm_start=False)
```

É uma expecialização do `sklearn.linear_model.SGDClassifier`.

Perceptron

- Função custo: linear



Avaliação dos resultados

- $ACC = 96.13\%$
- $AUC = 0.9615$
- $PRE(C1) = 0.9358$
- $REC(C1) = 0.9623$
- $PRE(C2) = 0.9772$
- $REC(C2) = 0.9608$

	$\hat{C}_1$ (Predita)	$\hat{C}_2$(Predita)
C_1	20.40	0.80
C_2	1.40	34.30

Tabela: Matriz de confusão

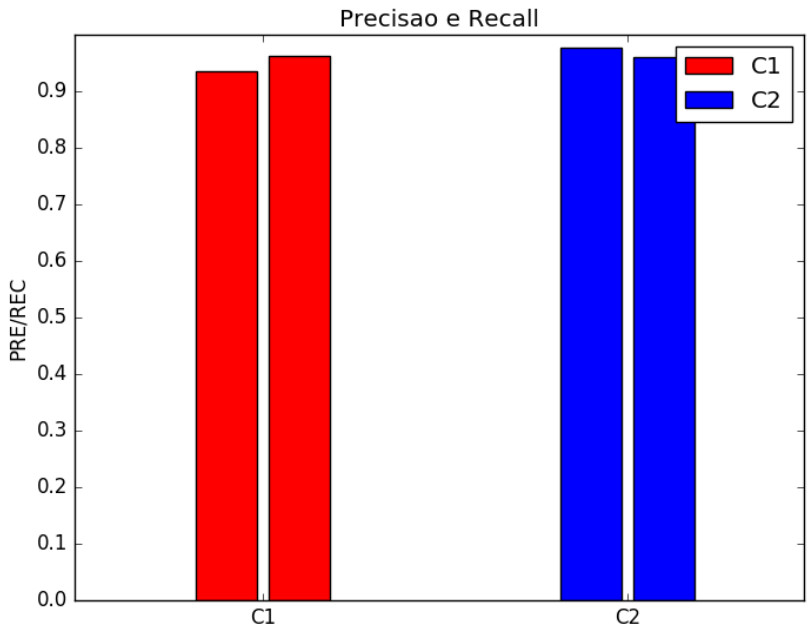
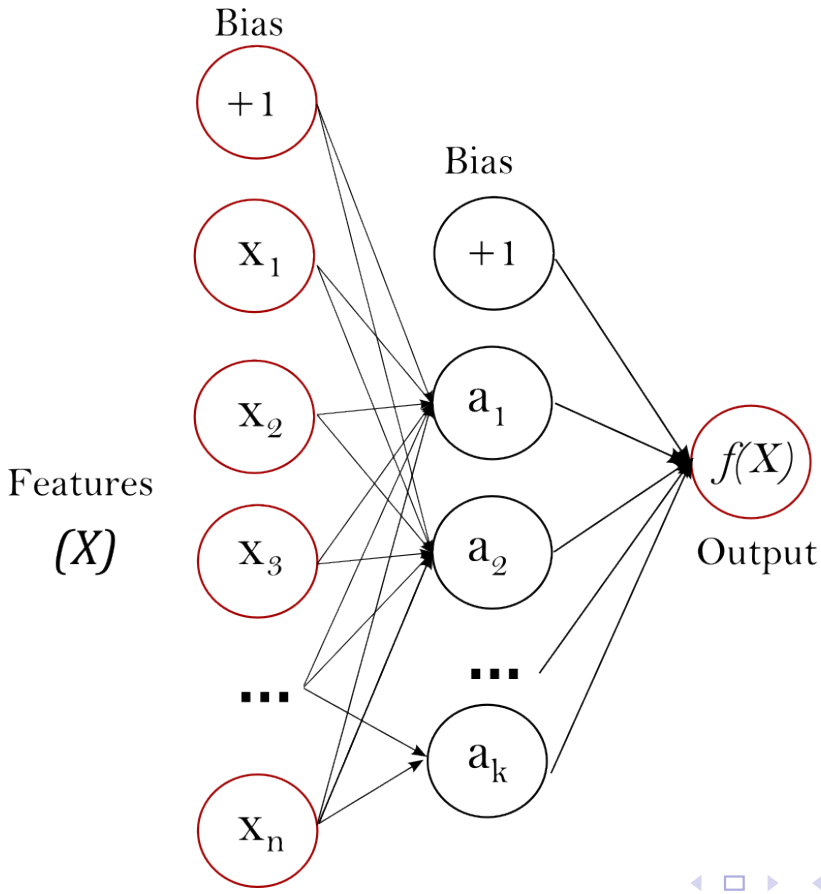


Figura: Precisão e Recall

Perceptron de Múltiplas Camadas

- Camada de entrada que não realiza processamento (dimensão do vetor de entrada)
- Camada de saída: vetor com as estimativas das variáveis indicadoras (dimensão vetor de saída)



Perceptron de Múltiplas Camadas

- A função custo a ser utilizada é a Entropia cruzada, ou função verossimilhança:

Entropia Cruzada

$$l(\theta) = -\sum_{t=1}^{N} \nu(t) log(\hat{\nu}) + (1 - \nu(t)) log(1 - \hat{\nu}(t))$$

- A função de ativação nos neurônios das camadas intermediárias é:

Tangente hiperbólica

$$u(t) = \frac{1 - exp(-z(t))}{1 + exp(-z(t))}$$

- Método de ajuste dos parâmetros: Gradiente descendente estocástico
- Aprendizado através do Backpropagation

Perceptron de Múltiplas Camadas - Observações

- Capaz de aprender modelos não lineares
- Quando existem camadas escondidas, a função custo é não convexa
- Mais de um mínimo local
- Diferentes resultados a cada inicialização
- Sensível a escala das variáveis de entrada

Perceptron de Múltiplas Camadas - Observações

Padronização das variáveis de entrada

- MLP apresenta desempenho ruim sem padronização
- Média zero e desvio padrão 1
- A padronização é calculada para o conjunto de treinamento e as mesma transformação é aplicada para o conjunto de teste.

Perceptron de Múltiplas Camadas - Implementação

Scikit Learn

```
class sklearn.neural_network.MLPClassifier
(hidden_layer_sizes=(100, ), activation='relu',
 solver='adam', alpha=0.0001, batch_size='auto',
 learning_rate='constant', learning_rate_init=0.001,
 power_t=0.5, max_iter=200, shuffle=True,
random_state=None, tol=0.0001, verbose=False,
 warm_start=False, momentum=0.9, nesterovs_momentum=True,
 early_stopping=False, validation_fraction=0.1,
 beta_1=0.9, beta_2=0.999, epsilon=1e-08)
```

Perceptron de Múltiplas Camadas - Resultados

Avaliação dos resultados para uma camada intermediária com 21 neurônios

	$\hat{C}_1$ (Predita)	$\hat{C}_2$(Predita)
C_1	20.20	1.00
C_2	0.40	35.30

- $ACC = 97.35\%$
- $AUC = 97.08\%$
- $PRE(C1) = 0.9806$
- $REC(C1) = 0.9528$
- $PRE(C2) = 0.9725$
- $REC(C2) = 0.9888$

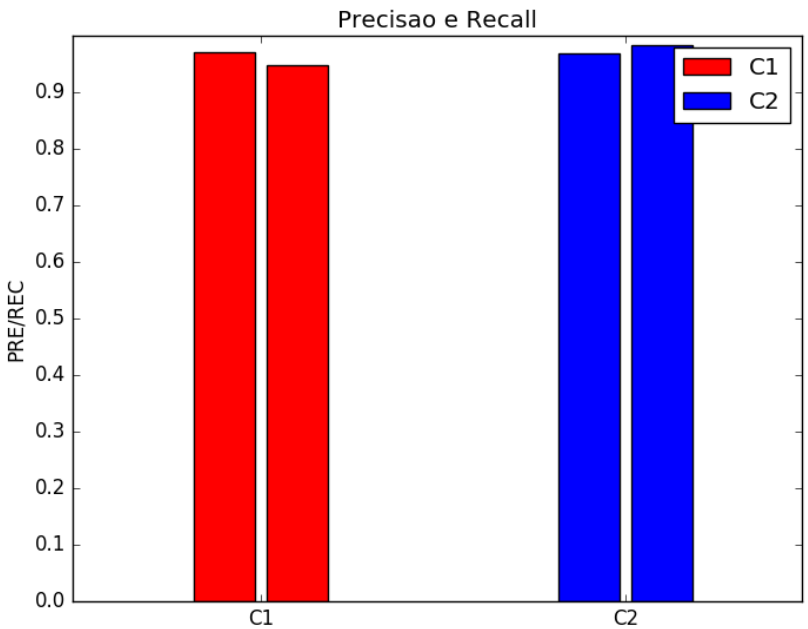


Figura: Precisão e Recall

Camada intermediária com 100 neurônios

	$\hat{C}_1$ (Predita)	$\hat{C}_2$(Predita)
C_1	20.40	0.80
C_2	0.30	35.40

- $ACC = 98.07\%$
- $AUC = 0.9769$
- $PRE(C1) = 0.9855$
- $REC(C1) = 0.9623$
- $PRE(C2) = 0.9779$
- $REC(C2) = 0.9916$

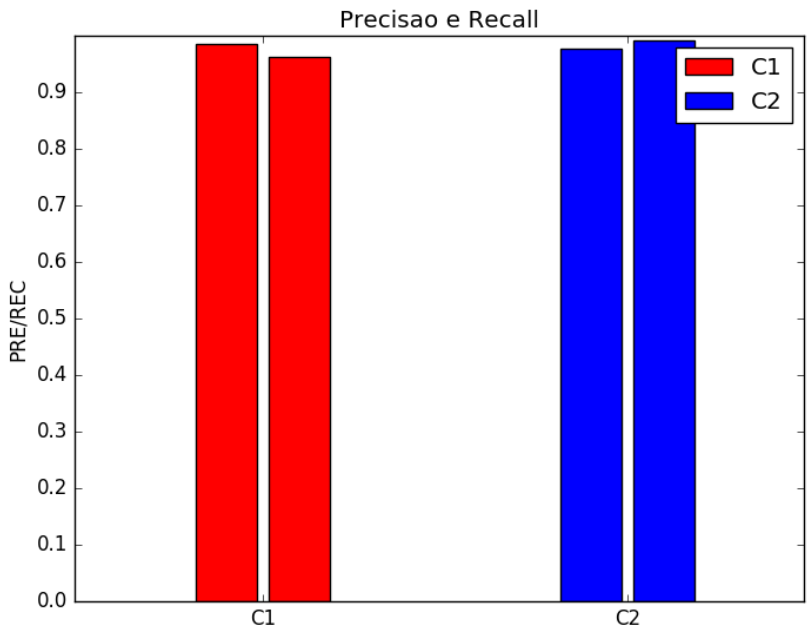


Figura: Precisão e Recall

Camadas intermediárias (10,10)

	$\hat{C}_1$ (Predita)	$\hat{C}_2$(Predita)
C_1	20.30	0.90
C_2	0.30	35.40

- $ACC = 97.89\%$
- $AUC = 0.9746$
- $PRE(C1) = 0.9854$
- $REC(C1) = 0.9575$
- $PRE(C2) = 0.9752$
- $REC(C2) = 0.9916$

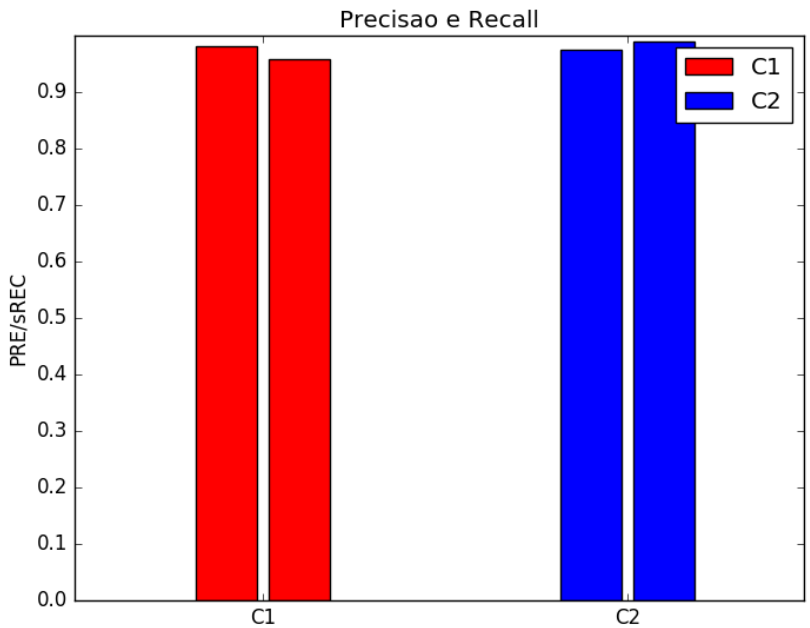


Figura: Precisão e Recall

Avaliação dos resultados

	$\hat{C}_1$ (Predita)	$\hat{C}_2$(Predita)
C_1	20.40	0.80
C_2	0.50	35.20

- $ACC = 97.72\%$
- $AUC = 0.9741$
- $PRE(C1) = 0.9761$
- $REC(C1) = 0.9623$
- $PRE(C2) = 0.9778$
- $REC(C2) = 0.9860$

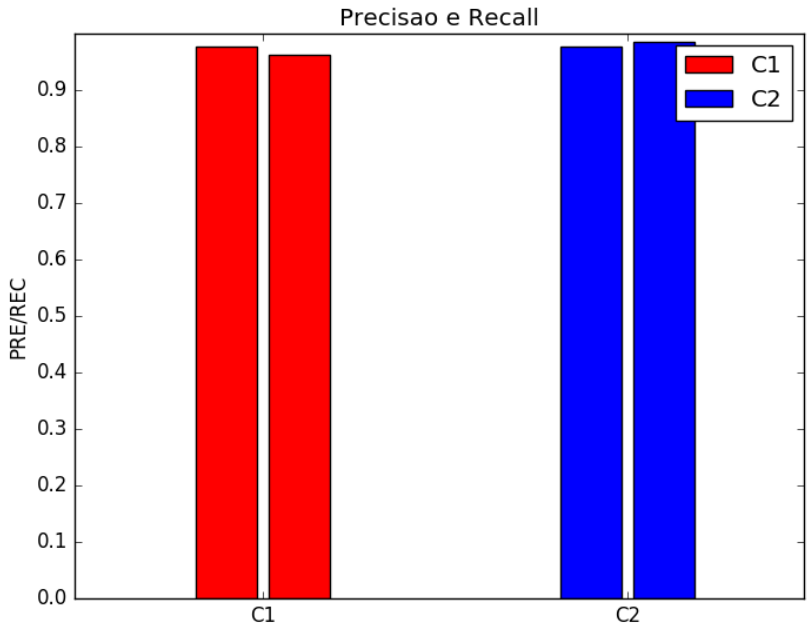


Figura: Precisão e Recall

Benchmark

	ACC	**AUC**
Baysiano Simples	0,9367	0,9266
Baysiano Quadrático	0,9578	0,9549
Regressão Logistica	0,9596	0,9563
Percepriton	0,9613	0,9615
MPL (21)	0,9735	0,9708
MPL(100)	**0,9807**	**0,9769**
MPL(10,10)	0,9789	0,9746
SVM	0,9772	0,9741