

# Informações importantes

- Lançamento de Nota da Aula Interativa: até 1 dia útil após a aula;
- Dúvidas sobre conteúdo e atividades: discussão nos fóruns;
- Prorrogação e 2º oportunidade de entrega de atividades: somente perante a apresentação do atestado médico/óbito de parentes de 1º grau;
- Atividade de reposição: somente da aula interativa;
- Perdeu alguma outra atividade? Não se preocupe! Entenda os critérios de aprovação do bootcamp no botão “Ajuda”, disponível na plataforma Canvas.

# Seleção de Modelos de Aprendizado de Máquina

Primeira Aula Interativa

Prof. Alberto de Sá Cavalcanti de Albuquerque

# Regressão

---

# Métricas de Qualidade para Regressões



- Erro médio absoluto.
- Erro médio quadrático.
- Erro mediano absoluto.
- Coeficiente de Determinação.

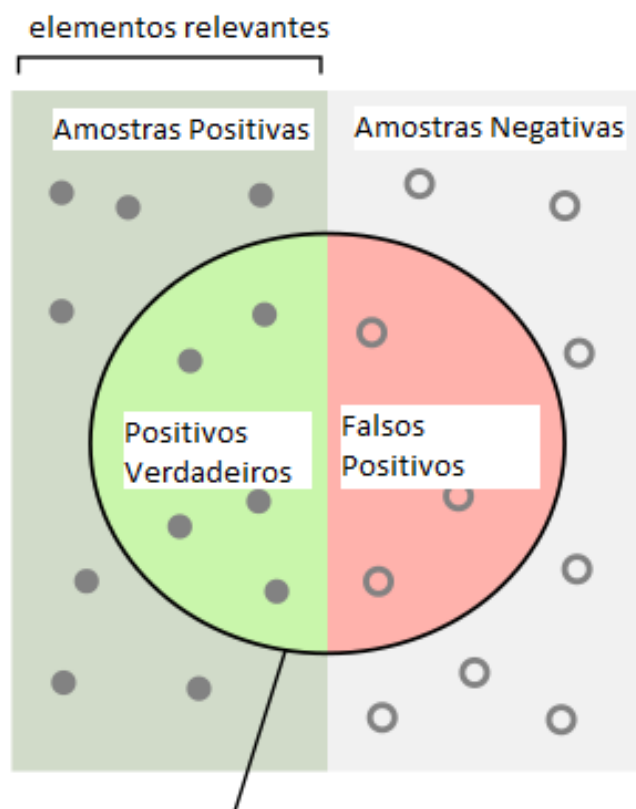
# Classificação

---

# As métricas fundamentais


- Acurácia: quantos elementos foram corretamente classificadas.
- Precisão: quantos elementos classificados positivos realmente são positivos.
- Recall: quantos elementos positivos foram percebidos?
- F-Score: Recall e Precisão igualmente importantes!

$$F_1 = 2 \times \frac{\textit{precisão} * \textit{recall}}{\textit{precisão} + \textit{recall}}$$



Elementos selecionados como positivos pelo classificador

Quanto dos elementos selecionados são realmente positivos?

$$\text{Precisão} = \frac{\text{Positivos Verdadeiros}}{\text{Positivos Verdadeiros} + \text{Falsos Positivos}}$$


Quanto itens relevantes foram selecionados?

$$\text{Recall} = \frac{\text{Positivos Verdadeiros}}{\text{Positivos Verdadeiros} + \text{Falsos Negativos}}$$


# Matriz de Confusão



	Positivos	Negativo
Positivo	Positivos verdadeiros	Falsos Negativos
Negativo	Falsos Positivos	Negativos verdadeiros



# Separador de frutas – matriz de confusão

- Os resultados: o classificador encontrou 210 maçãs, 255 pêssegos, 285 peras e 250 bananas.
- O que isso quer dizer..? Por enquanto, nada!
- Eis a matriz de confusão:

	Maçã	Pêssego	Pera	Banana
Maçã	150	90	10	0
Pêssego	30	130	80	10
Pera	30	30	180	10
Banana	0	5	15	230

# Como calcular as outras métricas usando essa?



- Acurácia!
  - Maças: 840 corretas, 160 erradas **84%**
  - Pêssegos: 755 corretas, 245 erradas **75%**
  - Peras: 825 corretas, 175 erradas **82,5%**
  - Bananas: 960 corretas, 40 erradas **96%**

# Como calcular as outras métricas usando essa?



- Precisão:
  - Maças: 150 corretas, 210 encontradas: **71.4%**
  - Pêssegos: 130 corretas, 255 encontrados: **51%**
  - Peras: 180 corretas, 285 encontradas: **63%**
  - Bananas: 230 corretas, 250 encontradas: **92%**

# Como calcular as outras métricas usando essa?



- Recall:
  - Maças: 150 corretas, 250 no total **60%**
  - Pêssegos: 130 corretas, 250 no total **52%**
  - Peras: 180 corretas, 250 no total **72%**
  - Bananas: 230 corretas, 250 no total **92%**

# Como calcular as outras métricas usando essa?

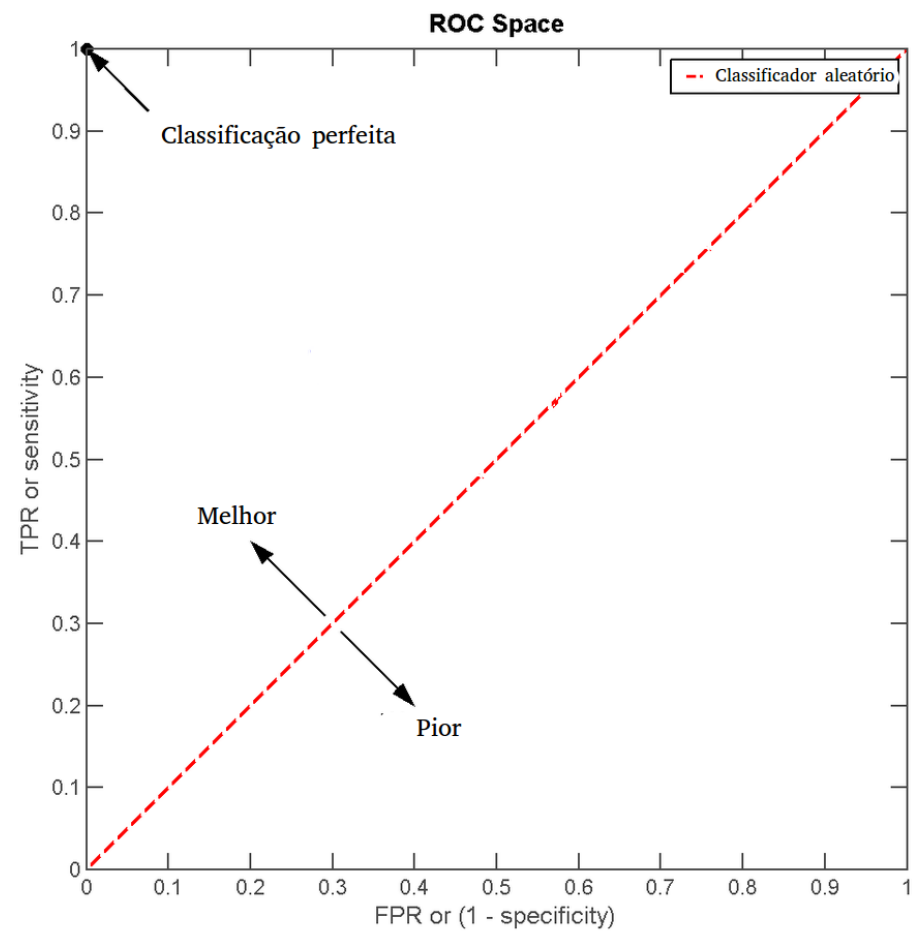


- F-Score:
  - Maçãs: **65%**
  - Pêssegos: **51,5%**
  - Peras: **67,2%**
  - Bananas: **92%**

# Área sob a curva ROC

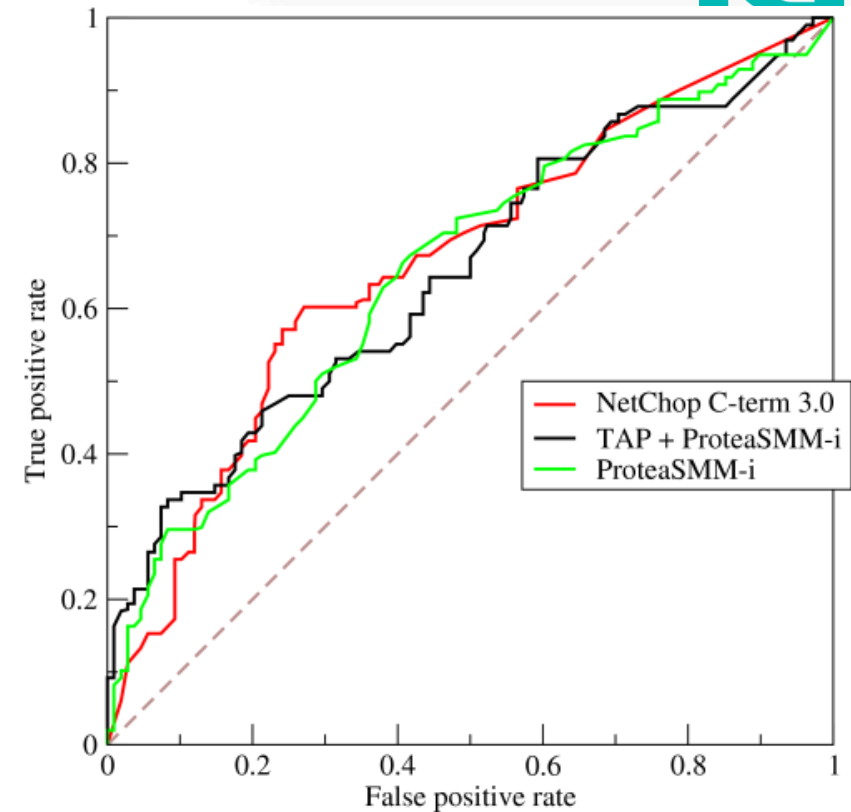


- Taxa de falsos positivos (FPR): Falsos positivos sobre soma de falsos positivos e negativos verdadeiros
  - “Quantos % dos negativos enganaram o classificador?”
- Quanto **maior** o recall, melhor!
- Quanto **menor** a FPR, melhor!



# Um segundo olhar

- A curva ROC: falando de forma mais precisa do que falei na aula. Indo além da “área sob o quadrado”. **Ela pode ser uma métrica mais robusta!**
- Deixe o seu classificador o menos rígido possível. Colete suas métricas.
- Vá deixando ele cada vez mais rígido, e vá jogando as métricas no gráfico
- Ao final, a curva ROC!
- O melhor modelo é o com a maior área sob a curva





# Classificação Multilabel

---

# Métricas



- Perda de Hamming - Quantas labels erradas?
- Perda 0-1 - Ou todas as labels dos elementos certas ou nada feito!

# Agrupamento

---

# Homogeneidade e Completude

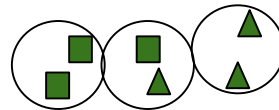
- Homogeneidade: meus agrupamentos são homogêneos?
- Completude: minha classe está em um só agrupamento?
- Acho ela uma das métricas mais difíceis.
- Ela é bem matemática.
- Fácil de fazer confusão.

# Homogeneidade e Completude

**Entropia:** Dado que  $x = (\text{Ocorrências no agrupamento} / \text{Total de pontos no agrupamento})$ , calcule:

$$\begin{array}{ll} \text{Homogeneidade:} & 1 - \frac{\text{Entropia das classes nos agrupamentos}}{\text{Entropia das classes}} & 1 - \frac{H(C|K)}{H(C)} \\ \\ \text{Completude:} & 1 - \frac{\text{Entropia dos agrupamentos nas classes}}{\text{Entropia dos agrupamentos}} & 1 - \frac{H(K|C)}{H(K)} \end{array}$$

# Homogeneidade e Completude



## Homogeneidade:

$$H(C|K) = ((2/6) \cdot \log(2/2) + (1/6) \cdot \log(1/2) + 0) + ((0 + (1/6) \cdot \log(2/2)) + (2/6) \cdot \log(2/2))$$



$$H(C) = ((3/6) \cdot \log(3/6))^2$$

$$1 - H(C|K)/H(C) = +-0.66$$

## Completude:



$$H(K|C) = ((2/6) \cdot \log(2/3) + 0) + ((1/6) \cdot \log(1/3) + (1/6) \cdot \log(1/3)) + ((2/6) \cdot \log(2/3) + 0)$$

$$H(K) = ((3/6) \cdot \log(3/6))^3$$

$$1 - H(K|C)/H(K) = +-0.41$$

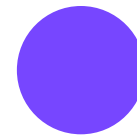
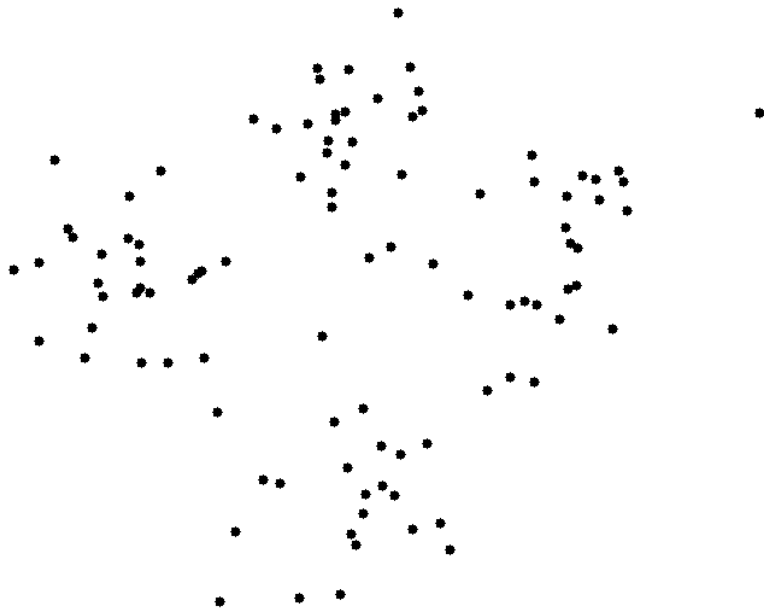


# K-Médias

- K centros.
- Cada ponto é designado para o seu centro mais próximo.
- Centro se move para o centroide dos seus pontos.
- Repita até convergir.

# K-Médias – Animação

iGTi

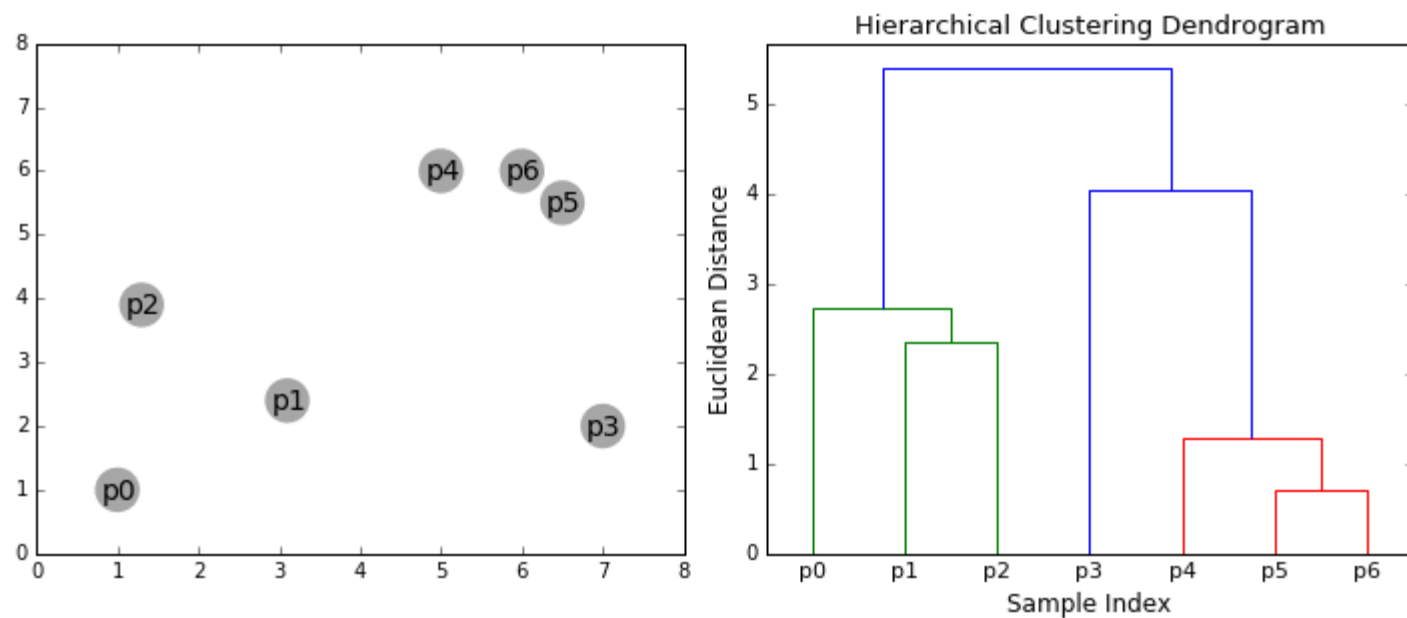




# K-Médias

- Escolha um número de clusters que você quer obter.  
(Experimente!)
- Busque o cluster mais próximo do cluster atual.  
Funda ambos em um único cluster.
- Repita até atingir o número de clusters desejado.
- **Conectividade:** pontos só podem formar clusters com seus vizinhos mais próximos.

# Agrupamento Hierárquico - Animação



**Na prática!**

---

# Calculando as métricas

- Usarei os exemplos do Scikit-Learn.
- Via de regra, para calcular as métricas, você deve ter duas matrizes de igual dimensão: a das classes previstas e a das classes obtidas!

# Erros médios quadráticos e Coeficiente de Determinação



```
>>> from sklearn.metrics import mean_squared_error
>>> y_true = [3, -0.5, 2, 7]
>>> y_pred = [2.5, 0.0, 2, 8]
>>> mean_squared_error(y_true, y_pred)
0.375
>>> y_true = [[0.5, 1], [-1, 1], [7, -6]]
>>> y_pred = [[0, 2], [-1, 2], [8, -5]]
>>> mean_squared_error(y_true, y_pred)
0.7083...
```

```
>>> from sklearn.metrics import r2_score
>>> y_true = [3, -0.5, 2, 7]
>>> y_pred = [2.5, 0.0, 2, 8]
>>> r2_score(y_true, y_pred)
0.948...
```

# Acurácia e Recall

```
>>> import numpy as np
>>> from sklearn.metrics import accuracy_score
>>> y_pred = [0, 2, 1, 3]
>>> y_true = [0, 1, 2, 3]
>>> accuracy_score(y_true, y_pred)
0.5
```

```
>>> from sklearn.metrics import recall_score
>>> y_true = [0, 1, 2, 0, 1, 2]
>>> y_pred = [0, 2, 1, 0, 0, 1]
>>> recall_score(y_true, y_pred, average='macro')
0.33...
>>> recall_score(y_true, y_pred, average='micro')
0.33...
>>> recall_score(y_true, y_pred, average='weighted')
0.33...
>>> recall_score(y_true, y_pred, average=None)
array([ 1.,  0.,  0.])
```

## average = None

retorna o recall de cada classe. Caso contrário, retorna diferentes tipos de médias do recall.

# Precisão

```
>>> from sklearn.metrics import precision_score
>>> y_true = [0, 1, 2, 0, 1, 2]
>>> y_pred = [0, 2, 1, 0, 0, 1]
>>> precision_score(y_true, y_pred, average='macro')
0.22...
>>> precision_score(y_true, y_pred, average='micro')
0.33...
>>> precision_score(y_true, y_pred, average='weighted')
...
0.22...
>>> precision_score(y_true, y_pred, average=None)
array([ 0.66...,  0.          ,  0.          ])
```

- **average = None** retorna a precisão de cada classe. Caso contrário, retorna diferentes tipos de médias da precisão.

# F1 Score



```
>>> from sklearn.metrics import f1_score
>>> y_true = [0, 1, 2, 0, 1, 2]
>>> y_pred = [0, 2, 1, 0, 0, 1]
>>> f1_score(y_true, y_pred, average='macro')
0.26...
>>> f1_score(y_true, y_pred, average='micro')
0.33...
>>> f1_score(y_true, y_pred, average='weighted')
0.26...
>>> f1_score(y_true, y_pred, average=None)
array([ 0.8,  0. ,  0. ])
```



# Matriz de Confusão e Área sob a curva ROC



```
>>> from sklearn.metrics import confusion_matrix
>>> y_true = [2, 0, 2, 2, 0, 1]
>>> y_pred = [0, 0, 2, 2, 0, 2]
>>> confusion_matrix(y_true, y_pred)
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]])
```

```
>>> import numpy as np
>>> from sklearn.metrics import roc_auc_score
>>> y_true = np.array([0, 0, 1, 1])
>>> y_scores = np.array([0.1, 0.4, 0.35, 0.8])
>>> roc_auc_score(y_true, y_scores)
0.75
```

# Perda de Hamming e Perda 0-1

```
>>> zero_one_loss(np.array([[0, 1], [1, 1]]), np.ones((2, 2)))  
0.5
```

- Quantos objetos continham labels erradas?

```
>>> hamming_loss(np.array([[0, 1], [1, 1]]), np.zeros((2, 2)))  
0.75
```

- Quantas labels foram classificadas incorretamente?

# Homogeneidade, Completude e Métrica-V



```
>>> from sklearn.metrics.cluster import completeness_score
>>> completeness_score([0, 0, 1, 1], [1, 1, 0, 0])
1.0
```

```
>>> from sklearn.metrics.cluster import homogeneity_score
>>> homogeneity_score([0, 0, 1, 1], [1, 1, 0, 0])
1.0
```

```
>>> print("%.6f" % v_measure_score([0, 0, 1, 1], [0, 0, 1, 2]))
...
0.8...
>>> print("%.6f" % v_measure_score([0, 0, 1, 1], [0, 1, 2, 3]))
...
0.66...
```

# Exemplo simples de classificação

```
1  from sklearn.ensemble import RandomForestClassifier
2  X = cancer.data # (m,n) numpy array
3  y = cancer.target # (m,) numpy array
4
5  # Create an instance of the classifier we want to use
6  clf = RandomForestClassifier()
7
8  clf.fit(X,y)
9  preds = clf.predict(X)
10
11 print(preds[:5,]) # Predicted classes
12 print(y[:5,]) # Actual classes
```

# Fontes e referências

- Acessos em 16/05/2018
- <https://deparkes.co.uk/2018/02/02/scikit-learn-simple-classification/>
- Documentação do scikit learn
- <https://dashee87.github.io/data%20science/general/Clustering-with-Scikit-with-GIFs/>
- <http://shabal.in/visuals.html>