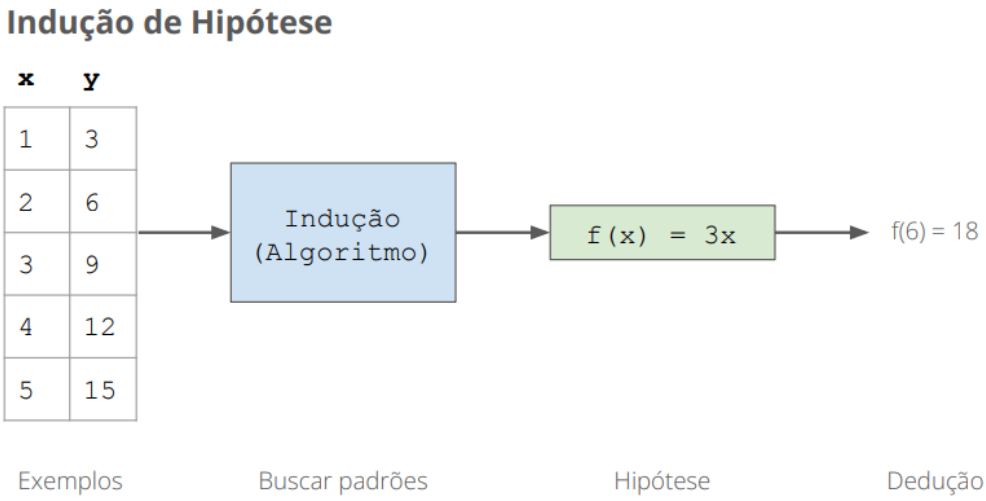
ALGORITMOS APRENDIZADO SUPERVISIONADO

O objetivo em aprendizado de máquina é utilizar um algoritmo de aprendizado para encontrar uma hipótese capaz de ser utilizada em dados novos (nunca antes vistos): **generalização**.



Viés Indutivo: conjunto de suposições (implícitas ou explícitas) realizadas por um algoritmo de aprendizado de máquina para realizar a indução de hipóteses.

No Free Lunch Theorem: não existe um algoritmo com melhor desempenho universal. Não é possível estabelecer a priori qual método de ML será melhor para a resolução de um problema específico, pois não há um algoritmo que tenha desempenho superior para todos os problemas de decisão.

K-NN

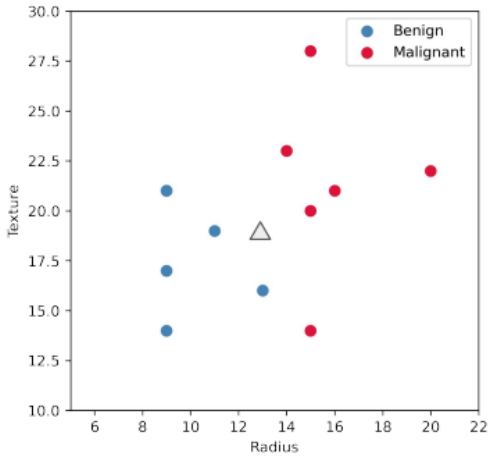
- Tem como premissa (Viés indutivo) = instâncias similares (próximas) pertencem à mesma classe (classificação) ou possuem valores semelhantes de atributo alvo (regressão).
- Não constrói um modelo preditivo.
- Aprendizado preguiçoso (lazy): só observa dados de treinamento quando precisa fazer predições para um objeto novo.
- Necessita de 3 “ingredientes”:
  1. Base de dados (treinamento).
  2. Medida de (dis)similaridade.
  3. Valor de k (número de vizinhos). K é um hiperparâmetro, ou seja, é definido experimentalmente.

Exemplo de execução com **k** = 1

△	13	18	???

1. **Calcular distâncias**
2. Ordernar instâncias
3. Selecionar k instâncias (1)
4. Atribuir rótulo majoritário

	Radius	Texture	Diagnosis	Distance (L2)
#1	14	23	Malignant	5.099
#2	15	28	Malignant	10.1980
#3	15	20	Malignant	2.8284
#4	16	21	Malignant	4.2426
#5	20	22	Malignant	8.0622
#6	15	14	Malignant	4.4721
#7	9	21	Benign	5
#8	9	14	Benign	5.6568
#10	11	19	Benign	2.2360
#11	9	17	Benign	4.1231
#12	13	16	Benign	2



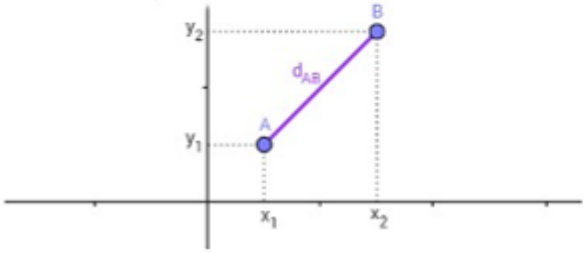
Distance functions

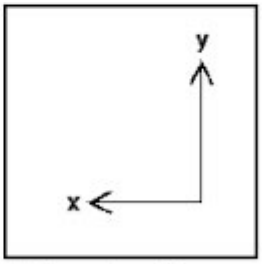
Euclidean $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$

Manhattan $\sum_{i=1}^{k}|x_i - y_i|$

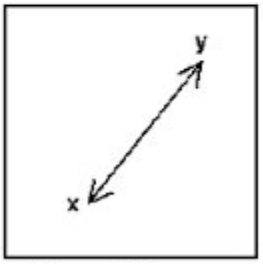
Distância Eucliada

$$d_{AB} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$





Manhattan



Euclidean

Medidas de distância utilizadas para atributos contínuos.

Para atributos discretos, recomenda-se utilizar **Simple Matching** ou **Jaccard**.

Medidas de Distância: Atributos mistos

- Quando há uma mistura entre atributos quantitativos e qualitativos, utiliza-se uma composição entre medidas (ex.: Distância Euclidiana + Coeficiente de Jaccard)

Coeficiente de Jaccard | Distância Euclidiana

	Profissão	Estado Civil	Idade	Salário	Bom Pagador
#1	Estudante	Solteiro	18	800	Não
#2	Engenheiro	Casado	24	5500	Sim
#3	Mecânico	Casado	53	2750	Sim

Para o caso acima, somar a distância dos atributos qualitativos com a distância dos atributos quantitativos.

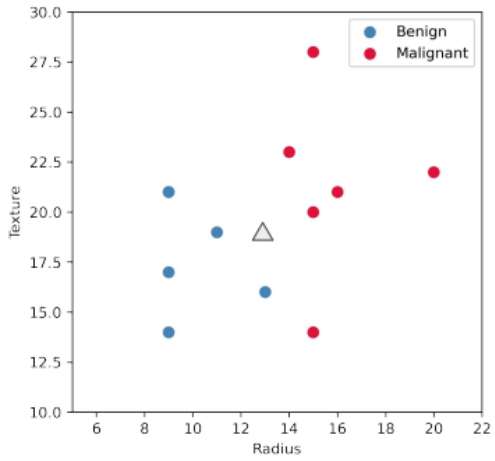
Importante normalizar os valores quantitativos.

Exemplo de execução com **k** = 1

△	13	18	???

1. Calcular distâncias
2. **Ordernar instâncias**
3. Selecionar k instâncias (1)
4. Atribuir rótulo majoritário

	Radius	Texture	Diagnosis	Distance (L2)
#12	13	16	Benign	2
#10	11	19	Benign	2.2360
#3	15	20	Malignant	2.8284
#11	9	17	Benign	4.1231
#4	16	21	Malignant	4.2426
#6	15	14	Malignant	4.4721
#7	9	21	Benign	5
#1	14	23	Malignant	5.099
#8	9	14	Benign	5.6568
#5	20	22	Malignant	8.0622
#2	15	28	Malignant	10.1980

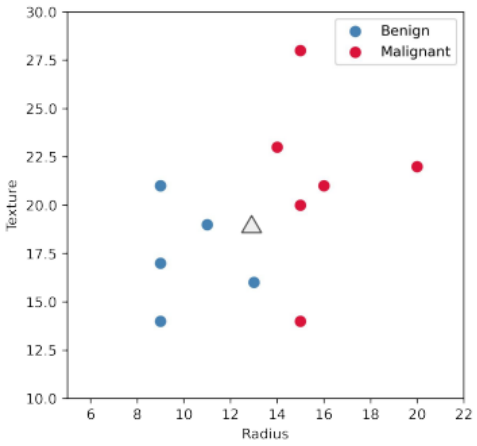


Exemplo de execução com **k** = 1

△	13	18	???

1. Calcular distâncias
2. Ordernar instâncias
3. **Selecionar k instâncias (1)**
4. Atribuir rótulo majoritário

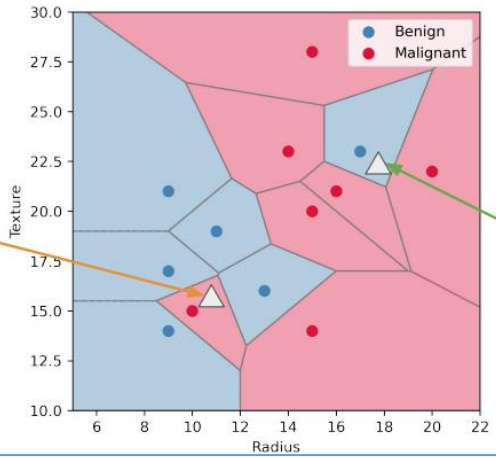
	Radius	Texture	Diagnosis	Distance (L2)
#12	13	16	Benign	2
#10	11	19	Benign	2.2360
#3	15	20	Malignant	2.8284
#11	9	17	Benign	4.1231
#4	16	21	Malignant	4.2426
#6	15	14	Malignant	4.4721
#7	9	21	Benign	5
#1	14	23	Malignant	5.099
#8	9	14	Benign	5.6568
#5	20	22	Malignant	8.0622
#2	15	28	Malignant	10.1980



O algoritmo k-NN é sensível a outliers, especialmente com **k** = 1

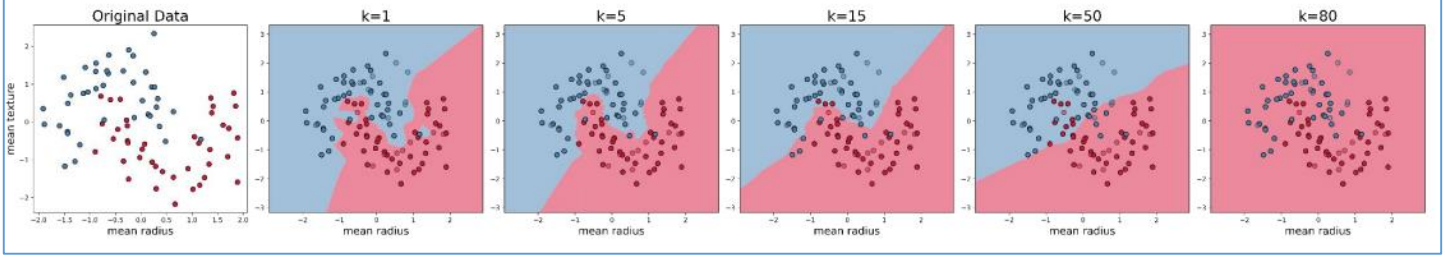
- Solução: aumentar o valor de **k**

Instância de teste classificada como "**Maligno**"



Instância de teste classificada como "**Benigno**"

- Valores **pequenos**: função de discriminação muito flexível (↓viés, ↑variância)
  - Sensível a ruído, classificação instável (**overfitting**)
- Valores **grandes**: função de discriminação muito robusta (↑viés, ↓variância)
  - Robusto a ruído, menos flexível, privilegia classe majoritária (**underfitting**)



Alternativa: **ponderar cada voto pela respectiva distância de cada instância**

- A contribuição de cada instância é multiplicada por uma **função de ponderação**
- Classe escolhida: maior soma ponderada. Permite valores maiores de **k**

Diagnosis	Distance (L2)	Weighted Distance (1/d)
Benign	2	0.500
Malignant	2.2360	0.448
Malignant	2.8284	0.354

$$\frac{1}{d(x^i, x^j)} \qquad \frac{1}{d(x^i, x^j)^2}$$

Benign: 0.5
Malignant: (0.448 + 0.354) = 0.802

O algoritmo k-NN é facilmente adaptado para a tarefa de **Regressão**

- Atributo Alvo: **média** (ponderada) dos valores dos k vizinhos mais próximos
- Exemplo: **k** = 3

$$\frac{1}{d(x^i, x^j)}$$

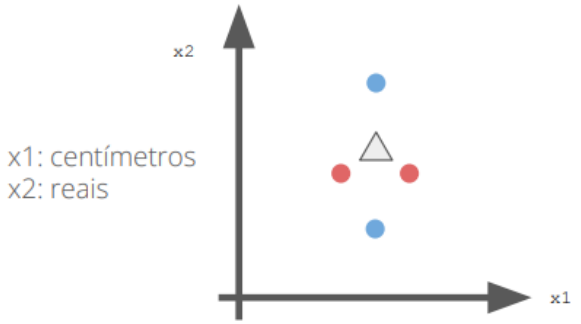
Radius	Texture	Survival	Distance (L2)	Weighted Distance (1/d)
13	16	**35**	2	0.5
11	19	**18**	2.2360	0.4472
15	20	**32**	2.8284	0.3535
9	17	**13**	4.1231	0.2425
16	21	**28**	4.2426	0.2357

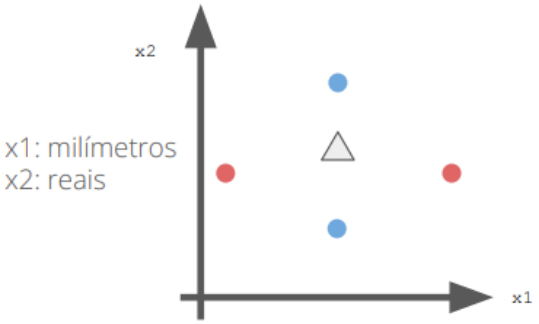
...

```
y = (0.5*35 + 0.4472*18 + 0.3535*32) / (0.5 + 0.4472 + 0.3535)
  = (17.5 + 8.0496 + 11.3120) / 1.3007
  = 36.8616 / 1.3007
  = 28.3398
```

É extremamente importante realizar uma etapa de pré-processamento!

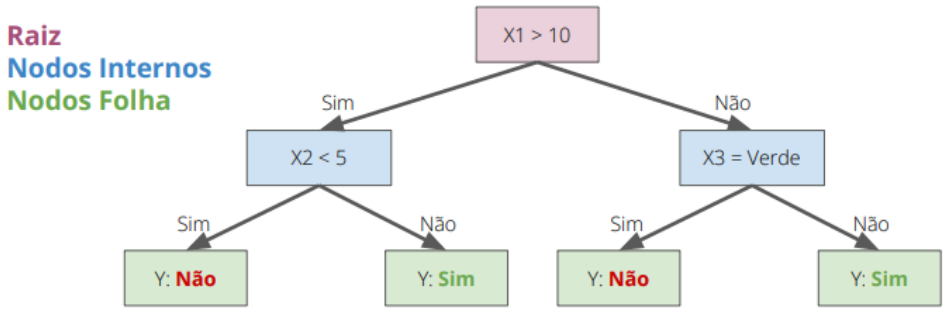
- O k-NN é afetado pela presença de **atributos irrelevantes ou redundantes**
  - Causam distorções no cálculo da distância
- É impactado pela **alta dimensionalidade** de dados (muitos atributos)
  - Em espaços altamente dimensionais (ex.: 300 atributos), a diferença entre vizinhos se torna mais sutil (pequenas variações causam baixo impacto)
- É sensível às **unidades de medida** dos atributos (solução: **normalização**)





Árvore de Decisão

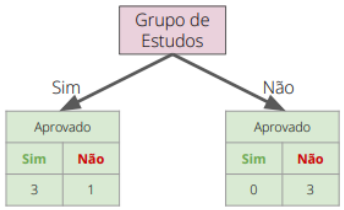
- Abordagem de modelagem preditiva para problemas de classificação e regressão.
- Algoritmo de Hunt (Top-Down): Não há backtracking (impureza é minimizada localmente em cada nó).



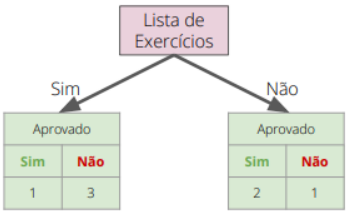
Exemplo: Quais fatores determinam a aprovação/reprovação dos alunos.

1. Coletar dados (abordagem orientada a dados).
2. Algoritmo de Indução (Top-Down).
   a) Decidir qual atributo será o próximo nodo (atributo que melhor prevê a aprovação/reprovação), utilizando **ÍNDICE GINI** para medir a IMPUREZA dos atributos.

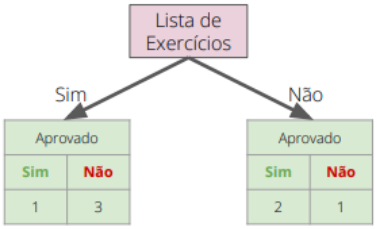
Lista de Exercícios	Grupo de Estudos	Horas de Estudo	Aprovado
Sim	Sim	2	Não
Sim	Não	5	Não
Não	Sim	6	Sim
Não	Sim	12	Sim
Sim	Sim	14	Sim
Sim	Não	20	Não
Não	Não	30	Não

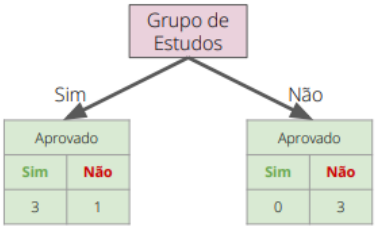


Lista de Exercícios	Grupo de Estudos	Horas de Estudo	Aprovado
Sim	Sim	2	Não
Sim	Não	5	Não
Não	Sim	6	Sim
Não	Sim	12	Sim
Sim	Sim	14	Sim
Sim	Não	20	Não
Não	Não	30	Não



Nenhum dos atributos separa perfeitamente os dados (impureza)

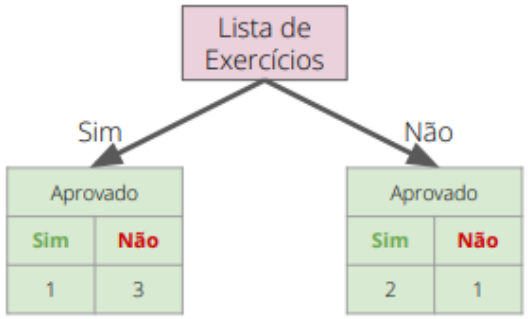




1º. Calcular impureza dos nodos folhas:

$$\text{Gini}(t) = 1 - \sum_{i=1}^{C} p(i \mid t)^2$$

- t : nodo em questão
- C: número de classes
- i: classe atual
- p(i|t): probabilidade de uma classe i em um nodo t



Gini = 0.375 Gini = 0.444

```
Gini = 1 - ( (1/(1+3))^2 + (3/(1+3))^2 )
     = 1 - ( (1/4)^2 + (3/4)^2 )
     = 1 - (0.0625 + 0.5625)
     = 1 - (0.625)
     = 0.375
```

```
Gini = 1 - ( (2/(2+1))^2 + (1/(2+1))^2 )
     = 1 - ( (2/3)^2 + (1/3)^2 )
     = 1 - (0.4444 + 0.1111)
     = 1 - (0.5555)
     = 0.4445
```

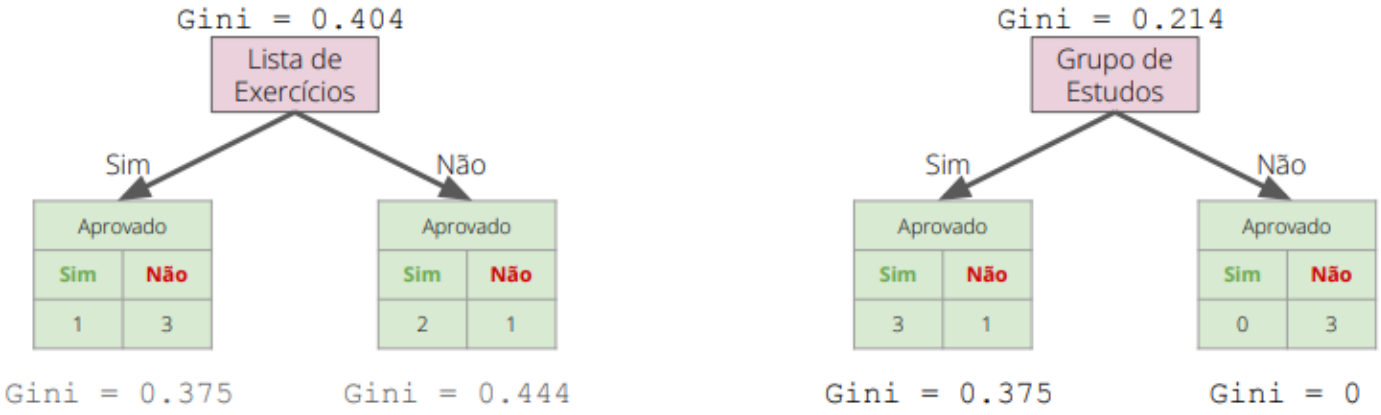
2º. Calcular impureza dos atributos:

$$\text{Gini}(a) = \sum_{i=1}^{K} \frac{n_i}{n} \text{Gini}(i)$$

a : atributo em questão
K: número de classes
i: nodo atual
ni: número de instâncias no nodo filho i
n: número de instâncias do nó pai

Exemplo: impureza do atributo “lista de exercícios”.

```
Gini = ( (4/7)*0.375 + (3/7)*0.444 )
     = ( 0.571*0.375 + 0.428*0.444 )
     = 0.404
```

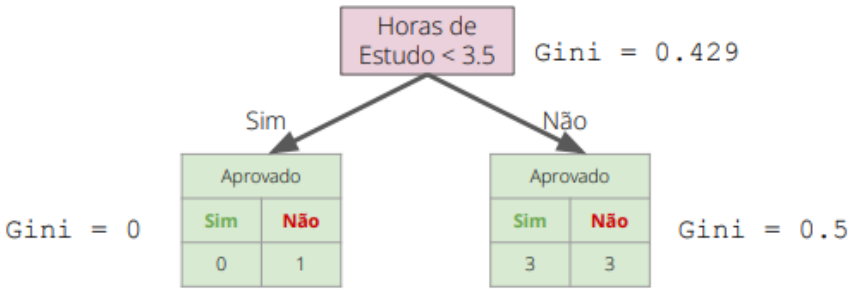


3º. Calcular impureza dos atributos contínuos:

- Ordenar linhas pelos valores (crescente)
- Calcular médias para valores intermediários → **(2+5) / 2 = 3.5**, **(5+6) / 2 = 5.5**, ...
- Calcular impureza para cada valor de média

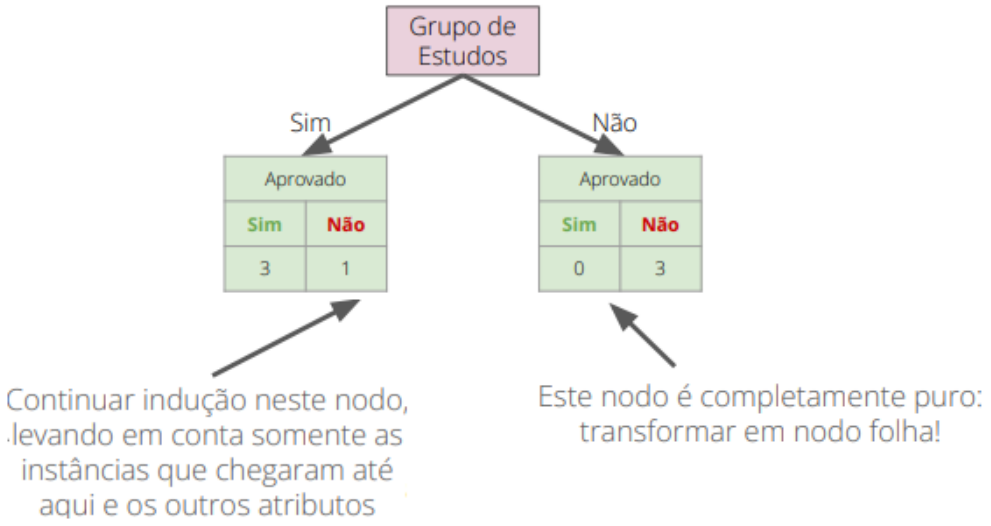
Horas de Estudo	Aprovado
2	Não
5	Não
6	Sim
12	Sim
14	Sim
20	Não
30	Não

Média	3.5	5.5	9	13	17	25
Gini	0.429	0.343	0.476	0.476	0.343	0.429

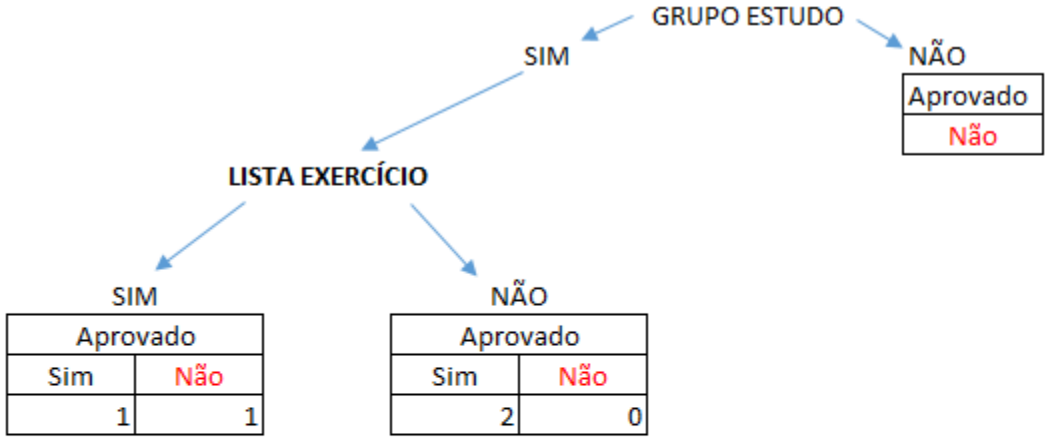


4º. Escolher melhor atributo (Gini menor = melhor):

Atributo	Índice Gini
Lista de Exercícios	0.404
Grupo de Estudos	**0.214**
Horas de Estudo < 5.5	0.343



5º. Calcular novamente impureza dos nodos folha e dos atributos, considerando somente as instâncias restantes:



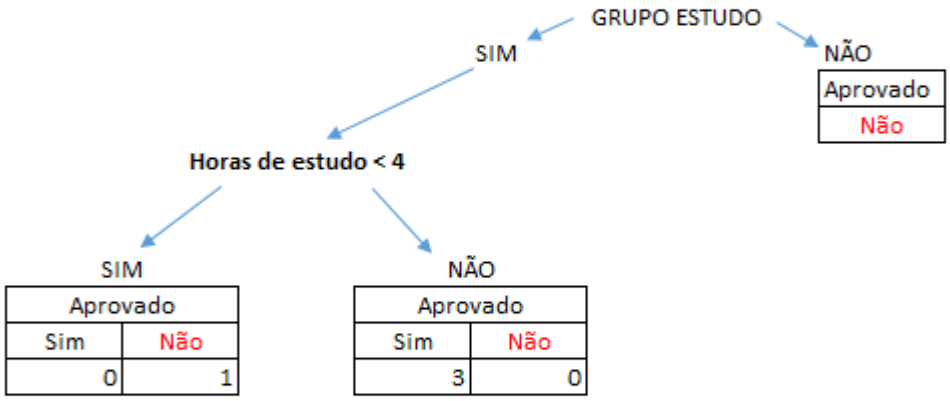
Gini: $1 - [(1 / 2)^2 + (1 / 2)^2]$
Gini: $1 - [0,25 + 0,25]$
Gini: 0,5

Gini: $1 - [(2 / 2)^2 + (0 / 2)^2]$
Gini: $1 - [1 + 0]$
Gini: 0

Gini (atributo) = $[(2 / 4) * 0,5] + [(2 / 4) * 0]$
Gini (atributo) = $[0,25] + [0]$
Gini (atributo) = 0,25

Horas de estudo	Aprovado
2	Não
6	Sim
12	Sim
14	Sim

Média	4	9	13
Gini	0	0,25	0,3375



Aprovado	
Sim	Não
0	1

Aprovado	
Sim	Não
3	0

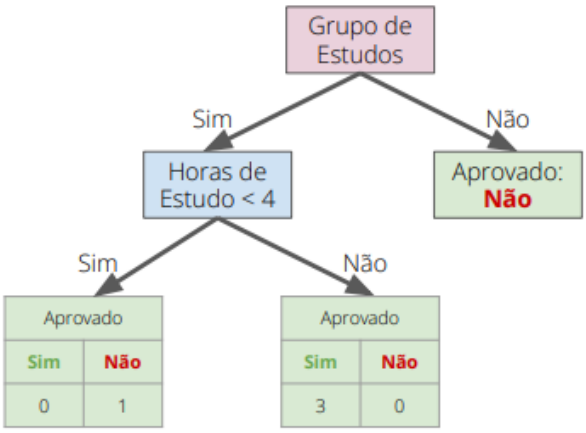
Gini: $1-[(0/1)^2+(1/1)^2]$
Gini: $1-[0+1]$
Gini: 0

Gini: $1-[(3/3)^2+(0/3)^2]$
Gini: $1-[1+0]$
Gini: 0

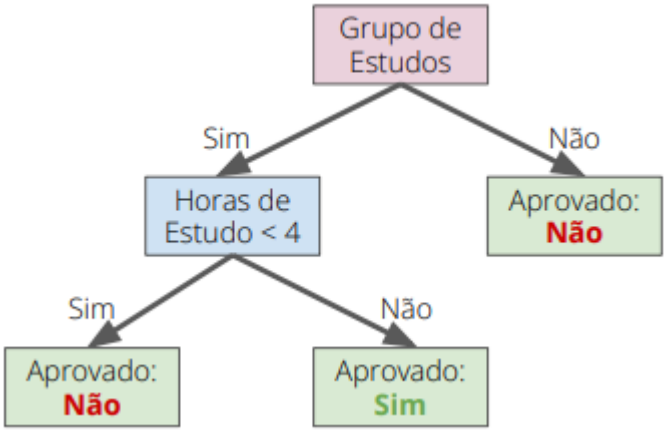
Gini (atributo) = [(1 / 4) * 0] + [(3 / 4) * 0]
Gini (atributo) = [0] + [0]
Gini (atributo) = 0

6º. Escolher novo melhor atributo (Gini menor = melhor):

Atributo	Índice Gini
Lista de Exercícios	0.25
Horas de Estudo < 4	**0**



7º. Temos a indução final, uma vez que podemos converter os nodos restantes em folha (há divisão perfeita).



- Árvores induzidas podem sofrer de overfitting e classificar novos exemplos de forma não confiável.
- Árvore induza também tende a ser muito grande e complexa, o que dificulta sua compreensão.
- Podar a árvore pode minimizar estes problemas e reduzir o erro de generalização.
- Tipos de poda:
    - Pré – poda: estratégia utilizada durante o processo de construção da árvore quando um critério é satisfeito. Interromper o crescimento da árvore segundo algum critério (valor de medida de impureza, número mínimo de instâncias atingido, etc).
    - Pós-poda: estratégia realizada após o término do processo de construção da árvore.
        - Crescer a árvore até a homogeneidade de classes.
        - Cortar os nós de maneira bottom-up.
        - Se erro de generalização melhorar após o corte, trocar sub-árvore por nó folha.

- Árvores de Decisão não possuem viés de restrição (são capazes de representar qualquer função de classificação de dados).
- Árvores de Decisão estão mais sujeitas a overfitting.
- Árvores de Decisão possuem viés de busca.
    - Atributos que geram maior redução de impureza estão nos níveis superiores.
    - Este viés implica em uma tendência a priorizar árvores mais curtas.

Aprendizado Bayesiano

- Espaço amostral (Ω): conjunto de todos possíveis resultados do fenômeno.
    - Discreto: Ω é finito. Análise envolve somatórios (∑).
    - Contínuo: Ω é infinito. Análise envolve integrais (∫).

- Probabilidade a priori (ou incondicional): probabilidade de algum evento na ausência de qualquer outra informação. Ex: P(Moeda = coroa) = 0.5 -> Neste caso, estamos desconsiderando (ou deixando de modelar) fatores físicos que influenciam o resultado, como velocidade e inclinação do arremesso, atrito do ar, tipo de solo, etc.

Probabilidade Conjunta: probabilidade de dois eventos A e B ocorrerem

- P(A): probabilidade do evento A ocorrer
- P(B): probabilidade do evento B ocorrer
- P(A∩B): probabilidade de A e B ocorrerem

P(A ∩ B) é P(A) * P(B)

Apenas quando A e B forem eventos **independentes**

P(Dado = 1) = 1/6 (0.167)
P(Dado = 2) = 1/6 (0.167)
P(Dado = 3) = 1/6 (0.167)
P(Dado = 4) = 1/6 (0.167)
P(Dado = 5) = 1/6 (0.167)
P(Dado = 6) = 1/6 (0.167)

P(DadoA = 1 ∩ DadoB = 1) = 1/6 * 1/6 = 1/36
P(DadoA = 1 ∩ DadoB = 2) = 1/6 * 1/6 = 1/36
P(DadoA = 1 ∩ DadoB = 3) = 1/6 * 1/6 = 1/36
...
P(DadoA = 6 ∩ DadoB = 6) = 1/6 * 1/6 = 1/36

Probabilidade Condicional: **Teorema de Bayes**

- Se A e B não forem eventos independentes, tem-se:

Likelihood

Prior

$$P(V|A) = \frac{P(A|V) \times P(V)}{P(A)}$$

Posterior

Evidence

Teorema de Bayes: Exemplo

Dados:

- Meningite causa rigidex no pescoço 50% das vezes
- Probabilidade a priori de se ter meningite é de 1/50.000
- Probabilidade a priori de se ter rigidez no pescoço é de 1/20

Problema: se um paciente está com rigidez no pescoço (evidência), qual a probabilidade a posteriori que o paciente esteja com meningite?

P(Rigidez | Meningite) = 0.5, P(Meningite) = 1/50.000, P(Rigidez) = 1/20
P(Meningite | Rigidez) = (P(Rigidez | Meningite) * P(Meningite)) / P(Rigidez) = (0.5 * 0.00002) / 0.05 = 0.0002 ou (0.02%)

Problema Prático: Classificação de manchas na pele como cancerígenas

- Assuma que temos duas classes, c_1 = benigno e c_2 = maligno
- Coletamos a informação de tamanho de várias manchas e o seu diagnóstico
- Qual a probabilidade de uma mancha **grande** não ser cancerígena (**benigno**)?

	Tamanho	Diagnóstico
#1	Grande	Benigno
#2	Médio	Maligno
#3	Grande	Maligno
#4	Grande	Maligno
#5	Pequeno	Benigno
#6	Pequeno	Maligno
#7	Médio	Maligno
#8	Médio	Benigno

Imputando os valores de p(G), temos valores normalizados de probabilidade (somam 1)

```
p(B | G) = p(G | B) * p(B) / p(G)
         = (1/3) * (3/8) / p(G)
         = 0.125 / 0.375 = 0.33

p(M | G) = p(G | M) * p(M) / p(G)
         = (2/5) * (5/8) / p(G)
         = 0.25 / 0.375 = 0.66
```

$$p(\mathbf{x}) = \sum_{j=1}^{k} p(\mathbf{x}|c_k)p(c_k)$$

Lei da Probabilidade Total

```
p(G) = [p(G|B)*P(B)] + [p(G|M)*P(M)]
     = [(1/3)*(3/8)] + [(2/5)*(5/8)]
     = 0.125 + 0.25
     = 0.375
```

Classificador Naive Bayes

Naive Bayes: Exemplo "Play Tennis" $p(c_j|x_1, x_2, ..., x_m) = \frac{p(c_j) \times \prod_{i=1}^{m} p(x_i|c_j)}{\prod_{i=1}^{m} p(x_i)}$

- Estimar a probabilidade de jogar tênis (ou não) com base no clima

Outlook	Temperature	Humidity	Windy	Play Tennis
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainny	Mild	High	True	No

Outlook			Temperature			Humidity			Windy			Play Tennis	
	Yes	No		Yes	No		Yes	No		Yes	No	Yes	No
Sunny	2	3	Hot	2	2	High	3	4	False	6	2	9	5
Overcast	4	0	Mild	4	2	Normal	6	1	True	3	3		
Rainy	3	2	Cool	3	1								
Sunny	2/9	3/5	Hot	2/9	2/5	High	3/9	4/5	False	6/9	2/5	9/14	5/14
Overcast	4/9	0/5	Mild	4/9	2/5	Normal	6/9	1/5	True	3/9	3/5		
Rainy	3/9	2/5	Cool	3/9	1/5								

Outlook	Temperature	Humidity	Windy	Play Tennis
Sunny	Cool	High	True	???

Para um novo dia

Outlook			Temperature			Humidity			Windy			Play Tennis	
	Yes	No		Yes	No		Yes	No		Yes	No	Yes	No
Sunny	2	3	Hot	2	2	High	3	4	False	6	2	9	5
Overcast	4	0	Mild	4	2	Normal	6	1	True	3	3		
Rainy	3	2	Cool	3	1								
Sunny	2/9	3/5	Hot	2/9	2/5	High	3/9	4/5	False	6/9	2/5	9/14	5/14
Overcast	4/9	0/5	Mild	4/9	2/5	Normal	6/9	1/5	True	3/9	3/5		
Rainy	3/9	2/5	Cool	3/9	1/5								

```
P(Yes|Sunny,Cool,High,True) = (2/9 * 3/9 * 3/9 * 3/9 * 9/14) / P(Sunny,Cool,High,True) = 0.0053 / P(Sunny,Cool,High,True)
P(No|Sunny,Cool,High,True)  = (3/5 * 1/5 * 4/5 * 3/5 * 5/14) / P(Sunny,Cool,High,True) = 0.0206 / P(Sunny,Cool,High,True)
```

Play = **No**

- O que acontece se um determinado valor de atributo não aparece na base de treinamento, mas aparece no exemplo de teste? Probabilidade correspondente será 0.
    - Solução: Estimador de Laplace:
      Adicionar 1 unidade fictícia para cada combinação de vetor-classe. Como resultado, probabilidades nunca serão zero!
      Exemplo: atributo Outlook, classe No):
        - Sunny = (3+1) / (5+3)
        - Overcast = (0+1) / (5+3)
        - Rainy = (2+1) / (5+3)
    - Nota: deve ser feito para todas as classes para não inserir viés nas probabilidades de apenas uma classe

Gradiente Descendente

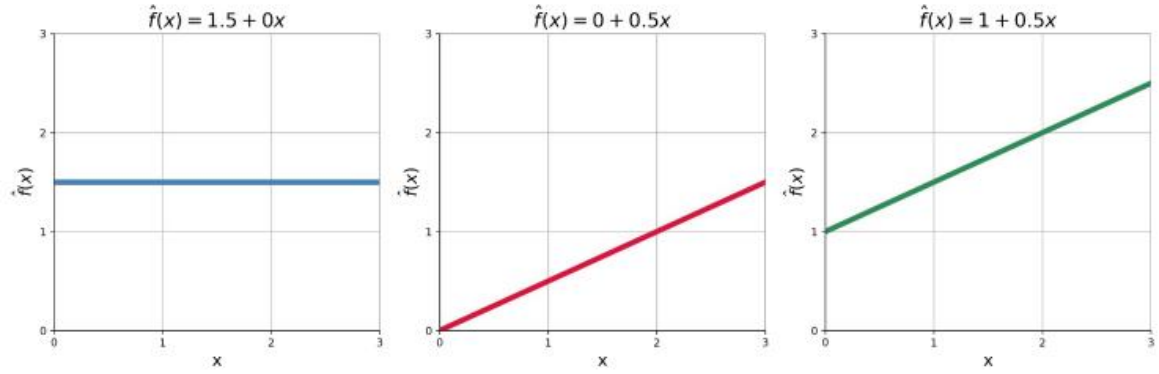
Regressão Linear **Univariada:**

- Somente um atributo. Função preditiva modela uma **relação linear**

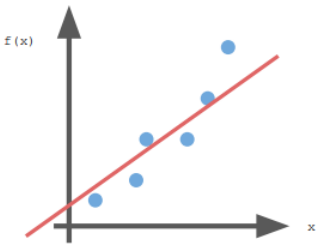
$$\hat{f}(x) = \theta_0 + \theta_1 x$$

Se $\theta_0 = 0$, nossa reta passa pela origem. θ_1 regula a angulação da reta. Para este valor de θ_1, é como se estivéssemos dizendo que o preço de uma casa é dado pela metade de seu tamanho

$\theta_0 = 1.5, \theta_1 = 0$ | $\theta_0 = 0, \theta_1 = 0.5$ | $\theta_0 = 1, \theta_1 = 0.5$



- Modelos de Regressão Linear possuem viés de restrição.
  - Dado um conjunto de dados, quais são os valores adequados de θ_0 e θ_1?
  - **Ideia:** escolher valores para que a saída do modelo se pareça (ao máximo possível) com os dados de treinamento. **Como fazer isso?**



$$\hat{f}(x^i) - f(x^i)$$

Um jeito possível para medir este "encaixe" do modelo com os dados é calcular a **diferença** entre o resultado predito e o real

Porém, notem que o resultado deste cálculo possui sinal. Se superestimarmos o valor, teremos um valor **positivo**. Se subestimarmos o valor, teremos um valor **negativo**.

Para evitarmos o "cancelamento" de erros, vamos **elevar ao quadrado a diferença** entre o valor predito e o real

$$\left(\hat{f}(x^i) - f(x^i)\right)^2$$

$$\frac{1}{N}\sum_{i=1}^{N}\left(\hat{f}(x^i) - f(x^i)\right)^2$$

Agora podemos calcular a diferença entre uma predição individual do modelo e o resultado real. Como queremos que nosso modelo se aproxime de **todos** os resultados (em média), vamos calcular os resíduos de todas as instâncias e calcular o **valor médio**

Assim, temos uma maneira de quantificar o "custo" de predição. Quanto maior for este valor, pior o modelo. Quando este valor atinge 0, o modelo não comete erros

Função de Custo: Erro Quadrático Médio

Podemos dizer, então, que o que queremos é **minimizar** esta função de custo. O modelo possui controle de seus parâmetros, então queremos que ele encontre valores de maneira a minimizar o erro quadrático médio

$$J(\theta_0, \theta_1) = \frac{1}{2N}\sum_{i=1}^{N}\left(\hat{f}(x^i) - f(x^i)\right)^2$$

Esta divisão por 2 está ali por conveniência matemática (simplificar a derivada)

$$\min_{\theta_0,\theta_1} J(\theta_0, \theta_1)$$

Como **minimizar J**(θ_0, θ_1)? - **Gradiente Descendente** (Intuição)

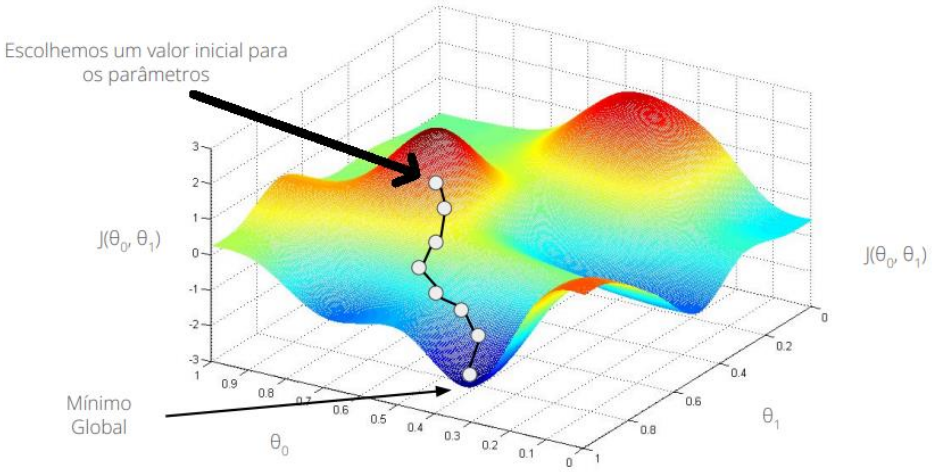
- "Olhar para todas as direções e dar um passo na direção de maior declive"
  - **Derivada:** descobrir a direção de maior declive
  - α: Hiperparâmetro que regula o tamanho do passo

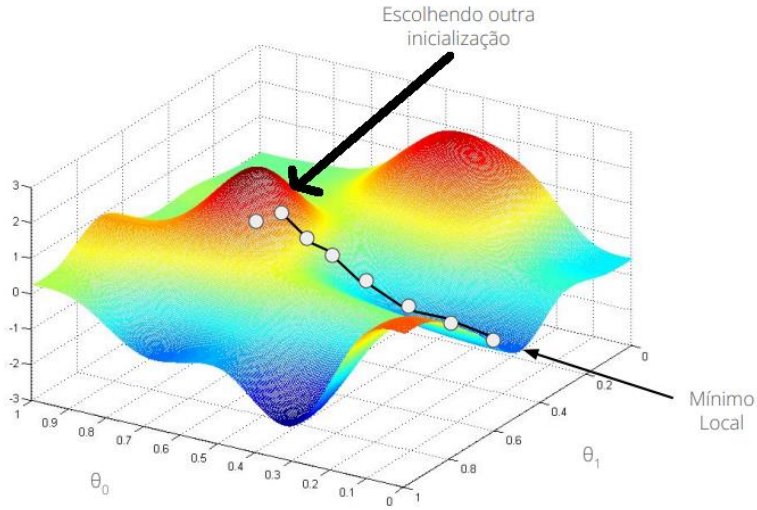
Repetir, até convergir: {

$$\theta_0 := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

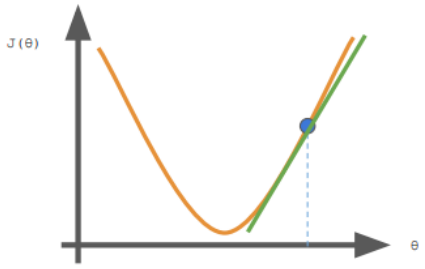
}





Como **minimizar J**(θ_0, θ_1)? - **Gradiente Descendente** (Entendendo a derivada)

- Assuma a existência de uma função de custo com apenas um parâmetro θ

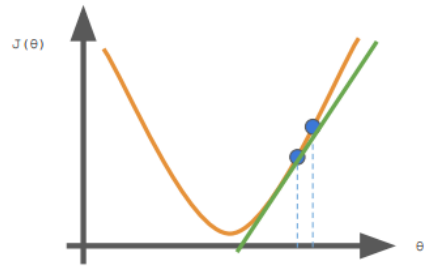


$$\theta := \theta - \alpha \frac{d}{d\theta} J(\theta)$$

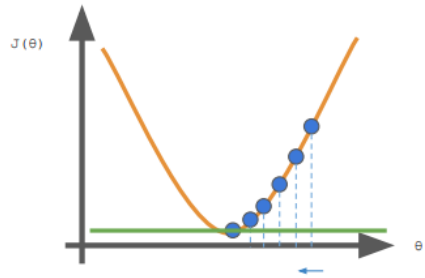
A derivada da função J com relação a θ fornece a inclinação da reta que é tangente à função no ponto em questão

$\theta := \theta - \alpha$ **(número positivo)**

Notem que a inclinação desta reta é positiva, então o resultado deste cálculo de derivada será positivo



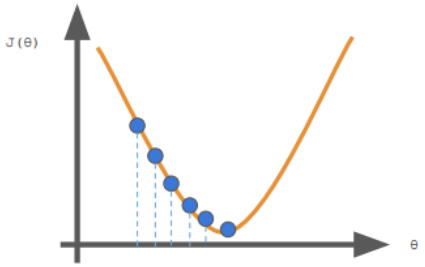
Como o valor de α é positivo e pequeno, atualização de θ será negativa e pequena



Até que o valor da nossa derivada atinge 0 (mínimo global)

$\theta := \theta - \alpha$ **(número positivo)**

O gradiente aponta para o máximo da função, então damos um passo na direção contrária



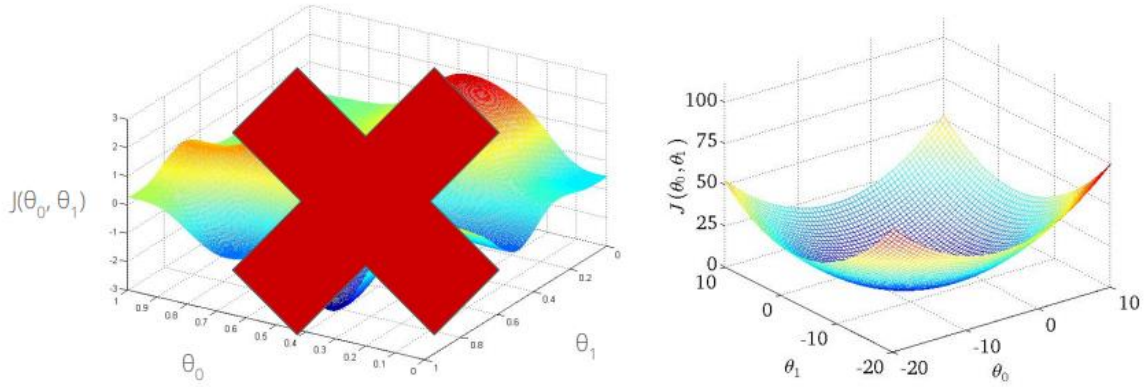
Se tivéssemos escolhido um valor inicial de θ à esquerda, o valor da derivada será negativo, então daremos um passo no sentido oposto (sentido positivo)

$\theta := \theta - \alpha$ **(número negativo)**

- Como saber que tamanho de passo devo dar?
  - A escolha da taxa de aprendizado (α) é um dos hiperparâmetros mais importantes em algoritmos de aprendizado de máquina baseado em otimização.
  - Valores muito baixos: convergência muito lenta.
  - ○ Valores muito altos: risco de divergência.

Loss Landscape da função de custo de Regressão Linear

- Nossa função de custo J pode ter este formato, certo? **ERRADO**
- O Erro Quadrático Médio define um problema de otimização **convexo**
  - Possui um único mínimo (que é o mínimo global)
  - "Bowl-shaped" - formato de tigela



Regressão Linear Multi-variada: **Notação**

- x_j^i: valor do j-ésimo atributo da i-ésimo instância
- Vamos assumir a existência de um novo atributo: $x_0 = [1, 1, ..., 1]^N$
- Considere o vetor de parâmetros $\Theta = [\theta_0, \theta_0, ..., \theta_m]$

$$\hat{f}(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + ... + \theta_m x_m$$

$$\hat{f}(\mathbf{x}) = \theta_0 + \sum_{i=1}^{m} \theta_i x_i$$

$$\hat{f}(\mathbf{x}) = \sum_{i=0}^{m} \theta_i x_i$$

$$\hat{f}(\mathbf{x}) = \Theta^T \mathbf{x}$$

Regressão Linear Multi-variada: **Gradiente Descendente**

Repetir, até convergir: {

$$\theta_j := \theta_j - \alpha \frac{1}{N} \sum_{i=1}^{N} \left(\Theta^T \mathbf{x}^i - f(\mathbf{x}^i) \right) x_j^i$$

Para j = [0, 1, ..., m]

}

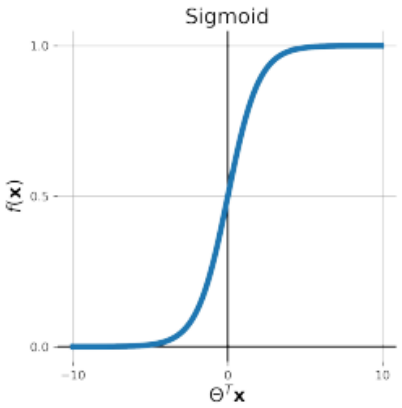
- Heurística para escolha de α
    - Começar com valores pequenos (ex.: 0.001).
    - Incrementar o valor por algum fator (ex.: 5, 10, etc.) para agilizar convergência (sempre conferindo se os valores estão decrescendo a cada iteração).

Regressão Logística

- É um algoritmo de classificação.

- Regressão Linear: $\hat{f}(\mathbf{x}) = \Theta^T \mathbf{x}$
- Qual o modelo para regressão logística?
  - σ: função de ativação **Sigmóide**. Garante saída 0 ⩽ f(x) ⩽ 1

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$



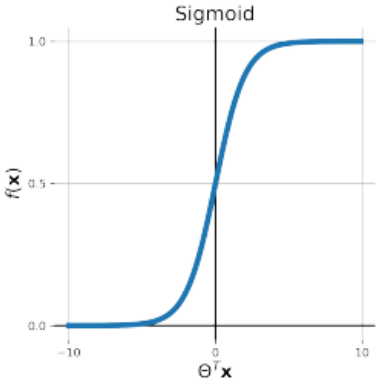
$$\hat{f}(\mathbf{x}) = \sigma(\Theta^T \mathbf{x})$$

$$\hat{f}(\mathbf{x}) = \frac{1}{1 + e^{-(\Theta^T \mathbf{x})}}$$

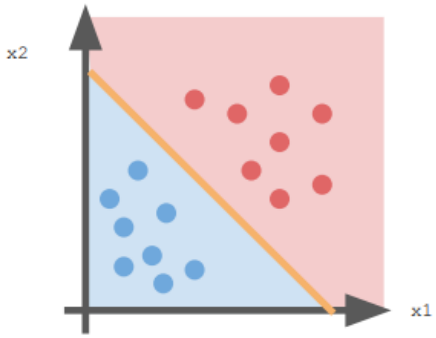
- Como temos uma saída entre [0, 1], podemos ter uma **interpretação probabilística** da saída do modelo

- Se $\sigma(\Theta^T \mathbf{x}) \geqslant 0.5$: classe positiva
- Caso contrário, classe negativa

- O que isso nos diz a respeito de $\Theta^T \mathbf{x}$?
  - Se $\Theta^T \mathbf{x} \geqslant 0$, classe positiva



Exemplo: problema com dois atributos



Instâncias exatamente no hiperplano separador possuem valor $\sigma(\Theta^T \mathbf{x}) = 0.5$

Toda instância onde $\sigma(\Theta^T \mathbf{x}) < 0.5$ estará abaixo do hiperplano e será classificada como **negativo**

Toda instância onde $\sigma(\Theta^T \mathbf{x}) \geqslant 0.5$ estará acima do hiperplano e será classificada como **positivo**

Regressão Logística: Entropia Cruzada Binária

$$J(\Theta) = -\frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}^i)\log(\hat{f}(\mathbf{x}^i)) + (1 - f(\mathbf{x}^i))\log(1 - \hat{f}(\mathbf{x}^i))$$

- Função de custo convexa. Como minimizar?
  - **Gradiente Descendente!**
  - Possui a mesma derivada que a função de custo da Regressão Linear!

Repetir, até convergir: {

$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\Theta) \longrightarrow \theta_j := \theta_j - \alpha\frac{1}{N}\sum_{i=0}^{N}\left(\hat{f}(\mathbf{x}^i) - f(\mathbf{x}^i)\right)x_j^i$$

Para j = [0, 1, ..., m]

}