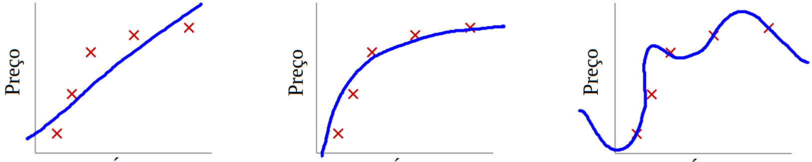
Aula 12 - Seleção e Avaliação de Modelos II

João Florindo

Instituto de Matemática, Estatística e Computação Científica
Universidade Estadual de Campinas - Brasil
florindo@unicamp.br

Outline

Cenários Possíveis



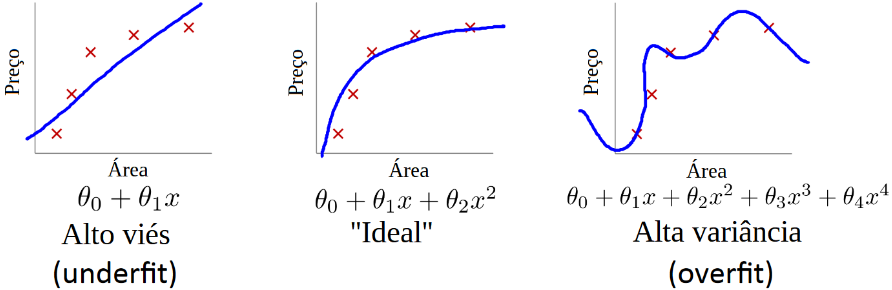
$\theta_0 + \theta_1 x$
Alto viés
(underfit)

$\theta_0 + \theta_1 x + \theta_2 x^2$
"Ideal"

$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$
Alta variância
(overfit)

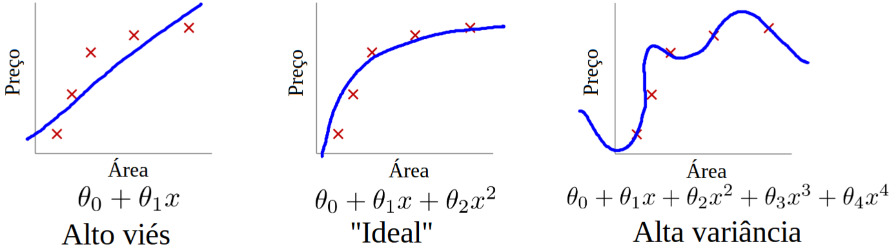
- Modelo muito simples - não se ajusta aos dados - *underfitting* - **viés** alto.
- Modelo muito complexo - ajuste perfeito ao treino, mas sem generalização no teste/validação - *overfitting* - **variância** alta.
- Modelo ideal - boa generalização.

Cenários Possíveis



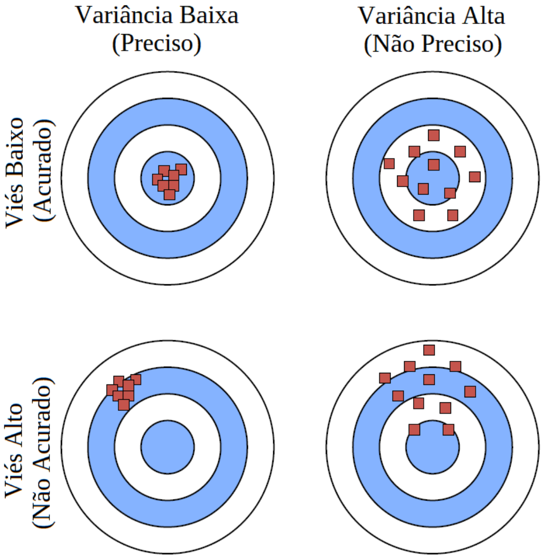
- Modelo muito simples - não se ajusta aos dados - *underfitting* - **viés** alto.
- Modelo muito complexo - ajuste perfeito ao treino, mas sem generalização no teste/validação - *overfitting* - **variância** alta.
- Modelo ideal - boa generalização.

Cenários Possíveis



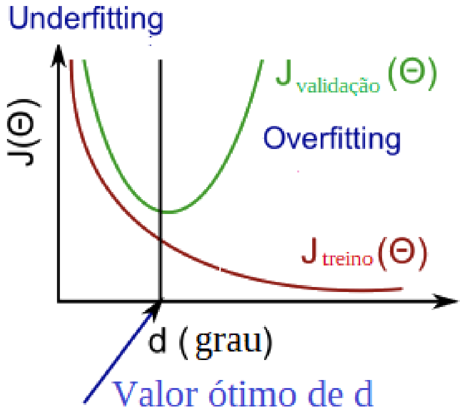
(underfit) (overfit)

- Modelo muito simples - não se ajusta aos dados - *underfitting* - **viés** alto.
- Modelo muito complexo - ajuste perfeito ao treino, mas sem generalização no teste/validação - *overfitting* - **variância** alta.
- Modelo ideal - boa generalização.


Variância Baixa
(Preciso)
Variância Alta
(Não Preciso)
Viés Baixo
(Acurado)
Viés Alto
(Não Acurado)

Para identificarmos qual problema está prevalecendo, comparamos o erro de treino com o de validação (ou de teste):

- Erro de treino: $J_{treino}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$
- Erro de validação: $J_{cv}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$



Viés alto (*underfit*):

- $J_{treino}(\theta)$ alto
- $J_{cv}(\theta) \approx J_{trenio}(\theta)$

Variância alta (*overfit*):

- $J_{treino}(\theta)$ baixo
- $J_{cv}(\theta) \gg J_{trenio}(\theta)$.

Viés/Variância e Regularização

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2.$$

Suponha que $h_\theta(x)$ original seja um polinômio de alto grau. Então:

- Se $\lambda \to \infty$, temos $\theta_j \to 0$ (reta-*underfit*).
- Se $\lambda = 0$, não há regularização (*overfit*).
- Buscamos λ ótimo!

Viés/Variância e Regularização

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2.$$

Suponha que $h_\theta(x)$ original seja um polinômio de alto grau. Então:

- Se $\lambda \to \infty$, temos $\theta_j \to 0$ (reta-*underfit*).
- Se $\lambda = 0$, não há regularização (*overfit*).
- Buscamos λ ótimo!

Viés/Variância e Regularização

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2.$$

Suponha que $h_\theta(x)$ original seja um polinômio de alto grau. Então:

- Se $\lambda \to \infty$, temos $\theta_j \to 0$ (reta-*underfit*).
- Se $\lambda = 0$, não há regularização (*overfit*).
- Buscamos λ ótimo!

Escolha de λ

- Dada uma função de hipótese complexa (ex. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$).
- Definimos a função de custo regularizada:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2.$$

- Definimos os erros sem regularização:

$$J_{treino}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m} (h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

$$J_{teste}(\theta) = \frac{1}{2m_{teste}} \sum_{i=1}^{m} (h_\theta(x_{teste}^{(i)}) - y_{teste}^{(i)})^2$$

Escolha de λ

- Dada uma função de hipótese complexa (ex. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$).
- Definimos a função de custo regularizada:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2.$$

- Definimos os erros sem regularização:

$$J_{treino}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J_{cv}(\theta) = \frac{1}{2m_{cv}}\sum_{i=1}^{m}(h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

$$J_{teste}(\theta) = \frac{1}{2m_{teste}}\sum_{i=1}^{m}(h_\theta(x_{teste}^{(i)}) - y_{teste}^{(i)})^2$$

Escolha de λ

- Dada uma função de hipótese complexa (ex. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$).
- Definimos a função de custo regularizada:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2.$$

- Definimos os erros sem regularização:

$$J_{treino}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J_{cv}(\theta) = \frac{1}{2m_{cv}}\sum_{i=1}^{m}(h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

$$J_{teste}(\theta) = \frac{1}{2m_{teste}}\sum_{i=1}^{m}(h_\theta(x_{teste}^{(i)}) - y_{teste}^{(i)})^2$$

Escolha de λ

- Testar sequência de valores de λ.
- EX.: 0, 0.01, 0.02, 0.04, 0.08, $\cdots$, 10.24 (ou 10.00).
- Para cada λ obter parâmetros como usual:

$$\operatorname*{argmin}_{\theta} J(\theta)$$

  e calcular $J_{cv}(\theta)$ correspondente.
- Escolher θ cujo λ resultou no menor $J_{cv}(\theta)$.
- Usar estes mesmos parâmetros no cálculo de $J_{teste}(\theta)$.

Escolha de λ

- Testar sequência de valores de λ.
- EX.: 0, 0.01, 0.02, 0.04, 0.08, $\cdots$, 10.24 (ou 10.00).
- Para cada λ obter parâmetros como usual:

$$\operatorname*{argmin}_{\theta} J(\theta)$$

  e calcular $J_{cv}(\theta)$ correspondente.
- Escolher θ cujo λ resultou no menor $J_{cv}(\theta)$.
- Usar estes mesmos parâmetros no cálculo de $J_{teste}(\theta)$.

Escolha de λ

- Testar sequência de valores de λ.
- EX.: 0, 0.01, 0.02, 0.04, 0.08, $\cdots$, 10.24 (ou 10.00).
- Para cada λ obter parâmetros como usual:

$$\underset{\theta}{\operatorname{argmin}}\, J(\theta)$$

  e calcular $J_{cv}(\theta)$ correspondente.
- Escolher θ cujo λ resultou no menor $J_{cv}(\theta)$.
- Usar estes mesmos parâmetros no cálculo de $J_{teste}(\theta)$.

Escolha de λ

- Testar sequência de valores de λ.
- EX.: 0, 0.01, 0.02, 0.04, 0.08, $\cdots$, 10.24 (ou 10.00).
- Para cada λ obter parâmetros como usual:

$$\underset{\theta}{\text{argmin}}\, J(\theta)$$

  e calcular $J_{cv}(\theta)$ correspondente.
- Escolher θ cujo λ resultou no menor $J_{cv}(\theta)$.
- Usar estes mesmos parâmetros no cálculo de $J_{teste}(\theta)$.

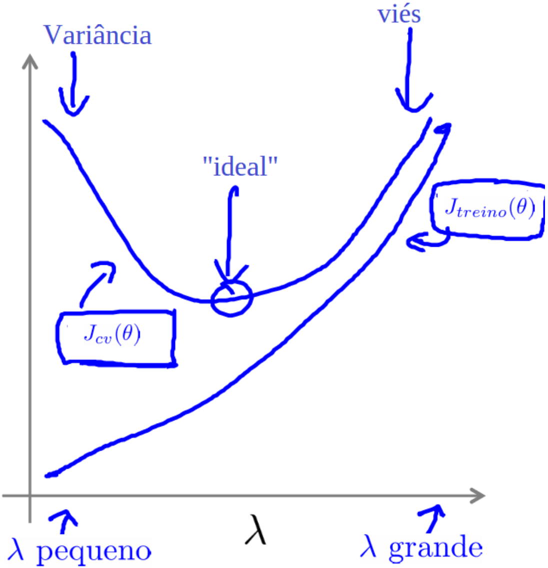
Escolha de λ

- Testar sequência de valores de λ.
- EX.: 0, 0.01, 0.02, 0.04, 0.08, $\cdots$, 10.24 (ou 10.00).
- Para cada λ obter parâmetros como usual:

$$\underset{\theta}{\operatorname{argmin}}\, J(\theta)$$

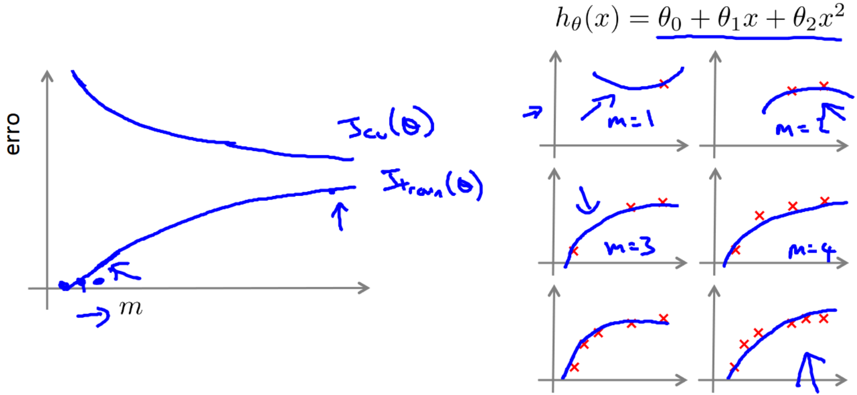
  e calcular $J_{cv}(\theta)$ correspondente.
- Escolher θ cujo λ resultou no menor $J_{cv}(\theta)$.
- Usar estes mesmos parâmetros no cálculo de $J_{teste}(\theta)$.

Influência do λ em J_{treino} e J_{cv}

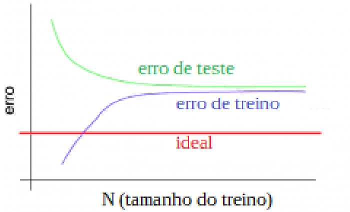


Curvas de Aprendizado

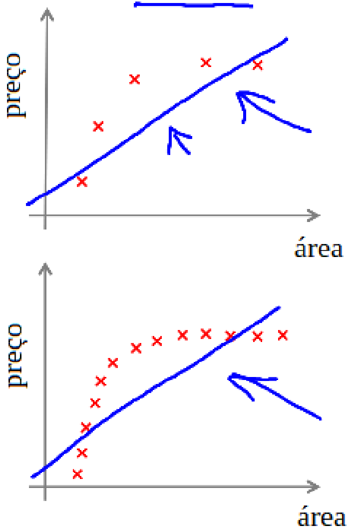
Erro de treino e de validação (ou teste) em função do número *m* de amostras de treino.



Curvas de Aprendizado - Alto Viés

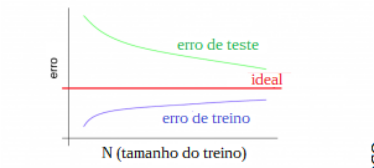


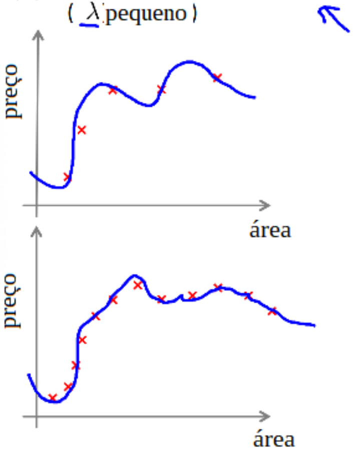
$$h_\theta(x) = \theta_0 + \theta_1 x$$



Curvas de Aprendizado - Alta Variância

$$h_\theta(x) = \theta_0 + \theta_1 x + \cdots + \theta_{100} x^{100}$$





Passos Gerais

- Obter mais exemplos de treinamento → corrige variância alta
- Usar menos atributos → corrige variância alta
- Usar mais atributos → corrige viés alto
- Adicionar atributos polinomiais (x_1^2, x_2^2, $x_1 x_2$, etc.) → corrige viés alto
- Aumentar λ → corrige viés alto
- Diminuir λ → corrige variância alta

Passos Gerais

- Obter mais exemplos de treinamento $\rightarrow$ corrige variância alta
- Usar menos atributos $\rightarrow$ corrige variância alta
- Usar mais atributos $\rightarrow$ corrige viés alto
- Adicionar atributos polinomiais (x_1^2, x_2^2, $x_1 x_2$, etc.) $\rightarrow$ corrige viés alto
- Aumentar λ $\rightarrow$ corrige viés alto
- Diminuir λ $\rightarrow$ corrige variância alta

Passos Gerais

- Obter mais exemplos de treinamento $\rightarrow$ corrige variância alta
- Usar menos atributos $\rightarrow$ corrige variância alta
- Usar mais atributos $\rightarrow$ corrige viés alto
- Adicionar atributos polinomiais (x_1^2, x_2^2, x_1x_2, etc.) $\rightarrow$ corrige viés alto
- Aumentar λ $\rightarrow$ corrige viés alto
- Diminuir λ $\rightarrow$ corrige variância alta

Passos Gerais

- Obter mais exemplos de treinamento $\rightarrow$ corrige variância alta
- Usar menos atributos $\rightarrow$ corrige variância alta
- Usar mais atributos $\rightarrow$ corrige viés alto
- Adicionar atributos polinomiais (x_1^2, x_2^2, x_1x_2, etc.) $\rightarrow$ corrige viés alto
- Aumentar λ $\rightarrow$ corrige viés alto
- Diminuir λ $\rightarrow$ corrige variância alta

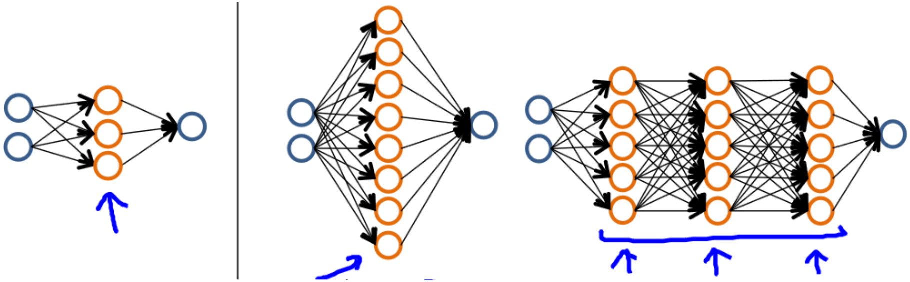
Passos Gerais

- Obter mais exemplos de treinamento $\rightarrow$ corrige variância alta
- Usar menos atributos $\rightarrow$ corrige variância alta
- Usar mais atributos $\rightarrow$ corrige viés alto
- Adicionar atributos polinomiais (x_1^2, x_2^2, $x_1 x_2$, etc.) $\rightarrow$ corrige viés alto
- Aumentar λ $\rightarrow$ corrige viés alto
- Diminuir λ $\rightarrow$ corrige variância alta

Passos Gerais

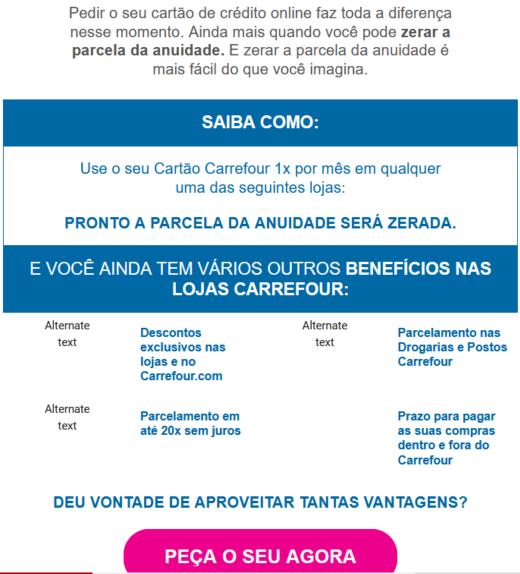
- Obter mais exemplos de treinamento $\rightarrow$ corrige variância alta
- Usar menos atributos $\rightarrow$ corrige variância alta
- Usar mais atributos $\rightarrow$ corrige viés alto
- Adicionar atributos polinomiais (x_1^2, x_2^2, $x_1 x_2$, etc.) $\rightarrow$ corrige viés alto
- Aumentar λ $\rightarrow$ corrige viés alto
- Diminuir λ $\rightarrow$ corrige variância alta

Redes Neurais e *Overfitting*



Outline

Exemplo Prático - Anti-Spam



Exemplo Prático - Anti-Spam

- Definir os atributos relevantes do email (x): 100 palavras indicativas de *spam*/não-*spam* e

$$x_j = \begin{cases} 1 & \text{se a palavra } j \text{ aparece no email} \\ 0 & \text{caso contrário.} \end{cases}$$

Exemplo:

$$x \in \mathbb{R}^{100} = \begin{bmatrix} \text{florindo} \\ \text{saiba} \\ \text{parcela} \\ \text{fatura} \\ \vdots \\ \text{agora} \\ \vdots \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ \vdots \\ 1 \\ \vdots \end{bmatrix}$$

Exemplo Prático - Anti-Spam

Estratégias para reduzir o erro:

- Coletar mais dados.
- Desenvolver atributos sofisticados com base no roteamento do email (cabeçalho).
- Desenvolver atributos sofisticados com base no corpo do email.
- Desenvolver algoritmos sofisticados de pré-processamento, p.ex., detectando erros de ortografia.

Exemplo Prático - Anti-Spam

Estratégias para reduzir o erro:

- Coletar mais dados.
- Desenvolver atributos sofisticados com base no roteamento do email (cabeçalho).
- Desenvolver atributos sofisticados com base no corpo do email.
- Desenvolver algoritmos sofisticados de pré-processamento, p.ex., detectando erros de ortografia.

Exemplo Prático - Anti-Spam

Estratégias para reduzir o erro:

- Coletar mais dados.
- Desenvolver atributos sofisticados com base no roteamento do email (cabeçalho).
- Desenvolver atributos sofisticados com base no corpo do email.
- Desenvolver algoritmos sofisticados de pré-processamento, p.ex., detectando erros de ortografia.

Exemplo Prático - Anti-Spam

Estratégias para reduzir o erro:

- Coletar mais dados.
- Desenvolver atributos sofisticados com base no roteamento do email (cabeçalho).
- Desenvolver atributos sofisticados com base no corpo do email.
- Desenvolver algoritmos sofisticados de pré-processamento, p.ex., detectando erros de ortografia.

Análise de Erro

- Começar por um método simples de implementação rápida e testá-lo no conjunto de validação.
- Fazer curvas de aprendizado para ver se mais dados, atributos, etc. ajudam.
- Análise de erro: Examinar manualmente os exemplos no conjunto de validação para os quais o algoritmo erra.
- Tentar identificar alguma tendência nesses erros para, por exemplo, adicionar um atributo que trate aqueles casos.

Análise de Erro

- Começar por um método simples de implementação rápida e testá-lo no conjunto de validação.
- Fazer curvas de aprendizado para ver se mais dados, atributos, etc. ajudam.
- Análise de erro: Examinar manualmente os exemplos no conjunto de validação para os quais o algoritmo erra.
- Tentar identificar alguma tendência nesses erros para, por exemplo, adicionar um atributo que trate aqueles casos.

Análise de Erro

- Começar por um método simples de implementação rápida e testá-lo no conjunto de validação.
- Fazer curvas de aprendizado para ver se mais dados, atributos, etc. ajudam.
- Análise de erro: Examinar manualmente os exemplos no conjunto de validação para os quais o algoritmo erra.
- Tentar identificar alguma tendência nesses erros para, por exemplo, adicionar um atributo que trate aqueles casos.

Análise de Erro

- Começar por um método simples de implementação rápida e testá-lo no conjunto de validação.
- Fazer curvas de aprendizado para ver se mais dados, atributos, etc. ajudam.
- Análise de erro: Examinar manualmente os exemplos no conjunto de validação para os quais o algoritmo erra.
- Tentar identificar alguma tendência nesses erros para, por exemplo, adicionar um atributo que trate aqueles casos.

Exemplo

- Anti-Spam: $m_{cv} = 500$ exemplos de validação.
- Algoritmo classifica erradamente 100 emails, que examinamos manualmente e categorizamos com base em
  1. Tipo de email, ex.: medicamento, réplica, roubo de senha, outros.
  2. Atributos que ajudariam o algoritmo.

Exemplo

EXEMPLO:

Tipo	Atributo
Medicamento: 12	Erros deliberados de ortografia: 5
Réplica/*fake*: 4	Rota incomum: 16
Roubo de senha: 53	Pontuação incomum (ex.: muitas exclamações): 32
Outros: 31	⋮

- Concentra-se em novos atributos, algoritmos, etc. para tratar especificamente dos casos mais frequentes: roubo de senha / pontuação incomum.

Métricas de Erro

- Métricas de erro são fundamentais na tomada de decisões em *machine learning*.
- EX.: Devemos considerar os radicais das palavras? Distinguir maiúsculas e minúsculas?
- Com radicais: 5% de erro; sem radicais: 3% de erro. Melhor usar!
- Distinguindo maiúsculas: 3.2% de erro; sem distinção: 3% de erro. Melhor sem!

Métricas de Erro

- Métricas de erro são fundamentais na tomada de decisões em *machine learning*.
- EX.: Devemos considerar os radicais das palavras? Distinguir maiúsculas e minúsculas?
- Com radicais: 5% de erro; sem radicais: 3% de erro. Melhor usar!
- Distinguindo maiúsculas: 3.2% de erro; sem distinção: 3% de erro. Melhor sem!

Métricas de Erro

- Métricas de erro são fundamentais na tomada de decisões em *machine learning*.
- EX.: Devemos considerar os radicais das palavras? Distinguir maiúsculas e minúsculas?
- Com radicais: 5% de erro; sem radicais: 3% de erro. Melhor usar!
- Distinguindo maiúsculas: 3.2% de erro; sem distinção: 3% de erro. Melhor sem!

Métricas de Erro

- Métricas de erro são fundamentais na tomada de decisões em *machine learning*.
- EX.: Devemos considerar os radicais das palavras? Distinguir maiúsculas e minúsculas?
- Com radicais: 5% de erro; sem radicais: 3% de erro. Melhor usar!
- Distinguindo maiúsculas: 3.2% de erro; sem distinção: 3% de erro. Melhor sem!

Outline

- Imagine uma regressão logística $h_\theta(x)$ para diagnosticar câncer ($y = 1$) ou sem câncer ($y = 0$).
- E que obtivemos 1% de erro no teste (99% de acerto). Excelente resultado???
- Porém, só 0.5% dos pacientes têm câncer! Temos **classes desbalanceadas** (*skewed*).
- Nosso resultado não é mais interessante! A função trivial abaixo teria 0.5% de erro!

```
def predictCancer(x):
return 0
```

- Imagine uma regressão logística $h_\theta(x)$ para diagnosticar câncer ($y = 1$) ou sem câncer ($y = 0$).
- E que obtivemos 1% de erro no teste (99% de acerto). Excelente resultado???
- Porém, só 0.5% dos pacientes têm câncer! Temos **classes desbalanceadas** (*skewed*).
- Nosso resultado não é mais interessante! A função trivial abaixo teria 0.5% de erro!

```
def predictCancer(x):
return 0
```

- Imagine uma regressão logística $h_\theta(x)$ para diagnosticar câncer ($y = 1$) ou sem câncer ($y = 0$).
- E que obtivemos 1% de erro no teste (99% de acerto). Excelente resultado???
- Porém, só 0.5% dos pacientes têm câncer! Temos **classes desbalanceadas** (*skewed*).
- Nosso resultado não é mais interessante! A função trivial abaixo teria 0.5% de erro!

```
def predictCancer(x):
return 0
```

- Imagine uma regressão logística $h_\theta(x)$ para diagnosticar câncer ($y = 1$) ou sem câncer ($y = 0$).

- E que obtivemos 1% de erro no teste (99% de acerto). Excelente resultado???

- Porém, só 0.5% dos pacientes têm câncer! Temos **classes desbalanceadas** (*skewed*).

- Nosso resultado não é mais interessante! A função trivial abaixo teria 0.5% de erro!

```
def predictCancer(x):
return 0
```

- Convenção: $y = 1$ na presença da classe rara, no caso, a presença de câncer.

Cenários possíveis na classificação binária - **Matriz de Confusão**:

		Classe Verdadeira	
		1	0
Classe Predita	1	Verdadeiro Positivo	Falso Positivo
	0	Falso Negativo	Verdadeiro Negativo

Precision/Recall

Precision

De todos os pacientes para os quais o algoritmo retornou $y = 1$, qual proporção **realmente** tem câncer?

$$\frac{\text{Número de verdadeiros positivos}}{\text{Número de previsões positivas}} = \frac{\text{Número de verdadeiros positivos}}{\text{Número de verdadeiros positivos} + \text{número de falsos positivos}}$$

Recall

De todos os pacientes que realmente têm câncer, qual proporção o algoritmo **detectou** como tendo câncer?

$$\frac{\text{Número de verdadeiros positivos}}{\text{Número real de positivos}} = \frac{\text{Número de verdadeiros positivos}}{\text{Número de verdadeiros positivos} + \text{número de falsos negativos}}$$

Trade off Precision/Recall

- Seja a regressão logística em que prevemos $y = 1$ se $h_\theta(x) \geq 0.5$ e $y = 0$ se $h_\theta(x) < 0.5$ (*threshold* 0.5).
- Se quisermos prever $y = 1$ (câncer) apenas se tivermos muita certeza, subimos o *threshold* para 0.7 ou 0.9.
- Assim temos **precision mais alto** e **recall mais baixo**.
- Já se quisermos evitar de passar batido um caso de câncer, baixamos o *threshold*, p.ex., para 0.3.
- E temos **recall mais alto** e **precision mais baixo**.

Trade off Precision/Recall

- Seja a regressão logística em que prevemos $y = 1$ se $h_\theta(x) \geq 0.5$ e $y = 0$ se $h_\theta(x) < 0.5$ (*threshold* 0.5).
- Se quisermos prever $y = 1$ (câncer) apenas se tivermos muita certeza, subimos o *threshold* para 0.7 ou 0.9.
- Assim temos **precision mais alto** e **recall mais baixo**.
- Já se quisermos evitar de passar batido um caso de câncer, baixamos o *threshold*, p.ex., para 0.3.
- E temos **recall mais alto** e **precision mais baixo**.

Trade off Precision/Recall

- Seja a regressão logística em que prevemos $y = 1$ se $h_\theta(x) \geq 0.5$ e $y = 0$ se $h_\theta(x) < 0.5$ (*threshold* 0.5).
- Se quisermos prever $y = 1$ (câncer) apenas se tivermos muita certeza, subimos o *threshold* para 0.7 ou 0.9.
- Assim temos **precision mais alto** e **recall mais baixo**.
- Já se quisermos evitar de passar batido um caso de câncer, baixamos o *threshold*, p.ex., para 0.3.
- E temos **recall mais alto** e **precision mais baixo**.

Trade off Precision/Recall

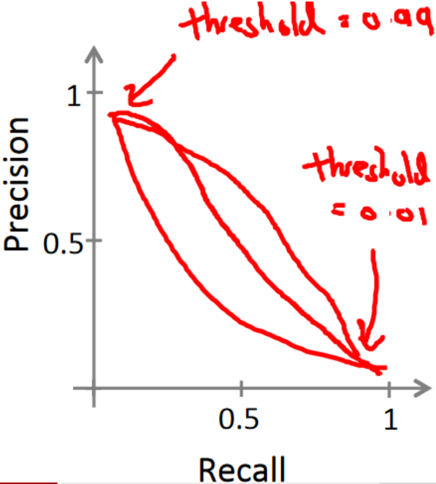
- Seja a regressão logística em que prevemos $y = 1$ se $h_\theta(x) \geq 0.5$ e $y = 0$ se $h_\theta(x) < 0.5$ (*threshold* 0.5).
- Se quisermos prever $y = 1$ (câncer) apenas se tivermos muita certeza, subimos o *threshold* para 0.7 ou 0.9.
- Assim temos **precision mais alto** e **recall mais baixo**.
- Já se quisermos evitar de passar batido um caso de câncer, baixamos o *threshold*, p.ex., para 0.3.
- E temos **recall mais alto** e **precision mais baixo**.

Trade off Precision/Recall

- Seja a regressão logística em que prevemos $y = 1$ se $h_\theta(x) \geq 0.5$ e $y = 0$ se $h_\theta(x) < 0.5$ (*threshold* 0.5).
- Se quisermos prever $y = 1$ (câncer) apenas se tivermos muita certeza, subimos o *threshold* para 0.7 ou 0.9.
- Assim temos **precision mais alto** e **recall mais baixo**.
- Já se quisermos evitar de passar batido um caso de câncer, baixamos o *threshold*, p.ex., para 0.3.
- E temos **recall mais alto** e **precision mais baixo**.

Curva de *Precision*/*Recall*

- Podemos fazer uma curva variando o *threshold* e prevendo $y = 1$ se $h_\theta(x) >$ threshold.



F_1 *score* (F *score*)

- Como expressar *precision* e *recall* em uma única medida?

	Precision (P)	*Recall* (R)	Média	F_1 score
Algoritmo 1	0.5	0.4	0.45	0.44
Algoritmo 2	0.7	0.1	0.4	0.18
Algoritmo 3	0.02	1.0	0.51	0.04

- Que tal a média?

$$\frac{P+R}{2}$$

- Algoritmo 3 prevê sempre $y = 1$ e tem média melhor que o Algoritmo 1. Portanto a média **não** serve!

F_1 *score* (F *score*)

- Como expressar *precision* e *recall* em uma única medida?

	Precision (P)	*Recall* (R)	Média	F_1 score
Algoritmo 1	0.5	0.4	0.45	0.44
Algoritmo 2	0.7	0.1	0.4	0.18
Algoritmo 3	0.02	1.0	0.51	0.04

- Que tal a média?

$$\frac{P+R}{2}$$

- Algoritmo 3 prevê sempre $y = 1$ e tem média melhor que o Algoritmo 1. Portanto a média **não** serve!

F_1 *score* (F *score*)

- Solução é o F_1 score:

$$2\frac{PR}{P+R}$$

- Se $P \approx 0$ OU $R \approx 0$, o score é pequeno e este só fica maior se AMBOS forem relativamente altos.

- Serve para definir um *threshold* ótimo (maior F_1 score no conjunto de validação).

F_1 *score* (F *score*)

- Solução é o F_1 score:

$$2\frac{PR}{P+R}$$

- Se $P \approx 0$ OU $R \approx 0$, o score é pequeno e este só fica maior se AMBOS forem relativamente altos.

- Serve para definir um *threshold* ótimo (maior F_1 score no conjunto de validação).

F_1 *score* (F *score*)

- Solução é o F_1 score:

$$2\frac{PR}{P+R}$$

- Se $P \approx 0$ OU $R \approx 0$, o score é pequeno e este só fica maior se AMBOS forem relativamente altos.

- Serve para definir um *threshold* ótimo (maior F_1 score no conjunto de validação).

Notas

- Uma alternativa à curva de *precision*/*recall* é a curva ROC (*Receiver Operating Characteristic*) - TPR em função do FPR:

$$TPR = \frac{\text{Número de verdadeiros positivos}}{\text{Número de verdadeiros positivos} + \text{número de falsos negativos}}$$

$$FPR = \frac{\text{Número de falsos positivos}}{\text{Número de falsos positivos} + \text{número de verdadeiros negativos}}$$

- Área sob essas curvas pode ser usada como métrica de desempenho do modelo.

- Essas curvas podem ser adaptadas para múltiplas classes: curva média do *one-vs-all*.

Notas

- Uma alternativa à curva de *precision*/*recall* é a curva ROC (*Receiver Operating Characteristic*) - TPR em função do FPR:

$$TPR = \frac{\text{Número de verdadeiros positivos}}{\text{Número de verdadeiros positivos} + \text{número de falsos negativos}}$$

$$FPR = \frac{\text{Número de falsos positivos}}{\text{Número de falsos positivos} + \text{número de verdadeiros negativos}}$$

- Área sob essas curvas pode ser usada como métrica de desempenho do modelo.

- Essas curvas podem ser adaptadas para múltiplas classes: curva média do *one-vs-all*.

Notas

- Uma alternativa à curva de *precision*/*recall* é a curva ROC (*Receiver Operating Characteristic*) - TPR em função do FPR:

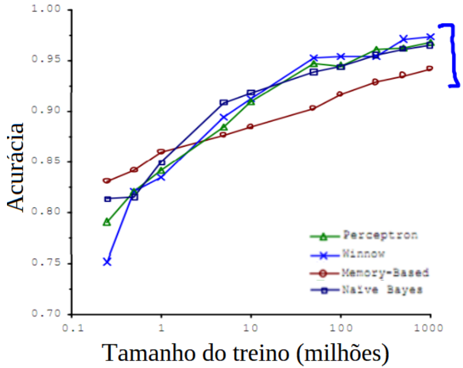
$$TPR = \frac{\text{Número de verdadeiros positivos}}{\text{Número de verdadeiros positivos} + \text{número de falsos negativos}}$$

$$FPR = \frac{\text{Número de falsos positivos}}{\text{Número de falsos positivos} + \text{número de verdadeiros negativos}}$$

- Área sob essas curvas pode ser usada como métrica de desempenho do modelo.
- Essas curvas podem ser adaptadas para múltiplas classes: curva média do *one-vs-all*.

Outline

Grandes Conjuntos de Dados



Grandes Conjuntos de Dados

- "O vencedor não é o melhor algoritmo, mas sim quem tem mais dados!".

- Condições:

    1. Os atributos devem ser suficientes para prever y corretamente. Teste útil: um especialista humano poderia prever y com base apenas em x?

    2. O algoritmo de aprendizado deve ter muitos parâmetros. Assim teremos $J_{treino}(\theta)$ pequeno (viés baixo) e o uso de um conjunto de treinamento muito grande provavelmente não causará *overfit* (variância baixa), de modo que $J_{treino}(\theta) \approx J_{teste}(\theta)$ e, portanto, $J_{teste}(\theta)$ será pequeno, como desejado!

Grandes Conjuntos de Dados

- "O vencedor não é o melhor algoritmo, mas sim quem tem mais dados!".
- Condições:

1. Os atributos devem ser suficientes para prever y corretamente. Teste útil: um especialista humano poderia prever y com base apenas em x?
2. O algoritmo de aprendizado deve ter muitos parâmetros. Assim teremos $J_{treino}(\theta)$ pequeno (viés baixo) e o uso de um conjunto de treinamento muito grande provavelmente não causará *overfit* (variância baixa), de modo que $J_{treino}(\theta) \approx J_{teste}(\theta)$ e, portanto, $J_{teste}(\theta)$ será pequeno, como desejado!

Grandes Conjuntos de Dados

- "O vencedor não é o melhor algoritmo, mas sim quem tem mais dados!".
- Condições:

1. Os atributos devem ser suficientes para prever y corretamente. Teste útil: um especialista humano poderia prever y com base apenas em x?
2. O algoritmo de aprendizado deve ter muitos parâmetros. Assim teremos $J_{treino}(\theta)$ pequeno (viés baixo) e o uso de um conjunto de treinamento muito grande provavelmente não causará *overfit* (variância baixa), de modo que $J_{treino}(\theta) \approx J_{teste}(\theta)$ e, portanto, $J_{teste}(\theta)$ será pequeno, como desejado!

Grandes Conjuntos de Dados

- "O vencedor não é o melhor algoritmo, mas sim quem tem mais dados!".
- Condições:
  1. Os atributos devem ser suficientes para prever y corretamente. Teste útil: um especialista humano poderia prever y com base apenas em x?
  2. O algoritmo de aprendizado deve ter muitos parâmetros. Assim teremos $J_{treino}(\theta)$ pequeno (viés baixo) e o uso de um conjunto de treinamento muito grande provavelmente não causará *overfit* (variância baixa), de modo que $J_{treino}(\theta) \approx J_{teste}(\theta)$ e, portanto, $J_{teste}(\theta)$ será pequeno, como desejado!