# Aula06

DATA SCIENCE IPT

TURMA 02

LINKED EDUCATION

V2019

# Treinamento e Testes

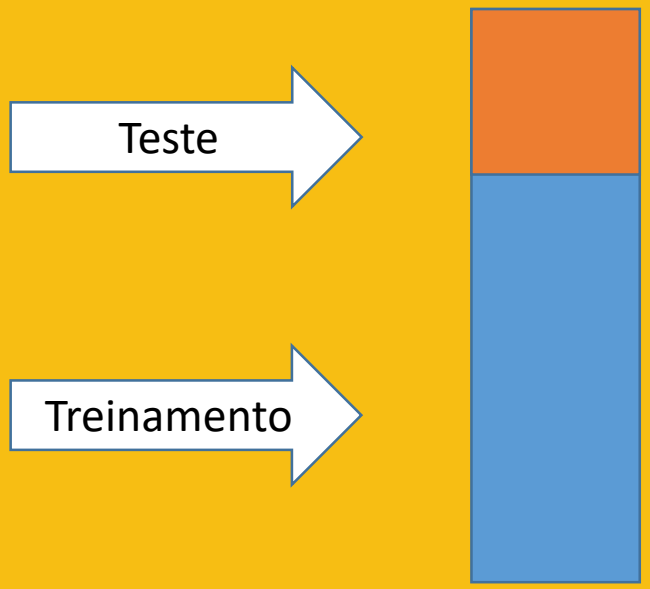
Conteúdo

Em Machine Learning, queremos que nosso modelo possa ser genérico nas predições e isso, obviamente, significa que ele deve funcionar bem não só nas amostras usadas no treinamento. Assim, é interessante ter um conjunto de amostras para testes que nos permita avaliar o “poder” de generalização do modelo.

# Treinamento e Testes



Uma técnica usual (split) é dividir as amostras entre treinamento e testes....e observar os resultados em ambas as amostras.




Tipicamente, em torno de 70% da amostra para treinamento

# Treinamento e Testes

## Conteúdo

Partir do notebook split.ipynb
Calcular erro na amostra de treinamento e de testes para as hipóteses

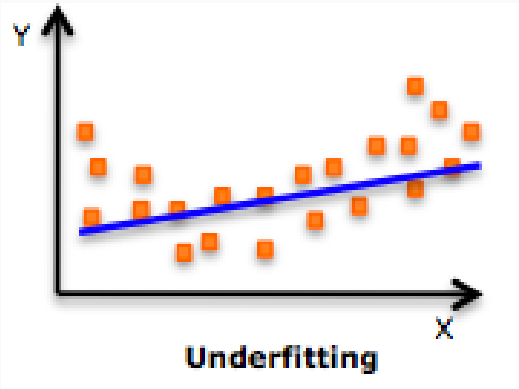
Regressão com horsepower somente
Regressão com horsepower + potência de quadrado à décima de horsepower.

Traçar gráficos : potência x consumo com os dois modelos e o dataset original

# Underfitting overfitting bias variance

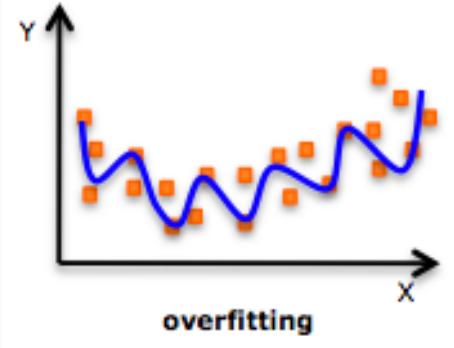
Conteúdo

Imagine, em uma regressão linear, se tentarmos “aprender” uma reta que represente as amostras do gráfico abaixo.




É bastante intuitivo que no nosso modelo, nossa hipótese $y=h(x)=\theta_0+\theta_1 x$ tem pouca flexibilidade
para representar as amostras de forma generalizada. Nesse caso, o custo será alto tanto no dataset de treinamento como no de validação/teste. É um caso de **underfitting,** é um modelo de complexidade muito baixa para representar as amostras de forma generalizada. Tem poucos “graus de liberdade”.

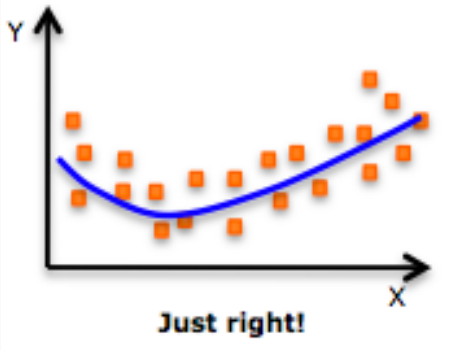
Por outro lado, imagine agora que usamos uma função polinomial de alto grau para representar as amostras....




Nesse caso, o polinômio de alto grau consegue se adaptar com baixo custo ao data set de treinamento (o erro do treinamento é baixo). Porém, no data set de validação/testes, tipicamente o erro aumenta e há variação grande do erro para diferentes datasets de treinamento. É um caso de **overfitting**. O modelo pega detalhes do dataset de treinamento que não são úteis para generalizar para qualquer dataset de validação/teste.

Analogia : é como aquela prova que o aluno decora todos os exercícios da lista, mas se enrola com uma nova questão. Sabe tudo da lista, mas não “pegou” os conceitos essenciais.

Obviamente, na escolha da complexidade do modelo há um ponto ótimo, que representa as amostras com baixo erro nos datasets de testes e com pouca variação do erro nos testes com a variação das amostras de treinamento...obviamente, nesse caso, o modelo também se comporta bem nas amostras de treinamento.

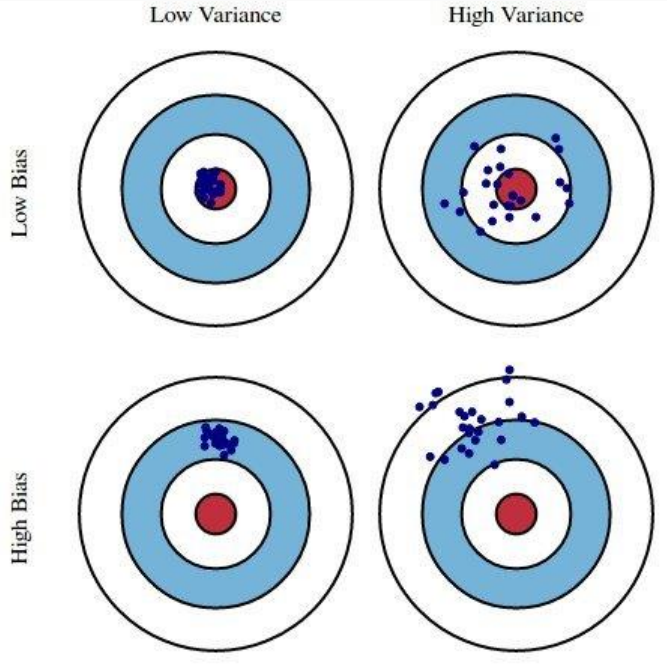


# Underfitting overfitting bias variance

Conteúdo

**Bias (viés) :** quando a diferença média entre hipótese (y estimado) e o real (y) é alta, para diferentes datasets de treinamento, dizemos que o bias é alto.

**Variance (variância) :** quando ocorre variação relevante nos erros no dataset de testes quando o dataset de treinamento é alterado, dizemos que a variância é alta.




Definindo Erro = $E((Y-Ye)^2)$ ..podemos dividi-lo em

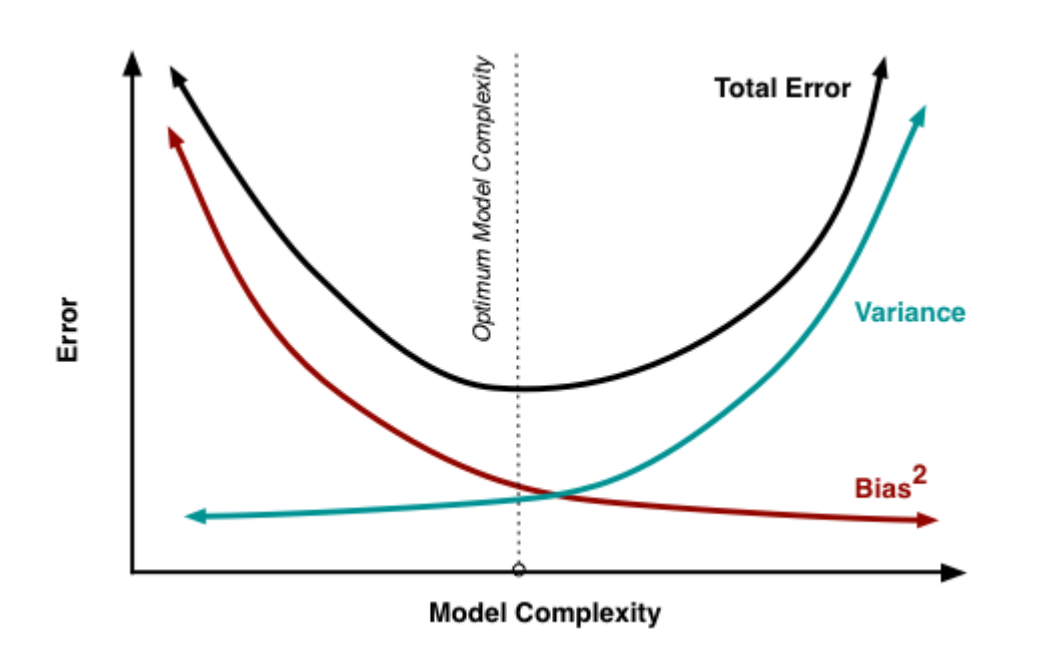
$Erro=(E[Ye]-Y)^2 + E[(Ye-E[Ye])^2] + \sigma e^2$

**$Erro=bias^2 + variance + erro\_irredutível^2$**

Obs : E é a esperança

# Trade-off Bias Variance

Normalmente, quando aumentamos progressivamente a complexidade do modelo a partir de uma situação de underfitting (tipicamento alto bias), o erro médio quadrático vai diminuindo (pela diminuição do erros de bias) até o ponto ótimo, depois, pelo aumento da variância, passa a aumentar...caminhando para o overfitting.



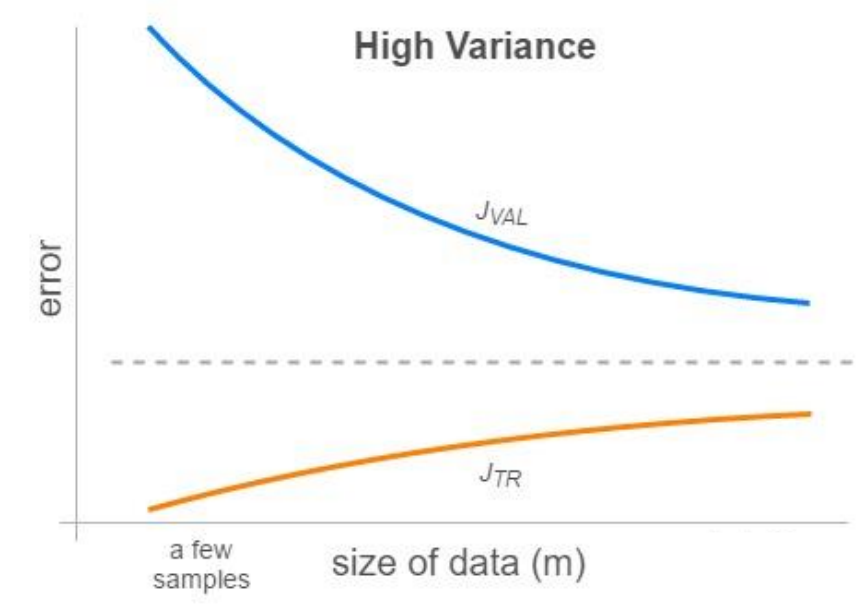
# Underfitting – overfitting-bias-variance

## Conteúdo

Erros nos data sets de treinamento e testes em alto bias e alta variância

**x**

Tamanho do dataset de treinamento

# Underfitting – overfitting-bias-variance

Conteúdo

Erros de Bias e Variance....**Como tratá-los?**

**Erro de bias (bias$^2$)**

Inserir outras features

Dar mais "graus de liberdade" ao modelo.

Exemplo : aumentar o grau do polinômio

Aumentar número de amostras de treinamento (até não melhorar mais)

**Erro de variância**

Retirar features

Aumentar número de amostras de treinamento (até não melhorar mais)

**Regularização (coming soon!)**

**Bagging (coming soon)**

**Há ainda o Erro irredutível$^2$ ...o ruído**

Faça uma cópia de do notebook split.ipynb, gerando complex.ipynb e procure a complexidade (no caso, até que grau da potência de horsepower) mais adequada(?) nesse modelo de pot x consumo apenas. Explique

# Regularização



Uma das possibilidades de minimizar o overfitting é usar regularização. Na regularização, “forçamos” os “thetas” a se manterem baixos por meio de uma “penalização” na função custo, onde os thetas aparecem ao quadrado e assim..se crescerem, aumentam muito o custo...

# Regularização



No caso da Regressão Linear, o custo “regularizado” é dado por :

$$J(\Theta)=(1/2m)^{*}\sum_{i=1}^{m}(\Theta_0 xo^i + \Theta_1 x1^i + \cdots \Theta_n xn^i - yi)^2 + \lambda \sum_{i=1}^{n} \Theta_i^2$$

Observe que o termo $\Theta_0$ não é regularizado! São regularizados apenas os termos que multiplicam as features. Portanto, é simples incluir as “penalizações” na função custo...e no gradiente?

# Regularização



A parcela do gradiente do termo de penalizado é trivial:

$$\nabla penalização = (1/m) * \lambda\Theta$$

Assim $\nabla J = \left(\frac{1}{m}\right) * (Xt(X\theta - y) + *\lambda\Theta)$

$*\lambda\Theta$ não é feito para $\Theta_0$

# Regularização

Conteúdo

Vamos trabalhar com o notebook regularization.ipynb

# Seleção de features 1/n



## A matriz de correlação pode ajudar na seleção das features para a regressão....

$$Correlation = \frac{Cov\,(x,y)}{\sigma x * \sigma y}$$

Population Covariance Formula

$$Cov(x,y) = \frac{\Sigma(x_i-\bar{x})(y_i-\bar{y})}{N}$$

Sample Covariance

$$Cov(x,y) = \frac{\Sigma(x_i-\bar{x})(y_i-y)}{N-1}$$

Features correlacionadas (módulo alto) com o target podem ser candidatas a features do modelo.

Features muito correlacionadas entre si, podem ser "repetições", basta uma...

# Seleção de features

Conteúdo

Vamos trabalhar com o notebook
Correlations_res.ipynb

# Ainda a correlação (2)



A correlação Pearson mede correlações “lineares”....

Analisar o notebook correlations2_res.ipynb: há uma feature e mais três, que são, respectivamente o triplo, quadrado e o cubo da primeira.

Observe que o maior índice de correlação é entre a feature e o seu triplo. O índice “default” para .corr() é “Pearson”..eis a explicação.

Vamos trocar por índice de correlação “Spearman” que mede correlações também não lineares...tudo 1.0!

# Ainda a correlação (3)



No terceiro exemplo, correlations3_res.ipynb, há uma relação perfeita quadrática, mas hora é positiva e hora é negativa...daí a somatória é zero!!!...Vamos descartar essa feature??

# Ainda a correlação (4)



No terceiro exemplo, correlations4.ipynb, há uma relação perfeita quadrática entre feature e target. Tentamos uma regressão linear...não funciona...já a rede neural, ok...Assim, o problema era o modelo (regressão linear) e não a feature.

# Se sobrar tempo

**Explique a diferença de Pearson e Spearman**


Cursos com Alta Performance de Aprendizado