Aula05

DATA SCIENCE IPT

TURMA 02

LINKED EDUCATION

V2019

Conteúdo

- IA,
- Machine Learning e
- Deep Learning

ARTIFICIAL INTELLIGENCE (IA)

"Campo de estudo que procura entender e emular comportamento inteligente em termos de processos computacionais" (Schalkoff, 1990)

COMPORTAMENTOS INTELIGENTES:

- ✓ RACIOCINAR
- ✓ RECONHECER PADRÕES
- ✓ APRENDER
- ✓ FALAR

POR QUE IA AGORA?

ABUNDÂNCIA DE DADOS

+

CAPACIDADE DE PROCESSAMENTO

+

MELHORES ALGORITMOS

TIMELINE

DADOS

- 1991 WWW
- 2007 Smartphones

PROCESSAMENTO

- 2002 Cloud
- 2006 Hadoop
- 2009 GPUs

ALGORITMOS

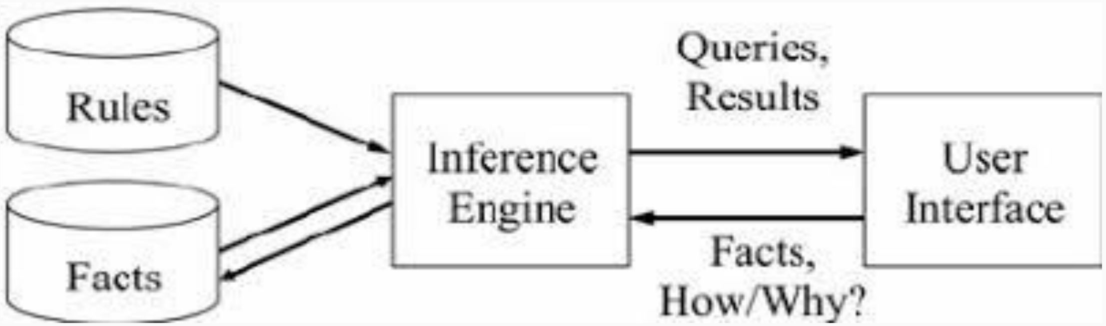
- 1986 Backpropagation
- 1989 CNN
- 2006 Deep Learning

IA X Machine Learning

Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.

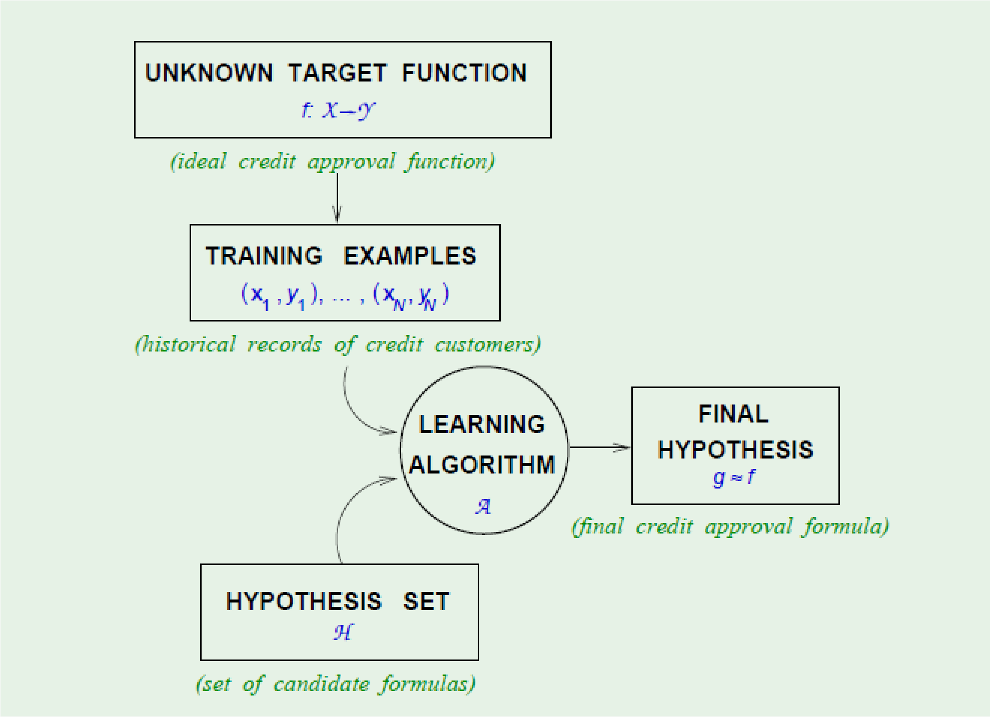
Arthur Samuel 1959

Sistema baseado em regras.



É IA, mas **não** é Machine Learning, pois a inteligência foi explicitamente definida por regras!

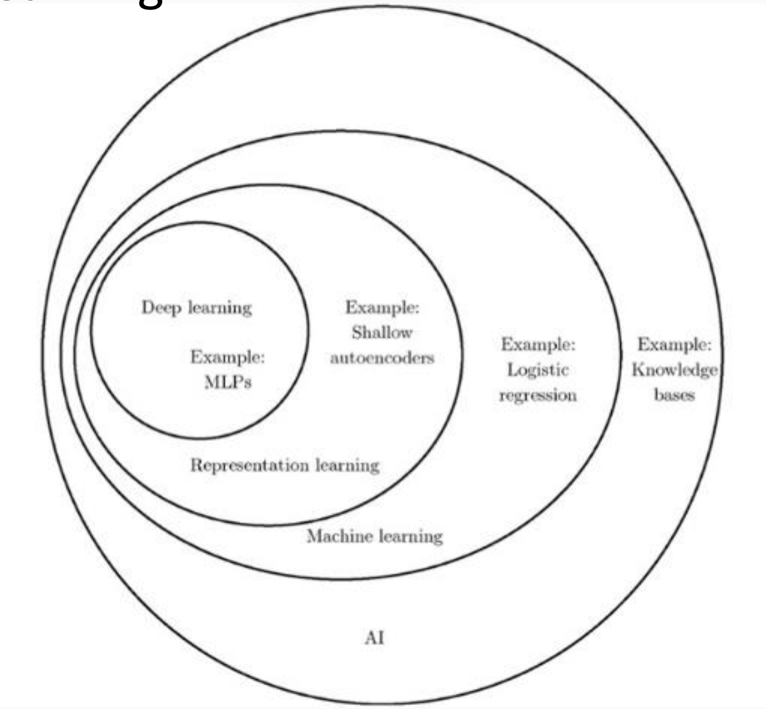
Machine Learning Summary



FONTE : ABU-MOSTAFA et al. Learning from Data

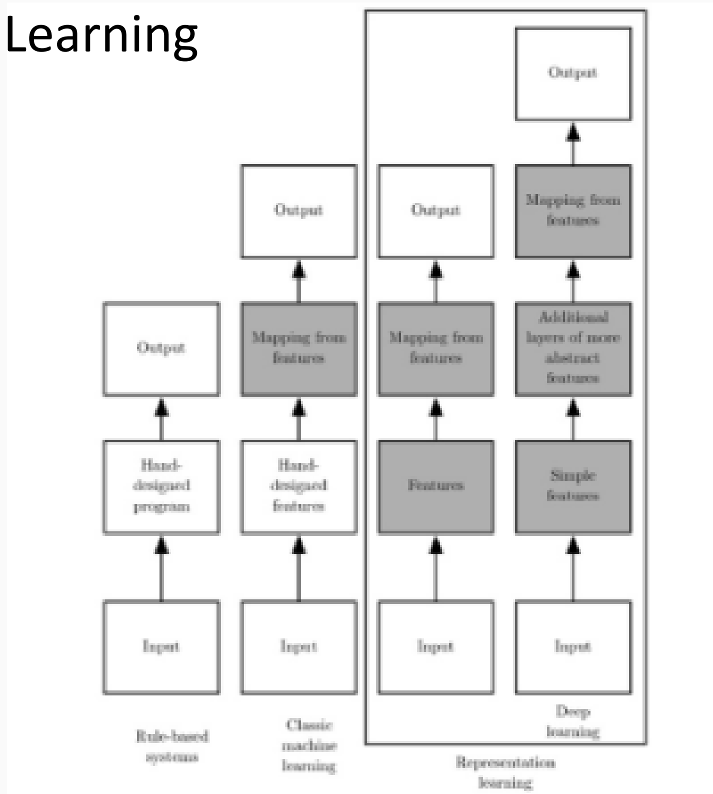
IA, ML e DL> Introdução

IA, Machine Learning, Deep Learning



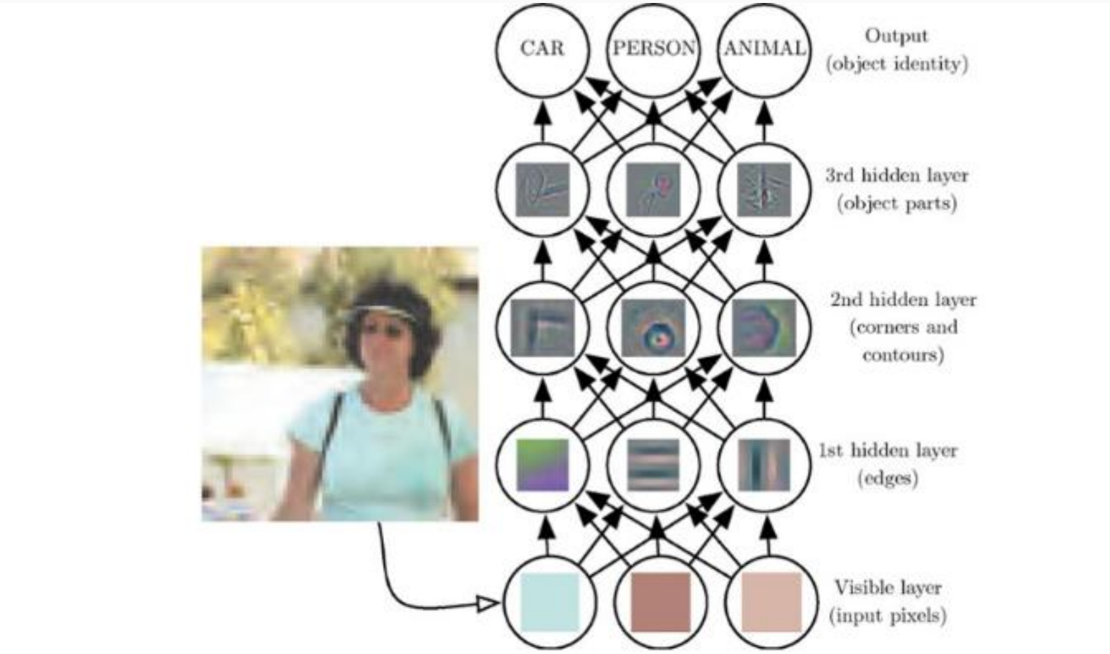
Fonte : BENGIO, Yoshua. Deep Learning

IA, Machine Learning, Deep Learning



Fonte : BENGIO, Yoshua. Deep Learning

Classificação com MLP



Fonte : BENGIO, Yoshua. Deep Learning

IA, ML e DL> Introdução

Conteúdo

Impacto das diferentes representações (features)

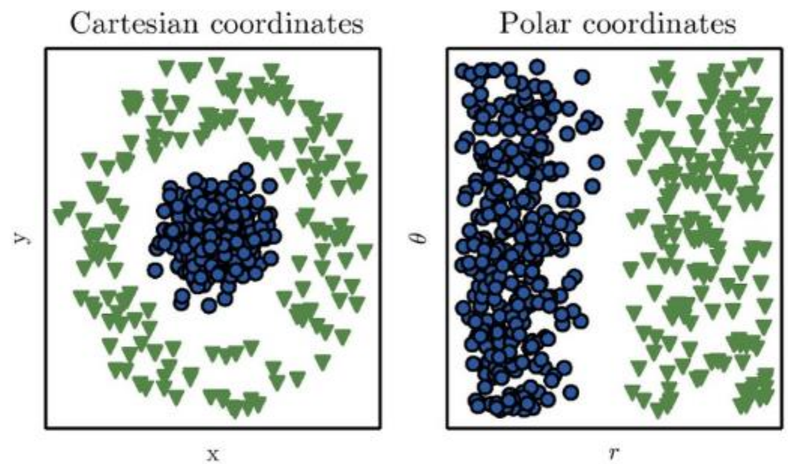


Figure 1.1: Example of different representations: suppose we want to separate two categories of data by drawing a line between them in a scatterplot. In the plot on the left, we represent some data using Cartesian coordinates, and the task is impossible. In the plot on the right, we represent the data with polar coordinates and the task becomes simple to solve with a vertical line. (Figure produced in collaboration with David Warde-Farley.)

Fonte : BENGIO, Yoshua. Deep Learning

- Exemplos de utilização de ML/DL

Utilizações de Machine Learning

- Carros Autônomos
- Diagnósticos Médicos
- Investimentos em bolsa de valores
- Detecção de fraudes
- Processamento de linguagem natural (NLP)
- Recomendações
- Reconhecimento Facial/de padrões
- Avaliação de crédito
- Automação de atividades humanas em processos

Brincando com carro autônomo

https://www.youtube.com/watch?v=ppFyPUx9RIU

BIBLIOGRAFIA

1. BENGIO, Yoshua. Deep Learning (Adaptive Computation and Machine Learning). MIT Press, 2017.

2. Aladino, Ethem.Introduction To Machine Learning. MIT Press, 2014.

3. Abu-Mostafa et al. Learning from Data, AMLBook.com, 2012.

Tipos de Aprendizado

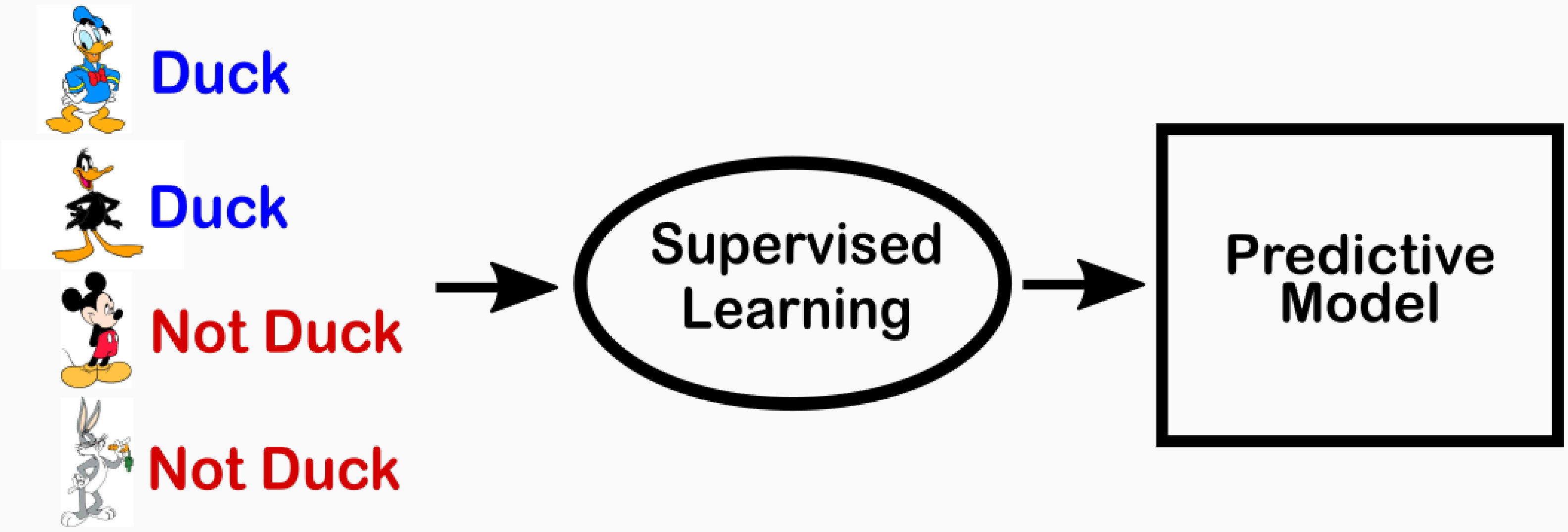
- APRENDIZADO SUPERVISIONADO
- APRENDIZADO NÃO SUPERVISIONADO
- APRENDIZADO POR REFORÇO

Tipos de Aprendizado

APRENDIZADO SUPERVISIONADO :

Sabemos a resposta correta (output) para cada entrada no data set (input). Acreditamos haver relação entre inputs e outputs.

Exemplo : Temos as notas de ensino médio dos alunos e sabemos se cada um deles foi ou não aprovado nos vestibulares. Podemos treinar a máquina mostrando instâncias de aprovados ou não aprovados para que, depois, possamos prever a aprovação ou não de um aluno nos vestibulares em função de suas notas.



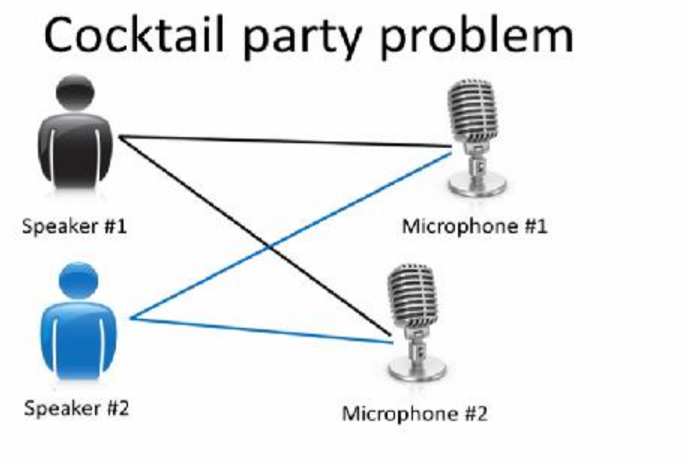
Tipos de Aprendizado

APRENDIZADO NÃO SUPERVISIONADO :

Para o data set procuramos derivar alguma estrutura entre os dados. Porém, o resultado não pode ser avaliado como "correto" ou "errado". Os dados não são **rotulados**.

Exemplo : Temos as notas de ensino médio dos alunos e queremos agrupá-los por semelhança (?).

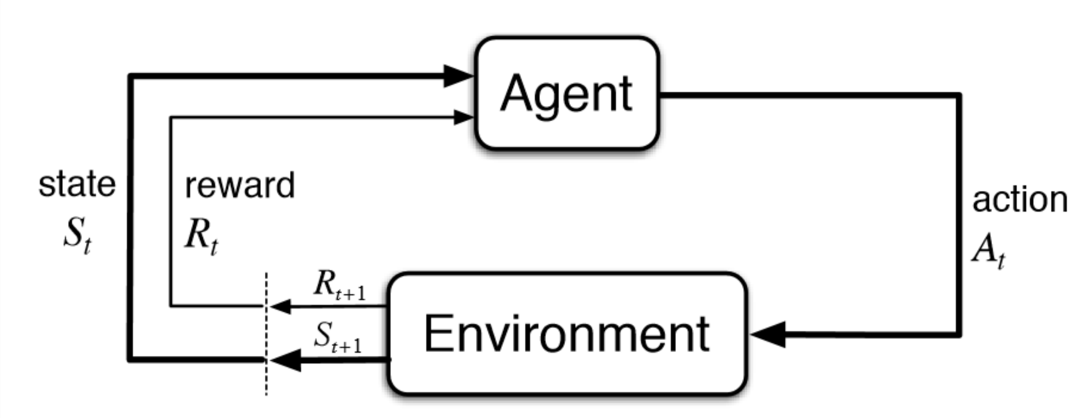
Exemplo clássico de aprendizado não supervisionado : separar as vozes que falam simultaneamente



Tipos de Aprendizado

APRENDIZADO POR REFORÇO :

Um agente interage com um ambiente. Ele está em um estado s e toma uma ação A, que o leva a um novo estado e a uma recompensa. O objetivo é maximizar a soma das recompensas. O aprendizado leva ao mapeamento otimizado entre estados e ações.



Regressão x Classificação

REGRESSÃO

Queremos um output contínuo com base no input.

Exemplo : queremos prever o valor de uma ação.

CLASSIFICAÇÃO

Queremos apenas mapear um input para determinada classe.

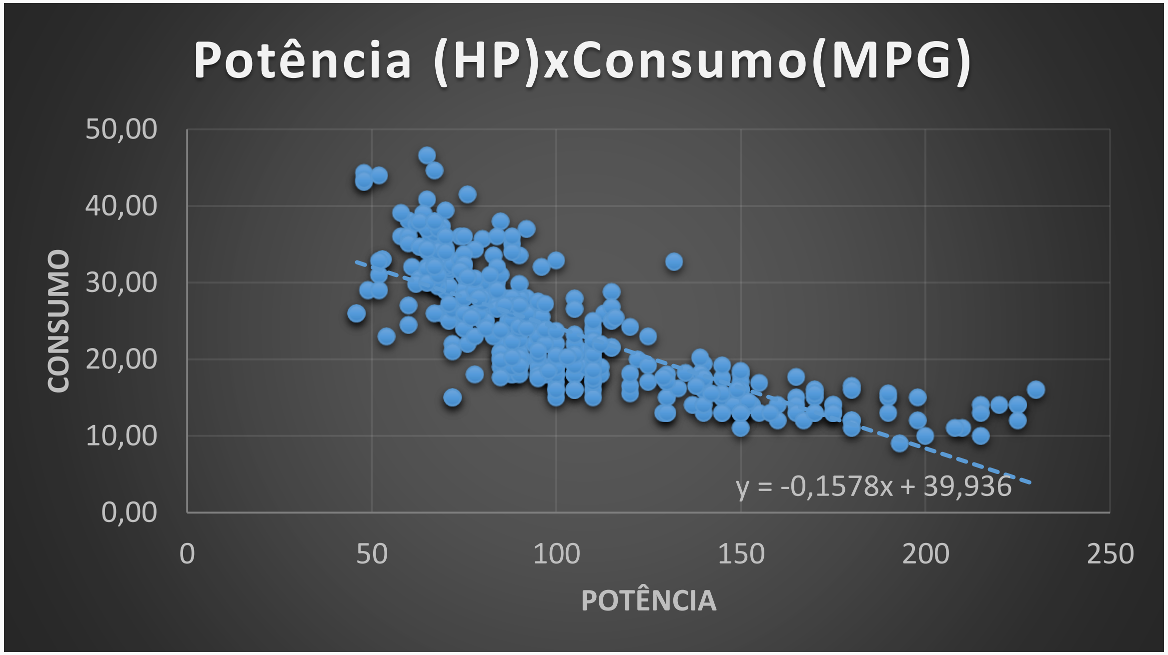
Exemplo : queremos apenas saber se a ação vai subir ou descer.

- Nosso primeiro problema será prever o consumo de um carro (MPG) em função de diversas “features” (peso, potência, aceleração etc.). Usaremos os dados do Machine Learning Repository da UCI (https://archive.ics.uci.edu/ml/index.php).

- Veja arquivo : cars-uci-Linked.csv

Regressão

No nosso primeiro modelo estimaremos o consumo em função apenas da potência. Espera-se um consumo pior para os carros de maior potência. Fazendo a importação de dados para o Excel, retirando as amostras com NA, traçando-se um gráfico de dispersão com tendência (linear) obtemos :



Regressão

Nossa função hipótese (h(x), a função que representa o modelo), temos :

$y_e = h(x) = \theta_0 + \theta_1 x$ (1)

Sendo :

y_e o output gerado pelo modelo (consumo em MPG)

θ_0 e θ_1 os parâmetros da reta estimada (desconhecidos)

x o input para a estimação (potência)

Regressão

Conteúdo

Nosso aprendizado é supervisionado pois sabemos o valor de potência real para cada amostra. Como nosso objetivo é ter um bom modelo com base no aprendizado da amostra, devemos estimar um erro desse modelo.

Primeira alternativa (ruim) de estimativa de erro da amostra :
Se fizermos a média (entre os m elementos do data set) entre o previsto e o real para todos os elementos da amostra:

Erro 1 $= \frac{1}{m} \sum_{i=1}^{m} (y_e - y)$ (2)

Sendo y , o valor real da potência para aquela amostra. Essa **não** é uma boa alternativa, pois um erro acima do esperado pode anular um outro igualmente abaixo do esperado (erros simétricos em relação ao esperado).

Erro 2 $= \frac{1}{m} \sum_{i=1}^{m} (y_e - y)^2$ (3)

A média da somatória dos quadrados dos desvios parece ser uma **boa alternativa**, pois desvios acima contam como desvios abaixo do esperado e não se anulam. Esse é o chamado **erro médio quadrático.**

Regressão

O nosso “aprendizado” será encontrar os thetas que minimizam o erro médio quadrático para as amostras disponíveis...daí teremos um modelo treinado!

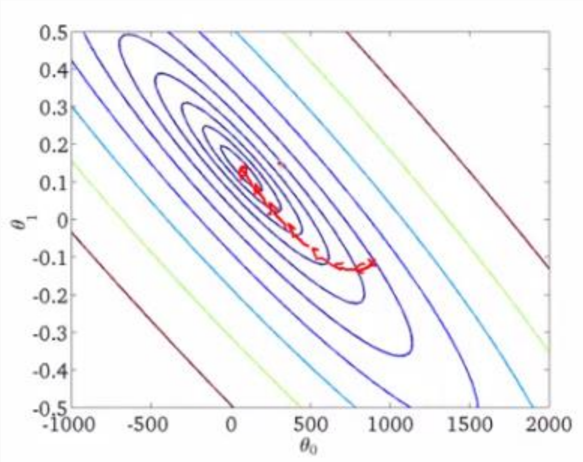
$$\text{Consumo (mpg)} = h(x) = \theta_0 + \theta_1 \text{Potência(HP)}$$

- Minimizando o erro médio quadrático com o algoritmo Gradient Descent

GRADIENT DESCENT

No Módulo 1, partimos de um modelo linear com uma feature (potência) para prever o consumo do carro (MPG). Obtivemos (por força bruta) o par θ_0 e θ_1 que minimizava a função custo adotada (erro médio quadrático).

Agora, vamos usar métodos melhores para otimizar. Começaremos com o Gradiente Descent.

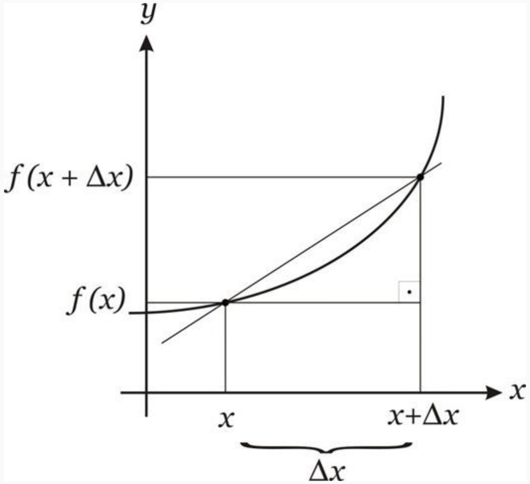


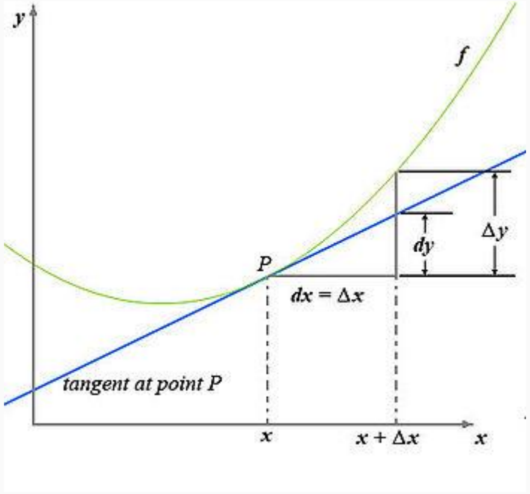
Na figura ao lado, o custo (como no nosso exemplo de regressão) depende de duas variáveis. Com o algoritmo Gradient Descent, vamos, passo a passo, "caminhando" na direção que nos leva ao mínimo da função custo que queremos minimizar.

Para isso, usamos o conceito de "gradiente"

Passo 1 : A derivada

$$\frac{dy}{dx} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$$



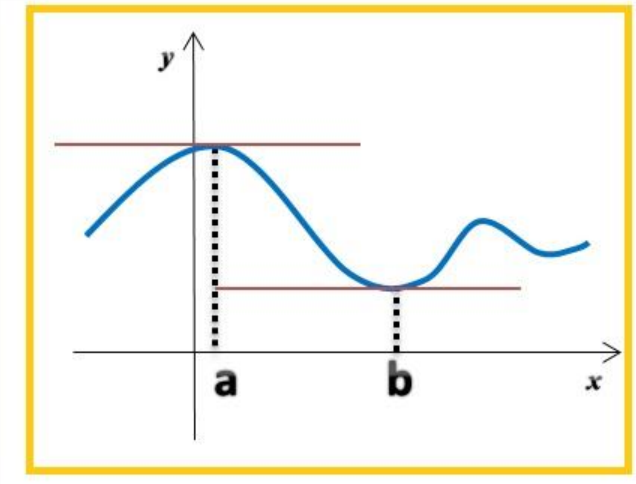


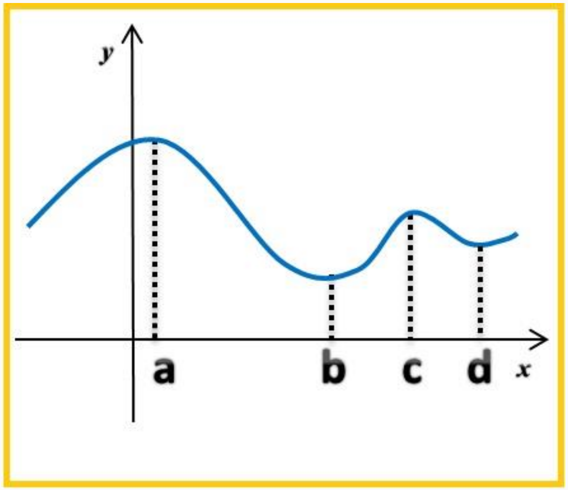
Quando Δx tende a 0, geometricamente a derivada no ponto tende para a tangente (o valor da derivada é o coeficiente angular da reta tangente).

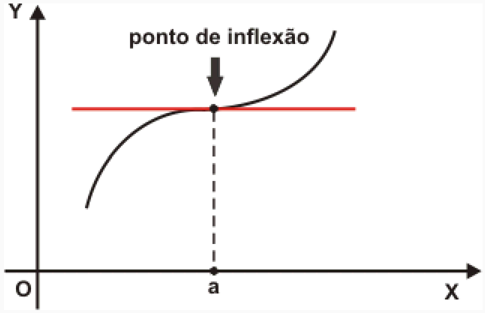
GRADIENT DESCENT

Passo 2 : A derivada e os pontos de máximo e mínimo da função

Se a função tem pontos de máximo/mínimo, eles ocorrem em pontos onde a derivada é nula (veja pontos **a** e **b** abaixo)...porém, ponto de derivada nula pode ser apenas um máximo/mínimo local (pontos c e d) (não global), ou nem ser ponto de máximo/mínimo local (inflexão)







Qual é a função da derivada segunda para esses casos?

Passo 3 : O vetor gradiente (funções com n variáveis)

O gradiente de uma função f, denotado por ∇f ou grad f, é a função vetorial cujas componentes são as *derivadas parciais, ou seja,

$$\nabla f = \left(\frac{df}{dx1}, \frac{df}{dx2}, \ldots\ldots., \frac{df}{dxn}\right)$$

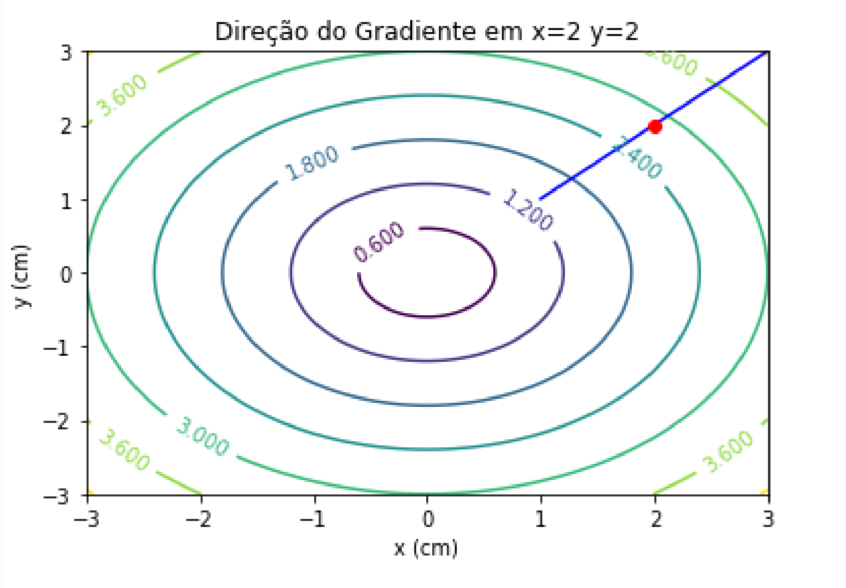
* Derivada em relação a cada uma das n variáveis

GRADIENT DESCENT

Exemplo : $f=z=f(x,y)=x^2+y^2$

$\nabla f = (\frac{df}{dx}, \frac{df}{dy})=(2x,2y)$

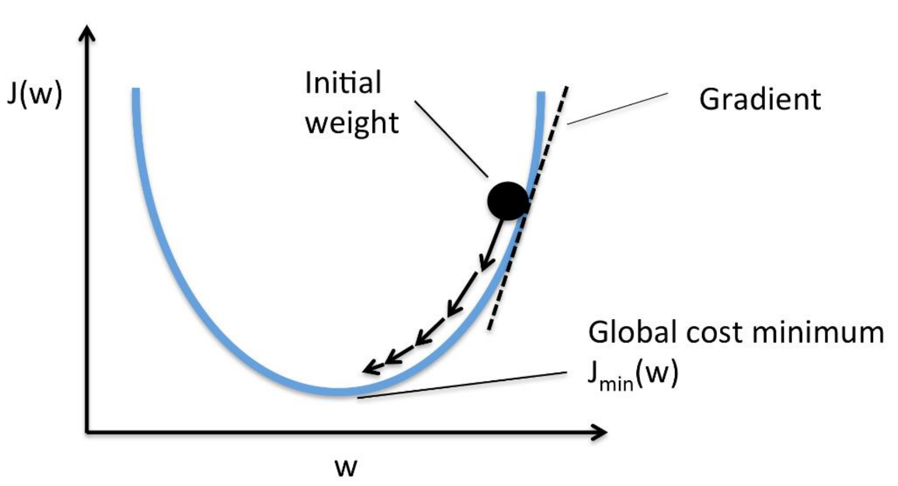
Em $x=2$, $y=2 \Rightarrow \nabla f =(4,4)$



O gradiente tem componentes 4 em x e y no ponto $(x,y)=(2,2)$.

GRADIENT DESCENT

O vetor gradiente indica a direção e o sentido de maior crescimento da função. Assim, se queremos minimizar uma função custo, uma boa ideia é caminharmos na mesma direção e sentido oposto ao gradiente! Essa é a principal ideia do algoritmo Gradient Descent



GRADIENT DESCENT

Conteúdo

Vamos minimizar a função $f(x,y)=3x^2+y^2-200$ usando o Gradient Descent

Partindo de um ponto x,y = 3,4 calculamos o gradiente nesse ponto

$\nabla f = \frac{df}{dx}, \frac{df}{dy}$ = 6x, 2y ..no ponto x,y = 3,4 => ∇f = 18 , 8

Daí nos movemos em x e y na direção contrária do gradiente (multiplicado por um “step”..). Vamos usar step = 0.1

Novo x = 3 – 0.1*18 = 1.2
Novo y = 4 –0.1*8 = 3.2

Recalculando o gradiente em x,y=1.2, 3.2 => ∇f = 7.2 , 6.4 e o
Novo x = 1.2 – 0.1*7.2 = 0.48
Novo y = 3.2 –0.1*6.4 = 2.56

x e y convergirão para 0 e 0..vamos fazer isso em Python

GRADIENT DESCENT

Gradient Descent na Regressão Linear

Partindo de um modelo ye=$\theta_0 + \theta_1 x$ (ye => y estimado pelo modelo e x vindo da amostra)

O erro médio quadrático desse modelo é

(1/m) $\sum_{i=1}^{m}(ye^i - yi)^2$ sendo y, o y real da amostra e m o número de itens da amostra

Ou

(1/m) $\sum_{i=1}^{m}(\theta_0 + \theta_1 \mathbf{x}^i - \boldsymbol{y}^i)^2$ **x** e **y** são vetores com m elementos

Podemos obter o par θ_0, θ_1 que minimiza o erro médio aplicando Gradient Descent

Gradient Descent na Regressão Linear

Para aplicar o Gradient Descent, temos que obter o gradiente da função que dá o erro médio quadrático (fmq), ou seja, obter $\frac{dfmq}{d\theta 0}$ e $\frac{dfmq}{d\theta 1}$

$fmq(\theta_0, \theta_1) = (1/m) \sum_{i=1}^{m} (\theta_0 + \theta_1 \mathbf{x}^i - \boldsymbol{y}^i)^2$

Para derivar *fmq em relação a* θ_0 e θ_1....vamos usar algumas propriedades :

1/m é constante, continuará multiplicando a derivada

A derivada da soma é a soma das derivadas...assim... a somatória continua.

Podemos fazer t= $(\theta_0 + \theta_1 \mathbf{x}^i - \boldsymbol{y}^i)$...assim :

$fmq(\theta_0, \theta_1) = (1/m) \sum_{i=1}^{m} t^2$

Pela regra da cadeia : $\frac{dfmq}{d\theta} = \frac{dfmq}{dt} \frac{dt}{d\theta}$

GRADIENT DESCENT

Conteúdo

Derivadas de fmq em relação a θ_0 e θ_1

$$\frac{dfmq}{d\theta 0} = \overbrace{(2/m)\sum_{i=1}^{m}(\theta_0 + \theta_1\mathbf{x}^{\mathrm{i}} - \boldsymbol{y}^{\boldsymbol{i}})}^{\frac{dfmq}{dt}} \quad \overbrace{1}^{\frac{dt}{d\theta_0}}$$

$$\frac{dfmq}{d\theta 1} = \underbrace{(2/m)\sum_{i=1}^{m}(\theta_0 + \theta_1\mathbf{x}^{\mathrm{i}} - \boldsymbol{y}^{\boldsymbol{i}})}_{\frac{dfmq}{dt}} \quad \underbrace{\boldsymbol{x}^{\boldsymbol{i}}}_{\frac{dt}{d\theta_1}}$$

GRADIENT DESCENT

No **Gradient Descent**, usando um alpha, partimos de um par θ_0, θ_1 que vai sendo atualizado (simultaneamente) para :

$\theta_0 = \theta_0 -$ alpha* $\frac{dfmq}{d\theta 0}$

$\theta_1 = \theta_1 -$ alpha* $\frac{dfmq}{d\theta 1}$ e criamos um critério de parada....(número de iterações, variação de custo entre iterações etc.)

Gradient Descent

Regressão linear (consumo do carro em função da potência) usando gradiente descent

Standardization

Conteúdo

Para aplicarmos o método **Gradient Descent** (e outros métodos), é muito importante efetuarmos “feature scaling”, uma maneira de trazer as diversas features para escalas próximas. Particularmente, no Gradient Descent, escalas diferentes, atrapalham a boa convergência.

Uma das maneiras possíveis, de fazer o “feature scaling” é por “standardization (Z-score normalization)”:

Xs = (x - x_{medio})/σ Xs é a feature X após standardization, Exemplo

x_{medio} é o (1/m) $\sum x$

σ é $\sqrt{\Sigma(x - xmedio)^2/m}$

Feature sem standardization			Feature c/standardization		
2	m	10	-0,875576037	m=10	
3	Média	7,7	-0,721966206	Média=	0
4	Desv. Padrão c/n=10	6,51	-0,568356375	Desvio Desv. Padrão c/n=10	1
5			-0,414746544		
3			-0,721966206		
2			-0,875576037		
9			0,19969278		
12			0,660522273		
23			2,350230415		
14			0,967741935		

Existe também min-max scaling....como é?

Observe que após standardization, a média da feature é zero e o desvio padrão é 1

Scalers

Partindo de scalers.ipynb,

Transforme a feature f com min-max e standardization

Regressão com n features

Conteúdo

- Generalizando a regressão linear com “n” features e Gradient Descent

Regressão com n features

Contexto :

Nas aulas anteriores, para chegarmos aos "parâmetros desejados" θ_0 e θ_1 na regressão linear com apenas uma feature (potência), nossa ideia foi minimizar o "erro médio quadrático" :

Erro= (1/m) $\sum_{i=1}^{m}(ye^{i} - yi)^2$ sendo y, o y real da amostra e m o número de itens da amostra

Ou

Erro= (1/m) $\sum_{i=1}^{m}(\theta_0 + \theta_1\mathbf{x}^{\mathbf{i}} - \boldsymbol{y}^{\boldsymbol{i}})^2$ **x** e **y** são vetores com m elementos (1 feature)

Em seguida, para usar o método Gradient Descent na minimização do erro, obtivemos o Gradiente do erro, fazendo as derivadas parciais em relação a θ_0 e θ_1

Regressão com n features

Podemos definir uma nova função custo $J(\theta)$ como :

$$J(\theta) = (1/2m) \sum_{i=1}^{m} (\overbrace{\theta_0 \mathbf{x_0}^{\mathbf{i}} + \theta_1 \mathbf{x_1}^{\mathbf{i}} + \ \ldots. \ + \theta_n \mathbf{x_n}^{\mathbf{i}}}^{ye} - \boldsymbol{y}^{\boldsymbol{i}})^2$$

O que mudou do erro médio para $J(\theta)$? :

a) Um fator ½ que simplificará as derivadas de $J(\theta)$ em relação a θ (o quadrado cancelará o ½ nas derivadas parciais)

b)O termo " constante, que não multiplica feature" é θ_0 . Para isso, "inventamos" uma feature $\mathbf{x_0}^{\mathbf{i}}$ com valor 1 para todo i ...isso permitirá a generalização da derivada da função $J(\theta)$ para todo θ

c)Houve a generalização para n features

Regressão com n features

Conteúdo

Obtendo o Gradiente :

Se X for uma matriz onde a primeira coluna for de 1´s e as próximas colunas forem de features, podemos generalizar (para “n” features) :

$$\frac{dJ}{d\theta k} = (1/m)\sum_{i=1}^{m}(\theta_0 \mathbf{x}_0{}^{i} + \theta_1 \mathbf{x}_1{}^{i} + \cdots .. + \theta_n \mathbf{x}_n{}^{i} \; - \; \boldsymbol{y}^{i})\boldsymbol{x}_k{}^{i}$$

observe que :

$x_0{}^i = 1$ para todo i

k é a coluna na matriz X (começa do zero)

i é o índice da amostra (que vai de 1 até m) e é também a linha da matriz X.

Regressão com n features

Para o caso de duas features, teríamos a seguinte matriz X e vetor y (exemplos)

$$X = \begin{matrix} 1 & 0.3 & 0.9 \\ 1 & 0.2 & 0.8 \\ 1 & 0.4 & 0.7 \end{matrix} \quad Y = \begin{matrix} 0.2 \\ 0.1 \\ 0.3 \end{matrix}$$

A primeira coluna (índice 0 na verdade) teria só 1´s. A segunda (índice 1) seria a coluna da primeira feature e a terceira coluna (índice 2) seria a coluna da segunda feature. Note que temos m=3 amostras

Como :

$$\frac{dJ}{d\theta k} = (1/m)\sum_{i=1}^{m}(\theta_0 x^i_0 + \theta_1 \mathbf{x_1}^i + \cdots .. + \theta_n \mathbf{x_n}^i \;-\; \boldsymbol{y}^i)\boldsymbol{x_k}^i$$

No exemplo da matriz acima, temos :

$$\frac{dJ}{d\theta 0} = (1/m)((\theta_0 1 + \theta_1 0.3 + \theta_2 0.9 - 0.2)1 + (\theta_0 1 + \theta_1 0.2 + \theta_2 0.8 - 0.1)1 + (\theta_0 1 + \theta_1 0.4 + \theta_2 0.7 - 0.3)1)$$

$$\frac{dJ}{d\theta 1} = (1/m)((\theta_0 1 + \theta_1 0.3 + \theta_2 0.9 - 0.2)0.3 + (\theta_0 1 + \theta_1 0.2 + \theta_2 0.8 - 0.1)0.2 + (\theta_0 1 + \theta_1 0.4 + \theta_2 0.7 - 0.3)0.4)$$

$$\frac{dJ}{d\theta 2} = (1/m)((\theta_0 1 + \theta_1 0.3 + \theta_2 0.9 - 0.2)0.9 + (\theta_0 1 + \theta_1 0.2 + \theta_2 0.8 - 0.1)0.8 + (\theta_0 1 + \theta_1 0.4 + \theta_2 0.7 - 0.3)0.7)$$

Regressão com n features

Conteúdo

Observe que

$$\nabla J(\theta) = \begin{bmatrix} \frac{dJ}{d\theta 0} \\ \frac{dJ}{d\theta 1} \\ \cdots\cdots\cdots \\ \frac{dJ}{d\theta n} \end{bmatrix} = 1/m\ \mathbf{X}^t(\mathbf{X}\theta - \mathbf{y})$$

E...pelo método Gradient Descent, devemos atualizar até a convergência, simultaneamente o vetor θ :

$$\theta \leftarrow \theta - \frac{\alpha}{m} \mathbf{X}^t(\mathbf{X}\theta - \mathbf{y})$$

A velha história... O novo theta é o theta anterior menos o alfa que multiplica o gradiente!..só que estamos agora trabalhando com matrizes e vetores....

Prática de regressão linear

Exercício

Notebook: Regressao_1.ipynb
Partir do dataset cars-uci-Linked.csv

Standardizar o dataset
Obter theta0 e theta1 para feature potência
Verificar erro médio quadrático na amostra de treinamento(?)
Qual o consumo para Potência=160hp?

Prática de regressão linear

Exercício

Fazer cópia de Regressao_1.ipynb => Regressao_2.ipynb
Partir do dataset cars-uci-Linked.csv

Obter thetas para features potência e peso
Verificar erro médio quadrático na amostra de treinamento
Comparar com erro médio do modelo só com potência

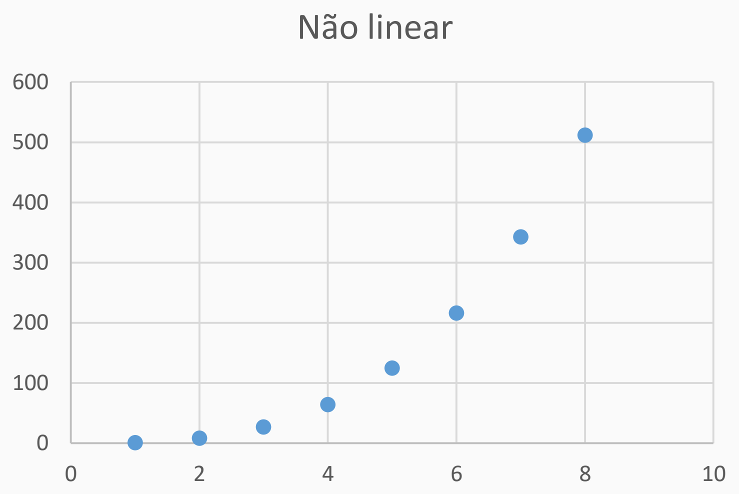
Regressão Polinomial

- Colocando features (não thetas) ao quadrado..ao cubo..etc...

Regressão Polinomial

Usando termos polinomiais (um ou vários) em nossa regressão, pode ser mais fácil a adaptação a determinados data sets (não lineares, obviamente, como no gráfico ao lado). Por exemplo, podemos usar, no caso do carro, o quadrado da potência, além da potência linear.

No novo modelo “matricial”, ficou fácil colocar novas features.



Prática de regressão polinomial

Exercício

Notebook: Regressao_poli (partir do notebook Regressao_2)
e do dataset cars-uci-Linked.csv

Obter thetas para features potência e peso linear e quadráticas
Verificar erro médio quadrático na amostra de treinamento

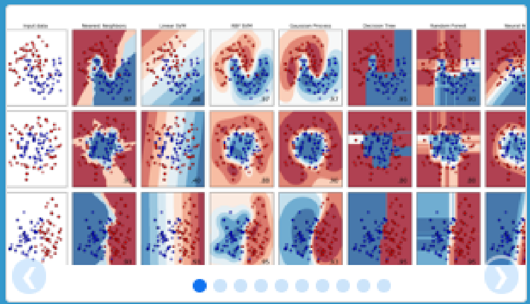
Scikit-Learn (Machine Learning in Python)

Conteúdo

scikit learn

Home | Installation | Documentation | Examples

Google Custom Search



scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, ... — Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, ... — Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, ... — Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency
Algorithms: PCA, feature selection, non-negative matrix factorization. — Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning
Modules: grid search, cross validation, metrics. — Examples

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.
Modules: preprocessing, feature extraction. — Examples

Prática de regressão linear

Exercício

Notebook: Regressao_poli_sklearn.ipynb
Partir do dataset cars-uci-Linked.csv

Obter thetas para features potência e peso linear e quadráticas
Verificar erro médio quadrático na amostra de treinamento

Se sobrar tempo...desafio

Partindo de Regressao_poli_sklearn.ipynb, obter desafio_poli_sklearn.ipynb
Usar dataset cars-uci-Linked.csv

Tentar diminuir o erro na amostra de treinamento com novas features

Como selecionar as features?

Coming soon!

LINKED
EDUCATION

Cursos com Alta Performance de Aprendizado