Introdução

Inteligência na Web e Big Data

Fabricio Olivetti de França e Thiago Ferreira Covões
folivetti@ufabc.edu.br, thiago.covoes@ufabc.edu.br

Centro de Matemática, Computação e Cognição
Universidade Federal do ABC

UFABC

CMCC
Centro de Matemática, Computação e Cognição
Center for Mathematics, Computation and Cognition

Introdução

Mineração de Dados

Extração e descoberta de conhecimentos de bases de dados.

Interdisciplinar: aprendizado de máquina, estatística e banco de dados.

Objetivo: conhecimento entregue de forma legível ou modelo computacional.

Principais Tarefas

Agrupamento dos Dados:

- Segmentação
- Sumarização
- Redução

Principais Tarefas

Classificação:

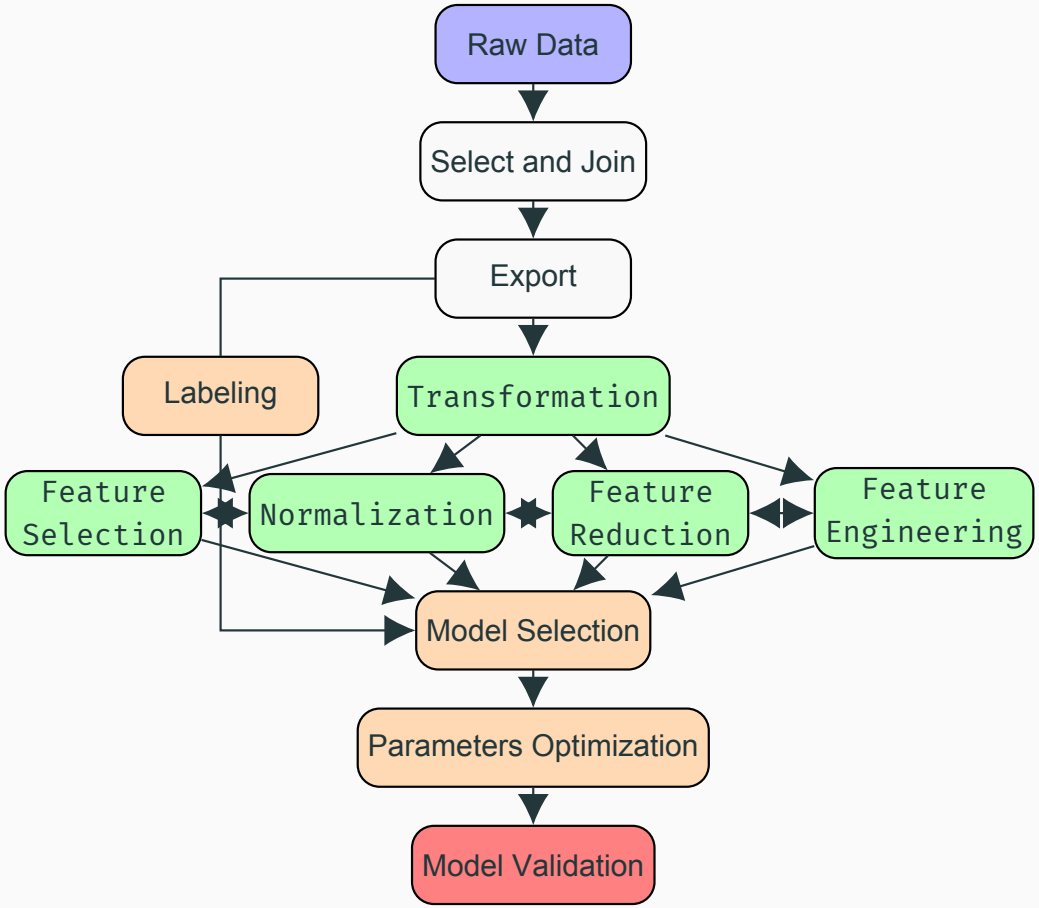
- Rotular dados em classes
- Aplicável em dados ainda não rotulados
- Automatizar tarefa

Principais Tarefas

Regressão:

- Modelo matemático descritivo
- Entender fenômenos

Pipeline


Raw Data
Select and Join
Export
Labeling
Transformation
Feature Selection
Normalization
Feature Reduction
Feature Engineering
Model Selection
Parameters Optimization
Model Validation

Muitos Dados

Novas tecnologias:

- Poder de processamento
- Capacidade de armazenamento
- Capacidade de coleta de dados

Poder de Processamento

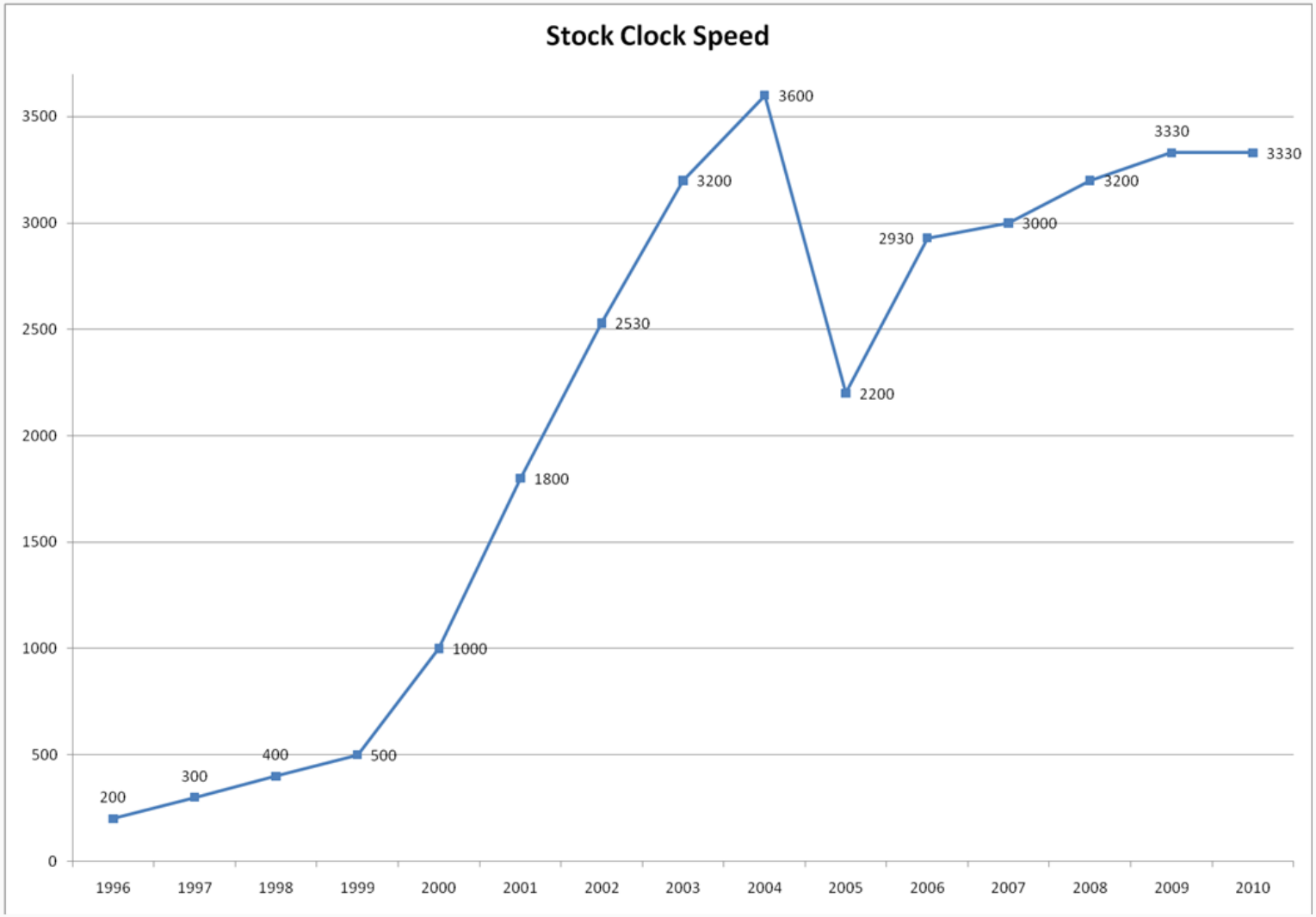


Figura 1: Fonte: maximumpc.com

Poder de Processamento

Não leva em conta apenas o clock:

- Instruções utilizando menos ciclos
- Cache maior
- Pipeline de instruções / Paralelização
- Mais instruções $\rightarrow$ otimização
- Intrinsics
- Multicore

Armazenamento

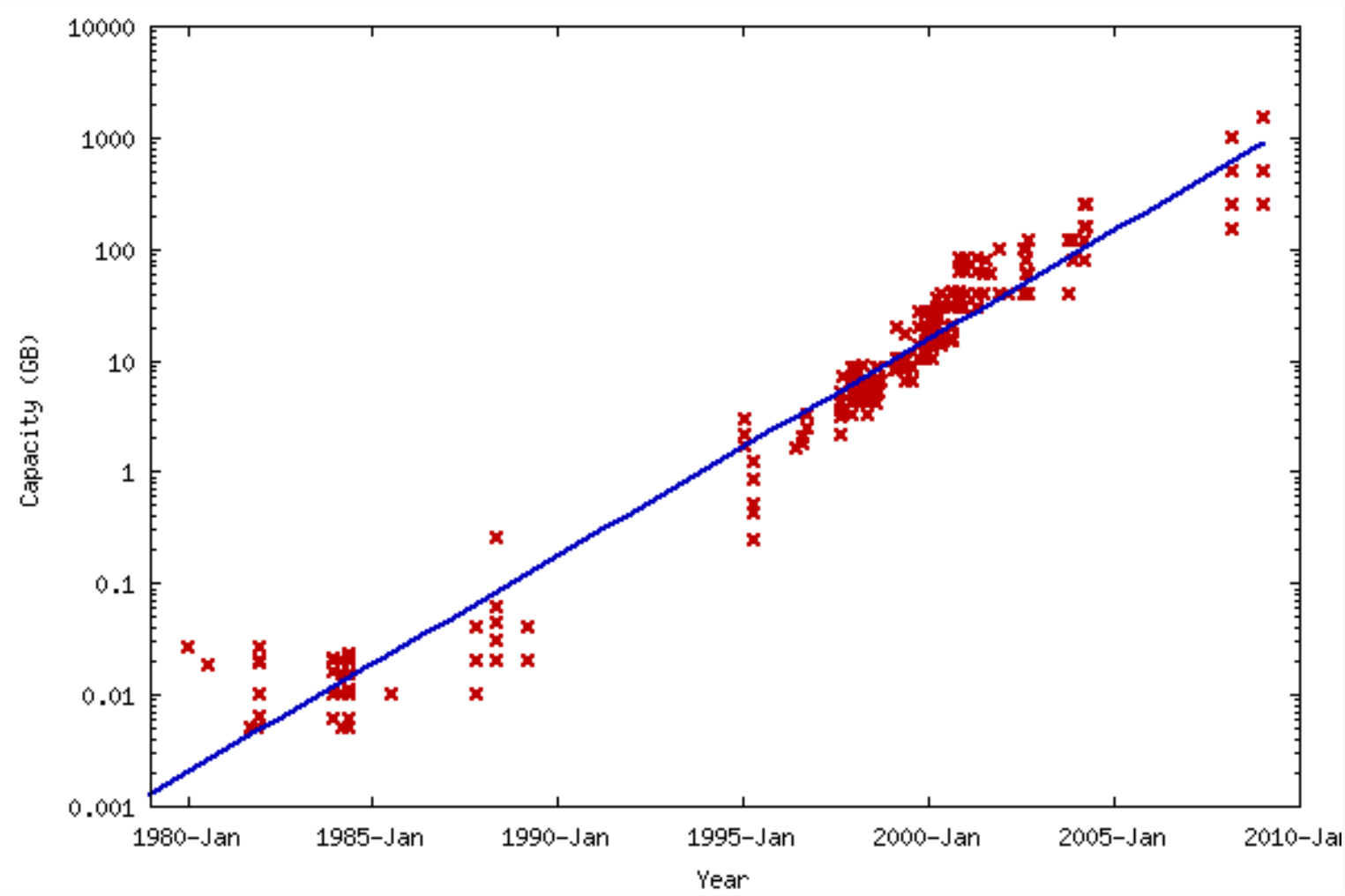


Figura 2: Fonte: wikimedia.org

Armazenamento

- Velocidade e durabilidade também aumentaram.
- Custo dimiuiu com o tempo.
- Computadores capazes de gerenciar múltiplos discos.

Coleta

Novas tecnologias, novas fontes:

- Transações de compras na internet
- Dados de navegação (Google, Facebook)
- Dados de satélite
- Experimentos com genes

Coleta x Armazenamento

O crescimento de capacidade de armazenamento e coleta levou a uma imensa quantidade de dados, que devem ser processados.

Coleta x Armazenamento

Em 2012, a Amazon vendia 158 itens por segundo: **13.651.200** itens em um dia!

Assumindo que cada transação de um cliente continha 5 itens, teremos **2.730.240** transações em um dia.

Coleta x Armazenamento

Em um ano teremos **996.537.600** transações para serem analisadas por técnicas de mineração de dados.

Se cada transação for representada por 24 bytes (otimista), teríamos cerca de 24 Gigabytes de dados.

SPAM

Em 2010, a estimativa era de 3 bilhões de spam enviados por dia.

Sem imagens, um e-mail de spam tem um tamanho médio de 5kb (2007).

Em um dia teremos 15TB de informação para processar!

Os 4 V’s do Big Data

Durante o modismo do Big Data foram definidos 4 desafios dessa área:

- Volume
- Velocidade
- Variedade
- Veracidade

Veracidade

Os resultados extraídos da base de dados são relevantes?

Possibilidade alta de ruídos aleatórios.

Limites Estatísticos

Torture os dados, e confessará qualquer coisa. — *Ronald Coase, economista.*

Limites Estatísticos

Uma crítica da área de Big Data é que a grande quantidade de dados pode te ajudar a encontrar padrões que, na verdade, eram esperados.

Princípio de Bonferroni

Se seu modelo estatístico retorna mais casos positivos do que o esperado, então seu modelo (ou hipótese) não funciona.

Exemplo Fictício

Hipótese: grupos criminosos se encontram em um hotel para planejar o crime.

Objetivo: encontrar pessoas sem relação aparente que, por pelo menos duas vezes, se hospedaram no mesmo hotel no mesmo dia.

Exemplo Fictício

Experimento: rastrear 10^9 pessoas hospedadas em hotéis em um período de 1000 dias.

Cada pessoa ficou em um hotel cerca de 1% desse período (10 dias).

Temos 10^5 hotéis e cada hotel comporta 100 pessoas e assumimos que eles estão sempre lotados.

Exemplo Fictício

Pergunta: se todos agirem aleatoriamente, nosso modelo estatístico encontrará algo suspeito nos dados coletados?

Probabilidades

Duas pessoas $\mathbf{p}, \mathbf{q}$ escolhem um dia específico:

$$P(p, d1) \times P(q, d2) = \frac{1}{100} \times \frac{1}{100} = 10^{-4}$$

Probabilidades

Duas pessoas $\mathbf{p}, \mathbf{q}$ escolhem um hotel específico:

$$P(p, h1) \times P(q, h2) = \frac{1}{10^{-5}} \times \frac{1}{10^{-5}} = 10^{-10}$$

Probabilidades

Essas duas pessoas escolhem o mesmo hotel:

$$P(p, q, h) = 10^{-10} \times 10^{-5} = 10^{-5}$$

Probabilidades

Elas escolhem o mesmo dia:

$$P(p, q, h, d) = 10^{-4} \times 10^{-5} = 10^{-9}$$

Probabilidades

Isso vai ocorrer por duas vezes:

$$P(p, q, h, d1, d2) = 10^{-9} \times 10^{-9} = 10^{-18}$$

Probabilidades

Como temos 5×10^5 pares de dias, a probabilidade de observar tal situação em um deles é:

$$5 \times 10^5 \times 10^{-18} = 5 \times 10^{-13}$$

Probabilidades

Como temos um total de 5×10^{17} pares de pessoas para investigar, a quantidade esperada de observações que faremos em nossa base de dados será:

$$5 \times 10^{17} \times 5 \times 10^{-13} = 2.5 \times 10^{5}$$

Exemplo Fictício

Se temos 10 pares de criminosos que apresentou esse comportamento, teremos que procurá-los dentro das 250.000 pares de pessoas que fizeram o mesmo…

Conclusão

Tenha certeza que a hipótese não permita um número de eventos aleatórios maior do que o que se busca.

Exercício

Em uma base de dados de avaliações de produtos, queremos detectar usuários que são pagos para avaliar uma série de produtos positivamente. É razoável imaginar que se dois usuários avaliaram positivamente em comum 90% dos produtos, então esses foram pagos para isso?

Volume

Como armazenar e tratar a grande quantidade de dados coletada?

Sistemas de Arquivos Distribuídos

- Dividir um arquivo muito grande em várias máquinas conectadas em rede
- Redundância dos dados
- Transparência no uso

Arquivos seguem a filosofia *write-once-read-many-times*.

Ou seja, os arquivos armazenados se tornam somente leitura.

Isso não é problema, pois queremos extrair informações, não alterar os dados.

Tarefas simples e comuns como calculo de estatística básica, procedimentos de seleção, filtragem, junção não são triviais.

Demandam algoritmos que podem ser aplicados localmente em cada uma das máquinas e combinados ao final.

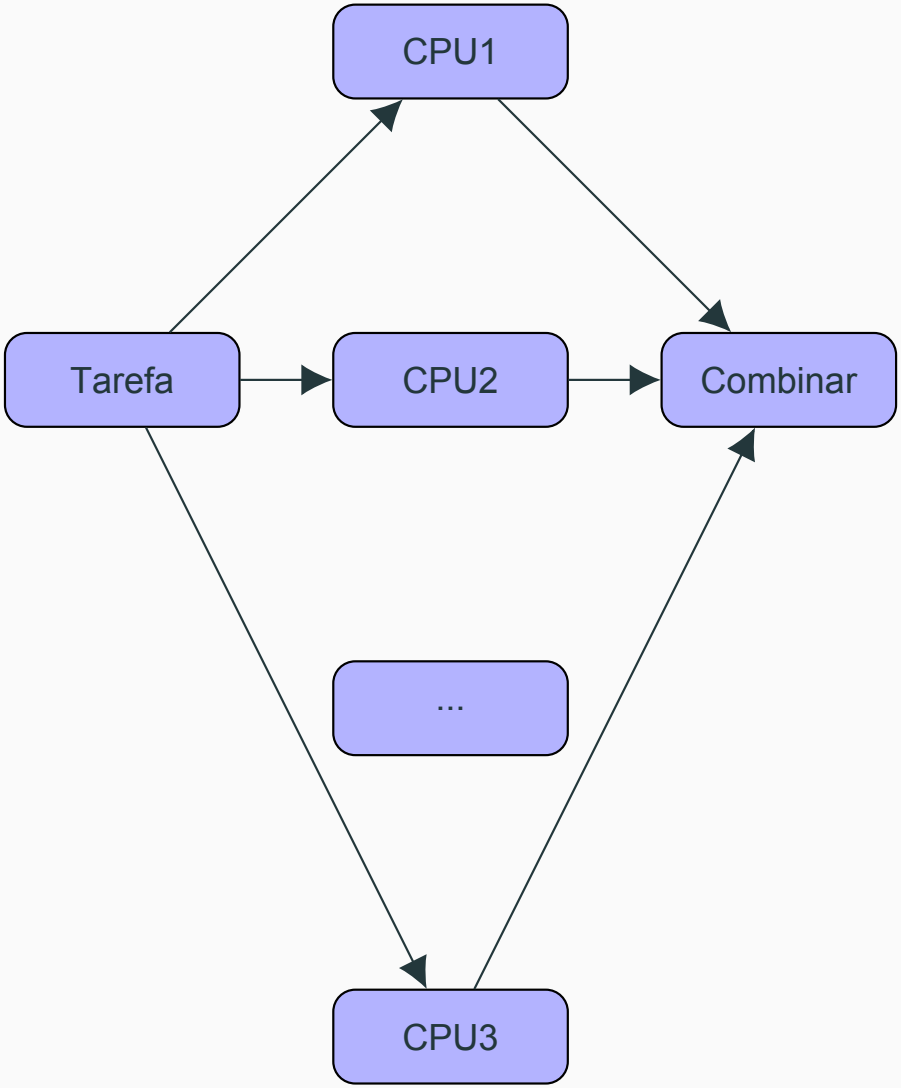
Race condition

A imutabilidade dos dados evita o problema de Race Condition.

Não existe alteração, não existe mudança de estado, não existe conflito!

Particionando a Tarefa

Possibilidade de executar tarefas localmente.



Paradigma Funcional

Um paradigma de programação que não permite mudança de estados e dados mutáveis.

Computação como sequência de funções matemáticas.

Paradigma Funcional

λ-calculus: formalização matemática de computação.

Turing completa.

Propriedades interessantes para processamento em paralelo e distribuído.

Transparência Referencial

Dada a expressão:

$x + f(x)$

Podemos substituir $f(x)$ pelo valor avaliado sem alterar o resultado final:

$x + y$

Pureza

Funções que permitem uma transparência referencial são chamadas de funções puras.

$f(x)$ sempre retornará o mesmo valor ao passar o mesmo valor de x.

Contra Exemplo

```c
int f(int * x) {
  *x = *x + 1;
  return *x;
}
```

```
System.currentTimeMillis()
```

Imutabilidade

Em linguagens funcionais puras uma variável não pode ter seu valor alterado.

Uma variável x passada para uma função $f()$ não pode ter seu valor alterado por ela.

Funcional x Paralelo

Como as variáveis são imutáveis, não existe conflito de acesso.

Se dois processos tentam acessar o conteúdo de uma mesma variável, tenho certeza que ambos receberão o mesmo valor.

Funcional x Paralelo

A pureza das funções e expressões matemáticas, permite que elas sejam avaliadas em qualquer ordem:

$f(x) = sin(x) + cos(x) * sqrt(x)$

Não importa quem vai ser calculado primeiro, sin, cos ou $sqrt$, o resultado final será o mesmo!

Funcional x Paralelo

$$f(x) = sin(x) + cos(x) * sqrt(x)$$

Cada *thread* pode ficar responsável por computar uma parte da expressão.

Linguagens Funcionais

Por muito tempo o uso dessas linguagens foi acadêmico.

Atualmente muitas empresas já fazem uso desse paradigma.

Linguagens Funcionais

Erlang - criado pela Ericsson para programar sistemas tolerante a falhas.

OCaml - uso em sistemas financeiros, programas livres de bugs.

Scala - elementos funcionais no Java.

Linguagens Funcionais

Closure - dialeto do Lisp que roda no Java Virtual Machine.

Haskell - linguagem acadêmica que encontrou seu lugar na indústria (anti-spam do Facebook).

Velocidade

Cenário que recebemos um fluxo contínuo de dados e queremos processá-los em tempo real.

Como tratar?

Algoritmos Incrementais

Algoritmos de aprendizado de máquina que aprendem incrementalmente.

O algoritmo deve se adaptar durante o processo de aprendizagem (mudança de tendências).

Apenas uma quantidade limitada de de dados pode residir em memória.

Variedade

Maior parte dos algoritmos de Aprendizado de Máquina assumem uma entrada de dados bem definida.

Geralmente um vetor multidimensional.

Variedade

Muitas bases de dados de interesse não possuem esse formato:

- Textos
- Imagens
- Som
- Vídeos

Pré-processamento

Necessidade de transformar tipos desestruturados em uma estrutura vetorial.

Cuidado para não perder informação no meio do caminho!

Exercício

Pense em exemplo de tarefas que demanda a combinação de dados de imagem, som e texto.

Conclusão

Trabalhar com grandes quantidades de dados traz diversos problemas desafiadores.

O simples volume de dados não implica em maior facilidade na extração de informação.

Algumas das formas de tratamento dos problemas serão apresentados nas próximas aulas.