Sessão 02

WiDS Recife Live Coding, 21/12/2019

Como serão os live codings?

- Sessões ao-vivo todos os sábados das 14h às 15h
- Código e slides serão disponibilizados no nosso site
- O objetivo é treinar para participar do Datathon em 2020

Recapitulando!

Recapitulando

- O que faz uma Cientista de Dados
- Usando o Pandas e o scikit-learn
- Convertendo dados de string para número
- Slides da sessão passada: https://github.com/widsrecife/live-coding

Começando!

Roteiro

- Selecionar as melhores features para treinar o modelo
- Treinar um modelo usando árvores de decisão
- Avaliar um modelo utilizando o conjunto de testes
- Como se organizar para participar do Datathon no Kaggle

O problema

- Nosso conjunto de dados é composto por atributos de vestidos
- Queremos treinar um modelo onde a gente envie os atributos do vestido e ele diga **qual a melhor época do ano para usá-lo**

Etapas para resolver o problema

1. Importar os dados
2. Explorar os dados
3. Dividir os dados em um conjunto de treinamento e um conjunto de teste
4. Treinar o modelo com o conjunto de treinamento
5. Avaliar o modelo com o conjunto de testes

4. Treinar o modelo

- 4. Treinar o modelo
  - Escolher as variáveis (features) que iremos utilizar
  - Transformar essas features para números
  - SelectKBest e chi2 do scikit-learn

4. Treinar o modelo

- 4. Treinar o modelo
  - Uniformizar as variáveis

Função .apply() do pandas

- Utilizamos a função apply() em uma Series (uma única coluna):
    - df["Season"].apply(string_para_minusculo)
- Mas também podemos usar o apply() em um DataFrame (um dataframe é um conjunto de Series)
    - Para isso precisamos **especificar se queremos aplicar a função nas colunas ou nas linhas**
    - axis=”index”: aplica nas colunas do dataframe
    - axis=”columns”: aplica nas linhas do dataframe

Função .apply() do pandas

- Só que quando aplicada na Series a função recebia o valor de cada instância dos dados, agora aplicada no DataFrame a função recebe cada Series (a coluna inteira)
- A gente pode usar um decorator do Python que lida com a Series automaticamente
    - @np.vectorize
    - def vec_string_para_minusculo(valor):
    - return str(valor).lower()

4. Treinar o modelo

- 4. Treinar o modelo
  - Escolher que algoritmo iremos utilizar
  - DecisionTreeClassifier do scikit-learn

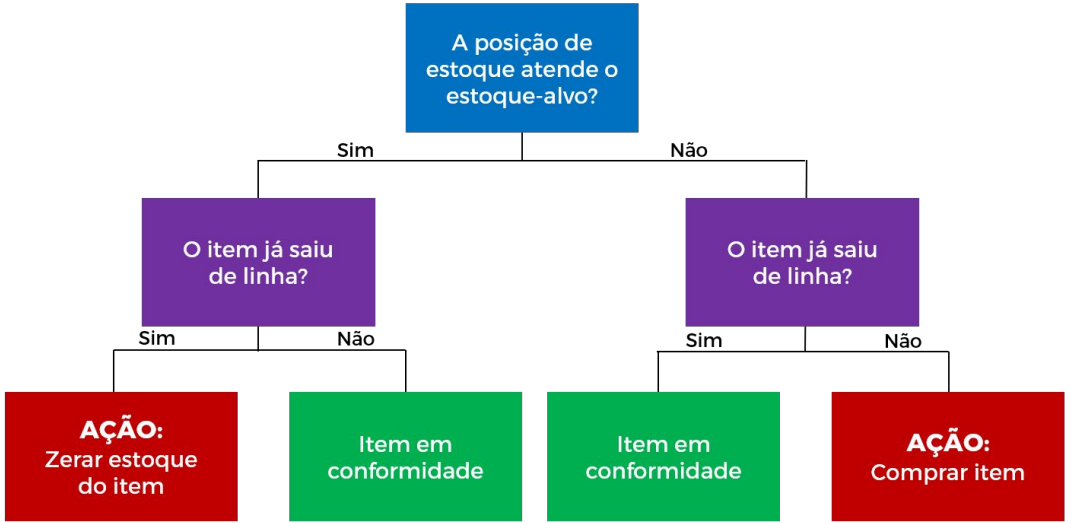
Escolhendo um algoritmo de aprendizagem de máquina

- O scikit-learn implementa vários algoritmos de classificação: https://scikit-learn.org/stable/supervised_learning.html
- Como a API é bem consistente, mesmo tendo vários algoritmos diferentes nós os utilizamos da mesma forma: primeiro chamando o método fit() para treinar o modelo e depois o método predict() para obter o resultado da predição

Árvores de decisão

- Uma árvore de decisão é uma estrutura onde cada nó possui um teste sobre uma feature do conjunto de dados e cada ramo representa um caminho para o resultado deste teste [Kamber (2001)]
- Desta forma, as folhas da árvore representam as classes atribuídas a cada instância do conjunto de dados

Árvores de decisão



Fonte: https://cromasolutions.com.br/aprenda-a-fazer-a-arvore-de-decisao-para-facilitar-a-tomada-de-decisoes/

4. Treinar o modelo

- 4. Treinar o modelo
  - Codificação das variáveis

Codificação das variáveis

- Tínhamos usado o OrdinalEncoder, mas os dados **não são ordinais** e o algoritmo que estamos utilizando leva isso em consideração, então não devemos utilizar esta codificação
- Existem dois tipos de variáveis categóricas
    - Variáveis nominais: não existe ordenação dentre as categorias; exemplo: sexo
    - Variáveis ordinais: existe uma ordenação entre as categorias; exemplo: escolaridade

Codificação das variáveis

- Vamos utilizar então o **OneHotEncoder**
- One hot encoding:
  - Transformar cada valor de feature em uma feature binária, onde os valores possíveis são 0 ou 1
  - Exemplo:
    - Feature cor:
      - Vermelho
      - Roxo
    - Para cada valor de cor será criada uma nova coluna
    - Cor_vermelho, cor_roxo

5. Avaliar o modelo

- 5. Avaliar o modelo
  - Método score() do DecisionTreeClassifier
  - Nos dá a acurácia média do conjunto de testes

E onde entra o kagglee?

Partindo do que vimos para o Kaggle

- Cada etapa (dividir os dados, explorar os dados, treinar, avaliar etc.) **é uma etapa importante para melhorar os resultados do modelo**
- Numa competição normalmente dividimos as tarefas entre as integrantes do time
- Então não se preocupe porque você não precisa ser especialista em todas as etapas (mas é interessante no início a gente trabalhar em todas para descobrir quais gostamos mais de fazer)

Tarefinha de casa

Tarefinha de casa

- Experimentar treinar o modelo com outras features (menos features? mais features? features diferentes?) para descobrir se fazendo isso conseguimos aumentar o score
- **Prazo**: semana que vem, 21/12/2019
- Link do Colab:

Obrigada!

E até semana que vem!

—

Referências

- https://towardsdatascience.com/scikit-learn-decision-trees-explained-803f3812290d
- https://chrisalbon.com/machine_learning/trees_and_forests/visualize_a_decision_tree/