Sessão 03

WiDS Recife Live Coding, 04/01/2020

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

- Usamos o Pandas para explorar os dados
- Usamos o scikit-learn para selecionar as melhores features e treinar o modelo com o algoritmo de Árvore de Decisão
- Slides da sessão passada: https://github.com/widsrecife/live-coding

Começando!

Roteiro

- Avaliar o modelo do conjunto de testes
- Conhecer algumas métricas para avaliar a performance de um modelo e porque a escolha da métrica é algo importante
- Avaliar o modelo para identificar “onde ele está errando”
- 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

5. Avaliar o modelo

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

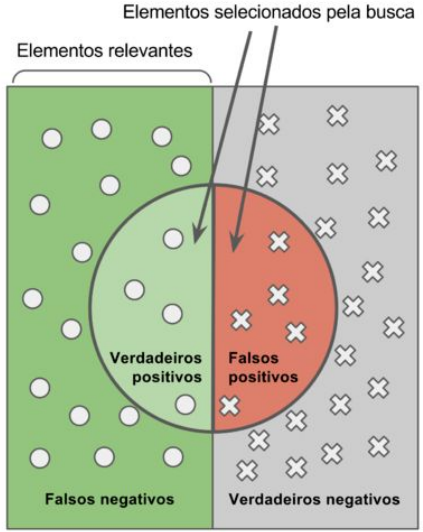
Tipos de problemas de classificação

- Classificação binária
  - Classificar elementos que pertencem a **dois possíveis grupos**
- Classificação multiclasse (multiclass)
  - Classificar elementos em **três ou mais classes**. Nem todos os algoritmos são capazes de lidar com problemas multiclasse

Métricas

- Precisão (precision)
  - Foram selecionados X itens para a classe, destes **quantos são relevantes**?
- Evocação (recall)
  - Quantos elementos relevantes foram selecionados considerando o total de elementos da classe?

Métricas



Precisão = Revocação =

Fonte: https://pt.wikipedia.org/wiki/Precis%C3%A3o_e_revoca%C3%A7%C3%A3o

Métricas

- F1 score
  - Essa métrica combina precision e recall da seguinte forma:
    `F1 = 2 * (precision * recall) / (precision + recall)`
  - No scikit a gente pode escolher os tipos de média quando o problema for multiclasse, por exemplo:
    - 'macro': calcula a métrica para cada label (cada classe) e mostra a média (não leva em conta o class imbalance)
    - 'weighted': calcula a métrica para cada label e acha a média levando em conta a quantidade de instâncias para cada classe

Métricas

- Mais sobre métricas: https://scikit-learn.org/stable/modules/model_evaluation.html

Qual a melhor métrica?

- Isso vai depender do problema que estamos resolvendo. No caso da competição (o Datathon) **eles que dizem** qual métrica será usada para gerar os scores.
- Vamos ver como foi no Datathon desse ano: https://www.kaggle.com/c/widsdatathon2019/overview/evaluation
    - > Submissions are evaluated on **Area under the ROC curve** between the predicted probability and the observed target (has_oilpalm)

Qual a melhor métrica?

- A métrica utilizada foi a **Area Under the Receiver Operating Characteristic Curve (ROC AUC)** (não falamos sobre ela mas ela trabalha com as probabilidades em vez de trabalhar com as labels das predições): https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html#sklearn.metrics.roc_auc_score

5. Avaliar o modelo

- 5. Avaliar o modelo
  - Ver onde o modelo está errando

5. Avaliar o modelo

- 5. Avaliar o modelo
  - Tunar os parâmetros do modelo

Quais os melhores parâmetros para um modelo?

- Os hiperparâmetros são parâmetros dos algoritmos de aprendizagem de máquina que se assemelham a engrenagens pois podemos ajustá-los para conseguir um desempenho melhor para o modelo
- O scikit-learn fornece duas classes que nos ajudam a escolher os melhores parâmetros para um modelo: **GridSearchCV** e **RandomSearchCV**

Classes do Scikit para model tuning

- **GridSearchCV**
  - Testa todas as combinações dos parâmetros
- **RandomSearchCV**
  - Faz testes aleatórios de acordo com a quantidade de testes que queremos

Quais os parâmetros para o DecisionTreeClassifier?

- Existem vários, vamos utilizar estes:
  - criterion: função para calcular se um nó deve ser criado durante a criação da árvore
  - splitter: estratégia para definir o split de cada nó (como cada nó deve ser dividido)
  - max_depth: a profundidade máxima de um nó da árvore

E onde entra o kaggle?

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)

Obrigada!

E até semana que vem!

—

Referências

- https://scikit-learn.org/stable/modules/model_evaluation.html
- https://www.kaggle.com/c/widsdatathon2019/overview/evaluation