

Aprendizado de Máquina

Aula 4.2 - Comitês: Algoritmos


Adriano Rivolli

rivolli@utfpr.edu.br

Especialização em Inteligência Artificial

Universidade Tecnológica Federal do Paraná (UTFPR)
Câmpus Cornélio Procópio
Departamento de Computação

Conteúdo

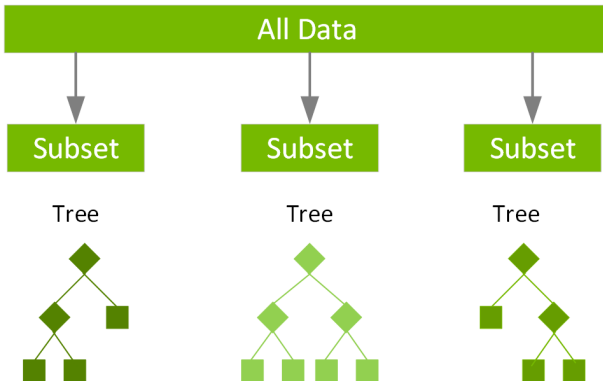
- 
- 1** Random Forest
 - 2** XGBoost
 - 3** Outros comitês

Random Forest


Floresta Aleatória

- Uma implementação de *Bagging*
 - ▶ Combina Bootstrapping com seleção de atributos aleatórios
- Gera diferentes modelos de árvores de decisão cada modelo **vota** em uma classe
- O erro do modelo estabiliza a partir de um limite de árvores
 - ▶ Na implementação do Sklearn o valor padrão é 100
 - ▶ Na implementação em R o valor padrão é 500
- Os modelos também são usados para medir a importância das variáveis preditivas

Random Forest (exemplo)



Treinamento

- 
- Cada árvore é construída a partir de uma amostra com reposição do conjunto de treinamento usando apenas alguns atributos preditivos
 - O número de atributos e as configurações da árvore podem ser definidas por hiperparâmetros

Hiperparâmetros

- **n_estimators** : Número de árvores da floresta
 - ▶ Opções: inteiro positivo, Padrão: 100
- **max_features** : Quantidade de atributos usados para cada árvore
 - ▶ Opções: 'sqrt', 'log2', float (entre 0 e 1), Padrão: 'sqrt'
 - ▶ Quanto menor, maior será a redução da variância, mas também maior será o aumento do viés.
- **Hiperparâmetros da árvore de decisão** :
 - ▶ max_depth, min_samples_split, ...

Predição

- Cada modelo faz uma predição que é considerado um voto

▶ *"In contrast to the original publication, the scikit-learn implementation combines classifiers by averaging their probabilistic prediction, instead of letting each classifier vote for a single class."*


Fonte: <https://scikit-learn.org/stable/modules/ensemble.html#random-forests-and-other-randomized-tree-ensembles>

Importância dos atributos

- Há 2 maneiras de calcular a importância dos atributos em um modelo RF
- **Impureza dos atributos** (scikit-learn)
 - ▶ Utiliza a importância relativa dos atributos usado nas diferentes árvores
 - ▶ Pode não funcionar bem para problemas com alta dimensionalidade
 - ▶ Gerado utilizando apenas os dados de treinamento
- Permutação
 - ▶ Troca valores de atributos e mede o impacto no erro
 - ▶ Uma medida mais robusta, porém demanda uma etapa adicional no processo de treinamento

XGBoost

eXtreme Gradient Boosting

- 
- Sistema escalável para boosting de árvores de decisão
 - No ano de 2015, das 29 competições disponível no Kaggle, 17 delas venceram usando XGBoost
 - Implementação otimizada e altamente eficiente do algoritmo de *Gradient tree boosting*
 -

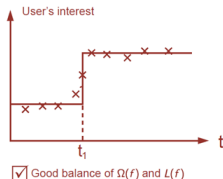
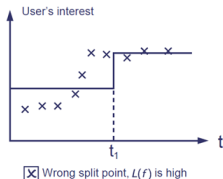
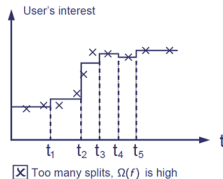
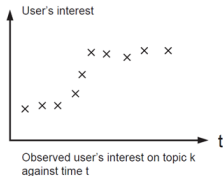
Gradient Tree Boosting

- Conhecido como *Gradient Boosting Machine* (GBM)
- Aplica a técnica de Boosting para árvores de decisão:
 - ▶ Realiza subamostragem dos atributos
 - ▶ Utiliza gradientes descendentes para encontrar a direção em que a função de perda está diminuindo mais rapidamente
- Soma as previsões de cada árvore de decisão ponderadas por um fator de aprendizado (*learning rate*)

Função de custo e Regularização

- A função objetivo otimizada pelo XGBoost consiste em 2 partes: **função de custo** e **termo de regularização**
- Função de custo:
 - ▶ Erro quadrado médio
 - ▶ Função logística
- Termo de regularização
 - ▶ Penaliza modelos complexos
 - ▶ Baseado no número de folhas e na soma dos valores preditos por cada folhas

Compromisso viés e variância



Hyperparâmetros

<https://xgboost.readthedocs.io/en/stable/parameter.html>

■ Geral

- ▶ Relacionado com a escolha do booster a ser utilizado
- ▶ Configurações gerais

■ Booster

- ▶ Dependente das escolhas gerais

■ Learning task

- ▶ Relacionado a tarefa de aprendizado
- ▶ Questões específicas do aprendizado

Hiperparâmetros: Geral

- **booster** : Qual *booster* utilizar
bgtree, bglinear ou dart
- **device** : Dispositivo que irá executar o treinamento
cpu, cuda, gpu, ...
- **verbosity** : usado para debug
0 (silent), 1 (**warning**), 2 (info), 3 (debug)
- **nthread** : Números de *threads* usadas
valor inteiro, Padrão: **total disponível**

Hiperparâmetros: Tree Booster


- **eta** ou **learning_rate** : taxa de aprendizado
valor real entre 0 e 1, Padrão: **0.3**
- **gamma** : Valor mínimo de redução da função de custo
valor real entre 0 e ∞ , Padrão: **0**
- **max_depth** : Altura máxima da árvore
valor inteiro entre 0 e ∞ , Padrão: **6**
- **min_child_weight** : Soma mínima dos pesos das instâncias em um nó
valor inteiro entre 0 e ∞ , Padrão: **0**
- **subsample** : Proporção das instâncias que serão amostradas em cada interação
valor real entre 0 e 1, Padrão: **0.5**

Hiperparâmetros: Learning task

- **objective** : Especifica a tarefa de aprendizado e o tipo da resposta
reg:squarederror, binary:logistic, multi:softmax, multi:softprob, ...
- **eval_metric** : Medida de avaliação utilizada no processo de validação interna
rmse, logloss, auc, ..., Padrão: *depende de cada tarefa*
- **seed** : Especifica a semente da geração aleatória
valor inteiro entre 0 e ∞
- **num_classes** : Define o número de classes a serem previstas
valor inteiro de 2 a ∞ , Padrão: **número de classes**

Outros comitês

Outros algoritmos de comitês

- 
- AdaBoosting
 - CatBoosting
 - ExtraTreesClassifier
 - Gradient Boosting Machine