Boosting

Victor Freguglia; Leonarcho Uchoa Pedreira

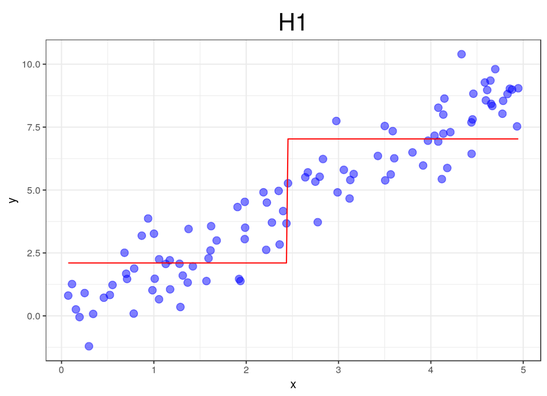
O Conceito de Boosting

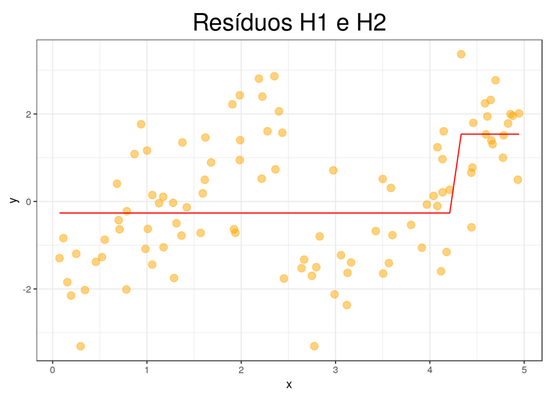
- Combinar um grande número de preditores simples para compor um bom preditor.
- Os preditores (árvores de decisão) são ajustados sequencialmente de maneira a melhorar o desempenho do anterior, ao compreender melhor as regiões com altas taxas de erro.
- Se pensarmos que cada árvore tem um “voto” na decisão final, o método nos fornece um comitê em que a decisão final é a ponderação de todos os votos. Aqueles que têm grande convicção têm mais poder na decisão.

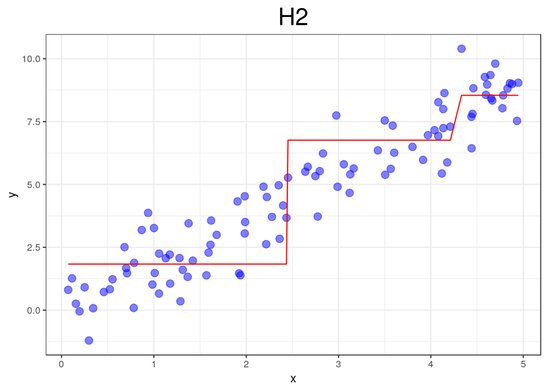
Visualização

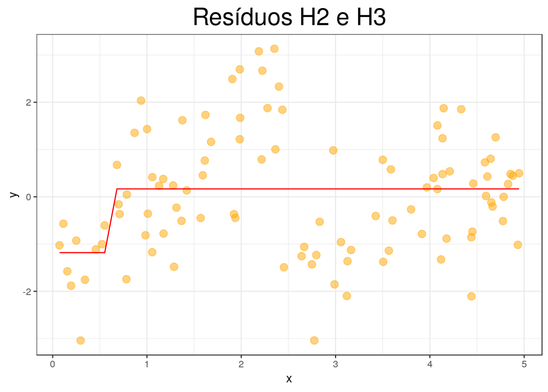
Primeiras iterações

Resíduos

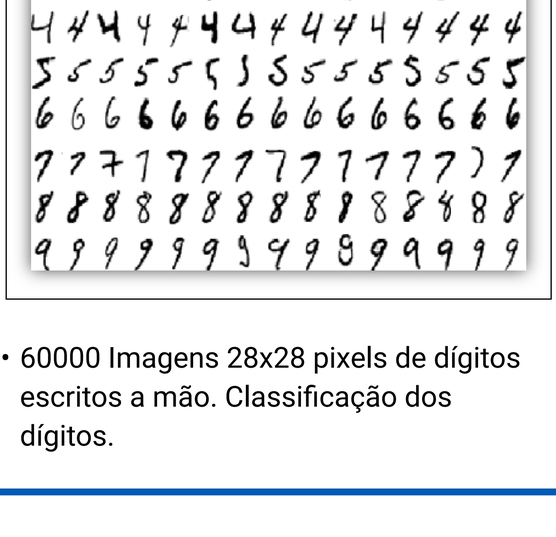


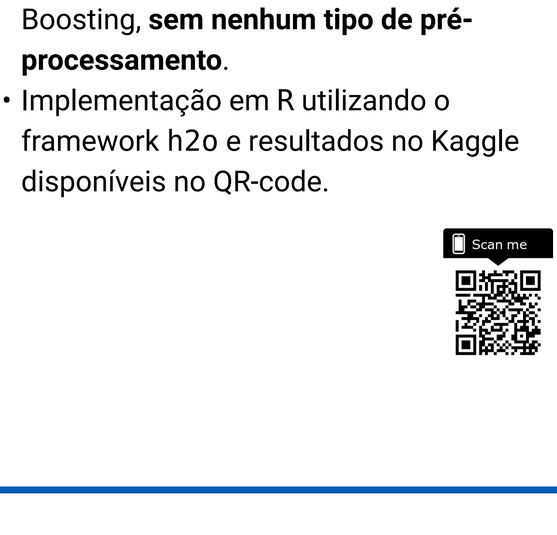






...





Uma aplicação

Conjunto de dados MNIST

- 60000 Imagens 28x28 pixels de dígitos escritos a mão. Classificação dos dígitos.

Resultados

- Acurácia de 97.33% no conjunto de teste da competição no Kaggle, utilizando Gradient Boosting, com taxa de aprendizado $\alpha = 0.08$, árvores com 7 nós nos classificadores e 600 passos de Boosting, **sem nenhum tipo de pré-processamento**.
- Implementação em R utilizando o framework h2o e resultados no Kaggle disponíveis no QR-code.

Scan me

Gradient Boosting

Direção fornecida da solução pelo método do Gradiente e sua conexão com os resíduos fornecem a intuição e a engrenagem por detrás do método.

Algoritmo

- $(y_i, x_i), i = 1, \ldots, N$;
- $L(\cdot, \cdot)$ - função perda;
- Inicie com o preditor constante $H_0 = \arg\min_c \sum_{i=1}^{N} L(y_i, c)$;
- Para $m = 1, \ldots, M$ faça:

1. Calcular os pseudo-resíduos
$$r_{im} = -\left[\frac{\partial L(y_i, G(x_i))}{\partial G(x_i)}\right]_{G(x_i)=H_{m-1}(x_i)};$$
2. Ajustar uma nova árvore h_m aos resíduos (r_{im}, x_i);
3. Calcule o peso do voto γ_m como
$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L\left(y_i, H_{m-1}(x_i) + \gamma h_m(x_i)\right);$$
4. Atualize o modelo:
$$H_m(x) = H_{m-1}(x) + \gamma_m h_m(x);$$

- Defina o preditor final como
$$G(x) = H_M(x).$$

Vantagens

- Alto poder de preditivo (ganhador de vários concursos recentes)
- Otimização direta da função perda (regressão, classificação, regressão quantílica, Modelo de Cox, etc...).
- Não requer nenhum tipo de pré-processamento e pode ser paralelizado.

Desvantagens

- Alto custo computacional;
- Pode ser muito sensível a dados muito ruidosos;
- Requer refinamento cuidadoso;

Refinamento

Principais refinamentos do Gradient Boosting incluem:

- Subamostragem: Sortear um subconjunto de tamanho $N' < N$ da amostra de treino para o ajuste do preditor a cada passo.
- Penalização: Encolher os preditores por algum valor $\alpha < 1$, isto é, substituir o passo de atualização por
$$H_m(x) = H_{m-1}(x) + \alpha\gamma_m h_m(x).$$
- Número de nós terminais: Utilizar diferentes números de nós terminais para as árvores.

Implementação

- Pacote `gbm`: Generalized Boosted Regression Models;
- Pacote `xgboost`: Extreme Gradient Boosting;
- Plataforma `h2o`: www.h2o.ai;
- Pacote `LightGBM`: Implementação da Microsoft para gbm;

Conclusão

- Boosting produz preditores muito eficazes;
- Extretamente versátil;
- Apesar de incluir diversos parâmetros de refinamento, exceto em casos específicos, muitos não tem grande efeito na qualidade final das predições; Por outro lado, podem drasticamente reduzir o custo computacional ou alterar a quantidade necessária de passos até produzir um bom preditor;
- A forma com que o algoritmo é construído causa a impressão de que o overfitting deve ocorrer, mas a quantidade necessária de passos para que ele de fato ocorra é muito grande.

Referências

- Friedman, J. H. (2001), ‘Greedy function approximation: a gradient boosting machine’, Annals of statistics pp. 1189–1232.
- Friedman, J., Hastie, T. & Tibshirani, R. (2001), The elements of statistical learning, Vol. 1, Springer series in statistics New York, NY, USA:.
- Schapire, R. E. (1990), ‘The strenght of weak learnability’, Machine Learning 5, 197–227.