Universidade Estadual de Campinas
Instituto de Matemática, Estatística e Computação Científica
Departamento de Estatística

Avaliação 2

ME906 - Métodos em Aprendizado Supervisionado Máquina
Professor Dr. Ronaldo Dias

Nome	**RA**	**Contribuição**
Andre Saul Juarez Castro	272208	20%
Isabella Nascimento Peres da Silva	238015	20%
João Pedro de Campos Formigari	236144	20%
Mateus Lee Yu	236235	20%
Pedro Luis Rebollo	217460	20%

Campinas - SP

1 Árvores de Regressão (Regression Trees)

Seja $\{(\mathbf{X}_{p\times 1}^{(i)}, Y^{(i)})_{i=1}^{n}\}$ o conjunto de treino dos nossos dados, onde o vetor $\mathbf{X}_{p\times 1}^{(i)} = [X_1 \cdots X_p]^T$ representa as p características e $Y^{(i)}$ uma variável contínua que estamos interessados em predizer. Suponha que queremos predizer o valor de y para $\mathbf{x}$ pertencente a região R_j, o modelo de árvores de regressão calcula $\hat{y}$ como sendo a média da variável resposta da região R_j, formalmente

$$\hat{y}_{R_j} = \sum_{j=1}^{J} \mu_j \mathbb{1}_{\{\mathbf{x}\in R_j\}}(\mathbf{x}), \quad \text{onde} \quad \mu_j = \frac{1}{n_j} \sum_{i:\mathbf{x}^{(i)}\in R_j} y^{(i)} \quad \text{e} \quad n_j = \sum_{i:\mathbf{x}^{(i)}\in R_j} 1.$$

As regiões $R_1, \ldots, R_J$ serão obtidas minimizando a soma dos quadrados dos resíduos (SQR), dada por

$$SQR = \sum_{j=1}^{J} \sum_{i\in R_j} \left(y^{(i)} - \hat{y}_{R_j}\right)^2,$$

infelizmente por conta do custo computacional requerido não é possível considerar todos as partições possíveis das J regiões, logo utilizaremos um algoritmo com uma abordagem conhecida como divisão binária recursiva (*recursive binary splitting*), é um método de cima para baixo (*top-down*) pois começa com todas as observações em uma única região, é ambicioso (*greedy*) por que realizamos cada uma das divisões baseadas no melhor valor possível da soma dos quadrados dos resíduos.

Algoritmo 1: Divisão Binária Recursiva

Passo 1: Começar com todas as observações em uma única região;

Passo 2: Para cada $j = 1, \ldots, p$:

Para cada ponto s:

Dividir em duas regiões $R_1(j,s) = \{\mathbf{X}|X_j < s\}$ e $R_2(j,s) = \{\mathbf{X}|X_j \geq s\}$ e calcular

$$\sum_{i:\mathbf{x}^{(i)}\in R_1(j,s)} \left(y^{(i)} - \hat{y}_{R_1}\right)^2 + \sum_{i:\mathbf{x}^{(i)}\in R_2(j,s)} \left(y^{(i)} - \hat{y}_{R_2}\right)^2; \tag{1}$$

Passo 3: Escolher o par (j,s) que minimiza a equação 1;

Passo 4: Para cada uma das regiões R_1 e R_2, voltar para o passo 2;

Passo 5: Parar até o critério de parada for satisfeito.

Após o procedimento das divisões é provável que temos um sobre-ajuste nos dados do conjunto de treino, para contornar esse problema podemos realizar uma poda na árvore (*tree pruning*) para obtermos uma sub árvore tal que a variância seja reduzida e o vício maior. Uma estratégia seria obter todas as sub árvores possíveis e escolher aquela com o menor erro de validação cruzada, contudo o custo computacional torna esse caminho inviável.

O critério de custo de complexidade pode ser utilizado na poda em uma árvore T_0, ao invés de considerarmos

todas as sub árvores possíveis, olhamos para as sub árvores $T \subset T_0$ que minimizam

$$\sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} \left(y_i - \hat{y}^{(i)}\right)^2 + \alpha|T|, \quad \alpha > 0 \quad \text{e} \quad |T| \text{ é o número de nó terminal de } T. \tag{2}$$

Algoritmo 2: Construção da Árvore de Regressão

Passo 1: Começar com uma árvore T_0 obtida no algoritmo 1;

Passo 2: Obter uma sequência de sub árvores utilizando a equação 2;

Passo 3: Usar validação cruzada K-fold para obter α;

Passo 4: Retornar a árvore do passo 2 com o valor de α obtido.

---

2 Árvores de Classificação (Classification Trees)

As árvores de classificação são semelhantes as árvores de regressão com a diferença de que a variável resposta Y é qualitativa ao invés de quantitativa, o modelo de árvores de classificação classifica y para a classe com a maior proporção nas observações da região, formalmente

$$\hat{y}_{R_m} = \arg\max_k \hat{p}_{mk}, \quad \text{onde} \quad \hat{p}_{mk} = \frac{1}{n_m} \sum_{i:\mathbf{x}^{(i)} \in R_m} \mathbb{1}_{\{y^{(i)}=k\}}$$

é a proporção de observações pertencentes a classe k na região m.
As regiões podem ser encontradas utilizando o mesmo algoritmo das árvores de regressão, porém utilizamos outras métricas no lugar da soma dos quadrados dos resíduos, uma forma simples seria a taxa de erro de classificação

$$E = 1 - \arg\max_k \hat{p}_{mk},$$

porém na prática não é utilizada por conta da falta de sensibilidade, uma alternativa seria o índice de Gini definido por

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}),$$

que mede a variância total entre as classes, outro método que é numericamente similar com o índice de Gini seria, a entropia dado por

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log(\hat{p}_{mk}).$$

3 Floresta Aleatória (Random Forest)

A floresta aleatória é um método *ensemble*, ou seja, as predições são calculadas a partir de um conjunto de modelos, quando temos variáveis quantitativas utilizaremos a média aritmética de um conjunto de árvores de regressão e caso seja qualitativa classificamos para a classe com maior frequência nas predições de um conjunto de árvores de classificação.
O modelo de cada uma das B árvores de decisões são ajustadas a partir de amostras com reposição do conjunto de treino (*bootstrap*), porém utilizaremos um subconjunto de dimensão m das p características, usualmente escolhemos $m \approx \sqrt{p}$.
A razão por não utilizar todas as variáveis preditivas está no objetivo de descorrelacionar as árvores, suponha que temos uma variável preditora muito forte, se utilizarmos as p variáveis todas as árvores utilizarão a mesma variável para realizar as divisões, logo as árvores se tornarão altamente correlacionadas, porém se utilizarmos um subconjunto das características as árvores são descorrelacionadas e a variância de cada uma delas é reduzida.

Com o intuito de entender de forma mais prática sobre o método de Floresta Aleatória, ele foi aplicado em um banco de dados que possui mais de 4600 observações e 58 variáveis, de tal forma que as observações são dados de e-mails em que a variável principal é se eles são SPAMs ou não. Dessa forma, o método foi proposta para classificar diferentes e-mails como SPAM ou não e quais características dos mesmos que poderiam estar associadas a esse fato.

Após uma análise prévia, foi decidido que o valor de m, dado que $m \approx \sqrt{p}$, seria igual a 8.

A Tabela 1 apresenta os resultados obtidos pelos modelos treinados no conjunto de teste. Nota-se que a precisão de acertos foi bem alta. A coluna representada por "0"indica que não é SPAM e "1"que é SPAM.

Tabela 1: Matriz de Confusão - Floresta Aleatória

	0	1
0	792	32
1	40	517

Com o método de Floresta Aleatória, obtivemos uma acurácia de mais de 94,7% na classificação dos e-mails como SPAM ou não, tendo acertado a classificação de 1309 dos 1381 testes feitos.

4 Bagging

4.1 Introdução

Bagging, que é uma abreviação de *Bootstrap Aggregating*, é uma técnica de aprendizado de máquina projetada para melhorar a estabilidade e a precisão de algoritmos de aprendizado de máquina. Desenvolvida por Leo Breiman em 1994, ela se destaca por reduzir a variância e ajudar a evitar o overfitting. Embora seja aplicável a vários tipos de métodos de aprendizado, é mais comumente associada com árvores de decisão.

O princípio fundamental do Bagging é que uma combinação de modelos de aprendizado aumenta o resultado geral do processo do que um único modelo. Isso é especialmente verdadeiro para modelos com alta variância, como árvores de decisão.

4.2 Metodologia do Bagging

Diferentemente de outras técnicas de ensemble, o Bagging aplica a amostragem tipo bootstrap para criar diversos conjuntos de treinamento, cada um gerando uma instância independente do modelo.

Etapas do Processo de Bagging:

1. Inicialmente, selecionam-se B conjuntos de treinamento de forma aleatória do dataset original, utilizando a técnica de amostragem com substituição, conhecida como bootstrap. Cada conjunto bootstrap representa uma amostra única e independente do conjunto de dados original.

2. Um modelo de predição é treinado separadamente em cada um destes conjuntos bootstrap. No contexto de árvores de decisão, cada árvore é construída com base em um desses conjuntos.

3. As previsões de todos os modelos são então combinadas para formar a saída final. O método de combinação varia conforme o tipo: classificação ou regressão.

Equação do Bagging para Regressão: Para problemas de regressão, as previsões dos modelos são combinadas utilizando a média, conforme a equação a seguir:

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^b(x) \tag{3}$$

onde $\hat{f}^b(x)$ é a previsão do modelo treinado com o b-ésimo conjunto bootstrap, e a média dessas previsões é calculada para produzir a previsão final do Bagging.

Equação do Bagging para Classificação: A agregação das previsões nos modelos de classificação é realizada por meio de votação majoritária, conforme expresso na seguinte equação:

$$\hat{f}_{\text{bag}}(x) = \text{moda}(\hat{f}^1(x), \hat{f}^2(x), \ldots, \hat{f}^B(x)) \tag{4}$$

4.3 Vantagens

- Reduz a variância de modelos individuais, diminuindo o risco de overfitting. Esta característica é particularmente benéfica para algoritmos de alta variância, como árvores de decisão.

- Melhora a robustez e a precisão do modelo final em comparação com um único modelo de treinamento.

- Simples de implementar e paralelizar, tornando-o uma técnica prática para melhorar modelos existentes.

4.4 Desvantagens

- Aumenta o custo computacional devido à necessidade de treinar múltiplos modelos. Esse custo adicional pode ser significativo, especialmente em conjuntos de dados grandes ou com modelos complexos.

- Pode não ser tão eficaz se os dados forem muito ruidosos ou se os modelos individuais forem muito enviesados. Nestes casos, o Bagging pode não conseguir melhorar suficientemente a precisão do modelo.

- Modelos de ensemble como o Bagging podem ser mais difíceis de interpretar do que um único modelo de decisão, comprometendo a transparência do processo de tomada de decisão.

4.5 Aplicação

Com o intuito de entender de forma mais prática sobre o método de Bagging, ele foi aplicado a um banco de dados que é resultado de um compilado de mais de 4600 e-mails coletados de diferentes pessoas e de 57 características diferentes desses e-mails, com o intuito de elencar quais eram e quais não eram SPAMs e quais características dos mesmos que poderiam estar associadas a esse fato.

Após uma análise prévia, foi decidido que o valor de m, dado que $m \approx \sqrt{p}$, seria igual a 8. A partir disso foram-se selecionados os conjuntos de treinamento e foram ajustados os modelos de predição.

A Tabela 2 apresenta os resultados obtidos pelos modelos treinados no conjunto de teste. Nota-se que a precisão de acertos foi bem alta. A coluna representada por "0"indica que não é SPAM e "1"que é SPAM.

Tabela 2: Matriz de Confusão - Bagging

	0	1
0	808	28
1	32	513

Com o método de Bagging, obtivemos uma acurácia de mais de 95,6% na classificação dos e-mails como SPAM ou não, tendo acertado a classificação de 1321 dos 1381 testes feitos.

5 Boosting

5.1 Introdução

Nesta seção será apresentado o método Boosting restringido ao contexto de árvores de decisão em problemas de classificação.

Boosting é uma abordagem que pode ser utilizada em diversos métodos de aprendizado de máquina com finalidade de aprimoramento das previsões obtidas através do uso dos mesmos, nesse caso, obtidas através das árvores de decisão.

Por mais que este não tenha um início claro a motivação por trás do seu desenvolvimento ocorreu por volta de 1988 quando os autores Kearns e Valliant, apresentaram a hipótese em suas pesquisas de que talvez um conjunto de *weak learners* pudesse resultar em um único *strong learner*, esta foi sendo estudada até que puderam ser constatados os benefícios do seu uso, sendo hoje o Boosting amplamente utilizado em diversas áreas de atuação de aprendizado de máquina.

5.2 Metodologia

Agora será exibida a metodologia empregada na utilização do Boosting.

Conforme já comentado anteriormente a ideia por trás deste procedimento é de que utilizar um aprendizado lento com o ajuste de mais de uma árvore de decisão ao invés de uma única, que poderia resultar em *overfitting*, tende a acarretar em melhores previsões.

Para a realização do Boosting são necessários três parâmetros, sendo esses:

- B = número de árvores de decisão
- λ = parâmetro de encolhimento
- d = número de divisões em cada árvore

Onde B é escolhido através de *cross-validation*, e indo ao oposto de Bagging e Random Forests, um B muito grande pode resultar em *overfitting*, porém este na maioria das vezes ocorre lentamente. Os valores mais comuns para *lambda* são de 0.01 ou 0.001, estando este controlando a taxa de aprendizado do Boosting, vale ressaltar que escolher pequenos valores para esse paâmetro pode exigir, afim de ter um boa performance, grandes valores para B. Já d normalmente ocasiona em boas previsões quando tomado o valor de 1, ou seja, cada árvore consiste em apenas uma divisão (ou *splits*) e o modelo se trata de um modelo aditivo, de forma que este parâmetro pode então ser visto como um controlador da ordem de interação do modelo.

Visto os parâmetros utilizados no boosting, segue-se o algoritmo utilizado na sua implementação:

Primeiramente define-se $\hat{f}(x) = 0$ e $r_i = y_i$, onde $1 \leq i \leq n$, sendo n o número de observações no conjunto de treinamento.

Depois realiza-se B vezes os procedimentos: Ajustar uma árvore $\hat{f}^b$ com d divisões ao conjuto de treinamento (X, R). Atualiza-se $\hat{f}$, adicionando a essa uma versão encolhida da árvore obtida no passo anterior, ou seja, $\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x)$. Atualiza-se os resíduos de forma que $r_i \leftarrow r_i - \lambda \hat{f}^b(x)$. Onde $b = 1, 2, ..., B$.

A última etapa consite na retono do modelo (*boosted model*) obtido pelo agoritmo, dado então por:

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x) \tag{5}$$

5.3 Vantagens

- Possui um algoritmo de fácil compreensão e consequentemente de fácil implementação
- Apresenta menos viés do que outros métodos a serem utilizados em aprendizado de máquina, devido ao uso que este faz da combinação de *weak learners*

5.4 Desvantagens

- Este método não é robusto a valores discrepantes, podendo esseso distorcer os resultados a serem obtidos de maneira expressiva
- Pode ser computalcionalmente caro, visto que os estimadores são obtidos através daqueles já encontrados anteriormente, o que torna o algoritmo mais lento quando comparado com o Bagging por exemplo

5.5 Aplicação

Com o intuito de entender de forma mais prática sobre o método de Boosting, ele foi aplicado, como no caso anterior, a um banco de dados que é resultado de um compilado de mais de 4600 e-mails coletados de diferentes pessoas e de 57 características diferentes desses e-mails, com o intuito de elencar quais eram e quais não eram SPAMs e quais características dos mesmos que poderiam estar associadas a esse fato.

Após uma análise prévia, foi decidido que o valor de m, dado que $m \approx \sqrt{p}$, seria igual a 8. Além disso, foi definido a criação de 500 árvores com um número de divisões d igual a 1.

A Tabela 3 apresenta os resultados obtidos pelos modelos treinados no conjunto de teste. Nota-se que a precisão de acertos foi bem alta. A coluna representada por "0"indica que não é SPAM e "1"que é SPAM.

Tabela 3: Matriz de Confusão - Boosting

	0	1
0	809	27
1	31	514

Com o método de Boosting, obtivemos uma acurácia de mais de 95,8% na classificação dos e-mails como SPAM ou não, tendo acertado a classificação de 1323 dos 1381 testes feitos.

6 Avaliação das Técnicas de Classificação

A matriz de confusão é uma das formas para avaliarmos a performance de um classificador, após ajustarmos um modelo, realizamos as predições em um conjunto de teste de tamanho n e sumarizamos em uma tabela $K \times K$, onde a entrada na i-ésima linha e j-ésima coluna representa o número de observações

pertencentes a classe j que foram classificadas na classe i. A matriz de confusão pode ser encontra na tabela 4.

	Classe Verdadeira		
Classe Predita	Classe 1	$\cdots$	Classe K
Classe 1	$n_{1,1}$	$\cdots$	$n_{1,K}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$
Classe K	$n_{K,1}$	$\cdots$	$n_{K,K}$

Tabela 4: Matriz de Confusão

A acurácia das predições pode ser calculada somando os elementos da diagonal principal e dividindo pelo número de observações do conjunto de teste.

$$\text{Acurácia} = \frac{1}{n}\sum_{i=1}^{K} n_{i,i}$$

Referências

[1] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (1st ed.). Springer.

[2] Breiman, L. (1994). Bagging Predictors.Department of Statistics University of California Berkeley, California, 10-13.