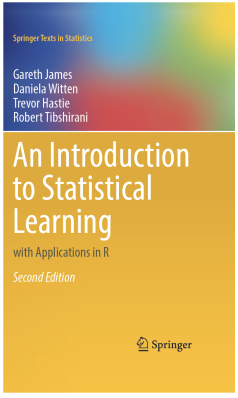
Aula 6: Revisão da Semana 2

Machine Learning

Paulo Orenstein

Verão, 2025
IMPA



Capítulo 4: Métodos lineares para classificação

Créditos de figuras e slides: James, Witten, Hastie e Tibshirani

Modelos lineares generalizados (GLM)

- Quando y é real, usamos regressão linear:

$$\hat{\beta}_0, \ldots, \hat{\beta}_p = \underset{\tilde{\beta}_0, \ldots, \tilde{\beta}_p}{\operatorname{argmin}} \sum_{i=1}^{n} (y_i - (\tilde{\beta}_0 + \tilde{\beta}_1 x_{i1} + \cdots + \tilde{\beta}_p x_{ip}))^2$$

- E se y é inteiro positivo? binário? categórico?

- Metodologia geral: modelos lineares generalizados
  1. Escrever log-verossimilhança de Y
  2. Modelar parâmetro que multiplica y na log-verossimilhança como uma função linear de x e $\hat{\beta}$
  3. Encontrar o valor de $\hat{\beta}$ que maximiza a log-verossimilhança

- Vantagens: escolha de perda e parametrização é automática; permite inferência assintótica

Exemplo: regressão logística

- Suponha que $Y \sim \text{Bern}(p)$, de modo que $\mathbb{P}[Y = y] = (p)^y(1-p)^{1-y}$, e

$$L(y, p) = -\left(y \log(p) + (1-y)\log(1-p)\right) = -\left(y \log(p/(1-p)) + \log(1-p)\right)$$

- Daí, escolhemos $p(x)$ via

$$\log\left(\frac{p(x)}{1-p(x)}\right) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$$

- Finalmente, estimamos os coeficientes via

$$\begin{aligned}(\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p) &= \underset{\tilde{\beta}_0,\ldots,\tilde{\beta}_p}{\text{argmin}} - \left(\frac{1}{n}\sum_{i=1}^{n} y_i \log\frac{\hat{p}(x_i)}{1-\hat{p}(x_i)} + \log(1-\hat{p}(x_i))\right) \\ &= \underset{\tilde{\beta}_0,\ldots,\tilde{\beta}_p}{\text{argmin}} - \left(\frac{1}{n}\sum_{i=1}^{n} y_i(\hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \ldots + \hat{\beta}_p x_{ip}) - \log\left(\frac{1}{1+e^{\hat{\beta}_0+\hat{\beta}_1 x_{i1}+\ldots+\hat{\beta}_p x_{ip}}}\right)\right)\end{aligned}$$

- O problema é convexo, então basta usar algum método de gradiente para resolvê-lo

Outros classificadores

- Métodos discriminativos: estimam $\mathbb{P}[Y|X = x]$ diretamente
  - Regressão logística (e GLMs em geral)

- Métodos generativos: estimam $\mathbb{P}[Y|X = x]$ via $\mathbb{P}[X|Y = k]$ e $\mathbb{P}[Y]$
  - LDA: assume que $X|Y = k \sim N(\mu_k, \Sigma)$; fronteira de decisão linear
  - QDA: assume que $X|Y = k \sim N(\mu_k, \Sigma_k)$; fronteira de decisão quadrática
  - Naive Bayes: assume que $\mathbb{P}[X|Y = k] = \prod_{j=1}^{p} \mathbb{P}[X_j|Y = k]$; fronteira arbitrária

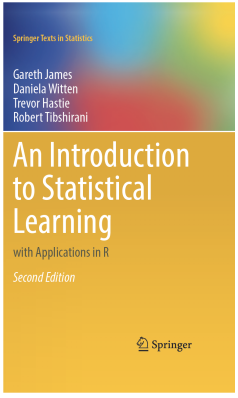
- Qual é melhor? Se hipóteses generativas são razoáveis, então são melhores; senão, métodos discriminativos. De modo geral, hipóteses generativas são mais fortes

Avaliação de classificadores

- Em geral, para métodos de classificação, usamos a perda 0-1: $\frac{1}{n}\sum_{i=1}^{n}\mathbb{I}_{[\hat{y}_i \neq y_i]}$
- Ela não é bem informativa: será que os erros são balanceados entre classes?
- Uma descrição melhor é a matriz de confusão:

		Actual class	
		Negative	Positive
Predicted class	Negative	True Negative (TN)	False Negative (FN)
	Positive	False Positive (FP)	True Positive (TP)

- Mudando o threshold t na classificação $\mathbb{I}_{[\hat{\mathbb{P}}[Y=1|X]>t]}$, mudamos os erros cometidos
- Uma forma agregada de ver o classificador: curva ROC e AUC



Capítulo 5: Métodos de reamostragem

Créditos de figuras e slides: James, Witten, Hastie e Tibshirani

1. Estimando performance futura

- Para ter uma noção realista do sucesso de um método, precisamos estimar performance futura
- Isso pode ser feito via o erro de teste, *i.e.,* num conjunto de dados intocados até então.
  - É a última etapa do processo
  - Regra de ouro: conjunto de teste só pode ser usado depois do modelo ser escolhido; senão está roubando (acontece muito)
- Filosofia:
  - Modelo é treinado nos dados de treino; erro de treino por si só não é informativo sobre performance futura
  - Se houver hiperparâmetro (parâmetro não escolhido no treino), usamos conjunto de validação
  - Depois do modelo ser escolhido, retreinamos em treino+validação, avaliamos erro de teste
  - Na hora de entregar o modelo, retreinamos em treino+validação+teste, e usamos o erro de teste como estimativa de performance futura

Como implementar essa filosofia

- Conjunto de validação: dados particionados em treino e validação
  - Vantagens: simples de implementar, e simples computacionalmente
  - Desvantagens: estimativas têm alta variância, dados não são reutilizados, erros sobrestimados
- LOOCV: média de estimativa em n folds; cada um tirando o ponto x_i, $i = 1, \ldots, n$
  - Vantagens: melhor uso dos dados, estimativas mais precisas
  - Desvantagens: computacionalmente muito intensivo, média de estimativas correlacionadas
- 5 ou 10CV: média de estimativa em 5 ou 10 folds; cada um tirando o ponto x_i, $i = 1, \ldots, n$
  - É um meio-termo, e muito popular
- Regra de 1 desvio-padrão para promover parsimônia
- Vale sempre? Não: dependência nos dados (*e.g.*, séries temporais e rolling + block CV)

2. Estimando incerteza

- Em casos particulares, desenvolvemos teoria de distribuição (Normal, t, χ^2, F)
- Mas de maneira mais ampla essa teoria não está disponível. Como fazer de forma não-paramétrica?
- Bootstrap: usar o próprio dataset para simular mais datasets via reamostragem com repetição;
- Reamostramos até ter as mesmas n amostras para garantir amostras de mesmo tamanho
- A partir daí, obtemos a distribuição de bootstrap, e estimamos quaisquer quantidades inferenciais
- Por que isso deveria funcionar? Estamos substituindo a distribuição populacional F pela empírica $\hat{F}$, e ao invés de amostrar de F amostramos de $\hat{F}$. Há razões teóricas para dar certo
- Vale sempre? Não, mas são casos relativamente patológicos