UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

Tópicos Especiais em Fundamentos de Computação: Quando os modelos encontram os dados

Alunos: André Riccioppo, Gabriel M. Almeida e Hudson Romualdo

Goiânia-GO

2024

Sumário

1 Introdução

Este relatório é um entregável parte da avaliação da disciplina de Tópicos Especiais em Fundamentos de Computação, do programa de pós-graduação do Instituto de Informática da Faculdade Federal de Goiás (INF/UFG). Os tópicos apresentados nas próximas seções são baseadas no capítulo 8 "When Models Meet Data"do livro texto (DEISENROTH; FAISAL; ONG, 2020), base da disciplina, além de referências destacadas e listadas no corpo e no fim deste documento.

O capítulo 8 é o primeiro da segunda parte do livro intitulada "Problemas centrais de Aprendizado de Máquina". Nessa segunda parte é iniciada a discussão sobre aprendizado de máquina introduzindo quatro pilares:

- Regressão (Capítulo 9)
- Redução de Dimensionalidade (Capítulo 10)
- Estimação de Densidade (Capítulo 11)
- Classificação (Capítulo 12)

Nesse contexto, o capítulo 8 serve como preparação para esses conteúdos pois aborda os três principais componentes de qualquer sistema de aprendizado de máquina: dados, modelos e aprendizado.

O restante deste trabalho está distribuído da seguinte forma. O Capítulo 2 descreve as ideias de treinamento e previsão no contexto de "algoritmos de aprendizado de máquina". No Capítulo 3 apresentaremos a estrutura empírica de minimização de risco. O capítulo 4 apresenta o princípio da máxima probabilidade. O capítulo 5 demonstra a ideia de modelos probabilísticos. No capítulo 6 é apresentada uma linguagem gráfica para especificar modelos probabilísticos e é discutida a seleção de modelos.

2 Dados, modelos e aprendizagem

Existem três conceitos que estão no núcleo do aprendizado de máquina: Dados, Modelos e Aprendizagem.

Uma das competências que a Inteligência Artificial busca reproduzir de nós humanos é a capacidade de aprender. Considerando que nós aprendemos com nossas experiências de vida, como fazer com que uma máquina aprenda se ela não pode passar por essas experiências? A resposta é: através de **dados**. O aprendizado de máquina é inerentemente orientado a dados e seu objetivo é desenvolver metodologias para extrair padrões valiosos desses dados.

Modelos são processos que geram dados relacionadas aos dados utilizados ao longo do treinamento em seu aprendizado.Um determinado modelo aprende a partir de dados se a sua performance melhora ao processar tais dados. Uma vez que o modelo apresenta bons resultados mesmo com dados ainda não vistos, pode-se dizer que o modelo aprendeu.

Aprendizado pode ser entendido como o processo de encontrar automaticamente padrões e estruturas nos dados, otimizando os parâmetros do modelo.

1 Dados como Vetores

A forma mais comum de se obter dados é de forma tabular onde cada linha representa uma determinada instancia ou exemplo do conjunto de dados e as colunas representam uma determinada característica desse conjunto, conforme pode ser observado na tabela 1.

Name	Gender	Degree	Postcode	Age	Annual salary
Aditya	M	MSc	W21BG	36	89563
Bob	M	PhD	EC1A1BA	47	123543
Chloé	F	BEcon	SW1A1BH	26	23989
Daisuke	M	BSc	SE207AT	68	138769
Elisabeth	F	MBA	SE10AA	33	113888

Tabela 1 – Dados de recursos humanos que não estão em um formato numérico

Entretanto para fins de treinamento os dados precisam ter uma representação numérica. Nesse sentido, é necessário fazer escolhas sobre quais características são interessantes para o modelo e, uma vez considerando um campo não numérico como importante, como representa-lo numericamente.

A coluna "Gender" por exemplo, pode ser mapeada para valores 0 e 1, sendo 0 masculino e 1 feminino. Por vezes, é importante ter conhecimento do domínio dos dados para poder fazer essa conversão, é o que podemos observar na coluna "Degree". Os graus de instrução presentes

nessa coluna podem ser representados em 3 grupos, inciando na graduação com o valor 1, em seguida o mestrado com valor 2, finalizando no doutorado com o valor 3. Outro exemplo é a coluna "Postcode" que armazena o código postal de determinada pessoa. Trata-se deu um dado referente ao endereço e, portanto, pode ser transformado em duas outras características, Latitude e Longitude, conforme tabela 2.

Gender ID	Degree	Latitude (in degrees)	Longitude (in degrees)	Age	Annual Salary (in thousands)
-1	2	51.5073	0.1290	36	89.563
-1	3	51.5074	0.1275	47	123.543
+1	1	51.5071	0.1278	26	23.989
-1	1	51.5075	0.1281	68	138.769
+1	2	51.5074	0.1278	33	113.888

Tabela 2 – Dados de recursos humanos que estão em um formato numérico

Mesmo colunas numéricas e que, portanto, podem ser diretamente lidas por um algoritmo de aprendizado de máquina requerem considerações quanto as suas unidades, escalas e restrições. Deve-ser normalizar os dados considerando a média e o desvio padrão do conjunto de dados ou de algumas colunas, separadamente.

Uma vez que temos dados (*dataset*) representados em um formato numérico, podemos então partir para o aprendizado de máquina. Essencialmente, cada *datapoint* ou seja, cada linha, da tabela apresentada na figura 2 é representada como um vetor de números reais. Observando a orientação da tabela inicialmente o vetor pode ser representado na forma de uma linha. Entretanto, para boa parte dos algoritmos de aprendizado de máquina, o vetor que representa o *datapoint* é representado como um vetor coluna.

Para representar o número de exemplos no conjunto de dados, vamos utilizar a letra *N*. Nesse caso, representamos os índices dos exemplos da seguinte forma: $n = 1, ..., N$. Cada exemplo é um vetor e, portanto, é representado por uma letra em minúsculo e em negrito: $\boldsymbol{x}_n$. Cada coluna representa uma característica particular de interesse sobre o exemplo, e indexamos as características como $d = 1, ..., D$.

Vamos considerar o problema de prever o salário anual a partir da idade, baseado nos dados na figura 2. Para prever o salário anual a partir da idade podemos considerar cada vetor $\boldsymbol{y}_n$ representando o salário associado com cada o vetor $\boldsymbol{x}_n$ representando a idade. Nesse exemplo, é comum nomear o vetor $\boldsymbol{y}_n$ de *target* e o vetor $\boldsymbol{x}_n$ de *input*. Sendo então o *dataset* um conjunto de pares de *input* e *target* pode ser representado da seguinte forma: $(\boldsymbol{x}_1, \boldsymbol{y}_1), ..., (\boldsymbol{x}_n, \boldsymbol{y}_n), ..., (\boldsymbol{x}_N, \boldsymbol{y}_N)$. Todo o conjunto de *datapoints* $\boldsymbol{x}_n, ..., \boldsymbol{x}_N$ pode ser representado dessa forma $\boldsymbol{X} \in \mathbb{R}^{N \times D}$. A figura 1 ilustra o *dataset* utilizado nesse exemplo consistindo das duas últimas colunas da tabela representada na figura 2. Estamos interessados no salário de uma pessoa de sessenta anos ($x = 60$) ilustrado como uma linha vermelha tracejada vertical, que não faz parte dos dados de treinamento.

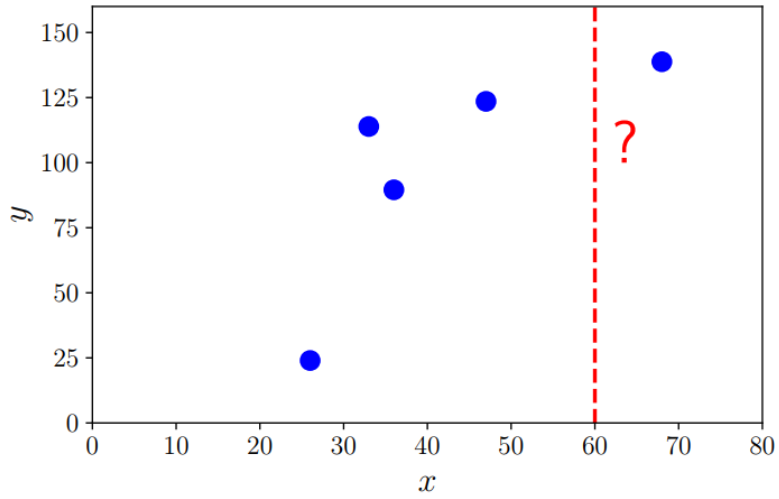

Figura 1 – Representação gráfica da relação entre os *datapoints*

Uma vez que os *datapoins* são representados por vetores, podemos utilizar conceitos de álgebra linear para representar o problema de aprendizado de máquina e definir o nosso **modelo** preditor. Duas abordagens para representar modelos são apresentadas neste livro: o preditor como uma função, e um preditor como uma modelo probabilístico.

2 Modelos como funções

Um preditor é uma função que, quando ao receber uma determinada entrada (no nosso caso, um vetor de características), produz um saída. De forma simplificada, considere a saída como um único número, ou seja, um escalar de valor real. Isso pode ser escrito como: $f : \mathbb{R}^D \rightarrow \mathbb{R}$.

Ou seja, uma função f tal que o conjunto $\mathbb{R}^D$ representando a entrada é mapeado para um número escalar $\mathbb{R}$ que representa a saída. A Figura 2 ilustra uma possível função que pode ser usada para calcular o valor da previsão para valores de entrada x.

3 Modelos como distribuições de probabilidade

Em vez de considerar um preditor como uma única função, podemos considerar os preditores como modelos probabilísticos, ou seja, modelos que descrevem a distribuição das possíveis funções.

A figura 3 combina uma visualização da linha de regressão com uma representação da incerteza ou variabilidade das previsões. Isso fornece uma compreensão mais completa não apenas da tendência central observada nos dados, mas também da confiança e da faixa de possíveis resultados para novas observações, como a previsão do salário para uma idade específica, neste caso, 60 anos.

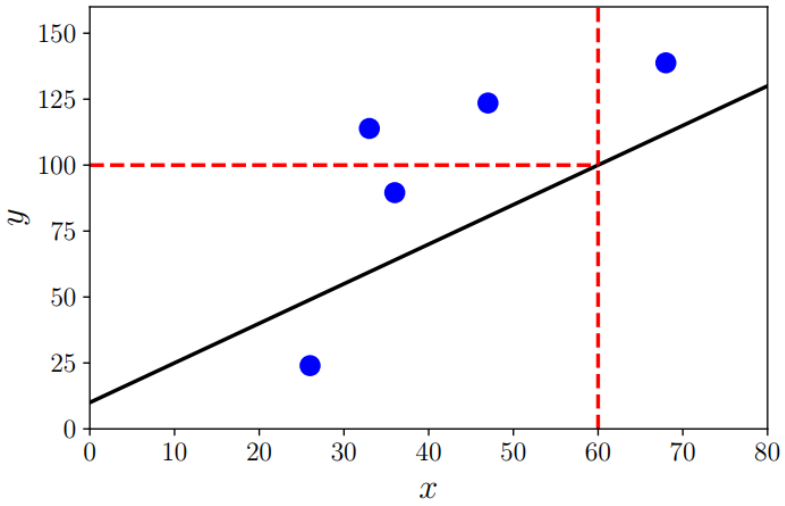

Figura 2 – A função (diagonal sólida preta) e a sua previsão em $x = 60$, ou seja, $f(60) = 100$

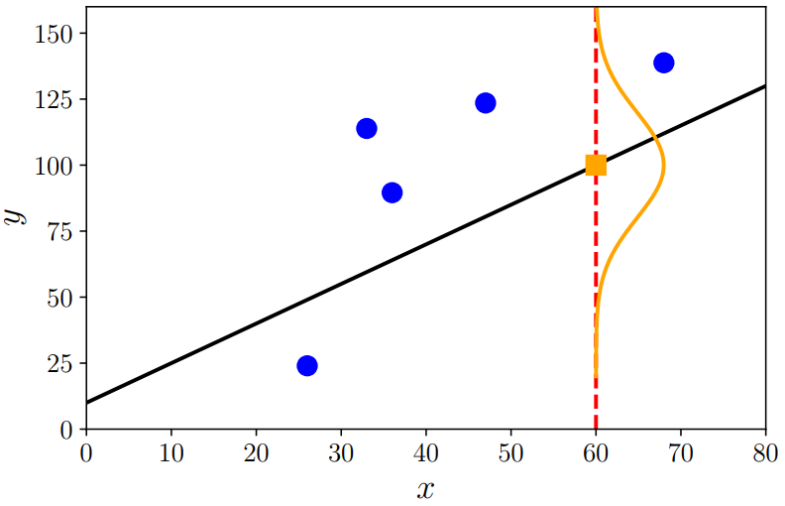

Figura 3 – Função (diagonal sólida preta) e sua incerteza preditiva em $x = 60$ (representada como uma Gaussiana)

4 Aprender é encontrar parâmetros

O objetivo da aprendizagem é encontrar um modelo e seus parâmetros correspondentes de modo que o preditor resultante tenha um bom desempenho em dados não vistos. Existem, conceitualmente, três fases algorítmicas distintas ao discutir algoritmos de aprendizagem de máquina:

- Predição ou inferência;
- Treinamento ou estimativa de parâmetros;
- Ajuste de hiperparâmetros ou seleção de modelo.

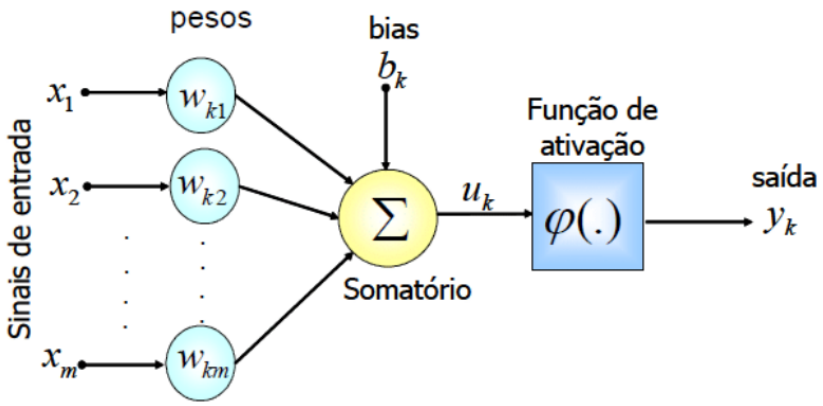

Figura 4 – Modelo de um neurônio artificial. (SOARES; SILVA, 2011)

A fase de predição ocorre quando usamos um preditor treinado em dados de teste nunca antes vistos. Em outras palavras, os parâmetros e a escolha do modelo já estão fixos e o preditor é aplicado a novos vetores que representam novos pontos de dados de entrada.

A fase de treinamento ou estimativa de parâmetros é quando ajustamos nosso modelo preditivo com base nos dados de treinamento. Gostaríamos de encontrar bons preditores com base nos dados de treinamento, e há duas estratégias principais para fazer isso: encontrar o melhor preditor com base em alguma medida de qualidade (às vezes chamada de encontrar uma estimativa pontual) ou usar inferência Bayesiana. Usamos métodos numéricos para encontrar bons parâmetros que "se ajustem" aos dados, e a maioria dos métodos de treinamento podem ser considerados abordagens de escalada (*hill-climbing*) para encontrar o máximo de um objetivo, por exemplo, o máximo de uma probabilidade.

Frequentemente, precisamos tomar decisões de modelagem de alto nível sobre a estrutura do preditor, como o número de componentes a serem usados ou a classe de distribuições de probabilidade a serem consideradas. A escolha do número de componentes é um exemplo de hiperparâmetro, e esta escolha pode afetar significativamente o desempenho do modelo. O problema de escolher entre diferentes modelos é chamado de seleção de modelos.

A distinção entre parâmetros e hiperparâmetros é um tanto arbitrária e é principalmente motivada pela distinção entre o que pode ser numericamente otimizado versus o que precisa usar técnicas de pesquisa.

Para exemplificar, a figura 4 representa uma rede neural artificial, modelo escolhido para determinada tarefa de predição. Por hiperparâmetros podemos entender como a escolha sobre a quantidade de neurônios e de camadas de neurônios da rede neural. Essa escolha não é realizada de forma aleatório mas existe certo nível de empirismo. Já os parâmetros estão relacionados aos pesos de cada neurônio, ajustados ao longo do treinamento verificando-se, por exemplo, a distância entre um sinal de entrada $x_1, x_2, ..., x_m$ cuja saída y_m já é conhecida e, a predição y_k realizada pela rede neural.

3 Minimização de Risco Empírico

Já se perguntou como modelos de aprendizado de máquina realizam predições? Como os modelos "aprendem"com dados?

A parte de “aprendizado” do aprendizado de máquina se resume a estimar parâmetros com base em dados de treinamento. O ERM (Empirical Risk Minimization), é um princípio que guia o processo de aprendizado de um modelo de aprendizado de máquina de forma análoga ao de uma bússola que aponta para a direção certa.

A direção certa no contexto do aprendizado quer dizer: minimizar a diferença entre os valores previstos pelo modelo e os valores reais, obtidos dos dados de treinamento. À medida que essa diferença se torna cada vez menor, mais precisas são as previsões do modelo preditor que está sendo treinado.

Para clarificar esse entendimento, vamos lançar mão dos princípios gerais da minimização de risco empírico que nos permitem questionar o que é aprendizado sem construir explicitamente modelos probabilísticos. Existem quatro escolhas principais de design que estão associadas às perguntas abaixo e que serão apresentadas detalhadamente nas subseções a seguir:

1 - Qual é o conjunto de funções que permitimos que o preditor assuma?

2 - Como medimos o desempenho do preditor nos dados de treinamento?

3 - Como construímos preditores apenas a partir de dados de treinamento que têm bom desempenho em dados de teste não vistos?

4 - Qual é o procedimento de busca no espaço dos modelos?

1 Classe de hipóteses de funções

Classes de hipóteses de funções referem-se ao conjunto de todas as funções possíveis que um modelo pode aprender para mapear entradas para saídas, dadas uma arquitetura e um conjunto de parâmetros. Em outras palavras, é o conjunto de todas as possíveis funções h que o modelo pode selecionar durante o processo de treinamento.

Exemplos de Classes de Hipóteses

- Regressão Linear: a classe de hipóteses inclui todas as funções lineares.

- Árvores de Decisão: a classe de hipóteses inclui todas as possíveis árvores de decisão que podem ser construídas com um determinado conjunto de características e critérios de divisão.
- Redes Neurais: a classe de hipóteses inclui todas as funções que podem ser representadas pela arquitetura da rede neural e seus parâmetros.
- Máquinas de Vetores de Suporte (SVM): a classe de hipóteses inclui todas as funções que podem ser representadas por hiperplanos de decisão no espaço de características, com diferentes margens e configurações de kernel.

A escolha da classe de hipóteses é crucial, pois determina a capacidade do modelo de aprender diferentes tipos de padrões nos dados de treinamento. Uma classe de hipóteses muito simples pode não ser capaz de capturar a complexidade dos dados (*underfitting*), enquanto uma classe de hipóteses muito complexa pode ajustar-se excessivamente aos dados de treinamento (*overfitting*).

2 Função de perda para treinamento

Funções de perda (ou de custo) são usadas para quantificar o erro de uma hipótese (função) em relação aos dados de treinamento. Elas medem a discrepância entre as previsões do modelo e os valores reais dos dados de treinamento. O objetivo do treinamento do modelo é minimizar essa função de perda, ou seja, a cada previsão minimizar a diferença entre o real e o previsto que foi empiricamente observado.

Exemplos de Funções de Perda

- Erro Quadrático Médio (*Mean Square Error, MSE*): $\ell(y, \hat{y}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$

  Penaliza grandes erros mais severamente do que pequenos erros e é utilizada principalmente em problemas de regressão.

- Erro Absoluto Médio (*Mean Absolute Error, MAE*): $\ell(y, \hat{y}) = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$

  Penaliza todos os erros de maneira linear e também é utilizada em problemas de regressão.

- Entropia Cruzada (*Cross-Entropy Loss*)

  Utilizada em problemas de classificação.

- Hinge Loss

  Usada em Máquinas de Vetores de Suporte (SVM).

Definida da classe de hipóteses de funções que compõe o preditor e a função de perda para treinamento é possível definir a equação de risco empírico:

$R_{\text{emp}}(f, X, y) = \frac{1}{N} \sum_{n=1}^{N} \ell(y_n, \hat{y}_n)$,

onde:

- f é o preditor;
- X é a matriz de exemplos;
- y é o vetor de rótulos;
- $\ell(y_n, \hat{y}_n)$ é a função de perda;
- N é o número de exemplos;
- $\hat{y}_n = f(x_n, \theta)$ é a predição do modelo para o exemplo x_n.

Até então falamos sobre como compor um preditor selecionando a classe de hipóteses adequada e como realizar o treinamento desse preditor utilizando funções de perda para medir a discrepância entre o previsto $\hat{y}$ com base em X e o real y.

Mas nosso interesse em um preditor não é que ele performe bem nos dados de treinamento. Ao invés disso, o objetivo é que sua performance seja satisfatório (baixo risco esperado) em dados que ele não conhece, dados de teste. Duas questões surgem desse desejo de obter esse risco baixo, comumente chamado de generalização, em um preditor:

- Como devemos alterar nosso procedimento de treinamento para generalizar bem?
- Como estimamos o risco esperado a partir de dados (finitos)?

Abordaremos essas questões nas seções a seguir.

3 Regularização para reduzir *Overfitting*

Ao realizar o treinamento de um preditor, na prática, temos apenas um conjunto finito de dados e, portanto, dividimos nossos dados em um conjunto de treinamento e um conjunto de teste. O conjunto de treinamento é usado para ajustar o preditor, e o conjunto de teste (não visto pelo algoritmo de aprendizado de máquina durante o treinamento) é usado para avaliar o desempenho do preditor, ou seja, sua capacidade de generalização. O *Overfitting* ocorre quando o preditor ajusta-se demais aos dados de treinamento e não generaliza bem para novos dados. Em outras palavras, temos um sobreajuste (*overfitting*) quando a média de discrepância entre o previsto e o real (risco empírico) é pequena no conjunto de treinamento, mas uma grande

média é observada no conjunto de teste. Isso tende a ocorrer quando temos poucos dados para treinamento ou uma classe de hipótese complexa.

Portanto, precisamos de alguma forma enviesar a busca pelo minimizador do risco empírico, introduzindo um termo de penalidade, o que torna mais difícil para o otimizador retornar um preditor excessivamente ajustado aos dados de treinamento. No aprendizado de máquina, o termo de penalidade é referido como regularização e é aplicado adicionando-se uma penalidade á função de perda do modelo.

Um tipo largamente utilizado de regularização é a L2 (*Ridge Regression*) que adiciona uma penalidade proporcional ao quadrado dos coeficientes do modelo modificando a função de perda conforme a seguir:

$$\ell_{\text{reg}}(y, \hat{y}, \theta) = \ell(y, \hat{y}) + \lambda \|\theta\|_2^2,$$

onde:

- $\ell(y, \hat{y})$ é a função de perda original,
- λ é o hiperparâmetro de regularização,
- $\|\theta\|_2^2$ é a norma $L2$ (quadrado da norma Euclidiana) dos parâmetros do modelo θ.

4 Validação cruzada para avaliar o desempenho da generalização

Conforme mencionado anteriormente, medimos o erro de generalização estimando-o aplicando o preditor em dados de teste. Estes dados são também, às vezes, referidos como o conjunto de validação. O conjunto de validação é um subconjunto dos dados de treinamento disponíveis que reservamos.

Diferentemente da divisão dos dados em conjuntos de treinamento e teste, a Validação Cruzada é uma técnica usada para avaliar o desempenho de um modelo de aprendizado de máquina dividindo os dados em conjuntos em múltiplas divisões.

Um tipo de Validação Cruzada largamente utilizada é a *K-Fold Cross-Validation* cujos passos para implementação são:

- Divisão dos dados

  Os dados são divididos aleatoriamente em K subconjuntos (ou "*folds*") de tamanhos aproximadamente iguais. Se $K = 5$, por exemplo, os dados serão divididos em 5 partes.

- Treinamento e validação

O processo de treinamento e validação é repetido K vezes. Em cada iteração, o $k-$ésimo *fold* é utilizado como conjunto de validação e os outros $k-1$ *folds* são usados para treinamento.

- Cálculo da métrica de desempenho

  A cada iteração a métrica de desempenho (por exemplo, erro quadrático médio, etc.) é calculada usando o conjunto de validação e armazenada.

- Média das métricas

  Após as K iterações, calcula-se a média das métricas de desempenho obtidas em cada iteração. Esta média fornece uma estimativa da capacidade de generalização do modelo.

O janelamento dos *folds* de treinamento e validação pode ser melhor observado na figura 5. Neste exemplo, o conjunto de dados foi dividido em $K = 5$ partes, sendo $K - 1$ utilizadas como conjunto de treinamento e uma como conjunto de validação. A cada iteração um dos *folds* figura como conjunto de validação. Na iteração seguinte esse mesmo *fold* compõe o conjunto de treinamento.

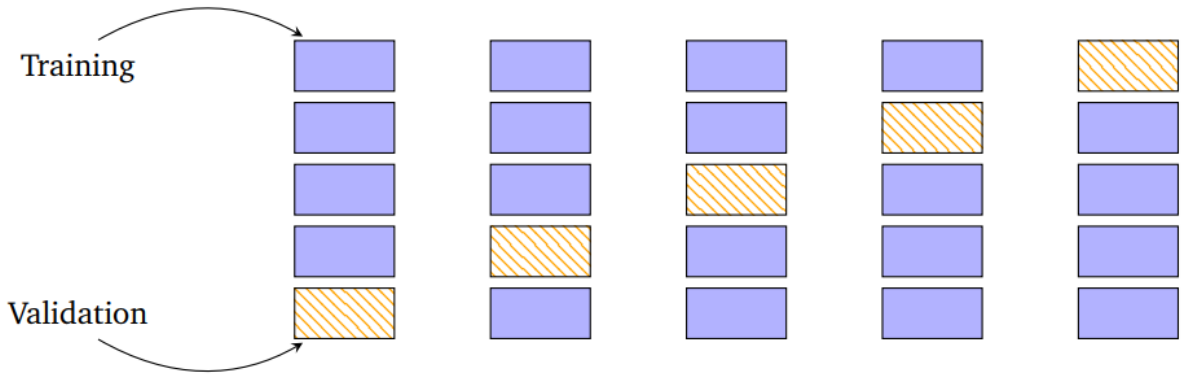

Figura 5 – Validação cruzada *K-Fold*. *Folds* de treinamento (azul) e *fold* de validação (laranja listrado).

A eficiência no treinamento em função da utilização dos dados para treinamento e validação em diferentes iterações garantindo que cada exemplo do conjunto de dados seja utilizado nos dois cenários é evidente. Entretanto, tal abordagem também contribui com a redução do viés que pode ser introduzido com a divisão simples entre treinamento e teste além de contribuir com uma estimativa mais robusta da capacidade de generalização do modelo.

Entretanto, a utilização da técnica *k-Fold* possui desvantagens. Além da complexidade de implementação, o custo computacional para a pode ser elevado, especialmente para grandes conjuntos de dados e modelos complexos pois o modelo será treinado K vezes.

4 Estimativa de parâmetros

Na seção anterior o problema não foi modelado explicitamente usando distribuições de probabilidade. Aqui veremos como usar essas distribuições de probabilidade para modelar a incerteza decorrente do processo de observação e a incerteza quanto aos parâmetros dos preditores.

Será introduzida a verossimilhança, que é análoga ao conceito de funções de perda na minimização de risco empírico, apresentado anteriormente. Já o conceito de prior ou distribuição a priori é análogo ao conceito de regularização, também apresentado na seção anterior.

1 Estimação de Máxima Verossimilhança

A ideia por trás da estimação de máxima verossimilhança (EMV) é definir uma função a partir dos parâmetros que permita encontrar um modelo que se adeque bem aos dados. Esse problema é focado na função de verossimilhança, ou, de maneira mais precisa, no seu logaritmo negativo. Esse sinal negativo decorre da convenção que desejamos maximizar a verossimilhança, mas a literatura da otimização numérica que é aplicada aqui, tende a estudar a minimização de funções.

Para dados representados por uma variável randômica x e para uma família de densidades de probabilidade $p(x|\theta)$ parametrizadas por θ, a log-verossimilhança negativa é dada por $\mathcal{L}_x(\theta) = -\log p(x|\theta)$.

A notação $\mathcal{L}_x(\theta)$ enfatiza o fato que o parâmetro θ está variando e o dado x está fixo. É comum descartar a referência a x e escrever a log-verossimilhança negativa como uma função apenas de θ, $\mathcal{L}(\theta)$ quando a variável randômica que representa a incerteza nos dados está livre do contexto.

Vamos interpretar qual é a densidade de probabilidade $p(x|\theta)$ através da modelagem usando um valor fixo de θ. Ela é uma distribuição que modela a incerteza dos dados para uma dada configuração de parâmetros. Para um determinado conjunto de dados x, a verossimilhança permite expressar preferências sobre configurações diferentes dos parâmetros θ e podemos escolher as configurações que mais "provavelmente" geraram os dados.

Em uma visão complementar, se considerarmos os dados como fixos (porque foram observados) e variarmos os parâmetros θ, o que o $\mathcal{L}(\theta)$ nos diz? Diz o quão provável uma configuração específica de θ é para as observações x. De acordo com essa segunda visão, a estimação de verossimilhança máxima nos dá o parâmetro mais provável de θ para o conjunto de dados.

Considerando o cenário de aprendizado supervisionado, onde temos os pares (x_1, y_1), $\ldots, (x_N, y_N)$ com $x_n \in \mathbb{R}^D$ e os rótulos $y_n \in \mathbb{R}$, estamos interessados em construir um preditor que a partir de um vetor de características x_n produza uma predição y_n. Ou seja, especificamos a distribuição probabilística condicional dos rótulos a partir dos exemplos para aquele θ específico.

Um exemplo comumente utilizado é a especificação da probabilidade condicionada dos rótulos dada uma distribuição Gaussiana. Assumimos que podemos explicar a observação da incerteza através de um ruído Gaussiano independente com média 0, $\mathcal{E}_n \sim \mathcal{N}(0, \sigma^2)$. Também se assume que o modelo linear $x_n^\top \theta$ é usado para a predição. Dessa forma, a verossimilhança Gaussiana para cada par exemplo-rótulo (x_n, y_n) pode ser especificada como $p(y_n|x_n, \theta) = \mathcal{N}(y_n|x_n^\top \theta, \sigma^2)$. A Figura 3 mostra uma verossimilhança Gaussiana para um dado parâmetro θ.

Assumimos que no conjunto de exemplos $(x_1, y_1), \ldots, (x_N, y_N)$, eles são independentes e identicamente distribuídos.

A palavra independente implica que a probabilidade envolvendo todo o conjunto de dados ($\mathcal{Y} = \{y_1, \ldots, y_N\}$ e $\mathcal{X} = \{x_1, \ldots, x_N\}$) pode ser fatorizada em um produto de probabilidades de cada exemplo individual $p(\mathcal{Y}|\mathcal{X}, \theta) = \prod_{n=1}^{N} p(y_n|x_n, \theta)$, onde $p(y_n|x_n, \theta)$ é uma distribuição em particular. No exemplo acima, consideramos essa distribuição em particular como sendo uma distribuição Gaussiana.

A expressão identicamente distribuída significa que cada termo no produto desta fórmula faz parte da mesma distribuição e todos eles compartilham os mesmos parâmetros.

Sob o ponto de vista de otimização, é mais fácil calcular funções que podem ser decompostas em somas de funções mais simples. Dessa forma, em aprendizado de máquina, considerando que $\log(ab) = \log(a) + \log(b)$, temos que a log-verossimilhança negativa é dada pela fórmula $\mathcal{L}(\theta) = -\log p(\mathcal{Y}|\mathcal{X}, \theta) = -\sum_{n=1}^{N} \log p(y_n|x_n, \theta)$.

Apesar de ser tentador interpretar que θ é observado e fixo, essa interpretação está incorreta. A log-verossimilhança negativa $\mathcal{L}(\theta)$ é uma função de θ, portanto para achar um bom vetor de parâmetros θ que explique bem os dados $(x_1, y_1), ..., (x_n, y_n)$, temos que minimizar a log-verossimilhança negativa $\mathcal{L}(\theta)$ com respeito a θ.

Continuando o raciocínio do exemplo anterior, onde foi explicada a verossimilhança Gaussiana, a log-verossimilhança negativa pode ser reescrita como:

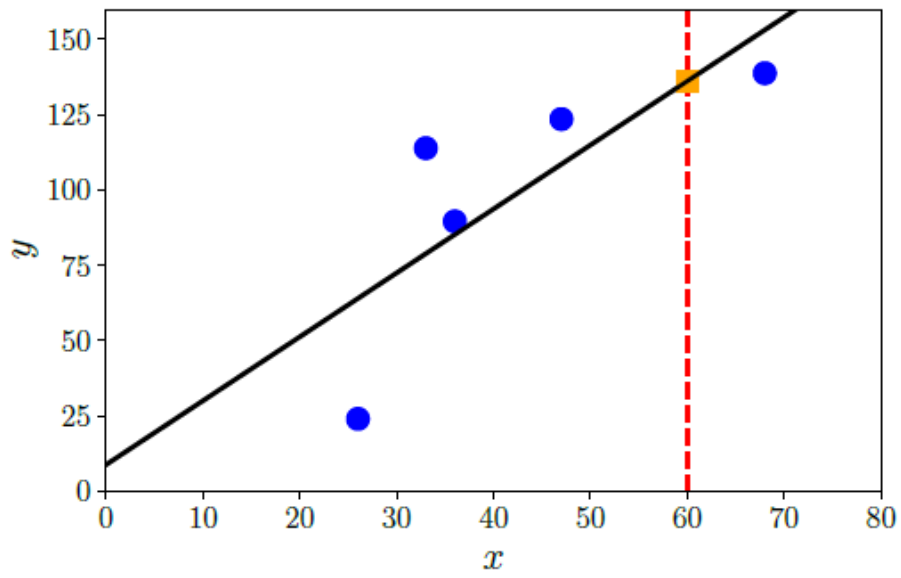

Figura 6 – Para os dados fornecidos, a estimação da máxima verossimilhança dos parâmetros resulta na linha diagonal preta. O quadrado laranja mostra o valor da predição da verossimilhança máxima em x = 60

$$\begin{aligned}
\mathcal{L}(\theta) &= -\sum_{n=1}^{N} \log p(y_n|x_n,\theta) = -\sum_{n=1}^{N} \log \mathcal{N}(y_n|x_n^\top\theta,\sigma^2) \\
&= -\sum_{n=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_n - x_n^\top\theta)^2}{2\sigma^2}\right) \\
&= -\sum_{n=1}^{N} \log \exp\left(-\frac{(y_n - x_n^\top\theta)^2}{2\sigma^2}\right) - \sum_{n=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma^2}} \\
&= \frac{1}{2\sigma^2}\sum_{n=1}^{N}(y_n - x_n^\top\theta)^2 - \sum_{n=1}^{N} \log \frac{1}{\sqrt{2\pi\sigma^2}}
\end{aligned} \tag{4.1}$$

Como σ é conhecido, o segundo termo da última equação é constante e minimizar $\mathcal{L}(\theta)$ corresponde a resolver o problema de mínimos quadrados expressado no primeiro termo.

Porém, o problema da otimização resultante para verossimilhanças Gaussianas que corresponde à estimação da máxima verossimilhança possui uma solução fechada, o que será tratado na apresentação que tratar de regressão linear. A Figura 6 mostra um conjunto de dados de regressão e uma função que é induzida por parâmetros de verossimilhança máxima.

2 Estimação de Máxima a Posteriori

Se tivermos conhecimento anterior sobre a distribuição dos parâmetros θ, podemos multiplicar um termo adicional à verossimilhança. Esse termo adicional é a distribuição de

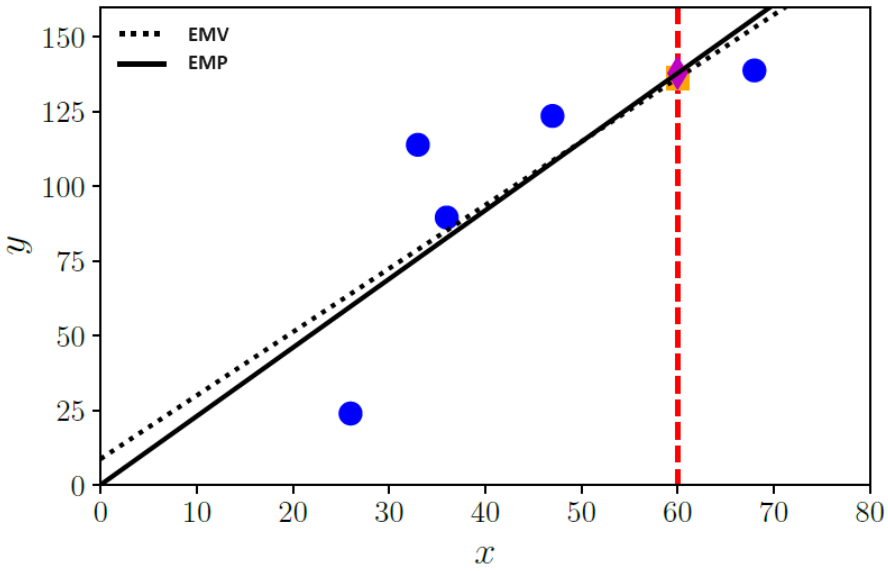

Figura 7 – Comparação das predições usando EMV e EMP em $x = 60$. A distribuição a priori adiciona um viés à inclinação da reta e leva a interseção para mais próximo de 0. Nesse exemplo, o viés que move a interseção para mais próximo de 0, aumenta a inclinação.

probabilidade a priori de parâmetros $p(\theta)$. Conhecendo esse prior, depois de observar alguns dados x, como atualizamos a distribuição de θ? Em outras palavras, como devemos representar o fato que temos um conhecimento mais específico de θ depois de observar os dados x? O teorema de Bayes, apresentado anteriormente no livro, traz uma ferramenta para atualizar a distribuição de probabilidades de variáveis aleatórias. Ele permite calcular a distribuição posterior $p(\theta|x)$ (o conhecimento mais específico) partindo da distribuição a priori $p(\theta)$ e da função de verossimilhança $p(x|\theta)$ que vincula os parâmetros θ e os dados observados x:

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)}$$

Estamos interessados em encontrar o parâmetro θ que maximiza essa distribuição posterior. Como a distribuição $p(x)$, que é a probabilidade marginal dos dados, não depende de θ, podemos ignorar o valor do denominador para a otimização e obter

$$p(\theta|x) \propto p(x|\theta)p(\theta)$$

Essa relação de proporção esconde a densidade dos dados $p(x)$, que pode ser difícil de estimar. Ao invés de estimar o mínimo da log-verossimilhança negativa, agora vamos estimar o mínimo da log-posterior negativa, que é chamada de estimativa da máxima a posteriori (EMP).

A Figura 7 mostra o efeito de adicionar uma distribuição a priori Gaussiana de média 0 no conjunto de dados mostrado na Figura 6.

Seguindo o exemplo que trata da distribuição Gaussiana, assumimos que o vetor de parâmetros é distribuído como uma distribuição Gaussiana multivariada de média 0. Como a distribuição a priori conjugada de uma distribuição Gaussiana também é uma distribuição Gaussiana, espera-se que a distribuição posterior também seja Gaussiana.

A ideia de incluir conhecimento a priori de onde estão os melhores parâmetros é amplamente utilizada em aprendizagem de máquina. Uma visão alternativa, demonstrada anteriormente, é a ideia de regularização, que introduz um termo adicional com um viés que aproxima os parâmetros à origem. A estimação de máxima a posteriori pode ser considerada uma ponte entre as abordagens probabilísticas e não probabilísticas, ao reconhecer a necessidade de uma distribuição a priori, mas apenas produzir um ponto estimado dos parâmetros.

Por fim, vale observar que a estimativa de máxima verossimilhança θ_{ML} possui algumas propriedades:

- consistência assintótica: a EMV converge para o valor verdadeiro no limite de observações infinitas, mais um erro aleatório que é aproximadamente normal.
- a quantidade de amostras necessárias para obter essas propriedades é bem grande.
- a variância do erro decresce em 1/N, sendo N o número de pontos de dados.
- em um regime de dados "pequeno", a EMV pode levar ao overfitting.

Tanto o princípio da estimação da máxima verossimilhança (EMV) como o da estimação da máxima a posteriori (EMP) usam a modelagem probabilística para raciocinar sobre a incerteza dos dados e parâmetros do modelo. Mas eles não usam modelagem probabilística em sua extensão total. Eles produzem um ponto estimado do preditor, ou seja, apenas um conjunto de valores de parâmetros que representa a melhor predição. Uma outra forma, apresentada na sequência, é que os valores de parâmetros também sejam tratados como variáveis aleatórias, ao invés de estimar os melhores valores da distribuição. Com isso, a distribuição completa dos parâmetros pode ser usada para fazer predições.

3 Ajuste do modelo

Considere que temos um conjunto de dados e estamos interessados em ajustar um modelo parametrizado a esses dados. Quando falamos de ajustar, geralmente nos referimos a otimizar ou aprender os parâmetros do modelo de forma que minimizem alguma função de perda (por exemplo, a log-verossimilhança negativa). Estimação de máxima verossimilhança e estimação de máxima a posteriori são dois algoritmos comumente usados para isso.

A parametrização do modelo define a classe de modelos M_θ com a qual podemos operar. Por exemplo, na definição de uma regressão linear, podemos definir que a relação entre as entradas

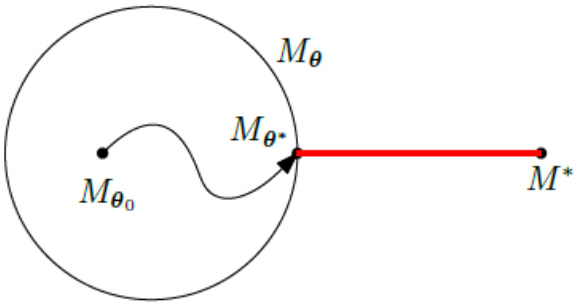

Figura 8 – Ajuste do Modelo. Em uma classe parametrizada M_θ de modelos, podemos otimizar os parâmetros θ do modelo para minimizar a distância até o modelo M^* verdadeiro, que é desconhecido.

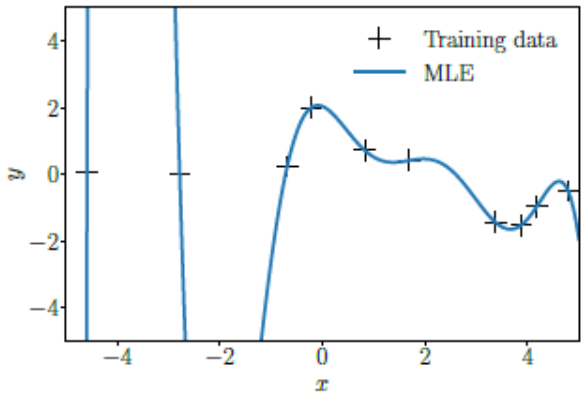

Figura 9 – Overfit

x e as observações y seja $y = ax + b$, onde $\theta := \{a, b\}$ são os parâmetros do modelo. Nesse caso, os parâmetros θ descrevem a família de funções afins. Assumindo que os dados venham de um modelo M^* desconhecido, dado um conjunto de dados de treinamento, otimizamos θ de forma que M_θ seja o mais próximo possível de M^*. Aqui proximidade é definida pela função objetiva que vamos otimizar (por exemplo, a perda ao quadrado).

A Figura 8 mostra uma configuração onde temos uma classe de modelos pequena (delimitada pelo círculo M_θ) e o modelo de geração dos dados M^* está fora do conjunto de modelos considerados. Depois da otimização, onde obtemos os melhores parâmetros θ^*, podemos distinguir três casos diferentes: overfit, underfit e o ajuste adequado.

De maneira aproximada, o overfit se refere à situação onde a classe de modelos parametrizada é muito rica para modelar o conjunto de dados gerado por M^*. Exemplo: o conjunto de dados foi gerado por uma função linear, mas M_θ foi definido pela classe de polinômios de sétima ordem. Geralmente modelos com overfit possuem um grande número de parâmetros. Isso causa problemas quando a predição se distanciar dos dados. A Figura 9 mostra um exemplo de overfit no contexto de uma regressão onde os parâmetros do modelo são aprendidos pelas médias da verossimilhança máxima.

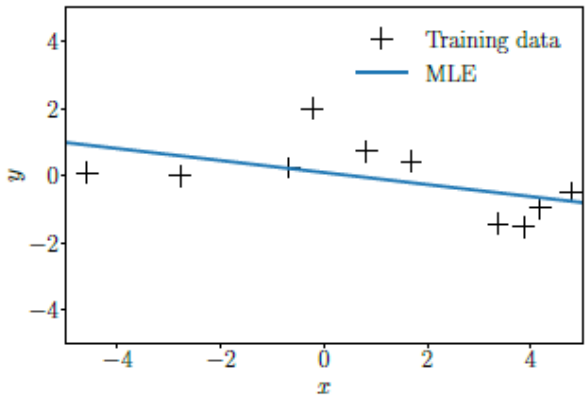

Figura 10 – Underfit

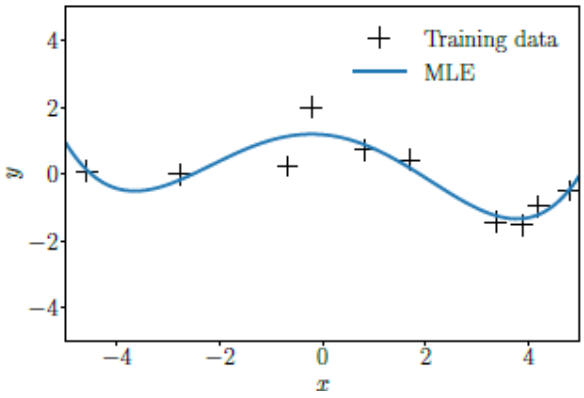

Figura 11 – Ajuste adequado

O underfit se refere à situação oposta, onde a classe de modelos M_θ não é rica o suficiente. Exemplo: o conjunto de dados foi gerado por uma função sinusoidal, mas θ parametriza linhas retas. Esses modelos geralmente possuem poucos parâmetros. Exemplo: o ajuste adequado se refere à situação onde a classe de modelos é correta e é rica o suficiente para descrever o conjunto de dados. A Figura 10 mostra um exemplo de um modelo que não foi flexível o suficiente.

No terceiro caso, demonstrado na Figura 11 a classe de modelos é adequada e o rica o suficiente para descrever o conjunto de dados.

Na prática, geralmente são definidos modelos de classes muito ricas, com vários parâmetros, como no caso das redes neurais profundas. Para mitigar o problema do overfitting, pode-se usar regularização ou distribuição a priori, por exemplo. Isso será mais discutido adiante.

5 Modelagem Probabilística e Inferência

Em aprendizado de máquina geralmente estamos preocupados com a interpretação e análise de dados, como por exemplo a predição de eventos futuros ou tomada de decisão. Para facilitar essa tarefa, geralmente construímos modelos que descrevem o processo generativo que gera os dados observados.

Como exemplo podemos descrever as saídas de um experimento de jogar uma moeda (cara ou coroa) em dois passos. No primeiro passo é definido o parâmetro μ, que descreve a probabilidade de "cara"como o parâmetro de uma distribuição de Bernoulli. No segundo passo obtemos a amostra de uma experiência $x \in \{\text{cara, coroa}\}$ da distribuição de Bernoulli $p(x|\mu) = \text{Ber}(\mu)$. O parâmetro μ dá origem a um conjunto específico de dados X e depende da moeda utilizada. Como não conhecemos μ anteriormente e não podemos observa-lo diretamente, precisamos mecanismos para aprender algo sobre μ, dadas observações obtidas de experimentos de jogar a moeda. A modelagem probabilística pode ser usada para isso.

1 Modelos Probabilísticos

Os modelos probabilísticos representam os aspectos incertos de um experimento como distribuições probabilísticas. O benefício de usar esses modelos probabilísticos é que eles oferecem um conjunto de ferramentas consistentes vindos da teoria da probabilidade para modelagem, inferência, predição e seleção de modelos.

Na modelagem probabilística, a distribuição conjunta (ou conjugada) das variáveis observadas x e dos parâmetros ocultos θ são de importância central e encapsulam as seguintes informações:

- A verossimilhança e a distribuição a priori.
- A verossimilhança marginal $p(x)$, que tem parte importante na seleção do modelo (como será demonstrado), pode ser calculada usando a distribuição conjunta e integrando seus parâmetros.
- A distribuição posterior, que pode ser obtida dividindo a distribuição conjunta pela verossimilhança marginal.

Como apenas a distribuição conjunta vai ter essas propriedades, o modelo probabilístico é especificado pela distribuição conjunta de todas suas variáveis aleatórias.

2 Inferência Bayesiana

Como já explicado, uma das principais tarefas em aprendizado de máquina é descobrir o valor das variáveis ocultas θ associadas ao modelo e aos dados a partir das variáveis observadas x. Usando a estimativa de verossimilhança máxima ou estimativa da máxima a posteriori, o objetivo do algoritmo de estimação de parâmetros é resolver um problema de otimização e obter um valor único que seria o melhor θ. A partir dessa estimação de θ^*, ela será usada para realizar as predições. De maneira mais específica, a distribuição preditiva será dada por $p(x|\theta^*)$, onde o θ^* é usado na função de verossimilhança.

Porém, ao focar apenas em algumas estatísticas da distribuição posterior, acabamos perdendo informação que pode ser crítica em um sistema que utilize a predição $p(x|\theta^*)$ para tomar decisões, sistemas esses que podem usar funções objetivas diferentes. Então, trabalhar com a distribuição posterior completa pode ser útil e levar a decisões mais robustas. A inferência Bayesiana trata de encontrar essa distribuição posterior.

A partir de um conjunto de dados $\mathcal{X}$, da distribuição a priori $p(\theta)$ e uma função de verossimilhança $p(\mathcal{X}|\theta)$, obtemos a distribuição posterior aplicando o teorema de Bayes:

$$p(\theta|\mathcal{X}) = \frac{p(\mathcal{X}|\theta)p(\theta)}{p(\mathcal{X})}, \qquad p(\mathcal{X}) = \int p(\mathcal{X}|\theta)p(\theta)d(\theta)$$

A ideia é explorar o teorema de Bayes para inverter a relação entre os parâmetro θ e os dados $\mathcal{X}$ (dados pela verossimilhança) e obter a distribuição posterior $p(\theta|\mathcal{X})$.

A implicação de ter uma distribuição posterior nos parâmetros é que ela pode ser usada para propagar a incerteza dos parâmetros aos dados. Nesse caso, com uma probabilidade $p(\theta)$ a predição seria $p(x) = \int p(x|\theta)p(\theta)d(\theta) = \mathbb{E}_\theta[p(x|\theta)]$ e não dependeria mais dos parâmetros θ, que foram marginalizados ou integrados. A equação mostra que a predição é uma média sobre todos os valores plausíveis do parâmetro θ, onde a plausibilidade é encapsulada pela probabilidade $p(\theta)$.

Conhecendo, então, a estimação de parâmetros e a inferência Bayesiana, podemos comparar essas duas abordagens de aprendizado:

- A estimação de parâmetros por EMV ou EMP fornece um ponto estimado θ^* dos parâmetros e o problema computacional a ser resolvido é uma otimização. Já a inferência Bayesiana fornece uma distribuição posterior e o problema computacional a ser resolvido é uma integração.
- Predições com pontos estimados são diretas, enquanto predições no framework Bayesiano exigem a resolução de outro problema de integração.
- A inferência Bayesiana permite incorporar conhecimento anterior, o que não é feito facilmente no contexto de estimação de parâmetros.

- A propagação da incerteza dos parâmetros para a predição pode ser importante para os sistemas de tomada de decisão, ao avaliar e explorar os riscos presentes nesse contexto.

Na inferência Bayesiana existem alguns desafios que surgem pela utilização de problemas de integração. Mais especificamente, se não forem escolhidos um prior conjugado nos parâmetros, as integrais a serem resolvidas não são tratáveis de forma analítica e não será possível calcular a distribuição posterior, as predições ou a verossimilhança máxima de uma forma fechada. Nesses casos, podem ser usadas aproximações estocásticas (como método de Monte Carlo via Cadeias de Markov) ou aproximações determinísticas (como aproximações de Laplace, inferência variacional ou propagação de expectativas).

Mesmo com esses desafios, a inferência Bayesiana tem sido aplicada a uma grande variedade de problemas, que incluem: modelagem de tópicos em grande escala, predição de cliques, aprendizado por reforço em sistemas de controle, sistemas de ranqueamento online e sistemas de recomendação em larga escala.

Existem ferramentas genéricas, como a otimização Bayesiana que são muito úteis para uma busca eficiente de meta parâmetros de modelos ou algoritmos.

3 Modelos de Variáveis Latentes

Na prática, pode ser útil ter variáveis latentes adicionais z (além dos parâmetros θ) como parte de um modelo. Essas variáveis latentes são diferentes dos parâmetros θ porque não parametrizam o modelo explicitamente. Mas elas podem descrever o processo de geração de dados, contribuindo para a capacidade de se interpretar o modelo. Elas podem, ainda, simplificar a estrutura do modelo e permitir definir estruturas de modelos mais simples e mais ricas. Essa simplificação da estrutura do modelo geralmente vem junto com um número menor de parâmetros do modelo. O aprendizado em modelos com variáveis latentes pode ser feito usando algoritmo de maximização de expectativas.

Alguns exemplos onde essas variáveis latentes podem ser úteis são utilização de Principal Component Analysis (PCA) para redução de dimensionalidade, estimação de densidade com modelos de mistura Gaussiana, modelagem de séries temporais com modelos ocultos de Markov ou sistemas dinâmicos e meta aprendizagem e generalização de tarefas.

Apesar da introdução dessas variáveis latentes poderem fazer a estrutura do modelo e do processo generativo mais fáceis, o aprendizado nesses modelos geralmente é mais difícil.

Esses modelos também permitem definir o processo que gera dados a partir dos parâmetros. Chamando os dados de x, os parâmetros do modelo de θ e as variáveis latentes de z, podemos obter a distribuição condicional $p(x|z, \theta)$ que nos permite gerar dados para quaisquer parâmetros e variáveis latentes de modelos. Dado que z são variáveis latentes, podemos colocar um priori $p(z)$ nelas.

Modelos com variáveis latentes podem ser utilizados para aprendizado e inferência de parâmetros. Para facilitar a aprendizagem (por exemplo, pela estimação de máxima verossimilhança ou inferência Bayesiana), segue-se um procedimento de dois passos.

Primeiro, calcula-se a verossimilhança do modelo $p(x|\theta)$, que não depende de variáveis latentes.

Depois, usa-se essa verossimilhança para estimação de parâmetro ou inferência Bayesiana, onde usamos as expressões já demonstradas de estimação de parâmetros e inferência Bayesiana.

Como a função de verossimilhança $p(x|\theta)$ é a distribuição preditiva dos dados a partir dos parâmetros do modelo, precisamos marginalizar as variáveis latentes de forma que $p(x|\theta) = \int p(x|z,\theta)p(z)dz$, onde $p(x|z,\theta)$ é conhecido e $p(z)$ é o priori nas variáveis latentes. Observe que a verossimilhança não deve depender das varíáveis latentes z, mas é apenas uma função dos dados x e dos parâmetros θ do modelo.

A partir dessa fórmula, a verossimilhança permite a estimação dos parâmetros pela verossimilhança máxima. A EMP também pode ser usada diretamente nos parâmetros θ. Mas, a inferência Bayesiana da verossimilhança em um modelo de variáveis latentes funciona da seguinte forma: Adiciona-se um priori $p(\theta)$ aos parâmetros do modelo e usa-se o teorema de Bayes para obter uma distribuição posterior usando os parâmetros do modelo, dado um conjunto de dados $\mathcal{X}$, conforme a fórmula:

$$p(\theta|\mathcal{X}) = \frac{p(\mathcal{X}|\theta)p(\theta)}{p(\mathcal{X})}$$

A distribuição posterior, então, pode ser usada para predições no framework de inferência Bayesiana.

Um desafio que existe nesse modelo de variáveis latentes é que a verossimilhança $p(\mathcal{X}|\theta)$ exige a marginalização das variáveis latentes. Porém, quando escolhermos a priori conjugada $p(z)$ para $p(x|z,\theta)$, a marginalização não pode ser tratada analiticamente e deve se recorrer a aproximações.

De forma similar aos parâmetros a posteriori, podemos calcular a distribuição posterior nas variáveis latentes de acordo com $p(z|\mathcal{X}) = \frac{p(\mathcal{X}|z)p(z)}{p(\mathcal{X})}, p(\mathcal{X}|z) = \int p(\mathcal{X}|z,\theta)p(\theta)d(\theta)$, onde $p(z)$ é a distribuição a priori nas variáveis latentes e $p(\mathcal{X}|z)$ exige que os parâmetros θ do modelo sejam integrados.

Considerando a dificuldade de resolver integrais analiticamente, fica claro que a marginalização tanto das variáveis latentes como dos parâmetros do modelo ao mesmo tempo não é possível, de forma geral. Uma quantidade que é mais fácil de se calcular é a distribuição posterior das variáveis latentes, mas condicionadas aos parâmetros do modelo, como na fórmula $p(z|\mathcal{X},\theta) = \frac{p(\mathcal{X}|z,\theta)p(z)}{p(\mathcal{X}|\theta)}$ onde $p(z)$ é o priori nas variáveis latentes e $p(\mathcal{X}|z,\theta)$ é conhecido.

Podemos explorar o fato que todos elementos de um modelo probabilístico são variáveis aleatórias para definir uma linguagem unificada que os represente, como o exemplo de uma linguagem gráfica para representar a estrutura de modelos probabilísticos que será apresentada a seguir.

6 Modelos Gráficos Dirigidos e Seleção de Modelos

Os Modelos Gráficos Dirigidos (*Directed Graphical Models* - DGMs) são uma ferramenta poderosa para representar e manipular dependências probabilísticas entre variáveis em sistemas complexos. Eles fornecem uma maneira compacta e visual de especificar modelos probabilísticos, facilitando a interpretação e a análise de dependências entre variáveis aleatórias. Neste trabalho, abordaremos o conceito de Modelos Gráficos Dirigidos, suas aplicações, e como a seleção de modelos é crucial para encontrar o equilíbrio entre a complexidade do modelo e o ajuste aos dados. Discutiremos também a Navalha de Occam e como a abordagem Bayesiana incorpora automaticamente este princípio. Além disso, exploraremos o processo de seleção de modelos em diferentes contextos e a importância da verificação de independência condicional utilizando separação-d (*d-separation*).

1 Modelos Gráficos Dirigidos

Um Modelo Gráfico Dirigido é uma representação visual de um modelo probabilístico através de um grafo direcionado $G = (V, E)$, onde os nós $v_i \in V$ representam variáveis aleatórias, e as arestas direcionadas $e_{ij} \in E$ representam dependências condicionais entre essas variáveis. Um exemplo comum de DGMs são as Redes Bayesianas.

A distribuição conjunta $p(x_1, x_2, ..., x_n)$ de um conjunto de variáveis aleatórias $x_1, x_2, ..., x_n$ em um DGM pode ser fatorada como o produto das distribuições condicionais de cada variável, dado seus pais no grafo:

$$\prod_{i=1}^{n} p(x_i \mid \text{Pais}(x_i)) \tag{6.1}$$

onde $\text{Pais}(x_i)$ denota o conjunto de variáveis que têm arestas direcionadas para x_i.

a Exemplo

A Figura 12 apresenta dois exemplos de representação gráfica de modelos probabilísticos. Na primeira figura (Fig. 12.a), temos a representação da seguinte correlação entre variáveis:

$$p(a, b, c) = p(c \mid a, b)p(b \mid a)p(a) \tag{6.2}$$

onde a variável c depende das variáveis a e b, a variável b depende apenas da variável a, e a variável a não depende nem de b e nem de c.

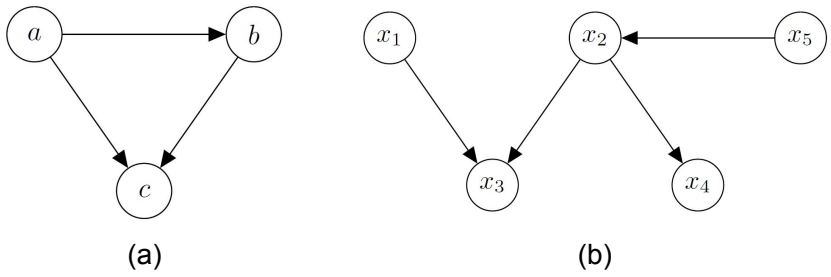

Figura 12 – Exemplo de representação gráfica de modelos probabilísticos.

Já na Fig. 12.b nós temos outro exemplo de relação de dependência entre as variáveis. Note, que neste exemplo temos 5 variáveis e a relação de dependência entre elas pode ser descrita como:

$$p(x_1, x_2, x_3, x_4, x_5) = p(x_1)p(x_2 \mid x_5)p(x_3 \mid x_1, x_2)p(x_4 \mid x_2)p(x_5) \tag{6.3}$$

onde as variáveis x_1 e x_5 não dependem de nenhuma variável do modelo, a variável x_2 e x_4 dependem de apenas uma variável, x_5 e x_2, respectivamente. Por fim, a variável x_3 depende de duas variáveis, sendo elas x_1 e x_2.

Assim, dada a representação gráfica das relações entre as variáveis, é possível fatorar e identificar a relação entre elas, e escrever essa relação matematicamente, como na Equação 6.3. Além disso, conseguimos a partir da equação matemática representar a relação das variáveis graficamente.

A representação gráfica em modelos probabilísticos oferece vantagens significativas, como a visualização clara e intuitiva das dependências entre variáveis, facilitando a identificação de relações de dependência e independência condicional. Isso melhora a comunicação do modelo e reduz erros na modelagem. Além disso, simplifica a decomposição da distribuição conjunta em produtos de distribuições condicionais, tornando os cálculos de inferência mais eficientes. Por fim, auxilia no design de novos modelos e validação de hipóteses, resultando em modelos mais robustos e interpretáveis.

A independência condicional é uma propriedade chave nos DGMs, e pode ser determinada pela técnica de separação-d (d-separation). Para verificar se um conjunto de nós A é condicionalmente independente de outro conjunto B dado um conjunto C, consideramos todas as trilhas/caminhos possíveis entre os nós de A e B. Consideramos um caminho bloqueado quando:

1. As setas no caminho se encontram cabeça-cauda ou cauda-cauda em um nó que pertence ao conjunto C.

2. As setas se encontram cabeça-cabeça em um nó e nem o nó, nem seus descendentes estão em C.

Se todos os caminhos estão bloqueados, A é dito ser d-separado de B por C, e a distribuição conjunta satisfará esta independência condicional.

Denotamos a relação de independência condicional como $A \vDash B \mid C$, caso seja verdade, ou $A \nvDash B \mid C$ em caso falso. Na Figura 12, podemos ver que a relação de independência condicional entre $A = \{x_3\}$, $B = \{x_5\}$ e $C = \{x_2\}$ é verdadeira. Isto é, $p(x_3 \vDash x_5 \mid x_2)$, uma vez que as setas em x_2 se encontram do tipo cabeça-cauda e $x_2 \in C$, satisfazendo assim a propriedade 1 descrita e bloqueando o caminho. Já para o caso em que $A = \{x_1\}$, $B = \{x_2\}$ e $C = \{x_3\}$ a independência condicional é falsa. Isto é, $p(x_1 \nvDash x_2 \mid x_3)$, uma vez que as setas em x_3 se encontram do tipo cabeça-cabeça, entretanto a propriedade 2 não é satisfeita, pois $x_3 \in C$. Assim o caminho não é bloqueado e a relação de independência condicional é falsa.

Por fim, os Modelos Gráficos Dirigidos, como redes Bayesianas, têm ampla aplicação em diversas áreas devido à sua capacidade de representar e manipular estruturas complexas de dependência probabilística de forma intuitiva e eficiente. Em bioinformática, são usados para modelar redes genéticas e prever interações biológicas. Na inteligência artificial e aprendizado de máquina, auxiliam em tarefas como classificação, predição e diagnóstico, permitindo a integração de conhecimento prévio e dados observacionais. Em sistemas de recomendação, ajudam a inferir preferências de usuários com base em interações passadas. Além disso, em economia e finanças, são aplicados para modelar incertezas e interdependências entre variáveis econômicas, melhorando a previsão e gestão de riscos. A versatilidade dos DGMs em lidar com incertezas e realizar inferências robustas torna-os ferramentas valiosas em campos que exigem análise de dados complexos e inter-relacionados.

2 Seleção de Modelo

A seleção de modelos é o processo de escolher o modelo mais adequado entre um conjunto de modelos candidatos. Este processo envolve equilibrar a complexidade do modelo e o ajuste aos dados, buscando o modelo mais simples que explica os dados de maneira razoável, conforme o princípio da Navalha de Occam. A Navalha de Occam é um princípio que sugere que, entre várias explicações igualmente válidas, a mais simples tende a ser a melhor. Em termos de seleção de modelos, isso implica preferir modelos com menos parâmetros ou suposições desnecessárias.

Na seleção de modelos Bayesiana, colocamos uma priori $p(M)$ sobre o conjunto de modelos e utilizamos o teorema de Bayes para calcular a probabilidade a posteriori de cada modelo dado os dados D:

$$p(M_i \mid D) = \frac{P(D \mid M_i)p(M_i)}{p(D)} \tag{6.4}$$

onde $P(D \mid M_i)$ é a evidência do modelo M_i, representando a capacidade do modelo de prever os dados observados, e $p(M_i)$ é a priori do modelo M_i.

Assumindo uma priori uniforme sobre os modelos, a posteriori é proporcional à evidência $P(D \mid M_i)$ recompensando modelos na proporção de quanto eles previram os dados que ocorreram. Modelos mais simples geralmente têm uma maior probabilidade a posteriori porque são menos propensos a superajustar os dados.

Por exemplo, considere dois modelos M_1 e M_2, onde M_1 é mais simples que M_2. Se D representa o espaço de todos os conjuntos de dados possíveis, a evidência para cada modelo pode ser representada por $P(D \mid M_1)$ e $P(D \mid M_2)$, que representa a probabilidade dos dados dado o modelo M_1 e M_2, respectivamente. Se um conjunto de dados específico cai em uma região C do espaço onde M_1 prevê melhor que M_2, então M_1 será escolhido, mesmo que M_2 seja mais complexo.

a Nested Cross-Validation

Uma técnica utilizada para escolha de modelos no contexto de aprendizado de máquina é o *Nested Cross-Validation*. A *nested cross-validation* é uma técnica estatística utilizada em aprendizado de máquina para avaliar o desempenho de um modelo e realizar a seleção de hiper parâmetros de maneira robusta. É especialmente útil em situações onde a quantidade de dados é limitada e o risco de *overfitting* é alto.

A *nested cross-validation* envolve dois níveis de validação cruzada. O primeiro nível é externo que divide o conjunto de dados original em várias partes. Em cada iteração, uma parte é reservado como conjunto de teste, enquanto as partes restantes são utilizados para treinamento e validação interna. No nível interno, dentro do conjunto de treinamento do nível externo, realiza-se uma segunda rodada de validação cruzada para a seleção e ajuste de hiper parâmetros. Em cada iteração do nível interno, o conjunto de treinamento é dividido novamente em múltiplas partes, onde se realizam os ajustes necessários.

3 Fatores Bayesianos para comparação de modelos

Os Fatores de Bayes são uma ferramenta poderosa no contexto da inferência Bayesiana, utilizada para a comparação de modelos estatísticos. Eles fornecem um método quantitativo para avaliar a evidência a favor de um modelo em relação a outro, com base na probabilidade dos dados observados sob cada modelo.

O Fator de Bayes entre dois modelos M_1 e M_2 é definido como a razão das probabilidades marginais dos dados sob cada modelo, podemos representar como:

$$\frac{p(D \mid M_1)}{p(D \mid M_2}, \tag{6.5}$$

onde $p(D \mid M_1)$ e $p(D \mid M_2$ são as evidências ou probabilidade marginal dos modelos M_1 e M_2, respectivamente.

Os Fatores de Bayes são uma ferramenta valiosa para a comparação de modelos, oferecendo uma abordagem quantitativa e intuitiva para avaliar a evidência relativa entre diferentes modelos. Embora possam ser computacionalmente exigentes, sua capacidade de incorporar informações a priori e penalizar a complexidade de maneira natural os torna uma escolha poderosa em muitas aplicações estatísticas e de aprendizado de máquina.

Concluindo, a seleção de modelo é uma etapa crucial em diversos campos de estudo e aplicação, onde a escolha do modelo adequado pode influenciar significativamente a precisão e a eficácia das previsões e inferências. Em aprendizado de máquina, a seleção de modelo é vital para determinar o algoritmo mais apropriado para tarefas como classificação, regressão ou agrupamento, levando em conta o trade-off entre bias e variância. Na economia, modelos preditivos são selecionados para melhor capturar as dinâmicas do mercado e prever tendências econômicas, auxiliando na tomada de decisões estratégicas. Em ciências naturais e sociais, a seleção de modelo é usada para identificar as melhores representações de fenômenos complexos, como mudanças climáticas ou comportamentos sociais, baseando-se em critérios de ajuste e complexidade do modelo. Na bioinformática, a escolha do modelo certo pode melhorar a compreensão de interações biológicas e predições sobre doenças. Métodos como validação cruzada, critérios de informação (AIC, BIC) e fatores de Bayes são frequentemente empregados para comparar modelos e garantir que o modelo escolhido seja o mais simples e eficaz possível para explicar os dados observados, evitando overfitting e melhorando a capacidade de generalização.

7 Conclusão

Neste trabalho, nós apresentamos conceitos sobre o conteúdo do Capítulo 8 do livro texto (DEISENROTH; FAISAL; ONG, 2020). Apresentamos os conceitos sobre dados como vetores, modelos como funções, modelos como distribuições de probabilidade e como encontrar parâmetros. Em seguida, descrevemos a classe de hipóteses de funções, funções de perda para treinamento, regularização para redução de *overfitting* e validação cruzada. Apresentamos conceitos de estimativa de parâmetros, estimação de máxima verossimilhança, máxima a posteriori e ajustes de modelo. Em seguida, discutimos sobre modelagem probabilística e inferência de modelos, modelos probabilísticos, inferência Bayesiana, modelos de variáveis latentes. Por fim, apresentamos definições de modelos gráficos dirigidos, seleção de modelo e fatores Bayesianos para comparação de modelos.

Referências

DEISENROTH, M. P.; FAISAL, A. A.; ONG, C. S. *Mathematics for machine learning*. [S.l.]: Cambridge University Press, 2020. Citado 2 vezes nas páginas 2 e 30.

SOARES, P. L. B.; SILVA, J. P. da. Aplicação de Redes Neurais Artificiais em Conjunto com o Método Vetorial da Propagação de Feixes na Análise de um Acoplador Direcional Baseado em Fibra Ótica. *Revista Brasileira de Computação Aplicada*, v. 3, p. 58–72, 2011. Citado na página 7.