

Modelagem estatística

Chegou a hora de aprofundarmos um pouco mais no universo da modelagem. Afinal, é através dela que iremos conseguir aplicar tudo o que aprendemos sobre ciência de dados até aqui.

Bora lá?

Para começar: o que é modelagem estatística?

Modelagem estatística é um processo que tenta expressar uma ou mais variáveis (chamadas de variáveis resposta) através de um outro conjunto de variáveis (chamadas de explicativas). Essa expressão é dada através da seguinte lógica:

variável resposta = função (variável explicativa)

Da forma matemática tradicional, essa lógica é dada por:

$$y = f(x)$$

Traduzindo, y é a variável resposta, f é a função matemática e x é a variável explicativa.

Escrevendo dessa maneira, tudo parece lindo né? Porém, sabemos que na prática nem sempre é possível criar uma função perfeita que explique a variável target (y).

Isso acontece pelo fato de que a relação entre as variáveis resposta e explicativas não é exata, muito menos perfeita. A variável resposta é influenciada por x , mas também pode sofrer influência de outros fatores aleatórios, que não conseguimos controlar ou prever que irão ocorrer.

Além disso, é importante destacar que estamos trabalhando com variáveis aleatórias. Isto é, tanto nossa variável resposta quanto as explicativas podem assumir qualquer valor dentro de um espaço de probabilidades, portanto não existe uma regra exata que define o valor assumido por cada uma delas.

Essa questão de aleatoriedade nos obriga a reconhecer que vivemos em um mundo **estocástico**. Isto é, um mundo em que fenômenos aleatórios são inerentes a qualquer evento e, independentemente de haver um modelo extremamente confiável que explique tal evento, jamais será possível obter o resultado exato.

EM RESUMO, NENHUM MODELO NUNCA ACERTARÁ 100% O VALOR VERDADEIRO DE UMA VARIÁVEL

RESPOSTA. ISSO SE DÁ PELA EXISTÊNCIA DE FATORES ALEATÓRIOS QUE INTERFEREM NESSA VARIÁVEL E QUE SIMPLEMENTE NÃO CONSEGUIMOS CONTROLAR OU “MODELAR”.

Por isso, sempre incluímos um “erro” na nossa função de modelagem. Esse erro representa tudo aquilo que influencia o y e que não conseguimos exatamente observar e escrever matematicamente. Não espere erros nulos, justamente por conta do que discutimos acima. O modelo faz previsões e não bruxaria, certo?

Considerado isso, veja como fica a expressão acima:

$$y = f(x) + \text{erro}$$

Agora sim temos uma expressão muito mais coerente com o mundo real!

Se até nós erramos, porque nosso modelo também não pode errar?

É importante destacar que essa $f(x)$ representa nossa equação do modelo, isto é:

$$f(x) = \beta_0 + \beta_1 \cdot x_1 + \dots + \beta_n \cdot x_n$$

Lembre-se que é sempre relevante inserir novas variáveis no modelo, e verificar o quanto a performance dele melhora. Geralmente, quando adicionamos novas variáveis que sejam significativas, a tendência é que esse erro diminua, pois são variáveis que de fato explicam melhor o comportamento do que queremos prever.

ENTENDENDO MAIS SOBRE VARIÁVEIS E SUAS INFLUÊNCIAS

Vamos entender um pouco melhor sobre a influência das variáveis em um modelo.

De maneira geral, conforme vimos acima, a performance do modelo tende a melhorar conforme adicionamos novas variáveis. Entretanto, é preciso ter um certo critério ao adicioná-las pois, dependendo de certas características, elas podem tanto melhorar quanto piorar a modelagem de um problema.

Que características são essas?

SIGNIFICÂNCIA DAS VARIÁVEIS NO MODELO

Primeiramente, é preciso avaliar se uma determinada variável é minimamente explicativa para o modelo. Essa avaliação é feita através de um teste de hipóteses para o beta (coeficiente) que multiplica essa variável na equação, e é realizado para todas as variáveis incluídas.

Lembra de teste de hipóteses?

No caso, a hipótese nula assumida (H_0) é a de que o valor de beta para essa variável em questão é nulo. Isto é, ela não influencia em nada o modelo. A hipótese alternativa é a de que esse beta é diferente de zero.

E COMO SABEMOS SE REJEITAMOS OU NÃO A HIPÓTESE NULA?

Para isso, analisaremos o nível de significância considerado e o respectivo p-valor obtido nesse teste. Geralmente, o nível de significância padrão é de 5%, mas ele pode oscilar de acordo com a decisão de quem faz o teste. Assim, ao realizá-lo para a variável em questão, obteremos um respectivo p-valor e, através dele, decidimos se aceitamos ou rejeitamos H_0 .

Quando o p-valor é menor que o nível de significância considerado, rejeita-se a hipótese nula. Ou seja, conclui-se que o beta que multiplica essa variável na equação é diferente de zero. Entretanto, um p-valor baixo não necessariamente significa que a variável é importante. Ele apenas mostra se de fato há um mínimo de “influência” dessa variável no modelo.

Esse teste é realizado para cada uma das variáveis consideradas na modelagem. Nos próximos materiais, veremos com mais detalhes os valores dos testes e como são disponibilizados em código.

BELEZA, MAS COMO DESCUBRO A IMPORTÂNCIA DE CADA VARIÁVEL?

Para verificar a importância real, analisa-se os coeficientes (betas) **ajustados** de cada uma das variáveis. Esse ajuste é necessário para colocar todos os coeficientes na mesma escala e, assim, poder compará-los. Dessa forma, fica viável escolher as variáveis com os maiores valores dentre os coeficientes ajustados.

! Atenção:

É comum e necessário olhar sempre o valor do r-quadrado durante o uso de um modelo linear, aquela famosa medida que indica o quanto se explicou os dados considerados.

Porém, a obtenção de um r-quadrado alto ao adicionar variáveis no modelo não significa necessariamente que a variável adicionada é relevante. Quanto mais variáveis adicionadas (isto é, quanto mais informação é inserida) o r-quadrado tende a aumentar. Entretanto, esse aumento não reflete a **qualidade** da variável inserida.

Para analisar a qualidade, olhamos outra medida: o r-quadrado **ajustado**. Essa medida “ajustada” aumenta conforme o novo

termo inserido efetivamente melhora o modelo e, da mesma forma, diminui quando o contrário acontece. Ou seja, quando uma variável nova é inserida e não é realmente importante, o r-quadrado ajustado

Vars	R-Sq	R-Sq (adj)
1	72.1	71.0
2	85.9	84.8
3	87.4	85.9
4	89.1	82.3
5	89.9	80.7

diminui, como forma de penalizar o modelo.

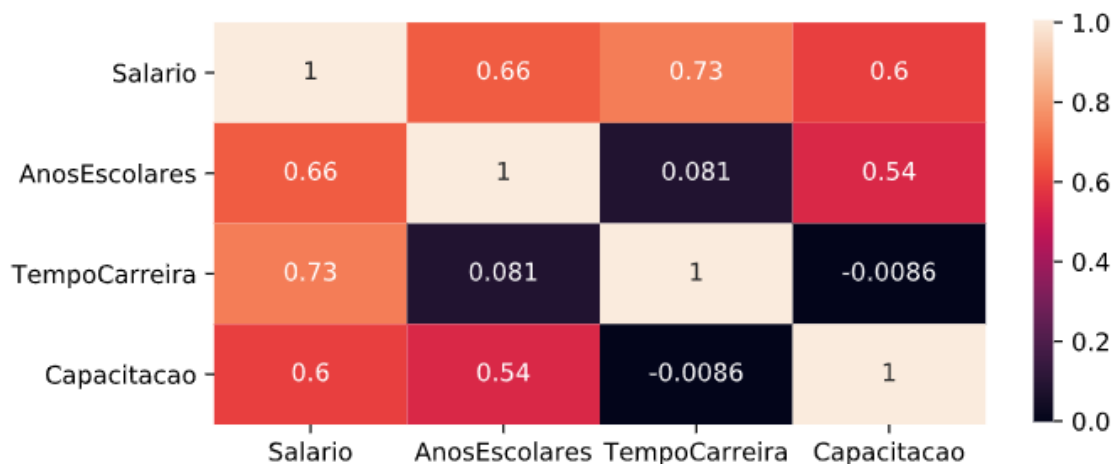
Assim, podemos concluir que aumentar o número de variáveis explicativas nem sempre significa aumentar a performance! Afinal, quanto mais variáveis forem adicionadas, mais complexo o modelo se torna e, como veremos mais à frente, menos generalizável ele tende a ser.

MULTICOLINEARIDADE

Quando trabalhamos com modelos lineares que envolvem mais de uma variável (como a regressão múltipla que aprendemos) é preciso avaliar se as variáveis não estão relacionadas entre si. Isto é, avaliar se não são **correlacionadas**.

Vamos entender melhor com um exemplo.

Suponha que você crie um modelo de previsão de salários, baseado em anos de estudo, tempo de carreira e capacitação. Como é necessário verificar se essas variáveis estão realmente relacionadas com o salário, você plota um gráfico de correlação:



Ao visualizar o gráfico, percebe-se que todas as variáveis estão relacionadas com a variável resposta salário. Entretanto, as variáveis “Capacitacao” e “AnosEscolares” estão bastante correlacionadas entre si.

QUAL O PROBLEMA DISSO?

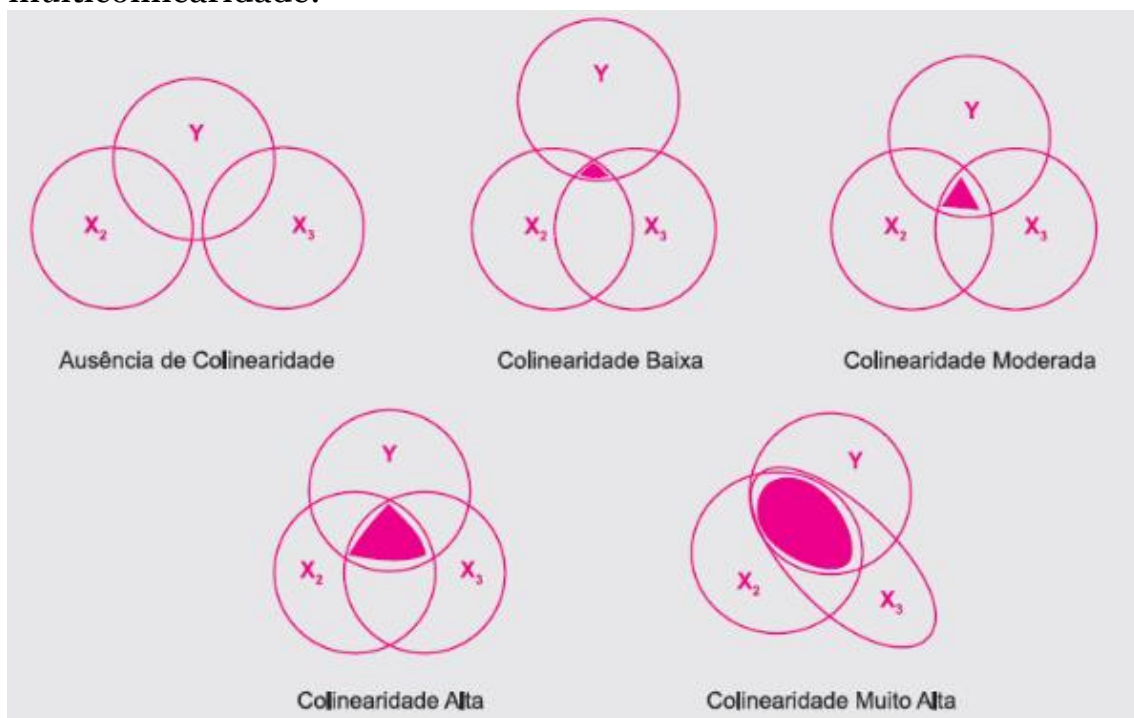
Uma correlação leve entre as variáveis preditoras não tem problema e é mais do que esperado. O problema começa quando essa correlação fica

forte demais, o que torna difícil conseguir isolar o efeito de cada uma na variável resposta.

Lembre-se que o objetivo principal é o de entender o quanto cada uma das variáveis preditoras, mantendo todas as demais constantes, interfere na resposta. Por isso, precisamos isolar o efeito de cada uma delas.

Nesse exemplo, provavelmente estamos capturando apenas parte do efeito da variável “Capacitacao” no salário previsto, enquanto podemos estar superestimando o efeito de “AnosEscolares”. Assim, a multicolinearidade torna os coeficientes (betas) do modelo bastante instáveis (alta variância), tornando difícil interpretá-los.

Veja no gráfico abaixo uma visualização prática do conceito de multicolinearidade:



A medida que a colinearidade vai aumentando entre as variáveis preditoras, percebe-se que o efeito compartilhado entre elas começa a ser refletido em Y de maneira exagerada, tornando difícil detectar o efeito individual de cada uma delas na variável resposta.

COMO RESOLVER ISSO?

Uma possibilidade seria a de excluir uma das variáveis que estão correlacionadas entre si, e verificar o modelo novamente. Como essas variáveis capturavam o efeito uma da outra, o modelo não seria tão prejudicado assim.

Há também algumas outras técnicas para lidar com esse problema, como a Análise de Componentes Principais (PCA), que serão vistas mais a frente no curso.

PROGRAMANDO COM MACHINE LEARNING: UMA MUDANÇA DE PARADIGMA

Lembra quando você começou a aprender a programar?

Qual era a ideia principal? A ideia era escrever todos os passos necessários para a execução de um programa, como o print de um simples “Hello World”, por exemplo.

Na programação tradicional, é preciso definir cada etapa do programa (ou código) para que o computador consiga executá-lo. Afinal, estamos lidando com uma máquina, que requer um passo a passo de tudo aquilo que precisa ser feito.

Agora, imagine que você queira criar um código de reconhecimento de imagens de animais. Seguindo a programação tradicional, você teria que “ensinar” para a máquina todos os padrões possíveis e existentes de imagens desse tipo. Meio complexo fazer isso, não é mesmo?

E qual a solução para esse problema?

Ora, usando machine learning (ML) conseguimos resolver esse problema. O modo de “ensinar” o computador utilizado por ML é através de exemplos, e não através de um passo a passo (contendo todas as situações possíveis) escrito em código, como no modo tradicional.

Pense só o quanto seria trabalhoso escrever um código com todas as possibilidades de cores, formatos e poses de animais em uma imagem. Inviável, não é mesmo?

Assim, no exemplo das imagens de animais, podemos reunir uma base de dados com as mais variadas imagens, que servirão de exemplos para o computador. Quando uma nova imagem for apresentada, ele terá como “referência” os exemplos que ele conseguiu aprender anteriormente, graças ao uso de ML.

Se pararmos para pensar, esse processo é extremamente parecido com nosso método de aprendizagem. Quando aprendemos através de exemplos, conseguimos replicar esse aprendizado em novos problemas que aparecerão.

Detalhe importante: aprender é diferente de decorar. Assim como nós humanos não aprendemos nada decorando o conteúdo, o computador também se comporta dessa maneira. Como veremos adiante, é de extrema importância que o algoritmo aprenda o comportamento geral dos dados, ao invés de simplesmente decorar um comportamento específico de um dataset.

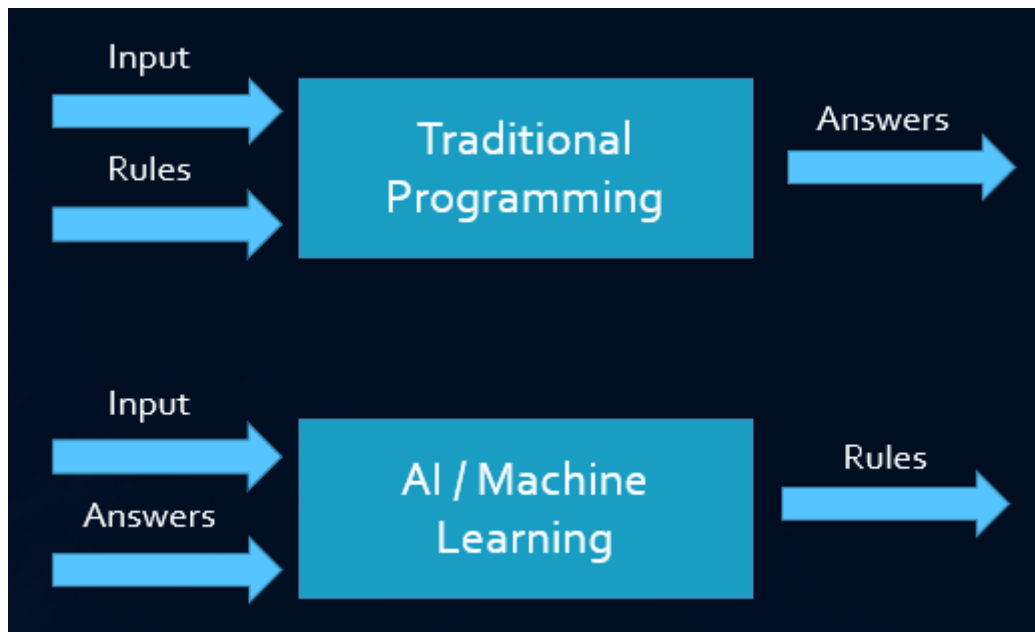


Imagem via [Quora](#)

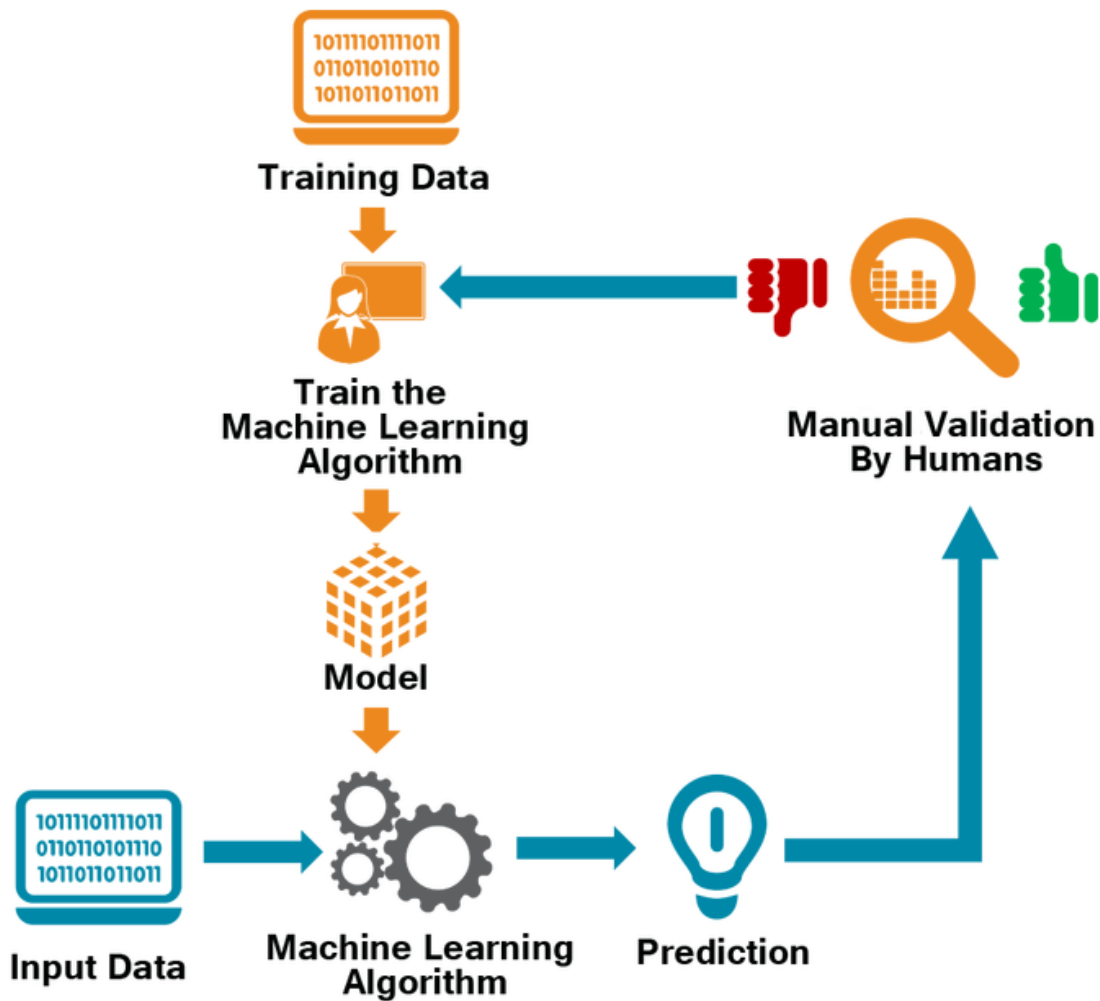


Imagem via: <http://www.technofist.com/>

COMO IDENTIFICAR SE UM MODELO É REALMENTE EFICIENTE?

Nesse momento, entramos em uma das questões mais importantes da área de ciência de dados. Certamente, você aprendeu e ainda aprenderá diversos modelos e técnicas para tratamento e previsões de dados, não é mesmo?

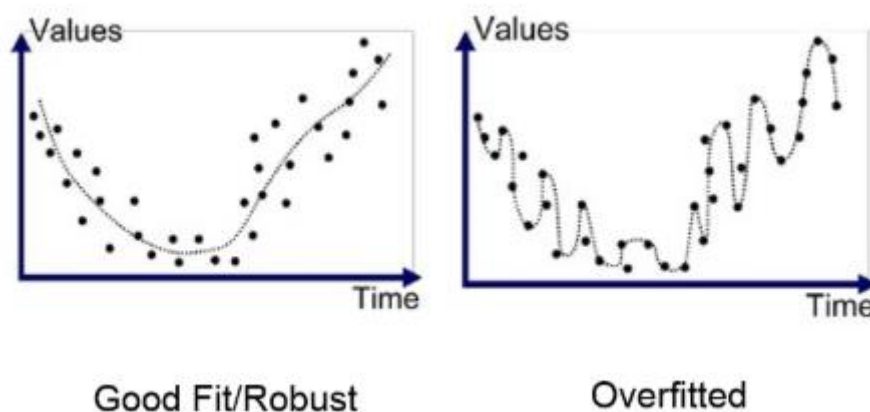
Entretanto, a principal característica a ser considerada em qualquer modelo é a sua capacidade de **generalização**. Ou seja, ao receber novos conjuntos de dados referentes ao problema para o qual ele foi construído, ele deve ser capaz de fazer previsões coerentes acerca desse conjunto, previsões que se aproximam bem com a realidade.

Vamos voltar ao exemplo anterior do modelo de previsão de salários, em que usamos dados de escolaridade, tempo de carreira e capacitação para prever o possível salário de uma pessoa de acordo com essas variáveis. Durante a fase de treinamento, a performance do modelo parecia ideal. Depois dessa etapa de treinamento, você pegou uma nova base de dados (diferente da que você utilizou antes) que contém novos dados de escolaridade, tempo de carreira e capacitação. Ao inseri-la no modelo, você percebe que as previsões estão bem ruins. O que pode ter acontecido?

Isso ocorre principalmente quando o modelo, ao invés de generalizar (“aprender”), “memoriza” os dados que recebeu durante o treinamento. Ou seja, ele não consegue fazer previsões de dados que nunca viu antes, pois ele não aprendeu um padrão geral de reconhecimento.

Essa situação é exatamente o que conhecemos como **overfitting**. Ou seja, durante o treinamento a performance do modelo é altíssima devido a um sobreajuste do mesmo aos dados de treino. Entretanto, ao se deparar com novas entradas, a performance é extremamente ruim.

Veja como isso se reflete graficamente:

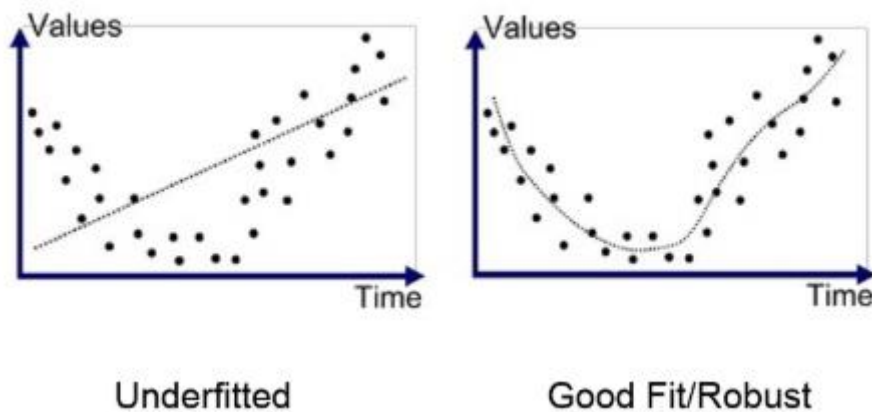


Da mesma forma, também podemos nos deparar com a situação

contrária: obter um modelo que simplesmente não é capaz de explicar os dados. Esse problema é menos comum, mas também acontece e é conhecido como **underfitting**.

Veja

só:



O underfitting aparece principalmente quando consideramos poucas variáveis explicativas para explicar o comportamento da variável resposta, que tende a ser mais complexo. É por isso que o modelo generaliza tão pouco, pois necessita de mais “recursos” para explicar esse comportamento.







Por isso, nem sempre aumentar o volume de dados é uma solução nesse caso, mas sim adicionar mais variáveis (aumentar o espaço de hipóteses) para melhorar o desempenho do modelo e garantir um ajuste adequado.

Mas que tanto de problema! Como escapamos disso tudo?

Calma que tem solução!

VIÉS E VARIÂNCIA: O FAMOSO *TRADE-OFF*

Antes de conhecer esses conceitos, comece fazendo o seguinte exercício:

Grupo 1	Grupo 2
  	  

De acordo com a tabela acima, indique a que grupo pertence os seguintes animais:

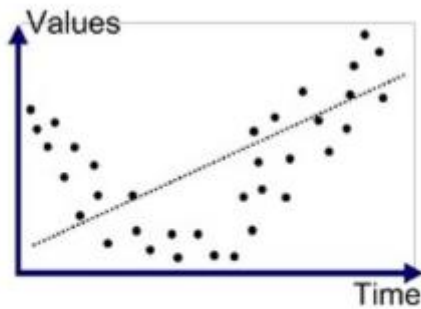


Imagem adaptada. Original via <https://juliocprocha.wordpress.com/>
Existem algumas maneiras de classificar esses animais, dependendo da linha de pensamento que decidirmos seguir. Uma linha de pensamento possível poderia ser a de dividir esses animais entre mamíferos ou opívaros (aves). A segunda poderia ser classificá-los de acordo com a presença ou não de asas em cada um deles, por exemplo.

Portanto, veja que é preciso assumir uma hipótese para que o exercício seja resolvido. Isto é, assumimos uma determinada linha de pensamento (uma suposição) para simplificar o processo de classificação dos animais. Esse é o conceito de **viés**.

De uma maneira mais conceitual, o viés (ou bias) pode ser entendido como uma simplificação ou generalização feita com o intuito de atingir uma determinada conclusão. Modelos mais simples (com poucas variáveis, por exemplo) costumam ser mais enviesados, pois tendem a simplificar muito o comportamento dos dados.

Conforme já vimos acima, esse seria o exemplo de um modelo com alto viés:



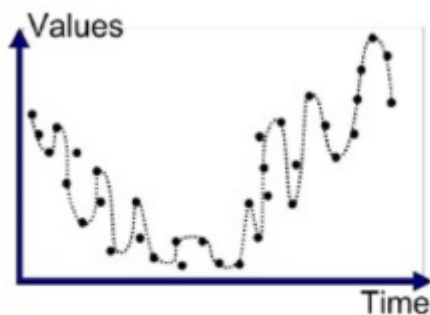
Underfitted

Veja que a reta estabelecida pelo modelo pouco explica o verdadeiro comportamento dos pontos de dados. Afinal, eles possuem um comportamento mais complexo, que não consegue ser representado e tampouco generalizado através de uma reta.

Seguindo na discussão sobre esse famoso *trade-off*, temos a **variância**. Trata-se de uma medida que indica o quanto as estimativas do modelo estão distantes do valor esperado, ou seja, o quão “longe” nossas estimativas estão dos dados reais.

Lembra quando calculamos as distâncias dos pontos de dados até a reta de regressão? Pois bem, no caso de alta variância, temos distâncias muito distintas dos pontos de dados reais em relação à linha determinada pelas previsões do modelo. Isso acontece quando o modelo passa a “aprender” padrões muito específicos dos dados de treinamento, que não se replicam em conjuntos de dados distintos.

Conforme exemplo anterior, esse é um exemplo de um modelo com alta variância:



Overfitted

Perceba que essa curva estabelecida no gráfico praticamente decorou o comportamento específico dos dados de treinamento. Por isso, esse modelo não consegue generalizar o comportamento de novos conjuntos de dados.

Portanto, o desempenho de modelos com alta variância tende a ser ruim, pois a generalização se torna difícil de alcançar e as chances de overfitting são altas. Da mesma forma, modelos altamente enviesados são ruins por fazerem generalizações não compatíveis com o verdadeiro comportamento dos dados, o que também gera possibilidade de overfitting.

EITA! MAS QUAL SOLUÇÃO ESCOLHER ENTÃO?

É exatamente por isso que o termo *trade-off* é mencionado. O ideal é atingir o equilíbrio entre viés e variância, de forma que se tenha um modelo com boa performance e, ao mesmo tempo, generalizável.

Na figura abaixo, a linha tracejada representa exatamente esse equilíbrio, que corresponde justamente ao valor mínimo da curva “Total error”. Perceba que qualquer alteração de complexidade do modelo resulta em desequilíbrio entre viés (bias) e variância, levando aos temidos underfitting/overfitting.

Observe:

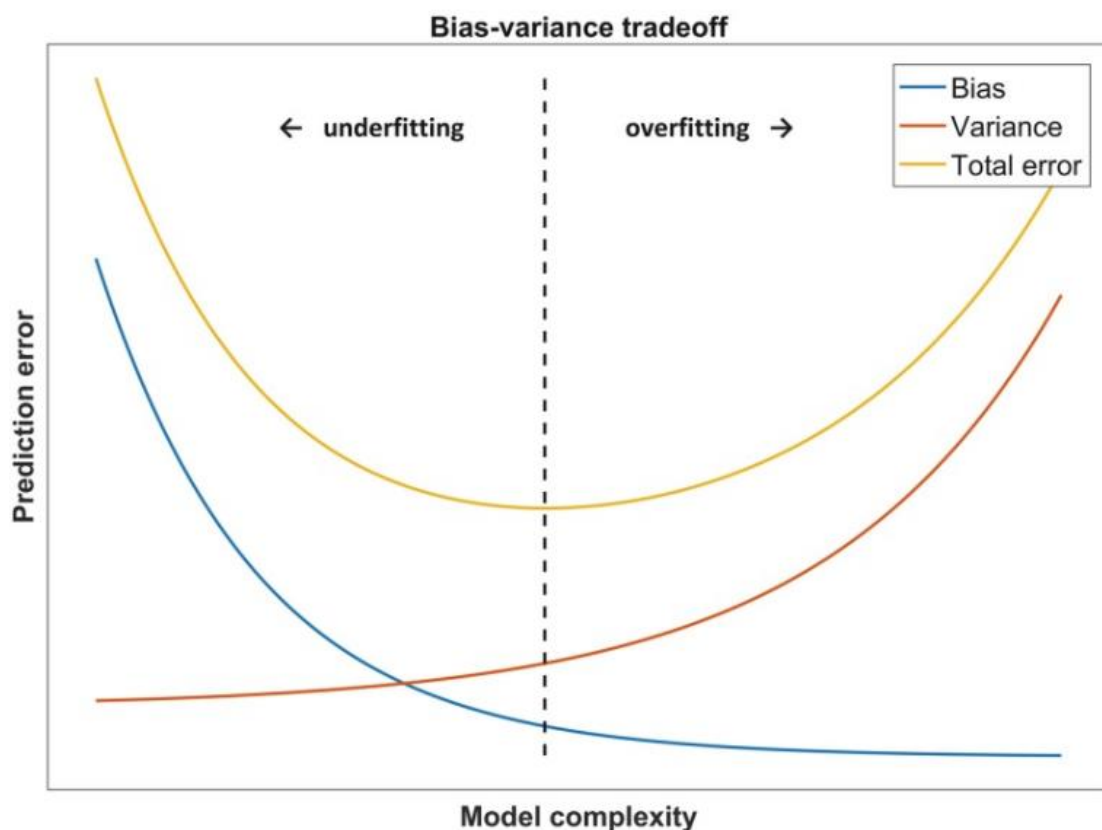


Imagem via: <https://www.ncbi.nlm.nih.gov/>

Claro que não existe uma receita pronta para sempre obter o melhor desempenho. Entretanto, certifique-se que você tem uma base de dados confiável, explore bastante os dados e teste sempre o uso de novas variáveis (features) e até modelos distintos.

Não se preocupe: você estudará esse conteúdo com mais detalhes mais pra frente no curso. Porém, grave isso: não existe resposta certa em ciência de dados. O caminho é sempre construir um modelo (não importa qual) que consiga ser generalizável e que produza resultados coerentes e positivos de acordo com aquilo que se deseja prever!