Passo a Passo análise de dados

1. **Entender o que está nos dados**

Pegar as variáveis uma a uma, ou as principais caso tenham muitas colunas e vá plotando gráficos para entender melhor do que se trata. Por exemplo, caso tenhamos uma coluna de gênero, é interessante ver quantos homens e mulheres tem, Caso seja idade, veja a média, mínima, plotar um histograma. Basicamente fazer uma análise geral sobre as variáveis do modelo a fim de ter um melhor entendimento sobre a base de dados.

1. **Fazer perguntas maiores**

**2.1) Como fazer perguntas?**

Perguntar coisas que você quer saber sobre o tema, use a criatividade para isso, por exemplo, em um dataset de cartões, quem será que gasta mais? ou menos? Qual o gênero que gasta mais? qual a faixa etária que tem maiores gastos? Basicamente explorar as relações que queremos entender através de perguntas.

É importante ressaltar a importância de ir afunilando essa análise até mirar em uma ou duas variáveis que são mais importantes para a base, se for um dataset de supermercado, foquemos nos produtos, em um de obesidade, no peso.

**2.2) Não consegui pensar em nada, o que fazer?**

Caso não consiga fazer essas perguntas, vá em code e veja alguns exemplos de usuários do kaggle que fizeram essa mesma base de dados, use o Chat GPT ao seu favor e faça consultas sobre isso.

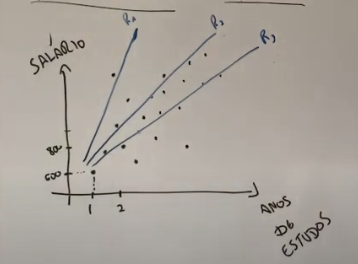
## **Semana 2: Introdução a Machine Learning, Lidando com Grandes Datasets no Pandas, Dicas de Pandas e Leituras Recomendadas**

1. **Introdução a machine learning**

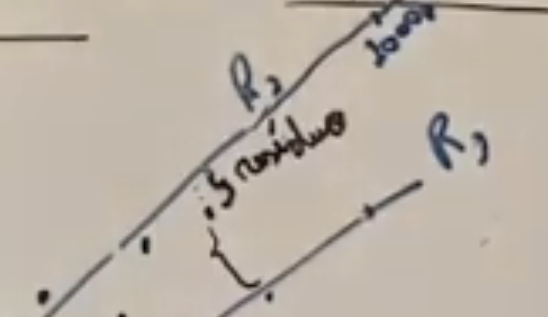
Um modelo de machine learning se baseia em encontrar padrões através de fórmulas matemáticas. A forma de encontrar esse padrão muda de modelo para modelo.

**1.1) Regressão Linear**

Para efeito de contextualização, vamos prever o salário de uma pessoa através da quantidade de anos de estudo. O que a regressão linear vai fazer para encontrar o padrão que foi dito anteriormente é a partir dos pontos colocados no gráfico, ela vai tentar achar a reta que melhor representa esse fenômeno.

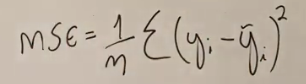


Nesse caso, o modelo de regressão linear achou 3 retas, e para escolher a melhor dentre elas, vai escolher a que tem menor taxa de erro. O erro seria a distância entre um ponto e o valor que ele foi predito na reta.



**1.2) Loss Function**

Também chamada de Função Perda, é criada pois apenas somar os erros de cada instância da base de dados não é muito eficiente, uma vez que muitas vezes podemos estar somando valores negativos com positivos e bagunçando o resultado da função. Logo, temos que achar uma forma de calcular o erro de forma eficiente, uma das formas é usar o erro ao quadrado, também chamado de MSE:



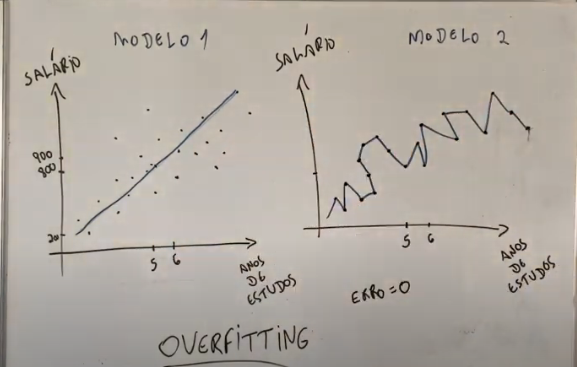
Elevando o erro ao quadrado, não vamos ter problemas com valores negativos

O modelo de machine learning vai sempre tentar minimizar o valor da Loss Function, tentando achar a menor perda possível para o modelo final.

1. **Como um modelo de machine learning aprende**

**2.1) Overfitting**

É basicamente quando o modelo “decora” o comportamento dos dados do dataset, fazendo com que quando novos dados cheguem, ele não consiga prever adequadamente, mesmo tendo uma Loss Function muito baixa. Isso acontece pois o modelo não entendeu o comportamento, apenas decorou uma está cada ponto.



Nesse caso, por exemplo, apesar do modelo 2 ter uma menor Loss Function, ele não aprendeu o comportamento da base como o modelo 1 fez.

**Como garantir que o modelo não está Overfittado?** Dividindo a base de dados entre treino e teste, iremos validar o modelo com outra base de dados diferente da de treinamento, sabendo que caso tenhamos um Score alto no treino e no teste baixo, significa que o modelo está Overfittado.

**2.2) Construção de Datasets Reais**

Para a análise de dados, nem sempre teremos um Dataset bonitinho montadinho com target e apenas aplicando o modelo de ML, muitas vezes precisaremos usar de mergers e inner joins para criar base de dados que vão ser convenientes para o modelo de negócio em questão. Por exemplo:

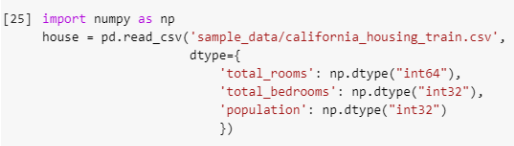
**Criar um modelo de ML que preveja se um cliente vai fazer uma compra nos primeiros 7 dias de cadastro a partir dos dados de cadastro e de compras apenas:** Nesse caso, teríamos que criar uma nova base de dados, usando ID, Data\_cadastro, Data\_cadastro 7, e a partir dessas colunas vamos consultar a base de dados de compras fazendo uma consulta para ver se o cliente comprou algo entre as duas datas previstas, sendo o retorno desta consulta a nossa variável target.

1. **Lidando com Datasets gigantes no pandas**

Quando estamos lidando com datasets muito grandes, seria interessante determinar a qual tipo cada coluna importada pertence, e podemos fazer isso com read\_csv, fazendo de modo a economizar mais memória usando tipos que exigem menos dela. Por exemplo, int8 consome 8x menos memória que int64, mas tem um limite do valor mínimo, assim como float16 e float64. Para analisar se podemos ou não usar float16 ou int8, temos que analisar o máximo e mínimo de cada coluna que queremos aplicar a eles. Além disso, uma estratégia muito eficiente na hora de lidar com datasets gigantes é converter string para “category”, ganhando um bom desempenho no processo.

**3.1) Importando o tipo correto**

Dá pra escolher qual o tipo de cada coluna direto na importação:

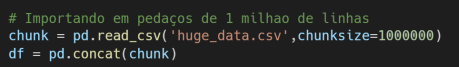


Outro aspecto que é importante ressaltar é que nem sempre o modelo vai precisar de todas as colunas disponíveis no dataset. Por exemplo, dentro de um dataset de clientes, dificilmente vai ser preciso ter informações sensíveis, como CPF, dados de cartão, entre outros. Para fazer essas importações filtradas usaremos use\_cols:



**3.2) Importação em CHUNKS**

Ao invés de importar o arquivo inteiro sozinho, é possível usar o parâmetro chunksize do read\_csv(), fazendo com que o arquivo seja importado em diversos pedaços, otimizando drasticamente o uso de memória RAM do pc.



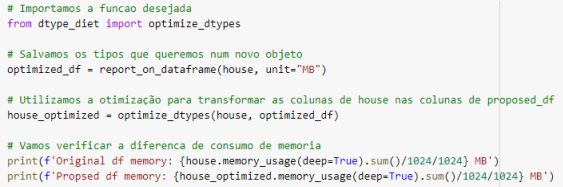
**OBS:** Em tratamento de grandes bases de dados, onde cada alteração vai demandar muitos minutos pra rodar, é interessante primeiro testar as transformações para uma parte da base de dados, e depois que tiver tudo pronto e preparado para aplicação, roda o resto da base de dados, diminuindo o tempo total de execução. Para fazer isso, pode-se usar o nrows para selecionar uma certa quantidade de linhas:



Outra informação importante é que existem bibliotecas em python que fazem a detecção do tipo mais eficiente para cada coluna automaticamente, chamada de dtype\_diet:



Esse comando vai retornar apenas recomendações dos tipos mais eficientes, caso queira realmente não ter trabalho e fazer ele mudar automaticamente, pode-se usar o optimize\_dtype:



**3.3) Pandas MODIN**

Pandas modin é uma biblioteca que permite usar todos os cores, aumentando a velocidade de execução do pandas, com os mesmos comandos, segue abaixo como importar:

