

Machine Learning e Quantum Computing

Lucas A. Leite^{a,1}^aGrupo SAMPA - Instituto de Física da USP

Professor/Grupo SAMPA

Abstract—Escrever alguma coisa como abstract desse relatório e estudo.**Keywords**—Machine Learning, Quantum Computing, Artigo

1. Introdução

Vamos começar nosso estudo embarcando em Machine Learning buscando um overview de técnicas e com uma quantidade balanceada de teoria de Statistical Learning e uma abordagem um pouco mais matemática, pretendo também deixar citações de onde as discussões e teoria estão sendo tiradas e podem ser aprofundadas ou lidas na íntegra, ao mesmo tempo, vou buscar trazer códigos com aplicações dos conceitos estudados.

Vamos começar com o estudo de Machine Learning clássico mas sempre que possível buscar conceitos de Quantum Computing para podermos trabalhar com essa infraestrutura no futuro.

2. Início

De forma bem vaga, definirei Machine Learning (ML) como um método de aproximação de uma função preditora do comportamento de um sistema específico, que é treinada com dados relativos a tal sistema. Eu vou supor que o leitor já tenha uma certa familiaridade com a finalidade do estudo de ML.

Vou passar, então, desse tópico de apresentação e motivação do estudo, direto para trabalhar com as principais estruturas de ML que temos: Regressão, Classificação e Clusterização. Trabalharemos com Redes Neurais também.

3. Regressão Linear

3.1. Modelo

Costumemente o tópico e o framework de ML é introduzido a partir de Classificação e exemplificações. com kNN, porém não farei desta forma e vou introduzir o conteúdo de uma forma um pouco mais crua e direta, pulando motivações, como já foi dito.

Sendo assim, vamos direto ao assunto. Quando estamos trabalhando com ML a ideia e a construção de uma função que será otimizada e que em Regressão Linear (RL) é descrita como:

$$f(\mathbf{X}) = \hat{\mathbf{Y}} = \mathbf{W}^T \mathbf{X} \approx \mathbf{Y}$$

Veja, nos temos um conjunto de dados $\mathbb{D} = \{(\mathbf{X}, \mathbf{Y})\}$ no qual \mathbf{X} são nossos *inputs* e \mathbf{Y} que são nossos *outputs*, ambos dados reais do nosso sistema/problema, outras nomenclaturas são usadas na literatura como (feature,label), (preditor, predição). Nossos grandes Romeu e Julieta são o foco principal quando tratamos de aprendizado supervisionado, que é quando temos justamente o *output* para podermos ter um parâmetro de comparação de erro ou acerto. É válido mencionar que o nome linear é sobre a linearidade nos parâmetros \mathbf{W} e não nos inputs, onde podemos fazer transformações.

Bem, voltando a nossa função definida acima, ela pode ser chamada de **Função de predição** ($\hat{\mathbf{Y}}$), veja bem $\hat{\mathbf{Y}}$ e \mathbf{Y} são diferentes, uma é a nossa função predição, que vem da relação linear descrita, o outro é o valor de output real que temos nos dados, respectivamente.

3.2. Otimização

Regressão Linear vem com um chute inicial para a função preditora que depois será otimizada para que cheguemos num valor de coefi-

cientes \mathbf{W} que melhor descrevam o comportamento dos dados que temos.

Assim, com esse conceito de comparação, vamos criar uma função de perda, chamaremos de Loss function ou Loss. O conceito geral é que essa função deve medir a diferença entre os dados preditos pelo nosso modelo e os valores reais que temos.

Um dos exemplo mais comuns de Loss function são os *Resíduos Quadrados* que tem uma forma de:

$$l = (y - \hat{y})^2$$

Porém podemos considerar o conjunto \mathbb{D} e somar o Loss considerando todos os dados que possuímos, isso nós chamaremos de Cost Function ou Função de custo, a otimização do processo está realmente nesta função.

$$J = (\mathbf{Y} - \hat{\mathbf{Y}})^2$$

De forma inocente vamos assumir que o melhor modelos que podemos ter é aquele com que cometeremos o menor número de erro, dessa forma minimizando a função de custo J . O único parâmetro que podemos levar em conta para a minimização da função de custo é o \mathbf{W} , dessa forma vamos derivar com respeito a ele:

$$\begin{aligned} \frac{\partial J}{\partial \mathbf{W}} &= \frac{\partial}{\partial \mathbf{W}} (\mathbf{Y} - \mathbf{W}\mathbf{X})(\mathbf{Y} - \mathbf{W}\mathbf{X})^T \\ &= \frac{\partial}{\partial \mathbf{W}} [\mathbf{Y}^2 + \mathbf{W}\mathbf{X}\mathbf{X}^T\mathbf{W}^T - \mathbf{Y}\mathbf{X}^T\mathbf{W}^T - \mathbf{W}\mathbf{X}\mathbf{Y}^T] \\ &= \mathbf{X}\mathbf{X}^T\mathbf{W}^T + \mathbf{W}\mathbf{X}\mathbf{X}^T - \mathbf{Y}\mathbf{X}^T - \mathbf{X}\mathbf{Y}^T = 2(\mathbf{W}\mathbf{X}\mathbf{X}^T - \mathbf{Y}\mathbf{X}^T) \\ &= 0 \end{aligned}$$

Assim podemos achar o valor de \mathbf{W} :

$$\mathbf{W} = \frac{\mathbf{Y}\mathbf{X}^T}{\mathbf{X}\mathbf{X}^T}$$

Porém esse valor só será possível de obter dessa forma, se e somente se, $\mathbf{X}\mathbf{X}^T$ for inversível.

3.3. Gradiente Descendente

Isso nem sempre é o que acontece, sendo assim, vamos criar uma técnica que é mais geral, chamamos essa técnica de **Gradiente Descendente**, e ele funciona dessa forma:

1º Teremos um chute inicial de parâmetros \mathbf{W}_0 e a partir desse chute inicial iremos buscar convergir para o mínimo em pequenos passos e atualizações, assim a cada interação o \mathbf{W} atualizado será o seguinte:

$$\mathbf{W}_{n+1} = \mathbf{W}_n - \eta \nabla_{\mathbf{W}} J$$

Onde o η é o que chamamos de step size, porém essa é uma denominação confusa, visto que esse hiperparâmetro seria mais como um step weight. O termo do gradiente é essencialmente o que fizemos anteriormente para o caso de $\mathbf{X}\mathbf{X}^T$ inversível. Bem e qual é o peso ideal de cada passo de otimização? Isso é algo bem nebuloso, a única informação relevante referente a essa questão é que esse hiperparâmetro não deve ser muito grande, porém também não muito pequeno, sweet spot geralmente é $\eta = 0,01$, porém é um chute. Outra possibilidade quando tratando desse problema é utilizar um step size ajustável a

alguma premissa que você queira, por exemplo, se o gradiente for muito grande, quero que η seja grande, ou pequeno, premissas desse tipo.

Bem, agora pode restar a dúvida, quando devemos parar de otimizar o nosso parâmetro W ? Em geral as principais formas usadas são número de interações ou ganho negligenciável de otimização, dando um retorno para que o sistema para de otimizar o parâmetro.

3.4. Gradiente Descendente Estocástico

Bem, vimos sobre alguns aspectos do GD, porém não discutimos sobre um caso que pode ser limitante para o uso dele... A quantidade de dados!

Caso tenhamos uma quantidade muito grande de dados calcular o GD será uma tarefa muito cara computacionalmente e levará muito tempo. Sendo

3.5. Verossimilhança

3.6. Generalized Linear Model (GLM)

4. Tratamento dos dados e divisão

Este capítulo é totalmente dedicado à praticas que são essenciais quando tratamos do conjunto de dados que possuímos. Bem quando temos um conjunto de dados do tipo:

$$\mathbb{D} = \{(\mathbf{X}, \mathbf{Y})\}$$

E queremos usar ele para treinar um modelo de descrição do nosso problema, nós utilizamos técnicas para o fitting dos dados segundo nossa função de predição. Porém, isso não gera a dúvida sobre 'o que garante que o nosso fit esta realmente certo?', 'será que se eu treinar meu modelo para ele cometer o mínimo de erros possíveis é a melhor solução?'. Para responder a primeira pergunta e acabando respondendo a segunda também, podemos fazer um teste, quebramos nosso conjunto de dado \mathbb{D} em dois pedaços, um \mathbb{D}_{treino} e o outro \mathbb{D}_{teste} segunda uma proporção digamos 80% e 20% respectivamente. Podemos então treinar o modelo com apenas os 80% e depois testar nosso modelo para predizer os dados no conjunto de 20%. Dessa forma, podemos ter um feedback de como o modelo vai atuar quando ele tiver diante de dados reais, não antes vistos.

Essa prática é o que chamamos de divisão Treino/Teste, e o modelo deve ser usado no conjunto de dados de teste apenas quando tiver fundamentalmente pronto, e é preciso ter cuidado para que não exista vazamento das informações de resposta para dentro do dataframe de treino do modelo, além do cuidado de que os dados sejam selecionados aleatoriamente.

Além desta técnica comentada, existem diversas outras, falarei em seguida de algumas outras:

4.1. Treino/Validação/Teste

Essa quebra é muito similar a que comentamos agora acima, porém invés de quebrar apenas em dois subconjuntos, quebraremos em três. O de **Treino**, que é usado para treinar o modelo, o de **validação** que é usado para analisar as métricas e para fazer mudança no modelo segundo os resultados obtidos nessas métricas e, por fim, o subconjunto de **Teste**, que tem o mesmo efeito da outra quebra explicada acima, que é usada apenas para testar o modelo num ambiente similar a realidade onde performará o modelo.

4.2. K-Fold

Essa abordagem é um pouco diferente das ultimas duas comentadas, aqui nós iremos pegar nosso dataset \mathbb{D} e quebra-lo em K subdatasets de mesmo tamanho, e faremos um revesamento onde pegamos um desses K subconjuntos e chamaremos de subconjunto de Teste e os demais serão os subconjuntos de Treino. Faremos isso até que todos os subconjuntos tenham sido utilizados como conjuntos de Teste e tivermos consequentemente gerado K modelos. Por fim, tomamos a

média desses K modelos gerados como o modelo que vai descrever o comportamento do problema que temos.

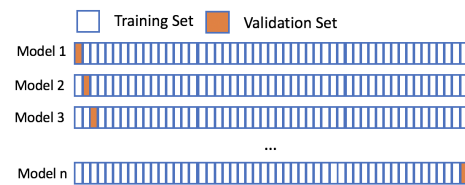


Figure 1. Esboço da estratégia de K-fold e Leave One Out.

4.3. Leave One Out

Na mesma linha em que pensamos na estratégia do K-fold, podemos levar ao limite, onde invés de quebrarmos nosso data set \mathbb{D} em K partes, vamos levar ao máximo possível e tomar cada um dos dados, Deixando um dos dados para Teste e os demais como Treino. Faremos depois o mesmo processo que fizemos no K-Fold, treinaremos diversos modelos e pegaremos a média deles.

Logo num conjunto com N dados, treinaremos N modelos e tomaremos a média deles.

5. Equation

Equation 1, shows the Schrödinger equation as an example.

$$\frac{\hbar^2}{2m} \nabla^2 \Psi + V(\mathbf{r})\Psi = -i\hbar \frac{\partial \Psi}{\partial t} \quad (1)$$

The `amssymb` package was not necessary to include, because `stix2` font incorporates mathematical symbols for writing quality equations. In case you choose another font, uncomment this package in `tau-class/tau.cls/math` packages.

If you want to change the values that adjust the spacing above and below the equations, play with `\setlength{\eqskip}{8pt}` value until the preferred spacing is set.

6. Adding codes

This class¹ includes the `listings` package, which offers customized features for adding codes in \LaTeX documents specifically for C, C++, \LaTeX and Matlab.

You can customize the format in `tau-class/tau.cls/listings` style.

```
1 function fibonacci_sequence(num_terms)
2     % Initialize the first two terms of the sequence
3     fib_sequence = [0, 1];
4
5     if num_terms < 1
6         disp('Number of terms should be greater than or
7         equal to 1.');
```

```
7         return;
8     elseif num_terms == 1
9         fprintf('Fibonacci Sequence:\n%d\n',
10        fib_sequence(1));
11        return;
12    elseif num_terms == 2
13        fprintf('Fibonacci Sequence:\n%d\n%d\n',
14        fib_sequence(1), fib_sequence(2));
15        return;
16    end
17
18    % Calculate and display the Fibonacci sequence
19    for i = 3:num_terms
20        fib_sequence(i) = fib_sequence(i-1) +
21        fib_sequence(i-2);
22    end
23
24    fprintf('Fibonacci Sequence:\n');
25    disp(fib_sequence);
```

¹Hello there! I am a footnote :)

```
23 end
```

Code 1. Example of Matlab code.

If line numbering is defined at the beginning of the document, I recommend placing the command `\nolinenumbers` at the start and `\linenumbers` at the end of the code.

This will temporarily remove line numbering and the code will look better.

7. References

The default formatting for references follows the IEEE style. You can modify the style of your references. See appendix for more information.

8. Appendix

8.1. Alternative title

You can make the following modification in `tau-class/tau.cls`/title preferences section to change the position of the title.

```
1 \newcommand{\titlepos}{\centering}
```

Code 2. Alternative title.

This will move the title to the center.

8.2. Info environment

An example of the info environment declared in the ‘`tauenvs.sty`’ package is shown below. Remember that *info* and *note* are the only packages that translate their title (English or Spanish).

Information

Small example of info environment.

8.3. Equation skip value

With the `\eqskip` command you can change the spacing for equations. The default *eqskip* value is 8pt.

```
1 \newlength{\eqskip}\setlength{\eqskip}{8pt}
2 \expandafter\def\expandafter\normalsize\expandafter{%
3   \normalsize%
4   \setlength\abovedisplayskip{\eqskip}%
5   \setlength\belowdisplayskip{\eqskip}%
6   \setlength\abovedisplayshortskip{\eqskip-\
   ↪ baselineskip}%
7   \setlength\belowdisplayshortskip{\eqskip}%
8 }
```

Code 3. Equation skip code.