O que é Machine Learning?

"É uma abordagem para se tomar várias pequenas decisões com relação aos dados disponíveis."

- Cassie Kozyrkov.

"É uma aplicação da Inteligência Artificial que dá a habilidade aos computadores de aprender e melhorar a sua performance sem que sejam explicitamente programados ou então com o mínimo de intervenção humana."

- Wikipedia

Exemplos de algoritmos:

- Filtro de Spam é um exemplo de programa de Machine Learning no qual aprende a sinalizar um spam dado uma série de exemplos (ou então chamado de conjunto de dados) cujo apresenta e-mails que sejam spams e não-spams (chamados de ham).

Observação

Diferentemente da programação usual, em Machine Learning a função do programador é alimentar o sistema com exemplos. E é trabalho do algoritmo de ML montar uma receita de como interpretar estes exemplos.

Referências

1. MFML 001 - What is machine learning?
2. What is Machine Learning? A definition
3. Hands-On Machine Learning with Sckit-Learn and TensorFlow

O que é Supervised Learning?

"É quando tu mostras ao sistema qual é o resultado correto que vai como uma entrada."

- Cassie Kozyrkov.

Em Supervised Learning, o conjunto de dados utilizado para treino inclui as soluções desejadas, chamadas de labels. Nesse tipo de modelo ML, os algoritmos aplicam o que aprenderam no passado em novos dados usando exemplos pré-classificados para prever eventos futuros. Começando com a análise de um conjunto de dados conhecido, o algoritmo de aprendizado produz uma função para fazer previsões sobre os valores de saída. Após o sistema ser suficientemente treinado, então ele se torna capaz de fornecer alvos para qualquer nova entrada.

Exemplos de tarefas:

- Tarefa de Classificação: Filtro de Spam é um exemplo de algoritmo de Supervised Machine Learning no qual aprende a classificar o tipo de e-mail como spam ou ham, uma vez que o filtro foi treinado com muitos exemplos pré-classificados.

Exemplos de algoritmos:

- k-Nearest Neighbors
- Linear and Logistic Regressions
- Support Vector Machines (SVMs)
- Decision Trees and Random Forests
- Neural Networks → algumas NN também podem ser unsupervised.

Referências

1. Machine Learning: Supervised versus Unsupervised - what's the difference?
2. What is Machine Learning? A definition
3. Hands-On Machine Learning with Sckit-Learn and TensorFlow

O que é Unsupervised Learning?

"É quando tu fornece uma entrada sem nenhum tipo de classificação ou rótulo, de forma que tu digas ao sistema: 'coloque coisas semelhantes juntas'."

"É um cartão de Rorschach para te ajudar a sonhar. Tu vais tentar diversas vezes até que aqueles agrupamentos façam sentido e te inspirem."

- **Cassie Kozyrkov.**

Em Unsupervised Learning, o conjunto de dados usado para treino não apresenta nenhum tipo de pré-classificação ou label (rótulo). Neste tipo de aprendizado, o sistema não busca por encontrar a saída correta, mas sim explorar os dados e buscar por algum tipo de padrão.

Exemplos de tarefas:

- Tarefa de Clusterização (Agrupamento): crie quatro grupos com estas fotos, colocando coisas semelhantes juntas.

Exemplos de algoritmos:

- Clustering:
  - k-Means
  - Hierarchical Cluster Analysis (HCA)
  - Expectation Maximization
- Visualization e Dimensionality Reduction:
  - Principal Component Analysis (PCA)
  - Kernel PCA
  - Locally-Linear Embedding (LLE)
  - T-distributed Stochastic Neighbor Embedding (t-SNE)
- Association rule learning:
  - Apriori
  - Eclat

Referências

1. Machine Learning: Supervised versus Unsupervised - what's the difference?
2. What is Machine Learning? A definition
3. Livro: Hands-On Machine Learning with Sckit-Learn and TensorFlow

O que é Distribuição Amostral?

"É a distribuição de freqüência de uma medida estatística sobre várias amostras ou re-amostras"

- **Practical Statistics for Data Scientists**

O termo "distribuição amostral de uma estatística" faz referência à distribuição de uma medida estatística de diversas amostras aleatórias retiradas da mesma população.

Referências

1. Wikipedia
2. Livro: Practical Statistics for Data Scientists

O que é o Teorema de Limite Central e o que ele implica para distribuições amostrais?

"É a tendência de que a distribuição amostral assuma a forma de uma distribuição normal (Gaussiana) a medida em que aumenta o tamanho da amostra"

- **Practical Statistics for Data Scientists**

O Teorema do Limite Central diz que a média retirada de múltiplas amostras assemelha-se à uma distribuição normal, mesmo que a população não seja "distribuída normalmente". Ou seja, implica que a distribuição de probabilidades de uma medida estatística de diversas amostras de uma mesma população assemelha-se à uma Gaussiana a medida que se aumenta o tamanho da amostra.

Referências

1. Livro: Practical Statistics for Data Scientists
2. Sampling Distribution: Central Limit Theorem in practice

O que é o p-value?

"Quanto menor for o p-value mais ridícula parece ser a nossa hipótece nula."
"É a probabilidade de vermos alguma coisa pelo mínimo tão ruim quanto àquela que vimos naquele mundo em que construímos a nossa hipótese nula."

- **Cassie Kozyrkov**

O que o p-value faz é responder a pergunta: o quão estatisticamente significante seria uma tomada de decisão?

Referências

1. What's a p-value?
2. How do you get the p-value?
3. How do you use p-values and significance levels

Defina o Teorema de Bayes e cite uma aplicação.

"É o senso comum aplicado à Matemática."
"Bayes via o mundo de uma forma que não se é possível saber como realmente é a realidade, mas o que podemos fazer é aumentar nosso conhecimento a medida em que vamos coletando cada vez mais informações e evidências."

- **Derek Muller**

"O Teorema de Bayes fornece um mecanismo formal para atualizar probabilidades."

- **Estatística Básica, Bussab & Morettin**

Descreve a probabilidade de um evento ocorrer, baseado em um conhecimento prévio que pode estar relacionado àquele evento.

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)} = \frac{P(E|H)P(H)}{P(H)P(E|H)+P(-H)P(E|-H)}$$

Existe uma realidade que a gente não sabe qual é, mas ao interagir com essa realidade a nossa crença de como é aquela realidade vai se atualizando e a cada interação com o ambiente a gente vai tendo mais certeza de qual seria a realidade por trás na qual a gente não tem acesso.

Demonstração

$$P(A \cap B) = P(B|A)P(A)$$
$$P(B \cap A) = P(A|B)P(B)$$
$$P(A \cap B) = P(B \cap A)$$
$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$

Aplicação

Um exemplo de aplicação da Forma de Bayes foi para encontrar a caixa preta do vôo AF447 (avião que caiu saindo do Brasil com destino à França). Ou seja, encontrar região com a maior probabilidade de queda do avião, a medida que mais e mais informações eram adicionadas a realidade do evento.

Referências

1. Veritasium
2. O Físico Turista
3. Wikipedia
4. Livro: Estatística Básica.

Tenho uma moeda e quero checar se ela é honesta. Como proceder?

É preciso ser feito um teste de hipótese, e para isso primeiro precisamos definir qual é a hipótese nula (ou em outras palavras qual é o mundo que a gente vive), e criar uma hipótese alternativa.

Então, criando as duas hipóteses:

H_0 : A moeda é honesta → Nossa hipótese nula é o mundo no qual a moeda é honesta.
H_1 : A moeda não é honesta → Nossa hipótese alternativa é o mundo no qual a moeda não é honesta.

Qual é a probabilidade de obtermos "coroa" em n jogadas?

1ª jogada: Prob(coroa) = 50%
2ª jogada: Prob(coroa) = 50% x 50% = 25%
3ª jogada: Prob(coroa) = 50% x 50% x 50% = 12.5%
4ª jogada: Prob(coroa) = 50% x 50% x 50% x 50% = 6.3%
5ª jogada: Prob(coroa) = 50% x 50% x 50% x 50% x 50% = 3.1%
6ª jogada: Prob(coroa) = 50% x 50% x 50% x 50% x 50% x 50% = 1.6%

Então, em um mundo regido pela hipótese nula, a probabilidade de obtermos seis "coroas" em seis jogadas é de 1.6%. A pergunta é: isto é estatisticamente suficiente para rejeitarmos H_0? Na verdade, com as informações que temos, e com a maneira como estamos estruturando a solução, não podemos ter certeza para rejeitarmos uma hipótese e favorecer outra. Então, o que fazer?

Vamos definir um intervalo (α) no qual eu posso interpretar a probabilidade de ocorrência como sendo estatisticamente significante ou não para favorecer uma hipótese à outra. Digamos que escolhemos um $\alpha = 5\%$, ou seja:

1ª jogada: Prob(coroa) = 50%
2ª jogada: Prob(coroa) = 50% x 50% = 25%

3ª jogada: Prob(coroa) = 50% x 50% x 50% = 12.5%

4ª jogada: Prob(coroa) = 50% x 50% x 50% x 50% = 6.3%

_____ $\alpha = 5\%$

5ª jogada: Prob(coroa) = 50% x 50% x 50% x 50% x 50% = 3.1%

6ª jogada: Prob(coroa) = 50% x 50% x 50% x 50% x 50% x 50% = 1.6%

Esta linha separa aquilo que eu defini como sendo um evento aleatório possível e um evento "marcado". Por exemplo, se nosso mundo fosse H_1, ou seja o mundo em que a moeda não é honesta. Então a probabilidade de obtermos coroa seis vezes seguidas seria:

1ª jogada: Prob(coroa) = 100%

2ª jogada: Prob(coroa) = 100%

3ª jogada: Prob(coroa) = 100%

4ª jogada: Prob(coroa) = 100%

5ª jogada: Prob(coroa) = 100%

6ª jogada: Prob(coroa) = 100%

Pois nós vivemos nesse mundo e portanto não teria outra ocorrência se não obter "coroa". Mas estamos assumindo que vivemos em H_0, então a linha de $\alpha = 5\%$ separa aquilo que parece ser um evento normal daquilo me parece um tanto suspeito.

Dado as seis ocorrências de "coroa", e sabendo que a probabilidade disso ocorrer em H_0 é de 1.6%, e dado que eu defini um intervalo $\alpha = 5\%$, eu posso rejeitar a minha hipótese nula H_0 com 95% de certeza.

Logo, os passos para eu saber se a moeda é ou não honesta são:

1. Definir duas hipóteses, neste caso moeda honesta e moeda não honesta.
2. Calcular as probabilidades de ocorrência em H_0.
3. Definir um intervalo de confiança.
4. Comparar as probabilidades de ocorrência com o intervalo de confiança definido.

O que é um intervalo de confiança?

"Expressa o grau de incerteza relacionado a um certo efeito."

- **Physiotutors**

"É um teste estatístico feito de forma visual"

- **StatQuest**

"É mais uma maneira de entender os erros em potencial de uma estatística de uma amostra."

- **Practical Statistics for Data Scientists**

Uma maneira de definir o CI: é o intervalo que engloba x% da bootstrap sampling distribution de um estatística da amostra. Generalizando, pode-se dizer que um CI de x% ao redor de uma estimativa da amostra deve, em média, conter estimativas da amostra similares em x% das vezes, quando é feito um processo de amostragem semelhante.

Do ponto de vista da Ciência de Dados, um CI pode ser definido como uma ferramenta para se ter uma idéia do quão variável pode ser um resultado da amostra. Portanto, isso pode ser usado para quantificar (ou seja, dar uma estimativa) dos erros em potenciais, e também para entender se é preciso usar uma amostra maior no problema em questão.

Coisas para se tomar cuidado

- Quanto maior for o nível de confiança (CL) → porcentagem associada ao CI
  - Mais "gordo" será o intervalo
- Quanto menor for o tamanho da amostra
  - Mais "gordo" será o intervalo → neste caso maior será a incerteza
  - Então quanto mais confiabilidade tu precisas, e com menos dados disponíveis, maior deverá ser o CI. Faz sentido!

O que significa dizer:

- *Um intervalo de confiança (CI) de 95%?*

- *Bem, é simplesmente dizer que é um intervalo que contém 95% dos valores médios de uma determinada feature em N amostras.*
    - *De forma que qualquer coisa fora deste intervalo, ocorre apenas em 5% das vezes.*
    - *Logo, o p-value fora do CI: $p < 0.05$ (5%).*

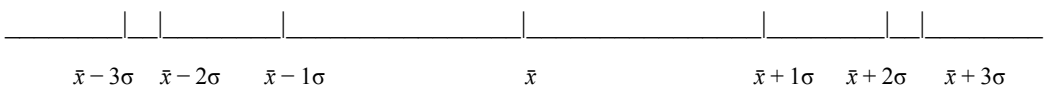
Referências

1. Confidence Intervals Explained (Calculation & Interpretation)
2. StatQuest: Confidence Intervals
3. Livro: Practical Statistics for Data Scientists

Como calcular um intervalo de confiança quando a distribuição amostral é Gaussiana?

Se a distribuição amostral é Gaussiana:

- Então ela é simetricamente centrada no seu valor médio.



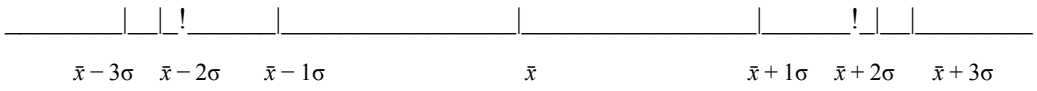
Sigma, é a o desvio padrão relacionado a medida. Portanto, os CI são calculados através do nível de dispersão da tua medida. Ou seja, do quanto um conjunto de medidas aleatórias se afasta do valor médio da amostra. Então, seria basicamente contar quantas vezes uma medida cai entre cada intervalo. Por exemplo:

- Para um CI de $1\,\sigma$: nº de medidas no intervalo $[\bar{x} - 1\sigma\,,\ \bar{x} + 1\sigma]$ dividido pelo tamanho total da amostra = 68%

Outro exemplo:

- Qual o CL correspondente ao CI de $1.5\,\sigma$?
    - Aqui entra o que se chama de z-score.

$\bar{x} - 1.5\sigma$ $\bar{x} + 1.5\sigma$



- Reformulando a pergunta: quantas medidas vão estar no intervalo $[\bar{x}-1.5\sigma, \bar{x}+1.5\sigma]$?
    - $Z = (\bar{x}+1.5\sigma - \bar{x}) / \sigma = 1.5 \rightarrow$ como é positivo, então devemos procurar o valor correspondente na tabela para z's positivos
        - $Z \rightarrow$ corresponde à 0.9332
            - 0.9332 - 0.5 = 0.4332
                - 0.4332 * 2 = 0.8664 → então um intervalo de confiança de 1.5 σ corresponde à um CL de 86.64%

O que é Bootstrapping e como pode ser usado para calcular intervalos de confiança?

"É uma maneira de estimar a distribuição amostral de uma estatística através da retirada de amostras adicionais com reposição, da própria amostra e recalcular a estatísitca para cada reamostragem."

- Practical Statistics for Data Scientists

Bootstrap é mais ou menos baseado na Lei dos Grandes Números que diz que com dados o suficiente a distribuição empírica é uma boa aproximação da distribuição real.

É como se fosse um processo de replicação da amostra. Só que ao invés de replicar a amostra, é feito um processo de reamostragem com reposição. Ou seja, retira uma amostra, calcula a estatística e guarda a info, devolve a amostra; retira uma segunda amostra, calcula a estatística e guarda a info, devolve a amostra; retira uma terceira amostra, calcula a estatística e guarda a info, devolve a amostra; …

Resumindo

É um processo de replicação de forma que tu consiga criar uma população hipotética suficientemente grande que englobe todo o "conhecimento" da tua amostra original.

Como calcular intervalos de confiança?

Basicamente tem dois métodos:

1. Percentile method: onde é usado 95% dos valores obtidos pelo bootstrapping, ou seja se calcula os limites do intervalo entre 2.5% e 97.5%

Por exemplo, seja * os valores obtidos. Todos os valores obtidos vão preencher todo o intervalo que representa a distribuição obtida. Ou falando de outra forma, os *'s vão preencher o intervalo correspondente à 0-100%.

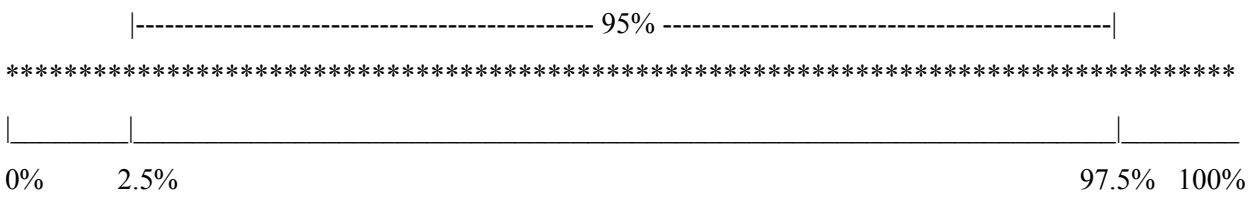
Representação de todas as medidas:

****
|_____|
0% 100%

Representando um intervalo contendo 95% das "medidas centrais"

****
|_____|_____
0% 2.5%

****
_____|_____
97.5%



2. Standard Error (SE): quais estatísticas sabemos após fazer bootstrapping?
    - Boots-média
    - Boots-população = tamanho das reamostras X nº de reamostras
    - Boots- σ

Para calcular o CI associado, é usual assumir um CL de 95%. Logo, é preciso calcular o z-score ($z_{\alpha/2}$), onde α refere-se ao CL. Neste caso z = 1.96. Desta forma, o CI é representado por: [média - 1.96*SE , média + 1.96*SE]

Referências

1. Livro: Practical Statistics for Data Scientists
2. Moderndive

O que são problemas de: Classificação, Regressão e Clusterização?

Classificação

Um problema de classificação é quando é os targets/outputs do ansatz utilizado para encontrar o P(Y|X), são variáveis categóricas. Ou seja, é quando o teu modelo de ML mapeia os teus inputs (X) em outputs discretos

Regressão

Um problema de regressão é quando o modelo de ML mapeia os inputs (X) em output reais ou contínuos.

Clusterização

Um problema de clusterização é quando os inputs não são mapeáveis, ou seja não são etiquetados. Desta forma, o problema de clusterização é simplesmente o de agrupar os inputs com relação à "sabores" de features semelhantes.

Quais são os passos em qualquer projeto de Machine Learning?

1. **Definir o problema:** é o estágio inicial do projeto, e provavelmente o mais importante de todos. E aqui em que o time passa a ganhar conhecimento sobre o problema e sobre a noção de business em que está inserido.
    a. Descrição do problema
    b. Solução ideal
    c. Entendimento e insights do problema
    d. Requerimentos técnicos
2. **Pesquisa:** é o que vai servir de base para o restante dos passos e também serve como alicerce para todo o restante do projeto.
    a. Estrutura de dados e fontes
    b. Formas de solução
    c. Arquitetura do modelo
    d. Pesquisa de algoritmos
    e. Requerimentos
        i. Hardware

ii. Software

3. **Mineração de dados (Aggregation, mining, scraping):** os dados precisam ser *diversos, unbiased, abundantes e representativos*.
4. **Preparação dos dados (Preprocessing, augmentation):** transformar os dados de maneira que podem ser para treino e validação.
5. **Implementação do modelo:** muito dificilmente a implementação vai ser baseada na construção de uma modelo desde o "início". Geralmente a implementação é feita por transfer learning.
6. **Treinamento:** é o momento quando os dados pré processados, e já separados em treino/validação/teste, são utilizados para treinar o modelo de ML que posteriormente será validado.
    a. Ajuste de hiperparâmetros: valores de regularização, taxa de aprendizado, número de camadas, profundidade (ou degraus)
    b. Parâmetros de rede.
7. **Validação:** é onde o modelo treinado vai ser posto "into trial". Alguma métricas que avaliam a performance são:
    a. Matriz de confusão → é o número de predições feitas pelo modela para cada classe.
    b. Accuracy → (TP + TN) / (TP + TN + FP + FN)
    c. Recall → TP / (TP + FN)
    d. Precision → TP / (TP + FP)
        i. Interessante que existe uma relação inversa entre recall e precision. Se um aumenta muito, o outro diminui bastante; se outro aumenta bastante, o um diminui muito.
    e. F1-score → 1 / 0.5(1/recall + 1/precision) → é a média harmônica não-ponderada.
    f. F0.5-score ou Fβ-score → quando é necessário dar mais peso para precision ($\beta = 0.5$) ou mais peso para o recall ($\beta = 2$).
    g. ROC ou AUC- curve → é um gráfico mostrando a performance de classificação do modelo plotando a taxa de (TP vs TN). A área abaixo da curva dá a probabilidade do modelo classificar um exemplo aleatório de TP mais vezes que um exemplo aleatório de TN.
8. **Fine-tuning:** é o processo de refinar o modelo através de modificações dos hiperparâmetros. O propósito disso é fazer com que o modelo perform melhor na validação.

9. **Teste:** é quando o modelo é de fato testado. Se não performar bem, então é preciso voltar ao passo 4, pois o conjunto de teste pode ser usado apenas uma vez. Do contrário, nossa análise e previsão passam a ser enviesadas, uma vez que já teremos informação sobre o conjunto de teste.
10. **Inferência:** se tudo ocorreu bem nos passos anteriores, então é hora de aplicar o modelo em dados reais.
11. **Deploy:** é quando o modelo nasce e vai ao mundo!

Referências:

1. 10 Stages of a ML Learning Project in 2020

O que é: Underfitting e Overfitting?

Underfitting

É quando o modelo falha no processo de aprender os padrões do dataset. E portanto não performa bem no treino e nem no teste. O modelo não captou a complexidade dos dados.

Overfitting

É quando o modelo fita absurdamente bem os dados de treino, mas falha miseravelmente nos dados de teste. Isso ocorre quando os dados usado para treino não tenham a mesma distribuição que os dados usados para teste. O modelo captou muito bem a complexidade dos dados, mas não tem capacidade de generalizar para dados reais.

O que fazer quando o modelo: Underfitou ou Overfitou?

Underfitou

Pode se introduzir mais features que podem ser relevantes para o problema e incluir isso no algoritmo que foi usado antes, ou então simplesmente mudar o algoritmo ou o modelo para algum que seja mais apropriado para o problema que tu tá tentando resolver.

Overfitou

Neste caso, pode ser resolvido diminuindo o número de features presentes no conjunto de treino ou então diminuindo a complexidade do modelo. Processo de regularização → diminuir a complexidade do modelo. Ex. Decision tree: setar uma profundidade máxima.

Cite métricas para avaliação de modelos de Regressão e quando usar cada uma.

Erro quadrático médio (MSE)

Que é simplesmente a média do quadrado da diferença entre os valores de target e os preditos pelo modelo. Ela é bastante utilizada, mas é preciso tomar cuidado pois ela deixa bastante evidente a presença de outliers nos dados. No sentido de que como ela pega o quadrado da diferença então ela acaba penalizando até mesmo o menor dos erros, e portanta superestima o quão ruim o modelo pode ser.

Raiz quadrada do erro quadrático médio (RMSE)

Talvez a métrica de avaliação mais usada em modelos de regressão, e é simplesmente a raiz quadrada da média do quadrado da diferença entre os valores de target e os preditos pelo modelo. Ela é bastante utilizada quando se busca penalizar grandes erros de predição do modelo.

Erro absoluto médio (MAE)

É a diferença absoluta entre os valores de target e os preditos pelo modelo. Ela é mais robusta para se lidar com outliers e não penaliza tanto os erros quando a MSE. Ela não é indicada para problemas onde tu queres prestar mais atenção nos outliers.

Erro absoluto médio ponderado (MAPE)

É muito parecido com a MAE exceto que a diferença entre os valores de target e preditos são agora ponderamos pelo valor de target.

R^2

Essa métrica ajuda a comparar o modelo "do jeito que está agora" com um baseline constante (que é definido pela média das features), e com isso dá uma idéia do quão melhor é o modelo do que simplesmente "a própria média".

Adjusted-R^2

Tem praticamente a mesma finalidade que o R^2, porém existem algumas melhorias nesta métrica. Existe um problema com o R^2 pois quanto mais features se adiciona ao modelo mais cresce o valor da métrica sem que necessariamente haja uma melhora no modelo. O R^2-ajustado adiciona um peso para cada feature adicionada.

Referências

1. Regression: An Explanation of Regression Metrics and What Can Go Wrong

Cite métricas para avaliação de modelos de Classificação e quando usar cada uma.

Confusion Matrix ou Error Matrix

É uma matriz que descreve toda a performance de classificação do modelo em um conjunto cujo os valores verdadeiros são conhecidos. Ela nos dá a informação de todas as possíveis variações de classificação:

- True Positive (TP) → Actual (y) Positive and Predicted ($\hat{y}$) Positive
- True Negative (TN) → Actual (y) Negative and Predicted ($\hat{y}$) Negative
- False Positive (FP) or Type 1 error → Actual (y) Negative but Predicted ($\hat{y}$) Positive
- False Negative (FN) or Type 2 error → Actual (y)Positive but Predicted ($\hat{y}$) Negative

É um resumo da performance de classificação do algoritmo.

Accuracy

Simplesmente é a razão entre os acertos do modelo pelo total. No entanto esta métrica não leva em conta os casos de classificação errada (misclassification), ou seja em situações no qual é muito importante ter um controle dos erros de tipo 1 e 2, uma boa acurácia não é garantia de que o modelo vai performar bem.

Cumulative Accuracy Profile

É uma curva que representa o número cumulativo de resultados positivos no eixo-y versus o número cumulativo de um parâmetro de classificação. Quanto maior for a área abaixo dessa curva, melhor é a performance do modelo.

F1-Score

Existe uma relação entre Recall e Precision, quando um aumenta o outro diminui, e quando o outro aumento o um diminui. Então o F1-score é a média harmônica não ponderada entre Precision e Recall. Varia no intervalo [0,1] e basicamente nos diz quão preciso o classificador é e ao mesmo tempo quão robusto, ou seja quantas previsões são classificadas corretas e quantas são misclassification.

- Precision → é a razão entre o número de True Positives e o número total de elementos não classificados como Positivos. (É o que eu sei dividido pelo o que realmente é)
- Recall → é a razão entre o número de True Positives e o número total de elementos que pertencem a classe dos Positivos. (É o que eu sei dividido pelo o que eu acho que é)

ROC ou AUC

É um gráfico mostrando a performance de classificação do modelo plotando Recall (Sensitivity) vs (1 - Specificity), ou seja a taxa de TP vs a taxa de FP (que é a taxa das instâncias negativas que foram incorretamente classificadas como positivas). E o que faz a curva ser plotada é justamente as taxas que foram calculadas para cada threshold, então é basicamente isso que monta a curva.

E área abaixo da curva dá a probabilidade do modelo classificar um exemplo aleatório de TP mais vezes que um exemplo aleatório de TN.

A ROC indica o quão bem separada as probabilidade das classes positivas estão das classes negativas.

LogLoss

É um tipo de métrica que funciona bem em problemas de multi-classes e que penaliza classificações falsas. No geral minimizar LogLoss retorna uma maior acurácia.

Referências

1. Metrics to Evaluate Your Machine Larning Algorithm
2. Cumulative Accuracy Profile
3. Precision and Recall
4. Recall, Precision, F1, ROC, AUC and Everything

Cite métricas para avaliação de modelos de Clusterização e quando usar cada uma.

Fora o Elbow-method, que não é uma métrica, é só uma maneira de tentar encontrar um número ótimo de clusters aplicando o k-means (k-means ++ → para lidar com o k-means random initialization trap). Eu conheço:

Within Clusters Sum of Squares (WCSS)

É a soma do quadrado da distância de cada ponto de um cluster até o seu centróide. Então o número ótimo de clusters é o que minimiza WCSS. No entanto, tem uma pegadinha aqui, pois quanto mais clusters for adicionado, mais WCSS vai diminuir, simplesmente pois cada ponto pode corresponder a um cluster. Desta forma, o valor mínimo de WCSS para o número de clusters igual ao número de ponto seria zero. O que não faz sentido! Por isso que se usa o WCSS associado ao Elbow-method, é estritamente visual. Pois, haverá uma grande diferença entre os valores de WCSS a medida que se adiciona os clusters, até que essa diferença passa a se tornar não tão significativo. Então este será o valor ótimo de k. Então a resposta para k vai sempre depender do problema em questão, e se de alguma forma tu “te inspirou” pelo resultado.

Silhouette Coefficient

É um coeficiente que diz o quão bem posicionado é cada ponto. Se S(i), S é o coeficiente e i é o ponto, for:

- Próximo de 0: então o ponto está muito próximo da intersecção de dois clusters
- Próximo de -1: então o ponto deveria ter sido colocado em outro cluster
- Próximo de 1: então o ponto está bem posicionado no cluster

É uma métrica muito custosa de ser pois é preciso calcular a distância de cada ponto i dos outros n-1 pontos para cada i. Então, para um conjunto muito grande pode ser impraticável o cálculo. Grau de complexidade $O(n^2)$.

Referências

1. Elbow Method (clustering)
2. Assessments Metrics For Clustering Algorithms

Como lidar com dados faltantes?

Depende do tipo de dados, se são dados numéricos, texto, imagens… Se faz sentido simplesmente excluir todas as informações das outras features relacionadas a esses dados faltantes, ou então se tem sentido gerar pontos com o valor médio da feature (se for numérica). Mas no geral, se mais de 30% da feature corresponde a dados faltantes, então se exclui a feature do modelo, pois talvez o impacto de mantê-la seja muito pior do que o de excluí-la.

Descreva em palavras os seguintes algoritmos para classificação: Logistic Regression, Naive Bayes, Decision Tree, Random Forest, SVM, Neural Networks e XGBoost.

Logistic Regression

É um algoritmo que estima a probabilidade de classificação de um target. É definido um threshold (0.5, por exemplo), então se a probabilidade predita for maior que 50%, então diz-se que o target foi classificado na categoria 1, se for menor que 50%, então foi classificado na categoria 0. Fazendo uma comparação com Física, Regressão Logística é a mesma coisa que uma distribuição de Fermi-Dirac.

Naive Bayes

Todo modelo de ML é um Ansatz diferente para tu econtrar P(target | feature). Esse método é usado em problemas de múltiplas classes e o ponto de partida é a Forma de Bayes. Porém fazendo uma suposição bastante simples, que é dizer que as features são independentes. Com isso tu pode usar a regra da cadeia para escrever as probabilidades condicionais para cada feature/classe em termos de uma expansão.

Decision Tree

É um algoritmo que faz uma série de perguntas para o conjunto de dados. Dependendo das respostas, o algoritmo faz cortes ortogonais nos dados separando da melhor maneira possível cada um na sua respectiva classe predita. Minimiza a entropia ou diminui o índice de impureza (Gini).

Random Forest

É um ensemble de Decision Trees treinados via bagging. São várias DT rodando em subconjuntos do conjunto de treino, e no final a RF seria mais ou menos como se fosse a média de todas as DT.

SVM

É um algoritmo versátil que serve tanto para problemas de classificação quanto regressão. No caso de classificação, o SVM tenta criar uma linha de separação, uma fronteira entre as classes, que é chamada de decision boundary, de forma a maximizar o gap, ou seja a distância entre os pontos de cada classe dessa fronteira de separação. Já no caso da regressão, é praticamente o mesmo mecanismo de criar a decision boundary. Só que agora a idéia é captar/fitar o máximo de instâncias dentro da decision boundary.

Neural Network

É um algoritmo que funciona em camadas onde cada uma é responsável por aprender uma determinada feature do input e passar isso adiante para a camada próxima. Então é uma rede com várias camadas e cada camada tem uma série de neurônios. Cada neurônios têm "uma probabilidade de" serem ativados e então interagir com neurônios de outras camadas. O que determina se eles vão ou não ser ativados é uma coisa chamada de função de ativação. Comparando com a Física, isso é mais ou menos um Modelo de Ising com interação de primeiros vizinhos.

XGBoost

É um algoritmo baseado decision trees, especialmente construído pensando em rapidez e performance de código e modelo de ML, que usa gradient boosting. Ou seja, o modelo é construído refitando os erros do modelo anterior de forma que a função custo seja atualizada e otimize cada boost,

Referências

1. A Gentle Introduction to XGBoost

2. XGBoost Algorithm: Long She May Rein!

Descreva em palavras os seguintes algoritmos para clusterização: k-means, Hierarchical Clustering e DBSCAN.

k-means

É um algoritmo que ajuda a agrupar o data set em features semelhantes, mas que também possibilita determinar o número ótimo de agrupamentos em um conjunto de dados. Por exemplo para um k = 2, basicamente a intuição é que é escolhido dois pontos aleatórios que serão o input para a posição dos centróides de cada cluster. O passo seguinte é então calcular a distância de cada ponto ao centróide mais próximo. Em seguida se determina a posição dos centróides, quase como que fosse o centro de massa do cluster, e repete o processo de calcular a distância de cada ponto ao centróides e associar o ponto ao centróide mais próximo. Esse processo é feito quantas vezes for necessário até que todos os pontos sejam associados a um cluster.

Hierarchical Clustering

Tem duas abordagens Agglomerative e Divisive, eu conheço mais ou menos a primeira e a outra praticamente nada da segunda. A idéia de Agglomerative Clusterings é tu fazer de cada ponto um cluster. Então tu pega o pontos mais próximos, e fazem deles um cluster. Na seqüência, tu pega os dois clusters mais próximos, e transforma em um só. E repete esse processo até restar apenas um único cluster. Agora a pergunta é: Por que fazer todo esse processo se no final vamos ter apenas um cluster? E a resposta é simples, pois HC guarda a informação de todo o processo de aglomeração no formato de dendrogramas. E é através da análise dos dendrogramas que se pode concluir o número de clusters presentes do data set. Então, dendrogramas é mais ou menos como se fosse a memória do algoritmo de HC.

DBSCAN

É um algoritmo de agrupamento de dados baseado em densidade. Então tem duas coisas que é preciso levar em consideração aqui (e eu acho que são só essas mesmo). Uma é a distância mínima entre dois pontos para considerar que eles fazem parte do mesmo cluster (eps), e o número de pontos mínimo para considerar uma região sendo densa (minPoints). E é praticamente isso. Todo e qualquer ponto que não for agrupado é considerado como ruído/outlier.

Quais cuidado devemos ter para preparar os dados para cada um dos algoritmos acima?

Logistic Regression

Como preparar dados de texto para ML?

Não dá para diretamente modelar em cima de dados de texto crus. É preciso passar por uma série de passos que podem ser resumidos dentro de duas etapas antes de se montar o modelo de ML.

1. Feature extraction ou encoding:
    a. Que resumindo significa criar um Bag of Words onde se vai vetorizar o texto, já que o computador entende somente dados binários.
        i. Splitar o texto em uma lista de tokens
        ii. Criar dicionário
2. Managing Vocabulary:
    a. Sparse matrix
    b. Remover pontuações, ou tokens que não pertence ao alfabeto
    c. Transformar tudo para lowercase letters.
    d. Remover stopwords
    e. Montar um dicionário de gírias (caso seja necessário)

f. Arrumar palavras erradas.

O que é Feature Engineering?

Isso é diretamente linkado com aquele ditado “Trash in, trash out”. O que diz que a qualidade do teu resultado depende inteiramente da qualidade dos teus dados. Então Feature Engineering é o processo que une as etapas de “entendimento do problema” e capturar a essência e importância das features do dataset. São dois passos, basicamente:

1. Selecionar as features mais relevantes entre todas as features disponíveis para ser fazer o treino do modelo de ML
2. Combinar features, se necessário e se isso fizer sentido dado o problema, para produzir uma outra feature que seja mais relevante. Por exemplo usando Dimensionality Reduction: eu já ouvi falar de PCA, que é um algoritmo para identificar e detectar correlação entre variáveis (features). Então, se uma forte correlação é encontrada, então isso pode ser usado para reduzir a dimensão das features.

O que é Covariate Shift?

Se refere a mudança do tipo de distribuição entre o conjunto de teste e o conjunto de treino. Seria como tu criar um modelo e treinar ele em um determinado conjunto de treino e tentar aplicar esse mesmo classificador treinado num conjunto de teste totalmente diferente do conjunto de treino.

Tu construiu um modelo de claassificação que tem boa acurácia. O teu chefe quer saber de quanto em quanto tempo o modelo precisa ser retreinado. Como fazer para descobir?

Supondo que tu tem um modelo que roda todo o mês e faz previsões para o mês seguinte. Então tu pode treinar o modelo para o mês M e prever o mês M+1. Passado o mês M+1, observa os erros e guarda a informação. E faz o mesmo processo para M+2, M+3,... e guarda as informações sobre os erros de previsão. Passado um tempo, é natural que o modelo passe a performar mal, com isso dá para ter uma idéia de quanto em quanto tempo precisa ser retreinado.

Então, usa um modelo fixo. Compra as previsões os os números reais e analisa o tempo que leva para o modelo deteriorar.

O que são Ensemble Methods?

Resumindo, é pegar vários algoritmos de ML e juntar para formar um algoritmo maior que tira vantagem dos outros algoritmos e otimiza a performance de processo.

- Bagging: É usar o mesmo algoritmo de ML para treinar os classificadores, porém em subconjuntos aleatórios do conjunto de treino. Resumindo, é fazer bootstrap com o mesmo algoritmo de ML.
- Stacking: Seria mais ou menos como usar a saída de um modelo como input para outro.

- Boosting: A idéia é treinar o classificador de uma maneira sequencial. Ou seja, fazer refits dos erros do modelo anterior

Qual a diferença entre Bagging e Boosting?

Bagging

O nome é bem intuitivo. Bagging é juntar um apanhado de algoritmos de ML do mesmo tipo e fazer com que cada um deles seja aplicado em resamples com reposição do conjunto de treino. É aplicar bootstrap usando o mesmo algoritmo de ML.

Boosting

É combinar vários algoritmos e treinar o classificador de forma sequencial, de maneira que cada algoritmo corrija ou faça um refit sobre os erros do anterior.

O que é o problema de desbalanceio em problemas de classificação?

Se refere aos casos em que se tem números muito diferentes de instâncias para diferentes categorias. E isso acaba se tornando um problemão em ML pois um conjunto de dados desbalanceado vai introduzir bias na previsão do modelo no sentido de “dar preferência para a categoria mais comum”.

Como lidar com o desbalanceamento em problemas de classificação?

Teoricamente tem duas maneiras de se lidar com esse problema:

Undersampling

Selecionar aleatoriamente um subconjunto com amostras da classe com mais instâncias, que seja mais ou menos do mesmo tamanho da classe com menos instâncias. Só que a desvantagem é que muita informação relevante pode acabar sendo deixado de lado nesse processo de left-out.

Oversampling

Duplicar aleatoriamente amostras da classe com menos instâncias até ficar equivalente com o tamanho da classe com mais instâncias. Uma outra forma e gerar dados novos a partir de interpolação dos pontos mais próximos. Só que esse processo de oversampling pode levar a overfitting do modelo já que é mais provável ter as mesmas amostras no treino.

Referências:

1. Dealing with unbalanced data in ML

O que é backtesting?

É um tipo de teste que se faz em dados históricos, a fim de prever o que pode acontecer no futuro. Por exemplo, é o que esse cara aqui tentou fazer. Ele praticamente previu a crise do corona. O certo seria ele ter cortado os dados em antes do corona e agora no corona. E aplicar backtesting (moving window ou cumulative window) nos dados pré-corona. O que ele fez foi simplesmente um split aleatório. Ou seja tem leakage aí.

O que são Convolutional Neural Networks? Como funcionam? E Recurrent Neural Networks?

É

O que é o problema de Leakage em ML?

É quando vaza informação do conjunto de teste para o conjunto de treino. Por exemplo o cara do LinkedIn que previu a crise do corona usando dados históricos misturados com dados do corona.

Explique o Bias x Variance tradeoff.

Bias é uma medida de quão restritivo é o modelo, já variance é uma medida da variabilidade do modelo. O ideal é que se tenha um modelo que capture bem a complexidade do conjunto de dados no treino e generalize bem no teste. Só que isso é praticamente impossível e é justamente essa relação entre Bias e Variance, ou seja restrição e variabilidade, que impede que os algoritmos de ML generalizar tão bem além do treino. Então resumindo, é o conflito de se tentar minimizar bias e variance ao mesmo tempo. Um modelo com muito bias e pouca variance vai underfitar, já um com pouco bias e muito variance vai overfitar.

Referências

1. Bias-variance tradeoff
2. Bias-Variance Tradeoff To Avoid Under/Overftting

O que é Interpretabilidade em algoritmos de ML? Cite algumas técnicas para Interpretabilidade.

Não sei se é bem isso, mas é o grau em que humanos conseguem consistentemente prever o resultado de um modelo. Então, quanto mais interpretável for um modelo de ML, mais fácil será entender o porquê de certas decisões ou previsões.

Interpretability dá a habilidade para os modelos de ML explicar ou apresentar o seu comportamento de forma que os humanos sejam capazes de entender.

Interpretabilidade não é necessária quando o problema é bem conhecido ou quando o modelo não tem um impacto significante.

Técnicas

- Interpretabilidade intrínseca: é a construção de modelos auto explicativos que incorporam interpretabilidade da sua estrutura. Ex. uma Decision Tree
- Post-hoc: é a construção de um segundo modelo para fornecer explicações para um modelo que já existe.

Referências

1. Interpretable Machine Learning
2. Techniques for interpretable ML

As evidências que coletamos fazem com que a nossa hipótese nula parecer ridícula.

Machine Learning: Validation vs Testing