1ª. ATIVIDADE

ALUNO				**MATRÍCULA**	
DISCIPLINA	MACHINE LEARNING			**DATA**	
PROFESSOR	LINDENBERG ANDRADE			**TIPO DE AVALIAÇÃO**	ATIVIDADE I
TURMA	4NA	**CÓDIGO DA TURMA**	NLN0400104NNA	**NOTA**	

Fundamentos de Machine Learning - Parâmetros e Hiperparâmetros

1. Introdução ao Machine Learning

Machine Learning (Aprendizado de Máquina) é um campo da inteligência artificial que permite aos sistemas aprenderem com dados, identificar padrões e tomar decisões com mínima intervenção humana. Em vez de serem explicitamente programados para cada tarefa, os algoritmos de Machine Learning constroem um modelo a partir de dados de entrada, que então pode ser usado para fazer previsões ou classificações em novos dados.

Existem diferentes tipos de aprendizado de máquina, incluindo:

Aprendizado Supervisionado: O modelo é treinado em um conjunto de dados rotulado, onde cada exemplo de entrada tem um resultado desejado correspondente. O objetivo é aprender uma função que mapeia entradas para saídas. Exemplos incluem regressão (prever um valor contínuo) e classificação (prever uma categoria discreta).

Aprendizado Não Supervisionado: O modelo é treinado em um conjunto de dados não rotulado, onde o objetivo é encontrar padrões ou estruturas ocultas nos dados. Exemplos incluem agrupamento (clusterização) e redução de dimensionalidade.

Aprendizado por Reforço: Um agente aprende a tomar decisões em um ambiente para maximizar uma recompensa. É comumente usado em robótica e jogos.

Independentemente do tipo de aprendizado, a construção de um modelo de Machine Learning envolve a escolha de um algoritmo, o treinamento desse algoritmo com dados e a avaliação de seu desempenho. Durante esse rocesso, dois conceitos cruciais surgem: **parâmetros** e **hiperparâmetros**.

2. Parâmetros de Modelo

Os **parâmetros** de um modelo de Machine Learning são as variáveis internas que o modelo aprende a partir dos dados de treinamento. Eles são ajustados durante o processo de treinamento para que o modelo possa fazer previsões precisas. Em essência, os parâmetros são o que o modelo 'memoriza' ou 'aprende' sobre os dados para realizar sua tarefa. Eles não são definidos manualmente pelo engenheiro de Machine Learning, mas sim descobertos pelo algoritmo de aprendizado.

Características dos Parâmetros:

Aprendidos pelos Dados: São ajustados automaticamente pelo algoritmo de otimização (e.g., Gradiente Descendente) durante o treinamento.

Internos ao Modelo: Fazem parte da estrutura interna do modelo e definem como ele mapeia as entradas para as saídas.

Essenciais para Previsões: Uma vez que o modelo é treinado, esses parâmetros são fixos e usados para fazer previsões em novos dados.

Exemplos de Parâmetros em Modelos Comuns:

Regressão Linear: Em um modelo de regressão linear simples, a equação é geralmente $y = mx + b$, onde m é a inclinação (coeficiente) e b é o intercepto. m e b são os parâmetros do modelo que são aprendidos a partir dos dados para encontrar a linha que melhor se ajusta aos pontos

de dados.

Redes Neurais Artificiais: Em redes neurais, os parâmetros são os **pesos** e **vieses** (biases) das conexões entre os neurônios. Durante o treinamento, esses pesos e vieses são ajustados para minimizar a função de perda, permitindo que a rede aprenda padrões complexos nos dados.

Máquinas de Vetores de Suporte (SVM): Os parâmetros de um SVM incluem os vetores de suporte e os pesos associados a eles, que definem o hiperplano de separação.

Árvores de Decisão: Embora as árvores de decisão não tenham parâmetros no sentido tradicional de pesos e vieses, a estrutura da árvore (quais características
usar para dividir, em que pontos e quais são as folhas finais) é aprendida a partir dos dados e pode ser considerada análoga aos parâmetros.

Em resumo, os parâmetros são o coração do modelo de Machine Learning, moldando sua capacidade de generalizar e fazer previsões com base nos dados que ele viu durante o treinamento. A qualidade desses parâmetros é diretamente influenciada pela quantidade e qualidade dos dados de treinamento, bem como pela escolha do algoritmo e seus hiperparâmetros.

3. Hiperparâmetros

Ao contrário dos parâmetros, os **hiperparâmetros** são configurações externas de um modelo de Machine Learning que não são aprendidas a partir dos dados. Eles são definidos antes do processo de treinamento e controlam como o modelo é treinado e como ele aprende. A escolha dos hiperparâmetros pode ter um impacto significativo no desempenho do modelo, na sua capacidade de generalização e na velocidade de treinamento.

Características dos Hiperparâmetros:

Definidos Antes do Treinamento: São especificados manualmente pelo engenheiro de Machine Learning ou por

técnicas de otimização de hiperparâmetros.

Externos ao Modelo: Não são ajustados pelo algoritmo de otimização durante o treinamento.

Controlam o Processo de Aprendizado: Influenciam a arquitetura do modelo, a complexidade e a forma como os parâmetros são aprendidos.

Importância do Ajuste de Hiperparâmetros:

A escolha correta dos hiperparâmetros é crucial para o sucesso de um modelo de Machine Learning. Hiperparâmetros mal ajustados podem levar a:

Underfitting (Subajuste): O modelo é muito simples para capturar os padrões nos dados de treinamento, resultando em baixo desempenho tanto nos dados de treinamento quanto nos dados de teste. Isso pode ocorrer se a complexidade do modelo for muito baixa (e.g., uma árvore de decisão muito rasa).
Overfitting (Sobreajuste): O modelo aprende os dados de treinamento de forma muito específica, incluindo o ruído, e não consegue generalizar bem para novos dados. Isso pode ocorrer se a complexidade do modelo for muito alta (e.g., uma árvore de decisão muito profunda).

O objetivo do ajuste de hiperparâmetros é encontrar a combinação ideal que minimize o erro de generalização do modelo, ou seja, que permita ao modelo ter um bom desempenho em dados não vistos.

Exemplos de Hiperparâmetros em Modelos Comuns:

Regressão Linear (com Regularização):

`alpha` (para Ridge ou Lasso Regression): Controla a força da regularização. Um valor alto de `alpha` aumenta a regularização, o que pode ajudar a prevenir o overfitting.

Árvores de Decisão e Random Forests:

`max_depth` : A profundidade máxima da árvore. Controla a complexidade da árvore. Uma profundidade muito grande pode

levar a overfitting.

`min_samples_split` : O número mínimo de amostras necessárias para dividir um nó interno. Valores maiores podem suavizar o modelo e reduzir o overfitting.

`n_estimators` (para Random Forest): O número de árvores no ensemble. Mais árvores geralmente melhoram o desempenho, mas aumentam o tempo de computação.

Máquinas de Vetores de Suporte (SVM):

`C` : O parâmetro de regularização. Controla a penalidade por erros de classificação. Um `C` pequeno favorece margens maiores (mais simples), enquanto um `C` grande favorece margens menores (mais complexas).

`kernel` : O tipo de função kernel (e.g., linear, polinomial, RBF). Define a transformação dos dados para encontrar um hiperplano de separação.

`gamma` (para kernel RBF): Define a influência de uma única amostra de treinamento. Valores altos significam que apenas pontos próximos são considerados, o que pode levar a overfitting.

Redes Neurais Artificiais:

`learning_rate` : A taxa na qual os pesos do modelo são atualizados durante o treinamento. Uma taxa muito alta pode fazer com que o treinamento oscile e não convirja; uma taxa muito baixa pode tornar o treinamento muito lento.

`batch_size` : O número de amostras de treinamento processadas antes que os parâmetros do modelo sejam atualizados. Afeta a estabilidade do gradiente e a velocidade de treinamento.

`num_epochs` : O número de vezes que o algoritmo de treinamento passará por todo o conjunto de dados de

treinamento. Mais épocas podem levar a overfitting.

`num_layers` / `num_neurons` : O número de camadas ocultas e o número de neurônios em cada camada. Definem a capacidade do modelo.

O processo de encontrar os melhores hiperparâmetros é chamado de **otimização de hiperparâmetros** ou **ajuste de hiperparâmetros**. Isso geralmente envolve a experimentação com diferentes combinações de valores de hiperparâmetros e a avaliação do desempenho do modelo usando técnicas como validação cruzada.

4. Métodos de Busca de Hiperparâmetros

Encontrar a combinação ideal de hiperparâmetros é um desafio, pois o espaço de busca pode ser vasto e a avaliação de cada combinação pode ser computacionalmente cara. Diversas estratégias foram desenvolvidas para otimizar esse processo:

4.1. Busca em Grade (Grid Search)

A **Busca em Grade** é o método mais direto e exaustivo para otimização de hiperparâmetros. Nela, você define um conjunto discreto de valores para cada hiperparâmetro que deseja otimizar. O algoritmo então treina e avalia o modelo para todas as combinações possíveis desses valores. A combinação que resulta no melhor desempenho (geralmente medido por validação cruzada) é selecionada.

Vantagens: Simples de implementar e entender. Garante que a melhor combinação dentro do espaço de busca definido será encontrada.
Desvantagens: Custo Computacional Elevado: O número de combinações cresce exponencialmente com o número de hiperparâmetros e o número de valores para cada um. Isso pode tornar a busca inviável para muitos hiperparâmetros ou para um grande número de valores. **Ineficiente em Espaços de Busca Grandes**: Pode gastar muito tempo explorando regiões do espaço de busca que não são promissoras.

Exemplo: Se você tem dois hiperparâmetros, `max_depth` com valores [3, 5, 7] e `n_estimators` com valores [100, 200, 300] , o Grid Search testará 3 * 3 = 9 combinações.

4.2. Busca Aleatória (Random Search)

A **Busca Aleatória** difere da Busca em Grade por não testar todas as combinações. Em vez disso, ela amostra aleatoriamente combinações de hiperparâmetros de distribuições especificadas para cada hiperparâmetro. O número de amostras é predefinido. Embora não garanta a exploração de todas as combinações, pesquisas mostram que a Busca Aleatória pode ser mais eficiente que a Busca em Grade em espaços de busca de alta dimensionalidade, pois tem maior probabilidade de encontrar combinações promissoras que o Grid Search poderia perder se os valores importantes não estivessem na grade definida.

Vantagens: Mais Eficiente que Grid Search: Especialmente em espaços de busca de alta dimensionalidade, onde alguns hiperparâmetros podem ter mais impacto que outros. **Mais Rápido**: Você pode controlar o número de iterações, tornando-o mais rápido que o Grid Search para o mesmo número de combinações testadas.

Desvantagens: Não garante a descoberta da melhor combinação, mas tem uma boa probabilidade de encontrar uma combinação próxima do ótimo.

Exemplo: Em vez de definir valores discretos, você pode definir um intervalo para `max_depth` (e.g., de 1 a 20) e para `n_estimators` (e.g., de 50 a 500). A Busca Aleatória selecionará aleatoriamente N combinações dentro desses intervalos.

4.3. Otimização Bayesiana

A **Otimização Bayesiana** é uma abordagem mais sofisticada que constrói um modelo probabilístico (chamado de função substituta ou surrogate function) do desempenho do modelo em relação aos hiperparâmetros. Esse modelo é usado para guiar a busca, sugerindo a próxima combinação de

hiperparâmetros a ser avaliada com base nas
avaliações anteriores. O objetivo é minimizar o número de avaliações caras do modelo real.

Vantagens: Mais Eficiente: Geralmente converge para a melhor combinação de hiperparâmetros em menos iterações do que Grid Search ou Random Search. **Lida bem com espaços de busca complexos**: Adequada para funções de custo ruidosas ou não convexas.

Desvantagens: Mais complexa de implementar e entender. Pode ser mais lenta por iteração devido à construção do modelo probabilístico.

4.4. Outros Métodos

Existem outros métodos, como algoritmos genéticos, otimização por enxame de partículas e otimização baseada em gradiente, que também podem ser usados para otimização de hiperparâmetros, cada um com suas próprias vantagens e desvantagens.

A escolha do método de otimização de hiperparâmetros depende de fatores como o número de hiperparâmetros, o tempo computacional disponível e a complexidade do problema.

Exercícios:

1) Qual a principal diferença entre parâmetros e hiperparâmetros em um modelo de Machine Learning? Dê um exemplo de cada.
2) Explique o que é overfitting e underfitting. Como a escolha inadequada de hiperparâmetros pode levar a esses problemas?
3) Descreva brevemente a diferença entre Grid Search e Random Search para otimização de hiperparâmetros. Quais as vantagens e desvantagens de cada um?
4) Em uma Rede Neural Artificial, quais seriam exemplos de parâmetros e hiperparâmetros?
5) Por que a validação cruzada é importante no processo de ajuste de hiperparâmetros?

1ª. ATIVIDADE

Referências:

- FILHO, Oscar Gabriel. Inteligência Artificial e Aprendizagem de Máquina: aspectos teóricos e aplicações. 1. ed. São Paulo: Edgard Blücher, 2023. 462 p. ISBN 978-65-5506-620-3
- HUYEN, Chip. Projetando sistemas de Machine Learning: processo interativo para aplicações prontas para produção. Tradução de Cibelle Ravaglia. 1. ed. São Paulo: Alta Books, 2024. 384 p. ISBN 978-8550819679
- LEKHANSH, T. Parameters vs Hyperparameters in Machine Learning. Medium, 2 jul. 2024. Disponível em: https://medium.com/@tyagi.lekhansh/parameters-vs-hiperparameters-in-machine-learning-56241f3b14e5. Acesso em: 18 ago. 2025.
- OURNAL OF ENGINEERING RESEARCH AND REVIEWS. Hyperparameter Tuning in Machine Learning: A Comprehensive Review. Journal of Engineering Research and Reviews, [s. l.], v. 6, n. 2, p. 1-10, 7 jun. 2024. Disponível em: https://journaljerr.com/index.php/JERR/article/view/1188. Acesso em: 18 ago. 2025.