Guia para Busca de Parâmetros

Prof. Leandro M. Almeida

1 Objetivo

Este guia tem como finalidade orientar a realização de uma busca sistemática de hiperparâmetros em modelos de classificação supervisionados. O foco é maximizar o desempenho preditivo, garantindo boas práticas metodológicas de validação e evitando problemas comuns, como vazamento de dados (*data leakage*) e avaliação enviesada.

2 Base de Dados

- **Origem:** Utilizar a base de dados definida para o grupo em atividades anteriores.
- **Características:** Descrever brevemente o número de instâncias, número de atributos, balanceamento das classes e/ou outras informações relevantes para contextualizar o problema.

3 Preparação dos Dados

1. **Carregamento do Dataset:** Carregar a base de dados do grupo
2. **Divisão em Conjuntos de Treino e Teste:**
   - Reservar 80% dos dados para treinamento e 20% para teste.
   - Utilizar estratificação de classes para manter a proporção das classes (ver link no classroom a respeito).
3. **Pré-processamento:**
   - Aplicar normalização, padronização ou outros métodos adequados para os atributos numéricos.

- Realizar *encoding* (por exemplo, *one-hot*) em variáveis categóricas, se necessário.
- Tratar valores ausentes de forma consistente (imputação).
- Ajustar qualquer transformação apenas no conjunto de treinamento e replicar no conjunto de teste para evitar *data leakage*.

4 Seleção de Modelos

- Selecionar **algoritmos de classificação** distintos, conforme definido na atividade principal.
- Listar, para cada modelo, os principais hiperparâmetros que serão ajustados (por exemplo, *n_estimators*, *max_depth*, *C* etc.).

5 Busca de Hiperparâmetros

5.1 Método de Busca

- Utilizar o ***RandomizedSearchCV*** (ou biblioteca equivalente) do *scikit-learn*.
- Executar, no mínimo, **20 iterações** para cada modelo.

5.2 Validação Cruzada

- Durante a busca, empregar **validação cruzada estratificada k-fold** (por exemplo, *k=5*) para avaliar o desempenho.
- Garantir a estratificação para lidar com possíveis desbalanceamentos.

5.3 Definição do Espaço de Hiperparâmetros

- Determinar faixas ou distribuições para cada hiperparâmetro. Exemplos:
  - **SVM:** {`C`: distribuição log-uniforme, `kernel`: [`rbf`, `linear`]}.
  - **Random Forest:** {`n_estimators`: distribuição inteira de 50 a 300, `max_depth`: distribuição inteira de 3 a 20}.

5.4 Registro de Desempenho

- Armazenar o histórico de desempenho (média e desvio-padrão) obtido em cada iteração durante a validação cruzada.
- Plotar a evolução dos resultados de busca para maior clareza (opcional).

6 Monitoramento e Avaliação

1. **Seleção da Melhor Configuração**: A melhor combinação de hiperparâmetros para cada modelo será escolhida com base na métrica de interesse (por exemplo, recall ou F1-score).
2. **Treinamento Final**: Reajustar cada modelo com toda a base de treinamento (80%) usando os hiperparâmetros selecionados.
3. **Avaliação no Conjunto de Teste**:
   - Avaliar o desempenho dos modelos no *dataset* de teste.
   - Calcular métricas como: **acurácia, precisão, recall, F1-score, AUC-ROC** (para problemas binários) e criar matriz de confusão.

7 Comparação e Análise

- **Comparação de Desempenho**: Apresentar gráficos (barras, boxplots ou curvas ROC) comparando métricas relevantes entre os modelos.
- **Overfitting vs. Underfitting**: Verificar se há grande discrepância entre as métricas em treino e teste, indicando overfitting.
- **Melhores Hiperparâmetros**: Registrar e discutir os hiperparâmetros que resultaram nos melhores desempenhos.
- **Conclusões**: Descrever qual modelo teve melhor desempenho em dados não vistos e possíveis implicações dos resultados.

8 Cuidados Metodológicos

8.1 Prevenção de Data Leakage

- Ajustar o pré-processamento e transformações *apenas* com dados de treino, aplicando-os posteriormente nos dados de teste.

- Separar treino e teste desde o início do processo para evitar vazamentos (por exemplo, estatísticas de normalização calculadas com o conjunto de teste).

8.2 Validação Adequada

- Utilizar **validação cruzada estratificada** para lidar com desbalanceamentos.
- Manter as mesmas métricas nas diferentes fases de avaliação para comparações consistentes.

8.3 Reprodutibilidade

- Definir `random_state` para todos os processos aleatórios.
- Documentar as versões de bibliotecas e parâmetros utilizados, se possível.

8.4 Eficiência Computacional

- Preferir ***RandomizedSearchCV*** a *GridSearchCV* em espaços grandes de parâmetros.
- Utilizar `n_jobs=-1` para paralelizar a busca, quando possível.
- Aplicar técnicas de *early stopping* (se disponíveis) para reduzir tempo de treinamento excessivo.

8.5 Interpretação Criteriosa

- Considerar múltiplas métricas (acurácia, precisão, recall, F1-score, AUC, etc.) para maior profundidade de análise.
- Analisar possíveis razões para discrepâncias, como tamanho de dados ou ruído nas variáveis.