Checklist para Projetos de Machine Learning

Este checklist pode guiá-lo através de seus projetos de Machine Learning. Ele foi adaptado e aprimorado para o contexto brasileiro a partir do checklist original de Aurélien Géron e inclui oito etapas principais. Sinta-se à vontade para adaptar este checklist às suas necessidades específicas.

1. Definição do Problema e Visão Geral

- **Definir o objetivo em termos de negócio:** Qual problema de negócio estamos tentando resolver?
- **Como a solução será utilizada?** Quem são os usuários finais? Como será integrada aos processos existentes?
- **Quais são as soluções/alternativas atuais (se houver)?** Quais são seus pontos fortes e fracos?
- **Como enquadrar o problema?**
    - Aprendizado supervisionado, não supervisionado, por reforço?
    - Classificação, regressão, clusterização, redução de dimensionalidade?
    - Processamento em lote (batch) ou online (tempo real)?
- **Como o desempenho deve ser medido?** Quais métricas são relevantes (acurácia, precisão, recall, F1-score, AUC, RMSE, MAE, etc.)?
- **A métrica de desempenho está alinhada com o objetivo de negócio?**
- **Qual seria o desempenho mínimo necessário para atingir o objetivo de negócio?** (Definir um baseline)
- **Existem problemas comparáveis?** É possível reutilizar experiências, artigos acadêmicos, ferramentas ou modelos pré-treinados?
- **Há especialistas no domínio disponíveis para consulta?**
- **Como o problema seria resolvido manualmente?** Isso pode dar insights sobre features e regras.
- **Listar as premissas feitas até agora** (por você ou outros).
- **Verificar as premissas, se possível.**

2. Obtenção dos Dados

Nota: Automatize ao máximo para obter dados atualizados facilmente.

- **Listar os dados necessários e a quantidade estimada.**
- **Encontrar e documentar as fontes de dados.**
- **Verificar o espaço de armazenamento necessário.**
- **Verificar obrigações legais (como LGPD) e obter as autorizações necessárias.** Cuidado especial com dados sensíveis.
- **Obter as autorizações de acesso aos dados.**
- **Criar um ambiente de trabalho adequado** (com espaço de armazenamento suficiente e seguro).
- **Coletar os dados.**
- **Converter os dados para um formato manipulável** (ex: CSV, banco de dados), sem alterar os dados originais.
- **Garantir que informações sensíveis sejam removidas ou protegidas** (ex: anonimização, pseudonimização).
- **Verificar o tamanho e tipo dos dados** (séries temporais, amostras, dados geográficos, texto, imagem, etc.).
- **Versionar os dados** (para reprodutibilidade).
- ***Separar um conjunto de teste (test set), guardá-lo e nunca analisá-lo durante a exploração e treinamento*** (evitar *data snooping*). Idealmente, 10-20% dos dados.

3. Exploração dos Dados (Análise Exploratória - EDA)

Nota: Tente obter insights de um especialista no domínio para estas etapas.

- **Criar uma cópia dos dados para exploração** (amostrando para um tamanho gerenciável, se necessário).
- **Utilizar ferramentas como Jupyter Notebooks ou similar para registrar a exploração.**
- **Estudar cada atributo (feature) e suas características:**
  - Nome
  - Tipo (categórico, numérico - int/float, discreto/contínuo, texto, estruturado, etc.)
  - Percentual de valores ausentes (% missing)
  - Ruído e tipo de ruído (estocástico, outliers, erros de arredondamento, etc.)
  - Potencial utilidade para a tarefa
  - Tipo de distribuição (Gaussiana, uniforme, logarítmica, bimodal, etc.) - *Visualizar com histogramas, boxplots.*
- **Para tarefas supervisionadas, identificar o(s) atributo(s) alvo (target).**
- **Visualizar os dados:** Gráficos de dispersão (scatter plots), matrizes de correlação, mapas de calor, etc.
- **Estudar as correlações entre os atributos** (e com o atributo alvo).
- **Analisar como o problema seria resolvido manualmente (insights dos especialistas).**
- **Identificar transformações promissoras a serem aplicadas** (log, raiz quadrada, etc.).
- **Identificar dados extras que seriam úteis** (voltar à etapa "Obtenção dos Dados").
- **Documentar os aprendizados e insights.**

4. Preparação dos Dados (Pré-processamento e Engenharia de Atributos)

- *Notas:*
  - *Trabalhe em cópias dos dados (mantenha o dataset original intacto).*
  - *Escreva funções reutilizáveis para todas as transformações aplicadas, por cinco motivos:*
    - *Para preparar facilmente novos dados (fresh data).*
    - *Para aplicar em projetos futuros.*
    - *Para preparar o conjunto de teste.*
    - *Para preparar novas instâncias em produção.*
    - *Para tratar as escolhas de pré-processamento como hiperparâmetros.*
- **Limpeza dos dados (Data Cleaning):**
  - Corrigir ou remover outliers (opcional, com cautela).
  - Preencher valores ausentes (ex: com zero, média, mediana, moda, ou usando algoritmos como KNNImputer) ou remover linhas/colunas (com cautela).
- **Seleção de Atributos (Feature Selection) (opcional):**
  - Remover atributos que não fornecem informação útil para a tarefa (ex: IDs, atributos com variância zero, atributos com alta correlação entre si).
- **Engenharia de Atributos (Feature Engineering), onde apropriado:**
  - Discretizar atributos contínuos (binning).
  - Decompor atributos (ex: data/hora em ano, mês, dia, dia da semana; categóricos em múltiplas features).
  - Aplicar transformações promissoras (ex: log(x), sqrt(x), x^2, interações entre features).
  - Agregar atributos para criar novos (ex: média de compras nos últimos 3 meses).
  - Tratar atributos categóricos (One-Hot Encoding, Label Encoding, Target Encoding, etc.).
- **Escalonamento de Atributos (Feature Scaling):** Padronizar (StandardScaler) ou normalizar (MinMaxScaler) os atributos numéricos.

5. Seleção e Treinamento Inicial de Modelos

- *Notas:*
  - *Se os dados forem muito grandes, considere amostrar conjuntos de treinamento menores para treinar diversos modelos rapidamente (ciente de que isso pode penalizar modelos complexos como redes neurais grandes ou Random Forests).*
  - *Automatize estas etapas o máximo possível.*
- **Treinar rapidamente vários modelos "básicos" de diferentes categorias:**
  - Lineares (Regressão Linear/Logística, SVM Linear)
  - Baseados em Árvores (Decision Tree, Random Forest, Gradient Boosting - XGBoost, LightGBM, CatBoost)
  - Naive Bayes
  - K-Nearest Neighbors (KNN)
  - Redes Neurais (simples, como MLP)
  - Utilizar parâmetros padrão inicialmente.
- **Medir e comparar o desempenho:**
  - Usar validação cruzada (N-fold cross-validation) no conjunto de *treinamento* para obter uma estimativa mais robusta.
  - Calcular a média e o desvio padrão da métrica de desempenho escolhida para cada modelo.
- **Analisar as variáveis mais significativas para cada algoritmo.**
- **Analisar os tipos de erros que os modelos cometem** (matriz de confusão, análise de resíduos).
  - Quais dados um humano usaria para evitar esses erros?
- **Fazer uma rodada rápida de seleção e engenharia de atributos baseada nos resultados.**
- **Iterar rapidamente mais uma ou duas vezes nos cinco passos anteriores.**
- **Selecionar os 3 a 5 modelos mais promissores**, preferindo modelos que cometem tipos diferentes de erros (potencial para Ensembles).

6. Ajuste Fino (Fine-Tuning) e Combinação de Modelos

- *Notas:*
  - *Utilize o máximo de dados possível nesta etapa (conjunto de treinamento completo).*
  - *Automatize o que for possível.*
  - *Rastreie seus experimentos (parâmetros, código, datasets, métricas) usando ferramentas como MLflow, DVC, Weights & Biases.*
- **Ajustar os hiperparâmetros usando validação cruzada:**
  - Trate suas escolhas de transformação de dados como hiperparâmetros (ex: método de imputação de missing values, tipo de escalonamento).
  - Explore o espaço de hiperparâmetros:
    - **Grid Search:** Bom para poucos parâmetros e valores discretos.
    - **Random Search:** Geralmente mais eficiente que Grid Search para muitos parâmetros.
    - **Otimização Bayesiana:** Eficiente quando o treinamento é muito longo (ex: usando Gaussian Process).
- **Testar métodos de Ensemble:** Combinar os melhores modelos (Voting, Averaging, Stacking) geralmente melhora o desempenho individual.
- ***Uma vez confiante no modelo final, avalie seu desempenho no conjunto de teste (test set) uma única vez*** para estimar o erro de generalização.
- ***Não ajuste mais o modelo após medir o erro de generalização no conjunto de teste!*** Fazer isso levaria ao overfitting no conjunto de teste.

7. Apresentação da Solução

- **Documentar o que foi feito:** Arquitetura da solução, escolhas de design, resultados dos experimentos.
- **Criar uma apresentação clara e concisa:**
  - Comece com a visão geral e o objetivo de negócio.
  - Explique como a solução atinge o objetivo de negócio.
  - Destaque os pontos interessantes observados durante o projeto.
  - Descreva o que funcionou e o que não funcionou.
  - Liste as premissas feitas e as limitações do sistema.
- **Comunicar os resultados chave** através de visualizações impactantes ou declarações fáceis de lembrar (ex: "a renda mediana é o principal preditor dos preços dos imóveis").
- **Tornar os resultados reprodutíveis.**

8. Lançamento, Monitoramento e Manutenção (Deploy & MLOps)

- **Preparar a solução para produção:**
  - Integrar com as fontes de dados de produção.
  - Escrever testes unitários e de integração.
  - Conteinerizar a aplicação (ex: Docker).
  - Definir a infraestrutura de deploy (Cloud, on-premises).
- **Escrever código de monitoramento para verificar o desempenho do sistema em produção em intervalos regulares.**
  - Configurar alertas para quedas de desempenho.
  - Monitorar *data drift* (mudança na distribuição dos dados de entrada) e *concept drift* (mudança na relação entre features e target).
- **Cuidado com a degradação lenta:** Modelos tendem a "apodrecer" à medida que os dados evoluem.
- **A medição de desempenho pode exigir um pipeline humano** (ex: via serviço de crowdsourcing ou feedback de usuários).
- **Monitorar a qualidade dos dados de entrada** (ex: sensor com defeito, output de outro time desatualizado). Crucial para sistemas online.
- **Definir uma estratégia de retreinamento:**
  - Retreinar os modelos regularmente com dados atualizados.
  - Automatizar o processo de retreinamento e deploy (CI/CD para ML).
- **Ter um plano de rollback** caso o novo modelo apresente problemas em produção.