


# Engenharia de Computação
# Inteligência Artificial
# 15 - Algoritmos Ensemble
# Prática guiada

# Retomando e avançando

## O que faremos

- Comparar DT, KNN e SVM com sete configurações ensemble.
- Usar a mesma base da Aula 11.
- Interpretar métricas, dispersão e custo.

## Para discutir

A combinação melhorou em relação aos modelos individuais?

# Roteiro de 100 minutos

## Parte 1: execução e compreensão (50 minutos)

| Etapa | Tempo |
|---|---|
| Execução e leitura do código | 15 min |
| Interpretar métricas e matriz | 15 min |
| Experimentos controlados | 20 min |

## Parte 2: adaptação ao Trabalho 02 (50 minutos)

| Etapa | Tempo |
|---|---|
| Mapear a base do Trabalho 02 | 15 min |
| Adaptar pipeline e executar | 25 min |
| Registrar decisões e pendências | 10 min |

# Arquivos e ambiente

## Pacote de código

- ensemble.py: implementação completa.
- requirements.txt: dependências.
- README.md: execução e interpretação.

## Comandos

```
pip install -r requirements.txt
python ensemble.py
```

## Observação

Não precisamos baixar dados adicionais.

# Base de dados

## Breast Cancer Wisconsin

- 569 exemplos; 30 atributos numéricos.
- Classe 1: maligno. Classe 0: benigno.
- Mesmo problema de classificação da Aula 11.

## Para discutir

Qual erro é um falso negativo neste cenário?

# Protocolo de execução

## Divisão estratificada

- 80% treinamento; 20% teste reservado.
- Validação cruzada estratificada com cinco folds no treino.
- Seed 42; mesmas partições para todos os modelos.

## Observação

A execução padrão não consulta o teste.

# Leitura do código: preparação dos dados

```
data = load_breast_cancer()
X, y = data.data, (data.target == 0).astype(int)
X_train, X_test, y_train, y_test = train_test_split(
  X, y, test_size=0.2,
  stratify=y, random_state=args.seed,
)
cv = StratifiedKFold(
  5, shuffle=True, random_state=args.seed
)
```

## Leitura do código

Para que servem a estratificação e a semente? Qual conjunto será reservado até a avaliação final?

# Estimar desempenho

| Modelo | Média F1 | Desvio-padrão F1 |
|---|---|---|
| A | 0,92 | 0,02 |
| B | 0,93 | 0,08 |

**Atividade**

Exemplo fictício: qual é mais estável? A diferença de médias é suficiente para declarar superioridade?

# Modelos individuais

## Função models

- DecisionTreeClassifier: sem escalonamento.
- KNeighborsClassifier: StandardScaler no pipeline.
- SVC: StandardScaler e probability=True.

## Observação

A opção de probabilidades é usada no soft voting; não garante calibração perfeita.

# Leitura do código: combinação de modelos

```
base = [('dt', dt), ('knn', knn), ('svm', svm)]
VotingClassifier(base, voting='hard')
VotingClassifier(base, voting='soft')
BaggingClassifier(
  estimator=dt, n_estimators=n,
  random_state=seed
)
```

**Pergunta**

Quais linhas combinam os três estimadores diferentes? Qual utiliza várias árvores?

# Voting e bagging

## No código

- VotingClassifier: hard e soft com os mesmos estimadores base.
- BaggingClassifier: árvores, bootstrap e número de estimadores definido.

## Para discutir

Por que os estimadores individuais são clonados e reajustados pela validação?

# Random Forest e boosting

## No código

- RandomForestClassifier: diversidade de amostras e atributos.
- AdaBoostClassifier: combinação sequencial de estimadores.
- GradientBoostingClassifier: minimização sequencial da perda.

## Para discutir

O número de estimadores tem o mesmo significado em todas as famílias?

# Stacking

## No código

- Estimadores base: DT, KNN, SVM.
- Metaestimador: regressão logística.
- CV interna: gera predições OOF no treino de cada fold externo.

## Observação

A validação externa avalia o conjunto completo, não apenas os estimadores base.

# Leitura do código: avaliação comum

```
scoring = ['accuracy', 'precision', 'recall', 'f1']
result = cross_validate(
  model, X_train, y_train,
  cv=cv, scoring=scoring
)
scores = result['test_f1']
print(scores.mean(), scores.std())
```

**Observação**

A chave ‘test_f1‘ em cross_validate refere-se à validação de cada fold, não ao conjunto de teste reservado.

# Checkpoint: primeira execução

**Atividade**

Execute `python ensemble.py`. Identifique a melhor média F1, o desvio-padrão e o tempo médio de treinamento. Compare com DT, KNN e SVM.

# Experimentos controlados

**Atividade**

Execute `python ensemble.py -estimadores 10` e depois com 100. Mantenha a seed 42. Compare os CSVs e registre o efeito do número de estimadores.

**Observação**

Voting e stacking não variam com esse parâmetro no código.

# Escolha e avaliação final

## Depois de congelar a configuração

- Execute `python ensemble.py -estimadores 50 -teste` (substitua 50 pela configuração escolhida).
- Analise as quatro métricas e as matrizes de confusão.
- Não volte a ajustar com base no teste.

## Para discutir

Se o ranking mudar entre validação e teste, como interpretar?

# Variabilidade e discussão

**Atividade**

Compare, se houver tempo, outras seeds predefinidas (por exemplo, 10 e 21) como repetições completas do protocolo. Registre todos os resultados; não selecione a melhor seed.

**Observação**

Tempos são dependentes do equipamento e não constituem garantia de desempenho.

# Reconhecer sua base

**Atividade**

Abra a base escolhida pelo grupo (UCI ou Kaggle). Identifique alvo, tipos de atributos, classes, valores ausentes, desequilíbrio e possíveis variáveis que vazam informação.

## Mapa de adaptação

| Parte | O que adaptar |
|---|---|
| Dados | carregamento e alvo |
| Pipeline | imputação, escala, categorias |
| Divisão | estratificação, grupos ou tempo |
| Métricas | classe positiva ou média macro |
| Modelos | parâmetros adequados à base |

**Observação**

Não alterar a divisão para favorecer um modelo.

# Adaptação orientada (25 min)

**Atividade**

Substitua o carregamento da base. Execute pelo menos um modelo individual e um ensemble. Garanta que as transformações estejam dentro de pipelines. Registre as dificuldades e uma estratégia para completar a comparação.

# Conexão com as metaheurísticas

## Para temas de otimização

- A representação, função objetivo e restrições devem ser preservadas entre AG, ACO/PSO e método imune conforme o enunciado.
- Se o objetivo envolve classificadores, utilize somente dados de validação para otimizar.

## Observação

Ensemble não substitui as metaheurísticas exigidas.

# Encerramento: registro do grupo

**Atividade**

Registre base, alvo, protocolo, seed, modelos, métricas, resultados iniciais, limitações e próximos passos para o Trabalho 02.

# Dúvidas



# Verificação de aprendizagem

**Atividade**

Explique por que o pipeline evita vazamento, por que o teste é reservado e se o ensemble escolhido apresentou ganho relevante em relação ao baseline.