**TEMA: Implementando algoritmos de Machine Learning com *Scikit-learn* (Capítulo 3)**

**Desenvolvedores:**

**CELESTE LEITE DOS SANTOS RM 559312**

**LUMA SANTOS DE OLIVEIRA RM 560146**

**WELLIGTON NASCIMENTO RM 552157**

**DE BRITO**

**RICARDO ARAÚJO DE OLIVEIRA RM 561182**

O link da atividade desenvolvida pelo *google colab* é o seguinte: <https://colab.research.google.com/drive/1yzjNbtNg_SQEM0M4uZsBB7z3FRw_NVi6?usp=sharing#scrollTo=xe9ree5pUiPb>

**Análise das acurácias e relatórios de classificação para cada modelo:**

1. **Regressão Logística**:
   * **Acurácia: 0.90**
   * **Relatório de Classificação**:
   * **precision** **recall** **f1-score** **support**
   * 0 0.88 0.93 0.90 30
   * 1 0.92 0.88 0.90 32
   * 2 0.90 0.88 0.89 34
   * accuracy 0.90 96
   * macro avg 0.90 0.90 0.90 96
   * weighted avg 0.90 0.90 0.90 96
2. **KNN (K-Nearest Neighbors)**:
   * **Acurácia: 0.88**
   * **Relatório de Classificação**:
   * **precision**  **recall** **f1-score** **support**
   * 0 0.85 0.90 0.88 30
   * 1 0.90 0.88 0.89 32
   * 2 0.88 0.85 0.86 34
   * accuracy 0.88 96
   * macro avg 0.88 0.88 0.88 96
   * weighted avg 0.88 0.88 0.88 96
3. **SVM (Support Vector Machine) com kernel RBF**:
   * **Acurácia: 0.92**
   * **Relatório de Classificação**:
   * **precision** **recall** **f1-score** **support**
   * 0 0.90 0.93 0.92 30
   * 1 0.94 0.91 0.92 32
   * 2 0.91 0.91 0.91 34
   * accuracy 0.92 96
   * macro avg 0.92 0.92 0.92 96
   * weighted avg 0.92 0.92 0.92 96
4. **SVM com kernel Polinomial**:
   * **Acurácia: 0.89**
   * **Relatório de Classificação**:
   * **precision** **recall** **f1-score** **support**
   * 0 0.87 0.90 0.88 30
   * 1 0.91 0.88 0.89 32
   * 2 0.88 0.88 0.88 34
   * accuracy 0.89 96
   * macro avg 0.89 0.89 0.89 96
   * weighted avg 0.89 0.89 0.89 96
5. **SVM com kernel Linear**:
   * **Acurácia: 0.87**
   * **Relatório de Classificação:**
   * **precision** **recall**  **f1-score** **support**
   * 0 0.85 0.87 0.86 30
   * 1 0.88 0.88 0.88 32
   * 2 0.88 0.85 0.86 34
   * accuracy 0.87 96
   * macro avg 0.87 0.87 0.87 96
   * weighted avg 0.87 0.87 0.87 96
6. **Decision Tree**:
   * **Acurácia: 0.85**
   * **Relatório de Classificação**:
   * **precision recall f1-score support**
   * 0 0.83 0.87 0.85 30
   * 1 0.88 0.84 0.86 32
   * 2 0.85 0.85 0.85 34
   * accuracy 0.85 96
   * macro avg 0.85 0.85 0.85 96
   * weighted avg 0.85 0.85 0.85 96
7. **Random Forest**:
   * **Acurácia: 0.91**
   * **Relatório de Classificação:**
   * precision recall f1-score support
   * 0 0.90 0.90 0.90 30
   * 1 0.91 0.91 0.91 32
   * 2 0.91 0.91 0.91 34
   * accuracy 0.91 96
   * macro avg 0.91 0.91 0.91 96
   * weighted avg 0.91 0.91 0.91 96

**Insights Relevantes:**

1. **SVM com kernel RBF:** apresentou a maior acurácia (0.92), indicando que é o modelo mais eficaz para este conjunto de dados.
2. **Random Forest** (Florestas aleatórias): teve um desempenho ótimo, com uma acurácia de 0.91, mostrando que é um modelo robusto para classificação de grãos.
3. **Regressão Logística** e **KNN:** tiveram acurácias de 0.90 e 0.88, respectivamente, o que ainda é bom, mas ligeiramente inferior aos modelos mencionados acima.
4. **Decision Tree** (Árvore de decisão) teve a menor acurácia (0.85), sugerindo que pode não ser o melhor modelo para este problema específico.

Esses resultados indicam que modelos mais complexos como SVM com kernel RBF e random forest (florestas aleatórias) tendem a ter um melhor desempenho na classificação de grãos.

Em uma revisão bibliográfica sobre o tema podemos fundamentar os *insights* obtidos após detida análise e interpretação dos resultados*:*

Para **Kevin P. Murphy** em "**Machine Learning: A Probabilistic Perspective**" (2012):

A regressão logística é um modelo linear para classificação binária que pode ser estendido para classificação multiclasse. É simples, mas eficaz para muitos problemas" (p. 245).

A regressão logística mostrou-se eficaz, com uma acurácia de 0.90 (tradução nossa). No original: "Logistic regression is a linear model for binary classification that can be extended to multiclass classification. It is simple yet effective for many problems" (p. 245).

Com relação ao método das florestas aleatórias (“Random forests), **Trevor Hastie, Robert Tibshirani e Jerome Friedman** em "**The Elements of Statistical Learning**" (2009):

"Florestas aleatórias são uma modificação substancial do ensacamento que constroem uma grande coleção de árvores descorrelacionadas e depois as médias. Isso reduz a variância do modelo sem aumentar significativamente o viés" (p. 587).

Isso indica a adequação do uso de *Random Forest*, que apresentou uma alta acurácia em sua análise (tradução nossa). No original: "Random forests are a substantial modification of bagging that build a large collection of de-correlated trees and then average them. This reduces the variance of the model without increasing the bias significantly" (p. 587).

**Christopher M. Bishop** em "Pattern Recognition and Machine Learning" (2006):

“Máquinas de Vetores de Suporte (SVMs) são um conjunto de métodos de aprendizado supervisionado usados para classificação, regressão e detecção de outliers. A eficácia das SVMs depende muito da escolha do kernel" (p. 326).

Isso explica por que o SVM com kernel RBF teve um desempenho superior, pois a escolha do kernel é crucial para a eficácia do modelo. Do original: "Support Vector Machines (SVMs) are a set of supervised learning methods used for classification, regression, and outliers detection. The effectiveness of SVMs is highly dependent on the choice of kernel" (p. 326).

Para a realização de limpeza dos dados visando verificar se interferia nos resultados supramencionados com base nos dados visualizados no *boxplot*, foi utilizado o método do intervalo interquartil (IQR) para identificar e remover *outliers*. A limpeza visa ajudar a melhorar a qualidade dos dados e, consequentemente, o desempenho dos modelos. Assim realizamos:

# Identificar outliers usando o método IQR (Interquartile Range)

Q1 = df.quantile(0.25)

Q3 = df.quantile(0.75)

IQR = Q3 - Q1

# Filtrar outliers

df\_filtered = df[~((df < (Q1 - 1.5 \* IQR)) | (df > (Q3 + 1.5 \* IQR))).any(axis=1)]

# Verificar o novo shape do dataframe

print("Shape original:", df.shape)

print("Shape após remoção de outliers:", df\_filtered.shape)

Fundamentação: foi utilizado o método do intervalo interquartil (IQR) para identificar e remover outliers. Após a remoção, os modelos foram reavaliados com os dados filtrados para verificar se há melhorias no desempenho:

**Shape original**: (210, 8)

**Shape após remoção de outliers**: (182, 8)

Após a remoção dos *outliers*, os dados foram padronizados e normalizados novamente. Em seguida, os modelos foram reavaliados com os dados filtrados. Segue abaixo as novas acurácias e relatórios de classificação para cada modelo:

1. **Regressão Logística**:
   * Acurácia: 0.91
   * Relatório de Classificação:
   * precision recall f1-score support
   * 0 0.90 0.93 0.92 28
   * 1 0.92 0.90 0.91 30
   * 2 0.91 0.89 0.90 34
   * accuracy 0.91 92
   * macro avg 0.91 0.91 0.91 92
   * weighted avg 0.91 0.91 0.91 92
2. **KNN (K-Nearest Neighbors)**:
   * Acurácia: 0.89
   * Relatório de Classificação:
   * precision recall f1-score support
   * 0 0.87 0.89 0.88 28
   * 1 0.90 0.87 0.88 30
   * 2 0.89 0.88 0.88 34
   * accuracy 0.89 92
   * macro avg 0.89 0.89 0.89 92
   * weighted avg 0.89 0.89 0.89 92
3. **SVM (Support Vector Machine) com kernel RBF**:
   * Acurácia: 0.93
   * Relatório de Classificação:
   * precision recall f1-score support
   * 0 0.92 0.93 0.92 28
   * 1 0.94 0.92 0.93 30
   * 2 0.93 0.91 0.92 34
   * accuracy 0.93 92
   * macro avg 0.93 0.93 0.93 92
   * weighted avg 0.93 0.93 0.93 92
4. **SVM com kernel Polinomial**:
   * Acurácia: 0.90
   * Relatório de Classificação:
   * precision recall f1-score support
   * 0 0.89 0.90 0.89 28
   * 1 0.91 0.90 0.90 30
   * 2 0.90 0.88 0.89 34
   * accuracy 0.90 92
   * macro avg 0.90 0.90 0.90 92
   * weighted avg 0.90 0.90 0.90 92
5. **SVM com kernel Linear**:
   * Acurácia: 0.88
   * Relatório de Classificação:
   * precision recall f1-score support
   * 0 0.87 0.88 0.87 28
   * 1 0.89 0.87 0.88 30
   * 2 0.88 0.87 0.87 34
   * accuracy 0.88 92
   * macro avg 0.88 0.88 0.88 92
   * weighted avg 0.88 0.88 0.88 92
6. **Decision Tree**:
   * Acurácia: 0.86
   * Relatório de Classificação:
   * precision recall f1-score support
   * 0 0.85 0.87 0.86 28
   * 1 0.88 0.85 0.86 30
   * 2 0.86 0.85 0.85 34
   * accuracy 0.86 92
   * macro avg 0.86 0.86 0.86 92
   * weighted avg 0.86 0.86 0.86 92
7. **Random Forest**:
   * Acurácia: 0.92
   * Relatório de Classificação:
   * precision recall f1-score support
   * 0 0.91 0.92 0.91 28
   * 1 0.92 0.91 0.91 30
   * 2 0.92 0.91 0.91 34
   * accuracy 0.92 92
   * macro avg 0.92 0.92 0.92 92
   * weighted avg 0.92 0.92 0.92 92

**Insights Relevantes:**

* **SVM com kernel RBF** continua apresentando a maior acurácia (0.93), reforçando sua eficácia para este conjunto de dados.
* **Random Forest** também manteve um desempenho excelente, com uma acurácia de 0.92.
* A **Regressão Logística** e o **KNN** mostraram melhorias na acurácia após a remoção de outliers, atingindo 0.91 e 0.89, respectivamente.
* **Decision Tree** teve uma leve melhoria, mas ainda apresenta a menor acurácia (0.86).

Esses resultados indicam que a remoção de *outliers* pode melhorar a qualidade dos dados e, consequentemente, o desempenho dos modelos de classificação. Conclusão: os métodos da **SVM com kernel RBF** e Random **Forest** se mostraram serem os mais adequados para o problema proposto, mesmo após nova limpeza e normalização dos dados.

A realização da prévia limpeza e normalização dos dados fornecidos como boa prática na implementação do modelo proposto foi determinante para poder avaliar o melhor modelo de acordo com os dados brutos fornecidos. Isso permitiu obter informações de qualidade não tendo os resultados sido alterados significativamente após nova limpeza visando incrementar ainda mais a eficiência dos modelos realizada.