Bootcamp IGTI: Analista de Machine Learning

Desafio

Módulo 2	**Modelos Preditivos e Séries Temporais**

Objetivos

Exercitar os seguintes conceitos trabalhados no Módulo:

- ✓ Análise exploratória de dados (EDA - *Exploratory Data Analysis*).
- ✓ Comparação e treinamento de modelos de classificação.

Enunciado

Neste desafio, serão abordados conceitos apresentados durante a disciplina Modelos Preditivos e Séries Temporais (MPT). Será utilizado o dataset “Banknote authetication”, disponível no UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets/banknote+authentication).

Este dataset consiste de atributos contínuos (variance, skewness, curtosis e entropy) extraídos de imagens de cédulas reais e falsificadas através de uma etapa de processamento de sinais (transformação de Wavelet), além da indicação se o exemplo é de uma cédula genuína (class = 0) ou falsificada (class = 1).

Atividades

Os alunos deverão desempenhar as seguintes atividades:

1. Acessar o ambiente Google Colaboratory (recomendado) ou qualquer ambiente de desenvolvimento em Python.

2. Carregar o dataset https://pycourse.s3.amazonaws.com/banknote_authentication.txt para análise utilizando o pandas. Exemplo:

```
# leitura do dataset
df = pd.read_csv('https://pycourse.s3.amazonaws.com/banknote_authentication.txt',
                 header=None,
                 names=['variance', 'skewness', 'curtosis', 'entropy', 'class'])
```

3. Divida o dataset utilizando a função *train_test_split*, conforme:
    a. Conjunto de treino (70%);
    b. Conjunto de teste (30%);
    c. random_state=1.
4. Utilize a variável "class" como saída e as demais como entrada dos modelos.
5. Para implementação dos algoritmos, utilize as seguintes definições (do sklearn):
    a. Algoritmo KNN:

```
clf_KNN = KNeighborsClassifier(n_neighbors=5)
```

    b. Algoritmo Árvore de Decisão (Decision Tree):

```
clf_arvore = DecisionTreeClassifier(random_state=1)
```

    c. Algoritmo Floresta Aleatória (Random Forest):

```
clf_floresta = RandomForestClassifier(max_depth=8, random_state=1)
```

    d. Algoritmo SVM:

```
clf_svm = SVC(gamma='auto',kernel='rbf', random_state=1)
```

    e. Algoritmo Rede MLP:

```
clf_mlp = MLPClassifier(hidden_layer_sizes=(2,),
                        solver='lbfgs',
                        random_state=1)
```

Respostas Finais

Os alunos deverão desenvolver a prática e, depois, responder às seguintes questões objetivas: