Trabalho de Aprendizado de Máquina

Nomes dos componentes do Grupo 7:
Andressa Magalhães
Jaime Terra
Leoncio Santana
Renan Baqui

1. Descrição do problema

O problema é realizar uma previsão precisa de diagnóstico de câncer de mama entre maligno ou benigno através de 30 atributos diferentes de resultados de exames de cada paciente.

2. Justificativa e descrição da ferramenta escolhida

A justificativa é criar uma aplicação que possa identificar com precisão a presença de câncer de mama através de dados diagnósticos fornecidos.

O scikit-learn é uma biblioteca de aprendizado de máquina de software livre para Python. A biblioteca apresenta vários algoritmos de classificação, regressão e *clustering*, incluindo máquinas de vetor de suporte, florestas aleatórias, aumento de gradiente, *k-means* e DBSCAN, e é projetado para interoperar com as bibliotecas numéricas e científicas do Python NumPy e SciPy.

Atualmente o scikit-learn é uma das bibliotecas mais populares de machine learning no GitHub e uma justificativa para sua utilização é que possui ótima integração com muitas outras bibliotecas Python, como matplotlib and plotly para plotar e numpy para vetorização de arrays.

Para resolução do problema utilizamos a plataforma Google Colab que é um serviço de nuvem gratuito hospedado pela Google para incentivar a pesquisa de Aprendizado de Máquina e Inteligência Artificial.

```
[ ] %matplotlib inline
    from sklearn.datasets import load_breast_cancer
    from sklearn.model_selection import train_test_split
    from sklearn.neighbors import KNeighborsClassifier
    import matplotlib.pyplot as plt
    import numpy as np

    breast = load_breast_cancer()
    breast.keys()
```

```
dict_keys(['data', 'target', 'target_names', 'DESCR', 'feature_names', 'filename'])
```

Importação das bibliotecas e do dataset através do Google Colab

```
[ ]  X_train, X_test, y_train, y_test = train_test_split(breast['data'], breast['target'], random_state = 30)

     print(X_train.shape)
     print(X_test.shape)

     (426, 30)
     (143, 30)
```

Métodos da biblioteca scikit-learn para dividir a base em dois: o conjunto de treino e o conjunto de teste

3. Descrição do dataset escolhido com o endereço de onde foi obtido

Este dataset chama-se *Breast Cancer Wisconsin (Diagnostic)* e é fornecido pela *University of California, Irvine* através de seu repositório de machine learning no endereço:

https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

Os *features* são computados a partir de uma imagem digitalizada de um procedimento diagnóstico chamado *fine-needle aspiration* (FNA) de uma massa mamária. Eles descrevem as características dos núcleos celulares presentes na imagem.

O plano de separação descrito acima foi obtido usando *Multisurface Method-Tree* (MSM-T) [K. P. Bennett, "Decision Tree Construction Via Linear Programming." Proceedings of the 4 Midwest Artificial Intelligence and Cognitive Science Society, pp. 97-101, 1992], um método de classificação que usa programação linear para construir uma árvore de decisão. Os recursos relevantes foram selecionados usando uma pesquisa exaustiva no espaço de 1-4 *features* e 1-3 planos de separação.

O programa linear real usado para obter o plano de separação no espaço tridimensional é descrito em: [K. P. Bennett e O. L. Mangasarian: “Robust Linear Programming Discrimination of Two Linearly Inseparable Sets”, Optimization Methods and Software 1, 1992, 23-34].

O dataset breast_cancer é padrão da biblioteca scikit-learn e não necessita de um download de arquivo de uma fonte externa.

Atributos:
radius (mean)
texture (mean)
perimeter (mean)
area (mean)
smoothness (mean)
compactness (mean)
concavity (mean)
concave points (mean)
symmetry (mean)
fractal dimension (mean)
radius (standard error)
texture (standard error)
perimeter (standard error)
area (standard error)
smoothness (standard error)
compactness (standard error)
concavity (standard error)
concave points (standard error)
symmetry (standard error)

fractal dimension (standard error)
radius (worst)
texture (worst)
perimeter (worst)
area (worst)
smoothness (worst)
compactness (worst)
concavity (worst)
concave points (worst)
symmetry (worst)
fractal dimension (worst)

Total de Amostras: 569
Dimensionalidade: 30
Atributos Faltantes: Nenhum
Features: Reais e positivos
Distribuição de Classes: 212 - Maligno, 357 - Benigno
Criadores: Dr. William H. Wolberg, W. Nick Street, Olvi L. Mangasarian
Doador: Nick Street
Data: Novembro, 1995

Referências:
W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear feature extraction for breast tumor diagnosis. IS&T/SPIE 1993 International Symposium on Electronic Imaging: Science and Technology, volume 1905, pages 861-870, San Jose, CA, 1993.

O.L. Mangasarian, W.N. Street and W.H. Wolberg. Breast cancer diagnosis and prognosis via linear programming. Operations Research, 43(4), pages 570-577, July-August 1995.

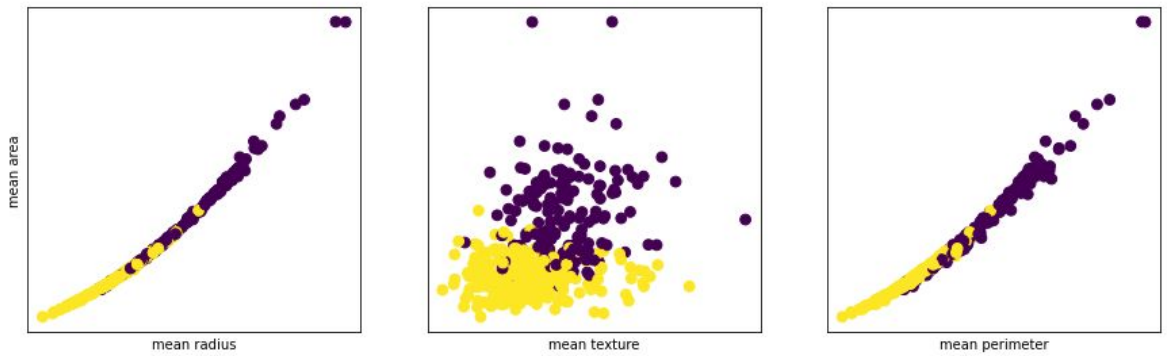
W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machine learning techniques to diagnose breast cancer from fine-needle aspirates. Cancer Letters 77 (1994) 163-171.

Verificação de algumas características do dataset através de código:

```
print(breast['data'].shape)
breast['data'][:3] #contém os atributos das cinco primeiras pacientes
```

```
(569, 30)
array([[1.799e+01, 1.038e+01, 1.228e+02, 1.001e+03, 1.184e-01, 2.776e-01,
        3.001e-01, 1.471e-01, 2.419e-01, 7.871e-02, 1.095e+00, 9.053e-01,
        8.589e+00, 1.534e+02, 6.399e-03, 4.904e-02, 5.373e-02, 1.587e-02,
        3.003e-02, 6.193e-03, 2.538e+01, 1.733e+01, 1.846e+02, 2.019e+03,
        1.622e-01, 6.656e-01, 7.119e-01, 2.654e-01, 4.601e-01, 1.189e-01],
       [2.057e+01, 1.777e+01, 1.329e+02, 1.326e+03, 8.474e-02, 7.864e-02,
        8.690e-02, 7.017e-02, 1.812e-01, 5.667e-02, 5.435e-01, 7.339e-01,
        3.398e+00, 7.408e+01, 5.225e-03, 1.308e-02, 1.860e-02, 1.340e-02,
        1.389e-02, 3.532e-03, 2.499e+01, 2.341e+01, 1.588e+02, 1.956e+03,
        1.238e-01, 1.866e-01, 2.416e-01, 1.860e-01, 2.750e-01, 8.902e-02],
       [1.969e+01, 2.125e+01, 1.300e+02, 1.203e+03, 1.096e-01, 1.599e-01,
        1.974e-01, 1.279e-01, 2.069e-01, 5.999e-02, 7.456e-01, 7.869e-01,
        4.585e+00, 9.403e+01, 6.150e-03, 4.006e-02, 3.832e-02, 2.058e-02,
        2.250e-02, 4.571e-03, 2.357e+01, 2.553e+01, 1.525e+02, 1.709e+03,
        1.444e-01, 4.245e-01, 4.504e-01, 2.430e-01, 3.613e-01, 8.758e-02]])
```

Exemplos dos 3 primeiros arrays com os 30 atributos do dataset



Scatter plot do conjunto de treino: é possível verificar que a dispersão proporcionará uma boa precisão do modelo

4. Apresentação dos resultados

```
[ ]  #criando o modelo
     knn = KNeighborsClassifier(n_neighbors = 1)
     knn.fit(X_train, y_train)
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=1, p=2,
                     weights='uniform')
```

Criação do modelo com o algoritmo K-Nearest Neighbor: o parâmetro n_neighbors (n_vizinhos) pode ser alterado e é proporcional à acurácia

```
#testando o modelo
X_new = np.array([[1.223e+01, 1.956e+01, 7.854e+01, 4.610e+02, 9.586e-02, 8.087e-02,
       4.187e-02, 4.107e-02, 1.979e-01, 6.013e-02, 3.534e-01, 1.326e+00,
       2.308e+00, 2.724e+01, 7.514e-03, 1.779e-02, 1.401e-02, 1.140e-02,
       1.503e-02, 3.338e-03, 1.444e+01, 2.836e+01, 9.215e+01, 6.384e+02,
       1.429e-01, 2.042e-01, 1.377e-01, 1.080e-01, 2.668e-01, 8.174e-02]])
X_new.shape
```

```
(1, 30)
```

```
[9] prediction = knn.predict(X_new) #a previsão é de que a paciente tenha o diagnóstico benigno (1), que está correto
    prediction
```

```
array([1])
```

Teste do modelo com um array previamente conhecido: neste exemplo o modelo fez a previsão de que a paciente teria o diagnóstico benigno (1) corretamente

```
#acurácia do modelo
knn.score(X_test, y_test)
```

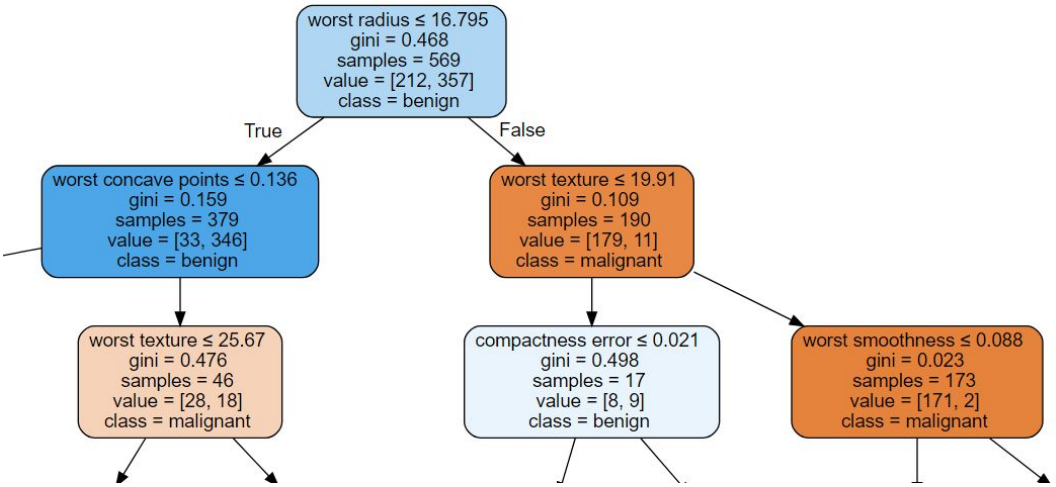
```
0.9370629370629371
```

Definitivamente a métrica mais importante do modelo de classificação: a sua acurácia, ela pode ser aumentada ao incrementar os argumentos dos parâmetros random_state e n_neighbors nos métodos de criação do modelo

```
[23] from sklearn.datasets import load_breast_cancer
     from sklearn import tree
     X, y = load_breast_cancer(return_X_y=True)
     clf = tree.DecisionTreeClassifier()
     clf = clf.fit(X, y)
```

```
[24] tree.plot_tree(clf)
```

Código para a construção de modelo de árvore de decisão e plot do seu gráfico



Parte da árvore de decisão gerada utilizando a biblioteca graphviz, a cor de cada nó da árvore representa uma classe diferente (azul = benigno e laranja = maligno) e o tom da cor indica o seu valor para a regressão

5. Discussão dos resultados

Para compararmos os resultados dos dois modelos de classificação, utilizamos o método classification_report da biblioteca sklearn.metrics, conforme o código abaixo:

```
print(knn.score(X_test, y_test))
print(clf.score(X_test, y_test))
previsto = knn.predict(X_test)

print(classification_report(y_test, previsto, target_names=breast.target_names))
```

- K-vizinhos mais próximos

	precision	recall	f1-score
Maligno	0.94	0.85	0.89
Benigno	0.92	0.97	0.94
macro avg	0.93	0.91	0.92
weighted avg	0.92	0.92	0.92

- Árvore de decisão

	precision	recall	f1-score
Maligno	0.87	0.88	0.88
Benigno	0.93	0.92	0.93
macro avg	0.90	0.90	0.90
weighted avg	0.91	0.91	0.91

Comparações estatísticas dos dois algoritmos diferentes para criação dos modelos de classificação: é possível verificar pequenas vantagens e desvantagens

Podemos verificar através da tabela estatística comparativa acima que os resultados para os dois modelos de classificação foram bastante semelhantes, visto que ambos apresentam uma boa taxa de precisão com cerca de 90% e uma taxa de recall bem razoável. No entanto, vale ressaltar que o *f1 score* para uma situação real de diagnóstico de câncer de mama não é suficiente, pois geraria muitos falsos positivos.