#Título:

##Usando a ciência de dados / aprendizado de máquina para estudos das probabilidades do câncer de mama.

###Descrição: Ferramentas do Python, como Numpy, Pandas, SkLearn, PyTorch, Seaborn, serão usadas para a coleta de informações e cálculos das probabilidades da incidência desta doença.

#Breve descrição das funções das bibliotecas importadas

Pandas: coleta e exibição dos dados em formato de tabela. A coleta ocorre com a leitura de um arquivo \*.csv .

SKLearn: abreviação para scikit-learn, é o responsável por fazer a separação entre dados de treino e teste (aprendizado de máquina).

Numpy: arranja os valores em formato de array para tratamentos nas respectivas funções.

PyTorch: biblioteca responsável por criar a rede neural para realizar a classificação das previsões.

##Breve descrição de toda operação

Uma série de atributos e seus respectivos valores serão dispostos em formato de tabela através do arquivo importado (\*.csv), após a preparação dos valores para tratamento, o PyTorch criará uma rede neural que examinará as combinações de todos os atributos e fará uma previsão da probabilidade de ter, ou não a doença, exibindo, no final, o percentual de acerto de suas previsões. Essa operação é feita usando probabilidades e juntamente com o aprendizado de máquina, o computador irá calcular a probabilidade de uma condição ser verdadeira ou falsa, baseado nos cálculos aprendidos durante a leitura dos arquivos carregados. Eis o motivo pelo qual temos a condição de treino e teste.

OBS. Vale salientar que o processo visa auxiliar um diagnóstico mais preciso e, em caso algum, substituir o exame minucioso de um especialista.

#Iniciando a operação

- Importações

- Leitura dos arquivos de entrada representadas pelas classes que são os atributos independentes e as saídas que são os atributos dependentes, ou seja, o que será previsto.

- Após o carregamento dos arquivos, podemos ver a incidência de casos verdadeiros e falsos, informados no gráfico da linha Linha[11].

- Separa-se os registros em previsores\_treinamento, classe\_treinamento, previsores\_teste, classe\_teste para definir o conteúdo/registro que servirão de base para APRENDIZADO de máquina e outra parte para TESTE do que foi aprendido. Linha[12].

- No caso, particionamos 70% do total de registros para aprendizado e 30% para treino. Resultando na quantidade de registros separados conforme linhas 14 a 17.

- O PyTorch trata os dados no formato de tensor. Convertemos a tabela para uma matriz usando a biblioteca Numpy e, logo em seguida o PyTorch cuida de converter para o formato de Tensor. Linhas[20 e 21].

- Com os dados convertidos, armazenamos em uma variável para ser usada para ajustes dos pesos. Estes pesos são dimensionados pela função PyTorch conforme a influência que exerce sobre outros atributos juntamente com suas previsões. Linha[25-27]

- Cria-se um classificador que irá definir um modelo de rede neural para montar a estrutura combinatória e fazer as previsões.

- A definição de um critério e o modelo de otimização, serão usados para refinar os valores durante o treinamento do modelo. O aprendizado de máquina está sendo desenvolvido neste momento. Linha[32]. A otimização consiste numa “releitura” dos valores lidos anteriormente para fazer um refinamento, melhorando a acurácia de suas previsões.

- No código, linhas abaixo representam a matriz dos valores lidos dentro dos parâmetros.

- Na linha[41], submetemos os valores de TESTE ao mesmo modelo criado anteriormente para ver as previsões obtidas pelo modelo usando o mesmo classificador. Linha[43]

- À seguir, temos as previsões como respostas do teste. Na linha[47] temos a taxa de acerto obtida pela comparação do teste e do que o computador previu. A classe\_teste contém os valores das variáveis dependentes já definidos. O computador fez os cálculos em cima desse teste e comparou com os registros contidos na classe\_teste original. Como resultado, temos uma taxa de acerto de 90%, indicando que o modelo consegue fazer uma boa previsão com os atributos que a tabela original oferece.

- Na linha[48] podemos ver na “matriz de confusão”, a quantidade de erros e acertos numa leitura cruzada em diagonal da parte superior esquerda para a inferior direita que informa as previsões que eram verdadeiras positivas e as verdadeiras negativas. O cruzamento diagonal oposto , lido da parte superior direita para a inferior esquerda informam os valores que a previsão errou.

Obs. Nas linhas do código, contém observações extras que poderão auxiliar no entendimento do trecho de código.