Inteligência Artificial

Trabalho de Implementação - Aprendizado de Máquina - Profa. Heloisa 2020/2

==========

Este trabalho consiste na execução e validação de algoritmos de aprendizado supervisionado para classificação (árvore de decisão) e não supervisionado para agrupamento (K-means) estudados na disciplina, com o uso de classes e funções disponíveis em bibliotecas Python e descritas neste documento. **Todas as funções necessárias estão exemplificadas nas videoaulas do Tópico 7.**

Os conjuntos de dados podem ser selecionados no repositório UCI (http://archive.ics.uci.edu/ml/) ou no site da ferramenta Weka (http://www.cs.waikato.ac.nz/ml/weka/). Pode também ser usado um dos conjuntos de dados do pacote sklearn.datasets, desde que apropriado para o tipo de tarefa, **exceto os conjunto Iris e Linnerrud**.

As etapas que devem ser executadas e relatadas são:

Parte 1 – Avaliação de árvore de decisão com métricas para classificação

- Selecionar conjunto de dados adequado para problemas de classificação (conjunto com classes nominais);
- Se for um conjunto de dados externo a biblioteca scikit-learn, ler o conjunto de dados no formato *.csv* com a função read.csv(...) (biblioteca Pandas)*;*
- Se for um conjunto de dados do pacote sklearn.datasets, carregar o conjunto com a função load_<nome_do_arquivo>;
- Separar o conjunto de dados em matriz de atributos (X) e vetor de classes (y).
- Transformar os atributos nominais em binários (se existirem), usando a classe OneHotEnconder do módulo sklearn.preprocessing ou usando a função get_dummies(...) da biblioteca Pandas;
- Dividir o conjunto de dados em um conjunto de treinamento e um conjunto de teste usando a função train_test_split do módulo sklearn.model_selection;
- Aplicar o algoritmo de indução de árvore de decisão no conjunto de treinamento usando a função fit(...) da classe DecisionTreeClassifier do módulo sklearn.tree, com o atributo criterion='entropy'**;**
- Gerar a figura da árvore de decisão gerada com a função plot_tree(...) do módulo sklearn.tree;
- Fazer a classificação dos dados de teste usando a função predict(...) da classe DecisionTreeClassifier do módulo sklearn.tree;
- Fazer a avaliação do modelo gerado usando os dados de teste e mostrando os resultados das funções confusion_matrix(...) e classification_report(...) do modulo sklearn.metrics.

Parte 2 – Determinação do melhor número de grupos usando a soma quadrática das distâncias como índice interno (Método do "cotovelo")

- Selecionar conjunto de dados adequado para problemas de agrupamento, que seja diferente do usado na parte 1 (se for um conjunto de dados que tem o atributo classe, ela pode ser retirada);
- Se for um conjunto de dados externo a biblioteca scikit-learn, ler o conjunto de dados no formato *.csv* com a função read.csv(...) (biblioteca Pandas)*;*
- Se for um conjunto de dados do pacote sklearn.datasets, carregar o conjunto com a função load_<nome_do_arquivo>;
- Fazer as transformações que forem necessárias: transformar atributos nominais em binários usando a classe OneHotEnconder do módulo sklearn.preprocessing ou usando a função get_dummies(...) da biblioteca Pandas e normalizar os atributos contínuos com a função fit(...) da classe MinMaxScaler do módulo sklearn.preprocessing;
- Definir um intervalo de valores para número de grupos (por exemplo, de 2 a 15);
- Aplicar o algoritmo k-means no conjunto de dados (sem a classe) para todos os valores do intervalo definido usando a função fit(...) da classe Kmeans do módulo sklearn.cluster**;**
- Coletar o valor do índice '*soma quadrática das distâncias*' para todos os agrupamentos encontrados, usando o atributo inertia_ da classe Kmeans do módulo sklearn.cluster**;**
- Plotar o gráfico dos índices encontrados para cada número de grupos e determinar qual é o melhor de acordo com o método do cotovelo explicado na aula.

Entrega:

- O trabalho deve ser acompanhado de um relatório que documente claramente o procedimento adotado e descreva **todas as etapas definidas para a Parte 1 e Parte 2**.
- Pode ser utilizado o framework Jupyter Notebook ou o ambiente Google Colaboratory. Nesse caso, o relatório e o código de execução devem estar no mesmo documento, no formato.ypnb.
- Alternativamente pode ser usada a linguagem Python fora do framework. Nesse caso, o relatório deve entregue em .pdf e o código em formato padrão do Python.
- O trabalho pode ser feito em duplas.
- Entregar (no ava – tarefa de arquivo único) com os arquivos nos formatos especificados acima.
- DATA DE ENTREGA: 15/04/2021