Aprendizagem Automática 2022/23

Ficha prática 10 - Comités de classificadores

Exercício#10.1 Breast cancer

Na ficha prática #9.2 foi proposto uma aplicação duma árvore de decisão ao conjunto dados “breast cancer”. Nesta ficha propõe-se a aplicação dum algoritmo de comité baseado em árvores de decisão. O algoritmo Random Forest.

1. Carregue o conjunto de dados "breast_cancer" usando a função "load_breast_cancer()".

2. Faça uma divisão treino/teste com os valores por omissão (treino=75%, teste=25%).

3. Construa um modelo usando Random Forest com os valores por omissão. Calcule o desempenho sobre ambos os conjuntos. Existem evidências de sobre-ajustamento? Como pode tentar reduzi-lo?

4. Construa novos modelos alterando o nº de atributos testados e/ou aplicando pré-poda. Compare o desempenho entre os vários modelos construídos.

5. À semelhança das árvores de decisão é possível calcular a importância dos atributos. Observe num gráfico de barras, a importância de cada atributo. Compare o gráfico com aquele obtido com uma árvore de decisão, na ficha 9.2.

6. Construa um modelo usando Gradient Boosting Trees com os valores por omissão. Calcule o desempenho sobre ambos os conjuntos. Existem evidências de sobre-ajustamento? Como pode tentar reduzi-lo?

7. Teste novos modelos alterando a pré-poda e/ou taxa de aprendizagem. Compare o desempenho obtido.

8. Apresente um gráfico de barras que mostre a importância de cada atributo. Compare o gráfico com aqueles obtidos com uma árvore de decisão e Random Forest.

9. Pode otimizar os algoritmos de comité fazendo um fine-tune de hiperparâmetros com a função GridSearchCV.

Por fim, deve-se analisar sempre de um modo crítico os resultados em termos de desempenho versus a complexidade, e peso computacional, dos algoritmos e procedimentos que utilizou: o uso de algoritmos de comité melhorou o desempenho significativamente ? a otimização / Grid Search melhorou quanto o desempenho anterior ? Pode ainda avaliar o tempo necessário para correr os seus programas usando o módulo time (módulo standard do Python) com a função time.time(), e comparar o desempenho atingido versus o tempo computacional necessário.