Relatório do Projeto - Paradigmas da Programação

Leonardo Carmo

Descrição

O projeto consiste em implementar uma versão do algoritmo Floresta Aleatória utilizando a linguagem Haskell. Este algoritmo consiste em gerar diversas Árvores de Decisão e prever basendo no voto gerado pela maioria das árvores individuais. Cada árvore é gerada utilizando um conjunto amostrado do total de exemplos com reposição (bootstrap) e somente parte das possíveis vaiáveis preditoras, amostradas sem reposição.

Nesta implementação, as variáveis preditoras podem ser tanto categóricas quanto numéricas, contudo os tipos necessitam ser consistentes entre a mesma "coluna". Além disso, todo número encontrado será tratado como um tipo númerico. Para gerar as árvores individuais, o critério de seleção utilizado foi o índice gini e todo nó possui apenas dois filhos.

Arquivos

- *ReadData*: Função utilizada para ler um arquivo *.csv* e gerar os dados passados para o algoritmo. Possui a definição do tipo *Exemple*, que guarda cada par de lista de variáveis preditoras e a variável dependente. Assume que os dados do arquivo já foram tratados (sem nulos e tipos consistentes) e com a variável dependente sendo a última coluna.
- *Aleatorio*: Possui as funções responsáveis por realizar a amostragem dos exemplos dado uma semente.
- *DecisionTree*: Implementação de uma Árvore de Decisão binária de classificação utilizando gini como critério de separação. Possui uma função auxiliar para calcular a acurácia dos resultados.
- *RandomForest*: Utiliza dos arquivos descritos anteriormente para gerar uma Floresta Aleatória.
- *Main*: Arquivo utilizado para definir os parâmetros e executar o algoritmo.

Parâmetros

- *minSampleSplit*: Número mínimo de exemplos em um nó para tentar separá-lo em nós filhos.
- *minSampleLeaf*: Número mínimo de exemplos em um nó filho para ser uma separação válida.
- *maxDepth*: Profundidade máxima da árvore gerada, sendo uma árvore de tamanho 1 uma única folha.
- *nEstimators*: Número de Árvores de Decisão geradas pela Floresta Aleatória.
- *nSamples*: Quantidade de amostragens a serem feitas nos exemplos para cada árvore.
- *nFeatures*: Quantidade de variáveis preditoras a serem utilizadas para cada árvore.

Como Utilizar

Modificar no arquivo *Main* os parâmetros desejados e especificar os nomes dos arquivos de treino e de teste.

Após configurado, apenas utilizar `stack run` no terminal para executá-lo.

Dificuldades Encontradas

As principais dificuldades encontradas durante o processo podem ser sumarizadas em alguns pontos:

- Houve um atraso inicial para compreeder a estruturação do arquivo cabal para permitir a separação do projeto em múltiplos arquivos e importar bibliotecas externas.
- Compreender a utilização de bibliotecas externas devido a falta de um maior número de exemplos simples, utilizando as principais funções da mesma. Por exemplo, inicialmente seria utilizado a biblioteca cassava, porém foi descartada na dificuldade de gerar uma lista com as variáveis preditoras e a classe para cada exemplo sem assumir conhecimento prévio do arquivo a ser lido.
- Realizar a estruturação dos tipos e funções para gerar os algoritmos, devido ao grande número de etapas e (aparente) ausência de funções facilitadoras, como permitir indexar elementos de listas de listas como uma matrix para selecionar colunas.

Vídeo

O vídeo demonstando a utilização do projeto pode ser acessado no link a seguir:

https://drive.google.com/file/d/1z08S2vxAZhab2f3GGMBuFO__YZDICZdk/view?usp=sharing