Analyse

1. Decision tree

* 說明方法:

使用到的operator和parameter

1. 處理樹節點的struct，裡面有存子節點的vector、節點代表的index值為int、attribute和label的值為double，以及表示是否為葉節點的bool值為bool
2. 處理模型資料的struct，裡面有存attribute名稱的vector、存輸入資料的vector、存attribute值的vector，以及取得attribute值的function
3. 建立決策樹的struct，裡面有(2的模型資料、樹的結構為vector、決策樹的constructor、build用來實作樹的連接
4. 建立決策樹中包含使用dfs遍歷決策樹的function、C4.5演算法中需要的function，getchoose用來找最大attribute的index，getGainRatio是計算增益率為了選擇分裂屬性，getGain是計算訊息增益，getInfoD計算entropy，還有用來分析模型的運算macro f1-score，所以使用了二維陣列儲存統計的實際值和預測值的關係。

* Performance:

Solution quality: 根據macro f1-score對模型的分析，得到兩個結果，一是使用模型預測模型資料的分數，計算結果是0.966335，二是使用模型預測測試資料的分數，計算結果是0.768770。

Running time: 執行時間花費8.127秒

1. Nearest neighbor

* 使用到的operator和parameter:

1. 存輸入資料的struct，DataPoint裡面有atrribute值跟label值
2. readCSV將資料從csv檔讀取並存到資料結構中
3. calculateDistance用來計算兩個DataPoint的Euclidean distance
4. customPartition利用中心值對資料進行分割
5. quickSelect用來找第K個最短距離
6. knnClassify利用KNN進行分類，找出與測試資料最鄰近的K個點，並得到最多的label
7. processPart用來對測試資料的一部份進行KNN分類，並存結果到result
8. calculateF1Score用來計算KNN分類的 macro F1-score，其中還會用到calculatePrecision對應到P值，calculateRecall對應到R值。

* Performance:

Solution quality: 根據macro f1-score對模型的分析，得到兩個結果，一是使用模型預測模型資料的分數，計算結果是0.956446，二是使用模型預測測試資料的分數，計算結果是0.915423。

Running time: 執行時間花費54.92秒

1. 分析以上兩種演算法:

Decision tree和nearest neighbor這兩種分類演算法分別使用不同的方法，前者使用tree結構上的attribute來進行分類，後者則利用找資料間的Euclidean distance來進行分類，從實驗結果上分析，我發現nearest neighbor在預測上的正確率較高，而Decision tree在執行時間上花費較少時間。