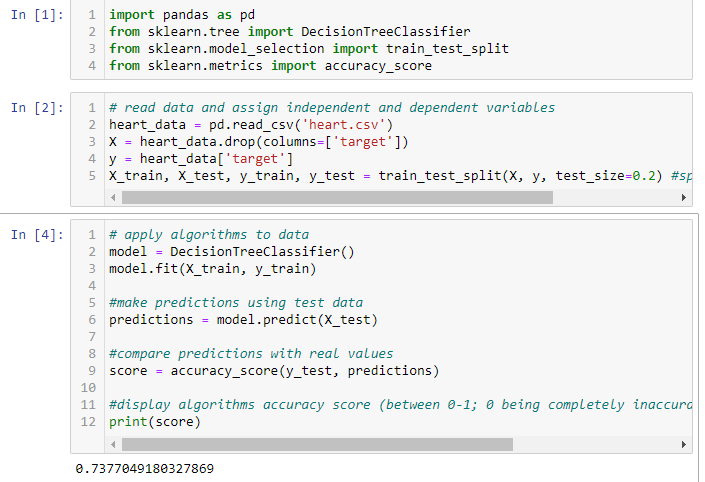
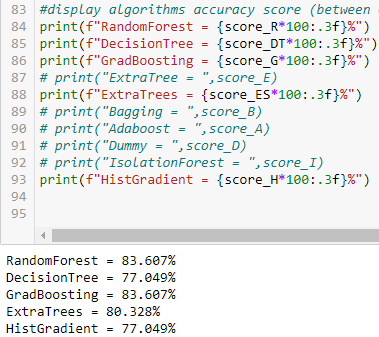
BioDigital Machine Learning (stages)

-Research on machine learning fundamentals (DecisionTreeClassifier)  
The first stage was understanding how machine learning worked and how to use it. The most widely accepted editor to run machine learning models on: was Jupyter Notebook. The screenshots shown in this document are Python code run on Jupyter Notebook to develop Machine Learning models.  
The initial step to develop a model was to run an algorithm on a dataset and let it learn the data to be able to make predictions. The best algorithm to start with was the DecisionTreeClassifier, since it was efficient and overall easier to use. Before data was acquired, the algorithms were tested on sample datasets with similar structures. The first dataset tested was on heart.csv, a dataset showing whether or not a patient had heart disease and the different information which correlated to their result (e.g. age, sex, maximum heart rate, cholesterol etc.) .



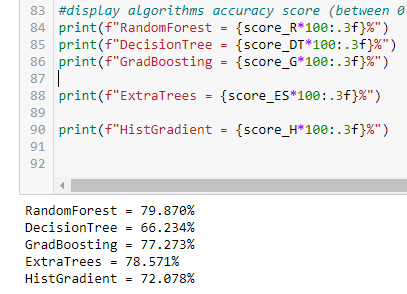
The dataset was split into two parts in the ratio 4:1, 80% used to train the model and 20% held separately to test the accuracy of the model. The accuracy of the DecisionTreeClassifier model is display at the very bottom of the screenshot (0.7377), approximately 73.8% accuracy.

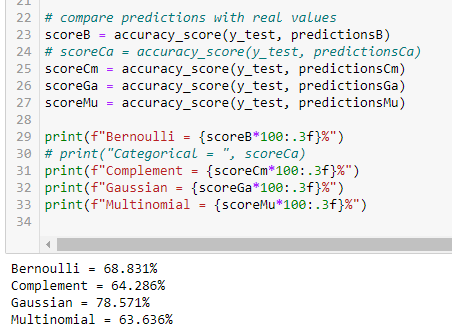
-Exploring different machine learning algorithms  
The next step was to research different algorithm classifiers and test them on the dataset.

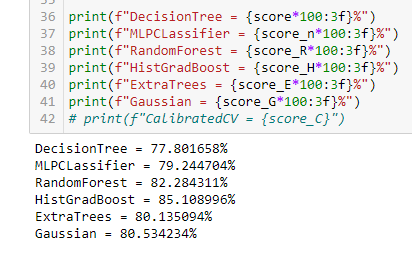
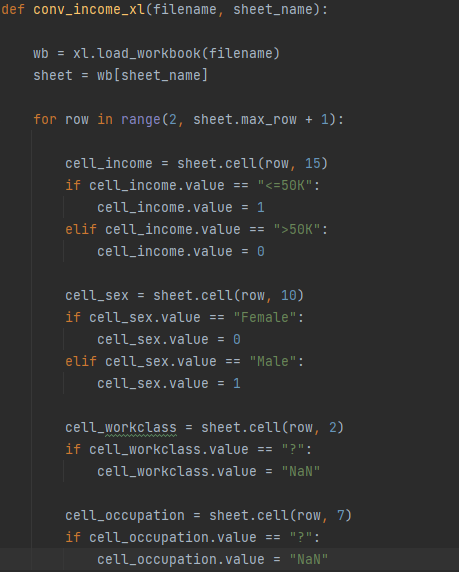


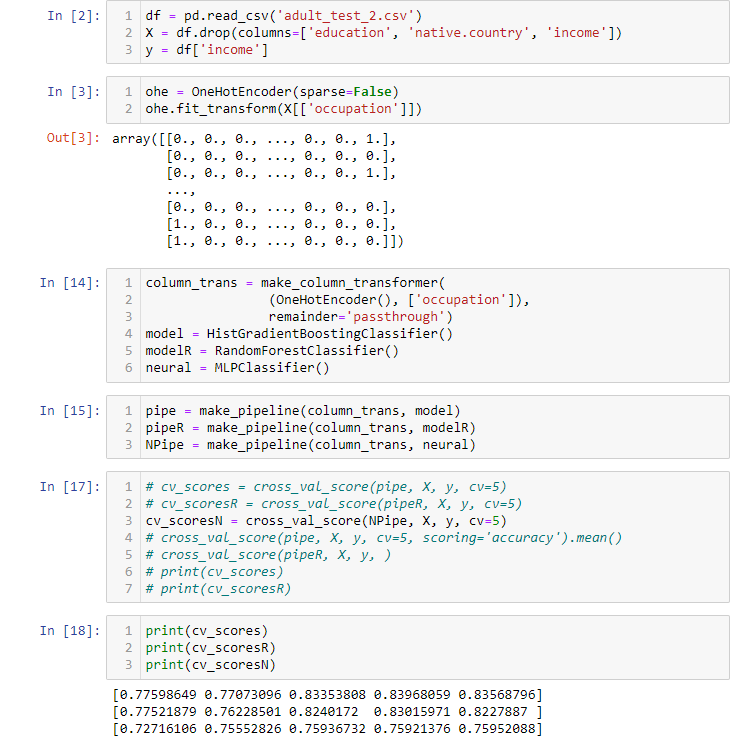
The accuracy of different algorithm classifiers that were explored are shown above. The other classifiers that did not work well with the dataset were hidden for the purposes of this document.

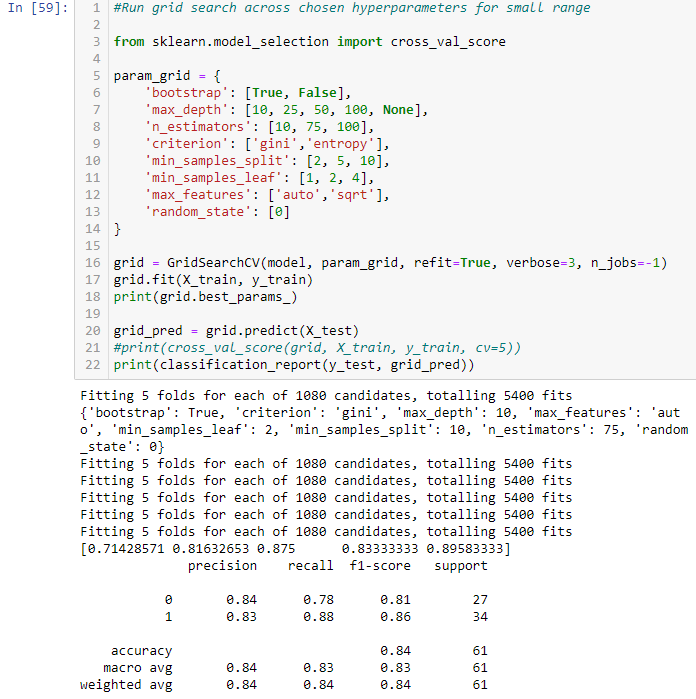
-Testing ensemble classification algorithms on sample datasets (heart.csv, diabetes.csv)  
The algorithms which were made up of multiple decision trees outperformed the single tree classifiers (DecisionTree, ExtraTree). A comparison of these ‘ensemble’ classifiers were carried out on both the heart.csv and a slightly larger dataset (diabetes.csv, approximately twice the size of heart.csv).



-Testing Support Vector algorithms, Naïve Bayes algorithms accuracy on heart.csv  
Other groups of classifiers were also tested using the available datasets.  
  


-Testing of best performing (manually selected) algorithms (HistGrad, RanFor, Gaus, ExtraTrees) on larger dataset (adult.csv).  
Since the data to be worked on is expected to have a large amount of entries, a larger sample dataset is required to see how the classifier algorithms perform.  
  
-Investigation of pre-processing data methods- imputing (through prediction/substitution), removing entries.  
Before the models could be developed some data preprocessing steps had to take place to clean the dataset. The previous datasets were already fairly clean datasets so no preprocessing was required. Some preprocessing, using an automation function on a separate editor (PyCharm), was used to clean the data.  


-Implementation of preprocessing tools (libraries), such as OneHotEncoder. Used on adult.csv to factor in features such as occupation.  
Another way to implement the preprocessing steps was to use an inbuilt tool called OneHotEncoder, which can convert the string data into a readable array format for the classifiers.  
  
-Implementation of cross\_val\_score tool on existing scripts (test on diabetes.csv, heart.csv, adult.csv).  
A more reliable method of measuring the performance of the classifiers was to use a cross validation method. This splits the data into n number of folds and assigns each fold as either the test or train set and outputs a score of the model. A mean of these scores can then be calculated and compared with the other classification algorithms.  
-Comparing ensemble classifiers (GradientBoosting, RandomForest, ExtraTrees, HistGrad) on covtype.csv (500k+ entries).  
Classifiers were compared on currently, the largest dataset, close to the expected entries of the data.  


-Optimisation of RandomForest (manual max\_depth alterations) and GradientBoosting (manual learning rate/max\_depth/estimators alterations) classifiers on diabetes.csv, at constant states.  
Carrying out further research uncovered that algorithms could potentially be optimised to a dataset by altering their hyperparameters. Initially some manual alterations were carried out and the best scoring hyperparameters values are displayed below.  
  
-Using grid search methods to tune parameters for different algorithms (ExtraTrees, RandomForest) on iris/breast cancer datasets.  
Instead of manual search for the best hyperparameters, an inbuilt tool was available to carry out the task. However this process is computationally expensive and takes a lot of time, so the tests were executed on smaller datasets (breast cancer and iris datasets on sklearn).  
  
RandomForest grid search is shown above. GradientBoosting grid search is shown below.  
