Sepsis Scoring System and Mortality Prediction

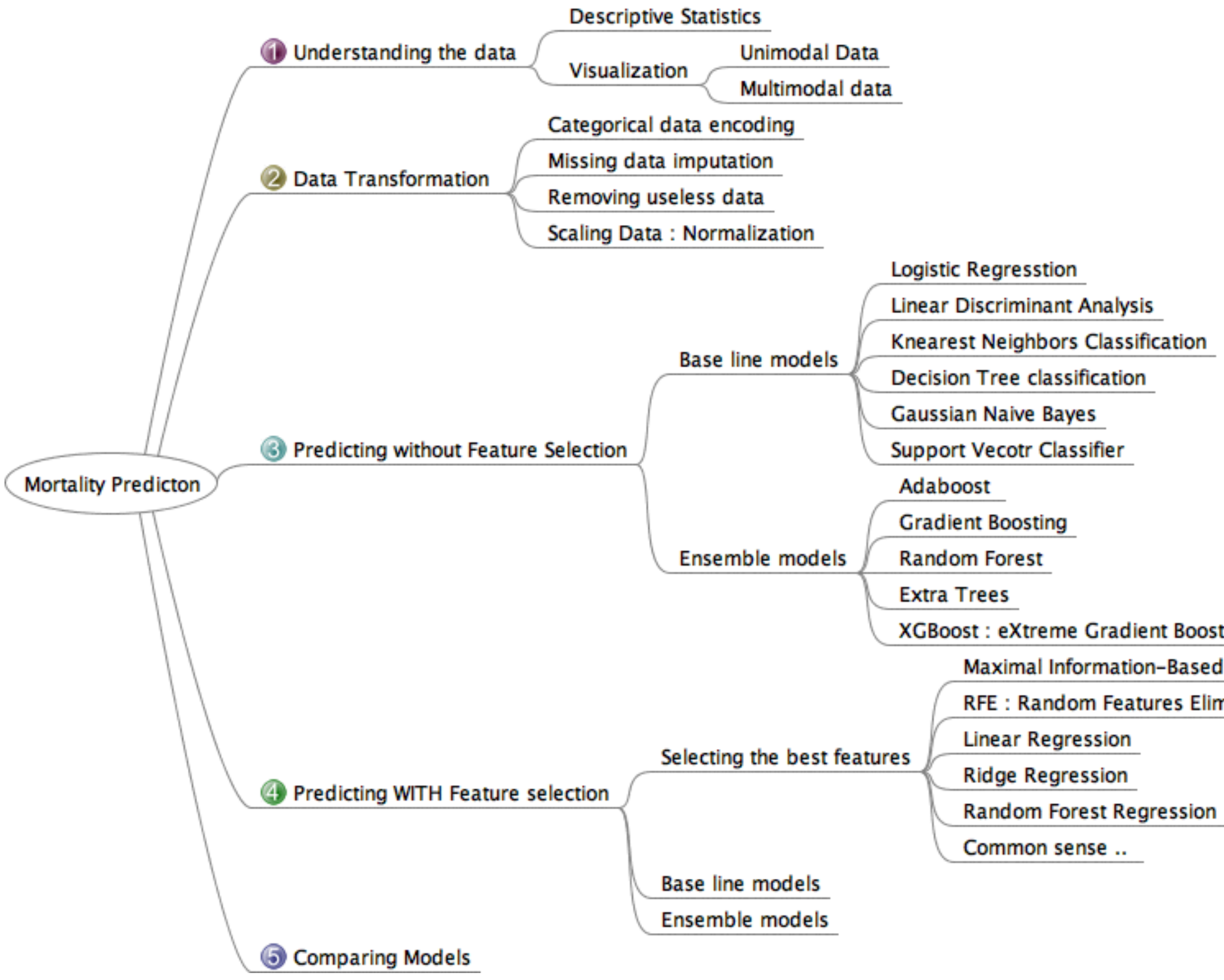
***this presentation is for results reporting, for details and explanations check the notebook attached. Notebook's outline is in the next slide for reference**

Notebook's Outline

- Sepsis scoring system and mortality prediction
- Datasets
- Phase 1 : predicting mortality
  - Methodology
  - Analysis Environment
  - Understand the data
  - Predicting outcome WITHOUT feature selection
  - Predicting outcome WITH feature selection
  - Classification across all datasets
- Phase 2 :Visualizing thresholds

PHASE I

Analysis protocol



Analysis protocol

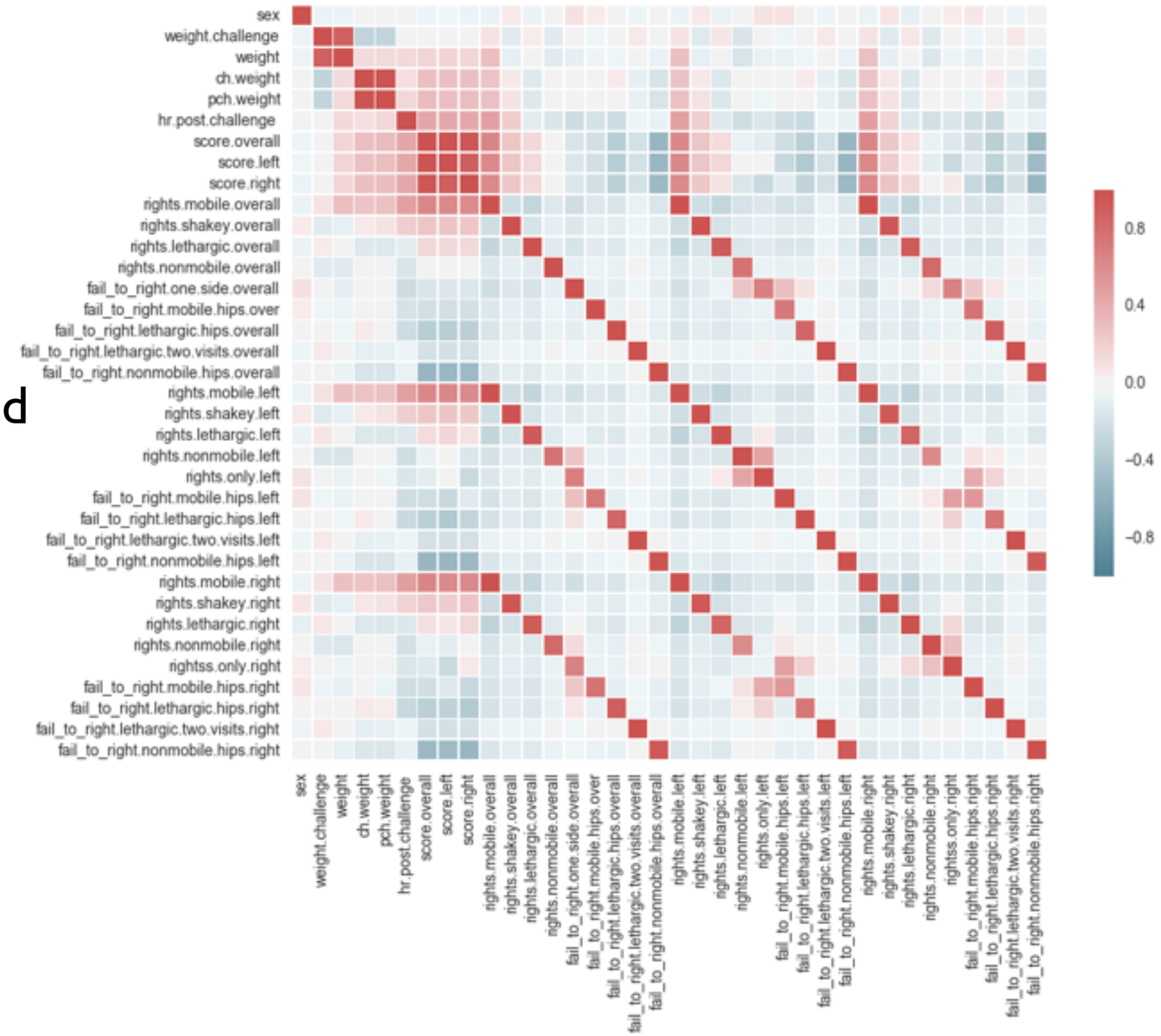
Note : All the results you will see in the following slides are made on the full vertical dataset, No BCG.

We have 4 data sets in total , a slide summarizing results on these datasets will be provided at the end

- Full dataset NO BCG
- NO BCG 24h
- Full dataset BCG
- BCG 24h

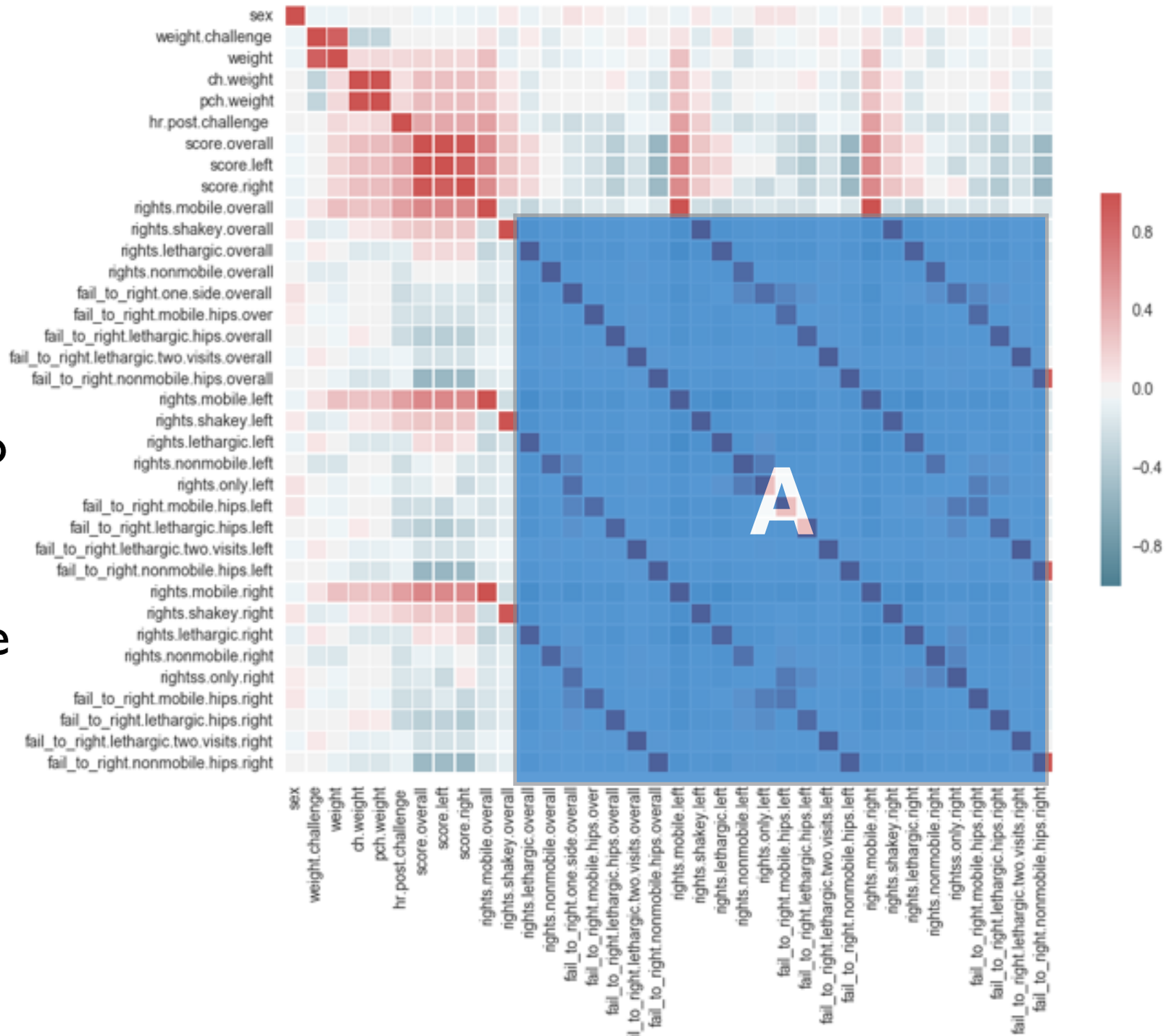
Sepsis Data : correlated features

- Lots of correlated observations
- Correlated attributes tend to be eliminated by classification algorithms



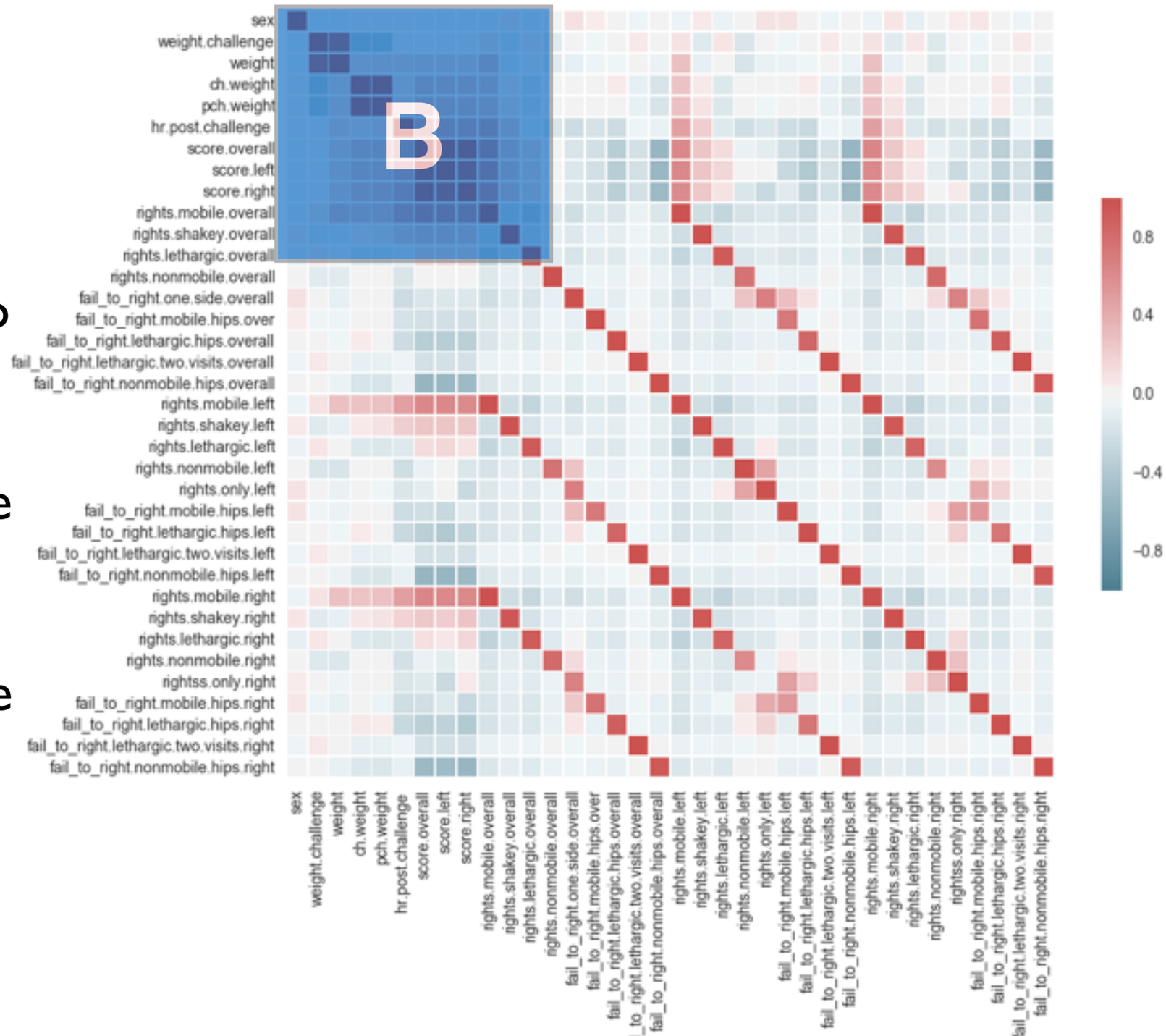
Sepsis Data : correlated features

- Lots of correlated observations
- Correlated attributes tend to be eliminated by classification algorithms
- Features in (A) will tend to be ignored by most classification algorithms since they tend to describe the data the same way



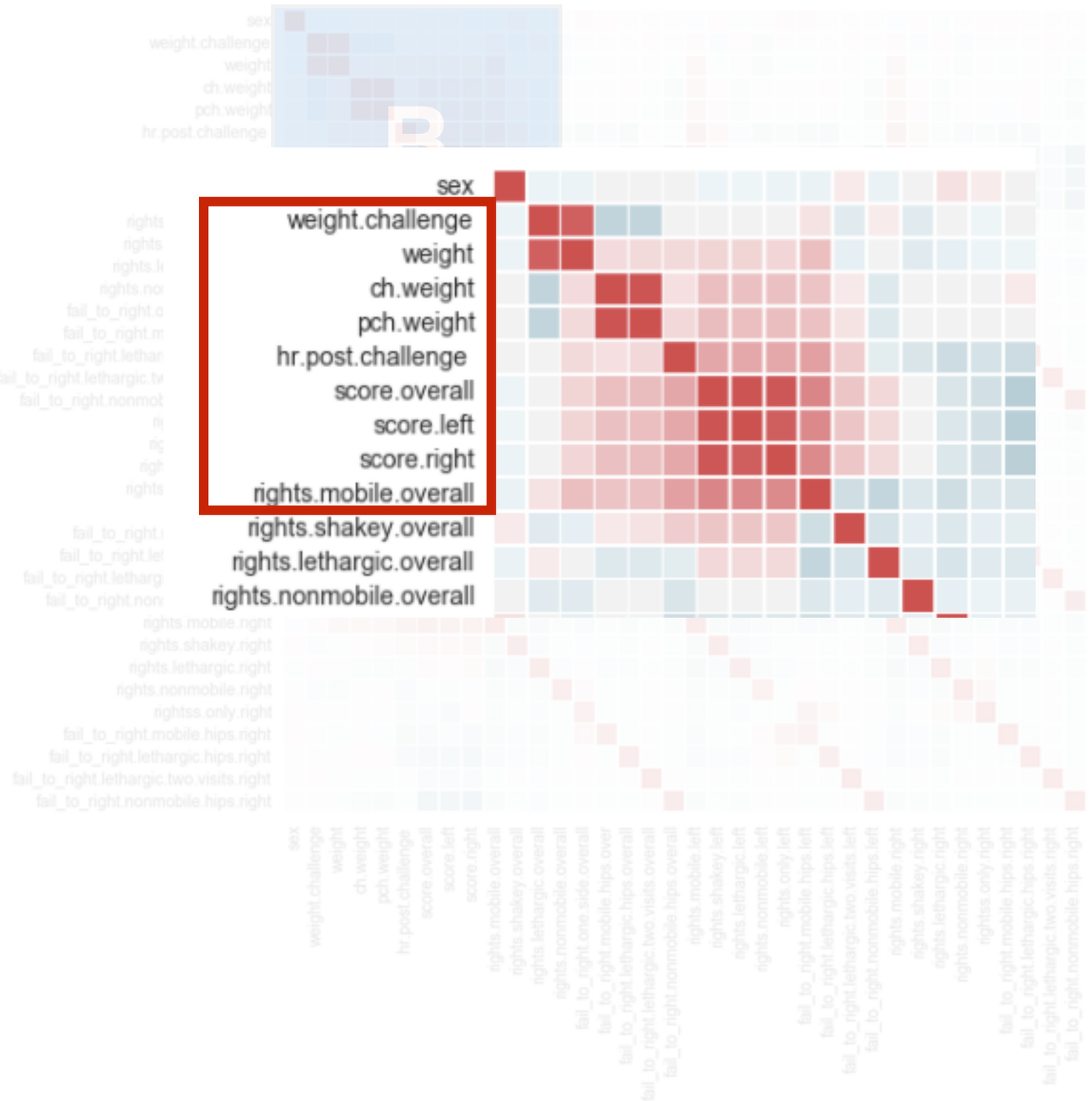
Sepsis Data : correlated features

- Lots of correlated observations
- Correlated attributes tend to be eliminated by classification algorithms
- Features in (A) will tend to be ignored by most classification algorithms since they tend to describe the data the same way
- Features in (B) will be more deterministic, feature selection algorithms will tend to pick those for features importance detection



Sepsis Data : correlated features

- Lots of correlated observations
- Correlated attributes tend to be eliminated by classification algorithms
- Features in (A) will tend to be ignored by most classification algorithms since they tend to describe the data the same way
- Features in (B) will be more deterministic, feature selection algorithms will tend to pick those for features importance detection



Classification : Algorithms used

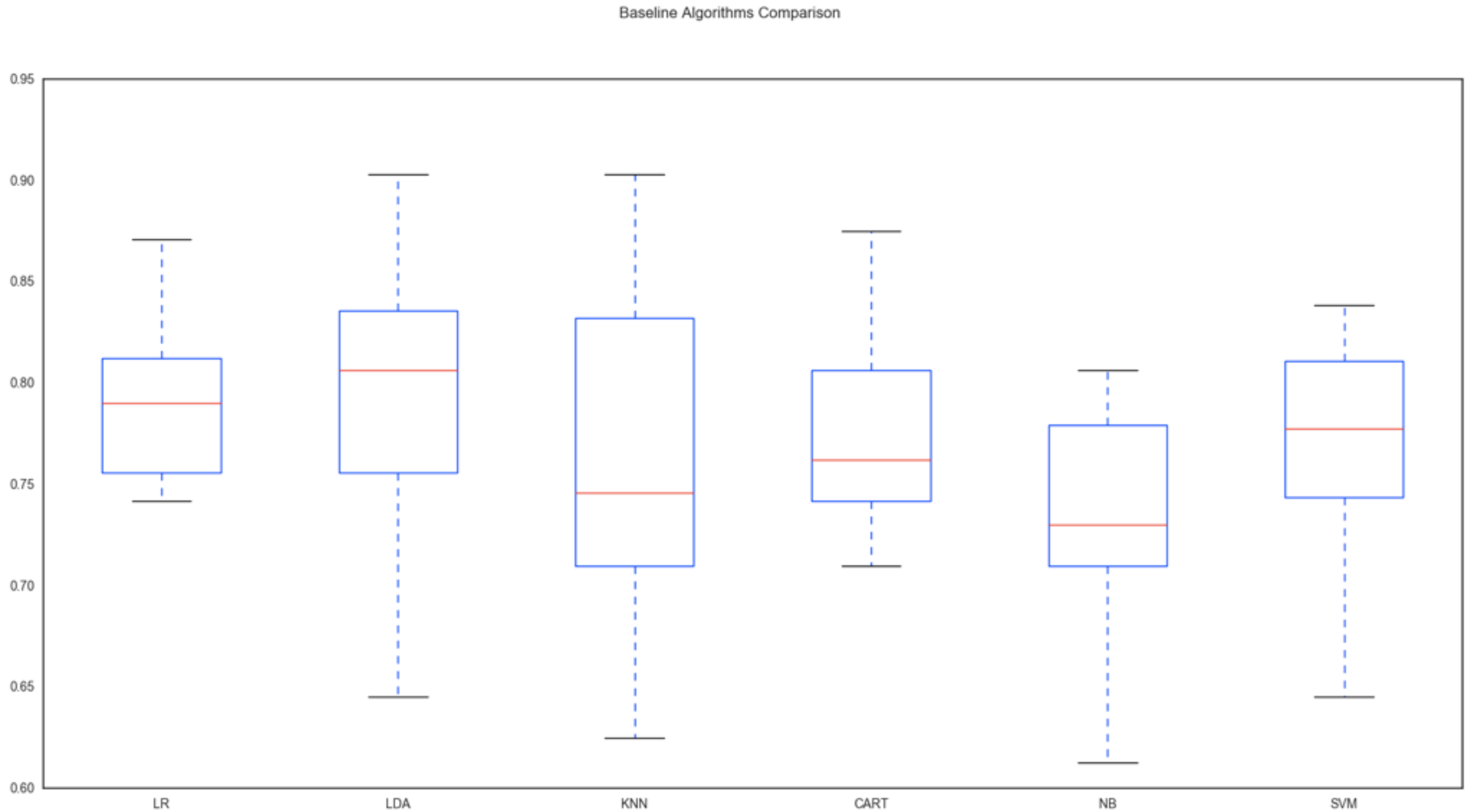
- In order to classify accurately the data, we need to benchmark different algorithms and test for accuracy and ROC
- We always hold 20% of the data for validation, the other 80% are used for classification/testing
- To avoid overfitting, we take some preventive measure, we standardize the data (same scale for all data, variance 0 and std 1) and we do cross-validation when we build the model to make sure models will be generalizable

Predicting mortality without feature selection

Baseline Algorithms

Predicting mortality without feature selection

Baseline Algorithms



Predicting mortality without feature selection

Baseline Algorithms

Algorithm	Accuracy
Logistic Regression	0.78226
Least Discriminant Analysis	0.795464
K Nearest Neighbour	0.760685
CART (decision tree)	0.776310
Naive Bayes	0.734879
Support Vector Machine	0.773085

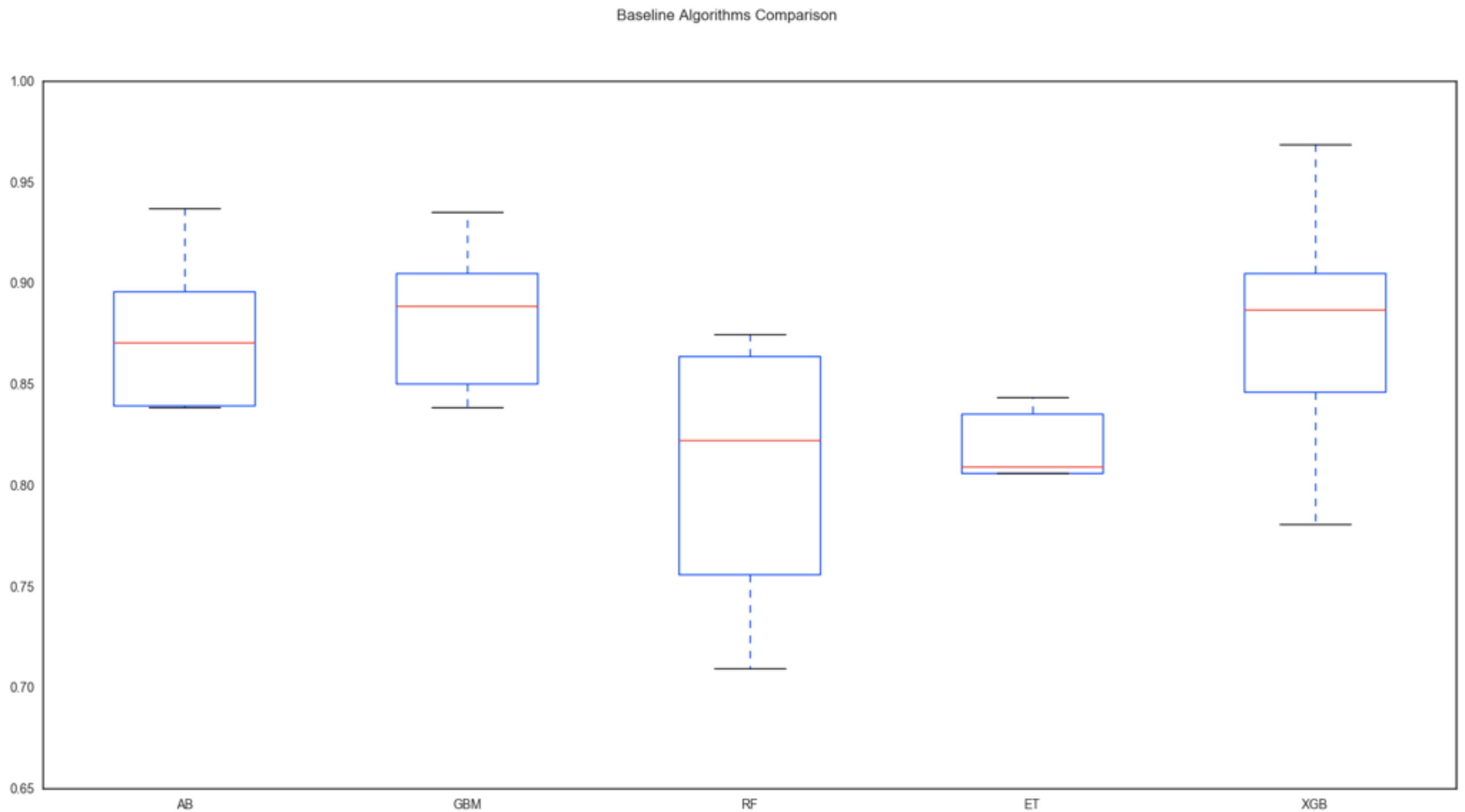
- Decent classification without feature selection
- LDA seems to outperform the other algorithms

Predicting mortality without feature selection

Ensemble Algorithms

Predicting mortality without feature selection

Ensemble Algorithms



Predicting mortality without feature selection

Ensemble Algorithms

Algorithm	Accuracy
Adaboost	0.872077
Gradient Boosting	0.872177
Random Forest	0.804940
Extended Trees	0.811391
XGboost	0.868851

- Boosting Algorithms definitely outperforms the first round of classifications
- Random Forest and Extended Trees perform less than the other methods most likely because of the noise created by correlated features that alter the classification accuracy
- AB, GBM and XGboost win this round of classifications

Predicting mortality without feature selection

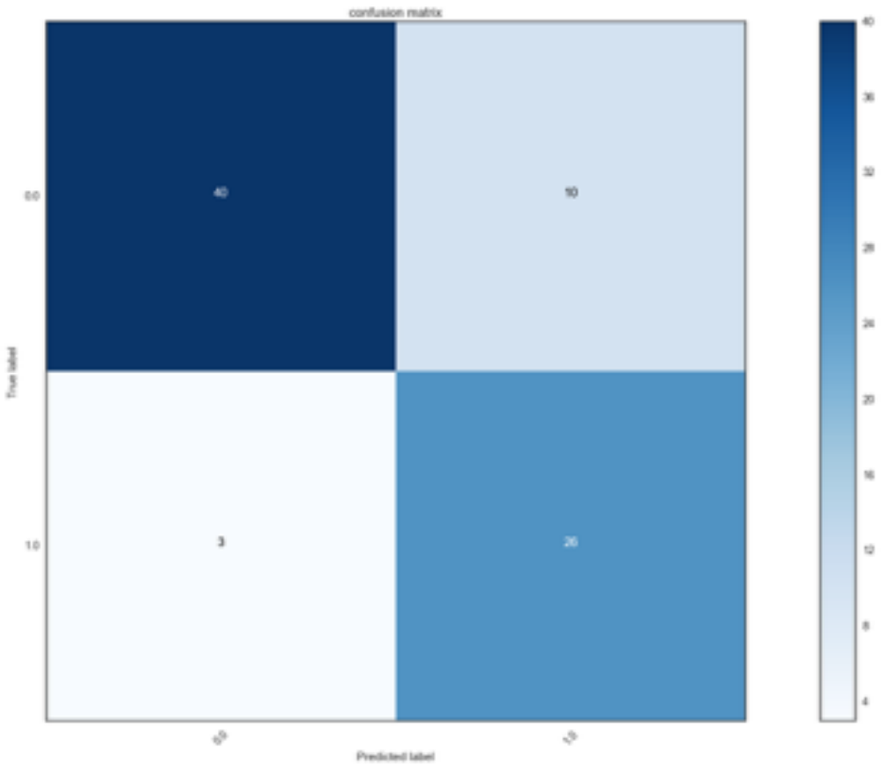
Validation using the 20% dataset

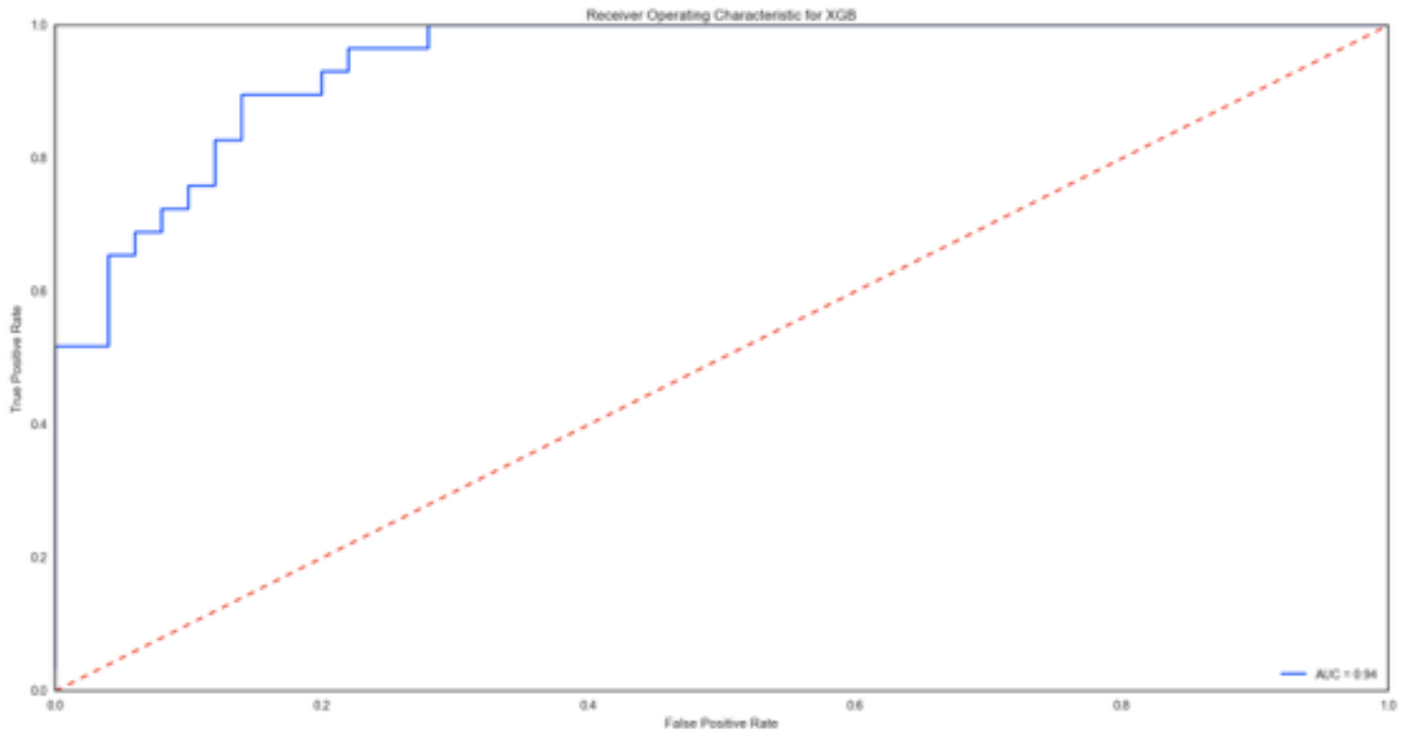
Algorithm	Accuracy	ROC
Adaboost	0.822784810127	0.89
Gradient Boosting	0.835443037975	0.95
Random Forest	0.79746835443	0.91
Extended Trees	0.784810126582	0.85
XGboost	0.835443037975	0.94
Logistic Regression	0.79746835443	0.90
LDA	0.784810126582	0.87
K Nearest Neighbour	0.79746835443	0.85
CART (decision tree)	0.810126582278	0.81
Naive Bayes	0.79746835443	0.88
Support Vector Machine	0.79746835443	0.87

Predicting mortality without feature selection

Validation using the 20% dataset

- Even without feature selection we can accurately predict the outcome (live/die) using the boosting algorithms
- Gradient Boosting and XGboost are the big winners of this first classification





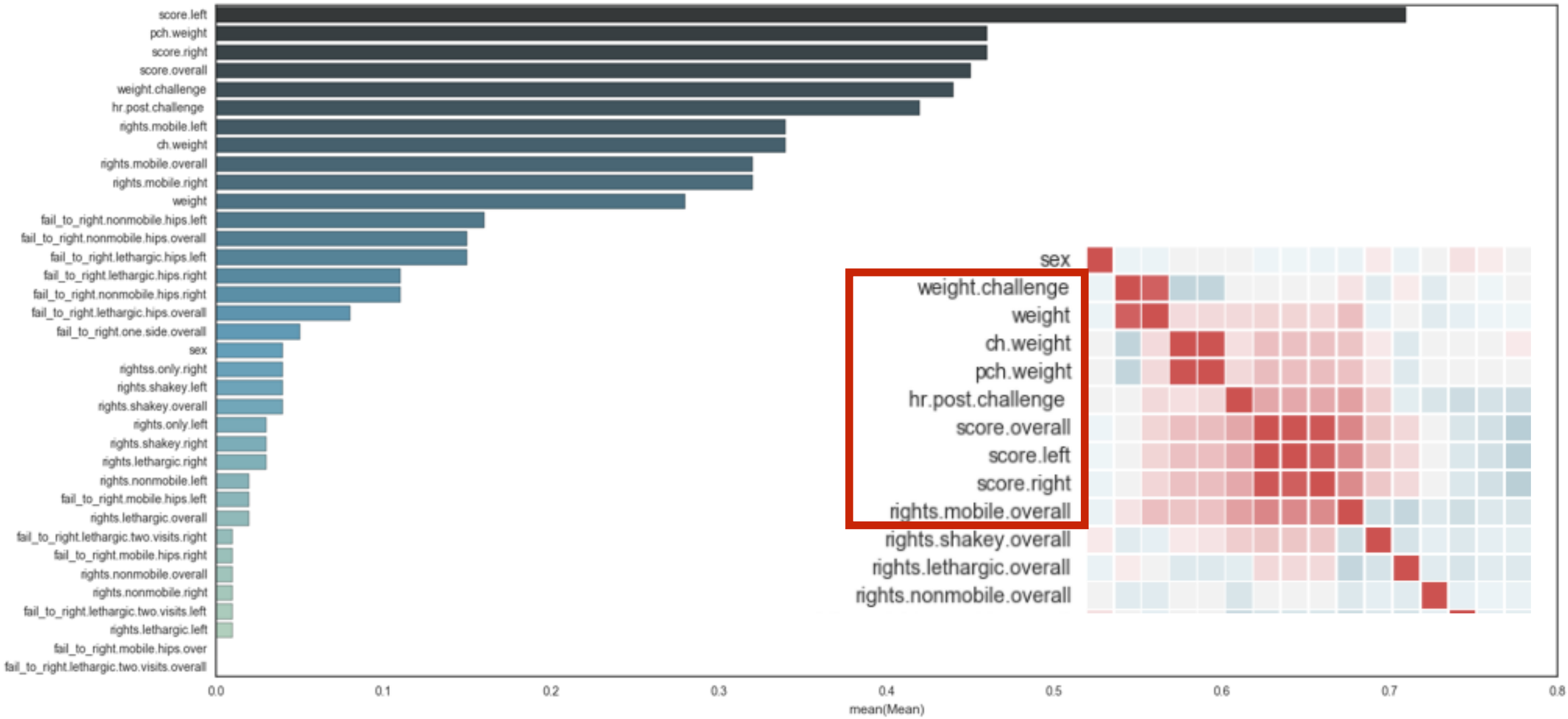
Predicting mortality with feature selection

Feature Selection

- The purpose of this step is eliminate features that are useless. We can predict outcome accurately without feature selection this means that there is a chance we do even better when we reduce the noise caused by features that are not useful for the classification
- There are different ways to consider an important feature :
  - A feature may be important independently of other features
  - A feature may be weak but combined with another weak feature they can be considered as an important one
  - A feature may be deterministic, but combined with other feature it can become a noisy feature
- There are different algorithms to 'predict/calculate' feature importance
- We opted to use different approaches and get the features that perform well in most cases.

Predicting mortality with feature selection

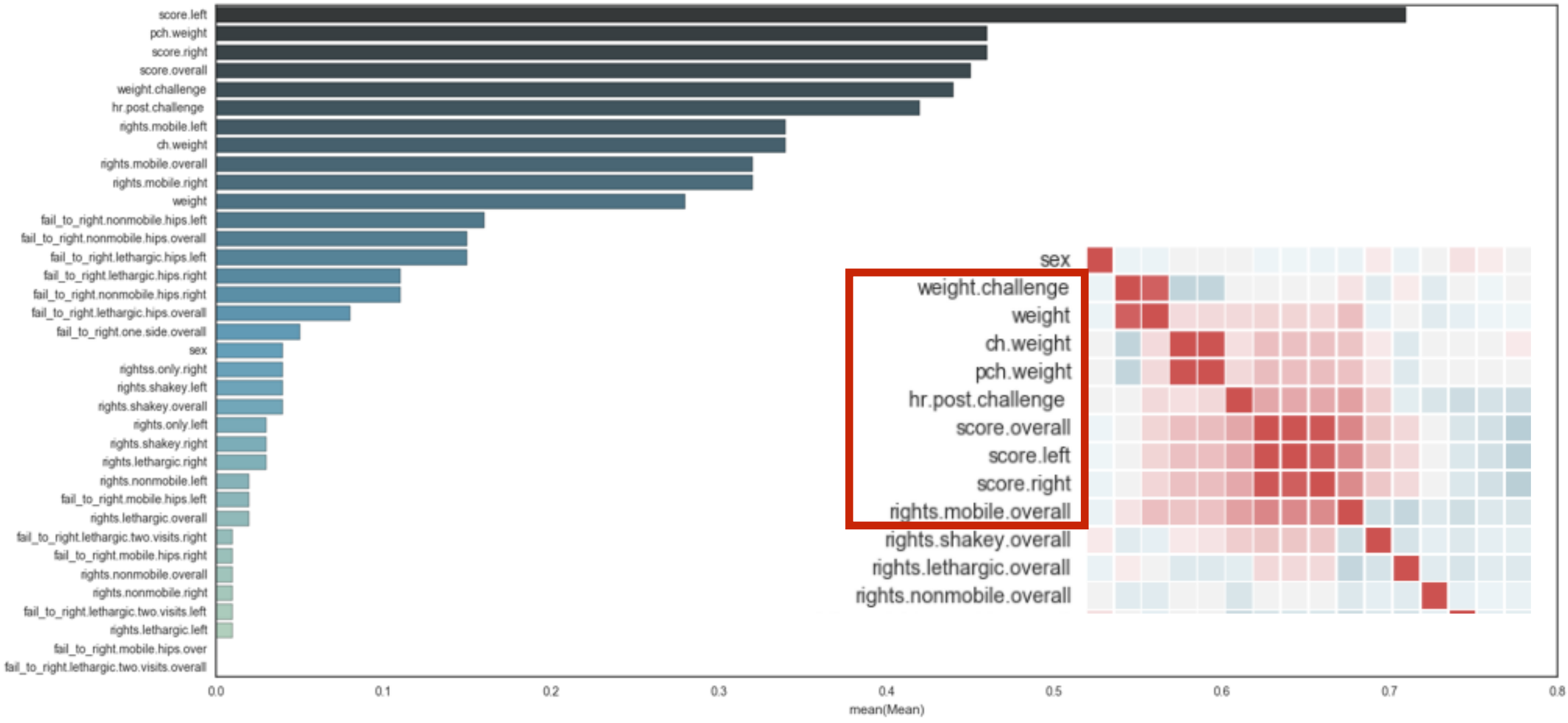
Feature Selection



- There result of feature selection reminds the first part of the analysis where we visually inspected feature correlation, which reinforce that testing different FS methods and using the mean score could be a good approach

Predicting mortality with feature selection

Feature Selection



- Features that we will be using in this section will be then : "weight.challenge", "pch.weight", "score.left", "score.right", "score.overall", “hr.post.challenge"

Predicting mortality with feature selection

Baseline Algorithms

Algorithm	Accuracy (old)	Accuracy (new)
Logistic Regression	0.78226	0.795665
Least Discriminant Analysis	0.795464	0.818044
K Nearest Neighbour	0.760685	0.763609
CART (decision tree)	0.776310	0.804839
Naive Bayes	0.734879	0.824496
Support Vector Machine	0.773085	0.801915

- Feature reduction/selection helped to improve prediction for baseline algorithms, making them perform like ensemble methods on the first round (almost, comparable)

Predicting mortality with feature selection

Ensemble Algorithms

Algorithm	Accuracy (old)	Accuracy (new)
Adaboost	0.872077	0.856048
Gradient Boosting	0.872177	0.868750
Random Forest	0.804940	0.817742
Extended Trees	0.811391	0.792540
XGboost	0.868851	0.891028

- Interestingly, ensemble method did not profit from feature selection, except xgboost which is now performing better than both methods combined, without and with feature selection
- What will be interesting to see is if this feature selection will help XGboost predict accurately on the validation set

Predicting mortality with feature selection

Validation using the 20% dataset

Algorithm	Accuracy(old)	ROC(old)	Accuracy(new)	ROC(new)
Adaboost	0.822784810127	0.89	0.886075949367	0.93
Gradient Boosting	0.835443037975	0.95	0.886075949367	0.97
Random Forest	0.79746835443	0.91	0.822784810127	0.91
Extended Trees	0.784810126582	0.85	0.822784810127	0.92
XGboost	0.835443037975	0.94	0.886075949367	0.96
Logistic Regression	0.79746835443	0.90	0.835443037975	0.89
LDA	0.784810126582	0.87	0.835443037975	0.90
K Nearest Neighbour	0.79746835443	0.85	0.848101265823	0.94
CART (decision tree)	0.810126582278	0.81	0.810126582278	0.81
Naive Bayes	0.79746835443	0.88	0.810126582278	0.88
Support Vector Machine	0.79746835443	0.87	0.79746835443	0.89

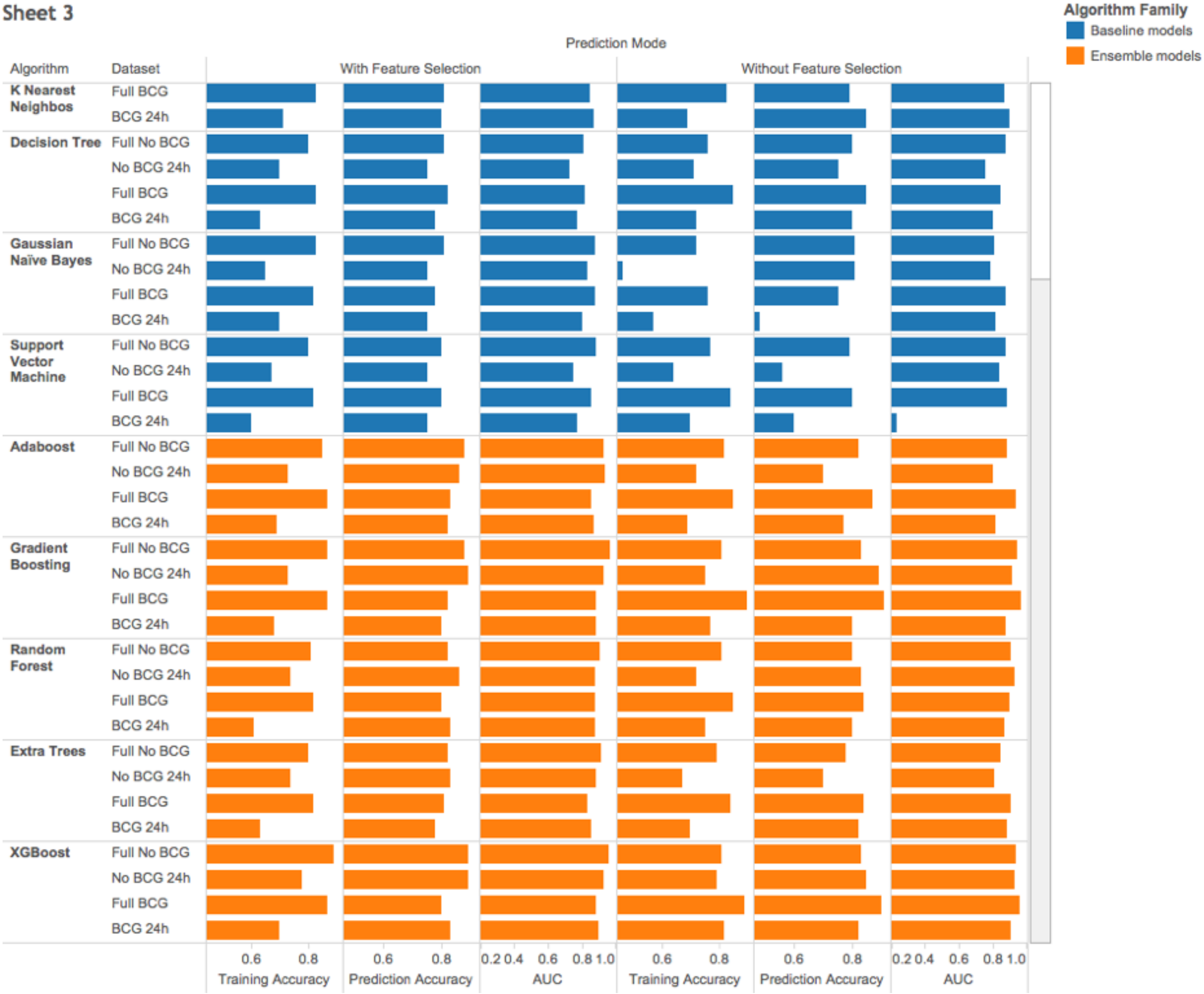
Predicting mortality with feature selection

Validation using the 20% dataset

- The feature selection definitely improves mortality prediction
- The result was reproduced in all datasets (bcg/no bcg, full/24h)
- The features that are predicted to be important are almost the same across all sites.
- Ensemble methods are definitely better than the other classifiers
- XGboost outperforms the other algorithms but overall the classification accuracy is decent for all the classifiers.

Predicting mortality : All in one plot

Sheet 3


Algorithm Family
Baseline models
Ensemble models
Prediction Mode
Algorithm
Dataset
With Feature Selection
Without Feature Selection
K Nearest Neighbos
Full BCG
BCG 24h
Decision Tree
Full No BCG
No BCG 24h
Full BCG
BCG 24h
Gaussian Naïve Bayes
Full No BCG
No BCG 24h
Full BCG
BCG 24h
Support Vector Machine
Full No BCG
No BCG 24h
Full BCG
BCG 24h
Adaboost
Full No BCG
No BCG 24h
Full BCG
BCG 24h
Gradient Boosting
Full No BCG
No BCG 24h
Full BCG
BCG 24h
Random Forest
Full No BCG
No BCG 24h
Full BCG
BCG 24h
Extra Trees
Full No BCG
No BCG 24h
Full BCG
BCG 24h
XGBoost
Full No BCG
No BCG 24h
Full BCG
BCG 24h
0.6 0.8
Training Accuracy
0.6 0.8
Prediction Accuracy
0.2 0.4 0.6 0.8 1.0
AUC
0.6 0.8
Training Accuracy
0.6 0.8
Prediction Accuracy
0.2 0.4 0.6 0.8 1.0
AUC

PHASE II

Sacrificing Accuracy for Interpretability

PHASE II

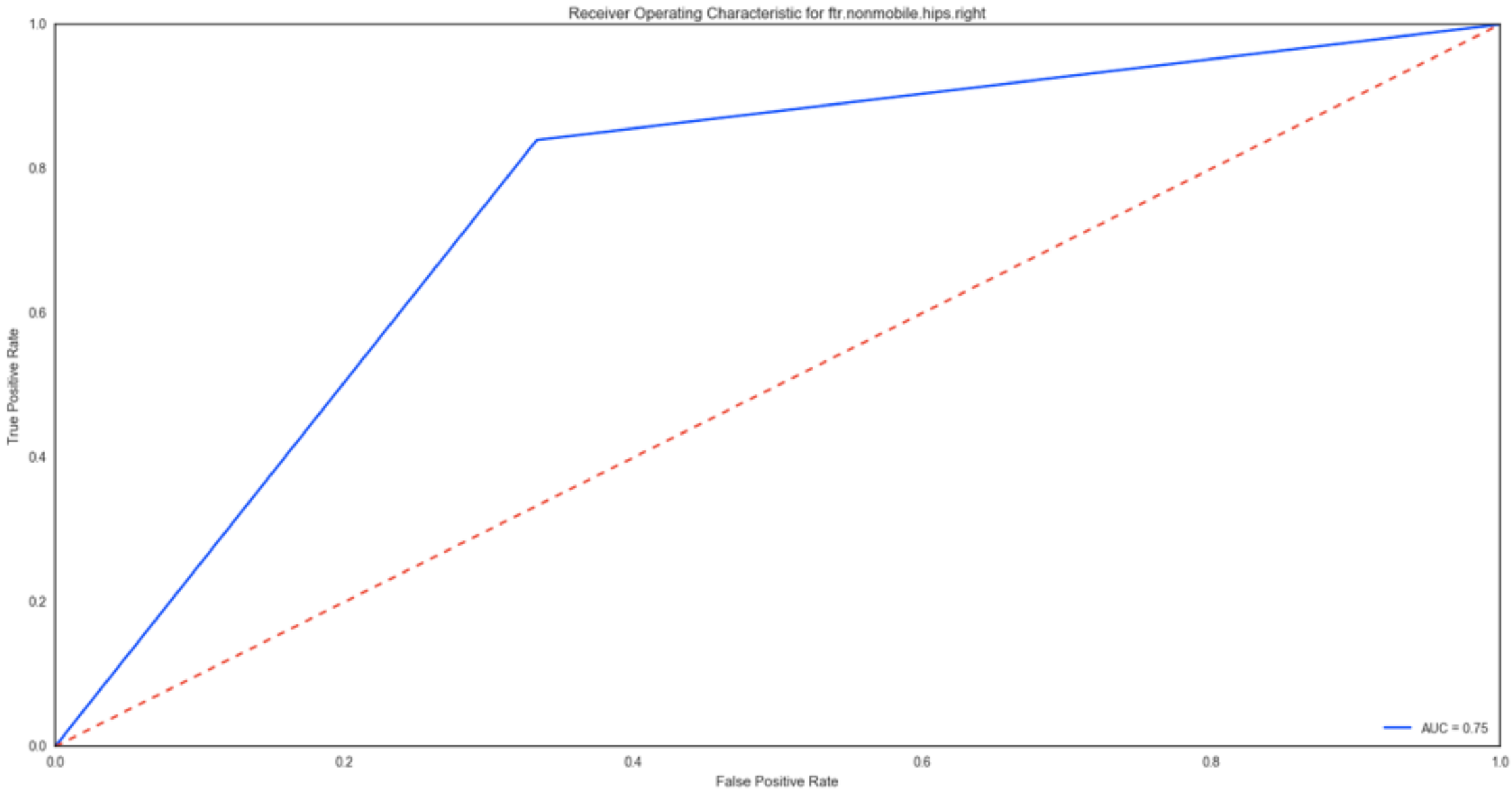
Understanding How all features work together

Going through a decision tree

- Classification algorithm tend to make the rationale behind a classification as a black box, although it can be easy to understand, a user tend to require tangible cutoffs in order to make decision quickly
- One of the best options to visualize these cutoffs is to use an algorithm that makes it easy to ‘see’ how a decision is being made
- We can sacrifice model accuracy in order to understand how a decision whether a mouse will live or die is being made
- We will use a decision tree for that (as simple one)
- Let’s measure how much accuracy we will lose

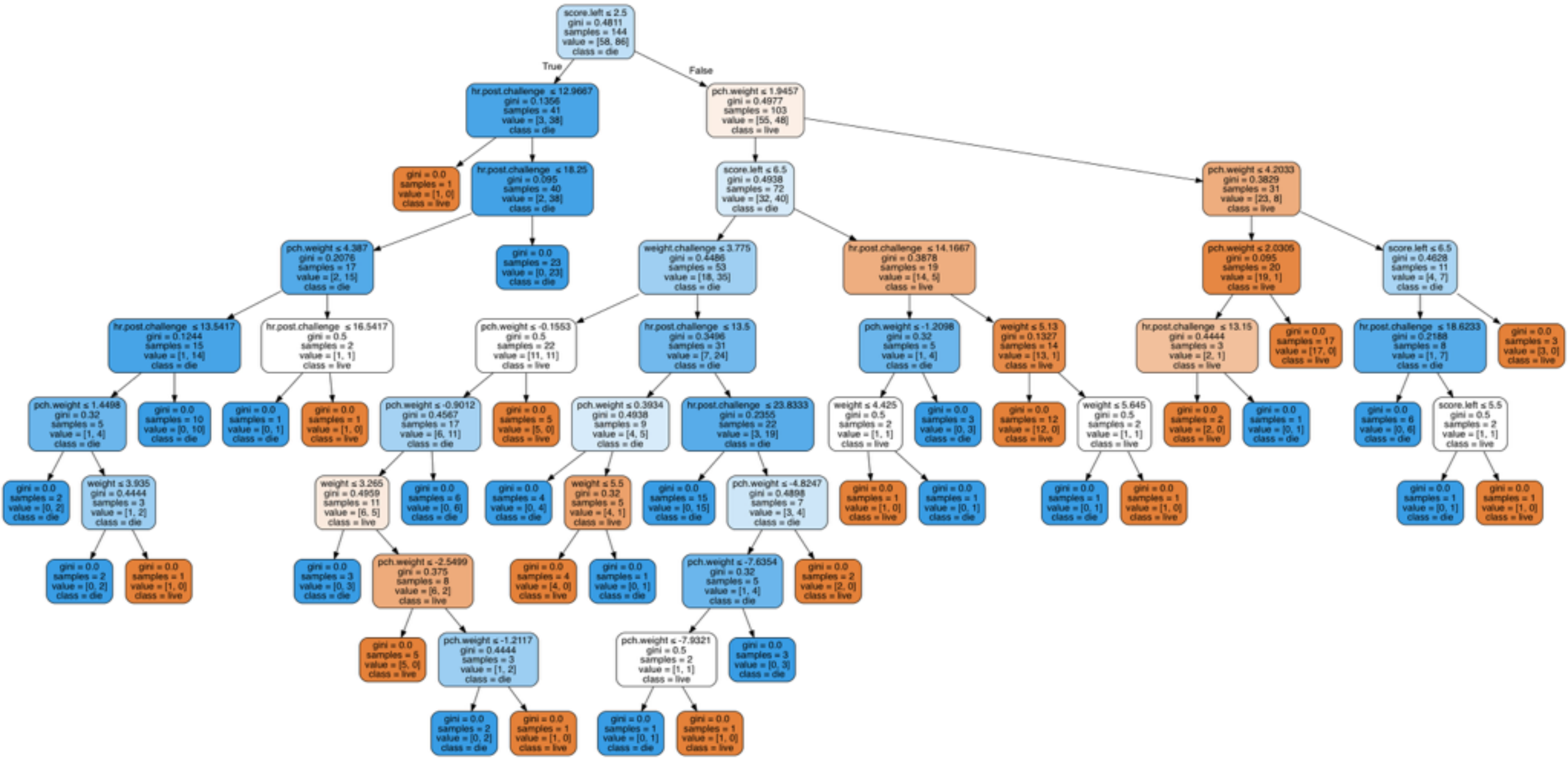
Going through a decision tree

Model Accuracy : 0.783783783784
ROC : 0.75



Going through a decision tree

Let's visualize how the decision is being made



Going through a decision tree

How to read the tree

- Imagine that all data (all rows) start in a single bin at the top of the tree.
- All features are considered to see how the data can be split in the most informative way– this uses the gini measure by default
- At the top we see the most informative condition is score.left <= 2.5. If this condition is true, take the left branch to get to the 41 samples of value = [3. 38]. This means there are 38 examples of class/target 1 (die), and 3 examples of class/target 0 (live). The other 103 samples, of the 144 total, go to the right bin.
- This splitting continues until the split creates a bin with only one class