National College of Ireland

The college for a learning society

Knowledge Discovery in Healthcare databases: feature selection in diabetes classification

User Configuration Manual

Svetlana Nicolenco
X01419950

Submitted as part of the requirements for the degree of
MSc in Data Analytics at the School of Computing,

National College of Ireland,
Dublin, Ireland.

August 2016

Supervisor: Dr. Eugene O'Loughlin

Contents

1. Introduction 3
2. Application Environment 3
2.1. Hardware 3
2.2. Software 3
3. Project development 4
3.1. Data extraction 4
3.2. Data report: 4
3.3. Data validation parameters: 5
3.4. Modelling with ‘fscaret’: 5
3.5. Modelling with caret: gbm, glmnet, svm, nnet 5
Model tuning ‘gbm’ case study 2 6
Model tuning ‘Svm case study 2 6
Model tuning ‘Glmnet’ case study 2 6
Model tuning ‘Nnet’ case study 2 6
3.6. Modelling with caret: gbm, glmnet, svm, nnet: Case study 3 6
Model tuning ‘gbm’ case study 3 7
Model tuning ‘Svm case study 3 7
Model tuning ‘Glmnet’ case study 3 7
Model tuning ‘Nnet’ case study 3 7
3.7. Modelling with caret: gbm, glmnet, svm, nnet: Case study 4 7
Model tuning ‘gbm’ case study 4 7
Model tuning ‘Svm case study 4 7
Model tuning ‘Glmnet’ case study 4 7
Model tuning ‘Nnet’ case study 4 8
4. Exploratory Data Analysis (EDA) 8
4.1. EDA Master Dataset: 8
4.1. Classification tree laboratory subset: 8
5. Future Work 8

1. Introduction

This configuration hand book facilitates the understanding of technical details involved in the process flow of a feature selection for improving classification accuracy in medical domain, as well as learn the internal specifics of the various artefacts developed in this project. Different aspects of data extraction and manipulation, configuration parameters, development environment and packages utilisation, are presented in more detail. For specifics and concepts related to feature extraction please refer to the project report.

2. Application Environment

The output of this project were developed and executed in the following environment.

2.1. Hardware

Dell Latitude E5250, 16GB RAM, Intel(R) Core(TM) i5-5300U CPU @ 2.30GHz, LITEONIT LCS-256L9S-11 22 SCSI Disk Device, Intel(R) HD Graphics 5500 has been used as a main development environment.

2.2. Software

- Initial pre-processing of data: merging, cleaning, identification of outliers, variable analysis was conducted in IBM SPSS Statistics V23.0 software. The initial progress of data acquisition, processing data and changes in naming convention was logged in the Codebook of the same program.
- R Studio version 3.3.1, update 2016-06-21, platform x86_64-w64-mingw32/x64 (64-bit), the Foundation for Statistical Computing was used for some Exploratory Data Analysis and construction of predictive models (R Core Team, 2012).
- Packages below have been utilised for feature selection:

  gbm (Ridgeway et al. 2013),

  glmnet (Friedman, Hastie, Tibshirani, 2010),

  caret (Kuhn et al., 2012),

  fscaret (Szlek, 2013),

  pROC (Robin et al., 2011),

  kernlab (Karatzoglou et al, 2004),

  rpart 3.1-39 (Therneau and Atkinson 1997),

  nnet 7.2-42 (Venables and Ripley 1999),

  party 0.9-96 (Hothorn et al. 2006),

  rattle (Williams, G., 2011)

  rpart.plot (Therneau, Atkinson and Ripley, 2014)

  RColorBrewer (Harrower and Brewer, 2013)

  Partykit (Hothorn and Zeleis, 2015)

3. Project development

3.1. Data extraction

Data sources:

Centers for Disease Control and Prevention (CDC), 2016, National Center for Health Statistics (NCHS). National Health and Nutrition Examination Survey Questionnaire (or Examination Protocol, or Laboratory Protocol). Hyattsville, MD: U.S. Department of Health and Human Services, Centers for Disease Control and Prevention

Laboratory data: Standard Biochemistry Profile

http://wwwn.cdc.gov/Nchs/Nhanes/Search/DataPage.aspx?Component=Laboratory&CycleBeginYear=2011

Demographic data:

http://wwwn.cdc.gov/Nchs/Nhanes/Search/DataPage.aspx?Component=Demographics&CycleBeginYear=2011

Dietary data: Dietary Interview - Total Nutrient Intakes, First Day

http://wwwn.cdc.gov/Nchs/Nhanes/Search/DataPage.aspx?Component=Dietary&CycleBeginYear=2011

The National Health and Nutrition Examination Surveys (NHANES) data year 2011-2012 was used in this study. Multi-stage complex survey of non-institutionalised population of United States conducted as a continuous cycle and released on two year basis. It is a representative stratified probability sample and includes Demographic, Examination, Laboratory, and Questionnaire components with extensive well documented data files. The main objectives of the NHANES survey are to investigate the connection between diet habits, nutritional value and public health matters in general and specific groups, to investigate emerging public health matters and new sophisticated technologies. Materials were produced by Federal agencies of the United States, the data designed for the exploration of the multiple predictors and complex relationship is open to the public, as all data is de-identified and may be reproduced without permission (National Centre for Health Statistics, 2013).

Process flow:

Demography, Laboratory, Dietary data files was downloaded from the NHANES website as SAS transport files (.XPT). Files have been processed in SPSS using (.sav) format, then merged by sequence number (SEQN), as it is the unique identifier. Merged file consists of 250 variables and 9756 individual cases. The pooled dataset contained total interviewed population and those who had completed a medical examination in the Mobile Exam Centre.

Only respondents that participated in all three parts of the survey have been considered for inclusion in the final dataset, which consists of 5510 cases. Variables of interest have been selected from the full dataset on the basis discussed in “dataset narrative” of the report paper and consist of 73 variables.

3.2. Data report:

Initial data collection report (refer Appendix-A1).

Changing in the naming convention (refer Appendix-A2)

Missing value analysis (MVA) can be found in the (refer Appendix-A3)

Descriptive statistics report can be found in the (refer Appendix-A4)

3.3. Data validation parameters:

Source code (refer Appendix-B1)

Master Dataset consists of 73 variables in total, 71 predictor, sequence number and binary outcome (0/1). Repeated ten-fold cross–validation was run 3 times to determine the mean generalization accuracy of each learning method on test data. Distribution of the outcome in classification tasks has been preserved with a function 'createDataPartition' available in 'caret' package. Dataset has been split in ratio 70/30, 3826 cases in train dataset and 1638 in the test dataset. To control the randomness and to warrant that the same resamples are used between calls function 'set.seed' has been used. This will assure reproducible results in all used models.

3.4. Modelling with 'fscaret':

Source code (refer Appendix-B1)

Output: feature importance with 'fscaret' (refer Appendix-C1)

Tuning settings give the option to specify the models to test from 270 available in this package and indicate a limitations. Data pre-processing step in "fscaret" included inspection of near zero variance predictors (the percentage of unique values is less than 20), test for multi-collinearity (variables with correlation of more than 0.9 removed). Setting preprocessData has been altered as TRUE. Time limit convention can be adjusted, as default value is 24 hour period. The function myTimeLimit is fixed in seconds for building a single model, for this experiment has been tuned to 60. For two-class classification task function 'funcClassPred' has been used and limited to models: "glm", "gbm", "treebag", "ridge", "lasso", "rpart", "svmRadial". Results are presented in the next section.

3.5. Modelling with caret: gbm, glmnet, svm, nnet

Four algorithms has been chosen for prediction and accuracy comparison between Master Dataset, laboratory subset and Pima Indian diabetes dataset. Methods of experiment are 'gbm', glmnet, svmRadial and pcaNNet, with built in feature selection (Kuhn, 2012). Tuning parameters for models has been set in 'trainControl' function available in 'caret' package. Variables standardised with the 'preProc' function, and were centred and scaled, prior to fitting the model sequence. The coefficients are always returned on the original scale. ROC was used to select the optimal model using the largest value. The grid of tuning parameters is controlled with the "tuneLength" function in caret package. Complexity parameter within range of values (c) 0.1 – 1000 is selected by tuneLength = 5 (Kuhn; 2008).

Case study 2

Source code (refer Appendix-B1)

Output: feature importance with ‘caret’ (refer Appendix-C2)

Output: ROC curves predictions (refer Appendix-C4)

Model tuning ‘gbm’ case study 2

Stochastic Gradient Boosting default boosting parameters are: number of trees is 100, interaction depth is 1 and shrinkage is 0.001. Custom tuning parameters have been specified by the use of the expand.grid function. Experimental grid for the ‘caret’ train process has been set to: max tree depth (interaction.depth = c(1, 5, 9)), boosting iterations (n.trees = (1:30)*50), shrinkage = c(.001,.01,.1), minimal terminal node size (n.minobsinnode = 20).

Modelling parameters of ‘gbm’ method, in particular shrinkage of 0.0001, considerably increased computational complexity. Given the size of the master dataset, tuning this parameters for gbm increased computational time and took more than 20 hours and considered not practical for the research. The final values used for the model were: n.trees = 500, interaction.depth = 5, shrinkage = 0.01, and n.minobsinnode = 20. With this settings on a train dataset ROC =0.85, sensitivity =0.33, specificity =0.98.

Model tuning ‘Svm’ case study 2

Default setting of the train() function of Support Vector Machines with Radial Basis Function Kernel derived estimates for sigma=0.01 and c=0.25. ROC was used to select the optimal model using the largest value 0.87, sensitivity 0.45, and specificity 0.95. The final values used for the model were sigma = 0.01 and C = 0.25. Number of Support Vectors : 1316. There were 9 predictors of which 9 had non-zero influence.

Additional sensitivity analysis around this two values has been done with expand.grid() of ‘caret’s train(). A data frame of values has been built with all the combination of the parameter settings. Tuning the model parameters with the ‘tuneGrid’ function to sigma values .01, .005, 0.001, and cost 0.25, 0.15, 0.1. The final values used for the model were sigma = 0.001 and C = 0.1, slight improvement with customised parameters: ROC 0.88, sensitivity 0.49 and specificity 0.95. Number of Support Vectors used 1368 in the final model.

Model tuning ‘Glmnet’ case study 2

Tuning parameter 'alpha' was held constant at a value of 1. ROC was used to select the optimal model using the largest value. The final values used for the model were alpha = 1 and lambda = 0.01. ROC 0.88, sensitivity 0.44, specificity 0.96. Area under the curve: 0.8863

Model tuning ‘Nnet’ case study 2

Neural Network Model with PCA Pre-Processing Created from 3826 samples and 71 variables. PCA needed 55 components to capture 99 percent of the variance. Tuning parameter 'size' was held constant at a value of 10, Tuning parameter 'decay' was held constant at a value of 0.1. Master dataset: ROC 0.80, Sensitivity 0.45, Specificity 0.92.

3.6. Modelling with caret: gbm, glmnet, svm, nnet: Case study 3

Source code (refer Appendix-B2)

Output: ROC curves predictions (refer Appendix-C7)

Output: Parallel plot ROC importance with ‘caret’ (refer Appendix-C4)

Model tuning ‘gbm’ case study 3

However for smaller subset of laboratory data, including this parameter in the model increased sensitivity of the results. ROC was used to select the optimal model using the largest value. The final values used for the model were n.trees = 150, interaction.depth = 4, shrinkage = 0.1 and n.minobsinnode = 20. ROC 0.86, sensitivity 0.42, specificity 0.97. Slight improvement in accuracy even with different set of features. As final, a gradient boosted model with Bernoulli loss function, performed 150 iterations.

Model tuning ‘Svm case study 3

Tuning parameter 'sigma' was held constant at a value of 0.09484751. The final values used for the model were sigma = 0.09 and C = 0.25. ROC was used to select the optimal model using the largest value. ROC 0.87, sensitivity 0.44, specificity 0.96. Tuning the parameters expand.grid(sigma = c(.05, 0.01, 0.005), C = c(0.10, 0.05, 0.20, 0.25, 0.15)). ROC was used to select the optimal model using the largest value. The final values used for the model were sigma = 0.01 and C = 0.25. ROC 0.88, sensitivity 0.5, specificity 0.95, slight improvement on ROC and considerable improvement on sensitivity.

Model tuning ‘Glmnet’ case study 3

The final values used for the model were alpha = 1 and lambda = 0.0001. ROC was used to select the optimal model using the largest value. ROC 0.89, sensitivity 0.43, specificity 0.97.

Model tuning ‘Nnet’ case study 3

Lab subset with only 9 predictors has a considerable improvement in accuracy, ROC 0.88, Sensitivity 0.48, Specificity 0.95, even though different features have been selected for the model. PCA needed 9 features to capture 99 percent of the variance.

3.7. Modelling with caret: gbm, glmnet, svm, nnet: Case study 4

Source code (refer Appendix-B3)

Output: Parallel plot ROC importance with ‘caret’ (refer Appendix-C5)

Output: ROC curves predictions (refer Appendix-C8)

Model tuning ‘gbm’ case study 4

ROC was used to select the optimal model using the largest value. The final values used for the model were n.trees = 400, interaction.depth = 3, shrinkage = 0.01 and n.minobsinnode = 20. A final model uses a gradient boosted model with Bernoulli loss function. 400 iterations were performed. There were 8 predictors of which 8 had non-zero influence.

Model tuning ‘Svm’ case study 4

ROC was used to select the optimal model using the largest value. The final values used for the model were sigma = 0.05 and C = 0.2. Number of Support Vectors: 348.

Model tuning ‘Glmnet’ case study 4

Tuning parameter 'alpha' was held constant at a value of 1. ROC was used to select the optimal model using the largest value. The final values used for the model were alpha = 1 and lambda = 0.01.

Model tuning 'Nnet' case study 4

Neural Networks with Feature Extraction, Resampling: Cross-Validated (10 fold, repeated 3 times). Tuning parameter 'size' was held constant at a value of 10. Tuning parameter 'decay' was held constant at a value of 0.1. Neural Network Model with PCA Pre-Processing Created from 538 samples and 8 variables. PCA needed 8 features to capture 99 percent of the variance a 8-10-2 network with 112 weights options were - decay=0.1

4. Exploratory Data Analysis (EDA)

4.1. EDA Master Dataset:

Source code (refer Appendix-B4)

4.1. Classification tree laboratory subset:

Source code (refer Appendix-B5)

Output: Plot (refer Appendix-C8)

5. Future Work

In addition to the code updates that would be applicable for implementation of future work proposed in the thesis report, the following code related tasks are recommended and can be taken up as an enhancement activity;

- Improve specificity of the prediction model by tuning more parameter settings.
- Utilisation of H2O environment for large datasets to improve the computational speed.
- Applying this process flow in a Spark in-memory processing environment.

Appendix –A (SPSS outputs)

A-1. Data collection report

```
GET
  FILE='C:\Users\s\Documents\SvetaMSDataAnalytics\thesis\ThesisData\LabFoodDemo2011.sav'.
DATASET NAME DataSet1 WINDOW=FRONT.

SAVE OUTFILE='C:\Users\s\Documents\SvetaMSDataAnalytics\thesis\ThesisData\masterData2011EDA.sav'
  /COMPRESSED.
CODEBOOK  SEQN [s] RIAGENDR [n] RIDAGEYR [s] RIDRETH1 [n] LBDSCASI [s] LBXSC3SI [s]
LBDSGLSI [s]
    LBDSIRSI [s] LBDSPHSI [s] LBDSTPSI [s] LBXSNASI [s] LBXSKSI [s] LBXSCLSI [s] LBXSOSSI [s]
DR1TPROT
    [s] DR1TSUGR [s] DR1TFIBE [s] DR1TATOC [s] DR1TATOA [s] DR1TRET [s] DR1TVARA [s]
DR1TACAR [s]
    DR1TBCAR [s] DR1TCRYP [s] DR1TLYCO [s] DR1TLZ [s] DR1TVB1 [s] DR1TVB2 [s] DR1TNIAC [s]
DR1TVB6 [s]
    DR1TFOLA [s] DR1TFA [s] DR1TFF [s] DR1TFDFE [s] DR1TCHL [s] DR1TVB12 [s] DR1TB12A [s]
DR1TVC [s]
    DR1TVD [s] DR1TVK [s] DR1TCALC [s] DR1TPHOS [s] DR1TMAGN [s] DR1TIRON [s] DR1TZINC [s]
DR1TCOPP [s]
    DR1TSODI [s] DR1TPOTA [s] DR1TSELE [s] DR1TCAFF [s] DR1TTHEO [s] DR1TALCO [s] DR1TMOIS
[s] DR1TS040
    [s] DR1TS060 [s] DR1TS080 [s] DR1TS100 [s] DR1TS120 [s] DR1TS140 [s] DR1TS160 [s] DR1TS180 [s]
    DR1TM161 [s] DR1TM181 [s] DR1TM201 [s] DR1TM221 [s] DR1TP182 [s] DR1TP183 [s] DR1TP184 [s]
DR1TP204
    [s] DR1TP205 [s] DR1TP225 [s] DR1TP226 [s] DR1_320Z [s]
  /VARINFO POSITION LABEL TYPE FORMAT MEASURE ROLE VALUELABELS MISSING
ATTRIBUTES
  /OPTIONS VARORDER=VARLIST SORT=ASCENDING MAXCATS=200
  /STATISTICS COUNT PERCENT MEAN STDDEV QUARTILES.
```

A-2. Recoding Variables Report

```
OUTFILE='C:\Users\s\Documents\SvetaMSDataAnalytics\thesis\ThesisData\masterData2011EDA.sav'
  /COMPRESSED.
RECODE LBDSCASI LBXSC3SI LBDSGLSI LBDSIRSI LBDSPHSI LBDSTPSI LBXSNASI LBXSKSI
LBXSCLSI LBXSOSSI
    DR1TPROT DR1TSUGR DR1TFIBE DR1TATOC DR1TATOA DR1TRET DR1TVARA DR1TACAR
DR1TBCAR DR1TCRYP DR1TLYCO
    DR1TLZ DR1TVB1 DR1TVB2 DR1TNIAC DR1TVB6 DR1TFOLA DR1TFA DR1TFF DR1TFDFE
DR1TCHL DR1TVB12 DR1TB12A
    DR1TVC DR1TVD DR1TVK DR1TCALC DR1TPHOS DR1TMAGN DR1TIRON DR1TZINC DR1TCOPP
DR1TSODI DR1TPOTA
    DR1TSELE DR1TCAFF DR1TTHEO DR1TALCO DR1TMOIS DR1TS040 DR1TS060 DR1TS080
DR1TS100 DR1TS120 DR1TS140
    DR1TS160 DR1TS180 DR1TM161 DR1TM181 DR1TM201 DR1TM221 DR1TP182 DR1TP183 DR1TP184
DR1TP204 DR1TP205
    DR1TP225 DR1TP226 DR1_320Z (ELSE=Copy) INTO Bcalcium Bbicarbonate Bglucose Biron Bphosphorus
    Bprotein Bsodium Bpotassium Bchloride Bosmolality Fprotein Fsugar Ffiber FvitE Falphatocopherol
    Fretinol FvitA Facarotene Fbcarotene Fbcryptoxan Flycopene Flutein FvitB1 FvitB2 Fniacin FvitB6
    Ftotfolate Ffolicacid Ffolate FfolateDFE Fcholine FvitB12 FaddvitB12 FvitC FvitD FvitK Fcalcium
    Fphosphorus Fmagnesium Firon Fzinc Fcopper Fsodium Fpotassium Fselenium Fcaffeine Ftheobromine
    Falcohol Fmoisture Fsfa40 Fsfa60 Fsfa80 Fsfa100 Fsfa120 Fsfa140 Fsfa160 Fsfa180 Fmfa161 Fmfa181
    Fmfa201 Fmfa221 Fpfa182 Fpfa183 Fpfa184 Fpfa204 Fpfa205 Fpfa225 Fpfa226 Totwater.
VARIABLE LABELS  Bcalcium 'Total calcium (mmol/L)' /Bbicarbonate 'Bicarbonate (mmol/L)'
    /Bglucose 'Glucose, serum (mmol/L)' /Biron 'Iron, refigerated (umol/L)' /Bphosphorus 'Phosphorus '+
    '(mmol/L)' /Bprotein 'Total protein (g/L)' /Bsodium 'Sodium (mmol/L)' /Bpotassium 'Potassium '+
    '(mmol/L)' /Bchloride 'Chloride (mmol/L)' /Bosmolality 'Osmolality (mmol/Kg)' /Fprotein
```

```
    'Protein (gm)' /Fsugar 'Total sugars (gm)' /Ffiber 'Dietary fiber (gm)' /FvitE 'Vitamin E as '+
    'alpha-tocopherol (mg)' /Falphatocopherol 'Added alpha-tocopherol (Vitamin E) (mg)' /Fretinol
    'Retinol (mcg)' /FvitA 'Vitamin A, RAE (mcg)' /Facarotene 'Alpha-carotene (mcg)' /Fbcarotene
    'Beta-carotene (mcg)' /Fbcryptoxan 'Beta-cryptoxanthin (mcg)' /Flycopene 'Lycopene (mcg)'
    /Flutein 'Lutein + zeaxanthin (mcg)' /FvitB1 'Thiamin (Vitamin B1) (mg)' /FvitB2 'Riboflavin '+
    '(Vitamin B2) (mg)' /Fniacin 'Niacin (mg)' /FvitB6 'Vitamin B6 (mg)' /Ftotfolate 'Total folate '+
    '(mcg)' /Ffolicacid 'Folic acid (mcg)' /Ffolate 'Food folate (mcg)' /FfolateDFE 'Folate, DFE '+
    '(mcg)' /Fcholine 'Total choline (mg)' /FvitB12 'Vitamin B12 (mcg)' /FaddvitB12 'Added vitamin '+
    'B12 (mcg)' /FvitC 'Vitamin C (mg)' /FvitD 'Vitamin D (D2 + D3) (mcg)' /FvitK 'Vitamin K (mcg)'
    /Fcalcium 'Calcium (mg)' /Fphosphorus 'Phosphorus (mg)' /Fmagnesium 'Magnesium (mg)' /Firon
    'Iron (mg)' /Fzinc 'Zinc (mg)' /Fcopper 'Copper (mg)' /Fsodium 'Sodium (mg)' /Fpotassium
    'Potassium (mg)' /Fselenium 'Selenium (mcg)' /Fcaffeine 'Caffeine (mg)' /Ftheobromine
    'Theobromine (mg)' /Falcohol 'Alcohol (gm)' /Fmoisture 'Moisture (gm)' /Fsfa40 'SFA 4:0 '+
    '(Butanoic) (gm)' /Fsfa60 'SFA 6:0 (Hexanoic) (gm)' /Fsfa80 'SFA 8:0 (Octanoic) (gm)' /Fsfa100
    'SFA 10:0 (Decanoic) (gm)' /Fsfa120 'SFA 12:0 (Dodecanoic) (gm)' /Fsfa140 'SFA 14:0 '+
    '(Tetradecanoic) (gm)' /Fsfa160 'SFA 16:0 (Hexadecanoic) (gm)' /Fsfa180 'SFA 18:0 '+
    '(Octadecanoic) (gm)' /Fmfa161 'MFA 16:1 (Hexadecenoic) (gm)' /Fmfa181 'MFA 18:1 (Octadecenoic) '+
    '(gm)' /Fmfa201 'MFA 20:1 (Eicosenoic) (gm)' /Fmfa221 'MFA 22:1 (Docosenoic) (gm)' /Fpfa182
    'PFA 18:2 (Octadecadienoic) (gm)' /Fpfa183 'PFA 18:3 (Octadecatrienoic) (gm)' /Fpfa184 'PFA 18:4 '+
    '(Octadecatetraenoic) (gm)' /Fpfa204 'PFA 20:4 (Eicosatetraenoic) (gm)' /Fpfa205 'PFA 20:5 '+
    '(Eicosapentaenoic) (gm)' /Fpfa225 'PFA 22:5 (Docosapentaenoic) (gm)' /Fpfa226 'PFA 22:6 '+
    '(Docosahexaenoic) (gm)' /Totwater 'Total plain water drank yesterday (gm)'.
EXECUTE.
DATASET ACTIVATE DataSet1.

SAVE OUTFILE='C:\Users\s\Documents\SvetaMSDataAnalytics\thesis\ThesisData\masterData2011EDA.sav'
  /COMPRESSED.

SAVE OUTFILE='C:\Users\s\Documents\SvetaMSDataAnalytics\thesis\ThesisData\masterData2011renamed.sav'
  /COMPRESSED.
RECODE RIAGENDR RIDAGEYR RIDRETH1 (ELSE=Copy) INTO Gender Age Race.
VARIABLE LABELS  Gender 'Gender' /Age 'Age in years at screening' /Race 'Race/Hispanic origin'.
EXECUTE.
DATASET ACTIVATE DataSet1.

SAVE OUTFILE='C:\Users\s\Documents\SvetaMSDataAnalytics\thesis\ThesisData\masterData2011renamed.sav'
  /COMPRESSED.
CODEBOOK  SEQN [s] Gender [n] Age [s] Race [n] Bcalcium [s] Bbicarbonate [s] Bglucose [s] Biron [s]
    Bphosphorus [s] Bprotein [s] Bsodium [s] Bpotassium [s] Bchloride [s] Bosmolality [s] Fprotein [s]
    Fsugar [s] Ffiber [s] FvitE [s] Falphatocopherol [s] Fretinol [s] FvitA [s] Facarotene [s]
    Fbcarotene [s] Fbcryptoxan [s] Flycopene [s] Flutein [s] FvitB1 [s] FvitB2 [s] Fniacin [s] FvitB6
    [s] Ftotfolate [s] Ffolicacid [s] Ffolate [s] FfolateDFE [s] Fcholine [s] FvitB12 [s] FaddvitB12
    [s] FvitC [s] FvitD [s] FvitK [s] Fcalcium [s] Fphosphorus [s] Fmagnesium [s] Firon [s] Fzinc [s]
    Fcopper [s] Fsodium [s] Fpotassium [s] Fselenium [s] Fcaffeine [s] Ftheobromine [s] Falcohol [s]
    Fmoisture [s] Fsfa40 [s] Fsfa60 [s] Fsfa80 [s] Fsfa100 [s] Fsfa120 [s] Fsfa140 [s] Fsfa160 [s]
    Fsfa180 [s] Fmfa161 [s] Fmfa181 [s] Fmfa201 [s] Fmfa221 [s] Fpfa182 [s] Fpfa183 [s] Fpfa184 [s]
    Fpfa204 [s] Fpfa205 [s] Fpfa225 [s] Fpfa226 [s] Totwater [s]
  /VARINFO POSITION LABEL TYPE FORMAT MEASURE ROLE VALUELABELS MISSING
ATTRIBUTES
  /OPTIONS VARORDER=VARLIST SORT=ASCENDING MAXCATS=200
  /STATISTICS COUNT PERCENT MEAN STDDEV QUARTILES.
```

A-3. Missing Value Analysis report

Univariate Statistics

	N	Mean	Std. Deviation	Missing		No. of Extremes[a]	
				Count	Percent	Low	High
SEQN	5510	67058.33	2818.680	0	.0	0	0
RIAGENDR	5510	1.50	.500	0	.0	0	0
RIDAGEYR	5510	42.22	20.527	0	.0	0	0
RIDRETH1	5510	3.26	1.180	0	.0	0	0
LBDSCASI	5510	2.3555	.08761	0	.0	127	123
LBXSC3SI	5510	25.01	2.182	0	.0	128	92
LBDSGLSI	5510	5.5387	2.01863	0	.0	0	213
LBDSIRSI	5505	15.1151	6.45839	5	.1	18	200
LBDSPHSI	5510	1.2390	.20352	0	.0	70	196
LBDSTPSI	5503	71.92	4.745	7	.1	108	143
LBXSNASI	5510	138.96	2.161	0	.0	148	67
LBXSKSI	5509	3.9416	.34043	1	.0	88	151
LBXSCLSI	5510	103.91	2.778	0	.0	179	70
LBXSOSSI	5510	277.45	4.815	0	.0	109	141
DR1TPROT	5510	81.2776	41.36444	0	.0	0	249
DR1TSUGR	5510	118.2115	78.99173	0	.0	0	222
DR1TFIBE	5510	17.0004	10.71187	0	.0	0	237
DR1TATOC	5510	8.4348	6.81571	0	.0	0	190
DR1TATOA	5510	.5683	2.99134	0	.0	0	169
DR1TRET	5510	408.76	557.311	0	.0	0	76
DR1TVARA	5510	604.60	709.643	0	.0	0	112
DR1TACAR	5510	375.00	1326.693	0	.0	0	119
DR1TBCAR	5510	2123.76	4716.272	0	.0	0	163
DR1TCRYP	5510	84.91	240.291	0	.0	0	107
DR1TLYCO	5510	5006.44	9319.543	0	.0	0	250
DR1TLZ	5510	1543.78	3683.412	0	.0	0	160
DR1TVB1	5510	1.5967	.89488	0	.0	0	222
DR1TVB2	5510	2.0099	1.23702	0	.0	0	186
DR1TNIAC	5510	25.3008	14.91374	0	.0	0	222
DR1TVB6	5510	2.0746	1.50470	0	.0	0	202
DR1TFOLA	5510	409.18	259.528	0	.0	0	226
DR1TFA	5510	192.73	189.153	0	.0	0	208
DR1TFF	5510	216.52	146.156	0	.0	0	207
DR1TFDFE	5510	544.29	377.030	0	.0	0	217
DR1TCHL	5510	323.5097	190.38719	0	.0	0	247
DR1TVB12	5510	5.0867	6.35720	0	.0	0	116
DR1TB12A	5510	.9521	2.43121	0	.0	0	240
DR1TVC	5510	84.9161	99.58847	0	.0	0	212
DR1TVD	5510	4.7232	5.66884	0	.0	0	184

DR1TVK	5510	115.2109	276.06070	0	.0	0	95
DR1TCALC	5510	950.47	591.258	0	.0	0	237
DR1TPHOS	5510	1364.09	672.169	0	.0	1	232
DR1TMAGN	5510	293.96	151.371	0	.0	0	219
DR1TIRON	5510	15.1356	8.99539	0	.0	0	225
DR1TZINC	5510	11.0468	6.62060	0	.0	0	238
DR1TCOPP	5510	1.2595	1.06238	0	.0	0	91
DR1TSODI	5510	3544.34	1830.885	0	.0	0	227
DR1TPOTA	5510	2617.81	1266.684	0	.0	4	217
DR1TSELE	5510	113.6756	62.94041	0	.0	0	221
DR1TCAFF	5510	125.50	179.801	0	.0	0	213
DR1TTHEO	5510	36.25	76.403	0	.0	0	219
DR1TALCO	5510	8.5376	27.49930	0	.0	0	212
DR1TMOIS	5510	2824.7076	1528.57177	0	.0	0	222
DR1TS040	5510	.5076	.53204	0	.0	0	251
DR1TS060	5510	.2869	.30243	0	.0	0	240
DR1TS080	5510	.2373	.27371	0	.0	0	188
DR1TS100	5510	.4475	.44080	0	.0	0	226
DR1TS120	5510	.7593	1.39537	0	.0	0	127
DR1TS140	5510	2.1080	1.85017	0	.0	0	242
DR1TS160	5510	13.9631	8.56926	0	.0	0	225
DR1TS180	5510	6.3473	4.08811	0	.0	0	233
DR1TM161	5510	1.1116	.85371	0	.0	0	238
DR1TM181	5510	26.6146	16.43556	0	.0	0	240
DR1TM201	5510	.2995	.28087	0	.0	0	195
DR1TM221	5510	.0289	.11884	0	.0	0	58
DR1TP182	5510	16.8419	11.44122	0	.0	0	237
DR1TP183	5510	1.7417	1.30626	0	.0	0	237
DR1TP184	5510	.01232	.035309	0	.0	0	244
DR1TP204	5510	.1490	.13372	0	.0	0	266
DR1TP205	5510	.0325	.10335	0	.0	0	175
DR1TP225	5510	.0239	.03861	0	.0	0	123
DR1TP226	5510	.0643	.17797	0	.0	0	170
DR1_320Z	5510	1078.0789	1178.20738	0	.0	0	276

a. Number of cases outside the range (Mean - 2*SD, Mean + 2*SD).

A-4. Descriptive Statistics Report

Descriptive Statistics

	N	Minimum	Maximum	Mean	Std. Deviation	Skewness		Kurtosis	
	Statistic	Statistic	Statistic	Statistic	Statistic	Statistic	Std. Error	Statistic	Std. Error
Glucose, serum (mmol/L)	5464	1.72	34.25	5.5365	2.01348	5.007	.033	38.547	.066
Gender	5464	1.00	2.00	1.4976	.50004	.010	.033	-2.001	.066
Age in years at screening	5464	12.00	80.00	42.1713	20.52488	.224	.033	-1.141	.066
Race/Hispanic origin	5464	1.00	5.00	3.2549	1.18040	-.337	.033	-.548	.066
Total calcium (mmol/L)	5464	2.00	2.83	2.3557	.08697	.199	.033	.691	.066
Bicarbonate (mmol/L)	5464	16.00	37.00	25.0117	2.17743	-.103	.033	.608	.066
Iron, refigerated (umol/L)	5464	.90	54.10	15.0876	6.37894	.893	.033	1.846	.066
Phosphorus (mmol/L)	5464	.52	2.13	1.2391	.20308	.366	.033	.487	.066
Total protein (g/L)	5464	53.00	95.00	71.9272	4.74321	.173	.033	.821	.066
Sodium (mmol/L)	5464	127.00	153.00	138.9607	2.14974	-.533	.033	2.449	.066
Potassium (mmol/L)	5464	2.80	5.60	3.9409	.33597	.510	.033	1.498	.066
Chloride (mmol/L)	5464	90.00	115.00	103.9173	2.77018	-.533	.033	1.345	.066
Osmolality (mmol/Kg)	5464	252.00	304.00	277.4531	4.77006	.215	.033	1.953	.066
Protein (gm)	5464	.00	356.26	80.9822	40.91908	1.256	.033	2.813	.066
Total sugars (gm)	5464	.00	851.43	117.6369	76.54733	1.735	.033	5.907	.066
Dietary fiber (gm)	5464	.00	93.20	16.9290	10.55043	1.582	.033	4.268	.066
Vitamin E as alpha-tocopherol (mg)	5464	.00	90.46	8.3331	6.37033	3.115	.033	19.265	.066
Added alpha-tocopherol (Vitamin E) (mg)	5464	.00	51.87	.5526	2.93690	8.594	.033	94.821	.066
Retinol (mcg)	5464	.00	8025.00	400.3327	408.90797	5.862	.033	78.182	.066

Vitamin A, RAE (mcg)	5464	.00	8890.00	589.7967	527.57479	3.717	.033	32.878	.066
Alpha-carotene (mcg)	5464	.00	15376.00	357.9526	912.67427	5.951	.033	57.259	.066
Beta-carotene (mcg)	5464	.00	40096.00	2056.8190	3598.14256	3.876	.033	20.674	.066
Beta-cryptoxanthin (mcg)	5464	.00	4875.00	82.5421	197.50652	12.698	.033	266.062	.066
Lycopene (mcg)	5464	.00	95206.00	4974.7354	9009.61235	3.766	.033	19.688	.066
Lutein + zeaxanthin (mcg)	5464	.00	61965.00	1503.1018	3162.50036	6.853	.033	69.912	.066
Thiamin (Vitamin B1) (mg)	5464	.00	11.91	1.5910	.88351	1.857	.033	8.474	.066
Riboflavin (Vitamin B2) (mg)	5464	.00	16.76	1.9973	1.18642	2.405	.033	14.604	.066
Niacin (mg)	5464	.00	137.65	25.1756	14.69311	1.914	.033	6.753	.066
Vitamin B6 (mg)	5464	.00	17.63	2.0619	1.48194	3.317	.033	20.267	.066
Total folate (mcg)	5464	.00	2316.00	406.4299	252.56679	1.771	.033	5.488	.066
Folic acid (mcg)	5464	.00	2149.00	191.8281	186.09716	2.878	.033	13.982	.066
Food folate (mcg)	5464	.00	1272.00	214.6667	138.47502	1.793	.033	5.661	.066
Folate, DFE (mcg)	5464	.00	3818.00	540.8995	369.12776	2.176	.033	8.364	.066
Total choline (mg)	5464	.00	1602.40	321.6209	187.32543	1.493	.033	3.706	.066
Vitamin B12 (mcg)	5464	.00	84.91	4.9817	4.80241	4.978	.033	55.200	.066
Added vitamin B12 (mcg)	5464	.00	26.34	.9459	2.40391	4.469	.033	27.689	.066
Vitamin C (mg)	5464	.00	1247.70	83.8907	93.83193	2.736	.033	13.747	.066
Vitamin D (D2 + D3) (mcg)	5464	.00	69.30	4.6404	5.11068	3.288	.033	20.737	.066
Vitamin K (mcg)	5464	.00	2201.70	111.0808	162.70590	5.161	.033	38.328	.066
Calcium (mg)	5464	12.00	5075.00	948.1168	586.02691	1.567	.033	4.428	.066
Phosphorus (mg)	5464	.00	6453.00	1357.9680	660.28120	1.218	.033	2.912	.066
Magnesium (mg)	5464	7.00	1402.00	292.1536	147.79552	1.557	.033	4.694	.066
Iron (mg)	5464	.00	82.44	15.0528	8.86324	1.911	.033	6.759	.066

Zinc (mg)	5464	.00	78.42	11.0009	6.56544	1.966	.033	8.102	.066
Copper (mg)	5464	.03	14.84	1.2395	.78565	4.966	.033	59.446	.066
Sodium (mg)	5464	7.00	17070.00	3536.6493	1813.60051	1.509	.033	4.342	.066
Potassium (mg)	5464	.00	9868.00	2602.5066	1225.02322	1.057	.033	2.016	.066
Selenium (mcg)	5464	.00	659.10	113.0544	61.04984	1.458	.033	4.346	.066
Caffeine (mg)	5464	.00	2410.00	124.6056	173.18296	3.335	.033	20.022	.066
Theobromine (mg)	5464	.00	937.00	36.0584	74.93512	4.397	.033	28.443	.066
Alcohol (gm)	5464	.00	374.40	8.3849	26.45226	5.689	.033	45.317	.066
Moisture (gm)	5464	197.98	15505.85	2813.9184	1513.42551	1.909	.033	7.081	.066
SFA 4:0 (Butanoic) (gm)	5464	.00	5.05	.5074	.52672	2.171	.033	7.534	.066
SFA 6:0 (Hexanoic) (gm)	5464	.00	2.94	.2864	.29867	2.276	.033	8.757	.066
SFA 8:0 (Octanoic) (gm)	5464	.00	2.32	.2341	.23802	2.514	.033	10.988	.066
SFA 10:0 (Decanoic) (gm)	5464	.00	4.90	.4447	.42126	2.174	.033	8.506	.066
SFA 12:0 (Dodecanoic) (gm)	5464	.00	18.58	.7367	1.06518	5.869	.033	60.570	.066
SFA 14:0 (Tetradecanoic) (gm)	5464	.00	14.42	2.0976	1.79088	1.832	.033	5.055	.066
SFA 16:0 (Hexadecanoic) (gm)	5464	.00	72.35	13.9278	8.47380	1.406	.033	3.159	.066
SFA 18:0 (Octadecanoic) (gm)	5464	.00	33.43	6.3320	4.04610	1.402	.033	3.285	.066
MFA 16:1 (Hexadecenoic) (gm)	5464	.00	10.37	1.1087	.84822	2.119	.033	8.988	.066
MFA 18:1 (Octadecenoic) (gm)	5464	.00	127.17	26.5101	16.05938	1.347	.033	2.860	.066
MFA 20:1 (Eicosenoic) (gm)	5464	.00	3.08	.2960	.26832	3.191	.033	17.574	.066
MFA 22:1 (Docosenoic) (gm)	5464	.00	2.35	.0263	.08181	14.085	.033	282.393	.066

PFA 18:2 (Octadecadienoic) (gm)	5464	.00	93.80	16.7442	11.05481	1.528	.033	3.748	.066
PFA 18:3 (Octadecatrienoic) (gm)	5464	.00	11.25	1.7308	1.26144	2.030	.033	6.924	.066
PFA 18:4 (Octadecatetraenoic) (gm)	5464	.00	.49	.0120	.03400	5.527	.033	44.041	.066
PFA 20:4 (Eicosatetraenoic) (gm)	5464	.00	1.28	.1477	.13028	1.793	.033	5.163	.066
PFA 20:5 (Eicosapentaenoic) (gm)	5464	.00	1.76	.0308	.09428	7.705	.033	87.960	.066
PFA 22:5 (Docosapentaenoic) (gm)	5464	.00	.60	.0231	.03197	6.444	.033	73.219	.066
PFA 22:6 (Docosahexaenoic) (gm)	5464	.00	2.75	.0614	.15750	6.955	.033	71.253	.066
Total plain water drank yesterday (gm)	5464	.00	12886.88	1076.3177	1178.28898	2.362	.033	10.254	.066
Valid N (listwise)	5464								

Appendix –B

(Source code used in this project)

B-1 R Code - Case study 1 and Case study 2

```
### Svetlana Nicolenco
### 01419950
### MS Data Analytics
### School of Computing
### National College pf Ireland
### August 2016

### variable importance
### caret package
### models GBM, SVM, GLMNET and Neural Network


rm(list=ls(all=TRUE))
#install.packages("fscaret", dependencies = c("Depends", "Suggests"))
library(fscaret)
library(caret)
library(pROC)
library(glmnet)
library(kernlab) ### SVM

setwd("C:/Users/s/Documents/R")
a = read.csv("masterData2011Glgr.csv", sep = ",", header = TRUE,
stringsAsFactors = TRUE)
str(a)
summary(a)
table(is.na(a))

### glucose data in the dataset was binned in 2 groups "norm" "high" glucose
### class to factor
a$Glgr = factor(a$Glgr)
levels(a$Glgr)
###create copy of the dataset and seqn N removed
a1 = a[,-1]
#transform data no "norm" and "high" for caret
a1$Glgr <- ifelse(a1$Glgr ==0 ,'norm','high')
a1$Glgr <- as.factor(a1$Glgr)
levels(a1$Glgr)
table(a1$Glgr)
prop.table(table(a1$Glgr))

### create train and test datasets ratio 70/30
set.seed (1001)
split <- createDataPartition(a1$Glgr, p=.70, list=FALSE)
trainDF <- a1[ split,]
testDF <- a1[-split,]
dim(trainDF)
dim(testDF)

### packages for feature selection in fscaret
names(getModelInfo())
# limit models to use in ensemble and run fscaret
data(funcClassPred)
funcClassPred
getModelInfo()$glm$type
getModelInfo()$gbm$type
```

```
fsModels <- c("glm", "gbm", "treebag", "ridge", "lasso", "rpart",
"svmRadial")
set.seed (007)
myFS<-fscaret(trainDF, testDF, myTimeLimit = 40, preprocessData=T,
              Used.funcRegPred = fsModels, with.labels=TRUE,
              supress.output=FALSE, no.cores=2)

names(myFS)
# analyze results
print(myFS$VarImp)
print(myFS$PPlabels)
results <- myFS$VarImp$matrixVarImp.MSE
results$Input_no <- as.numeric(results$Input_no)
results <- results[c("SUM","SUM%","ImpGrad","Input_no")]
myFS$PPlabels$Input_no <-  as.numeric(rownames(myFS$PPlabels))
results <- merge(x=results, y=myFS$PPlabels, by="Input_no", all.x=T)
results <- results[c('Labels', 'SUM')]
results <- subset(results,results$SUM !=0)
results <- results[order(-results$SUM),]
print(results)
results1 = head(results, 10)
#results written to temorary directory
tempdir()

### nice plot of top ten important feaatures.
p <- ggplot(results1, aes(x=Labels, y=SUM)) +
  geom_bar(stat="identity", fill="#53cfff") +
  coord_flip() +
  theme_light(base_size=16) +
  xlab("Predictors") +
  ylab("") +
  ggtitle("Feature Importance 'fscaret'") +
  theme(plot.title=element_text(size=12))
p


### models GBM, SVM, GLMNET, rpart, c5.0, Nnet
target = 'Glgr'
predictors = names(trainDF)[names(trainDF) != target]

### train control function parameter tuning
set.seed(7)
TC = trainControl(method = "repeatedcv",
                  number = 10, repeats = 3,
                  classProbs = TRUE,
                  summaryFunction = twoClassSummary,
                  returnResamp='final')

# train the GBM model
# with this size of the dataset and shrinkage parameter 0.0001,
computationally too expensive
#gbmGrid =  expand.grid(interaction.depth = c(2:5),
                       #n.trees=seq(1,501,10),
                       #shrinkage = c(0.0001, .001, .01, .1),
                       #n.minobsinnode = 20)

gbmGrid =  expand.grid(interaction.depth = c(2:5),
                       n.trees = (1:10)*50,
                       shrinkage = c(.001,.01,.1),
                       n.minobsinnode = 20)
set.seed(7)
gbmMod = train(trainDF[,predictors], trainDF[,target],
               method='gbm', trControl=TC, tuneLength = 5,
               metric = "ROC", tuneGrid = gbmGrid,
```

```
                verbose = FALSE, preProc = c("center", "scale"))


# train the SVM model
SVMgrid <- expand.grid(sigma = c(.01, .005, 0.001),
                       C = c(0.25, 0.15, 0.1))

set.seed(7)
svmMod = train(trainDF[,predictors], trainDF[,target],
               method = "svmRadial", tuneLength = 5,
               preProc = c("center", "scale"),
               metric = "ROC",
               tuneGrid = SVMgrid,
               trControl = TC)
svmMod

# train the glmnet model - alpha=0 ridge regression, alpha= 1 lasso, 0 <
alpha < 1 elastic net!
lassoGrid = expand.grid(.alpha=1,.lambda=c(0.1, 0.01, 0.001, 0.0001))
set.seed(7)
glmnetMod = train(trainDF[,predictors],
                  trainDF[,target],
                  method='glmnet',
                  preProc = c("center", "scale"),
                  metric = "ROC",
                  tuneLength = 5,
                  tuneGrid = lassoGrid,
                  trControl=TC)
glmnetMod

# train the Neural network model with feature extraction 'pcaNNet'
NNgrid <- expand.grid(size=c(10), decay=c(0.1))
set.seed(7)
NNMod = train(trainDF[,predictors],
               trainDF[,target],
               method = 'pcaNNet',
               preProc = c("center", "scale"),
               metric = "ROC",
               tuneLength = 5,
               trControl=TC,
               tuneGrid=NNgrid)
NNMod

# collect resamples
results <- resamples(list(GBM=gbmMod, SVM=svmMod, GLMNET=glmnetMod,
NNet=NNMod))

# summarize the distributions
summary(results)
# boxplots of results
bwplot(results)

###
splom(results, metric = "ROC")
xyplot(results, metric = "ROC")
parallelplot(results, metric = "ROC")
dotplot(results, metric = "ROC")
rocDiffs <- diff(results, metric = "ROC")
summary(rocDiffs)

summary(gbmMod)
summary(svmMod)
summary(glmnetMod)
summary(NNMod)
```

```
gbmMod
svmMod
glmnetMod
NNMod


plot(gbmMod)
plot(svmMod)
plot(glmnetMod)
plot(NNMod)

gbmMod$finalModel
svmMod$finalModel
glmnetMod$finalModel
NNMod$finalModel

### VARIABLE IMPORTANCE FOR MODELS

gbmImp <- varImp(gbmMod, scale = FALSE)
gbmImp
plot(gbmImp, top = 10)

svmImp <- varImp(svmMod, scale = FALSE)
svmImp
plot(svmImp, top = 10)

glmnetImp <- varImp(glmnetMod, scale = FALSE)
glmnetImp
plot(glmnetImp, top = 10)

NNImp <- varImp(NNMod, scale = FALSE)
NNImp
plot(NNImp, top = 10)

predictors(glmnetMod)
predictors(gbmMod)
predictors(NNMod)
predictors(svmMod)

################################################
# evalutate model - Estimating Performance For Classification
################################################
### type 'Raw' gives a class prediction, in our case 'high' and 'norm'
# class prediction
GBMpredictions <- predict(object=gbmMod, testDF[,predictors], type='raw')
SVMpredictions <- predict(object=svmMod, testDF[,predictors], type='raw')
GLMNETpredictions <- predict(object=glmnetMod, testDF[,predictors],
type='raw')
NNpredictions <- predict(object=NNMod, testDF[,predictors], type='raw')

### probabilities
head(GBMpredictions, 3)
head(SVMpredictions, 3)
head(GLMNETpredictions, 3)
head(NNpredictions, 3)

confusionMatrix(GBMpredictions, testDF$Glgr)
confusionMatrix(SVMpredictions, testDF$Glgr)
confusionMatrix(GLMNETpredictions, testDF$Glgr)
confusionMatrix(NNpredictions, testDF$Glgr)
```

```
### start by looking at class predictions and using the caret 'postResample'
function to get an accuracy score
### print(postResample(pred=predictions,
obs=as.factor(testDF[,outcomeName])))
postResample(pred=GBMpredictions, obs=as.factor(testDF[,target]))
postResample(pred=SVMpredictions, obs=as.factor(testDF[,target]))
postResample(pred=GLMNETpredictions, obs=as.factor(testDF[,target]))
postResample(pred=NNpredictions, obs=as.factor(testDF[,target]))

# get predictions on your testing data
###  prob gives you the probability on how sure the model is about it's
choice
GBMpredictions1 = predict(object=gbmMod, testDF[,predictors], type='prob')
SVMpredictions1 = predict(object=svmMod, testDF[,predictors], type='prob')
GLMNETpredictions1 = predict(object=glmnetMod, testDF[,predictors],
type='prob')
NNpredictions1 = predict(object=NNMod, testDF[,predictors], type='prob')

### probabilities
head(GBMpredictions1, 3)
head(SVMpredictions1, 3)
head(GLMNETpredictions1, 3)
head(NNpredictions1, 3)

### to be in control of the threshold and also like to use AUC score which
requires probabilities, not classes.
gbmAUC = roc(ifelse(testDF[,target]=="norm",0,1), GBMpredictions1[[2]])
plot(gbmAUC)
plot(gbmAUC, type = "S", print.thres = .5)
print(gbmAUC$auc)

svmAUC = roc(ifelse(testDF[,target]=="norm",0,1), SVMpredictions1[[2]])
plot(svmAUC)
plot(svmAUC, type = "S", print.thres = .5)
print(svmAUC$auc)


glmnetAUC = roc(ifelse(testDF[,target]=="norm",0,1), GLMNETpredictions1[[2]])
plot(glmnetAUC)
plot(glmnetAUC, type = "S", print.thres = .5)
print(glmnetAUC$auc)

nnetAUC = roc(ifelse(testDF[,target]=="norm",0,1), NNpredictions1[[2]])
plot(nnetAUC)
plot(nnetAUC, type = "S", print.thres = .5)
print(nnetAUC$auc)

Sys.info()
traceback()
sessionInfo()

citation("caret")
citation("fscaret")
citation("pROC")
citation("svmRadial")
citation("kernlab")
```

B-2 R Code - Case study 3

```
### Svetlana Nicolenco
### 01419950
### MS Data Analytics
### School of Computing
### National College pf Ireland
### August 2016

### variable importance
### caret package
### models GBM, SVM, GLMNET and Neural Network

rm(list=ls(all=TRUE))
library(caret)
library(pROC)
library(glmnet)
library(kernlab) ### SVM

setwd("C:/Users/s/Documents/R")
a = read.csv("masterData2011Glgr.csv", sep = ",", header = TRUE,
stringsAsFactors = TRUE)
str(a)
summary(a)
table(is.na(a))

### glucose data in the dataset was binned in 2 groups "norm" "high" glucose
### class to factor
a$Glgr = factor(a$Glgr)
levels(a$Glgr)
###create copy of the dataset and seqn N removed
## subset lab data only
myvars <- c("Bcalcium", "Bbicarbonate", "Biron", "Bphosphorus", "Bprotein",
"Bsodium", "Bpotassium", "Bchloride", "Bosmolality", "Glgr")
a1 <- a[myvars]
summary(a1)

#transform data no "norm" and "high" for caret
a1$Glgr <- ifelse(a1$Glgr ==0 ,'norm','high')
a1$Glgr <- as.factor(a1$Glgr)
levels(a1$Glgr)
table(a1$Glgr)
prop.table(table(a1$Glgr))

### create train and test datasets ratio 70/30
set.seed (1001)
split <- createDataPartition(a1$Glgr, p=.70, list=FALSE)
trainDF <- a1[ split,]
testDF <- a1[-split,]
dim(trainDF)
dim(testDF)

### models GBM, SVM, GLMNET, rpart, c5.0, Nnet
target = 'Glgr'
predictors = names(trainDF)[names(trainDF) != target]

set.seed(7)
TC = trainControl(method = "repeatedcv",
                  number = 10, repeats = 3,
                  classProbs = TRUE,
                  summaryFunction = twoClassSummary,
                  returnResamp='final')
```

```
# train the GBM model
gbmGrid =  expand.grid(interaction.depth = c(2:5),
                       n.trees = (1:10)*50,
                       shrinkage = c(0.0001, .001, .01, .1),
                       n.minobsinnode = 20)
set.seed(7)
gbmMod = train(trainDF[,predictors], trainDF[,target],
               method='gbm',
               trControl=TC,
               metric = "ROC",
               tuneGrid = gbmGrid,
               verbose = FALSE,
               tuneLength = 5,
               preProc = c("center", "scale"))
gbmMod

# train the SVM model
SVMgrid <- expand.grid(sigma = c(.05, 0.01, 0.005),
                       C = c(0.10, 0.05, 0.20, 0.25, 0.15))
set.seed(7)
svmMod = train(trainDF[,predictors], trainDF[,target],
               method = "svmRadial",
               tuneLength = 5,
               preProc = c("center", "scale"),
               metric = "ROC",
               tuneGrid = SVMgrid,
               trControl = TC)
svmMod

# train the glmnet model - ??= 0 ridge regression, . ??= 1 lasso . 0 < ?? < 1
elastic net!
lassoGrid = expand.grid(.alpha=1,.lambda=c(0.1, 0.01, 0.001, 0.0001))
set.seed(7)
glmnetMod = train(trainDF[,predictors],
                  trainDF[,target],
                  method='glmnet',
                  preProc = c("center", "scale"),
                  metric = "ROC",
                  tuneLength = 5,
                  tuneGrid = lassoGrid,
                  trControl=TC)
glmnetMod

# train the Neural network model with feature extraction 'pcaNNet'

NNgrid <- expand.grid(size=c(10), decay=c(0.1))
set.seed(7)
NNMod = train(trainDF[,predictors],
              trainDF[,target],
              method = 'pcaNNet',
              preProc = c("center", "scale"),
              metric = "ROC",
              tuneLength = 5,
              trControl=TC,
              tuneGrid=NNgrid)
NNMod

# collect resamples
results <- resamples(list(GBM=gbmMod, SVM=svmMod, GLMNET=glmnetMod,
NNet=NNMod))

# summarize the distributions
summary(results)
# boxplots of results
```

```
bwplot(results)
splom(results, metric = "ROC")
xyplot(results, metric = "ROC")
parallelplot(results, metric = "ROC")
dotplot(results, metric = "ROC")
rocDiffs <- diff(results, metric = "ROC")
summary(rocDiffs)

summary(gbmMod)
summary(svmMod)
summary(glmnetMod)
summary(NNMod)

gbmMod
svmMod
glmnetMod
NNMod

plot(gbmMod)
plot(svmMod)
plot(glmnetMod)
plot(NNMod)

gbmMod$finalModel
svmMod$finalModel
glmnetMod$finalModel
NNMod$finalModel

### VARIABLE IMPORTANCE FOR MODELS
gbmImp <- varImp(gbmMod, scale = FALSE)
gbmImp
plot(gbmImp)

svmImp <- varImp(svmMod, scale = FALSE)
svmImp
plot(svmImp)

glmnetImp <- varImp(glmnetMod, scale = FALSE)
glmnetImp
plot(glmnetImp)

NNImp <- varImp(NNMod, scale = FALSE)
NNImp
plot(NNImp)

################################################
# evalutate model - Estimating Performance For Classification
################################################
### type 'Raw' gives a class prediction, in our case 'high' and 'norm'
# class prediction
GBMpredictions <- predict(object=gbmMod, testDF[,predictors], type='raw')
SVMpredictions <- predict(object=svmMod, testDF[,predictors], type='raw')
GLMNETpredictions <- predict(object=glmnetMod, testDF[,predictors],
type='raw')
NNpredictions <- predict(object=NNMod, testDF[,predictors], type='raw')

### probabilities
head(GBMpredictions, 3)
head(SVMpredictions, 3)
head(GLMNETpredictions, 3)
head(NNpredictions, 3)

confusionMatrix(GBMpredictions, testDF$Glgr)
confusionMatrix(SVMpredictions, testDF$Glgr)
```

```
confusionMatrix(GLMNETpredictions, testDF$Glgr)
confusionMatrix(NNpredictions, testDF$Glgr)

### start by looking at class predictions and using the caret 'postResample'
function to get an accuracy score
### print(postResample(pred=predictions,
obs=as.factor(testDF[,outcomeName])))
postResample(pred=GBMpredictions, obs=as.factor(testDF[,target]))
postResample(pred=SVMpredictions, obs=as.factor(testDF[,target]))
postResample(pred=GLMNETpredictions, obs=as.factor(testDF[,target]))
postResample(pred=NNpredictions, obs=as.factor(testDF[,target]))

# get predictions on your testing data
###  prob gives you the probability on how sure the model is about it's
choice
GBMpredictions1 = predict(object=gbmMod, testDF[,predictors], type='prob')
SVMpredictions1 = predict(object=svmMod, testDF[,predictors], type='prob')
GLMNETpredictions1 = predict(object=glmnetMod, testDF[,predictors],
type='prob')
NNpredictions1 = predict(object=NNMod, testDF[,predictors], type='prob')

### probabilities
head(GBMpredictions1, 3)
head(SVMpredictions1, 3)
head(GLMNETpredictions1, 3)
head(NNpredictions1, 3)

### to be in control of the threshold and also like to use AUC score which
requires probabilities, not classes.
gbmAUC = roc(ifelse(testDF[,target]=="norm",0,1), GBMpredictions1[[2]])
plot(gbmAUC)
plot(gbmAUC, type = "S", print.thres = .5)
print(gbmAUC$auc)

svmAUC = roc(ifelse(testDF[,target]=="norm",0,1), SVMpredictions1[[2]])
plot(svmAUC)
plot(svmAUC, type = "S", print.thres = .5)
print(svmAUC$auc)

glmnetAUC = roc(ifelse(testDF[,target]=="norm",0,1), GLMNETpredictions1[[2]])
plot(glmnetAUC)
plot(glmnetAUC, type = "S", print.thres = .5)
print(glmnetAUC$auc)

nnetAUC = roc(ifelse(testDF[,target]=="norm",0,1), NNpredictions1[[2]])
plot(nnetAUC)
plot(nnetAUC, type = "S", print.thres = .5)
print(nnetAUC$auc)

#results written to temorary directory
tempdir()
Sys.info()
traceback()
sessionInfo()

citation("caret")
citation("fscaret")
citation("pROC")
citation("svmRadial")
citation("kernlab")
```

B-3 R Code - Case study 4

```
### Svetlana Nicolenco
### 01419950
### MS Data Analytics
### School of Computing
### National College pf Ireland
### August 2016

### variable importance - Pima dataset
### caret package
### models GBM, SVM, GLMNET and Neural Network

rm(list=ls(all=TRUE))
library(caret)
library(pROC)
library(glmnet)
library(kernlab) ### SVM

setwd("C:/Users/s/Documents/R")
a = read.csv("Pima.txt", sep = ",", header = TRUE, stringsAsFactors = TRUE)
str(a)
summary(a)
str(a)
table(is.na(a))

### glucose data in the dataset was binned in 2 groups "norm" "high" glucose
### class to factor
a$Target = factor(a$Target)
levels(a$Target)
###create copy of the dataset and seqn N removed
## subset lab data only
myvars <- c("Bcalcium", "Bbicarbonate", "Biron", "Bphosphorus", "Bprotein",
"Bsodium", "Bpotassium", "Bchloride", "Bosmolality", "Glgr")
a1 <- a[myvars]
summary(a1)
a1=a
#transform data no "norm" and "high" for caret
a1$Target <- ifelse(a1$Target ==0 ,'norm','high')
a1$Target <- as.factor(a1$Target)
levels(a1$Target)
table(a1$Target)
prop.table(table(a1$Target))

### create train and test datasets ratio 70/30
set.seed (1001)
split <- createDataPartition(a1$Target, p=.70, list=FALSE)
trainDF <- a1[ split,]
testDF <- a1[-split,]
dim(trainDF)
dim(testDF)

### models GBM, SVM, GLMNET, rpart, c5.0, Nnet
target = 'Target'
predictors = names(trainDF)[names(trainDF) != target]

set.seed(7)
TC = trainControl(method = "repeatedcv",
                  number = 10, repeats = 3,
                  classProbs = TRUE,
                  summaryFunction = twoClassSummary,
                  returnResamp='final')
```

```
# train the GBM model
gbmGrid =  expand.grid(interaction.depth = c(2:5),
                       n.trees = (1:10)*50,
                       shrinkage = c(0.0001, .001, .01, .1),
                       n.minobsinnode = 20)
set.seed(7)
gbmMod = train(trainDF[,predictors], trainDF[,target],
               method='gbm',
               trControl=TC,
               metric = "ROC",
               tuneGrid = gbmGrid,
               verbose = FALSE,
               tuneLength = 5,
               preProc = c("center", "scale"))
gbmMod

# train the SVM model
SVMgrid <- expand.grid(sigma = c(.05, 0.01, 0.005),
                       C = c(0.10, 0.05, 0.20, 0.25, 0.15))
set.seed(7)
svmMod = train(trainDF[,predictors], trainDF[,target],
               method = "svmRadial",
               tuneLength = 5,
               preProc = c("center", "scale"),
               metric = "ROC",
               tuneGrid = SVMgrid,
               trControl = TC)
svmMod

# train the glmnet model - ??= 0 ridge regression, . ??= 1 lasso . 0 < ?? < 1
elastic net!
lassoGrid = expand.grid(.alpha=1,.lambda=c(0.1, 0.01, 0.001, 0.0001))
set.seed(7)
glmnetMod = train(trainDF[,predictors],
                  trainDF[,target],
                  method='glmnet',
                  preProc = c("center", "scale"),
                  metric = "ROC",
                  tuneLength = 5,
                  tuneGrid = lassoGrid,
                  trControl=TC)
glmnetMod


# train the Neural network model with feature extraction 'pcaNNet'

NNgrid <- expand.grid(size=c(10), decay=c(0.1))
set.seed(7)
NNMod = train(trainDF[,predictors],
              trainDF[,target],
              method = 'pcaNNet',
              preProc = c("center", "scale"),
              metric = "ROC",
              tuneLength = 5,
              trControl=TC,
              tuneGrid=NNgrid)
NNMod

# collect resamples
results <- resamples(list(GBM=gbmMod, SVM=svmMod, GLMNET=glmnetMod,
NNet=NNMod))

# summarize the distributions
summary(results)
```

```
# boxplots of results
bwplot(results)
splom(results, metric = "ROC")
xyplot(results, metric = "ROC")
parallelplot(results, metric = "ROC")
dotplot(results, metric = "ROC")
rocDiffs <- diff(results, metric = "ROC")
summary(rocDiffs)

summary(gbmMod)
summary(svmMod)
summary(glmnetMod)
summary(NNMod)


gbmMod
svmMod
glmnetMod
NNMod

plot(gbmMod)
plot(svmMod)
plot(glmnetMod)
plot(NNMod)


gbmMod$finalModel
svmMod$finalModel
glmnetMod$finalModel
NNMod$finalModel

### VARIABLE IMPORTANCE FOR MODELS
gbmImp <- varImp(gbmMod, scale = FALSE)
gbmImp
plot(gbmImp)


svmImp <- varImp(svmMod, scale = FALSE)
svmImp
plot(svmImp)

glmnetImp <- varImp(glmnetMod, scale = FALSE)
glmnetImp
plot(glmnetImp)

NNImp <- varImp(NNMod, scale = FALSE)
NNImp
plot(NNImp)

#################################################
# evalutate model - Estimating Performance For Classification
#################################################
### type 'Raw' gives a class prediction, in our case 'high' and 'norm'
# class prediction
GBMpredictions <- predict(object=gbmMod, testDF[,predictors], type='raw')
SVMpredictions <- predict(object=svmMod, testDF[,predictors], type='raw')
GLMNETpredictions <- predict(object=glmnetMod, testDF[,predictors],
type='raw')
NNpredictions <- predict(object=NNMod, testDF[,predictors], type='raw')

### probabilities
head(GBMpredictions, 3)
head(SVMpredictions, 3)
head(GLMNETpredictions, 3)
```

```
head(NNpredictions, 3)

confusionMatrix(GBMpredictions, testDF$Target)
confusionMatrix(SVMpredictions, testDF$Target)
confusionMatrix(GLMNETpredictions, testDF$Target)
confusionMatrix(NNpredictions, testDF$Target)

### start by looking at class predictions and using the caret 'postResample'
function to get an accuracy score
### print(postResample(pred=predictions,
obs=as.factor(testDF[,outcomeName])))
postResample(pred=GBMpredictions, obs=as.factor(testDF[,target]))
postResample(pred=SVMpredictions, obs=as.factor(testDF[,target]))
postResample(pred=GLMNETpredictions, obs=as.factor(testDF[,target]))
postResample(pred=NNpredictions, obs=as.factor(testDF[,target]))

# get predictions on your testing data
###  prob gives you the probability on how sure the model is about it's
choice
GBMpredictions1 = predict(object=gbmMod, testDF[,predictors], type='prob')
SVMpredictions1 = predict(object=svmMod, testDF[,predictors], type='prob')
GLMNETpredictions1 = predict(object=glmnetMod, testDF[,predictors],
type='prob')
NNpredictions1 = predict(object=NNMod, testDF[,predictors], type='prob')

### probabilities
head(GBMpredictions1, 3)
head(SVMpredictions1, 3)
head(GLMNETpredictions1, 3)
head(NNpredictions1, 3)


### to be in control of the threshold and also like to use AUC score which
requires probabilities, not classes.
gbmAUC = roc(ifelse(testDF[,target]=="norm",0,1), GBMpredictions1[[2]])
plot(gbmAUC)
plot(gbmAUC, type = "S", print.thres = .5)
print(gbmAUC$auc)

svmAUC = roc(ifelse(testDF[,target]=="norm",0,1), SVMpredictions1[[2]])
plot(svmAUC)
plot(svmAUC, type = "S", print.thres = .5)
print(svmAUC$auc)


glmnetAUC = roc(ifelse(testDF[,target]=="norm",0,1), GLMNETpredictions1[[2]])
plot(glmnetAUC)
plot(glmnetAUC, type = "S", print.thres = .5)
print(glmnetAUC$auc)

nnetAUC = roc(ifelse(testDF[,target]=="norm",0,1), NNpredictions1[[2]])
plot(nnetAUC)
plot(nnetAUC, type = "S", print.thres = .5)
print(nnetAUC$auc)
```

B-4 R Code - EDA

```
### Svetlana Nicolenco
### 01419950
### MS Data Analytics
### School of Computing
### National College pf Ireland
### August 2016

# EXPLORATORY DATA ANALYSIS

rm(list=ls(all=TRUE))
setwd("C:/Users/s/Documents/R")
a = read.csv("masterData2011EDA.csv", sep = ",", header = TRUE,
stringsAsFactors = TRUE)
str(a)
summary(a)
table(is.na(a))

library(mlbench)
library(caret)

### check our target variable status - column BGlucose
#levels(a$BGlucoseGr)
#a$Glgr = factor(a$Glgr) ### remove hashtags if needed to convert to factor
#levels(a$Glgr)

# Create a copy of dataset
a1 = a[,-1]
scale(a1, center = TRUE, scale = TRUE)

# CORRELATION MATRIX
correlationMatrix <- cor(a1)
# summarize the correlation matrix
print(correlationMatrix)
# find attributes that are highly corrected (ideally >0.75)
highlyCorrelated <- findCorrelation(correlationMatrix, cutoff=0.75)
# print indexes of highly correlated attributes
print(highlyCorrelated)
print(names(a1))

### variable exploration with smoothed conditional mean.
qplot(Bcalcium, Bglucose,     data  =     a1,   geom  =     c("point",
      "smooth"))
qplot(Bpotassium, Bglucose,   data  =     a1,   geom  =     c("point",
      "smooth"))
qplot(Bosmolality, Bglucose,  data  =     a1,   geom  =     c("point",
      "smooth"))
qplot(Bchloride, Bglucose,    data  =     a1,   geom  =     c("point",
      "smooth"))
qplot(Bbicarbonate,     Bglucose,   data  =     a1,   geom  =     c("point",
      "smooth"))
qplot(Biron,      Bglucose,   data  =     a1,   geom  =     c("point",
      "smooth"))
qplot(Bsodium,    Bglucose,   data  =     a1,   geom  =     c("point",
      "smooth"))
qplot(Age,  Bglucose,   data  =     a1,   geom  =     c("point",  "smooth"))
qplot(Totwater,   Bglucose,   data  =     a1,   geom  =     c("point",
      "smooth"))
```

B-5 R Code EDA

Classification tree on laboratory subset

```
### Svetlana Nicolenco
### 01419950
### MS Data Analytics
### School of Computing
### National College pf Ireland
### August 2016
### thesis code - variable importance
### EDA classification tree

rm(list=ls(all=TRUE))
setwd("C:/Users/s/Documents/R")
a = read.csv("masterData2011Glgr.csv", sep = ",", header = TRUE,
stringsAsFactors = TRUE)
str(a)
summary(a)
table(is.na(a))

### glucose data in the dataset was binned in 2 groups "norm" "high" glucose
### class to factor
a$Glgr = factor(a$Glgr)
levels(a$Glgr)

###create copy of the dataset and seqn N removed
## subset lab data only
myvars <- c("Bcalcium", "Bbicarbonate", "Biron", "Bphosphorus", "Bprotein",
"Bsodium", "Bpotassium", "Bchloride", "Bosmolality", "Glgr")
a1 <- a[myvars]
summary(a1)

#transform data no "norm" and "high"
a1$Glgr <- ifelse(a1$Glgr ==0 ,'norm','high')
a1$Glgr <- as.factor(a1$Glgr)
levels(a1$Glgr)
table(a1$Glgr)
prop.table(table(a1$Glgr))

### plot rpart tree - EDA
library(rpart)
library(rattle)                 # Fancy tree plot
library(rpart.plot)             # Enhanced tree plots
library(RColorBrewer)               # Colour selection
library(party)                  # Alternative decision tree algorithm
library(partykit)               # Convert rpart object to Binary


form <- as.formula(Glgr ~ .)
Tree <- rpart(form,data=a1, control=rpart.control(minsplit=20,cp=0))
plot(Tree)
text(Tree)
#
prp(Tree)                    # Will plot the tree
prp(Tree,varlen=3)               # Shorten variable names
# Interatively prune the tree
TreePrun <- prp(Tree,snip=TRUE)$obj # interactively trim the tree
prp(TreePrun) # display the new tree
fancyRpartPlot(Tree)
```

```
Tree2 <- rpart(form,data=a1)
prp(Tree2)
fancyRpartPlot(Tree2)

Tree2 <- rpart(form,data=a1, cp=.02)
prp(Tree2)
fancyRpartPlot(Tree2)

Tree2 <- rpart(form,data=a1, space=0)
prp(Tree2)
fancyRpartPlot(Tree2)

prp(Tree2, main="Glucose level",
    type=4, fallen=T, branch=.3, round=0, leaf.round=9,
    clip.right.labs=F, under.cex=1,
    box.palette="GnYlRd",
    yesno=TRUE,
    prefix="glucose\n", branch.col="gray", branch.lwd=2,
    extra=101, under=T, lt=" < ", ge=" >= ", cex.main=1.5)

prp(Tree2, extra=101, yesno=TRUE, box.palette=c("pink", "palegreen3"))

# Trgrid <- expand.grid(.cp=(1:50)*0.01)
# Trgrid <- expand.grid(.cp=(1:50)*0.001)
Trgrid <- expand.grid(.cp=(1:50)*0.0001)
set.seed(7)
TrMod = train(trainDF[,predictors],
              trainDF[,target],
              method = 'rpart',
              #preProc = c("center", "scale"),
              metric = "ROC",
              trControl=TC,
              tuneGrid=Trgrid,
              tuneLength = 30)
```

Appendix – C (Sample Display Outputs)

C-1. Case study 1

(Output generated with ‘fscaret’ package of feature importance)

	Labels	SUM
12	Bosmolality	187.7137088
2	Age	106.9598529
9	Bsodium	56.7399113
7	Bphosphorus	39.0912783
11	Bchloride	30.8471428
14	Fsugar	20.2366111
10	Bpotassium	13.7787872
47	Fsfa80	6.9432744
46	Fsfa60	6.5319263
48	Fsfa120	6.4808785
24	FvitB2	6.1762426
5	Bbicarbonate	5.7368644
6	Biron	5.6977475
45	Fmoisture	5.5937375
16	FvitE	5.4522539
18	FvitA	4.9361570
55	Fpfa183	4.9142795
54	Fpfa182	4.9090523
37	Fmagnesium	4.9045761
17	Fretinol	4.7830395
57	Fpfa204	4.6523128
30	FvitB12	4.5242668
25	Fniacin	4.5149793
23	FvitB1	4.5036320
13	Fprotein	4.4698009
15	Ffiber	4.4649363
27	Ffolicacid	4.3708304
49	Fsfa180	4.3125086
4	Bcalcium	4.3116783
35	Fcalcium	4.2770292
32	FvitC	4.2647915
44	Fcaffeine	4.2379528
36	Fphosphorus	4.0793158
19	Facarotene	3.8869800
38	Firon	3.7549741
52	Fmfa201	3.7157917
26	FvitB6	3.6777158
41	Fsodium	3.3492697
50	Fmfa161	3.2993790
29	Fcholine	3.2359098
22	Flycopene	3.1482987
61	Totwater	3.1356182
31	FaddvitB12	2.9745212
1	Gender	2.9502211
51	Fmfa181	2.9372049
40	Fcopper	2.8926882
60	Fpfa226	2.4299898
43	Fselenium	2.3361961
28	Ffolate	2.3278241
8	Bprotein	2.2718637
42	Fpotassium	2.2370557
34	FvitK	2.2210861
39	Fzinc	2.0497856
3	Race	1.9900926
20	Fbcarotene	1.9074550
59	Fpfa225	1.8852080
58	Fpfa205	1.7198534
21	Fbcryptoxan	1.7057266
33	FvitD	1.6136786
53	Fmfa221	1.5686621
56	Fpfa184	0.5475267

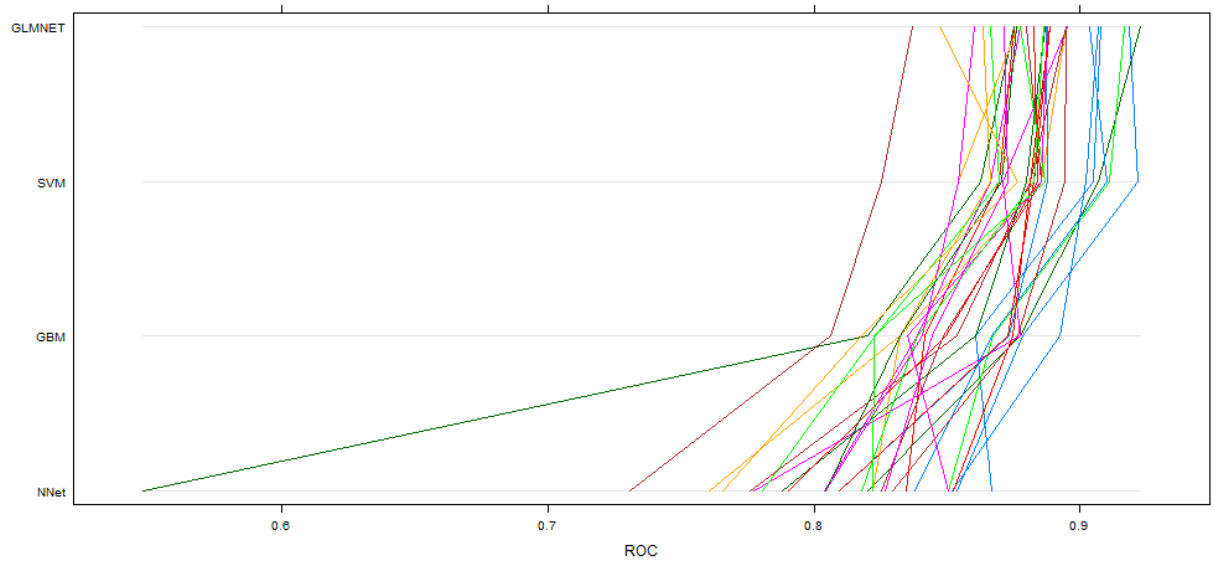
C-2. Case study 2

(Output generated with ‘caret’ package of top 20 feature importance)

gbm		pcaNNet		glmnet		svmRadial	
Bosmolality	2322.33	Bphosphorus	0.6082	Bosmolality	2.021302	Bphosphorus	0.6082
Bsodium	1657.55	Fsugar	0.5785	Bsodium	1.655227	Fsugar	0.5785
Age	816.32	Bchloride	0.5663	Bphosphorus	0.326756	Bchloride	0.5663
Bphosphorus	307.42	Fsfa40	0.5558	Bchloride	0.096315	Fsfa40	0.5558
Bchloride	253.95	Fsfa60	0.5478	Age	0.082612	Fsfa60	0.5478
Fsugar	125.15	Bsodium	0.5472	Bpotassium	0.081869	Bsodium	0.5472
Fretinol	82.68	Biron	0.5467	Biron	0.029907	Biron	0.5467
Bbicarbonate	75.42	Fsfa100	0.5466	FvitB2	0.024340	Fsfa100	0.5466
Bpotassium	74.18	Fcalcium	0.5464	Fsfa100	0.020039	Fcalcium	0.5464
Biron	73.15	Fsfa140	0.5455	Bbicarbonate	0.006776	Fsfa140	0.5455
FvitC	53.61	Fsfa80	0.5446	Fsfa60	0.003753	Fsfa80	0.5446
Fcholine	45.88	Fsfa120	0.5407	Fbcryptoxan	0.000000	Fsfa120	0.5407
FvitD	44.25	Ftheobromine	0.5405	Fmfa201	0.000000	Ftheobromine	0.5405
Fsfa40	41.99	Fsfa160	0.5401	Ffiber	0.000000	Fsfa160	0.5401
Fmoisture	41.77	Fphosphorus	0.5319	Ffolicacid	0.000000	Fphosphorus	0.5319
Fpfa225	41.71	Bcalcium	0.5311	Firon	0.000000	Bcalcium	0.5311
Fpfa204	40.53	Fpfa182	0.5310	Totwater	0.000000	Fpfa182	0.5310
Fbcarotene	40.01	Fretinol	0.5309	FvitE	0.000000	Fretinol	0.5309
FvitE	39.95	Fsfa180	0.5298	FvitA	0.000000	Fsfa180	0.5298
FvitK	39.02	FvitE	0.5296	Fpfa204	0.000000	FvitE	0.5296

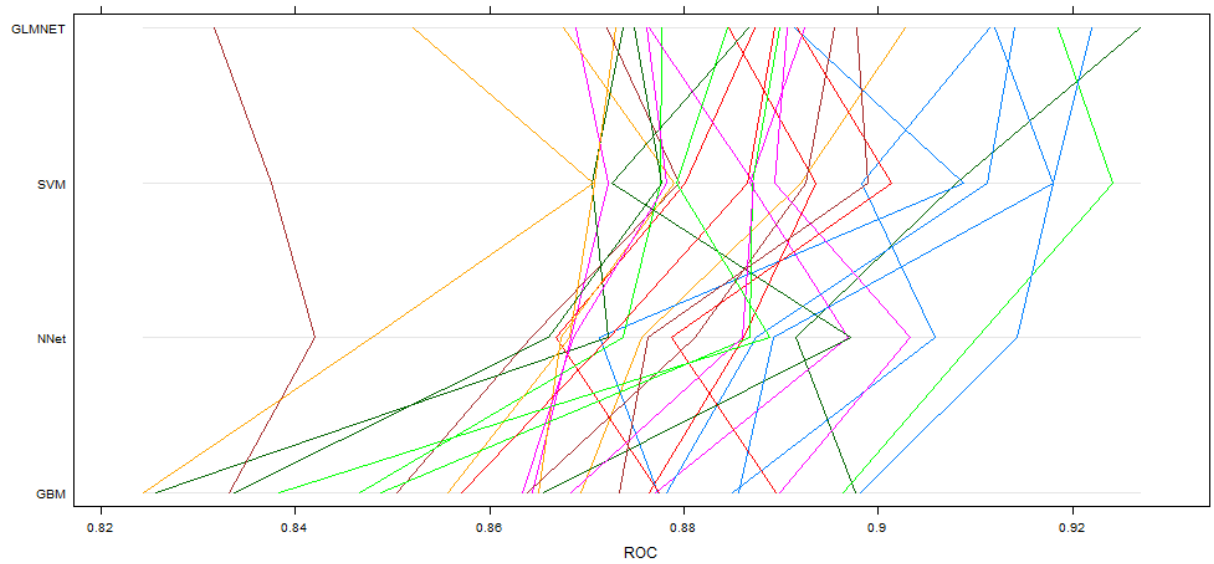
C-3 Case study 2

Output (Parallel plots, ROC curves models ‘gbm’ ‘glmnet’ ‘svmRadial’ PCAnnet’)



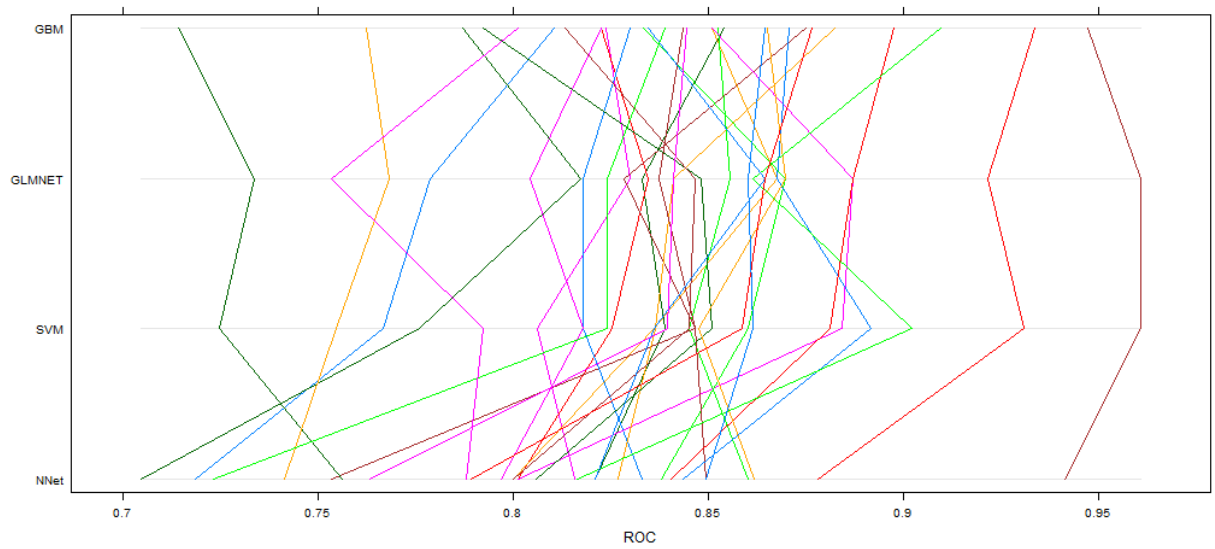
C-4 Case study 3

Output (Parallel plots, ROC curves models ‘gbm’ ‘glmnet’ ‘svmRadial’ PCAnnet’)



C-5 Case study 4

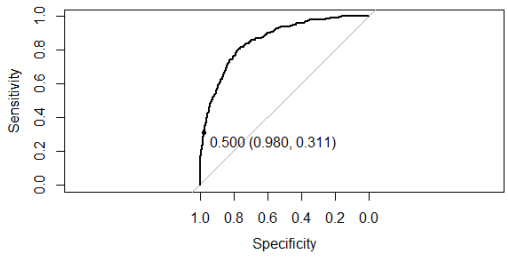
Output (Parallel plots, ROC curves models ‘gbm’ ‘glmnet’ ‘svmRadial’ PCAnnet’)



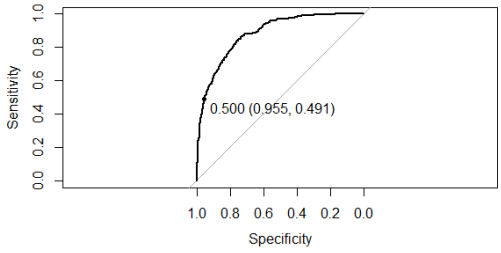
C-6 ROC curves predictions on a test set

Case study 2

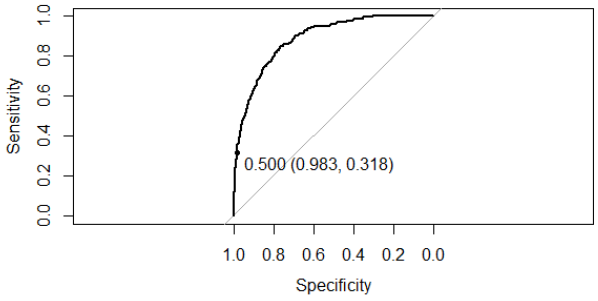
Method ‘gbm’
Area under the curve: 0.86



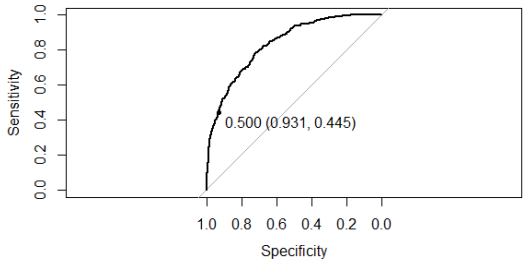
Method ‘svmRadial
Area under the curve: 0.88



Method ‘glmnet’
Area under the curve: 0.89



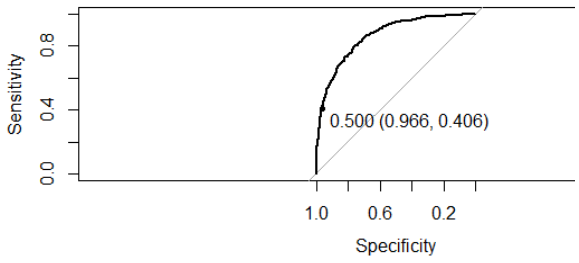
Method ‘nnet’
Area under the curve: 0.84



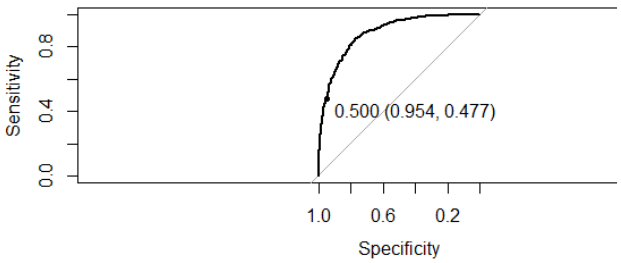
C-7 ROC curves predictions on a test set

Case study 3

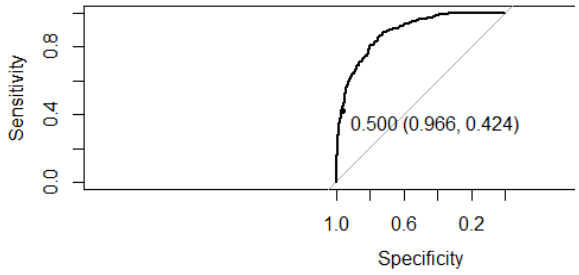
Method ‘gbm’
Area under the curve: 0.86



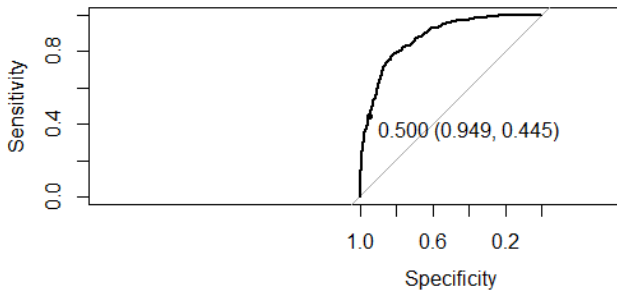
Method ‘svmRadial
Area under the curve: 0.89



Method ‘glmnet’
Area under the curve: 0.89



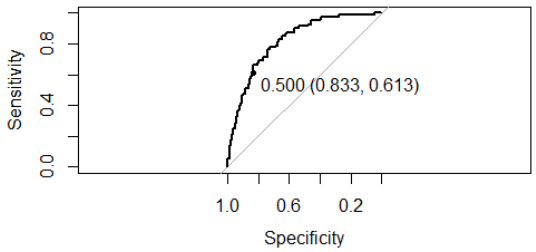
Method ‘nnet’
Area under the curve: 0.88



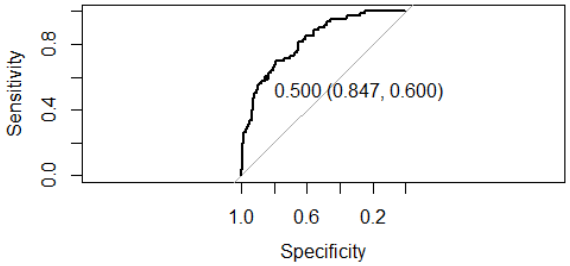
C-7 ROC curves predictions on a test set

Case study 4

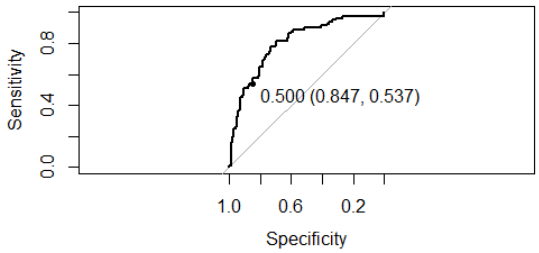
Method ‘gbm’
Area under the curve: 0.82



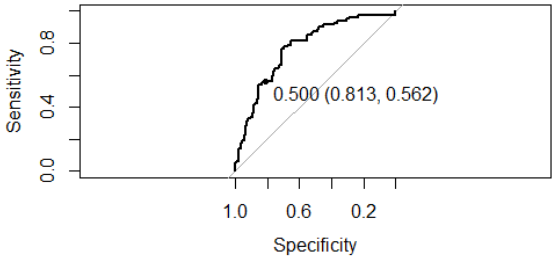
Method ‘svmRadial
Area under the curve: 0.82



Method ‘glmnet’
Area under the curve: 0.80



Method ‘nnet’
Area under the curve: 0.77



C-9 EDA Classification Tree

(Output generated with ‘rpart’ and ‘rattle’ packages on laboratory subset)