12-10-2018

Predictive Models in Brain Connectivity Analysis

Guenadie Nibbs
gxn2511@rit.edu

Predictive Models in Brain Connectivity Analysis

By

Guenadie Nibbs

A thesis submitted in partial fulfillment of the requirement for the Degree of Master of Science in Applied Statistics

Department of Applied Statistics

College of Science

Rochester Institute of Technology

Rochester, NY

Dec 10, 2018

Committee Approval:

Dr. Peter Bajorski Date

Thesis Advisor/Committee Member

Dr. Robert Parody Date

Committee Member

Dr. Minh Pham Date

Committee Member

Abstract

Neuroscience have been the field with most significant contributions to the study of the human brain. The development of new techniques for image acquisition has made possible the improvement of extracting quality information of brain activity. Utilizing functional MRIs, is possible to measure brain activity, based on changes of the oxygen level in the bloodstream, at certain period of time. This imaging data is transformed into numerical values using a software that maps all the information into a data object. Taking advantage of the obtainability of functional connectivity information of the human brain, the present study shows a widespread process to build predictive models, with built-in cross-validation, capable of classifying relevant subject's traits. The investigation propose four powerful statistical methods (Regression, Logistic Regression, Linear Discriminant Analysis and Random Forest) to predict subjects traits based on relationships between brain regions. The final model will be able to use any functional brain connectivity data, which makes this process a generalized approach that others researchers could use to assess further features of the human brain.

Acknowledgment

I would first like to thanks my thesis advisor Dr. Peter Bajorski for always being willing to help and giving me invaluable advises on my academic and professional path. In addition, I would like to express my gratitude to Dr. Parody and Dr. Fokoue, who always had their door open to give me guidelines whenever I needed to talk regarding any topic. My sincere thanks also goes to Shawna Hayes who was very kind and oriented me in a variety of topics while I was pursuing my degree. My fellow classmates who were always willing to study together as a team: Marcos Soriano, Aqil Azad, Charles Muttay and Ahmed Aldhaheri.

Table of Contents

Objective

Understanding the human brain have been one of the most important topics studied by neuroscience. This field has come up with different imaging theories that are used to quantify the properties of brain networks and their components. The brain has been modeled as a complex network system[1] under the premise that neurons make up an interconnected structure into the nervous system. The mainly components of these networks are the Nodes and their Links. When nodes are seen as the regions of interest, based on a certain parcellation scheme, inter-regional relationships will be defined by three types of connectivity: structural, functional and effective.

While structural connectivity approaches are focused on the anatomical parts, which refers to the existence of tracts connecting different brain areas, functional connectivity approaches are focused on the statistical dependence of the signals coming from different areas of the brain. The effective connectivity is similar to the functional but with the peculiarity that it brings the causation elements to the analysis. This type of connectivity allows to determine, when the activation of one area directly causes a change (activation or depression) in another area, or provoke any other special signal. For the effective connectivity, unlike the functional, is possible to evaluate directionality and causality because it provide information about neural interactions.

Neuroimaging and Graph Theory provide a well-established framework to describe brain connectivity, providing a robust and non-invasively method to explore the entire human brain. The present work tries to investigate the existence of relationships among different brain regions and their subject's characteristics using statistical analysis. While neuroscience field has been expanding the knowledge of the human brain, statistical analysis could bring a new interesting approach and provide new evidence on the development of understanding complex networks in the human brain. This analysis will be based on information gathered from the fMIRs of 820 subjects, performed under similar conditions (Resting-State), to apply different statistical techniques that allows the creation of an adequate model to classify/predict subjects traits based on brain connectivity.

[1] The human connectome.

Human Brain and Connectivity

The human brain could be described as the command center of the nervous system. This powerful organ contains a complex network system formed under a crucial organizational structure. It receives signals from all sensors of the body and interpret them to output information that controls movements and decision-making. The largest part of the human brain is the cerebrum, which is divided into two hemispheres. Underneath lies the brainstem, and behind that sits the cerebellum. This complexity, of how these parts interacts, is often study by Neuroscience and the biological field of Networks Sciences.

Brain networks are composed of two important elements, the main unit of the network best known as Nodes and the pairwise relationships between them well known as Links[2]. The nodes are the interacting units within the network, while links could be expressed by three types of connectivity: structural, functional and effective.

Structural connectivity is represented by anatomical connections and could be macroscopically analyzed using Tractography Fiber[3] from Diffusion Tensor[4] of MRIs. On the other hand, functional and effective connections are inferred from measuring the activity of remotes nodes using fMRI or EEG[5] signals. In functional brain networks, measured brain areas interactions focus on statistical dependencies such as coherence, correlations and transfer entropy. Because effective connectivity is measured using neural interactions, is possible to determine directionality and causation, whereas in functional connectivity is not possible to define any kind of directionality.

Imaging Acquisition Methods (MRI vs fMRI)

Development on new imaging acquisition methods and new tools containing dynamical systems of neuroimaging, have made possible the improvement in the quality of brain networks analysis. The method of choice, to extract information of the human brain, have been the Magnetic Resonance Imaging well known as MRI. This technique uses powerful electromagnetic fields and radio waves to take detailed pictures from inside of the body.

---

[2] Also known as Edges.
[3] 3D modelling technique used to represent nerve tracts.
[4] The imaging technique based on water diffusion used in MRIs.
[5] Electroencephalogram.

The standard MRI procedure is used only for structural connectivity because it can only detect physical changes coming from the brain regions. This process uses water molecules of hydrogen nuclei to determine variation of brain tissue respect to space. In contrast, fMRI procedure detect blood oxygen levels to determine the activity[6] coming from those brain regions respect to time.

Generally speaking, the MRI oversees the anatomical structure of the human brain based on a spatial resolution, while the fMRI oversees the metabolic function based on a longitudinal resolution.

Regions of Interest (ROIs)

All information extracted through fMRIs could be transformed into numerical data using an imaging processing software. The data coming from these images will be given by voxels, which represent an individual value into the three dimensional space. Depending on the objective of every researcher, the characteristics of a node could vary. For this reason, nodes should represent, meaningfully and accurately, the elements to be investigate in the system.

In a brain network, theoretically, the most accurate representation of a node will be an individual neuron, having all synapses representing its links. The issue with this approach is that existing technology can only account for areas over 1 mm, while neurons sizes are around 0.004 mm. Furthermore, all signals coming from those nodes are weaker in comparison with other alternatives representations and hence, harder to interpret because they contain more noise.

The minimum sizes a node could take is the same size of a voxel (1 mm). The issue with this representation is that there are around a four million of voxels in a human brain image, each one of them with around eight thousand synapses. Applying computational procedures or even recording this amount of data will represent a huge challenge for any researcher.

Because of these technical limitations, the bigger the amount of neurons or voxels used to represent a single node, the easier will be performing computational analyses on them. In this grouped representation, all interacting neurons and synapses within that given space will represent a singular node in the brain.

[6] Measured by Blood Oxygen Levels Dependence (BOLD)

The challenge with this representation comes with the fact that because nodes can be built freely in terms of size and location, the selection of these features needs to be done carefully depending on the researcher objective. At the moment of selecting the spatial area of each node, is necessary that the area shares similar features so the created node makes sense.

This is how researchers has come up with the definition of Regions of Interest. They represent the variaerity of ranges in the number of nodes and their locations used to create a bigger new one; and the parcellation scheme they have to use to maintain an accurate interpretation of the results. This means that depending on the characteristics[7] of the ROIs, interpretation of the results could differ.

Parcellation Scheme in MRIs and fMRIs

Parcellation schemes split the spatial brain area into a set of non-overlapping regions that present homogeneity with respect to their components. Regions of interest not always will be at the same level of an individual voxel, but it could be at the same level of a set of them. This situation gives place to the existence of two different modalities that could be categorized in: single voxel-based and aggregates voxels-based. In the single voxel-based modality, an individual voxel will represent a single node. This modality has been seen as one of the best representations of relationships within the system[8].

The aggregates voxels-based modality could take two sides, Multi-Voxel analysis[9] and Brain Atlases. The multi-voxel analysis allows the researcher to define any structure of interest[10], while brain atlases provide a pre-defined set of regions with certain base on the brain structure. Because these methods are based in voxels combinations to create the main unit (nodes) as a bigger entity of the system, blood oxygen level dependence signals needs to be average within the ROIs.

Brain atlases have been the predominant option in this topic, not necessarily for been the best way to define regions of interest, but having the best consistency study-to-study. This is possible thanks to the fact that, when researchers use the same set ROIs, studies become comparable between them. The Automate Anatomical Labeling (AAL) has been the most used brain atlas. This software

[7] Nodes size, location and parcellation scheme.
[8] Allowing model-free analysis.
[9] Free ROIs-based analysis.
[10] Customized nodes sizes and parcellation.

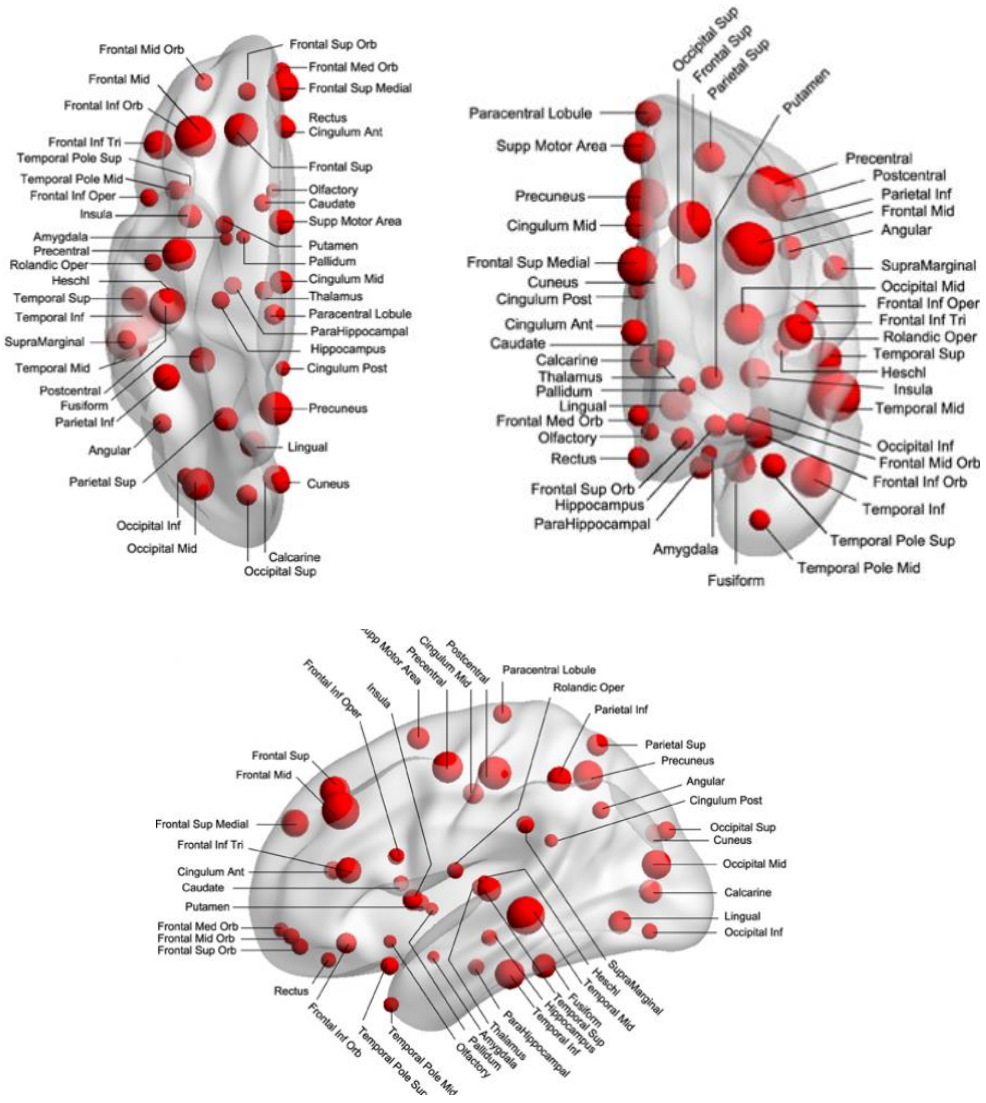
is a digital atlas containing representation of macroscopic brain structures with a 116 (ROIs) parcellation, 58 on each hemisphere of the brain. The following list correspond to the labeling of each region.

1. Precentral_L	31. Cingulum_Ant_L	61. Parietal_Inf_L	91. Cerebelum_Crus1_L
2. Precentral_R	32. Cingulum_Ant_R	62. Parietal_Inf_R	92. Cerebelum_Crus1_R
3. Frontal_Sup_L	33. Cingulum_Mid_L	63. SupraMarginal_L	93. Cerebelum_Crus2_L
4. Frontal_Sup_R	34. Cingulum_Mid_R	64. SupraMarginal_R	94. Cerebelum_Crus2_R
5. Frontal_Sup_Orb_L	35. Cingulum_Post_L	65. Angular_L	95. Cerebelum_3_L
6. Frontal_Sup_Orb_R	36. Cingulum_Post_R	66. Angular_R	96. Cerebelum_3_R
7. Frontal_Mid_L	37. Hippocampus_L	67. Precuneus_L	97. Cerebelum_4_5_L
8. Frontal_Mid_R	38. Hippocampus_R	68. Precuneus_R	98. Cerebelum_4_5_R
9. Frontal_Mid_Orb_L	39. ParaHippocampal_L	69. Paracentral_Lobule_L	99. Cerebelum_6_L
10. Frontal_Mid_Orb_R	40. ParaHippocampal_R	70. Paracentral_Lobule_R	100. Cerebelum_6_R
11. Frontal_Inf_Oper_L	41. Amygdala_L	71. Caudate_L	101. Cerebelum_7b_L
12. Frontal_Inf_Oper_R	42. Amygdala_R	72. Caudate_R	102. Cerebelum_7b_R
13. Frontal_Inf_Tri_L	43. Calcarine_L	73. Putamen_L	103. Cerebelum_8_L
14. Frontal_Inf_Tri_R	44. Calcarine_R	74. Putamen_R	104. Cerebelum_8_R
15. Frontal_Inf_Orb_L	45. Cuneus_L	75. Pallidum_L	105. Cerebelum_9_L
16. Frontal_Inf_Orb_R	46. Cuneus_R	76. Pallidum_R	106. Cerebelum_9_R
17. Rolandic_Oper_L	47. Lingual_L	77. Thalamus_L	107. Cerebelum_10_L
18. Rolandic_Oper_R	48. Lingual_R	78. Thalamus_R	108. Cerebelum_10_R
19. Supp_Motor_Area_L	49. Occipital_Sup_L	79. Heschl_L	109. Vermis_1_2
20. Supp_Motor_Area_R	50. Occipital_Sup_R	80. Heschl_R	110. Vermis_3
21. Olfactory_L	51. Occipital_Mid_L	81. Temporal_Sup_L	111. Vermis_4_5
22. Olfactory_R	52. Occipital_Mid_R	82. Temporal_Sup_R	112. Vermis_6
23. Frontal_Sup_Medial_L	53. Occipital_Inf_L	83. Temporal_Pole_Sup_L	113. Vermis_7
24. Frontal_Sup_Medial_R	54. Occipital_Inf_R	84. Temporal_Pole_Sup_R	114. Vermis_8
25. Frontal_Med_Orb_L	55. Fusiform_L	85. Temporal_Mid_L	115. Vermis_9
26. Frontal_Med_Orb_R	56. Fusiform_R	86. Temporal_Mid_R	116. Vermis_10
27. Rectus_L	57. Postcentral_L	87. Temporal_Pole_Mid_L	
28. Rectus_R	58. Postcentral_R	88. Temporal_Pole_Mid_R	
29. Insula_L	59. Parietal_Sup_L	89. Temporal_Inf_L	
30. Insula_R	60. Parietal_Sup_R	90. Temporal_Inf_R	

Right Hemisphere → _R

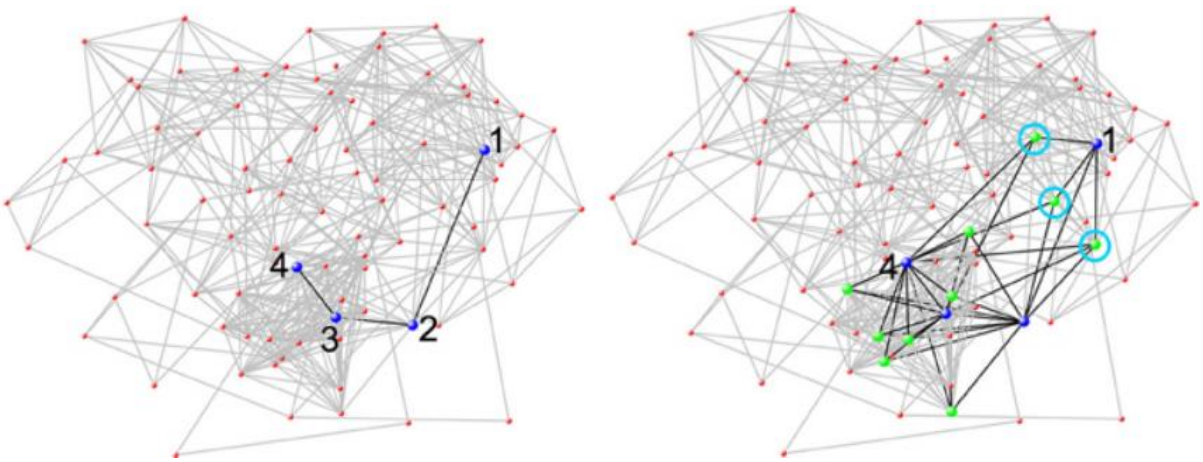
Left Hemisphere → _L

The labeled areas could be represented with spheres that are between 10-20 millimeters of size with individual coordinates for locations with no overlapping. The fMRIs time-series[11] within each region is averaged and the new calculated time-series is used to generate the brain network as the interactions of the new nodes.



In the same manner, all pairwise association could be represented graphically using a network graph. Points in red represents the nodes and gray lines represent all possible synapsis (links).

[11] BOLD signals



Statistical Analysis

Exploratory Data Analysis

The data used to perform the following analysis comes from the Human Connectome Project. The main objective of this project is building a network map[12] that provide a better understanding of anatomical and functional connectivity of the human brain, as well as producing the data that will make this work easier for the researchers. The data consist of the extracted information from a neuroimaging sequences coming from a fMRIs[13], which provide a measure of brain activity based on its functions over time. A connectivity matrix was calculated for each subject with the averaged time series based on the Automate Anatomical Labeling's Regions of Interest.

The final object consist of a fourth-dimensional array with the following lengths:

- Dimension 1 – Brain Activity – 1200 points (Times Series).
- Dimension 2 – ROIs/Nodes – 116 (Whole brain scans) (ALL-Based).
- Dimension 3 – fMRI – 4 Scans.
- Dimension 4 – Subjects – 820 patients.

The data was also pre-processed for motion and distortion correction using the imaging mapping software. The following output represent a preview of the final dataset.

```
## , , 1, ID = 100206
##
##
```

[12] The connectome.
[13] Using a 3.0 Tesla Siemens Scanner.

```
##                 [,1]         [,2]         [,3]
##   [1,]  0.1236054431 -0.020382756  0.106195805
##   [2,]  0.0800266609  0.010988130  0.007470173
##   [3,]  0.0144267235  0.015400782 -0.079054707
##   [4,] -0.0001230086  0.002695427 -0.076813373
##   [5,]  0.0153635991  0.056480333 -0.081829650
##
## , , 2, ID = 100206
##
##
##                [,1]        [,2]        [,3]
##   [1,] -0.01173671 -0.05686451 0.078815854
##   [2,] -0.07736850 -0.15765601 0.006740389
##   [3,] -0.17109888 -0.20602204 0.041703167
##   [4,] -0.06120633 -0.07538534 0.031231134
##   [5,] -0.09076419 -0.11931941 0.023624715
##
## , , 1, ID = 100307
##
##
##                [,1]        [,2]        [,3]
##   [1,]  0.15101818  0.23247057  0.13967349
##   [2,]  0.08053443  0.01286557  0.11152290
##   [3,] -0.03390613 -0.04879962  0.02367411
##   [4,] -0.07632881 -0.05166325 -0.04149430
##   [5,] -0.06947184 -0.06160221 -0.10885455
##
## , , 2, ID = 100307
##
##
##                 [,1]         [,2]        [,3]
##   [1,]  0.247344314  0.285289135  0.17308879
##   [2,]  0.116162643  0.112810976  0.03426607
##   [3,] -0.043211263 -0.009433342 -0.06245121
##   [4,]  0.045134515  0.045080710 -0.18601745
##   [5,] -0.006203855  0.088373547 -0.09191019
```

The ID correspond to the reference number of every subjects. Every ID will be repeated four times throughout the array, each of them containing the information of a particular scan. The scan information will be given as a matrix, where columns represent the regions of interest and the rows will be the time series corresponding to the information of the brain activity over time. Now let's take a look at the information provided on subjects characteristics.

```
##     Subject Gender   Age
## 304  167743      F 26-30
## 713  572045      F 31-35
## 816  727654      F 22-25
```

```
## 803  705341      M 31-35
## 970  996782      F 26-30
## 757  627852      M 26-30
## 792  687062      M 26-30
## 219  149236      F 22-25
## 153  133726      F 31-35
## 637  424939      M 26-30
```

This sample output contains information about subject's characteristics that will be matched with the array information using the reference ID number. The dataset provide Gender and Age ranges for all the subjects involved in the study.

Since the main interest of this investigation is determining possible relationships between brain's regions to predict subject's traits, the first approach will be calculating correlation between those regions. Because the extensive amount of data, correlation will be calculated first for the following sample of the array.

- (2) Subjects
- (3) ROIs
- (2) Scans
- (5) Time Points

```
## , , ID = 100206
##
##
##               [,1]      [,2]
##    [1,]  1.0000000 1.0000000
##    [2,] -0.6064211 0.9132721
##    [3,]  0.9810192 0.3811120
##    [4,] -0.6064211 0.9132721
##    [5,]  1.0000000 1.0000000
##    [6,] -0.7080398 0.4742222
##    [7,]  0.9810192 0.3811120
##    [8,] -0.7080398 0.4742222
##    [9,]  1.0000000 1.0000000
##
## , , ID = 100307
##
##
##              [,1]      [,2]
##    [1,] 1.0000000 1.0000000
##    [2,] 0.9029716 0.9396038
##    [3,] 0.9272139 0.8164144
##    [4,] 0.9029716 0.9396038
```

```
##    [5,] 1.0000000 1.0000000
##    [6,] 0.7677725 0.8429427
##    [7,] 0.9272139 0.8164144
##    [8,] 0.7677725 0.8429427
##    [9,] 1.0000000 1.0000000
```

The resulting array consist of one correlation matrix for each of the subjects in the sampled dataset. The columns represent the number of scans analyzed and the rows represent the correlations of all possible combination of ROIs. At this point, the only interest is just keeping those unique correlations. Ones were removed because correlations between same regions always will be one, and repeated correlations were removed because their inverse always will have the same correlation value. For example, the correlation value between one and six will be the same as six and one.

```
##             [,1] [,2] [,3]
##  [1,] -0.6064211    1    1
##  [2,]  0.9810192    1    1
##  [3,] -0.7080398    1    1
##  [4,]  0.9132721    2    1
##  [5,]  0.3811120    2    1
##  [6,]  0.4742222    2    1
##  [7,]  0.9029716    1    2
##  [8,]  0.9272139    1    2
##  [9,]  0.7677725    1    2
## [10,]  0.9396038    2    2
```

When extracting the values of interest coming from the previous array, a new matrix is generated containing the correlation values with their respectively region combination per row. To validate that all values where extracted correctly, is necessary to build an expression that calculates all possible combinations, taking into account the removals performed based on the dimension of the data sample.

$$Unique.Corrs = \frac{ROIs(ROIs+1)}{2} - ROIs * Length(Scans) * Length(Subjects)$$

In this case, the calculated unique.corrs value is twelve, matching the actual first dimension of the correlation matrix. After validating the process of extracting meaningful correlations from the regions combination of interest, these results could be replicate for the whole dataset provided in the first array. To make the process more efficient, a correlation function was created to performed

the same process in any pre-specified subset of the data. The following output represent the results of executing the function over the whole dataset.

```
##            [,1] [,2]
##  [1,] 0.8819580    1
##  [2,] 0.7799713    1
##  [3,] 0.9227818    1
##  [4,] 0.6540562    1
##  [5,] 0.7055715    2
##  [6,] 0.6076018    2
##  [7,] 0.7610454    2
##  [8,] 0.6326059    2
##  [9,] 0.9130051    3
## [10,] 0.8701052    3
## [11,] 0.8870973    3
## [12,] 0.9141837    3
```

Features Selection and Summarization

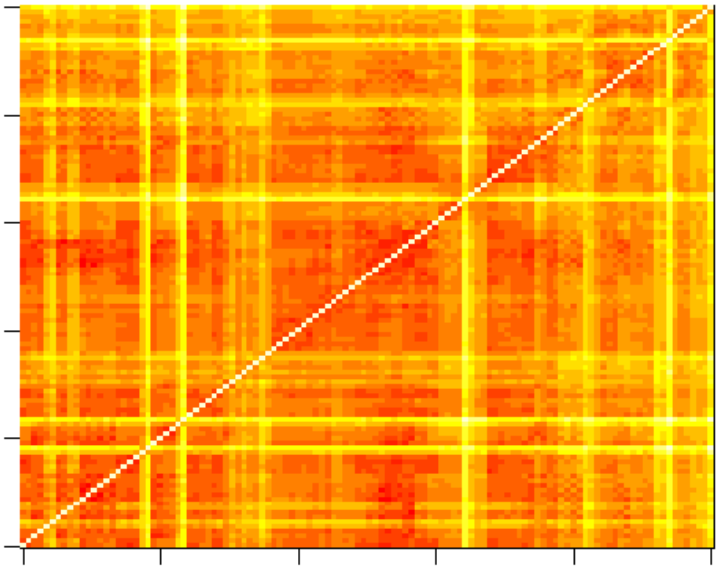
Having determined the correlation value for each regions combination, is important to identify which of those regions have the best subject-to-subject consistency, those are the correlations that are not much different between scans. Using analysis of variance could be assessed any potential difference between correlations among the four scans of each subject. The best way to approach this problem is setting up a One-Way ANOVA, using correlations as the dependent variable (Response) and subjects as the independent variable (Input). Because the independent variable has a large number of possible levels, which might have to be chosen at random, subjects will be set as a random factor.

To get an accurate interpretation of correlations consistency, ANOVA should not be applied on the whole dataset at once. Instead, the process needs to be applied individually over each region combination. These results will be used to extract the variance components of the random effects and build a matrix using the variability coming from residuals. The next output represent a sample of the residual's matrix created using the previous models.

```
##         [,1]   [,2]   [,3]   [,4]   [,5]   [,6]   [,7]   [,8]   [,9]
##  [1,] 100.00  44.47  50.85  48.98  62.88  70.20  52.10  50.60  62.00
##  [2,]  44.47 100.00  49.56  49.85  65.63  71.61  52.32  54.28  62.23
##  [3,]  50.85  49.56 100.00  47.49  62.32  66.86  36.59  43.77  54.36
##  [4,]  48.98  49.85  47.49 100.00  59.56  67.74  43.93  40.62  56.74
##  [5,]  62.88  65.63  62.32  59.56 100.00  60.33  54.26  63.54  55.08
##  [6,]  70.20  71.61  66.86  67.74  60.33 100.00  64.74  68.31  62.01
##  [7,]  52.10  52.32  36.59  43.93  54.26  64.74 100.00  47.32  53.45
```

```
##  [8,]  50.60  54.28  43.77  40.62  63.54  68.31  47.32 100.00  60.68
##  [9,]  62.00  62.23  54.36  56.74  55.08  62.01  53.45  60.68 100.00
## [10,]  63.15  65.64  51.24  55.72  59.52  65.14  51.89  55.68  53.48
##         [,10]
##  [1,]  63.15
##  [2,]  65.64
##  [3,]  51.24
##  [4,]  55.72
##  [5,]  59.52
##  [6,]  65.14
##  [7,]  51.89
##  [8,]  55.68
##  [9,]  53.48
## [10,] 100.00
```

The dimension in this matrix is conformed by the regions identifiers, which means that any indexing (i, j) will represent the variability coming from the error for the combination between region i and region j.



This matrix plot represent the level of variability coming from the error. Assuming the desire results are the lower error percentages between regions, this plot shows with darker colors, where those percentages are concentrated for every region combination.

Predictive Models with Built-in Cross-Validation

At this point of the work, there is enough information to start developing predictive models with brain relationships coming from functional connectivity. This process will involve three steps: features summarization, model building with built-in cross-validation and accuracy assessment.

The first subject characteristic selected as the response variable was Gender, which happens to have three categories. The input variables will be given by the averaged correlations within each subject. There is going to be as many predictor as region combination selected for the model. Because correlations cannot be summed, the following expression was used to summarized the features values.

$$Y = sqrt(mean(corrs^2))$$

The three categories corresponding to the response variable are:

- Male
- Female
- Undefined

```
#Gender Categories
## [1] "M" "F" "U"
```

Because the response is a categorical variable, the problem leans to be a classification type. The error percentages previously calculated will serve as reference to include meaningful features in the model. Since all error percentages were ordered from lowest to highest, every sample taken from this dataset, ensure the best predictors. The set of ranked features for each model will be selected by the following criteria:

- Case 1: The first set of predictors, that will be selected, correspond to the set of correlations where the variability coming from the error was lower than the first quartile of the distribution.
    - Results: 1666 predictors[14] – Random Forest.
- Case 2: The second set of predictors will be given by the set of correlations where the variability coming from the error is the minimum value of the distribution.
    - Results: 1 predictor – Logistic Regression – The Best Predictor
- Case 3: The third set of predictors will be given by the set of correlations supported by the statistical technique used to model the data without causing complete or quasi-separation.

[14] Completely ranked dataset.

The supported set with less predictors of all statistical technique will be replicated over the rest for comparison purposes.

- Logistic Regression AIC: 120 predictors
- Logistic Regression Best Predictors: 120 predictors
- Linear Discriminant Analysis: 120 predictors
- Random Forest: 120 predictor

Finally, the "Undefined" category of the response variable will be excluded from the analysis, because only contains one subject of the 820, which makes it irrelevant.

Ranked Predictors and Model Assumptions

The following output present a sample of the dataset containing all region combinations with their error percentages. These results are already ranked for the best error percentages[15].

```
head(bestCombs.df,12)
```

```
##        [,1] [,2] [,3]
##  [1,] 27.47   11   61
##  [2,] 31.78   13   61
##  [3,] 32.36   12   14
##  [4,] 32.64   11   62
##  [5,] 33.24   11   13
##  [6,] 33.35    8   66
##  [7,] 34.14    7   61
##  [8,] 34.50   65   67
##  [9,] 34.55   62   65
## [10,] 34.66   12   62
## [11,] 34.86   12   66
## [12,] 35.29   65   66
```

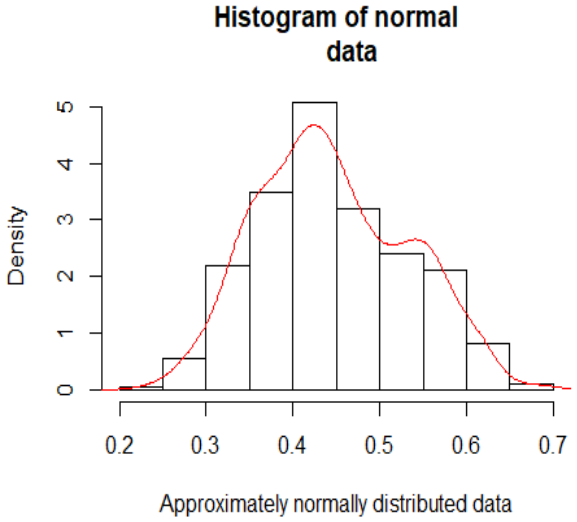
The first column correspond to the calculated error percentage, the second column is the first region and the third column is the second region of the combination. Based on that sample, it was possible to obtain the following set:

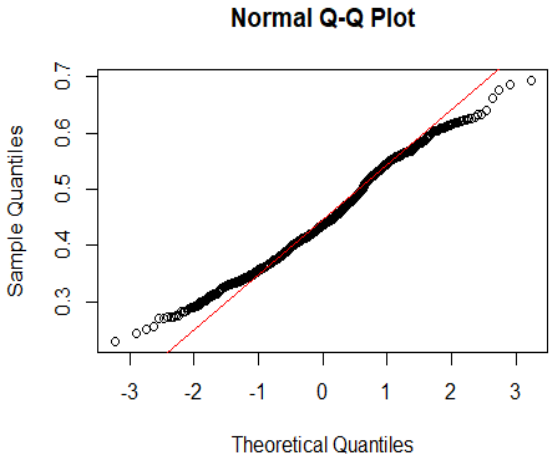
```
##    Group.1         x Gender   Age
## 1   100206 0.4417249      M 26-30
## 2   100307 0.3409603      F 26-30
## 3   100408 0.6088309      M 31-35
## 4   100610 0.4345944      M 26-30
## 5   101006 0.3359919      F 31-35
## 6   101107 0.6758970      M 22-25
```

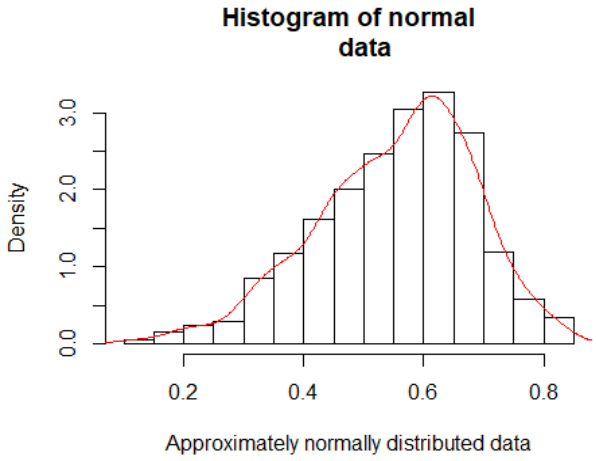
[15] Lowest (Best) to Highest (Worst).

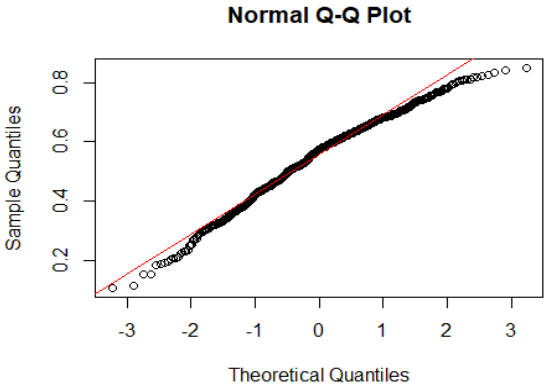
```
## 7   101309 0.4469654      M 26-30
## 8   101915 0.4621885      F 31-35
## 9   102008 0.5967210      M 22-25
## 10  102311 0.5345915      F 26-30
## 11  102513 0.4547970      M 26-30
## 12  102816 0.4242340      F 26-30
```

Now let's take a look to the assumptions of the new dataset (n < Q1) and the original dataset.









Both dataset samples seems to approximate the normal distribution. The main difference between them correspond to the level of the skewness they present, while the whole dataset seems to have

a more centered distribution, the sample dataset is skewed to the left. As expected, the filtered data is centered towards higher correlation values, because of the lower error percentages.

Model 1: Logistic Regression with the Best Predictor - Gender

Now let's see, how the simplest model with intercept, can predict subject's gender based on the relationship between the brain region combination with the lowest error percentage using logistic regression.

```
##
## Call:
## glm(formula = as.factor(Gender) ~ corrsLowest, family = binomial(link = "l
ogit"),
##     data = corr.charLowest)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7999  -1.0277  -0.5818   1.0742   2.2592
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -3.7360     0.3839  -9.731   <2e-16 ***
## corrsLowest   6.2796     0.6623   9.481   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1125.7  on 818  degrees of freedom
## Residual deviance: 1015.7  on 817  degrees of freedom
## AIC: 1019.7
##
## Number of Fisher Scoring iterations: 3
```

The generated model shows p-values lower than 0.05, which indicates statistical significance of the predictor. The following results correspond to the predictions calculated using the model for the 819 subjects.

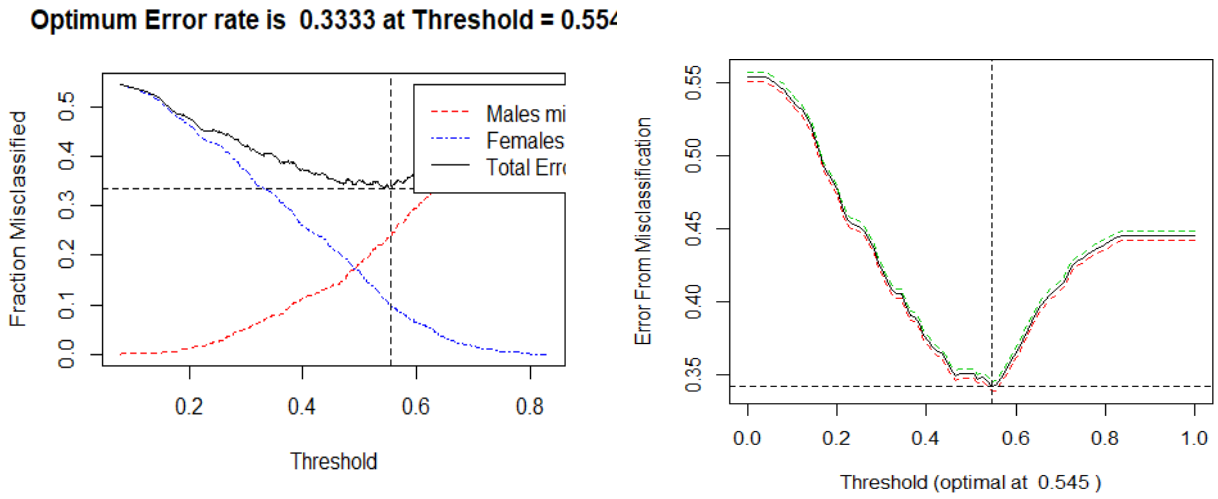
```
##      CorrHatFac
##         F   M Sum
##   F   274 180 454
##   M   111 254 365
##   Sum 385 434 819
```

```
#Accuracy Percentage
```

```
## [1] 64.47%
```

The model with one predictor and the intercept was capable of successfully classified 274 females as females, and 254 males as males. On the other side, it incorrectly classified 111 females as males, and 180 males as females. Based on these results, the model performed predictions with a 64.47 percent of accuracy. The following step consist on using a model selection procedure to determine which predictors are more relevant for the next sample of the analysis.

The following output compares misclassification tables using optimal thresholds with and without cross-validation.




```
## The optimal threshold for the 10-fold cross validation is: 0.5454545454545
46
## gender.prediction   F   M
##             FALSE 364 187
##             TRUE   90 178
## [1] 66.18%
## The error rate for 10-fold cv is:0.338217338217338
```

The model with cross validation got an accuracy rate of 66.18 percent, while the non-cross-validated model with optimal threshold had a 66.67 percent.

Model 2: Logistic Regression with AIC Selection - Gender

```
#Showing a summary of the results

##
## Call:
## glm(formula = as.factor(Gender) ~ V81 + V72 + V45 + V1 + V49 +
##     V53 + V14 + V92 + V36 + V78 + V66 + V84 + V70 + V94 + V41 +
##     V111 + V79 + V116 + V15 + V8 + V118 + V13 + V102 + V114 +
```

```
##     V18 + V115 + V64 + V34 + V20 + V5 + V40 + V108 + V31 + V3 +
##     V99 + V51 + V113 + V82 + V83 + V95, family = binomial(link = "logit"),
##     data = corr.charAIC)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1475  -0.4474  -0.0482   0.4524   2.6914
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -10.8962     1.8717  -5.821 5.83e-09 ***
## V81          14.3861     2.3386   6.152 7.67e-10 ***
## V72         -16.3667     2.9872  -5.479 4.28e-08 ***
## V45           3.8330     0.9874   3.882 0.000104 ***
## V1            2.4593     1.2533   1.962 0.049726 *
## V49           8.3699     1.5247   5.490 4.03e-08 ***
## V53          -4.3834     1.2058  -3.635 0.000278 ***
## V14           6.4554     1.4576   4.429 9.48e-06 ***
## V92         -13.3200     2.4067  -5.535 3.12e-08 ***
## V36           6.4035     1.4169   4.519 6.20e-06 ***
## V78          -6.0896     1.4844  -4.102 4.09e-05 ***
## V66           2.7489     1.4629   1.879 0.060227 .
## V84          -5.6306     1.4932  -3.771 0.000163 ***
## V70           4.9579     1.3940   3.557 0.000376 ***
## V94           9.6088     2.2833   4.208 2.57e-05 ***
## V41          -6.2282     1.9216  -3.241 0.001190 **
## V111          8.9915     2.6827   3.352 0.000803 ***
## V79           5.7141     1.6490   3.465 0.000530 ***
## V116         -4.2167     1.2642  -3.336 0.000851 ***
## V15           5.9021     1.4787   3.991 6.57e-05 ***
## V8           -3.2683     1.2285  -2.660 0.007804 **
## V118          5.3734     1.5065   3.567 0.000361 ***
## V13          -4.9213     1.5338  -3.208 0.001334 **
## V102          5.5252     1.4497   3.811 0.000138 ***
## V114         -3.7408     1.4365  -2.604 0.009211 **
## V18           4.3220     1.6005   2.700 0.006924 **
## V115         -5.2198     1.5612  -3.343 0.000827 ***
## V64          -4.0766     1.5484  -2.633 0.008471 **
## V34           3.9466     1.6674   2.367 0.017939 *
## V20          -2.7104     1.2269  -2.209 0.027160 *
## V5            2.9672     1.6945   1.751 0.079946 .
## V40          -3.5030     1.3383  -2.618 0.008857 **
## V108          2.2147     1.2300   1.800 0.071783 .
## V31           2.6211     1.2872   2.036 0.041721 *
## V3            4.1641     1.9762   2.107 0.035107 *
## V99           2.3656     1.2334   1.918 0.055123 .
## V51          -5.0552     2.9176  -1.733 0.083163 .
## V113         -2.7361     1.4944  -1.831 0.067108 .
## V82          -3.8616     1.6072  -2.403 0.016275 *
## V83           2.6319     1.5487   1.699 0.089243 .
```

```
## V95          -2.5757     1.1182  -2.303 0.021259 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1125.68  on 818  degrees of freedom
## Residual deviance:  534.79  on 778  degrees of freedom
## AIC: 616.79
##
## Number of Fisher Scoring iterations: 6
```

```
#Contingency table and marginal sums
cTab.AIC <- table(corrset.aic$data$Gender, CorrHatFac.AIC)
addmargins(cTab.AIC)
```

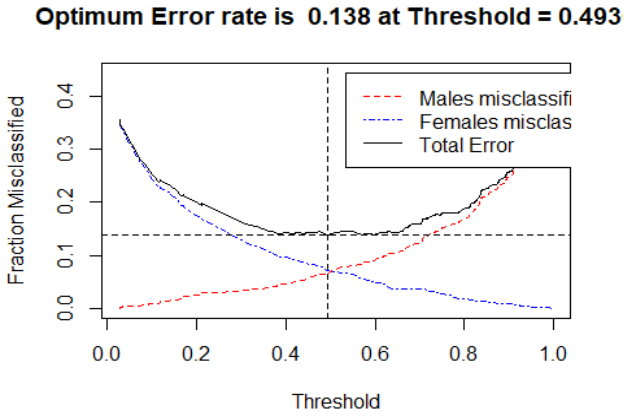
```
##      CorrHatFac.AIC
##         F   M Sum
##   F   384  70 454
##   M    47 318 365
##   Sum 431 388 819
```

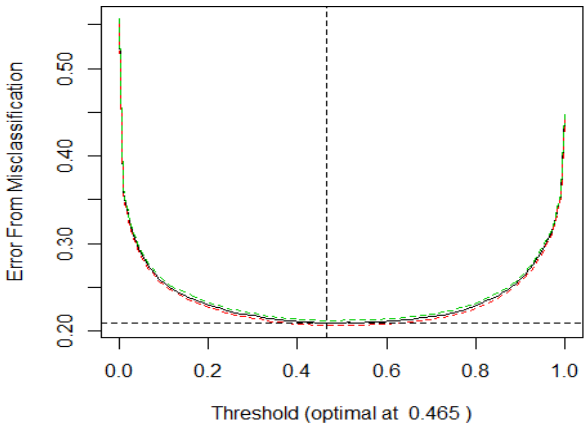
```
#Accuracy Percentage
round(sum(diag(cTab.AIC)) / sum(cTab.AIC)*100 , 2)
```

```
## [1] 85.71%
```

After the selection procedure, the resulting model possess 40 predictors, where 34 of those have p-values lower than 0.05, expressing statistical significance. Comparing AIC values[16], the model with just one predictor has a higher value of 1019.7 versus the last model with 616.79, which express better evidence towards this model. The AIC model was capable of successfully classified 384 females as females, and 318 males as males. On the other side, it incorrectly classified 47 females as males, and 70 males as females. Based on these results, the model performed predictions with an 85.71 percent of accuracy. Now is time to validate how the model will perform in practice using 10-fold cross-validation.

[16] The lower the better.





The following output compares misclassification tables using optimal thresholds with and without cross-validation.

```
## Misclassification table using optimal threshold
## opt.pred   F   M
##    FALSE 395  54
##    TRUE   59 311

## [1] 86.20%

## The optimal threshold for the 10-fold cross validation is: 0.4646464646464
65
## gender.prediction   F   M
##             FALSE 387  51
##             TRUE   67 314
## [1] 85.59%
## The error rate for 10-fold cv is:0.144078144078144
```

There is not much difference coming from the error rates. While the model without cross-validation and optimal threshold was able to predict with an accuracy rate of 86.20%, the cross-validated model got an 85.59% accuracy rate. All three models had similar performance for the AIC selected predictors.

Model 3: Logistic Regression with Ranked Predictors - Gender

```
## Call:
## glm(formula = as.factor(Gender) ~ ., family = binomial(link = "logit"),
##     data = corr.charLR)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9840  -0.3319  -0.0196   0.3787   2.9902
##
```

```
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -16.83044    4.85017  -3.470 0.000520 ***
## V1            4.06852    3.55390   1.145 0.252291
## V2           -1.70836    3.59235  -0.476 0.634391
## V3            5.27994    2.56971   2.055 0.039910 *
## V4           -2.37441    4.20075  -0.565 0.571913
## V5            1.12381    2.57893   0.436 0.663008
## V6            3.15330    2.54732   1.238 0.215757
## V7            5.06552    2.90380   1.744 0.081081 .
## V8           -5.17593    3.12187  -1.658 0.097326 .
## V9           -7.06325    2.78726  -2.534 0.011273 *
## V10           1.67898    2.63952   0.636 0.524716
## V11          -3.55372    3.15284  -1.127 0.259680
## V12           4.01480    2.34874   1.709 0.087387 .
## V13          -5.40351    2.77452  -1.948 0.051469 .
## V14           7.35878    3.40858   2.159 0.030858 *
## V15           8.73171    2.06423   4.230 2.34e-05 ***
## V16           5.44934    3.22889   1.688 0.091472 .
## V17          -4.95084    2.63277  -1.880 0.060044 .
## V18           8.48805    2.61464   3.246 0.001169 **
## V19           4.08331    3.24316   1.259 0.208011
## V20          -4.21336    2.65278  -1.588 0.112222
## V21         -13.40283    5.28108  -2.538 0.011152 *
## V22          -1.17588    3.40804  -0.345 0.730072
## V23          -3.99715    2.88580  -1.385 0.166020
## V24          -4.31020    2.73866  -1.574 0.115526
## V25          -2.33461    3.17528  -0.735 0.462189
## V26          -1.55138    2.01594  -0.770 0.441562
## V27          -2.18409    2.37163  -0.921 0.357092
## V28           1.05519    3.36646   0.313 0.753946
## V29          -1.21789    2.08861  -0.583 0.559820
## V30          -2.86585    2.80545  -1.022 0.307005
## V31           7.85406    3.40988   2.303 0.021261 *
## V32          -0.29939    3.11477  -0.096 0.923425
## V33           1.79562    3.36106   0.534 0.593173
## V34           4.81499    2.48245   1.940 0.052427 .
## V35           4.92248    2.91682   1.688 0.091485 .
## V36           8.61552    1.96997   4.373 1.22e-05 ***
## V37          -0.26404    3.23659  -0.082 0.934982
## V38           2.00601    2.64578   0.758 0.448337
## V39          -4.13753    3.33993  -1.239 0.215418
## V40          -5.02944    2.32823  -2.160 0.030757 *
## V41          -3.65050    3.36498  -1.085 0.277987
## V42           1.77433    2.42785   0.731 0.464888
## V43          -2.21675    1.82199  -1.217 0.223730
## V44           3.58716    3.12456   1.148 0.250947
## V45           5.02299    1.29709   3.873 0.000108 ***
## V46           0.16493    3.49299   0.047 0.962339
## V47           1.61355    4.94383   0.326 0.744139
```

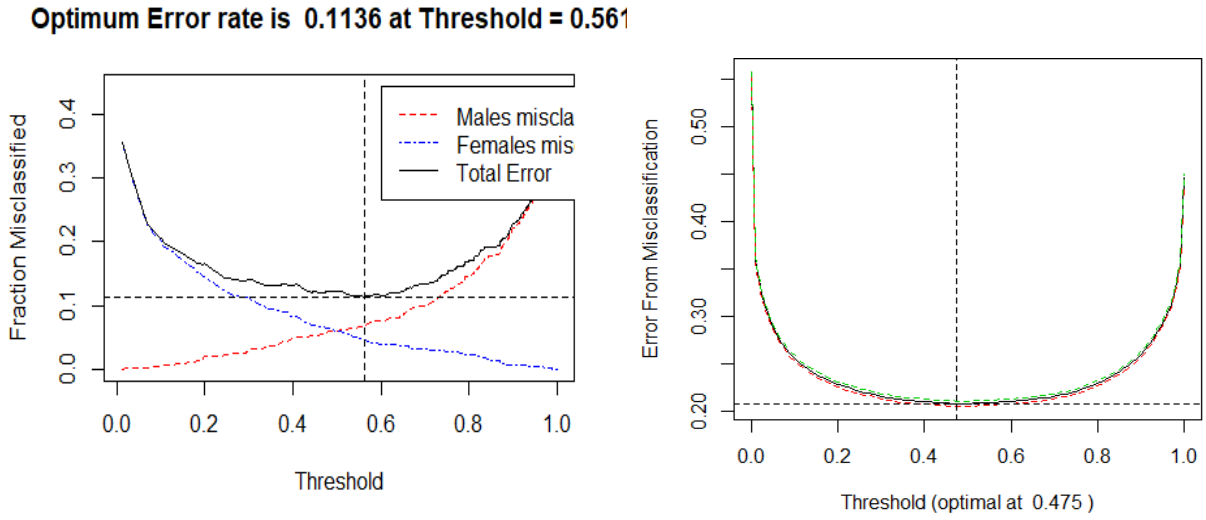
```
## V48           2.18061    1.75890   1.240 0.215066
## V49          11.60237    2.33447   4.970 6.69e-07 ***
## V50          -4.62781    3.14644  -1.471 0.141343
## V51          -9.18095    3.84816  -2.386 0.017042 *
## V52           0.47310    2.66853   0.177 0.859280
## V53          -6.74053    1.87875  -3.588 0.000334 ***
## V54          20.77548    8.28740   2.507 0.012180 *
## V55           2.39387    2.59692   0.922 0.356629
## V56           3.56943    2.26425   1.576 0.114927
## V57           1.68344    3.82624   0.440 0.659957
## V58         -11.82469    6.76102  -1.749 0.080300 .
## V59          -5.96736    3.89679  -1.531 0.125682
## V60           2.51504    3.71389   0.677 0.498280
## V61           1.48568    3.24913   0.457 0.647487
## V62           4.86983    2.65070   1.837 0.066182 .
## V63          -4.91599    2.79070  -1.762 0.078144 .
## V64          -5.70067    3.27429  -1.741 0.081676 .
## V65         -12.35326    6.56107  -1.883 0.059726 .
## V66           1.34283    3.30314   0.407 0.684353
## V67          -6.55650    5.03033  -1.303 0.192441
## V68           3.79365    2.60910   1.454 0.145945
## V69           5.44436    2.59903   2.095 0.036192 *
## V70          -0.80733    2.61125  -0.309 0.757188
## V71           2.77591    2.92519   0.949 0.342637
## V72         -19.33270    4.05879  -4.763 1.91e-06 ***
## V73           1.59415    6.16632   0.259 0.796002
## V74          -0.89093    2.97820  -0.299 0.764825
## V75           1.74384    2.83182   0.616 0.538024
## V76           9.08507    7.51352   1.209 0.226600
## V77          -0.16976    2.99892  -0.057 0.954857
## V78          -5.77552    2.35615  -2.451 0.014236 *
## V79           8.80062    2.29499   3.835 0.000126 ***
## V80           0.97323    8.05802   0.121 0.903867
## V81          10.16573    3.54151   2.870 0.004099 **
## V82          -9.14538    3.25397  -2.811 0.004946 **
## V83           4.07491    3.09443   1.317 0.187888
## V84          -5.23863    2.15977  -2.426 0.015285 *
## V85           6.88705    4.85802   1.418 0.156288
## V86          -1.76269    2.52668  -0.698 0.485408
## V87           0.36032    5.26484   0.068 0.945436
## V88          -1.06489    2.18178  -0.488 0.625492
## V89           2.85573    2.23818   1.276 0.201985
## V90           3.99123    3.15016   1.267 0.205157
## V91          -0.03803    2.90096  -0.013 0.989541
## V92         -18.96632    7.35364  -2.579 0.009904 **
## V93          -0.47182    2.21347  -0.213 0.831204
## V94          10.45989    3.28747   3.182 0.001464 **
## V95          -3.80219    2.25645  -1.685 0.091982 .
## V96           2.71555    2.63779   1.029 0.303253
## V97           8.14971    5.10449   1.597 0.110360
```

```
## V98           3.25078    2.26334   1.436 0.150924
## V99           4.37004    3.71561   1.176 0.239543
## V100         -2.95125    2.95542  -0.999 0.317994
## V101         -1.97238    3.83430  -0.514 0.606969
## V102          5.38394    3.32042   1.621 0.104918
## V103         -0.34987    1.96531  -0.178 0.858706
## V104          0.38002    2.96253   0.128 0.897932
## V105         -1.55366    2.16217  -0.719 0.472408
## V106          6.78504    7.60361   0.892 0.372208
## V107         -2.15777    4.90537  -0.440 0.660025
## V108          3.37098    1.60759   2.097 0.036001 *
## V109          4.38324    2.41808   1.813 0.069879 .
## V110         -2.03526    2.57050  -0.792 0.428490
## V111          5.44921    6.99906   0.779 0.436238
## V112         -3.43345    2.61566  -1.313 0.189302
## V113         -3.19998    3.33536  -0.959 0.337353
## V114         -3.02525    1.79012  -1.690 0.091034 .
## V115         -4.99377    2.54012  -1.966 0.049303 *
## V116         -9.96946    2.82168  -3.533 0.000411 ***
## V117         -0.87289    3.14468  -0.278 0.781338
## V118          7.22483    3.53859   2.042 0.041179 *
## V119          6.80636    2.83647   2.400 0.016413 *
## V120         -5.27296    3.10517  -1.698 0.089484 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1125.68  on 818  degrees of freedom
## Residual deviance:  457.53  on 698  degrees of freedom
## AIC: 699.53
##
## Number of Fisher Scoring iterations: 7

##      CorrHatFac.LR
##         F   M Sum
##   F   398  56 454
##   M    44 321 365
##   Sum 442 377 819

## [1] 87.79%
```

The model obtained using logistic regression contains 29 variables with p-values lower than 0.05for statistical significance, intercept included. While the AIC value is lower than the one predictor model, the AIC is higher than the AIC selected predictors. . Based on these results, the model performed predictions with an 85.71 percent of accuracy. Now is time to validate how the model will perform in practice using 10-fold cross-validation.



The following output compares misclassification tables using optimal thresholds with and without cross-validation.

```
## Misclassification table using optimal threshold
## opt.pred   F   M
##    FALSE 417  56
##    TRUE   37 309

## [1] 88.64%

## The optimal threshold for the 10-fold cross validation is: 0.4747474747474
75
## gender.prediction   F   M
##             FALSE 401  48
##             TRUE   53 317
## [1] 87.67%
## The error rate for 10-fold cv is:0.123321123321123
```

There is not much difference coming the accuracy of the models above. While the model without cross-validation and optimal threshold was able to predict with an accuracy rate of 88.64%, the cross-validated model got an 87.67% accuracy rate. All three models had similar performance for the 120 ranked predictors.

Model 4: Linear Discriminant Analysis with Predictors - Gender

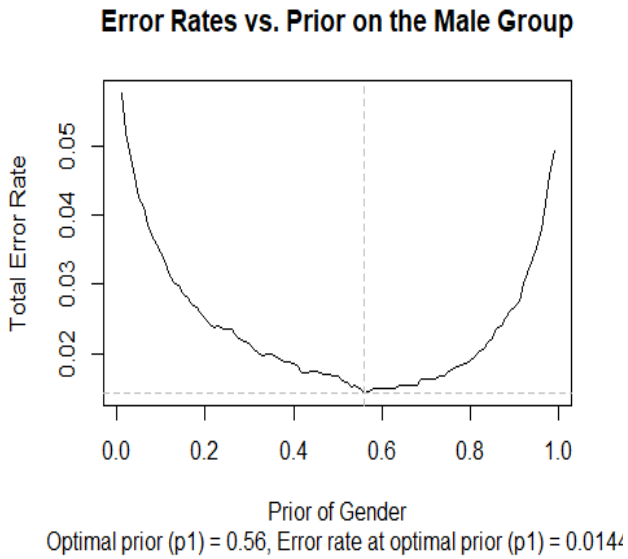
The second statistical technique proposed for this classification problem was Linear Discriminant Analysis. This method allows characterizing two or more classes of objects based on means and variances, whose results must be used as a linear classifier.

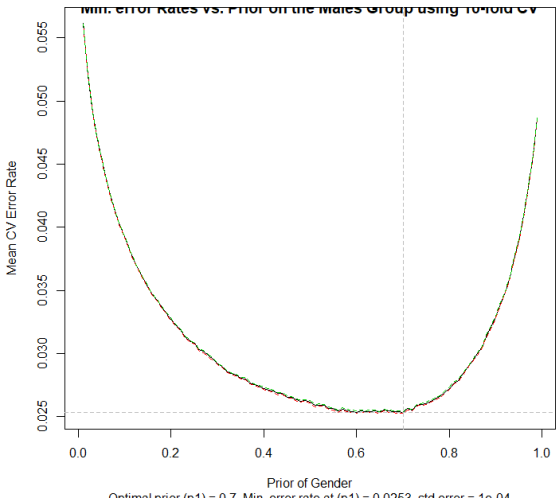
```
##         Length Class  Mode
## prior     2    -none- numeric
## counts    2    -none- numeric
## means   240    -none- numeric
## scaling 120    -none- numeric
## lev       2    -none- character
## svd       1    -none- numeric
## N         1    -none- numeric
## call      3    -none- call
## terms     3    terms  call
## xlevels   0    -none- list

##          Actual
## Predicted   F   M
##         F 402  44
##         M  52 321

## [1] 88.28%
```

The LDA model was capable of successfully classified 402 females as females, and 321 males as males. On the other side, it incorrectly classified 52 females as males, and 44 males as females. Based on these results, the model performed predictions with an 88.28 percent of accuracy. Now is time to validate how the model will perform in practice using 10-fold cross-validation.





The following output compares misclassification tables using optimal thresholds with and without cross-validation.

```
Classif.table
```

```
##
##       F   M
##   F 402  44
##   M  52 321
```

```
## [1] 88.28%
```

```
# Misclassification error of the 100 rounds of 10 fold CV
CV.error <- CV.error.f.1()
```

```
## Average misclassified cases using 10 fold CV =  171.63
## Standard error of total minimized error of misclassification =  0.5054561
```

```
##       F   M
##   F 294 257
##   M 160 108
```

```
## [1] 49.08%
```

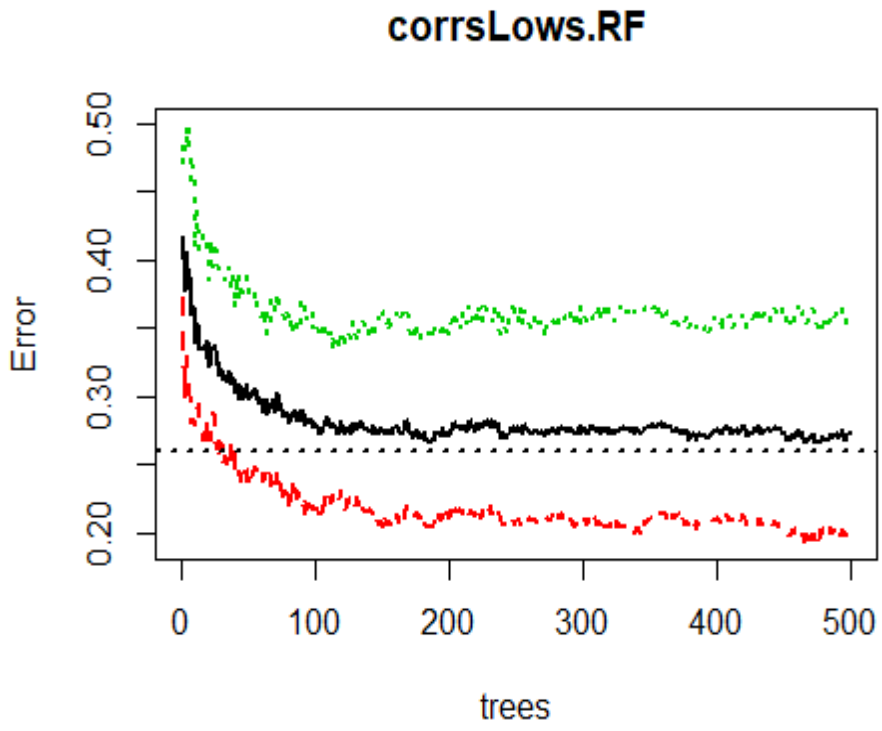
The LDA technique is presenting a different behavior compared to the other techniques. The accuracy rate for the trained model and the best prior is almost the same. On the other hand, the cross-validated model using best prior, is presenting a lower accuracy rate of 49.08%.

Model 5: Random Forest with Ranked Predictors - Gender

The last statistical technique to be applied in this classification problem will be Random Forest. This is a more general technique that uses a multitude of decision trees to determine which class is the best for the object to be classified.

```
## Random Forest
##
## 819 samples
## 120 predictors
##   2 classes: 'F', 'M'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 738, 737, 736, 738, 738, 737, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.728052  0.4440291
```

```
##
## Tuning parameter 'mtry' was held constant at a value of 11
```

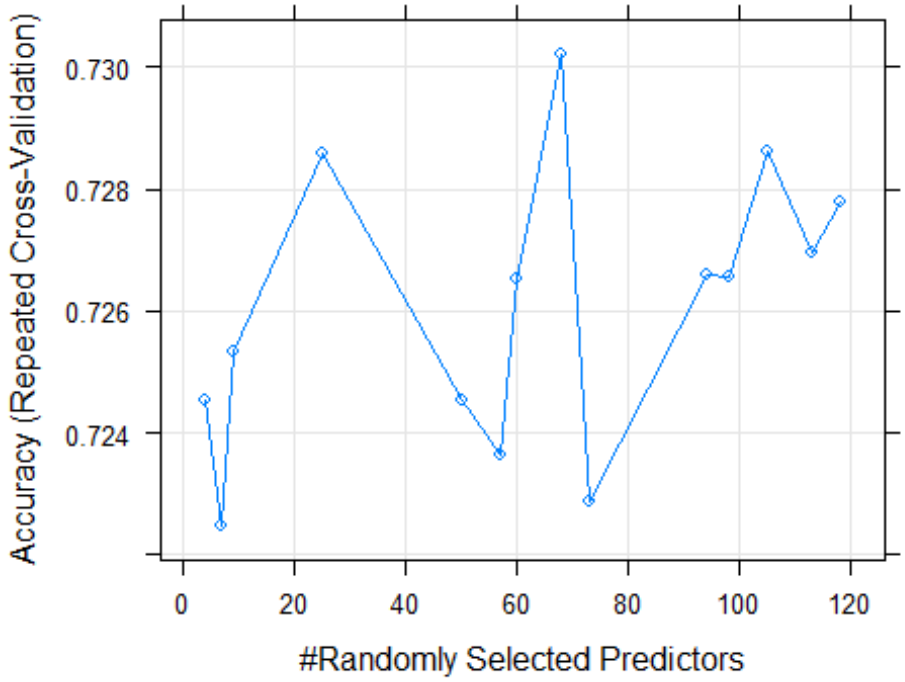


The accuracy rate corresponding to the random forest technique when using 120 ranked predictors with mtry[17] constant is 72.80 percent.

```
## Random Forest
##
## 819 samples
## 120 predictors
##   2 classes: 'F', 'M'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 737, 736, 737, 736, 738, 737, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##     4   0.7245164  0.4355670
##     7   0.7224595  0.4316812
##     9   0.7253294  0.4371518
##    25   0.7285918  0.4437038
```

[17] If mtry is no constant a set of predictors will be selected at random.

```
##    50   0.7245267  0.4358481
##    57   0.7236387  0.4346460
##    60   0.7265290  0.4403687
##    68   0.7302224  0.4479660
##    73   0.7228603  0.4335629
##    94   0.7265786  0.4410374
##    98   0.7265691  0.4410037
##   105   0.7286017  0.4453208
##   113   0.7269409  0.4412480
##   118   0.7277939  0.4432234
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 68.
```



Using a dynamic value for mtry, its shows that the best configuration for this set of ranked predictors correspond to best 68, where the accuracy rate ends up at 73.02 percent.

Model 6: Regression with Ranked Predictors - Age

The second subject characteristic selected as the response variable was Age, which happens to have a five ordinal type categorical variable. Because the first statistical technique to apply will be regression, which manage continuous responses, the following transformation is necessary.

```
corr.charRE$AgeCod[corr.charRE$Age == "22-25"] <- 23.5
corr.charRE$AgeCod[corr.charRE$Age == "26-30"] <- 28
corr.charRE$AgeCod[corr.charRE$Age == "31-35"] <- 33
corr.charRE$AgeCod[corr.charRE$Age == "36+"] <- 36
```

Similar to the previous models, the regression analysis will be performed using the ranked predictors, thus the model accuracy could be compare with the others at the same level. Now the summary of the model is presented.

```
## Call:
## lm(formula = AgeCod ~ ., data = corr.charRE)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.8707 -2.2918  0.0086  2.4354  8.3629
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 36.059421   3.812125   9.459  < 2e-16 ***
## V1          -3.221107   3.059959  -1.053  0.29286
## V2           3.944721   3.093685   1.275  0.20270
## V3          -4.119666   2.234707  -1.843  0.06568 .
## V4          -3.756276   3.646059  -1.030  0.30326
## V5          -0.702781   2.379033  -0.295  0.76777
## V6          -0.878356   2.210614  -0.397  0.69124
## V7           2.920904   2.700048   1.082  0.27972
## V8           2.030905   2.739726   0.741  0.45877
## V9          -0.886810   2.333384  -0.380  0.70402
## V10          3.471999   2.152741   1.613  0.10723
## V11         -3.520009   2.401257  -1.466  0.14312
## V12          1.526494   2.088241   0.731  0.46503
## V13         -0.662649   2.437520  -0.272  0.78581
## V14         -0.603333   2.889385  -0.209  0.83466
## V15         -1.006256   1.710633  -0.588  0.55656
## V16         -0.062363   2.729969  -0.023  0.98178
## V17          3.293064   2.251382   1.463  0.14400
## V18          0.499859   2.153971   0.232  0.81656
## V19          0.311833   2.809856   0.111  0.91167
## V20         -2.501441   2.095747  -1.194  0.23305
## V21         -9.025169   4.643123  -1.944  0.05232 .
## V22          8.187431   2.840418   2.882  0.00407 **
## V23         -0.345812   2.391837  -0.145  0.88508
## V24          2.414605   2.311054   1.045  0.29647
## V25          5.727392   2.810637   2.038  0.04195 *
## V26         -5.575796   1.731629  -3.220  0.00134 **
## V27          1.588582   2.001758   0.794  0.42770
## V28          4.449219   2.874064   1.548  0.12206
## V29          0.791144   1.809377   0.437  0.66207
## V30          0.119861   2.452972   0.049  0.96104
```

```
## V31          2.635439   2.895063   0.910  0.36297
## V32         -1.441686   2.690293  -0.536  0.59221
## V33         -1.097037   2.924615  -0.375  0.70770
## V34         -0.946230   2.111123  -0.448  0.65414
## V35         -4.968160   2.482858  -2.001  0.04578 *
## V36         -0.308552   1.582182  -0.195  0.84544
## V37          3.049985   2.646271   1.153  0.24949
## V38         -3.279255   2.298886  -1.426  0.15418
## V39          1.682060   2.772075   0.607  0.54419
## V40         -0.492762   2.024349  -0.243  0.80775
## V41         -3.291159   2.929262  -1.124  0.26159
## V42         -2.680493   2.146202  -1.249  0.21210
## V43          3.378759   1.651655   2.046  0.04116 *
## V44          0.578282   2.770722   0.209  0.83473
## V45         -2.172712   1.169483  -1.858  0.06361 .
## V46          2.527324   2.981010   0.848  0.39684
## V47          2.430394   4.003736   0.607  0.54403
## V48         -2.459040   1.551085  -1.585  0.11334
## V49         -3.413571   1.793460  -1.903  0.05741 .
## V50         -6.474533   2.682880  -2.413  0.01607 *
## V51         -1.892125   3.252336  -0.582  0.56091
## V52         -0.841029   2.279961  -0.369  0.71233
## V53          3.091753   1.691082   1.828  0.06794 .
## V54          7.165981   7.124494   1.006  0.31485
## V55         -3.042102   2.297575  -1.324  0.18592
## V56         -2.462247   1.824397  -1.350  0.17757
## V57          2.539803   3.434199   0.740  0.45981
## V58          2.447733   5.804761   0.422  0.67339
## V59         -0.303263   3.296894  -0.092  0.92674
## V60         -1.896050   3.185981  -0.595  0.55195
## V61          1.348752   2.733497   0.493  0.62187
## V62          2.689739   2.172202   1.238  0.21604
## V63         -5.980263   2.512553  -2.380  0.01757 *
## V64          4.376951   2.797697   1.564  0.11816
## V65         -7.471391   5.781401  -1.292  0.19668
## V66          0.088726   2.946397   0.030  0.97599
## V67          7.716587   4.177376   1.847  0.06514 .
## V68         -3.409332   2.188314  -1.558  0.11969
## V69          0.436587   2.193178   0.199  0.84227
## V70          4.109876   2.193712   1.873  0.06142 .
## V71          1.742735   2.392792   0.728  0.46666
## V72          0.351053   3.200243   0.110  0.91268
## V73         -2.980404   5.198215  -0.573  0.56659
## V74         -6.389469   2.513355  -2.542  0.01123 *
## V75         -1.198395   2.439948  -0.491  0.62347
## V76          1.691629   6.571425   0.257  0.79693
## V77          3.341946   2.664199   1.254  0.21012
## V78         -0.309661   2.037186  -0.152  0.87923
## V79         -3.479120   1.893818  -1.837  0.06662 .
## V80         -1.366272   7.418309  -0.184  0.85393
```

```
## V81          3.128931   2.951677   1.060  0.28949
## V82         -0.267533   2.819736  -0.095  0.92444
## V83          0.130288   2.666107   0.049  0.96104
## V84         -5.856971   1.938523  -3.021  0.00261 **
## V85          4.589318   4.224905   1.086  0.27774
## V86          2.951686   2.240163   1.318  0.18806
## V87         -6.669455   4.239618  -1.573  0.11614
## V88          1.842232   1.982811   0.929  0.35316
## V89          0.290086   2.070916   0.140  0.88864
## V90         -3.633582   2.702141  -1.345  0.17916
## V91         -0.173392   2.381239  -0.073  0.94197
## V92          3.731865   6.275424   0.595  0.55225
## V93          0.009575   1.970768   0.005  0.99612
## V94         -1.840717   2.820104  -0.653  0.51416
## V95         -3.190036   2.095833  -1.522  0.12844
## V96         -0.463682   2.273817  -0.204  0.83847
## V97          0.440872   4.282425   0.103  0.91803
## V98          0.837597   2.000288   0.419  0.67554
## V99         -0.080477   3.069807  -0.026  0.97909
## V100        -2.988035   2.448997  -1.220  0.22284
## V101         2.035410   3.413008   0.596  0.55112
## V102         4.335541   2.597115   1.669  0.09549 .
## V103        -0.935374   1.662980  -0.562  0.57398
## V104         0.961077   2.571799   0.374  0.70874
## V105        -0.346836   1.901066  -0.182  0.85529
## V106        -0.749740   6.669970  -0.112  0.91053
## V107        -3.070646   4.081667  -0.752  0.45212
## V108        -4.344954   1.488466  -2.919  0.00362 **
## V109        -0.735741   2.050528  -0.359  0.71985
## V110        -1.248119   2.214298  -0.564  0.57316
## V111         5.140838   5.777060   0.890  0.37384
## V112         0.109820   2.159888   0.051  0.95946
## V113        -1.605397   2.860955  -0.561  0.57488
## V114        -0.223579   1.566308  -0.143  0.88653
## V115         4.346832   2.268017   1.917  0.05570 .
## V116         5.241648   2.332630   2.247  0.02494 *
## V117         0.650741   2.617017   0.249  0.80370
## V118        -0.825011   3.060201  -0.270  0.78755
## V119        -0.431792   2.363876  -0.183  0.85512
## V120         0.954559   2.666351   0.358  0.72045
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.279 on 699 degrees of freedom
## Multiple R-squared:  0.2881, Adjusted R-squared:  0.1658
## F-statistic: 2.357 on 120 and 699 DF,  p-value: 5.935e-12

##              actuals predicteds
## actuals    1.0000000  0.4912312
## predicteds 0.4912312  1.0000000
```

```
mape <- mean(abs((RealPredsRE$predicteds - RealPredsRE$actuals))/RealPredsRE$
actuals)
mape*100
```

```
## [1] 9.093%
```

The model obtained using regression analysis contains 12 variables with p-values lower than 0.05for statistical significance, intercept included. While the r-squared has a value of 0.2881, the adjusted r-squared has a lower value of 0.1658. Based on these results, the model performed predictions with a 9.1 percent of accuracy. Now is time to validate how the model will perform in practice using 10-fold cross-validation.

```
## Linear Regression
##
## 820 samples
## 120 predictors
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 737, 738, 738, 738, 739, 739, ...
## Resampling results:
##
##   RMSE      Rsquared    MAE
##   3.574496  0.09225774  2.960404
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

After cross-validate the regression model, there is not much difference in terms of the behavior of the model. The r-square measure is for both, the cross-validated and the no cross-validated is around 9 percent.

Model 7: Random Forest with Ranked Predictors – Age

Maintaining the response variable as continuous is possible apply Random Forest to the same set of ranked predictor and evaluate the performance improvement.

```
## Random Forest
##
## 820 samples
## 120 predictors
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 739, 739, 737, 737, 737, 740, ...
```

```
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   3.432033  0.0975931  2.842126
##
## Tuning parameter 'mtry' was held constant at a value of 11
```

The Random Forest with constant mtry got similar results to the previous model. The r-squared value is also close to 9 percent.

Model 8: Random Forest with Ranked Predictors – Age as Categorical

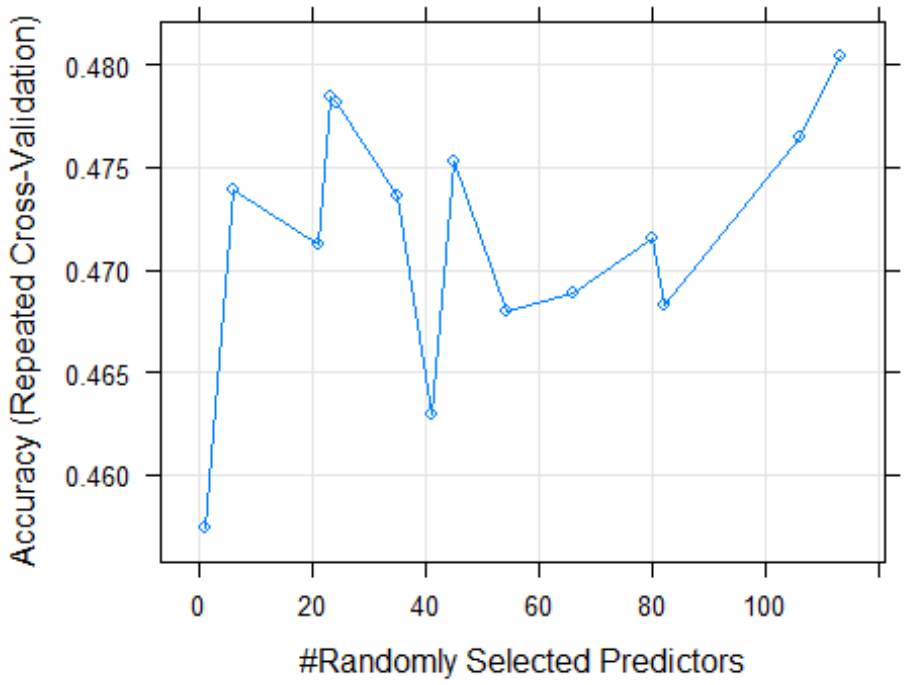
Initially the ordinal response variable was transformed to numerical type as a way to avoid losing order information. This process was done taking the mid-point of the range of every category. Because the model did not perform well, the same statistical technique will be apply over the same set of values but using a classification perspective. Random Forest allows to perform models for both, prediction and classification cases. The first configuration will be maintaining mtry constant over the whole procedure.

```
## Random Forest
##
## 820 samples
## 120 predictors
##   4 classes: '23.5', '28', '33', '36'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 737, 738, 738, 739, 738, 737, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.4699289  0.1235139
##
## Tuning parameter 'mtry' was held constant at a value of 11
```

The model was able to classified categories with an accuracy of 47 percent. Now lets see if there is a change coming of setting mtry dynamic.

```
## Random Forest
##
## 820 samples
## 120 predictors
##   4 classes: '23.5', '28', '33', '36'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
```

```
## Summary of sample sizes: 737, 739, 737, 737, 737, 739, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##     1   0.4573678  0.09110569
##     6   0.4739267  0.12944764
##    21   0.4712190  0.12935359
##    23   0.4784931  0.14028927
##    24   0.4781756  0.14082136
##    35   0.4735987  0.13380489
##    41   0.4629691  0.11774828
##    45   0.4752790  0.13937075
##    54   0.4679813  0.12664317
##    66   0.4688200  0.13006895
##    80   0.4715458  0.13493838
##    82   0.4683036  0.12999767
##   106   0.4764338  0.14336838
##   113   0.4804694  0.15063575
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 113.
```



Configuring mtry value as dynamic, its shows that the best configuration for this set of ranked predictors correspond to the best 113, where the accuracy rate ends up at 48.04 percent.

Accuracy Assessment and Recommendations

Having applied four different statistical methods[18] to classify/predict two relevant subject's traits, is possible to make assessments on how these models performed based on the accuracy rate obtained with each method. For contrast purposes, all models were performed using the same set of ranked predictors, which makes possible to determine the best choice using similar amount of computational resources. The following table shows a summary of the accuracy measurement for each technique at every level of optimization.

Model Configuration					Accuracy			
Predictors	Response	Statistical Technique	Details	AIC/R-Sq	Standard	Optimal/Mtry	CV	BesttPred
1	Gender	Logistic Regression	Best Predictor	1019.70	64.47	66.67	66.18	Females
120	Gender	AIC Logistic Regression	Ranked Predictors (40 Pred. left)	616.79	85.71	86.20	85.59	Males
120	Gender	Logistic Regression	Ranked Predictors	699.53	87.79	88.64	87.67	Females
120	Gender	Linear Discriminant	Ranked Predictors	NA	88.28	88.28	49.08	Females
120	Gender	Random Forest	Both cross-validated - Mtry dynamic vs Mtry constant	NA	NA	73.02	72.80	Females
120	Age	Regression		9.10	NA	NA	NA	NA
120	Age	Random Forest	Mtry constant	9.75	NA	NA	NA	NA
120	Age	Random Forest	Mtry constant vs Mtry dynamic	NA	NA	48.40	46.99	Females

[18] Regression, Logistic Regression, Linear Discriminant, Random Forest.

Both prediction and classification analysis will get a different accuracy measurement at each level of the process. The standard level correspond to the training of the model using the entire dataset and using the same values to predict. The second level corresponds to the same standard process but adding an optimization technique to determine the best threshold. The last level represent a cross-validation procedure utilizing the optimal threshold.

The motivation of using cross-validation falls in the fact that when a model is fitted, it usually fit the same training dataset. Without cross-validation, the accuracy measure only tells how the model perform in that specific dataset. The main interest in this case is having an accuracy measure that could represent the correctness of the model for any new dataset of this type. For this reason, the goodness of the model only will be evaluated based on cross-validated results.

The first interesting thing to point out is how good the procedure used to select the best features was. The model with only one predictor and the intercept was capable of predicting accurately 66% of the cases. It should be noted that every feature to be included will be worse than the previous one when they follow the ranked order. This means that at some point adding more features will not be significant for the increment of the accuracy rate.

Selecting 120 ranked predictors to perform each statistical technique, was the optimal point between getting an adequate accuracy rate, manage viable computational times and avoiding irrelevant predictors. The following output represent the top 6 best predictors.

```
head(bestCombs.df)

##       [,1] [,2] [,3]
## [1,] 27.47   11   61
## [2,] 31.78   13   61
## [3,] 32.36   12   14
## [4,] 32.64   11   62
## [5,] 33.24   11   13
## [6,] 33.35    8   66
```

1. Precentral_L	16.	31.	46.	61. Parietal_Inf_L
2. Precentral_R	17.	32.	47.	62. Parietal_Inf_R
3. Frontal_Sup_L	18.	33.	48.	63. SupraMarginal_L
4. Frontal_Sup_R	19.	34.	49.	64. SupraMarginal_R
5. Frontal_Sup_Orb_L	20.	35.	50.	65. Angular_L
6. Frontal_Sup_Orb_R	21.	36.	51.	66. Angular_R
7. Frontal_Mid_L	22.	37.	52.	67. Precuneus_L
8. Frontal_Mid_R	23.	38.	53.	68. Precuneus_R
9. Frontal_Mid_Orb_L	24.	39.	54.	69. Paracentral_Lobule_L
10. Frontal_Mid_Orb_R	25.	40.	55.	70. Paracentral_Lobule_R
11. Frontal_Inf_Oper_L	26.	41.	56.	71. Caudate_L
12. Frontal_Inf_Oper_R	27.	42.	57.	72. Caudate_R
13. Frontal_Inf_Tri_L	28.	43.	58.	73. Putamen_L
14. Frontal_Inf_Tri_R	29.	44.	59.	74. Putamen_R
15. Frontal_Inf_Orb_L	30.	45.	60.	75. Pallidum_L

Looking closely at the top six best region combination, it is noticeable that these low error regions share the same side of brain hemisphere. For example, *Frontal_Inf* and *Parietal_Inf* are in the left hemisphere, while *Frontal_Mid* and *Angular* are in the right side.

Evaluating the models, Linear discriminant technique had a good performance using the optimal prior, but it fell down in the cross-validation procedure going from 88.28 to 49.08 percent accuracy rate. For this reason, this was the first discarded technique of the three used to model gender. Random Forest also performed well using mtry set constant and little bit better when the parameter was dynamic. It went from 72.80 to 73.02 percent accuracy rate. It was the most robust technique, allowing to model gender using over a thousand predictors. The results with more than 200 predictors were not included because they did not affected much the accuracy rate[19]. Although the Random Forest model had a good performance and the best robustness, it was discarded because the last two model outperformed its results.

The statistical technique propose by this investigation, which is the best adequate to classify the subject gender based on functional connectivity, correspond to Logistic Regression. The AIC Logistic Regression model was capable of getting an 85.6 percent accuracy rate. Alternatively, the Logistic Regression model maintaining the entire set of ranked predictor was capable of getting an 87.7 percent accuracy rate. Is interesting to point out that the model with the AIC features was better classifying males, whereas the complete ranked model was better classifying females.

Even though the Logistic Regression technique was not as robust as the Random Forest, it was able to get better accuracy rates after cross-validation. Moreover, because this type of model is based purely on linear relationships, is easier to explain and be implemented by other researchers with low or no expertise in statistical analysis.

Speaking of Age as the second response variable, the first technique, corresponding to Regression analysis, failed trying to capture the pattern to predict the subject's age. This variable was given as an ordinal type level of measurement. The first approach consisted on converting each category to continuous and avoid losing information coming from the order. In the same way, Random Forest were performed using the same specification and also failed, getting an r-squared of 9.75 and 9.10 for the Regression technique.

The results changed for good when the variable was treated as a nominal type with five categories. The Random Forest technique using mtry dynamic was capable of getting 48.80 percent accuracy rate. Any set of predictors between 200 and 1600 where presenting similar rates of accuracy.

[19] Around 1 percent better, but 5 times slower on the computational side.

Bibliographic Reference

Vul E, Harris C, Winkielman P, Pashler H. Puzzlingly high correlations in fMRI studies of emotion, personality, and social cognition. Perspect. Psychol. Sci. 2009.

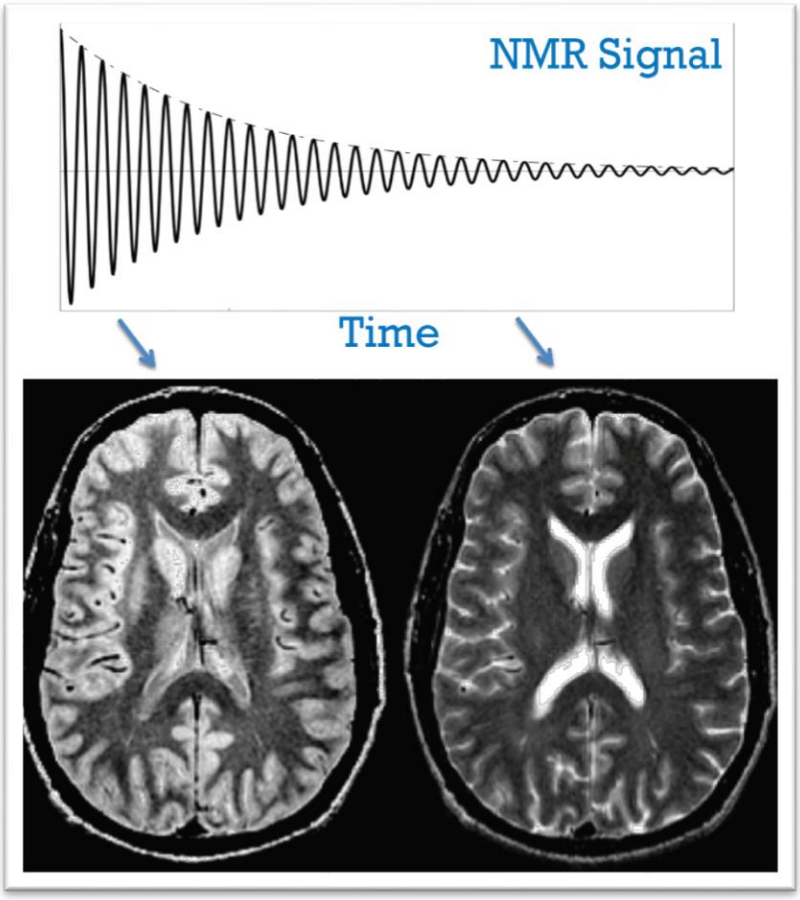
Finn ES, et al. Functional connectome fingerprinting: identifying individuals using patterns of brain connectivity. Nat Neurosci. 2015.

Van Essen DC, et al. The WU-Minn human connectome project: an overview. Neuroimage. 2013.

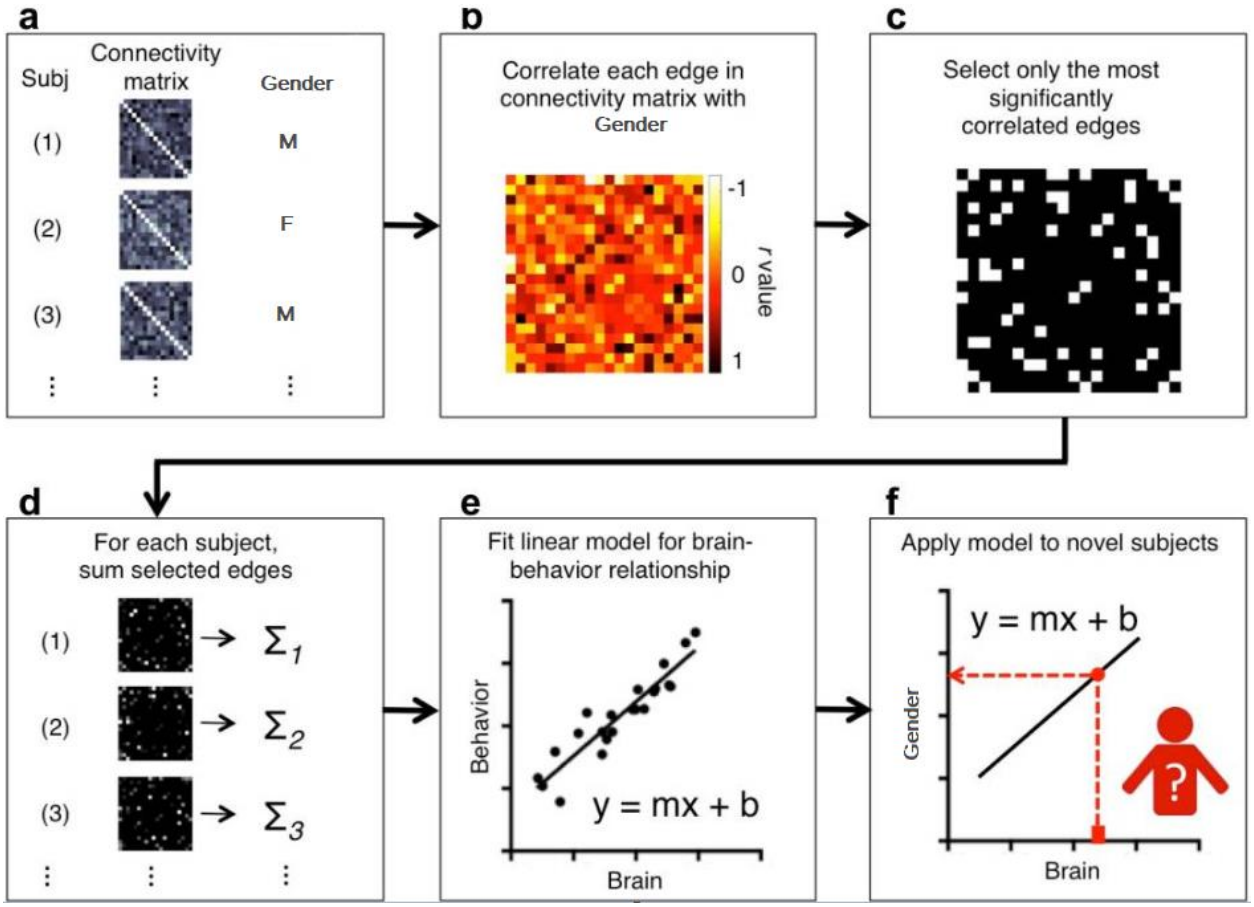
Gabrieli JD, Ghosh SS, Whitfield-Gabrieli S. Prediction as a humanitarian and pragmatic contribution from human cognitive neuroscience. Neuron. 2015.

Appendix

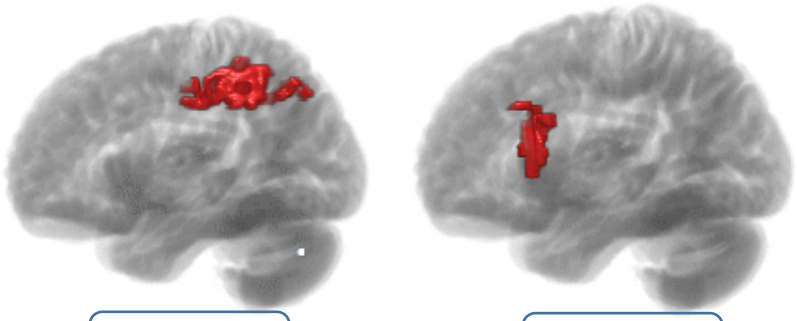
Brain Activity Measured by fMRI (Oxygen Levels)



Feature Selection Procedure



Best Region Combination



11

61

R Code

Reading in Data and Loading Packages

```
#################################### Reading Data ##########################
#################
#Loading Packages

library(memisc)#Reordering features
library(lme4)#Fitting linear mixed-effects model
library(nortest)#Testing normality
library(MASS)#Linear Discriminant Analysis
library(randomForest)#RF Decision trees
library(DAAG)# k-fold CV for continuos
library(caret)#k-fold CV linear regression
library(e1071)


#Reading in the Data
#Setting Working Directory
thesisPath<-"D:\\GuenadieNibbs\\Classes\\AppliedStatisticsMS\\Thesis\\Data\\"
setwd(thesisPath)
getwd()

#Loading Brain Activity data from file#
load("Scans.arr") #27.87 seg elapsed
#Array dimensions (Whole dataset)
dim(Scans.arr)
#Printing subset to show array structure (5 Nodes, 3 Regions, 2 Scans, 2 Subj
ects)
Scans.arr[1:5,1:3,1:2,1:2]

#Reordering array by subject's ID
Scans.arr<-reorder(Scans.arr,dim=4, indices = 1:820) #4.19 seg elapsed

#Loading Subjects characteristics
characteristics.df<-read.csv("BrainsubjectsCopy.csv", header = T, stringsAsFa
ctors = F)
dim(characteristics.df)
sample(characteristics.df,10)
```

Exploratory:

Data Sampling Parameters

```
#################################### DataSet Breakdown ####################
#################
#Creating a function to calculate correlations on pre-specified number of reg
ions.

#Defining Parameters#
```

```
#Brain Activity -> min=1 max=1200
inibac<-1
endbac<-1200
bac<-c(inibac:endbac)
#Regions -> min=1 max=116
iniReg<-1
endreg<-116
#regions<-c(1,2) -> To gather not sequential regions.
regions<-c(iniReg:endreg)
#MRIs -> min=1 max=4
inimri<-1
endmri<-4
mris<-c(inimri:endmri)
#Patients -> min=1 max=820
inipat<-1
endpat<-820
patiens<-c(inipat:endpat)

#Lengths#
bl<-length(bac)
rl<-length(regions)
ml<-length(mris)
pl<-length(patiens)

exp.reg<-(((rl*(rl+1))/2)-rl)#Amount of corrs within 1 scan
exp.scan<-(((rl*(rl+1))/2)-rl)*ml #Amount of corrs within ml scans (max. 4) a
nd 1 subject
exp.pat<-(((rl*(rl+1))/2)-rl)*ml*pl #Total amount of correlations within pls
(max. 820)
```

Features Selection:

Variance Components.

```
######################################## Variance Components ##################
########################
#Correlation Function with VC features (2 columns output: corr|Sub)
corr.fun.Esp<-function(a,b,c,d){
  scans.subset<-Scans.arr[a,b,c,d]
  corr.arr<-apply(scans.subset,c(3,4), cor)
  corr.vect<-as.vector(corr.arr[corr.arr[1:(2^2)]!=1,,])#Reg fixed at 2 becau
se they enter 2 at a time always.
  corr.mat<-as.matrix(unique(corr.vect))
  pat.sec<-rep(1:pl, each=4)# Fixed to 4
  corr.mat<-do.call(cbind,list(corr.mat,pat.sec,deparse.level = 0))
  return (corr.mat)
}

#Creating dataframe to host all region combinations C(116,2)
regComb<-expand.grid(x = 1:rl, y = 1:rl)#Creating all possible combinations o
```

```
f Regions
regComb<-unique(t(apply(regComb, 1, sort)))#Removing duplicates pairs
regComb<-regComb[regComb[,1]!=regComb[,2], , drop = F]#Removing X=Y

varComp.mat<-matrix(NA,nrow(regComb),1)#Creating empty matrix to be loaded
nrow(varComp.mat) == (((rl*(rl+1))/2)-rl)


#Calculating all variance component for the specified subset of regions #2244
.85 seg (37.4 min) elapsed
for(i in 1:nrow(regComb)){
  regs<-c(regComb[i,1],regComb[i,2])#Getting two regions at a time
  corrs<-corr.fun.Esp(bac,regs,mris,patiens)#Getting correlations for 1 combi
nation (3280)
  study.mod<-lmer(corrs[,1] ~ (1|corrs[,2]), data = as.data.frame(corrs)) #Ra
ndom Effects Model
  var.comp<-as.data.frame(VarCorr(study.mod),deparse.level = 0)[,c(1,4)]
  error.perct<-round((var.comp[2,2]/sum(var.comp[,2]))*100,2)
  varComp.mat[i,]<-error.perct
}

#Creating Error Percentage Matrix
errorPerct.mat<-matrix(NA,rl,rl)#Empty Matrix

#Feeling the matrix in the right order
n=1
for(i in 1:rl){

  for(j in 1:rl){
    if (i==j) {
      errorPerct.mat[i,j]<-NA
    } else if (j<i){
      #Nothing
    } else {
      errorPerct.mat[i,j]<-varComp.mat[n,1]
      errorPerct.mat[j,i]<-varComp.mat[n,1]
      n=n+1
    }}}

class(errorPerct.mat) <- "numeric" #Changin to numeric
errorPerct.mat[is.na(errorPerct.mat)]<-100 #Changing NAs with 100
errorPerct.mat[1:5,1:5] #Matrix Preview
#Plotting variability(Error%) Matrix
par(mar=c(1,1,1,1))
image(errorPerct.mat)
```

Criteria - Selection Cases.

```
##################################### Criteria Selection Cases #############
############################
```

```r
bestCombs.df<-cbind(varComp.mat,regComb)#Matching Error% with its region comb
ination
bestCombs.df<-bestCombs.df[order(bestCombs.df[,1]),]
dim(bestCombs.df)
head(bestCombs.df)

#Extracting lows t Error% reg combinations - Q1 and MIN as reference
#Getting Q1 and MIN
fiveMeasures<-summary(varComp.mat)
Q1<-as.numeric(sub('.*:', '', fiveMeasures[2]))# First quartile
min<-as.numeric(sub('.*:', '', fiveMeasures[1]))# Minimun value

############################################################ Case 1 - Lowest E
rror% (Min)
bestCombsMin.df<-subset(bestCombs.df,bestCombs.df[,1] == min)
head(bestCombsMin.df)
regCombLowest<-bestCombsMin.df[,2:3,drop = FALSE]
dim(regCombLowest)

#Calculating correlations for the lowest Error%
corrsMin<-matrix(NA,1,2)#Creating a matrix to be filled
for(i in 1:nrow(regCombLowest)){
  regs<-c(regCombLowest[i,1],regCombLowest[i,2])#Getting two regions at a tim
e
  corrs<-corr.fun.Esp(bac,regs,mris,patiens)
  corrsMin<-do.call(rbind,list(corrsMin,corrs))
}

corrsMin<-corrsMin[2:nrow(corrsMin),]#Removing first NAs row.
ID.names<-dimnames(Scans.arr)[[4]] #Extrating subjects ID
exp.scanTwo<-(((2*(2+1))/2)-2)*ml #Amount of corrs within ml scans (max. 4) a
nd 1 subject

#Generating Subjects ID's secuence to match correlations
pat.secMin<-rep(as.numeric(ID.names), each=exp.scanTwo, time=1)#Repeating eac
h ID as much as Corrs subjects exist
corrs.IDMin<-cbind(corrsMin,pat.secMin)#Binding with corr matrix
dim(corrs.IDMin)
head(corrs.IDMin,12)

#Summarizing correlations by Subjects.
corr.avgMin<-as.matrix(aggregate(corrs.IDMin[,1], list(corrs.IDMin[,3]), FUN
= function(avg){sqrt(mean(avg^2))}))
dim(corr.avgMin)
head(corr.avgMin,12)

#Matching characteristics with subjects
corr.charMin<-merge(corr.avgMin,characteristics.df, by.x="Group.1", by.y="Sub
ject")
corr.charMin<-subset(corr.charMin, Gender != "U")
```

```r
dim(corr.charMin)#Just one case categorized as "U".
head(corr.charMin,12)

################## Checking Normality ####################
#Density Plot
hist(corr.avgMin[,2],probability=T, main="Histogram of normal
     data",xlab="Approximately normally distributed data")
lines(density(corr.avgMin[,2]),col=2)

#QQ Norm Plot
qqnorm(corr.avgMin[,2]);qqline(corr.avgMin[,2], col = 2)

#Normality Test
lillie.test(corr.avgMin[,2]) #Kolmogorov-Smirnov based


########################################################## Case 2 - Error% lowe
r than Q1
bestCombs.df<-subset(bestCombs.df,bestCombs.df[,1] < Q1)
head(bestCombs.df,12)
regCombLow<-bestCombs.df[,2:3,drop = FALSE]
dim(regCombLow)

#Calculating correlations for reg. combs. with the low Error%  #563.04 seg (9
.38 min)
corrsComp<-matrix(NA,1,2)#Creating a matrix to be filled
for(i in 1:nrow(regCombLow)){
  regs<-c(regCombLow[i,1],regCombLow[i,2])#Getting two regions at a time
  corrs<-corr.fun.Esp(bac,regs,mris,patiens)
  corrsComp<-do.call(rbind,list(corrsComp,corrs))
}

corrsComp<-corrsComp[2:nrow(corrsComp),]#Removing first NAs row.
head(corrsComp,12)

pat.sec<-rep(as.numeric(ID.names), each=exp.scanTwo, time=1666)#Repeating eac
h ID as much as Corrs subjects exist
corrs.ID<-cbind(corrsComp,pat.sec)#Binding with corr matrix
dim(corrs.ID)
head(corrs.ID,12)

#Correlations averaged by subject
corr.avg<-as.matrix(aggregate(corrs.ID[,1], list(corrs.ID[,3]), FUN = functio
n(avg){sqrt(mean(avg^2))}))
head(corr.avg,12)

#Matching characteristics with subjects
corr.char<-merge(corr.avg,characteristics.df, by.x="Group.1", by.y="Subject")
corr.char<-subset(corr.char, Gender != "U")
dim(corr.char)
```

```
head(corr.char,12)

################## Checking Normality ####################
#Density Plot
hist(corr.avg[,2],probability=T, main="Histogram of normal
     data",xlab="Approximately normally distributed data")
lines(density(corr.avg[,2]),col=2)

#QQ Norm Plot
qqnorm(corr.avg[,2]);qqline(corr.avg[,2], col = 2)

#Normality Test
lillie.test(corr.avg[,2]) #Kolmogorov-Smirnov based

#Correlation Function for Features Selection
corr.fun.EspX<-function(a,b,c,d){
  scans.subset<-Scans.arr[a,b,c,d]
  corr.arr<-apply(scans.subset,c(3,4), cor)
  corr.vect<-as.vector(corr.arr[corr.arr[1:(2^2)]!=1,,])#Reg fixed at 2 becau
se they enter 2 at a time always.
  corr.matX<-as.matrix(unique(corr.vect))
  return (corr.matX)
}
```

Best Model Selection:

Model 1: Logistic Regression AIC with 120 Predictors and CV - Gender.

```
################################ AIC with 120 Features
regCombLowAIC<-regCombLow[1:120,]#Number of reg combinations
dim(regCombLowAIC)

#Calculating correlation for 200 combinations with AIC  #563.04 seg (9.38 min
)
corrsAIC<-matrix(NA,3280,1)#Creating a matrix to be filled
for(i in 1:nrow(regCombLowAIC)){
  regs<-c(regCombLowAIC[i,1],regCombLowAIC[i,2])#Getting two regions at a tim
e
  corrs<-corr.fun.EspX(bac,regs,mris,patiens)
  corrsAIC<-do.call(cbind,list(corrsAIC,corrs))
}

corrsAIC<-corrsAIC[,2:ncol(corrsAIC)]#Removing first NAs column.
dim(corrsAIC)

pat.secAIC<-rep(as.numeric(ID.names), each=exp.scanTwo, time=1)#Repeating eac
h ID as much as Corrs subjects exist
corrs.ID.AIC<-cbind(corrsAIC,pat.secAIC)#Binding with corr matrix
dim(corrs.ID.AIC)

#Correlations averaged by subject
```

```r
corr.avgAIC<-as.matrix(aggregate(corrs.ID.AIC[,1:(ncol(corrs.ID.AIC))],
                                 list(corrs.ID.AIC[,ncol(corrs.ID.AIC)]), FUN
= function(avg){sqrt(mean(avg^2))}))
dim(corr.avgAIC)

#Matching characteristics with subjects
corr.charAIC<-merge(corr.avgAIC,characteristics.df[,1:2], by.x="Group.1", by.
y="Subject")
corr.charAIC<-subset(corr.charAIC, Gender != "U")
corr.charAIC<-corr.charAIC[,2:ncol(corr.charAIC)]
corr.charAIC<- corr.charAIC[ , -which(names(corr.charAIC) %in% c("pat.secAIC"
))]
dim(corr.charAIC)#Just one case categorized as "U".
corr.charAIC[1:12,ncol(corr.charAIC):(ncol(corr.charAIC)-12)]

#Model Selection Using AIC and 150 Features
#Model Boundaries
lo<-glm(as.factor(Gender) ~ 1, family=binomial(link="logit"), data = corr.cha
rAIC)#Intercept Model
hi<-glm(as.factor(Gender) ~ ., family=binomial(link="logit"), data = corr.cha
rAIC)#Full Model

corrset.aic<-stepAIC(lo,direction='both',scope=list(upper=hi,lower=lo),trace
= 0)
#GLM results
summary(corrset.aic)

#Predicted probabilities
CorrHat.AIC<-fitted(corrset.aic)

#Choose a threshold for dichotomizing according to predicted probability
thresh.AIC<- nrow(corr.charAIC[corr.charAIC=="M",]) / nrow(corr.charAIC)#0.5
CorrHatFac.AIC <- cut(CorrHat.AIC, breaks=c(-Inf, thresh.AIC, Inf), labels=c(
"F", "M"))

#Contingency table and marginal sums
cTab.AIC <- table(corrset.aic$data$Gender, CorrHatFac.AIC)
addmargins(cTab.AIC)

#Accuracy Percentage
round(sum(diag(cTab.AIC)) / sum(cTab.AIC)*100 , 2)

################ Calculating Best Treshold ####################
Opt.error.rate.f <- function(Truth=(corr.charAIC$Gender=="M"),
                             Pred=corrset.aic$fitted.values) {
  # Truth is logical; # Pred is the probability
  # Cover rejection rates from 0% to 100%
  Threshold.v <- sort(Pred[Truth])
  k <- length(Threshold.v)
  Error.mat <- matrix(0,k,3)
```

```
  for (i in 1:k) {
    prediction <- (Pred>=Threshold.v[i])
    Classif.table <- table(prediction,Truth)
    Error.mat[i,1] <- Classif.table[1,2]
    Error.mat[i,2] <- Classif.table[2,1]
  }
  Error.mat[,3] <- apply(Error.mat[,1:2],1,sum)
  list(Error=Error.mat/length(Truth),Threshold=Threshold.v)
}

Opt.error.list <-Opt.error.rate.f()

#############################################################
Plot.Error.f <- function(lista=Opt.error.list,limit=1) {
  mat <- lista$Error
  cond <- (mat[,3]<limit)
  mat <- mat[cond,]
  Threshold <- lista$Threshold[cond]
  vect <- c(2,4,1)
  min.val <- min(mat[,3])
  ind <- match(min.val, mat[,3])
  matplot(Threshold, mat, type="l",col=vect, lty=vect,
          main=paste("Optimum Error rate is ",round(min.val,4),"at Threshold
=", round(Threshold[ind],4)),
          ylab="Fraction Misclassified")
  abline(h=min.val, lty=2)
  abline(v=Threshold[ind], lty=2)
  legend(Threshold[ind]+0.04, max(mat[,3]),
         legend =c("Males misclassified",
                   "Females misclassified",
                   "Total Error"),
         col=vect, lty=vect)

  opt.pred<-corrset.aic$fitted.values>=Threshold[ind]
  cat("Misclassification table using optimal threshold")
  mis.tbl<-table(opt.pred,corr.charAIC$Gender)  #retrieving misclassification
table at optimal value of threshold
  print(mis.tbl)

  #Accuracy Percentage
  cat("\n")
  round(sum(diag(mis.tbl)) / sum(mis.tbl)*100 , 2)
}

Plot.Error.f()

######################## Cross-Validation AIC ###################################
###
K<-10
```

```r
error.rate.f <- function(mat){
  # assumes mat is 2 by 2 table
  # Errror rate = 1 - correct classification rate
  1- sum(diag(mat))/sum(mat)
}

CV.error.f <- function(data=corr.charAIC,k=K,t1=0,t2=1,m=100,
                       Rounds=100) {
  # k-fold CV; uses m=100 thresholds from t1 to t2
  Threshold.v <- seq(from=t1, to=t2, length=m)
  Err.arr <- array(0,c(m,Rounds,k))
  for (i in 1:Rounds) {
    fold=sample(rep(1:k,length=nrow(data)))
    for (j in 1:k) {
      cond <- (fold==j)
      obj <- glm(as.factor(Gender) ~ .,
                 data = data[!cond,], family = "binomial")
      Pred <- predict(obj, newdata = data[cond,],
                      type = "response")
      Truth=(data[cond,]$Gender=="M") # Truth is logical
      for (ind in 1:m) {
        prediction <- (Pred>=Threshold.v[ind])
        Err.arr[ind,i,j] <- mean((prediction==F)&(Truth==T)) +
          mean((prediction==T)&(Truth==F)) # not using "table"
      }
    }
  }
  list(Err=Err.arr, Threshold=Threshold.v)
}

Plot.CV.Err.f <- function(lista=CV.error.list,from=0, to=1,
                          Truth=(corr.charAIC$Gender=="M"), Pred=obj$fitted)
{
  Threshold <- lista$Threshold
  cond <- ((Threshold>=from)&(Threshold<=to)) # range
  Threshold <- Threshold[cond]
  Err.arr <- lista$Err[cond,,]
  N <- prod(dim(Err.arr)[2:3]) # Rounds*k
  err <- apply(Err.arr,1,mean)
  err.se <- apply(Err.arr,1,sd)/sqrt(N)
  low <- err - 2*err.se;   high <- err + 2*err.se
  mat <- cbind(err,low,high)
  # now calculating the optimal threshold
  min.val <- min(err)
  ind <- match(min.val, err) # Threshold[ind] is the optimal threshold
  par(mar = c(4.5,4.5,0,1))
  matplot(Threshold,mat,type="l", xlab=paste("Threshold (optimal at ",
                                             round(Threshold[ind],3),")"), yl
ab="Error From Misclassification",lty=c(1,2,2))
  abline(h=min.val, lty=2);
```

```
  abline(v=Threshold[ind], lty=2)
  Threshold[ind] # the optimal threshold in case we need it

  cat(paste0("The optimal threshold for the ",K,"-fold cross validation is: "
,Threshold[ind]))
  gender.prediction <- (corrset.aic$fitted>Threshold[ind])
  (Classif.table.gender <- table(gender.prediction,corr.charAIC$Gender))
  print(Classif.table.gender)
  print(round(sum(diag(Classif.table.gender)) / sum(Classif.table.gender)*100
, 2))

  error.rate.f(Classif.table.gender)
  cat(paste0("The error rate for ",K,"-fold cv is:",error.rate.f(Classif.tabl
e.gender)))

  return(Threshold[ind])
}

CV.error.list<-CV.error.f()
optimal.threshold<-Plot.CV.Err.f()
```

Model 2: Logistic Regression with 1 (The Best) Predictor and CV - Gender.

```
###################################### Logistic Regression With Best Feature
regCombLowest<-regCombLow[1,,drop=0]#Number of reg combinations
dim(regCombLowest)

#Calculating correlation for 200 combinations with AIC  #563.04 seg (9.38 min
)
corrsLowest<-matrix(NA,3280,1)#Creating a matrix to be filled
for(i in 1:nrow(regCombLowest)){
  regs<-c(regCombLowest[i,1],regCombLowest[i,2])#Getting two regions at a tim
e
  corrs<-corr.fun.EspX(bac,regs,mris,patiens)
  corrsLowest<-do.call(cbind,list(corrsLowest,corrs))
}

corrsLowest<-corrsLowest[,2:ncol(corrsLowest)]#Removing first NAs column.
length(corrsLowest)

pat.secLowest<-rep(as.numeric(ID.names), each=exp.scanTwo, time=1)#Repeating
each ID as much as Corrs subjects exist
corrs.ID.Lowest<-cbind(corrsLowest,pat.secLowest)#Binding with corr matrix
dim(corrs.ID.Lowest)

#Correlations averaged by subject
corr.avgLowest<-as.matrix(aggregate(corrs.ID.Lowest[,1:(ncol(corrs.ID.Lowest)
)],
                                    list(corrs.ID.Lowest[,ncol(corrs.ID.Lowest)]
), FUN = function(avg){sqrt(mean(avg^2))}))
```

```r
dim(corr.avgLowest)

#Matching characteristics with subjects
corr.charLowest<-merge(corr.avgLowest,characteristics.df[,1:2], by.x="Group.1
", by.y="Subject")
corr.charLowest<-subset(corr.charLowest, Gender != "U")
corr.charLowest<-corr.charLowest[,2:ncol(corr.charLowest)]
corr.charLowest<- corr.charLowest[ , -which(names(corr.charLowest) %in% c("pa
t.secLowest"))]
dim(corr.charLowest)#Just one case categorized as "U".

corr.Lowest.mod<- glm(as.factor(Gender) ~ corrsLowest, family=binomial(link="
logit"), data = corr.charLowest)
summary(corr.Lowest.mod)

#Predicted probabilities
CorrHat.Lowest<-fitted(corr.Lowest.mod)

#Choose a threshold for dichotomizing according to predicted probability
thresh.lowest<- nrow(corr.charLowest[corr.charLowest=="M",]) / nrow(corr.char
Lowest)#0.5
CorrHatFac.Lowest <- cut(CorrHat.Lowest, breaks=c(-Inf, thresh.lowest, Inf),
labels=c("F", "M"))

#Contingency table and marginal sums
cTab.Lowest <- table(corr.Lowest.mod$data$Gender, CorrHatFac.Lowest)
addmargins(cTab.Lowest)

#Accuracy Percentage
round(sum(diag(cTab.Lowest)) / sum(cTab.Lowest)*100 , 2)

################ Calculating Best Treshold ####################
Opt.error.rate.f <- function(Truth=(corr.charLowest$Gender=="M"),
                             Pred=corr.Lowest.mod$fitted.values) {
  # Truth is logical; # Pred is the probability
  # Cover rejection rates from 0% to 100%
  Threshold.v <- sort(Pred[Truth])
  k <- length(Threshold.v)
  Error.mat <- matrix(0,k,3)
  for (i in 1:k) {
    prediction <- (Pred>=Threshold.v[i])
    Classif.table <- table(prediction,Truth)
    Error.mat[i,1] <- Classif.table[1,2]
    Error.mat[i,2] <- Classif.table[2,1]
  }
  Error.mat[,3] <- apply(Error.mat[,1:2],1,sum)
  list(Error=Error.mat/length(Truth),Threshold=Threshold.v)
}

Opt.error.list <-Opt.error.rate.f()
```

```r
#############################################################
Plot.Error.f <- function(lista=Opt.error.list,limit=1) {
  mat <- lista$Error
  cond <- (mat[,3]<limit)
  mat <- mat[cond,]
  Threshold <- lista$Threshold[cond]
  vect <- c(2,4,1)
  min.val <- min(mat[,3])
  ind <- match(min.val, mat[,3])
  matplot(Threshold, mat, type="l",col=vect, lty=vect,
          main=paste("Optimum Error rate is ",round(min.val,4),"at Threshold
=", round(Threshold[ind],4)),
          ylab="Fraction Misclassified")
  abline(h=min.val, lty=2)
  abline(v=Threshold[ind], lty=2)
  legend(Threshold[ind]+0.04, max(mat[,3]),
         legend =c("Males misclassified",
                   "Females misclassified",
                   "Total Error"),
         col=vect, lty=vect)

  opt.pred<-corr.Lowest.mod$fitted.values>=Threshold[ind]
  cat("Misclassification table using optimal threshold")
  mis.tbl<-table(opt.pred,corr.charLowest$Gender)  #retrieving misclassificat
ion table at optimal value of threshold
  print(mis.tbl)

  #Accuracy Percentage
  cat("\n")
  round(sum(diag(mis.tbl)) / sum(mis.tbl)*100 , 2)
}

Plot.Error.f()

######################## Cross-Validation Lowest ##############################
######
K<-10

CV.error.f <- function(data=corr.charLowest,k=K,t1=0,t2=1,m=100,
                       Rounds=100) {
  # k-fold CV; uses m=100 thresholds from t1 to t2
  Threshold.v <- seq(from=t1, to=t2, length=m)
  Err.arr <- array(0,c(m,Rounds,k))
  for (i in 1:Rounds) {
    fold=sample(rep(1:k,length=nrow(data)))
    for (j in 1:k) {
      cond <- (fold==j)
      obj <- glm(as.factor(Gender) ~ .,
                 data = data[!cond,], family = "binomial")
```

```
      Pred <- predict(obj, newdata = data[cond,],
                      type = "response")
      Truth=(data[cond,]$Gender=="M") # Truth is logical
      for (ind in 1:m) {
        prediction <- (Pred>=Threshold.v[ind])
        Err.arr[ind,i,j] <- mean((prediction==F)&(Truth==T)) +
          mean((prediction==T)&(Truth==F)) # not using "table"
      }
    }
  }
  list(Err=Err.arr, Threshold=Threshold.v)
}

Plot.CV.Err.f <- function(lista=CV.error.list,from=0, to=1,
                          Truth=(corr.charLowest$Gender=="M"), Pred=obj$fitte
d) {
  Threshold <- lista$Threshold
  cond <- ((Threshold>=from)&(Threshold<=to)) # range
  Threshold <- Threshold[cond]
  Err.arr <- lista$Err[cond,,]
  N <- prod(dim(Err.arr)[2:3]) # Rounds*k
  err <- apply(Err.arr,1,mean)
  err.se <- apply(Err.arr,1,sd)/sqrt(N)
  low <- err - 2*err.se;   high <- err + 2*err.se
  mat <- cbind(err,low,high)
  # now calculating the optimal threshold
  min.val <- min(err)
  ind <- match(min.val, err) # Threshold[ind] is the optimal threshold
  par(mar = c(4.5,4.5,0,1))
  matplot(Threshold,mat,type="l", xlab=paste("Threshold (optimal at ",
                                             round(Threshold[ind],3),")"), yl
ab="Error From Misclassification",lty=c(1,2,2))
  abline(h=min.val, lty=2);
  abline(v=Threshold[ind], lty=2)
  Threshold[ind] # the optimal threshold in case we need it

  cat(paste0("The optimal threshold for the ",K,"-fold cross validation is: "
,Threshold[ind]))
  gender.prediction <- (corr.Lowest.mod$fitted>Threshold[ind])
  (Classif.table.gender <- table(gender.prediction,corr.charLowest$Gender))
  print(Classif.table.gender)
  print(round(sum(diag(Classif.table.gender)) / sum(Classif.table.gender)*100
, 2))

  error.rate.f(Classif.table.gender)
  cat(paste0("The error rate for ",K,"-fold cv is:",error.rate.f(Classif.tabl
e.gender)))

  return(Threshold[ind])
}
```

```
CV.error.list<-CV.error.f()
optimal.threshold<-Plot.CV.Err.f()
```

Model 3: Logistic Regression with 120 (Best) Predictors and CV - Gender.

```
##################################### Logistic Regression With 120 Features
regCombLR<-regCombLow[1:120,,drop=0]#Number of reg combinations
dim(regCombLR)

#Calculating correlation for 200 combinations with AIC  #563.04 seg (9.38 min
)
corrsLR<-matrix(NA,3280,1)#Creating a matrix to be filled
for(i in 1:nrow(regCombLR)){
  regs<-c(regCombLR[i,1],regCombLR[i,2])#Getting two regions at a time
  corrs<-corr.fun.EspX(bac,regs,mris,patiens)
  corrsLR<-do.call(cbind,list(corrsLR,corrs))
}

corrsLR<-corrsLR[,2:ncol(corrsLR)]#Removing first NAs column.
dim(corrsLR)

pat.secLR<-rep(as.numeric(ID.names), each=exp.scanTwo, time=1)#Repeating each
ID as much as Corrs subjects exist
corrs.ID.LR<-cbind(corrsLR,pat.secLR)#Binding with corr matrix
dim(corrs.ID.LR)

#Correlations averaged by subject
corr.avgLR<-as.matrix(aggregate(corrs.ID.LR[,1:(ncol(corrs.ID.LR))],
                                list(corrs.ID.LR[,ncol(corrs.ID.LR)]), FU
N = function(avg){sqrt(mean(avg^2))}))
dim(corr.avgLR)

#Matching characteristics with subjects
corr.charLR<-merge(corr.avgLR,characteristics.df[,1:2], by.x="Group.1", by.y=
"Subject")
corr.charLR<-subset(corr.charLR, Gender != "U")
corr.charLR<-corr.charLR[,2:ncol(corr.charLR)]
corr.charLR<- corr.charLR[ , -which(names(corr.charLR) %in% c("pat.secLR"))]
dim(corr.charLR)#Just one case categorized as "U".

corr.LR.mod<- glm(as.factor(Gender) ~ ., family=binomial(link="logit"), data
= corr.charLR)
summary(corr.LR.mod)

#Predicted probabilities
CorrHat.LR<-fitted(corr.LR.mod)

#Choose a threshold for dichotomizing according to predicted probability
thresh.LR<- nrow(corr.charLR[corr.charLR=="M",]) / nrow(corr.charLR)#0.5
```

```r
CorrHatFac.LR <- cut(CorrHat.LR, breaks=c(-Inf, thresh.LR, Inf), labels=c("F"
, "M"))

#Contingency table and marginal sums
cTab.LR <- table(corr.LR.mod$data$Gender, CorrHatFac.LR)
addmargins(cTab.LR)

#Accuracy Percentage
round(sum(diag(cTab.LR)) / sum(cTab.LR)*100 , 2)

################ Calculating Best Treshold ####################
Opt.error.rate.f <- function(Truth=(corr.charLR$Gender=="M"),
                             Pred=corr.LR.mod$fitted.values) {
  # Truth is logical; # Pred is the probability
  # Cover rejection rates from 0% to 100%
  Threshold.v <- sort(Pred[Truth])
  k <- length(Threshold.v)
  Error.mat <- matrix(0,k,3)
  for (i in 1:k) {
    prediction <- (Pred>=Threshold.v[i])
    Classif.table <- table(prediction,Truth)
    Error.mat[i,1] <- Classif.table[1,2]
    Error.mat[i,2] <- Classif.table[2,1]
  }
  Error.mat[,3] <- apply(Error.mat[,1:2],1,sum)
  list(Error=Error.mat/length(Truth),Threshold=Threshold.v)
}

Opt.error.list <-Opt.error.rate.f()

#################################################################
Plot.Error.f <- function(lista=Opt.error.list,limit=1) {
  mat <- lista$Error
  cond <- (mat[,3]<limit)
  mat <- mat[cond,]
  Threshold <- lista$Threshold[cond]
  vect <- c(2,4,1)
  min.val <- min(mat[,3])
  ind <- match(min.val, mat[,3])
  matplot(Threshold, mat, type="l",col=vect, lty=vect,
          main=paste("Optimum Error rate is ",round(min.val,4),"at Threshold
=", round(Threshold[ind],4)),
          ylab="Fraction Misclassified")
  abline(h=min.val, lty=2)
  abline(v=Threshold[ind], lty=2)
  legend(Threshold[ind]+0.04, max(mat[,3]),
         legend =c("Males misclassified",
                   "Females misclassified",
                   "Total Error"),
         col=vect, lty=vect)
```

```r
  opt.pred<-corr.LR.mod$fitted.values>=Threshold[ind]
  cat("Misclassification table using optimal threshold")
  mis.tbl<-table(opt.pred,corr.charLR$Gender)  #retrieving misclassification
table at optimal value of threshold
  print(mis.tbl)

  #Accuracy Percentage
  cat("\n")
  round(sum(diag(mis.tbl)) / sum(mis.tbl)*100 , 2)
}

Plot.Error.f()

######################## Cross-Validation LR ###################################
##
K<-10

CV.error.f <- function(data=corr.charLR,k=K,t1=0,t2=1,m=100,
                       Rounds=100) {
  # k-fold CV; uses m=100 thresholds from t1 to t2
  Threshold.v <- seq(from=t1, to=t2, length=m)
  Err.arr <- array(0,c(m,Rounds,k))
  for (i in 1:Rounds) {
    fold=sample(rep(1:k,length=nrow(data)))
    for (j in 1:k) {
      cond <- (fold==j)
      obj <- glm(as.factor(Gender) ~ .,
                 data = data[!cond,], family = "binomial")
      Pred <- predict(obj, newdata = data[cond,],
                      type = "response")
      Truth=(data[cond,]$Gender=="M") # Truth is logical
      for (ind in 1:m) {
        prediction <- (Pred>=Threshold.v[ind])
        Err.arr[ind,i,j] <- mean((prediction==F)&(Truth==T)) +
          mean((prediction==T)&(Truth==F)) # not using "table"
      }
    }
  }
  list(Err=Err.arr, Threshold=Threshold.v)
}

Plot.CV.Err.f <- function(lista=CV.error.list,from=0, to=1,
                          Truth=(corr.charLR$Gender=="M"), Pred=obj$fitted) {
  Threshold <- lista$Threshold
  cond <- ((Threshold>=from)&(Threshold<=to)) # range
  Threshold <- Threshold[cond]
  Err.arr <- lista$Err[cond,,]
  N <- prod(dim(Err.arr)[2:3]) # Rounds*k
  err <- apply(Err.arr,1,mean)
```

```
  err.se <- apply(Err.arr,1,sd)/sqrt(N)
  low <- err - 2*err.se;   high <- err + 2*err.se
  mat <- cbind(err,low,high)
  # now calculating the optimal threshold
  min.val <- min(err)
  ind <- match(min.val, err) # Threshold[ind] is the optimal threshold
  par(mar = c(4.5,4.5,0,1))
  matplot(Threshold,mat,type="l", xlab=paste("Threshold (optimal at ",
                                             round(Threshold[ind],3),")"), yl
ab="Error From Misclassification",lty=c(1,2,2))
  abline(h=min.val, lty=2);
  abline(v=Threshold[ind], lty=2)
  Threshold[ind] # the optimal threshold in case we need it

  cat(paste0("The optimal threshold for the ",K,"-fold cross validation is: "
,Threshold[ind]))
  gender.prediction <- (corr.LR.mod$fitted>Threshold[ind])
  (Classif.table.gender <- table(gender.prediction,corr.charLR$Gender))
  print(Classif.table.gender)
  print(round(sum(diag(Classif.table.gender)) / sum(Classif.table.gender)*100
, 2))

  error.rate.f(Classif.table.gender)
  cat(paste0("The error rate for ",K,"-fold cv is:",error.rate.f(Classif.tabl
e.gender)))

  return(Threshold[ind])
}

CV.error.list<-CV.error.f()
optimal.threshold<-Plot.CV.Err.f()
```

Model 4: Linear Discriminant with 120 (Best) Predictors and CV - Gender.

```
###################################### Linear Discriminant Analysis With Best
Features
regCombLDA<-regCombLow[1:120,,drop=0]#Number of reg combinations
dim(regCombLDA)

#Calculating correlation for 200 combinations with AIC  #563.04 seg (9.38 min
)
corrsLDA<-matrix(NA,3280,1)#Creating a matrix to be filled
for(i in 1:nrow(regCombLDA)){
  regs<-c(regCombLDA[i,1],regCombLDA[i,2])#Getting two regions at a time
  corrs<-corr.fun.EspX(bac,regs,mris,patiens)
  corrsLDA<-do.call(cbind,list(corrsLDA,corrs))
}

corrsLDA<-corrsLDA[,2:ncol(corrsLDA)]#Removing first NAs column.
dim(corrsLDA)
```

```r
pat.secLDA<-rep(as.numeric(ID.names), each=exp.scanTwo, time=1)#Repeating eac
h ID as much as Corrs subjects exist
corrs.ID.LDA<-cbind(corrsLDA,pat.secLDA)#Binding with corr matrix
dim(corrs.ID.LDA)

#Correlations averaged by subject
corr.avgLDA<-as.matrix(aggregate(corrs.ID.LDA[,1:(ncol(corrs.ID.LDA))],
                                 list(corrs.ID.LDA[,ncol(corrs.ID.LDA)]), FUN
= function(avg){sqrt(mean(avg^2))}))
dim(corr.avgLDA)

#Matching characteristics with subjects
corr.charLDA<-merge(corr.avgLDA,characteristics.df[,1:2], by.x="Group.1", by.
y="Subject")
corr.charLDA<-subset(corr.charLDA, Gender != "U")
corr.charLDA<-corr.charLDA[,2:ncol(corr.charLDA)]
corr.charLDA<- corr.charLDA[ , -which(names(corr.charLDA) %in% c("pat.secLDA"
))]
dim(corr.charLDA)#Just one case categorized as "U".

corr.LDA.mod<-lda(as.factor(Gender) ~., data = corr.charLDA)
summary(corr.LDA.mod)

#Predicted probabilities
CorrHat.LDA<-predict(corr.LDA.mod, corr.charLDA)

#Contingency table and marginal sums
cTab.LDA<-table(CorrHat.LDA$class, corr.charLDA$Gender,dnn=c("Predicted","Act
ual"))
cTab.LDA

#Accuracy Percentage
round(sum(diag(cTab.LDA)) / sum(cTab.LDA)*100 , 2)

################ Calculating Best Prior for LDA ####################
prior<-seq(0.01,0.99,.01) #avoided using 0 as prior

error.mat<-matrix(0,nrow = length(prior),ncol = 2)

for(i in 1:99)
{
  obj<-lda(as.factor(Gender) ~., data = corr.charLDA, prior=c(1-prior[i],prio
r[i])) #build model
  Pred.obj <- predict(obj, corr.charLDA) #predict using the model
  missed<-sum(Pred.obj$class != corr.charLDA$Gender) #find the number of misc
lassified observations

  error.mat[i,1] <- missed  #total misclassified
}
```

```r
error.mat[,2]<-error.mat[,1]/n  #misclassification error

error.optimal<-min(error.mat[,2]) #minimum error
optimal<- 1-prior[which(error.mat[,2]==error.optimal)][1]  #optimal p1

#plotting Error rate vs. Prior probability
matplot(1-prior,error.mat[,2],xlab="Prior of Gender",
        ylab="Total Error Rate",pch=18,type="l",main="Error Rates vs. Prior o
n the Male Group",
        sub=paste0("Optimal prior (p1) = ",optimal, ", Error rate at optimal
prior (p1) = ",round(error.optimal,4)))
abline(v=optimal, h=error.optimal,lty=2,col="grey75")

#########################################################################
#Classification Table
lda.new <- lda(as.factor(Gender) ~., data = corr.charLDA, prior=c(optimal,1-o
ptimal))
prediction <- predict(lda.new, corr.charLDA) # doing prediction

#classification table at optimum prior
Classif.table <- table(prediction$class,corr.charLDA$Gender)
Classif.table

#Accuracy Percentage
round(sum(diag(Classif.table)) / sum(Classif.table)*100 , 2)


######################## Cross-Validation LDA ################################
###
CV.error.f.2 <- function(data=corr.charLDA,k=10,Rounds=100) {

  prior<-seq(0.01,0.99,.01) #avoided using 0 as prior
  error.mat<-matrix(0,nrow = length(prior),ncol = 4)

  for(i in 1:99)
  {
    Cost <- matrix(0,Rounds)
    for (j in 1:Rounds)
    {
      fold=sample(rep(1:k,length=nrow(data)))
      for (ind in 1:k)
      {
        cond <- (fold==ind)
        obj <-lda(as.factor(Gender) ~., data = data[!cond,],prior=c(1-prior[i
],prior[i]))
        Pred <- predict(obj, newdata = data[cond,])$class
        Truth=(data[cond,]$Gender==1)

        prediction <- (Pred==1) #Severe predictions
```

```
        count <- sum((prediction==F)&(Truth==T)) +
          sum((prediction==T)&(Truth==F)) # not using "table"

        Cost[j] <- Cost[j] + count #the of the folds are added to give the to
tal cost in a round
      }
    }
    error.mat[i,1] <- mean(Cost/n)  #Average misclassification rate
    sd.err<-sd(Cost/n)/sqrt(length(Cost)) #calculating standard error for ave
rage misclassification rate
    error.mat[i,2] <- error.mat[i,1]-sd.err #lower bound
    error.mat[i,3] <- error.mat[i,1]+sd.err #upper bound
    error.mat[i,4] <- sd.err
  }

  error.optimal<-min(error.mat[,1]) #minimum error
  optimal<- 1-prior[which(error.mat[,1]==error.optimal)][1]  #optimal p1
  std.error<- error.mat[,4][which(error.mat[,1]==error.optimal)][1]
  #plotting Error rate vs. Prior probability
  matplot(1-prior,error.mat[,1:3],xlab="Prior of Gender", col=1:3, lty=c(1,2,
2),
          ylab="Mean CV Error Rate",type="l",main=paste0("Min. error Rates vs
. Prior on the Males Group using ",k,"-fold CV"),
          sub=paste0("Optimal prior (p1) = ",optimal, ", Min. error rate at (
p1) = ",round(error.optimal,4),", std.error = ",round(std.error,4)))

  abline(v=optimal, h=error.optimal,lty=2,col="grey75")
}

CV.error.f.2()

####### CV with normal prior
CV.error.f.1 <- function(data=corr.charLDA,k=10,Rounds=100) {
  Cost <- matrix(0,Rounds)
  for (i in 1:Rounds)
  {
    fold=sample(rep(1:k,length=nrow(data)))
    for (j in 1:k)
    {
      cond <- (fold==j)
      obj <-lda(as.factor(Gender) ~., data = data[!cond,])
      Pred <- predict(obj, newdata = data[cond,])$class
      Truth=(data[cond,]$Gender=="M")

      prediction <- (Pred=="M") #Male predictions

      count <- sum((prediction==F)&(Truth==T)) +
        sum((prediction==T)&(Truth==F)) # not using "table"
```

```
      Cost[i] <- Cost[i] + count #the of the folds are added to give the tota
l cost in a round
    }
  }
  cat("Average misclassified cases using 10 fold CV = ", mean(Cost),"\n")
  cat("Standard error of total minimized error of misclassification = ", sd(C
ost)/sqrt(length(Cost)))
  Cost
}

# Misclassification error of the 100 rounds of 10 fold CV
CV.error <- CV.error.f.1()
```

Model 5: Random Forest with 120 (Best) Predictors and CV - Gender.

```
############################################################## Random Forest
regCombRF<-regCombLow[1:120,,drop=0]#Number of reg combinations
dim(regCombRF)

#Calculating correlation for 200 combinations with AIC  #563.04 seg (9.38 min
)
corrsRF<-matrix(NA,3280,1)#Creating a matrix to be filled
for(i in 1:nrow(regCombRF)){
  regs<-c(regCombRF[i,1],regCombRF[i,2])#Getting two regions at a time
  corrs<-corr.fun.EspX(bac,regs,mris,patiens)
  corrsRF<-do.call(cbind,list(corrsRF,corrs))
}

corrsRF<-corrsRF[,2:ncol(corrsRF)]#Removing first NAs column.
dim(corrsRF)

pat.secRF<-rep(as.numeric(ID.names), each=exp.scanTwo, time=1)#Repeating each
ID as much as Corrs subjects exist
corrs.ID.RF<-cbind(corrsRF,pat.secRF)#Binding with corr matrix
dim(corrs.ID.RF)

#Correlations averaged by subject
corr.avgRF<-as.matrix(aggregate(corrs.ID.RF[,1:(ncol(corrs.ID.RF))],
                                list(corrs.ID.RF[,ncol(corrs.ID.RF)]), FUN =
function(avg){sqrt(mean(avg^2))}))
dim(corr.avgRF)

#Matching characteristics with subjects
corr.charRF<-merge(corr.avgRF,characteristics.df[,1:2], by.x="Group.1", by.y=
"Subject")
corr.charRF<-subset(corr.charRF, Gender != "U")
corr.charRF<-corr.charRF[,2:ncol(corr.charRF)]
corr.charRF<- corr.charRF[ , -which(names(corr.charRF) %in% c("pat.secRF"))]
dim(corr.charRF)#Just one case categorized as "U".
```

```
corr.RF.mod<-randomForest(corr.charRF[,1:(ncol(corr.charRF)-1)],as.factor(cor
r.charRF$Gender))

cTab.RF<- corr.RF.mod$confusion
cTab.RF
round(sum(diag(cTab.RF)) / sum(cTab.RF)*100 , 2)

##################################################### Random Forest CV
#Define training control
seed<-set.seed(123)
trainContrRFGen <- trainControl(method="repeatedcv", number=10, repeats=3)
metric <- "Accuracy"
set.seed(seed)
mtry <- sqrt(ncol(corr.charRF))

#10 fold CV Random Forest
tunegrid <- expand.grid(.mtry=mtry)
modelRFGen <- train(as.factor(Gender) ~., data=corr.charRF, method="rf", metr
ic=metric, tuneGrid=tunegrid, trControl=trainContrRFGen)
print(modelRFGen)

# Random Search
trainContrRFGen2 <- trainControl(method="repeatedcv", number=10, repeats=3, s
earch="random")
modelRFGen2 <- train(as.factor(Gender) ~., data=corr.charRF, method="rf", met
ric=metric, tuneLength=15, trControl=trainContrRFGen2)
print(modelRFGen2)
plot(modelRFGen2)
```

Model 6: Regression with 120 (Best) Predictors and CV - Age.

```
######################################## Prediction Type  ###################
######################
#################################### Regression Analysis
regCombRE<-regCombLow[1:120,,drop=0]#Number of reg combinations
dim(regCombRE)

#Calculating correlation for 200 combinations with AIC  #563.04 seg (9.38 min
)
corrsRE<-matrix(NA,3280,1)#Creating a matrix to be filled
for(i in 1:nrow(regCombRE)){
  regs<-c(regCombRE[i,1],regCombRE[i,2])#Getting two regions at a time
  corrs<-corr.fun.EspX(bac,regs,mris,patiens)
  corrsRE<-do.call(cbind,list(corrsRE,corrs))
}

corrsRE<-corrsRE[,2:ncol(corrsRE)]#Removing first NAs column.
dim(corrsRE)

pat.secRE<-rep(as.numeric(ID.names), each=exp.scanTwo, time=1)#Repeating each
```

```r
ID as much as Corrs subjects exist
corrs.ID.RE<-cbind(corrsRE,pat.secRE)#Binding with corr matrix
dim(corrs.ID.RE)

#Correlations averaged by subject
corr.avgRE<-as.matrix(aggregate(corrs.ID.RE[,1:(ncol(corrs.ID.RE))],
                                list(corrs.ID.RE[,ncol(corrs.ID.RE)]), FUN =
function(avg){sqrt(mean(avg^2))}))
dim(corr.avgRE)

#Matching characteristics with subjects
corr.charRE<-merge(corr.avgRE,characteristics.df[,c(1,3)], by.x="Group.1", by
.y="Subject")

corr.charRE$AgeCod[corr.charRE$Age == "22-25"] <- 23.5
corr.charRE$AgeCod[corr.charRE$Age == "26-30"] <- 28
corr.charRE$AgeCod[corr.charRE$Age == "31-35"] <- 33
corr.charRE$AgeCod[corr.charRE$Age == "36+"] <- 36

corr.charRE<-corr.charRE[,2:ncol(corr.charRE)]
corr.charRE<- corr.charRE[ , -which(names(corr.charRE) %in% c("pat.secRE","Ag
e"))]
dim(corr.charRE)

set.seed(123)  # setting seed to reproduce results of random sampling
trainingRows <- sample(1:nrow(corr.charRE), 0.8*nrow(corr.charRE))  # row ind
ices for training data
trainingDataRE <- corr.charRE[trainingRows, ]  # model training data
testDataRE  <- corr.charRE[-trainingRows, ]

#Predictive Model
corr.RE.mod<-lm(AgeCod ~ ., data=corr.charRE)
summary(corr.RE.mod)

#Prediction Accuracy
CorrHat.RE<-predict(corr.RE.mod, testDataRE)
RealPredsRE <- data.frame(cbind(actuals=testDataRE$AgeCod, predicteds=CorrHat
.RE))
cor(RealPredsRE)

mape <- mean(abs((RealPredsRE$predicteds - RealPredsRE$actuals))/RealPredsRE$
actuals)
mape*100

############################ Cross-Validation RE ############################
#Define training control
set.seed(123)
trainContrRE2 <- trainControl(method = "repeatedcv", number = 10, repeats = 1
0)
```

```
#Train the model
modelRE <- train(AgeCod ~ ., data = corr.charRE, method = "lm",
                 trControl = trainContrRE2)

# Summarize the results
print(modelRE)
```

Model 7: Random Forest with 120 (Best) Predictors and CV - Age.

```
######################################################## Random Forest CV
######################### Prediction Type Analysis
#Define training control
seed<-set.seed(123)
trainContrRFAge <- trainControl(method="repeatedcv", number=10, repeats=10)
metric <- "RMSE"
set.seed(seed)
mtry <- sqrt(ncol(corr.charRE))

#Train the model
tunegrid <- expand.grid(.mtry=mtry)
modelRFAge.Pred <- train(AgeCod ~., data=corr.charRE, method="rf", metric=met
ric, tuneGrid=tunegrid, trControl=trainContrRFAge)

# Summarize the results
print(modelRFAge.Pred)
```

Model 8: Random Forest with 120 (Best) Predictors and CV - Age as Categorical.

```
######################### Classification Type Analysis
#Using Response as Categorical Variable
metric <-  "Accuracy"
modelRFAge.Class <- train(as.factor(AgeCod) ~., data=corr.charRE, method="rf"
, metric=metric, tuneGrid=tunegrid, trControl=trainContrRFAge)
print(modelRFAge.Class)

# Random Search
control <- trainControl(method="repeatedcv", number=10, repeats=3, search="ra
ndom")
modelRFClass2 <- train(as.factor(AgeCod) ~., data=corr.charRE, method="rf", m
etric=metric, tuneLength=15, trControl=control)
print(modelRFClass2)
plot(modelRFClass2)
```