Experiences in finding relationship between genes, aging and cortical thicknesses: a machine learning approach

Joonas Puura, Prabhant Singh, Sriyal Himesh Jayasinghe

Advisor: prof. Raul Vicente

June 20, 2018

Abstract

Several genes have been associated to neurodegenerative diseases. We would like to link those genes' expressions to measured cortical thicknesses. By knowing which gene expression combinations are factors in decrease of cortical thicknesses, we would have early indicators on possible future disease. In this paper we describe our experiences in trying to apply several different machine learning and statistical techniques in order to solve the aforementioned task.

1 Introduction

"Neuroimaging research indicates that human intellectual ability is related to brain structure including the thickness of the cerebral cortex. [...] Most studies indicate that general intelligence is positively associated with cortical thickness in areas of association cortex distributed throughout both brain hemispheres [MCP+13]". Given the description, the main goal of this project is to find out whether there is any connection between expressions of certain genes and changes in cortical thicknesses. If made the connection, we would gain another tool in preventive measures against neurodegenerative diseases which are linked with reduction in cortical thicknesses [ZIBC+13].

Knowing that having certain genes would mean that there is a higher chance of developing neurodegenerative (e.g. Alzheimer) disease later in life, would help us in applying medical procedures earlier and check up on the patients, who are more likely to develop the disease, more.

In this paper we give an overview of trying to link genes expressions' to cortical thicknesses. Given a sample size of 160, with each sample having age, sex and list of expressed genes, together with measurements from 149 areas in the brain, we apply different analysis and machine learning techniques in order to try and see if there is a connection.

The paper is formatted to first give a information to the reader about dataset and about the methods used. The following is the work process used with descriptions approaches we used and how they were applied.

Last part is for discussion of what was found and what are possible ways forward.

2 Background

In this section we give a brief overview of required knowledge in order to understand what has been done during the project described in this article.

2.1 Dataset

Out dataset contains the list of 149 points of cortical thicknesses in the skull and 6 genotypes plus age and sex. The 6 genotypes included were:

- APOE - Apolipoprotein E protein
- CR1 - Complement receptor type 1 protein
- PICALM - Phosphatidylinositol binding clathrin assembly protein
- BIN1 - Bridging Integrator-1 protein

- CLU - Clusterin protein
- ABCA7 - ATP-binding cassette sub-family A member 7 protein

Several articles [MRF+14, SXT+17]have suggested that some of those above mentioned genes have an influence on several brain atrophy and cognitive impairments such as Alzheimer.

Our goal in this experiment is to find the relation between genes listed above and mean cortical thicknesses of measured brain parts. Our current approach focuses on removing genes from the dataset one by one to see of there's any reduction in the RMSE.

2.2 Regression analysis

Regression analysis is a predictive modeling technique, which studies the relationships between a dependent/target and independent/predictor variables [7Ty]. Regression analysis helps to identify significant relationships between dependent variable and independent variable or strength of impact of multiple independent variables on a dependent variable [7Ty].

2.2.1 Regression on categorical data

As our dataset consists mostly of categorical features - what kind of gene does a certain person exhibit. Usually regression analysis on categorical data is done with conversion to numerical form and then by fitting the regression function. The transformation does not necessarily have to be done for all models, as some models do not have such prerequisites.

2.3 Evaluation measures

In order to know how well trained models manage to predict we need some kind of measures. Two of the more common measures used are MSE and RMSE.

Mean squared error

$$MSE = \frac{1}{n}\Sigma_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

, where n is the number of samples, $\hat{Y}_i$ is the i-th predicted value, and Y_i is the i-th ground truth value

Root-mean-square error

$$RMSE = \sqrt{MSE}$$

RMSE is often used instead of MSE, because it provides for better interpretability, as then the units would be in the same scale.

2.4 Data encoding

In order to use certain models we have to encode (transform our data) into other form, which is understandable by the machine learning libraries we are going to apply.

2.4.1 Categorical data encoding methods

Encoding, which is used on

2.5 Scikit-learn

Scikit-learn [PVG+11] is an open source library for Python, which provides use of several machine learning algorithms. It has several algorithms for machine learning tasks such as clustering, classification and regression. In our project we use it to build regression models by making use of its model RandomForestRegressor [324].

RandomForest models in Scikit-learn requires data to be encoded into numerical form, which means that we have to encode our categorical features in numeric values. There are certain dangers in doing that. For example

2.6 H2O

H2O[tea15] is an open source machine learning / data analysis platform mainly meant for big data analysis.

2.7 Optimized pipeline

To make another optimal pipeline to compare with H2O model, we made use of TPOT[OUA+16][OBUM16] to give us an optimal pipeline.
The optimized pipeline is a ExtraTreeRegressor with parameters being max features of 0.85 and 100 estimator, built on top of a K-nearest neighbor regressors with k being 50 neighbors and with uniform weights.
Our optimized pipeline gave us similar results as H2O model has managed to give us.

One-hot encoding. Encode categorical integer features using a one-hot aka one-of-K scheme. The input to this transformer should be a matrix of integers, denoting the values taken on by categorical (discrete) features. The output will be a sparse matrix where each column corresponds to one possible value of one feature.
This encoding is needed for feeding categorical data to many scikit-learn estimators, notably linear models and support vector machines with the standard kernels.

For experiments with random forest and optimized pipeline we used Label encoder with one-hot encoding in SK-Learn [One].

Categorical encoding Basic categorical encoding is done as follows: each unique value in the labels set for the specific feature gets an bijective relation with set of natural numbers.

2.8 Decision Tree

The decision trees are used to fit a sine curve with addition noisy observation. As a result, it learns local linear regressions approximating the sine curve. Decision tree regression gave worse results than predicting mean for the cortical thickness.

2.9 Random forest

Our dataset contains only 8 categorical values, to prevent information loss during one hot encoding we conducted tests on Random forest regressor with both H2O and Sk-learn libraries. The H2O library gave a significant improvement of RMSE over SkLearn regressor. The H2O Optimizes it's random forest by itself by using Repeated Crossvalidation so comparing it with Naive Random forest was not ideal. A naive random forest regressor with 1000 estimators gave worse results than simply predicting the mean. Hence we tried to find our own optimal model with TPOT [OUA+16] which uses genetic algorithms to find an optimal model for the specified dataset.

3 Work process

In this section we give an overview of what techniques we used and how did we apply them during this project.

3.1 Data extraction

Our dataset consisted of 160 samples. The data came in two parts:

- Excel table with data on subjects age, sex and genotypes
- Folders of structured measurements for each subject: one file per subject for each hemisphere.

We wrote a script to extract measurements for each subjects, aggregate them into single table and then join them up with age, sex and genotypes data.

3.2 Correlations between brain areas thicknesses

In order to see how much different brain areas are related to each other we can plot a pairwise correlation matrix. From the matrix (Fig. 1) we can see, that there are areas which are strongly correlated and that there are areas which have almost no correlation at all. We also note that rectangles of same level of correlations appear, which means that there are clusters of areas which are more correlated together. Also there seems to be strong correlation between same areas between different hemispheres. There also seem to be some areas which have really low correlations with every other area.

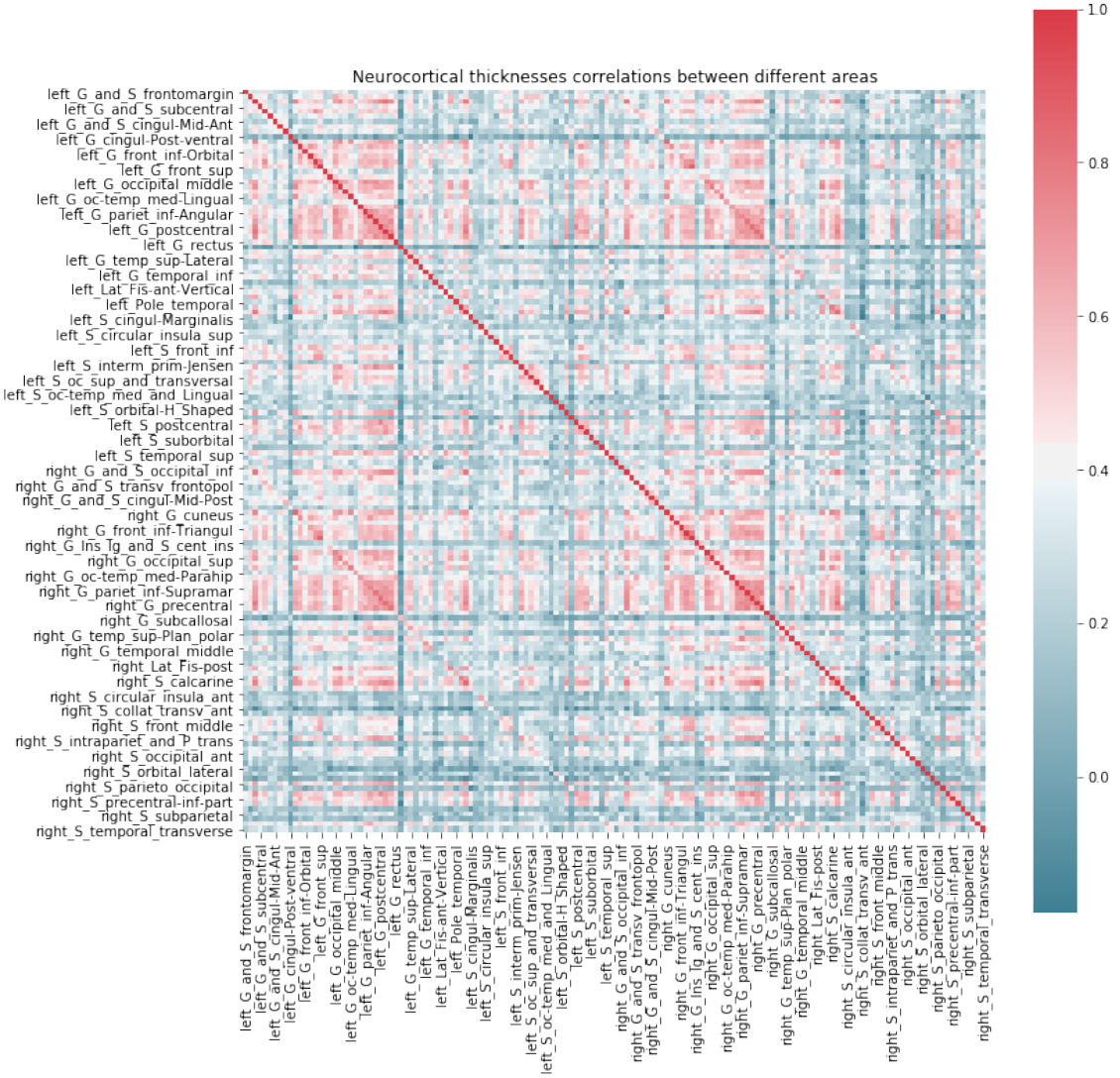


Figure 1: Correlation matrix between different brain areas thicknesses

3.3 Exploring dataset

As the cortical dataset has 148 measures per sample we tried to make use of dimensionality reduction algorithms. We tried [vdMH08], PCA [Pri] and SVD [Sin].

With t-SNE algorithm we used perplexity values 5 up to 50 with steps of 5. With PCA and SVD algorithms we set our reduced dimensionality size to 2. Our main goal here was to see if there are any visual clusters forming after using dimensional reduction algorithms.

As shown from the above figures we were unable to identify any meaningful clustering.Additionally based on the reduced dimensions of PCA and SVD algorithms we trained KNN models with different K values to see whether the models would be able to classify the genes.

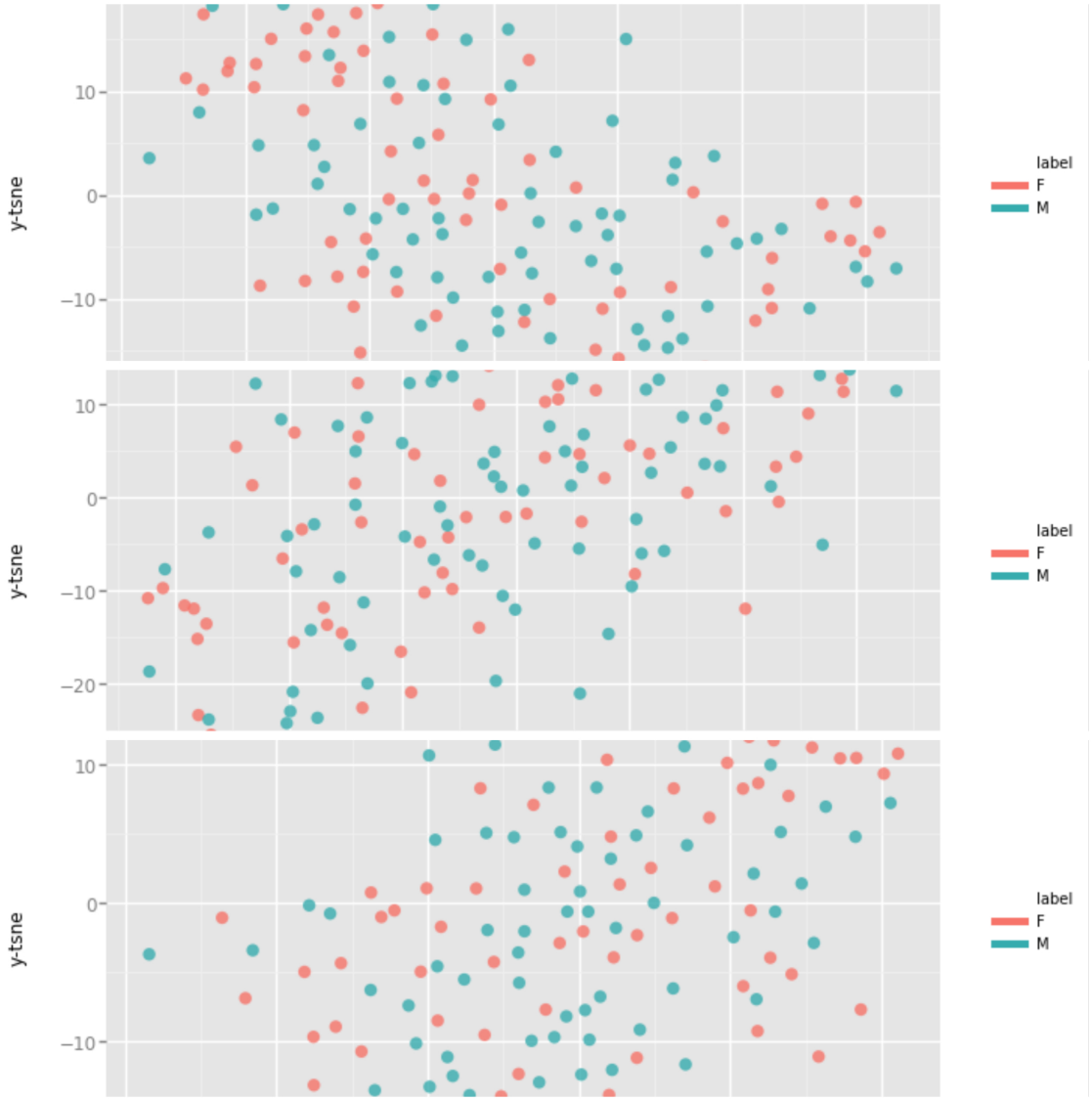


Figure 2: t-SNE results with perplexity values 5, 10 and 15. Coloring according to subject being Male or Female

3.4 Evaluating that models work

In order to be sure that our models are able to learn from categorical data we generate synthetic data, with some underlying function and see if the model is able to learn the function from inputs and outputs.

3.4.1 Generating synthetic data

In order to be sure that models actually work we generate synthetic data as was discussed and done in a blog post [Are]. The article discussed that some models are not implemented properly in software packages and therefore it is useful to evaluate if they actually work on categorical data.

Generating similar data We tried to make our synthetic data be a bit similar to our input data by having one numerical value (age) and others categorical value (binary and multi-class categorical features). Often, when we do supervised machine learning (e.g. classification, regression) we try to approximate some kind of a relation (function) between input and output.

Therefore, in order to generate our synthetic data, we made up a function, which we are then, after using it to generate data, trying to again approximate: $y = (100 - age) + sex\_coeff + gene1\_coeff + gene2\_coeff + ... + gene6\_coeff + +noise$ where sex_coeff is $male = 10$ if subject

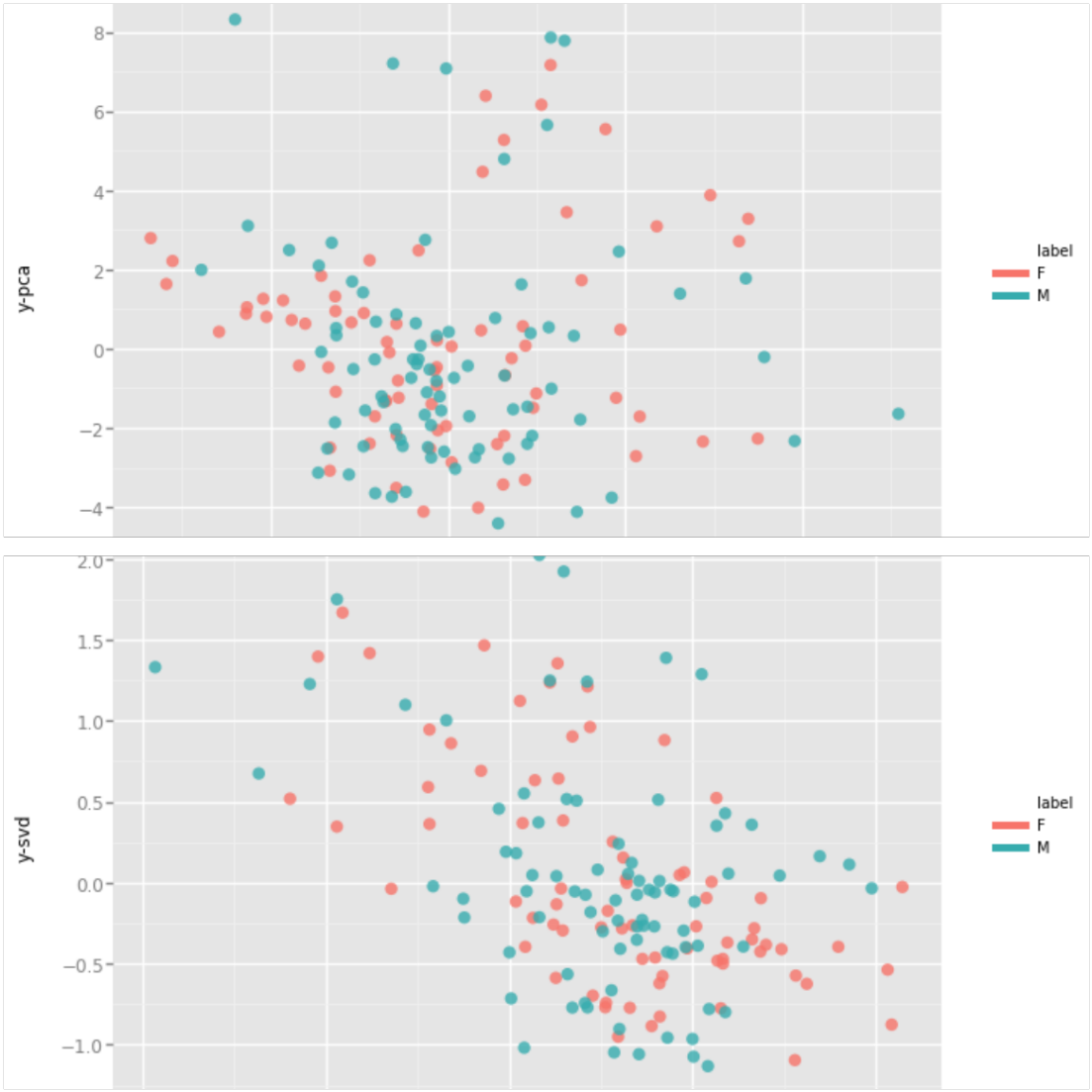


Figure 3: PCA and SVD results. Coloring according to subject being Male or Female

or *female* = 5 if female. Each *genex_coeff* attribute will be either $a = 3$, $b = 1$ or $c = 5$. We also add some noise, a random value between -5 and 5, in order to represent noisiness of real world data. We limit age to be an integer between 50 and 70. We sample all our values randomly from a uniform distribution. The code sample can be seen on Figure 4

Our second generation function is to only generate categorical data - by removing age feature.

3.4.2 Permutation test

Although, we do not do any specific calculations, we can visually interpret if model's output results could be significant or not. We run something similar to permutation test [Res].

In order to check if our models actually learn from real labels we first run several trials on real labels (with different train-test splits), record the mean result and then run trials on datasets, which have their labels (target values) shuffled. For testing if our models work on synthetic data, we shuffled our dataset labels 100 times. On each of the shuffled dataset we ran 20+ trials with different train-test splits and calculated means of mean squared errors.

To see if the model has learned from the data, we should get considerably lower MSE on correct labels, while on shuffled labels we should mostly get worse results.

```python
def gen_func():
    age = randint(50, 70)
    sex = choice(["Male", "Female"])
    gene1 = choice(["A", "B", "C"])
    gene2 = choice(["A", "B", "C"])
    gene3 = choice(["A", "B", "C"])
    gene4 = choice(["A", "B", "C"])
    gene5 = choice(["A", "B", "C"])
    gene6 = choice(["A", "B", "C"])
    noise = random() * 10 - 5

    label = score_age(age) + score_sex(sex) + score_gene(gene1) +  score_gene(gene2) +\
        score_gene(gene3) +  score_gene(gene4) +  score_gene(gene5) +  score_gene(gene6) + noise

    return [age, sex, gene1, gene2, gene3, gene4, gene5, gene6, label]
```

Figure 4: Python code for generating synthetic data

3.5 Training and evaluating models

In this section we see if models are able to work on categorical data, as there has been some discussion of some software packages not working correctly [Are]. We generate categorical synthetic and see if the models are able to learn on it.

3.5.1 Training SKLearn's RandomForestRegression model

Results on synthetic data. We trained RandomForestRegressor on the synthetic data for which we applied categorical encoding, eg. A, B, C were encoded as 0, 1, 2 respectively, which consisted only on categorical data. We can see (Figure 5) that that the results we got from training on real labels were considerably smaller than on shuffled labels dataset. This can be considered as empirical evidence to RandomForestRegressor model being able to successfully learn from categorical data.

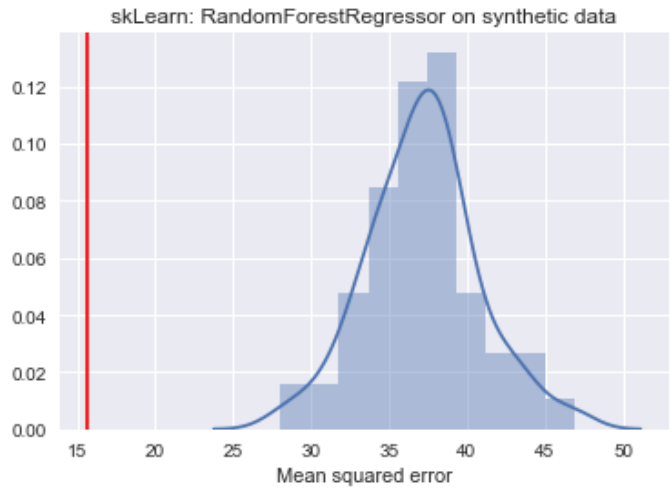


Figure 5: Red line represents mean of squared errors on real labels, distribution is on shuffled data

Results on real data. We applied the same methods as we did on the synthetic on the real dataset. For genes categorical encoding was applied - if there were 3 different possibilities, the gene was encoded as 0, 1, 2. We show a selection of prediction results for 9 different brain areas (Figure 6). In those figures we plot the red line as mean MSE for the cases when training on real labels and the blue distribution as results we get when we randomize our labels.

When going over the results in (Figure 6) we see that there is one plot (1st column, 3rd row), where there seems to be significant difference between random labels and true labels. This belonged to region left_G_and_S_cingul-Mid-Post. This could imply that there is some relation between features and cortical thicknesses in that certain brain region.

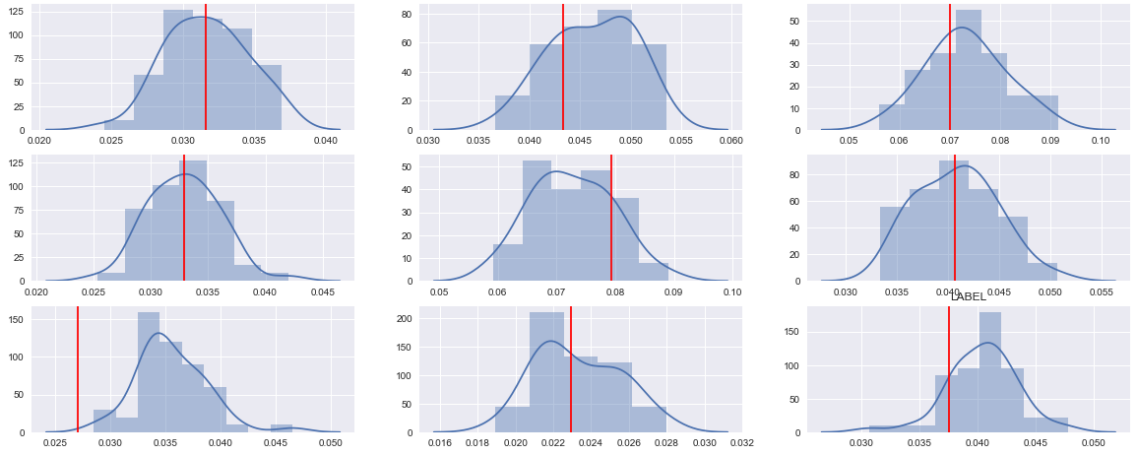


Figure 6: Example of results on checking model's viability for brain parts. Red line represents the mean of squared errors on real labels, distribution is on shuffled data

3.5.2 Training other models

We also trained H2O models and our own optimized pipeline method. The results from those two did not differ too much.

3.5.3 Training KNN model with PCA and SVD dimensionality reduction

One of the ideas we investigated, was to see if instead of getting a mapping from genes expressions' to cortical thicknesses, we wanted to reverse it and see if we can get relation of cortical thicknesses to genes' expressions.

To do this we applied PCA and SVD dimensionality reduction algorithms on the cortical thickness data by reducing the dimensionality from 148 thickness features down to 3. Afterwards we trained KNN models with 3, 5, 7 and 9 neighbors for each of the gene type between the reduced thicknesses data set and genes' expressions. We plotted the confusion matrices for the results after predicting on the test data sets, with random split of 80% train and 20% test (Figures. 7, 8, 9, 10).

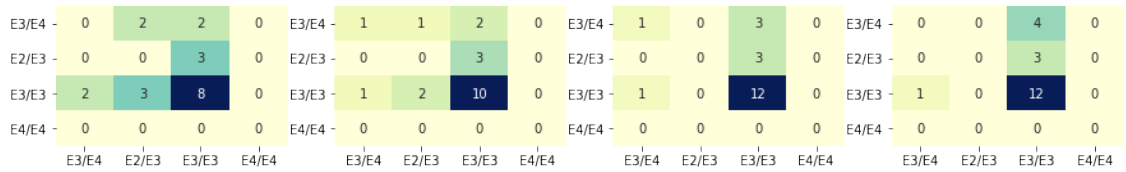


Figure 7: Apoe gene classification confusion matrix for K values 3,5,7,9. Vertical axis is ground truth values and horizontal axis is predicted values

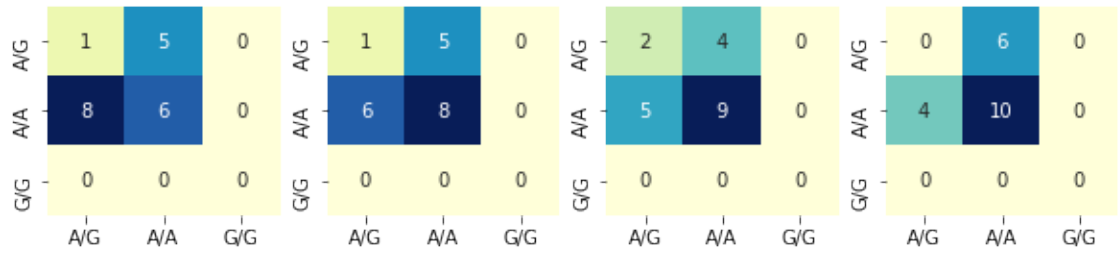


Figure 8: bin1 gene classification confusion matrix for K values 3,5,7,9. Vertical axis is ground truth values and horizontal axis is predicted values

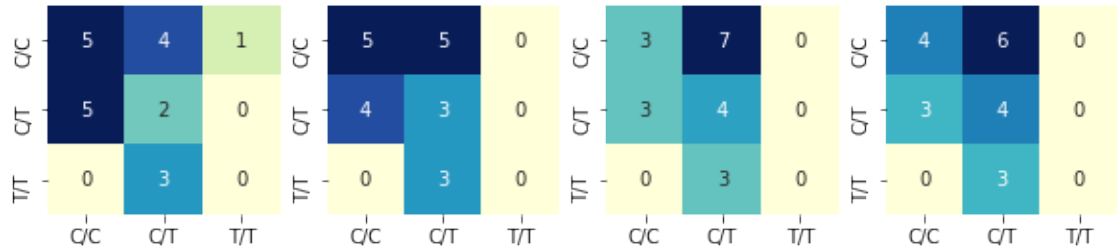


Figure 9: clu gene classification confusion matrix for K values 3,5,7,9. Vertical axis is ground truth values and horizontal axis is predicted values

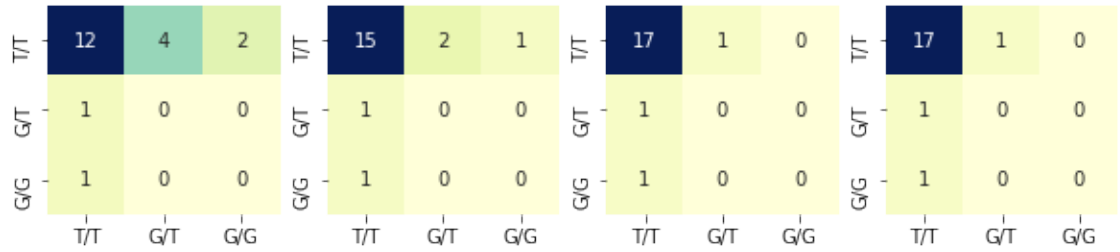


Figure 10: abca7 gene classification confusion matrix for K values 3,5,7,9. Vertical axis is ground truth values and horizontal axis is predicted values

What we really noticed in this case, is that in our dataset, the genes' expressions are not uniformly distributed. This is not the problem of our dataset, but some expressions are way more common than the other ones. Due to this we have a really imbalanced data set, which skews our accuracy measures. This problem could be alleviated by having more data and applying some techniques in order to make the dataset more balanced.

In this case we can see that the some models might seem to get pretty good results, but they are often achieved by doing something stupid - for example being almost a constant classifier, which seems to happen in Figure 10. This just means that almost all the points we trained on, were of T/T class, so wherever we pick a point in th plane, the closest K ones would always be T/T.

3.6 Finding more interesting brain areas

By checking the model's mean of trials against true labels vs trials against shuffled labels we can get some intuition on the brain parts, which are more promising to have a relation between its cortical thickness and genotypes, sex and age.

Our first step was to find out, which brain parts are more promising. To do that, we first trained models for each brain region's thickness data. For each brain region we also run a number of trials, every time reshuffling the target labels. By now comparing the results on real labels to those we got by randomly permutating target labels, we can find out how unlikely is it for the results for true labels to appear in a random setting.

If the result from training on real data is on the left-end tail of the distribution or to the left of the distribution at all, then it might imply that there is a considerable relation between input and output data. We give an example of it in the following sections.

3.7 Checking on more-promising brain sectors

From our previous steps we filtered out some more promising sectors for relations between its cortical thicknesses and age, sex and genotype. Currently, we filtered according to our visual interpretation of results. We picked the ones which are left of the shuffling tests' distribution or far end on the left tail to it. Instead of visual picking, some statistical test could be instead used in order to pick more promising ones (e.g. t-Test or calculating percentiles). Full list of images for all brain areas are listed in Appendix A.

By agreement, we decided the following ones to be significant enough:

- right_S_oc-temp_med_and_Lingual
- right_S_parieto_occipital
- left_G_and_S_cingul_Mid_Ant
- right_G_temp_sup-Ptan_polar

Closer look at specific regions One of the more promising brain areas to have a relation was area G_and_S_cingul_Mid_Ant in the left hemisphere. From initial testing, using a SKLearn's RandomForestRegression model we saw that it had a promising separation between random data and true data. We then tried training some other models on it, to see if we can get any better results on that area. We trained a H2O random forest regression model on it. We can see, that training on true labels indeed gives us better separation showing that there is some relationship between that brain region and our input data (Fig. 11).

Figure 11: Model trained on left_G_and_S_cingul_Mid_Ant thicknesses. Red line represents the mean of squared errors on real labels, green distribution are trials on correct labels (different train-test splits) and blue distribution on shuffled data. Y-axis is the number of trials

We also tried training on area right_S_oc-temp_med_and_Lingual region. From the results (Fig. 12) we can see that the mean of trials on real labels is at the most-end of left-tail of random shufflings' distribution. However, after we removed feature giving us information on age (Fig. 13) we see that the distributions are way more collided.

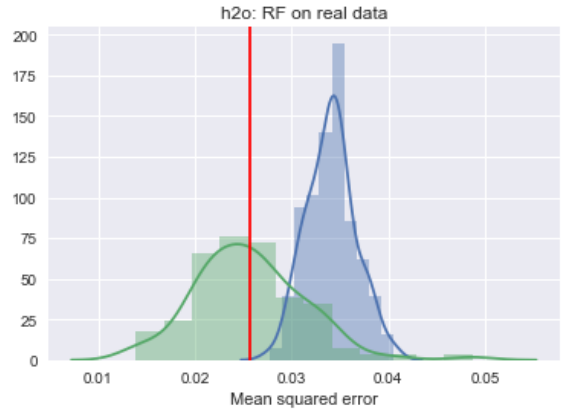


Figure 12: Model trained on right_S_oc-temp_med_and_Lingual region. Red line represents the mean of squared errors on real labels, green distribution are trials on correct labels (different train-test splits) and blue distribution on shuffled data. Y-axis is the number of trials outputting resulting MSE being in corresponding bin

Figure 13: As previous figure, but with age feature removed. We can see that age played a major role.

3.8 Feature importance

One of the more important questions, we wanted answers for, was to find out how genes expressions and their combinations affect cortical thicknesses. In this paper we played around with three different ways in order to investigate usefulness of variables.

3.8.1 Investigating features by removal

One way to investigate the effect of features is to train two models - one with and the other without it. For example we train one model with having the certain feature and the other model without having it. If the RMSE gets higher considerably, then could sign that this feature was previously put to a great use.

After testing with the removals, there was a minute difference when removing gene expression data, but there was a more significant increase in RMSE, when we removed age (Fig. 13) or sex features.

3.8.2 Feature_importances variable

SKLearn's package's random forest model instances also keeps an array called *feature_importances_*. After having trained the model instance, the array will then be a list of importance of features assigned for each of the features. From there we can learn, how important did the model think the feature to be.

Eg. on *rightS_oc − temp_med_and_Lingual* area we got the following *feature_importances_* results:

```
array([0.12929222 (sex), 0.36137515 (age), 0.10674656, 0.09864512,
0.12065848, 0.09475595, 0.08852653])
```

. This says that, for brain area *rightS_oc − temp_med_and_Lingual* , the age played the most important role in guessing what is the correct cortical thickness for specific gene expressions.

3.8.3 LIME interpretations

We tried to look into different ways, on how to interpret results of ensemble methods. One of the libraries we found is called LIME [RSG16]. Lime helps to interpret the trained models in an easier, more comprehensible and visualized ways.

In this case we used LIME [RSG16] to analyze our trained random forest models. Practically, what lime allows us to do, is to check for each test instance, how was it graded - what features played a role in defining the result to this test instance.

Example results on a test instance on model trained using a *rightS_oc − temp_med_and_Lingual* brain areas as a reference is given in Figure 14. Form the figure we can read, that for the specific train instance, value 2.02 was predicted and what were factors which pulled the value away from

the mean. E.g. $sex = 1$, means that the subject being male, reduced its predicted cortical thickness by 0.09, age being over 71 reduced it by a further 0.09 and some on specific expressions of genes.

By going over samples, we think that this tool could bring some good insights into, on how you regression model is predicting results.

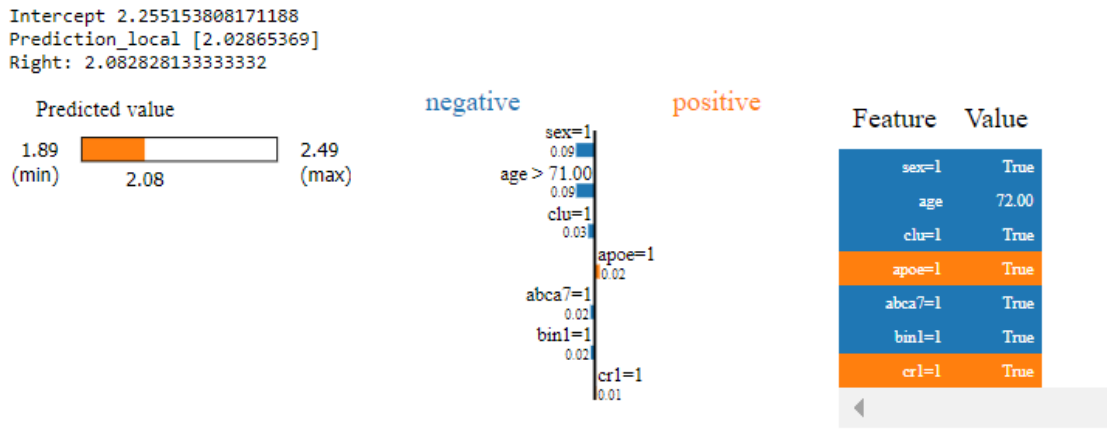


Figure 14: LIME Interpretation for an instance on a trained random forest regression model

4 Discussion

In this section we give a summary on what we tried and what were the interesting things we found. We also discuss on some possible ways forward for extending the investigation into this topic.

4.1 Results

In this project, by making use of permutation testing, we found that there are a few areas in brain which seem to be more dependent on our given dataset features - right_S_oc-temp_med_and_Lingual, right_S_parieto_occipital, left_G_and_S_cingul_Mid_Ant and right_G_temp_sup-Ptan_polar. By doing further testing on the first region in the list, we found that in this case age seems to be a feature of greatest importance and after removing it the predictions got considerably worse.

A backwards approach was also attempted by trying to find genes from cortical thicknesses. In this case we found that due the fact that expressions are not drawn randomly from uniform distribution, but some expressions are more likely to appear in general giving us an imbalanced dataset.

The dataset was also being analyzed by different exploratory methods, such as dimensionality reduction algorithms and by looking at correlations of cortical thickness data.

4.2 Ways forward

There could be many more tests being run as this paper has some lack of empirical results in a nicely formatted tables. What have been discussed, together with our advisor Raul Vicente, is that there could be a lot to gain in prediction accuracies if we were to include some domain specific knowledge. Also an increase in size of dataset might go a long way here.

There was also a question, on how to find sets of gene expressions which contribute to decrease of cortical thicknesses the most. Currently we have not found a good way to deal with this problem, but this is certainly an important question to try and find an answer to.

5 Acknowledgements

We would like to thank our advisor professor Raul Vicente for sharing his valuable insights and knowledge for this project. We would also like to acknowledge support from StudyITin.ee programme for supporting our studies.

References

[324] 3.2.4.3.2. sklearn.ensemble.randomforestregressor — scikit-learn 0.19.1 documentation. http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html. (Accessed on 06/20/2018).

[7Ty] 7 types of regression techniques you should know. https://www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/. (Accessed on 06/20/2018).

[Are] Are categorical variables getting lost in your random forests? https://roamanalytics.com/2016/10/28/are-categorical-variables-getting-lost-in-your-random-forests/. (Accessed on 06/20/2018).

[MCP+13] Kyle Menary, Paul F. Collins, James N. Porter, Ryan Muetzel, Elizabeth A. Olson, Vipin Kumar, Michael Steinbach, Kelvin O. Lim, and Monica Luciana. Associations between cortical thickness and general intelligence in children, adolescents and young adults. *Intelligence*, 41(5):597–606, sep 2013.

[MRF+14] K. Morgen, A. Ramirez, L. Frolich, H. Tost, M. M. Plichta, H. Kolsch, F. Rakebrandt, O. Rienhoff, F. Jessen, O. Peters, H. Jahn, C. Luckhaus, M. Hull, H. J. Gertz, J. Schroder, H. Hampel, S. J. Teipel, J. Pantel, I. Heuser, J. Wiltfang, E. Ruther, J. Kornhuber, W. Maier, and A. Meyer-Lindenberg. Genetic interaction of PICALM and APOE is associated with brain atrophy and cognitive impairment in Alzheimer's disease. *Alzheimers Dement*, 10(5 Suppl):S269–276, Oct 2014.

[OBUM16] Randal S. Olson, Nathan Bartley, Ryan J. Urbanowicz, and Jason H. Moore. Evaluation of a tree-based pipeline optimization tool for automating data science. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, GECCO '16, pages 485–492, New York, NY, USA, 2016. ACM.

[One] http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.onehotencoder.html.

[OUA+16] Randal S. Olson, Ryan J. Urbanowicz, Peter C. Andrews, Nicole A. Lavender, La Creis Kidd, and Jason H. Moore. *Applications of Evolutionary Computation: 19th European Conference, EvoApplications 2016, Porto, Portugal, March 30 – April 1, 2016, Proceedings, Part I*, chapter Automating Biomedical Data Science Through Tree-Based Pipeline Optimization, pages 123–137. Springer International Publishing, 2016.

[Pri] Principal component analysis - wikipedia. https://en.wikipedia.org/wiki/Principal_component_analysis. (Accessed on 06/20/2018).

[PVG+11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[Res] Resampling (statistics) - wikipedia. https://en.wikipedia.org/wiki/Resampling_(statistics). (Accessed on 06/20/2018).

[RSG16] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144, 2016.

[Sin] Singular-value decomposition - wikipedia. https://en.wikipedia.org/wiki/Singular-value_decomposition. (Accessed on 06/20/2018).

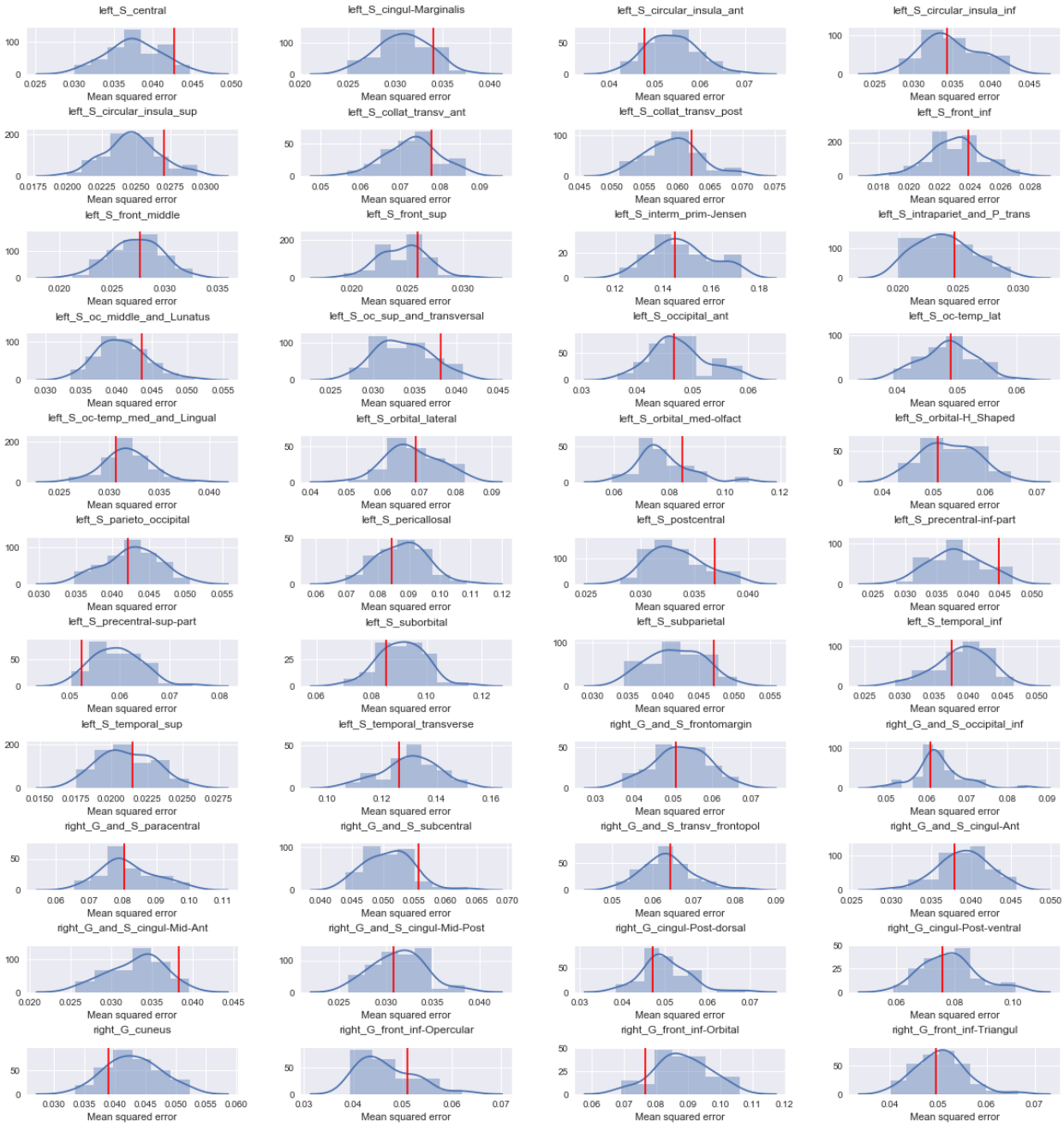
[SXT+17] Qiying Sun, Nina Xie, Beisha Tang, Rena Li, and Yong Shen. Alzheimer's disease: From genetic variants to the distinct pathological mechanisms. *Frontiers in Molecular Neuroscience*, 10:319, 2017.

[tea15] The H2O.ai team. *h2o: Python Interface for H2O*, 2015. Python package version 3.1.0.99999.

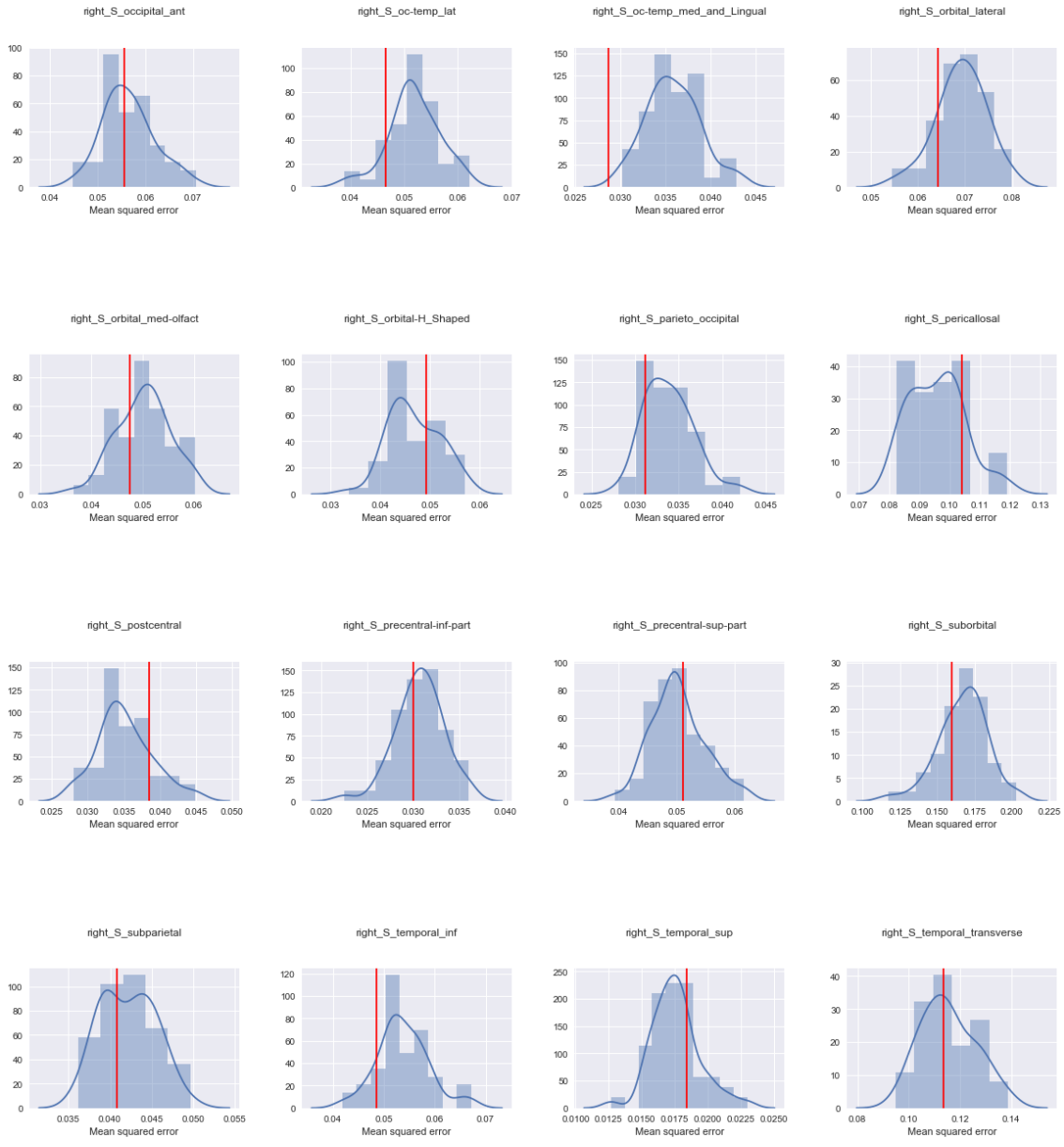
[vdMH08] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[ZIBC+13] Mojtaba Zarei, Naroa Ibarretxe-Bilbao, Yaroslau Compta, Morgan Hough, Carme Junque, Nuria Bargallo, Eduardo Tolosa, and Maria Jose Martí. Cortical thinning is associated with disease stages and dementia in parkinson's disease. *Journal of neurology, neurosurgery, and psychiatry*, 84(8):875–81, Aug 2013.

A RFRegression on all brain areas


left_S_central
left_S_cingul-Marginalis
left_S_circular_insula_ant
left_S_circular_insula_inf
left_S_circular_insula_sup
left_S_collat_transv_ant
left_S_collat_transv_post
left_S_front_inf
left_S_front_middle
left_S_front_sup
left_S_interm_prim-Jensen
left_S_intrapariet_and_P_trans
left_S_oc_middle_and_Lunatus
left_S_oc_sup_and_transversal
left_S_occipital_ant
left_S_oc-temp_lat
left_S_oc-temp_med_and_Lingual
left_S_orbital_lateral
left_S_orbital_med-olfact
left_S_orbital-H_Shaped
left_S_parieto_occipital
left_S_pericallosal
left_S_postcentral
left_S_precentral-inf-part
left_S_precentral-sup-part
left_S_suborbital
left_S_subparietal
left_S_temporal_inf
left_S_temporal_sup
left_S_temporal_transverse
right_G_and_S_frontomargin
right_G_and_S_occipital_inf
right_G_and_S_paracentral
right_G_and_S_subcentral
right_G_and_S_transv_frontopol
right_G_and_S_cingul-Ant
right_G_and_S_cingul-Mid-Ant
right_G_and_S_cingul-Mid-Post
right_G_cingul-Post-dorsal
right_G_cingul-Post-ventral
right_G_cuneus
right_G_front_inf-Opercular
right_G_front_inf-Orbital
right_G_front_inf-Triangul
Mean squared error


right_G_front_middle
Mean squared error
right_G_front_sup
Mean squared error
right_G_Ins_lg_and_S_cent_ins
Mean squared error
right_G_insular_short
Mean squared error
right_G_occipital_middle
Mean squared error
right_G_occipital_sup
Mean squared error
right_G_oc-temp_lat-fusifor
Mean squared error
right_G_oc-temp_med-Lingual
Mean squared error
right_G_oc-temp_med-Parahip
Mean squared error
right_G_orbital
Mean squared error
right_G_pariet_inf-Angular
Mean squared error
right_G_pariet_inf-Supramar
Mean squared error
right_G_parietal_sup
Mean squared error
right_G_postcentral
Mean squared error
right_G_precentral
Mean squared error
right_G_precuneus
Mean squared error
right_G_rectus
Mean squared error
right_G_subcallosal
Mean squared error
right_G_temp_sup-G_T_transv
Mean squared error
right_G_temp_sup-Lateral
Mean squared error
right_G_temp_sup-Plan_polar
Mean squared error
right_G_temp_sup-Plan_tempo
Mean squared error
right_G_temporal_inf
Mean squared error
right_G_temporal_middle
Mean squared error
right_Lat_Fis-ant-Horizont
Mean squared error
right_Lat_Fis-ant-Vertical
Mean squared error
right_Lat_Fis-post
Mean squared error
right_Pole_occipital
Mean squared error
right_Pole_temporal
Mean squared error
right_S_calcarine
Mean squared error
right_S_central
Mean squared error
right_S_cingul-Marginalis
Mean squared error
right_S_circular_insula_ant
Mean squared error
right_S_circular_insula_inf
Mean squared error
right_S_circular_insula_sup
Mean squared error
right_S_collat_transv_ant
Mean squared error
right_S_collat_transv_post
Mean squared error
right_S_front_inf
Mean squared error
right_S_front_middle
Mean squared error
right_S_front_sup
Mean squared error
right_S_interm_prim-Jensen
Mean squared error
right_S_intrapariet_and_P_trans
Mean squared error
right_S_oc_middle_and_Lunatus
Mean squared error
right_S_oc_sup_and_transversal
Mean squared error
Mean squared error
Mean squared error
Mean squared error


right_S_occipital_ant
right_S_oc-temp_lat
right_S_oc-temp_med_and_Lingual
right_S_orbital_lateral
right_S_orbital_med-olfact
right_S_orbital-H_Shaped
right_S_parieto_occipital
right_S_pericallosal
right_S_postcentral
right_S_precentral-inf-part
right_S_precentral-sup-part
right_S_suborbital
right_S_subparietal
right_S_temporal_inf
right_S_temporal_sup
right_S_temporal_transverse
Mean squared error