Group 7 - Brain Mapping
COE3803 Final Project

Heather Whittaker, Cori Harber, Lisa Meyer-Baese, Erika Moreno, Hope Mumme

Problem Statement

The overarching goal is to aid in automatic segmentation of neuro-imaging data (Fig. 1). Given a sub-patch of macro-scale neuro-images (Fig. 2), the objective is to classify this small section of an image to a brain area -- hypothalamus (thalamus), striatum, or cortex.

The solution pipeline spanned data augmentation, feature extraction, dimensionality reduction, and classification. Data augmentation was leveraged to generate a dataset with 92,160 total images. Features evaluated included entropy, edge detection, object detection, shape indices, histogram of oriented gradients, and more. Two sets of features were evaluated, the second with more features than the first. Multiple dimensionality reduction approaches were iterated over including no reduction, Scaled PCA with 3 components, Scaled PCA with 5 components, Kernel PCA, and Gaussian Random Process. For each of these reductions, Logistic Regression, k-Nearest Neighbors, Decision trees, Linear SVMs, and Kernel SVMs were iterated over for classification. Transformations and classification followed a 70-30 train-test split procedure.

Ultimately, the highest achieved accuracy was 97.8% using no dimensionality reduction and a kernel SVM as the classifier. Scores are reported as well as error analysis.

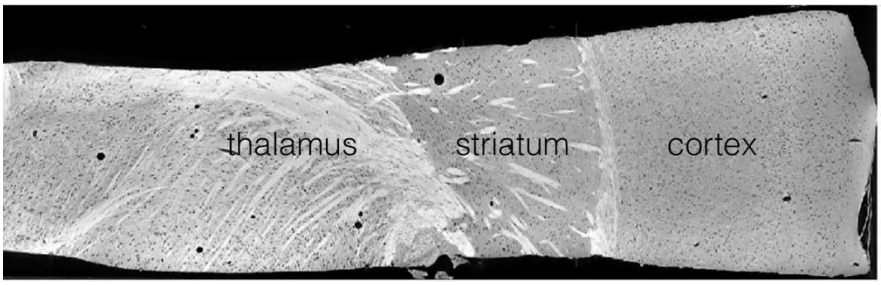


Figure 1: Neuro-Imaging Scan

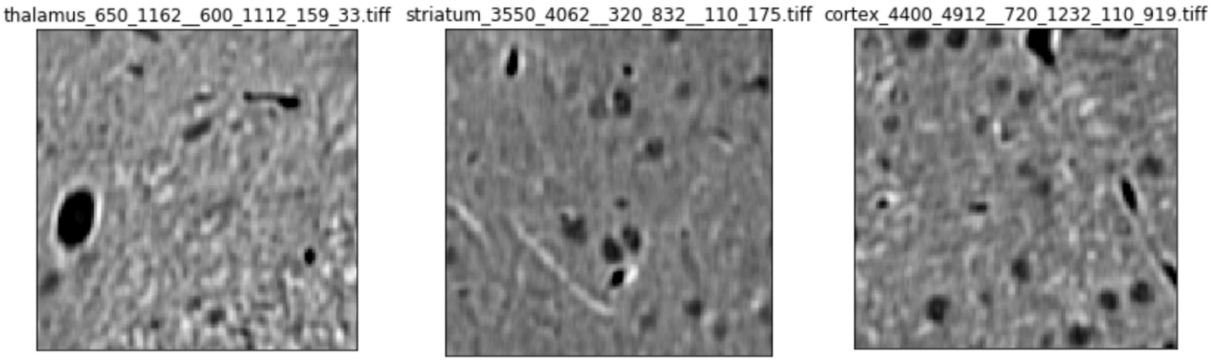


Figure 2: Sub-patch from Thalamus, Striatum, and Cortex Brain Areas

Approach

After exploring available images, a long preprocessing procedure was developed due to the number of images and the amount of transformations performed as described in the problem statement above. The pipeline handled over 90,000 images post-data augmentation. Augmenting the original data set to include quadrant-sub-sections and rotated-sub-sections of each image provided more detail about the key characteristics of each brain area and led to higher accuracy in classification. Figure 3 shows the workflow of this project.

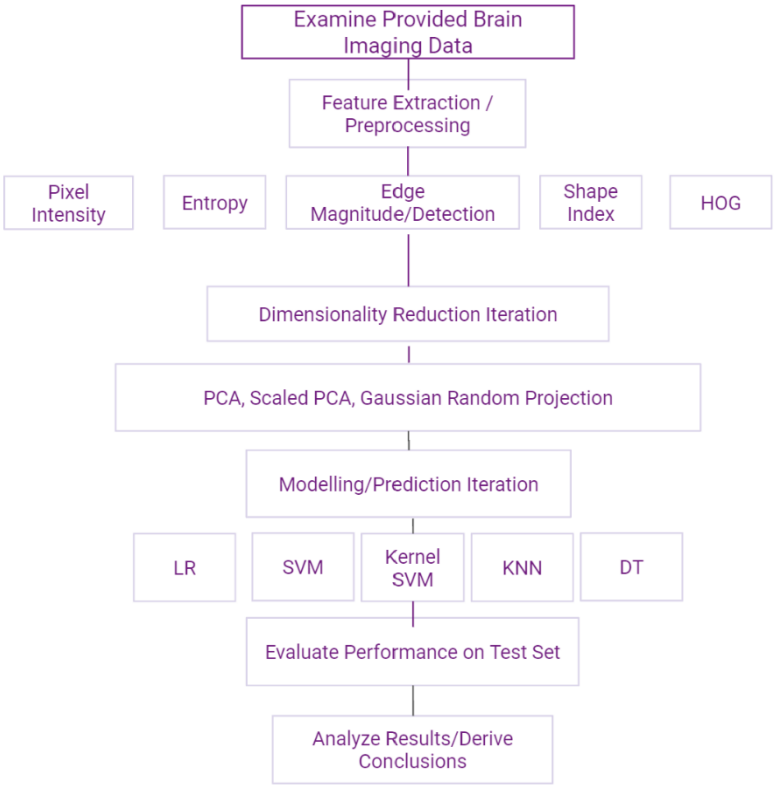


Figure 3. Workflow

Methods

Creating a baseline

To establish a baseline, the mean, variance, and sum of values from the entropy filter were used on the original images. Then a logistic regression classifier was implemented to obtain an accuracy value of the classification results and to provide a baseline for classification using only a couple of features. The training accuracy was 73.9% and the validation accuracy was 74.7%. These classification results are relatively high with only three features on the original images. The next step is data augmentation.

Data Augmentation

The original data set obtained contained 6,144 images. Half of these images, 3072, were identified as being part of the training set, while the other half was for testing. There were 1024

images representative of each of the three classes in each of the two sets. For example, 1024 labeled cortex images were a part of the training data set.

Image classification models are prone to overfitting to the training data. Therefore, to train a model that not only fits the training data but also generalizes well on the test data, both the bias and the variance should be minimized (Luis, 2017). Training on more data is one way to achieve this. The data set was increased in size by a factor of 14 through data augmentation by both turning and cropping the original image as seen in Figure 4.

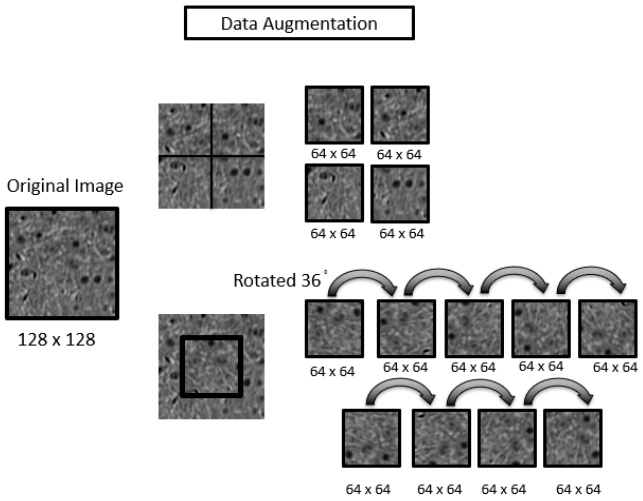


Figure 4: Data Augmentation steps

Feature Extraction

Feature Set #1

Data features provided in a raw image consist of pixels. Rather than use raw pixel-level features for classification, a few different feature extraction techniques were used in order to increase the amount of gleaned information. Entropy, pixel intensity, shape index, Histogram of Oriented Gradients (HOG), and canny edge detection filters were applied to the images to extract numerical values from the images.

Entropy is a measure of the image's overall complexity in a certain area of the image. The entropy filter is able to detect small changes in the image's gray scale distribution. It is especially helpful for this analysis since the images are very similar to the naked eye. Using the filter catches the slight differences between the different brain regions.

Shape Index values come from the eigenvalues of the Hessian. It is used to describe curvature as a single value. It is especially helpful in finding the structure of an image region from the local shapes it contains like the circular blood vessels.

Histogram of Oriented Gradients is an algorithm that detects the orientation of an image region. It is invariant to image lighting and is key in detecting elongated regions like blood vessels or axons.

The canny filter is used to detect edges from the Gaussian. It is able to remove the noisiness of an image and determine where there is an edge. A convolutional neural network (CNN) was used to detect bounding boxes of detected objects (cells); number of events, max

area, average area, average percent difference width/height, and max difference percent height features were found (to explore “spherical-ness”). However, with the 64x64 images, the majority of images detected nothing, and ~10% of the 128x128 parent images were unable to find objects themselves. The distribution of missing objects (191 Striatum images, 394 Cortex, 50 Thalamus) was too close, so canny edge detection was not included in the final feature matrix.

Once the images were processed, mean, variance, and sum values were extracted from each filter application and appended into a feature matrix. A pair plot (Appendix Fig. 13) to see how useful our features are and how they relate to each other. The feature matrix consisted of the image names as the rows and the feature names as the columns. Example outputs of the features are shown below in Fig. 5.

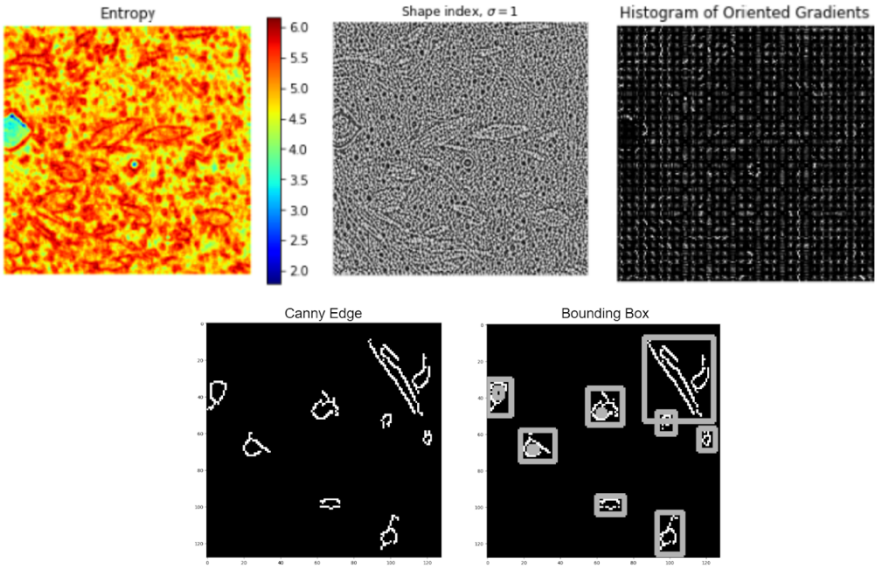


Figure 5. Shows output of entropy, shape index, histogram of oriented gradients, canny edge detection, and bounding box filters

Features Used: ['Entropy Mean', 'Entropy Variance', 'Entropy Sum', 'Shape Index Mean', 'Shape Index Variance', 'Shape Index Sum', 'Pixel Intensity Mean', 'Pixel Intensity Variance', 'Pixel Intensity Sum','HOG Mean', 'HOG Variance', 'HOG Sum']

Feature Set #2

As exploratory work to find useful features, principal components were learned for each brain area such that each basis set would account for at least 60% of the explained variance within data for each brain area. Images are then projected onto each basis set and approximated with its principal components. Examples of this projection is shown in Figures 6 and 7.

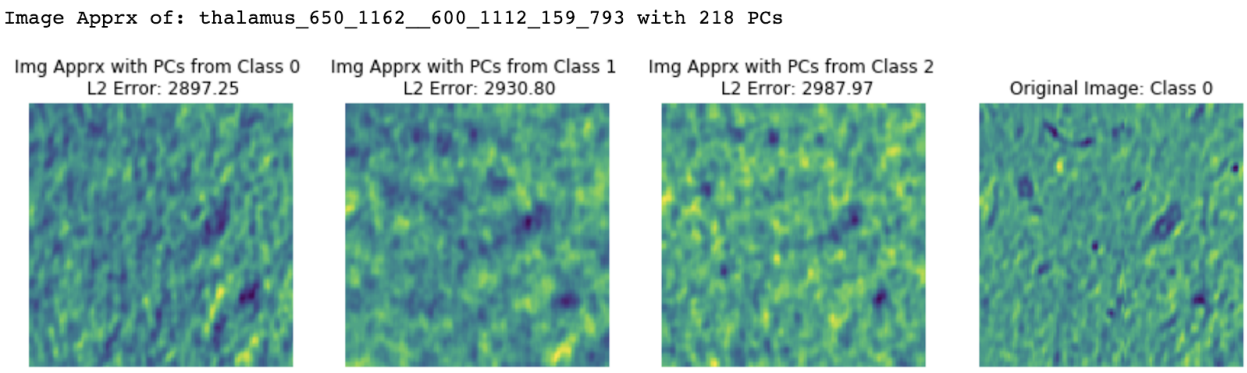


Figure 6: 128 x 128 Image Approximation

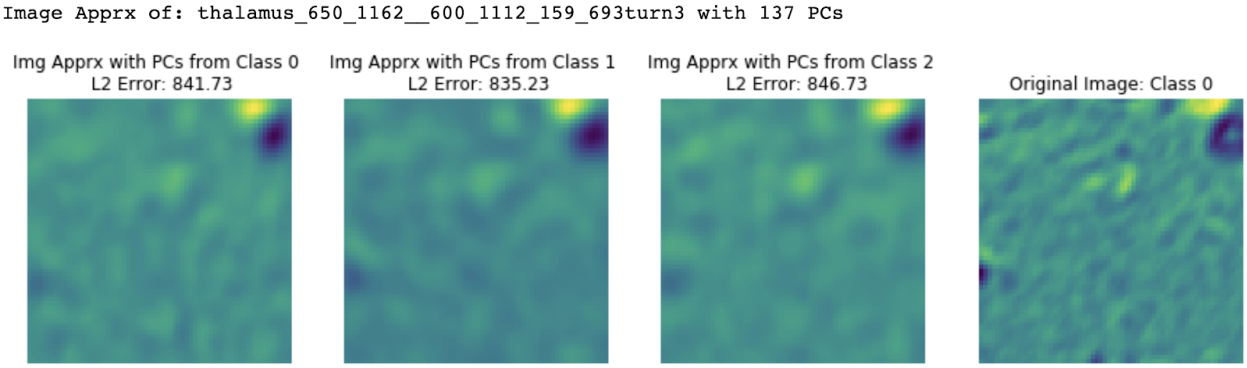


Figure 7: 64 x 64 Image Approximation

As seen in Figure 14, the principal components created by each brain area are plotted. Figure 15 shows a scatter plot of thalamus images aligning more with the principal components created from the same class. Figures 16 and 17 show the same for striatum and cortex respectively. The L2 Norm of the accuracy from the approximation made by each class' principal components (Equation 1) and the mean, standard deviation, and L2 norm of the coefficients of the projections were used as features.

The hypothesis is that images of a certain class will have a lower L2 norm of accuracy (error) from approximations made by that class' principal components than the L2 norms of error from approximations made by the other classes' principal components and that the alphas (coefficients of the image projections) will be different enough such that they can be used to differentiate brain areas (classes). Figure 18 shows the distribution of the L2 norm of the accuracy from the approximation made by each class' principal components. Figure 19 shows the distribution of the mean, standard deviation, and L2 norm of the coefficients of the projections onto each class' principal components.

Although an image could be perfectly approximated with k principal components where k is the size of the original data space (128^2=16,384 dimensions for 128x128 image), the number of principal components used to generate principal components is seen in Figures 20 and 21. A minimum of 60% variance can be accounted for across all 3 brain areas using 218 principal components for the original data of size 128x128 pixels and using 137 principal components for the augmented data of size 64 x 64 pixels. The number of principal components used to generate these features is a hyperparameter.

Features: ['Entropy Mean', 'Entropy Variance', 'Entropy Sum', 'Shape Index Mean', 'Shape Index Variance', 'Shape Index Sum', 'Pixel Intensity Mean', 'Pixel Intensity Variance', 'Pixel Intensity Sum', 'class', 'train', 'HOG Mean', 'HOG Variance', 'HOG Sum', 'Avg Alpha Class 0', 'Avg Alpha Class 1', 'Avg Alpha Class 2', 'Sum Alpha Class 0', 'Sum Alpha Class 1', 'Sum Alpha Class 2', 'L2 Norm Alpha Class 0', 'L2 Norm Alpha Class 1', 'L2 Norm Alpha Class 2', 'L2 Norm Accuracy Class 0', 'L2 Norm Accuracy Class 1', 'L2 Norm Accuracy Class 2']

Dimensionality Reduction

Dimensionality reduction was used to simplify the feature set and extract features producing optimal classification accuracy. Principal component analysis (PCA) was used to identify the n features that produce largest variance within the dataset. PCA takes the eigenvalue decomposition of the covariance matrix, containing the correlations between the features; the n-largest eigenvalues found represent the highest ranked features. Scaled PCA (sPCA) involves normalizing and scaling features so they have the properties of a standard normal distribution. Gaussian Random Projection (GRP) focuses on projecting points to a lower-dimensional space while preserving the distances between points. GRP creates a random matrix R using Gaussian distribution and projects R onto K dimensions. GRP is different from other dimensionality reduction techniques in that it trades a controlled amount of error for faster processing times.

In terms of implementation, PCA was evaluated using 3 and 5 components. Elementary cross validation was employed when selecting the gamma parameter for Kernel PCA for the feature sets. Separability was assessed using interactive 3D plots as well as component-by-component cross section visualization.

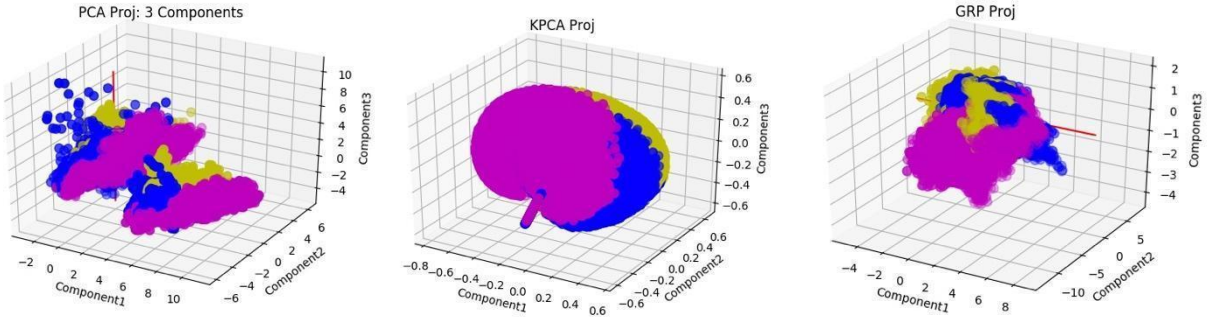


Figure 8. Reduction Visualized on Feature Set 1. PCA Right, KPCA Middle, GRP Right

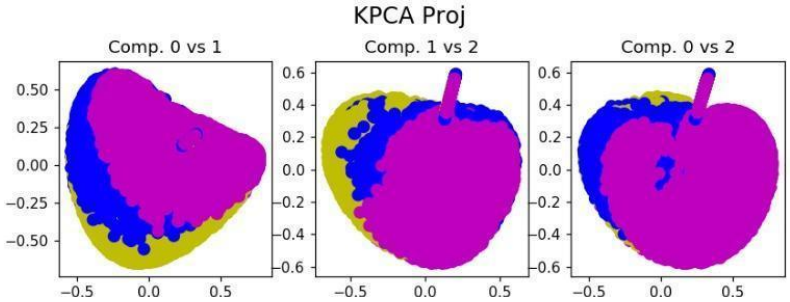


Figure 9. Reduction Cross Sections Visualized for Kernel PCA on Feature Set 1

Image Classification

Classifiers are characterized by 3 attributes: linear vs non-linear, parametric vs non-parametric, and discriminative vs. generative. Classifiers used with this image data set were chosen based on the first and second attributes listed previously. The projection of the image data into 3D space is seen in Figures 22 and 23. The underlying distribution of the data is unknown, and the data is nearly linearly separable. Therefore, non-linear non-parametric classifiers such as k-nearest neighbors, kernel support vector machine, and decision tree are expected to do better than linear parametric classifiers such as linear SVM and logistic regression.

The distribution K-nearest neighbors (kNN) classifier finds the k nearest points and determines the points class by calculating the most occuring class among its neighbors. Hyperparameter selection can be used with kNN to optimize the classifier. Decision trees provide a form of discriminative classification, meaning it relies more on data quality and statistics than making assumptions about the data. A decision tree is an iterative process that involves splitting the data into different partitions and branches. Logistic regression is based on the loss function for least squares regression and finds the coefficients, beta, that minimizes the least squares error between the input data and the predicted data. Logistic regression predicts the probability of outcomes and not the actual outcomes themselves. Linear support-vector machine (SVM) classifier is a supervised learning model that linearly separates the classes of data using hyperplanes. Kernel SVM classifiers here used the radial basis function (rbf) kernel. This classifier resulted in the highest accuracy out of all five classifiers. The formula for the K SVM model used is shown in equation 2 below.

$$K(X_1, X_2) = exponent(-\gamma \|X_1 - X_2\|^2)$$

Equation 2. Euclidean distance is represented by (X1-X2)

Results/Discussion

With a total of 5 different dimensionality reduction techniques and 5 different classifiers. Fig. 10 outlines the testing accuracy of all the five different classifiers for each of the five different dimensionality reduction steps. The highest average accuracy was prior to dimensionality reduction at 97.6%. Out of all dimensionality reduction methods, scaled PCA with 5 features came in first place with an average accuracy across all classifiers of 85.21%. KSVM was the best performing classifier in all five cases, achieving 97% test-set accuracy, as seen in Figure 10.

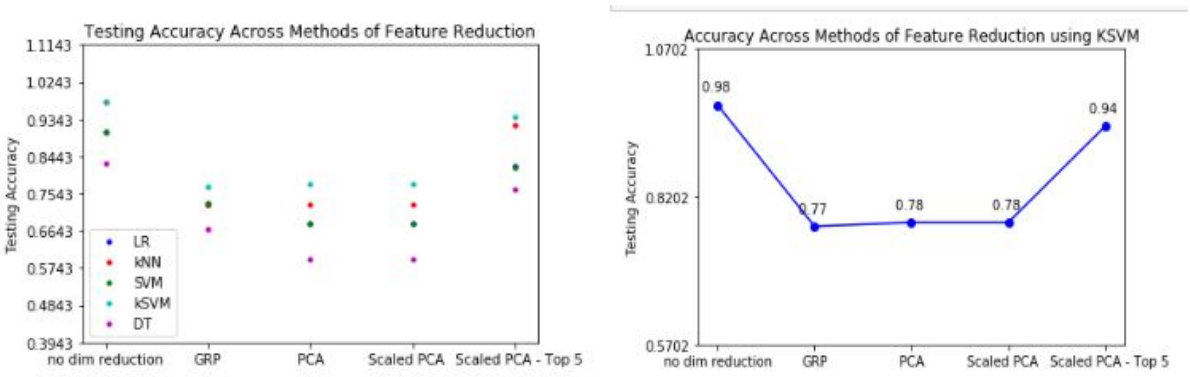


Figure 10: Testing Accuracy for Feature Set 1 across feature reduction methods for all five classifiers (left). Testing accuracies for Feature Set 1 using kSVM across dimensionality reduction techniques (right)

Shown in Figure 11, the testing accuracy for linear classifiers logistic regression and linear SVM improved significantly indicating that the additional features created from the principal components learned for each brain area increased the data's separability.

Classification Using Engineered Features

Name	Train Acc	Test Acc
LR	0.902670	0.905208
kNN	0.995310	0.976338
SVM	0.901679	0.903466
KSVM	0.979119	0.975286
DT	0.834239	0.826420

Classification Using Engineered Features

Name	Train Acc	Test Acc
LR	0.950530	0.952495
kNN	0.985806	0.963636
SVM	0.950917	0.953112
KSVM	0.985032	0.978407
DT	0.834239	0.826420

Figure 11: Best Accuracies Achieved are without Dimensionality Reduction Feature Set 1(Left). Feature Set 2 (additional features) (Right)

Further insight on the performance of kSVM, for the scaled PCA top 5 features, can be obtained from looking at the confusion matrix and ROC curve. The area under the ROC for each class is very close to 1 meaning that the model is able to distinguish between the positive and negative cases well. kSVM most correctly classified images belonging to class 3, and had the most difficulty with images belonging to class 2.

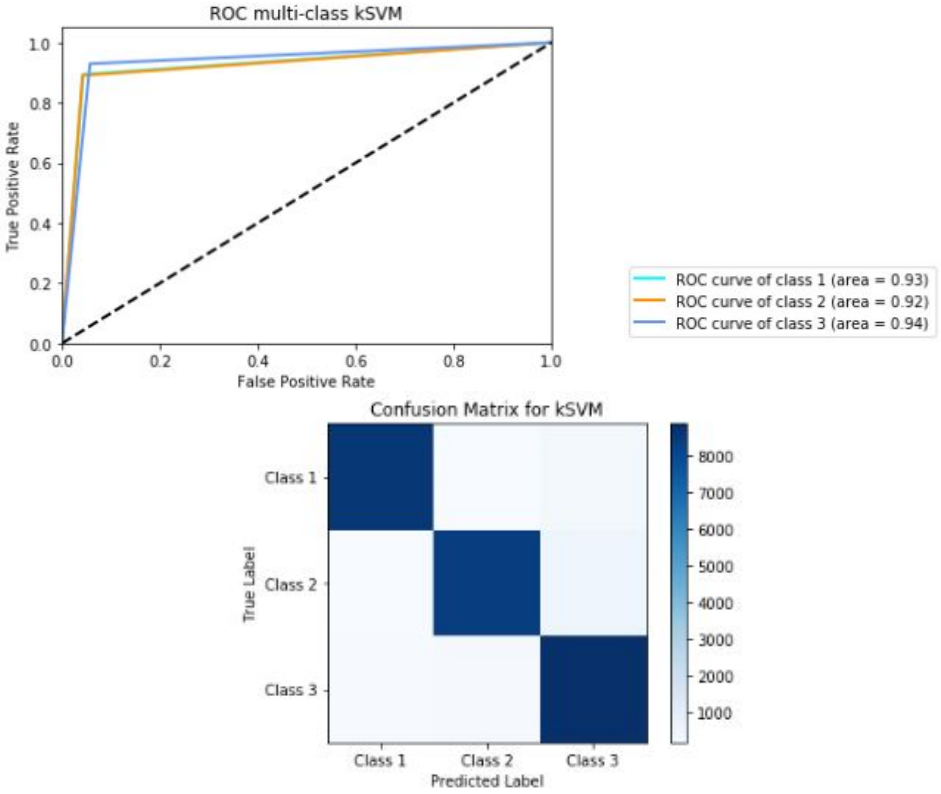


Figure 12: Multi-class Classification by the Best Performing Classifier Kernel SVM

All the data was split into a training or test set. A portion of the training data could have been hidden from the classifier to create a validation data set. Under the assumption that the collected data is representative of the future data, classifying on a validation set would determine if the models could generalize to new examples and therefore indicate if the models are too complex or not complex enough. Furthermore, hyperparameters could be optimized using the validation set.

Conclusion

The goal of this project was, given a high-resolution and multi-slice neuro-image, to automatically segment the image into distinct brain regions. In this work, the classifiers differentiated between the brain areas Hypothalamus, Striatum, and Cortex. To achieve this goal a pipeline was developed to accomplish data augmentation, feature selection, dimensionality reduction, visualization, classification, and error analysis. Results achieved exceeded 97% test-set accuracy, with optimal results stemming from Kernel SVMs and kNN classifiers using either no dimensionality reduction or PCA with 5 components.

Ultimately, the success of this work could potentially aid in automated neuro-image segmentation for large-scale neuro-images, alleviating manual annotation workloads for humans and speeding analytical pipelines.

Appendix:

Equations

Equation 1: L2 Norm of the Accuracy of the Image Approximation using Principal Components

$$\text{Accuracy} = \|y\text{-}\hat{y}\|_2$$

Additional Figures

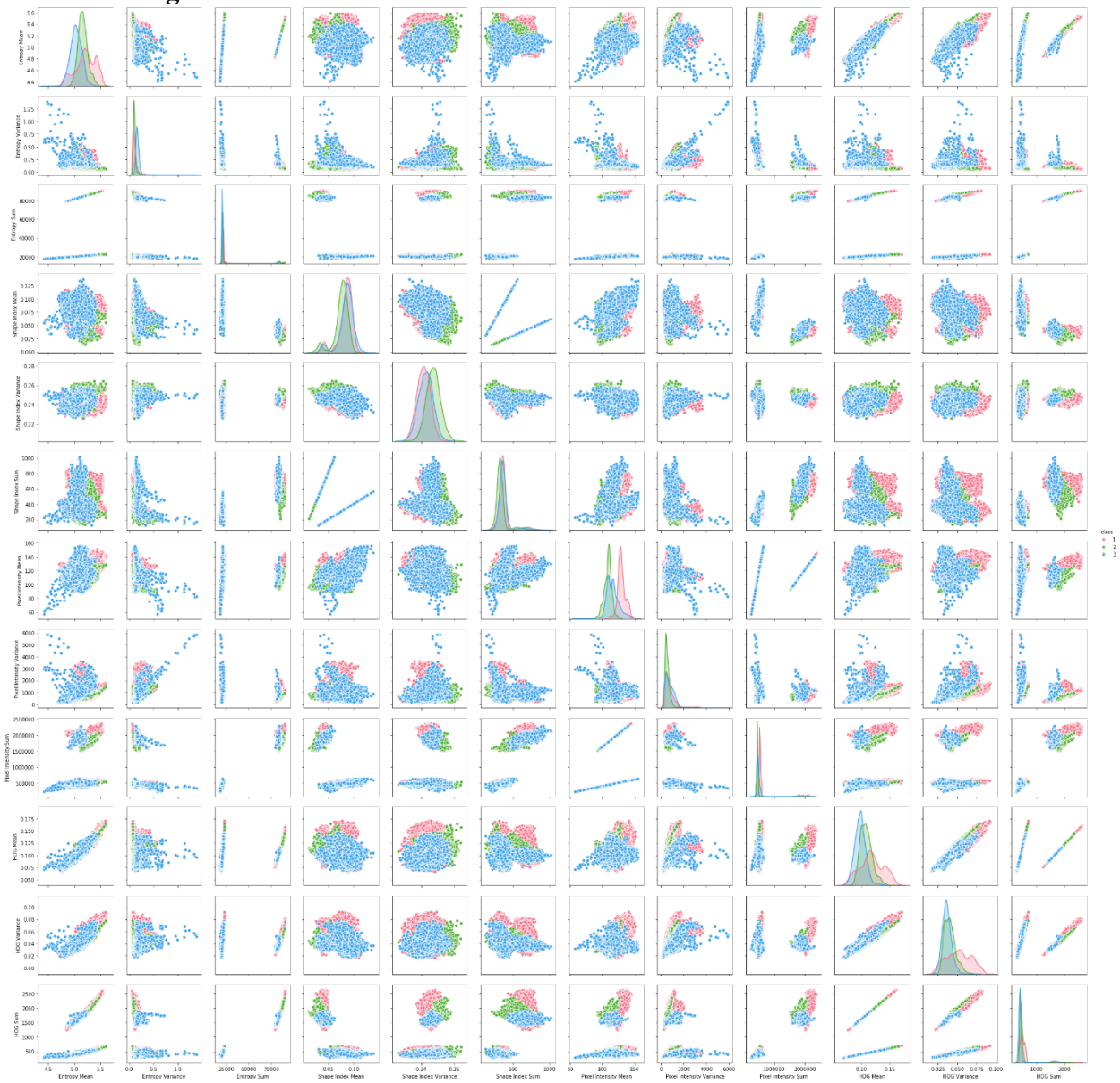


Figure 13: Pair Plot of Features from Filters

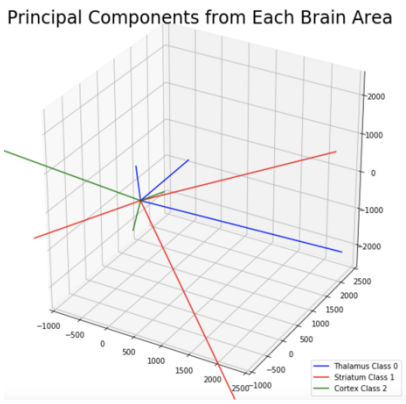


Figure 14: Basis Sets of Principal Components Learned from Different Brain Areas

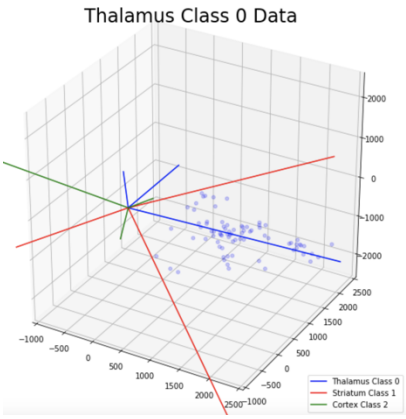


Figure 15: 3D Projection of Thalamus 128 x 128 Test Data Plotted with Principal Components

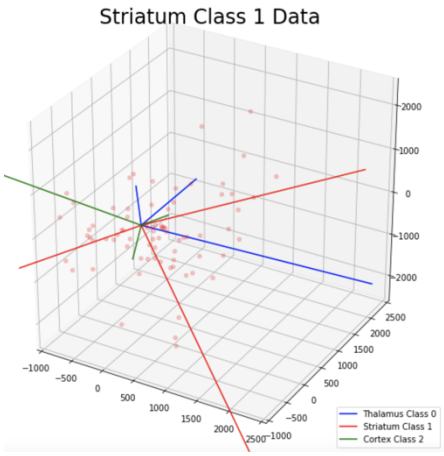


Figure 16: 3D Projection of Striatum 128 x 128 Test Data Plotted with Principal Components

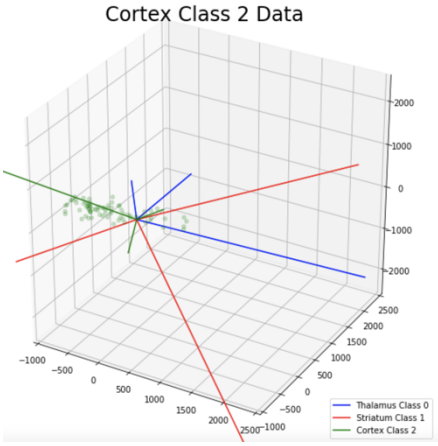


Figure 17: 3D Projection of Cortex 128 x128 Test Data Plotted with Principal Components

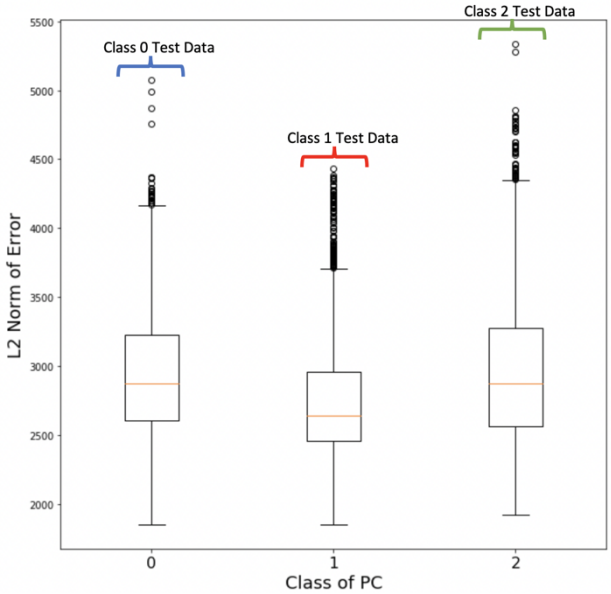
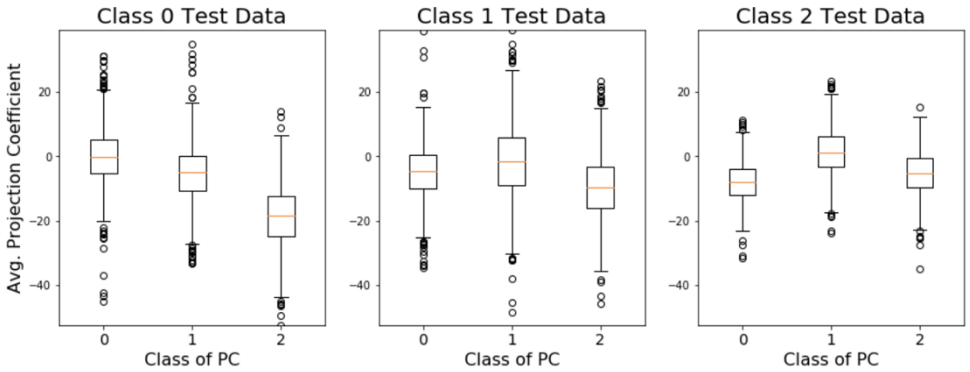


Figure 18: L2 Norm of Error $||y\text{-}y\_predicted||_2$ for All 128 x 128 Test Data



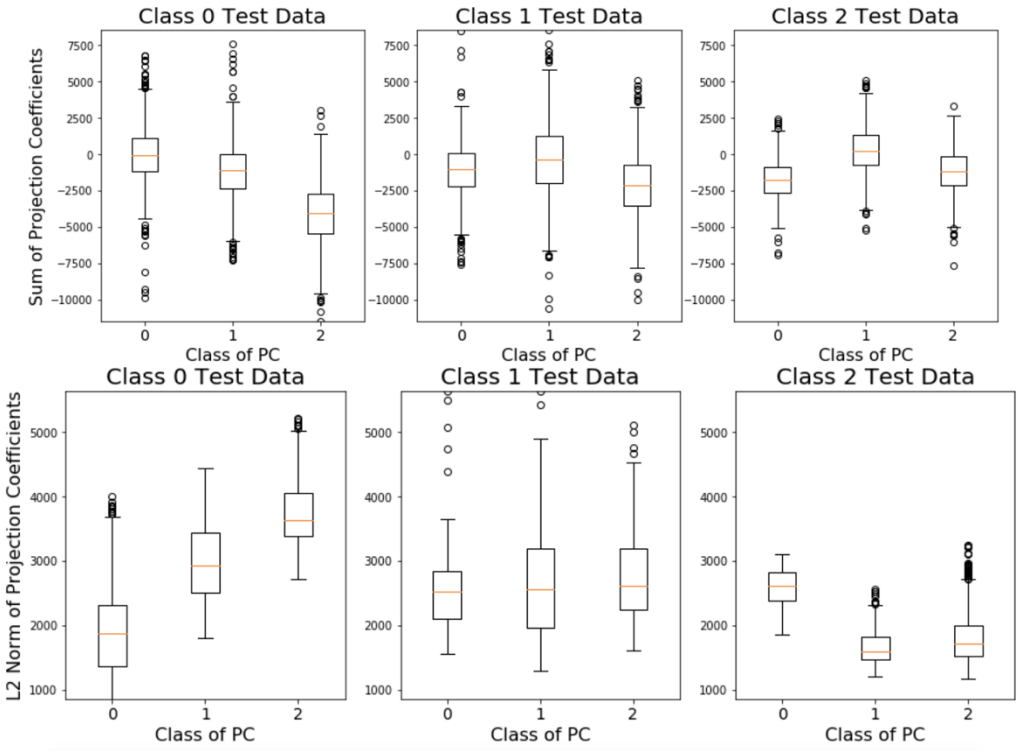


Figure 19: Distribution of Mean, Sum, and L2 Norm of the Coefficients used to Project the 128 x 128 Test Data onto PC Basis Set

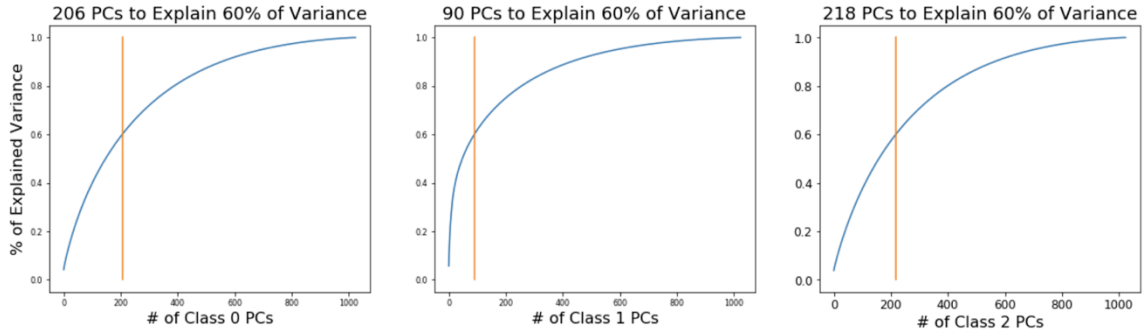


Figure 20: Number of Principal Components Needed to Explain 60% Variance in the 128 x 128 Image Data Set

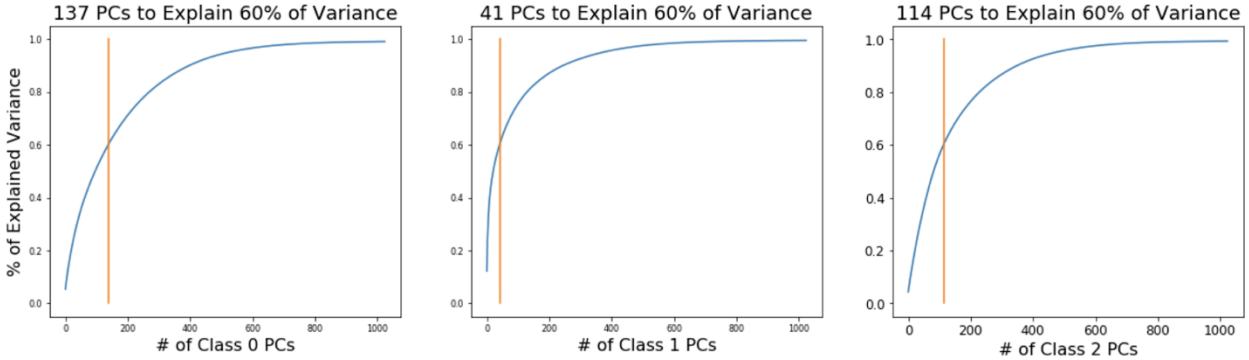


Figure 21: Number of Principal Components Needed to Explain 60% Variance in the 64 x 64 Image Data Set

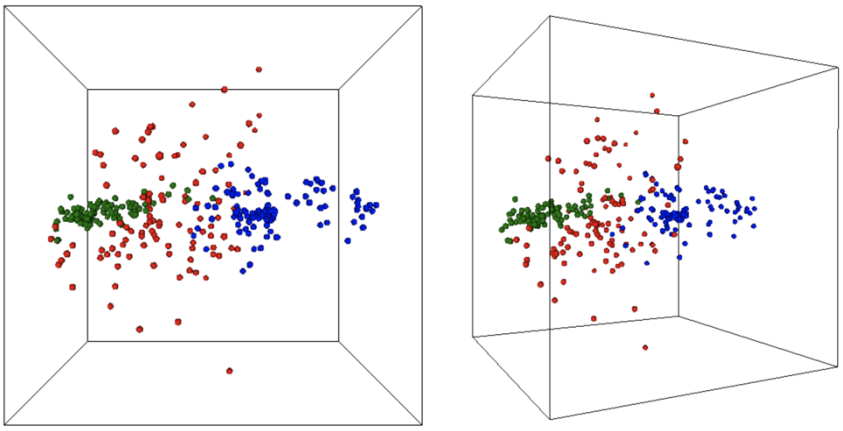


Figure 18: 128 x 128 Test Data Projected to a 3D Space
Front View (left) and Left View (right)

Figure 22: 128 x 128 Test Data Projected to a 3D Space in Left Side View

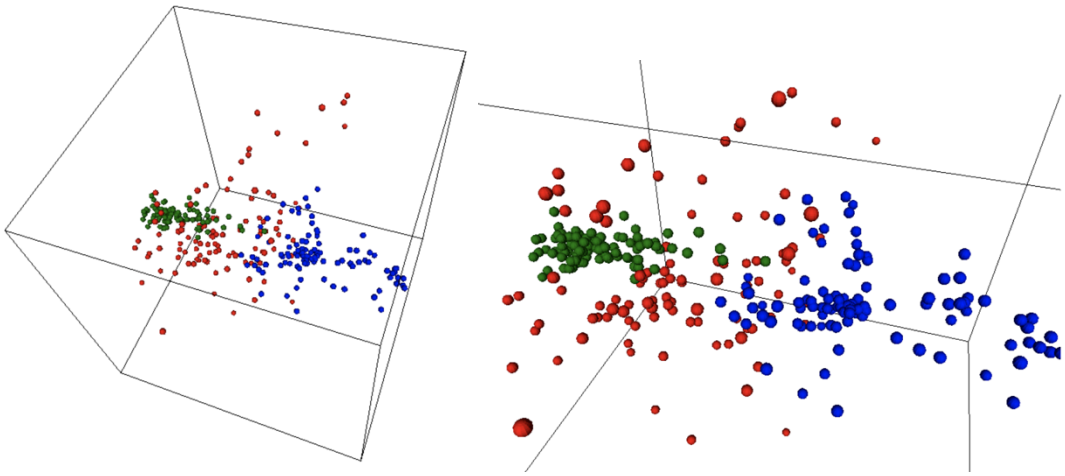


Figure 23: 128 x 128 Test Data Projected to a 3D Space in Top View (Left) and Zoomed (Right)

References

Hackeling, G. (2017). *Mastering machine learning with scikit-learn: learning to implement and evaluate machine learning solutions with scikit-learn*. Birmingham, UK: Packt Publishing Ltd.

Luis, Wang, & Jason. (2017, December 13). The Effectiveness of Data Augmentation in Image Classification using Deep Learning. Retrieved from https://arxiv.org/abs/1712.04621.

Canny edge detector¶. (n.d.). Retrieved from https://scikit-image.org/docs/dev/auto_examples/edges/plot_canny.html

Entropy¶. (n.d.). Retrieved from https://scikit-image.org/docs/dev/auto_examples/filters/plot_entropy.html.

Histogram of Oriented Gradients¶. (n.d.). Retrieved from https://scikit-image.org/docs/dev/auto_examples/features_detection/plot_hog.html.

Shape Index¶. (n.d.). Retrieved from https://scikit-image.org/docs/dev/auto_examples/features_detection/plot_shape_index.html.

Individual Contributions:

Hope:

- Feature extraction – Entropy, Pixel Intensity, Shape Index, Histogram of Oriented Gradients; mean, variance, sum for each feature.

- Made Poster and wrote some of paper

Cori:

- Feature Engineering (additional features added to create Feature Set 2)
- Figures 14-23
- Wrote Feature Set #2 portion of report
- Wrote part of results/discussion section
- Proofread and edited the full report

Heather:

- Feature extraction – Canny Edge Detection, Object Detection, feature statistics
- Dimensionality reduction – Instantiation, training, tuning five reduction techniques
- Visualization: 3D plots and cross sections for each transformation (both feature-sets all reduction methods)
- Classification – Instantiation, tuning, training, fitting, and testing of each classifier
- Result reports of each combination of feature sets, reduction methods, and classifiers

Lisa:

- Created original data frame that held a list of all the image file names
- Data augmentation – taking all original images and creating four sub-patch images, rotating images patches by 36° at a time to create 15 images total.
- Result analysis and visualization – looking at classification accuracy, precision, recall, F1 score and Cohens kappa score, creating individual confusion matrices and ROC curves, and creating summative figures to show accuracy changes across different methods

Erika:

- Took the information from the original images and created the baseline model. Used logistic regression to see how good we were before data augmentation.
- Compared how the final features relate to each other with a pair plot
- Researched the different filters to understand what each one delivered and explained it in the written report.