MACHINE LEARNING ON FMRI IMAGES

Team DataStars

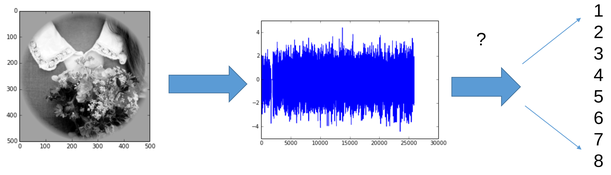
Kevin Kao, Ananth Subramaniam, Valerie Liu

University of California, Berkeley

{kkao, ananthsub, valliu}@cs.berkeley.edu

Problem

Given a stimulus image, can we predict which ROI in the brain will be most activated, i.e. have the highest mean BOLD response among the voxels in its region? Several modifications to the problem were made during our exploration of the data.



Tools

Some of the tools we used include:

Data Wrangling

All of the data wrangling was done in Python, which usually only involved applying various filters and feature extraction techniques on the images. We relied on packages like skimage, OpenCV, and mahotas to explore various ways of featurizing our data. One other package we considered was py-LogGabor, but the documentation was lacking so we decided against it.

Data Modelling

Here, we had to choose between using Python or Scala, scikit-learn or BIDMach. Python was enticing because of its familiarity, and since all our data exploration and wrangling work was done in Python. Scala with BIDMach had an obvious performance advantage. We tried both, but in the end, we stuck with scikit-learn due to some technical problems with transferring and reading in data from IPython and BIDMach. Additionally, our unfamiliarity with Scala made it difficult for us to manipulate the data with BIDMach.

matplotlib

SciPy NumPy

IP[y]: IPython

OpenCV

scikit learn

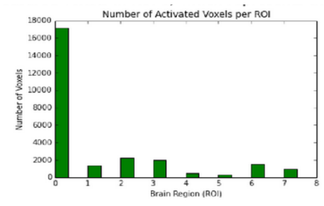
BID DATA

MATLAB SIMULINK

Methods

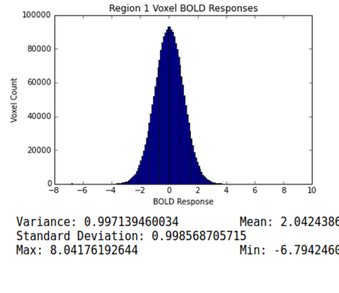
Data Exploration

Distribution of voxels among ROI



Region 0	Region 1	Region 2	Region 3	Region 4	Region 5	Region 6	Region 7
17127	1331	2208	1973	484	314	1550	928

Distribution of BOLD responses by ROI



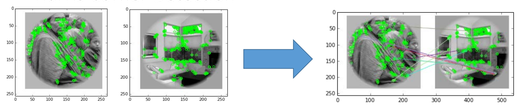

```
Variance: 0.997139460034          Mean: 2.04243862601e-17
Standard Deviation: 0.998568705715
Max: 8.04176192644                Min: -6.79424606241
```

Data Wrangling

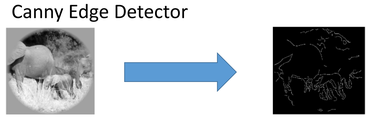
- Replaced NaN values from the BOLD responses with either 0.0 (resting state) or mean BOLD response of voxel

Feature Extraction/Engineering

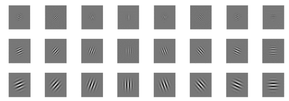
- Harris Corner Detection



- Canny Edge Detector



- Gabor Filter Banks
  - Applied at various spatial frequencies and orientations and drew out statistics.



Data Modelling

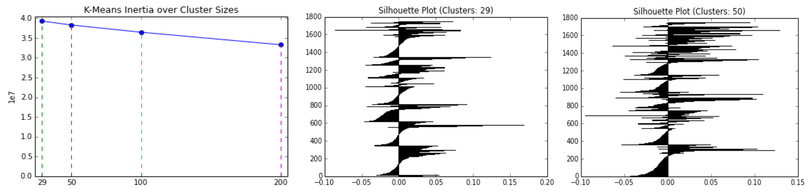
Here's a list of various techniques we tried, along with their corresponding problems.

- **K-Means Clustering:**
  - **Input:** the matrix of images to voxel BOLD responses
  - **Output:** clustered images
  - Essentially, we wanted to cluster images based on their BOLD responses to see if those with similar BOLD responses produced a pattern among the clustered images.
  - Attempted with several cluster sizes
- **Regression Models:**
  - **Linear Regression, SVC Regression, Decision Tree Regressor**
  - **Input:** a stimulus image
  - **Output**: a particular voxel's BOLD response
  - Given an image, the regression model should be able to predict a single voxel's response.
  - Need to train one model per voxel, i.e. approximately 25,000 models total
  - Initially tried with the pixels of the original full resolution 500x500 images as the features.
  - Later tried with features extracted from the images
- **Binary Classification Models:**
  - **Linear SVC, C-SVC, Decision Tree Classifier, Random Forest Classifier**
  - **Input**: an image
  - **Output**: the ROI that activates most after viewing the picture
  - Labels for each image were generated based on the highest mean BOLD response across ROI
  - One classifier per ROI – label 1 if most activated ROI is the same as classifier ROI, 0 otherwise

Results

1. K-Means:

Mostly could not notice any coherence among images in a cluster, even though voxel responses were similar within clusters.

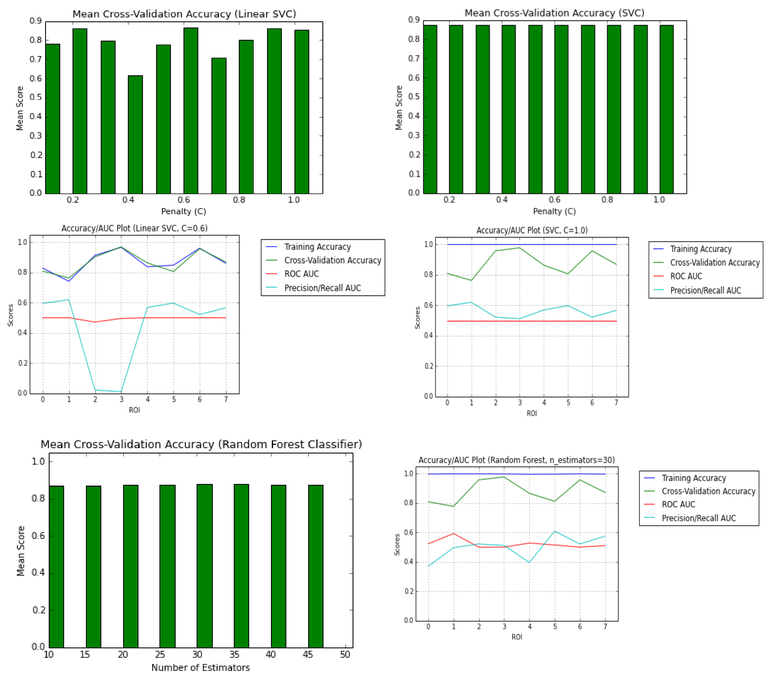


2. Regression Models

- Unexpectedly bad across the board, with negative cross-validation R-squared scores.
- Linear regression performed the worst, with low training and cross-validation scores.
- Support vector regression had a training score of 0.85.
- Decision tree regression had a training score of 0.93.

3. Binary Classification Models

- Much better performance than regression models, with accuracies up in 80-90% range



Lessons Learned

- Domain expertise is necessary for good feature extraction
- Feature engineering is difficult
- Machine learning requires a lot of time and resources
- Trying a variety of modelling techniques is good.