Machine Learning Project

WORD PREDICTION FROM FMRI OF BRAIN ACTIVITY

Author

Aaditya Aanand
Pulkit Sharma
Arvaman

• Problem Statement:

There is a dictionary of 60 words and there are 300 persons. Each word has an associated 218 – D feature vector where the features describe some human defined attributes. Each person is being shown a word from the dictionary along with its "Line Drawing" and then FMRI image of the brain is captured which is represented as a 21764 – D feature vector of voxel intensities. The goal is to predict which word (from two candidate words) the person was thinking about, given the FMRI image of their brain.

• Literature Review:

Following is the summary of a paper studied regarding collection of FMRI data:

Predicting the brain activity for the given noun: Generation of FMRI

This is a single-subject model which operates under the crucial assumption that **the brain activity for concrete nouns is a linear combination of contributions from each of its semantic features.**

To generate the FMRI for a noun, the model first maps the noun to a set of intermediate semantic features which are obtained from a Google text corpus where each feature is defined as the number of co-occurrences of that verb with the input noun in the text corpus. The final step generates FMRI as a weighted sum of brain images contributed by each of the semantic features. Specifically, the predicted activation Av at voxel v in the brain image for word w is given by:

$$Av = \sum_{i=1}^{n} Cv(i) fi(w)$$

fi(w) is the Co-Occurrence value of the ith semantic feature with the input word w, n is the number of features and $Cv(i)$ is a learned parameter that specifies the magnitude of activation of the ith intermediate semantic feature contributes to voxel v.

This way, there is a significant difference of accuracies between the word only and word picture experiment.

- **Methods Applied:**

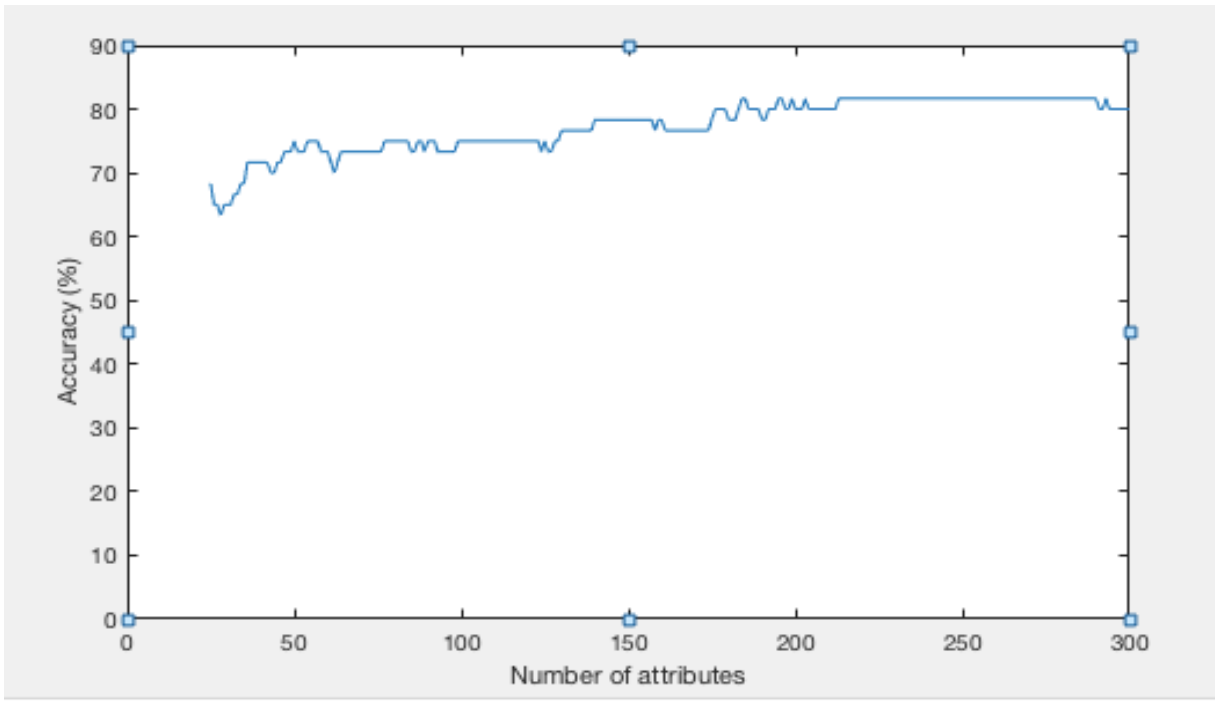
1. Dimensionality Reduction:

- **Low Variance Filter:** Removed all data columns than a given threshold chosen by plotting and choosing the low density region as the columns with little changes in the data carry very little information. The number of attributes reduced to about 10000 from 21764.

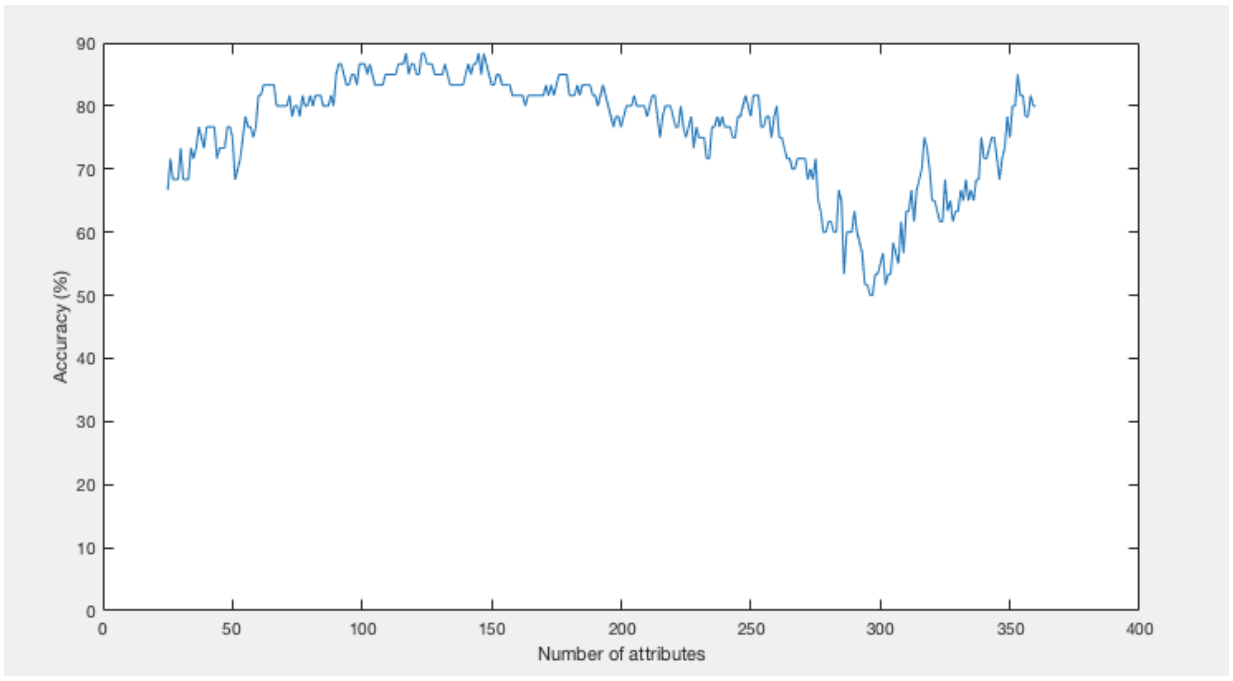
- **Singular Value Decomposition:** We made an assumption that the training and test datasets come from the same distribution as by SVD we could reduce the features to min (N, D) and projecting test data from its Eigen vectors would have reduced the attributes to less than 60 (not an ideal number as important attributes would then be missed). We thus projected test examples into the required dimensional space from the top eigenvectors of the training data. We employed SVD in the following two manners.

(a) **Training Data:** Consisted of 300 examples with around 10000 attributes (reduced via low variance filter). We plotted the number of attributes with the accuracy achieved by our best model and achieved 81.67% accuracy at features around 250.

(b) **Training & Test Data:** Both consisted of total 360 examples with 10000 attributes thus we got the possibility of reducing the attributes until 360. We again chose the top Eigen vectors from this combined dataset and plotted the number of attributes with our best model. We achieved an accuracy of 88.33% at around 117 attributes.



[Training Data]



[Training & Test Data]

- **Canonical Correlation Analysis (CCA):** We chose the 250 dimensional data derived by SVD and the semantic word feature vector of 218 dimensions (centered data) to compute pairs of canonical variables and projected data onto these canonical axes to give the highest correlation between these data. It can be computed by doing SVD on a correlation matrix.

2. Models Applied and Experimental Results:

Linear Regression

(a) Least Squared: Learned the weight vector but the matrix XX(transpose) was close to singular and thus the results of this model could not be trusted (achieved 66.67% accuracy).

(b) Kernelized Linear with Regularizer: Learned the weight vector and chose the value of the hyper parameter (λ=10) by plotting accuracy versus λ. We achieved an accuracy of 81.67% with SVD (training data) and the best accuracy of 88.33% with SVD (training and test data).

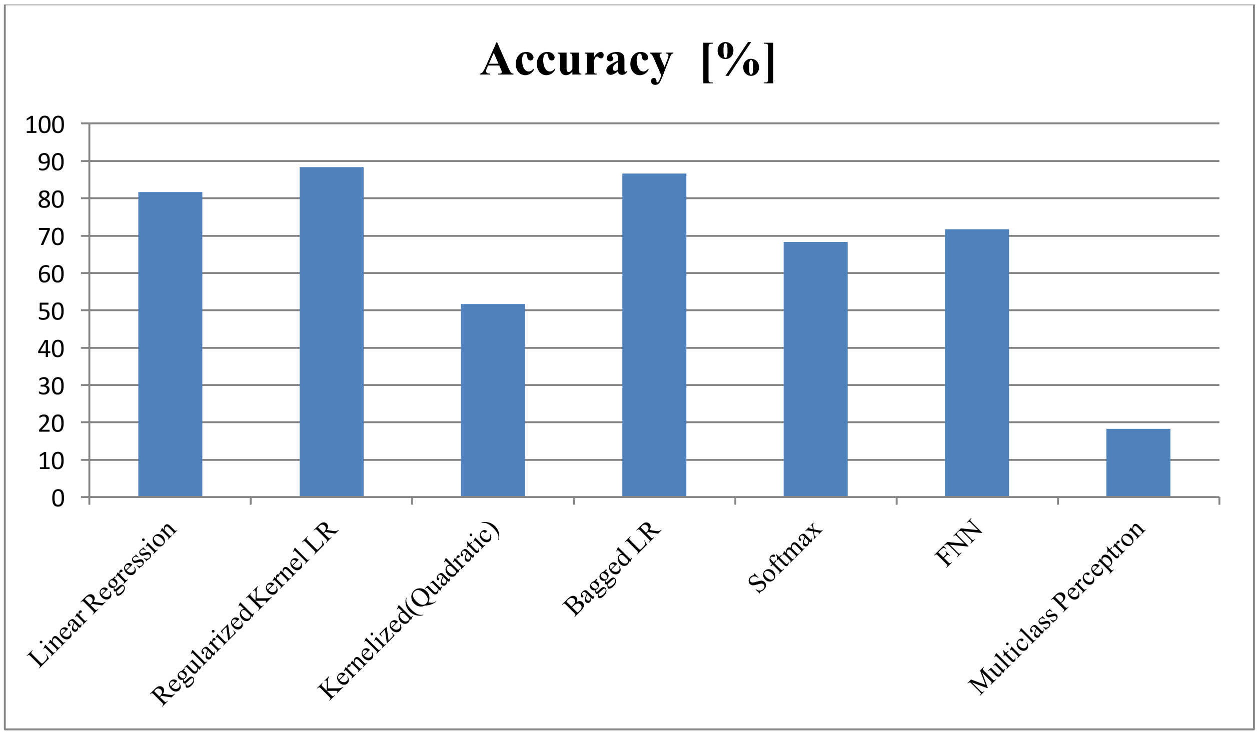
(c) Kernelized Quadratic & Cubic: Achieved accuracies of 51.67% for both the cases and thus easily could suggest a linear mapping (though very little test examples).

(d) Bagged Ridge Regression: Employed a couple of variations in bagging. First of all, we selected examples with replacement, made 6 copies and used an average model to reduce variances and achieved 86.67% accuracy. Secondly divided the training dataset into 6 parts (50 examples each) and used an averaged model to achieve 76.67% accuracy.

Multiclass Logistic or Softmax Regression: Assigned probabilities of belonging to a particular class by learning from data and chose the word based on higher probability of occurring in the candidate classes. Achieved an accuracy of 68.33%.

Feed Forward Neural Net: Used a single hidden layer and sigmoid activation function to achieve 71.67% accuracy.

<u>Multiclass Perceptron</u>: Predicted the exact class out of 60 classes i.e. the exact word and achieved 18.33% accuracy (probably low due to little examples and we predicted the exact word label rather than predicting from two words).



- **<u>Things Learnt in Project:</u>**

  - **Importance of Regularizer:** It comes to the rescue when the matrix becomes singular.

  - **Importance of Kernels:** Kernels can be helpful to identify non-linear mapping.

  - **Dimensionality Reduction:** It is very useful and efficient in many cases for extraction of information.