**Cardiac Arrest Classification Using Feature Reduction Technique**

**Illinois Institute of Technology**

**Machine Learning CS 584**

**Final Report**

***Priyank Shah (A20344797)***

***Shilpa Mandal (A20343042)***

**Abstract:**

In this paper, we have implemented machine learning techniques to classify cardiac arrhythmic arrest. We attempted to find better alternative to existing approach Neural Networks (BPNN back propagation neural network). The research is based on feature reduction PCA and extreme learn machine ELM, along with using techniques like KNN and Random Forest. We find that through this system, performance of existing algorithm can be improved in terms of comparatively lesser time and higher accuracy.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Introduction:**

Machine Learning has immense importance in solving practical problems in various domain of medicine, music, investment, recommendations, sentiment analysis, churn prediction etc. In our paper we focus on the application of different machine learning algorithm in the medical domain for cardiac arrhythmic classification collected from standard 12 lead ECG recordings. The aim is to distinguish between the presence and type of cardiac arrhythmia and to classify it in one of the 16 group.

**Dataset:**

The data contains records of patients comprising of clinical measurements, patient's information and cardiologist' decision. There are 279 features for 452 patient's record and target i.e cardiologist's diagnosis is either normal or one of 15 different classes of arrhythmia. 5 feature set are missing some values Among the 279datasets (that is 1.79% of overall data), hence we planned to eliminate those feature set from the process to achieve better efficiency.

In Classification, Class 01 refers to normal ECG. Class 02 refers to Ischemic changes, Class 03 to old anterior myocardial infarction, class 04 to old inferior myocardial infraction, class 05 to Sinus tachycardy, Class 06 to sinus brandycardy, class 07 to Ventricular Premature Contraction(PVC), class 08 to Supraventricular Premature Contraction, class 09 to Left bundle branch block,class 10 to Right bundle branch block. Class 11 to 14 refers to degree Atrio Venticular block, class 15 to Flutter, and class 16 refers to the rest. Arrhythmic Dataset

**Approach:**

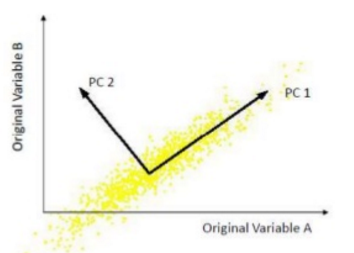
In this research we have tried classify arrhythmia cardiac arrest into the 16 categories by applying machine learning algorithm like PCA (principal component analysis) for feature reduction and further using the reduced features to build various models using ELM (extreme learn machine), KNN (K Nearest Neighbor) and RF (Random Forest).

***Principal Component Analysis:***

Principal Component Analysis is a linear projection method to reduce the number of parameters by mapping the features into a space of lower dimensionality.

In this form of unsupervised learning the subset of original dataset i.e the principal components (linear combination of original features) capture the maximum variance of the data in the low-dimensional representation. These are uncorrelated with one another and are orthogonal in original dimension space.

The first principal component is the direction of greatest variability(covariance) in the data. Second is the next orthogonal (uncorrelated) direction of greatest variability, and so on. Projection along PC1 discriminate the data most along any one axis.



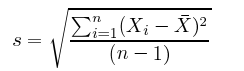
In PCA, covariance (or correlation) matrix is split into scale part (eigen values) and direction part (eigenvectors).

Mean is simply the average value of all x’s in the set X, which is found by dividing the sum of all data points by the number of data points, n.



Subtracting the mean from each of the data dimensions makes covariance calculation easier and helps in center and scaling the features.

Standard deviation, as fun as that sounds, is simply the square root of the average square distance of data points to the mean.



The data is normalized by subtracting mean from it and further dividing by standard deviation.

When a matrix performs a linear transformation, eigenvectors trace the lines of force it applies to input; when a matrix is populated with the variance and covariance of the data, eigenvectors reflect the forces that have been applied to the given. The eigen value denotes the amount of variability captured along that dimension. Some information is lost in feature reduction but if the eigenvalues are small we don't loose much information.

There is minimum loss is the dimensional space is spanned by eigenvectors and ordered by eigen values.





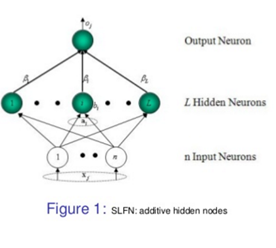
Here  is the eigenvector of S matrix. The eigenvectors are found from the covariance matrix and are next ordered by eigen value. For n original dimensions, correlation matrix is nxn and has up to n eigenvectors so the number of PCs is n.

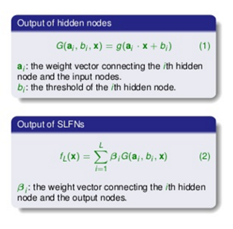
Ignoring the components of lesser significance based on the eigenvalues the first top components are chosen.

Applying this technique we reduced the number of features from 278 to 79.

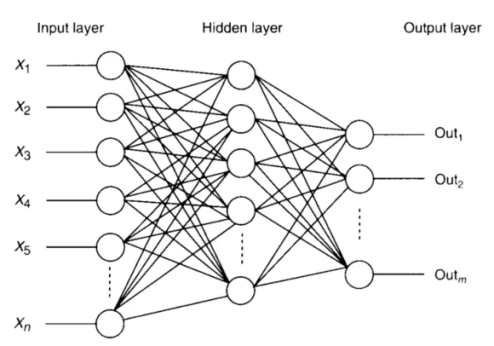
***Extreme Learn Machine:***

Gradient descent based methods are mainly used in many learning algorithm of feed forward neural networks. Traditionally, all parameters of the feed forward need to tune iterative and require a very long. time to learn. The application of general gradient based learning method like back propagation neural network is in practice slower due to improper learning steps or easily converge to local minima. Popular machine learning techniques like Neural Network (NN) and support vector machine (SVM) face some challenging issues such as: intensive human intervene, slow learning speed, poor learning scalability. In our research we try to implement ELM algorithm using PCA.



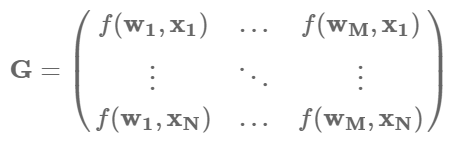
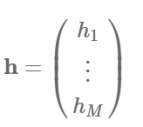
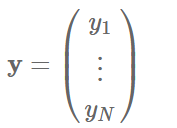


*Extreme Learning Machine*, is an easy way of training Artificial Neural Networks with a hidden layer. ELM was proposed by [Huang et al., 2006]. It is used in a multilayered structure with one neural hidden layer (Single Layer Feedforward Network, SLFN). The first step is to initialize at random the weights connecting the input and the hidden layer. Thus, it will only be necessary to optimize the weights connecting the hidden layer and the output layer. In order to do this, the *Moore-Penrose pseudoinverse* matrix will be used.

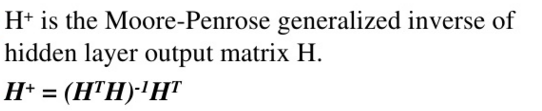


ELM substantially decreases the computational time spent in adjusting the weights since it does not use any search method for fitting the hidden weights. Instead, as we will explain below, the Moore-Penrose pseudo inverse matrix is a simple and fast algebraic calculation.

The equation 1 can also be expressed as y=G⋅h where ***h*** is the vector of weights connecting the hidden and output layer, ***y*** is the output vector and ***G*** can be calculated as:

 ,, 

ELM are single layered [feed forward neural network](https://en.wikipedia.org/wiki/Feedforward_neural_network) for [classification](https://en.wikipedia.org/wiki/Statistical_classification) or [regression](https://en.wikipedia.org/wiki/Regression_analysis). Here where the weights connecting inputs to hidden nodes are randomly assigned and never updated. These weights between hidden nodes and outputs are learned in a single step, which essentially amounts to learning a linear model.



Thus we see ELM follows three main steps :

|  |
| --- |
|  |



In the above equation,

 represent the *i*th input vector and output vector, *b* is bias of *i*th hidden neuron and input weight vector connecting *i*th hidden

neuron to input layer.

Each component of **H** represent output of hidden layer.

When input weights *i* **w** and biases *i b* of hidden neuron are

invariable, **H** is determined with input vector *j* **x** . In that case

SLFNs are linear system. So, In case of **H** has inverse matrix,

we can get **β** through **H** inverse ⋅**T**

The Moore-Penrose generalized inverse matrix is application of linear algebra theorem. For a general linear system Ax=y, we say is least square solution if



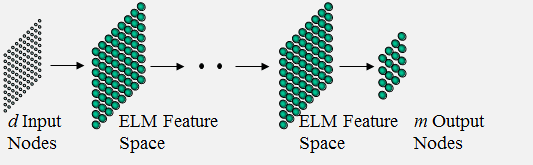
The mathematical model can be written as,



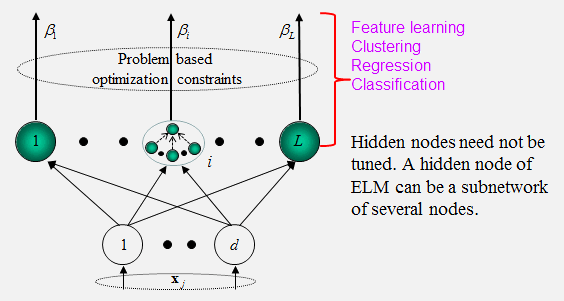
After choosing 1) the arbitrarily value for input weightsand

biases of hidden neurons, 2) the hidden layer output matrix is calculated **H** and finally 3) the optimal is found using the above equation.

ELM actually fills gaps among them and proposes that it needn’t have different learning algorithms for different SLFNs if universal approximation and classification capabilities are considered.



 ELM considers multi-hidden-layer of networks as a white box and trained layer-by-layer. Different from Deep Learning, all the hidden neurons in hierarchical ELM as a whole are not required to be iteratively tuned. ELM theory was confirmed in biological systems in 2013.



\

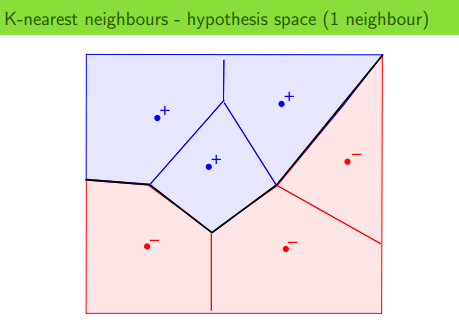
**Advantages and disadvantages of the ELM algorithm:**

Elm needs less training time compared to BP and SVM. The prediction performance of ELM is usually the bit better then these techniques. As mentioned earlier, ELM is much faster than the traditionally used gradient descent .This is due to optimal coefficients, which are calculated using an algebraic calculation instead of an iterative stochastic method. The challenges are 1) how to find optimal solution. 2) Problem of local minima or easy fitting.

ELM problems, in which the parameters of hidden nodes are assigned randomly and the desired output weights can be determined analytically, are suitable for being outsourced to the cloud.

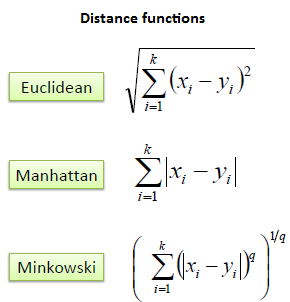
***K Nearest Neighbor:***

KNN is a simple and lazy learning or instance based learning algorithm where the function is only approximated locally and all computation is deferred until classification. It is used for both regression and classification. A non parametric model is one that can not be characterized by a fixed set of parameters. A family of non parametric models is Instance Based Learning. Instance based learning is based on the memorization of the dataset.



It stores all available cases and classifies new cases based on similarity matrix. KNN is conceptually simple, the able to solve complex problems. It basically assumes that all instances are points in n dimensional space. A distance measure is needed to determine the "closeness" of instances. Then an instance is classified by finding its nearest neighbors and by picking the most popular class among the neighbors. It can be useful to assign weight to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/*d*, where *d* is the distance to the neighbor.

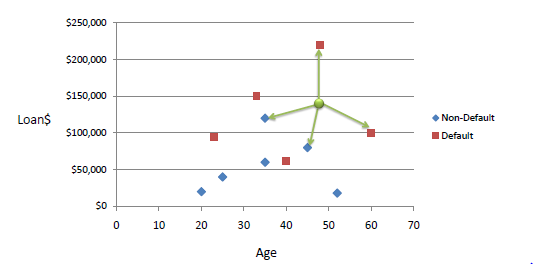
|  |
| --- |
|  |
| A case is classified by a majority vote of its neighbors, with the  case being assigned to the class most common amongst its K  nearest neighbors measured by a distance function. If K = 1,  then the case is simply assigned to the class of its nearest neighbor. |

 all

A commonly used distance metric for [continuous variables](https://en.wikipedia.org/wiki/Continuous_variable) is [Euclidean distance](https://en.wikipedia.org/wiki/Euclidean_distance). For discrete variables, such as for text classification, another metric can be used, such as the overlap metric (or [Hamming distance](https://en.wikipedia.org/wiki/Hamming_distance)).

|  |
| --- |
| It should also be noted that all three distance measures are only  valid for continuous variables. In the instance of categorical variables the Hamming distance must be used. It also brings up the issue of standardization of the numerical variables between 0 and 1 when  there is a mixture of numerical and categorical variables in the data  set. |
|  |

As parameter selection best choice of k depends upon the data. Choosing the optimal value for K is best done by first inspecting the data. In general, a large K value is more precise as it reduces the overall noise but there is no guarantee. Cross-validation is another way to retrospectively determine a good K value by using an independent dataset to validate the K value. generally, larger values of k reduce the effect of noise on the classification, but make boundaries between classes less distinct. A good k can be selected by various [heuristic](https://en.wikipedia.org/wiki/Heuristic_(computer_science)) techniques



In the above example of data concerning credit default, Age and Loan are two numerical variables (predictors) and Default is the target. Using KNN they are being classified into two groups default and non-default.

[Feature extraction](https://en.wikipedia.org/wiki/Feature_extraction) and dimension reduction can be combined in one step using [principal component analysis](https://en.wikipedia.org/wiki/Principal_Component_Analysis) (PCA), [linear discriminant analysis](https://en.wikipedia.org/wiki/Linear_discriminant_analysis) (LDA), or [canonical correlation analysis](https://en.wikipedia.org/wiki/Canonical_correlation) (CCA) techniques as a pre-processing step, followed by clustering by k-NN on [feature vectors](https://en.wikipedia.org/wiki/Feature_(machine_learning)) in reduced-dimension space. In [machine learning](https://en.wikipedia.org/wiki/Machine_learning) this process is also called low-dimensional [embedding](https://en.wikipedia.org/wiki/Embedding).

In k-NN regression, the k-NN algorithm is used for estimating continuous variables. One such algorithm uses a weighted average of the k nearest neighbors, weighted by the inverse of their distance. This algorithm works as follows:

1. From the queried example to the labeled examples Compute the Euclidean or [Mahalanobis distance](https://en.wikipedia.org/wiki/Mahalanobis_distance).

2. Order the labeled examples by increasing distance.

3. Find a heuristically optimal number k of nearest neighbors, based on [RMSE](https://en.wikipedia.org/wiki/RMSE). This is done using cross validation.

4. With the k-nearest multivariate neighbors Calculate an inverse distance weighted average.

|  |
| --- |
|  |

|  |
| --- |
|  |

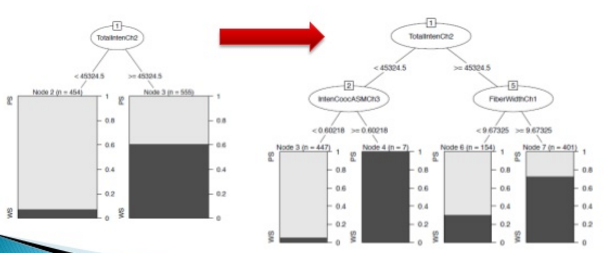
**Advantages and disadvantages of the KNN algorithm:**

The cost of the learning process is practically none. No assumptions about the characteristics of the concepts to learn have to be done. Complex concepts can be learned by local approximation using simple procedures

One major drawback in calculating distance measures directly from the training set is in the case where variables have different measurement scales or there is a mixture of numerical and categorical variables. One solution is to standardize the training set. If Using the standardized distance on the same training set, the unknown case returned a different neighbor then it won't not a good sign of robustness. It is computationally expensive to find the k nearest neighbors when the dataset is very large. Performance depends on the number of dimensions that we have (curse of dimensionality).

***Random Forest:***

Decision Trees Learning is one of the most widely used and practical methods for inductive inference and is an important tool in machine learning or predictive analytics. It recursive partitioning models are decision support tool which uses a tree like graph of decisions and their possible consequences.



Random Forest are an [ensemble learning](https://en.wikipedia.org/wiki/Ensemble_learning) method for [classification](https://en.wikipedia.org/wiki/Statistical_classification), [regression](https://en.wikipedia.org/wiki/Regression_analysis) and other tasks, that operate by constructing a multitude of [decision trees](https://en.wikipedia.org/wiki/Decision_tree_learning) at training time. The class that is the [mode](https://en.wikipedia.org/wiki/Mode_(statistics)) of the classes (classification) or mean prediction (regression) of the individual trees are the predictions .Those predictions are then combined into a single (mega) prediction that should be as good or better than the prediction made by any one classifer.

*Random* forest can handle a large number of features, and it's helpful for estimating which or your variables are important in the underlying data being modeled. To classify an observation in a dataset a decision tree is composed which comprises of a series of decisions. The algorithm to induce a random forest will create a bunch of random decision trees automatically. The training algorithm for random forests applies the general technique of [bootstrap aggregating](https://en.wikipedia.org/wiki/Bootstrap_aggregating), or bagging, to tree learners.

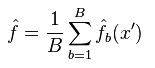
|  |
| --- |
|  |

This bootstrapping procedure leads to better model performance because it decreases the [variance](https://en.wikipedia.org/wiki/Bias%E2%80%93variance_dilemma) of the model, without increasing the bias. This means that while the predictions of a single tree are highly sensitive to noise in its training set, the average of many trees is not, as long as the trees are not correlated. Simply training many trees on a single training set would give strongly correlated trees (or even the same tree many times, if the training algorithm is deterministic); bootstrap sampling is a way of de-correlating the trees by showing them different training sets.

The training algorithm for random forests applies the general technique of [bootstrap aggregating](https://en.wikipedia.org/wiki/Bootstrap_aggregating), or bagging, to tree learners. Given a training set X = x1, ..., xn with responses Y= y1, ..., yn, bagging repeatedly (B times) selects a [random sample with replacement](https://en.wikipedia.org/wiki/Bootstrapping_(statistics)) of the training set and fits trees to these samples:

|  |
| --- |
|  |

After training, predictions for unseen samples x' can be made by averaging the predictions from all the individual regression trees on x' or by taking the majority vote in the case of decision trees



This is an very effective technique of decreasing the [variance](https://en.wikipedia.org/wiki/Bias%E2%80%93variance_dilemma) of the model, without increasing the bias resulting bootstrapping procedure leads to better model performance.

Random forest relationship with KNN

Random Forest and KNN both can be viewed as so-called weighted neighborhoods schemes. These are models built from a training set \{(x_i, y_i)\}_{i=1}^n that make predictions \hat{y} for new points x' by looking at the "neighborhood" of the point, formalized by a weight function W:



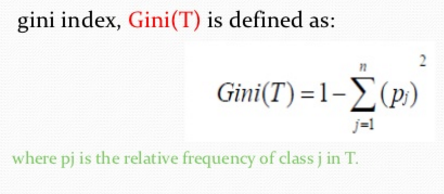
Here, W(x_i, x') is the non-negative weight of the i'th training point relative to the new point x'. For any particular x', the weights must sum to one. Weight functions are given as follows:

* In *k*-NN, the weights are W(x_i, x') = \frac{1}{k} .if *xi* is one of the *k* points closest to *x'*, and zero otherwise.
* In a tree, W(x_i, x') = \frac{1}{k'} if *xi* is one of the *k'* points in the same leaf as *x'*, and zero otherwise.
* Since a forest averages the predictions of a set of *m* trees with individual weight functions W_j, its predictions are

\hat{y} = \frac{1}{m}\sum_{j=1}^m\sum_{i=1}^n W_{j}(x_i, x') \, y_i = \sum_{i=1}^n\left(\frac{1}{m}\sum_{j=1}^m W_{j}(x_i, x')\right) \, y_i.

This shows that the neighborhood of *x'* depends in a complex way on the structure of the trees, and thus on the structure of the training set.

Every time a split of a node is made on variable m the gini impurity criterion for the two descendent nodes is less than the parent node. Adding up the gini decreases for each individual variable over all trees in the forest gives a fast variable importance that is often very consistent with the permutation importance measure.



Proximities are one of the most useful tools in random forests. The proximities originally formed a NxN matrix. After a tree is grown, put all of the data, both training and oob, down the tree. If cases k and n are in the same terminal node increase their proximity by one. At the end, normalize the proximities by dividing by the number of trees.

For estimating the performance for classification the common metric accuracy is used. However when the classes are not balanced other metrics need to be considered. One of them is Kappa statistics. It takes into account the expected error rate.



where O is the observed accuracy and E is the expected accuracy under chance agreement.

[Random forest](http://en.wikipedia.org/wiki/Random_forest) is a highly versatile machine learning method with numerous applications ranging from marketing to healthcare and insurance. It can be used to [model the impact of marketing](http://epubl.ltu.se/1653-0187/2008/014/LTU-PB-EX-08014-SE.pdf) on customer acquisition, retention, and churn or to [predict disease risk and susceptibility](http://www.biomedcentral.com/1472-6947/11/51) in patients.

**Advantages and Disadvantages of Random Forest:**

It has methods for balancing error in class population unbalanced data sets. Generated forests can be saved for future use on other data. Prototypes are computed that give information about the relation between the variables and the classification. It computes proximities between pairs of cases that can be used in clustering, locating outliers, or (by scaling) give interesting views of the data. The capabilities of the above can be extended to unlabeled data, leading to unsupervised clustering, data views and outlier detection. It offers an experimental method for detecting variable interactions.

Random forests have been observed to over fit for some datasets with noisy classification/regression tasks. For data including categorical variables with different number of levels, random forests are biased in favor of those attributes with more levels. Therefore, the variable importance scores from random forest are not reliable for this type of data.

**Result and Discussion:**

In this experiment, I verified the result with 10-fold cross validation. 9 part of dataset were used for training purpose and 1 part of dataset was used for testing purpose. Data is randomly shuffle in order to achieve better results.

Knn algorithm took aprx 11 to 12 second to compute and shows results with average accuracy of 55 %, while implementing the Knn with PCA helped to improve the accuracy till 62% and timing by 10th time of Knn execution time. As per case paper, VF-15 shows to achieve the accuracy of 62%, the best as per their experiments with different algorithms, that’s what we achieved with knn with PCA algorithm. Main purpose of this experiment is to convert higher dimension dataset into lower dimension dataset to achieve more accuracy and decrease the timing, we were successful to demonstrate those advantages. PCA helped to reduce timing while maintaining the almost equal level of accuracy.

Here are the screen-shots which indicated the result of Knn and Knn with PCA:

|  |
| --- |
|  |



**Conclusion:**

In a nutshell, we conclude that PCA helps in significant reduction of execution time and also increase the accuracy of result. Due to time constraint we were unable to implement VF15, but we did the case study and from the research paper we followed, describe that it has achieved the accuracy of 62%. In our case, with Knn without feature reduction we were able to get accuracy around 42% with the random shuffle of data. With the introduction to PCA in Knn algorithm, accuracy increased till 65% and execution time reduced to 10 times lower. Principal Component Analysis has remarkable learning speed and classifying performance.

**Future Scope:**

In future we can implement Component Reduction technique on neural network to see if there are any positive changes in outcomes. There are certainly room for development. We can use deep learning method to generate new features.

**References:**

[1] <http://www.saedsayad.com/k_nearest_neighbors.htm>

[2] <https://docs.python.org/2/library/operator.html>

[3] <http://blog.yhat.com/posts/random-forests-in-python.html>

[4] <http://docs.scipy.org/doc/numpy-1.10.0/reference/generated/numpy.linalg.svd.html>

[5]<http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=4353024>

[6] <https://www.youtube.com/watch?v=LIPtRVDmj1M#t=3.020476>

[7] <http://pythonhosted.org/milk/randomforests.html>

[8] http://elm.readthedocs.org/en/latest/usage.html