Unsupervised Learning Methods for Handwritten Digit Classification and Facial Recognition

Michael Walton
Department of Computer Science, Georgia Institute of Technology

November 9, 2015

Abstract

In this work we explore two labeled image datasets from an unsupervised learning perspective. We apply various dimensionality reduction techniques and visualize the transformed features extracted by theses algorithm. Specifically, we utilize Principal Components Analysis (PCA), Independent Components Analysis, Gaussian Random Projections, and Feature Agglomeration. The 'expressiveness' of the components are discussed in terms of the degree of the data's variance captured. Next we explore clustering in the high-dimensional spaces formed by taking the raw pixel intensities as feature dimensions; We compare observations from these methods with clustering using k-means and Expectation Maximization (EM) of projected versions of the data using the proxy features generated by dimensionality reduction transformations. Finally we apply all these methods as preprocessing for a supervised image classification problem using artificial neural network and compare the effects of each method.

1 Introduction

The dimensionality reduction algorithms considered in the present study are Principal Components Analysis (PCA), Independent Components Analysis, Gaussian Random Projections, and Feature Agglomeration. The clustering algorithms considered are Expectation Maximization and k-means. Neural Network models are implemented using the `sk-nn` wrapper module for pylearn2 and `Theano`. Implementations are based on the `scikit-learn` python library [3] The python modules `scipy, numpy, pandas` and `scikit-learn` [1] [2] are used extensively throughout the implementation for experiment scripting, analysis and visualization.

2 Methods

In the sections that follow, we will provide a brief overview of the handwritten digits and the Olvetti faces datasets used in this study. In addition, we will briefly discuss the implementation details for

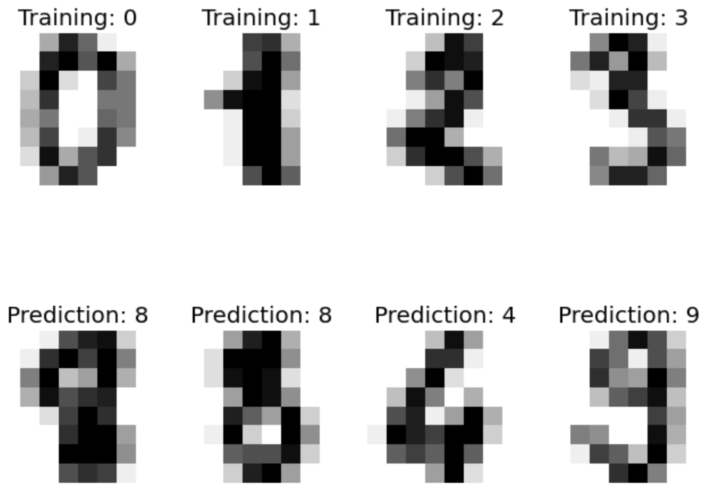


Figure 1: Hand Written Digits Example [2]

the considered algorithms.

2.1 Digit classification Dataset

The first dataset considered in this study is a collection of 8 x 8p gray-scale images of handwritten digits. Each input instance $x \in X$ is projected, or 'flattened' from $x \in \mathbb{R}^{8\times8}$ into a 64-dimensional vector $\hat{x} \in \mathbb{R}^{64}$ the new feature array $\hat{X} \in \mathbb{R}^{64\times m}$ where $m = 1796$ is the number of instances in the dataset; target labels valued $\{0, 1...9\}$ are one-hot encoded as an array of bit strings $\boldsymbol{y}$ where, given a target label n the n^{th} bit of the corresponding string is 1 and 0 for all other bit positions. In the supervised context, our objective is to derive a hypothesis $h \in H, h : x \in \mathbb{R}^{64} \mapsto y \in \{0, 1...9\}$ approximating the target concept $c(x)$. We define the preferred hypothesis as the h which minimizes $error_{\mathbb{D}}(h)$ on the actual distribution $\mathbb{D}$ by learning on the training set $S \subset D \sim \mathbb{D}$ where D is the available dataset of (instance, label) pairs $\{(x_i, y_i)\}$ The objective is some tbd. 'best' function which partitions the instance space by mapping each example $x \in X$ to a corresponding bitstring $y \in \boldsymbol{y}$ where the n^{th} bit of y indicates the target label index.

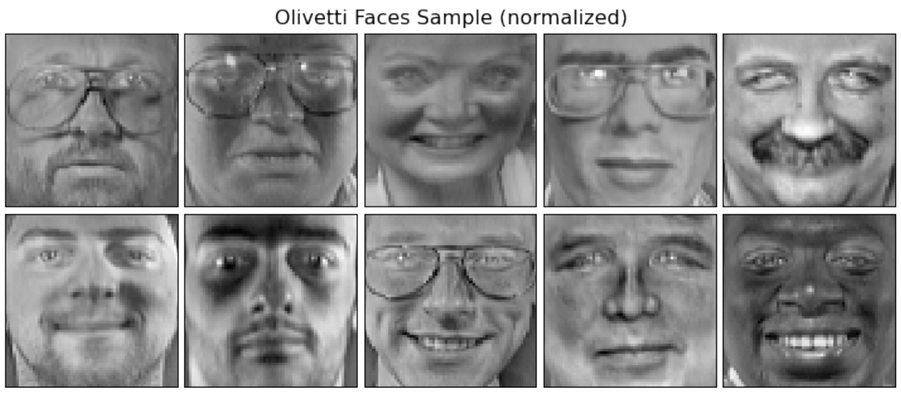


Figure 2: Olvetti Faces Dataset [2]

2.2 Olvetti Faces Dataset

The olvetti faces dataset was gathered between 1992 - 1994 at AT&T labs Cambridge. According to the originating study: "There are ten different images of each of 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement)."

Each input instance $x \in X$ is 'flattened' similarly to the digits dataset. In the olvetti faces data, we have $x \in \mathbb{R}^{64 \times 64}$ transformed into a 4096-dimensional vector $\hat{x} \in \mathbb{R}^{4096}$ the new feature array $\hat{X} \in \mathbb{R}^{4096 \times m}$ where $m = 400$ is the number of instances in the dataset; target labels valued $\{0, 1...39\}$ are one-hot encoded as an array of bit strings $\boldsymbol{y}$ similarly to the digit classification task. Supervised learning objectives and target concepts for the olvetti data are defined similarly to the digit task.

2.3 Dimensionality Reduction

The objective in PCA is to take dataset D with high dimensional vectors $d \in \mathbb{R}^m$ and decompose it into a collection of orthogonal components formed by the singular value decomposition of D which maximizes the variance represented by the components. This representation may then be used to project elements d into a m-dimensional space through linear combination of the principal components (eigenvectors) into transformed features. Furthermore, the associated eigenvalues of the principal components are proxy variables which encode the proportion of the variance explained by the corresponding component; therefore we may select the first $m < n$ components of $PCA(D)$ to maximally represent the variance of D while reducing the dimensionality of the instance space.

While PCA extracts features which are mutually orthogonal while maximizing the variance spanned by each dimension; ICA selects features which are maximally statistically independent. ICA assumes that there exists in D some set of underlying, independent, non-normally distributed source processes S; elements d are assumed to be linear combinations of these independent sources. ICA's job, therefore, is to learn the underlying S, making it exceedingly well suited for blind source separation. Applied to our datsets, we hope to observe PCA learning 'global' features such as average faces and skin tone normalization while we anticipate ICA to learn various local features such as eyes, noses etc.

In our implementation, random projections are computed by generating a random matrix with components selected according to a Gaussian process with 0 mean and $\sigma = 1/n$-components. More interestingly, agglomerative clustering recursively merges related features through inter-feature hierarchical clustering. Agglomeration starts with all features as individual clusters and then recursively extracts new features which minimize the intra-cluster variance taken across the clusters being merged.

2.4 Clustering

The objective of k-means is to partition the instance space into k clusters. To begin, k centroids are selected from a uniform distribution over the instance space. Instances are assigned to the cluster minimizing the euclidean distance between each point and the nearest corresponding centroid. The centroids are then recomputed as the mean of the points in the cluster. This procedure is repeated until a convergence critera is met.

In a loosely analogous way, Expectation Maximization (EM) cycles between an estimate and model update state, however instead of assigning each instance a discrete cluster label (as in k-means) EM estimates the probability distribution assigning probabilities of each point being in a particular cluster. EM is defined in terms of a vector of means $\Theta =< \theta_0, \theta_1 ... \theta_n >$ of n assumed underlying Gaussian processes. In the expectation step, the probabilities of each instance being generated by each Gaussian process with mean θ is computed. In the maximization step, the model parameters Θ are updated such that the probability of a correct association for all instances is maximized.

2.5 Supervised Learning and Image Classification

In our neural network implementation, we elected to use the tangent hyperbolic function *tanh* in place of the standard sigmoid transfer function. This was based on its comparable non-linearity and less compute intensive calculation allowing larger networks and faster weight updates. Input layer dimensionality was dependent on the size of the transformed feature vectors extracted by the dimensionality reduction or clustering algorithms. Weight updates were performed with stochastic gradient descent using the rule:

$$\omega_i^n \leftarrow \omega_i^{n-1} - \eta \frac{\partial \epsilon_n(\omega)}{\partial \omega_i} = \omega_i^{n-1} - \eta \frac{\partial y_\omega(x_n)}{\partial \omega_i}(y_\omega(x_n) - t_n) \tag{1}$$

We update the weight vector ω of layer i, on the n^{th} iteration, given a randomly selected instance x_n, t_n by computing the old ω_i^{n-1} less the approximated error ϵ gradient of weight parameters ω with respect to ω_i scaled by some learning constant η. The second equation is an expansion where the error gradient is explicitly denoted in terms of the input vector x_n, target output t_n and network transformation $y_\omega(x_n)$ using parameters ω.

3 Results

Our first experiments involved purely unsupervised explorations of the digit and faces datasets. In both cases, we leveraged the domain knowledge of knowing how many distinct classes exist in the datasets. We chose $k = 10$ for both datasets. This is appropriate for the digit task as we know there are exactly 10 possible labels. Interestingly, numbers with high inter-class variability form easily interpretable clusters (for example $0, 3, 4, 5, 6$). However numbers which may be more difficult to disambiguate (such as $1, 2, 7$) are not as readily recognizable. After trying several values for k in olvetti faces dataset, we wanted to explore clusterings that were much smaller than the number of actual labels in the dataset; this forces the algorithms to learn inter-class clusterings by computing average faces.

3.1 Dimensionality Reduction Experiments

As previously discussed, the magnitude of eigenvalues should be a monotonically non-increasing function of n principal components (eigenvectors). This corresponds to the amount of importance or variance represented by each principal component. This characteristic is readily observable in our experiments; for the digits dataset, more than 50% of the variance is represented in the first five

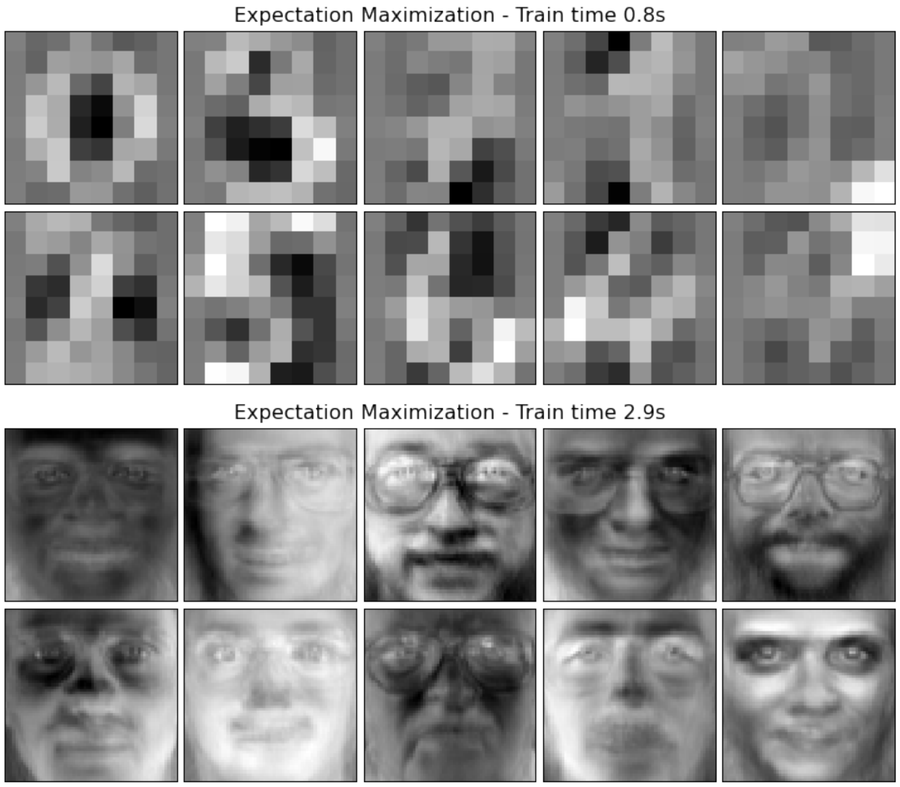
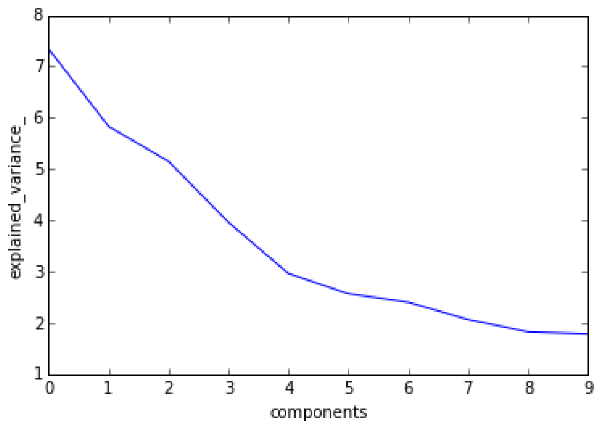

Expectation Maximization - Train time 0.8s
Expectation Maximization - Train time 2.9s

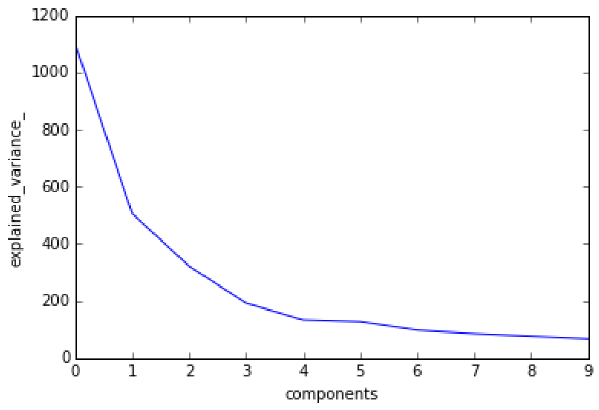
Figure 3: Expectation Maximization Cluster Centers

components extracted by PCA. Applied to the faces dataset, this effect is even more dramatic with a comparable proportion of the variance captured by the first three PCs. It is also conceptually illuminating to visualize the principal components themselves (as in fig 5). Here we visualize the first 10 'eigenfaces' decomposed from the Olivetti dataset. Intuitively, the PCA has extracted a set of global features which represent the variance in skin tone (PC1), lighting orientation (PC2&3), possibly gender (PC4) and a set of facial primitives that may be linearly combined to reconstruct the input images.

The feature agglomeration algorithm also gives interesting insight into the underlying structure of the data. Here we visualize agglomerations into 32 feature clusters for the digit and face datasets. In the case of digits, we observe that large areas of the ambient space around the central region where numbers are most densely distributed are merged into clusters. Conversely we also note that much of the entropy between clusters is focused around the region of the image where numbers frequently occur. Further, the algorithm seems to be picking out some rough structure which constructs correlated vertical line segments across the center of the pixel space. The far more obvious case can be observed in the face agglomeration where eye, nose, mouth, facial edge and ambient pixel space regions are readily identifiable.



(a) Digits PCA



(b) Faces PCA

Figure 4: Distribution of Explained Variance Over Components

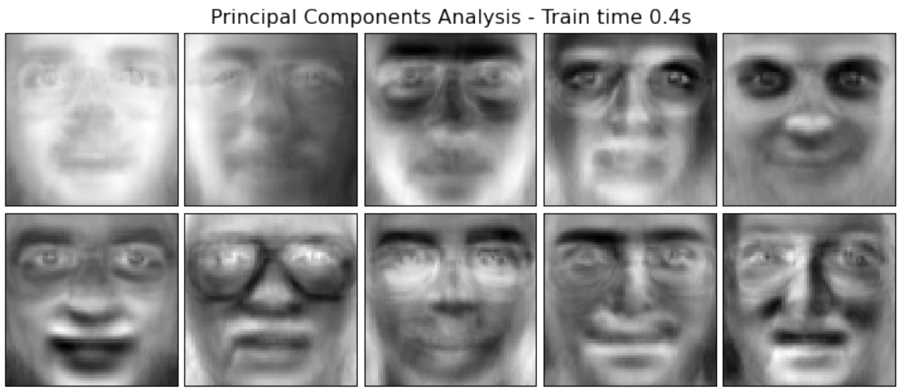


Figure 5: Principal Component Eigen Faces [0:9]

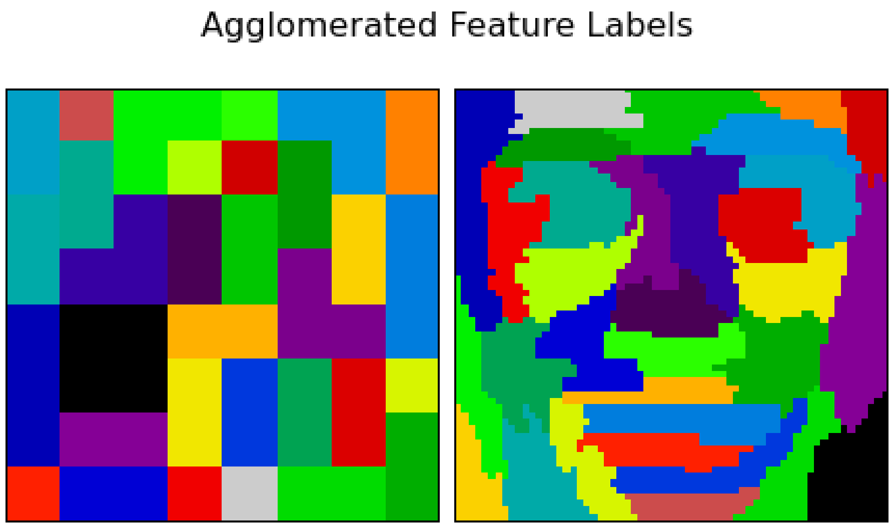
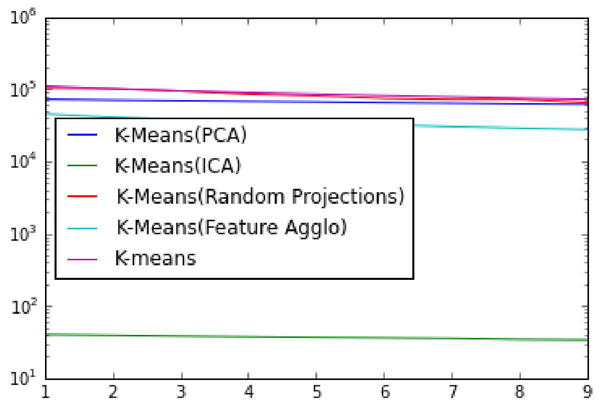
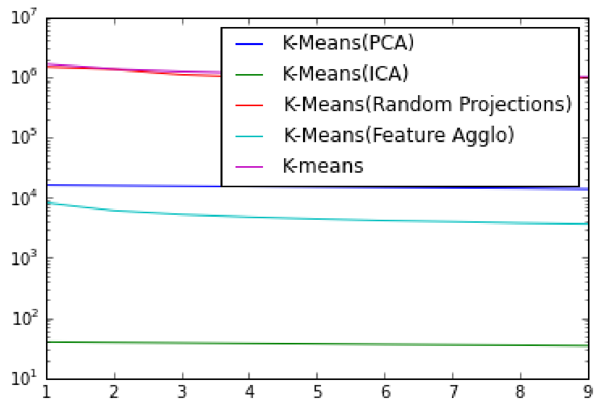


Figure 6: Feature Agglomerations



(a) Digits Inertia



(b) Faces Inertia

Figure 7: Inertia of K-means clustering on Extracted Feature Sets

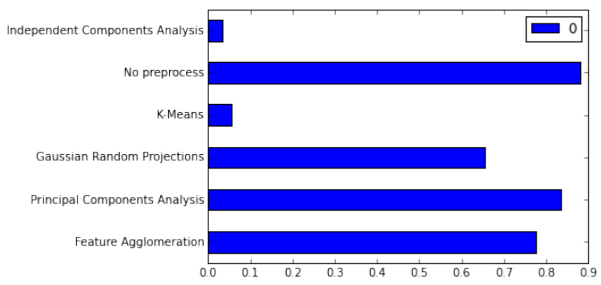
3.2 Clustering Using Extracted Features

To explore the impact of dimensionality reduction on clustering algorithms, a series of experiments were conducted in which the two datasets were clusterd using k-means and expectation maximization after performing feature extraction with n constant components and variable m clusters. The relative 'tightness' of the clusters was measured using inertia (defined as the intra-cluster sum of squares). Generally, random projects showed little impact on inertia vs. the un-transformed data; PCA and feature agglomeration showed a slight reduction in inertia of the digit clusters and roughly two orders of magnitude reduction in face cluster inertia. Independent component analysis however drastically reduced the inertia of clusterings of both datasets.

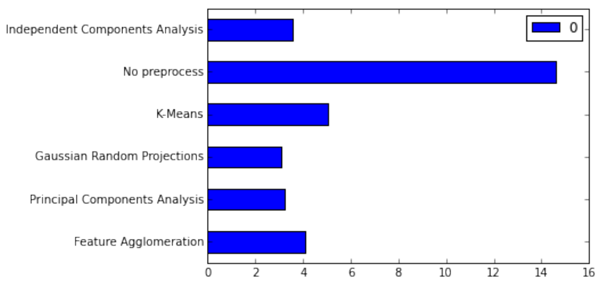
The drastic improvement in clustering observed in the ICA case follows from the fact that ICA maximizes the statistical independence between features. PCA and feature agglomeration are simplifying the clustering problem by finding combinations of correlated features. In PCA, we select orthogonal components that span the broadest variance; similarly in agglomeration we merge features with minimal intra-cluster variance. These are really two sides of a similar coin, in the PCA / agglomeration case, we are grouping along the dimensions of the space; in k-means and EM we group the points that exist in the corresponding space.

3.3 Supervised Learning Experiments

For our supervised learning experiments, we constructed pipelines which encapsulated the preprocessors (clustering or feature extraction) with the classifier (neural network). The Olvetti faces dataset was split with 33% held out for testing and the rest for computing the preprocessing parameters and



(a) F1 Scores



(b) Wallclock Training Times

Figure 8: Neural Network Performance

training the ANN. Although one could fit the preprocessing stage to the full dataset this could overfit the available data, thus hurting the accuracy of our estimated generalization error and possibly exhibiting overly-optimistic performance.

Interestingly, the best classification performance was achieved when no preprocess was applied to the data. This is a situation which is unique to neural networks which in perform a kind of embedded feature extraction and classification within the same model through weighted linear combination and non-linear transformations of the input. Therefore methods such as PCA and feature agglomeration hinder performance, if only slightly, by discarding small amounts information represented in low-variance principal components that could have an impact on the network's learning. Independent component analysis and k-means have a drastically negative impact on classification performance.

The most drastic impact of the preprocessing methods is on the time taken to train the neural network which is greatly reduced by dimensionality reduction. This occurs because a smaller feature space leads to less requisite nodes in the input layer which exponentially decreases the number of weight parameters between the input and hidden layers and thus the number of weight updates that must be computed per iteration. In this kind of scenario, tradeoffs could be considered between sacrificing some performance in the interest of faster training times.

4 Conclusion

In this study we have explored the value and utility of unsupervised methods in image classification. In particular we have used clustering and dimensionality reduction methods to gain insights into the underlying structure of handwritten digits and the Olvetti faces datasets. We have also compared the effects of dimensionality reduction in conjunction with clustering. Finally, we applied feature extraction and clustering methods as a preprocess for training a neural network. We visualized the centroids of clustering algorithms as well as decompositions such as sets of eigenfaces. In the supervised learning experiments we observed all the algorithms negatively impacted the classifier's accuracy due to loss of information, however the training times were much faster using the extracted feature sets.

References

[1] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. [Online; accessed 2015-09-15].

[2] 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.

[3] Pushkar. Abagail, 2015.