CS410 Artificial Intelligence Project Report

Chacha Chen[1] and Sijun Li[2]

[1] *515021910302 chacha1997@sjtu.edu.cn*
[2] *515030910124 1997_lsj@sjtu.edu.cn*

January 15, 2018

Abstract

Artificial Intelligence Based Techniques, including classifiers and statistical learning methods, neural networks, natural language processing, are becoming more and more prevalent. The power of machine learning methods shows great potential in Bioinformatics, like microarray analysis.In this project, some typical machine learning approaches will be taken in this project in a straightforward yet thorough way, based on the Human Gene Expression Dataset. Following the results, the comparative differences between these methods will be discussed. The conclusion on the differences and the application scope of those techniques will be drawn in the end according to the results of our experiment.[1]

1 Introduction

As microarrays technologies are becoming more and more prevalent, the challenges associated with processing and analyzing the large set of human gene data in order to get more biological insight have also increased. Classical machine learning methods, as well as deep learning methods, turn out to be good approaches to address these challenges, which enable us to access a better interpretation of microarray data(Quackenbush, 2006).

Classical methods, such as support vector machine, logistic regression, k-nearest neighbors and decision tree, have the ability to achieve satisfactory accuracy and are relatively robust in classification tasks, which aim to classify examples into a set of pre-defined categories. After analyzing the results of our experiment, their different application scopes and capabilities to deal with different dataset will be summarized to some extent.

Deep learning, a powerful set of learning techniques using neural networks based on representation learning, has become a ubiquitous and cutting-edge machine learning approach recently. In this project, A typical neural network technique will be exploited to analyze the given gene chip data, including a continuous optimization process. Furthermore, the experiment results will be analyzed thoroughly to give a rough yet intuitive interpretation of deep learning.

2 Methods

First, the problem is decomposited into three stages: Data Preprocessing, Model Construction and Performance Assessment. PCA(Principal Component Analysis) is used in the data preprocessing stage. Both classical methods and deep learning techniques are used in Model Construction stage. Finally, cross Validation is the main method to assess the model performance. Specifically, classical methods are employed in both binary classification and multi-class classification tasks, while deep learning methods are mainly used for multi-class classification.

2.1 Data Preprocessing

During this stage, we first do the standardization of the features, and then applying principal Component Analysis (PCA) to reduce the dimensionality.

Standardize(or Z-score normalization) Standardizing the features so that they are centered around 0 with a standard deviation of 1 is an essential requirement for machine learning algorithms, as it is especially crucial in order to compare the similarities between features based on certain distance measures.

[1] Our code can be downloaded at `https://github.com/Chacha-Chen/CS410_AI_Project`

Standard scores, or z score, of the samples are calculated as follows:

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

Often, in Principal Component Analysis, a standardization is performed first.

PCA(Principal Component Analysis) After noise and outliers are handled manually, PCA is applied to reduce the dimension of the dataset.

Principal Component Analysis, as the name indicated, is an algorithm which linearly map the data to a new coordinate system, where the greatest variance by any projection of the data lies along the first coordinate which is the first principal component, the second greatest variance along the second coordinate, and so on(**einasto2011sdss**). The intuition of PCA is that if principal components are chosen to represent the original data, only a commensurately small amount of information will be lost. Dimensionality reduction is done by using only the first few principal components to represent the original dataset.

Figure 1 shows an example of PCA coordinates transformation.

The general steps for performing a PCA are listed as follows:

- Standardize the scale of the data, which has been done in the first stage.
- Obtain the Eigenvectors and Eigenvalues from the covariance matrix or correlation matrix, or perform Singular Vector Decomposition.
- Sort eigenvalues in descending order and choose the k eigenvectors that correspond to the k largest eigenvalues where k is the number of dimensions of the new feature subspace ($k \leq d$).
- Construct the projection matrix W from the selected k eigenvectors.
- Transform the original dataset X via W to obtain a k-dimensional feature subspace Y ($Y = XW$).

2.2 Model Construction

Classification models using both classical machine learning methods and deep learning techniques are constructed during this stage.

2.2.1 Classical Methods

Logistic Regression (Kleinbaum and Klein, 2010) One way to address the binary classification problem is Logistic Regression. The hypothesis function is as below:

$$h_\theta(x) = g(\theta^T x) \tag{2}$$

$$z = \theta^T x \tag{3}$$

$$g(z) = \frac{1}{1 + e^{-z}} \tag{4}$$

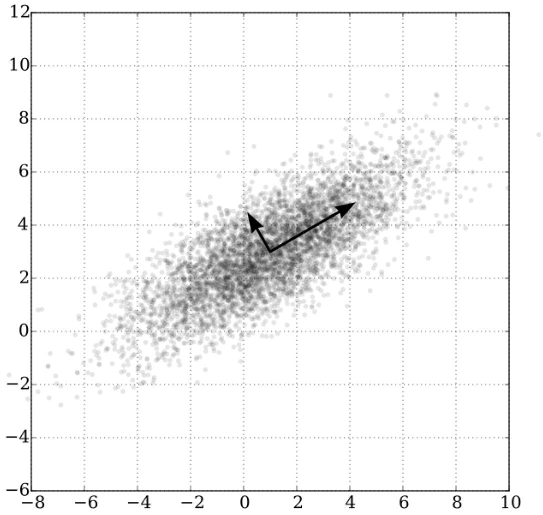

Figure 1: *PCA of a multivariate Gaussian distribution*

Support Vector Machine Among the best "off-the-shelf" supervised learning algorithm, SVM shows a strong ability of generalization and robustness against noise and interference in most cases. The intuition behind SVM can be simply interpreted as an effort to find a decision boundary which maximizes the margin.

SVM for binary classification The mathematical formulation of a binary classification SVM is to solve the simplified Lagrangian dual problem as listed below:

maximize

$$\int(c_1, ..., c_n) = \sum_{i=1}^{n} c_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i c_i k(\vec{x}_i, \vec{x}_j) y_j c_j$$

subject to $\sum_{i=1}^{n} c_i y_i = 0, and \quad 0 \leq c_i \leq \frac{1}{2n\lambda}$ for all i.

SVM for multi-class classification A multi-class classifier is constructed by combining several binary SVM classifiers(Fung and Mangasarian, 2005). Using one-against all method, the implementation of multi-class SVM is to construct k SVM models, where k is the number of classes. The m^{th} SVM is trained with all the examples in the m^{th} class with positive labels, and all other examples with negative labels(Fung and Mangasarian, 2005).

Decision Tree Decision tree(Safavian and Landgrebe, 1991) builds classification or regression models in the form of a tree structure. It breaks down a dataset into smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes.

K-nearest Neighbors KNN(Zhang and Zhou, 2007) is a fundamental and simple machine learning algorithm that categorizes an input by using its k nearest neighbors.The main steps of KNN algorithms are as follows:

- Calculate dx, xi, i=1, 2,..., n; where d denotes the Euclidean distance between the points.
- Arrange the calculated n Euclidean distances in non-decreasing order.
- Take the first k distances from this sorted list.
- Find those k-points corresponding to these k-distances. Let k_i denotes the number of points belonging to the i^{th} class among k points i.e. $k \leq 0$
- If $k_i > k_j \forall i \neq j$ then put x in class i.

2.2.2 Deep Learning

Deep neural networks(Baldi and Hornik, 1989) have been used to train the data as well.

Weight Initialization Although all zero initialization is easy and simple, it will increase the probability of being stuck in a bad local solution as well as eliminate asymmetry between neurons. Consequently, a random initialization is computed with function $w = np.random.randn\frac{n}{\sqrt[2]{n}}$ to each neuron's weight vector where n is the number of its inputs.

Regularization Regularization serves as an approach to sparsify and lower the model complexity and prevent over-fitting problem. L2 regularization is used because L2 is smoother while L1 is more aggressive, which results in some useful features being eliminated.

Loss Function The loss is composed of 2 parts: the difference between a predicted label and its real label, and the L2 regularizer as described above.

Parameter Updating The gradient descent methods are adopted for updating parameters. Also, RMSprop is served as our updating strategy. Some adjustments to solve the early-stopping trap are done in the Adagrad method.

Batch Normalization As a commonly applied method in neural network, Batch normalization is able to increase the speed of convergence as well as automatically choose the learning rate. Moreover, batch normalization could improves the ability of generalization, which leads to a higher accuracy and make the regularization function better, too.

2.3 Performance Assessment

Cross Validation Cross Validation is a technique to evaluate the models by partitioning the original samples into a training set to train and a test set to evaluate.

Specifically, in k-fold cross validation, the original sample is randomly partitioned into k equal-sized subsamples, while a single subsample is retained as the validation data to test the model, and the remaining $k-1$ subsamples are used as training data. The process is then repeated k times, with each of the k subsamples used exactly once as the validation data. Finally, an average value of the k results is derived as an estimation.

3 Experiment

The problem can be formulated as follows: given a set of observation vectors $\{x_1, x_2, ..., x_m\}$ with its labels$\{y_1, y_2, y_3, ...y_m\}$, our goal is to construct a classifier with the ability to predict which category it belongs to given a new set of observation and optimize the classifier to obtain relatively better performance.

3.1 Data Preprocessing

Instances without a label are excluded from the dataset. Further, too few instances within the same class are not suitable for training, too. So the labels with no more than 10 instances are excluded from the dataset. At last, 3613 samples with 92 different labels are used for training.

For binary classification task, labels related to cancer or tumor are denoted as $\{1\}$ class while non-cancer labels are denoted as $\{0\}$ class. For, multi-class classification task, labels are denoted by 93 numerical numbers.

Standardization and PCA After standardization of the dataset, PCA is applied to do the dimensionality reduction, since the task is a large p, small n problem. Also, an evaluation of the variance percentage is conducted, which makes it possible to choose a best final dataset for later training and evaluation.

3.2 Classical methods

Classical methods are exploited to address both the binary classification problem and the multi-class classification problem. In practice, the training is realized by using Matlab, together with the application of parallel pool for the training time optimization. In this part, 5-fold cross validation is used to evaluate the accuracy of the models.

3.2.1 Binary Classification

In this section, classical binary classification methods, including Support Vector Machines, Logistic Regression, Decision Tree and Nearest Neighbor are being adopted.

The implementations of Decision Tree, KNN and Logistic Regression are rather straightforward, since there are not much computation complexity. For support vector machine, three different kernel functions: 'linear', 'Gaussian' as well as 'polynomial kernel function', are tried in an attempt to find the key features and their connections of the dataset.

3.2.2 Multi-class Classification

Decision Tree, KNN as well as SVM are able to solve the multi-class classification tasks. In our experiment, three multi-class classifiers are trained by using these three methods. The results and the comparison is demonstrated in the next section.

3.3 Deep Learning

Keras, a highly integrated machine learning framework, is used for the implementation of deep learning neural networks in the experiment. In detail, 5 hidden layers with 256, 384, 384, 256, 256 neural units respectively are constructed to train, also with an additional layer for L2 regularization. The output layer consists of 93 neural units, corresponding to the number of labels for classification. To settle the problem of unbalanced data distribution, the minority are over-sampled and the majority are under-sampled in the experiment. Batch normalization is also used in every layer in order to achieve a better performance. The performance assessment is done by comparing the accuracy and the loss on both training set and testing set.

4 Result

In this part, our experiment results will be demonstrated and discussed.

4.1 Data Preprocessing

After applying the standardization and PCA, an analysis is done to decide the final dataset being used for training. The relationship between the percentage of variance and the number of dimensions is showed in table 1. Without the loss of generality, both 441 dimensions dataset and 1063 dimensions dataset are chosen for training.

Table 1: *PCA variance preserved*

Variance Percentage	Number of Dimensions
95%	1063
90%	441
85%	126

4.2 Classical Methods

In this part, both binary classifiers and multi-class classifiers using different algorithms are being trained.

Binary Classification First, distribution graphs of the first two principal components and three principal components of the dataset have been demonstrated as Figure2 and Figure 3.

From these two figures, it is still hard to tell whether the dataset is linearly separable or not.

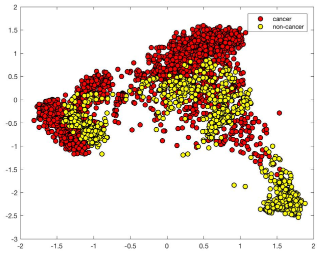

Figure 2: *2-D Distribution of Dataset for Binary Classification*

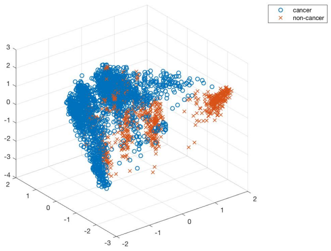

Figure 3: *3-D Distribution of Dataset for Binary Classification*

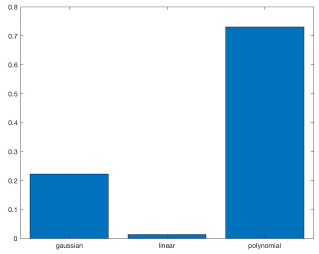

Figure 4: *Loss of SVM using different kernel functions*

Consequently, three different support vector machines using different kernel functions are being trained in order to examine the dataset more thoroughly. The results are shown in Figure 4. A linear kernel SVM performs the best, thus making it possible to draw a safe conclusion that our data is linearly separable.

Further, KNN, Decision Tree as well as Logistic Regression were employed to generate different binary classifiers. Their performances can be seen as in Table 2 and Table 3.

Multi-class Classification Same as binary classifiers, a fuzzy distribution graph is drawn at first, as shown in Figure 5.

Table 2: *Binary Classifiers with 3613*412 dataset*

Methods	Accuracy	Training Time
KNN	89.8%	25.4sec
SVM	99.0%	19.492sec
Decision Tree	95.4%	14.218sec
Logistic Regression	99.2%	151.31sec

Table 3: *Binary Classifiers with 3613*1062 dataset*

Methods	Accuracy	Training Time
KNN	59.9%	60.184sec
SVM	99.1%	74.109sec
Decision Tree	95.8%	43.742sec
Logistic Regression	98.8%	599.43sec

Subsequently, KNN, SVM and Decision Tree are exploited to train the data and the results are shown in Table 5.

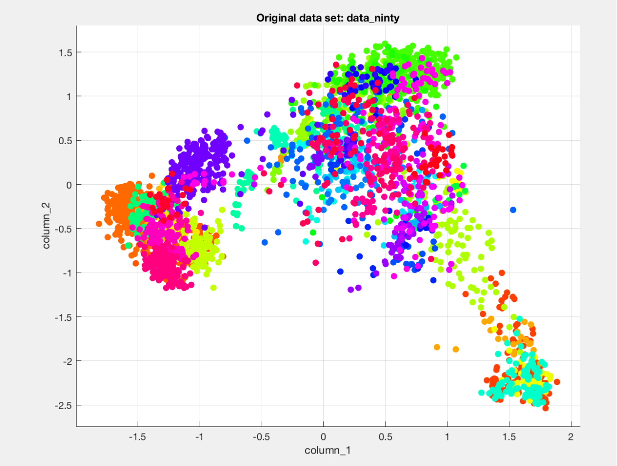

Figure 5: *2-D Distribution of Dataset for Multi-class Classification*

Table 4: *Multi-class Classifiers with 3613*412 dataset*

Methods	Accuracy	Training Time
KNN	77.4%	24.127sec
SVM	85.5%	595.14sec
Decision Tree	61.7%	74.648sec

Table 5: *Multi-class Classifiers with 3613*1062 dataset*

Methods	Accuracy	Training Time
KNN	49.4%	53.733sec
SVM	72.9%	1200.7sec
Decision Tree	60.8%	157.52sec

4.3 Deep Learning

The final results of our deep learning networks on Tensorboard are displayed in Figure 6 and Figure 7. Intuitively, The curves in these figures show the training process.

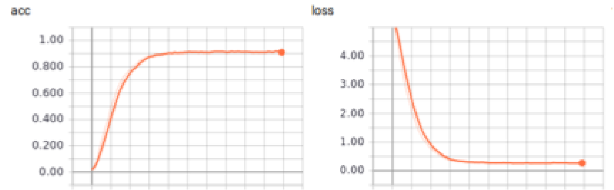

Figure 6: *Accuracy and Loss of Training set*

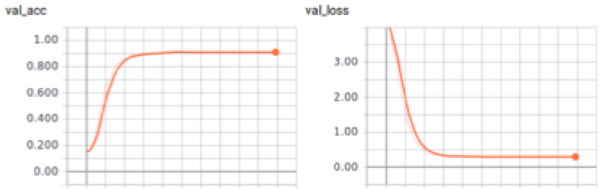

Figure 7: *Accuracy and Loss of Testing set*

The accuracy of both training set and testing set is similar to each other, which indicates that the model is relatively robust with low bias and low variance.

5 Discussion

In this part, the comparison between different kinds of classical machine learning methods, along with the difference between classical methods and deep learning methods will be the main topics. The selection of PCA parameters will also be involved.

5.1 PCA

As is demonstrated in Table 1, the dimensionality varies as the variance changes. However, if the performance of models is taken into account as shown in Table 2,3,4,5, it could be observed that there is a trade-off between the minimal loss of information and the accuracy along with the computation cost.

To be specific, the more dimensions, the less loss of information, while the complexity and computation cost will increase as more dimensions are included, which affects the accuracy. Consequently, a relatively neutral choice is made by choosing the medium size of the dataset, which, in this case, means that the 3613*412 dataset with a variance percentage of 90% is chosen to derive our classifiers.

5.2 Binary Classifier vs. Multi-calss Classifier

In our experiment, binary classifiers obtain both a higher accuracy and a fast training speed. Since the complexity of the model and the data is low in the binary classification case, it is reasonable and natural that the binary classifier is more robust and faster than the multi-class classifier in the experiment.

5.3 Comparison between classical machine learning methods

As for the capability, Logistic Regression is limited to perform binary classification tasks, while the other three could do both binary classification and multi-task classification. As for each of their application scopes are discussed below.

Logistic Regression vs. Support Vector Machines Logistic Regression predicts probabilities that can be interpreted as decision confidence, while SVMs do not penalize examples for which the correct decision is made with sufficient confidence. On the other hands, SVMs are good for generalization and have the ability to give sparse solutions using the kernel trick.

KNN and Decision Tree KNN and Decision Tree have some favorable properties: they are simple enough to perform the classification without many calculations. On the other hand, both Decision Tree and KNN could have a high classification error rate while the size of training set is small compared with the number of features, as is shown by the experiment results.

5.4 Classical Methods vs. Deep Learning Methods

Data dependencies As we could seen from the experiment result, when the dataset is not very large, using classical methods such as SVM could result in much more robust and accurate classifiers than deep learning methods. In other words, the most significant difference between deep learning and classical machine learning methods is that the performance of deep learning methods changes as the scale of data increases. When the data is small, deep learning performance is not as good as classical methods, since deep learning algorithms need a large amount of data to learn the features and connections. Deep Learning could be illustrated as a process of feature decomposition and feature composition afterwards. In contrast, traditional machine learning algorithms with their empirical rules prevail deep learning algorithms under this scenario.

Hardware dependencies In most cases, deep learning is used as an method to address large dataset problems, thus resulting in a high dependency on high-end machines.In contrast, classical machine learning algorithms are able to work on low-end machines.

Application scope As classical machine learning methods have been developed for years, it is applied in many fields. Meanwhile, as a newly developed technique, deep learning is gaining its own popularity, and have a large application potential. In fields like Computer Vision, Information Retrieval, Bioinformatics, Natural Language Processing etc., deep learning is growing at a tremendous speed and is proving to be one of the best techniques to be discovered with state-of-the-art performances.

References

Baldi, Pierre and Kurt Hornik (1989). "Neural networks and principal component analysis: Learning from examples without local minima". In: *Neural networks* 2.1, pp. 53–58.

Fung, Glenn M and Olvi L Mangasarian (2005). "Multicategory proximal support vector machine classifiers". In: *Machine learning* 59.1-2, pp. 77–97.

Kleinbaum, David G and Mitchel Klein (2010). "Analysis of matched data using logistic regression". In: *Logistic regression*. Springer, pp. 389–428.

Quackenbush, John (2006). "Microarray analysis and tumor classification". In: *New England Journal of Medicine* 354.23, pp. 2463–2472.

Safavian, S Rasoul and David Landgrebe (1991). "A survey of decision tree classifier methodology". In: *IEEE transactions on systems, man, and cybernetics* 21.3, pp. 660–674.

Zhang, Min-Ling and Zhi-Hua Zhou (2007). "ML-KNN: A lazy learning approach to multi-label learning". In: *Pattern recognition* 40.7, pp. 2038–2048.