Mice Protein Expression and Mice Treatment Group Prediction

Julia Zheng[1]

1 Computer Science & Engineering, Michigan State University, East Lansing, Michigan, USA

* zhengjul@msu.edu

Abstract

Introduction

Down syndrome (DS) is a genetic abnormality characterized by cognitive disability and unique physical features [1]. DS is usually caused by an extra copy of chromosome 21, but it can be caused by Robertsonian translocation and isochromosomes. Robertsonian translocation refers to a breakage in the chromosomal center (centromere) and consequent chromosome fusion that occurs during fetal development [2]; while isochromosomes is an abnormality where a chromosomes has two short arms or two long arms instead of one short and one long arm [3]. It is hypothesized that Down syndrome is caused by overexpression of genes related to Hsa21, which encodes for over 500 regulatory regions, biochemical enzymes, and cell surface receptors [4]. DS is a biologically complex abnormality that is not fully understood.

Despite the high incidence of DS, there are no existing pharmacotherapies to treat DS-associated learning disabilities. Memantine is a common drug prescribed to treat moderate to severe Alzheimer's Disease [5]. Memantine is an N-methyl-D-aspartate (NMDA) receptor antagonist, which is important to central nervous system diseases like Alzheimers' Disease [6]. Since Down syndrome affects cognition and learning, memantine has been hypothesized to treat DS [7]. Thus, mouse models with an extra copy of Hsa21 related genes have been engineered to test and learn the efficacy of potential Down syndrome pharmaceuticals.

The biological complexity of DS makes identifying the associated genes and pathways difficult. In this study, mice protein expression data is used to identify subsets of the proteins that are discriminant between memantine-treated mice and control mice. The mice data set is retrieved

from a machine learning repository hosted by University of Califonia, Irvine [7]:
https://archive.ics.uci.edu/ml/datasets/Mice+Protein+Expression .

Previous Work

Statistical and computational approaches to testing drugs and learning the changes in treatment group have been used in literature. For example, Naïve Bayes method has been used with and without feature selection in the Ts65Dn drug experiment by Nguyen et al. [8] to predict locomotor activities. Ts65Dn is a mouse model of DS and has been engineered to be trisomic for about two-thirds of the orthologous genes to humans [9]. Supervised and unsupervised learning has been applied to biological data with success [10]. In Down syndrome research, Ts65Dn has been featured in many studies to understand the biological mechanisms, such as learning and memory, and to test drug efficacy. Although memantine has been demonstrated to antagonize NMDA receptors in both Ts65Dn mice with DS and normal mice, there is not much know about the consequential protein expression changes [7].

The mice protein data set has been previously analyzed in two studies using different methods. For example, in the study by Ahmed et al. [9], the selected statistical test, K-means, on the drug effects on Ts65Dn and controls showed no correlation between drugs and cognitive improvement. Whereas in Higuera et al. [7], unsupervised learning method Self-Organizing Feature Maps (SOM) was used to analyze the mice protein expression data, which was successful in discriminating control from individuals treated with drugs. These two papers demonstrate the importance of carefully choosing classifiers. For the mice protein express data set in this study, various features selection methods and classifiers will be tested to see how they compare to each other and which is the best for this data set.

Data Set Description

The mouse protein expression data set contains eight classes of mice (see Table 1), which are separated based on genotype, learning behavior, and treatment. Depending on the genotype, a mouse can be trisomic (Down syndrome) or control (wild type). The behavior describes whether the mice have been stimulated to learn (context-shock) or not stimulated (shock-context). There are between seven to ten mice in each of the eight classes, for a total of 38 control mice and 34 trisomic mice (Down syndrome). For each individual, fifteen independent measurements are

taken for each protein. Hence, there are 570 samples for the control mice, and 510 samples for the trisomic mice. In total, there are 1080 samples.

Table 1. Mice classes and treatment groups.

Label	Mice class	#Mice	Genotype	Stimulation to learn	Treatment
0	c-SC-s	9	control	Shock-Context	saline
1	c-SC-m	10	control	Shock-Context	memantine
2	c-CS-s	9	control	Context-Shock	saline
3	c-CS-m	10	control	Context-Shock	memantine
4	t-SC-s	9	Ts65Dn	Shock-Context	saline
5	t-SC-m	9	Ts65Dn	Shock-Context	memantine
6	t-CS-s	7	Ts65Dn	Context-Shock	saline
7	t-CS-m	9	Ts65Dn	Context-Shock	memantine

The data set contains 77 proteins/protein modifications, all of which have detectable signals in the nuclear fraction of the cortex. Reverse phase protein arrays (RPPA) has been used to measure the protein concentrations. All protein expression values are real numbers between zero and nine. Some samples have less than fifteen measurements. Unfortunately this high-throughput method does not allow repeat experiments on individual measurements. Therefore, the final data set has missing values.

In the study where the data set is from [7], memantine has been able to rescue Ts65Dn mice with DS from cognitive deficits. Intelligence is measured in this study by context fear conditioning (CFC), which is learning that occurs when an aversive stimulus, like electric shock, is associated with a context [7]. The simplest CFC method has two groups of mice, one shocked earlier and one shocked later to control for the effect of the electric shock [7]. The context-shock (CS) mice classes are placed into a new cage and allowed to explore for several minutes before given an electric shock. Wild type, normal mice in CS condition learn that the new cage is the context where aversive stimulus is present and they will freeze in fear when exposed to the new cage in the future. However, Ts65Dn mice in CS condition do not learn to associate a place with aversion, and when exposed to the same cage do not freeze. This is a learning impairment that may be corrected if the Ts65Dn mice are treated with memantine. The shock-context (SC) mice are placed into a new cage, immediately shocked, and then allowed to explore. Wild type mice in SC condition do not associate the new cage with aversive stimulus and thus do not freeze in the future. It is anticipated that Ts65Dn mice in SC condition will not freeze as well due to learning impairment. To control for possible side-effects of the act of injecting, saline is given to control groups. Therefore, wild type and Ts65Dn mice each will have four treatment groups: CS-memantine, CS-saline, SC-memantine, and SC-saline. This there are a total of eight mice classes.

Description of Analysis Conducted

Data Pre-processing

The original data received is a comma-separated value (csv) file with 82 columns and 1081 rows. First row is the header information and included all columns names, hence this is a labeled data set. The first column is a mouse identification value, which is removed during data set pre-processing because it is not a measurement of the mouse protein expression. Columns 79 through 82 are labels relevant to the treatment group and not to the protein expression, thus they are also removed during pre-processing. In total, there are 77 columns/features and 1080 rows of labeled data relevant to the experiment. There are missing values in this data set. Of the 77*1080 = 83160 values in the table, 376 are null valued. Hence, this data set has 0.4521% empty values. From visually inspecting the data, all values in the table are real-numbers between zero and nine, so the missing values are filled in with the mean of the nonzero values. The data set is split into 60% train, 20% validate, and 20% test data sets. The train set is used on the classifiers to learn about the patterns of the underlying distribution. The validate set is used to check the classifiers performance and tune the hyper-parameters of the classifiers if possible. The learned classifier is used to separate the test data set by classes, and the results are compared with the actual classes to determine the accuracy. To compare the effect of feature selection, the data set in examined by all of the classifiers with and without feature selection.

Feature Reduction

The mouse protein expression data set may have redundant properties, hence feature reduction is used prior to training classifiers. Feature reduction is a way of reducing the number of features by obtaining informative features that may contribute most to the prediction of target variables. Feature reduction on this data set is done by principal component analysis (PCA) and linear discriminant analysis (LDA). The input to PCA and LDA is standardized by z-score, which normalizes the mean. PCA is an unsupervised technique that chooses the component axes that maximize the variance within the data set [11]. PCA accomplishes the feature reduction by summarizing the correlations between features of the original training set and then performing eigenvalue decomposition on its covariance matrix [12]. LDA is a supervised technique that aims to separate the eight classes by choosing the component axes that maximize the separate between classes [12]. LDA assumes that features are independent and normally distributed [12].

Classification

All analysis in this paper are coded in Python3.6. There are seven classifiers examined in this paper: Naïve Bayes, Support Vector Machine (SVM), Random Forests, K-Nearest Neighbor (KNN), Multi-Layer Perceptron (MLP), and AdaBoost. Naïve Bayes is implemented from scratch, while the other five classifiers are from Python package Scikit-Learn [13]. There is an extra classifier from package Auto-Sklearn called autoclassifier that automatically finds and tunes an ensemble of classifiers [14]. The autoclassifier is an artifical learning software built on top of Scikit-Learn algorithms [14]. The goal of utilizing an array of seven classifiers is to compare the performance and to determine the best techniques for this data set.

Naïve Bayes classifier is implemented by assuming the underlying distributions are multivariate Gaussian [12]. This assumption creates hyperquadratic decision boundary that dependent on prior probabilities [12]. The prior probabilities are calculated on the training set, the posterior probabilities of the test set are calculated with multivariate Gaussian model, and the conditional probabilities are then determined using the naïve Bayes rule. For each sample, the maximum *a posteriori* principle is used to estimate the sample class. The Naïve Bayes classifier is first trained with the training set. There are no hyper-parameters to tune in this implementation of the Naïve Bayes classifier.

SVM classifier is a supervised learning method that maps data to high-dimensional feature space and discerns linear decision boundaries. SVM is implemented with Scikit-Learn and the optimal parameters are determined by grid search through parameters including kernel function, gamma, and C. Kernel function can be linear and Radial Basis Function (RBF). RBF is a Gaussian kernel method that creates non-linear combinations of the features and generate values correlated with the Euclidean distance between points [15]. The gamma parameters controlling the distance of the training samples [16] and is grid searched on 0.0000001 and 0.0001; meanwhile, the C parameter is a trade-off between correct classification and maximization of decision function's margin [16] and varies from 1, 1.5, 10, 50, 100, to 200.

Random Forests classifier is a supervised ensemble classifier implemented in Scikit-Learn. Random Forests of multiple decision trees are fitted on subsets of the training set [17]. This classifier uses averaging across decision trees to improve prediction accuracy and to reduce over-fitting [17]. The number of estimators is grid searched from ten to 300

at every ten intervals.

AdaBoost classifier is a boosting ensemble learning method that utilizes a weighted majority vote with weak classifiers on subsets of the training set to produce a high performance ensemble [12]. AdaBoost is implemented in Scikit-Learn and it uses a sequence of decision trees on repeatedly modified training subsets [12]. Every decision tree learner has a weight depending on its performance, where incorrectly classified samples are adjusted to be focused on by the consequent learners [18]. The number of estimators is grid searched from ten to 300 at every ten intervals; whereas, the boost algorithm is either SAMME or SAMME.R [18].

KNN classifier is a non-parametric, supervised classifier that doesn't assume anything about the underlying distribution [12]. Minkowski distance is the default distance metric in Scikit-Learn, and it is used to calculate the distance of a point to its surrounding neighbours [19]. KNN assigns the data point to the class shared by the majority of its K closest neighbors [19]. Grid search is used to find the optimal K value ranging from one to 100.

MLP classifier is a supervised, artificial neural network implemented in Scikit-Learn. Each layer of perceptron learns a function mapping, and in between the input and output layers there can be multiple linear or non-linear, hidden layers [20]. MLP iteratively updates the loss function and the model parameters via backpropagation training method [21]. Grid search is used to find the optimal MLP parameters for the training set. The hidden layer sizes are sampled from a 2020 bladder cancer diagnosis paper [22] that also utilizes MLP: (20,20),(80,20),(80,20,20),(80,40,40,20), and (40,40,20,20,20,10). Learning rate is set to adaptive. Optimization algorithms test include Limited-memory Groyden-Fletcher-Foldfarb-Shanno (L-BFGS), Stochastic Gradient Descent (SGD), and an extension of SGD called Adam [21]. Activation functions determine the outputs of MLP [21], and the following three were test: Tanh, logistic regression, and Rectified Linear Unit (ReLU).

Autoclassifier from Auto-Sklearn is a supervised ensemble that automatically searches for optimal learning algorithms [23]. Autoclassifier combines fifteen classifiers, fourteen feature preprocessing algorithms, and it identifies similar data sets gathered in the past to rapidly identify a optimial ensemble [23]. No validation set is used because Autoclassifier does not allow manual cross validation.

Post-Processing

The results from running each classier multiple times are recorded and the accuracy calculate from the confusion matrix. To produce precision-recall graphs for performance evaluation, the classification results are micro-averaged. Micro-averaging gives provides a generalized view of the results by summing the values of confusion matrix and dividing by the aggregate number of class samples [24]. Micro-averaging is a useful metric for class imbalance data sets [24]. Precision-recall curves are drawn using the micro-average of each class across all classification results prior to validation. Only SVM, Random Forests, KNN, MLP, and AdaBoost can produce precision-recall curves due to implementation.

Results and Analysis

Feature Reduction Tests

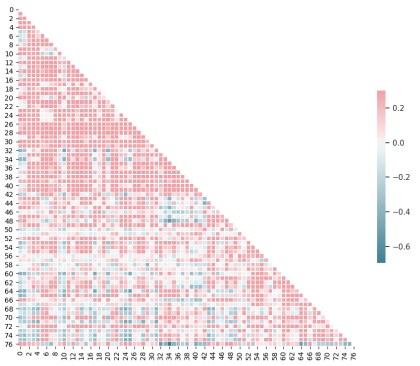

Fig 1. Feature correlation map relevant to LDA.

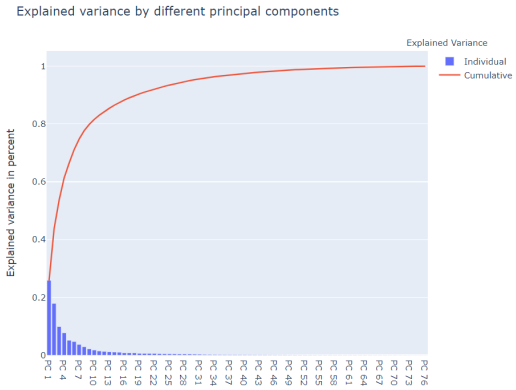

Fig 2. Explained variance by principal components.

In this study, some features have high correlation and not all features are informative (see in Fig 1. Prior to LDA pre-processing, three collinear features are removed. The LDA pre-processed training set has seven linear discriminant components chosen by the method. Fig 2 compares the explained variance in PCA between different principal components. The first principal component is the most informative at 25.84% explained variance. When the first twenty-two principal components are retained after feature reduction with PCA, over 92% cumulative explained variance is achieved (see red line in Fig 2). Hence, PCA pre-processed training set has twenty-two principal components. Therefore, feature reduction using PCA results in twenty-two principle components, while using LDA results in eight linear discriminant components.

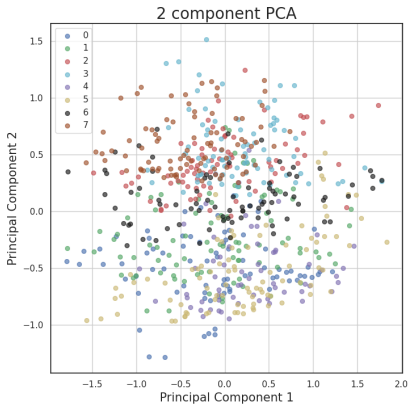

Fig 3. Graph of data set on two PCA component axes on MinMaxScaler data.

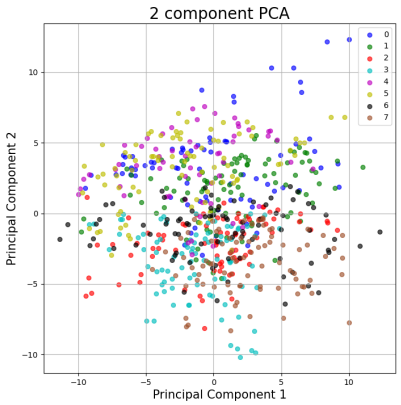

Fig 4. Graph of data set on two PCA component axes on z-score data.

Two component axes graphs are shown in Fig 3, Fig 4, and Fig 5. PCA transformed data points are graphed on two principal component axes, however there is significant overlap of all eight classes when normalized with z-score (see Fig 4) and with MinMaxScaler to between 0 and 1 (see Fig 3). LDA transformed data points are plotted on two LDA component axes with high degree of separation between the classes. Hence, these two figures justify the methodology of choosing more components for PCA than LDA.

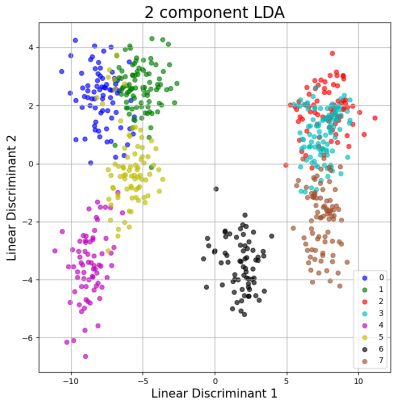

Fig 5. Graph of data set separated on two LDA axes.

Classification Results

Table 2. KNN and Random Forests Results.

	KNN			**Random Forests**		
	Original	PCA	LDA	Original	PCA	LDA
Cross Validation	**99.50%**	98.13%	97.21%	98.36%	96.50%	95.83%
CV Stdev	0.39%	0.14%	**0.10%**	0.57%	0.57%	0.87%
Count runs	127	125	124	24	23	21
Tuned Accuracy	**99.43%**	98.13%	96.30%	96.55%	96.55%	95.30%
Accuracy Stdev	0.78%	0.14%	**0.05%**	00.57%	0.57%	0.61%
Count Final Runs	126	125	124	22	22	21

Results for KNN and Random Forests classifiers are in Table 2. Overall, KNN has better prediction accuracy on this data set after tuning than Random Forests. Original data set input into SVM and AdaBoost resulted in higher accuracy than the PCA and LDA data sets. One possible explanation for this is phenomenon is that there are more features in this data set that are informative and significantly aid classifiers' learning than has been extracted by the applied feature reduction. Accuracy before and after classifier tuning with held out validation set are different, where the accuracy is higher before tuning. Performance on all test data sets are lower before classifier tuning, which may be a sign of over-fitting.

Table 3. SVM and AdaBoost Results.

	SVM			**AdaBoost**		
	Original	PCA	LDA	Original	PCA	LDA
Cross Validation	**99.05%**	93.08%	95.37%	49.01%	74.40%	58.35%
CV Stdev	0.23%	0.28%	**0.05%**	0.63%	0.53%	0.93%
Count runs	105	105	104	105	103	103
Tuned Accuracy	**99.05%**	93.08%	95.42%	47.68%	75.24%	58.27%
Accuracy Stdev	0.23%	0.20%	0.41%	**0.10%**	0.92%	0.58%
Count Final Runs	105	105	104	105	103	103

Results for SVM and AdaBoost classifiers are in Table 3. AdaBoost has significantly worse performance than SVM, reaching as low as 47.68% prediction accuracy. AdaBoost's highest prediction accuracy is with PCA data set input. Similar to Table 2 results, prediction accuracy is higher for original test data set compared with feature reduction test data set, and classifier perform is better hyper-parameter tuning than after.

Table 4. MLP and Naïve Bayes Results.

	MLP			**Naïve Bayes**		
	Original	PCA	LDA	Original	PCA	LDA
Cross Validation	88.71%	97.58%	**97.93%**	N/A	N/A	N/A
CV Stdev	4.11%	**0.94%**	1.07%	N/A	N/A	N/A
Count runs	92	91	90	N/A	N/A	N/A
Tuned Accuracy	92.83%	96.83%	**97.02%**	87.34%	86.98%	87.58%
Accuracy Stdev	1.64%	1.17%	**1.11%**	1.62%	1.71%	2.86%
Count Final Runs	92	91	90	23	23	23

Results for MLP and Naïve Bayes classifiers are in Table 4. Naïve Bayes classifier does not have hyper-parameter tuning on held out validation set due to implementation. Differing from Table 2 and Table 3 results, MLP classifier has better performance after classifier tuning than before. However, MLP has worse performance on the original test set compared with KNN, Random Forests, and SVM.

Results for Scikit-Learn's Autoclassifier is in Table 5. Autoclassifier does not have hyper-parameter tuning on held-out validation set due to Scikit-Learn implementation. Prediction accuracy for feature reduced,

PCA and LDA test sets is the the highest out of all classifiers. Autoclassifier performance on the original test set is only second to Naïve Bayes, which means it is close to optimal.

Table 5. Autoclassifier Results

Autoclassifier			
	Original	PCA	LDA
Cross Validation	N/A	N/A	N/A
CV Stdev	N/A	N/A	N/A
Count runs	N/A	N/A	N/A
Tuned Accuracy	98.89%	**99.07%**	97.22%
Accuracy Stdev	1.49%	**0.38%**	0.76%
Count Final Runs	4	4	4

Overall, the highest prediction accuracy classifier on the original test set is KNN with 99.50% accuracy before tuning, and the second highest accuracy classifier is SVM with 99.05% before tuning. The highest prediction accuracy classifier on feature reduced, PCA and LDA test sets is the autoclassifier. The lowest prediction accuracy classifier is AdaBoost on the original data set with 47.68% accuracy.

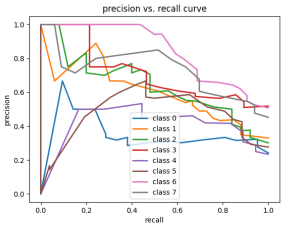

Fig 6. KNN precision recall graph on unscaled data set.

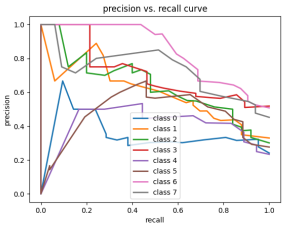

Fig 7. KNN precision recall graph on PCA data set.

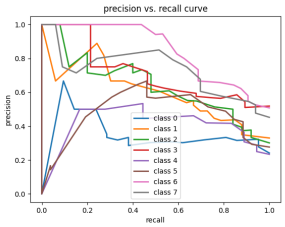

Fig 8. KNN precision recall graph on LDA data set.

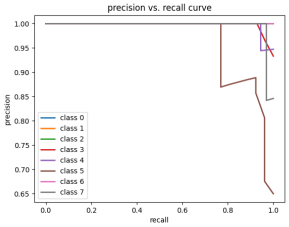

Fig 9. Random Forests precision recall graph on PCA data set.

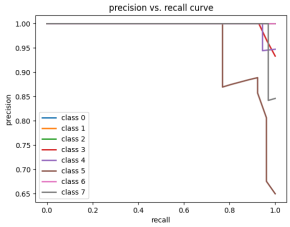

Fig 10. Random Forests precision recall graph on PCA data set.

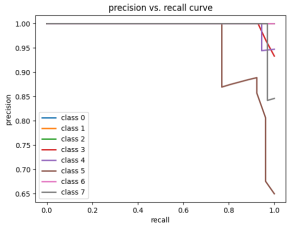

Fig 11. Random Forests precision recall graph on LDA data set.

Figures 6 through 20 are precision-recall (PR) graphs for each class of mice in each classification group. Since observations are not balanced between each class (see Table 1), Receiver Operating Characteristic (ROC) curve cannot be used. PR curves represent the trade-off between true positive rate, which is the y-axis, and sensitivity, which is the x-axis. Random Forests PR curves (see Figures 9, 10, 11) are closest to perfect classifier PR curve. AdaBoost PR curves (see Figures 15, 16, 17) and MLP

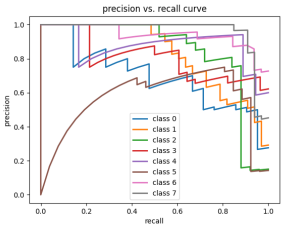

Fig 12. SVM precision recall graph on unscaled data set.

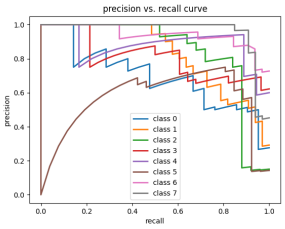

Fig 13. SVM precision recall graph on PCA data set.

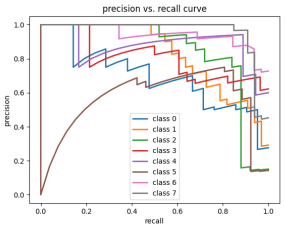

Fig 14. SVM precision recall graph on LDA data set.

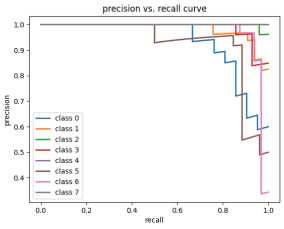

Fig 15. AdaBoost precision recall graph on unscaled data set.

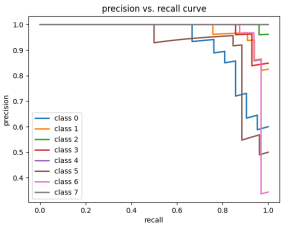

Fig 16. AdaBoost precision recall graph on PCA data set.

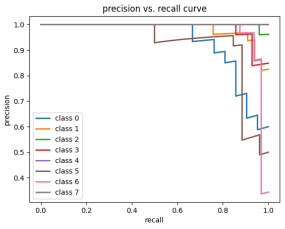

Fig 17. AdaBoost precision recall graph on LDA data set.

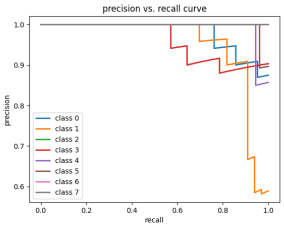

Fig 18. MLP precision recall graph on unscaled data set.

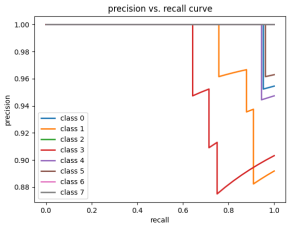

Fig 19. MLP precision recall graph on PCA data set.

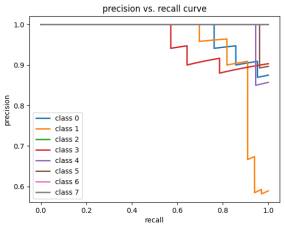

Fig 20. MLP precision recall graph on LDA data set.

PR curves (see Figures 18, 19, 20) are worse than Random Forests PR, but they are still good classifiers with over 90% PR-area-under-the-curve (AUC). KNN PR curves (see Figures 6, 7, 8) and SVM PR curves (see Figures 12, 13, 14) are significantly worse than the PR curves of Random Forests, AdaBoost, and MLP. Thus, Random Forests classifier have the best performance when PR is measured, while KNN and SVM classifiers have the worse performance.

Taking in the analysis of prediction accuracy and PR-AUC, the researcher is confused with the results. Prediction accuracy measures predict KNN and SVM to be the best performing classifiers, while indicating AdaBoost is the worst performing classifier. Meanwhile, PR-AUC measures predict AdaBoost, Random Forests, and SVM to be

great classifiers, while indicating KNN and SVM are the worst performing classifiers. It is known that prediction accuracy measures don't take into account of class imbalance, while PR-AUC on micro-averaged classifier outputs do take into consideration of class imbalance. Hence, it's possible that class imbalance is causing significant error in the prediction accuracy analysis.

Table 6. F1 Scores.

Original	Naïve Bayes	SVM	Random Forests	KNN	MLP	AdaBoost	Autoclassifier
c-CS-s	0.18	0.96	0.96	0.94	0.86	0.24	1
c-CS-m	0.00	0.95	0.95	0.86	0.67	0.31	0.98
c-SC-s	0.00	1.00	1.00	1.00	1.00	0.22	1.00
c-SC-m	0.00	1.00	0.98	0.98	0.99	0.60	1.00
t-CS-s	0.00	1.00	1.00	1.00	0.93	0.27	1.00
t-CS-m	0.00	0.92	0.97	0.95	0.89	0.00	0.97
t-SC-s	0.00	1.00	1.00	1.00	1.00	0.52	1.00
t-SC-m	0.00	1.00	0.98	0.98	0.98	0.06	1.00
PCA							
c-CS-s	0.00	0.85	0.95	0.99	0.95	0.47	0.96
c-CS-m	0.00	0.80	0.91	0.93	0.95	0.50	0.93
c-SC-s	0.00	0.98	0.98	1.00	0.98	0.60	1.00
c-SC-m	0.00	0.94	0.97	0.98	0.99	0.73	1.00
t-CS-s	0.20	1.00	1.00	0.93	1.00	0.30	1.00
t-CS-m	0.00	0.89	0.95	0.97	0.95	0.50	1.00
t-SC-s	0.00	1.00	0.98	1.00	1.00	0.74	1.00
t-SC-m	0.00	0.95	0.97	0.98	1.00	0.51	1.00
LDA							
c-CS-s	0.00	0.94	0.95	0.95	0.92	0.85	0.96
c-CS-m	0.00	0.84	0.84	0.91	0.82	0.70	0.87
c-SC-s	0.00	1.00	1.00	1.00	0.98	0.41	1.00
c-SC-m	0.26	1.00	0.97	1.00	0.98	0.00	1.00
t-CS-s	0.00	0.95	0.95	1.00	0.97	0.82	0.95
t-CS-m	0.00	0.86	0.88	0.91	0.90	0.65	0.92
t-SC-s	0.00	1.00	1.00	1.00	1.00	1.00	1.00
t-SC-m	0.00	1.00	0.97	1.00	0.97	0.00	1.00

As a tie-breaker, F1 metric, a well-known performance metric that takes class imbalance into consideration, is brought into the analysis (See Table 6,). F1-score is a harmonic mean of weighted Precision and Recall [25]. Naïve Bayes has the lowest F1 measurements with seven out of eight classes at zero F1-score, which is due to zero-valued recalls. AdaBoost has the second lowest F1-scores. The highest F1-score is Autoclassifier from Scikit-Learn, with second highest F1 performance measures being tied between SVM and Random Forests.

Further improvements can be made on the methods. The held out validation set may not be as effective as K-fold validation, so future works should investigate cross validation and hyper-parameter tuning with K-fold validation. Another issue is the MLP classifier's hidden layers some times don't converge. Future works can improve MLP by further transforming the data or increase the number or iterations. Furthermore, the pre-processing of the data set and treatment of missing values may not be optimal. Current method is to take the average of existing values in the

same column, but this may confound the result. The future work may investigate the effect of removing the six columns containing missing features, which is effectively removing eight percent of the data set. Therefore, the methodology can be improved in future works.

Obtained prediction accuracy analysis have strange patterns that needs to be discussed. It is strange that KNN with K=1 has the best accuracy. Theoretically, accuracy improves as K grows to infinity, but K=1 is selected by grid search algorithm from a range of K values between one and 100. One possible explanation for this phenomenon is that the test set are very similar to the training set. Another possibility is that the data set is not easily separable with linear decision boundaries, leading to better performance with KNN than SVM when prediction accuracy is measured. Additionally, AdaBoost has the worse performance overall across all data sets when it is an ensemble. It is possible that the underlying classifiers AdaBoost utilizes are inadequate for this data set. One way to improve AdaBoost is to change its weak learners from Decision Trees to a classifier more capable of handling multi-class, non-linearly separable data. Another strange problem is that some classifiers, including Naïve Bayes, KNN, Adaboost, and Random Forests, are less accurate after tuning with the held out validation set. This issue may be partially due to over-fitting the classifiers to the training and validation data sets. Issues involving over-fitting and AdaBoost implementations should be examined in future works.

The three performance metrics used in analysis all differ. F1 performance metric and prediction accuracy both indicate the second highest performance classifier for this data set is SVM, while PR-AUC considers SVM to be tied for last place. However, prediction accuracy suggests that Naïve Bayes is a great classifier while F1-score suggests that Naïve Bayes is the worst performing classifier. PR-AUC indicates AdaBoost is tied for the second best performing classifier for this data set, but F1-score and prediction accuracy measures disagree. One of the ways to improve performance measurements of PR-AUC and F1-score is to make sure baselines are optimal via precision-recall-gain algorithm [25]. Another way to improve PR-AUC and F1-scores are to optimize each of their thresholds, while keeping in mind that precision affects F1-scores [26]. Overall, performance analyses can be investigated more thoroughly and future work can likely improve the implementation of the performance metrics.

Summary and Conclusion

Down syndrome pharmacotherapies are under active development to treat its cognitive and behavioural deficits. The mice protein expression data set

is fed as input into two different feature reduction schemes and seven different classifiers with the goal of discerning the protein assay profiles of eight classes of mice.

Feature reduction techniques PCA and LDA are applied during pre-processing. There are seven linear discriminant components chosen by LDA; meanwhile, there are twenty-two principal components chosen by PCA that make up more than 92% of the information about the data set.

Three performance matrics are applied on each of the classified treatment groups. Prediction accuracy is a performance metric that measures number of misclassifications without considering class imbalance, while PR-AUC and F1 are performance metrics that consider class imbalance. Difference classifiers evaluated with three performance metrics indicate contradictory results. For example, KNN classifier gives the highest performance when measured by prediction accuracy, but PR-AUC indicates KNN is tied for the worst performance classifier. PCA feature reduction test set classified with Random Forests classifier gives 96.55% on prediction accuracy. On the same data set, Random Forests gives over 91% F1-score on each of the eight classes, while on according to PR-AUC Random Forests is the best classifier for this data set. Notably, Autoclassifier from Scikit-Learn performs the best in prediction accuracy and F1 metrics.

Given more time and computational resources, the future works to extend this paper should focus on improving the methods and the performance metrics. Most importantly, PR-AUC and F1-scores baselines and thresholds should be optimized. During pre-processing, the features with missing data can be removed and the results evaluated in the same way to compare. Additionally, the classifiers can utilize K-fold cross validation of the training set. Therefore, future works will focus on improving pre-processing, cross validation, and performance metrics.

Supporting information

Additional charts. This document contains supplementary information including tables and charts that did not fit in the paper.

Python code Github link:
https://github.com/zhengjul/CSE802-Project-

Acknowledgments

The author would like to acknowledge Clara Higuera from university Complutense, Kateneleen J. Gariner, University of Colorado, and Krzysztof J. Cios, Virginia Commonwealth University for providing the data set.

References

1. Plaiasu V. Down Syndrome - Genetics and Cardiogenetics. Maedica (Buchar). 2017;12(3):208–213.

2. Kim SR, Shaffer LG. Robertsonian translocations: mechanisms of formation, aneuploidy, and uniparental disomy and diagnostic considerations. Genet Test. 2002;6(3):163–168.

3. Akbas E, Altintas ZM, Celik SK, Dilek UK, Delibas A, Ozen S, et al. Rare Types of Turner Syndrome: Clinical Presentation and Cytogenetics in Five Cases. Laboratory Medicine. 2012;43(5):197–204. doi:10.1309/LMEZQXK85CDP4HYN.

4. Sturgeon X, Le T, Ahmed MM, Gardiner KJ. Pathways to cognitive deficits in Down syndrome. Prog Brain Res. 2012;197:73–100.

5. Olivares D, Deshpande V, Shi Y, Lahiri D, Greig N, Rogers J, et al. N-methyl D-aspartate (NMDA) receptor antagonists and memantine treatment for Alzheimer's disease, vascular dementia and Parkinson's disease. Current Alzheimer Research. 2016;9.

6. Newcomer J, Farber N, Olney J. NMDA receptor function, memory, and brain aging. Dialogues in clinical neuroscience. 2000;2.

7. Higuera C, Gardiner K, Cios K. Self-Organizing Feature Maps Identify Proteins Critical to Learning in a Mouse Model of Down Syndrome. PLoS One. 2015;10.

8. Nguyen C, Costa A, Cios K, Gardiner K. Machine learning methods predict locomotor response to MK-801 in mouse models of down syndrome. Journal of neurogenetics. 2011;25:40–51.

9. Ahmed M, Dhanasekaran R, Block A, Tong S, Costa A, Stasko M, et al. Protein Dynamics Associated with Failed and Rescued Learning in the Ts65Dn Mouse Model of Down Syndrome. PLoS One. 2015;10.

10. Larrañaga P, Calvo C, Santana R, Bielza C, Galdiano J, Inza I, et al. Machine learning in bioinformatics. Biefings in Bioinformatics. 2006;7.

11. Martinez A, Kak A. PCA versus LDA. IEEE Transactions on Patter Analysis and Machine Intelligence. 2001;23.

12. Duda R, Hart P, Stork D. Pattern Classification, 2nd Edition. Wiley-Interscience; 2000.

13. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. 2011;12:2825–2830.

14. Feurer M, Klein A, Eggensperger K, Springenberg J, Blum M, Hutter F. Efficient and Robust Automated Machine Learning. In: Cortes C, Lawrence ND, Lee DD, Sugiyama M, Garnett R, editors. Advances in Neural Information Processing Systems 28. Curran Associates, Inc.; 2015. p. 2962–2970. Available from: http://papers.nips.cc/paper/5872-efficient-and-robust-automated-machine-learning.pdf.

15. Gao D, Zheng T. Support vector machine classifiers using RBF kernels with clustering-based centers and widths. IEEE International Joint Conference on Neural Networks Proceedings. 2007;2.

16. scikit-learn 0.22.2 : RBF SVM parameters;. Available from: https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html.

17. scikit-learn 0.22.2 : RandomForestClassifier;. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#.

18. scikit-learn 0.22.2 : AdaBoostClassifier;. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html.

19. scikit-learn 0.22.2 : KNeighborsClassifier;. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html.

20. Multi-layer Perceptron;. Available from: https://scikit-learn.org/stable/modules/neural_networks_supervised.html.

21. scikit-learn 0.22.2 : MLPClassifier;. Available from: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html.

22. Lorencin I, Andelic N, Spanjol J, Car Z. Using multi-layer perceptron with Laplacian edge detector for bladder cancer diagnosis. Artificial Intelligence in Medicine. 2020;102.

23. AUTO-SKLEARN;. Available from: https://www.automl.org/automl/auto-sklearn/.

24. Asch V. Macro- and micro-averaged evaluation measures [[BASIC DRAFT]]. 2013;.

25. Flach P, Kull M. Precision-Recall-Gain Curves: PR Analysis Done Right. Advances in neural information processing systems. 2015;.

26. Lipton Z, Elkan C, Naryanaswamy B. Thresholding Classifiers to Maximize F1 Score. Mahine Learning and Knowledge Discovery in Databases. 2014;8725.