The Importance of Proper Feature Selection in Diagnosing Adults with Autistic Spectrum Disorder

Marius Kleppe Larnøy
Department of Computer Science
California State University, Northridge
Northridge, United States
marius.larnoey.62@my.csun.edu

Thomas Nguyen
Department of Computer Science
California State University, Northridge
Northridge, United States
thomas.nguyen@my.csun.edu

***Abstract*—In this paper we will take a closer look on how applying four different feature selection algorithms impacts three selected supervised machine learning algorithms and their ability to correctly classify adults with Autistic Spectrum Disorder (ASD). Early detection of ASD can be a can be very important for correct treatment and further research on the underlying causes of ASD [1, 2], and machine learning as a tool to help recognize autistic traits in people of all ages.**

***Keywords*—autistic spectrum disorder, classification, supervised learning, feature selection**

***Limitations*—In this paper we have worked with a limited size dataset, and the instances collected in the set are all adults. With this in mind, we tried to keep the algorithms as general as possible, but our results only reflect testing done using an adult training dataset. Due to the limited amount of time we had to work on the project, this limited our ability to perform more sophisticated tests on the dataset.**

I. Introduction

Autism Spectrum Disorder (ASD) is considered as a brain development disorder that limits communication and social behaviors [9]. As the psychological research field grows increasingly interdisciplinary [2], application of data analysis methodologies has become a big part of the diagnostics research field [2]. Machine learning in particular has seen a rise in popularity, and in this paper, we are exploring the impact of optimizing feature selection [17] before applying supervised machine learning algorithms to a dataset.
Limitations of the referenced papers:
[1] mentions risk of overfitting due to utilizing a decision tree algorithm. [2-8] acknowledges that data used is not a perfect representation across the entire autistic spectrum and certain sub-populations. Many of the screening methods in use today is not up to date and does not follow DSM-5 [9]. [10] mentions how generalized learning models can lead to misdiagnosis, while [11] brings up the point that incorrectly choosing features with high correlation can lead to high redundancy in the dataset. [13] acknowledges some limitations of using SVM to classify, mainly that it risks being a very specialized classifier which is sensitive to outliers. [14] discusses the lack of papers on the field of ASD classification using machine learning, while also pointing out that application of ML in medicinal diagnostics can be very helpful. [15] goes into depth about the importance of feature selection, but is limited to improving the Firefly Algorithm, which we will not be using for our testing purposes. The choices in which samples to use for training in [16] could affect the result, as they opted to use machine-diagnosed samples rather than clinically.

II. Hypothesis

Feature selection is the process where one determines which features in the dataset that contributes the most in predicting the target feature and which ones are irrelevant or redundant. Performing feature selection before modeling can help to reduce overfitting, improve accuracy, and reduce training time. There are many feature selection methods that can be applied to the dataset. [7] used the chi-square testing and information gain to determine the effective features. [11] used a swarm intelligence based binary firefly feature selection wrapper to select 10 important features among 21 features of ASD dataset. [3] discussed the greedy forward feature selection with nested cross validation in order to get reliable performance.
Our hypothesis for this project is that proper feature selection can help increase both test accuracy and performance. In this paper, we propose four different feature selection methods [18]: Variance Threshold feature removal, Univariate feature selection, Feature Importance, and Pearson Correlation:

- Low-variance threshold feature removal is a technique that removes any feature whose variance is below a specified threshold.
- Univariate feature selection uses statistical tests to determine those features which have the strongest relationship with the target feature.
- Feature Importance method computes the score for each feature in the dataset. The features are then ranked by score, and we can keep a number of features based on the specified score threshold.
- Pearson correlation technique computes a correlation matrix to see the correlation of the input features with

the target feature and then selects those features which have correlation values above the specified value.

For modeling, we selected three prediction models: Decision Tree, Random Forest, and K-Nearest Neighbors (KNN) [17]. After applying data preprocessing and each feature selection method, the modified dataset is passed into the three models. The performance results will be compared for each prediction model to determine which model works best.

As this is a classification task, correlating and redundant features can drastically impact runtime and accuracy negatively as the dataset increases in size. Proper preprocessing of the data and programming with scalability in mind is therefore essential to be able to create the most accurate models.

III. Tools and Methodology

A. Tools

For this project, we have decided to utilize the following tools to perform the data preprocessing, feature selection, and model training as mentioned in the hypothesis section:

- We used the Python programming language for code development.
- We used the Python Sci-kit library for data preprocessing, feature selection, and machine learning algorithms [18].
- We used Python matplotlib and other packages to generate the plots and tables to explore and evaluate the results.

We used GitHub for version control. All the source code, documents, and research papers for each phase of the project have been properly checked in, reviewed by us. Google Docs were also utilized for efficient document sharing.

B. Methodology

As we discussed in the hypothesis section, we plan to use Variance Threshold feature removal, Univariate feature selection, Feature Importance, and Pearson Correlation for feature selections. For machine learning algorithms, we used Decision Tree, Random Forest, and KNN. Below are the steps that we developed in Python to implement the preprocessing, feature selection, and machine learning models:

Data preprocessing: for any missing data, we dropped that instances since the number of missing data instances is small compared with the total instances. The dataset's attributes consist of numerical and categorical data, we used the *LabelEncoder()* method from Sci-Kits preprocessing library to convert the categorical features into numeric values. Finally, after preprocessing was done, the data was normalized.

Feature selections: below are the Python methods from Sci-Kits feature selection library that we used for feature selection [18]:

- For Variance Threshold, we used *VarianceThreshold(threshold=(.8 * (1 - .8)))* and dropped any feature which has a threshold below 0.2.
- For Univariate method, we used *SelectKBest(score_func=chi2,...)* and dropped any feature which has a score less than 10.
- For Feature Importance, we used *ExtraTreesClassifier()* algorithm to rank the feature importance and dropped any feature which has a score less than 0.03.
- For Pearson correlation, we used *dataframe.corr()* method to generate the correlation matrix. Then we extracted the target correlation column and dropped any feature which has correlation value less than 0.2.

Note that there are no strict rules to choosing the threshold to drop the features. After running each feature selection method, we examined the score results and choose the appropriate threshold values so that we could retain the features that we think they are important for predicting models.

1. **Machine learning models:** in Python, we used DecisionTreeClassifier(), RandomForestClassifier(), and KNeighborsClassifier() for Decision Tree, Random Forest, and KNN algorithms respectively. The modified dataset was passed into these classifier methods to obtain the predicting results. Then we used the metrics.accuracy_score(y_test, y_pred) to obtain the accuracy scores. The details of these results were explained in detail in the Data and Results section.

IV. Data and Results

We used a public dataset for this project, which we collected from Kaggle.com [12], and is provided by Dr. Fadi Fayez Thabtah of the Department of Digital Technology, Manukau Institute of Technology, Auckland, New Zealand. This dataset contains 704 instances, with 21 attributes, including a binary target feature, and the samples given in this dataset are collected from autism screening of adults (18+ years old). The attributes of the dataset include binary, continuous and categorical values, and were originally collected using a research tool named ASDTests. This tool collects 20 features, 10 behavioral and 10 of individual characteristics which has been proven effective in separating ASD cases from control cases in behavior science.

The individual characteristics:

- Age (number)
- Gender (String: Male/Female
- Ethnicity (String: Country)
- Jaundice (yes/no, medical condition)
- Close family with PDD: pervasive developmental disorder (yes/no)

- Country of residence (String: country)
- Has the individual taken the test before (yes/no)
- Relation, who is performing the test (String: e.g oneself, a parent, health care professional)
- Age Description, what age range the individual is in (in this dataset, all samples are labelled ‘18 or above’)
- Screening Score, generated from the 10 behavioral features (Integer)

A. Results from Part 2

For part 2 of the project we converted the categorical data into numerical data using the LabelEncoder from Sci-kit’s preprocessing library. We then ran tests using three different supervised learning algorithms provided by Sci-kit: Decision Tree, Random Forest and KNN. We ran the tests using random sampling, with five different sample-sizes (121, 242, 363, 484 and 609). Without applying any feature selection, the mean accuracy of the three algorithms were: 88.42% (decision tree), 94.3% (forest) and 68.6% (KNN).

For our initial testing of feature selection importance in part 2 we did tests using three different feature selection tools; VarianceThreshold, SelectKBest and ExtraTreesClassifier, provided by Sci-Kit.
The feature selection algorithm we saw the most improvement with was the ExtraTreesClassifier with an exclusion threshold set at 0.03. The mean accuracy for the different machine learning algorithms were; 92.6% for Decision tree, 97.4% for Random Forest, and 96.06% for KNN. KNN saw an increase in accuracy of almost 27.5%. This supported our hypothesis of how important feature selection can be used for accuracy. When it came to performance, feature selection did not impact runtime neither positively nor negatively during our testing, but this was the expected result as the samples we used were relatively small. For Decision tree, mean runtime was 0.0024s, for forest it was 0.367s, and for KNN it was 0.012s. This held true across all tests.

B. Results from Part 3

We took a closer look at our initial preprocessing of the data, in an effort to increase accuracy and performance, and to reduce redundancy in the dataset. Before applying any of the feature selection algorithms, we went over the dataset and removed unimportant features. In particular we removed ‘age_desc’, which is a feature describing the age of the person, given in a range. As this dataset only provided instances of adults, they all had the same age description ’18 or above’. When all data in a column is equal, this means that the feature is redundant. We then converted all the categorical data into numerical data using LabelEncoder [18], and as another necessary step for reliable results, normalized all the data to a range between 0 and 1.

In addition to the three feature selection methods we applied in part 2, we added another method to give us even more ground for comparison. Our fourth feature selection method checks for

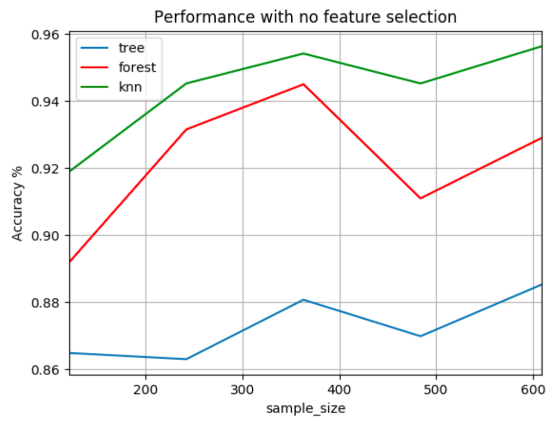

Fig. 1: Performance with improved preprocessing, no feature selection: 19 features

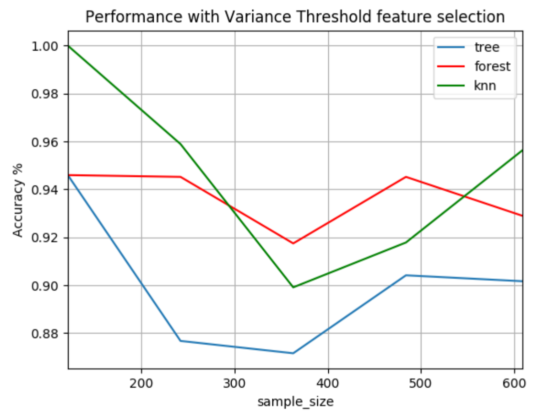

Fig. 2: Performance with improved preprocessing, variance threshold feature selection: 19 features

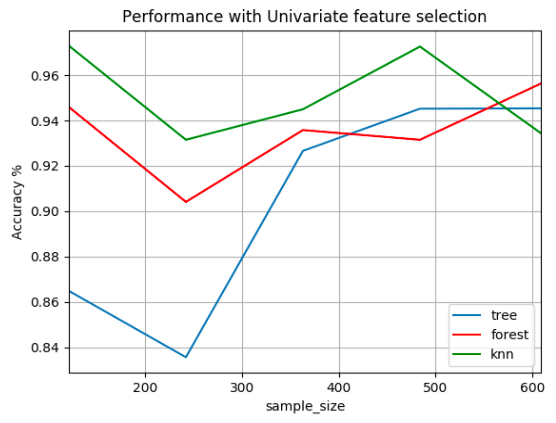

Fig. 3: Performance with improved preprocessing, univariate feature selection: 12 features

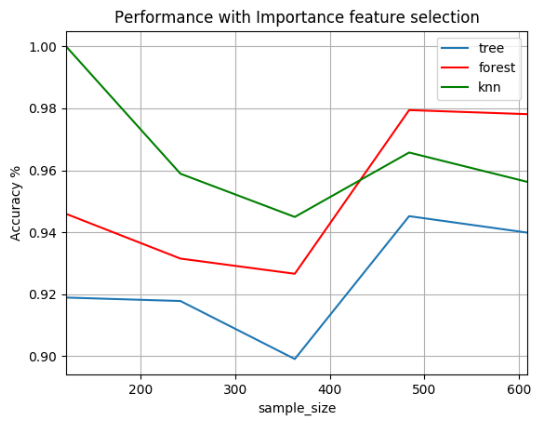

Fig. 4: Performance with improved preprocessing, importance feature selection: 11 features

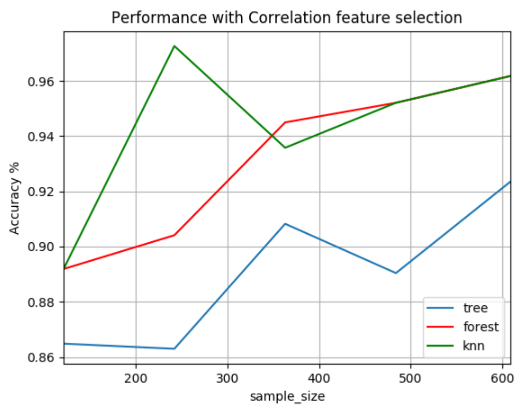

Fig. 5: Performance with improved preprocessing, correlation feature selection: 12 features

correlation between features, and keeps a top percentage given a threshold.

For our main testing, we applied each feature selection method to each machine learning algorithm in five iterations, using random sampling to perform testing on five different sample sizes. Fig. 1 displays the accuracy of each machine learning algorithm on different sample sizes, with no feature selection method applied to prune the dataset, and Table I displays the performance data of the different algorithms. The KNN algorithm benefited heavily from normalizing the dataset, which in retrospect is a step we definitely should have added during Part 2.

The Variance Threshold gave the biggest increase in accuracy for the decision tree algorithm, with a mean accuracy increase of 5% (Fig 2 and Table II). The random forest algorithm responded best in terms of accuracy from the univariate feature selection, with a mean accuracy increase of 2% (Fig 3, Table III). Lastly, the KNN algorithm received the biggest mean accuracy increase from applying Variance Threshold feature selection. On average, all three algorithms performed on the same level or better as during our last round of testing.

Performance-wise, the mean execution time went down for all algorithms compared to our results in Part 2. Comparing the runtime of Decision Tree without and with applying variance threshold feature selection, mean runtime went from 0.0034s to 0.002s, a 41% decrease, and experienced a shorter runtime for across the board after testing all the feature selection tools. While not significant for the performance using our limited sized dataset, 41% decrease is still a proof of concept which is applicable to larger datasets. The random forest algorithm had a mean runtime decrease of 1% after applying univariate feature selection, and across all the tests the random forest algorithm saw no significant performance increase. For KNN, the largest mean performance increase came from applying correlation feature selection, with a decrease of 35%. This also shows that the most accurate algorithm is not necessarily the most runtime efficient one.

The tables in the next section gives a full overview over both accuracy and performance.

V. Comparison

A. Comparing results of different feature selection methods with different predictive models.

These are the results of the classifiers without feature selection and for each of the feature selection methods. In each table, we sampled the dataset with increasing sizes and measured the running time and the accuracy score for each sample.

TABLE I. Accuracy and Performance with no feature selection: 19 descriptive features

Samples	Tree time	Tree %	Forest Time	Forest %	KNN Time	KNN %
121	0.011	0.838	0.250	0.946	0.009	0.973
242	0.002	0.808	0.251	0.945	0.004	0.918
363	0.001	0.890	0.238	0.954	0.005	0.963
484	0.001	0.904	0.237	0.938	0.006	0.938
609	0.002	0.891	0.233	0.929	0.007	0.956
Mean	**0.0034 s**	**86.62 %**	**0.2418 s**	**94.24 %**	**0.0062 s**	**94.96 %**

TABLE II. Accuracy and Performance with Variance Threshold feature selection: 19 descriptive features

Samples	Tree time	Tree %	Forest Time	Forest %	KNN Time	KNN %
121	0.001	0.838	0.256	1.000	0.003	1.000
242	0.002	0.877	0.244	0.659	0.004	0.959
363	0.002	0.853	0.237	0.945	0.006	0.954
484	0.002	0.856	0.241	0.904	0.008	0.952
609	0.003	0.880	0.256	0.929	0.009	0.956
Mean	**0.002s**	**86%**	**0.247s**	**88.7%**	**0.006s**	**96.42%**

TABLE III. ACCURACY AND PERFORMANCE WITH UNIVARIATE FEATURE SELECTION: 12 DESCRIPTIVE FEATURES

Samples	Tree time	Tree %	Forest Time	Forest %	KNN Time	KNN %
121	0.002	0.856	0.245	1.000	0.004	1.000
242	0.002	0.904	0.245	0.959	0.003	0.973
363	0.001	0.890	0.236	0.963	0.005	0.945
484	0.001	0.966	0.235	0.938	0.006	0.959
609	0.001	0.934	0.234	0.956	0.006	0.934
Mean	**0.0014s**	**91%**	**0.239s**	**96.3%**	**0.0048s**	**96.2%**

TABLE IV. ACCURACY AND PERFORMANCE WITH IMPORTANCE FEATURE SELECTION: 11 DESCRIPTIVE FEATURES

Samples	Tree time	Tree %	Forest Time	Forest %	KNN Time	KNN %
121	0.001	0.892	0.236	0.838	0.003	0.946
242	0.001	0.849	0.234	0.918	0.003	0.932
363	0.001	0.890	0.236	0.954	0.004	0.963
484	0.001	0.818	0.244	0.932	0.005	0.938
609	0.001	0.934	0.237	0.978	0.006	0.956
Mean	**0.001s**	**87.66%**	**0.238s**	**92.4%**	**0.0042s**	**94.7%**

TABLE V. ACCURACY AND PERFORMANCE WITH CORRELATION FEATURE SELECTION: 12 DESCRIPTIVE FEATURES

Samples	Tree time	Tree %	Forest Time	Forest %	KNN Time	KNN %
121	0.001	0.811	0.246	0.838	0.002	0.919
242	0.001	0.890	0.237	0.945	0.003	0.932
363	0.001	0.890	0.240	0.945	0.004	0.954
484	0.002	0.911	0.240	0.945	0.005	0.959
609	0.001	0.923	0.238	0.962	0.006	0.962
Mean	**0.0012s**	**88.5%**	**0.24s**	**92.7%**	**0.004s**	**94.52%**

Performance wise, the Decision Tree was the fastest one, then KKN is second, and Random Forest is the worst. It is reasonable because the Random Forest algorithm in this paper is an ensemble of 100 decision trees. The KNN algorithm is slower than Decision Tree algorithm since constructing a binary tree and search the tree is faster than computing Euclidean distances and comparing the distance of all samples to obtain the k nearest neighbors to predict a query sample. While our testing does not include sample sizes large enough to cause performance issues, it is important to keep in mind for further application of these concepts. Proper feature selection can therefore help with limiting the chance of having to pick performance over accuracy, a potentially damaging choice for the outcome of model predictions made in a practical setting.

Each of the machine learning algorithms improved their mean accuracy in one or more cases of feature selection application. Decision Tree and KNN gained the most accuracy from the Variance Threshold method, and the Random Forest algorithm increased the most after applying univariate feature selection. It is important to note that while this was the case for our results, increased knowledge about when to apply different types of feature selection and comparing different feature selection methods is recommended to be able to train the most accurate models.

B. *Comparing with other's previous works*

In the previous papers, different feature selection methods with different machine learning models were used to reduce the number of input features and evaluated the performance of these methods. [7] used Information Gain and Chi-Square feature selection methods with Logistic Regression model for classification, [15] used Firefly Algorithm for feature selection with classification and regression models, [16] used backward feature selection with SVM (Support Vector Machine) predictive model. These feature selection methods from these papers achieved the performance similar of what we have achieved in this paper. However, our feature section methods: Variance Threshold feature removal, Univariate feature selection, Feature Importance, and Pearson Correlation were simple since these feature selection methods don't require to have a predictive mode to work, they are more efficient.

VI. CONCLUSION

In this paper, we explored four different feature selection methods: Variance Threshold feature removal, Univariate feature selection, Feature Importance, and Pearson Correlation and applied these methods to reduce the number of features of the ASD adult dataset. Our results demonstrate why feature selection is an important step of preprocessing while applying machine learning algorithms. When we get a dataset, some of the features are irrelevant and redundant for predicting the target, and these features need to be removed in order to reduce overfitting, improve accuracy, and reduce training time. As shown in the "data and results" and "comparison" sections, the results from applying the feature selection methods clearly show the benefits of using these methods while preprocessing the data. These results confirm our hypothesis.

Applying machine learning to the medical diagnostics field can be very helpful but requires caution. The algorithms lack vital knowledge that medical professionals have, and while ML can help to reduce assessment time of patients with potential ASD, it is important to have a critical view when utilizing such powerful tools.

There are other feature selection methods such as Wrapper methods [11] or Embedded methods [7, 9]. Wrapper methods requires a predictive model to be used to evaluate the combination of features and assign a score based on model accuracy. Embedded methods such as LASSO is a regularization method that uses optimization with constraints for predictive algorithms. This paper can be extended to include these selection methods and compare the results with the methods being used in this current paper.

References

[1] M. B. Usta *et al.*, "Use of machine learning methods in prediction of short-term outcome in autism spectrum disorders," *Psychiatry and Clinical Psychopharmacology,* vol. 29, no. 3, pp. 320-325, 2019/07/03 2019, doi: 10.1080/24750573.2018.1545334.

[2] D. Bone, M. Goodwin, M. Black, C.-C. Lee, K. Audhkhasi, and S. Narayanan, "Applying Machine Learning to Facilitate Autism Diagnostics: Pitfalls and Promises," *Journal of Autism and Developmental Disorders,* vol. 45, no. 5, pp. 1121-1136, 2015.

[3] D. Bone, S. L. Bishop, M. P. Black, M. S. Goodwin, C. Lord, and S. S. Narayanan, "Use of machine learning to improve autism screening and diagnostic instruments: effectiveness, efficiency, and multi-instrument fusion," *Journal of Child Psychology and Psychiatry,* vol. 57, no. 8, pp. 927-937, 2016.

[4] D. P. Wall, J. Kosmicki, T. F. Deluca, E. Harstad, and V. A. Fusaro, "Use of machine learning to shorten observation-based screening and diagnosis of autism," Translational psychiatry, vol. 2, no. 4, p. e100, 2012.

[5] D. P. Wall, R. Dally, R. Luyster, J.-Y. Jung, and T. F. DeLuca, "Use of Artificial Intelligence to Shorten the Behavioral Diagnosis of Autism (Fast and Accurate Behavioral Detection of Autism)," vol. 7, no. 8, p. e43855, 2012.

[6] M. Duda, R. Ma, N. Haber, and D. P. Wall, "Use of machine learning for behavioral distinction of autism and ADHD," *Translational Psychiatry,* vol. 6, no. 2, p. e732, 2016.

[7] F. Thabtah, N. Abdelhamid, and D. Peebles, "A machine learning autism classification based on logistic regression analysis," *Health Information Science and Systems,* vol. 7, no. 1, pp. 1-11, 2019.

[8] M. N. Parikh, H. Li, and L. He, "Enhancing Diagnosis of Autism With Optimized Machine Learning Models and Personal Characteristic Data," *Frontiers in Computational Neuroscience,* vol. 13, 2019.

[9] F. Thabtah, "Autism Spectrum Disorder Screening: Machine Learning Adaptation and DSM-5 Fulfillment," vol. 129311, ed, 2017, pp. 1-6.

[10] K. K. Hyde *et al.*, "Applications of Supervised Machine Learning in Autism Spectrum Disorder Research: a Review," *Review Journal of Autism and Developmental Disorders,* vol. 6, no. 2, pp. 128-146, 2019.

[11] V. Ravindranath and S. Ra, "A machine learning based approach to classify Autism with optimum behaviour sets," *International Journal of Engineering and Technology,* vol. 7, 2018.

[12] Faizunnabi."Autism Screening." https://www.kaggle.com/faizunnabi/autism-screening (accessed 10/03, 2019).

[13] Alessandro Crippa *et al.*, "Use of Machine Learning to Identify Children with Autism and their Motor Abnormalities" *J Autism Dev Discord*

[14] Bram van den Bekerom "Using Machine Learning for Detection of Autism Spectrum Disorder" *University of Twente, The Netherlands,* 20th Student Conf. IT, 2017.

[15] Li Zhang *et al.*, "Feature Selection using firefly optimization for classification and regression models" *Decision Support Systems (2017).*

[16] JA Kosmicki *et al.*, "Searching for a minimal set of behaviors for autism detection through feature selection-based machine learning" *Transl Psychiatry (2015) 5, e514; doi:10.1038/tp.2015.7.*

[17] J.D. Kelleher, B.M Namee, A. D'Arcy, *Fundamentals of Machine Learning For Predictive Data Analytics,* Massachusetts Institute of Technology, MIT Press, 2015.

[18] Pedregosa *et al.*, Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-2830, 2011.