Multidiszciplináris tudományok, 10. kötet. (2020) 3 sz. pp. 212-220 **https://doi.org/10.35925/j.multi.2020.3.26**

EFFICIENCY COMPARISON OF PYTHON AND RAPIDMINER

Hina Ghous
PhD student, Institute of Information Technology, University of Miskolc, Hungary
3515 Miskolc, Miskolc-Egyetemváros, e-mail: hs.hinashaikh@gmail.com

László Kovács
professor, Institute of Information Technology, University of Miskolc, Hungary
3515 Miskolc, Miskolc-Egyetemváros, e-mail: kovacs@iit.uni-miskolc.hu

Abstract
Machine learning is an important technique that helps companies, institutes, and humans to improve the quality of decision making. As there are many different free tools on the Internet for machine learning, we need comparisons, benchmarks to provide help in the selection of the appropriate analysis technique. This paper aims at providing a comparative study of the two most powerful and open-source machine learning tools Python and RapidMiner by using most common supervised machine learning techniques Decision Tree, Support Vector Machines, and Neural Networks on data analytic of PIMA Indian Diabetes Dataset and Mushroom Classification Dataset. The results reveal that the usage of RapidMiner provides better performance in terms of both accuracy and execution time.

Keywords: Machine learning, Classification benchmark, Python, RapidMiner

1. Introduction

Machine learning (ML) is an artificial intelligence technology (AI) in which computer programs analyzes the observation data to discover hidden relationships. Machine learning is seeing increasing usage across industries for many factors. Enormous amounts of data are being collected and can be processed digitally in a cost-effective way. Because of the improved computational capacity accessible today at competitive rates, there are numerous open-source platforms, toolkits, and libraries that can be used to develop and operate ML applications.

ML also contributed to promising technological advancements, especially in healthcare, that could redefine diagnosis and treatment in the years ahead. Scientists are already working on ML models that forecast vulnerability to disease or help in the early detection of diseases. Data mining is the approach utilized as a part of this article, whereby using advanced methods for data processing to identify previously undetected relationships between data objects. Data mining consists of three key techniques: regression, classification, and clustering. The heart of this study is based on a classification problem, to predict diabetes in a patient using different machine learning algorithms in two amazing tools Python and Rapid Miner [1].

This research article focuses on the following popular machine learning algorithms: Decision Tree, Neural Networks and Support Vector Machine (SVM). These algorithms will be used for two different sizes of datasets for diabetes detection and eatable mushrooms. The dataset named "PIMA Indian Diabetes" and "Mushroom Classification" was taken from the kaggle.com, a Machine Learning Repository [18-19]. In particular, Python and RapidMiner are chosen to evaluate the datasets and make a

comparison based on the accuracy and performance matrix. The key goal is to select the best tool for future analyzes of data medical diagnoses.

2. Literature Survey of Related Works

Machine learning has been applied to many health-care applications, a variety of studies was conducted and a significant amount of work has been performed in the classification area. Many of the researches included a real question of how to evaluate the efficiency of supervised learning algorithms and classifiers, or which tools can be considered good to invest and can enhance the performance of classifiers. In [2], a comparative study was conducted on 20 different real datasets, to evaluate the two tools Matlab and OpenCV by applying some machine learning techniques. The authors wanted to explore similar environments to identify the best algorithm for better results.

In [3], a comparative analysis has been done on the Dialysis dataset using two popular tools 'Python vs Weka'. The key purpose was to choose which tool performs well by using machine learning algorithms. The authors used Decision Tree, KNN, Naïve Bayes and SVM algorithms in Python and Weka and the results revealed that the use of Python offers the best performance in terms of correct/incorrect instances, accuracy, and recall. In [4], the authors performed some experiments on twitter data by using three diverse classifiers techniques: KNN, SVM, and NB algorithms and the results showed that the SVM classifier obtained the most satisfactory result with ITC as the function, where Precision, Recall and F1-Score were all obtained at 95%. In [5], the authors used two separate environments, called RapidMiner and WEKA, to examine comparatively the effects of classification and clustering of SMS spam management. The same dataset was analyzed in both environments and experiments were carried out using different machine learning techniques and the outcome of both the environments was the same, considering SVM as the best machine learning technique.

A broader comparative study was done in [6, 7], the authors implemented several machine learning techniques like Artificial Neural Networks (ANN), k-Nearest Neighbors (KNN) and Bayesian Network Classifiers to compare and evaluate which technique gives better results.

In their paper, Christa et al. [8] examined and did a comparative study on different data mining methods such as KNIME, RapidMiner, Weka, Tanagra, and Orange. These methods have their features, yet special advantages. RapidMiner and Weka have the strongest and most active consumer groups than the other data mining applications.

3. Classification Implemented Using Supervised Machine Learning Algorithms

Supervised learning, according to Brownlee [9], inspects input data called training data and has a predefined label or outcome. A model is designed through a training process in which predictions must be made and corrected if those predictions are incorrect. The training process keeps going until a desired level of accuracy is achieved on the training data. Classification and regression are examples of problems that apply supervised learning. The preceding highlights on various supervised learning algorithms are used in this study.

3.1. Decision Tree

A decision tree is an algorithm for learning, using a "divide and conquer" technique to distinguish instances. It is comprised of leaf nodes marked with a class and test nodes linking two or more subtrees. Each test node calculates a result based on a specific attribute, and that result decides how deep a tree

can go. According to the key concept of the decision tree is to divide the unknown data recursively until every data belongs to a specific class. In general, the algorithm for the decision tree is implemented in 2 phases. Tree-building is the first phase and in that phase, the tree is divided until all the data have its class in a top-down fashion. The second phase is tree pruning, where predictions and accuracy are improved in a bottom-up fashion.

3.2. Support Vector Machine

Support Vector Machine (SVM) has been used to solve the problem of pattern recognition. SVM is a learning algorithm with a classification method performed in 2 steps. The first step is to map all data into n-dimension space of training set, where n is the number of features and values of features are the value co-ordinates. The second step is to identify the hyper-plane which differentiates between the two classes. The classification is performed by creating a proper hyperplane among instances of different classes. According to the article which classifies patients as diabetic or non-diabetic according to the following dot product:

$$y = \bar{w}\bar{x} - b \tag{1}$$

where $\bar{x}$ is a feature vector of patients with their attributes and $\bar{w}$ is the normal vector of the separation plane and b is a bias parameter obtained by the training process.

3.3. ANN Framework

Bishop in his study [12] has discussed many types of ANN models, where a feed-forward neural network with stochastic gradient descent using the back-propagation algorithm was most popular among all. This type of neural network is referred to as a supervised network and generally used for prediction, pattern recognition and mapping. The purpose of this type of neural network is to build a model that maps the input correctly to the output using training data so that then model can help in predicting the output when the desired output is uncertain. A feed-forward with the back-propagation typically operates on the Hidden States which serves as a bridge between an input layer and an output layers with one or more hidden layers in-between, consisting of several neurons. Assume a hidden layers x and an output layer y, and a w_{xy} relation weight between them. First, the neuron determines a weighted total of its inputs and then calculates the output using an activation function. From the output layer, the training system then carefully re-calibrate the error backward by changing all the weights of the neuron in the layers of the network, and then sends back to the training data (each step of this is called a training epoch). The cycle continues until an appropriate level of error tolerance is reached, at which the network is said to be trained.

3.4. Tools

This entire classification process can be achieved with the assistance of defined algorithms built using licensed as free or commercial software products. In this experiment, two popular data analyst tools with free license (freeware) were used for the implementation of this process; Python and RapidMiner Studio.

3.4.1. Python

Python is a high-level, general-purpose, object-oriented, and integrated programming language. Python is known as the comprehensive library of open-source data mining tools, web applications, and machine learning. Python packed with several plugins and extensions to assist different tasks for large active communities [20]. Python has proven to be an effective programming language [20] and shows many applications in scientific calculations Python Package Index (PyPI) runs thousands of Python third party packages. High-quality libraries and open source tools render Python as an excellent choice for machine learning model development. The large range of libraries for machine learning comprises TensorFlow, Keras, PyTorch, NLTK, Theano Python, Seaborn, Scikit-learn, and Numpy, etc. This study uses the Jupiter Notebook (Python 3) for model implementations and Python libraries used for building machine learning models are Numpy, Pandas, Scikit-Learn, and Matplotlib.

3.4.2. RapidMiner

RapidMiner is an open-source tool for data mining that can be used as a stand-alone framework for data analysis or embedded into other software as a data mining tool. RapidMiner is a powerful tool for data integration, analytical Extract Transform Load (ETL), data analysis, and reporting into a single suite. RapidMiner has a very effective Graphical User Interface (GUI) for the design of analytical processes. It contains a variety of Repositories for process, operators, data and helps in metadata management. It helps in bugs fixing and error detection. It is a visualization tool, easy to use without coding and it is a complete and versatile package, containing hundreds of available approaches for data integration, machine learning, and simulation.

4. Experimental Design

The two datasets with different sizes used in this experiment are taken from kaggle.com and Machine Learning Repository [18-19]. The smaller dataset “Pima Indian Diabetes” contains 768 Indian female patients, aged from at least 20 – 21. The conditional response feature takes two classes '0' or '1. Where ‘1’ represents patients is positive for diabetes and ' 0 ' represents patient is negative for diabetes. 268 (37.8%) cases belongs to Class ' 1 ' and 500 (62.2%) cases belongs to Class ' 0 '. The second dataset is much larger contains 8125 instances with 22 attributes and the conditional-response feature takes two classes edible=e or poisonous=p. All the instances in the datasets are used for both training and testing phases in a ratio of 7:3, 70% data for the training dataset, and 30% data for testing dataset.

As to explain the experiment, Figure 1, demonstrates the process, highlights the functionality of the classification process, machine learning algorithms, and data mining tools, Python and RapidMiner.

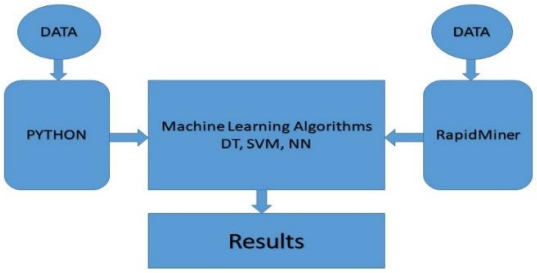


Figure 1. *Classification processes with machine learning algorithms and experimental tools.*

As for having a fair comparison, the same datasets are deployed in Python and RapidMiner using three machine learning algorithms listed above to find the best fitting algorithm for these two tools.

As for the data mining tools, this project is using Jupiter Notebook (Python 3) and RapidMiner (version 9.3.1.0.).

For the classification process, the datasets were divided into two sets, training dataset, and testing dataset. The training and testing dataset represents supervised machine learning features.

For both the tools, all classification model were built, during the training process, a set of positive outcomes and negative outcomes are running through each model classifier separately. Then for testing, based on the results stored from the training process, a testing dataset is running through those models. Finally, to evaluate its performance, the outcomes of those observations are examined.

Decision Tree and Support Vector Machine models are built using the Scikit-learn library from python. The Decision tree model was created using the "Gini-Index" feature in both tools and for SVM, a linear kernel is used. ANN framework was created using deep learning techniques. For Python, Keras library was used to build the model with 3 hidden layers in the manner (12-8-1) with "relu" and "sigmoid" activation function, and the model was fitted using 150 epochs.

5. Results and Discussion

This section analyzes the results from the experiments performed on PIMA Indian diabetes dataset and Mushroom classification dataset in different setting tools and algorithms. This study compares various classification outcomes, considering the accuracy percentage of the number of correctly classified instances. A comparison of the algorithms is performed using different parameters such as accuracy, precision, recall, F1-score, execution time. The comparisons are tabled out as below:

Table 1. *Comparison of the Performance*

Performance features	Machine Learning Algorithms	Python (PIMA)	RapidMiner (PIMA)	Python (Mushroom)	RapidMiner (Mushroom)
Accuracy	Decision Tree	67.96%	75.22%	100%	98.69%
	SVM	79.22%	77.39%	97.82%	78.79%
	ANN	77.99%	78.70%	100%	99.30%
Precision	Decision Tree	57.74%	63.86%	100%	97.53%
	SVM	79.36%	72.28%	96.22%	95.04%
	ANN	69.72%	71.23%	100%	99.84%
Recall	Decision Tree	48.23%	66.25%	100%	100%
	SVM	58.82%	56.25%	99.40%	62.28%
	ANN	65.29%	65.00%	100%	98.81%
F1-score	Decision Tree	52.56%	65.03%	100%	98.75%
	SVM	67.56%	63.38%	97.79%	63.38%
	ANN	67.43%	67.97%	100%	99.32%
Execution Time	Decision Tree	14.99 sec	0.037 sec	14.99 sec	0.025 sec
	SVM	17.75 sec	0.031 sec	45.88 sec	0.040 sec
	ANN	74.77 sec	0.031 sec	2108.0sec	0.178 sec

The tables below compare the Python and RapidMiner using 5 different performance factors which are accuracy, precision, recall, F1-score, and execution time. Table 3 deals with the accuracy of the models based on how many correct instances are in both tools. The table below shows that RapidMiner gave the best results in Decision Tree and ANN algorithms and Python is good in predicting using the SVM algorithm.

Table 3. *Comparison of Accuracy*

Machine Learning Algorithm	Python (PIMA)	RapidMiner (PIMA)	Python (Mushroom)	RapidMiner (Mushroom)
Decision Tree	57.74%	75.22%	100%	98.69%
SVM	79.36%	77.39%	97.82%	78.79%
ANN	69.72%	78.70%	100%	99.30%

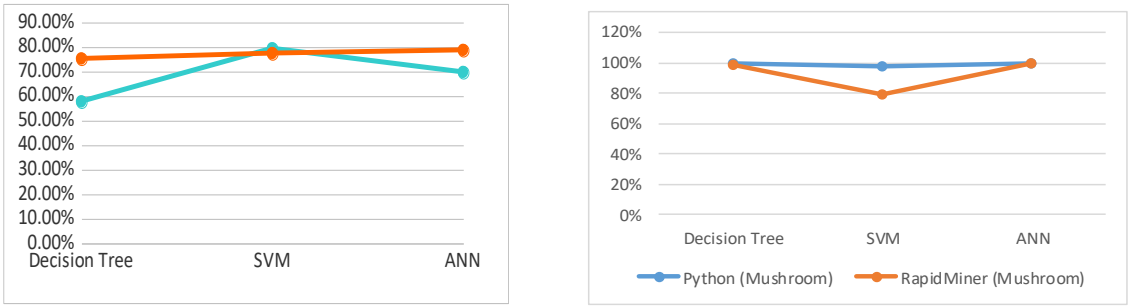


Figure 2*. Graph Visualization of comparison of accuracy*

Table 4 shows the Precision results of both Python and RapidMiner. As it is observed, the tool with the highest precision is Python, because in Python normal and abnormal classification mostly takes values equal to 1 and RapidMiner deals with some negative values.

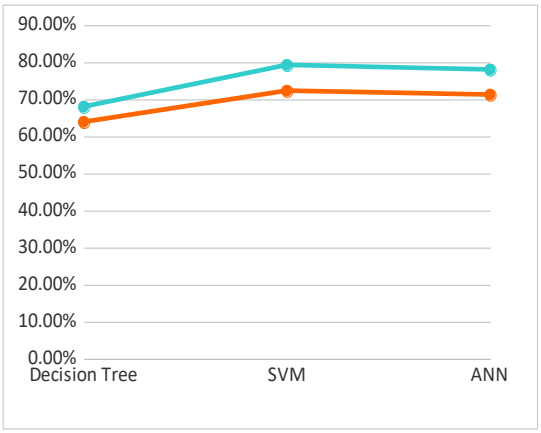
Table 4. *Comparison of Precision*

Machine Learning Algorithm	Python (PIMA)	RapidMiner (PIMA)	Python (Mushroom)	RapidMiner (Mushroom)
Decision Tree	67.96%	63.86%	100%	97.53%
SVM	79.22%	72.28%	96.22%	95.04%
ANN	77.99%	71.23%	100%	99.84%

Table 5 focuses on how many true instances are predicted correctly. It can be perceived from the table that Python performance well for SVM and ANN in terms of recall because in python recall rate equal to 1.

Table 5. *Comparison of Recall*

Machine Learning Algorithm	Python (PIMA)	RapidMiner (PIMA)	Python (Mushroom)	RapidMiner (Mushroom)
Decision Tree	48.23%	66.25%	100%	100%
SVM	58.82%	56.25%	99.40%	62.28%
ANN	65.29%	65.00%	100%	98.81%



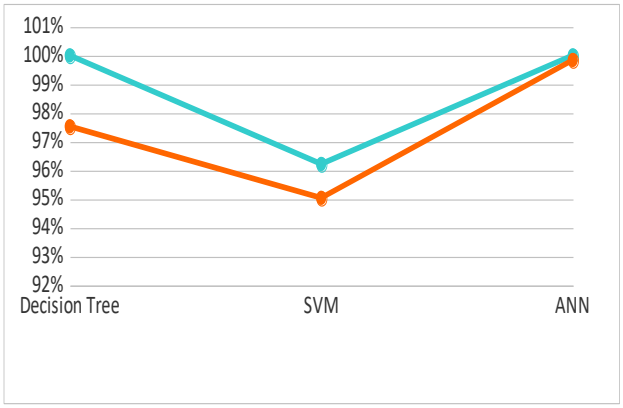


***Figure 3**. Graph Visualization of Comparison of Precision*

Table 6 focuses on the comparison of both the tools using the F1-score. By looking at the table, it can be considered as RapidMiner is a good approach for F1-score for the smaller dataset and approximately equal score for the larger dataset.

***Table 6.** Comparison of F1-score*

Machine Learning Algorithm	Python (PIMA)	RapidMiner (PIMA)	Python (Mushroom)	RapidMiner (Mushroom)
Decision Tree	52.56%	65.03%	100%	98.75%
SVM	67.56%	63.38%	97.79%	63.38%
ANN	67.43%	67.97%	100%	99.32%

Table 7 deals with the execution time, including the time taken for a model to train and test, the whole process. As it is known, the smaller the execution time, the more efficient the classifier is. For this aspect, RapidMiner has good performance because RapidMiner is pre-optimized, unlike Python.

***Table 7.** Comparison of Execution Time*

Machine Learning Algorithm	Python (PIMA)	RapidMiner (PIMA)	Python (Mushroom)	RapidMiner (Mushroom)
Decision Tree	14.99 sec	0.037 sec	14.99 sec	0.025 sec
SVM	17.75 sec	0.031 sec	45.88 sec	0.040 sec
ANN	74.77 sec	0.031 sec	2108.0sec	0.178 sec

It can be summarized from the above comparisons that, if we consider accuracy and execution time, then RapidMiner can be the best option to consider. However, it seems that Python has the highest performance in terms of accuracy in some algorithms but RapidMiner gives very little execution time and provides a more detailed process of the model.

6. Conclusion and future work

In this paper, we compared the accuracy and cost-efficiency of two known machine learning tools, Python and RapidMiner. For the test we have used two datasets, PIMA Indian Diabetes Dataset (smaller) and Mushroom classification (larger) datasets to evaluate the performance by using different machine learning algorithms including Decision Tree, Support Vector Machine (SVM), and Artificial Neural Networks (ANN). Rapid Miner and Python have shown similar performance in terms of accuracy. But regarding the execution time, there was a big difference in two tools, regardless of the size of datasets Python required significantly more execution time. Therefore, it is important to improve Python's interpreter and render it for more efficiency in the future.

7. Acknowledgement

The described article was carried out as part of the EFOP-3.6.1.-16-2016-00011 "Younger and Renewing University – Innovative Knowledge City institutional development of the University of Miskolc aiming at intelligent specialization " project implemented in the framework of the Szechenyi 2020 program. The realization of the project is supported by the European Union, co-financed by the European Social Fund.

References

[1] Dwivedi, S., Kasliwal, P., and Soni, S.: Comprehensive study of data analytics tools (RapidMiner, Weka, R tool, Knime), in Editor (Ed.): 'Book Comprehensive study of data analytics tools (RapidMiner, Weka, R tool, Knime)' (IEEE, 2016, edn.), pp. 1-8. **https://doi.org/10.1109/CDAN.2016.7570894**

[2] Elsayed, A.A., and Yousef, W.A.: Matlab vs. opencv: A comparative study of different machine learning algorithms, arXiv preprint arXiv:1905.01213, 2019

[3] Mitrpanont, J., Sawangphol, W., Vithantirawat, T., Paengkaew, S., Suwannasing, P., Daramas, A., and Chen, Y.-C.: A study on using Python vs Weka on dialysis data analysis, in Editor (Ed.): 'Book A study on using Python vs Weka on dialysis data analysis' (IEEE, 2017, edn.), pp. 1-6. **https://doi.org/10.1109/INCIT.2017.8257883**

[4] Tiun, S.: Experiments on Malay short text classification, in Editor (Ed.): 'Book Experiments on Malay short text classification' (IEEE, 2017, edn.), pp. 1-4. **https://doi.org/10.1109/ICEEI.2017.8312371**

[5] Zainal, K., Sulaiman, N., and Jali, M.: An analysis of various algorithms for text spam classification and clustering using RapidMiner and Weka. International Journal of Computer Science and Information Security 2015, 13(3):66.

[6] Potdar, K., and Kinnerkar, R.: A comparative study of machine learning algorithms applied to predictive breast cancer data. International Journal of Science and Research 2016, 5(9):1550-1553.

[7] Palaniappan, R., Sundaraj, K., and Sundaraj, S.: A comparative study of the svm and k-nn machine learning algorithms for the diagnosis of respiratory pathologies using pulmonary acoustic signals. BMC Bioinformatics 2014, 15(1):223. **https://doi.org/10.1186/1471-2105-15-223**

[8] Christa, S., Madhuri, K.L., and Suma, V.: A comparative analysis of data mining tools in agent based systems, arXiv preprint arXiv:1210.1040, 2012

[9] Brownlee, J.: A tour of machine learning algorithms. Machine Learning Mastery 2013:25.
[10] Suykens, J.A., and Vandewalle, J.: Least squares support vector machine classifiers. Neural Processing Letters 1999, 9(3):293-300. **https://doi.org/10.1023/A:1018628609742**
[11] El-Halees, A.M.: Filtering Spam E-Mail from Mixed Arabic and English Messages: A Comparison of Machine Learning Techniques, Filtering Spam E-Mail from Mixed Arabic and English Messages: A Comparison of Machine Learning Techniques., 2009, 6(1).
[12] Bishop, C.M.: Neural networks for pattern recognition (Oxford university press, 1995). **https://doi.org/10.1201/9781420050646.ptb6**
[13] White, H.: Learning in artificial neural networks: A statistical perspective. Neural Computation 1989, 1(4):425-464. **https://doi.org/10.1162/neco.1989.1.4.425**
[14] Oliphant, T.: Python for Scientific Computing Computing in Science & Engineering, 2007. **https://doi.org/10.1109/MCSE.2007.58**
[15] Perez, F., Granger, B.E., and Hunter, J.D.: Python: an ecosystem for scientific computing. Computing in Science & Engineering 2010, 13(2):13-21. **https://doi.org/10.1109/MCSE.2010.119**
[16] Kotu, V., and Deshpande, B.: Predictive analytics and data mining: concepts and practice with rapidminer (Morgan Kaufmann, 2014. 2014) **https://doi.org/10.1016/B978-0-12-801460-8.00013-6**
[17] Wang, A.H.: Machine learning for the detection of spam in twitter networks, in Editor (Ed.): 'Book Machine learning for the detection of spam in twitter networks' (Springer, 2010, edn.), pp. 319-333. **https://doi.org/10.1007/978-3-642-25206-8_21**
[18] Kaggle.com Machine Learning Repository. URL: https://www.kaggle.com/uciml/pima-indians-diabetes-database Accessed: 22 December, 2020
[19] Kaggle.com Machine Learning Repository. URL: https://www.kaggle.com/uciml/mushroom-classification Accessed: 12 June, 2020]
[20] M. Makai. Why Use Python? - Full Stack Python, 2017. URL: https://www.fullstackpython.com/why-use-python.html Accessed: 6 April, 2020