IEREK PRESS
Academic Research Community

http://www.press.ierek.com

ARChive

CrossMark

ISSN (Print: 2537-0154, online: 2537-0162)

International Journal on:

The Academic Research Community Publication

DOI: 10.21625/archive.v3i1.442

# [ANT]: A Machine Learning Approach for Building Performance Simulation: Methods and Development

**Mahmoud M. Abdelrahman[1], Ahmed Mohamed Yousef Toutou[2]**

[1]*Kafrelsheikh University - Kafrelsheikh - Egypt. The American University in Cairo - Cairo – Egypt*

[2]*Master Student in Alexandria University, B.Sc. Mansoura University*

## Abstract

In this paper, we represent an approach for combining machine learning (ML) techniques with building performance simulation by introducing four methods in which ML could be effectively involved in this field i.e. Classification, Regression, Clustering and Model selection . Rhino-3d-Grasshopper SDK was used to develop a new plugin for involving machine learning in design process using Python programming language and making use of scikit-learn module, that is, a python module which provides a general purpose high level language to non-specialist user by integration of wide range supervised and unsupervised learning algorithms with high performance, ease of use and well documented features. ANT plugin provides a method to make use of these modules inside Rhino\Grasshopper to be handy to designers. This tool is open source and is released under BSD simplified license. This approach represents promising results regarding making use of data in automating building performance development and could be widely applied. Future studies include providing parallel computation facility using PyOpenCL module as well as computer vision integration using scikit-image.



***Keywords***

Machine Learning; ML; building performance simulation; Rhino3d; Python; Scikit-learn; grasshopper.

## 1. Introduction

### 1.1. Machine learning in architecture and design.

Machine learning could be defined as: ”Learning from data” (Abu-Mostafa, Magdon-Ismail, and Lin 2012) by considering a set of n samples of data which is used to predict properties of unknown data, each sample could have more than one variable (multivariable data), these variables are called features. Learning problems could be separated into two main categories i.e. supervised and unsupervised learning, the former consists of training data and targets set by users and the machine role is to classify a new unlabeled input data (classification) or to predict an output value of continuous data type (regression). While the latter only consists of data without targets and, in this case, the machine role is to predict a relation between these data (clustering) or determine the distribution of data within input space (density estimation) or to project data from multi-dimensional space into 2d or 3d space (Pedregosa et al. 2011).In architecture and design, machine learning has been increasingly used in several studies during the last two decades due to the complexity and non-linearity of real-world multi-variable relations. Its applications in architecture and design are rapidly mutating.

By reviewing the CumInCAD database ( a Cumulative Index about publications in Computer Aided Architectural Design supported by the sibling associations ACADIA, CAADRIA, eCAADe, SIGraDi, ASCAAD and CAAD futures ), it has been monitored that machine learning topics (including supervised, unsupervised and convolution neural networks CNN) are growing over the past few years. Figure (1) shows the number of papers which are related to this topic.

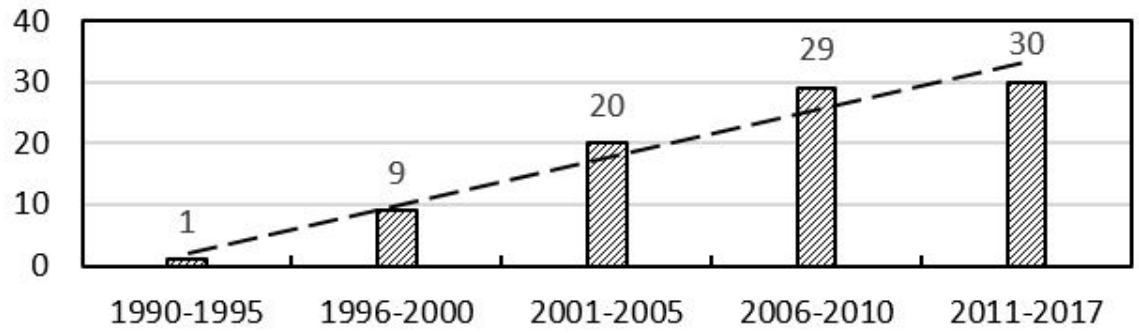


Figure 1. Machine learningrelated papers by year (collected from CumInCAD website).

Table 1. Review of recent architecture-and-design-basedMachine Learning articles.

| Category | Reference |
|---|---|
| Architecture and design | (Vardouli 2013, Paterson et al. 2013, Cutellic and Lotte 2013, Bernhard 2013, Fernando et al. 2010, Tamke et al. 2016, Silvestre, Ikeda, and Guéna 2016, Stasiuk, Thomsen, and Thompson 2014) |
| Urban design | (Chen et al. 2017, Laskari 2014, Sokmenoglu, Cagdas, and Sarıyıldız 2010, Wassermann 2010, Stouffs et al. 2013, Carlos Sandoval Olascoaga 2016, Oh et al. 2006) |
| Robotics, Automation and Fabrication | (Decker 2015, Pinochet 2016, Chi-Li Cheng 2016, Arenas and Falcón 2013, Harrison 2016) |
| Building performance and Simulation | (Wilkinson et al. 2013, Merrick, Maher, and Saunders 2008, Mahalingam 2005, Khosrowpour, Gulbinas, and Taylor 2016, Davis 2016, Standfest 2014, de Wilde et al. 2013, Yildiz, Bilbao, and Sproul 2017, Paudel et al. 2017, Kontokosta and Tull 2017, Banihashemi, Ding, and Wang 2017, Deb et al. 2016, Cui et al. 2016, Chou and Ngo 2016, Zarkadis, Ridi, and Morel 2014) |

In spite of its emerging importance and popularity, Machine learning requires a good knowledge in programming languages which is not affordable to all designers and engineers. However, visual programming tools such as Rhino\Grasshopper recently has gained increasing interest by a large body of architects and designers as the most popular algorithmic modeling tool due to its ease of use, user-friendly interface, flexibility, scalability in addition to its ability to be functionally extended by its plugin ecosystem to interact with both other software and other hardware using its software development kit (SDK) e.g. C# and Python RhinoCommon and grasshopper sdk.

### 1.2. Scikit-learn.

Scikit-learn is a popular open-source machine learning module (Pedregosa et al. 2011) developed and distributed under simplified BSD license, coded in Python and written to be simple, efficient and handy to non-expert users, as well as , it brings high level machine learning algorithms i.e. supervised and unsupervised learning to be usable in different contexts. There are many other Python machine learning modules but in this project, scikit-learn has been chosen for the following reasons (Buitinck et al. 2013): Its ease of use, robustness i.e. (code has followed

specific quality guidelines such as consistency and unit-testing), documentation and examples covering all features, efficiency (time and resources), table (2) summarizes a comparison between machine learning modules .However, this library and its classes and functions are designed to be integrated with scientific and numerical modules in python such as NumPy (Walt, Colbert, and Varoquaux 2011), and SciPy (Oliphant 2007) because datasets are represented as either Numpy arrays of Scipy sparse matrices (Buitinck et al. 2013).

Table 2. Comparison between Python-based ML modules (classes time efficiency andlicense) (Pedregosa et al. 2011).

| Algorithm | scikit-learn | mlpy | pybrain | pymvpa | mdp | shogun |
|---|---|---|---|---|---|---|
| Support vector classification | 5.2 | 9.47 | 17.5 | 11.52 | 40.48 | 5.63 |
| Lasso (LARS) | 1.17 | 105.3 | - | 37.35 | - | - |
| Elastic Net | 0.52 | 73.7 | - | 1.44 | - | - |
| k-Nearest Neighbors | 0.57 | 1.41 | - | 0.56 | 0.58 | 1.36 |
| PCA (9 components) | 0.18 | - | - | 8.93 | 0.47 | 0.33 |
| k-Means (9 clusters) | 1.34 | 0.79 | * | - | 35.75 | 0.68 |
| License | BSD | GPL | BSD | BSD | BSD | GPL |

### 1.3. Python for science and computation.

Python is one of the most popular programming languages among scientific computing community for its steadily expanding development ecosystem besides having a full range of sophisticated features such as first-class functions, garbage collection and expansion handling in addition to being an object oriented scripting language which can run on many platforms. Also it can be managed to make the best of the hardware speed available. Furthermore it is easy to use, subject to multidisciplinary large body of user community, above and beyond, it can be embedded into existing applications and interact with a variety of software. These reasons made this language rapidly gains interest by scientific computation users.

This research aims to make use of Python scikit-learn module and Rhino/Grasshopper for the sake of bringing sophisticated machine learning libraries to be manageable to architects and designers who are not specialized in machine learning but at the same time concerned with using its approaches in their work by introducing a plugin for Rhino\Grasshopper called Ant developed using C# and Python.

## 2. Plugin structure.

The structure of the plugin has followed a particular method consisted of three main segments, these are: 1) Datasets such as data instances, targets and training data, 2) Grasshopper interaction such as components, numerical inputs etc. . . ,and 3) Python modules such as Numpy, SciPy and scikit-learn as shown in figure (2).

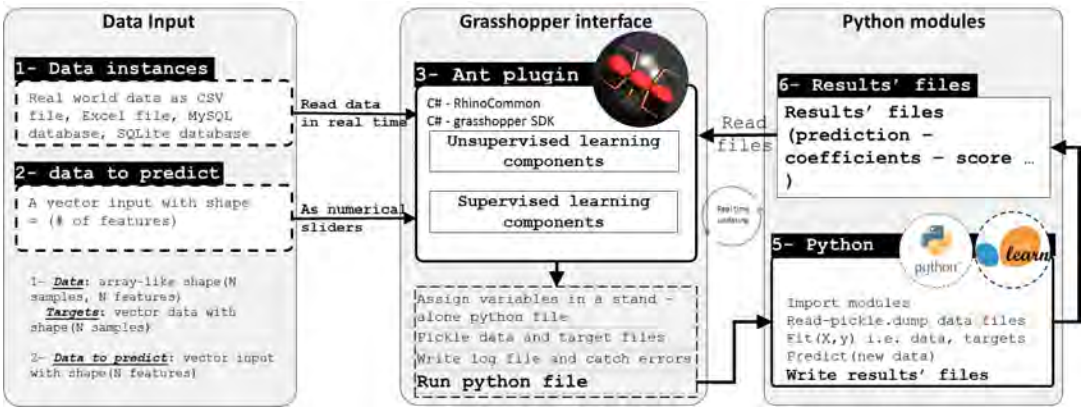


Figure 2. Structure of Antplugin.

### 2.1. Input data

The data input is categorized into three main portions:

- *Data instances* i.e. the real-world data which are used for the fitting process, it comes in form of 2d array with shape *(#_samples, #_features)*. The term "features" refers to the explanatory variables which are used for training, it should have a label and be of numerical data type either continuous, discrete or binary.

- *Targets* i.e. the response variables for each data sample, it comes in a form of vector $x_i \in \mathbb{R}^+$, $i = 1, \ldots n : n = \#\,of\ samples$ , that is each sample refers to one target as follows:

$$Input\ data = \overset{\begin{matrix} Features\ (X) & & & Targets\ (Y) \end{matrix}}{\begin{bmatrix} i_0\ f_0 & i_0\ f_1 & \ldots & i_0\ f_n \\ i_1\ f_0 & i_1\ f_1 & \ldots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ i_n\ f_0 & \ldots & \ldots & i_n\ f_n \end{bmatrix}} \Rightarrow \begin{bmatrix} t_0 \\ t_1 \\ \vdots \\ t_n \end{bmatrix}$$

Where $i_n$ refers to the input number, $f_n$ refers to the feature number and $t_n$ refers to the target number.

The present plugin has been developed so that it is able to read data of different types such as comma separated values (CSV) file, Microsoft excel files (.xls, .xlsx) files or structured query language e.g. MySQL data file.

- *Data to predict:* These data are the unknown data numeric variables are predicted for instance, given unlabeled observations X, returns the predicted labels y. It comes in a form of 1D array of shape (#_features, 1).

The grasshopper data-set components are shown in figure (3-a). The data input has been dsigned to follow an exact structure as shown in figure (3-b)

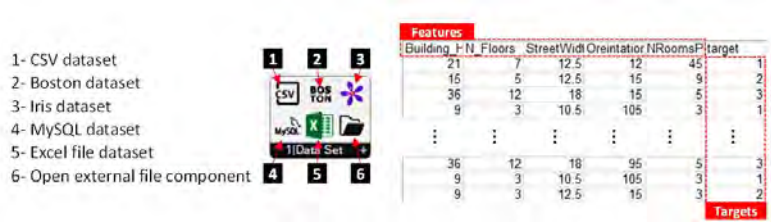


Figure 3. a. Data sets' components b. The structure of input data

### 2.2. Grasshopper interaction

The plugin interface has been developed using C# grasshopper sdk. Machine learning components has been categorized into sets based on their functions e.g. (Linear Models, Support Vector Machines, Linear and quadratic discriminant analysis, Stochastic Gradient Descent, Nearest Neighbors...etc.) and each category consists of a group of components as shown in figure (5).

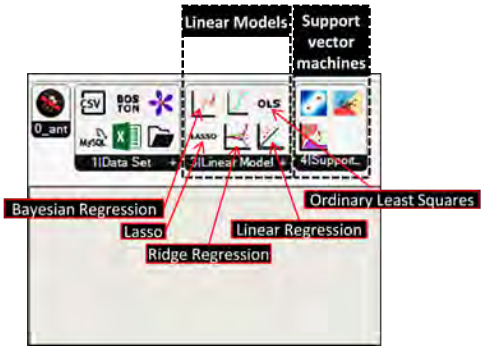


Figure 4. Machine learning categories based on their functions.

On the other hand, the development of each ML component has followed certain steps following the same steps as scikit-learn module for example, Support Vector Classification (SVC) is shown in figure (5) and the main inputs and outputs are demonstrated in table (3). A pseudo code of the fitting algorithm is illustrated in listing (1).

Table 3. The required input and the output of the support vector classifier.

| **Input** | **Description** |
|---|---|
| _Data | Data (X) instances derived from any data set components as shown in figure 3 |
| _Targets | Targets (y), derived from the dataset components [array] |
| _Working Folder | [String] |
| _Predict | The data which is required to be fitted. [array of numerical variables] |
| _Fit? | Boolean toggle to start fitting data. |
| Output | Description |
| Data Log | Log console to report errors and exceptions. |
| Prediction Result | The resulted value [numerical value] |

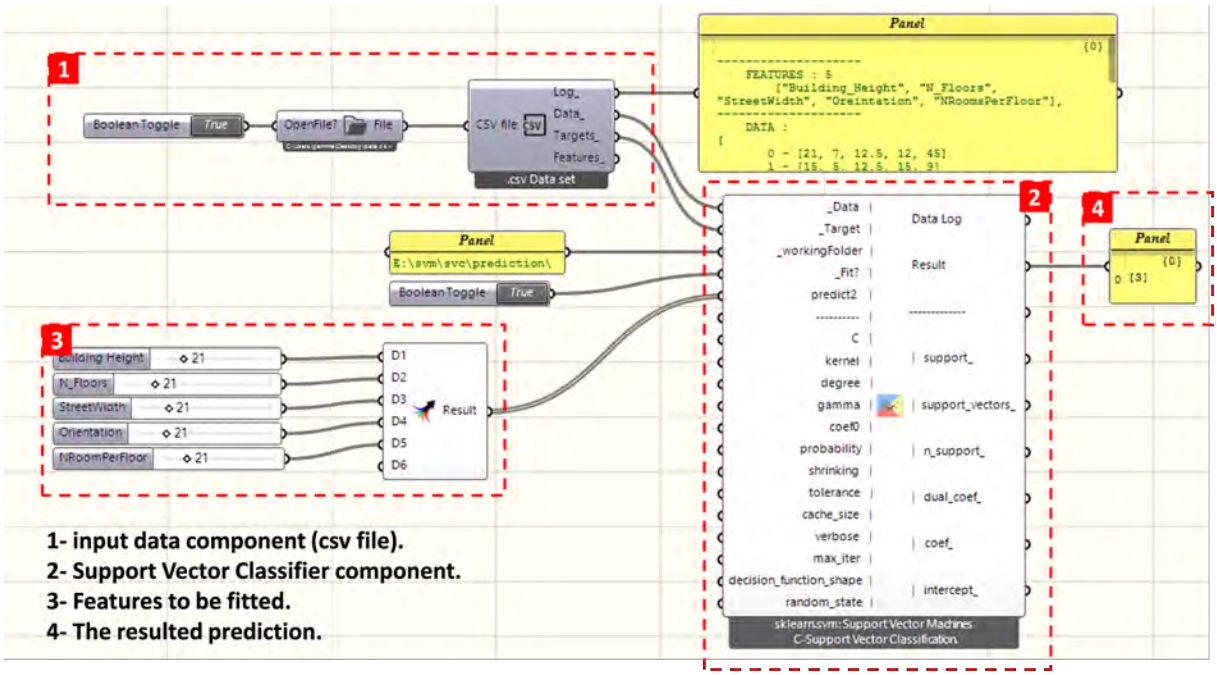


Figure 5. An example of classifier component (support vectorclassifier).

If ***fit*** is true:

Read data as ***d***

Read targets as ***t***

Read working directory as ***dir***

Read prediction data as p

Read data from stand-alone dummy python file as ***f***.

Write ***d, t, dir, p*** to ***f***.

Save ***f*** at working ***dir*** as ***nf.py***

Run ***nf.py*** (***nf.py*** runs and saves results as ***Res***)

While (***nf.py*** hasn't finished):

Wait

Read ***Res*** as output.

If any changes happen to **d or t or p**:

Repeat.

**Listing 1. A pseudo code of the algorithm used in the classifier component.**

### 2.3. Complexity and efficiency

The complexity of this tool is mainly derived from the complexity of the similar functions of scikit learn. For example: the complexity of the ordinary least square is: For X: X is a matrix of size (n, p) $\rightarrow$ $O(np^2)$ assuming that n$\geq$p (Pedregosa et al. 2011), while that of the support vector machines, the complexity scales between $O(n_{features} * n^2_{samples})$and $(n_{features} * n^3_{samples})$. A detailed explanation and best practices are demonstrated by (Pedregosa et al. 2011).

## 3. Future developments

The current state of the ANT plugin could enable regular users to perform machine-learning estimations, and predictions. It also gives users the ability to update data instances in real-time in addition to update calculations automatically whenever any input changes. It also could be efficiently integrated with other plugins such as Galapagos, Ladybug and Honeybee, ButterFly, Octopus... as it depends on numerical variables as input and output. However, it is still under progress. Thus, Future development includes involving a wide range of supervised and unsupervised learning, providing well-structured documentation and including plotting facility based on matplotlib module(Hunter 2007). Furthermore, it is intended to add an image processing module components to be integrated with machine learning. This could be attained by adjoining the scikit-image module in addition to increasing efficiency by including parallel computing module i.e. PyOpenCL.

## 4. Conclusion

This paper presents a method for benefiting from the potentials of Machine learning and Python scikit-learn to be handy to architects and designers who aren't specialists at Machine Learning and coding. Machine learning has become an active field of interest by a large portion of designers due to its potentials as well as the advancement in computer resources, efficiency and algorithms. Selecting scikit-learn for this purpose is justified due to its consistency, ease of use, efficiency and robust documentation. On the other hand, Rhino\Grasshopper has been selected as it is considered one of the most popular tools used by architects and designers, it also provides a wide range of flexibility such as the ability to be integrated with other software and hardware due to its robust sdk. The combination of these tools has led to developing a plugin called ANT which could be thought of as an open-source free platform which brings scikit-learn classes and functions in a form of components inside grasshopper. These components could be integrated with any other plugins and tools such as but not limited to: LadyBug and HoneyBee (a parametric environmental plugin for grasshopper) by (Roudsari, Pak, and Smith 2013), ButterFly (OpenFoam-based CFD plugin for grasshopper), Galabagos (Genetic algorithm plugin for grasshopper) etc... This plugin development is still in progress. Future works include adding image processing module scikit-image and parallel computing module PyOpenCL in addition to enhancing efficiency of the current components.

## 5. References

1.Abu-Mostafa, Yaser S, Malik Magdon-Ismail, and Hsuan-Tien Lin. 2012. Learning from data. Vol. 4: AML-Book New York, NY, USA:.

2.Arenas, Ubaldo, and José Manuel Falcón. 2013. "ALOPS Constructive Systems–Towards the Design and Fabrication of Unsupervised Learning Construction Systems."

3.Banihashemi, Saeed, Grace Ding, and Jack Wang. 2017. ”Developing a Hybrid Model of Prediction and Classification Algorithms for Building Energy Consumption.” Energy Procedia no. 110:371-376. doi: https://doi.org/10.1016/j.egypro.2017.03.155.
4.Bernhard, Mathias. 2013. Frequency Analysis of Wood Textures: Encoding of the grain pattern's orientation distribution for classification, comparison and search queries. Paper read at eCAADe 2013: Computation and Performance–Proceedings of the 31st International Conference on Education and research in Computer Aided Architectural Design in Europe, Delft, The Netherlands, September 18-20, 2013.
5.Buitinck, Lars, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, and Jaques Grobler. 2013. ”API design for machine learning software: experiences from the scikit-learn project.” arXiv preprint arXiv:1309.0238.
6.Carlos Sandoval Olascoaga, Wenfei Xu, Hector Flores. 2016. Crowd-Sourced Neighborhoods - User-Contextualized Neighborhood Ranking. In Proceedings of the 34th eCAADe Conference. University of Oulu, Oulu, Finland.
7.Chen, Nai Chun, Jenny Xie, Phil Tinn, Luis Alonso, Takehiko Nagakura, and Kent Larson. 2017. Data Mining Tourism Patterns - Call Detail Records as Complementary Tools for Urban Decision Making. In CAADRIA. Liverpool University, Suzhou, China.
8.Chi-Li Cheng, June-Hao Hou. 2016. Biomimetic Robotic Construction Process - An approach for adapting mass irregular-shaped natural materials. In Herneoja, Aulikki; Toni Österlund and Piia Markkanen (eds.), Complexity & Simplicity - Proceedings of the 34th eCAADe Conference - Volume 1, University of Oulu, Oulu, Finland, 22-26 August 2016, pp. 133-142.
9.Chou, Jui-Sheng, and Ngoc-Tri Ngo. 2016. ”Time series analytics using sliding window metaheuristic optimization-based machine learning system for identifying building energy consumption patterns.” Applied Energy no. 177:751-770. doi: https://doi.org/10.1016/j.apenergy.2016.05.074.
10.Cui, Can, Teresa Wu, Mengqi Hu, Jeffery D. Weir, and Xiwang Li. 2016. ”Short-term building energy model recommendation system: A meta-learning approach.” Applied Energy no. 172:251-263. doi: https://doi.org/10.1016/j.apenergy.2016.03.112.
11.Cutellic, Pierre, and Fabien Lotte. 2013. Augmented Iterations: Integrating neural activity in evolutionary computation for design. Paper read at eCAADe 2013.
12.Davis, Daniel. 2016. ”Evaluating Buildings with Computation and Machine Learning.”
13.de Wilde, Pieter, Carlos Martinez-Ortiz, Darren Pearson, Ian Beynon, Martin Beck, and Nigel Barlow. 2013. ”Building simulation approaches for the training of automated data analysis tools in building energy management.” Advanced Engineering Informatics no. 27 (4):457-465. doi: https://doi.org/10.1016/j.aei.2013.05.001.
14.Deb, Chirag, Lee Siew Eang, Junjing Yang, and Mattheos Santamouris. 2016. ”Forecasting diurnal cooling energy load for institutional buildings using Artificial Neural Networks.” Energy and Buildings no. 121:284-297. doi: https://doi.org/10.1016/j.enbuild.2015.12.050.
15.Decker, Martina. 2015. Soft Robotics and Emergent Materials in Architecture. Paper read at Real Time—Proceedings of the 33rd eCAADe Conference.
16.Fernando, Ruwan, Robin Drogemuller, Flora Salim, and Jane Burry. 2010. Patterns, heuristics for architectural design support: making use of evolutionary modelling in design. Paper read at New Frontiers: Proceedings of the 15th International Conference on Computer-Aided Architectural Design Research in Asia.
17.Harrison, Paul. 2016. What Bricks Want: Machine Learning and Iterative Ruin. In [Proceedings of the 36th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA). Ann Arbor.
18.Hunter, John D. 2007. ”Matplotlib: A 2D graphics environment.” Computing In Science & Engineering no. 9 (3):90-95.
19.Khosrowpour, Ardalan, Rimas Gulbinas, and John E. Taylor. 2016. ”Occupant workstation level energy-use prediction in commercial buildings: Developing and assessing a new method to enable targeted energy efficiency programs.” Energy and Buildings no. 127:1133-1145. doi: https://doi.org/10.1016/j.enbuild.2016.05.071.
20.Kontokosta, Constantine E., and Christopher Tull. 2017. ”A data-driven predictive model of city-scale energy use in buildings.” Applied Energy no. 197:303-317. doi: https://doi.org/10.1016/j.apenergy.2017.04.005.
21.Laskari, Anna. 2014. Multidimensional Comparative Analysis for the Classification of Residual Urban Voids'.

Paper read at Proceedings of the 32nd eCAADe Conference.
22.Mahalingam, GANAPATHY. 2005. "A Computational Model of a Sensor Network for the Optimization and Control of Acoustical Performance Criteria in Spatial Enclosures." Proceedings of CAADRIA 2005:475-483.
23.Merrick, Kathryn, Mary Lou Maher, and Rob Saunders. 2008. "Achieving adaptable behaviour in intelligent rooms using curious supervised learning agents." Proc. CAADRiA 2008 Beyond Computer Aided Design:185-192.
24.Oh, Jean, Jie-Eun Hwang, Stephen F . Smith, and Kimberle Koile. 2006. "Learning from Main Streets - A machine learning approach identifying neighborhood commercial districts." Innovations in Design & Decision Support Systems in Architecture and Urban Planning:325-340
25.Oliphant, Travis E. 2007. "Python for scientific computing." Computing in Science & Engineering no. 9 (3).
26.Paterson, G, SM Hong, D Mumovic, and J Kimpian. 2013. Real-time Environmental Feedback at the Early Design Stages. Paper read at eCAADe 2013: Computation and Performance–Proceedings of the 31st International Conference on Education and research in Computer Aided Architectural Design in Europe, Delft, The Netherlands, September 18-20, 2013.
27.Paudel, Subodh, Mohamed Elmitri, Stéphane Couturier, Phuong H. Nguyen, René Kamphuis, Bruno Lacarrière, and Olivier Le Corre. 2017. "A relevant data selection method for energy consumption prediction of low energy building based on support vector machine." Energy and Buildings no. 138:240-256. doi: https://doi.org/10.1016/j.enbuild.2016.11.009.
28.Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, and Vincent Dubourg. 2011. "Scikit-learn: Machine learning in Python." Journal of Machine Learning Research no. 12 (Oct):2825-2830.
29.Pinochet, Diego. 2016. Making - Gestures: Continuous design through real time Human Machine interaction. In Proceedings of the 21st International Conference on Computer-Aided Architectural Design Research in Asia (CAADRIA 2016). Melbourne.
30.Roudsari, Mostapha Sadeghipour, Michelle Pak, and Adrian Smith. 2013. Ladybug: a parametric environmental plugin for grasshopper to help designers create an environmentally-conscious design. Paper read at Proceedings of the 13th International IBPSA Conference Held in Lyon, France Aug.
31.Silvestre, Joaquim, Yasushi Ikeda, and François Guéna. 2016. Artificial Imagination of Architecture with Deep Convolutional Neural Network. In Proceedings of the 21st International Conference on Computer-Aided Architectural Design Research in Asia (CAADRIA 2016). Melbourne.
32.Sokmenoglu, Ahu, Gulen Cagdas, and Sevil Sarıyıldız. 2010. Exploring the Patterns and Relationships of Urban Attributes by Data Mining. Paper read at Proceedings of the 28th eCAADe Conference, Zurich, Switzerland.
33.Standfest, Matthias. 2014. "Unsupervised Symmetric Polygon Mesh Mapping-The Dualism of Mesh Representation and Its Implementation for Many Layered Self-Organizing Map Architectures."
34.Stasiuk, D, MR Thomsen, and EM Thompson. 2014. "Learning to be a vault—implementing learning strategies for design exploration in inter-scalar systems." Newcastle upon Tyne, England:381-390.
35.Stouffs, R, P Janssen, S Roudavski, and B Tunçer. 2013. FEATURE RECOGNITION AND CLUSTERING FOR URBAN MODELLING. Paper read at Conference on Computer-Aided Architectural Design Research in Asia (CAADRIA 2013).
36.Tamke, Martin, Mateusz Zwierzycki, Henrik Leander Evers, Sebastian Ochmann, Richard Vock, and Raoul Wessel. 2016. "Tracking Changes in Buildings over Time-Fully Automated Reconstruction and Difference Detection of 3d Scan and BIM files."
37.Vardouli, Theodora. 2013. Performed by and Performative for Rethinking computational models for user participation in design. Paper read at eCAADe 2013: Computation and Performance–Proceedings of the 31st International Conference on Education and research in Computer Aided Architectural Design in Europe, Delft, The Netherlands, September 18-20, 2013.
38.Walt, Stéfan van der, S Chris Colbert, and Gael Varoquaux. 2011. "The NumPy array: a structure for efficient numerical computation." Computing in Science & Engineering no. 13 (2):22-30.
39.Wassermann, Klaus. 2010. "SOMcity: Networks, Probability, the City, and its Context." Proceedings of

eCAADe 2010:197-205.
40.Wilkinson, Samuel, Sean Hanna, Lars Hesselgren, and Volker Mueller. 2013. Inductive aerodynamics. Paper read at eCAADe 2013: Computation and Performance–Proceedings of the 31st International Conference on Education and research in Computer Aided Architectural Design in Europe, Delft, The Netherlands, September 18-20, 2013.
41.Yildiz, B., J. I. Bilbao, and A. B. Sproul. 2017. "A review and analysis of regression and machine learning models on commercial building electricity load forecasting." Renewable and Sustainable Energy Reviews no. 73:1104-1122. doi: https://doi.org/10.1016/j.rser.2017.02.023.
42.Zarkadis, N., A. Ridi, and N. Morel. 2014. "A Multi-sensor Office-building Database for Experimental Validation and Advanced Control Algorithm Development." Procedia Computer Science no. 32:1003-1009. doi: https://doi.org/10.1016/j.procs.2014.05.525.