Machine learning for fluid property correlations: Classroom examples with MATLAB

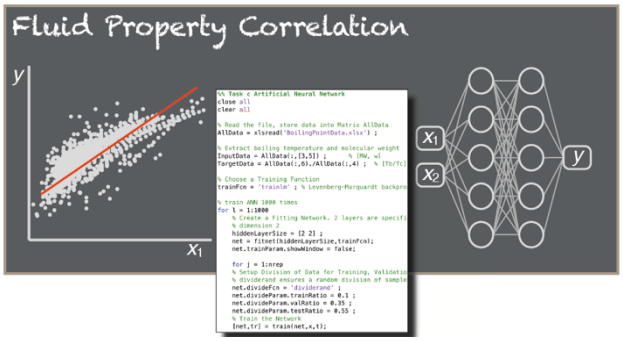
Lisa Joss and Erich A. Müller*
Department of Chemical Engineering, Imperial College London, U.K.

ABSTRACT

Recent advances in computer hardware and algorithms are spawning an explosive growth in the use of computer-based systems aimed at analyzing and ultimately correlating large amounts of experimental and synthetic data. As these machine learning tools become more widespread, it is becoming imperative that scientists and researchers become familiar with them, both in terms of understanding the tools and the current limitations of Artificial Intelligence, and more importantly being able to critically separate the hype from the real potential.

This manuscript presents a classroom exercise aimed at first-year science and engineering college students, where a task is set to produce an engineering-type correlation to predict the normal boiling point of organic compounds from an unabridged data set of > 6000 compounds. The exercise, which is fully documented in terms of the problem statement and the solution, guides the students to initially perform a linear correlation of the boiling point data with a plausible relevant variable (the molecular weight), and to further refine it using multivariate linear fitting employing a second descriptor (the acentric factor). Finally, the data is processed through an artificial neural network to eventually provide an engineering-quality correlation. The problem statements, data files for the development of the exercise and solutions are provided within a MATLAB environment, but are general in nature.

GRAPHICAL ABSTRACT



KEYWORDS
First-Year Undergraduate; Second-Year Undergraduate; Chemical Engineering; Computer-Based Learning; Mathematics/Symbolic Mathematics, Molecular Properties/Structure, Physical Properties

BACKGROUND

Research in artificial intelligence (AI), computational neural networks (CNN) and machine learning (ML) has been ongoing for decades but has only recently been in the spotlight with popular and widely-publicized applications such as voice[1],[2] and face recognition[3], personalized

online recommendation systems, language translation[4], and others largely driven by the recent exponential growth in available computational power. At the heart of ML applications is both the capacity and the requirement to handle large data sets[5] (hence the recent neology of "Big Data"). In parallel to their ostentatious appearance into aspects of daily life, recent advances both in hardware and software have allowed the unconventional application of these techniques to areas of the physical, chemical sciences[6,7], and engineering. The extraction of intermolecular force-fields from quantum mechanical calculations[8,9,10,11], the use of artificial intelligence in process system modelling[12] , the design and discovery of catalysts[13] and the prediction of properties of inorganic materials[14,15] are a but a few examples of the upcoming trends in engineering and physical science. It becomes imperative that college science and engineering students be exposed to these techniques, to the unique lingo, and more importantly be able to recognize, weeding out the media hype, how these new analytical tools can be successfully used both in labs and engineering environments.

ML researchers tend to discriminate the types of problems to be solved into two categories, *classification* problems and *regression* problems. Classification relates to the problem of identifying to which set of categories an observation composed of one or more *features* belongs. Binary categories (e.g. yes/no) are most often used in decision-making problems. Relevant examples come from the pharma and biomedical industries which have, in growing numbers, been employing ML to explore and screen candidate drug molecules in their ability to perform certain pharmacological function (see for example [16,17,18]). Regression problems are more commonplace in engineering and the interest here is to find analytical expressions that relate several variables amongst themselves. There are references that backdate decades ago on artificial intelligence programs for chemical education. Noticeable example is GEORGE[19], a terminal-based program which "solved" problems based on mass, volume and number of moles. Internally, GEORGE worked with dimensional analysis and by today's standard it would not be classified as ML, as the code itself followed (man-made) pre-developed algorithms. The actual promise of AI is the fact that the computer finds the underlying relationships on their own.

It is within this area that ML has the potential to have an impact in engineering. There is an inherent difficulty (and sometimes an impossibility) of performing experiments for fluid phase properties in extended ranges of temperature, pressure and composition. In lieu of a reliable theory, the design engineer must recourse to surrogate tools such as equations of state and/or correlations to be able to appropriately interpolate (or extrapolate) the available data. There are, generally, stiff restrictions in terms of the range of and accuracy of the correlations and questions on their ultimate validity. Notwithstanding these warning signs the use of correlations, be them empirical, or semi-empirical (as those based on theories, corresponding states and or group contribution methods) are the mainstay of Chemical Engineering Design[20]. A question stands to how AI can be used in this context. Can we replace the "human" element of correlating data empirically with a systematic procedure? Elements of such approaches are starting to appear in the literature[21,22,23] and will undoubtedly become ever more widespread.

A particular application of ML involves CNN, also called Artificial Neural Networks (ANN). ANN are computer models inspired by suggested models of how brain neurons interact with each other. To many ANN are still considered as a "black boxes", in spite that their inner workings were "revealed" to the readers of this journal over two decades ago[24]. A review of ANN with some application examples to Chemical Sciences and Engineering is found in ref 25. The current contribution explores the use of correlations and ML techniques in a prototypical chemical engineering / physical chemistry application: the correlation of

thermophysical data of fluids, with an example on the development, by the students, of a correlation for the normal boiling point of organic fluids.

We proceed in the current contribution to detail an exercise which allows students to gain some insight not only on the limiting factors of classical empirical correlations, but to do so in the context of the use of MATLAB® [26] as a programming tool and to showcase the capacity of ANN to solve an existing current design problem: the estimation of the normal boiling point of an (initially unknown) fluid. It is part of a larger 1st year undergraduate chemical engineering course in programming and computer competency. We share the belief that there is a future for the incorporation of data science into the mainstream activities of a chemical engineer and that there is reasonable space within most standard programming / numerical methods / computational methods courses to provide for the gradual introduction of these new concepts into the curriculum[27].

BOILING POINT CORRELATION

The normal boiling point is defined as the saturation (boiling) temperature of a liquid at 1 atmosphere of pressure. A related quantity, the standard boiling point, is defined by IUPAC as the saturation temperature of a fluid at a pressure of 1 bar. It is a simple enough property such that most students will be familiar with it. In order to exemplify the power of modern computers to manage large quantities of data, and to make the problem interesting we start out with a data set which is large enough to be impractical to sieve through manually. It corresponds to an unabridged table of boiling point information for organic compounds obtained from a standard database[28]. The table has not been sifted in any way and includes over 6000 points of experimental data, estimates obtained from group contribution methods and most likely a number of pseudodata of questionable origin. In this example, the students are asked to produce a correlation to calculate the normal boiling point of a fluid from the table provided. Initially, no brief is given in terms of the form and nature of the correlation and the students are guided towards the correlation process sequentially.

Normal linear regression

The most basic form of ML is the normal linear regression (or ordinary least squares) method. In first instance, we can guess that the boiling of point of a substance might depend on the molecular weight (MW) of the chemical compound: a thick, heavy cooking oil will boil at a higher temperature than water or an alcohol. A quick look in Wikipedia[29] confirms that somehow this assumption has some truth to it. A possible preliminary task is provided in Box 1: students are encouraged to manage a very large data set (which is difficult to sieve through, the file has over 6000 rows), plotting the bubble point data against the MW and attempting a simple linear regression fit.

Box 1. Visualizing the data

Retrieve the file **`BoilingPointData.xls`** (provided in the supplementary information). Note that this is an Excel file and you will have to import the data in a format MATLAB can read.

The data is tabulated into rows, one for each compound and columns that include a common name, molecular weight, critical temperature (in K), acentric factor and the normal boiling point (in K). Extract the data and place it into a matrix **`AllData`** to use further on.

Extract the boiling points from the data and place it in a column vector. Similarly, extract the molecular weight in a second vector. Plot the normal boiling point as a function of the molecular weight of the compounds in the data base. Fit a straight line through the data and find the R^2 value.

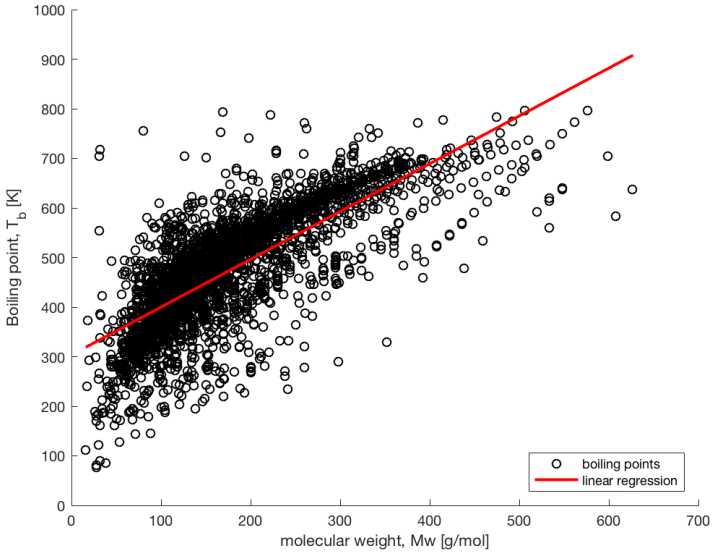


Fig 1. Plot of the boiling point data as a function of the molecular weight with an example of a linear regression to the data (red line).

The script for the solution is presented in the Supporting Information. Figure 1 shows the resulting plot of all the boiling points as a function of the molecular weight. It would seem that our choice of the molecular weight as our correlation variable is not particularly good at discriminating the data. While there is an evident trend, there is also a large scatter, obvious from the poor correlation coefficient (R^2 = 0.58). A more in-depth analysis would point out that the data set used has a few non-organic compounds with very high boiling point (e.g. sulfur, beryllium bromide, zirconium iodide, etc.), organic compounds that decompose before boiling (e.g. sucrose, cholesterol) and other data of questionable validity. The point of the exercise is not to discriminate the data, and we will keep the database unabridged. The average absolute deviation (AAD) of the predicted data,

$$AAD\ (\%) = \frac{1}{n}\sum_{i=1}^{n} \frac{abs(T_{b,i}^{\text{PREDICTED}} - T_{b,i}^{\text{EXPERIMENTAL}})}{T_{b,i}^{\text{EXPERIMENTAL}}}\ 100\% \qquad (1)$$

is 10.7 %, which would sound surprisingly good, given that state-of-the-art correlations, based on a group contribution method[30] report an accuracy of 4.3%. The resulting agreement is

misleading, as there is a smoothing effect on the AAD due to the very large data set used. Furthermore, the comparison is made on different data sets.

Multivariate regression

Other molecular features, apart from the MW, come into play when trying to correlate the boiling point. To search for these, the use of physical intuition might help out. As an example, consider methane, with a MW of 16 g/mol and water with a similar MW (18 g/mol). Both are roughly comparable in molecular size, but the normal boiling point of methane is 111.66 K while that of water is 373.15 K. Clearly, the specific interactions between molecules have a strong influence in the result. To improve our chance of obtaining a correlation, we must consider other properties to characterize our molecules. A plausible relevant property is Pitzer's acentric factor[31], ω. This is a dimensionless number that relates the deviation of the slope of the vapor pressure curve to that of the noble gases and helps quantify the relative strength of the intermolecular interactions. In the case of methane, the acentric factor is ω = 0.012 while for water it is 0.3449. At least in our example, it would seem that the acentric factor serves to recognize the difference in the intermolecular interactions between methane and water. We have now found two variables which serve to describe the peculiarities of each substance. In common ML lingo, these variables are called *features*. The process we will demonstrate has no limits in terms of how many features we use, and while any number could be chosen for our correlation, for simplicity, let us stop at this point and take these two: the molecular weight and the acentric factor. Side-tracking, if we were to use exclusively the acentric factor to correlate the boiling point, the results would be poorer that when we used the MW alone.

We will assume then that the normal boiling temperature T_b can then be expressed as a linear combination of the two features (this is called our *hypothesis* or *model*);

$$T_b = \theta_0 + \theta_1 x_1 + \theta_2 x_2 \tag{2}$$

where θ_i are the coefficients of the regression (the values we are looking for) and the independent variables are x_1 = MW and x_2 = ω.

An important step in any engineering correlation is to use as much theoretical insight as possible. We could, instead of finding the absolute value of the boiling point, focus on the ratio between the boiling point and the critical point. This has two advantages, on one hand, from a mathematical point of view, this guarantees that the variable we are looking for is in the order of one (actually, it will most likely be from 0.6 to 1). This *regularizes* the data, making the searches less prone to dealing with numerical issues of having large numbers summed or multiplied by small ones. On the other hand, there is a fundamental reason why this ratio is meaningful. In fluid-phase thermodynamics, it is common to invoke the corresponding states principle. This idea has a solid theoretical basis in Statistical Mechanics[32] and recognizes that there is a correspondence in the general physical behaviour of all pure fluids i.e. although all fluids boil, freeze, are gases etc. at distinct conditions, if an appropriate scale is used, their behaviour can be mapped into a single "universal" *conformal* trend. Here we use the critical temperature (T_c) as the energy scale to make the boiling temperature dimensionless. We can then use $y = T_b / T_c$ as our resulting target, hence eq. (2) takes the form

$$y = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 \tag{3}$$

where an assignment of $x_0 = 1$ is made to make the matrix algebra easier. We would only need a minimum of two data points on which to apply Equation (3) to uniquely determine the unknown values of θ_1 and θ_2. But out of the 6000+ data points we have, which should we choose? Could we not use them all? The answer is yes. Actually, the more data, the better. An equivalently valid answer is no. If we use all the data to "teach" our models, how will we test it? How will we know how good it is at predicting values (which is the ultimate objective of a correlation)? This introduces the concept of *training set*, a sub-set of the data used to obtain the required relationships. The resulting data, not included in the training set can then be used to *validate* the quality of the correlation.

The problem at hand is the solution of n equations of the type of Equation (3), each for a single data point in the training set, e.g. y is a column vector with n data points. For each one of the elements of y there is a corresponding column in the X matrix with the values of (1, MW, ω) for the given y. For example, a multivariate linear regression performed on a training set of 100 compounds extracted randomly from the data set would require the solution (the finding of the θ_i) in the following system:

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{100} \end{pmatrix} = \begin{pmatrix} 1 & MW_1 & \omega_1 \\ 1 & MW_2 & \omega_2 \\ \vdots & \vdots & \vdots \\ 1 & MW_{100} & \omega_{100} \end{pmatrix} \begin{pmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \end{pmatrix} \tag{4}$$

which in compact matrix form can be expressed as $y = X\theta$. The solution to the problem can be found using matrix algebra, invoking the ordinary least squares minimization, as[33]

$$\theta = (X^T X)^{-1} X^T y \tag{5}$$

which is ideally suited for solving with Matlab. Here X^T corresponds to the transpose of the matrix X while A^{-1} represents the inverse of matrix A. In Box 2 we present a plausible statement for a second task: the students are guided to perform a least squares analysis of a subset of the data, correlating it with both the MW and the acentric factor.

Box 2. Performing a multivariable correlation

While we could use all the data we have to produce a meaningful correlation, let us assume that not all the data was available. We will select a *training set*, or set of data that will be used to inform our ML algorithm. Extract from `AllData` 100 random data compounds. Using these selected compounds, build a 100×3 matrix, which we call X where each row corresponds to a training example (i.e. the data for a given compound). For each one of these rows the first column is the number 1 (corresponding to x_0) and the next two columns correspond to the values of x_1 and x_2 (molecular weight and acentric factor).
Create another column matrix y with all the corresponding expected results, i.e. each of the elements of the vector is the expected reduced boiling point (T_b / T_c) of each training example. The resulting y vector is a 100×1 vector.

The "solution" of this problem can be found directly as

$$\theta = (X^T X)^{-1} X^T y$$

Calculate the values of the coefficients θ_0 , θ_1 and θ_2 and evaluate the quality of the correlation.

A code for solving the problem is presented in the supplementary material. An (optional) 3-D plotting of the data and the solution is given in figure 2a. The correlation coefficient between the predicted and raw data has improved noticeably from the linear regression case as R^2 is now 0.85. The absolute deviation on the training set and on the entire data set will change every time that the code is run due to the random selection of the training set. The average numbers over 1000 repetitions are 2.57% and 2.68%, for the training set and entire data set, respectively. An interesting observation is that increasing the size of the training set does little to improve the overall correlation, indicating that we have reached the limit of what can be done with a linear fit of the data. Figure 2a shows how there is still a significant number of outliers, but the discrimination in terms of acentric factor and molecular weight has improved the overall correlation.

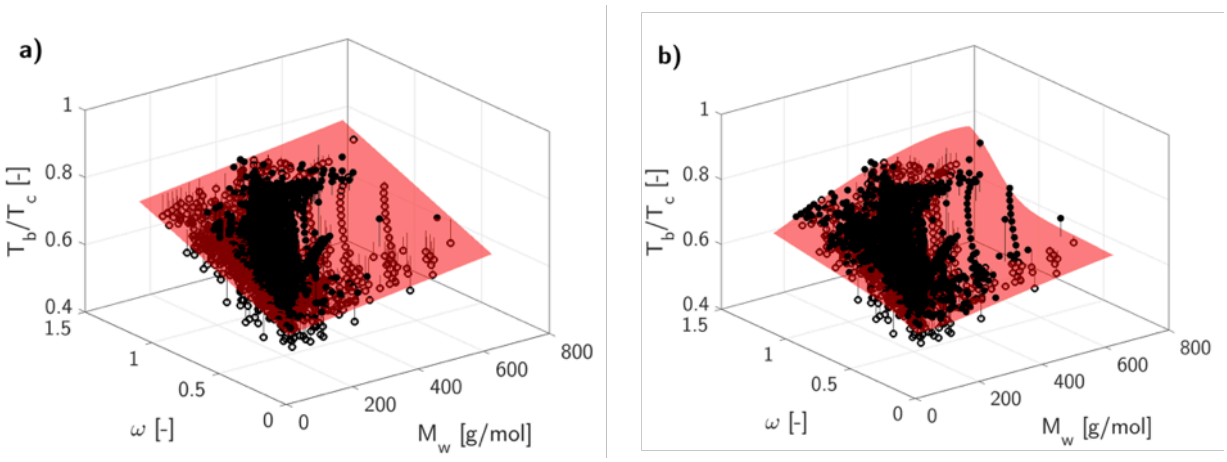


Fig 2. Reduced boiling point as a function of the molecular weight (MW) and the acentric factor (ω). Red surface is the result of a multivariate linear correlation (a) and an ANN (b). Symbols are the original data, shown as empty and filled circles for points lying below and above the surface, respectively.

Artificial Neural Networks

The above example would be qualified by some purists as a statistical regression rather than a "true" ML model. We had to provide the computer with a guess of the correlation form, Eq. (1), which is not always known or obvious and the resulting outcome is strongly dependent on the choice of this model.

Artificial neural networks (ANN) are computer systems based on a collection of interconnected computing elements (called artificial neurons in an analogy to biological systems) that have the ability to "learn" (progressively improve performance) based on the analysis of examples. Neurons are placed in layers between the input and output nodes which process the data. The

receiving neurons process signals and will relay them to another applying a weight that varies as the learning proceeds (See Figure 3).

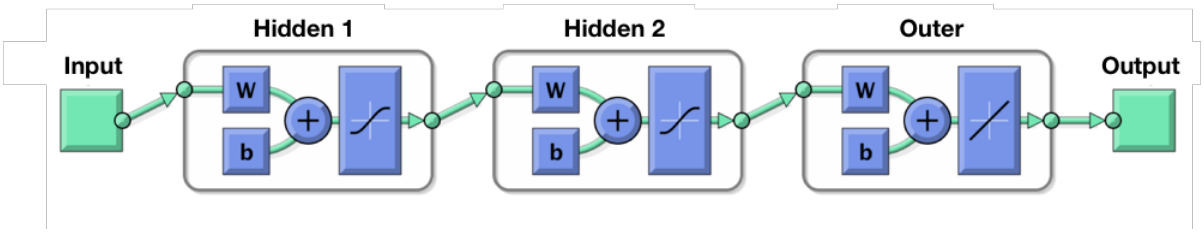


Fig. 3 Schematic of the ANN used to solve the boiling point correlation problem. The ANN is composed of two hidden layers with a tan-sigmoid transfer function and an outer layer with a linear transfer function (grey boxes). The weights (W) and biases (b) are optimized using the Levenberg-Marquardt algorithm. Green boxes represent the algorithm input (left) and output (right).

ANN are particularly well suited for processing very large data sets where there is very little understanding of the correlations between them. As such they have found common usage in spam filters, speech recognition, social networks, machine translation, medical diagnosis, drug discovery, etc. When training a ML model, the general practice is to divide the data set into three subsets. During the learning phase, the model is trained to match the input and outputs of the *training set*. this involves a number of iterations (also called epochs in the Matlab toolbox). The learning process stops once the error on the *validation set* is not reduced significantly anymore. Because both training and validation sets are used to choose the best ML model, the performance is assessed based on the *test set*, which has been reserved for this purpose.

To make a fair comparison, the students are asked to train an ANN with the same features as used in the linear regression. However, no implicit model is provided and no assumption is made on the relationships between the features and the target properties. In the original classroom exercise, students are guided in the use of ANN and the script for the code is provided. Box 3 provides a possible problem statement for this purpose. Most college-wide installations of MATLAB also provide for an "APP" that allows the same code to be performed. While the graphical user interface (GUI) provided within the app is very intuitive and easy to use, it gives the feeling of a "black box" exercise and defeats the learning outcome.

Box 3. Fit the data with a neural network

From the matrix **`AllData`** matrix retrieve two column vectors, one called **`InputData`** which will have two columns, the MW and the acentric factor of all the data in the original set and a second matrix called **`TargetData`** which contains the reduced boiling point data.

Use the following code to train a neural network and compare the performance of your network with that from the previous task.

```
% Solve an input-output fitting problem with a Neural Network

% Input and target data
x = InputData';
t = TargetData';

% Choose a Training Function
trainFcn = 'trainlm';  % Levenberg-Marquardt backpropagation

% Create a Fitting Network. 2 layers are specified here, each
% of dimension 2
hiddenLayerSize = [2 2];
net = fitnet(hiddenLayerSize,trainFcn);

% Setup division of data for Training, Validation, Testing
% dividerand ensures a random division of samples
net.divideFcn = 'dividerand';
net.divideParam.trainRatio = 10/100;
net.divideParam.valRatio = 35/100;
net.divideParam.testRatio = 55/100;

% Train the Network
[net,tr] = train(net,x,t);

% Estimate the output y based on the trained network (net)
% and the input x
y = net(x) ;
```

You may want to explore the effect of changing the number of layers. In the example above, only a 10% of the available data is used to train the network (some 600 data points) while the rest is used for validations and testing. Explore what happens when these ratios are changed.

The ANN model relies on a number of tuning parameters, of which the number of layers and the data division are of particular interest. Upon an increase in the number of layers, the model becomes more and more complex, and describes the data from the training set with a slightly increased accuracy. However, this comes at the cost of generalization, i.e., the model is so complex that there is the risk of over-fitting the data and performing badly on the prediction of the outcome of new data. This is typically referred to as the *bias-variance trade-off* of predictive models.

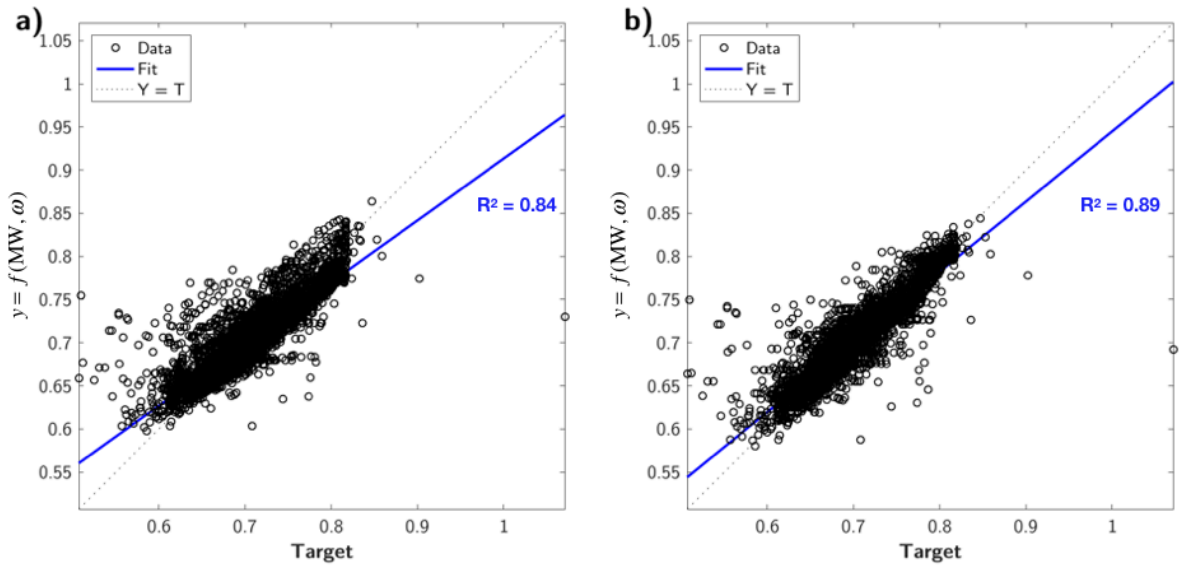


Fig. 4 Regression plot of the multivariate linear correlation (a) and an ANN correlation (b). All data points are plotted in terms of the output $y = f(MW, \omega)$ against the target T_b/T_c. The dashed line indicates the diagonal, and the blue solid line shows the best linear regression.

The more data is available in the training set, the more robust will the prediction be. However, the performance of the model will be more difficult to estimate as the number of samples in the validation and test sets are reduced. The general performance of the model on the entire data set can be evaluated from a regression plot, where the modelled outputs are plotted against the target outputs, as shown in figure 4. A perfect model (or regression) will yield points along the diagonal.

The result of the fit is similar to that of the multivariable regression, there is a coefficient of correlation of R^2 = 0.89 and an AAD of 2.18 %, which very marginally depends on the particular choice of training set, number of layers or training function. The only slight improvement as compared to the multivariate linear regression might seem somewhat disappointing, given the promise of ANN for fitting data, and its non-linear nature evidenced in figure 2b. But this should not be taken as a failure but rather as a recognition that the data itself may not be correlated any better with the limited choice of features chosen. Discrimination in terms of chemical families, organic/inorganics, etc. and the removal of obvious outliers and suspicious data will improve the fit and is essentially what is done with engineering correlations.

CLASS EXPERIENCE & CONCLUSIONS

MATLAB has an excellent Graphical User Interface (GUI) which allows students and researchers to set up an ANN and explore these ideas further. MATLAB is particularly suited for machine learning applications and has several built-in features and on-line teaching tools for this purpose. Given that the target audience of these exercises are first/second year chemical engineering students, the onus is not placed on the intricacies of the computer models, but on the ease with which data may be processed along with pointers towards advance learning

material. The example exposes students to the vocabulary of machine learning and provides them with a satisfactory resolution of a real-life non-trivial problem.

Student feedback on this particular problem has been exceptional. There is the recognition that this is an area that is becoming very topical and that the opportunities to incorporate these tools into design and engineering are flourishing.

We do recognize that a large proportion of the readers of this journal are composed of chemists, which as opposed to chemical engineers, would have different interpretations of the results and a different approach in terms of solving the principal question of the paper: obtaining a correlation for boiling points of a large ensemble of molecules. In fact, most chemists by training would recognize rapidly that while the MW is a relevant descriptor for the boiling point, it will be woefully insufficient to distinguish between families of molecules, where other descriptors might be more useful (such as the dipole moment, hydrogen bonding potential, etc.). The proposed method can be easily extended to cater for this collective and a rather simple modification of the input files and the script would be sufficient. However, we would like to point out that the objective of the paper is not to produce a physical correlation, for which there are many excellent examples in the scientific literature, but to illustrate how to harness the power of machine learning to process large data sets in a simple and effective way.

SUPPORTING INFORMATION

The Supporting Information is available on the ACS Publications website at DOI: 10.1021/acs.jchemed.XXXXXXX.

- Boiling point dataset (including critical temperature and acentric factor) (.xlsx)
- Sample Matlab scripts (zip file)
- Solutions to the three tasks and a list of advanced learning material (.docx)

AUTHOR INFORMATION

Corresponding Author

*E-mail: e.muller@imperial.ac.uk

ORCID

Lisa Joss: 0000-0002-1600-0203
Erich A. Müller: 0000-0002-1513-6686

REFERENCES

[1] Hinton, G. E.; Deng, L.; Yu, D.; Dahl, G. E.; Mohamed, A.; Jaitly, N.; Senior, A.; Vanhoucke, V.; Nguyen, P.; Sainath, T. N.; Kingsbury, B. Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. *IEEE Signal Process. Mag.* **2012**, *29*, 82−97.

[2] Deng, L.; Hinton, G.; Kingsbury, B. New types of deep neural network learning for speech recognition and related applications: An overview. *Int. Conf. Acous. Speech Signal Proc.* **2013**, 8599– 8603,

[3] Krizhevsky, A.; Sutskever, I.; Hinton, G. E. ImageNet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Sys.* **2012**, *25*, 1097−1105.

[4] Wu, Y.; et al. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. arXiv preprint arXiv:1609.08144. **2016**.

[5] T. M. Mitchell, *Machine Learning.* McGraw-Hill, New York 1997.
[6] Sumpter, B. G.; Getino, C.; Noid, D. W. Theory and applications of neural computing in chemical science. *Annu. Rev. Phys. Chem.* **1994**. *45,* 439-81.
[7] Schneider, F.; Guo, H. Machine Learning. *J. Phys. Chem. B*. **2018**, *122* (4), 1347–1347.
[8] Botu, V.; Batra, R.; Chapman, J.; Ramprasad, R. Machine Learning Force Fields: Construction, Validation, and Outlook. *J. Phys. Chem. C*, **2017**, *121*, 511–522.
[9] Behler, J. Perspective. Machine learning potentials for atomistic simulations. *J. Chem. Phys.* **2016,** *145*(17), 170901–170909.
[10] Handley, C. M.; Behler, J. Next generation interatomic potentials for condensed systems. *Eur. Phys. J. B*. **2014,** *87*, 152.
[11] Smith, J. S.; Isayev, O.; Roitberg A. E. ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost. *Chem. Sci.*, **2017,** *8*, 3192–3203.
[12] V. Venkatasubramanian, R.; Vaidyanathan; Y. Yamamoto. Process Fault Detection and Diagnosis Using Neural Networks–I. Steady-State Processes. *Comp. & Chem. Eng*. **1990,** *14*, 699–712.
[13] Goldsmith, B. R.; Esterhuizen, J.; Bartel, C. J.; Sutton, C.; Liu, J.-X. Machine Learning for Heterogeneous Catalyst Design and Discovery. *AIChE J.* **2018,** *64*, 2311–2323.
[14] Logan, W.; Agrawal, A.; Choudhary, A.; Wolverton, C. A General-Purpose Machine Learning Framework for Predicting Properties of Inorganic Materials. *NPJ Comput. Mater.* **2016,** *2,* 16028–7.
[15] Spellings, M; Glotzer, S. C. Machine learning for crystal identification and discovery, *AIChE J.* **2018,** *64* (6), 2198–2206.
[16] Pereira, J. C.; Caffarena, E. R.; Nogueira dos Santos, C. Boosting Docking-Based Virtual Screening with Deep Learning. *J. Chem. Inf. Model.* **2016,** *56* (12), 2495-2506.
[17] Subramanian, G.; Ramsundar, B.; Pande, V.; Denny R. A. Computational Modeling of β-Secretase 1 (BACE-1) Inhibitors Using Ligand Based Approaches. *J. Chem. Inf. Model.* **2016,** *56* (10), 1936–1949.
[18] Wang, J.; Ferguson, A. L. Nonlinear machine learning in simulations of soft and biological materials. *Mol. Simulation*. **2017,** *99* (2), 1090–1107.
[19] Cornelius, R.; Cabrol, D.; Cachet, C. Applying the Techniques of Artificial Intelligence to Chemistry Education, Chapter 11 of *Artificial Intelligence Applications in Chemistry*. ; Ed. T. H. Pierce, B. A. Hohne. ACS Symposium Series Volume 306, Washington D.C. 1986
[20] Poling, B. E.; Prausnitz, J. M; O'Connell J. P.; Reid, R. C. *Properties of Gases and Liquids.* 5th ed. Mc-Graw Hill, New York, 2000.
[21]Rostami, A.; Arabloo, M.; Ebadi, H. Genetic programming (GP) approach for prediction of supercritical CO_2 thermal conductivity. *Chem. Eng. Res. Des.* **2017,** *122,* 164–175.
[22] Mulero, A.; Cachadiña, I.; Valderrama, J. O. Artificial neural network for the correlation and prediction of surface tension of refrigerants. *Fluid Phase Equilib.* **2017,** *451*, 60–67.
[23] Meng, X.; Jia, M.; Wang, T. Neural network prediction of biodiesel kinematic viscosity at 313K. *Fuel*. **2014,** *121*, 133–140.
[24] Spining, M. T.; Darsey, J. A.; Sumpter, B. G.; Nold, D. W. Opening Up the Black Box of Artificial Neural Networks. *J. Chem. Educ.* **1994,** *71* (5), 406–411.
[25] Himmelblau, D. M. Accounts of Experiences in the Application of Artificial Neural Networks in Chemical Engineering. *Ind. Eng. Chem. Res.* **2008,** *47*(16), 5782–5796.
[26] MATLAB is a registered trademark of The Math Works, Inc. The version employed here is R2018a. Open source alternatives such as GNU Octave, Scilab and SageMath may also be used.

[27] Beck, D. A. C.; Carothers, J. M.; Subramanian, V. R.; Pfaendtner, J. Data science: Accelerating innovation and discovery in chemical engineering. *AIChE J*. **2016,** *62,* 1402–1416.
[28] Yaws, C. L. (2012; 2013; 2014). Yaws' Critical Property Data for Chemical Engineers and Chemists. http://app.knovel.com/hotlink/toc/id:kpYCPDCECD/yaws-critical-property/yaws-critical-property (accessed November 2018).
[29] Boiling point: From Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Boiling_point (accessed November 2018).
[30] Stein, S. E.; Brown, R. L. Estimation of Normal Boiling Points from Group Contributions. *J. Chem. Inf. Comp. Sci.* **1994,** *34,* 581-587.
[31] Pitzer, K. S. The Volumetric and Thermodynamic Properties of Fluids: I. Theoretical Basis and Virial Coefficients. *J. Am. Chem. Soc.* **1955,** *77*, 3427–3433.
[32] Reed, T.M.; Gubbins, K.E. *Applied Statistical Mechanics*, Mc-Graw Hill, New York, 1973.
[33] Freedman, D. A. *Statistical Models: Practice and Theory*, Cambridge University Press, Cambridge, 2009.