Customer Energy Profiling using Julia Lang with Big Data

Ebby Thomas [a], Hayden Klok [b], Yoni Nazarathy [b], Rahul Sharma [a],

[a] *School of Information Technology and Electrical Engineering, The University of Queensland*

[b] *School of Mathematics and Physics, The University of Queensland, Brisbane*

Abstract

In this paper, we present a procedure to aid network operators and electricity service utilities make use of the available big data in the realm of power systems. To illustrate the procedure, one of the data intensive applications in power networks - customer energy profiling - is selected. Existing protocols utilize a limited number of variables to forecast the electricity consumption, partly due to the complexities that arise when dealing with large number of variables. In contrast, our approach attempts to handle high dimensional data in the order of hundreds of variables. The variables are hinged on household specific characteristics that are available in the public domain and/or those could be fetched by the energy operators with minimal effort. Should the power system operators/retailers be capable of obtaining this data from their customers through methods such as questionnaires and surveys, customer energy modelling can be performed accurately together with the available real-time data. Treating high-dimensional data in the conventional paradigm may fail due to the large number of variables and increased complexity. We, therefore rely on a relatively new (at least in power systems) technical computing language - Julia computing. This paper walks the reader through the data analysis process and presents the resulting best-fit regression model that is able to accurately present the customer energy demand. We provide an instructional layout detailing all the computational working codes. The tool developed is unique in the sense that to the best of authors' knowledge, there is no package available in Julia that details all the data processing stages and model selection. Moreover, two packages within the tool- one that implements stepwise regression with categorical variable consideration and the second that implements Fisher's exact test- are unique in Julia. The virtue of our model is validated with another popular statistical model selection method, LASSO. Details of our methodology and the link to the codes on GitHub are provided so that others in the community may adopt and/or build on our approach.

Key words: Big Data, Regression, Linear Models....

1 Introduction

In recent years, a new trend has emerged in the applied science domains where data manipulation techniques have resulted in improvements of existing solutions. This trend of data manipulation is generally categorised as 'Big Data analysis'. Irrespective of the application, large datasets often share similar traits, such as their size, level of complexity and their scalability to much smaller, interpretable forms. Data processing is becoming centric to multiple domians from enterprise and marketing to space and science, Jagadish et al. (2014). Data has always been of particular interest in power engineering, and regression models have traditionally been used in load, energy and weather analysis, load prediction, day ahead demand, price forecasting and contingency analysis, Alfares & Nazeeruddin (2002), Chen et al. (1992). Following the recent introduction of renewable energy and in the pursuit of better power reliability, data driven methods are largely being used for various non-conventional purposes such as forecasting of wind and solar power, prediction of maintenance and upkeep of asset management, Endrenyi et al. (1998), Long et al. (2014). Moreover, the electric power industry is on the edge of a paradigm shift, both in terms of its technical and business aspects, Akhavan-Hejazi & Mohsenian-Rad (2018).

With the ever increasing amount of data measured worldwide, the word 'Big Data' is coined to represent 'unstructured' and 'high dimensional' data, Fan et al. (2014). Undeniably, many of the datasets used in various industries are massive by virtue of their data points, nevertheless, cannot be deemed as 'high dimensional' since the number of variables are often limited to a handful. In the modern day power systems, we come across highly unstructured data from sources such as

Email addresses: e.thomas@uq.edu.au (Ebby Thomas), h.klok@uq.edu.au (Hayden Klok), y.nazarathy@uq.edu.au (Yoni Nazarathy), rahul.sharma@uq.edu.au (Rahul Sharma).

measurement sensors, Advanced Metering Infrastructure (AMI), smart meters and electricity customer advocates, Kabalci (2016). Apart from online load data, big datasets may also be generated that may comprise of nameplate and maintenance details, socio-economic factors that may affect the customer energy usage profile, data from energy management and energy information system, Siano (2014) and many more which are often recorded offline. Given that the datasets of the near future could consist of hundreds of variables, there is an urgency to develop tools to accommodate this restoration of power systems. For instance, future power systems applications such as customer profiling and clustering, aggregator load management, energy planning and demand response would require a large amount of data to be managed and processed efficiently and systematically over reasonable time frames.

The main contributions of this paper are:

- Provide tools for energy profiling so that, by identifying individual energy usage traits, engineers/ retailers could initiate energy management strategies.
- Identify the key variables (from a large pool of variables) that affect customer energy consumption.

The paper is structured as follows. Section 2 gives a flavour of the nature of data used for modelling, as well as introduces Julia language. Section 3 details the data processing stages, followed by the modelling stage in Section 4. Detailed results and energy prediction is given in Section 5, before concluding with Section 6.

2 The Data and Software for Data Processing

2.1 The Data

The data as well as the codebook for this study is publicly available and is accessed from, *US, EIA, RECS* (2014). The dataset is the outcome of the Residential Energy Consumption Survey (RECS) in The United States over the year of 2009. The survey had 12,083 observants with 940 variables recorded for each observant.

2.1.1 Variable Types:

The dataset comprises of three types of variables:

- *Categorical:* Categorical variables take on values that have two or more categories or levels, however, there is no intrinsic ordering to the categories. For instance, the type of bulb used by an electricity customer is a categorical variable having a number of categories (incandascent, LED, fluorescent, halogen bulbs etc.), although, there is no agreed direct way of ordering these from the highest to lowest.
- *Ordinal:* Ordinal variables are similar to categorical variables, however in this case, there is a clear ordering of the variables, although the intervals between the levels may not be equal. For instance, suppose we define a variable describing the maximum education level that a customer possesses. We order it in three different classes such as high school, college graduate and university graduate. Though the spacing between these levels may not be equal, we are still able to order the levels from lowest to highest.
- *Numerical:* Numerical variables are similar to ordinal variables except for that the spacing between the levels are well defined. Mathematical operations can be performed on numerical variables to deduce meaningful insights. For instance, it is unilaterally agreed that the energy consumption is influenced by the household income of the customer. If the income of two households is \$50k and \$60k per annum respectively, this clearly differentiates the two households as well as conveys the information that the difference between their incomes is \$10k.

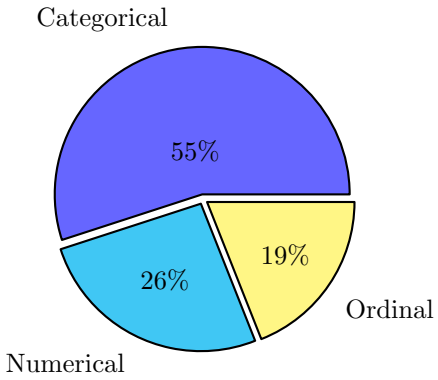

Fig. 1. Proportions of variables in the dataset (out of 588 variables).

Of the 940 variables, 352 had imputation flags. The remaining 588 variables consist of 323 categorical, 153 numerical and 112 ordinal with proportions as shown in Fig. 1.

2.1.2 The Response Variable:

The response variable to be predicted is the annual energy consumption of each individual household. The histogram of Fig. 2, shows the distribution of the resposne variable and indicates a 'log normal' distribution of energy consumption. As per classical statistics, the central limit theorem states that the "addition of random effects" yields a Normal/Gaussian distribution, whereas the less-well known versions establish that multiplication of positive random numbers result in an approximate log normal distribution. Note that multiplication plays a much more dominant role in natural processes than addition, for instance when "*amounts*", such as physical, biological, medical and chemical properties are modelled. A random variable 'X' is said to follow a log normal distribution if the random variable $Y = log(X)$ is normally distributed. The density function is,

$$f(x) = \frac{1}{x\sigma\sqrt{2\pi}} e^{\frac{-(lnx-\mu)^2}{2\sigma^2}}; x > 0, \quad (1)$$

where μ and σ are the mean and standard deviation of the distribution. For our dataset, $\mu = 11,288$ and $\sigma = 7,641$. Some of the properties of the log normal distribution and specific inferences for our dataset are given below:

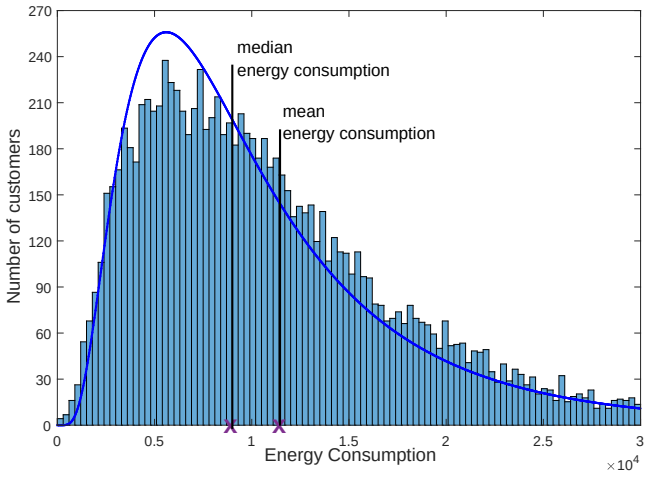

Fig. 2. Energy consumption characteristics.

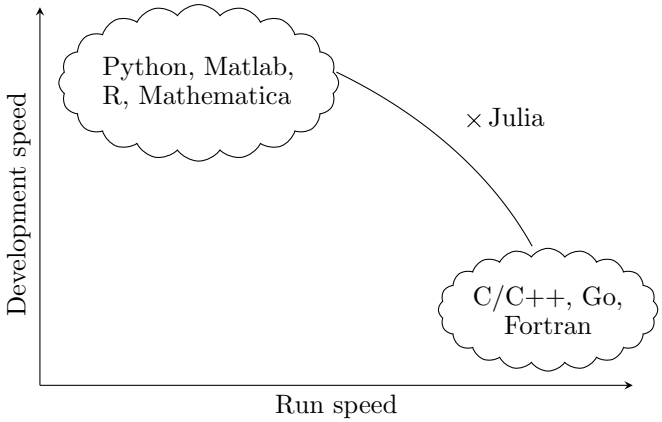

Fig. 3. Julia, Performance comparison.

- The lower limit for a log normal curve is definite, but the upper limit is theoretically unlimited (though in reality is limited by some practical constraints). In our case, we can be sure that the minimum energy that a customer could possibly use is theoretically zero, however, in a society, there are always customers who use large quantity of electric energy every year.
- The log normal distribution is positively skewed. This signifies there are a large number of low-usage/austere electricity customers in our case.
- Due to the fact that real-world quantities that follow log normal distributions are often skewed due to outliners, the median value is often taken as the estimate of the location parameter of the distribution. Hence in our case, the median value is taken as the representative of a typical energy customer, rather than the mean value (as marked in Fig. 2).

Many challenges are encountered during the data processing stages. The diversity in variable type and the large number of categorical variables as discussed earlier are undoubtedly the greatest hurdle. Secondly, the data is high dimensional with potentially hundreds of predictors influencing the response. Note that our aim to model the energy consumption may not necessarily relate to the objective of the 'EIA' for attempting this survey. Finally, sparsity, non-relevant and repeated data points and the fact that the data is in an unaggregated form - all increased to the complexity of working with the dataset. In the next sections, we explain how we managed to process the data and define the model, despite these challenges.

2.2 Software Tool for Data Processing, Julia Language

We relied on the relatively new technical computing language, Julia, to accomplish our research objective. Julia is a high level, high performance dynamic programming language with extensive numerical and scientific computational capability. An impressive list of mathematical and statistical libraries enables Julia to be used for most data processing applications. Julia is an open source platform, with many strong computational advantages including its speed, built-in parallel computing capability and external packages that cover many areas. One of its strongest attributes is that Julia not only has fast run times, but it is easy to learn and hence has fast development times, thereby reducing the overall programming time. Julia aims to extend the present frontier of computing languages, as shown in Fig. 3. We anticipate more members from the power engineering community will embrace this new tool over time.

2.2.1 Using Julia:

There are different ways to use Julia based on the user's preference:

- Juliabox: Juliabox is a server-side environment where Julia code can be run remotely in a web browser. It is free to sign up to Juliabox through an active Github, LinkedIn or Google account, no download is required.
- Julia REPL: The Julia REPL (Read-Eval-Print-Loop) can be downloaded from https://julialang.org/downloads/. Once installed, Julia can be run locally in the command-line mode.
- Atom Juno IDEs: Julia can be run in the Atom text editor by configuring teh Atom Juno IDE (Interactive Development Environment). This is the method that we used to develop the code.

3 Data Processing Stages

This section and the next describes our most significant research contribution. Our approach is outlined in the project flowchart, Fig. 4. The procedure involves many stages of data processing and modelling. The data processing stages are described in this section, while Section 4 the modelling stages. First data from the survey results are collected following an initial variable selection/screening. Various compatible variables are merged through aggregation and their values are normalised to limit the scale of the variables, thereby making the computation easier. The dataset observations are then randomised to ensure an impartial selection of model. Finally, missing data is imputed as per some standard practices.

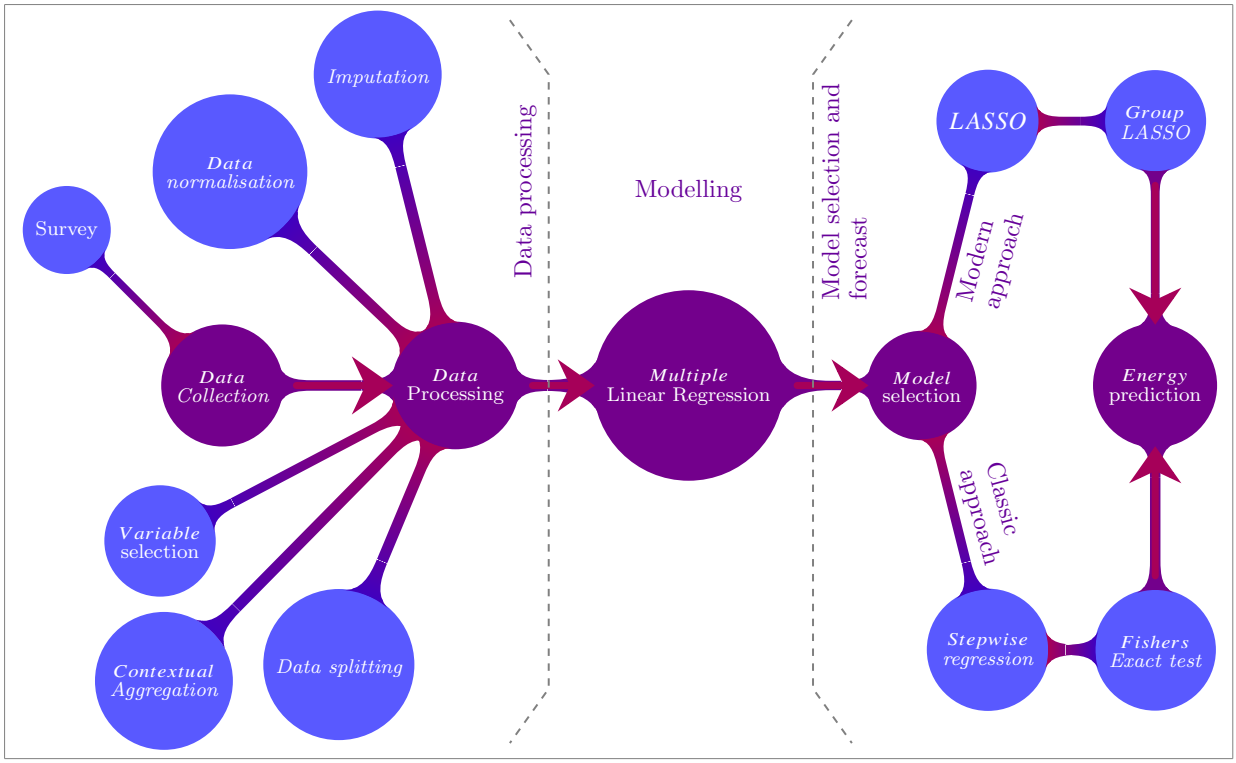

Fig. 4. Project Flowchart.

After data processing modelling is performed. Of the many modelling techniques available, we select Multiple Linear Regression. Model selection is performed using a backward elimination technique which is a common form of stepwise regression in statistics. The selected model is utilised to predict energy demand based on household specific characteristics. Although these are typical methods in data analysis problems, the magnitude, order and frequency of their usage widely vary based on application specific requirements. For some other applications, some of these stages could be skipped, while for others, this would need to be used as part of the procedure, depending on the nature of the dataset, the problem and the user's specific requirements.

3.1 Data Collection

Data collection process is expected to be extensive and often tedious in big data analytics. Some common sources of data are:
Log files: System generated data automatically recorded in designated files in the required format. An example is the data in web servers for access tracking.
Sensors: With the extensive use of control systems, the number of sensors that are added to any system are on the rise. They generate enormous data relating to power, temperature and vibration.
Survey results: Though survey results are often not very huge in terms of the number of observations, they could increase processing complexity due to their high dimensionality.

3.2 Subjective Variable Selection

Subjective variable selection is the first step in our data processing. In most cases, datasets received by a power engineer contain not just variables that are pertinent to the response, but also numerous other variables that are likely not relevant. In our case, in order to deal with the large amount of variables, first the variables corresponding to the imputation flags are removed from the model. The imputation process followed in the data processing is described in a later section. Next, we investigated all the remaining variables and assessed their possible cause and effect relationship to the response - the annual energy consumption. Thus, we eliminate nearly 66% of variables that we presumed to exhibit less significant relationship with the response. The exclusion of such a considerable number of variables is largely due to the survey also being aimed at other realms that power engineers would be least interested in.

As an example, one of the variables in the dataset is the daily distance travelled by the households. We claim that this variable do not improve the understanding of electricity consumption of the customer in any way and therefore, is eliminated from the variable set. Moreover, the origin of the dataset is in The United States. Information related to allied states and territories stands largely irrelevant in the Australian context, thus forming one of the many other reasons for the elimination of irrelevant variables. Again, we excluded some variables such as natural gas and gasoline usage characteristics

that are not directly related to our purpose. Majority of big datasets are subjected to a manual scrutiny as performed in here. This idea of utilising a sophisticated computational algorithm in conjunction with pragmatic manual approaches has been acknowledged extensively over the data analysis research community, Sjøvaag et al. (2012), Lewis et al. (2013).

3.3 Contextual Aggregation

Contextual aggregation of data is performed mainly to reduce the dimension of the dataset, thereby to decrease the complexity of data processing and to improve the interpretability of data. Aggregate data is obtained when groups of observations/measurements/variables are replaced with their summary statistics. Categorised as a data redundancy elimination method, contextual aggregation, as the name suggests, follows a logical aggregation of variables based on a case to case judgement, depending on the 'context' of the data collected as well as the demand of the application. For instance, in our dataset, the variables that indicate for the customer whether they cook food at home, the type of fuel they use for cooking, and if they use electricity, the frequency of their cooking - all these variables are merged to a single variable as per a pragmatic algorithm. Thus the 3 variables of diverse nature are combined to a single continuous variable, thereby not just decreasing the number of variables, but also improving the ease of analysis considerably. In our case, we used contextual aggregation in data processing for nearly 75 times ranging in various complexities.

3.4 Data Normalisation

Data normalisation is performed to transform the nature of the variables as well as to change the scale of data to fit it to the desired interval. Data discretisation, noise reduction and scalability is achieved through data normalisation which improves the data analysis efficiency, Bonsack & Skoryk (2017). Of the numerous normalisation methods such as min-max normalisation, Z-score normalisation and decimal scaling, here we proceed with the min-max normalisation, where the scale of normalised data is strained to fall between suitable limits as suggested by the dataset. Here, the minimum and maximum of a particular variable are identified (say L and U) and a normalised scale minimum and maximum are considered (say ℓ and u). Then any number, 'a' in the dataset is transformed into '$\hat{a}$' as below,

$$\hat{a} = \ell + \frac{(a - L)(u - \ell)}{U - L} \tag{2}$$

This puts $\hat{a}$ in the range $[L, U]$ when we set $[\ell, u]$ individually per variable. The normalisation procedures for data samples we followed are consistent with existing industry standards, Benini et al. (2014).

As an example, in the dataset, there are multiple categorical levels for the variable that characterises the frequency of cooking over a week. The customer answers 20 if no meal is cooked at all in the house, 10 if one meal is cooked every week, 5 if 3-4 meals are cooked every week, 4 if one meal is cooked every day, 2 if two meals are cooked every day. All these are merged to a single continuous variable after choosing a suitable scale, which is determined by the entire range of observation of the variable. Here, we use a linear scale starting from 0 to 15. The change of scale help not just to retrieve more information, but also to change the type of variable to numerical. We use data normalisation procedure extensively for over 30 times during the data processing. The scale of some of the variables before and after the normalisation process are given in Table.1, 'C' denotes a categorical variable.

Table 1
Actual scale of variables and their normalised scale.

Variable	Actual scale $[L, U]$	Normalised scale $[\ell, u]$
TVONWE1	0-5	0-12
TIMEON1	0-4	0-10
WASHLOAD	0-5	0-16
WINDOWS	0,20,30,40,50 (C)	0-30
MONEYPY	0-20 (C)	\$20k-\$130k

Performing aggregation with normalisation often results in the transformation of categorical variables into numerical variables, thereby making the insight and the interpretation from the model more powerful. An example by which we implement contextual aggregation together with data normalisation is shown below,

```
using DataFrames
if newDf[:COOKFOOD][i]==1 && newDf[:ELECCOOK][i]==1;
newDf[:NUMMEAL][i] = newDf[:NUMMEAL][i]
else newDf[:NUMMEAL][i]=0
newDf[:NUMMEAL] = df[:NUMMEAL]
for i=1:obs
if newDf[:NUMMEAL][i]==1
   newDf[:NUMMEAL][i]=18
   elseif newDf[:NUMMEAL][i]==2  ;  newDf[:NUMMEAL][i]=10
   elseif newDf[:NUMMEAL][i]==3  ;  newDf[:NUMMEAL][i]=7
   elseif newDf[:NUMMEAL][i]==4  ;  newDf[:NUMMEAL][i]=5
   elseif newDf[:NUMMEAL][i]==5  ;  newDf[:NUMMEAL][i]=3
   elseif newDf[:NUMMEAL][i]==6  ;  newDf[:NUMMEAL][i]=1
   end
end
```

(see file "**loadForecasting.jl**")

Possibilities of automating the data analysis until this point are seldom, as they are greatly application specific and lean on user preferences. At this stage, the column dimension of dataset is reduced to 69.

3.5 Randomisation of the Dataset

Randomisation is perfomed in data analysis to ensure that data collected among similar groups do not influence the validity of the model. Our case being a survey result, observations that are close could be the representative of households from the same geographical location. Here, randomisation is done to obtain a foolproof model by shuffling the observations of the dataset, Golbraikh et al. (2003). Now that the observations are randomised, train, validation and test sets are formed

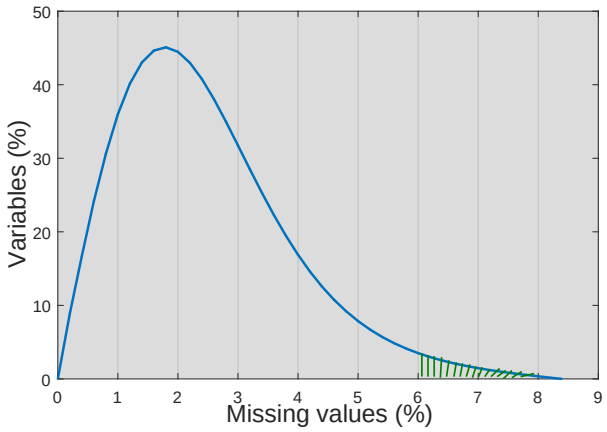

Fig. 5. Percentage of missing values.

for the model formulation, validation as well as testing respectively. While train set fits the best model for the system, the validation set is used to estimate the prediction error for model selection. The test set is used for the generalization error of the final chosen model. As the validation set is is used for the selection of final model, a separate test set is used to ensure that the error rate estimate on the final model is unbiased. The randomised dataset is split into train, validation and test sets in the ratio of 8 : 3 : 1. This is based on the common practice to choose anywhere between 50% and 80% of the total observations as the train set, Andrew Ng (2018). Randomisation is done in our dataset as follows,

```
trainProp = 0.3
trainRows = sample(1:obs, convert(Int,
floor(obs*trainProp)), replace = false)
testSet = newDf[ trainRows, :]
trainSet = newDf[ setdiff(1:obs, trainRows), :]
```

(see file "**loadForecasting.jl**")

3.6 Missing Data and Imputation

Missing data, quite common in big datasets, is a ubiquitous problem since statistical methods presume that every observation has complete information on all the variables, Allison (2001). Survey results may have missing data points when respondents refuse to answer due to feeling insecure in sharing the information, or even unsure and double minded about the question itself. Missing values could also be the result of superficial information after the respondent giving an answer that cannot be aligned to any of the pre-assigned levels of the variable, or simply due to an error whilst handling the data. Missing values could introduce chaos into the dataset, affecting the statistical parameters of each variable. There are many well established approaches to deal with this missing data. 'Complete case analysis' is a method in which data processing considers only the variables for which all observations are defined. However, this does not turn out to be the most appropriate in our case as our data has a relatively higher order of variables in comparison with the observations, therefore, chances of one or more observation to be missing in a variable is high.

Fig. 5 indicates Kernel Density Estimation (KDE) of percentage of missing values per variable. KDE is a technique to estimate the unknown probability distribution of a random variable based on the samples taken from the distribution. The probability density function is expressed as weights (kernels) and hence the name. Kernel estimators smooth out the contribution of each observed data point over a local neighbourhood of that data point.

Following this, we eliminate variables with more than 6% of their data points missing as indicated in the shaded region of Fig. 5. As a result, we eliminate six variables from our dataset. We now *impute* all the remaining missing observations. Imputation is the generic process referring to the replacement of the missing values based on clearly defined procedures, Royston et al. (2004). There are numerous, well documented application-specific imputation techniques such as mean value imputation, random imputation, KNN and K-means clustering, Schmitt et al. (2015).

```
# define colsToImpute (all variables that needs imputation)
for i in colsToImpute
   possibleValues = setdiff(newDf[i],0)
   proportions = [ count( x->(x==i), newDf[i])
   for i in possibleValues ]
   weightVect = Weights(proportions)
   for j in 1:varTotal
    if  newDf[i][j] == 0
    newDf[i][j] = sample(possibleValues,weightVect)
    end
   end
end
```

(see file "**loadForecasting.jl**")

Our imputation process, as shown above, first involves specifying a list of variables that requires imputation. Then, for each variable, a list of all its unique values is obtained though the function `setdiff`, along with the number of times (frequency) each value occurs (through `count`). These frequencies of occurrence are then used to calculate a weight vector for each of the identified unique values in the column though the dedicated `Weights` function. Finally, all null or missing values encountered in the column are replaced with a randomly selected value from the vector of observed values, based on the corresponding weight vector via the function `sample`. This process is repeated for all the variables that require imputed. This is a selective hybrid imputation process where the properties of randomness as well as weights are considered. This genre of imputation is generally classified as 'hot-deck' imputation.

3.7 Data Processing Summary

Following steps 3.1 to 3.6, we arrive at a final dataset with dimension $12,040 \times 69$. Starting from 940 variables, our dataset is passed through a number of dimensionality reduction stages, and this has been illustrated in Fig. 6. Dataset at each stage is indicated by D_x. Process 1 indicates the subjective variable selection, and Process 2, contextual aggregation. Process 3 and 4 shows the randomisation of the datasets and subsequent imputa-

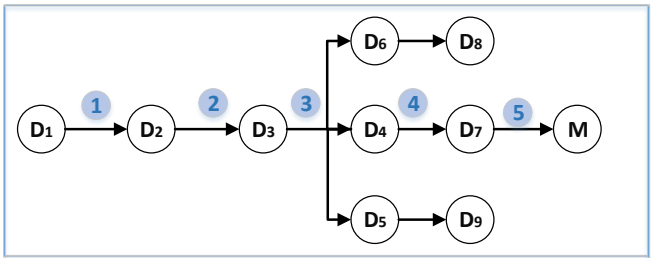

Fig. 6. Dataset dimensions at various data processing stages.

Table 2
Dataset dimensions at various data processing stages.

Item	Dimension	Description
D_1	12040 × 940	Raw dataset
D_2	12040 × 300	Post subjective variable selection
D_3	12040 × 75	Post contextual aggregation
(D_4, D_5, D_6)	(8459, 2719, 905) × 75	Randomised (train set, validation set, test set)
(D_7, D_8, D_9)	(8459, 2719, 905) × 69	Imputed (train set, validation set, test set)
M	As required	The model

tion respectively. Note that not all of the data processing stages described in the procedure incur a dimension reduction. Fine details of dimensions of the dataset at each stage of data processing is furnished in Table. 2.

Fig. 7 presents the share of the three different types of variables after the data processing. At this point, it is interesting to compare the pie chart with Fig. 1 that indicate the same prior to data processing. Note that the fraction of categorical variables reduced significantly, not because all the variables that belong to this class has been eliminated, but many of them have been converted to a numerical variable type, which in turn caused a sharp rise in the share of numerical variables.

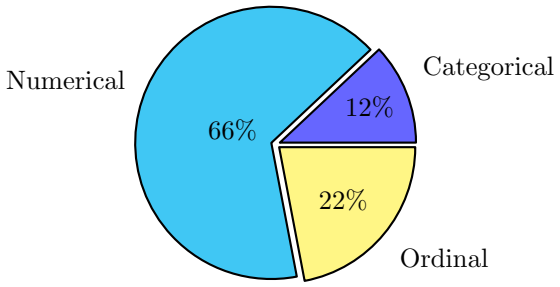

Fig. 7. Proportions of variables in the dataset after processing (out of 69 variables).

4 Models and Model Selection

After the data is passed on through a series of processing stages, we now have a clean dataset to work on. To deduce meaningful insights from the dataset, we need to appropriately model the data as per the user requirement. Associations between fewer number of variables can be performed through single summary statistics and significance testing. However, our model is far more complex with nearly 70 variables - a mixture of numerical, ordinal and categorical.

4.1 Models

There are numerous modelling techniques when high number of variables are involved. Here we discuss some of the most commonly used statistically based data modelling techniques capable of model description and associations.

Multiple Linear Regression

Linear regression is the single most used statistical technique to model the relationship between two or more variables through fitting a linear equation to the observed data. If the relationship between two variables are modelled, of which one is the dependent/response variable that is influenced by the other- the independent/predictor variable, we call it a simple linear regression. If the response variable is influenced by several predictor variables, we use Multiple Linear Regression (MLR) expressed as,

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots \beta_k x_k + \epsilon, \tag{3}$$

where, y is the response variable, $x_1, x_2, \ldots, x_k$ are the 'k' predictor variables influencing y, and ϵ is the error term associated. The terms $\beta_1, \ldots \beta_k$ are regression coefficients, the estimates of the unknown population parameters that describe the relationship between a predictor variable and the response. The coefficient value represents the mean change in the response, given a unit change in the corresponding predictor. The intercept β_0, is principally the expected value of response, $\hat{y}$, when there is no combined effect of the predictors. Neglecting the error term (note that the error term is often omitted in the expression), the expression resembles a straight line (recall $y = mx + c$) in a bidirectional plane, hence MLR possesses property of *linearity* and thus the name. The two main objectives of performing a MLR are,

- Establish a statistically significant relationship between the predictors and the response.
- Forecast the unobserved values using the established relationships.

Lasso

The Least Absolute Shrinkage and Selection Operator (LASSO or Lasso) is one of the most popular recent advancements in regression analysis that performs regularisation together with variable selection. Lasso enhances the interpretability of the statistical model and prediction accuracy through subset selection. Lasso imposes a constraint on the absolute value of the coefficients of the model parameters where their sum takes an upper bound. It uses penalty functions to minimise the absolute values of the regression coefficients, that drives the coefficients of non-significant parameter estimates quickly to zero, thereby getting them eliminated

from the model. At the same time, the variables that possess a strong association with the response are identified and retained in the model, thereby facilitating efficient feature selection. The tuning parameter 'λ' may be increased to push more variable coefficients to zero entailing a model with lesser predictors, and when $\lambda = 1$ we obtain the full model.
Lasso coefficients are the solutions to the optimisation problem,

$$\begin{aligned} PRSS(\beta) &= \sum_{j=1}^{n}(y_i - x_i^T\beta)^2 + \lambda \sum_{j=1}^{p}|\beta_j| \\ &=(y - X\beta)^T(y - X\beta) + \lambda|\beta_1| \end{aligned} \tag{4}$$

where PRSS is the Penalized Sum of Residual Squares.

Generalised Linear Model

Generalised Linear Model (GLM), an extension of Linear Models (LM), is one of the unified modelling processes in statistics to gather relationships between the model variables for behavioural and social analysis. Unlike LMs with many limitations that allow it to be used only in special contexts, GLM offers a generic method by unifying several methods such as MLR, Logistic and Poisson regression. While LMs assume that the residual or the error term follow a normal distribution, in GLM, the error term is flexible to follow any distribution from the family of exponentials including the Gaussian. This makes LM a specific case of the GLM. A GLM is expressed as,

$$f(Y) = b_0 + b_1X_1 + b_2X_2 + \ldots + b_kX_k + \epsilon$$

Some of the advantages that the GLM offers in comparison to LM are,

- Transformations of the model are smoother in GLM, which are restricted in LM. For instance, transformation to a logarithmic scale is not always possible in LMs as they encounter negative effects.
- Type of response variable is restricted to 'numerical' type in LMs, whereas, in GLM, it can have the widest flexibility.

At the center of GLM, we have the three components as defined by Nelder and Wedderburn, Nelder & Baker (1972). **Doubt area**

- The distribution model for the response, here the random component.
- The systematic component, here the linear predictor of the predictor variables.
- A link function connecting the mean of the response to the linear predictor, the intercept.

Neural Networks

The success of Neural Networks (NN) is largely associated with the availability of huge data, surge in the computing capacity of machines as well as the availability of superior optimisation methods. A NN presents a value at its output layer depending on the training set of data at its input layer as well as the operations at the hidden layer(s) that comprises of many neurons (nodes). The *number of iterations* determines the number of times that the algorithm gets executed and the *desired error level* is the demanded accuracy for the result. The training stops once it achieves either of the above. The paths between the neurons carry a *weight* and the neurons itself carry a *bias*. At any node, all the incoming neurons are multiplied with their path weights with the bias added to it. This value is passed through an *activation function* at each neuron which helps to impart non-linearity. Activation functions are dedicated non-linear functions such as sigmoid and tanh. We thus obtain a value at the output neuron, which is still random to a great extend.

Now, the weights and biases of neurons and paths are recalculated and adjusted by proceeding backward from the output layer all the way to the input layer. This is called *back propagation* and is one of the greatest advantages of NNs over conventional modelling techniques. Now, the *Learning Rate (LR)* defines the rate at which the network learns from the train dataset. For instance, in a NN system that distinguishes the images of cats and dogs, a low LR would require a large number of images of cats and dogs to be fed into the system, whereas with high LR, less number of images would suffice. Low LR comes with a high accuracy, evidently at a higher computational cost, and a high LR would mean better computational advantage, but at the accuracy at stake. Next is *momentum* that describes the influence of the past weights and biases in the current one. As an example, if the past outcomes(s) gave high errors continually, a high momentum in the next step would push the error term go down. Thus LR and Momentum act in synchronisation to yield an accurate and efficient training process. Numerous iterations of back propagation algorithm, complimented with a carefully selected LR and momentum and subsequent updation of the weights and biases delivers an accurate result in an NN.

We do not proceed with GLM as our situation is lesser complicated considering that we do not intent any model transformations. One of the caveat with NN is the chance of overfitting as sometimes, it may appear harmless to add more hidden layers to the network. Moreover, it requires specific tools and methodologies to be adopted. In our case, we confine ourselves to both MLR and Lasso as they are the most widely used methods for selecting the covariates in the model selection process. Further these methods are at an advantage due to their easy implementation and superior interpretability. Our aim is to reduce the error, $\epsilon = y - \hat{y}$, $\hat{y}$ being the forecast and y, the actual test dataset value.

Utilizing both these methods, we obtain a model that includes all the available predictor variables, or the 'full model'. However, the full model need not necessarily

represent the 'best model' for the user mainly due to two reasons. Firstly, including variables that are not evidently significant can conflict the prediction leading to statistical error. The condition called overfitting, occurs when the regression coefficients tend to represent the noise/error in the data rather than the genuine relationships in the population. This bring misleading and distortive regression coefficients, R-squared values as well as p-values. Secondly and specially in cases like ours as of power systems, energy operators in real scenarios would like to collect minimal, but statistically significant information from their customers. As an example, in our case, the full model contains about 70 predictors manifesting the customer energy consumption. However, it is not realistic for energy operators to obtain such a high volume of information from each customer due to many reasons such as policy regulations from the governmental side on privacy breach, difficulty in obtaining the data and on account of the customer comfort consideration.

Hence, our next objective is model selection that enables us to select the most appropriate model as per the requirement. Many classical as well as modern methods are being performed in statistical domain for model selection. While stepwise regression is a classical method that offers unmatched comprehensibility among all statistical methods, Lasso has become one of the favourite methods for statisticians recently due to its ease and advantages. After performing model selection with both the methods, the results and performances are discussed with a view to provide the users to set their preferred method based on their specific application.

4.2 Model Selection with MLR:- Stepwise Regression

Algorithm 1 Stepwise regression

Inputs :
Model, the full GLM model,
Thresh, threshhold value of 'p',
No. of iterations,
modelNames, variable names in the model,
varName, variable with the maximum 'p' value,
pMaxIndex, the index of varName,
pVal, value of 'p' of varName,
for i = 1 to obs **do**
  if pMaxIndex !=1 **then**
    $variableName = pMax[Model[i]]$
    Initiate *Fisher's exact test*
    if pVal[varName] ≥ thresh **then**
    Eliminate the variable from the model
    end if
    if pMaxIndex ==1 **then**
      $variableName = pMax[modelNames[2:end]]$
      Initiate Fisher's exact test
      if pVal[varName] ≥ thresh **then**
        Eliminate the variable from the model
      end if
    end if
  end if
end for

... see file **"loadforecasting.jl"**

Stepwise regression is further classified as forward selection and backward elimination. Forward selection provides an initial screening of variables where each variable is continually added to the model, whereas, in backward elimination, the method starts with the full model with all independent effects considered at first, and variables are dropped one at a time. Here we proceed with backward elimination as described in Alg. 1. First, we obtain the model with all the candidate variables. For each of these predictors, we identify their association to the response variable after observing their 'p-values'. The procedure to obtain the 'p-value' is described in the next section. It is this p-value that determines whether a candidate variable is retained or eliminated from the model, subsequently which the model is re-fitted with the new set of predictor variables. This process is continued until a stopping criteria is achieved. For energy operators, the maximum information that can be collected from their customers can form a stopping criteria, but in this paper, since we like to keep it in the more statistical way, we select the knee point as the stopping criteria, which also will be discussed later.

The categorical variables are treated as variables possessing 'dummy variables' and the data is sorted into mutually exclusive categories. Ordinal variables are treated either as continuous or categorical depending on their nature. The MLR model is implemented with the dedicated `GLM` package and `fit` command. The way in which formula, *fm* update of the model is performed is explained in the next paragraph.

```
using GLM
model = fit(LinearModel, fm, trainSet)
```

(see file "**loadForecasting.jl**")

Formula Update

In order to automate the process of both formula identification and formula update after every successive backward elimination step, the dedicated Julia function `formula` is used. This function helps the user to avoid type in hundreds of variables of a big dataset which can be cumbersome or sometimes, impossible as in our case. Here, the response variable is defined as a function of all predictor variables. These predictor variables, get eliminated, one at a time from the equation following each successive iteration. The process is executed as below,

```
dfNames = names(newDf)
responseVar = parse("KWH")
predictorVars = dfNames[dfNames .!= responseVar]
fm = Formula(responseVar, Expr(:call, :+, predictorVars...))
```

(see file "**loadForecasting.jl**")

Fishers Exact Test

To advance from the full model to the best model, it is of our interest to knock out the least important predictor variables and retain the significant ones in the model. To establish levels of association between predictors and the response, we perform significance tests for each candidate variable.

For each candidate variable, thus, we incline towards Fisher's exact test, a test that uses an exact, maximum accuracy method that has no distrubutional assumptions and explicitly maximises the model accuracy. To perform the test, we define two groups. Our first group, which we call as the 'base model', consist of the response and the whole set of predictors, or the full model. The second group consist of the response and all the predictors, but with one or more variable(s) missing, whose association with the base model need to be explicitly characterised. We call this missing variable(s) *Variable X* and we name this group as 'second model'. Our intention of using Fisher's exact test is to obtain the association (if any) of *Variable X* with the base model. This is done by obtaining the exact p-value (as the name of the test suggests) of the second model with that of the base model.

Now, p-value, or probability value is achieved for this pair of group through hypothesis testing. Traditionally, p-values are used in Statistical Hypothesis testing to determine if the null hypothesis should be rejected (low p-value) or not (high p-value). Here we use the p-value to order the association of variables. At each iteration, the candidate variable with the highest p-value is removed from the model.

From this section onward, we differentiate between *data variable* and *model variable*. Data variables are the variables obtained in the dataset after performing the steps 3.2 to 3.6, whereas, multiple levels of categorical variables contribute to the model variables. The virtue of using Fisher's exact test stems from the fact that each categorical variable induces multiple variables in the model. A categorical variable with 'n' levels induces '$n-1$' model variables in the model. Our use of Fisher's exact test will give all $n-1$ model variables the same p-value.

Consider a system with 'p' variables.

$$Y = \beta_0 + \beta_1 X_1 + \ldots \beta_l X_l + \ldots \beta_{l+k} X_{l+k} + \ldots \beta_p X_p + \epsilon, \quad (5)$$

where, $(X_l, X_{l+1} + \ldots X_{l+k})$, all correspond to a single categorical variable with '$k+1$' levels.
In this step, we carry out Fisher's exact test with

$$\begin{aligned} H_0 : \text{Model} =& \beta_0 + \beta_1 X_1 + \ldots \beta_{l-1} X_{l-1} + \beta_{l+k+1} X_{k+1} \\ &+ \ldots \beta_p X_p + \epsilon, \\ H_1 : \text{Model} =& \beta_0 + \beta_1 X_1 + \ldots \beta_p X_p + \epsilon, \end{aligned} \quad (6)$$

We set the Null Hypothesis (H_0) as there is no significant association between the base model and the second model and the alternate Hypothesis (H_A) to be against the null hypothesis stating that there exists a significant relationship between the two groups. Therefore, in our case, a high p-value (H_0 true) advocates that the association between the groups is just random and such variable(s) are eliminated from the model. A relatively small p-value (H_0 not true) suggests that there is significant relationship between the groups/models and such variable(s) are retained in the model.

Algorithm 2 Fisher's exact test

Inputs :
Dataset
Categorical columns
Response variable
Variable X

`function:` Obtain **Design Matrix**
`function:` Obtain **Base Model**
`function:` Obtain **second model** with Variable *X*

for all i=1 to *obs*, **do do**
`function:` Obtain **first model**, fit(base model)
`function:` Obtain **second model**, fit(second model)
Calculate SSR for first model and second model
Calculate SSR difference between models
Calculate $f0$ Stat
Calculate h_0 distribution
Calculate CCDF

end for

CCDF: Complementary Cumulative Distribution Function
SSR : Sum of Squared of Residuals
h_0 : Null hypothesis

In short, Fisher's exact test implemented here take each numerical data variable and maps it to an identical model variable, whereas, each categorical data variable with l levels is mapped to a single model variable.

The Fisher's exact test is implemented here as a stand alone function *FishersTest.jl* as shown in Alg. 2. To the best of authors' knowledge, this is only Julia package that implements Fisher's exact test.

4.3 Model Selection:- Lasso

We use Lasso to compare the results that we obtained from the stepwise regression. The reason for using Lasso, is its recent popularity in the statistical domain due to its high prediction accuracy and feature selection capability. We implement it here so that better flexibility is offered when encountered with data of diverse characteristics. Multiple Linear Regression may not be the most appropriate choice if the available dataset has $p >> n$, where 'p' is the number of model parameters and 'n' is the number of observations. Moreover, when there is a high correlation between the predictors (whose effects we omitted here), statisticians agree that Lasso could better exhibit feature selection than stepwise regression.

CCDF: Complementary Cumulative Distribution Function
SSR: Sum of Squared of Residuals
h_0: Null hypothesis

Julia has the capability of calling Python, C and R commands and execute them in the Julia console. With this convenience, we implement Lasso in 'R' and import it in Julia using the function `RCall`. Lasso is realised as per the algorithm Alg. 3.

Algorithm 3 Lasso

Inputs :
Dataset
Categorical columns
Categorical levels for factors
'Groups' array of factor levels Response variable
λ *values,* X

`function:` Obtain **Variable** X
`function:` Obtain **Design Matrix**
`function:` Obtain 'groups' for **Group Lasso**

for all x_i , **do do**
`function:` *Perform* fit_i , **gglasso**
`function:` *Obtain* **coefficients,** fit_i
`function:` *Perform* **validation** in validation set
Calculate standardised error of residuals:- validation vs prediction
Calculate R Squared error:- validation vs prediction
Calculate standardised error of residuals:- test vs prediction

end for

In stepwise regression, recall that we utilized the special case of Fisher's exact test to decide to keep or eliminate each predictor from the model. General Lasso solution that selects individual dummy variables - data variables- instead of the model variables, is insufficient in our case with many categorical (factor) variables. Choosing different contrasts for a categorical predictor will produce different solution in general. Thus in our case, we use a special case - *Group Lasso*. In Group Lasso, the categorical predictors are encoded are in multiple dummy variables as binary co-variates, often called as one-hot encoding. Since including/excluding only a few of these co-variates does not make mathematical sense, each categorical variable with multiple model variables is treated as a single entity in the model selection process. In other words, all the model variables corresponding to a predictor is either retained within the model or eliminated. The objective function for the Group Lasso is a generalisation of the standard Lasso objective,

$$\min_{\beta\in R^p}\left\{||y-\sum_{j=1}^{J}X_j\beta_j||_2^2+\lambda\sum_{j=1}^{J}||\beta_jK_j||\right\}, \quad (7)$$
$$||z||_{K_j}=(z^tK_jz)^{\frac{1}{2}},$$

Here the design matrix X and the co-variate vector β is replaced by a collection of design matrices X_j and co-variate matrices β_j where the penalty is defined by the vector K_j. If $K_j = I$, implies that the data variables for a candidate variable and its model variables are the same, then it reduces to standard Lasso.

For model selection in Lasso, we rely on the value λ. As explained, it is the λ that decides what predictors to be eliminated from the model and what not to. To obtain the best model, we select the Lasso model corresponding to a specific value of λ that optimises the standardised error of residuals. This process is explained in detail later.

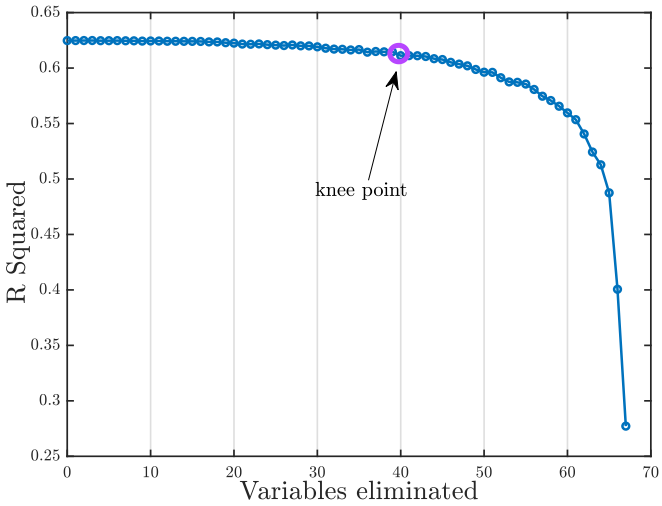

Fig. 8. R-Squared values on successive variable elimination.

5 Results

Validity and accuracy of the model is tested using the validation dataset -recall D_5 in Fig. 6. The'R-squared' values of each model following every subsequent variable elimination is calculated and presented in Fig. 8. R-squared value of the model is a statistical tool to measure the degree of performance of the model attributed to the performance of a particular predictor variable. R-squared varies between 0 and 1, where 0 says that the response cannot be predicted by any of the predictor variables and 1 indicating that it can be predicted without error from the corresponding variables.

Though do not quantify the performance of the model as by itself, in our case, finding R-squared helps the user to get an idea about the performance of the model in order to select the best model available through recommending a value for 'k', recall 'k' as the number of data points to be collected from the customers. According to the graph, R-squared shows a sharp fall after the elimination of nearly 40 variables, which we call *knee point*. Further elimination of variables after the knee point would mean that remaining predictor variables could barely quantify variability of the response. Hence, R-squared *may* to form the basis of selection of an appropriate value for 'k'. Note that the user can have other constraints on the number of variables that need be retained in the model, hence the word "may".

Now, to verify this knee-point, we also look at Fig. 9, that tells the standardised error of residuals. The standardised squared residuals are plotted against the predictor variables at various levels of variable involvement in the model. Here, we get a more accurate knee-point, which is after the 39^{th} variable removed from the model. In other words, the best model - as identified by

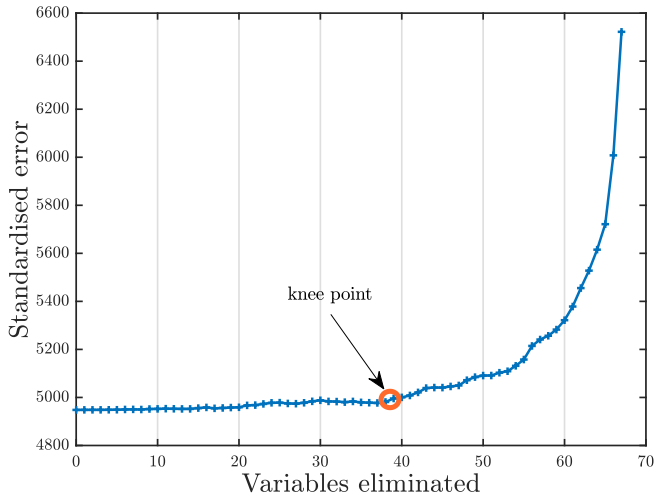

Fig. 9. Standard deviation of squared error on successive variable elimination- Stepwise regression

the stepwise regression model selection procedure is a model that has 29 predictor variables in the model.

The knee point is treated as an optimal point during variable elimination, above which describing the model would render too expensive owing to the presence of large number of variables and below which the model characteristics are not properly described due to the unavailability of variables. Based on this impression adopting stepwise regression, it is found that the model can be described effectively with just 29 variables. The 29 variables thus identified through stepwise regression are summarised in Table. 3 ordered as types and in their decreasing significance.

Now, we look at the model selection procedure with Lasso. Fig. 10 that illustrates the standardised error of residuals that are encountered at various stages of the variable elimination procedure. Not surprisingly, the trend of the graph agrees with that of the results in stepwise regression. From the graph, it is observed that the model accuracy shows a sudden fall after the 38^{th} variable is eliminated from the model, which claim that the best model is the model with 31 variables. The 30 variables thus identified through Lasso are summarised in Table. 3 ordered as types and in their decreasing significance.

In Fig. 11, we compare the classical method with the more modern method in their prediction accuracy in the light of the figures Fig. 9 and Fig. 10. When it comes to the prediction accuracy, it is seen that stepwise regression clearly outperforms Lasso.

Both methods clearly indicates the same trend over the entire variable elimination phase indicating similar results and performance accuracy. We see a slow increase at the start until the knee point - when less significant variables are eliminated, followed by a an abrupt increase after the knee point.

It is typical to observe in regression problems, a U

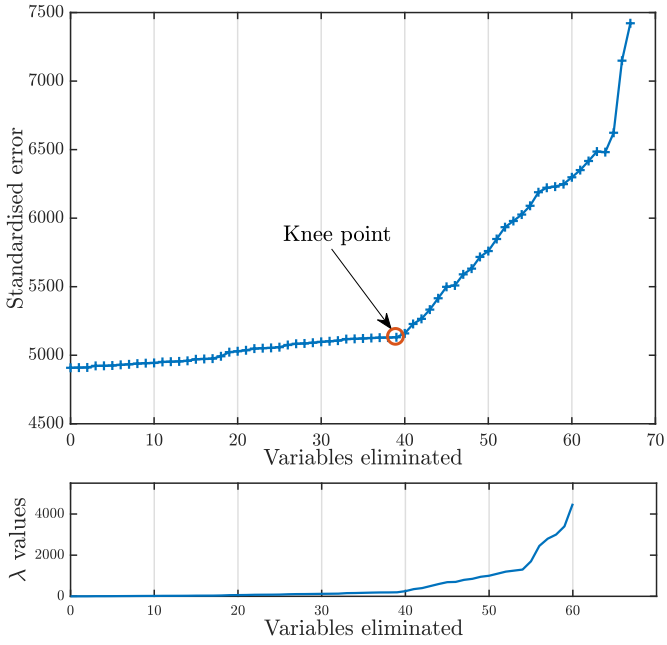

Fig. 10. Standardised error and λ values on successive variable elimination-Lasso

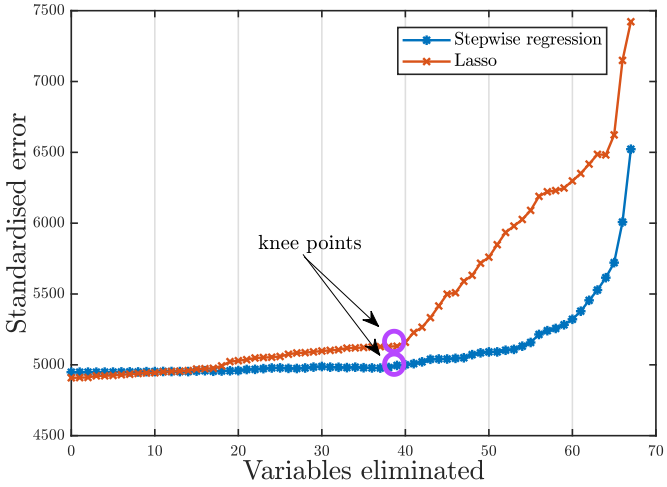

Fig. 11. Comparison of standardised errors on successive variable elimination-Stepwise regression vs Lasso

- shaped curve for the standardised error graph due to the increase of accuracy when less significant variables are eliminated first, and decrease of accuracy later when significant variables start to get eliminated. In our case, we see a contradiction largely due to the legitimacy of the manual variable screening of (less-significant) variables that is performed prior to the automated data processing stages. The response is described the best with all the predictors, nevertheless, not the 'most efficient way' of representation, since our way of looking the efficiency is in addressing the energy operator's ca-

Table 3
Identified most significant variables as per the knee point - Stepwise Regression and Lasso.

Stepwise regression		Lasso	
Numerical variables		Numerical variables	
1.	Number of bedrooms	1.	Number of bedrooms
2.	Number of full bathrooms	2.	Number of full bathrooms
3.	Number of half bathrooms	3.	Number of meals cooked
4.	Number of refrigerators	4.	Number of refrigerators
5.	Number of freezers used	5.	Number of freezers used
6.	Number of televisions	6.	Age of the most used refrigerator
7.	Total television operation time	7.	Number of televisions
8.	Number of ceiling fans	8.	Total television operation time
9.	Total light hours in house	9.	Number of ceiling fans
10.	Presence of sliding doors in heated areas	10.	Total light hours in house
11.	Number of households in the house	11.	Number of windows in heated areas
12.	Mean age of the households	12.	Number of households in the house
13.	Total area of the house	13.	Mean age of the households
14.	Total cooled area in the house - ACs	14.	Total family income of the household
	–	15.	Total area of the house
	–	16.	Total heated area in the house- ACs and heater
	–	17.	Total cooled area in the house - ACs
Categorical variables		Categorical variables	
15.	Age band of the house	18.	Age band of the house
16.	Wall type of the house	19.	Microwave usage frequency and time
27.	Toaster usage frequency and time	20.	Dish washer usage frequency and time
18.	Dish washer usage frequency and time	21.	Washing machine usage frequency and time
19.	Washing machine usage frequency and time	22.	Cloth drier usage frequency and time
20.	Cloth drier usage frequency and time	23.	Television usage on weekdays
21.	Television usage on weekdays	24.	Television usage on weekends
22.	Usage of well water pump (if any)	25.	Usage and frequency of heating equipment
23.	Usage of central air conditioner in last summer	26.	Type of fuel used by the water heater
24.	Usage of programmable thermostat for central AC	27.	Usage of central air conditioner in last summer
25.	Usage of hot tub (if any)	28.	Type of usage of natural gas at home (if any)
26.	Type of usage of natural gas at home (if any)	29.	Usage of electricity for space heating (if any)
27.	Usage of electricity for space heating (if any)	30.	Usage of electricity for water heating (if any)
28.	Usage of electricity for water heating (if any)		–
29.	Highest level of education of the main household		–

pacity to predict the energy based on fewer number of customer data points.

Energy Prediction

From obtaining the 'best model' from the 'full model' through model selection procedure, we proceed to our final objective - to predict the energy consumption of customers from the test predictors. For the same, we use the function from 'Hypothesis' package in Julia, `predict` as well as in R as below,

```
predictModel = predict(model, testSet)
```

(see file "**loadForecasting.jl**")

The function fits the regression onto the test dataset as per to the selected 'best model'. Varying levels of variable elimination result in varying levels of accuracy in prediction. To investigate this varying accuracy of the model, we examine the 'residuals' of the prediction. A residual, denoted by e is the difference between the observed value of the response, y and its corresponding predicted value, $\hat{y}$.

$$e = y - \hat{y} \tag{8}$$

Table 4
Standardised error in the test sets for selected models

Method	Standardised error
Stepwise regression	5091.8
Lasso	5167.4

Now, the standardised errors of residuals of the test dataset - recall D_6 in Fig. 6 - is given in Table. 4. As expected (since the overall trend of standardised error throughout the variable elimination is lower in stepwise regression), the standardised error for the stepwise regression is lower than that of Lasso. This means that considering the prediction accuracy, the more classical method of stepwise regression performs better than the modern method.

6 Conclusion

This paper discussed the common challenges and complexity encountered whilst handling large datasets in a power engineering context and addressed it in an easily implementable step-by-step manner. An instructional layout is provided with the access to the functional codes for each step carried out. The new high performance technical computing language, Julia is introduced and its capacity and compatibility in dealing with complex datasets is manifested.

The tool *'loadForecasting.jl'* is developed in Julia and provided the insights as desired. Starting from 940, most of the variables are eliminated by a manual subjective variable selection and their aggregated clustering. Many of these aggregate variables are then normalised in order to obtain better interpretable form of energy consumption characteristics. The train dataset is then randomised in order to prevent any possible bias responses and the entire dataset is scrutinised for missing data points following a special hot-deck imputation. The data processing stages is followed by energy modelling. Though the full model provides the most information, constraint considerations from prospective stakeholders demand the size of the model to be reduced considerably (and) as per requirement. For model formulation and model selection, we utilize tow popular approaches - a more classical Multiple Linear Regression and a recent method Lasso. In multiple linear regression, less significant variables are eliminated through iterative stepwise regression and backward elimination utilizing Fisher's exact test to take into account categorical variable associations. With Lasso, the λ value is varied to identify the best model and Group Lasso is utilized to consider the categorical variable interactions. The regression coefficients and their corresponding p-values are obtained from the both the models as a tabular.

After validation and arriving at the best models using the validation dataset, energy consumption of customers is predicted for the test dataset using dedicated Julia functions for both the methods.

For better use to the community, the paper provide the links to access all the functional, reproducable codes that are publicly available in GitHub. To the best of authors' knowledge, there are no other packages available in Julia to implement Fisher's exact test as well as stepwise regression with categorical variable consideration. Moreover, our tool *loadforecasting.jl* is unique in Julia by allowing the users to grasp the finest details of data analysis and modelling. Though this paper outlines the procedure to tackle an energy related problem, the same can be extended to numerous other applications including demand, asset and ancillary services management with minimal modifications in the codes.

References

Akhavan-Hejazi, H. & Mohsenian-Rad, H. (2018), 'Power systems big data analytics: An assessment of paradigm shift barriers and prospects', *Energy Reports* **4**, 91–100.

Alfares, H. K. & Nazeeruddin, M. (2002), 'Electric load forecasting: literature survey and classification of methods', *International journal of systems science* **33**(1), 23–34.

Allison, P. D. (2001), *Missing data*, Vol. 136, Sage publications.

Andrew Ng, K. K. (2018), 'Lecture notes in cs 230, deep learning'.

Benini, L., Mancini, L., Sala, S., Manfredi, S., Schau, E. & Pant, R. (2014), Normalisation method and data for Environmental Footprints, Technical report, European Union.

Bonsack, M. & Skoryk, V. (2017), 'The power of data normalisation: A look at the common information model', University Lecture.

Chen, S.-T., Yu, D. C. & Moghaddamjo, A. R. (1992), 'Weather sensitive short-term load forecasting using nonfully connected artificial neural network', *IEEE Transactions on Power Systems* **7**(3), 1098–1105.

Endrenyi, J., Anders, G. & Da Silva, A. L. (1998), 'Probabilistic evaluation of the effect of maintenance on reliability. an application [to power systems]', *IEEE Transactions on power systems* **13**(2), 576–583.

Fan, J., Han, F. & Liu, H. (2014), 'Challenges of big data analysis', *National science review* **1**(2), 293–314.

Golbraikh, A., Shen, M., Xiao, Z., Xiao, Y.-D., Lee, K.-H. & Tropsha, A. (2003), 'Rational selection of training and test sets for the development of validated qsar models', *Journal of computer-aided molecular design* **17**(2-4), 241–253.

Jagadish, H., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J. M., Ramakrishnan, R. & Shahabi, C. (2014), 'Big data and its technical challenges', *Communications of the ACM* **57**(7), 86–94.

Kabalci, Y. (2016), 'A survey on smart metering and smart grid communication', *Renewable and Sustainable Energy Reviews* **57**, 302–318.

Lewis, S. C., Zamith, R. & Hermida, A. (2013), 'Content analysis in an era of big data: A hybrid approach to computational and manual methods', *Journal of Broadcasting & Electronic Media* **57**(1), 34–52.

Long, H., Zhang, Z. & Su, Y. (2014), 'Analysis of daily solar power prediction with data-driven approaches', *Applied Energy* **126**, 29–37.

Nelder, J. A. & Baker, R. J. (1972), *Generalized linear models*, Wiley Online Library.

Royston, P. et al. (2004), 'Multiple imputation of missing values', *Stata journal* **4**(3), 227–41.

Schmitt, P., Mandel, J. & Guedj, M. (2015), 'A comparison of six methods for missing data imputation', *Journal of Biometrics & Biostatistics* **6**(1), 1.

Siano, P. (2014), 'Demand response and smart grids—a survey', *Renewable and sustainable energy reviews* **30**, 461–478.

Sjøvaag, H., Moe, H. & Stavelin, E. (2012), 'Public service news on the web: A large-scale content analysis of the norwegian broadcasting corporation's online news', *Journalism Studies* **13**(1), 90–106.

US, EIA, RECS (2014).
URL: *https://www.eia.gov/consumption/residential/data/2009/index.php?view=microdata/*

Appendix

All the working codes are found in the web hosting service, GitHub at https://givethe//code//links/. The codes for each of the data processing as well as model selection stages are given separately in separate files for ease of use for the users. The complete dataset, along with the codebook is also given in this link in GitHub.