BUAN 6341.003 - Applied Machine Learning - Assignment 1

Anurag Sethi

September 17, 2019

1 Introduction

This project focuses on the implementation of linear and logistic regression algorithms, that are two most fundamental algorithms in machine learning. The project takes uses *Appliance Energy Consumption Dataset* hosted at *UCI machine learning repository* a popular achive portal for open datasets for academic use. Thus, prediction models of electrical energy consumption in buildings can be useful for a number of applications: to determine adequate sizing of photovoltaics and energy storage to diminish power flow into the grid, to detect abnormal energy use patterns, to be part of an energy management system for load control. The project illustrates in section two using exploratory analysis techniques, the impact of various variables on the energy consumption. The exploratory analysis gives an insight to feature selection and is instrumental in determining the required variables. We should predict Appliance energy consumption for a house based on factors like temperature , humidity pressure as well as extracted features from date-time of the record. In order to achieve this , we need to develop a supervised learning model using regression algorithms.

The crux of the project is the experimentation section, this is where the major learning outcomes of the project lie. There are levers of the optimization algorithms like threshold, iterations and most importantly the learning rate that need that can be user controlled. The study of variation in the outcome of the optimization result with respect to these changing control parameters is important and insight into them enhances the understanding of these optimization algorithms. Interestingly, the project also opens up opportunities for subjective decision making,example when implementing logistic regression, there are important things to consider when making choices like selecting the threshold parameter to convert regression problem to classification. The codes for the project are implemented in python and are attached with the report in the submission.

About the dataset, The data set is at 10 min for about 4.5 months. The house temperature and humidity conditions were monitored with a ZigBee wireless sensor network. Each wireless node transmitted the temperature and humidity conditions around 3.3 min. Then, the wireless data was averaged for 10 minutes periods. The energy data was logged every 10 minutes with m-bus energy meters. There are a total of 19735 observations for the household spanning across 5 months with 29 independent variables.

Linear Regression in mathematical form with n predictor variables is represented as:

$$\text{DepVar}_t = beta_0 + \beta_1\text{IndepVar1} + \beta_2\text{IndepVar2...} + \beta_3\text{IndepVar(n)} + \epsilon \tag{1}$$

2 Data Preparation for regression analysis

This section is a detailed account of data-preprocessing and feature engineering, exploratory analysis and feature selection from the original dataset. The various steps are explained in each of the subsections.

2.1 Data Pre-processing and Feature Engineering

The dataset has 29 features to start with, there is dependent variable Appliances showing the energy consumption by appliances of the house. The other features can be broadly divided into four further categories, Weather Related Features,Humidity related features,Temperature Related Features: each category capturing the various entities at different parts of the house, also there are features like light, date, rv1 and rv2 which are put together as miscellaneous.

Descriptor Type	Variables
Weather Features	T out,Press MM HG,RH Out,Windspeed,Visibility, Tdewpoint
Humidity Features	RH1, RH2... RH9
Temperature Features	T1, T2... T9
Misc. Features	lights, Date, rv1, rv2
Dependent Variable	Appliances

In the pre-processing step we scale the numeric features using the z-score, implemented using the standard scaler operand inbuilt in python. Mathematically, it is given as

$$z = (x - u)/s \tag{2}$$

where u is the mean and s is the standard deviation of each numeric independent variable. This normalization aids in converging of the optimization algorithm as large spread of one independent variable increases the feature space, and gives disproportionate weight to one(or some) of the features. We engineer new feature from date variable, which we think will be relevant based on the understanding of the context. There are listed in the table below. The rationale behind this is that intuitively appliance consumption is affected by seasons (captured by month), day of week (captured by isweekend), and section of the day (captured by time of the day).

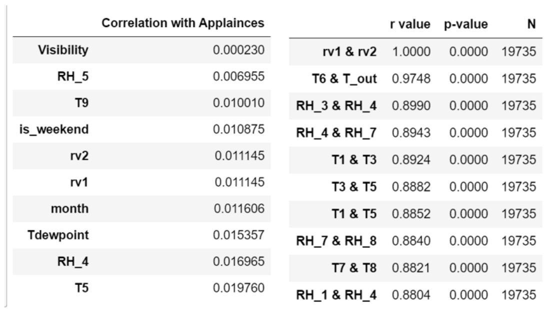
Variable Name	Description
Month	Descriptor labelled 1 to 5 representing month
Is Weekend	Flag if the day is a weekend or weekday
Time of Day	Section of the Day eg. night time, evening time, work hours etc.

2.2 Feature Selection

In this section we implement feature selection in two steps, for the tasks listed and experiment 1 and 2, we have to do implementations using a minimum of 15 variables. At this beginning stage we have 32 independent variables including the engineered ones, in the python code please refer to variable *XMaster* as it contains the original list of 30 independent columns. We reduce these 30 variables to 21 for first cut implementation using *pearson correlation.*

$$\rho = \frac{\text{cov}(X,Y)}{\sigma_x \sigma_y} \tag{3}$$

Left: Correlation with Appliances, Right : Pairwise Correlation among Dependent Variables



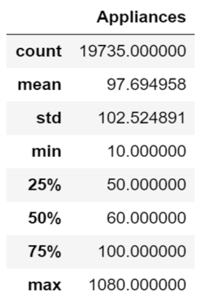
	Correlation with Applainces
Visibility	0.000230
RH_5	0.006955
T9	0.010010
is_weekend	0.010875
rv2	0.011145
rv1	0.011145
month	0.011606
Tdewpoint	0.015357
RH_4	0.016965
T5	0.019760

	r value	p-value	N
rv1 & rv2	1.0000	0.0000	19735
T6 & T_out	0.9748	0.0000	19735
RH_3 & RH_4	0.8990	0.0000	19735
RH_4 & RH_7	0.8943	0.0000	19735
T1 & T3	0.8924	0.0000	19735
T3 & T5	0.8882	0.0000	19735
T1 & T5	0.8852	0.0000	19735
RH_7 & RH_8	0.8840	0.0000	19735
T7 & T8	0.8821	0.0000	19735
RH_1 & RH_4	0.8804	0.0000	19735

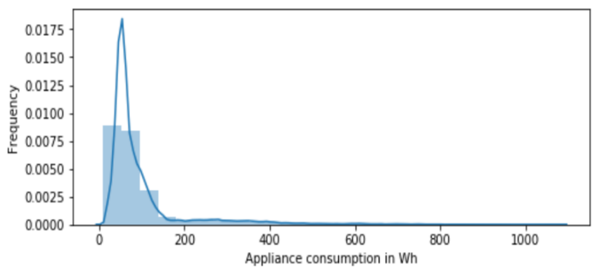
It is defined as covariance of two variables, divided by the product of standard deviation of the two variables. We first observe the co-relation of the independent variable with the dependent variables, and remove the dependent variables that have the score below a certain threshold. Secondly we want to make sure that the multicollinearity is minimised. We check the pairwise coorelation score among the dependent variables and drop the variables which show high values of pairwise correlation. Refer Figure above, here we remove *Visibiity, RH5 and T9* since they have very low correlation with the dependent variable. In the pairwise correlation analysis, we remove rv1, T6, Rh3, Rh4, T1, T5: since they high pairwise correlation greater than 0.88, and very low (0) p values making the pair wise correlation statistically significant. After this feature reduction we remove these 9 independent variables, and use 21 variables for the analysis. In the python code these are referred as XMaster-Updated.

2.3 Exploratory Analysis

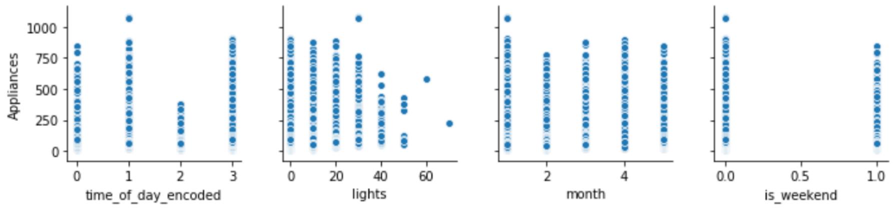
Distribution of target variables



	Appliances
count	19735.000000
mean	97.694958
std	102.524891
min	10.000000
25%	50.000000
50%	60.000000
75%	100.000000
max	1080.000000



Scatter plot of Energy consumption with respect to selected independent variables



Here we first see the distribution of the dependent variable, in the entire dataset, we have the mean consumption at 97 KwH, and comparatively large standard deviation at 102. It indicates that the spread is large, it could potentially be harder problem to get a regression fit, specially with a metric like RMSE, which is sensitive to the spread of dependent variable.

Next we plot the scatter plot of Dependent Variables with respect to selected columns and see any observable patterns that we expect. The most important observation is that the use of appliances is higher in morning time (time of day 1) and lower in work hours/afternoon (time of day 2). Correspondingly the appliance usage is slightly lower during weekends and has an inverse relation with lights usage. The appliance usage is higher in the month of January, possibly because of higher consumption in the winters.

3 Project Tasks

Project specifies a set of tasks that need to be done before carrying out the experiments. The task revolve around splitting the dataset into training and testing and writing building basic functions for linear and logistic regression.

Task1: The dataset is split into training and testing in the python code, we use the split ratio as 77.5/22.5 percent and divide the train and test set.

Task2: The function **linear regression mains** in the python code in the implementation of this task, it invokes dependent function for hypothesis calculation and gradient descent implementation as well.

Task3: The function **gradient descent mains** in the python code in the implementation of this task.

Task4: In the dataset the split was logistic regression is chosen at 150 KhH above as spike usage, else normal usage The rationale behind the split is explained in the section logistic regression and accuracy metrics are reported as well. Refer function in code.

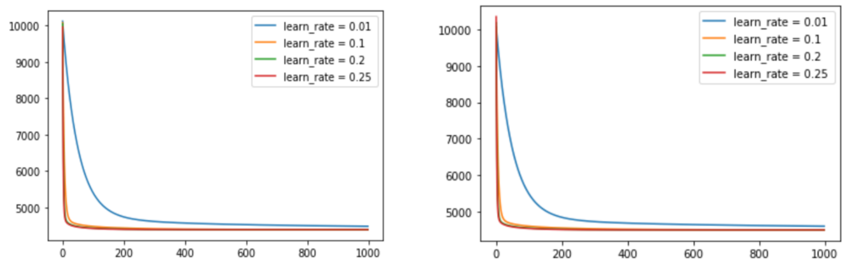
4 Project Experiments

Experiment 1

Here we build upon the implementation of Linear Regression and report cost calculated, model performance on train and test set. We choose the reduced set of 21 features. We vary the parameter learning rate and record the results for four values of alpha, namely **0.01, 0.1, 0.2, 0.25**. In this experiment threshold values are not set, we run the linear regression model the entirety of 1000 iterations, which is the maximum iterations set for this project for every run of regression model. Below is the summary of key metrics, for each value of the four learning rates we have final cost after maximum iterations on train and test data, train set error, test set error.

Learn Rate	CostTr Min	CostTe Min	RMSE Tr	RMSE Te
0.01	4475.476	4601.51	94.56775	96.04471
0.1	4384.916	4500.84	93.64740	95.07818
0.2	4384.484	4499.66	93.64278	95.06325
0.25	4382.482	4499.59	93.64275	95.06318

Left: Train Set Cost vs Iteration, Right Test Set Cost vs Iteration



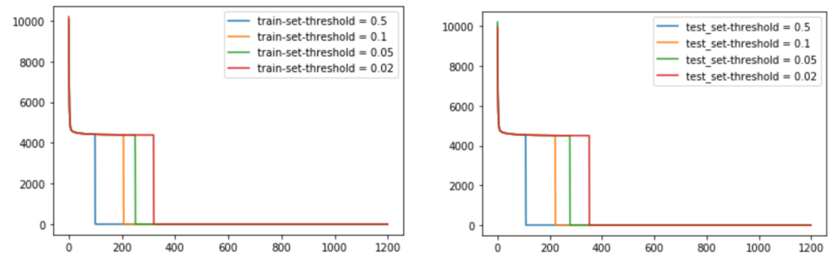
- In the key metrics table, we can observe that, while we are increasing the learning rate, we are getting better output, for the given set of learning rate the best results were for rate = 0.25. Also the train and test set RMSE values are close to each other, which is a positive indicator of the model being able to replicate the training performance on out of sample data points.
- We also tried running the model for learn rate 0.3, it did not converge, the cost increased exponentially after a few iterations and tended to infinity. Hence it is important to select correct learn rate, 0.25 learn rate is the right trade off as it converges sooner than learn rates lower than 0.25. It is important to note that the RMSE and Cost Metrics for this learn rate are not significantly different from the metrics obtained for 0.01, 0.1, 0.2, but since here we have large features of features that increase the dimension space and make it harder for algorithm to converge, 0.25 appears to be the right value for learn rate.
- Another important thing to note is that the model converges relatively quickly, if we refer to train and test cost vs iteration plots (which although look similar but are different), even the lowest learn rate gradient descent converges at around 400th iteration.

Important Note: Logistic Regression implemented in a seperate section

Experiment 2

We carry out experiment with multiple thresholds, in the range [**0.5,0.1,0.05,0.02**], this threshold is set upon the difference in the calculated cost for consecutive iterations, in the code implementation we add a condition to stop the iterations as soon as the marginal decrease in cost falls lower than a given threshold. We use the same functions as before, only difference being the threshold value was set as 0 for experiment 1, here we vary it and see the results. Also, it is important to note that the learn rate is fixed at 0.25 which was the optimal rate observed in experiment 1. Keeping the rate constant enables us to make proper comparison as we vary the relevant variable threshold here.

Left: Train Set Cost for thresholds, Right Test Set Cost for thresholds



Threshold	Iter-Tr	Iter-Te	CostTr Min	CostTe Min	RMSE Tr	RMSE Te
0.5	100	109	4417.3665	4535.8466	93.9859	95.4661
0.1	207	222	4391.2789	4507.6697	93.7150	95.1691
0.05	251	278	4387.9287	4503.8979	93.6787	95.1251
0.02	320	352	4385.8489	4503.8979	93.6572	**109.3342**

- The trivial observation here is that models with high threshold are quicker to converge, and we lower the threshold we require more iterations.
- **The most important observation** and the subsequent inference from it is that we see RMSE Test increasing as we decrease the threshold to 0.02, this is a clear indicator of the model overfilling the training data and performing poorly out of sample. If we furthur decrease the threshold to say 0.01, we will observe the following things, higher number of iterations required to converge, a slight marginal improvment in the train set RMSE, but we will also see a significant increase in the test set RMSE, thereby increasing the overfit.
- Keeping the factors in mind we select the optimal model, with the optimal learning rate and optimal threshold. Other things that we need to keep in mind when selecting the parameters after tuning is that the model should converge in time and work well out of sample. Below is the summary of the optimal model, in the form of the regression equation and the parameters like optimal learn rate and optimal threshold. We have also tabulated the Train and Test Error for the model. As mentioned before having test error close to train error is a good indicator of the model performing well out of sample.

Optimal Model for initial 21 variables

Regression Equation

$$\begin{aligned}\text{Appliances}_{usage} = 97.76 &+ 17.58 * \text{lights} && +59.98 * \text{RH1} + (-26.24) * \text{T2}\\ &+ (-46.83) * \text{RH2} && +46.60 * \text{T3} + (-15.26) * \text{T4}\\ &+ (-5.25) * \text{RH6} && +(-9.23) * \text{T7} + (-5.71) * \text{RH7}\\ &+ 11.44 * \text{T8} && +(-22.01) * \text{RH8} + (-4.34) * \text{RH9}\\ &+ 0.57 * \text{TOut} && +0.26 * \text{PressMMHg} + (-0.49) * \text{RHOut}\\ &+ 2.33 * \text{Windspeed} && +6.88 * \text{TDewpoint} + (-0.21) * \text{RV2}\\ &+ (-24.04) * \text{Month} && +(-1.87) * \text{IsWeekend} + 3.71 * \text{TimeOfDay} + \epsilon\end{aligned} \tag{4}$$

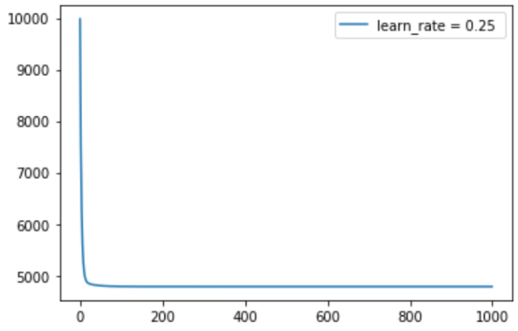
Learn Rate	Threshold	IterationsTrained	RMSE Tr	RMSE Te
0.25	0.05	256	93.678	95.126

Experiment 3

Here 10 features are randomly chosen features, we do a column sample in pandas and get [**rv2, T2, RH9,RH6, month, T8, lights, T4,Pressmmhg, RH2**] as our feature set to work on experiment 3. Just as before we run the model and calculate three important metrics, training cost of converged model, RMSE trainset, RMSE testset. In terms of tuning parameters we keep the learn rate at 0.25 and iterations as max iterations at 1000. It is important to note that we do not select threshold value since our computed optimal threshold in experiment 2 was for larger dimension set. The rate of convergence of the model is sensitive to the dimension of the training set, hence no threshold has to set to alter the natural convergence at max iterations. This is just to aid us to observe the natural behaviour of the linear regression model. The three key metrics are tabulated below.

Learn Rate	RMSE Tr	RMSE Te	Cost Tr Min
0.25	97.82	99.19	4784.4989

Left: Quick Convergence of Gradient Descent with 10 random features



- The first observation is that the model converges very quickly as speculated before, it happens because of reduction in the dimensions of the feature space.
- The train and test rmse are slightly higher (but not significantly), it implies that there is some information lost in the dropped variables, which decreases the effiency of fit by a small amount.
- In the regression equation below the coefficient of T2 changes for negative to positive. This behaviour can be explained by the effect of one of the ommitted variables that would be positively correlated with the dependent variable, the effect of that ommitted regressor is captured in T2.

$$\begin{aligned}\text{Appliances}_{usage} = 97.76 + 19.66 * \text{lights} \quad & +(23.89) * \text{T2} \\ + (11.31) * \text{RH2} \quad & +(-17.47) * \text{T4} \\ + (-24.87) * \text{RH6} \quad & +5.44 * \text{T8} + (-12.10) * \text{RH9} \\ + 3.23 * \text{PressMMHg} \quad & +(-0.33) * \text{RV2} \\ + (-22.43) * \text{Month} + \epsilon & \end{aligned} \tag{5}$$

Experiment 4

Doing **dimensional reduction**, We further narrow down to 10 features, the process followed is more comprehensive, first we convert the month, time of the day, isweekend columns to dummy variables using one hot encoding, we now go ahead and merge is new encoded columns with the other numerics.

As a secondary exercise we again calculate the correlation of the Appliance Energy usage with out new feature set of 36 variables, we sort the absolute value of correlation and pick the top 10 values. The feature set for 10 features we get are: [timeofdaysleeptime, timeofdayworkhours, lights, RHout, T2, T6, Tout, RH8, timeofdaynighttime, Windspeed]

Threshold	Iter-Tr	CostTr Min	RMSE Tr
0.5	18	4635.20	97.705
0.1	28	4634.99	97.729
0.05	39	4633.23	97.725
0.02	103	4631.56	97.715
0.001	279	4629.83	97.705

$$\begin{aligned} \text{Appliances}_{usage} = 97.68 &+ 16.10 * \text{lights} & &+(14.32) * \text{timeofdayworkhours} \\ &+ (3.92) * \text{windspeed} & &+(-12.74) * \text{timeofdaysleeptime} \\ &+ (6.970) * \text{timeofdaynighttime} & &+(-2.94) * \text{RH8} \\ &+ (-2.61) * \text{TOut} & &+2.54 * \text{T6} \\ &+ (4.99) * \text{T2} & &+(-1.45) * \text{RHOut} + \epsilon \end{aligned} \tag{6}$$

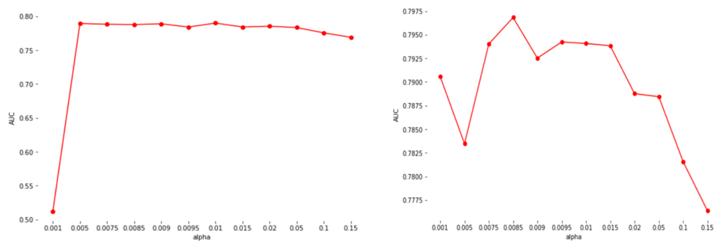
Comparaing All Models	21 Features	10 Random Features	10 Selected Features
RMSE Test	95.12	99.19	97.705
Cost Train	4387.92	4784.49	4629.83

- We run the model with selected 10 features for 10 different values of threshold, the most noticable results are that the models converge much quicker in much fewer iterations compared to the model with 21 dimension feature set, it implies that convergence is very sensitive to the dimension of the feature set. Also, another important thing is that compared to experiment 2, where we saw evidence of overfitting, that is test set error rising upon decreasing the threshold. It is not observed here. Again this tendency to not overfit can be attributed to reduction in dimensionality.
- Comparing the three models we see that using 21 feature set gave the lowest test set error and train cost. One of the reasons could be that we are losing information upon feature reduction. Some variance is there in the dropped columns that is not there in 10 column subset.
- The values of new introduced coefficients indicate some interesting trends, appliance usage is expected to drop post mid night, and peak in the night time, same is indicated by the sign of the coefficients.

Implementation of Logistic Regression

We implement logistic regression using SGDClassifier, prior to that we do following tasks, first is the creation of class column. We create a column called spike alert which basically corresponds to time periods with electricity consumption over 150. The rationale behind selecting it is that it is helpful to know the likelihood of consumption to go up, given time, season, weather, humidity conditions. It can potentially help the service provider in demand forecasting if such an analysis is scaled up for multiple household in a neighbourhood.

Area under ROC Curve vs Learn Rate, Left Test Set : Right Train Set



Optimal Values	Learn Rate	Area Under Curve
Train Set	0.0085	0.7968
Test Set	0.01	0.7900

$$h_\beta(x) = g(\beta^T x) \tag{7}$$

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log(h_\theta(x^{(i)})) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))]$$

We make sure to do stratified sampling, so make sure that the class distribution is uniform in train and test set. The spike in electricity is noted in 11.6 percent of the observations. Ideally we would have set the threshold to be higher, but it would lead to class imbalance. Equations above describe the hypothesis calculation and and the cost function summed up for each y value, similar to linear regression, it is minimized using a gradient descent algorithm using a specified learn rate. We apply the classifier for a grid of learning rates from 0.0001 to 0.15, and take a note of Area under the curve in both train and test set. In the table above which is derived from the graph, we observe that maximum AUC for training set is observed at 0.0085, at this point test set AUC is at lower level, it maximizes at learn rate 0.01. It is an evidence of the risk of over fitting if we select lower learning rate. Selecting lower learning rate will max out the iteration runs of cost optimization learning to the model being over-trained and work poorly on out of sample data. This completes **experiment 1** for logistic regression. For the implementation of logistic regression in **experiment 4 and experiment 5** we take 0.01 as the optimal value of learn rate.

$$\begin{aligned}
\text{Appliances}_{usage} = 0.82 &+ (-0.57) * \text{lights} & +(-0.04) * \text{timeofdayworkhours}\\
&+ (0.02) * \text{windspeed} & +(0.04) * \text{timeofdaysleeptime}\\
&+ (0.15) * \text{timeofdaynighttime} & +(0.17) * \text{RH8}\\
&+ (-0.001) * \text{TOut} & +(-0.0016) * \text{T6}\\
&+ (-0.148) * \text{T2} & +(-0.061) * \text{RHOut} + \epsilon
\end{aligned} \tag{8}$$

Prediction Accuracy(alpha=0.01)	Initial All Vars	Random 10 Vars	Selected 10 Vars
Train Set Accuracy	0.871	0.867	0.887
Test Set Accuracy	0.850	0.862	0.882

Here we can clearly observe that model with selected 10 features performs better, it can imply that the selected 10 features are more clearly able to explain the spikes in usage, than be regressors to predict usage in Kwh, also it is important to note that the differnece between the train and test error is minimal in the model with 10 selected features. Hence, the model performs well on the out of sample datapoints as well (this is partially helped by reduction in dimensionality as well)

5 Discussion

The results were satisfactory for both classification and regression problems. They are discussed in detail, in each of the experiment section, but there is a general scope of improving the regression results, classification results were somewhat good. To summarize the results, in the linear regression results were better when more number of features were used, in that case the trainRMSE, testRMSE and Cost were minimized. The optimal learn rate and threshold have been mentioned before and included in the experiment section. In classification the model with 10 selected features had the best accuracy, possible factors leading to this result have been explained in the logistic regression section.

There could be many steps taken to improve the prediction of results.

- The panel nature of the data should be considered. The dataset provided is a longitudanal data with one entity(household) observed over time. Considering the panel nature of the given dataset will significantly reduce the train and test set errors.
- More information and better data collection, the following data is of the spring time of the year, intuitively the energy consumption by Appliances. Having an year long data helps to keep into account the seasonality factor better as the energy consumption tends to tends to move in sync with changing seasonal cycles.
- Better feature selection techniques, the regression results had a low r-squared values, these were not included in the project report but were calculated. It implies that there was a lot of unexplained variation of the dependent variable that was missed in the modelling process. For instance, one implementation of Principal component analysis, where top 3 Principal components accounted for 90 percent of variation, again it was not presented in report since we were supposed to select at least 10 features for implementation. The model with 3 PC's had a relatively higher value of r-square and explained the variation of the dependent variable better.
- The last point here is subjective and can only be confirmed upon implementing but there is a chance that other methods like ensembles techniques like boosting trees could give a better result and reduce the RMSE values.

6 Key Takeaways

This project was a good exercise in hands on implementation of regression and gradient descent algorithms in python. There were many nuances related to parameters like learning rate and threshold which we were unaware of prior to this project. This project gave and opportunity to augment the learning of linear and logistic regression algorithms with bias - variance trade-off study when we evaluated the effect on train and test set errors. In real world, it is necessary to have the understanding of control parameters of a model, before tuning them to use to solve data driven business or a scientific problem. This project was aptly designed exercise in gaining understanding of the control parameters.