STAT 6560 Project Report

Analysis of Building Energy Consumption with ASHARE Dataset

Liangyu Liu*, Zi Yang, Ningyi Liu and Yilin Zheng

* Team Leader

Table of Contents

1. Introduction

In this project, we are interested in developing accurate models to predict energy usage for buildings equipped with one of four types of meters: chilled water, electricity, hot water, and steam. From houses and hotels to schools and skyscrapers, buildings in the United States consume almost 40% of the country's energy for cooling, heating, lighting and equipment operation. Any efficient climate protection strategy must take residential and commercial buildings, which are responsible for approximately 40% of U.S. carbon dioxide emissions, into account [1]. Since the energy consumption of buildings is significant, the prediction of energy use in buildings is important in order to improve their energy performance and reduce environmental impact. Furthermore, energy demand forecasting also helpa design and construct energy-saving buildings.

Energy consumption of a building is affected by internal factors such as primary use, number of floors and area of each floor, as well as external factors such as local weather (temperature, precipitation, wind speed, etc.). Such a complex situation makes it difficult to implement the prediction of building energy usage. So far, several prediction methods such as engineering models and artificial intelligence methods have been developed. The engineering methods, which use physical theories to characterize thermal dynamics and energy behaviors of buildings, have been well developed over the past fifty years. Al-Homoud reviewed a simplified engineering method called 'degree day' in which only one index, daily temperature, is analyzed. This method performs well for predicting the energy usage of small buildings [2]. Yik et al. used simulation tools to calculate cooling load profiles for different types of buildings [3]. Artificial Neural Networks (ANNs) have been studied for more than twenty years and are the most widely used artificial intelligence methods to estimate the energy use of buildings. This approach is effective for such complex application and solving nonlinear problems [4]. For instance, Kreider et al. studied a recurrent neural network on hourly energy consumption to predict energy needs for building heating and cooling using only the weather and timestamp datasets [5].

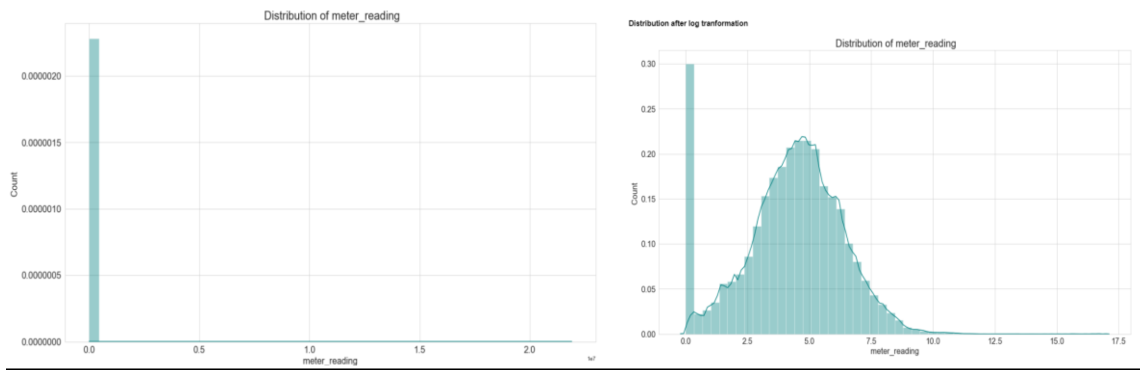
ASHRAE hosts a featured prediction competition named 'Great Energy Predictor III' [2]. This competition challenges participants to build counterfactual models across four energy types given historic meter reads and observed weather data. Here we propose a statistical method that carefully measure and select features then applies both linear and Gradient Boosting models to predict and analyze building energy consumption based on building meta data and weather conditions. A comparison of the predicted and actual energy usage indicated that the model can predict energy consumption within an acceptable error range. In the future, this model has the potential to be widely used to monitor energy consumption and measure energy savings for different kinds of buildings. Moreover, if additional data is available, this method will be more widely applicable to other sectors such as industrial facilities.

In conclusion, this report presents and evaluates the Linear and Gradient Boosting model we developed for estimating energy use in different U.S. buildings, which includes data cleaning, exploratory data analysis (EDA) and methods development. Previous research work using other statistical models and relevant applications are also introduced. Finally prediction results and further prospects are discussed for additional research reference.

2. Exploratory Data Analysis (EDA)

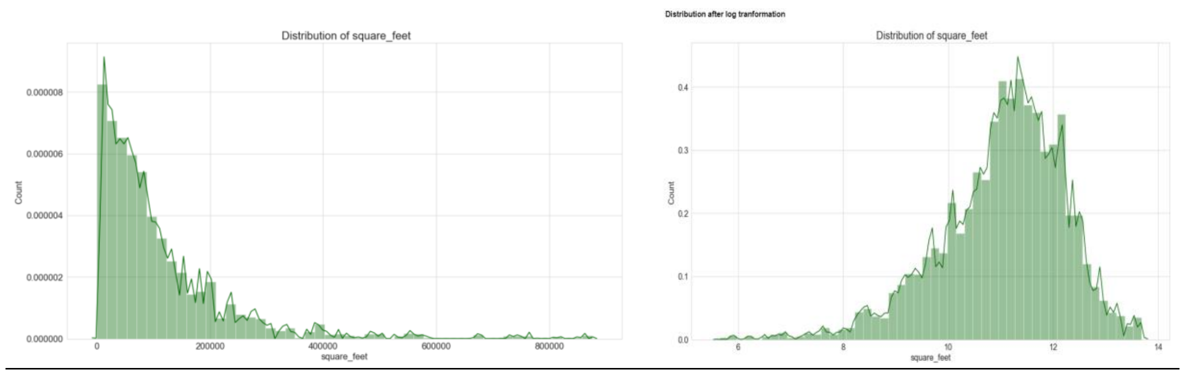
The original data was provided by American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE). Meter readings of four different types (electricity, natural gas, chilled water and hot water) were gathered from 1449 buildings and 16 sites. Weather data and building information were also gathered. The total time frame is one year and the measurement frequency is once per hour. There are 1688175 unique meter readings in total. Hence the data represents a sample from a larger population.

There are four different meter types: 0-electricity, 1-chilledwater, 2-steam, and 3-hotwater. Figure S1 indicates that electricity is the most frequent meter type measured. Both weather and building data have missing data points. For weather data, we impute the missing data points using the mean value of the day. For building data, the year of built and floor count have over 50% missing so we removed these two variables. Table S1 shows the variable description of the dataset after filling the missing data. After plotting the meter_reading data (Figure 1(a)), we found this variable is heavily skewed with a mean of 1.99×10^3 and a standard deviation of 1.53×10^5. After log transformation, the data seems to have a normal distribution (Figure 1(b)). Same applies to the square feet data (Figure 2).



(a) (b)

Figure 1: Distribution of meter_reading (a) before and (b) after log transformation.



(a) (b)

Figure 2: Distribution of square_feet (a) before and (b) after log transformation.

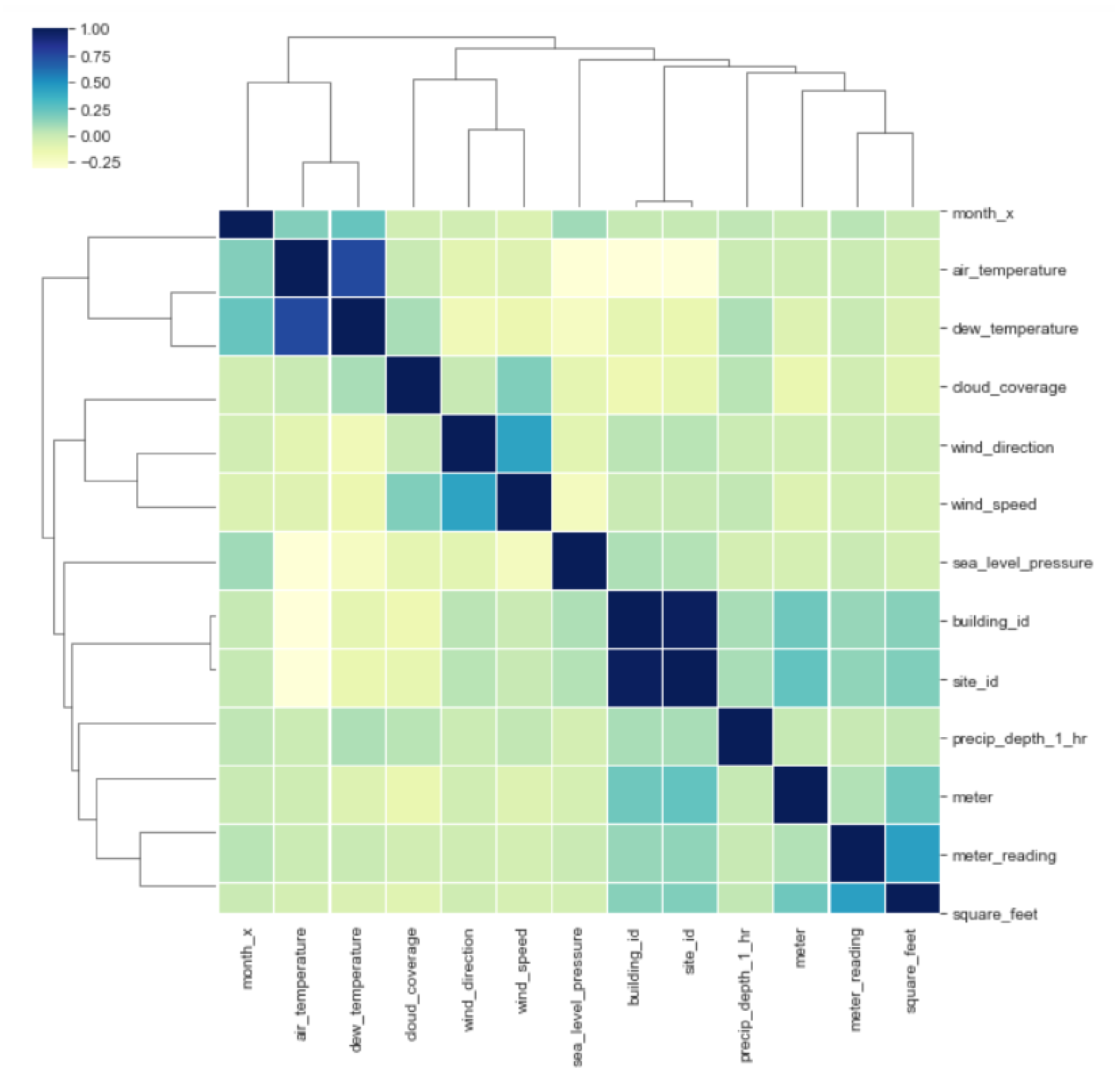


Figure 3: Correlation between meter_reading and numeric variables.

The correlation pattern is shown in Figure 3, which indicates that building_id and site_id represent the same variable, and air_temeprature is related to dew_temperature. There is also a strong relationship between wind_direction and wind_speed. Overall, square_feet, site_id, air_temperature and dew_temperature are most important variables for regression models that get built. The meter reading based on building type is shown in Figure 4. The Utility and Healthcare places tend to have the highest readings, while Religious Worship places the least. This is because Religious Worship buildings are used less often than Utility and Healthcare buildings. Figure 4 also indicates there are quite some meter_reading outliers for each type of building, which will be removed prior to running regression analysis.

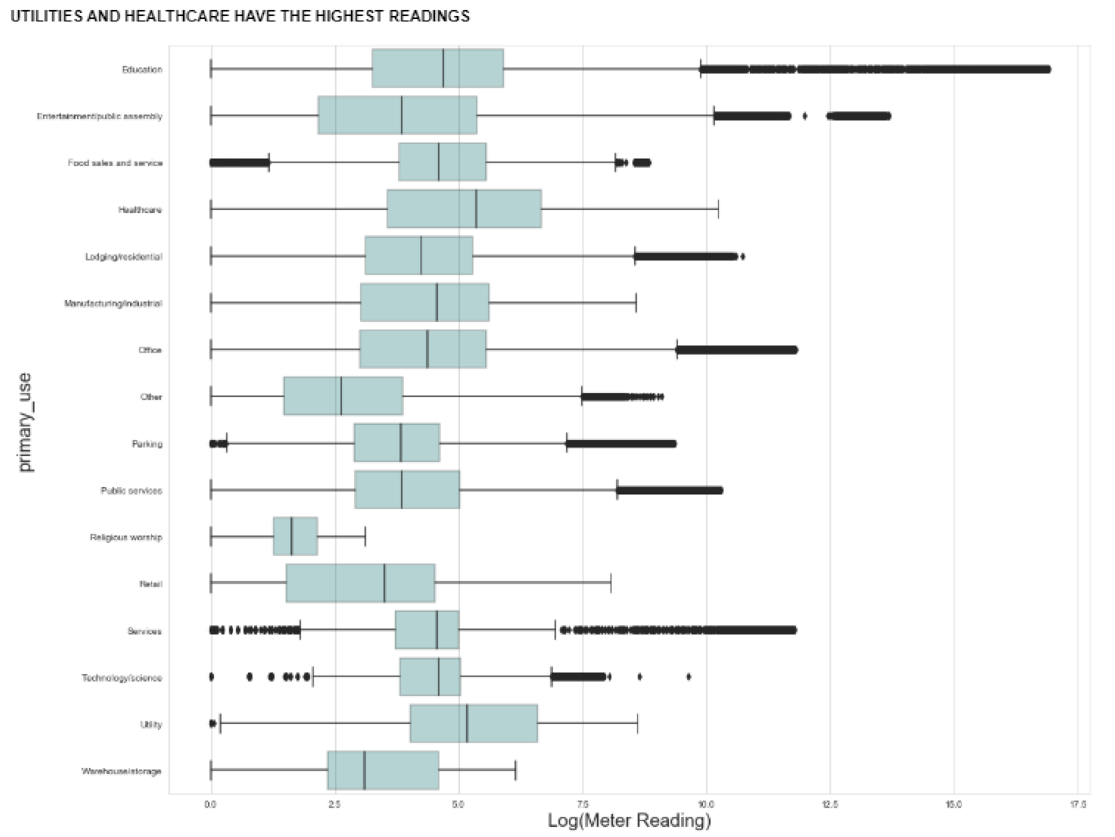


Figure 4: Meter_reading for different types of buildings.

Given the analysis on the data, we select the site id, primary use, square feet, month, air temperature, dew temperature, precop depth of a month, sea level pressure, wind direction, and wind speed to build our feature vector. We implement feature transformation like PCA or adding lag to make the model more efficient or specific. We also take the variance of the feature into consideration, it the variance of a feature is too large, we use normalization method or apply log function to make it a distribution similar to normal distribution.

Since features like the year built and floor count are not that important, we discard these features to make the training process faster. Though building id has a significant impact on the meter readings, it is a feature that is too specific. Including this feature in the training process may cause overfitting in the model delivered, which means, the model performed very well on this 1449 buildings but will make bad predictions on other buildings. We discard building id to make the model more general.

3. Methods

Meter reading prediction can be defined as a regression problem. Given observable variables (weather, building info, site id and time), we would like to build a model to predict the meter readings. To simplify the problem, we choose to focus on the electric meter (0) which is the most prevalent. For input, there are 7 weather variables, 2 building info variables, 1 site id variable and 1 time variable. The output is the predicted meter reading result. We evaluate two types of methods for this task: one is linear regression and the other is gradient descent boosting. For each model, in addition to using all variables as input, we also use PCA to reduce the dimension of weather features and compare the results.

The first multivariate method we use is Principal Component Analysis (PCA). To reduce the dimension of weather variables, we perform PCA on the 7 variables. We build the covariance matrix and correlation matrix using all the data points from different sites and perform a spectral decomposition of the covariance matrix. The largest 5 number of components are selected to keep 80% of the total variance.

For PCA, the assumptions are: a) there are no outliers. This assumption is justified because we already checked and removed the outliers during EDA. b) There are correlations between variables and the relationship between variables are linearly related. This assumption is valid as we can see that there are correlations between weather variables.

The second multivariate method we use is Linear Discriminant Analysis (LDA). We would like to check if weather features can be used to cluster different time frame or site ids. All 7 weather features are selected as the input. The goal is to check if we can separate different time frame or different sites using discriminant analysis. We calculate the within group variation and between group variation matrices and solve for the first 2 eigenvectors of the generalized eigendecomposition.

For LDA, the assumptions are: a) Homoscedasticity, different groups should have different covariance structure. This assumption is hard to check because we do not have true labels of different energy consumption groups. However, we can check the covariance structure of different building usage to get an idea of how good this assumption is. b) The discriminants that best separate different groups are assumed to be a linear combination of the original variables. For this assumption, we need to check the clustering result to see if the linear combination is valid.

Gradient boosting is based on the fact that we can find the fastest decreases for a cost function F in a small neighborhood ɾ. It is a machine learning technique for both regression and classification problems, and it is widely accepted by the industrial and academic department. Like other boosting models, Gradient Boosting is usually ensemble of weak prediction models (decision trees), which are generated by minimizing loss function.

For this specific case, we choose lightGBM optimization and l2_root with log function to build the loss function, which will be explained in the Result part. The features of lightGBM include leaf-wise growth strategy, histogram split search algorithm, and categorical feature support, which make LightGbm a gradient boosting framework efficient and accurate.

4. Results and Analysis

For all the models we develop, evaluation metric used for meter reading prediction is the Root Mean Squared Logarithmic Error (RMSLE), which is defined as:

$$\epsilon = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(log(p_i+1) - log(a_i+1))^2}$$

n is the total number of observations. $p_i$is the predicted meter reading. $a_i$is the ground truth meter reading. Smaller error means better prediction. The dataset has a dedicated testing set which will be used as the evaluation dataset.

4.1 PCA analysis of weather variables

First of all, we do the PCA analysis on the weather dataset, trying to find the correlation between the month, air temperature, cloud coverage, dew temperature, precipitation depth in one hour, sea level pressure wind direction and wind speed. We would like to reduce the number of variables in the weather feather and use the principal elements of weather to do further analysis. We will do the same analysis with the original data to check the results of our PCA analysis.

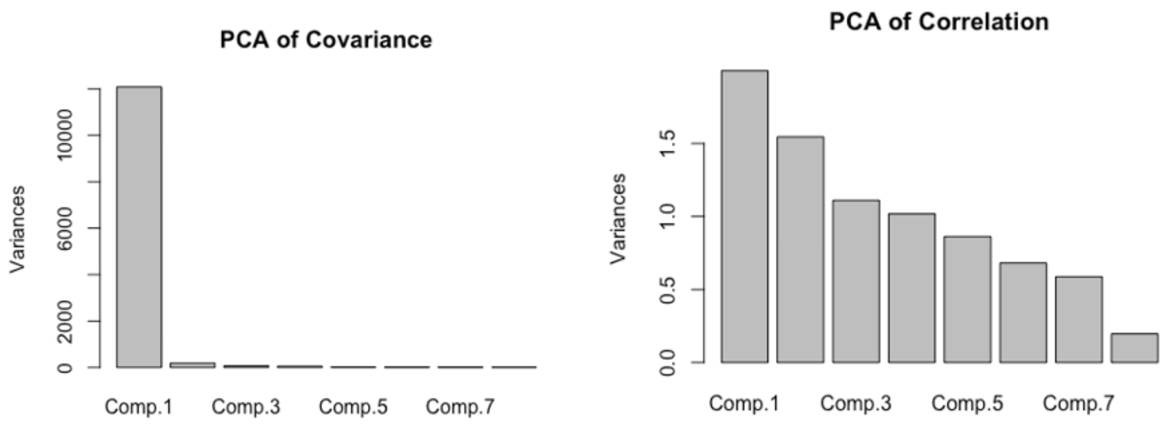
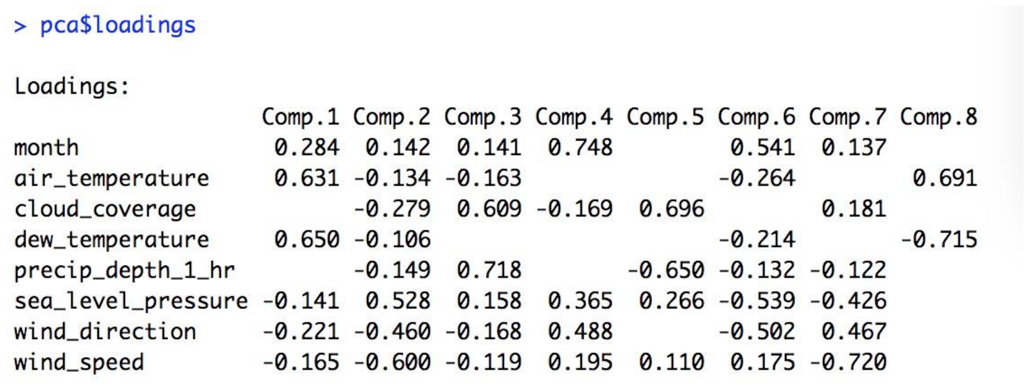


Figure 5: Scree plot of Principal Component Analysis

According to results of PCA using covariance matrix and correlation matrix (Figure 5) and numerical results in Figure S6, there is an obvious indication of scaling problem with our original data. Thus we choose to use the correlation matrix to do further analysis. Table 1 shows the result of our PCA.



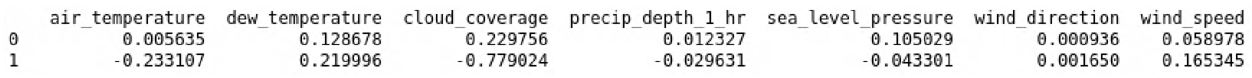
```
> pca$loadings

Loadings:
                   Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
month               0.284  0.142  0.141  0.748         0.541  0.137
air_temperature     0.631 -0.134 -0.163               -0.264         0.691
cloud_coverage            -0.279  0.609 -0.169  0.696         0.181
dew_temperature     0.650 -0.106                      -0.214        -0.715
precip_depth_1_hr         -0.149  0.718        -0.650 -0.132 -0.122
sea_level_pressure -0.141  0.528  0.158  0.365  0.266 -0.539 -0.426
wind_direction     -0.221 -0.460 -0.168  0.488        -0.502  0.467
wind_speed         -0.165 -0.600 -0.119  0.195  0.110  0.175 -0.720
```

Table 1: PCA loadings of weather variables

We choose 80% as the cut-off point. Thus according to the table above, we choose the first 5 components as our principal components. Combing the real-life meaning of these variables, we want to interpret the 5 principal components. 1) The first component is highly positively correlated to the air temperature and dew temperature. It also correlates positively to the month and negatively to the sea-level pressure, wind direction and wind speed. We can define it as the temperature component. 2) The second component is highly negatively correlated to the wind speed, sea-level pressure and wind direction. We define it as the atmosphere component. 3) The third component has the highest correlation to the precipitation depth in one hour and could coverage. It is clearly a precipitation component. 4) The fourth component is highly correlated to the month and the wind direction also plays an important role in it. As the wind direction depends on the month in one year, we define the fourth component as the date component. 5) The fifth component is a difference between cloud coverage and precipitation with some sea level and wind speed positive correlation, we say that the fifth component might be a thunder component.

4.2 LDA analysis of weather variables.

If we use weather features to cluster different sites, the first discriminant component is a combination of dew temperature, cloud coverage and sea level pressure. The second discriminant component is the difference between air temperature and dew temperature combined with cloud coverage. We can see that these two components can discriminate among different sites.



```
   air_temperature  dew_temperature  cloud_coverage  precip_depth_1_hr  sea_level_pressure  wind_direction  wind_speed
0         0.005635         0.128678        0.229756           0.012327            0.105029        0.000936    0.058978
1        -0.233107         0.219996       -0.779024          -0.029631           -0.043301        0.001650    0.165345
```

Table 2: LDA loadings of weather features with respect to sites

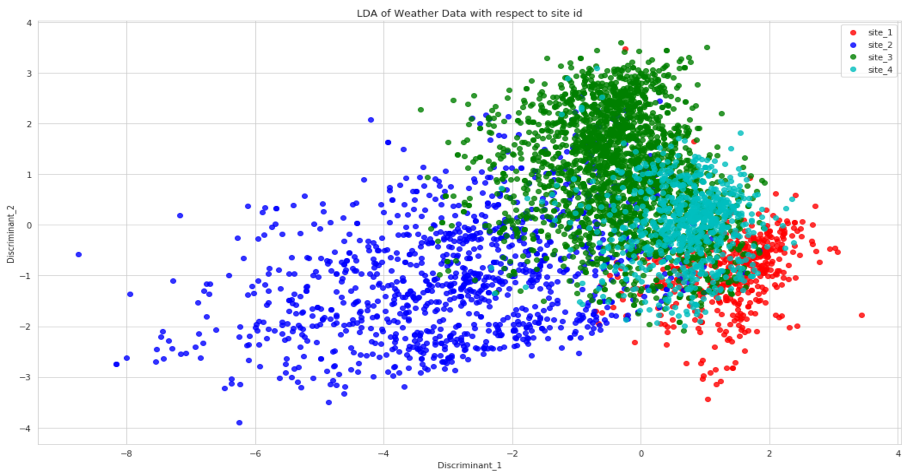
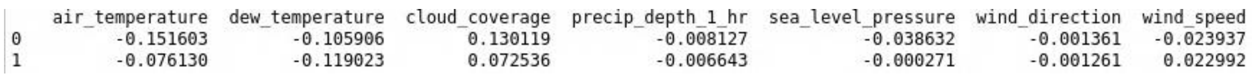


Figure 6: Discriminant analysis of weather features with respect to sites

If we use weather features to cluster different months, the first discriminant is the combination of temperature and cloud coverage. We can see that using the first discriminant alone can successfully discriminate between summer and winter months.



```
   air_temperature  dew_temperature  cloud_coverage  precip_depth_1_hr  sea_level_pressure  wind_direction  wind_speed
0        -0.151603        -0.105906        0.130119          -0.008127           -0.038632       -0.001361   -0.023937
1        -0.076130        -0.119023        0.072536          -0.006643           -0.000271       -0.001261    0.022992
```

Table 3: LDA loadings of weather features with respect to months

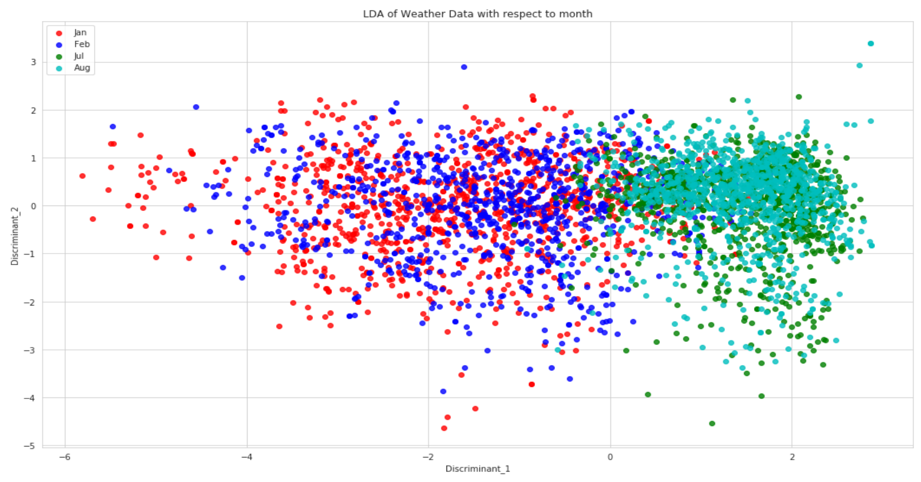
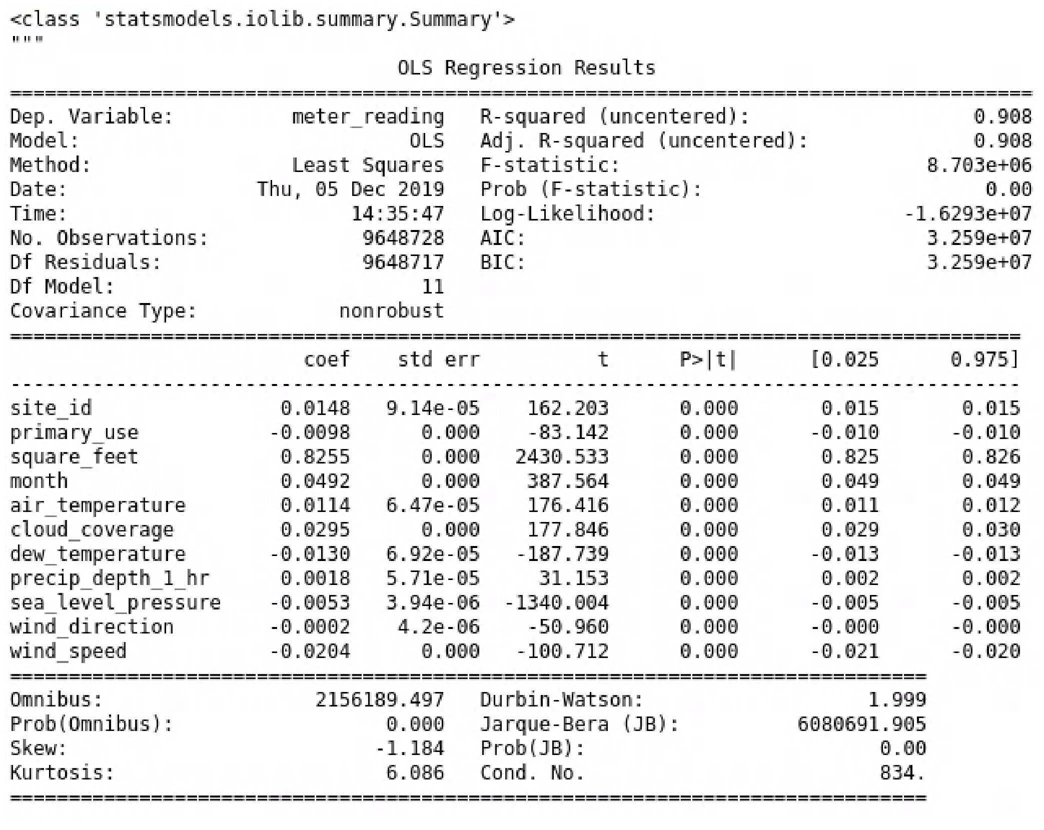


Figure 7: Discriminant analysis of weather features with respect to months

4.3 Prediction Results of Linear Regression Method

We first build a linear regression model as our prediction baseline. As discussed before, the input to our model are site_id, square_feet, primary_use, month, and weather variables. The results are shown in Figure 8. In addition, we change the weather variables to their first 5 principal components. The linear regression results are shown in supplementary Figure S5.



```
<class 'statsmodels.iolib.summary.Summary'>
"""
                                 OLS Regression Results
=======================================================================================
Dep. Variable:          meter_reading   R-squared (uncentered):                   0.908
Model:                            OLS   Adj. R-squared (uncentered):              0.908
Method:                 Least Squares   F-statistic:                          8.703e+06
Date:                Thu, 05 Dec 2019   Prob (F-statistic):                        0.00
Time:                        14:35:47   Log-Likelihood:                     -1.6293e+07
No. Observations:             9648728   AIC:                                  3.259e+07
Df Residuals:                 9648717   BIC:                                  3.259e+07
Df Model:                          11
Covariance Type:            nonrobust
======================================================================================
                        coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------------
site_id               0.0148   9.14e-05    162.203      0.000       0.015       0.015
primary_use          -0.0098      0.000    -83.142      0.000      -0.010      -0.010
square_feet           0.8255      0.000   2430.533      0.000       0.825       0.826
month                 0.0492      0.000    387.564      0.000       0.049       0.049
air_temperature       0.0114   6.47e-05    176.416      0.000       0.011       0.012
cloud_coverage        0.0295      0.000    177.846      0.000       0.029       0.030
dew_temperature      -0.0130   6.92e-05   -187.739      0.000      -0.013      -0.013
precip_depth_1_hr     0.0018   5.71e-05     31.153      0.000       0.002       0.002
sea_level_pressure   -0.0053   3.94e-06  -1340.004      0.000      -0.005      -0.005
wind_direction       -0.0002    4.2e-06    -50.960      0.000      -0.000      -0.000
wind_speed           -0.0204      0.000   -100.712      0.000      -0.021      -0.020
==============================================================================
Omnibus:                  2156189.497   Durbin-Watson:                   1.999
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          6080691.905
Skew:                          -1.184   Prob(JB):                         0.00
Kurtosis:                       6.086   Cond. No.                         834.
==============================================================================
```

Figure 8: Linear regression results using all the variables.

According to the table, we see that the P-value for all the variables are almost zero, which implies all of the variables are significant in our linear model. We have R-squared value of 0.988, which means that 98.8% of the variance in the response variable is predictable from the independent explanatory variables. The above features show the fitness of this model is good.

After log-transformation, the one unit increase of site id, primary use, square feet, month, air temperature, could coverage, dew temperature, precipitation in one hour, sea level pressure, wind direction and wind speed will cause 0.0148, -0.0098, 0.8225, 0.0492, 0.0114, 0.0295, -0.0130, 0.0018, -0.0053, -0.0002, -0.0204 unit increase of the log meter reading respectively.

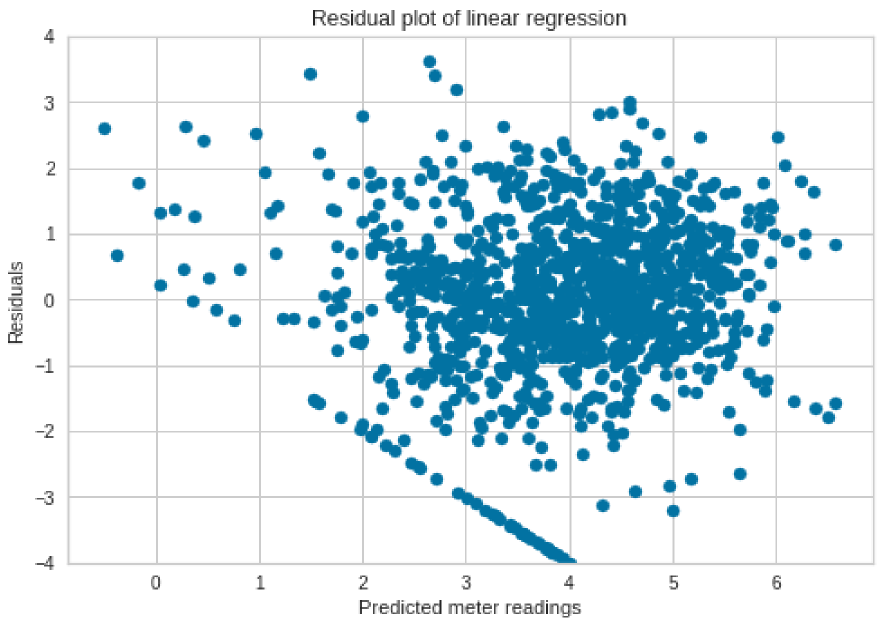


Figure 9: Residual plot of linear prediction (only a subset of data are plotted).

4.4 Prediction Results of LightGBM Method

We implemented 2 LightGBM models, one of which is trained with site_id, square_feet, primary_use, month, and weather variables, just like the linear model. Another model is trained with pca components instead of month and weather variables. From result from previous pca analysis, we use the first 5 components as the principle components which explain 80% of the weather parameters.

First we need to preprocess the training data, since it has na value and object type value. We fill the na values with functions like interpolating and drop some of them. Then since the meter readings and features like square feet have large variance, so we fit a log1p function to the data to get a distribution similar to normal distribution. Also, we encoded the category data like primary_use to 0-15

Training-wise, we train the model with 4-Folder validation to avoid overfitting, which means 0.25 of data is used as validation data. K-folder validation strategy use part of the data as validation dataset, if the training score keeps improving while validation score starts to slip back, we should stop training. The importance of training features is below.

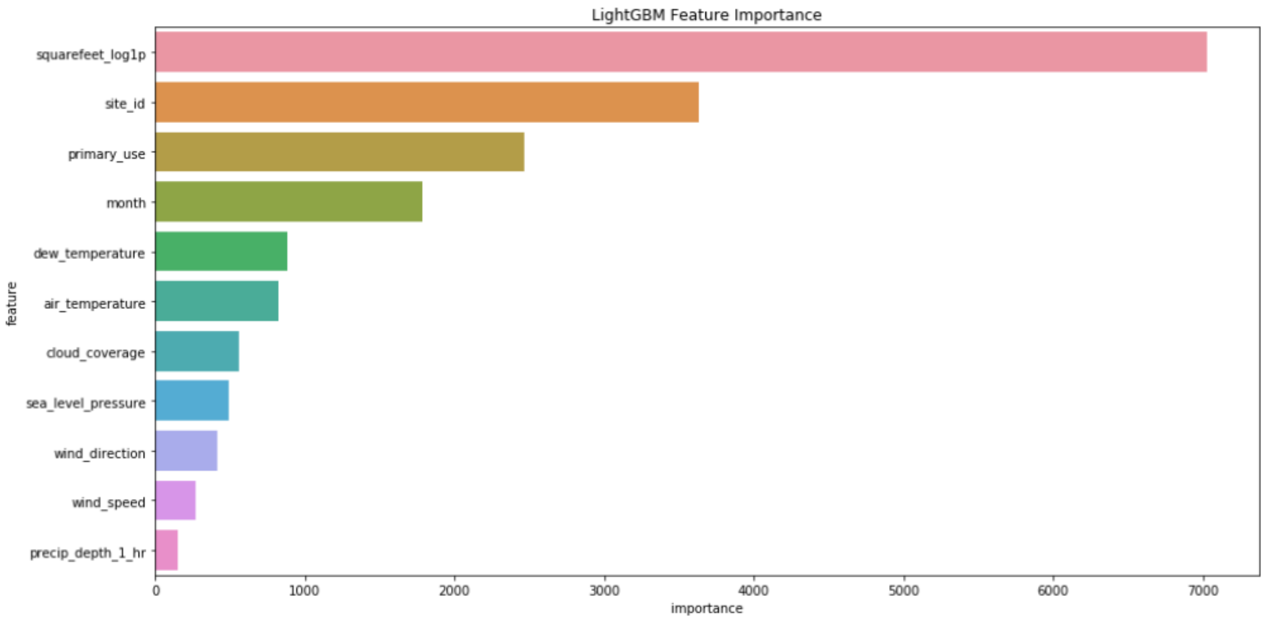
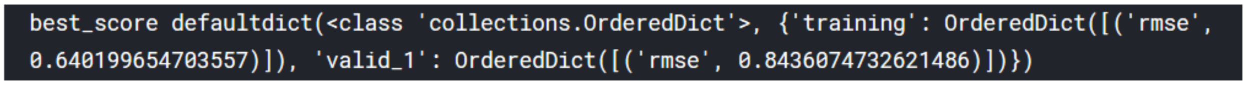


Figure 10: Importance of each feature (without PCA).

We stop training at the score is decreasing on the validation set. The best score for both training set and validation set is shown below. We have done the log transformation, so the result is actually RMSLE.



```
best_score defaultdict(<class 'collections.OrderedDict'>, {'training': OrderedDict([('rmse',
0.640199654703557)]), 'valid_1': OrderedDict([('rmse', 0.8436074732621486)])})
```

Figure 11: Training scores on training and validation dataset

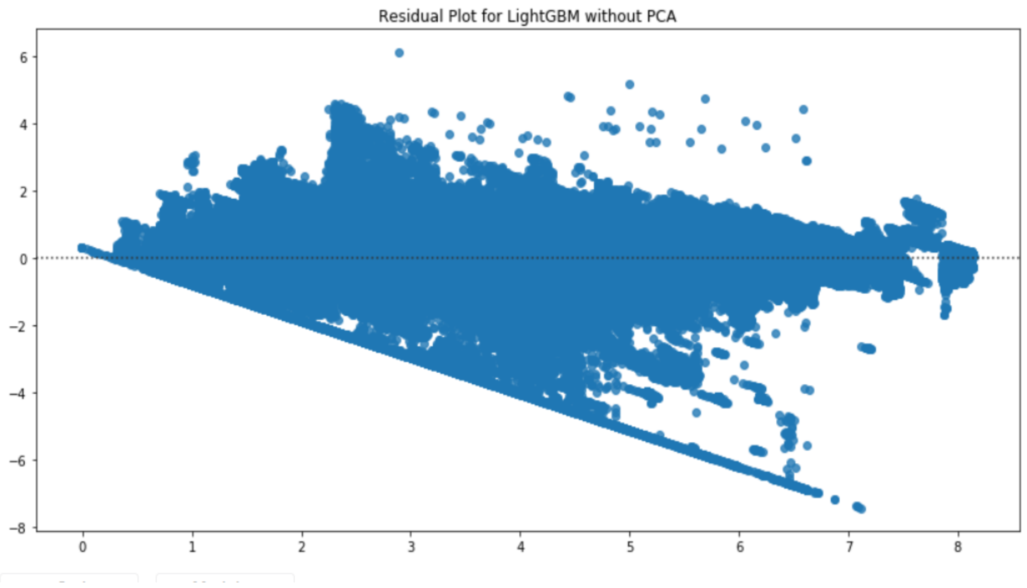


Figure 12: Residual Plot for LightGBM without PCA

Then we implement a LightGBM model with principal components. As described before, we also implement a model using the first 5 components as the principal components instead of all weather parameters. The importance of these components and other variables are below.

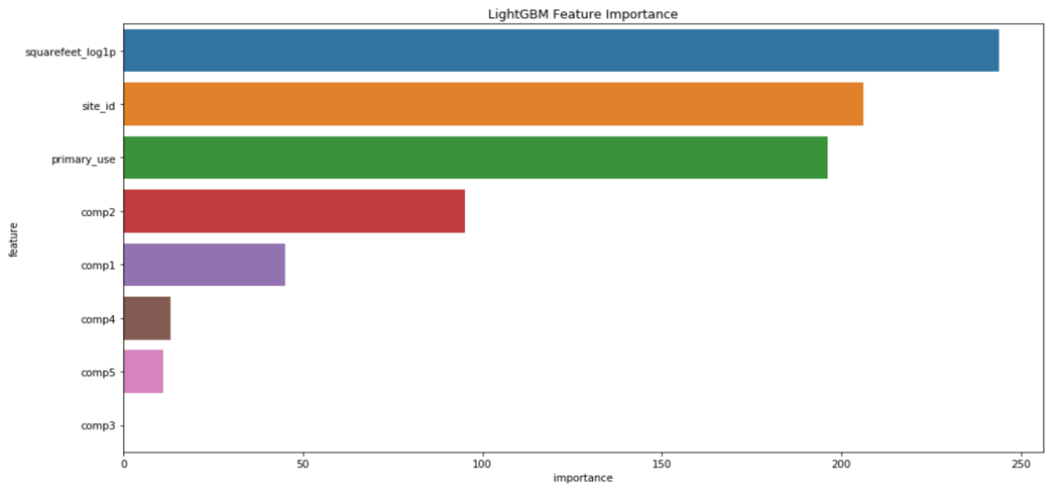
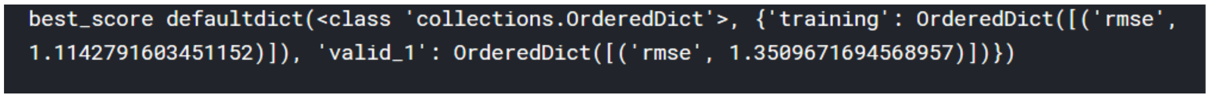


Figure 13: Importance of each feature (with PCA).

We stop training at the score is decreasing on the validation set. The best score for both training set and validation set is shown below. As mentioned, we have done the log transformation, so the score is actually RMSLE.



```
best_score defaultdict(<class 'collections.OrderedDict'>, {'training': OrderedDict([('rmse',
1.1142791603451152)]), 'valid_1': OrderedDict([('rmse', 1.3509671694568957)])})
```

Figure 14: Training scores on training and validation dataset

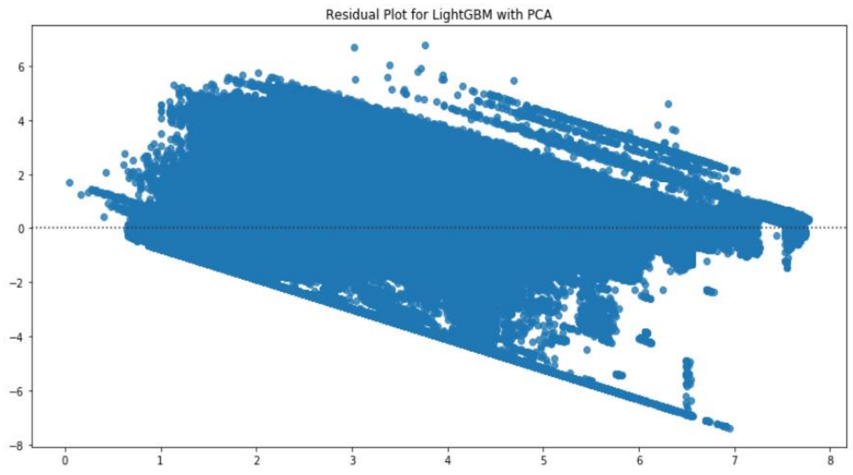
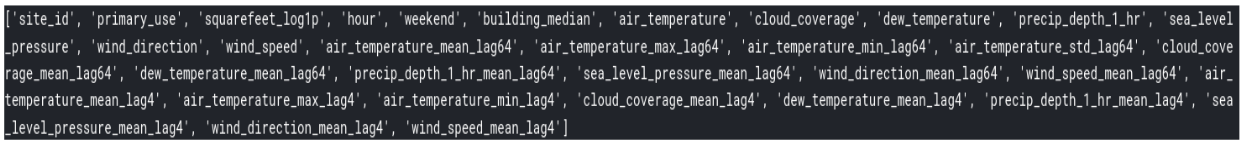


Figure15: Residual Plot for LightGBM with PCA

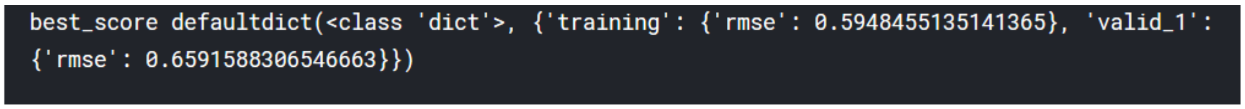
4.5 Improved Method

To reach the maximum accuracy possible and avoid overfitting at the same time, we use a specific way to deal with time series feature. Considering that in the different time of a day, human activities vary. And this is true for the weekends. We build different time features for accurate prediction. We did not include the specific date because we only have data of a whole year, including data will have the model overfit. We also includes lag features and window feature. The feature vector looks like below.



```
['site_id', 'primary_use', 'squarefeet_log1p', 'hour', 'weekend', 'building_median', 'air_temperature', 'cloud_coverage', 'dew_temperature', 'precip_depth_1_hr', 'sea_level
_pressure', 'wind_direction', 'wind_speed', 'air_temperature_mean_lag64', 'air_temperature_max_lag64', 'air_temperature_min_lag64', 'air_temperature_std_lag64', 'cloud_cove
rage_mean_lag64', 'dew_temperature_mean_lag64', 'precip_depth_1_hr_mean_lag64', 'sea_level_pressure_mean_lag64', 'wind_direction_mean_lag64', 'wind_speed_mean_lag64', 'air_
temperature_mean_lag4', 'air_temperature_max_lag4', 'air_temperature_min_lag4', 'cloud_coverage_mean_lag4', 'dew_temperature_mean_lag4', 'precip_depth_1_hr_mean_lag4', 'sea
_level_pressure_mean_lag4', 'wind_direction_mean_lag4', 'wind_speed_mean_lag4']
```

Figure 16: Feature vector with lag and window features.



```
best_score defaultdict(<class 'dict'>, {'training': {'rmse': 0.5948455135141365}, 'valid_1':
{'rmse': 0.6591588306546663}})
```

Figure 17: Training scores on training and validation dataset

The improved method achieved a RMSLE score of 0.66 on validation data set, making it the model with the smallest error.

4.6 Comparison of Accuracy of Different Models

The summary of prediction results of different methods is shown in Table 4. For linear regression, when training with all variables, the RMSLE is 1.38. When using principal components of weather variables, the RMSLE is 1.41. Training the Gradient Boosting model with all variables can reach a relatively low RMSLE on validation set, which is 0.84. Using principal components instead of weather variables can speed up the training process, but get a 1.35 error score, which indicates that the training result is kind of rough. Building a feature vector with generalized lag and window features, can deliver a more accurate result. But we only get a small improvement for the huge computational consumption.

Method	Linear Regression	Linear Regression + PCA	LightGBM	LightGBM + PCA	LightGBM + Additional Features
RMSLE	1.38	1.41	0.840	1.35	0.66

Table 4: Comparison of different methods in terms of prediction results

5. Conclusions

So what are the meaning of energy prediction? By comparing the prediction of a building's energy use and real readings of a specific one, we can get to the conclusion to questions like whether this building energy consumption is normal, and should the electrical system be replaced or not. By analyzing the correlation of readings and different parameters, we can get insights on how to distribute energy more efficiently in the future. With the prediction model, electrical company can make policy and balance electrical uses of a distinct easier or a company can plan its budget of a year more accurate with existing data.

In different cases we can implement different models. In general, training a model with extremely high accurate is time consuming, which requires data that are clean and not overlapped, training parameters that is carefully tweaked. For our problem, linear model and even Boosting models with PCA can only deliver a relatively rough result. We should choose from different training models and carefully manipulate the data set, to balance the training consumption and result accuracy. If an overall understanding of the relationship between readings buildings and weather, a linear model or a correlation plot may be enough, while more accurate models could support better market incentives and enable lower cost financing.

One caveat of the methods used is that the time series information is not considered. Variables at different time point are treated independently when there could be correlation along time. In addition, we did not train the model on all types of meter reading, and since different meter readings are related to different human activities, we may train the following models in a way that constructs different feature vectors for each one.

Reference

1. Environmental and Energy Study Institute. *Buildings & Built Infrastructure.* Retrieved from https://www.eesi.org/topics/built-infrastructure/description

Reference summary: This article states that buildings in the United States consume almost 40% of the country's energy and account for 40% of U.S. carbon dioxide emissions. This piece of information is used for providing background and references for this project in the introduction section.

2. Al-Homoud M.S. *Computer-aided building energy analysis techniques.* Building and Environment, 36 (4) (2001), pp. 421-433.

Reference summary: Al-Homoud reviewed a simplified engineering method called 'degree day' in which only one index, daily temperature, is analyzed. This method performs well for predicting energy usage of small buildings. This example is mentioned in introduction section as 'previous research work using engineering models and relevant applications'.

3. Yik F.W.H., Burnett J., Prescott I. *Predicting air-conditioning energy consumption of a group of buildings using different heat rejection methods.* Energy and Buildings, 33 (2) (2001), pp. 151-166.

Reference summary: Yik et al. used simulation tools to calculate cooling load profiles for different types of buildings. This example is mentioned in introduction section as 'previous research work using engineering models and relevant applications'.

4. Zhao H-X., Magoules F. *A review on the prediction of building energy consumption.* Renewable and Sustainable Energy Reviews, 16 (6) (2012), pp. 3586-3592.

Reference summary: This paper introduces the definition of Artificial Neural Networks (ANNs), which is mentioned as background and reference at the beginning of the report.

5. Kreider J.F., Claridge D.E., Curtiss P., Dodier R., Haberl J.S., Krarti M. *Building energy use prediction and system identification using recurrent neural networks.* Journal of Solar Energy Engineering, 117 (3) (1995), pp. 161-166.

Reference summary: Kreider et al. studied a recurrent neural network on hourly energy consumption to predict energy needs for building heating and cooling using only the weather and timestamp datasets. This example is mentioned in introduction section as 'previous research work using neural network models and relevant applications'.

6. Kaggle Inc. *ASHRAE-Great Energy Predictor III*. Retrieved from https://www.kaggle.com/c/ashrae-energy-prediction

Reference summary: This webpage introduces the featured prediction competition 'Great Energy Predictor III' in detail and it is the source of our project idea. The raw datasets we used can be found on 'Data' section.

Supplementary Material

Author Biography

Team leader: Liangyu Liu

i. Education: 2014-2018 B.S. in Mechanical Engineering at Shandong University
2018-2020 M.S. in Electrical and Computer Engineering at The Ohio State University
ii. Current affiliation: Electrical and Computer Engineering
iii. Research interests: Perception in Robotics

Team member 1: Zi Yang

i. Education: 2012-2015 B.S. in Materials Science and Engineering at The Ohio State University
2015-2020 PhD in Materials Science and Engineering at The Ohio State University
ii. Current affiliation: Materials Science and Engineering
iii. Research interests: Inorganic membranes used in water purification and gas separation, especially developing ceramic membranes for nanofiltration and ultrafiltration processes to remove dissolved salts and other species such as bacteria and proteins.

Team member 2: Ningyi Liu

i. Education: 2013-2015 B.A. in Arabic at Beijing Foreign Studies University
2015-2019 B.S. in Honor Mathematics & Actuarial Science at The Ohio State University
2019-2025 PhD in Statistics at The Ohio State University
ii. Current affiliation: Statistics

Team member 3: Yilin Zheng

i. Education: 2010-2014 B.A. in Electrical and Computer Engineering at Peking University
2014-2016 M.S. in Electrical and Computer Engineering at Purdue University
2016-2021 PhD in Electrical and Computer Engineering at The Ohio State University
ii. Current affiliation:Electrical and Computer Engineering
iii. Research interests: Machine learning methods in computer vision and graphic learning problems.

Additional Figures and Tables

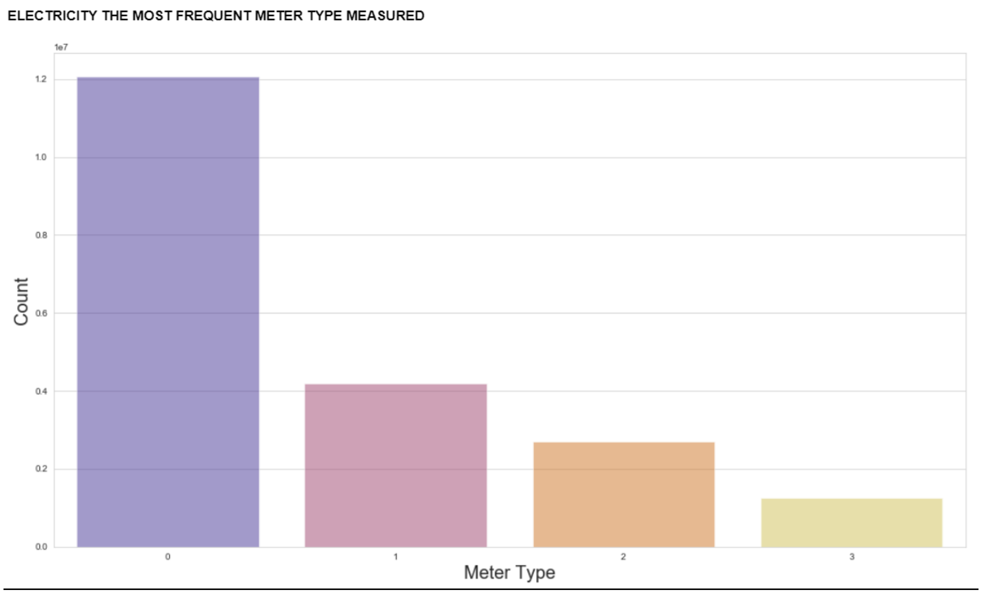
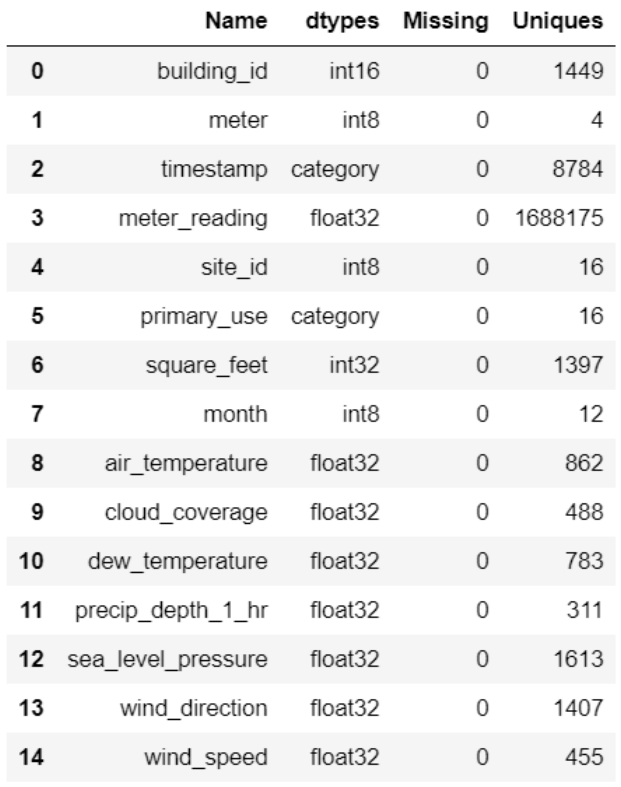


Figure S1: Count for each meter type: 0-electricity, 1-chilledwater, 2-steam, and 3-hotwater.



	Name	dtypes	Missing	Uniques
0	building_id	int16	0	1449
1	meter	int8	0	4
2	timestamp	category	0	8784
3	meter_reading	float32	0	1688175
4	site_id	int8	0	16
5	primary_use	category	0	16
6	square_feet	int32	0	1397
7	month	int8	0	12
8	air_temperature	float32	0	862
9	cloud_coverage	float32	0	488
10	dew_temperature	float32	0	783
11	precip_depth_1_hr	float32	0	311
12	sea_level_pressure	float32	0	1613
13	wind_direction	float32	0	1407
14	wind_speed	float32	0	455

Table S2: Variable description of the dataset

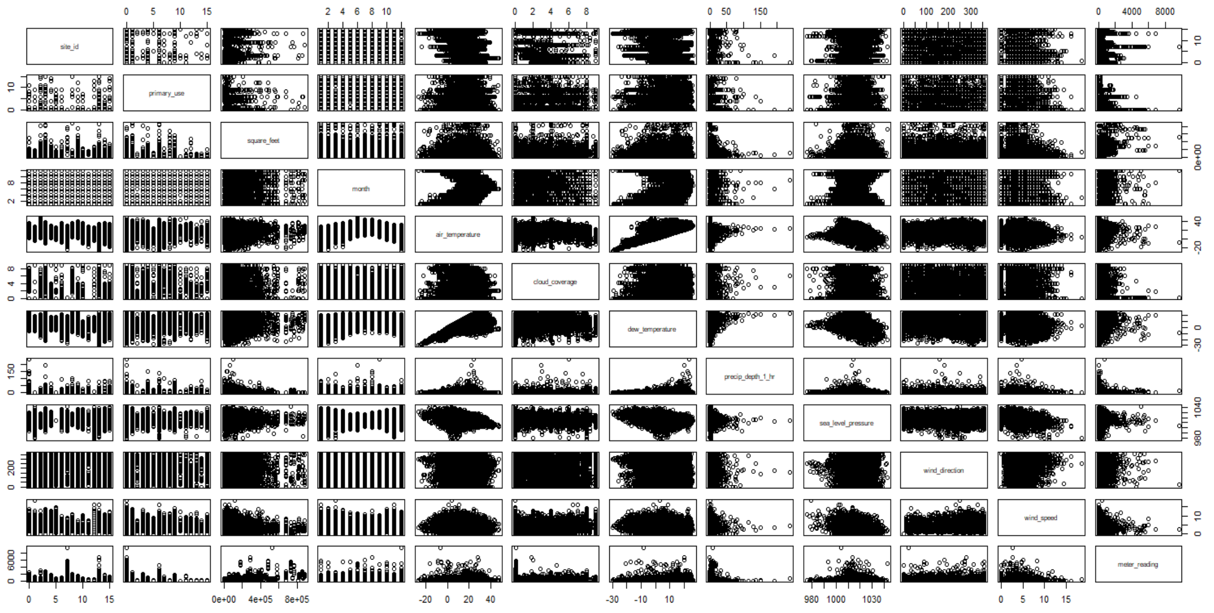


Figure S3: Scatter plots of all pairs of variables.

Figure S4 shows the plot of meter reading versus reading hour. Reading are higher during traditional work hours and this is to be expected. It seems the time of day may be a significant predictor in any subsequent model.

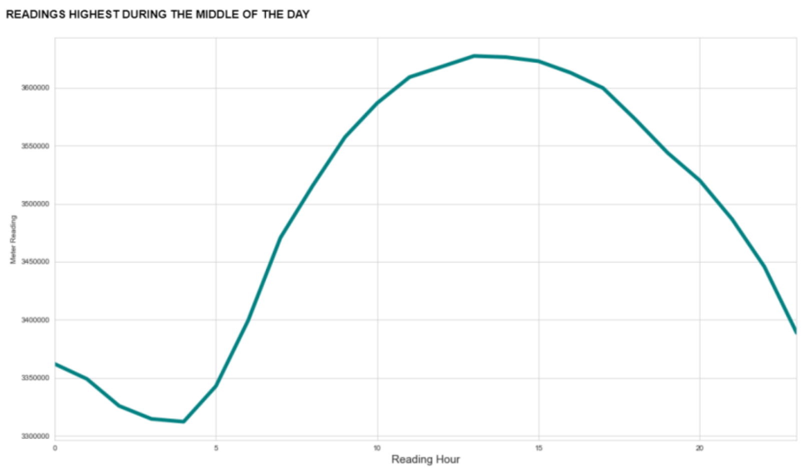
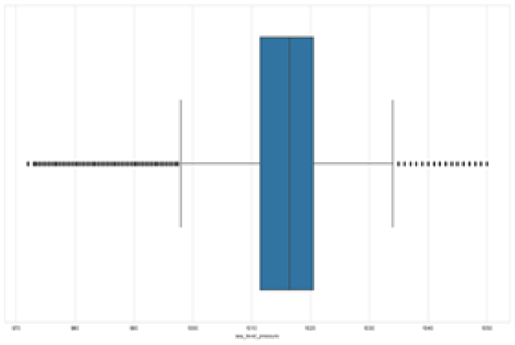
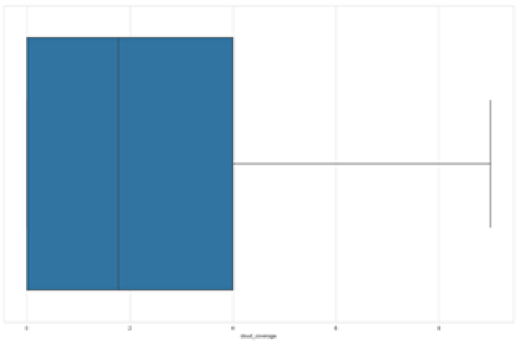


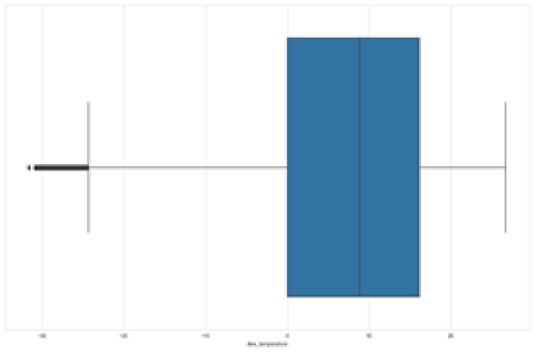
Figure S4: Hourly meter_reading during a day.



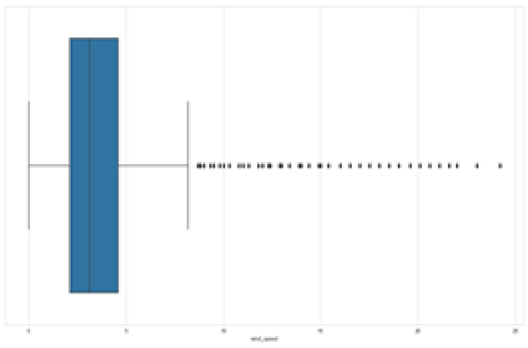
(a)



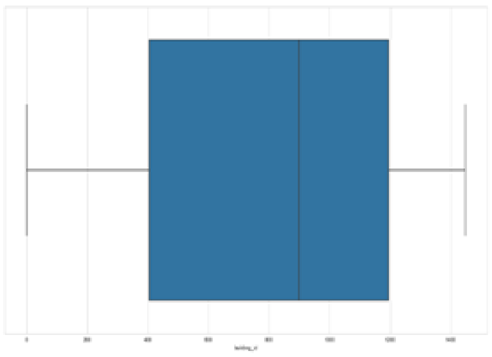
(b)



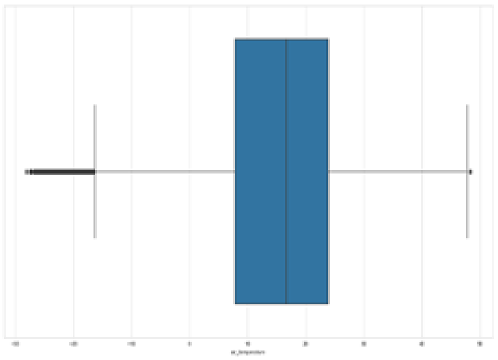
(c)



(d)

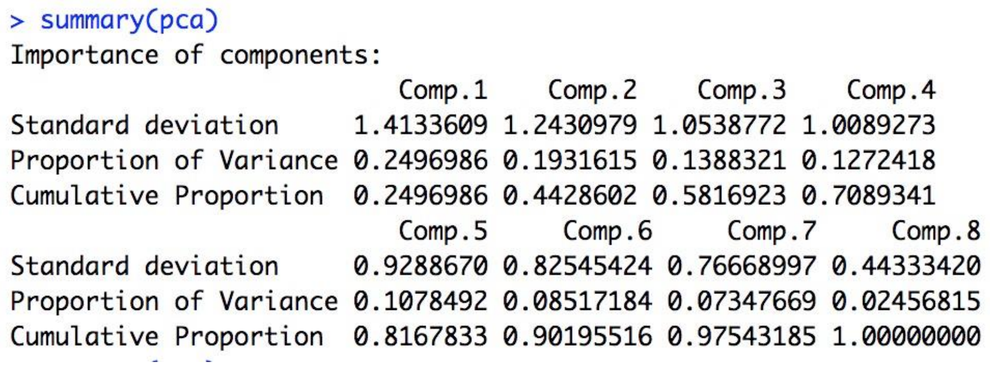


(e)



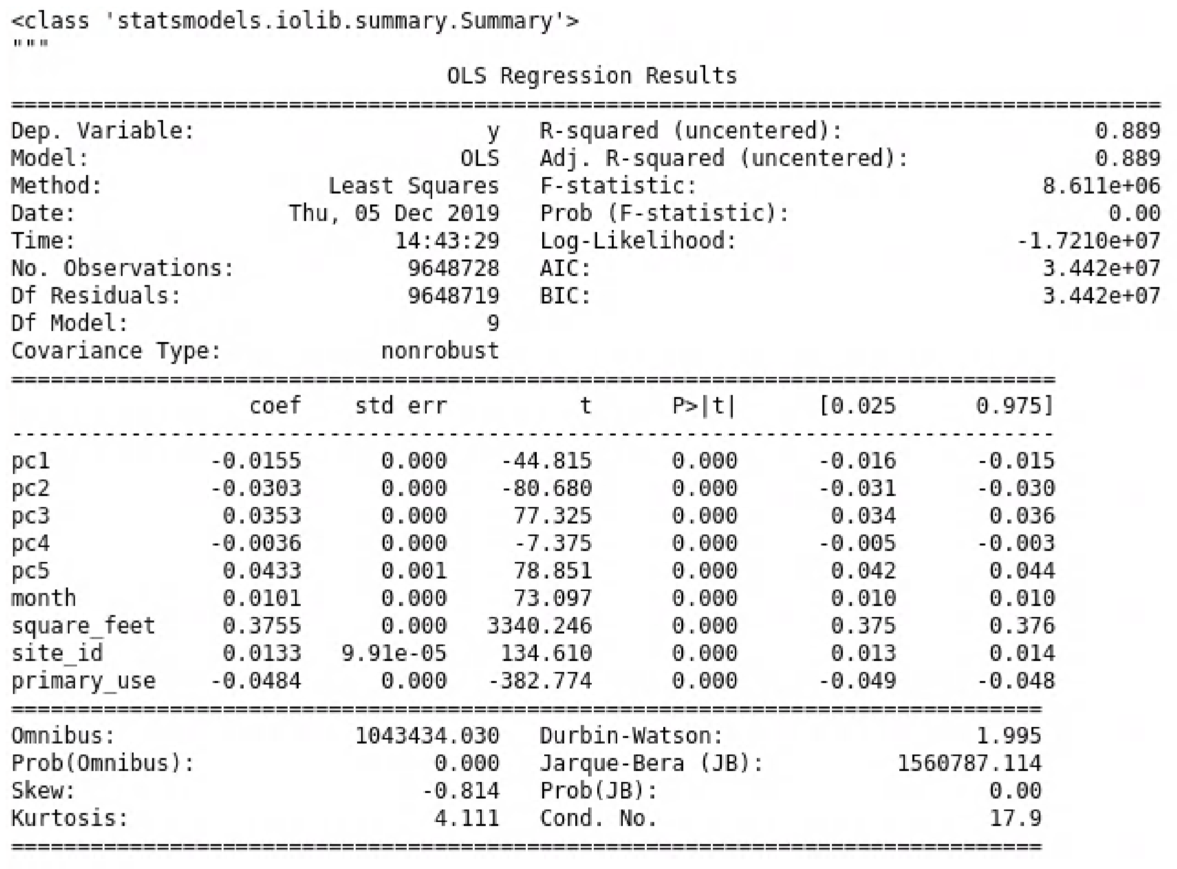
(f)

Figure S5: Boxplots for (a) sea_level_pressure (b) cloud_coverage (c) dew temperature (d) wind_speed (e) building_id (f) air_temperature.



```
> summary(pca)
Importance of components:
                          Comp.1    Comp.2    Comp.3    Comp.4
Standard deviation     1.4133609 1.2430979 1.0538772 1.0089273
Proportion of Variance 0.2496986 0.1931615 0.1388321 0.1272418
Cumulative Proportion  0.2496986 0.4428602 0.5816923 0.7089341
                          Comp.5     Comp.6     Comp.7     Comp.8
Standard deviation     0.9288670 0.82545424 0.76668997 0.44333420
Proportion of Variance 0.1078492 0.08517184 0.07347669 0.02456815
Cumulative Proportion  0.8167833 0.90195516 0.97543185 1.00000000
```

Table S2: Proportion of variance explained by different principal components of correlation matrix of weather variables



```
<class 'statsmodels.iolib.summary.Summary'>
"""
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                      y   R-squared (uncentered):                  0.889
Model:                            OLS   Adj. R-squared (uncentered):             0.889
Method:                 Least Squares   F-statistic:                         8.611e+06
Date:                Thu, 05 Dec 2019   Prob (F-statistic):                       0.00
Time:                        14:43:29   Log-Likelihood:                    -1.7210e+07
No. Observations:             9648728   AIC:                                 3.442e+07
Df Residuals:                 9648719   BIC:                                 3.442e+07
Df Model:                           9
Covariance Type:            nonrobust
===============================================================================
                  coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------
pc1            -0.0155      0.000    -44.815      0.000      -0.016      -0.015
pc2            -0.0303      0.000    -80.680      0.000      -0.031      -0.030
pc3             0.0353      0.000     77.325      0.000       0.034       0.036
pc4            -0.0036      0.000     -7.375      0.000      -0.005      -0.003
pc5             0.0433      0.001     78.851      0.000       0.042       0.044
month           0.0101      0.000     73.097      0.000       0.010       0.010
square_feet     0.3755      0.000   3340.246      0.000       0.375       0.376
site_id         0.0133   9.91e-05    134.610      0.000       0.013       0.014
primary_use    -0.0484      0.000   -382.774      0.000      -0.049      -0.048
==============================================================================
Omnibus:                  1043434.030   Durbin-Watson:                   1.995
Prob(Omnibus):                  0.000   Jarque-Bera (JB):          1560787.114
Skew:                          -0.814   Prob(JB):                         0.00
Kurtosis:                       4.111   Cond. No.                         17.9
==============================================================================
```

Table S3: Linear regression results using principal components of weather features.

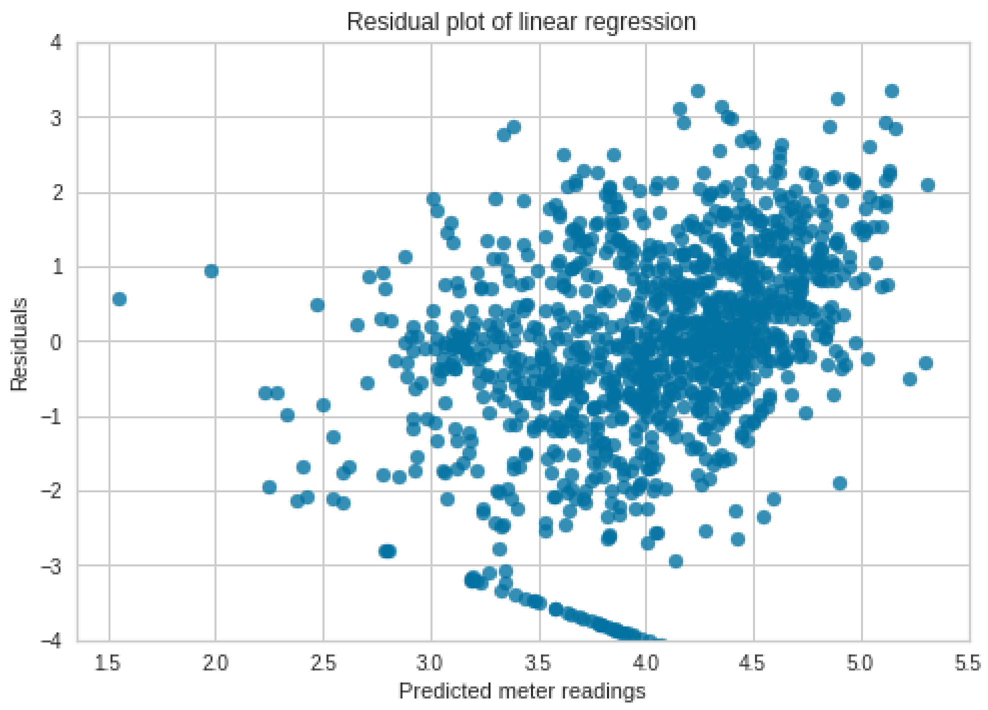


Figure S6: Residual plots of linear regression results using principal components of weather features.

Codes Used in the Report

Exploratory Data Analysis (EDA)

```
'''Importing Data Manipulattion Modules'''
import numpy as np
import pandas as pd
from scipy import stats
import os, gc

'''Seaborn and Matplotlib Visualization'''
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style("whitegrid")
%matplotlib inline

'''plotly Visualization'''
import plotly.offline as py
from plotly.offline import iplot, init_notebook_mode
import plotly.graph_objs as go
init_notebook_mode(connected = True)

'''Display markdown formatted output like bold, italic bold etc.'''
from IPython.display import Markdown
def bold(string):
    display(Markdown(string))
```

```
'''Variable Description'''
def description(df):
    summary = pd.DataFrame(df.dtypes,columns=['dtypes'])
    summary = summary.reset_index()
    summary['Name'] = summary['index']
    summary = summary[['Name','dtypes']]
    summary['Missing'] = df.isnull().sum().values
    summary['Uniques'] = df.nunique().values
    summary['First Value'] = df.iloc[0].values
    summary['Second Value'] = df.iloc[1].values
    summary['Third Value'] = df.iloc[2].values
    return summary
bold('**Variable Description of Data:**')
description(building6560)
```

```
sns.boxplot(x=building6560['variable name'])
```

```
'''Function to distribution plot'''
def distplot(variable, color):
    global ax
    font_size = 16
    title_size = 20
    plt.rcParams['figure.figsize'] = (18, 10)
    ax = sns.distplot(variable, color = color)
    plt.xlabel('%s' %variable.name, fontsize = font_size)
    plt.ylabel('Count ', fontsize = font_size)
    plt.xticks(fontsize = font_size)
    plt.yticks(fontsize = font_size)
    plt.title(' Distribution of '+'%s' %variable.name, fontsize = title_size)
    plt.show()

    '''Distribution of the Meter Reading'''
distplot(building6560['meter_reading'], 'teal')
```

```
'''Log tranformation of meter_reading'''
building6560['meter_reading'] = np.log1p(building6560['meter_reading'])

bold('**Distribution after log tranformation**')
distplot(building6560['meter_reading'], 'teal')
```

```
'''Distribution of the Meter Reading'''
distplot(building6560['square_feet'], 'darkgreen')
```

```
'''Log tranformation of meter_reading'''
building6560['square_feet'] = np.log1p(building6560['square_feet'])
bold('**Distribution after log tranformation**')
distplot(building6560['square_feet'], 'darkgreen')
```

```
bold('**READINGS HIGHEST DURING THE MIDDLE OF THE DAY**')
plt.rcParams['figure.figsize'] = (18,10)
temp_df = building6560.groupby('hour_x').meter_reading.sum()
temp_df.plot(linewidth = 5, color = 'teal')
plt.xlabel('Reading Hour', fontsize = 15)
plt.ylabel('Meter Reading')
plt.show()
```

```
bold('**UTILITIES AND HEALTHCARE HAVE THE HIGHEST READINGS**')
plt.rcParams['figure.figsize'] = (18, 15)
ax = sns.boxplot(data = building6560, y ='primary_use', x = 'meter_reading', color = 'teal', boxprops=dict(alpha=.3))
ax.set_xlabel('Log(Meter Reading)', fontsize = 20)
ax.set_ylabel('primary_use', fontsize = 20)
plt.show()
```

```
cg = sns.clustermap(building6560.corr(), cmap ="YlGnBu", linewidths = 0.1);
plt.setp(cg.ax_heatmap.yaxis.get_majorticklabels(), rotation = 0)
cg
```

```
bold('**ELECTRICITY THE MOST FREQUENT METER TYPE MEASURED**')
plt.rcParams['figure.figsize'] = (18, 10)
ax = sns.countplot(data = building6560, x ='meter', palette = 'CMRmap', alpha = 0.5)
ax.set_ylabel('Count', fontsize = 20)
ax.set_xlabel('Meter Type', fontsize = 20)
plt.show()
```

Data Preprocess

```
squarefeet_log1p = np.log1p(building_metadata['square_feet'])
squarefeet_log1p = pd.DataFrame(squarefeet_log1p)
squarefeet_log1p.columns=['squarefeet_log1p']
building_metadata=building_metadata.reset_index()
squarefeet_log1p=squarefeet_log1p.reset_index()
building_metadata=building_metadata.merge(squarefeet_log1p, how='left')
```

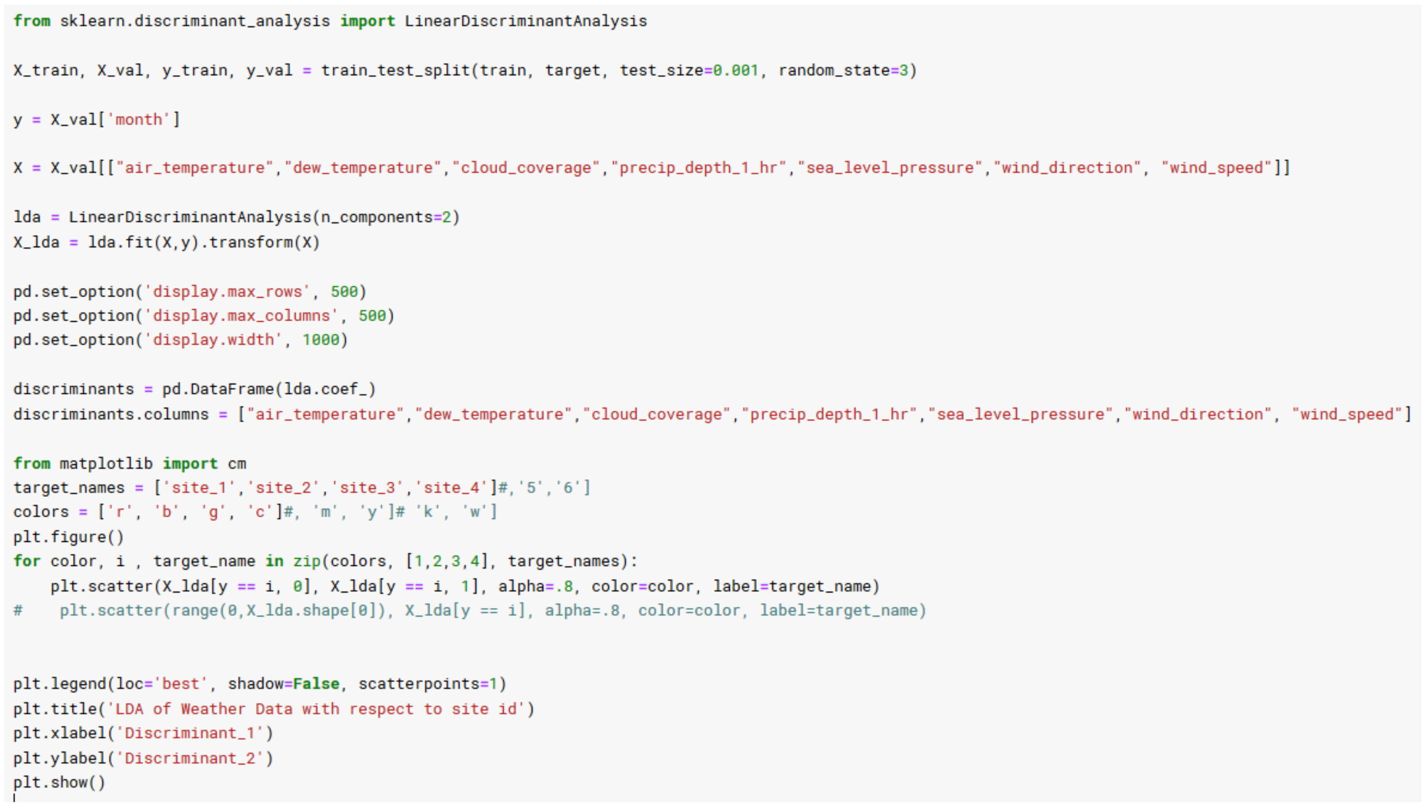
Linear Model

```
kf = KFold(n_splits=4)
models = []
for train_index, val_index in kf.split(train):
    train_features = train.iloc[train_index]
    train_target = target.iloc[train_index]

    val_features = train.iloc[val_index]
    val_target = target.iloc[val_index]

    pca = PCA(n_components = 5)
    X_train_pca = pca.fit_transform(train_features[weather_column])
    X_val_pca = pca.transform(val_features[weather_column])
    pca.explained_variance_ratio_
    pca.components_

    X_train_pca =  np.concatenate((X_train_pca,train_features[["month","square_feet","site_id","primary_use"]].values),axis=1)
    X_val_pca =  np.concatenate((X_val_pca,val_features[["month","square_feet","site_id","primary_use"]].values),axis=1)

    X_train_pca = pd.DataFrame(X_train_pca, columns = ['pc1','pc2','pc3','pc4','pc5','month','square_feet','site_id','primary_use'])
    X_val_pca = pd.DataFrame(X_val_pca, columns = ['pc1','pc2','pc3','pc4','pc5','month','square_feet','site_id','primary_use'])

    model = LinearRegression()
    model.fit(X_train_pca, train_target)
    models.append(model)
    val_pred = model.predict(X_val_pca)
    print(np.sqrt(mean_squared_error(val_target, val_pred)))
#     print(sklearn.metrics.r2_score(val_target,val_pred))
    del train_features, train_target, val_features, val_target
    gc.collect()
```

Residual plot

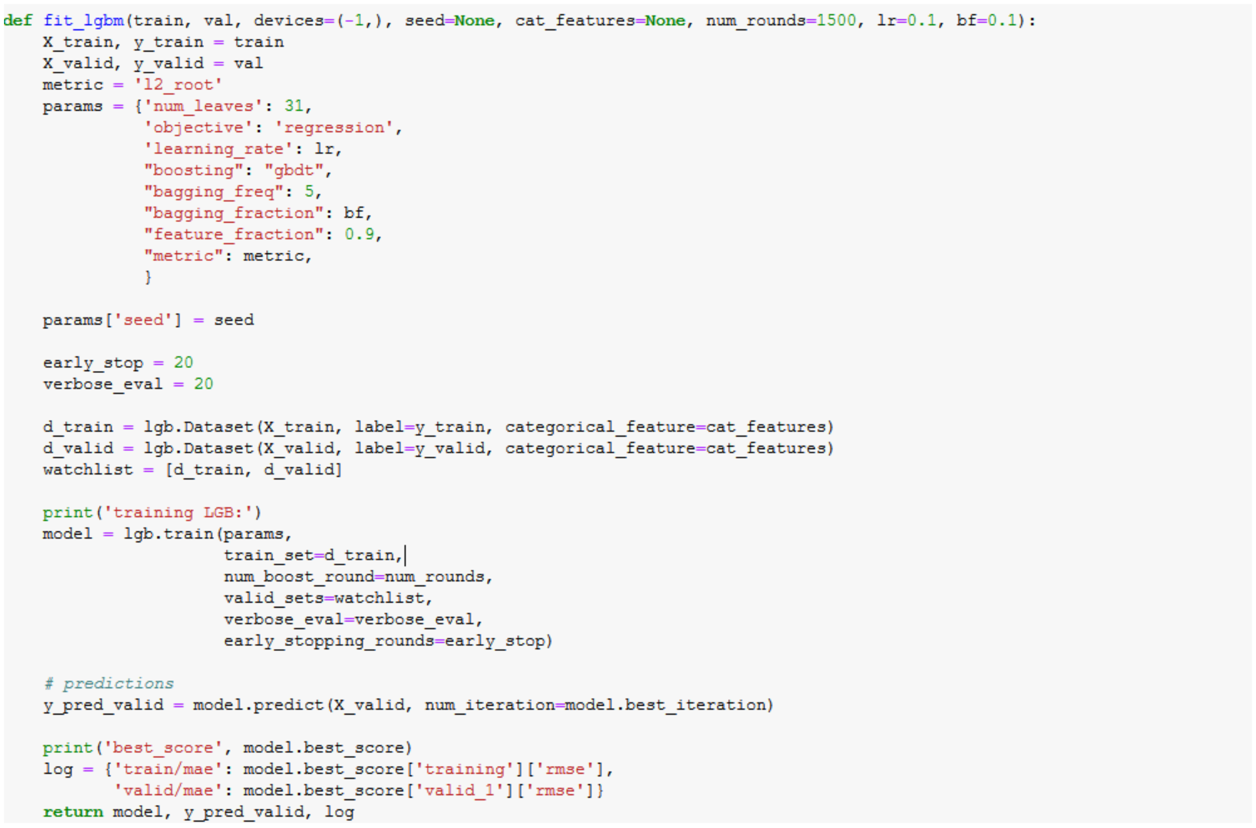
```
from matplotlib.pyplot import ylim
y_residual = y_val.values-y_pred
plt.scatter(y_pred,y_residual,alpha=0.8, color='b')
ylim(-4,4)
plt.title('Residual plot of linear regression')
plt.xlabel('Predicted meter readings')
plt.ylabel('Residuals')
plt.show()
```

Discriminant Analysis

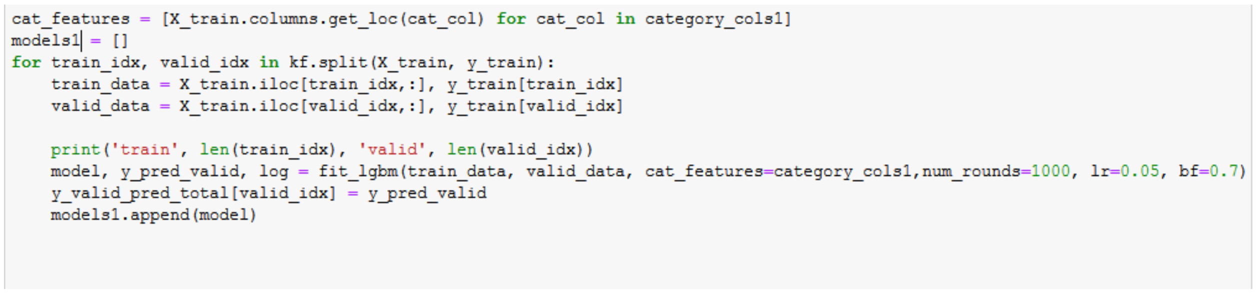


```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

X_train, X_val, y_train, y_val = train_test_split(train, target, test_size=0.001, random_state=3)

y = X_val['month']

X = X_val[["air_temperature","dew_temperature","cloud_coverage","precip_depth_1_hr","sea_level_pressure","wind_direction", "wind_speed"]]

lda = LinearDiscriminantAnalysis(n_components=2)
X_lda = lda.fit(X,y).transform(X)

pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)

discriminants = pd.DataFrame(lda.coef_)
discriminants.columns = ["air_temperature","dew_temperature","cloud_coverage","precip_depth_1_hr","sea_level_pressure","wind_direction", "wind_speed"]

from matplotlib import cm
target_names = ['site_1','site_2','site_3','site_4']#,'5','6']
colors = ['r', 'b', 'g', 'c']#, 'm', 'y']# 'k', 'w']
plt.figure()
for color, i , target_name in zip(colors, [1,2,3,4], target_names):
    plt.scatter(X_lda[y == i, 0], X_lda[y == i, 1], alpha=.8, color=color, label=target_name)
#    plt.scatter(range(0,X_lda.shape[0]), X_lda[y == i], alpha=.8, color=color, label=target_name)


plt.legend(loc='best', shadow=False, scatterpoints=1)
plt.title('LDA of Weather Data with respect to site id')
plt.xlabel('Discriminant_1')
plt.ylabel('Discriminant_2')
plt.show()
```

LightGBM Model

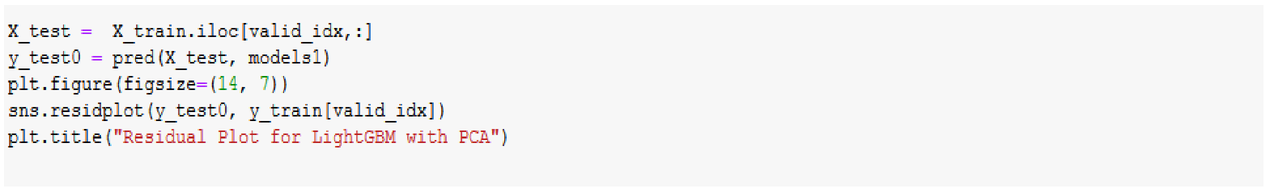


```
def fit_lgbm(train, val, devices=(-1,), seed=None, cat_features=None, num_rounds=1500, lr=0.1, bf=0.1):
    X_train, y_train = train
    X_valid, y_valid = val
    metric = 'l2_root'
    params = {'num_leaves': 31,
              'objective': 'regression',
              'learning_rate': lr,
              "boosting": "gbdt",
              "bagging_freq": 5,
              "bagging_fraction": bf,
              "feature_fraction": 0.9,
              "metric": metric,
              }

    params['seed'] = seed

    early_stop = 20
    verbose_eval = 20

    d_train = lgb.Dataset(X_train, label=y_train, categorical_feature=cat_features)
    d_valid = lgb.Dataset(X_valid, label=y_valid, categorical_feature=cat_features)
    watchlist = [d_train, d_valid]

    print('training LGB:')
    model = lgb.train(params,
                      train_set=d_train,
                      num_boost_round=num_rounds,
                      valid_sets=watchlist,
                      verbose_eval=verbose_eval,
                      early_stopping_rounds=early_stop)

    # predictions
    y_pred_valid = model.predict(X_valid, num_iteration=model.best_iteration)

    print('best_score', model.best_score)
    log = {'train/mae': model.best_score['training']['rmse'],
           'valid/mae': model.best_score['valid_1']['rmse']}
    return model, y_pred_valid, log
```



```
cat_features = [X_train.columns.get_loc(cat_col) for cat_col in category_cols1]
models1 = []
for train_idx, valid_idx in kf.split(X_train, y_train):
    train_data = X_train.iloc[train_idx,:], y_train[train_idx]
    valid_data = X_train.iloc[valid_idx,:], y_train[valid_idx]

    print('train', len(train_idx), 'valid', len(valid_idx))
    model, y_pred_valid, log = fit_lgbm(train_data, valid_data, cat_features=category_cols1,num_rounds=1000, lr=0.05, bf=0.7)
    y_valid_pred_total[valid_idx] = y_pred_valid
    models1.append(model)
```

Residual Plot



```
X_test =  X_train.iloc[valid_idx,:]
y_test0 = pred(X_test, models1)
plt.figure(figsize=(14, 7))
sns.residplot(y_test0, y_train[valid_idx])
plt.title("Residual Plot for LightGBM with PCA")
```