IoT home energy prediction

Data-driven modeling of Internet of Things

Steven Quitugua, Jake Setter, Steve Thudium

IoT home energy prediction

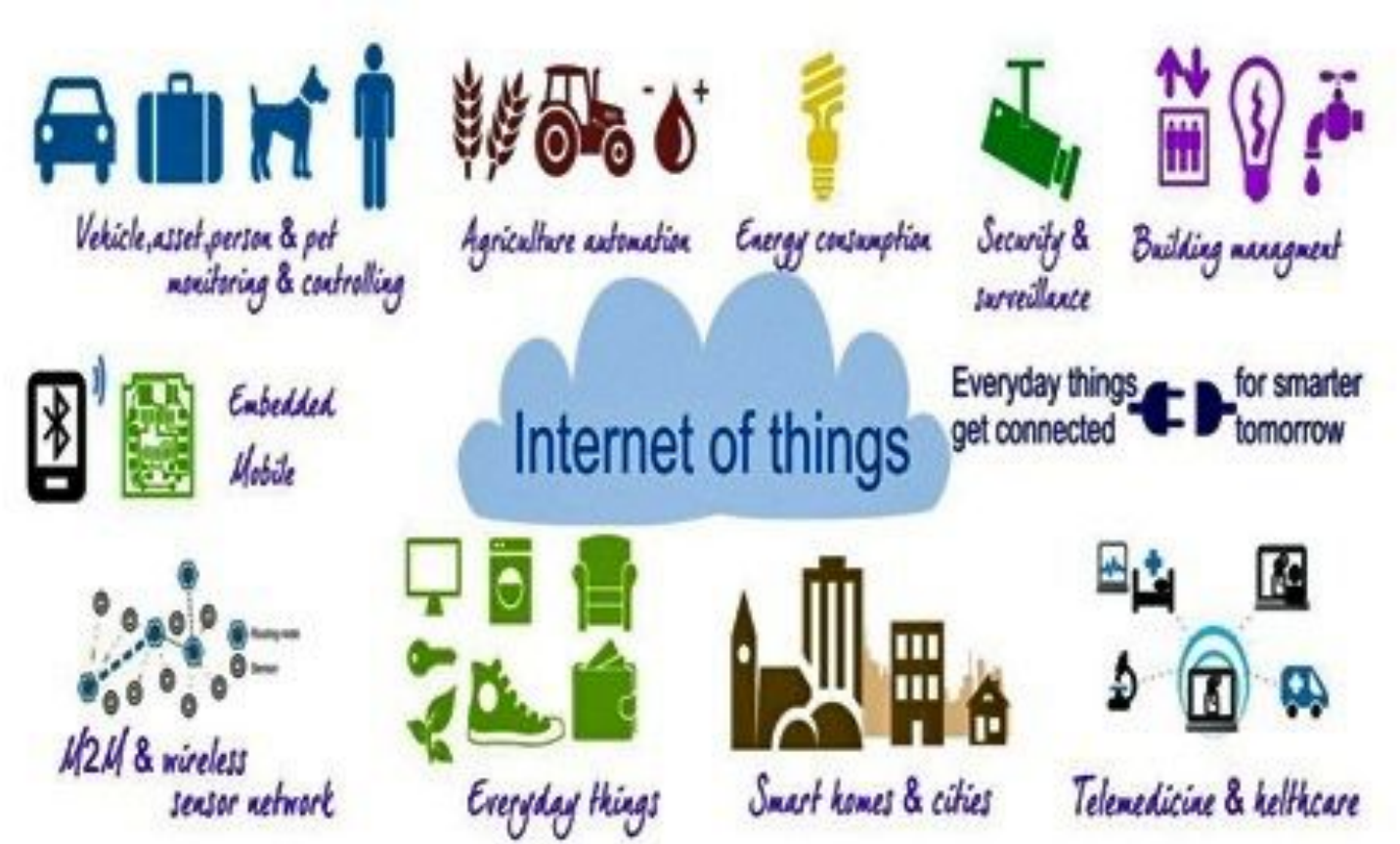
- Our project applies machine learning to home energy usage.
- Energy usage is predicted from data that is collected by sensors on an Internet of Things network

This presentation can be downloaded

URL = https://github.com/squitugua5821/Final_Project.git

This presentation can be downloaded

- This presentation will be available for download at that URL.



Autonomous electrical and mechanical devices sharing data on the Internet

- Connected appliances
- Smart home security systems
- Autonomous farming equipment
- Wearable health monitors
- Smart factory equipment
- Wireless inventory trackers
- Ultra–high speed wireless internet
- Biometric cybersecurity scanners
- Shipping container and logistics tracking

Internet of Things

- To quote Wikipedia, “The Internet of things is a system of interrelated computing devices, mechanical and digital machines having the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction.”
- Examples are connected appliances, home security, wireless inventory trackers, and autonomous farming equipment

Use case

- A home has an IoT network to collect sensor measurements, which include:
- Temperature and humidity readings from sensors in multiple rooms throughout the house
- Current weather at the local airport
- Measurements every 10 minutes
- Can this data be used to predict energy usage?
- If so, how … and how well?

Use case

- Our project is predicated on a use case.
- This use case is to predict energy usage for any home, using:
  - temperature and humidity sensors in multiple rooms
  - day of the week, hour of the day
  - local weather conditions
- Prediction comes from a regression model that is trained by historical data for the host home
- The model accounts for that particular home, with its unique characteristics, such as insulation and geographic location
- The model is train for the family behavior
- Periodically the model is retrained with the latest data to account for changing characteristics, like a new family member

Problem statement

- Analyze the available data for predictive value
- Train and evaluate multiple models with actual data.
- Select the linear or non-linear regression model that most accurately predicts home energy usage

Problem statement

- Identify a model that may be generalized for use with any home, using data that was recorded for a particular home
- Consider two possibilities -- either one model stands out above the rest, or all models are equal so any model will do.
- Analyze data characteristics -- maybe some features add no predictive value and don’t need to be used, or worse, reduce prediction accuracy
- Evaluate multiple models using that particular dataset
- Select the model that most accurately predicts home energy usage.
- This is a regression scenario, so the usual linear regressions models are used.
- Additionally, non-linear models are evaluated. Support vector regression uses a classification scheme. Random forest regressor uses decision trees.

Subject data

- IoT
- Energy usage, temperature, and humidity measurements; which were monitored with a ZigBee wireless sensor network
  - weather measurements from a public dataset for a nearby airport, Chievres Airport, Belgium, joined by date and time
  - measurement averages over every 10-minute interval during the first half of 2016
-

Dataset features

- **RH_1** – kitchen
- **T2** – living room
- **RH_2** – living room
- **T3** – laundry
- **RH_3** – laundry
- **T4** – office
- **RH_4** – office
- **T5** – bathroom
- **RH_5** – bathroom
- **T6** – outside
- **RH_6** –outside
- **T7** – ironing room
- **RH_7** – ironing room
- **T8** – teenager room
- **RH_8** -- teenager room
- **T9** – parent's room
- **RH_9** – parent's room

Chievres airport weather station

Dataset features

- Our dataset has features that fall under specific categories
  - Each row includes the data for a 10-minute interval, which is identified by a date and time
  - The total appliance energy usage during that 10-minute interval
  - The total lighting energy usage during that 10-minute interval
  - The temperature and relative humidity in several rooms throughout the house and the outside
  - The temperature, barometric pressure, relative humidity, wind speed, visibility, and dew point at the local airport

Prediction targets

Appliance -- total appliance energy usage during a 10-minute interval

Lights -- total lighting energy usage during a 10-minute interval

Prediction targets

- We want to emphasize that the variables that will be predicted are the appliance and lighting energy usage for a given 10-minute interval

Workflow

Left to right and down

- Import the dataset
- Encode features and impute missing data
- Evaluate and create new features
- Explore and analyze data
- Visualize energy usage
- Cull features & add dummy variables
- Run Recursive Feature Elimination
- Identify positive - negative correlation features
- Evaluate models

Workflow

- Our workflow is as shown. The workflow starts with importing the dataset and ends with evaluating the models.
- Each step is further developed in the following slides.
- Many of the steps are evaluating and analyzing the data before the models are built and evaluated.
- Read the steps
- Six models are evaluated with different feature subsets and scalars

Import the dataset

- The dataset includes data for a particular home.
- The dataset comes from the “UCI Machine Learning Repository.” The specific repository is the “Appliances energy prediction Data Set.”
- The data is directly downloaded into the Python notebook from https://archive.ics.uci.edu/ml/machine-learning-databases/00374/energydata_complete.csv
- The dataset is also kept in reserve locally as “./energydata_complete.csv”

Import the dataset

- We downloaded our dataset from the UC Irvine Machine Learning Repository

Encode features and impute missing data

- Encode date string
  - The date comes as a string, like “1/11/2016 5:00:00 PM.“
  - The date string is converted to the datetime format, with any seconds removed.
  - ”2016-01-11 17:00”
  - A datetime object simplifies parsing
- Rename features with meaning
  - Meaningful names mean meaningful visualizations
  - E.g., change “T1” to “T_kitchen”; “RH_1” to “RH_kitchen”
- Delete duplicate records
- No imputation is needed. Null value counts and row counts viz-a-viz the time range indicate no data is missing

Encode features and impute missing data

- We changed the date to a datetime object type, which is easier to work with
- We renamed features to give them meaningful names
- We deleted duplicate records
- We did not need to impute data to account for missing data. No dataset cells were empty or NA. And the number of dataset records is what we would expect for the date-time range.

Evaluate and create new features

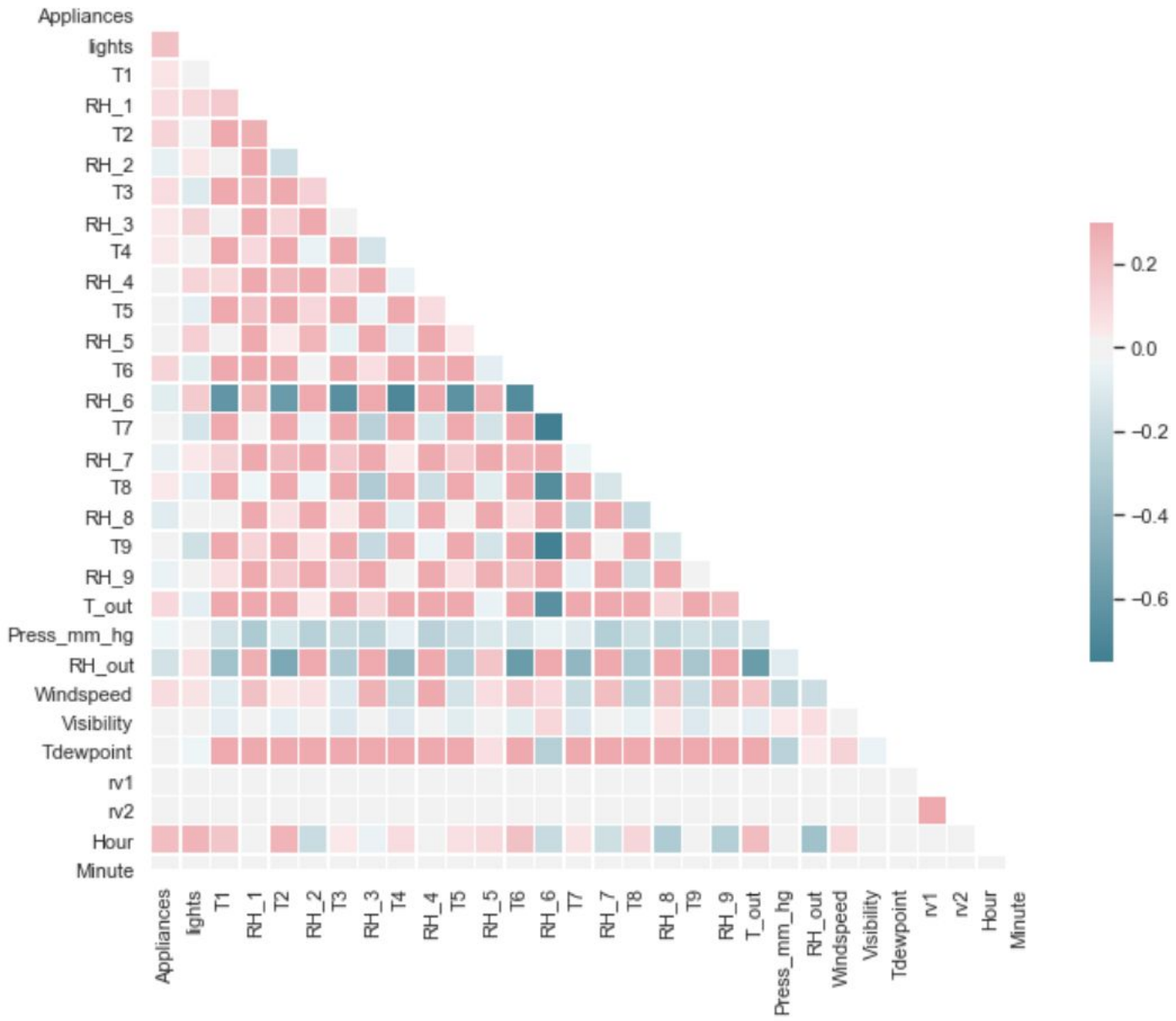
- Derive the following features from the datetime object
  - Day
  - Day_of_week
  - Time
  - Hour
  - NSM - number of seconds since last midnight
- Sort by Day and Time

Evaluate and create new features

- We created new features to facilitate and support our anticipated analysis and models.
- We derived several features from the date which support manipulating the date. For example, our energy heatmap focuses on the hour of the day, so we derive hour from date.
- We think that the relevant feature of a calendar date -- for use in our models -- is the day of the week, and so we added that new feature.
- The NSM is the number of seconds since midnight. This supports our models because the complex time is represented by simple format

Exploratory data analysis 1

- Correlation matrix heatmap – 6-month aggregate.
- Appliances and lights are weakly correlated with all features
- Features are weakly correlated, except for the outside humidity, RH_6, and temperature
- Dark red is weak positive correlation (+0.2)
- Dark blue is strong negative correlation (-0.7)

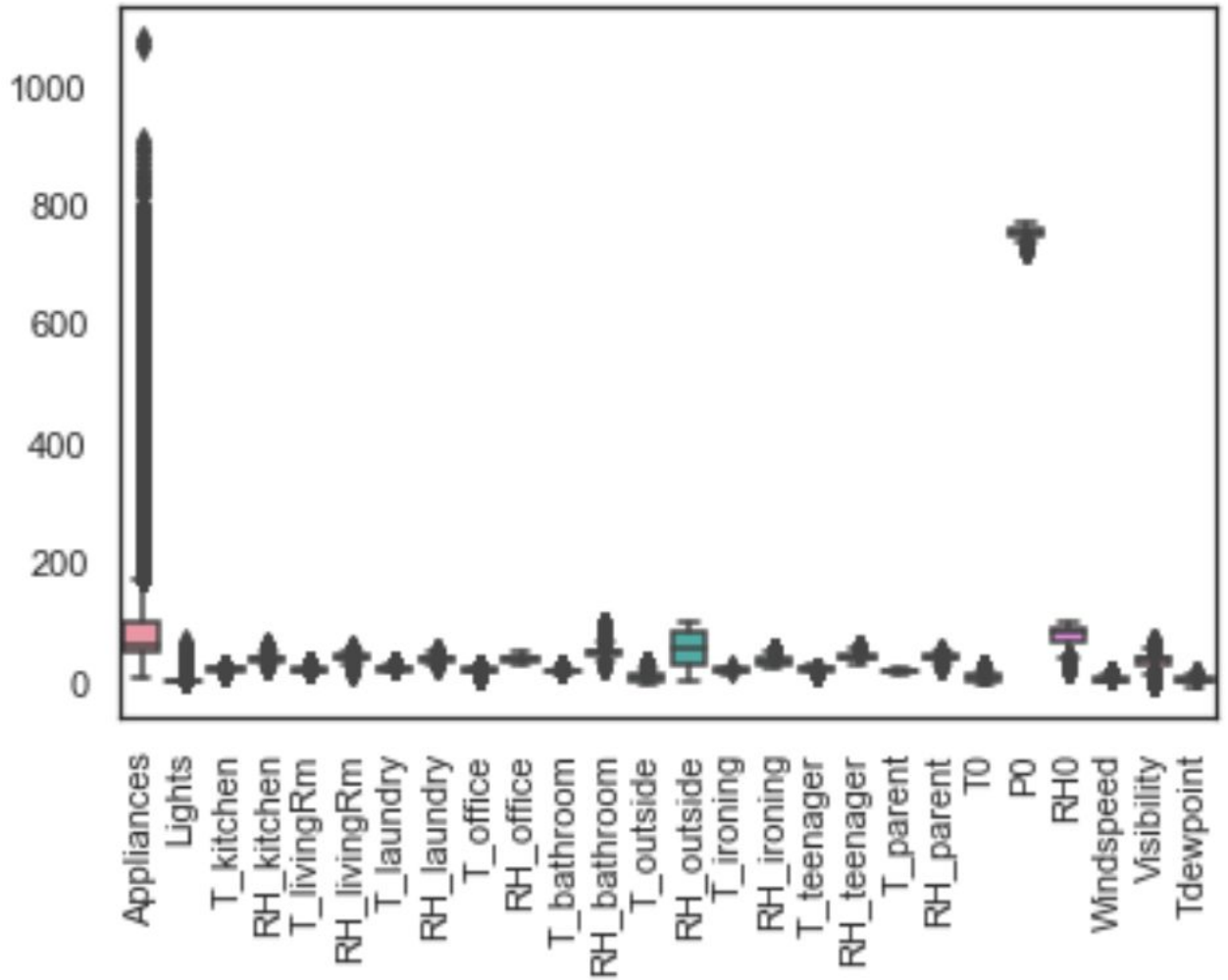


Exploratory data analysis 1

- We have 8 slides for visualizing the data and its properties.
- The first is a correlation map that represents the positive and negative correlation of each feature with all other features and the targets. Positive correlation is represented by red squares and negative with blue.
- In general, features are weakly correlated, including with the targets. The exception is the outside humidity, which is strongly and negatively correlated with inside temperatures and strongly and positively correlated with inside humidity. We thought that this feature could be used to replace those other features in our models, but this was proven wrong in subsequent analysis.

Exploratory data analysis 2

- The range of values for Appliances is wide and highly variable
- The range of values for Lights is not wide, but is highly variable
- The range of all other features is relatively narrow
- The mean of airport pressure (P0) is large

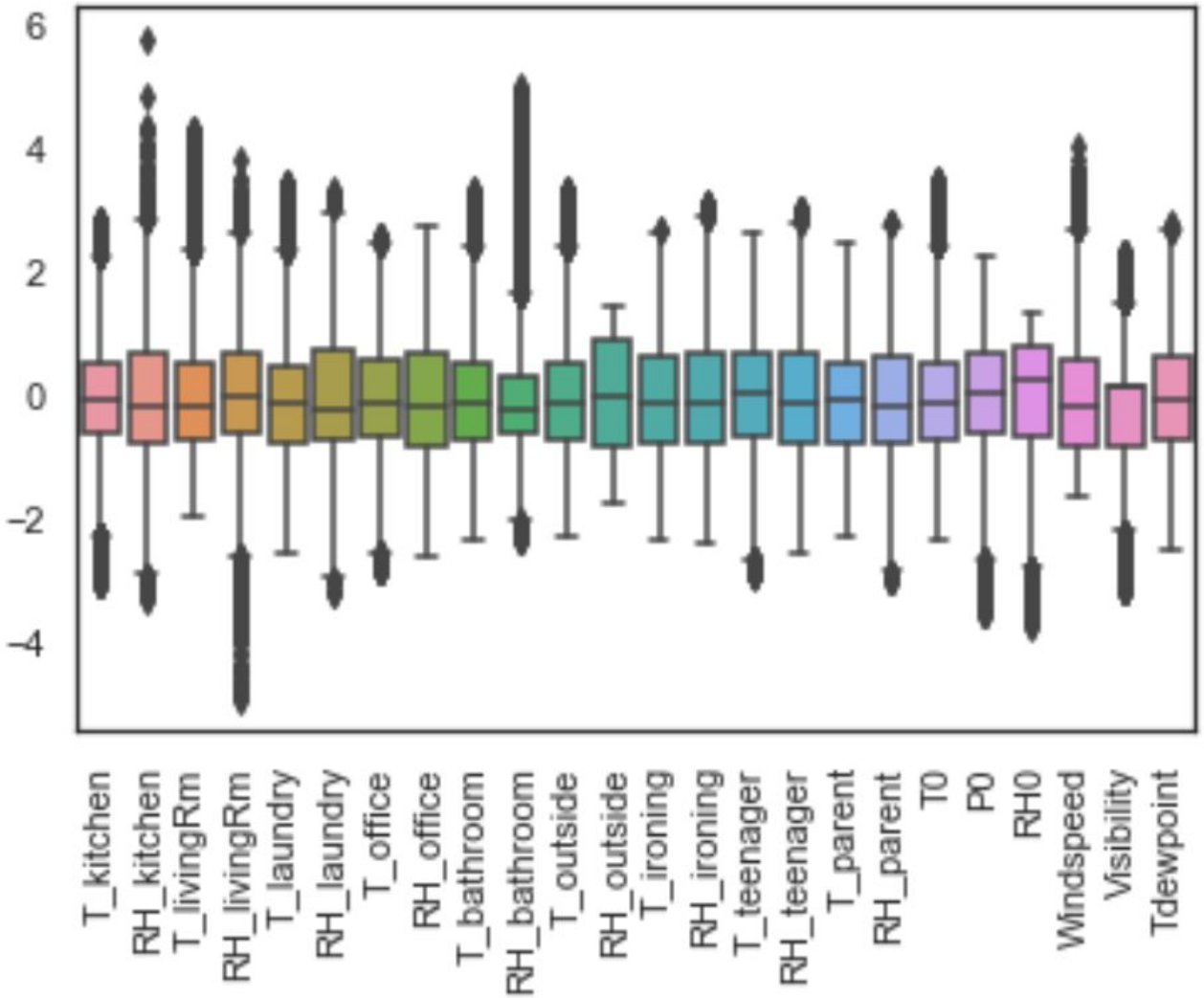


Exploratory data analysis 2

- Here we have box plots for the distribution of feature values.
- The first, appliances, shows a large variation in values, but most are limited to a small range that is between 10 and 80 kW-hours.
- The range of values for all other features, including lighting, is small.
- In the next chart we will omit the appliances and lighting and use normalization to closely examine features.

Exploratory data analysis 3

- Appliance and lighting is omitted.
- The above boxplots are normalized – subtract mean, divide by variance.
- There are outliers in the dataset
- Some room measurements are skewed for example room humidity in the bathroom is skewed right most likely due to the use of the shower and RH in the kitchen is skewed right possibly from increased humidity when cooking

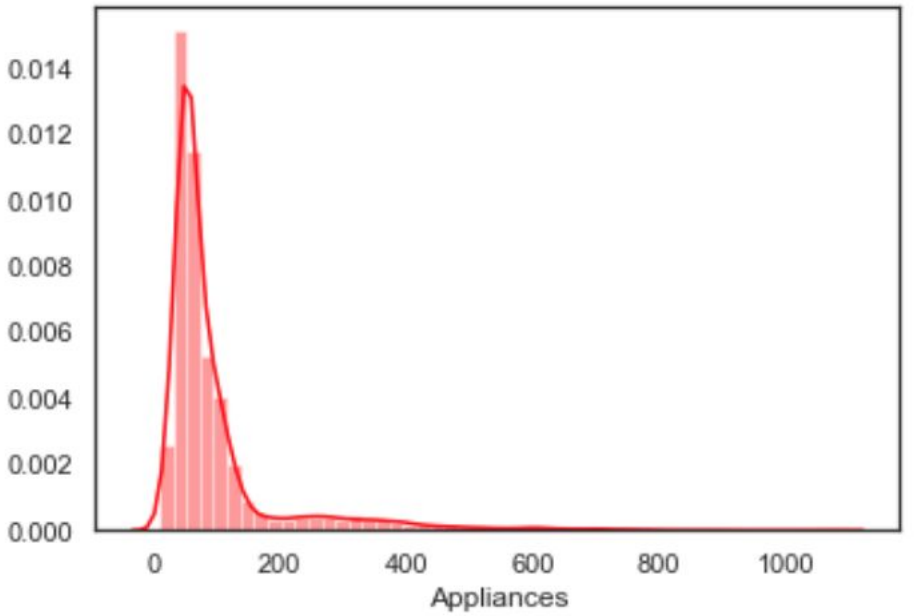


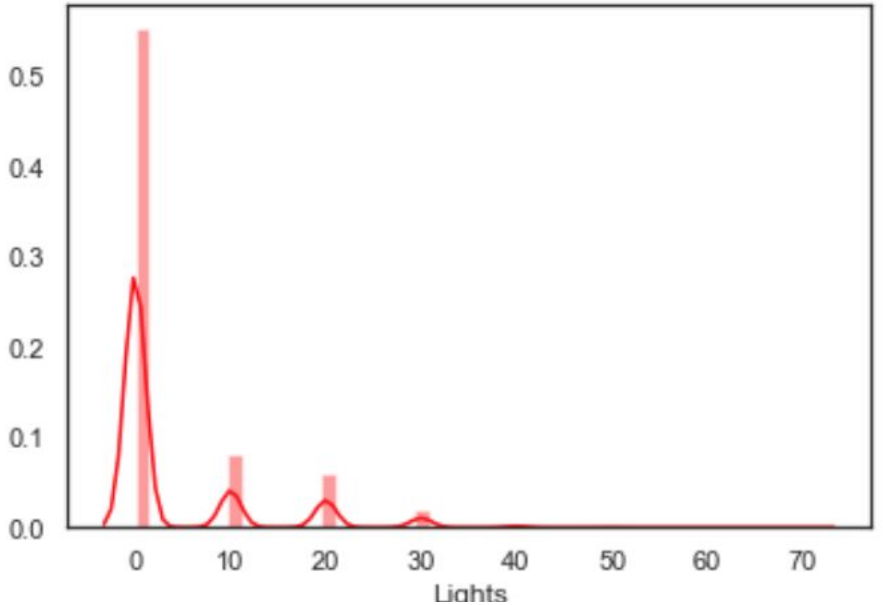
Exploratory data analysis 3

- This chart suggests that features have a bell-shaped distribution

Exploratory data analysis 4

- Distribution of appliance and lighting energy usage
- Most of the appliance energy (in a 10-minute interval) is between 10 and 180 Watt-hours, with occasional peak usage
- Most of the lighting energy is zero, since lights are typically off from midnight to sunset
- No obvious outliers need to be replaced (nothing extraordinary)



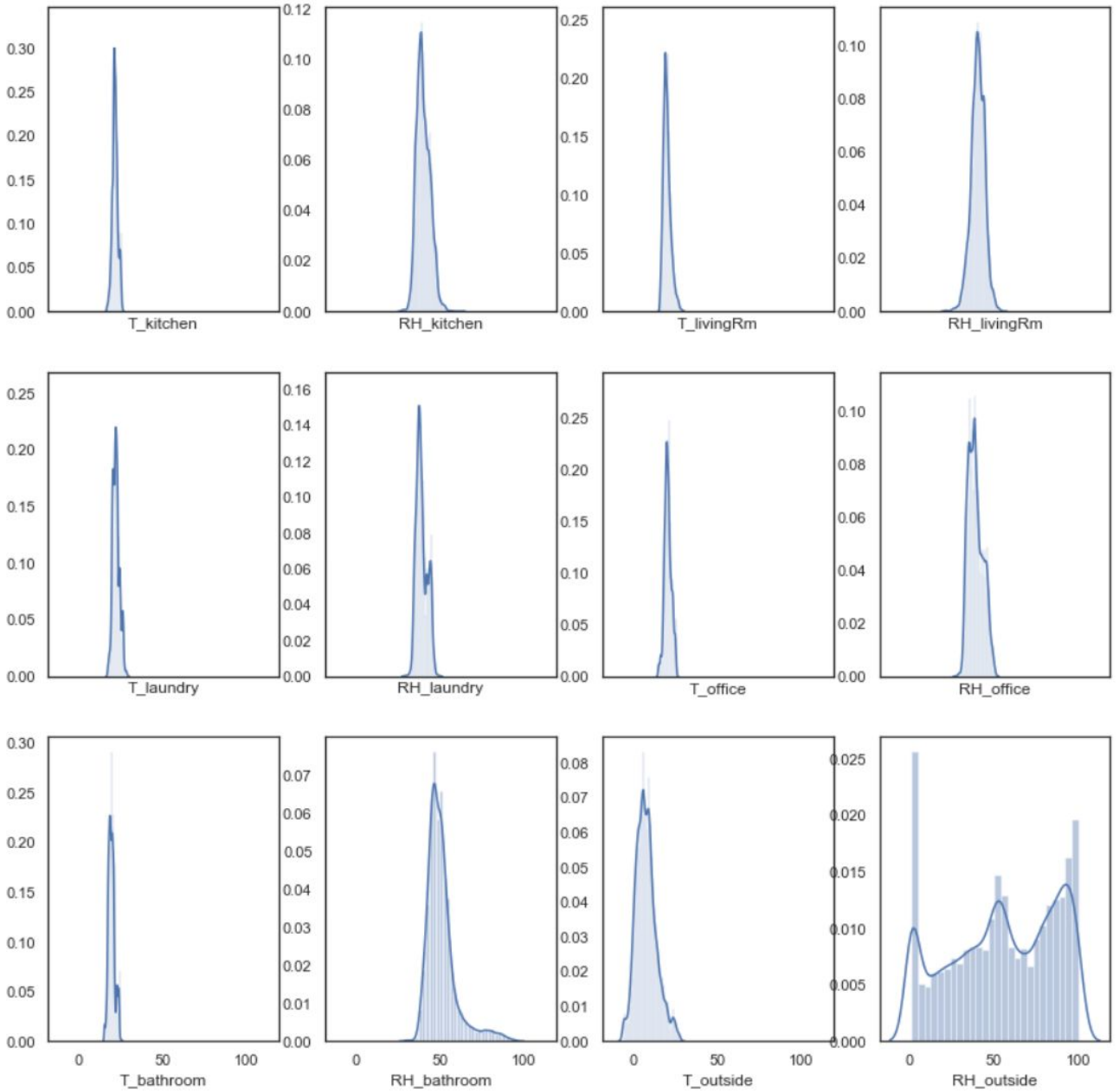


Exploratory data analysis 4

- Here we have two charts that show the distribution of values for appliances and lights.
- The top chart shows that the appliance energy -- in a 10-minute interval -- is between 10 and 180 Watt-hours, with occasional peak usage.
- Most of the lighting energy is zero, which makes sense, since lights are typically off from midnight to sunset.
- We see that lighting contributes a small fraction to the overall energy use, which either (1) belies the admonition to turn lights off when leaving a room or (2) suggests that that family does listen to their father.
- We see nothing extraordinary in these distribution and so conclude that they have no outliers that should be corrected.

Exploratory data analysis 5

- Distribution of 12 input features
- Indoor and outdoor temperature and humidity
- Most data is moderate, except the outside humidity, which varies randomly form 0 to 100%
- No obvious outliers need to be replaced

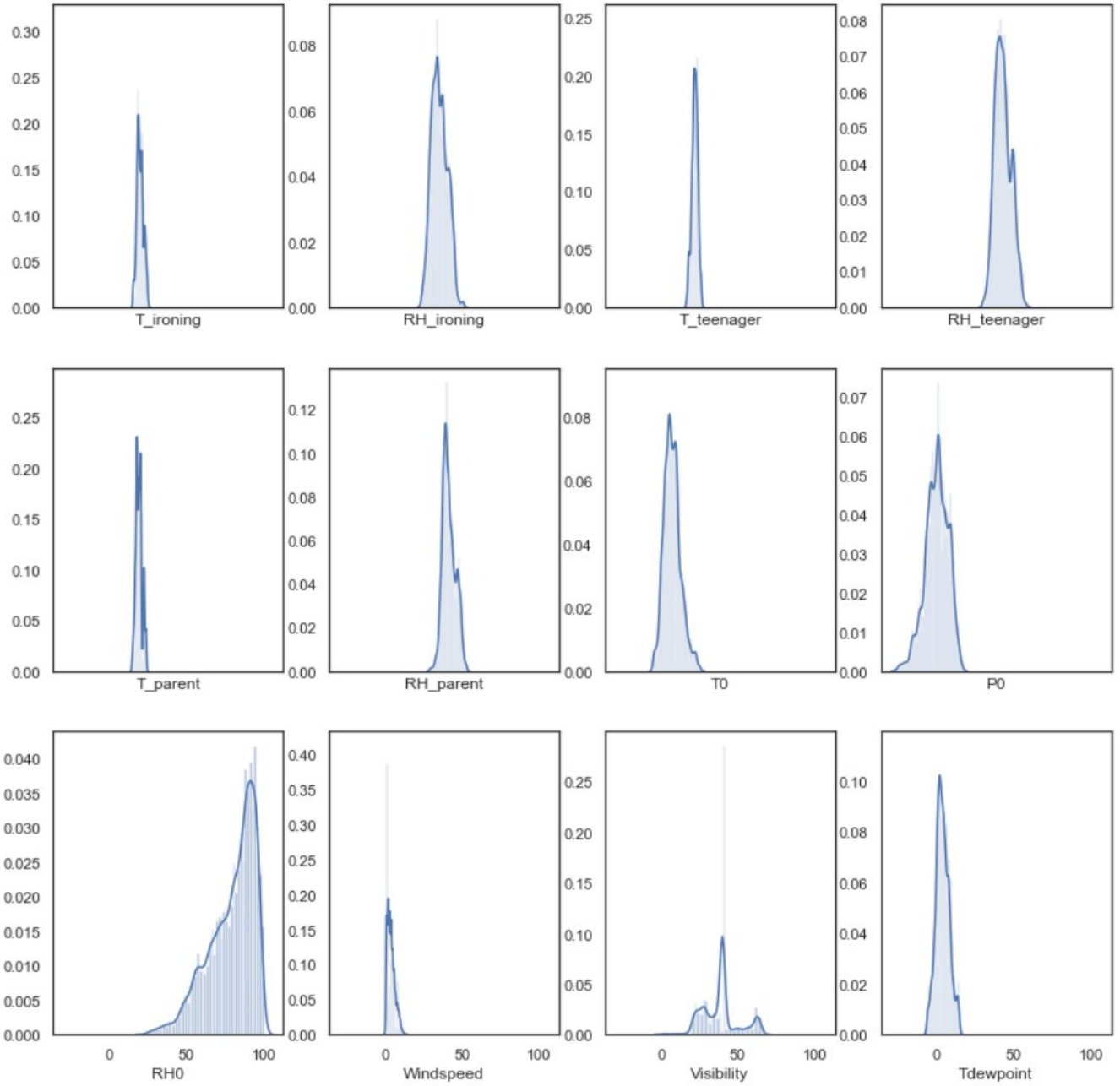


Exploratory data analysis 5

- There are 24 input features – this chart shows the value distributions of the first 12.
- These 12 show the inside temperature and humidity.
- They are as expected and show that the family maintains a comfortable house.
- The only variation is the outside humidity – in the bottom right corner -- which the family cannot control and is as one would expect.

Exploratory data analysis 6

- Another distribution of 12 input features
- Indoor temperature and humidity and airport weather
- Most data is moderate, except the airport humidity, which is variable with the peak at 80%, and falling rapidly
- No obvious outliers need to be replaced



Exploratory data analysis 6

- Here are the other 12 features, which are the other rooms and the airport weather.
- They appear appear as one would expect.

Appliance energy usage visualization

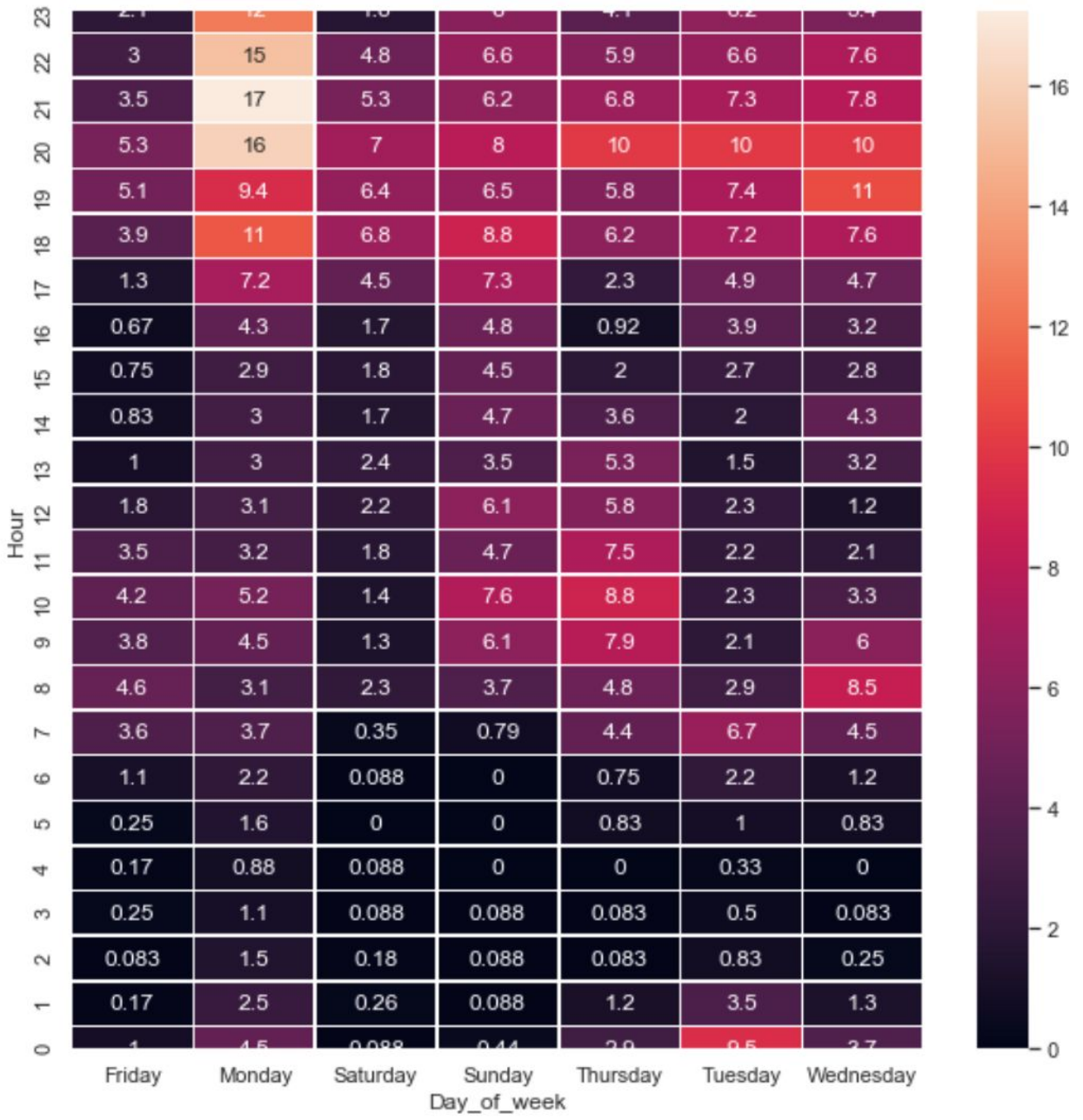
- Peak appliance usage is found on Thursday at 1800 to 1900 (6pm to 7pm) at 220 kW-hour.
- Least amount of usage as expected, is during the 10pm to 6am time frame.
- Probable factors: AC unit, cooking, people getting home from work, and entertainment appliances.

Appliance energy usage visualization

- This heatmap shows appliance energy usage for the 7 days of the week and 24 hours of the day.
- The peak hours are: Thursday at 1800 to 1900 (6pm to 7pm) at 220 kW-hour and non-peak hours: 10pm to 6am.

Lighting energy usage visualization

- Peak light usage is found on Monday from 9pm to 11pm at 15-17 kW-hour.
- Least amount of usage as expected, is during the midnight to 6am time frame.
- Probable factors: people getting home from work and getting ready for the next day.



Lighting energy usage visualization

- The lighting heatmap, depicts less lighting is used on Friday after 10pm – which could be due to people having plans at different locations.
- Compared to the rest of the week, the majority of lighting is used on Monday – possibly due to preparing for the week ahead.
- These heatmaps were made with matplotlib / seaborn

Cull features & add dummy variables

- Remove the following features, which are not be used in the ML models
  - date - replaced with Day_of_week, NSM
  - rv1 - in original dataset, but not used
  - rv2 - in original dataset, but not used
  - Date - not needed for ML prediction, which will use Day_of_week, NSM
  - Time - not needed for ML prediction, which will use Day_of_week, NSM
  - Hour - not needed for ML prediction, which will use Day_of_week, NSM
- Add dummy variables to replace the single feature Day_of_week. Instead 7 new features are added, like Day_of_week_Sunday

Cull features & add dummy variables

- We identified these features that are not applicable for the models.
- We created a new dataset without those features and replaced the Day_of_week categorical feature with seven dummy variables.

Run Recursive Feature Elimination

- Eliminating non-predictive features speeds up the prediction - don't waste CPU cycles on worthless features or risk overfitting, which is being bound too closely to a “particular” set of data
- A Scikit-Learn function, RFE, ranks features for predictive value. The 5 most predictive values are used
- The ranking for appliances and lighting is completely different
- Took a computer-hour, using only 5% (1000) records, randomly selected
- The literature suggests that an alternative, Boruta, provides equivalent results as RFE, but is less CPU intensive
  http://www.socr.umich.edu/people/dinov/courses/DSPA_notes/16_FeatureSelection.html

Run Recursive Feature Elimination

- We noticed that eliminating non-predictive features speeds up the prediction - don't waste CPU cycles on worthless features or risk overfitting, which is being bound too closely to a “particular” set of data
- We used an sklearn function, which ranks features for predictive value
- We built another dataset that only has the 5 most predictive values
- The computer ran for an hour, using only 1000 records, which were randomly selected from the larger dataset. Fortunately, training models is much quicker.

Appliances & Lighting feature ranking

Feature	Appliance rank	Lighting rank		Feature	Appliance rank	Lighting rank		Feature	Appliance rank	Lighting rank
T_kitchen	1	8		RH_outside	26	23		Visibility	23	15
RH_kitchen	2	9		T_ironing	4	2		Tdewpoint	18	20
T_livingRm	7	4		RH_ironing	25	14		Day_of_week_Friday	28	28
RH_livingRm	5	13		T_teenager	1	3		Day_of_week_Monday	21	24
T_laundry	13	7		RH_teenager	3	6		Day_of_week_Saturday	6	15
RH_laundry	11	17		T_parent	1	1		Day_of_week_Sunday	1	10
T_office	8	1		RH_parent	14	26		Day_of_week_Thursday	9	12
RH_office	20	5		T0	22	22		Day_of_week_Tuesday	17	1
T_bathroom	15	1		P0	27	27		Day_of_week_Wednesday	1	1
RH_bathroom	24	21		RH0	19	16		NSM	10	19
T_outside	16	18		Windspeed	12	11				

Appliances & Lighting feature ranking

- Here are the results from the ranking by recursive feature elimination.
- There are three separate tables, that divide the 32 features.
- The ranking for appliance and lighting is shown to the right of each feature.
- As you can see, the ranking is different for appliances and lighting.
- Therefore, we built separate datasets for appliance and lighting.

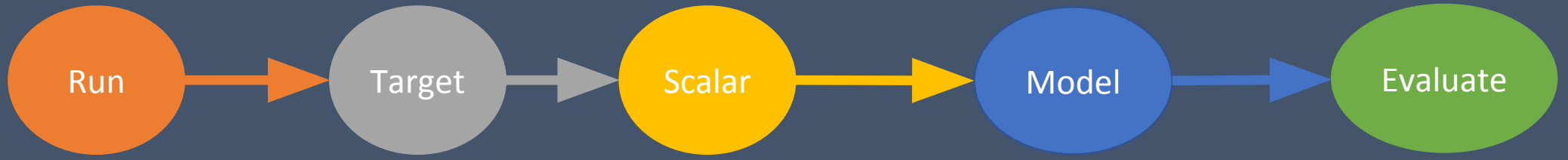
Identify positive - negative correlation features

- Linear regression models performed poorly when using features that are both positively and negatively correlated to the target (app/lights)
- Split the dataset into one with positive correlation and one with negative correlation.
- A total of four datasets are created, two for appliances and two for lighting, since the positive - negative mix is different for those two targets.
- Test this hypothesis during model evaluation

Identify positive - negative correlation features

- You have seen that features can be positively or negatively correlated with the targets.
- We suspect that linear regression models should only use either positively or negatively correlated features, because superimposing positive and negative slopes may cancel each other.
- We, therefore, created four datasets. Two for appliances with either positive or negative correlated features and two for lighting.
- If our hypothesis is correct, we should see a difference in the model performance.

Evaluate models through iteration



- full feature set
- top 5 RFE ranked for Appliances
- top 5 RFE ranked for Lights'
- positive correlation w/ Appliances
- positive correlation w/ Lights
- negative correlation w/ Appliances
- negative correlation w/ Lights

- Appliances
- Lights

- StandardScaler
- MinMaxScaler’
- PowerTransformer
- RobustScaler

- LinearRegression
- Lasso
- Ridge
- ElasticNet
- SVR
- RandomForestRegressor

- fit scalar
- split training & test
- train
- Evaluate r2 & MSE
- Export r2 & MSE

Easy ML Class

- Allows a developer to access machine learning models by simply giving known inputs and outputs.
- Iterate through combinations of scalars and machine learning models to identify the best model for any ml problem.
- Output dictionary object with each model ready to use for end user
- Below is an example of the dictionary object output:

```
Out[40]: {'Linear Regression (Power)': {'model': LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False),
           'MSE': 0.7215178712353634,
           'R2': 0.2689474534885328},
          'Lasso (Power)': {'model': Lasso(alpha=0.01, copy_X=True, fit_intercept=True, max_iter=1000,
                normalize=False, positive=False, precompute=False, random_state=None,
                selection='cyclic', tol=0.0001, warm_start=False),
           'MSE': 0.7432259090218253,
           'R2': 0.24695254950037737},
          'Ridge (Power)': {'model': Ridge(alpha=0.01, copy_X=True, fit_intercept=True, max_iter=None,
                normalize=False, random_state=None, solver='auto', tol=0.001),
           'MSE': 0.7215179299180283,
           'R2': 0.2689473940303915},
          'Elastic Net (Power)': {'model': ElasticNet(alpha=0.01, copy_X=True, fit_intercept=True, l1_ratio=0.5,
                      max_iter=1000, normalize=False, positive=False, precompute=False,
                      random_state=None, selection='cyclic', tol=0.0001, warm_start=False),
           'MSE': 0.7332857550440877,
           'R2': 0.2570240600863858},
          'SVR (Power)': {'model': sklearn.svm.classes.SVR,
           'MSE': 0.4843954102353143,
```

EasyML pseudocode

```
class Easy_ML(object):
        def __init__(self, X, y):
                #Initialize object pass in inputs and outputs and results dictionary


        def find_best_model():
                def transform_selection(transform_type):
                        this function is used in the find_best_model and takes an input of a transformation type and outputss the corresponding transformer object
                # reshape X and y
                #split X and y
                #iterate through all the combinations of scalars and models and add r2, MSE, and models to results dictionary
                #return results dictionary
```

Results

Model 1 (Appliances)

Models	Scalers: Min Max	Power	Robust	Standard
Elastic Net	0.0133	0.2570	0.1367	0.1335
Lasso	-0.0005	0.2470	0.1322	0.1253
Linear Regression	0.1448	0.2689	0.1448	0.1448
RFR	0.5974	0.7232	0.5986	0.5991
Ridge	0.1448	0.2689	0.1448	0.1448
SVR	0.1197	0.5092	0.1794	0.2325

Model 1 (Lights)

Models	Scalers: Min Max	Power	Robust	Standard
Elastic Net	0.0506	0.1939	0.2347	0.2295
Lasso	-0.0002	0.1897	0.2369	0.2237
Linear Regression	0.2382	0.2013	0.2382	0.2382
RFR	0.7033	0.5553	0.7014	0.7023
Ridge	0.2382	0.2013	0.2382	0.2382
SVR	0.3364	0.3304	0.1379	0.4139

Negative Correlation (Appliances)

Models	Scalers: Min Max	Power	Robust	Standard
Elastic Net	-0.0005	0.0734	0.0212	0.0212
Lasso	-0.0005	0.0715	0.0209	0.0205
Linear Regression	0.0202	0.0735	0.0202	0.0202
RFR	0.5780	0.7040	0.5763	0.5778
Ridge	0.0202	0.0735	0.0202	0.0202
SVR	-0.0794	0.3027	0.0447	0.0670

Negative Correlation (Lights)

Models	Scalers: Min Max	Power	Robust	Standard
Elastic Net	-0.0002	0.1407	0.1646	0.1628
Lasso	-0.0002	0.1346	0.1651	0.1564
Linear Regression	0.1651	0.1434	0.1651	0.1651
RFR	0.6820	0.5458	0.6848	0.6839
Ridge	0.1651	0.1434	0.1651	0.1651
SVR	0.2179	0.2350	-0.0279	0.2984

Positive Correlation (Appliances)

Models	Scalers: Min Max	Power	Robust	Standard
Elastic Net	0.0133	0.2353	0.1241	0.1227
Lasso	-0.0005	0.2305	0.1217	0.1161
Linear Regression	0.1303	0.2432	0.1303	0.1303
RFR	0.6056	0.7211	0.6062	0.6037
Ridge	0.1303	0.2432	0.1303	0.1303
SVR	0.1066	0.4780	0.1524	0.1972

Positive Correlation (Lights)

Models	Scalers: Min Max	Power	Robust	Standard
Elastic Net	0.0506	0.1259	0.1292	0.1290
Lasso	-0.0002	0.1246	0.1290	0.1275
Linear Regression	0.1287	0.1260	0.1287	0.1287
RFR	0.6691	0.5401	0.6693	0.6706
Ridge	0.1287	0.1260	0.1287	0.1287
SVR	0.0848	0.1584	0.0017	0.2087

Recursive Feature Elimination (Appliances)

Models	Scalers: Min Max	Power	Robust	Standard
Elastic Net	-0.0005	0.0494	0.0074	0.0074
Lasso	-0.0005	0.0486	0.0069	0.0063
Linear Regression	0.0074	0.0491	0.0074	0.0074
RFR	0.4229	0.5353	0.4252	0.4237
Ridge	0.0074	0.0491	0.0074	0.0074
SVR	-0.1791	0.0858	-0.0841	-0.0704

Recursive Feature Elimination (Lights)

Models	Scalers: Min Max	Power	Robust	Standard
Elastic Net	-0.0002	0.0929	0.1113	0.1109
Lasso	-0.0002	0.0920	0.1115	0.1099
Linear Regression	0.1113	0.0932	0.1113	0.1113
RFR	0.5109	0.4202	0.5101	0.5085
Ridge	0.1113	0.0932	0.1113	0.1113
SVR	-0.1265	0.0013	-0.0538	0.0314