Team Greex X: Tristan Tseng, Cass Wang, Lucy Wingard

Intro to AI

12/15/2019

# Kaggle Term Project

These are the steps I followd, in order:

1. As a first step, I load and preprocess the data.
2. I do some exploratory data analysis.
3. And Finally, I train a regressor model to predict the meter values.

## Data Pre-Processing

First of all, I loaded the reported measured meter values, and then the weather related features, and building metadata. These data frames are shown bellow:

A screenshot of a cell phone

Description automatically generated

The first thing that I noted was that the weather and metadata files have lots of missing values. In fact, just the “cloud\_coverage” column in the weather data frame, had about 50% missing values, as shown bellow,

A screenshot of a cell phone

Description automatically generated

I could do imputation to derive values for the missing entries, however, since there are so many records, I decided drop all rows with missing values, in a later stage.

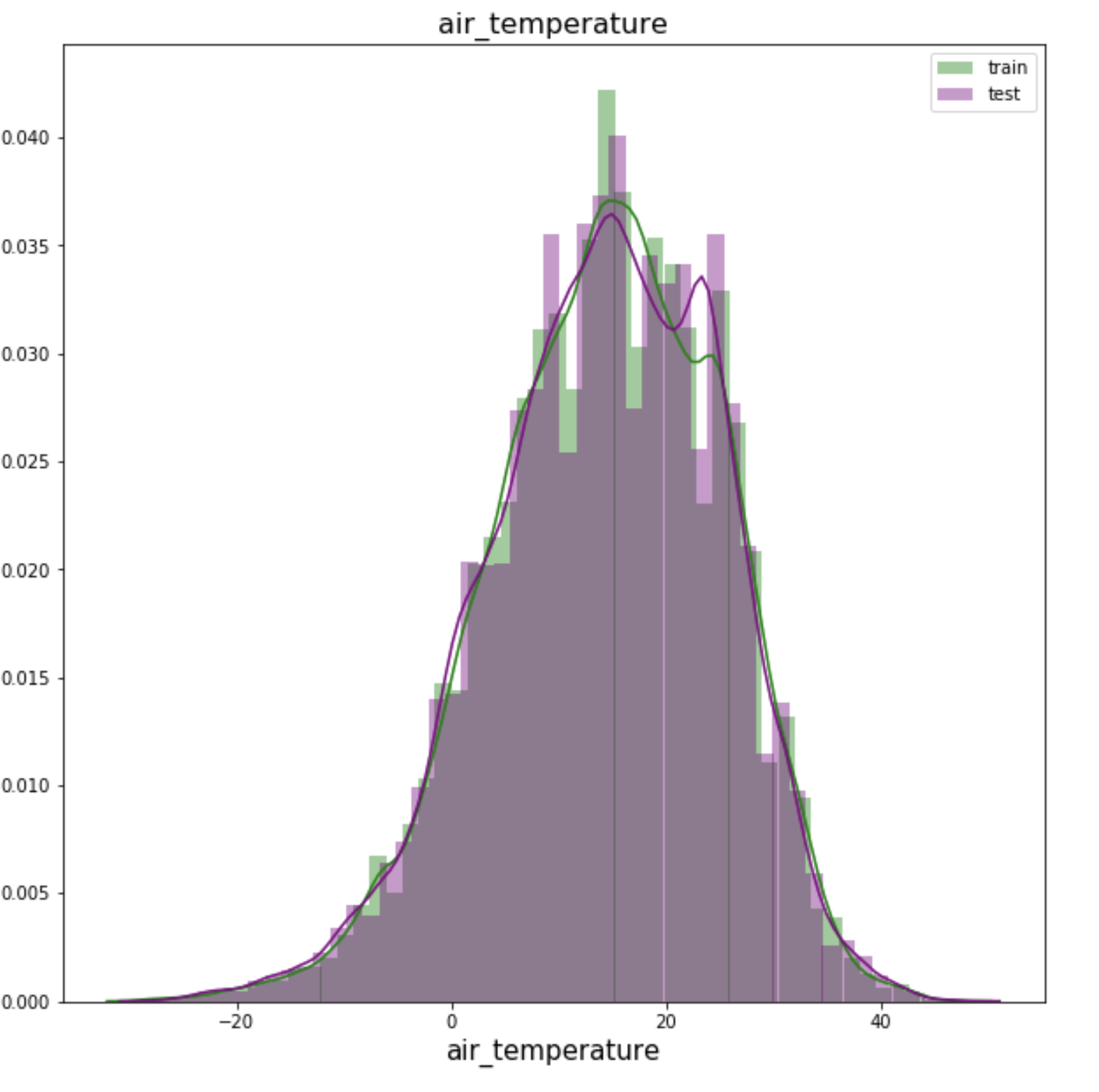
I also, merged the meter reading data frame with the meta-data, on the “building\_id” so that we have all the building features associated to each reading. Furthermore, to train a model that predicts the power consumption we need to consider the weather around each building. We can join the weather data frame with the data frame resulted form the previous stage using the “site\_id” as the key, to associate the weather status around each building’ area. As such, we will get a unified data frame with all features in it. Also, as a technical note, we reduced the numeric features to the ones that take smaller number of bytes, if larger data type is not necessary, and that way, we significantly reduced the memory allocated to our data (i.e. around 50-60% less memory usage).

## Exploratory Data Analysis

The distribution of the meter reading appears to be sharply picked on small reading values (most likely belonging to the regular homes) but with a very tall tail (probably associated to large corporations). Below I show this distribution after filtering out all the high-power consumption unit. The features in the weather data, on the other hand, show relatively smooth normal distribution. Figure 4, and 5, depicts such a distribution for “air\_temparature” and “wind\_speed”.

A screenshot of a cell phone

Description automatically generated



A screenshot of a cell phone

Description automatically generated

And finally, figure below, shows the distribution of the label, i.e. the “meter\_reading” column.

A screenshot of a cell phone

Description automatically generated

## Generating Features and Developing a Model

In order to make predictions, I had to generate a feature matrix that is ammeanable to model training. First, some features are not useful in a raw format, such as the ‘timestamp’. However, timestamp informs us about the parodic patterns of life. For example, the day of a week, or the hour of the day both are very well correlated to the pattern of energy consumption. So, instead of using these features in the raw format, I derived other features such as “day of the week”, “hour of the day”, “day of the year”, etc. And, added them to the feature vector, and removed the original column. Also, we removed all ID features, such as “site\_id” or “building\_id” from the feature matrix. And finally, I converted the categorical features, i.e. “meter” and “primary\_use”, using a one-hot coding approach. This, altogether, generated 39 features in my feature set.

Finally, we trained a random forest regression model. I selected RF, because it is an ensemble learning model and as a result is relatively robust with respect to the noise in the data. Furthermore, I searched for the optimal number of estimators as well as depth, for my model, using a simple grid search strategy. Specifically, I tried all combinations of 10, 20 number of estimators and 5, 10, and 20 different dept values. For the evaluation purpose, I used the 3-fold cross validation strategy to avoid being misled by the overfitting phenomenon. Table below illustrates the mean of the RMSE score on the three test folds, for each experiment.

|  |  |
| --- | --- |
| Parameter Setting (n\_est, max\_depth) | Cross-Validation RMSE |
| (10, 5) | 243.5 |
| (10,10) | 124.4 |
| (10,20) | 107.4 |
| (20,5) | 242.6 |
| (20,10) | 123.7 |
| (20,20) | 105.2 |

As we can see from the table, the model with 20 estimators and depth 20, resulted in the best performance. Higher number of estimators will lead to a more generalizable model, while higher number depth helps training a more expressive model. In combination these two resulted in the best performance on the test folds.