CSCI567 Fall 2015 - Mini Project

Team: MLCLASS_REIGNtheRAIN
Vaibhav Behl, Veeresh Beeram, Nishkala Jayasimha
University of Southern California
Los Angeles, 90007
vbehl@usc.edu, beeram@usc.edu, jayasimh@usc.edu

How Much Did It Rain ? II

In this report we describe our approach to building data models to predict hourly rainfall at a location using polarimetric radar data. The raw dataset and problem statement is taken from "How much did it Rain? II" competition on kaggle [1]. Current rainfall measurements with rain gauges are unreliable. Polarimetric radars are used to provide high quality weather data using radio wave pulses with both horizontal and vertical orientations. The problem focuses on predicting hourly rain gauge total from the given snapshots of polarimetric radar values. First we introduce the problem and describe the dataset. Then we identify important pre-processing steps to transform the data. After that, we describe the Feature Engineering methodology along with the machine learning models we used for this regression problem. Finally, we discuss the results obtained along with some insight into how the performance can be further increased.

1 Introduction

In this problem we attempt to estimate the amount of rainfall at a given place using data from weather radars. The dataset is from a doppler radar which measures reflectivity due to doppler effect of raindrops. The radio waves from radar are scattered by rain drop and ice particles in the atmosphere. The amount of scattering depends on the size of the raindrop/ice-particle which is given in the reflectivity values of the dataset. Since the particles are in vertical motion, the vertical-orientation waves have a phase differential in the reflected wave. The size of the raindrops directly affects the amount of precipitation at the point [3]. There can be variability in radar readings for same amount of rain due to stormy, snowy or other conditions that affect the radar reflectivity values. The rain gauges just capture the amount of water collected in the gauge in an hour. This data can be inaccurate due to uncertain events such as clogging. It is known that rain drops follow exponential drop size distribution [4]. A simple reflectivity and drop size distribution is captured by Marshall-Palmer model, which we use as a baseline. Our initial approach was to start with simple regression models and assess their performance. But as with most Kaggle competitions, our research for methods drove us towards ensemble models [6].

2 Data

The data consists of multiple sets of radar observations along with the corresponding rain gauge reading for each set, where each set can be uniquely identified by an `id`. A set here represents the radar observations made over an hour with the rain gauge reading recorded after the hour and associated with that set's `Expected` value. Thus, the multiple radar observations within each set can also be thought of as a time series data with which we associate a single target(`Expected`) value.

2.1 Description of the Data

The training dataset consists of 13,765,201 observations.Each of these has 22 observable features. This set was drawn from radar observations at gauges in the Midwestern US over 20 days each month during the corn growing season. We have been given the gauge reading at the end of each hour for the training set. Our test data consists of 8022757 observations. This set is obtained from radar observations at gauges in the Midwestern US over the remaining 10/11 days each month of the same year(s) as the training set. Few important columns of the observations are:

- The `id`- which is the unique number that is assigned for the set of observations over an hour.
- `minutes_past`–the minutes past the top of the hour. Each radar observation is a snapshot at that point in time.
- `radardist_km`- the distance of the radar from the gauge.
- `Ref`- radar reflectivity in dBZ (wrongly mentioned as km on data-description page)
- `RefComposite`- maximum reflectivity in the vertical column above the gauge. This is measured in dBZ.
- `RhoHv`- correlation coefficient.
- `Zdr`- Differential reflectivity in dB
- `Kdp`- Specific differential phase. This is measured in (deg/km).
- `Expected`- Actual gauge observation in mm at the end of the hour.
- `valid_time`- This feature was not in the original train.csv, but was derived from minutes_past field. It contains a numeric value indicating the time for which that particular radar observation is valid. This helps in calculating weighted-mean of observation.

Each of the Reflectivity values, correlation coefficient and differential phase values have 4 columns representing value above the gauge and the 10th, 50th and 90th percentile values in a 5x5 neighborhood around the rain gauge.
The test dataset is same as the training dataset, but the 'Expected' field is missing. We estimate this field.

2.2 Data preprocessing and cleaning

This dataset posed the following challenges - size of the training set, high percentage of missing values, many outliers and its unorthodox structure. We enumerate these problems below and explain how we solved them.

- This dataset has multiple radar measurements within an hour and a single rain gauge reading at the end of the hour. So, it has a many-to-one mapping between input observations and output target for every hour(`id`). This made it tricky to select random samples for cross-validation as we can't just randomly split the training set. So we split the training set based on the `id` (values belonging to one `id` group should be in either train set of test set). This computation was costly and so we decided to make ten separate splits of training data, out of which we can randomly select train and test set at run-time.
- During the course of the competition, it was mentioned that for a particular `id`, if all its `Ref` values are null, then that `id` will not count towards the final MAE score. Using this fact, we removed all the `id` values from the train set that satisfied this condition. This helped reduce the size of the training set to 9,125,329 rows or 731,556 unique hourly measurements.
- While generating the pre-processed training file, we also added a field called `valid_time`, which was derived from `minutes_past`. This was added to save computation cost associated with calculating it again and again for all `id` values.
- It was mentioned on the Kaggle forums that the `Expected` rain gauge values were converted from inch to mm because the rain gauges report their values in inches. So, we used this fact to further filter out values which did not have proper `Expected` values.

- Plotting a histogram (Figure 1) of Expected values on a log scale, it was observed that there were some `Expected` values which were abnormally high. This was attributed to faulty rain gauges and weather phenomenon like snow melting. To deal with this we had to set our outlier cut-off point. We used cross validation and found that a cut-off point of 243.6919 mm resulted in the highest accuracy. Also note that the highest ever single hour rainfall in US is around 356mm [5].

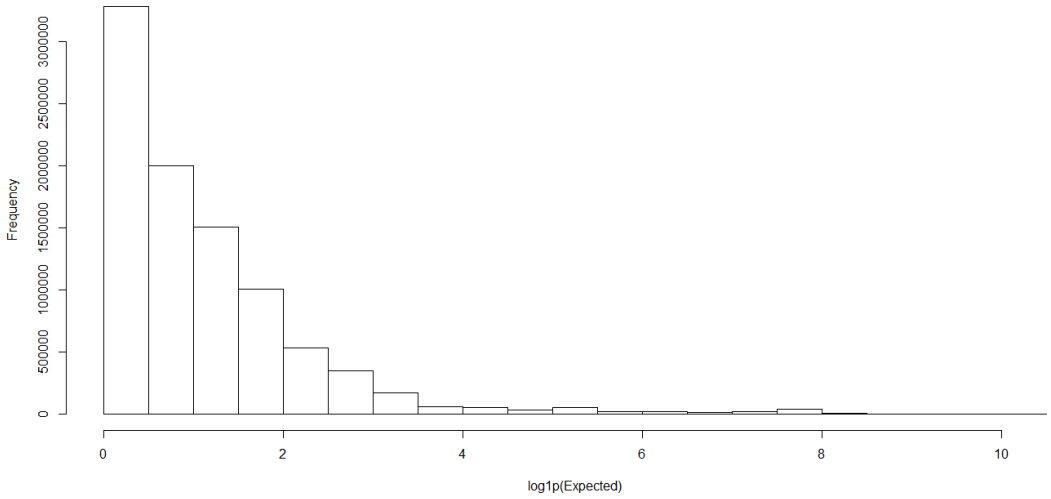

Figure 1: Histogram of Expected Values

3 Feature Engineering

As stated previously, the unorthodox structure of this dataset, where we have multiple rows of features(radar reading) for each target value (`Expected`), makes it difficult to directly apply regression algorithms on it. The usual structure of a dataset is of the form where we have a vector(single row) of features and a corresponding target value. Thus, we needed a way to convert our given dataset into the latter form while still retaining the time series information of the hourly radar readings.

To capture this time series information, we experimented with many different types of statistical measures [8].

Statistic Name	Description
Mean	Numerical measure of the central location of the data values
Median	Value at the middle when the data is sorted in ascending order
Interquartile Range	Difference between the upper and lower quartiles
Variance	Measure of how the data values is dispersed around the mean
Standard Deviation	square root of variance
Covariance	Measure of linear relationship between two variables
Skewness	It is defined by the formula: $\mu_3/\mu_2^{\frac{3}{2}}$ $\mu_3 - 3^{\text{rd}}centralmoment, \mu_2 - 2^{\text{nd}}centralmoment$
Kurtosis	It is defined by the formula: $\mu_4/\mu_2^2 - 3$ $\mu_4 - 4^{\text{th}}centralmoment, \mu_2 - 2^{\text{nd}}centralmoment$
Min,Max	Minimum and Maximum value

Another important factor that came into play during feature engineering was the abundance of missing values in data. This made the value of a lot of the derived statistical features as NA(missing). To solve this, we merged the time series data of similar features (eg. `Ref`, `Ref_5x5_10th`, `Ref_5x5_50th`, `Ref_5x5_90th`) and then calculated the statistical features on that merged time series data. This reduced the number of missing values while still allowing us to capture the

general patter. These combined features were some of the most important features in our model.

Using a combination of these statistical features we were able to extract a total of 100 features per `id`, after which we applied cross-validation to select the features which gave us the best performance improvement while avoiding over-fitting. We tried these features on various set of models which we will describe in the next section. Overall, feature engineering was the most challenging part of this competition because of the way data was structured.

4 Model and Learning Algorithms

Before we delve into the models we used, we should first try to understand the evaluation metric for this competition, which is "MAE" (Mean Absolute Error). The objective function for MAE that we want to minimize over is:

$$J = \sum_{i=1}^{N} \|y_i - \hat{y}_i\|$$

Where we have N samples, y_i is the true value of i^{th} sample and $\hat{y}_i$ is the predicted value. Using MAE means that all values will be penalized equally. Below we discuss a number of models that we experimented with and report our findings.

4.1 Sample Solution Benchmark - Marshall Palmer

This was the benchmark solution provided to us by this competition. It used only the `Ref` value to estimate the rainfall and thus didn't need any train data to perform its prediction. Still, it performed decently and got a score of 23.128 on the local CV, and a public and private score of 24.06968 and 25.09833 respectively. The formula for Marshall-Palmer is as follows:

$$\hat{y}_i = 200 R_i^{1.6}$$

Where R_i is the reflectivity value of radar (Ref column in dataset).

Marshall-Palmer was later used as a features where it greatly improved the training accuracy of different models.

4.2 Linear Regression(SGD)

We used the sklearn's Linear Regression with Stochastic Gradient Descent and tried both L2 and L1 norm as regularization. This model did not perform very well compared to other models we used. It got an average local CV score of 23.05. One of the reasons for it was that this algorithm did not know how to handle missing values, so we had to do data imputation and use mean value wherever there were missing values present.

4.3 Sklearn - Gradient Boosting Regression

Next we experimented with sklearn's GBR. We selected this because it provided us an option to select lad(least absolute deviation) as its loss function whereas most other regression algorithms only supported least squares. Still, we were not able to achieve a good accuracy on the validation set with this algorithm, with the average local CV score being 22.90. This algorithm also didn't handle missing values and thus required some type of data imputation. This could be the reason why it scored so low, compared to our best model which also used gradient boosting, but it supported missing values.

4.4 H2O - Random Forest Regression

This was one of the best models and performed quite well compared to our previous models. It got an average local CV score of 22.40. In the implementation we used the h2o's random forest model in regression mode. We optimized various hyper-parameters of this algorithm using cross-validation. This model was used as an ensemble for some time along with XGB, but the latter out-performed soon after and thus this model was dropped from usage.

4.5 Xgboost

This was our best model which gave the highest local CV score of 22.04 as well as our highest public and private score of 23.73757 and 24.75383 respectively. We used the R implementation of this library for our project. This was only gradient boosting library we found which supported handling missing values. It handled missing values by automatically finding its best imputation value based on the reduction on training loss. We believe that this was the deciding factor in this model being the best because a lot of training data had missing values and this model was able to deal with it elegantly.

4.6 Other Hybrid Techniques

We also applied other advanced techniques like using a classification algorithm to first classify which range the target value belongs in, and then using a regression algorithm to find its actual value. While in theory this algorithm sounds very nice and also gave a fantastic local CV score of 15!, it performed very poorly on public and private leaderboard, managing a score of over 25. We believe the low local CV score was due to our model overfitting the training data and the poor public score was because our model didn't generalize enough. We tried a bunch of times with different configuration, but reached the conclusion that it was not possible to classify which bin a value belongs to, thus not possible to directly classify outliers on test set from training data alone. This could be one of the reasons why no one was able to break below the 23.68 public score.

5 Results

5.1 Execution of code

To generate the sample solution you will need R Studio with all the necessary machine learning packages installed. After that executing the script 'generate_final_solution.R' will generate the required submission file. For more details refer the file 'how_to_run.txt' located at the root of the directory.

5.2 Final Results

We achieved our current best MAE using xgboost library and obtained a Public leaderboard score of 23.73757 and a Private Leaderboard score of 24.75383.

6 Other Insights

We noticed very late into the competition that the distribution of `Expected` vs number of records in an `Id` were very different for the training data(Figure 2) and the predictions (Figure 3) we generated. As it can be seen from the graphs below, a large density for training's `Expected` value lie within the interval 0 and 1. But, for our predictions, that interval is very sparse. This could be one reason why it was difficult to lower the prediction error beyond a certain limit.

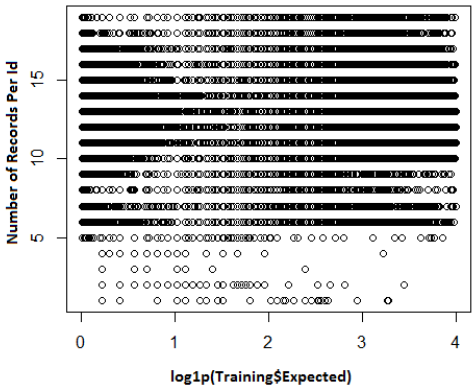

Figure 2: Distribution Pattern of Training's Expected values

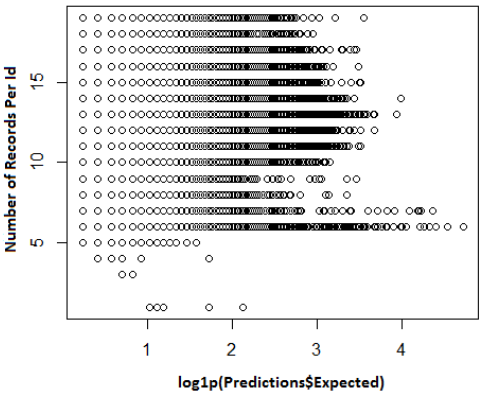

Figure 3: Distribution Pattern of Prediction's Expected values

7 References

[1] Kaggle - How much did it rain II, https://www.kaggle.com/c/how-much-did-it-rain-ii

[2] How does radar work? https://en.wikipedia.org/wiki/Weather_radar

[3]Doviak, R. J.; Zrnic, D. S. (1993). Doppler Radar and Weather Observations (2nd ed.). San Diego CA: Academic Press.

[4] Drop size distribution, http://glossary.ametsoc.org/wiki/Drop-size_distribution

[5] Notable local floods of 1942-43, Floods of July 18, 1942 in north-central Pennsylvania, with a section on Descriptive details of the storm and floods http://pubs.er.usgs.gov/publication/wsp1134B

[6] Kaggle Ensembling Guide http://mlwave.com/kaggle-ensembling-guide/

[7] Marshall, John S., and W. Mc K. Palmer. "The distribution of raindrops with size." Journal of meteorology 5.4 (1948): 165-166.

[8] Statistical Measures in R http://www.r-tutor.com/elementary-statistics/numerical-measures