How much did it rain -II

Akhilesh Saklecha **Ankit Agrawal** **Saurabh Gadia**

Computer Science
University of Southern California
{saklecha,ankituma,gadia}@usc.edu

Abstract

This paper presents the work done and results obtained for kaggle competition How much did it rain-ii. In this competition data from polarimetric radars and corresponding gauge reading on hourly basis was given for training purpose. The prediction about the hourly rainfall had to be made given only polarimetric radars data. This paper depicts the work done by each team member and corresponding result and inferences. We start from data cleaning, pre-processing, feature selection and feature enrichment followed by different techniques and models used for training and model selection. Results obtained at each phase.

1 Introduction

In Chinese mythology, the one who can predict the rain is known as the messenger who can talk to "the ruler of the ocean, the dragon king". Data prediction using machine learning techniques is becoming very popular these days. Data prediction techniques can also be used to predict rainfalls given the relevant data. 'How much did it rain' tries to do same with the radar polarimetric data, i.e. given the radar polarimetric readings, rainfall has to be predicted. This paper presents the work done by our team on this project. We start from feature engineering, models experimented, results obtained and conclusion.

2 Data Preprocessing

2.1 Original Data Description

Data provided by the competition organizers consists of hourly readings of factors like radar distance, reflectivity, differential reflectivity, correlation coefficient, and variants of these factors that can help in estimating amount of rainfall. Multiple readings of these factors were taken in a single hour. A unique ID is assigned to each hour and the final column contains the amount of rainfall recorded in that hour. More information about the data can be found at [1]
We analyzed the data in two ways- grouping the data by ID, and keeping the data as is and grouping the results. All the processing on the data below will be divided in two categories.

2.2 Cleaning and Missing Value Replacement

In the given data for some unique IDs, the rainfall recordings for 'Ref' field was completely missing for all the recordings of that hour. As per the competition forum and guidelines we discarded all the readings for that hour. Apart from this mandatory task we also had a lot of missing values for almost every feature. So to fill the 'NaN' values, we took two approaches:

1. We grouped data based on its ID and replaced the column values by its mean. This way we got single recording for each recording having mean in its respective columns. Even after

this grouping the data had some missing values because of absence of the column values in all recording of its ID. So further, we filled such missing data with the column mean. After filling the missing data we transformed all data of the column to zero mean and unit standard deviation.

2. We tried multiple approaches to replace the missing values for the complete data like replacing with mean, mode, and replacement with PCA. The last approach was made possible by using Microsoft Azure ML services.

3 Feature Engineering

3.1 Data Analysis

We first started analyzing the data by discovering the correlation between all the features. For preliminary data analysis we applied multiple statistical inference mechanisms to get most out of original data. Following is the list of data discovery methods we used:

3.1.1 Correlation Observation

We first computed the correlations between each of the columns whose result can be found at [2]. We discovered that there was no direct relationship of any feature with the predictor column. Following were the observation based on correlation data:

1. Ref, Ref_10, Ref_50, Ref_90 had a very good positive relationship of approximately 0.87
2. RefComposite, RefComposite_10th, RefComposite_50th, RefComposite_90 with average 0.90

Apart from this we did not get a strong correlation between any other features.

3.1.2 Mean, percentile, standard deviation and other statistics

After calculating the mean, percentile, standard deviation for each of the feature we discovered that the predictor column, Expected, followed Gamma Distribution. Almost 90% of the data had expected value between 0-10 mm. Complete statistics can be found at [3]

3.1.3 Plotting the scatter plots and histogram

With the support of above statistics we went further plotting the scatter plots of each feature against the predictor column. We also generated the frequency distribution of the Expected predictor value. We discovered that the data had plenty of outliers which would make it difficult to model the distribution for any predictor algorithm. Due to limitation of space the frequency distribution of 'Expected' values, and 'Reflectivity' vs 'Expected' plots are shown in Figure1,2. Rest of the plots can be found at [4]

3.2 New features

We added a new feature called 'Marshall Palmer evaluation' which contains the value predicted by Marshall Palmer relation using reflectivity. We also tried to add other new features like the variance of 'Ref', ratio of 'Ref' and 'radardist' after observing the correlation between various existing features . But we realized that these did not improve the accuracy as we expected. Another approach that we tried was to modify the 'Expected' value according to reflectivity so that it depicts the amount of rainfall during that particular reading instead of the complete hour. This too did not improve our results much.

4 Models Experimented

We used a lot of regressor models to get an idea about the problem. The major ones were Linear Ridge Regression, SVM, Random Forest, GBM, Kernelized SVM(RBF Kernel) from python's

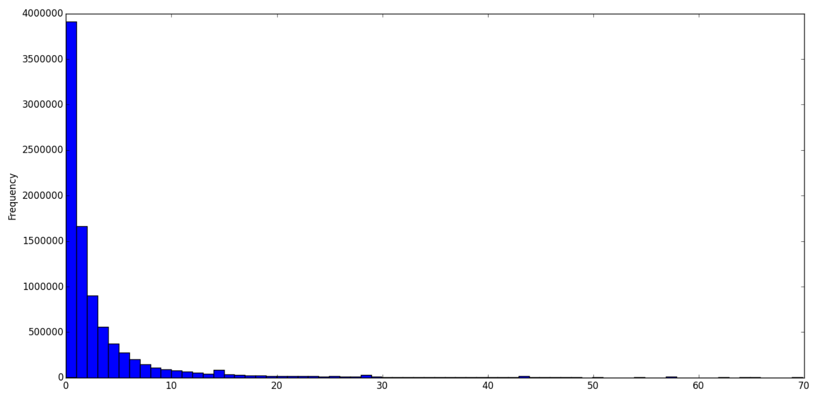

Figure 1: Histogram of 'Expected' value ¡ 70mm (follows Gamma distribution) .

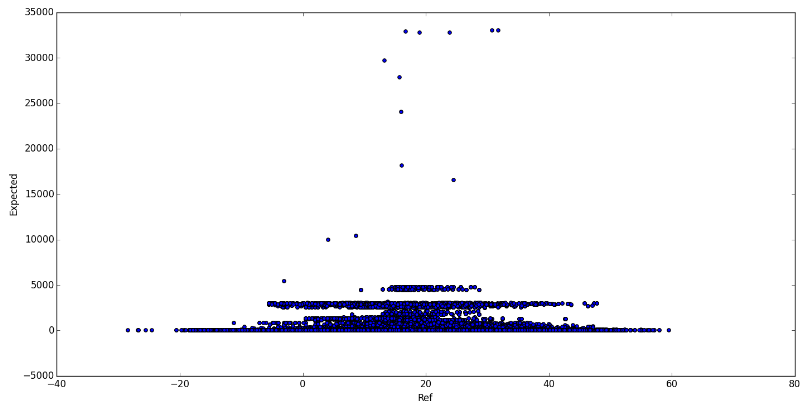

Figure 2: 'Ref' vs 'Expected'.

sklearn package. We also utilized Neural networks regression, Boosted decision tree regression from Microsoft's Azure Machine Learning platform . The results are shown in Table1

5 Best Prediction Approach- Ensemble of predictors

After experimenting with individual machine learning algorithms we chose to design an ensemble of best individual predictors and further tuned it to generate the final results for given test data. We generated the pipeline for complete process right from start of the hyper-parameter tuning of the individual predictors to generating the test output after tuning the ensemble of predictors. Below is the layout of complete strategy:

5.1 Data Partitioning

We divided the complete pre-cleaned training data into 3 parts:

- 40% data for training of individual predictors (Train 1 dataset) - Pipeline Stage I
- 40% data for training of ensemble of predictors (Train 2 dataset) Pipeline Stage II
- 20% hold out data set Local Test Prediction

Table 1: Results of various methods used

METHOD	RESULT
Linear Ridge Regression	24.66
SVM(RBF Kernel)	23.9
Boosted Decision Tree Regression	27.5
Neural Network Regression	25.67

Table 2: Pipeline I results

PREDICTOR	TRAIN ERROR	TEST ERROR	HYPER PARAMETERS
Gradient Boosting Regressor	1.9906680	22.9497564	n_estimators=600,Learning rate=0.3, Max_depth=6,Exp_Threshold=30
Extra Trees regressor	0.12215439	22.768551	N_estimators=600,Exp_Threshold=40
Random Forest regressor	0.8861031	22.96342	N_estimators=600,Exp_Threshold=25
XGBoost Regressor	1.9905	22.8325	max_depth=6, num_rounds=10, objective:count-poisson
Marhsall Palmer	24.0523	24.45652	Formula Available

For training we used 5 fold validation.

5.2 Pipeline Stage 1

After experimenting with the individual predictors in the beginning of the project, we chose the best ones among these individual predictors to be a part of our ensemble method.The process is as follows:

- Train individual predictors on train 1 data with tuned parameters
- Get the prediction results of all individual predictors on train 2 data
- Append the original values of train 2 'Expected' column to the predicted results from individual predictors, this acts as training data for pipeline stage II

The results of individual predictors are shown in Table2

5.3 Pipeline Stage II

After generating the results of individual predictors, we now have 5 new features for ensemble training. Following were the steps involved in tuning of hyper-parameters for ensemble learning:

1. **Predictor Selection**
   After getting the training data for this stage, we had to again choose the learning algorithm for training of stage II of pipeline. This was done by applying **GBR, Extra Trees, Random Forest, SVM, XGBoost regressors**. The best results were given by **Gradient Boosting Regressor** and so we decided to hyper tune it's parameters for final prediction.
2. **Feature Selection**
   Initially we were skeptical about whether to include the original features of stage I into stage II for ensemble training. We used Recursive greedy approach for feature selection and also used the knowledge of Feature_importance attribute of individual predictors used in above predictor selection process. Based on these feature selection process we discovered that results were better when only the outcomes of pipeline stage I were selected as features. Thus we just used the stage I outcome as predictor features and choose **Gradient Boosting Regressor** as our predictor method for training the ensemble model as it outperformed all other predictors.
3. **Feature Permutations**
   Among the predictor features we got the averaged result for combination of features. We

Table 3: Ensemble methods result

ENSEMBLE TRAINING METHOD	TRAIN ERROR	HOLDOUT DATA ERROR	KAGGLE SCORE
Gradient Boosting	1.5343579	22.17306585	23.85641

tried with combination of 2, 3, 4, 5 predictors as features to model our ensemble method and further discovered that ensemble method was performing best for below hyperparameters:

(a) N_estimators = 600
(b) Learning rate = 0.3
(c) Feature Set = Marhsall Palmer prediction, Gradient Boosting Regressor prediction, Extra Trees Regressor prediction
(d) Max_depth = 6

6 Results

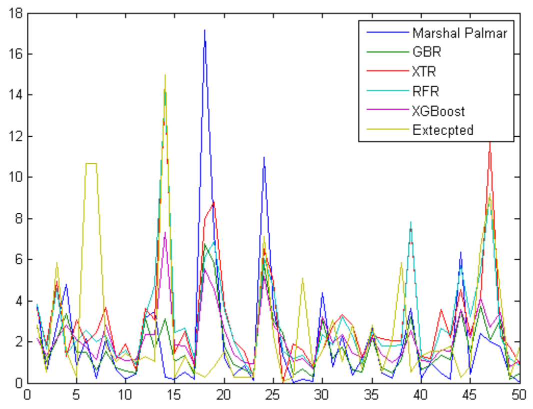

Figure 3: First 50 predictions by ensemble predictors .

Final results are shown in Table3.
The predictions for first 50 unique IDs are plotted in Figure3.
From our analysis we discovered that ensemble methods work better than all the individual regressors that we tried.

7 Conclusion

We learnt a lot from this project. There were a lot of hurdles in coming up with an effective way to replace the missing values, finding outliers, and finding the perfect regression algorithm for the problem. At the end our faith in ”United we stand divided we fall” was affirmed by the results of the problem.

References

[1] *Data description:* https://www.kaggle.com/c/how-much-did-it-rain-ii/data

[2] *Corelation Results:* https://drive.google.com/folderview?id=0B44HRKVuGCkFcmRtZ2tmWUgwR0U&usp=sharing

[3] *Statistics:* https://docs.google.com/document/d/1QZRaOE5T9sb6hEbBoffPVsBkLPtntNM05IJgJZiVa90/edit?usp=sharing

[4] *Plots:* https://drive.google.com/folderview?id=0B44HRKVuGCkFYlZ2YklKRnhFR3c&usp=sharing