Title

A combined approach for predicting sparse variables such as tips ratio and daily precipitation

Author

Li, Jinshu

Publication Date

2019

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

A combined approach for predicting sparse variables such as tips ratio and daily precipitation

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Statistics

by

Jinshu Li

2019

ABSTRACT OF THE THESIS

A combined approach for predicting sparse variables
such as tips ratio and daily precipitation

by

Jinshu Li
Master of Science in Statistics
University of California, Los Angeles, 2019
Professor Yingnian Wu, Chair

A sparse variable is a variable whose values are mostly zero. Because of its sparsity, satisfactory prediction results of a sparse variable usually cannot be obtained by either pure (i.e. single) regression or pure classification machine learning methods. Therefore, to resolve this difficulty, this thesis paper proposes a framework that combines a regression model and a classification model. Furthermore, two types of the combined regression and classification framework are discussed, and their differences are illustrated. Two sparse variables are selected as the case studies: taxi tips ratio (i.e. tips amount divided by total fare) and daily precipitation volume (i.e. total rainfall amount in one day). The author first employs Lasso regression to select relevant features for each sparse variable, with the best Lasso parameter determined by cross-validation (CV). Second, the author selects Logistic regression and the AdaBoost method as the classification methods, while the XGBoost method is chosen as the regression method. The hyperparameters are determined by fine-tuning. The author then surveys over the prediction results of the pure classification method, the pure regression method, and the combined method, using root mean square error (RMSE) as the metric. The results show that the pure regression method provides the least RMSE for both variables; however, it does not satisfy the sparsity requirement. On the contrary, the combined method, whose RMSE is close to the RMSE of the pure regression method, can also provide the sparse results, which makes it an efficient way to predict sparse variables like taxi tip ratio and daily precipitation.

The thesis of Jinshu Li is approved.

Qing Zhou

Hongquan Xu

Yingnian Wu, Committee Chair

University of California, Los Angeles

2019

To the people who guided me

To the people who have been consistently supporting me

To the people that I deeply love

and,

Thank you, mom and dad.

Special thanks to all professors in UCLA Statistics Department.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

Predicting the value of a random variable is an essential task in statistics and machine learning. It typically involves an activity where a mapping function is first inferred from a set of labeled training examples, then a prediction can be made based on the learned mapping function and a new testing example. The predictions based on such procedures are called supervised learning.

Supervised learning is focused on learning the relation between training input variables and training output variables. On the contrary, unsupervised learning, another kind of learning method, does not have labeled training examples (i.e. no obvious categories of input and output variables); instead, it focuses on investigating the hidden structure from the unlabeled training data (Murphy, 2012; Wu, 2019). Also, since no labeled training output is available, it is much harder to evaluate the performance of unsupervised learning methods than supervised learning methods. The widely-used unsupervised learning methods include Principal Component Analysis (PCA) (Pearson, 1901; Hotelling, 1933), Independent Component Analysis (ICA) (Hyvrinen, A. and Oja, 2000), embedding methods, clustering algorithms (Murphy, 2012; Wu, 2019), etc. Reinforcement learning (Sutton and Barto, 2014; Wu, 2019) is another kind of learning method that is newly emerged in the field of deep learning. Its learning strategy is between supervised learning and unsupervised learning, which involves mapping from state to action based on some policies. However, since most prediction activities are conducted under supervised learning, in this study the author will focus on supervised learning.

Many popular statistical models and machine learning methods belong to supervised learning. Depending on the format of outputs, they can be divided into two categories: regression methods and classification methods (Murphy, 2012; Wu, 2019). Regression models output continuous prediction results. Popular regression results include ordinary least-squares regression (OLS), ridge regression (Hoerl and Kennard, 1970), lasso regression (Tibshirani, 1996), kernel regression (Takeda et al., 2006), boosting machine (Schapire, 1990) (e.g. gradient boosting), and artificial neural network (ANN) (McCulloch and Pitts, 1943), etc. Unlike regression, classification models provide binary classes prediction (or multi-classes prediction). Popular classification models are logistic regression (Walker et al., 1967), support vector machine (SVM) (Cortes and Vapnik, 1995), boost machine (e.g. Adaboost), and ANN, etc.

Because of different output formats, classification models and regression models typically cannot be mix-used. In other words, for a given prediction problem, it can either be solved by a regression problem or a classification problem. If the output variable is continuous, a regression model is used, while classification models are more appropriate if the output variable is binary.

Sparse variable is a special variable x that has a positive probability weight on value zero (i.e. $P(x = 0) = \beta > 0$), and the rest of the probability weight is on the rest support of x (e.g. $P(x \neq 0) = 1 - \beta$). In other words, the sparse variable x is discrete between zero and the rest of support, but continuous on the rest of support. The definition of sparse variable x decides that it is more likely for x to be equal to zero than any other values, since $P(x = b) = 0$, where b is a nonzero value on the support. Thus, if we sample from the distribution of sparse variable x and store the results in a vector, the vector would be a sparse vector, in which most elements are zero.

Predicting a sparse variable's value based on learned relation is worth studying, since sparse variables are very common in engineering and daily life. For example, the taxi tips

ratio is defined as the tips amount divided by the total trip fare, which can be regarded as a sparse variable. Since giving tips is not mandatory in some regions, many customers in those regions will choose not to give tips, and this will lead to many zero values for the ratio variable. Another sparse variable example in hydrology is daily precipitation. For a given day, the daily precipitation would be zero if there is no rain on that day, which is very often. However, daily precipitation can also be a very high number if there is a heavy rain on that day.

Precisely predicting a sparse variable is not easy, and often more difficult than predicting other variables. This is because a sparse variable is neither discrete nor continuous, so directly applying either regression or classification methods may not yield desired results. Therefore, in this study, I propose a framework that combines a classification model and a regression model, for predicting the sparse variable. I will then compare the predicting performance of the proposed framework with a single classification and a single regression model.

The classification models selected in this study are logistic regression (Walker et al., 1967) and Adaboost (Freund and Schapire, 1997). Logistic regression is a traditional classification model that can be regarded as one of the generalized linear models (GLM). Adaboost belongs to boosting machine and consists of several weak classifiers. The final classification decision is made upon a perceptron based on those weak classifiers. The regression model selected in this study is extreme gradient boosting (XGB). XGB (Chen and Guestrin, 2016) is a newly emerged gradient boosting techniques that quickly drew a lot attention due to its good accuracy and efficiency. All the methods will be elaborated in the following methodology section.

All the aforementioned supervised learning methods require a design matrix (i.e. training examples) that consists of a number of features. To achieve a good prediction, it is necessary to select a subset of available features that are related to the output variable, but not highly related with each other. This is due to the fact that if the selected features are not related

to the output variable, then the designed model based on such features are weak in prediction. And if the selected features are highly related with each other, then collinearity may occur, which will lead to a non-full rank design matrix. Therefore, feature selection must be conducted before any models are built. Lasso regression is a linear regression model with a l_1 regularization, which is known for feature selection by selecting a relatively small number of regression coefficients to be non-zeros, and setting the coefficients of other features to be zeros. Thus, Lasso regression will be employed for feature selection before constructing machine learning models.

Here is an outline of this thesis: Section 2.1 elaborates on the Lasso regression for feature selection and cross-validation (CV) method for the determination of Lasso parameter. Section 2.2 discusses the proposed classification-regression framework for predicting sparse variables. Section 2.3 elaborates on the specific machine learning models used in this study, which include logistic regression, Adaboost, and XGBoost. Chapter 3 applies a pure classification model, a pure regression model and the proposed classification-regression framework on two sparse variables: taxi tips ratio and daily precipitation volume. Final remarks and conclusions are provided in Chapter 4.

CHAPTER 2

Methodology

The main contribution of this thesis is the proposal of a classification-regression framework to predict sparse variables (i.e. classification-regression framework type I and II). A flowchart of our proposed method of predicting sparse variables is shown in Figure 2.1. I first elaborate on how to use Lasso regression to select relevant features. Next, a classification-regression framework is proposed for predicting sparse variables. Then I discuss several classification and regression methods that can be used in the proposed framework, including logistic regression, Adaboost, and XGBoost. Finally, the proposed classification-regression is tested to predict two sparse variables: taxi tips ratio and daily precipitation volume, using real-world data.

2.1 Lasso regression for feature selection

Lasso regression is typically referred to the linear regression with l_1 regularization. For the following generic linear model:

$$y_i = \sum_{j=1}^{p} x_{ij}\beta_j + \varepsilon_i, \tag{2.1}$$

where y_i is the output (response, dependent variable); x_{ij} is the i_{th} observation of j_{th} feature (predictor, regressor); β_j is the regression coefficient of j_{th} feature;

The ordinary linear regression (OLS) is aimed at solving the following least square opti-

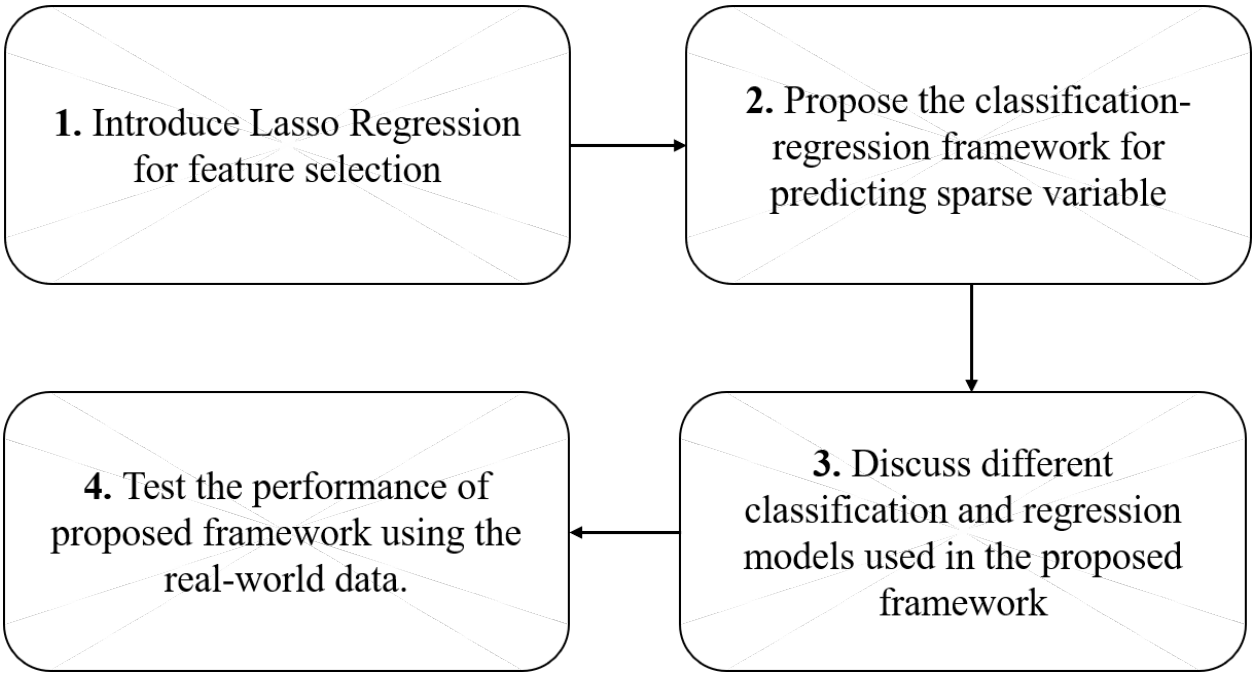

Figure 2.1: A flowchart of the proposed method.

mization problem:

$$\underset{\beta}{\textbf{minimize}} \quad RSS(\beta) = ||Y - X\beta||_2^2, \tag{2.2}$$

where RSS is the function of Residual Square, defined as square of the difference between the data and its prediction value. X is the design matrix; β is the regression coefficient vector; Y is the response vector.

OLS does not penalize on the β, which might lead to a non-sparse β vector. Thus, the l_1 regularization is added in the above optimization problem to make β vector as sparse as possible, shown as Eqn.(2.3).

$$\underset{\beta}{\textbf{minimize}} \quad RSS(\beta) + \lambda \cdot ||\beta||_1 \tag{2.3}$$

where $||\beta||_1$ is the l_1 norm of β vector; λ is the a tuning parameter (i.e. Lasso parameter). Since any norm is a convex function, so Eqn.(2.3) is still a convex optimization problem. However, $||\beta||_1$ is not differentiable, which prevents us using gradient method to solve that problem. We can choose other optimization method, such as sub-gradient method, proximal

gradient method, or ADMM (Alternating direction method of multiplier), to solve this non-smooth problem. Note this problem can also be transformed into a smooth problem and then solved easily, which is adopted by CVX and other popular solvers.

The Lasso (shrinkage) parameter λ controls the amount of regularization and the sparsity of β vector. For each λ, we have a solution (β vector). So, we can vary the value of λ trace out a path of solution and select the λ value, under which all solutions are stable.

Another method to determine Lasso parameter λ is to use K-fold cross-validation. It partitions the training data T into K separate equal-size sets and fits the model to the training set excluding the kth-fold T_k. This T_k will serve as the testing set for each fitted model and the test error (residual square) is calculated for each model. Figure 2.2 provides an illustrative diagram. The average residual square (CV error) is computed based on those k models. So, a range of λ value is first selected, then the CV error is calculated for each λ. The best λ value in the range is the λ with the minimum CV error. In this thesis, both methods of determining are conducted: I first employ the Lasso trace method, and then apply CV to validate the results.

2.2 Classification-regression framework for predicting sparse variables

As discussed, the sparse variable has a positive probability weight on zero, and the rest of the probability weight is on the rest of support. In other words, the sparse variable x is discrete between zero and the rest of support, but continuous on the rest of support. Thus, the prediction of its value based on the selected features is an interesting challenge. In this thesis, I propose and test the following four methods that can be used to predict such variable's value.

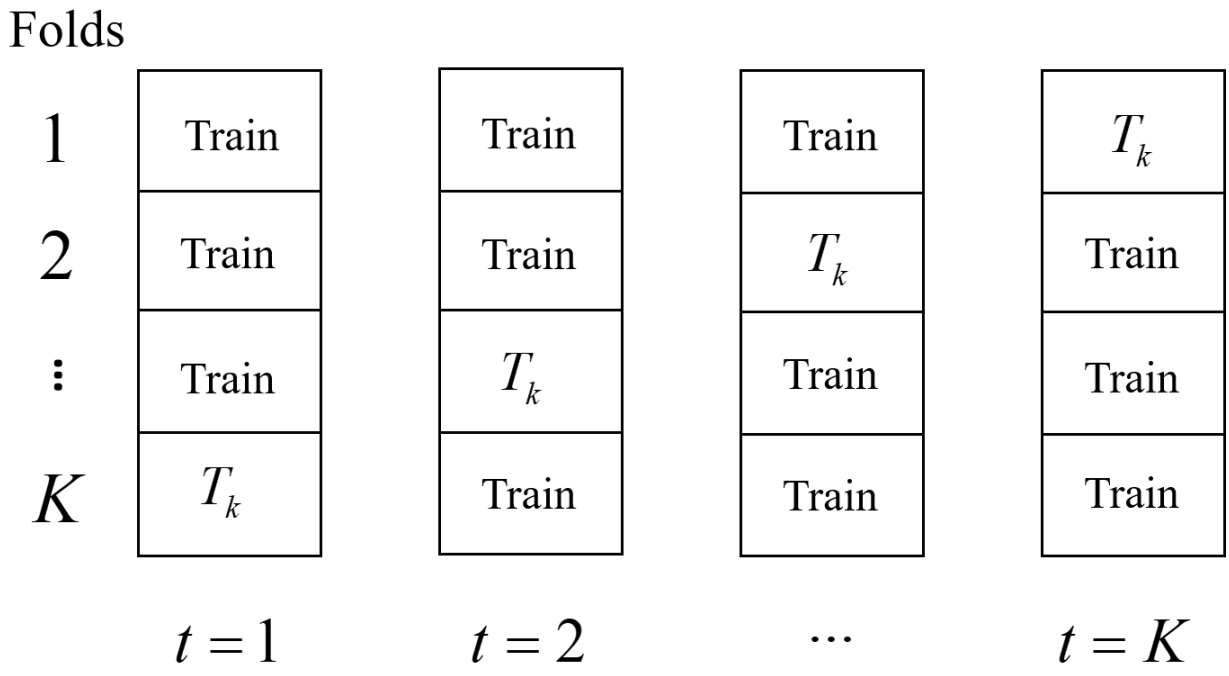

Figure 2.2: An illustrative diagram for K-folds cross-validation.

2.2.1 Pure regression method

Assume the sparse variable is continuous and select a regression model to predict the variable. Calculate the root mean square error (RMSE). The pseudo code is shown as follows:

$$f_r = train(X_{train}), \tag{2.4}$$

$$\hat{Y}_r = f_r(X), \tag{2.5}$$

$$RMSE_r = ||\hat{Y}_r - Y||_2, \tag{2.6}$$

where X_{train} is the training design matrix of selected features; X is the testing design matrix of selected features; f_r is a trained regression model; $\hat{Y}_r$ is the output vector based on the regression model and the testing design matrix; Y is the real testing value vector.

2.2.2 Pure classification method

Assume the sparse variable is discrete and select a classification model (0/1) to predict the variable. For every value that is classified as 1, multiply the average testing value. Calculate

the RMSE. The pseudo code is shown as follows:

$$f_c = train(X_{train}), \tag{2.7}$$

$$\hat{Y}_c = f_c(X), \tag{2.8}$$

$$\hat{Y}_c(\hat{Y}_c == 1) = mean(Y_{train}), \tag{2.9}$$

$$RMSE_c = ||\hat{Y}_c - Y||_2, \tag{2.10}$$

where X_{train} is the training design matrix of selected features; X is the testing design matrix of selected features; train(*) represents the training action based on *; f_c is a trained classification model; $\hat{Y}_c$ is the output vector based on the classification model and the testing design matrix; Y_{train} is the training value vector; Y is the real testing value vector. Note that the equations from (2.4) to (2.25) represent assigning the value of right side term to the left side term. And the expression $Y(Y == 1)$ indicates all elements in Y whose values are 1. Thus for example, Eqn.(2.9) means: a) Find all elements in $\hat{Y}_c$ whose values are 1; b) Assign the mean value of Y_{train} to all the elements found in a). The reason to use the mean value of Y_{train} as our predicting results instead of the predicted "1" is that we want to lower the RMSE as much as possible, and the mean value of Y_{train} is a much better choice than the original predicted "1".

2.2.3 Classification-Regression framework type I

We select a classification model to predict the variable, and we also select a regression model to predict the variable. Then, for every value that is classified as 1, equate it to the corresponding prediction in the regression model. Calculate the RMSE. The pseudo code is shown as follows:

$$f_c = train(X_{train}), \tag{2.11}$$

$$f_r = train(X_{train}), \tag{2.12}$$

$$\hat{Y}_c = f_c(X), \tag{2.13}$$

$$\hat{Y}_r = f_r(X), \tag{2.14}$$

$$[\hat{Y}_c(\hat{Y}_c == 1)] = [\hat{Y}_r(\hat{Y}_c == 1)], \tag{2.15}$$

$$\hat{Y}_{CR-1} = \hat{Y}_c, \tag{2.16}$$

$$RMSE_{CR-1} = ||\hat{Y}_{CR-1} - Y||_2, \tag{2.17}$$

where X_{train} is the training design matrix of selected features; X is the testing design matrix of selected features; f_c is a trained classification model; f_r is a trained regression model; $\hat{Y}_c$ is the output vector based on the classification model and the testing design matrix; $\hat{Y}_r$ is the output vector based on the regression model and the testing design matrix; $\hat{Y}_{CR-1}$ is the final prediction vector under this framework; Y is the real testing value vector. Remind that Eqn.(2.15) indicates: a) Find all the elements in $\hat{Y}_r$ whose positions corresponding to the elements in $\hat{Y}_c$ whose values are 1. b) Find all elements in $\hat{Y}_c$ whose values are 1. c) Assign the values of found elements in a) to the found elements in b), one-to-one correspondingly.

2.2.4 Classification-Regression framework type II

We select a classification model to predict the variable, based on the whole training design matrix. Next, we extract the observations in the training design matrix where the corresponding output is classified as 1. Those extracted observations are combined as a new design matrix, We then train a regression model based on this new design matrix to predict the variable values. These values from regression model are used as the prediction for the output classified as 1. The pseudo code is shown as follows:

$$f_c = train(X_{train}), \tag{2.18}$$

$$\hat{Y}_c = f_c(X), \tag{2.19}$$

$$X_{new} = X_{train}[Y_c == 1], \tag{2.20}$$

$$f_r = train(X_{new}), \tag{2.21}$$

$$\hat{Y}_r = f_r(X), \tag{2.22}$$

$$\hat{Y}_c(\hat{Y}_c == 1) = \hat{Y}_r, \tag{2.23}$$

$$\hat{Y}_{CR-2} = \hat{Y}_c, \tag{2.24}$$

$$RMSE_{CR-2} = ||\hat{Y}_{CR-2} - Y||_2, \tag{2.25}$$

where X_{train} is the training design matrix of selected features; X is the testing design matrix of selected features; X_{new} is the new design matrix (only contains extracted observations); f_c is a trained classification model; f_r is a trained regression model; $\hat{Y}_c$ is the output vector based on the classification model and the testing design matrix; $\hat{Y}_r$ is the output vector based on the regression model and the testing design matrix; $\hat{Y}_{CR-2}$ is the final prediction vector under this framework; Y is the real testing value vector.

Among the four methods, the first and second method are the ordinary regression and classification model, except that we multiply the average training output to the non-zero prediction in the classification model. They are the widely-used methods that are applied to predict sparse variables. However, both of them have the drawbacks: for the pure regression method, it treats the sparse variable as the continuous variable, which ignores the internal property of sparse variable, and the prediction result is not a sparse vector. This violates the goal to produce a sparse prediction vector. For the classification method, although it can provide a sparse prediction, the RMSE is expected to be high due to the assumed discreteness.

To alleviate these drawbacks, in this thesis, the aforementioned two types of Classification-Regression framework are proposed. It can be seen that the difference between type I and type II is how to train the regression model: type I uses the whole design matrix, while type II only uses part of the design matrix, whose observations corresponds to the output classified as label 1. We will evaluate and compare the performance of these two different types in this next section.

2.3 Selected classification and regression models

In the proposed framework, we need to choose a classification model and a regression model. In this thesis, we select two popular classification models: logistic regression and Adaboost; and one regression model: XGBoost. These models are explained briefly in this section.

2.3.1 Logistic regression

Logistic regression is one of generalized linear models (GLM), designed for classification. The random structure is:

$$y_i \sim Bernoulli(p_i), \tag{2.26}$$

[i.e. $Pr(y_i = 1|x_i,\beta) = p_i$]. and the systematic structure is:

$$logit(E(y_i)) = logit(p_i) = \log(\frac{p_i}{1-p_i}) = s_i = x_i^T\beta, \tag{2.27}$$

$$p_i = sigmoid(s_i) = \frac{e^{s_i}}{1+e^{s_i}} = \frac{1}{1+e^{-s_i}}, \tag{2.28}$$

where logit() is the logit function, also the link function in this GLM; p_i is the probability; x_i is the predictor of ith observation. β is the coefficient vector to be determined. In this thesis, we apply the deterministic version of logistic regression:

$$y_i = round(p_i), \tag{2.29}$$

round() is a function: round(x)=0 if x is in [0,0.5); round(x)=1 if x is in [0.5,1].

To compute β, we optimize the following loss function:

$$\max \prod_{i=1}^{n} Pr(y_i|x_i,\beta), \tag{2.30}$$

It is equivalent to:

$$\max \prod_{i=1}^{n} \frac{e^{y_i s_i}}{1+e^{s_i}} \tag{2.31}$$

Take the log:

$$\max \sum_{i=1}^{n} \log(Pr(y_i|x_i,\beta)) = \sum_{i=1}^{n} [y_i s_i - \log(1+e^{s_i})], \tag{2.32}$$

It is equivalent to:

$$\min -\sum_{i=1}^{n} [y_i s_i - \log(1+e^{s_i})]. \tag{2.33}$$

This is a convex optimization problem that can be easily solved. Also, the above logistic regression is the binary version, and it can be extended to multiclass classification. But in this thesis, it is enough to adopt the binary version.

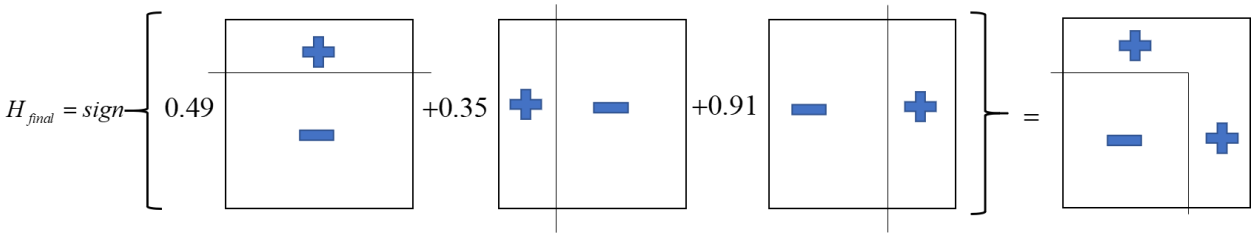

Figure 2.3: An illustrative diagram of Adaboost

2.3.2 Adaboost

Adaboost belongs to boosting machine for classification. It consists of many weak classifiers $h_k(x_i)$. The final classification result is a perceptron based on the weighted results of weak classifiers:

$$y_i = sign(\sum_{k=1}^{d} \beta_k h_k(x_i)), \tag{2.34}$$

where β_k is the weight of vote of classifier h_k.

How to add weak classifiers is the central question in Adaboost. Typically, each iteration adds a new weak classifier, and then assigns a bigger weight to the observation that is falsely classified. Then by adding a new classifier in the next iteration, we try to minimize our misclassifications. The final classifier is the linear combination of the selected classifiers. Figure 2.3 shows an example of this procedure.

To find the weak classifier h_k in each iteration. We first define the classifier at mth iteration:

$$F_m(x_i) = F_{m-1}(x_i) + \beta_m h_m(x_i). \tag{2.35}$$

Use exponential loss function:

$$L(h_m, \beta_m) = \sum_{i=1}^{n} \exp[-y_i(F_{m-1}(x_i)) + \beta_m h_m(x_i)] \propto \sum_{i=1}^{n} D_i \exp[-\beta_m y_i h_m(x_i)], \tag{2.36}$$

where $D_i \propto \exp[-(y_i F_{m-1}(x_i))]$. Since $y_i h_m(x_i)$ can only be +1 or -1, we simplify the loss function as:

$$L(h_m, \beta_m) = \sum_{i:h_m(x_i)=y_i} D_i e^{-\beta_m} + \sum_{i:h_m(x_i)\neq y_i} D_i e^{\beta_m} = (1-\varepsilon)e^{-\beta_m} + \varepsilon e^{\beta_m}, \tag{2.37}$$

where $\varepsilon = \sum_{i:h_m(x_i)\neq y_i} D_i$ is the error of h_m on the reweighted dataset. Also, it can be shown that minimizing $L(h_m, \beta_m)$ is equivalent to minimizing the error ε. Thus, we can choose a new weak classifier h_m by minimizing ε. And for a fixed h_m, its weight β_m is calculated by equating the two parts in $L(h_m, \beta_m)$ [i.e. this is to minimize $L(h_m, \beta_m)$]:

$$(1-\varepsilon)e^{-\beta_m} = \varepsilon e^{\beta_m}, \tag{2.38}$$

which leads to:

$$\beta_m = \frac{1}{2}\log(\frac{1-\varepsilon}{\varepsilon}). \tag{2.39}$$

2.3.3 XGBoost (XGB)

XGBboost is a gradient boosting machine but more principled. It expands the gradient boosting loss function by the second order Taylor expansion. Let the current function be $F_{m-1}(x)$:

$$F_{m-1}(x) = \sum_{k=1}^{m-1} h_k(x), \tag{2.40}$$

where $h_k(x)$ is the base function (classification and regression tree). We want to learn a new tree in the next iteration:

$$F_m(x) = F_{m-1}(x) + h_m(x), \tag{2.41}$$

The loss function of regression is:

$$L = \sum_{i=1}^{n} [y_i - (F_{m-1}(x_i) + h_m(x_i))]^2 = \sum_{i=1}^{n} (r_i - h_m(x_i))^2, \tag{2.42}$$

where r_i is the residual error for observation i. The gradient boosting is aimed at minimizing L to find the new $h_m(x)$. XGBoost further expand L by the Taylor expansion:

$$s_i = F_{m-1}(x_i) + h_m(x_i), \tag{2.43}$$

$$L(s_i, y_i) = (y_i - s_i)^2. \tag{2.44}$$

$$L(s_i, y_i) \approx L(\hat{s}_i, y_i) + L^{'}(\hat{s}_i, y_i)h_m(x_i) + \frac{1}{2}L^{''}(\hat{s}_i, y_i)h_m(x_i)^2, \tag{2.45}$$

where:

$$L^{'}(\hat{s}_i, y_i) = \frac{\partial}{\partial s}L(s_i, y_i), \tag{2.46}$$

$$L^{''}(\hat{s}_i, y_i) = \frac{\partial^2}{\partial s^2} L(s_i, y_i). \tag{2.47}$$

Let $r_i = -L^{'}(\hat{s}_i, y_i)$ and $w_i = L^{''}(\hat{s}_i, y_i)$. Then we can rewrite:

$$L(s_i, y_i) = \frac{w_i}{2}[\tilde{y}_i - h_m(x_i)]^2 + c, \tag{2.48}$$

where $\tilde{y}_i = r_i/w_i$ and c is a constant. We can minimize Eqn. (2.48) to find $h_m(x)$ by weighted least square method. Also, compared loss function (2.42) and (2.48), we see that XGBoost is typically faster than gradient boosting because it accounts for the curvature w_i.

CHAPTER 3

Case study

In this section, we will test our proposed frameworks by predicting two sparse variables. One is taxi tips ratio, the other is daily precipitation volume. Through these tests, we compare the performance of our proposed frameworks with the pure regression model and the pure classification model. RMSE is used as the criteria.

3.1 Taxi tips ratio

Taxi tips ratio is defined as the tips amount divided by the total trip fare. It represents the willingness of passengers to give tips, and it is very meaningful to the taxi drivers and taxi companies. Thus, how to predict taxi tips ratio based on some features is an interesting question. Since giving tips is not mandatory, many passengers would choose not to give tips, which will lead to a high weight on zero. If the tip ratio is not zero, it can be any positive value less or equal to one. Thus, taxi tips ratio can be regarded as a sparse variable.

3.1.1 Data description

In this study, we use data from the New York City Taxi commission about Green Taxis in Sep, 2015. The data are public at NYC government website and can be obtained at:
https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page.
There are 1.5 million records in total. To expiate the algorithms, we only use the first 500,000 observations to run algorithms.

In one observation, the features (predictors) include: Passenger count, Trip distance,

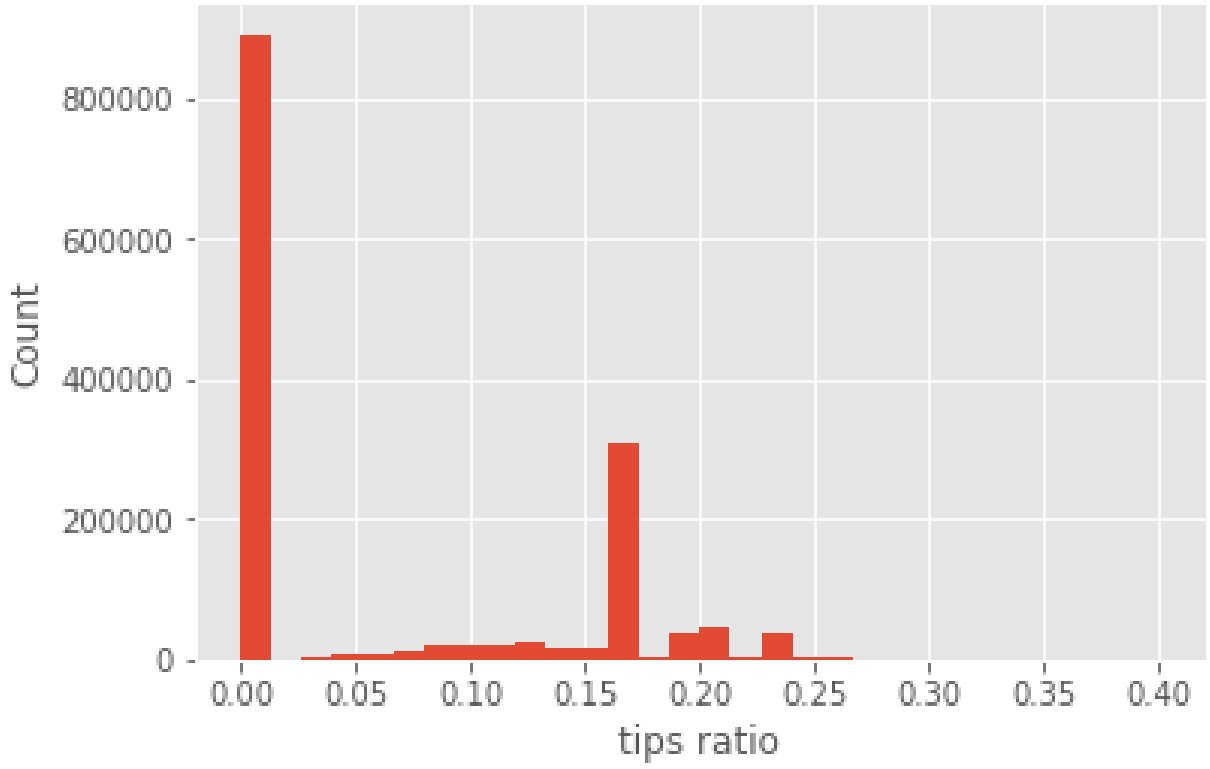

Figure 3.1: A histogram of tips ratio (sparse variable)

Fare amount, Extra, MTA tax, Tolls amount, and Improvement surcharge. The response is taxi tips ratio. We plot the histogram of the tips ratio (the sparse variable) to show its sparsity (i.e. most values are zero) in Figure 3.1 as an example. We also plot the histograms of two important features (predictors): trip distances and total fare amount behind, from which we can see that the features are not necessarily sparse.

3.1.2 Lasso for feature selection

Before running any predicting models, we need to first select effective predictors by Lasso regression introduced above. We first conduct the trace method to determine the Lasso parameter λ. The trace plot is shown in Figure 3.2. We can see in Figure 3.4 that only three features have non-zero coefficients when λ approaches to 0.001. To confirm this value, we conduct a 10-folds cross validation to find the best λ. The CV results indicate that $\lambda_{best} = 0.001$, which validates Figure 3.4. So, under $\lambda_{best} = 0.001$, we select the following

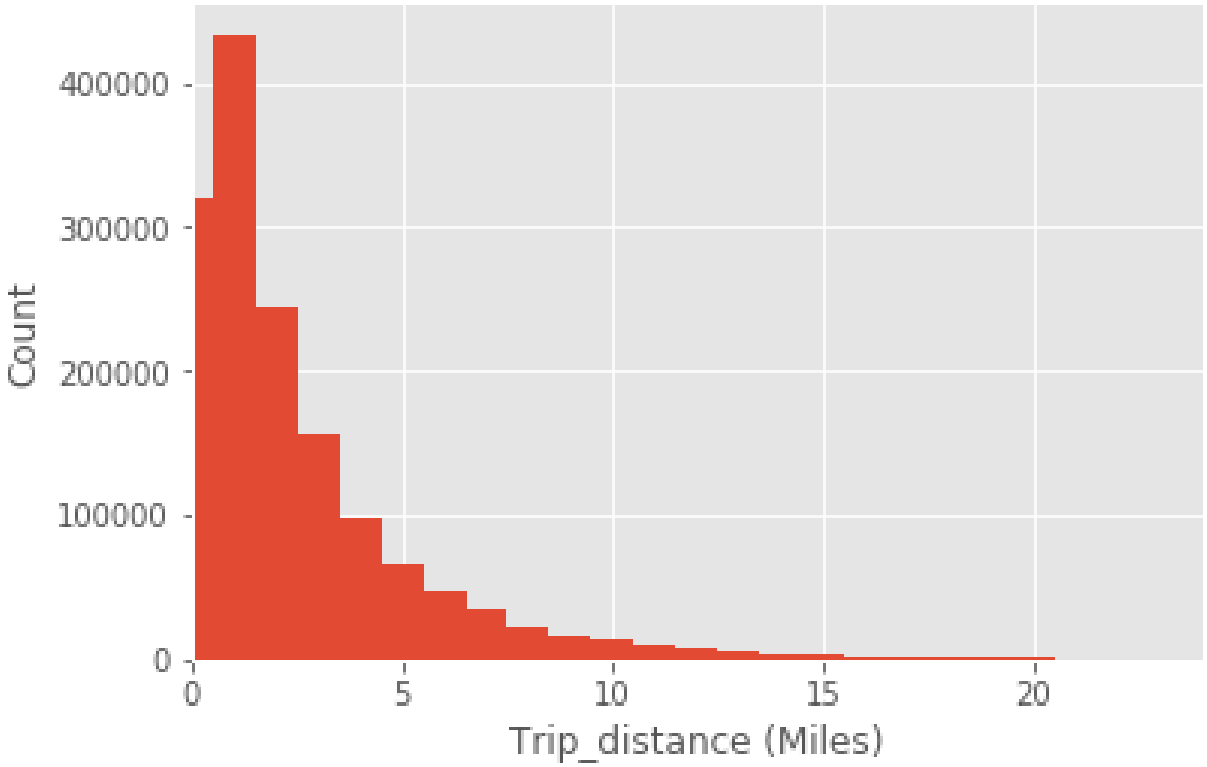

Figure 3.2: A histogram of trip distance (feature)

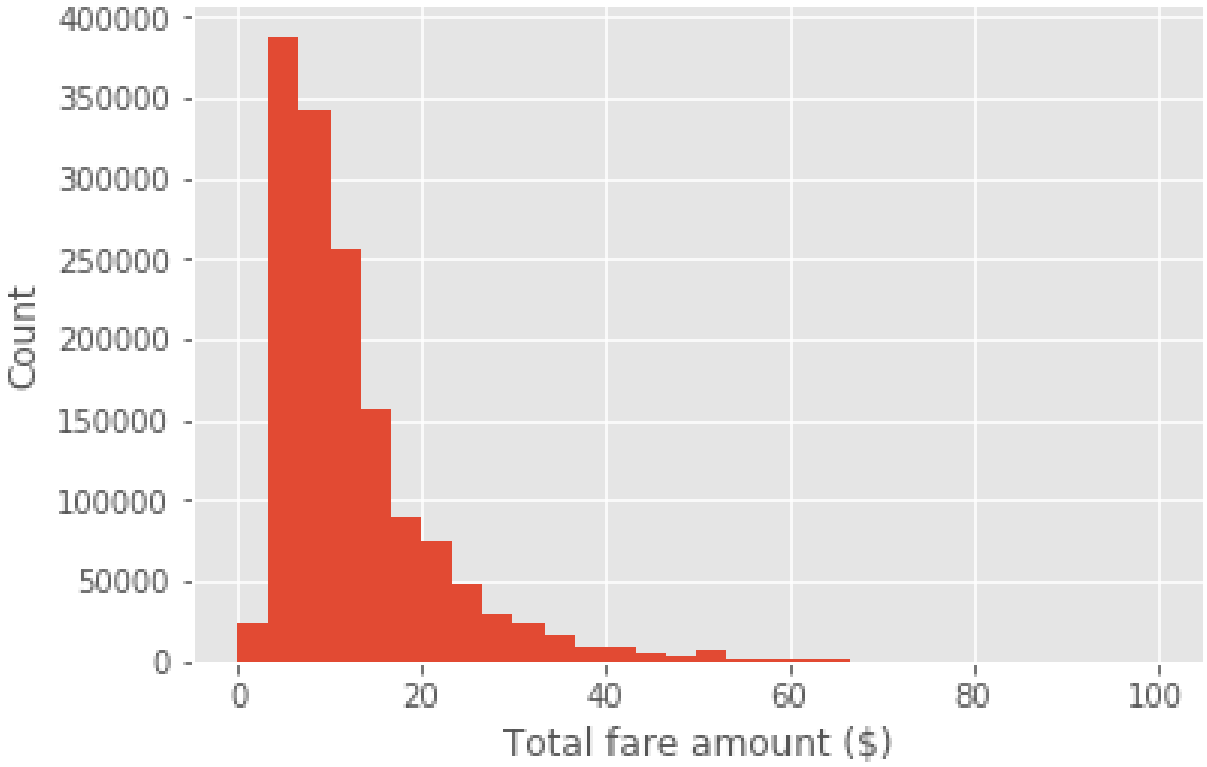

Figure 3.3: A histogram of total fare amount (feature)

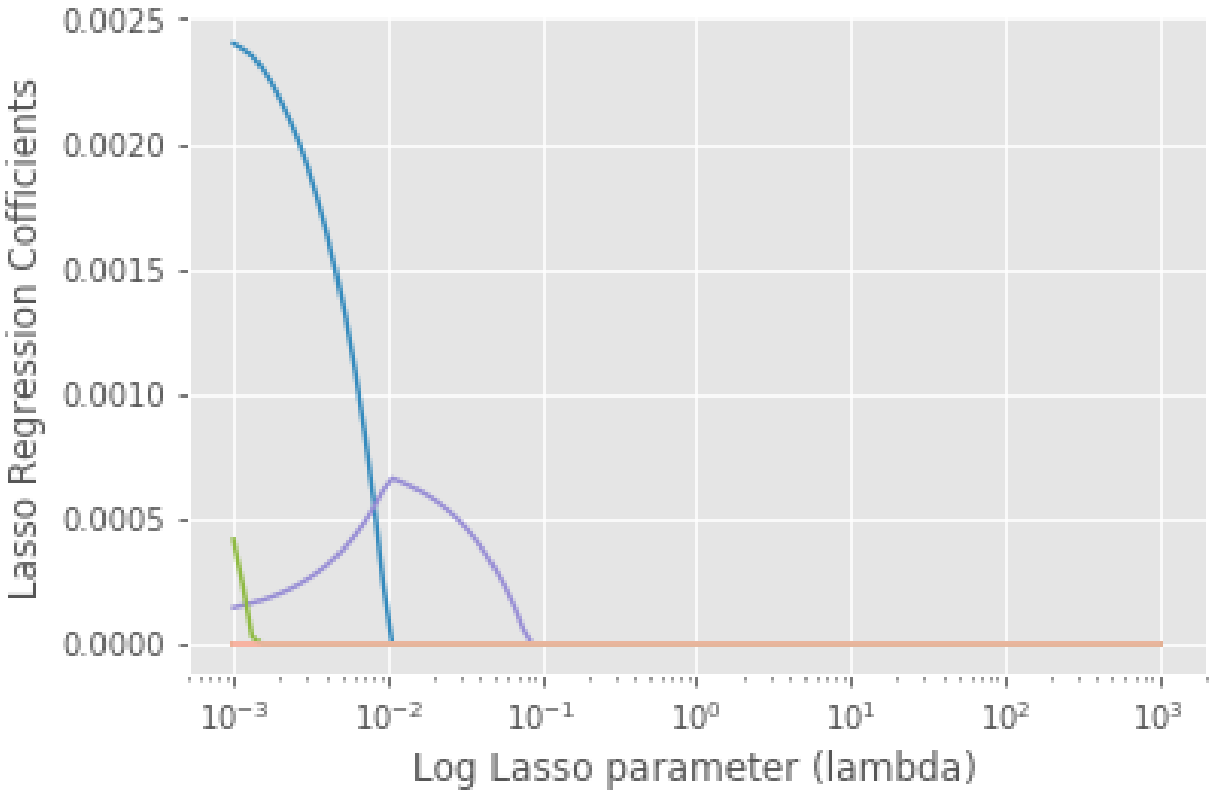

Figure 3.4: Trace plot for Lasso parameter of tips ratio

three predictors: Trip distance, Fare amount, and Tolls amount, whose lasso regression coefficients are non-zero. The design matrix used in the following models are based on these three predictors.

3.1.3 Predicting performance of proposed method

In this subsection, we compare the four aforementioned methods for predicting sparse variables: 1.Pure regression model; 2.Pure classification model; 3.Classification-Regression framework type I; 4.Classification-Regression framework type II. Logistic regression and XGBoost are selected as the pure classification and pure regression model, respectively. For the Classification-Regression framework, we use both logistic regression and Adaboost as the classification models, and the regression model is still XGBoost. The hyperparameters in each model are determined by fine-tuning. RMSE is used as the criteria and the results are shown in Table 3.1. Note that the RMSE in this chapter is calculated based on train-test split method, with the testing data ratio equal to 0.2.

Table 3.1: RMSE results of different frameworks for predicting tips ratio

Proposed frameworks	**RMSE**
1. Pure XGB	0.0871
2. Pure Logistric	0.1059
3. Logistic + XGB framework type I	0.1057
4. Adaboost + XGB framework type I	0.1034
5. Logistic + XGB framework type II	0.1081
6. Adaboost + XGB framework type II	0.1075

It can be seen from Table 3.1 that: First, except the pure regression model, all the rest five frameworks can provide spare predictions. Thus. although the pure regression (XGB) model has the least RMSE, its non-sparse predictions prevent it from being the best choice. The pure classification (Logistic) model behaves better than Type II classification-regression framework, but not as good as Type I classification-regression framework, which indicates

that type I is generally better than type II. Compare experiments 3 and 4, 5 and 6, we see that using Adaboost instead of logistic regression as the classification model in the proposed framework works better, regardless of in type I or II. This might be because that Adaboost has the stronger classification power.

In sum, to obtain sparse prediction results with the lowest RMSE, I suggest use method 4: (Adaboost + XGB framework type I) to predict taxi tips ratio in this section.

3.2 Daily precipitation volume

Daily precipitation volume is an important index in hydrology. It represents the total rainfall amount for a given day, typically with the unit of inches or mm. If daily precipitation volume can be predicted based on some features, the water resources can be deployed accordingly, which will maximize the irrigation benefits and minimize the flood risk. Thus, it is our desire to find a way to predict daily precipitation volume. Since for most regions, the rainfall does not appear on a daily basis, so it is obvious that daily precipitation volume can be regarded as a sparse variable.

3.2.1 Data description

In this study, we use the hydro data in Hobbs basin. Hobbs basin (USGS 01104430 HOBBS BK BELOW CAMBRIDGE RES NR KENDALL GREEN, MA) is in the Middlesex County, Massachusetts, USA, with Latitude $42^{23'}53"$ and Longitude $71^{16'}26"$. Figure 3.5 is a location map showing its relative position. The drainage area is 6.86 square miles, and the contributing drainage area is also 6.86 square miles, which indicates it is not a big basin. We collect the hydro data for Hobbs basin from 9/30/2006-1/1/2019, which is available at USGS official website: *https* : *//waterdata.usgs.gov/* (ID: 01104430). The testing data ratio is still 0.2.

Figure 3.5: A location map of Hobbs basin

In one observation, there are 11 features (predictors), including: Average reservoir storage, Average discharge, Maximum water temperature, Minimum water temperature, Average water temperature, Maximum specific conductance, Minimum specific conductance, Average specific conductance, Maximum air temperature, Minimum air temperature, and Average air temperature. We plot the histogram of the response variable (i.e daily precipitation volume) to show its sparsity (i.e. most values are zero) in Figure 3.6. The important features (average reservoir storage and average water temperature) are also plotted to characterize the watershed as follows.

3.2.2 Lasso for feature selection

As discussed, we need to first select effective predictors by Lasso regression. The trace plot is shown in Figure 3.5 to determine the Lasso parameter λ. We can see in Figure 3.9 that most regression coefficients are non-zero under a larger range of λ value, which prompts us to conduct a 10-folds cross validation to find the best λ. The CV results indicate that $\lambda_{best} = 0.001$. So, under $\lambda_{best} = 0.001$, we select the following three predictors: Maximum

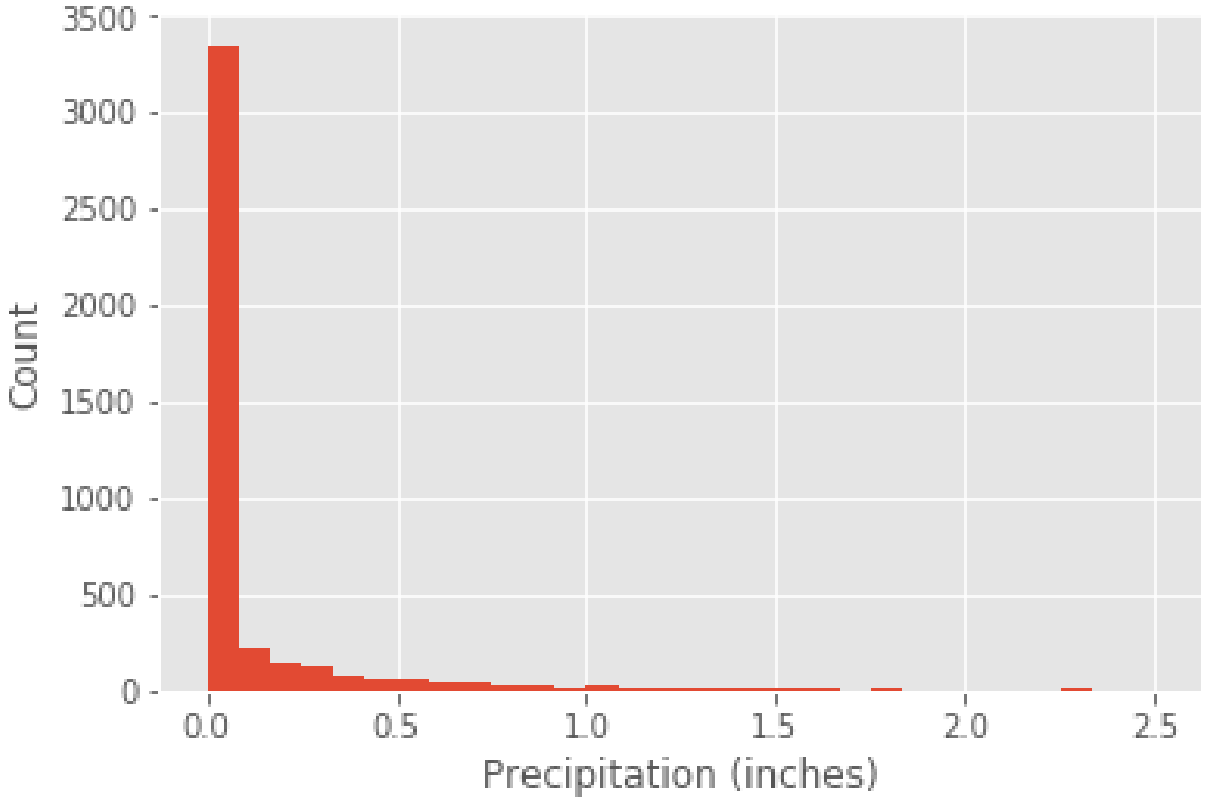

Figure 3.6: A histogram of daily precipitation

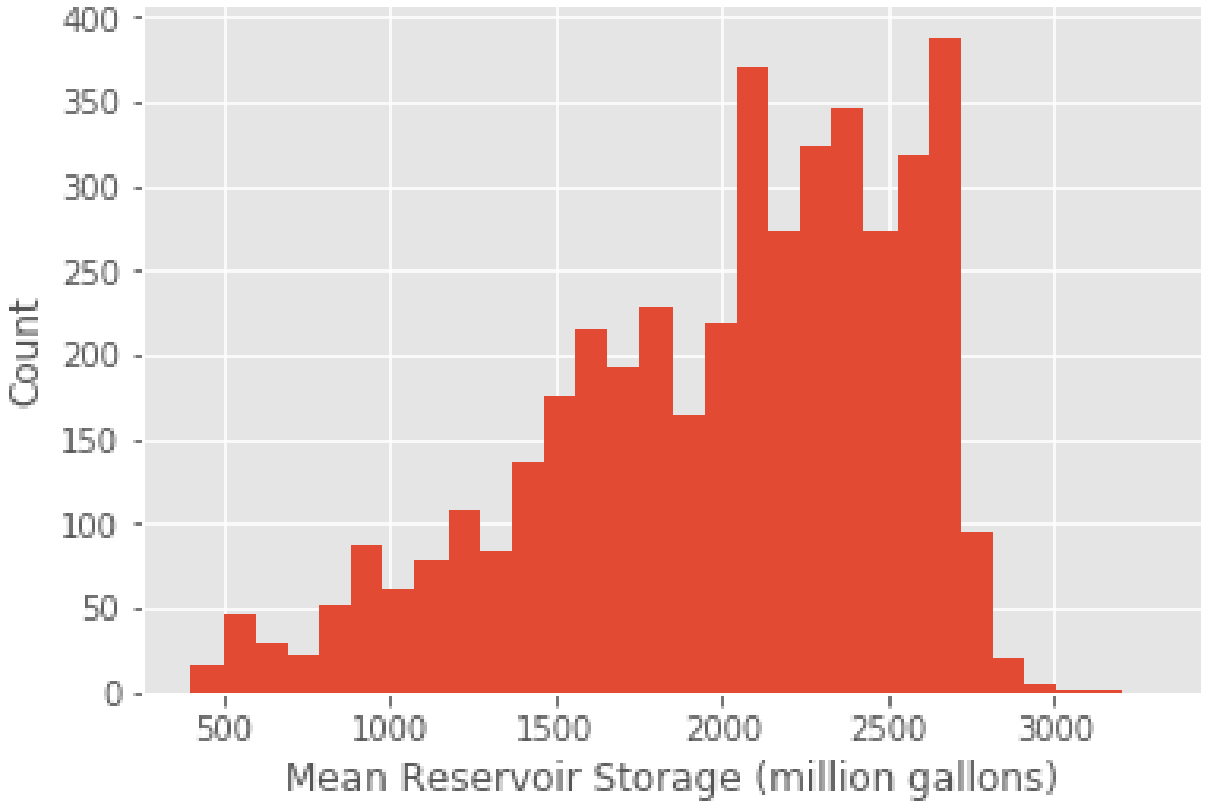

Figure 3.7: A histogram of mean reservoir storage

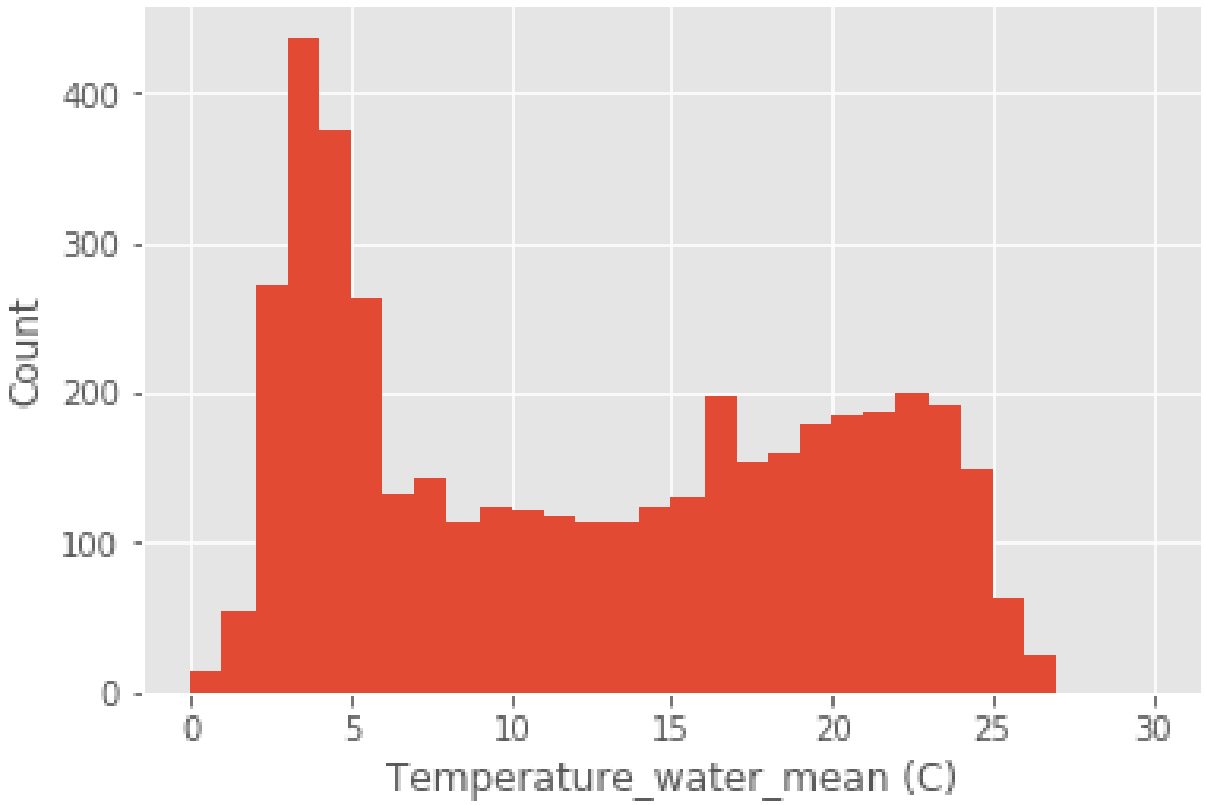

Figure 3.8: A histogram of mean water temperature

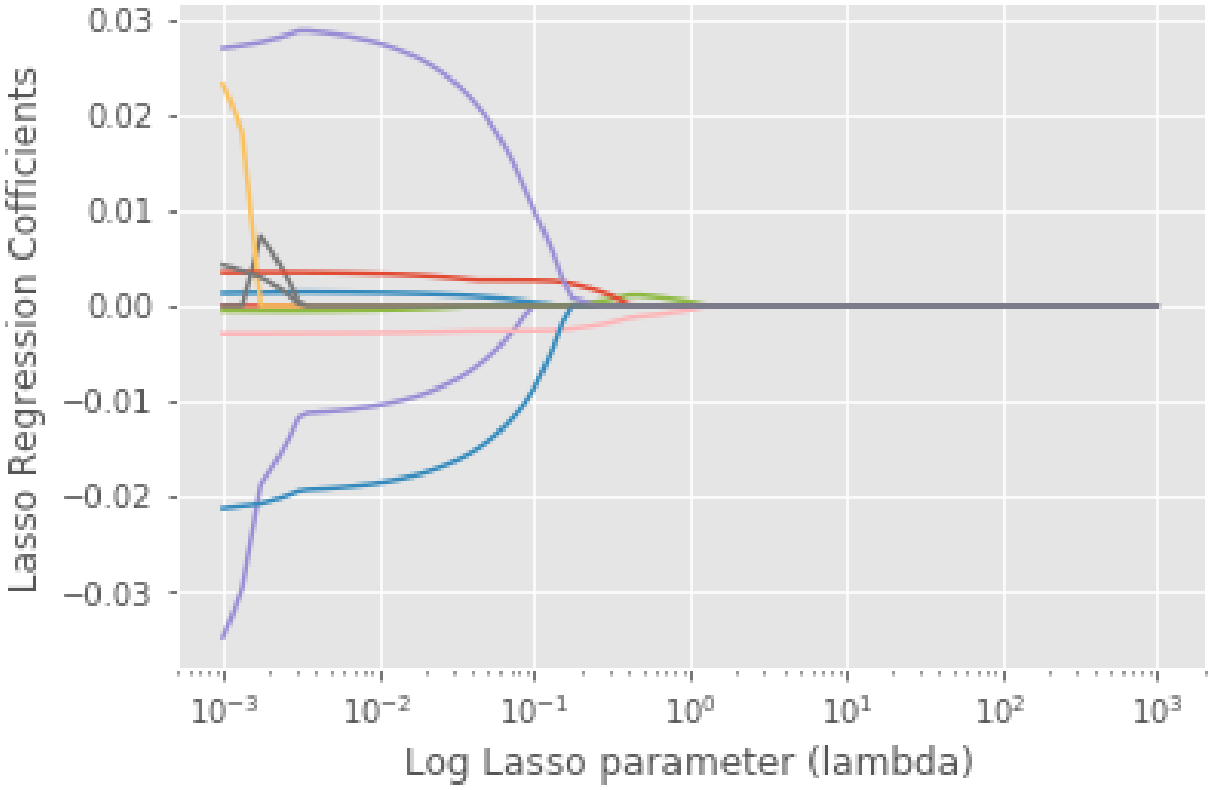

Figure 3.9: Trace plot for Lasso parameter of daily precipitation volume

water temperature, Average water temperature, Maximum air temperature, and Minimum air temperature, whose lasso regression coefficients are non-zero and significant. The design matrix used in the following models are based on these four predictors.

3.2.3 Predicting performance of proposed method

We compare the four aforementioned methods for predicting sparse variables: 1.Pure regression model; 2.Pure classification model; 3.Classification-Regression framework type I; 4.Classification-Regression framework type II. The selected classification models, regression models and other settings are the same with section 3.1.3. RMSE is used as the criteria and the results are shown in Table 3.2.

Table 3.2: RMSE results of different frameworks for predicting daily precipitation volume

Proposed frameworks	**RMSE**
1. Pure XGB	0.3011
2. Pure Logistric	0.3222
3. Logistic + XGB framework type I	0.3090
4. Adaboost + XGB framework type I	0.3126
5. Logistic + XGB framework type II	0.3165
6. Adaboost + XGB framework type II	0.3182

It can be seen from Table 3.2 that: The pure regression (XGB) model still has the least RMSE. But as we discussed, its non-sparse predictions prevent it from being the legal choice. The pure classification (Logistic) model, in this experiment, behaves as the worst. Type I framework outperforms Type II framework according to the comparison between experiments 3 and 5, 4 and 6. And this time, using logistic regression instead of Adaboost as the classification model in the framework works better.

In sum, to obtain sparse prediction results with the lowest RMSE, I suggest use method 3: (Logistic regression + XGB framework Type I) to predict daily precipitation volume in

this section. This is due to the fact that method 3 can not only provide sparse predictions, but also has the minimum RMSE.

CHAPTER 4

Discussion and conclusion

In this thesis, I describe sparse variables that have non-zero probability weights on zero. Four methods are discussed to predict sparse variables: 1.Pure regression model; 2.Pure classification model; 3.Classification-Regression framework type I; 4.Classification-Regression framework type II. Several widely used classification and regression models are introduced, such as: logistic regression, Adaboost, and XGBoost. Lasso regression is also elaborated for feature selection.

In the case studies, I select two sparse variables as examples: Taxi tips ratio and daily precipitation volume. By analyzing the results, it is found that Classification-Regression framework type I is the best method to predict sparse variables like taxi tips ratio and daily precipitation volume, since it obtains sparse prediction results with the lowest RMSE, for both examples. While the best framework is determined (i.e. type I), the optimal regression and classification models in this framework are flexible, which seems to depend on the particular problem (i.e. different sparse variables to be predicted).

REFERENCES

[1] Cortes, C. and Vapnik, V., 1995. *Support-vector networks. Machine learning.* 20(3), pp.273-297.

[2] Chen, T. and Guestrin, C., 2016. *Xgboost: A scalable tree boosting system.* In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794). ACM.

[3] Freund, Y. and Schapire, R.E., 1997. *A decision-theoretic generalization of on-line learning and an application to boosting.* Journal of computer and system sciences, 55(1), pp.119-139.

[4] Hotelling, H., 1933. *Analysis of a complex of statistical variables into principal components.* Journal of educational psychology, 24(6), p.417.

[5] Hoerl, A.E. and Kennard, R.W., 1970. *Ridge regression: Biased estimation for nonorthogonal problems.* Technometrics, 12(1), pp.55-67.

[6] Hyvrinen, A. and Oja, E., 2000. *Independent component analysis: algorithms and applications.* Neural networks, 13(4-5), pp.411-430.

[7] McCulloch, W.S. and Pitts, W., 1943. *A logical calculus of the ideas immanent in nervous activity.* The bulletin of mathematical biophysics, 5(4), pp.115-133.

[8] Murphy, K.P., 2012. *Machine learning: a probabilistic perspective.* MIT press.

[9] Pearson, K., 1901. LIII., 1901. *On lines and planes of closest fit to systems of points in space.* The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 2(11), pp.559-572.

[10] Schapire, R.E., 1990. *The strength of weak learnability.* Machine learning, 5(2), pp.197-227.

[11] Sutton, R.S. and Barto, A.G., 2018. *Reinforcement learning: An introduction.* MIT press.

[12] Tibshirani, R., 1996. *Regression shrinkage and selection via the lasso.* Journal of the Royal Statistical Society: Series B (Methodological), 58(1), pp.267-288.

[13] Takeda, H., Farsiu, S. and Milanfar, P., 2006. *Kernel regression for image processing and reconstruction.* Doctoral dissertation, University of California, Santa Cruz.

[14] Walker, S.H. and Duncan, D.B., 1967. *Estimation of the probability of an event as a function of several independent variables.* Biometrika, 54(1-2), pp.167-1791967.

[15] Wu, Y., 2019. *A Note on Machine Learning Methods.* UCLA Statistics, Based on lectures.