Leveraging Classification Models for River Forecasting

ABSTRACT

River forecasting has become essential to human life, and is the foundation for critical societal problems like hydropower generation and disaster avoidance. Prior work in this field has focused on applying regression models to gage and discharge prediction since these are naturally continuous dynamical functions. On the other hand, with discretized data, classifiers can be adopted to solve this problem by predicting a conditional probability distribution. Predicting this distribution is important in at least two ways: (1) the variance of the distribution can indicate the confidence of the predicted expected values, and (2) the distribution can be used for computing the probability that the gage or discharge exceeds or falls below some threshold. This paper presents a concrete river forecasting framework with classifiers including probabilistic graphical models (PGMs) and artificial neural network classifiers (ANNCs), and compares them with regression models including linear and artificial neural network regression (ANNR). The proposed framework is applied on real data for the Guadalupe river basin (Texas) thereby enabling a detailed comparison among various manners of forecasting studied, along with a set of guidelines for their best use. To the best of our knowledge, we are the first to use classifiers for regression problems in river modeling and forecasting. Due to its higher resilience to noise and better performance at predicting distributions, the proposed forecasting framework with classification models can be applied for other types of time series predictions.

KEYWORDS

River forecasting, spatial-temporal modeling, classification and regression

1 INTRODUCTION

River forecasting is essential to human life and subsistence, especially for flood or drought prediction, ever since early agriculture has started to spread. More recently, the advent of hydropower generation, especially in the context of run-of-river hydropower projects [17], requires fine grain forecasting capabilities for potential energy availability. However, progress in forecasting river behavior has stalled, mainly due to several challenges coming from its application domain. First, the hydrologic eco-system is characterized by many inter-related factors with highly non-linear dynamical dependencies [15]. Second, the metrics used for assessing certain models must rely on their application domain. For example, to forecast the dynamical behavior of rivers, the expected value of future gage or discharge is produced as prediction, while for flood avoidance and hydropower availability, the probability that discharge exceeds or is below a certain threshold is needed. Third, data can be partially unavailable or sparse. For example, precipitations are not measured at some locations. Even for those locations where precipitation is measured, data may be missing due to malfunctioning sensors or failing communication. In this paper, the set of features considered include gage, discharge, and precipitation at one location. In the real examples we used in this paper, 30% of all features are missing and with those that are available, 2% of data are missing. Finally, the observed data are noisy, making complex forecasting models prone to overfitting.

Much attention in river forecasting was placed on regression models, ranging from linear models, to artificial neural network regression (ANNR) [1]. However, by quantizing the output variable to multiple discrete levels, the problem may be transformed to a classification problem, and the continuous predicted value can be computed based on the classification results. The advantages of using classification models are twofold: (1) Clustering continuous values to discrete levels can alleviate the measurement noise, thereby reducing the danger of overfitting. (2) Classifiers usually produce a probability distribution for the predicted variable instead of just an expected value. In our work, we implement the river forecasting problem on probabilistic graphical models (PGMs) and artificial neural network classifiers (ANNCs). A neighbor smoothing (NBS) strategy is proposed to address the inherent overfitting problem of PGMs due to their larger number of parameters.

The rest of paper is organized as follows. In section 2, we introduce prior work on river modeling and detail the contributions of this paper. In section 3, the Guadalupe river dataset is introduced. In section 4, regression and classification models and their detailed implementation are described. Finally, in section 5, we describe our experiment setup and results on Guadalupe river data and discuss and compare different models proposed in this paper.

2 RELATED WORK

Past decades have witnessed great progress in river modeling and forecasting. Approaches used can be divided into two types: (1) conceptual models and (2) black-box models. Conceptual models aim at simulating the physical processes and transforming inputs to outputs guided by prior knowledge of the natural systems [16]. Rainfall, river flow, evaporation, soil moisture, underground water, carbon cycling, and any other factors relevant to the hydrologic cycle, are incorporated to formulate the model for river gage and flow forecasting. The development of conceptual models started early and have achieved good performance over the years. An example is SWAT (Soil and Water Assessment Tool), formed by combining and improving many other well-performed river models, such as CREAMS, SWRRB, etc. [4].

Black-box approaches, on the other hand, rely more on data instead of knowledge of physical procedure. With much more data available, the data-driven black-box models are witnessing an increased popularity for river modeling. Before 1990s, traditional linear models including autoregressive integrated moving average (ARIMA) were most widely used [20]. Although modifications to linear models were then made to improve its performance, such as the seasonal ARIMA (SARIMA) model which incorporates seasonality for model construction, they can only capture linear relations

[1] For clarity, "ANNR" and "ANNC" are adopted to differentiate between ANNs used for regression and classification, respectively.

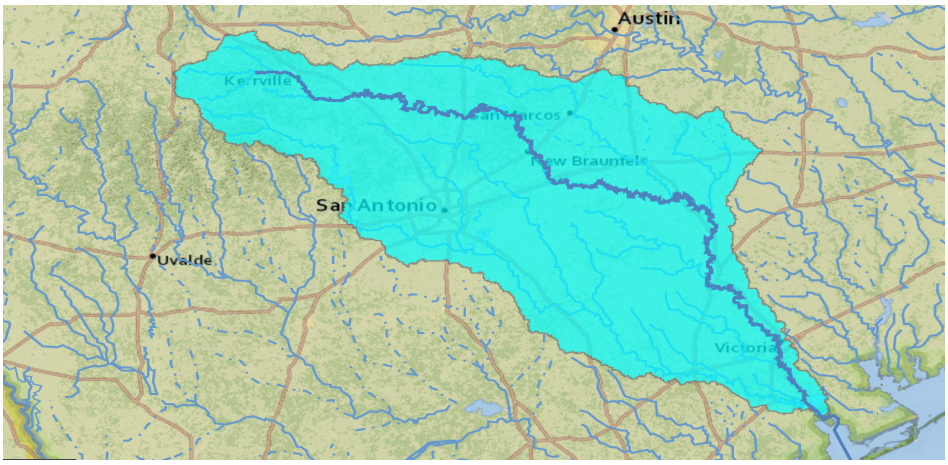

Figure 1: Guadalupe river basin is shown by the light blue area. The dark blue stream is the trace of main Guadalupe river. Figure generated from ArcGIS [3].

between predictors and predicted variables [27]. However, river flow forecasting is believed to be highly non-linear and not easily described by simple models [15]. In 1987, Karlsson *et al.* compared K-nearest neighbor models for rainfall-runoff modeling, but only found limited improvement in performance. Since 1995, a notable trend in research on river forecasting has been toward artificial neural network regression (ANNR) [15]-[5]. Hsu *et al.* used a neural network to model rainfall-runoff process, and achieved better performance than linear models [15]. Dawson *et al.* compared multi-layer perceptron (MLP) with radial basis function (RBF) network, and found MLP worked better [10]. Chau *et al.* leveraged genetic algorithms (GA) to initialize the weights of ANNR before training the model [8]. Asadi *et al.* proposed a hybrid ANNR by combining GA and Levenberg-Marquardt (LM) algorithm for learning feed forward neural networks [5].

As the predicted variables (e.g. water flow) are continuous, regression models have been naturally adopted by almost all the previous work, while little work was done in the realm of classification models for predicting a probability distribution. The idea of using classification models for regression problems can be dated back to 1984, when Breiman *et al.* performed inference using regression trees that partition the range of continuous variables into multiple sections [7]. However, this idea was explored by only a few researchers, since most forecasting problems are formulated as regression problems where the target is a single value, *i.e.*, the conditional expected value. River forecasting, however, is concerned with predicting both the expected value and the conditional probability distribution [18], which can be further used to answer questions like: What is the probability that the river gage exceeds the flooding level or falls below the drought level? Or what is the probability that the river discharge exceeds the threshold required by hydropower generation?

The contributions of this papers are summarized as follows:

- To the best of our knowledge, our work is the first to employ classification models including PGMs and ANNC for regression problems in river forecasting. This paper compares regression and classification models, as a guideline of model selection for river forecasting.
- We proposed a NBS strategy to address the overfitting problem of PGMs, thereby increasing their accuracy when used with limited training data.
- Instead of merely forecasting the expected value, we also predict a probability distribution of the target variable. Mean log-likelihood (MLL) is proposed to assess the quality of the predicted probability distribution.

3 DATASET

Guadalupe river (shown in Figure 1) is located in the southeast of U.S., contains abundant hydropower resources, and is also prone to fluctuations. Guadalupe basin has a 3256 km^2 catchment area. Its length is 370 km, starting from ($30°05'17''N$, $99°38'32''W$), and flowing into Gull of Mexico at ($28°24'07''N$, $96°46'57''W$). The average discharge of Guadalupe river is 34 m^3/s [2]. Data ranging from April to late July in 2016 are studied, as drastic fluctuations of river flow happen in this period, making it harder to forecast. The data, provided by United States Geological Survey (USGS) [1], include gages, discharges, and precipitations at 15-minute intervals. Therefore, each time series has 11520 time steps. Gages and discharges of six big nodes are predicted, four of which are joint nodes with parent nodes from multiple branches. They are selected because the prediction for joint nodes exhibits more challenges. Features of eleven nodes, including the six nodes to predict, are collected in total, serving as inputs for forecasting. These settings can scale to any number, and we are showing these joint nodes because they are potentially more suitable for hydropower generation due to the large discharges.

4 METHODOLOGY

In this section, we first provide an overview of river models by formulating the forecasting problem, and introducing regression and classification models for river modeling. We then show how two

classifiers, PGM and ANNC, can be used for the regression problems. Finally, we discuss metrics for assessing prediction performances.

4.1 Model Overview

4.1.1 Problem Formulation. We aim at forecasting two features characterizing river dynamics: gage and discharge. Gage is the water level, also called *water stage*. Discharge is the volume of water running per unit time, also called *water flow* or *runoff*. Along the river, there are several stations, also called *nodes*, measuring gage, discharge and precipitation at their own locations in real time. Suppose N nodes on a river are studied. We denote the gage at the n-th node ($n \in \{1, 2, ..., N\}$) as time series $G^{(n)} = \{G_1^{(n)}, G_2^{(n)}, ..., G_t^{(n)}, ...\}$, its discharge as $D^{(n)} = \{D_1^{(n)}, D_2^{(n)}, ..., D_t^{(n)}, ...\}$, and precipitation as $P^{(n)} = \{P_1^{(n)}, P_2^{(n)}, ..., P_t^{(n)}, ...\}$. Then river forecasting is formulated as follows:

- Input: $\{G^{(\mathbb{N})}_{t_c-t_h+1}, ..., G^{(\mathbb{N})}_{t_c}\}$, $\{D^{(\mathbb{N})}_{t_c-t_h+1}, ..., D^{(\mathbb{N})}_{t_c}\}$ and $\{P^{(\mathbb{N})}_{t_c-t_h+1}, ..., P^{(\mathbb{N})}_{t_c}\}$, where $\mathbb{N} = \{1, 2, ..., N\}$, t_c is current time, and t_h is history window size.
- Output: $\mathrm{E}(G^{(n)}_{t_c+t_l}|\mathcal{F}^{(\mathbb{N})}_{t_c})$ and $\mathrm{E}(D^{(n)}_{t_c+t_l}|\mathcal{F}^{(\mathbb{N})}_{t_c})$, where $n \in \mathbb{N}$, t_l is *lead time*, *i.e.* how long in the future we are predicting, and $\mathcal{F}^{(\mathbb{N})}_{t_c}$ is the accumulated information up to and including time t_c for nodes $\mathbb{N}$.

In this paper, the input features including gages, discharges and precipitations are called *predictors*, and the variables to predict are called *targets*. Due to the spatial dependencies introduced by river topology, the targets are only dependent of the predictors of neighboring nodes and the node itself. Therefore,

$$\begin{aligned} p(G^{(n)}_{t_c+t_l}|\mathcal{F}^{(\mathbb{N})}_{t_c}) &= p(G^{(n)}_{t_c+t_l}|\mathcal{F}^{(\mathbb{N}_n)}_{t_c}) \\ p(D^{(n)}_{t_c+t_l}|\mathcal{F}^{(\mathbb{N})}_{t_c}) &= p(D^{(n)}_{t_c+t_l}|\mathcal{F}^{(\mathbb{N}_n)}_{t_c}) \end{aligned} \tag{1}$$

where $\mathbb{N}_n \subset \mathbb{N}$ is the set of neighbors of node n and node n itself, and $p(.)$ is the probability density function (PDF).

For many applications such as flood forecasting and hydropower prediction, one needs to figure out the probability of the discharge being higher or lower than a certain threshold thr, *i.e.,* $P(D^{(n)}_{t_c+t_l} > thr|\mathcal{F}^{(\mathbb{N})}_{t_c})$ or $P(D^{(n)}_{t_c+t_l} < thr|\mathcal{F}^{(\mathbb{N})}_{t_c})$. Therefore, an ideal model should not only produce $\mathrm{E}(G^{(n)}_{t_c+t_l}|\mathcal{F}^{(\mathbb{N})}_{t_c})$ and $\mathrm{E}(D^{(n)}_{t_c+t_l}|\mathcal{F}^{(\mathbb{N})}_{t_c})$, but also estimate $p(G^{(n)}_{t_c+t_l}|\mathcal{F}^{(\mathbb{N})}_{t_c})$ and $p(D^{(n)}_{t_c+t_l}|\mathcal{F}^{(\mathbb{N})}_{t_c})$, the PDF of target variables.

4.1.2 Regression Models. Regression models typically have the following form:

$$y = f(\vec{x}) + \epsilon, \tag{2}$$

where $\vec{x}$ is the predictor vector, y is the target with continuous values, $f(\cdot)$ is the model function and ϵ is the error. Since gages and discharges are both continuous variables, they can be naturally predicted by regression models. These can be divided into (1) linear models where the output is a linear combination of predictors, and (2) non-linear models, such as ANNRs.

All previously used regression models [15][10][8][5] minimize the least square error $\sum_i (y_i - f(\vec{x}_i))^2$, where $(\vec{x}_i, y_i)$ is the i-th training instance. This is equivalent to maximum likelihood estimation only when the distribution of noise is *i.i.d. normal*. Thus, the probability that the gage will exceed flooding threshold thr, is estimated as $\hat{P}(y > thr|\vec{x}) = max\{\alpha : thr \le f(\vec{x}) + \hat{\sigma}\Phi^{-1}(1-\alpha)\}$, where $\hat{\sigma}$ is the estimated standard deviation of ϵ, and $\Phi(\cdot)$ is the cumulative distribution function (CDF) of the standard normal distribution. However, this approach does not hold under heteroscedasticity where ϵ is time-variant or input-dependent [12].

4.1.3 Classification Models. Classifiers solve the problem:

$$argmax_{y \in \mathbb{C}}\ P(y^{(d)}|\vec{x}), \tag{3}$$

where $\vec{x}$ is the predictor, $y^{(d)}$ is a discrete variable denoting a class, the superscript d indicates that the target y is a discrete variable, and $\mathbb{C}$ is the set of classes.

A regression problem can be transformed into a classification problem by discretizing the continuous target variable y into multiple levels. Let us denote the number of levels as K. Then, the range of y is split into K bins, with K centroids. Typical discretization approaches include [28]:

- Linear discretization. The range of y is uniformly divided into K bins where the median of each bin is a centroid. This may cause imbalance of different bins.
- Density-based discretization. Assuming $\frac{1}{K}, ..., \frac{K-1}{K}$ quantiles of y as bin boundaries, each bin contains the same amount of data and centroids are medians of bins.
- K-means algorithm. By clustering data into K bins, K-means algorithm is flexible to data with different characteristics. The centroid of each bin is the mean of all data clustered in this bin. [13]

Classification models produce an estimation of $P(y^{(d)}|\vec{x})$. Discriminative models, including ANNC, Multinomial Logistic Regression (MLR) and Random Forest (RF), first compute K scores for the K classes, and then output the class with the highest score. The normalized K scores estimate $P(y^{(d)}|\vec{x})$ over $y^{(d)} \in \mathbb{C}$ where $|\mathbb{C}| = K$. Generative models including PGMs first compute $\hat{P}(y^{(d)}, \vec{x})$ for all $y^{(d)} \in \mathbb{C}$, and then output $argmax_{y^{(d)} \in \mathbb{C}}\ \hat{P}(y^{(d)}, \vec{x})$. Since $\hat{P}(y^{(d)}|\vec{x}) = \frac{\hat{P}(y^{(d)}, \vec{x})}{P(\vec{x})}$, the normalized $\hat{P}(y^{(d)}, \vec{x})$ is an estimation of $P(y^{(d)}|\vec{x})$.

To predict $\mathrm{E}(y|\vec{x})$, instead of simply using the centroid of a bin [28][6], we leverage the estimated $\hat{P}(y^{(d)}|\vec{x})$, and compute the weighted average of bin centroids:

$$\hat{\mathrm{E}}(y|\vec{x}) = \sum_{i=1}^{K} \hat{P}(y^{(d)} = i|\vec{x}) B_i, \tag{4}$$

where B_i is the centroid of the i-th bin.

In the Appendix, we prove that under certain assumptions the risk of the classification models, *i.e.*, its error compared to the true regression function, is bounded by:

$$\mathbb{E}|\hat{\mathrm{E}}(y|\vec{x}) - \mathrm{E}(y|\vec{x})|^2 \le \frac{1}{K^2} + \frac{K}{n} \tag{5}$$

where K is the number of discrete levels, and n is the number of training data.

To predict the $p(y|\vec{x})$, two approaches are adopted: (1) maximum likelihood estimation (MLE) by fitting a pre-assumed type of distribution [23], and (2) kernel smoothing (KNS) which predicts $p(y|\vec{x})$

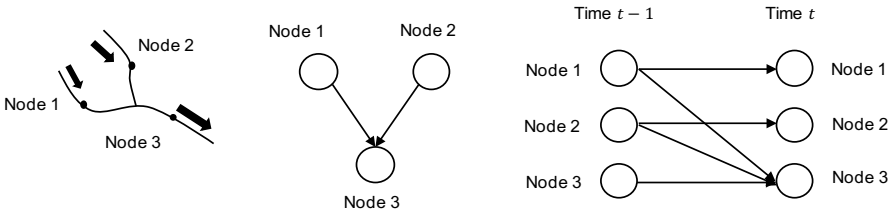

Figure 2: Left: river topology for node 1-3. Middle: a concise structure of node dependency. Right: node dependency extended with a time dimension.

	$Y=1$	$Y=2$	$Y=3$	$Y=4$	$Y=5$
$X_1=1, X_2=1$	$n_{1,1}$	$n_{1,2}$	$n_{1,3}$	$n_{1,4}$	$n_{1,5}$
$X_1=1, X_2=2$	$n_{2,1}$	$n_{2,2}$	$n_{2,3}$	$n_{2,4}$	$n_{2,5}$
$X_1=1, X_2=3$	$n_{3,1}$	$n_{3,2}$	$n_{3,3}$	$n_{3,4}$	$n_{3,5}$
...	...	...	...	...	...
$X_1=5, X_2=5$	$n_{25,1}$	$n_{25,2}$	$n_{25,3}$	$n_{25,4}$	$n_{25,5}$

Figure 3: Example of PGM look-up table. Each row corresponds to an input condition. Each column corresponds to an output level. Each entry $n_{i,j}$ is the number of training instances with the i-th condition and the j-th output.

by:

$$\hat{p}(y|\vec{x}) = \frac{1}{h}\sum_{i=1}^{K}\hat{P}(y^{(d)}=i|\vec{x})\mathrm{F}(\frac{y-B_i}{h}), \tag{6}$$

where h is a factor determining smoothness, and $\mathrm{F}(\cdot)$ is the kernel function with constraint $\mathrm{E}_y(\mathrm{F}(y))=0$ and $\int_{-\infty}^{\infty}\mathrm{F}(y)dy=1$. KNS works like kernel density estimation, but uses a discrete distribution instead of data samples to predict the continuous distribution [23].

4.2 Probabilistic Graphical Models

4.2.1 Graph Structure. A PGM uses a graph structure to describe the dependency of different variables. Since the river topology naturally defines the node dependencies, the graph structure simply follows this topology. Note that for almost all the rivers, there is a constant direction of the flows, and there are no cycles, which makes a directed acyclic graph (DAG) assumption reasonable. In this paper, a directed probabilistic graphical model is considered. A simple example is shown in Figure 2.

4.2.2 Look-up Table. When using PGMs for river forecasting, both predictors and targets need discretization. Given the graph structure, training PGMs requires estimating the conditional probability distribution (CPD) of target variables, given each combination of the predictor values. The CPD is estimated by counting the number of corresponding instances in the training set, and storing them in a look-up table. For example, suppose X_1, X_2 and Y are random variables, and both X_1 and X_2 have a link directing to Y. Suppose X_1, X_2 and Y each have five discrete levels. Then, $P(Y|X_1,X_2)$ is modeled by a look-up table as shown in Figure 3. In the inference phase, if the condition is $(X_1=1, X_2=2)$, then the second row is retrieved and normalized as the predicted CPD of the target.

The look-up table can be very sparse yet in the worst case, its size grows exponentially with the number of predictors and discrete levels. If there are nine predictors and one target each with eight discrete levels, the look-up table will include $8^{10}\approx 10^9$ entries. Even trained with a million instances of data, the sparsity (the number of zero entries over total number of entries) of this table is still greater than 99.9%, causing significant overfitting and very poor inference performance [11].

4.2.3 Neighbor Smoothing. To address the sparsity-based overfitting, we propose a neighbor smoothing (NBS) approach which adds to each row with its "neighbor" rows in the inference phase. To define neighbors, we first define the distance between two condition vectors in the look-up table $C_1: (X_1=x_1, X_2=x_2, ..., X_m=x_m)$ and $C_2: (X_1=x_1', X_2=x_2', ..., X_m=x_m')$ as

$$D(C_1,C_2)=\sum_{i=1}^{m}|x_i-x_i'|, \tag{7}$$

where m is the number of predictors.

Two conditions are defined as h-hop neighbors if their distance is h. For example, the condition vectors $(X_1=1, X_2=2)$ and $(X_1=1, X_2=3)$ are one-hop neighbors; $(X_1=1, X_2=2)$ and $(X_1=2, X_2=1)$ are two-hop neighbors. Two vectors $\vec{x}$ and $\vec{x'}$ are defined as h-hop neighbors if the conditions $(X=\vec{x})$ and $(X=\vec{x'})$ are h-hop neighbors. NBS estimates CPD by:

$$P(Y|\vec{X}=\vec{x})=\frac{1}{Z}\sum_{h=0}^{H}\sum_{\vec{x'}\in\mathbb{V}_h(\vec{x})}\alpha_h P(Y|\vec{X}=\vec{x'}) \tag{8}$$

where $\mathbb{V}_h(\vec{x})$ is the set of h-hop neighbors of $\vec{x}$, H is the pre-defined maximum number of hops, α_h is a decay factor for h-hop neighbors $(0<\alpha_{h+1}<\alpha_h<1)$, and Z is a normalizing coefficient used to ensure that the sum of $P(Y|\vec{X}=\vec{x})$ over all Y values is 1.

The intuition behind NBS is that it allows to increase the size of training data by Gibbs sampling with a small probability of

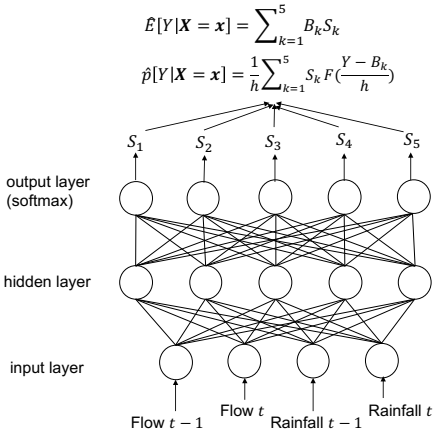

Figure 4: An example of using ANNC for predicting a distribution and the expected value. Assume that only two steps of history are included in inputs, and that the output variable is dicretized into five sections, and KNS is used for smoothing the distribution. S_1 to S_5 are the outputs of five neurons in the softmax layer, which are regarded as the estimate of the discrete distribution of target variable $Y|X$. B_k is the centroid of the k-th section of Y, and $F(.)$ is the kernel function.

distortion. By increasing the ratio of training size over parameter amount, overfitting is alleviated.

4.3 ANNC and ANNR

After discretizing targets of the training set, an ANNC is trained using the backpropagation algorithm, with K output neurons where K is the number of bins of the target variable. The label of each instance in the training set is a vector of length K, where only one of the K values is 1, indicating the correct bin. Since the outputs of softmax function are considered as a proxy of probability distribution of the target variable [26][14], we select the softmax function as the activation function of the output layer. In the inference phase, the K values of the output neurons are used as an estimation of the CPD of the K classes, i.e. $\hat{P}(y^{(d)}|\vec{x})$. Different from ANNC, the architecture of an ANNR has only one output neuron, and no activation function for the output layer (or equivalently, a linear activation function). ANNR is also trained with backpropagation algorithm [25].

4.4 Metrics

As mentioned before, river models can help with two types of questions: (1) what is the expected target value, and (2) what is the confidence interval or the probability that a given target feature falls in some range. While the first predicts an expected value, the second relies on a probability distribution. Correspondingly, two metrics are adopted to assess their prediction performance: R squared (R^2) and mean log-likelihood (MLL). Note that these metrics are applied to the testing set instead of training set.

4.4.1 R^2. R squared is defined as

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - f(\vec{x_i}))^2}{\sum_{i=1}^{N}(y_i - \bar{y_i})^2}, \tag{9}$$

where $f(\cdot)$ is the model function that predicts $\mathrm{E}(y_i|\vec{x_i})$, $\vec{x_i}$ is the input of i-th instance in the testing set, y_i is the true target value of i-th instance, and N is the size of testing set. R^2 measures the relative distance between true and predicted values. As it can be seen, it is smaller than 1, and higher values indicate better performance.

4.4.2 MLL. Mean log-likelihood assesses the quality of predicted probability distribution, and is defined as

$$MLL = \frac{1}{N}\sum_{i=1}^{N}\log \hat{p}(y_i|\vec{x_i}), \tag{10}$$

where N is the size of testing set, $\hat{p}(y|\vec{x_i})$ is the predicted CPD for the target value of the i-th instance, and y_i is the true target value of the i-th instance. Here is an intuitive explanation of using MLL. Suppose two models each predict a distribution with the same mean (μ) and different variances ($\sigma_1^2 < \sigma_2^2$). If the true target value exactly equals μ, then the model predicting smaller variance σ_1^2 is better, since a more concentrated distribution indicates more confidence of the mean value. If, on the other hand, the true target value is far from μ, then the model predicting a wider variance is better, since it allocates more probability to extreme values. However, in terms of R^2 where $f(\vec{x_i}) = \mu$, these two models always have the same performance.

5 EXPERIMENT

5.1 Setup

The experiment is conducted on Guadalupe river described in section 3. The workflow of experiments follows the commonly accepted procedure: (1) data preprocessing, (2) model calibration, and (3) model validation [9].

5.1.1 Data Preprocessing. Missing data appear in almost all the time series collected. Linear imputation is adopted to fill in the missing data. For ANNR and ANNC, data standardization helps accelerate training and alleviates the local optimum problem. To avoid output signal saturation, the input data are standardized to range $[0.1, 0.9]$ as suggested by prior work [24][15][9].

5.1.2 Model Calibration/Validation. Five models are compared: last-value forward (LVF), stepwise multiple linear regression (SWMLR), ANNR, PGM, and ANNC. Their setup configurations are shown in Figure 5. LVF simply uses current value as a prediction for future. SWMLR is a linear model with varying history steps [19]. ANNR and ANNC both have one hidden layer with sigmoid activation function, and hidden neurons twice the number of input neurons [22]. ANNR has one output neuron with linear activation function, while ANNC has K output neurons with softmax function where K is the discrete levels. Five-fold cross validation is adopted. The average values for R^2 and MLL over five folds are reported for each model.

Model	Input	Output	History Steps[1] t_h	Linearity	RvC	Number of Parameters
LVF	$G_{t_c}^{(n)}, D_{t_c}^{(n)}$	$G_{t_c+t_l}^{(n)}$, $D_{t_c+t_l}^{(n)}$	1	Linear	No	0
SWMLR	$\{G_{t_c-t_h+1}^{(\mathbb{N}_n)}, \dots, G_{t_c}^{(\mathbb{N}_n)}\}$ $\{D_{t_c-t_h+1}^{(\mathbb{N}_n)}, \dots, D_{t_c}^{(\mathbb{N}_n)}\}$ $\{P_{t_c-t_h+1}^{(\mathbb{N}_n)}, \dots, P_{t_c}^{(\mathbb{N}_n)}\}$		3	Linear	No	16-25
ANNR			5	Non-linear	No	1351-3361
PGM			1	Non-linear	Yes	>1M[2]
ANNC			5	Non-linear	Yes	1810-4090

Figure 5: Model setup.[1]History steps are selected by step-wise testing for SWMLR, ANNR and ANNC. It is set to 1 for PGM due to large memory requirements.[2]PGMs have an exponential number of parameters, but more than 99.9% are zero. Parameters for each target variable vary with the number of neighbor nodes and available features.

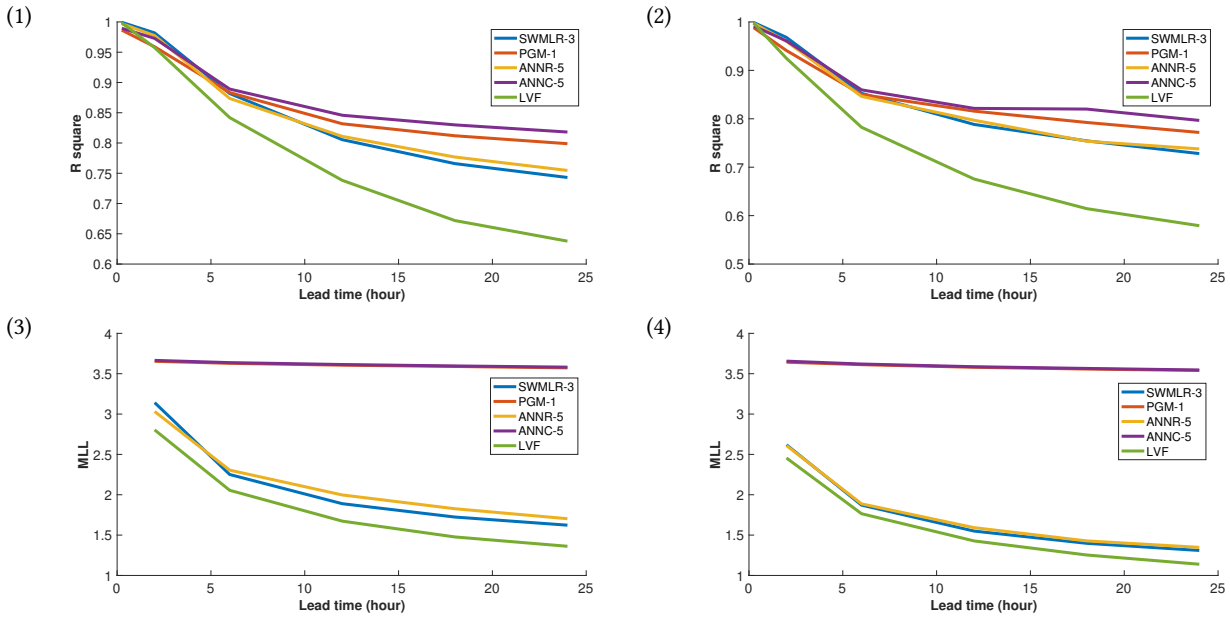

Figure 6: R square and MLL results of five models. (1) and (3) are results of gage forecasting; (2) and (4) are results of discharge forecasting.

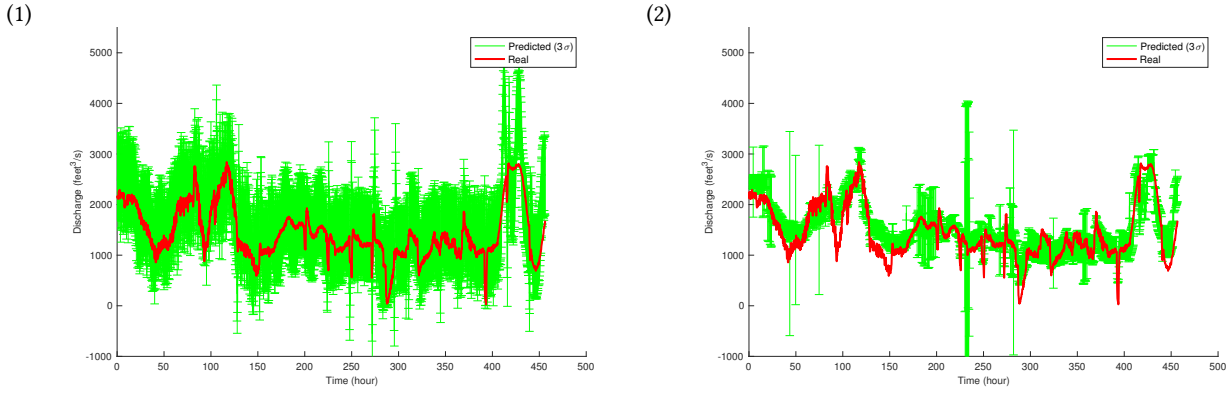

Figure 7: Predicted result vs. true values for one node with (1) SWMLR and (2) PGM. The green bar shows the three-standard-deviation range.

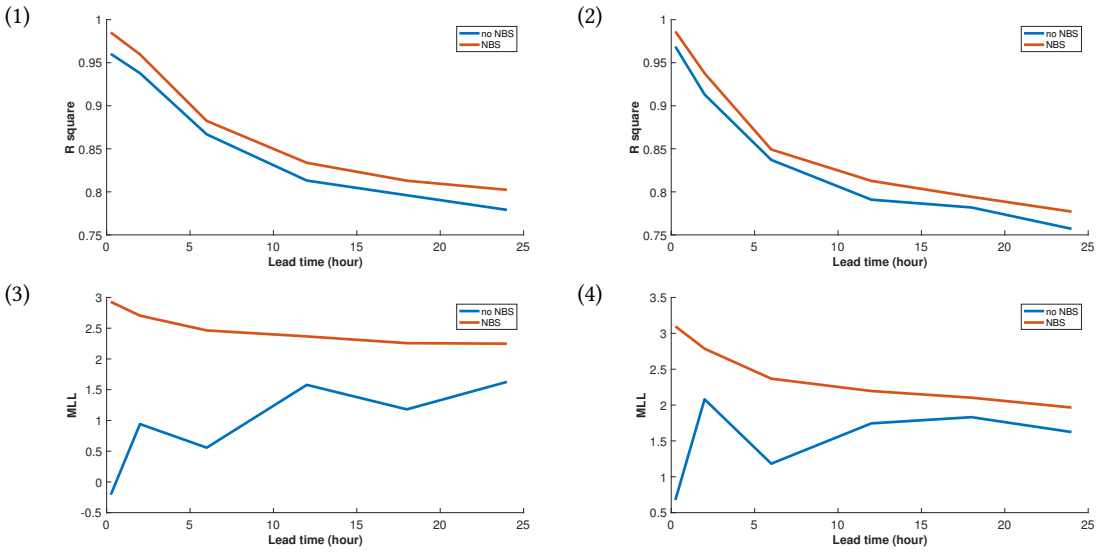

Figure 8: R square and MLL results of PGM with or without NBS. (1) and (3) are results of gage forecasting; (2) and (4) are results of discharge forecasting.

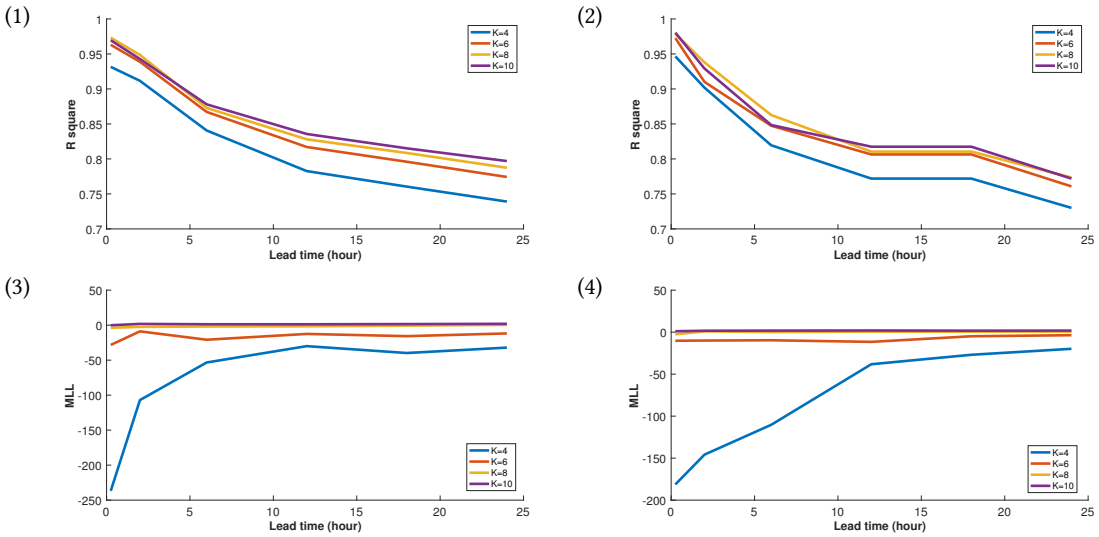

Figure 9: R square and MLL results of PGM with different discrete levels. (1) and (3) are results of gage forecasting; (2) and (4) are results of discharge forecasting.

5.2 Results

The experiments assume the following default settings. K-means discretization and KNS are used to transform between continuous and discrete values or distributions for classification models. Gaussian distribution is adopted for (*i*) the noise of regression models and (*ii*) the MLE approach of transforming discrete distribution to continuum for classification models. The number of discrete levels K is set to ten.

Different models are compared in Figure 6. The models are shown in the format "(model name)-(t_h)", where t_h is the history window size. For SWMLR, ANNR, and ANNC, t_h is selected by stepwise testing, while t_h is set to 1 for PGM due to the exponentially increase in number of parameters. KNS is used to estimate distribution.

Classification models are better for long term forecasting than regression. (*i*) In terms of R^2, classification models are better at long-term forecasting while regression models work better in immediate future, with a cutoff of six hours. Since longer-term forecasting models more complicated relation between predictors and targets, classification models can capture the causal effects better. Besides, classification models can mitigate effects of noise by

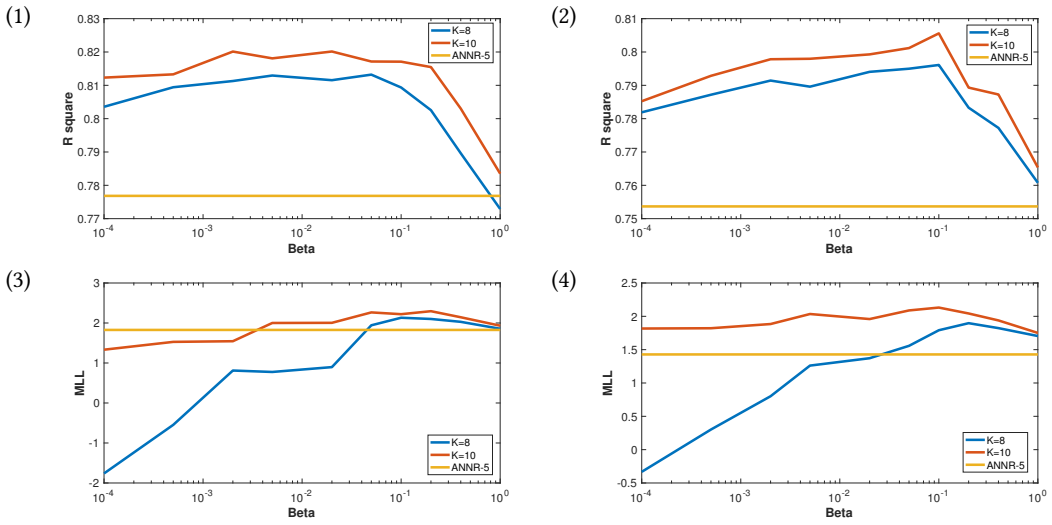

Figure 10: R square and MLL results of PGM with different decay factors for NBS. (1) and (3) are results of gage forecasting; (2) and (4) are results of discharge forecasting.

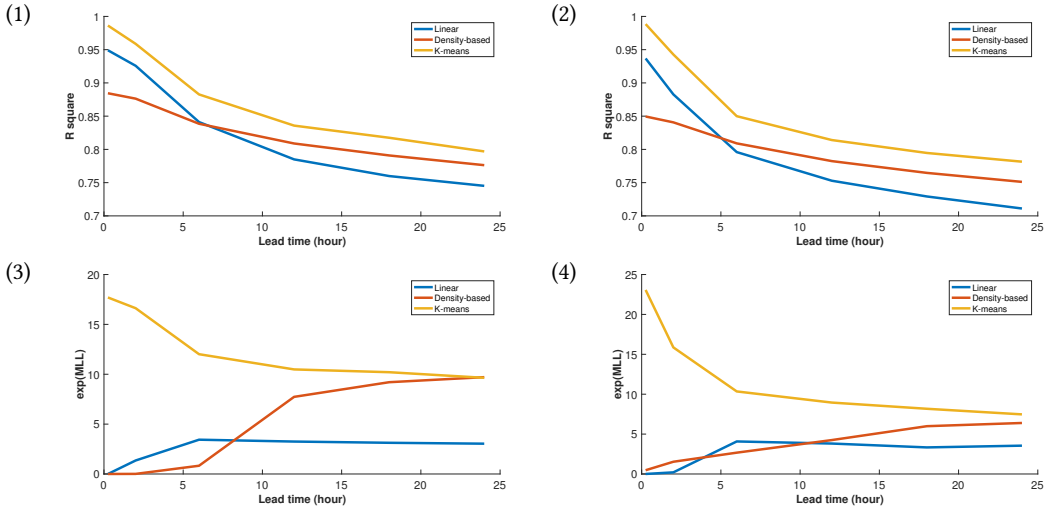

Figure 11: R square and MLL results of PGM with different discretization approach. (1) and (3) are results of gage forecasting; (2) and (4) are results of discharge forecasting.

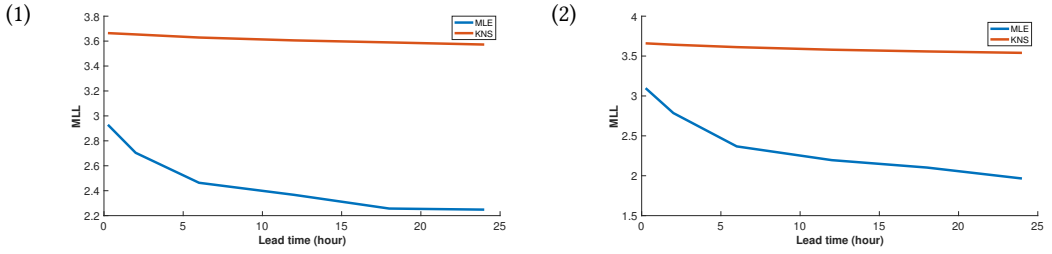

Figure 12: MLL results of the predicted distribution by PGM, under different approaches to converting discrete distribution to a continuous one. (1) is the result of gage forecasting; (2) is the result of discharge forecasting.

discretization, while regression models are more prone to extreme noise. However, for shorter-term gage and discharge prediction, even the LVF model has better performance than classification models which lose accuracy due to quantization. (*ii*) When MLL is considered, classification models always work better than regression models. The assumption made by regression models that noise should have stationary mean and variance does not always hold. However, classification models estimate distributions according to the inputs provided. Figure 7 shows the three-standard-deviation range of the prediction results for a node's discharge using SWMLR and PGM. Except for a few points, PGM captures the true value with higher confidence than SWMLR. Since PGM predicts the variance based on inputs while SWMLR predicts a constant variance, in general PGM works better than SWNLR. The few outlier points with high variance in the PGM graph correspond to time steps for which the input training data also has very high variance.

NBS and increased discretization make PGMs better at forecasting. Figure 8 shows a comparison of PGM with or without NBS. Up to three-hop neighbors are used to smooth the distribution. The decay factor for the h-hop neighbors is chosen by: $\alpha_h = \beta^h$, where β is set to 0.1. The reason of using exponential decay is because the number of neighbors increases exponentially with h. MLE is used to transform the resulting discrete distribution to a continuous one. With NBS, both R^2 and MLL are improved for all lead time values t_l. Different values of β are compared in Figure 9. ANNR-5 is selected as a benchmark. It is observed that too small or too large β both degrades R^2 and MLL. Note that the performance variation is moderate, indicating insensitivity to β selection.

As mentioned, NBS alleviates overfitting since it resembles increasing training size. Another perspective is that it reduces the number of parameters. As an extreme condition, each row of the look-up table becomes the same if $\beta = 1$ and max h is large enough to cover all rows, thereby reducing the degree of freedom to only $K - 1$, where K is the number of discrete levels.

Figure 10 compares the performance of PGM for different discrete levels K. Larger K generally leads to better performances, while for some t_l, $K = 10$ shows poorer R^2 than $K = 8$ due to overfitting.

From continuous to discrete and back: K-means is better than linear and density-based approaches; KNS is superior to MLE when used with PGMs. Linear, density-based and K-means discretization approaches are experimented with PGMs. Results are shown in Figure 11. For better visualization, $exp(MLL)$ is plotted instead of MLL, since linear and density-based approaches have extremely low MLL. $exp(MLL) = (\prod_{i=1}^{N} \hat{p}(y_i|\vec{x}_i))^{\frac{1}{N}}$ is the geometric mean of $\hat{p}(y_i|\vec{x}_i)$. The K-means approach shows the best R^2 and MLL for all t_l, since it captures the clusters best.

MLE and KNS, two different approaches for transforming the discrete distribution in a continuous one, are compared in Figure 12 with PGM. Bandwidth h is selected with asymptotic mean integrated square error (AMISE) approach [21]. KNS shows better estimation of PDF than MLE for all t_l.

6 CONCLUSIONS

In this paper, we compare regression and classification models for river gage and discharge forecasting. MLL is introduced as a metric to assess the estimated probability distribution, while R^2 is used to assess the expected prediction. Experiment results on real data for Guadalupe river (Texas) show that classification models always work better in terms of MLL. For R^2, regression models work better for shorter-term predictions, while classification models are better for longer-term predictions. To discretize continuous variables, K-means works better than linear and density-based approaches. To estimate continuous distribution using a discrete one, KNS is superior to MLE. The methodology can be extended to include additional predictors relevant for multi-modal renewable energy generation besides hydropower (e.g., wind or solar) or to other domains where time series forecasting is of interest.

REFERENCES

[1] *http://maps.waterdata.usgs.gov/mapper/index.html*.
[2] *https://en.wikipedia.org/wiki/Guadalupe_River_(Texas)*.
[3] *https://www.arcgis.com/*.
[4] Jeffrey G Arnold, Daniel N Moriasi, Philip W Gassman, Karim C Abbaspour, Michael J White, Raghavan Srinivasan, Chinnasamy Santhi, RD Harmel, Ann Van Griensven, Michael W Van Liew, et al. 2012. SWAT: Model use, calibration, and validation. *Transactions of the ASABE* 55, 4 (2012), 1491–1508.
[5] Shahrokh Asadi, Jamal Shahrabi, Peyman Abbaszadeh, and Shabnam Tabanmehr. 2013. A new hybrid artificial neural networks for rainfall–runoff process modeling. *Neurocomputing* 121 (2013), 470–480.
[6] Stamatia Bibi, Grigorios Tsoumakas, Ioannis Stamelos, and I Vlahavas. 2008. Regression via Classification applied on software defect estimation. *Expert Systems with Applications* 34, 3 (2008), 2091–2101.
[7] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. 1984. *Classification and regression trees*. CRC press.
[8] KW Chau, CL Wu, and YS Li. 2005. Comparison of several flood forecasting models in Yangtze River. *Journal of Hydrologic Engineering* 10, 6 (2005), 485–491.
[9] CW Dawson and RL Wilby. 2001. Hydrological modelling using artificial neural networks. *Progress in physical Geography* 25, 1 (2001), 80–108.
[10] Christian W Dawson and Robert L Wilby. 1999. A comparison of artificial neural networks used for river forecasting. *Hydrology and Earth System Sciences Discussions* 3, 4 (1999), 529–540.
[11] Nir Friedman and Yoram Singer. 1999. Efficient Bayesian parameter estimation in large discrete domains. In *Advances in neural information processing systems*. 417–423.
[12] James Douglas Hamilton. 1994. *Time series analysis*. Vol. 2. Princeton university press Princeton.
[13] John A Hartigan and Manchek A Wong. 1979. Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 28, 1 (1979), 100–108.
[14] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).
[15] Kuo-lin Hsu, Hoshin Vijai Gupta, and Soroosh Sorooshian. 1995. Artificial neural network modeling of the rainfall-runoff process. *Water resources research* 31, 10 (1995), 2517–2530.
[16] RK Kachroo. 1992. River flow forecasting. Part 1. A discussion of the principles. *Journal of Hydrology* 133, 1-2 (1992), 1–15.
[17] Jordan D Kern, Gregory W Characklis, Martin W Doyle, Seth Blumsack, and Richard B Whisnant. 2011. Influence of deregulated electricity markets on hydropower generation and downstream flow regime. *Journal of Water Resources Planning and Management* 138, 4 (2011), 342–355.
[18] Roman Krzysztofowicz. 2001. The case for probabilistic forecasting in hydrology. *Journal of hydrology* 249, 1 (2001), 2–9.
[19] William M Mendenhall and Terry L Sincich. 2016. *Statistics for Engineering and the Sciences*. CRC Press.
[20] JD Salas, JR Delleur, V Yevjevich, and WL Lane. 1980. Applied modeling of hydrologic time series, Water Resor. *Pub., Littleton, CO, USA* (1980).
[21] Simon J Sheather and Michael C Jones. 1991. A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society. Series B (Methodological)* (1991), 683–690.
[22] K Gnana Sheela and Subramaniam N Deepa. 2013. Review on methods to fix number of hidden neurons in neural networks. *Mathematical Problems in Engineering* 2013 (2013).
[23] Bernard W Silverman. 1986. *Density estimation for statistics and data analysis*. Vol. 26. CRC press.
[24] Murray Smith. 1993. *Neural networks for statistical modeling*. Thomson Learning.
[25] Donald F Specht. 1991. A general regression neural network. *IEEE transactions on neural networks* 2, 6 (1991), 568–576.
[26] Richard S Sutton and Andrew G Barto. 1998. *Reinforcement learning: An introduction*. Vol. 1. MIT press Cambridge.

[27] Claude D Tankersley, Wendy D Graham, and Kirk Hatfield. 1993. Comparison of univariate and transfer function models of groundwater fluctuations. *Water Resources Research* 29, 10 (1993), 3517–3533.
[28] Luís Torgo and Joao Gama. 1996. Regression by classification. *Advances in artificial intelligence* (1996), 51–60.

A APPENDIX

A.1 Problem Definition

Here we give a formal definition of the problem of bounding regression risk of classification models under the framework proposed in this paper. Given input-output pairs $\{X_i, Y_i\}$, $(i = 1, 2, ..., n)$ extracted from time-series data, where $X_i \in R^d$, $Y_i \in R$, and n is the size of the training set. Assume that:

(1) The original time series are stationary, meaning that $\{X_i, Y_i\}$, $(i = 1, 2, ..., n)$ are i.i.d. Assume they follow an underlying probability distribution $p(X, Y)$, where $X \in R^d$ and $Y \in R$.
(2) Y_i is bounded. For simplicity, assume $Y_i \in [0, 1]$.
(3) Prior distribution for $p(X)$ is given.

When using classification model, we first split the range of Y_i into K sections. Denote the $K-1$ splitting points and the two boundary points as $z_0, z_1, ..., z_K$, where $0 = z_0 \le z_1 \le ... \le z_K = 1$. Assume that the K sections have equal length ($= \frac{1}{K}$), and the centroid of each section is the middle point ($\frac{0.5}{K}, \frac{1.5}{K}...$).

Question: Can we give the upper bound for the risk $r(\hat{f}(x)) = \mathbb{E}|\hat{f}(x) - f_0(x)|^2$? (where $\hat{f}(x)$ is the output of the model, and $f_0(x)$ is the "true" regression function satisfying $f_0(x) = \mathbb{E}(Y_i|X_i = x)$)

A.2 Bounding the Risk

The fitted function of the classification model can be written as:

$$\hat{f}(x) = \sum_{j=1}^{K} \hat{P}(Y \in [z_{j-1}, z_j]|X = x) \frac{z_{j-1} + z_j}{2} \tag{11}$$

We can split the risk into bias and variance:

$$r(\hat{f}(x)) = \mathbb{E}|\hat{f}(x) - f_0(x)|^2 = |\mathbb{E}[\hat{f}(x)] - f_0(x)|^2 + \mathbb{E}|\hat{f}(x) - \mathbb{E}[\hat{f}(x)]|^2 \tag{12}$$

and bound the two terms respectively.

A.2.1 Bounding $Bias^2 = |\mathbb{E}[\hat{f}(x)] - f_0(x)|^2$. The fitted function for classification approach is:

$$\hat{f}(x) = \sum_{j=1}^{K} \hat{P}(Y \in [z_{j-1}, z_j]|X = x) \frac{z_{j-1} + z_j}{2} \tag{13}$$

where (for PGM model; and for simplicity, consider that X is a discrete variable)

$$\hat{P}(Y \in [z_{j-1}, z_j]|X = x) = \frac{\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(X_i = x, Y_i \in [z_{j-1}, z_j])}{P(X = x)} \tag{14}$$

Therefore,

$$\begin{aligned} \mathbb{E}[\hat{P}(Y \in [z_{j-1}, z_j]|X = x)] &= \frac{\frac{1}{n}\sum_{i=1}^{n} P(X_i = x, Y_i \in [z_{j-1}, z_j])}{P(X = x)} \\ &= P(Y \in [z_{j-1}, z_j]|X = x) \end{aligned} \tag{15}$$

Thus, combining (13) and (15) we have:

$$\begin{aligned} \mathbb{E}[\hat{f}(x)] &= \sum_{j=1}^{K} P(Y \in [z_{j-1}, z_j]|X = x) \frac{z_{j-1} + z_j}{2} \\ &= \sum_{j=1}^{K} \int_{z_{j-1}}^{z_j} p_{Y|X}(y|x) \frac{z_{j-1} + z_j}{2} dy \end{aligned} \tag{16}$$

Since

$$\begin{aligned} f_0(x) &= \mathbb{E}_{Y|X}[y|X = x] \\ &= \sum_{j=1}^{K} \int_{z_{j-1}}^{z_j} p_{Y|X}(y|x) y dy \end{aligned} \tag{17}$$

We have:

$$\begin{aligned} |\mathbb{E}[\hat{f}(x)] - f_0(x)| &\le \sum_{j=1}^{K} \int_{z_{j-1}}^{z_j} p_{Y|X}(y|x) \, |\frac{z_{j-1} + z_j}{2} - y| \, dy \\ &\le \sum_{j=1}^{K} \int_{z_{j-1}}^{z_j} p_{Y|X}(y|x) dy \cdot (z_j - z_{j-1}) \\ &= \frac{1}{K} \end{aligned} \tag{18}$$

Therefore,

$$Bias^2 \le \frac{1}{K^2} \tag{19}$$

A.2.2 Bounding $Var(\hat{f}(x)) = \mathbb{E}|\hat{f}(x) - \mathbb{E}[\hat{f}(x)]|^2$. Due to the fitted function (13),

$$\begin{aligned} & Var(\hat{f}(x)) \\ &= \sum_{j=1}^{K} \frac{P(X = x, Y \in [z_{j-1}, z_j])(1 - P(X = x, Y \in [z_{j-1}, z_j]))}{nP(X = x)^2} \cdot \\ &(\frac{z_{j-1} + z_j}{2})^2 \\ &= \sum_{j=1}^{K} \frac{P(Y \in [z_{j-1}, z_j]|X = x)(1 - P(Y \in [z_{j-1}, z_j]|X = x))}{n} \cdot \\ &(\frac{z_{j-1} + z_j}{2})^2 \\ &\le \sum_{j=1}^{K} \frac{P(Y \in [z_{j-1}, z_j]|X = x)}{n} (\frac{z_{j-1} + z_j}{2})^2 \\ &\le \sum_{j=1}^{K} \frac{P(Y \in [z_{j-1}, z_j]|X = x)}{n} \\ &= \frac{K}{n} \end{aligned} \tag{20}$$

A.2.3 Combining bias and variance.

$$Bias^2 + Var \le \frac{1}{K^2} + \frac{K}{n} \tag{21}$$

Therefore, when choosing $K = \Theta(n^{\frac{1}{3}})$, the risk is bounded as $O(n^{-\frac{2}{3}})$.