Uncertainty Quantification in SVM prediction

Anonymous authors
Paper under double-blind review

Abstract

This paper explores SVM models from the lens of uncertainty quantification (UQ), developed for regression and forecasting tasks. Unlike the Neural Network, the SVM solutions are typically more certain, stable, sparse, optimal and interpretable. However, there is only limited literature addressing uncertainty quantification (UQ) in SVM-based prediction. At first, We provide a comprehensive summary of existing Prediction Interval (PI) estimation and probabilistic forecasting methods developed in the SVM framework. Although SVMs offer globally optimal and stable solutions, the existing literature on UQ within the SVM framework still exhibits several critical gaps. In this work, we also address these gaps and extend contemporary UQ techniques to SVMs, for promoting their applicability across diverse domains for more reliable estimation. Our major contributions include the development of sparse SVM models for PI estimation and probabilistic forecasting, an investigation of the role of feature selection in PI estimation, and the extension of SVM regression to the Conformal Regression (CR) setting to construct more stable prediction sets with finite-sample guarantees. Extensive numerical experiments highlight that SVM-based UQ methods yield PIs and probabilistic forecasts that are less uncertain and comparable to, or even better than, those produced by modern complex deep learning and neural network models.

1 Introduction

Given the training set $T' = \{(x_i, y_i) : x_i \in \mathbf{R}^n, y_i \in \mathbf{R}, i = 1, 2, ...m\}$, sampled independently from the joint distribution of the random variables (X, Y), the goal of the regression task is to estimate a function that predicts the target variable y based on the input variable x well. However, in most of applications, the prediction of the regression model may not be perfectly accurate due to random relationship between Y and X. For example, predicting the impact of a specific drug on a patient's heart rate based on their Body Mass Index (BMI) may not be accurate and may involve a significant degree of uncertainty. In such cases, quantifying these uncertainties is crucial for making effective decisions.

The Prediction Interval (PI) estimation is most commonly used Uncertainty Quantification (UQ) technique in regression tasks. Given a high confidence $1 - \alpha \in (0, 1)$ and training set T', the PI tube is defined as a pair of functions (f_1, f_2). It is said to be well calibrated if it satisfies $P(f_1(X) \leq Y \leq f_2(X)|X) \geq 1 - \alpha$. The objective of the PI models is to obtain a PI tube with the minimum possible width while ensuring the target calibration. Therefore, the performance of a PI estimation method is mainly evaluated using two criteria: Prediction Interval Coverage Probability (PICP), which computes the fraction of y values within the PI tube, and Mean Prediction Interval Width (MPIW), quantifying the width of the PI tube.

The PI estimation models requires to explore the different characteristics of the conditional distribution $Y|X$, rather focusing only on $E(Y|X)$ as done in standard regression tasks. The basic approach of PI models involves estimating a pair of quantile functions (Koenker & Bassett Jr (1978)), say $(f_q(x), f_{1+q-\alpha}(x))$, of the conditional distribution $Y|X$, for some $0 \leq q \leq \alpha$, where the qth quantile function for given x is defined as infimum of functions satisfying $P(y \leq f_q(x)|x) = q$.

For time-series data, estimating the Prediction Interval (PI) for future observations using an auto-regressive approach is referred to as probabilistic forecasting. Both PI estimation and probabilistic forecasting models are widely investigated in the Neural Network (NN) architectures in the literature. The PI estimation

methods in the NN literature can primarily be divided into two main categories. A popular class of PI estimation methods assumes that the conditional distribution $Y|X$ follows a particular distribution (often normal) and obtains the quantile functions by computing the inverse of the corresponding cumulative distribution function. Some important of them are Bayesian method (MacKay (1992); Bishop (1995), Delta methodDe VlEAUX et al. (1998); Hwang & Ding (1997); Seber & Seber (2015)) and Mean Variation Estimation (MVE) method (Nix & Weigend (1994)). Some of the recent NN architecture for the probabilistic forecasting task with distribution assumptions are Mix Density Network (Bishop (1994); Zhang et al. (2020a)), Deep Auto-regressive Network (Deep AR) (Salinas et al. (2020)).

The other class of PI estimation and probabilistic forecasting methods believe in estimating the pair of quantile functions $(f_q(x), f_{1+q-\alpha}(x))$ in distribution-free setting without imposing any assumption regarding the distribution of $Y|X$. For estimation of the qth quantile function, $f_q(x)$, most of them minimizes the pinball loss function. The pinball loss-based NN model, also known as Quantile Regression Neural Network (QRNN) (Taylor (2000); Cannon (2011)) is the main PI estimation method, which has been utilized in various engineering applications. The pinball loss-based NN model has been frequently applied to probabilistic forecasting of wind (Wan et al. (2016)), electric load (Zhang et al. (2018; 2020b)), electric consumption (He et al. (2019)), flood risks (Pasche & Engelke (2024)) and solar energy (Lauret et al. (2017)). Some of the distribution-free PI estimation NN methods consider the minimization of a particularly designed loss function for the direct and simultaneous estimation of the bounds of the PI. Some of them are Lower Upper Bound Estimation (LUBE) NN, Quality-Driven (QD) Loss NN (Pearce et al. (2018)) and Tube loss NN (Anand et al. (2024)).

However, a well-calibrated High Quality (HQ) PI guarantees the target coverage level t only asymptotically, and may fail to achieve it on finite test samples. In real-world decision-making, especially in high-stakes applications, guarantees on finite test samples coverage are often essential. Conformal Regression (CR) (Vovk et al. (1999; 2005)) provides a principled framework through which PI models can be adapted to ensure such finite-sample coverage guarantees, making them more suitable for practical deployment.

The aforementioned UQ techniques derive a pair of quantile functions to quantify the uncertainty in the relationship between X and Y, However, their estimates themselves may also involve a certain degree of uncertainty. Therefore, researchers typically consider two main sources of uncertainty when assessing the overall uncertainty: data uncertainty (aleatoric uncertainty) and model uncertainty (epistemic uncertainty). Data uncertainty arises from the inherent noise or variability in the relationship between X and Y while model uncertainty primarily arises from the uncertainty in the estimated model parameters.

For UQ task in NN and deep learning regression, accurately capturing model uncertainty, in addition to data uncertainty is essential. This is due to the significant variability in the learned weights, as the underlying objective functions are typically non-convex. As a result, training the same model multiple times with the same dataset and hyperparameter settings can still lead to different local minima. Studies by Lakshminarayanan et al. (2017) and Pearce et al. (2018) have shown that ensemble methods are among the most effective approaches for addressing model uncertainty in NN. The key idea is to train the model multiple times with different random initializations on the same training set T', and then estimate model uncertainty by measuring the variability in the resulting learned parameters.

However, in contrast to NN, SVM does not suffer from such parameter uncertainty, as they obtain the global optimal solution that remains invariant to the choice of the initialization. It makes the SVM more trustworthy and less uncertain than NN. Apart from the global optima, the SVM also offers simple, interpretable and often sparse solutions along with an explicit mechanism to incorporate the regularization.

Despite these notable advantages, SVM remains surprisingly almost underexplored from the lens of the UQ. In contrast to detailed NN literature, only a limited number of SVM-based approaches have been proposed for PI estimation and probabilistic forecasting tasks. This paper provides a comprehensive survey of SVM models developed for UQ tasks. Despite the advantages of global optimal and stable solution, the existing SVM literature on UQ contains critical gaps. In this paper, we also bridge these gaps and further extend contemporary UQ techniques to the SVM framework to promote the SVM UQ application across diverse domains for a more reliable estimate.

We summarize the contribution of our work in detail as follows.

(a) First, we carefully review the existing literature on PI estimation and probabilistic forecasting methods in SVM. We outline the desirable properties of an ideal PI model and compare the PI estimation and probabilistic forecasting methods in SVM against them. We find that only two of SVM PI methods attain the global optimal solution but, none of them achieve a sparse solution vector.

(b) Building on this motivation, we develop a sparse SVM solution for PI estimation and probabilistic forecasting tasks by simply solving a pair of pinball loss problems with L_1 regularization via Linear Programming Problem (LPP). Our Sparse SVM PI model preserves the classical properties of SVM by achieving both a globally optimal and sparse solution. Unlike NN, the global optimal solution of Sparse SVM PI formulation ensures the zero parameter uncertainty while its sparse solution reduces the complexity of the solution. It enables the sparse SVM PI method to estimate a less uncertain and complex PI.

(c) Further, we investigate the importance of the feature selection in PI estimation particularly in high dimensional regression tasks. At first, we perform the feature selection task by developing a simple yet effective feature selection algorithm for PI estimation using our sparse SVM PI model. We show that our algorithm does not only successfully discard a significant percentage of features but, also improves the quality of the PI while learning the PI for high-dimensional data. To the best of our knowledge, there is no any existing literature which study the feature selection problem in context of PI estimation.

(d) Finally, we note that although SVM models inherently exhibit zero model uncertainty, they have not yet been extended to the conformal framework for regression tasks. In this work, we bridge this gap by adapting SVM regression to the conformal prediction setting, thereby ensuring finite-sample coverage guarantees. Compared to NN, we show that the conformal prediction sets produced by SVM models are more stable, certain and interpretable due to its global optimal solution.

(e) We conduct extensive experiments on artificial, real-world benchmark datasets to empirically analyze the quality of solution obtained by Sparse SVM PI model relative to existing SVM PI models. We show the advantages of the feature selection method in PI estimation task on benchmark high-dimensional datasets. Further, we empirically verify that the SVM based CR model solution is more stable, certain than NN based CR model. Finally, we perform the experiments to demonstrate that the SVM based probabilistic forecasting models can achieve comparable, and in some cases superior, PI quality relative to recent complex deep probabilistic forecasting models that involve significantly larger numbers of parameters on few benchmark datasets.

The remainder of this paper is structured as follows. Section 2 provides a systematic review of the prerequisite concepts required for understanding of the SVM models and UQ techniques. Section 3 provides a detail description of the several SVM models for PI estimation and probabilistic forecasting, highlighting their advantages and limitations. In Section 4, we develop Sparse SVM algorithms for PI estimation along with feature selection, probabilistic forecasting and extend SVM model in conformal regression setting . Section 5 presents the numerical results from extensive experiments, demonstrating the effectiveness of the SVM model for UQ tasks. Section 6 concludes this work by outlining its impact along with future research directions.

2 Related key concepts

In this section, we will outline key concepts and methods relevant to PI estimation techniques in SVM.

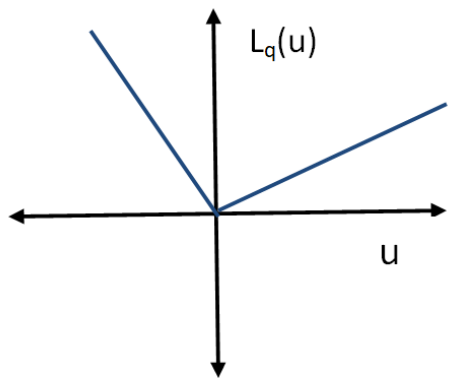


Figure 1: Pinball loss function

2.1 Quantile Regression and SVM

In distribution free setting, for a given quantile $q \in (0,1)$, the quantile value is estimated by minimizing the pinball loss function, which is given by

$$\rho_q(u) = \begin{cases} qu, & \text{if } \ u \geq 0, \\ (q-1)u, & \text{Otherwise.}, \end{cases} \tag{1}$$

For the estimation of the conditional quantile function, u represents the error obtained by the subtracting the estimates $f(x_i)$ from its target values y_i. For a given quantile $q \in (0,1)$, training set $T' = \{(x_i, y_i) : x_i \in \mathbf{R}^n, y_i \in \mathbf{R}, i = 1, 2, ...m\}$ and class of function F, let us suppose that f_T is the solution of the problem $\min\limits_{f \ \in F} \sum\limits_{i=1}^{m} \rho_q(y - f(x_i))$. Tekeuchi et al., have shown that the fraction of y values lying below the function $f_T(x)$ is bounded from above by qm and asymptotically equals qm with probability 1 under a very general condition in their work (Takeuchi et al. (2006)).

Given the training set, SVM models estimate the function in the form of $f(x) = w^T\phi(x) + b$, where ϕ maps the input variable x into the high dimensional feature space, such that for any pair of x_i and x_j in $\mathbf{R}^n$, $\phi(x_i)^T\phi(x_j)$ can be obtained by the well defined kernel function $k(x_i, x_j)$. By the use of the kernel trick and representer theorem Schölkopf et al. (2001), the SVM estimate $f(x) = w^T\phi(x) + b$ can be represented by the kernel generated function in the form of $\sum_{i=1}^{m} k(x_i, x)u_i + b$, where k is positive-semi definite kernel Mercer (1909). This representation eliminates the need for explicit knowledge of the mapping ϕ.

2.2 Support Vector Quantile Regression model

The Support Vector Quantile Regression (SVQR) model (Takeuchi et al. (2006)) minimizes the L_2-norm of the regularization along with the empirical risk computed by the pinball loss function. For q^{th} quantile function estimation, it seeks the solution of the problem

$$\min_{(w,b)} \frac{1}{2}w^T w + C\sum_{i=1}^{m} \rho_q(y_i - (w^T\phi(x_i) + b)), \tag{2}$$

which can be equivalently converted to the following Quadratic Programming Problem (QPP)

$$\begin{aligned} &\min_{(w,b,\xi,\xi^*)} \frac{1}{2}w^T w + C\sum_{i=1}^{m}(q\xi_i + (1-q)\xi_i^*) \\ &\text{subject to,} \\ &\quad y_i - (w^T\phi(x_i) + b) \leq \xi_i, \\ &\quad (w^T\phi(x_i) + b) - y_i \leq \xi_i^*, \\ &\quad \xi_i, \xi_i^* \geq 0, \ i = \ 1, 2, ..m, \end{aligned} \tag{3}$$

where $C \geq 0$ is the user defined parameter for trading-off the empirical risk against the model complexity.

To efficiently solve QPP (3), we often focus on obtaining the solution to its corresponding Wolfe dual problem, which is given by

$$\begin{aligned}
&\min_{(\alpha,\beta)} \sum_{i=1}^{m}\sum_{j=1}^{m}(\alpha_i-\beta_i)k(x_i,x_j)(\alpha_j-\beta_j) - \sum_{i=1}^{m}(\alpha_i-\beta_i)y_i \\
&\text{subject to,} \\
&\quad \sum_{i=1}^{m}(\alpha_i-\beta_i)=0, \\
&\quad 0 \leq \alpha_i \leq Cq,\ i=\ 1,2,..m, \\
&\quad 0 \leq \beta_i \leq C(1-q),\ i=\ 1,2,..m,
\end{aligned} \tag{4}$$

where $(\alpha_i, \beta_i), i=\ 1,2,..,m$, are Lagrangian multipliers.

After obtaining the optimal solution of the dual problem (4), $(\alpha_i^*, \beta_i^*), i=\ 1,2,..,m$, the qth quantile function is estimated by

$$f_q(x) = \sum_{i=1}^{m}(\alpha_i^* - \beta_i^*)k(x_i,x) + b. \tag{5}$$

The estimation of the bias term b can be obtained by using the KKT conditions of the primal problem (3). For this, we need to select the every training point (x_k, y_k) which corresponds to $0 < \alpha_k^* < Cq$ or $0 < \beta_k^* < C(1-q)$ and compute

$$b_k = y_k - \sum_{i=1}^{m}(\alpha_i^* - \beta_i^*)k(x_i,x_k). \tag{6}$$

The final value of bias b can be obtained by computing the the mean of all b_k.

2.3 Least Squares Support Vector Regression

For estimating the mean regression using training set T', the LS-SVR model Suykens et al. (2002) minimizes the least square loss function along with the L_2-norm of regularization in the following problem.

$$\begin{aligned}
&\min_{(w,b,\xi)} \frac{1}{2}w^T w + C\sum_{i=1}^{m}(\xi_i^2) \\
&\text{subject to,} \\
&\quad y_i - (w^T\phi(x)+b) = \xi_i,\ i=\ 1,2,..m.
\end{aligned} \tag{7}$$

The solution of problem (7) can be obtained through the following system of equations

$$\begin{bmatrix} 0 & e^T \\ e & K(A,A^T)+\frac{2}{C}I \end{bmatrix}\begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ Y \end{bmatrix}, \tag{8}$$

where $K(A, A^T)$ is an $m \times m$ kernel matrix constructed from the training set T', e is an m-dimensional column vector of ones, and I represents the $m \times m$ identity matrix. After obtaining the (b, α) from (8), the LS-SVR estimates the regression function for a given $x \in \mathbb{R}^n$ using

$$f(x) = \sum_{i=1}^{m} k(x_i,x)\alpha_i + b. \tag{9}$$

2.4 Probabilistic Forecasting

The task of probabilistic forecasting is basically an extension of the PI estimation in an auto-regressive setting. Consider the time series observations $T = \{x_1, x_2, ..., x_t\}$, recorded on t different time stamps. If

$p < t$ denotes the effective lag window, then auto-regressive models estimate the relationship between $z_i := (x_{i-p+1}, \ldots, x_i)$ and x_{i+1} for $i = p, p+1, \ldots, t-1$ using the training set T', and use this learned relationship to forecast future observations. Point forecasting models aim to estimate the conditional expectation $E(x_{i+1} \mid z_i)$. However, in many cases, such forecasts may incur significant errors due to inherent noise and volatility in the data. Probabilistic forecasting quantifies these uncertainties in the prediction by obtaining the PI.

The task of probabilistic forecasting is to estimate the PI for x_{i+1} given the input z_i for $i \geq t$. The SVM based probabilistic forecasting models obtain the estimate of the PI $[\hat{F}_q(z_i), \hat{F}_{1+q-\alpha}(z_i)]$, where $\hat{F}_q(z_i)$ and $\hat{F}_{1+q-\alpha}(z_i)$ are kernel generated functions, estimating of the qth and $(1-\alpha+q)^{th}$ quantiles of the conditional distribution $(x_{i+1} \mid z_i)$ for some $q \in (0, \alpha)$. Distribution-free probabilistic forecasting methods estimate quantile functions directly, without making any assumptions about the conditional distribution $(x_{i+1} \mid z_i)$.

2.5 Conformal Regression

Conformal Regression (Vovk et al. (1999; 2005)) provides a general framework for adjusting PI models to guarantee the target coverage $1-\alpha$ on finite test samples, assuming only that the data are exchangeable.

The split conformal regression (CR) approach (Papadopoulos et al. (2002); Papadopoulos (2008))starts by dividing the available training data T into two separate subsets: a training set I_1 used to fit the predictive model, and a calibration set I_2 used for PI adjustment. A nonconformity score function is then introduced to quantify the disagreement between predicted value for y_i, given input x_i and its actual observed value. These nonconformity scores are computed on the calibration set I_2. A obvious choice of the non-conformity score is the absolute residual value, computed by $|y_i - \hat{f}(x_i)|, \quad i \in I_2$, where $\hat{f}$ is the estimate of the mean regression model trained on I_1.

Romano et al. have developed the quantile regression based nonconformity score for obtaining the fully adaptive conformal prediction set in their work Conformalized Quantile Regression (CQR) (Romano et al. (2019)), which is given by

$$E_i = max\{\hat{F}_{lo}(x) - y_i,\ y_i - \hat{F}_{hi}(x)\}, \tag{10}$$

where $\hat{F}_{lo}(x)$ and $\hat{F}_{hi}(x)$ are the estimates of the qth and $(1+q-\alpha)^{th}$ quantile function on set I_1 for some $0 \leq q \leq \alpha$.

After computing the non-conformity scores, the CR methodology requires the computation of $(1-\alpha)(1+\frac{1}{|I_2|})$-th empirical quantile of the non-conformity score. In case of CQR, we denote it with $Q_{1-\alpha}(E_i, I_2)$ and obtain the prediction set on new test point x_{m+1} as

$$C(x_{m+1}) = [\hat{F}_{lo}(x_{m+1}) - Q_{(1-\alpha)}(E, I_2), \hat{F}_{hi}(x_{m+1}) + Q_{1-\alpha}(E, I_2)] \tag{11}$$

3 PI estimation in SVM

In this section, we gather in detail the PI estimation and probabilistic forecasting methods developed in the SVM literature and compare their advantages and limitations.

3.1 PI estimation through LS-SVR

One of the naive PI estimation method in SVM literature follows the normal assumption regarding the distribution of $Y|X$ and estimate its mean through (9) by training the LS-SVR model. The error distribution $\epsilon_i = y_i - f(x_i)$ follows a normal distribution with a mean of zero and variance σ. This variance can be estimated from the error computed on training set T'. The pair of quantile bounds required for PI is estimated as $(\hat{f(x)} + \epsilon_{\frac{\alpha}{2}}, \hat{f(x)} + \epsilon_{1-\frac{\alpha}{2}})$, where ϵ_q is the qth quantile of the error. A more refined and bias-corrected PI based on the LS-SVR model is proposed in (De Brabanter et al. (2010); Cheng et al. (2014)).

3.2 PI estimation through SVQR

Given the high confidence $1-\alpha$ with training set T', the PI model requires the estimation of the pair of quantile functions $(f_q(x), f_{1+q-\alpha}(x))$ of the conditional distribution Y/X. The SVQR model can be trained twice for the estimation of the pair of these quantile functions for some $0 \leq q \leq \alpha$. We detail the algorithm for PI estimation through SVQR in Algorithm 1. At Algorithm 1, the tuning of C refers to selecting the value of C from a specified range such that the SVQR estimate obtains the least coverage error.

Algorithm 1 PI estimation through SVQR

procedure :- PI THROUGH SVQR$(T, 1-\alpha)$
  Choose some $\bar{q} \in (0, 1-\alpha)$
  for each $q \in \{\bar{q}, (1+\bar{q}-\alpha)\}$ **do**
    Solve the QPP problem (4) by tuning the value of C. Obtain the solution (α^*, β^*).
    Estimate the function $f_q(x)$ using the (5).
  return $(f_{\bar{q}}(x), f_{1+\bar{q}-\alpha}(x))$

A key challenge in estimating prediction intervals (PI) using the quantile approach is to determine the good choice of $\bar{q}$ for obtaining the narrower PI. For a symmetric noise distribution, $\bar{q} = \frac{\alpha}{2}$ is expected to produce the PI with minimum width. However, this does not hold for an asymmetric noise distribution. In the latter case, $\bar{q}$ should be selected such that the resulting PI passes through the denser regions of the data cloud. Furthermore, for each choice of $\bar{q}$, the optimization problem (4) must be solved twice for $q = \bar{q}$ and $q = 1 + \bar{q} - \alpha$ to obtain the prediction interval (PI). It increases the overall computational complexity of the PI estimation process, making it both time-consuming and challenging in practice.

To simplify the PI estimation process, researchers have developed direct PI estimation methods, which solve a single optimization problem to obtain the both bounds of PI simultaneously. These methods are designed with a specialized loss function that can be minimized to obtain the both bounds of PI simultaneously. Some important of them are LUBE loss (Khosravi et al. (2010)), Quality-Driven (QD) loss (Pearce et al. (2018)) and Tube loss (Anand et al. (2024)) functions. We describe those also formulated within the SVM framework as follows.

3.3 PI estimation through Tube loss

(Anand et al. (2024)) have developed the Tube loss for PI estimation and probabilistic forecasting. It can be minimized directly to obtain the bounds of the PI simultaneously. The minimizer of the Tube loss function also guarantees the target coverage $1-\alpha$ asymptotically. Also, the PI tube can also be shifted up or down by tuning its parameter r so that it can cross through the denser region of data cloud for minimal PI width. Furthermore, the width of the PI tube can be explicitly minimized in its optimization problem through the parameter δ. It helps to improve the PI width, when the PI tube achieves a coverage higher than the target on the validation set.

The Tube loss function is a kind of two-dimensional extension of the pinball loss function (Koenker & Bassett Jr (1978)). For a given $1-\alpha \in (0,1)$ and $u_2 \leq u_1$, the Tube loss function is given by

$$\rho^r_{1-\alpha}(u_2, u_1) = \begin{cases} (1-\alpha)u_2, & \text{if} \quad u_2 > 0, \\ -\alpha u_2, & \text{if } u_2 \leq 0, u_1 \geq 0 \text{ and } \; ru_2 + (1-r)u_1 \geq 0, \\ \alpha u_1, & \text{if } u_2 \leq 0, u_1 \geq 0 \text{ and } \; ru_2 + (1-r)u_1 < 0, \\ -(1-\alpha)u_1, & \text{if } \; u_1 < 0, \end{cases} \tag{12}$$

where $0 < r < 1$ is a user-defined parameter and $(u_2,\, u_1)$ are errors, representing the deviations of y values from the bounds of PI.

Figure 2 illustrates the Tube loss for $(1-\alpha) = 0.9$ with $r = 0.5$. For $r = 0.5$, the Tube loss function is always a continuous loss function of u_1 and u_2, symmetrically positioned around the line $u_1 + u_2 = 0$. In all

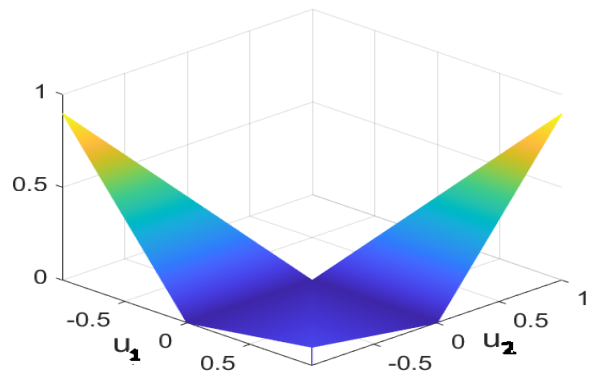


Figure 2: Tube loss function for $1-\alpha = 0.9$.

experiments with a symmetric noise distribution, the r parameter in the Tube loss function should be set to 0.5 to capture the denser region of y values.

The Tube loss SVM model seeks a pair of kernel generated functions

$$\mu_1(x) = \sum_{i=1}^{m} k(x_i, x)\alpha_i + b_1 \quad \text{and } \mu_2(x) = \sum_{i=1}^{m} k(x_i, x)\beta_i + b_2 \tag{13}$$

by minimizing the optimization problem

$$\min_{(\alpha,\beta,b_1,b_2)} J(\alpha,\beta,b_1,b_2) = \frac{\lambda}{2}(\alpha^T\alpha + \beta^T\beta) + \sum_{i=1}^{m} \rho^r_{1-\alpha}\big(y_i, \big(K(A^T, x_i)\alpha + b_1\big), \big(K(A^T, x_i)\beta + b_2\big)\ \big)$$
$$+\delta \sum_{i=1}^{m} \big|(K(A^T, x_i)(\alpha - \beta) + (b_1 - b_2)\big|, \tag{14}$$

where δ, r and λ are user-defined parameters and A is the $m \times n$ data matrix containing the training set. Further, details on the Tube SVM problem and its minimization using gradient descent method can be found in (Anand et al. (2024)).

3.4 PI estimation through LUBE loss

The LUBE method (Khosravi et al. (2010)) was originally developed in the NN framework but, was extended in the SVM framework later in (Shrivastava et al. (2014; 2015)) for probabilistic forecasting of electric price. For the given target confidence $(1-\alpha)$, and training set T', the LUBE SVM model seeks a pair of kernel generated functions of (13), $\mu_1(x)$ and $\mu_2(x)$, which are obtained by minimizing the following loss function

$$CWC = \frac{1}{R} MPIW\big(1 + \gamma\ (PICP)e^{-\eta(PICP-(1-\alpha))}\big). \tag{15}$$

Here, MPIW is the average width of estimated PI on training set, computed by $\frac{1}{m}\sum_{i=1}^{m}(\mu_2(x_i) - \mu_1(x_i))$. As discussed earlier, the PICP is the coverage of the estimated PI and computed using the $PICP = \frac{1}{m}\sum_{i=1}^{m} k_i$, where

$$k_i = \begin{cases} 1, & \text{if } \ y_i \in [\mu_1(x), \mu_2(x)]. \\ 0, & \text{Otherwise.} \end{cases}$$

Further, the $\gamma(PICP) = \begin{cases} 0, & \text{if } \ PICP \geq 1-\alpha, \\ 1, & \text{otherwise,} \end{cases}$, R is the range of response values y_i and η is the user-defined parameter.

	LS-SVR PI	SVQR PI	LUBE	Tube loss
Distribution-free method	No	Yes	Yes	Yes
Asymptotic guarantees	Only with normal noise	Yes	No	Yes
Direct PI estimation	Yes	No	Yes	Yes
PI tube movement	No	Yes	No	Yes
Global optimal solution	Yes	Yes	No	No
Re-calibration	No	No	Yes	Yes
Sparsity	No	No	No	No

Table 1: Comparisons of the Quantile , LUBE, QD loss and Tube loss based PI estimation models

The major problem with the LUBE cost function (15) is that it is very difficult to be optimized because, the PICP is a step function. Khosravi et al. have solved the LUBE cost function (15) using the Particle Swarm Optimization (PSO) (Kennedy & Eberhart (1995)) to estimate the PI. However, due to sub-optimal solution, high-quality PI is not always observed. Pearce et al. refine the LUBE cost function and use a sigmoidal function to approximate the PICP, allowing the application of the gradient descent method for training the NN for PI estimating in their work (Pearce et al. (2018)).

3.5 Comparison of PI estimation SVM models

In Table 1, we visualize the desirable properties for a PI estimation model and compare the SVM methods in light of them with a detailed discussion as follows.

(a) Distribution-free method : - The LS-SVR PI model assumes that the underlying noise distribution of the data is normal and may obtain poor estimate otherwise. In the literature, distribution-based PI methods often struggle to achieve consistent performance across various datasets. The SVQR, LUBE, and Tube loss methods provide PI estimation without assuming any specific distribution, allowing them to generate high-quality PI even in the presence of non-normal noise.

(b) Asymptotic guarantees:- A fundamental requirement in PI estimation models is that the obtained PI should guarantee the target coverage $1 - \alpha$ at least asymptomatically, which remain missing in the LUBE method. The LS-SVR PI model provides this guarantee only in presence of normal noise. The SVQR and Tube loss based PI methods provide this asymptotic guarantee. The asymptotic guarantee of the SVQR based PI model is implied from the asymptotic guarantee of the pinball loss function minimizer for quantile regression detailed in (Takeuchi et al. (2006)).

(c) Direct PI estimation:- As detailed in Algorithm 1, the SVQR PI model obtains the two bounds of PI by solving pair of SVQR problems one by one. The LUBE and Tube loss-based PI models simultaneously obtain the bounds of the PI by solving a single optimization problem.

(d) PI tube movement:- The PI tube movement is one another important desirable feature for PI estimation. This movement allows the PI to pass through the denser regions of the data cloud, helping to minimize the width of the PI, while achieving the target coverage. The centered PI is ideal only in the presence of the symmetric noise. However, in the presence of asymmetric noise in the data, the width of the prediction interval (PI) can be reduced by shifting it upward or downward without compromising its coverage. The SVQR and Tube loss-based PI models enable PI movement through their parameter $\hat{q}$ and r respectively.

(e) Global Optimal Solution:- One of the attractive feature of the standard SVM methods is that they guarantee a global optimal solution by solving a convex optimization problem. However, in PI estimation, only the SVQR and LS-SVR based PI model maintains this guarantee by minimizing the convex loss function in its optimization problems. The Tube loss problem (14) is non-convex and hence fails to guarantee the global optimal solution. Furthermore, the LUBE problem (15) is highly discontinuous and relies on meta-heuristic algorithms for its solution. It often makes the LUBE solution suboptimal, resulting in poor PI quality.

(f) Re-calibration:- As detailed in Algorithm 1, the SVQR PI model obtains the bound of PI by solving the pair of SVQR problems one-by one. It can not explicitly minimize the width of PI in its optimization problem. This limitation prevents SVQR PI from using the recalibration feature. In recalibration, PI models are retrained to reduce interval width, when empirical coverage obtained on validation set exceeds the target $1 - \alpha$ significantly. During retraining, the PI models increases the value of the parameter (δ in case of the Tube loss) that trade-off the width of the PI against the coverage in the optimization problem. The LUBE and Tube loss-based models explicitly incorporate the minimization of prediction interval (PI) width in their problems, thereby enabling recalibration, which is practically useful for further reducing the PI width.

(g) Sparsity:- Sparsity is yet another promising feature offered by the initial SVM models developed for the classification and regression. However, it remains missing with all PI estimation models developed in the SVM framework.

Only few of components of a sparse vector is non-zero. A sparse solution in a learning model is desirable and important as it offers significant advantages. In case of the linear model, it can help us to drop irrelevant features making the learning task more simple and effective. Further, in case of learning non-linear function also, it reduces the complexity of the model without compromising its predictive ability on train set that makes the model own better generalization ability on unseen test points.

In the SVM literature, two primary approaches are commonly used to obtain sparse solutions, which often leads to confusion. The first approach minimizes the empirical risk with an additional L_1-norm regularization term, while the second relies on the ϵ-insensitive loss function to minimize empirical error. However, when estimating a linear regression model of the form $w^T x + b$, with $w \in \mathbb{R}^n$, the latter approach can only induce sparsity in the dual variables (Lagrangian multipliers) and does not guarantee sparsity in w. As a result, it cannot eliminate irrelevant features during the learning process. In contrast, explicitly minimizing the L_1-norm of $w \in \mathbb{R}^n$ within the optimization problem directly enforces sparsity in w, thereby enabling feature selection.

4 Sparse PI estimation in SVM

In this section, we develop the Sparse Support Vector Quantile Regression (SSVQR) based PI estimation model. The SSVQR PI model inherits all properties of SVQR listed in Table 1 and also brings the sparse solution as well. Additionally, it addresses the feature selection problem in PI estimation efficiently and also can obtain the sparse estimates in CR setting.

In literature, there are few works that obtain the sparse solution while minimizing the pinball loss function or its variants. Some of the literature such as (Taylor (2000); Rastogi et al. (2018)) obtains the sparse solution for the classification task in the SVM framework while minimizing the variant of the pinball loss function. In (Tanveer et al. (2021)), authors have obtained the sparse solution by minimizing a variant of the pinball loss for clustering problem. For quantile estimation, there are a few literature like (Anand et al. (2020); Ye et al. (2025)) which attempt to develop the ϵ-insensitive variant of the pinball loss function, inspired by the ϵ-insensitive loss function used in standard SVM regression (Vapnik (2013)).

However, as discussed in above section, the minimization of the ϵ-insensitive loss function can not directly eliminate the features. In light of the above, we minimize a pair of pinball loss functions with an added L_1-norm regularization to obtain a sparse SVM solution for the PI estimation task. Li & Zhu (2008) have also employed the pinball loss function with L_1-norm regularization for sparse quantile estimation. However, for a stable solution, we reformulate the problem as a linear programming problem (LPP) and further extend it to address feature selection in PI estimation.

4.1 Sparse Support Vector Quantile Regression model

The SSVQR minimizes the pinball loss function with L_1- norm regularization. It seeks the solution of the following problem

$$\min_{(w,b)} \frac{1}{2}||w||_1 + C\sum_{i=1}^{m} \rho_q(y_i - (w^T\phi(x_i) + b)), \tag{16}$$

where $C \geq 0$ is the user defined parameter for trading of the regularization against the empirical loss. The solution to problem (16) is sparse, similar to LASSO regression, as the minimization of the L_1-norm regularization compels the weight coefficients to shrink to zero.

With the help of the kernel trick, the representor theorem rewrites the estimated function $f(x) = (w^T\phi(x)+b)$ as $\sum_{j=1}^{m} k(x_j, x)u_j + b$. Also, the pinball loss $\rho_q(u)$ in (1) is equivalent to $q\max(u, 0) + (1 - q)\max(-u, 0)$. Further, let us consider $u_i = y_i - \Big(\sum_{j=1}^{m} k(x_j, x_i)u_j + b\Big)$ and two slack variables $\xi_i = max(u_i, 0)$ and $\xi_i^* = max(-u_i, 0)$ to measure the error corresponding to the data points lying above and lower side of the regression function respectively.

Now, the problem (16) can be written as

$$\begin{aligned}
&\min_{(u,b)} \frac{1}{2}||u||_1 + C\sum_{i=1}^{m}(q\xi_i + (1-q)\xi_i^*),\\
&\text{subject to,}\\
&\xi_i = \max\Big(y_i - \Big(\sum_{j=1}^{m} k(x_j, x_i)u_j + b\Big), 0\Big), i = 1, 2, ...m,\\
&\xi_i^* = \max\Big(\Big(\sum_{j=1}^{m} k(x_j, x_i)u_j + b\Big) - y_i, 0\Big), i = 1, 2, ...m,
\end{aligned}$$

that can be simplified as

$$\begin{aligned}
&\min_{(u,b,\xi,\xi^*)} \frac{1}{2}||u||_1 + C\sum_{i=1}^{m}(q\xi_i + (1-q)\xi_i^*)\\
&\text{subject to,}\\
&\qquad \xi_i \geq y_i - \Big(\sum_{j=1}^{m} k(x_j, x_i)u_j + b\Big), \quad \xi_i \geq 0, \ i = \ 1, 2, ..m,\\
&\qquad \xi_i^* \geq \Big(\sum_{j=1}^{m} k(x_j, x_i)u_j + b\Big) - y_i, \quad \xi_i^* \geq 0, \ i = \ 1, 2, ..m,
\end{aligned}$$

which can be written as

$$\begin{aligned}
&\min_{(u,b,\xi,\xi^*)} \frac{1}{2}||u||_1 + C\sum_{i=1}^{m}(q\xi_i + (1-q)\xi_i^*)\\
&\text{subject to,}\\
&\quad y_i - \Big(\sum_{j=1}^{m} k(x_j, x_i)u_j + b\Big) \leq \xi_i,\\
&\quad \Big(\sum_{j=1}^{m} k(x_j, x_i)u_j + b\Big) - y_i \leq \xi_i^*,\\
&\quad \xi_i, \xi_i^* \geq 0, \ i = \ 1, 2, ..m.
\end{aligned} \tag{17}$$

Without loss of generality, let us consider the solution vector $u = r - p$, where r and p are vectors of positive numbers i,e., $r_i, p_i > 0, i = 1, 2, .., m$, then similar to the (Mangasarian et al. (2006)), the problem (17) can

be expressed as

$$\min_{(r,p,b,\xi,\xi^*)} \frac{1}{2}\sum_{i=1}^{m}(r_i+p_i) + C\sum_{i=1}^{m}(q\xi_i + (1-q)\xi_i^*)$$
$$\text{subject to,}$$
$$y_i - \Big(\sum_{j=1}^{m} k(x_j,x_i)(r_j-p_j)+b\Big) \le \xi_i,$$
$$\Big(\sum_{j=1}^{m} k(x_j,x_i)(r_j-p_j)+b\Big) - y_i \le \xi_i^*,$$
$$\xi_i, \xi_i^*, r_i, p_i \ge 0,\ i = 1,2,..m. \tag{18}$$

The above problem (18) is a LPP with $4m$ variables, $2m$ linear constraints and $4m$ non-negative constraints, which can be efficiently solved by any LPP solver. The optimal solution (r^*, p^*, b^*) of the LPP (18) determines the estimate of the qth quantile function using

$$f_q(x) = \sum_{i=1}^{m}(r_i^* - p_i^*)k(x_i,x) + b. \tag{19}$$

The asymptotical properties of the SSVQR model remain similar to the SVQR model detailed in (Takeuchi et al. (2006)).

4.2 PI estimation through SSVQR

We detail the algorithm for PI estimation through SSVQR in 2.The SSVQR PI preserves the properties of the SVQR PI, including the global optimal solution, PI tube movement, asymptotic guarantees, and distribution-free estimation, while also achieving a sparse solution.

Algorithm 2 PI estimation through SSVQR

1: **procedure** :- PI THROUGH SSVQR$(T, 1-\alpha)$
2: Choose some $\bar{q} \in (0, 1-\alpha)$
3: **for** each $q \in \{\bar{q}, (1+\bar{q}-\alpha)\}$ **do**
4: Solve the LPP problem (18) by tuning the value of C. Obtain the solution (r^*, p^*, b^*).
5: Estimate the function $f_q(x)$ using the (19).
6: **return** $(f_{\bar{q}}(x), f_{1+\bar{q}-\alpha}(x))$

4.3 Feature selection in PI estimation through SSVQR

Similar to other machine learning tasks, the PI estimation in high-dimensional settings also presents several challenges. The increased dimensionality not only increase the complexity of the PI bounds but also necessitates a larger sample size to ensure the quality of the estimate PI. Therefore, an efficient feature selection method is crucial for reducing the overall complexity of the PI estimation, particularly when dealing with high-dimensional data.

We detail the feature selection algorithm through SSVQR model for the linear PI estimation task in Algorithm 3. Here, a linear PI refers to the PI, where both bounds are linear functions of the input variables ,i,e. $f_{\bar{q}}(x) = w_{\bar{q}}^T x + b_{\bar{q}}$ $f_{1+\bar{q}-\alpha}(x) = w_{1+\bar{q}-\alpha}^T x + b_{1+\bar{q}-\alpha}$.

The input of Algorithm 3 is the training set $T' = \{(x_i, y_i), x_i \in \mathbf{R}^n, y_i \in \mathbf{R}, i = 1,2,....m\}$, specified confidence $(1-\alpha)$, and a very small number ϵ. It finally returns the selected feature set. In the next section, we have explored several real-world datasets with numerous features and successfully perform feature selection using the SSVQR method for linear PI estimation, without compromising the quality of the estimated PI.

Algorithm 3 Feature selection through SSVQR

procedure :- FEATURE SELECTION THROUGH SSVQR$(T, 1-\alpha, \epsilon)$
  Choose some $\bar{q} \in (0, 1-\alpha)$
  for each $q \in \{\bar{q}, (1+\bar{q}-\alpha)\}$ **do**
    Consider the linear kernel $k(x_i, x_j) = x_i^T x_j$ at the LPP (18).
    Solve the LPP (18) and obtain its solution (r^*, p^*, b^*).
    Obtain the w_q using $[x_1, x_2, ..., x_m](r^* - p^*)$.
  Compute $I_{\bar{q}} = \{i :, |w_{\bar{q}}(i)| \leq \epsilon\}$ and $I_{1+\bar{q}-\alpha} = \{i :, |w_{1+\bar{q}-\alpha}(i)| \leq \epsilon\}$
  Compute $I = I_{\bar{q}} \cap I_{1+\bar{q}-\alpha}$ and *Feature Set* $= \{1, 2, ..., m\} - I$
  return (*Feature Set*)

4.4 Conformal Regression in SVM

Finally, we extend the SVM models in CR setting for obtaining the finite sample test set guarantees. In split CR setting, we detail the SVM based CR algorithm as follows.

Algorithm 4 CR through SVQR

procedure :- CR THROUGH SVQR$(T, 1-\alpha)$
  Split the training set T' into I_1 and calibration set I_2.
  Choose some $\bar{q} \in (0, 1-\alpha)$
  for each $q \in \{\bar{q}, (1+\bar{q}-\alpha)\}$ **do**
    Solve the QPP problem (4) on I_1 by tuning the value of C. Obtain the solution (α^*, β^*).
    Estimate the function $f_q(x)$ using the (5).
  Evaluate the nonconformity score $E_i = max\{f_{\bar{q}}(x) - y_i,\ y_i - f_{1-\bar{q}+\alpha}(x)\}$ on I_2.
  Compute the $Q_{1-\alpha}(E, I_2) = (1-\alpha)(1 + \frac{1}{|I_2|})$-th empirical quantile of the non-conformity score E_i.
  return $C(x_{m+1}) = [f_{\bar{q}}(x_{m+1}) - Q_{(1-\alpha)}(E, I_2), f_{1-\bar{q}+\alpha}(x_{m+1}) + Q_{1-\alpha}(E, I_2)]$

Compared to neural network (NN)-based CR models, the SVM-based CR model offers not only greater interpretability but also more stable prediction sets. In contrast, NN-based CR models often produce varying prediction set across different training runs, even when trained on the same data splits (I_1 and I_2) and with identical hyperparameter settings. It is because that unlike SVMs, NN models often converge to different local optima during training. We empirically verify these advantages of the SVM based CR model in next section.

5 Experimental Results

In this section, we present the numerical results to analyze the quality of the PI obtained by the different SVM models through a series of experiments on simulated/artificial datasets. But, before this, we outline the objective of our experiments in view of our claim as follows.

(i) We need to evaluate the effectiveness of the SSVQR model and assess the quality of its PI estimate relative to other existing PI methods in SVM on simulated and benchmark datasets.

(ii) We aim to validate the effectiveness of Algorithm 3 for feature selection by demonstrating its ability to identify relevant features for PI estimation. Furthermore, we check that whether it improves the quality of PIs, particularly in high-dimensional real-world benchmark datasets under the linear PI estimation setting.

(iii) We need to compare the SVM based CR and NN based CR model to verify our claim that prediction sets obtained by SVM based CR model are more stable and less uncertain than that of NN.

(iv) We need to check the potential of SVM models for probabilistic foretasting tasks against the recently developed deep learning solutions.

Evaluation metrics and criteria

Now, we describe in detail the evaluation criteria that will be used for our experiments. In all our experiments, the objective is to estimate PI with a confidence level of either $1-\alpha = 0.95$ or $1-\alpha = 0.90$. We focus on estimating **centered** PIs throughout. For instance, to construct a PI with a 95% confidence level, we seek the pair of quantile functions $(f_q(x), f_{q+0.95}(x))$, where $0 \leq q \leq 0.05$.

In case of artificial datasets, we know the noise distribution and the true quantile function can be easily computed by the inverse of the cumulative distribution function. Therefore, the quality of the quantile function can be accurately assessed by computing the RMSE between the true and estimated quantile functions.

In the absence of information about the noise distribution, the quality of the quantile function can be evaluated using Coverage Probability (CP). For given a test set $\{(x_i, y_i) : x_i \in \mathbf{R}^n, y \in \mathbf{R}, i = 1, 2, ...k$ and estimate of the q_{th} quantile function $f_q(x)$, the CP is computed by $\frac{|\{y_i : y_i \leq f_q(x_i)\}|}{k}$. It measures the fraction of y values falling below the estimated quantile function.

For evaluating the overall quality of PI estimation, we use PICP and MPIW as assessment criteria. If the $[f_q(x), f_{1+q-\alpha}(x)]$ is the estimate of the PI, then the PICP on test set is computed by $\frac{|\{y_i : f_q(x_i) \leq y_i \leq f_{1+q-\alpha}(x_i)\}|}{k}$ and MPIW is given by $\frac{|f_{1+q-\alpha}(x_i)\} - f_q(x_i)|}{k}$.

An effective PI method must achieve the target $1-\alpha$ calibration while minimizing the PI width, as measured by the MPIW value. Furthermore, we define the Prediction Interval Coverage Error (PICE) as $\max(0, (1-\alpha) - PICP)$ to quantify the extent to which the model falls short of the target calibration (1-α). For comparing PI estimation models, a natural decision criterion is that the model with the lowest PICE should be considered the best. If all models successfully achieve the target calibration, the one with the minimum MPIW should be deemed the most optimal.

5.1 PI estimation through SSVQR model

At first, we shall perform experiments with the SSVQR PI estimation model on artificial and benchmark datasets.

Baseline methods

To evaluate the quality of the PIs produced by the SSVQR model, it is essential to establish appropriate baseline methods for comparison. One of the key strengths of SVM machines is their ability to always obtain the global optimal solution, maintaining their relevance and applicability in modern cutting-edge technology. As detailed in Table 1, PI estimation through SVQR and LS-SVR only obtains the global optimal solution but lacks the sparsity. In contrast, the SSVQR PI can obtain the optimal global solution as well as the sparse solution. In view of this, we find the SVQR and LS-SVR model are qualified enough to be compared with the SSVQR model for the PI estimation task in SVM framework. The Tube and LUBE loss PI estimation methods available in SVM framework do not guarantee the global optimal solution and their solution may vary with the choice of initialization. However, we have considered the Tube loss and an improved version of the LUBE loss, the QD loss function (Pearce et al. (2018)) in deep forecasting models to compare their performance with the SVM based probabilistic forecasting models.

Experimental Setup and Parameter Tunning

All SVM based PI estimation models have been implemented in MATLAB. To estimate both bounds of the PI, we utilize the RBF kernel, defined as $k(x_i, x_j) = e^{-\gamma||x_i - x_j||^2}$. As detailed in Algorithm 1 and Algorithm 2, SVQR and SSVQR require solving the QPP (4) and LPP (18) twice to obtain the quantile bounds of the PI respectively. We solve the QPPs of the SVQR PI model and the LPPs of the SSVQR model in MATLAB using the 'quadprog' and 'linprog' packages respectively.

The 'linprog' function employs the "dual-simplex-highs" algorithm by default, which may not scale efficiently for very large datasets. A more suitable open-source alternative for solving the LPPs in the SSVQR PI

	$(\bar{q},\ 1+\bar{q}-\alpha)$	RMSE(Lw,Up)	Spar (Lw,Up)	CP (Lw,Up)	PICP	PICE	MPIW	Time (s)
SVQR	(0.01, 0.96)	(1.8021, 1.4446)	(0%, 0%)	(0.0110, 0.9680)	0.957	0	2.9054	**0.4783**
	(0.015, 0.965)	(1.7046, 1.4653)	(0%, 0%)	(0.0150, 0.9720)	0.957	0	2.8244	0.5485
	(0.02, 0.97)	(1.6519, 1.4799)	(0%, 0%)	(0.0180, 0.9720)	0.954	0	2.7855	0.4712
	(0.025, 0.975)	(1.5857, 1.5016)	(0%, 0%)	**(0.0220, 0.9760)**	**0.954**	0	2.7379	0.4928
	(0.03, 0.98)	**(1.5030, 1.5667)**	(0%, 0%)	(0.0250, 0.9800)	0.955	0	**2.7212**	0.6054
SSVQR	(0.01, 0.96)	(1.8060, 1.4418)	(15%, 18%)	(0.0110, 0.9680)	0.957	0	2.9056	0.6157
	(0.015, 0.965)	(1.7051, 1.4714)	(15%, 18%)	(0.0150, 0.9700)	0.955	0	2.8275	0.5844
	(0.02, 0.97)	(1.6497, 1.4861)	(15%, 18%)	0.0160, 0.9720	0.956	0	2.7939	0.6169
	(0.025, 0.975)	**(1.5517, 1.4978)**	(15%, 20%)	**(0.0250, 0.9750)**	**0.95**	0	**2.6995**	0.5546
	(0.03, 0.98)	(1.5133, 1.5604)	**(16%, 20%)**	(0.0290, 0.9800)	0.951	0	2.7288	**0.515**

Table 2: Performance of the SVQR and SSVQR PI models on AD 1 dataset.

	$(\bar{q},\ 1+\bar{q}-\alpha)$	RMSE(Lw,Up)	Spar (Lw,Up)	CP (Lw,Up)	PICP	PICE	MPIW	Time (s)
SVQR	(0.01, 0.96)	**(4.2653, 5.7073)**	(0%, 0%)	(0.0100, 0.9470)	0.937	0.013	**8.3755**	0.805
	(0.015, 0.965)	(4.1993, 6.0698)	(0%, 0%)	(0.0210, 0.9520)	0.931	0.019	8.6865	0.699
	(0.02, 0.97)	(4.1312, 6.5751)	(0%, 0%)	(0.0220, 0.9610)	**0.939**	**0.011**	9.1619	0.6595
	(0.025, 0.975)	(4.0525, 6.8163)	(0%, 0%)	(0.0340, 0.9650)	0.931	0.019	9.3256	0.719
	(0.03, 0.98)	(4.0251, 7.4745)	(0%, 0%)	(0.0350, 0.9720)	0.937	0.013	9.978	0.6809
SSVQR	(0.01, 0.96)	**(4.1628, 5.5481)**	(20%, 15%)	(0.0080, 0.9390)	0.931	0.019	**8.1036**	**0.502**
	(0.015, 0.965)	(4.0206, 5.9335)	(18%, 18%)	(0.0100, 0.9520)	0.942	0.008	8.346	0.5173
	(0.02, 0.97)	(4.0156, 6.3077)	**(20%, 25%)**	**(0.0110, 0.9640)**	**0.953**	**0**	8.7801	0.5128
	(0.025, 0.975)	(3.9444, 7.1596)	(15%, 20%)	(0.0240, 0.9680)	0.944	0.006	9.5723	0.4694
	(0.03, 0.98)	(3.9309, 7.3834)	(20%, 20%)	(0.0290, 0.9740)	0.945	0.005	9.8075	0.5113

Table 3: Performance of the SVQR and SSVQR PI models on AD 2 dataset.

model is the "HiGHS" solver, which is specifically designed to handle large-scale linear programming problems involving thousands or even millions of variables and constraints with high efficiency.

The SVQR problem(4) or SSVQR (18) problem requires the supply of the two user defined parameters C and RBF kernel parameter γ for non-linear PI estimation. We have tunned the value of the these parameters using the grid search in the search space $\{2^{-8}, 2^{-7}, ..., 2^{7}, 2^{8}\} \times \{2^{-8}, 2^{-7}, ..., 2^{7}, 2^{8}\}$.

Artificial Datasets

First, we generate six distinct artificial datasets. In each dataset, the values of x_i are randomly sampled from a uniform distribution $U(-5, 5)$, while the corresponding y_i values are obtained by polluting the function $(1 - x_i + 2x_i^2)e^{-0.5x_i^2}$ with different types of noise as described below.

$$\textbf{AD 1:-}\quad y_i = (1 - x_i + 2x_i^2)e^{-0.5x_i^2} + \xi_i, \text{where}\ \ \xi_i \sim N(0, 0.6)$$
$$\textbf{AD 2:-}\quad y_i = (1 - x_i + 2x_i^2)e^{-0.5x_i^2} + \xi_i, \qquad \text{where}\ \ \xi_i \sim \chi^2(3)$$
$$\textbf{AD 3:-}\quad y_i = (1 - x_i + 2x_i^2)e^{-0.5x_i^2} + \xi_i, \text{where}\ \ \xi_i \sim N(0, 0.4)$$
$$\textbf{AD 4:-}\quad y_i = (1 - x_i + 2x_i^2)e^{-0.5x_i^2} + \xi_i, \text{where}\ \ \xi_i \sim N(0, 0.8)$$
$$\textbf{AD 5:-}\quad y_i = (1 - x_i + 2x_i^2)e^{-0.5x_i^2} + \xi_i, \text{where}\ \ \xi_i \sim U(-5, 5)$$
$$\textbf{AD 6:-}\quad y_i = (1 - x_i + 2x_i^2)e^{-0.5x_i^2} + \xi_i, \text{where}\ \ \xi_i \sim U(-4, 4)$$

In case of each dataset, 2500 data points are generated in which 1000 data points are considered for training and rest of them are considered for testing.

Artificial Datasets Results, Discussion and Analysis

We present the performance of the SVQR and SSVQR model for PI estimation task with the different values of $\bar{q}$ for each of simulated dataset in Table 2-7. The leftmost column of these Tables list the different pairs of target quantiles $(\bar{q},\ 1+\bar{q}-\alpha)$, required for the PI estimation with 0.95 confidence level. As detailed in Section 5.1 of this paper, for artificial datasets, the quality of the estimated upper and lower quantiles of the PI can be best evaluated by computing the RMSE between the estimated quantiles and their corresponding true quantile functions. The third column of the Tables 2-7 list these RMSE for different values of $\bar{q}$ and mean

	$(\bar{q},\ 1+\bar{q}-\alpha)$	RMSE(Lw,Up)	Spar (Lw,Up)	CP (Lw,Up)	PICP	PICE	MPIW	Time (s)
SVQR	(0.01, 0.96)	(1.7894, 1.5837)	(0%, 0%)	(0.0120, 0.9570)	0.945	0.05	2.9	0.4872
	(0.015, 0.965)	(1.7355, 1.6297)	(0%, 0%)	**(0.0130, 0.9590)**	**0.946**	**0.004**	2.887	0.5077
	(0.02, 0.97)	**(1.5696, 1.6407)**	(0%, 0%)	(0.0230, 0.9590)	0.936	0.014	2.7223	0.5004
	(0.025, 0.975)	(1.5070, 1.6927)	(0%, 0%)	(0.0280, 0.9680)	0.94	0.01	**2.71**	0.4907
	(0.03, 0.98)	(1.4694, 1.7477)	(0%, 0%)	(0.0300, 0.9750)	0.945	0.005	2.7394	**0.4745**
SSVQR	(0.01, 0.96)	1.8411, 1.5781	20%, 40%	0.0120, 0.9570	0.945	0.005	2.9462	**0.4367**
	(0.015, 0.965)	(1.8297, 1.5917)	(40%, 30%)	**(0.0120, 0.9570)**	**0.945**	**0.005**	2.9458	0.4563
	(0.02, 0.97)	(1.6999, 1.6969)	(40%, 40%)	(0.0180, 0.9600)	0.942	0.008	2.9026	0.4671
	(0.025, 0.975)	(1.6509, 1.7191)	**(40%, 40%)**	(0.0220, 0.9620)	0.94	0.01	2.8817	0.4501
	(0.03, 0.98)	**(1.4865, 1.7721)**	(30% , 40%)	(0.0290, 0.9700)	0.941	0.009	**2.7739**	0.4717

Table 4: Performance of the SVQR and SSVQR PI models on AD 3 dataset.

	$(\bar{q},\ 1+\bar{q}-\alpha)$	RMSE(Lw,Up)	Spar (Lw,Up)	CP (Lw,Up)	PICP	PICE	MPIW	Time (s)
SVQR	(0.01, 0.96)	(2.5953, 2.1197)	(0%, 0%)	(0.0160, 0.9530)	0.937	0.013	4.1549	0.723
	(0.015, 0.965)	(2.5595, 2.1292)	(0%, 0%)	(0.0160, 0.9530)	0.937	0.013	4.1268	0.6713
	(0.02, 0.97)	(2.4221, 2.2193)	(0%, 0%)	**(0.0200, 0.9610)**	**0.941**	**0.009**	4.0817	0.6704
	(0.025, 0.975)	**(2.2561, 2.2555)**	(0%, 0%)	(0.0290, 0.9630)	0.934	0.016	**3.9365**	0.6815
	(0.03, 0.98)	(2.2211, 2.3699)	(0%, 0%)	(0.0310, 0.9670)	0.936	0.014	4.0273	**0.6343**
SSVQR	(0.01, 0.96)	(2.5842, 2.1364)	(20%, 40%)	(0.0160, 0.9530)	0.937	0.013	4.1577	0.4704
	(0.015, 0.965)	(2.5475, 2.1911)	(20%, 20%)	(0.0160, 0.9560)	0.94	0.01	4.1802	0.4745
	(0.02, 0.97)	(2.4939, 2.2455)	(20%, 20%)	**(0.0190, 0.9610)**	**0.942**	**0.008**	4.183	0.4818
	(0.025, 0.975)	**(2.3617, 2.3037)**	**(40%, 20%)**	(0.0240, 0.9640)	0.94	0.01	**4.1002**	**0.4662**
	(0.03, 0.98)	(2.2689, 2.4147)	(40%, 20%)	(0.0290, 0.9700)	0.941	0.009	4.121	0.4884

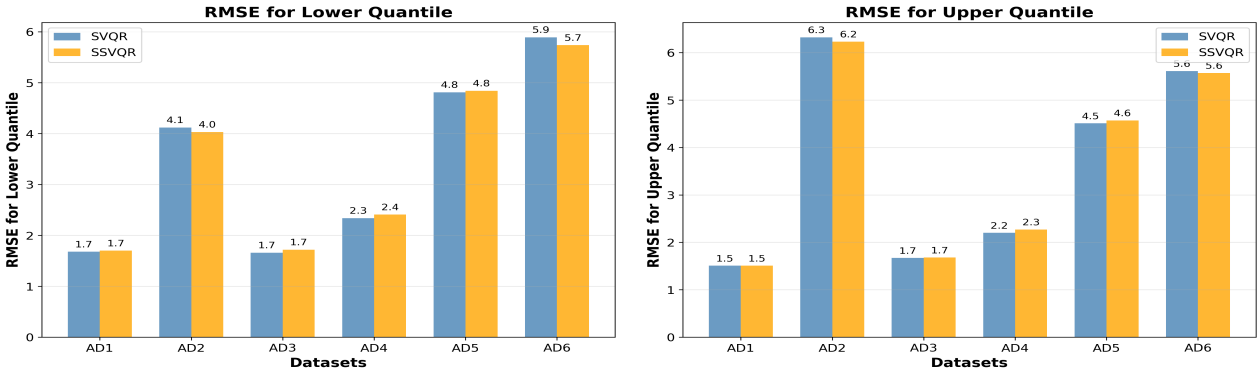
Table 5: Performance of the SVQR and SSVQR PI models on AD 4 dataset.

	$(\bar{q},\ 1+\bar{q}-\alpha)$	RMSE(Lw,Up)	Spar (Lw,Up)	CP (Lw,Up)	PICP	PICE	MPIW	Time (s)
SVQR	(0.01, 0.96)	(4.9694, 4.4016)	(0%, 0%)	**(0.0100, 0.9560)**	**0.946**	**0.004**	8.1044	0.7917
	(0.015, 0.965)	(4.8456, 4.4539)	(0%, 0%)	(0.0180, 0.9600)	0.942	0.008	8.0257	0.8009
	(0.02, 0.97)	(4.8100, 4.4860)	(0%, 0%)	(0.0190, 0.9620)	0.943	0.007	8.0223	0.736
	(0.025, 0.975)	**(4.7111, 4.5143)**	(0%, 0%)	(0.0280, 0.9640)	0.936	0.014	**7.943**	0.6895
	(0.03, 0.98)	(4.6921, 4.5368)	(0%, 0%)	(0.0290, 0.9680)	0.939	0.011	7.9481	**0.6191**
SSVQR	(0.01, 0.96)	(5.2658, 4.5044)	(40%, 30%)	(0.0050, 0.9610)	0.956	0	8.5344	0.5394
	(0.015, 0.965)	(4.8787, 4.5073)	(35%, 30%)	(0.0160, 0.9610)	0.945	0.005	8.1171	0.4835
	(0.02, 0.97)	**(4.8138, 4.5249)**	(30%, 30%)	**(0.0190, 0.9650)**	**0.946**	**0.004**	**8.0659**	**0.472**
	(0.025, 0.975)	(4.7464, 4.6037)	(25%, 30%)	(0.0260, 0.9670)	0.941	0.009	8.0692	0.5102
	(0.03, 0.98)	(4.6962, 4.7637)	**(30%, 40%)**	(0.0290, 0.9720)	0.943	0.007	8.1832	0.5068

Table 6: Performance of the SVQR and SSVQR PI models on AD 5 dataset.

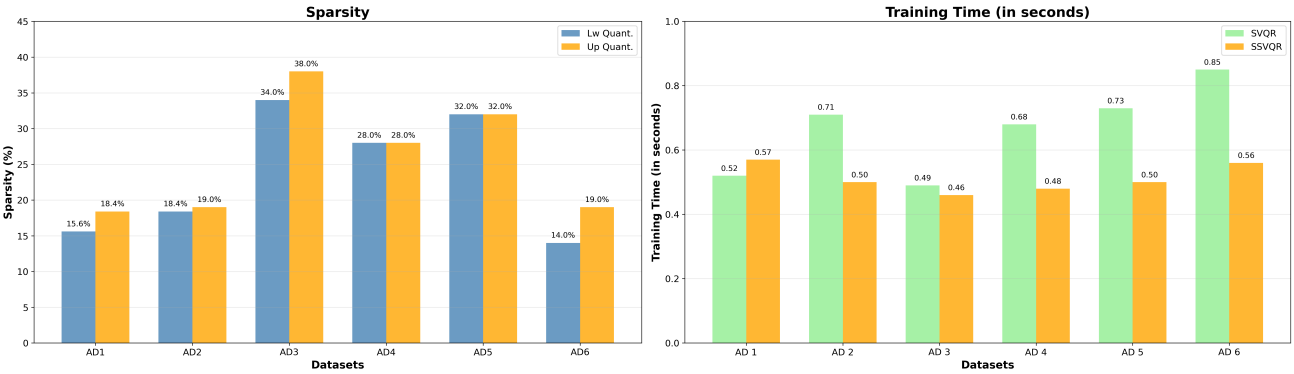
	$(\bar{q},\ 1+\bar{q}-\alpha)$	RMSE(Lw,Up)	Spar (Lw,Up)	CP (Lw,Up)	PICP	PICE	MPIW	Time (s)
SVQR	**(0.01, 0.96)**	(6.1510, 5.5057)	(0%, 0%)	**(0.0110, 0.9630)**	**0.952**	0	10.0826	**0.6437**
	(0.015, 0.965)	(6.0517, 5.5584)	(0%, 0%)	(0.0150, 0.9670)	0.952	0	**10.031**	0.6798
	(0.02, 0.97)	(5.9777, 5.6130)	(0%, 0%)	(0.0170, 0.9700)	0.953	0	10.0141	1.0752
	(0.025, 0.975)	(5.8879, 5.6868)	(0%, 0%)	(0.0230, 0.9760)	0.953	0	9.9987	0.9265
	(0.03, 0.98)	**(5.7599, 5.7156)**	(0%, 0%)	(0.0290, 0.9780)	0.949	0.001	**9.8878**	0.9445
SSVQR	(0.01, 0.96)	(5.9304, 5.4581)	(10%, 15%)	(0.0110, 0.9610)	0.95	0	9.7944	0.569
	(0.015, 0.965)	(5.9107, 5.5686)	(10%, 20%)	(0.0140, 0.9700)	0.956	0	9.9016	0.5656
	(0.02, 0.97)	(5.7836, 5.5880)	(15%, 25%)	**(0.0180, 0.9700)**	**0.952**	0	9.7807	0.5766
	(0.025, 0.975)	**(5.6486, 5.6038)**	**(20%, 20%)**	(0.0260, 0.9710)	0.945	0.005	**9.6443**	**0.5423**
	(0.03, 0.98)	(5.6184, 5.6506)	(10%, 15%)	(0.0300, 0.9760)	0.946	0.004	9.6653	0.5636

Table 7: Performance of the SVQR and SSVQR PI models on AD 6 dataset.



(a) (b)

Figure 3: Comparison of the quality of quantile function obtained by the SVQR and SSVQR PI models.



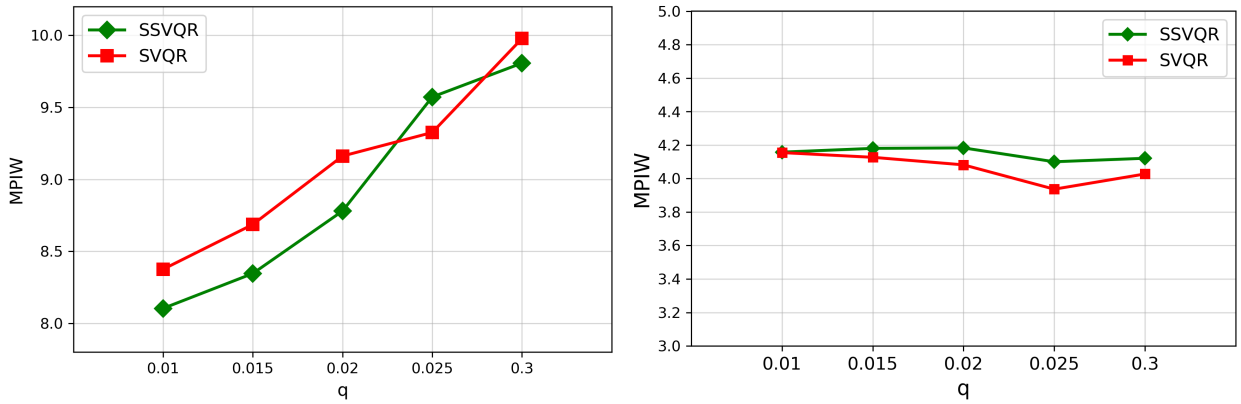
(a) (b)

Figure 4: (a) Qunatile functions estimated by the SSVQR model is sparse while SVQR model fails to obtain the sparse solution. (b) Average training time comparison of the SVQR and SSVQR models for PI estimation.

of them are plotted at Figure 3 for lower and upper quantile estimation separately for different simulated datasets. The quantile bounds estimated by the SSVQR model are comparable to, or slightly better than, those obtained from the SVQR model. Replacing L_2 regularization with L_1 regularization in the SVM quantile regression model does not result in significantly different estimates. However, the major advantage of using the SSVQR model is its ability to obtain the sparse solution vector. Figure 4(a) plots the sparsity of the solution vector corresponding to the upper and lower quantile bounds obtained by the SSVQR model. It highlights that the SSVQR model effectively reduces significant coefficients of the solution vector to near zero which enables the feature selection task in PI estimation and also simplify the overall prediction process.

We compare the overall average training time (in seconds) taken by the SVQR PI and SSVQR PI models for the PI estimation task across different artificial datasets in Figure 4(b). It shows that the SSVQR requires fewer seconds train the PI model than the SVQR model. We have solved the QPPs of the SVQR PI model and LPPs of the SSVQR model in MATLAB with 'quadprog' and 'linprog' packages respectively.

Another key observation from the numerical results in Tables 2-7 is the consistent performance of the SVQR and SSVQR PI models across different datasets. In all cases, both the SVQR and SSVQR PI models manage to approximately achieve 95% target coverage. We plot the MPIW values obtained by the SSVQR and SVQR PI models against different values of $\bar{q}$ on dataset AD2 and AD4 in Figure 5. On AD2 dataset, the MPIW values of the PI obtained by both models increase with $\bar{q}$. It is evident from Table 3 that this increase is



(a) (b)

Figure 5: Plot of the MPIW obtained by the SVQR and SSVQR PI models against $\bar{q}$ on (a) AD2 and (b) AD4 dataset.

not related to the PICP values obtained by the SVQR and SSVQR PI models. Actually, it is caused by the nature of noise present in the AD2 dataset. The AD2 dataset contains asymmetric noise from the (χ^2) distribution, leading to a higher density of data points in the lower region of the input-target space, which gradually decreases as we move upward. As $\bar{q}$ decreases, the resultant PI shifts downward, passing through denser regions of the data cloud, leading to lower MPIW values. It shows that the movement of the PI tube due to change of $\bar{q}$ values may lead to the narrower PI particularly in presence of the asymmetric noise in the data. Apart from the AD2 dataset, all other artificial datasets contain noise from symmetric distributions. In these datasets, the centered PI $((f_{0.025}(x), f_{0.975}(x))$ is expected to achieve the high quality PI. Figure 5(b) shows that at $q = 0.025$, both the SVQR and SSVQR PI models attain the lowest MPIW values on AD4 dataset.

SVM PI estimation methods on benchmark datasets

We have done experiments on the two popular benchmark datasets namely Boston Housing and Concrete and evaluate the quality of the PI estimated by the SSVQR, SVQR and LS-SVR based PI estimation methods for different value of the $\bar{q}$ with non-linear RBF kernel. Table 8 and 13 contains the numerical results obtained on the Boston Housing and Concrete datasets respectively. We can observe that the SSVQR, SVQR and LS-SVR PI models obtains a similar quality of the PI but SSVQR models always obtain the sparse solution vector.

	q	Spar (Lw, Up)	CP (Lw, Up)	PICP	PICE	MPIW	Time (s)
SVQR	(0.025, 0.925)	(0%, 0%)	(0.028, 0.930)	**0.90**	0	**28.55**	**0.1347**
	(0.05, 0.95)	(0%, 0%)	(0.052, 0.950)	0.90	0.00	33.06	0.1472
	(0.075, 0.975)	(0%, 0%)	(0.088, 0.972)	0.88	0.02	38.23	0.1670
SSVQR	**(0.025, 0.925)**	**(17%, 16%)**	(0.027, 0.927)	**0.90**	0	**28.62**	**0.0657**
	(0.05, 0.95)	(15%, 22%)	(0.048, 0.947)	0.90	0	32.77	0.0678
	(0.075, 0.975)	(14%, 28%)	(0.080, 0.967)	0.89	0.01	38.24	0.0681
LS-SVR	(0.025, 0.925)	(0%,0%)		0.91	0	31.19	**0.0064**
	(0.050, 0.950)	(0%,0%)		**0.91**	0	**30.18**	0.0067
	(0.075, 0.975)	(0%,0%)		0.89	0.01	31.19	0.0060

Table 8: Comparison of different SVM PI estimation methods on Boston Housing

5.2 Feature Selection through SSVQR

Next, we apply our SSVQR model for feature selection in PI estimation with linear kernel.

	q	Spar (Lw, Up)	CP (Lw, Up)	PICP	PICE	MPIW	Time (s)
SVQR	(0.025, 0.925)	(0%, 0%)	(0.0219, 0.8997)	0.8777	0.0723	32.5492	2.6141
	(0.05, 0.95)	(0%, 0%)	(0.0408, 0.9436)	**0.9028**	**0.0472**	**28.6541**	2.8405
	(0.075, 0.975)	(0%, 0%)	(0.0690, 0.9561)	0.8871	0.0629	32.1635	**2.5197**
SSVQR	(0.025, 0.925)	(12%, 12%)	(0.0340, 0.9029)	0.8689	0.0811	29.4962	0.1858
	(0.05, 0.95)	(12%, 12%)	(0.0583, 0.9272)	0.8689	0.0811	**27.8917**	**0.1795**
	(0.075, 0.975)	**12%, 12%**	(0.0777, 0.9515)	**0.8738**	0.0762	30.2547	0.1837
LS-SVR	(0.025, 0.925)	0%,0%		0.9126	0.0374	28.2429	**0.4125**
	(0.050, 0.950)	(0%,0%)		0.8932	0.0568	**27.3308**	0.4631
	(0.075, 0.975)	(0%,0%)		**0.9029**	0.0471	28.2429	0.3954

Table 9: Comparison of different SVM PI estimation methods on Concrete dataset

Experimental Setup:- All of the simulations were performed in the MATLAB. We use the 80% of data points for training and use rest of them for testing. We set the target calibration $1-\alpha = 0.95$. Following Algorithm 3, we perform feature selection using the SSVQR PI model for $\bar{q} = 0.025$ and present the results in Table 10. Since Algorithm 3 solves the problem 18 with linear kernel twice, we need to tune only the parameter C for each $\bar{q}$ which was selected from the grid of $\{2^i : i = -15, ..., 15\}$.

Datasets:- we use five popular real-world benchmark datasets namely Spambase (4601×58), Student Performance (395×16) , Boston Housing (505×14), UCI-secom (1568×591) and MADELON (2000×500). for the feature selection task using our proposed algorithm.

Analysis: - The numerical results presented in Table 10 clearly demonstrate that the SSVQR PI feature selection (as detailed in Algorithm 3) can significantly reduce the number of features while maintaining the quality of the PI, as measured by PICP and MPIW. On average, it could reduce the 69% of features on considered datasets and reduce the complexity of the PI estimation task significantly. The training time of the PI estimating task improves significantly after dropping irrelevant features through Algorithm 3. On average, it could improve the 53% improvement in training time. Also, reducing the significant numbers of features will reduce the tunning and testing time of the PI model.

For high-dimensional datasets such as UCI secom and MADELON, feature selection leads to a significant improvement in the quality of the estimated PI in terms of its coverage. The SSVQR based feature selection algorithm eliminates a significant number of irrelevant features and learn the PI in relatively much lower dimension which results in better PI coverage with linear functions. Table 11 lists the features dropped by the SSVQR-PI feature selection algorithm for each dataset.

Dataset	**Dimension**	**Before Feature Selection**			**After Feature Selection**			**% Reduced**
		PICP	**MPIW**	**Train Time(s)**	**PICP**	**MPIW**	**Train Time(s)**	**Features**
Spambase	(4601, 58)	0.9663	0.9330	60	0.9653	0.9340	52	46%
Student Perf.	(395, 16)	0.8861	6.8429	0.48	0.8861	6.8429	0.42	73%
Boston Housing	(505, 14)	0.9406	23.0647	0.81	0.9406	23.0647	0.7	38%
uci-secom	(1568, 591)	0.8758	1.6146	16	0.9204	1.6683	7	91%
MADELON	(2000, 500)	0.7100	2.0007	58	0.9375	2.0020	11	99%

Table 10: Performance comparison before and after feature selection using the SSVQR PI model

Dataset	**Dropped Features**
Spambase	2, 8, 11, 13, 16, 17, 18, 19, 20, 21, 22, 23, 25, 27, 31, 35, 37, 46, 47, 48, 49, 50, 53, 54, 55, 56
Student Perf.	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11
Boston Housing	2, 3, 4, 7, 10
UCI Secom	1, 4-20, 22-27, 28-41, 42-50, 52-58, 60-66, 68-70, 72, 74-87, 89, 91-110, 112-132, 134, 137-138, 140-157, 159-160, 162-187, 189-203, 205-224, 226-246, 247-296, 297-332, 333-362, 364-386, 387-400, 401-418, 420-422, 424-431, 434-435, 437-438, 440-466, 467, 469-481, 483, 489-498, 501-509, 512-520, 522-538, 540-545, 546, 548-560, 563-569, 571, 573-579, 580, 582-587
MADELON	0-89, 91-227, 229-275, 277-331, 333-444, 446-499

Table 11: Feature Selection by SSVQR PI

5.3 Probabilistic Forecasting with SVM models

In this section, we compare the performance of the SSVQR, SVQR and LS-SVR model for the probabilistic forecasting.

Datasets:- We consider three popular time-series datasets namely Female Births (365×1), Minimum Temperature (3651×1) and Beer Production (464×1). We have used the 70% of dataset as training set and rest of them are used for testing. Out of the training set, the last 10% of the observations have been used for the validation set.

Baseline Methods:- We also train several recent and widely adopted deep learning models for probabilistic forecasting developed in a distribution-free setting, including Quantile-based LSTM, Tube Loss LSTM, and Quality-Driven (QD) Loss LSTM models Pearce et al. (2018) to compare their performance against the SVM-based models. The QD loss Pearce et al. (2018) is the improved version of the LUBE model which can be used minimized with the gradient descent method in deep learning architecture. Also, we have added the popular Deep AR (Salinas et al. (2020)) probabilistic forecasting method for baseline comparison.

Experimental Setup:- The SSVQR, SVQR and LS-SVR model based probabilistic forecasting model were trained in MATLAB. All of the models were asked to obtain the probabilistic forecast for target calibration $1 - \alpha = 0.95$. For this, they estimate the 0.975^{th} and 0.025^{th} quantiles of the predictive distribution. All of them requires the tuning of the two parameters namely C and RBF kernel parameter γ. We have tunned the value of the these parameters using the grid search in the search space $\{2^{-15}, 2^{-14}, ..., 2^{14}, 2^{15}\} \times \{2^{-15}, 2^{-14}, ..., 2^{14}, 2^{15}\}$.

The deep learning based probabilistic forecasting models such as Quantile-based LSTM, Tube Loss LSTM, and Quality-Driven (QD) Loss LSTM models were trained in a Python environment using the Tensor-Flow library. All of them were evaluated on the test set by effectively tuning their parameters on the validation set.

Analysis :- We present the numerical results in Table 12. One major observation from the Table 12 is that the SVM based probabilistic forecasting models obtain competitive performance compared to complex LSTM based deep forecasting models after efficient tuning of their parameters.

On the **Female Births** dataset, the SVQR models achieve the narrowest prediction interval (PI) tube while maintaining the target coverage level of 0.95, followed by the Quantile LSTM and Tube LSTM models. On the **Minimum Temperature** dataset, all SVQR models meet the target coverage, with SSVQR emerging as the best-performing model, followed by the LS-SVR model. Similarly, on the **Beer Production** dataset, all SVR-based probabilistic forecasting models achieve the desired coverage, with SVQR being the best-performing model, followed by SSVQR. Also, the SSVQR based probabilistic forecasting model obtains the sparse solution.

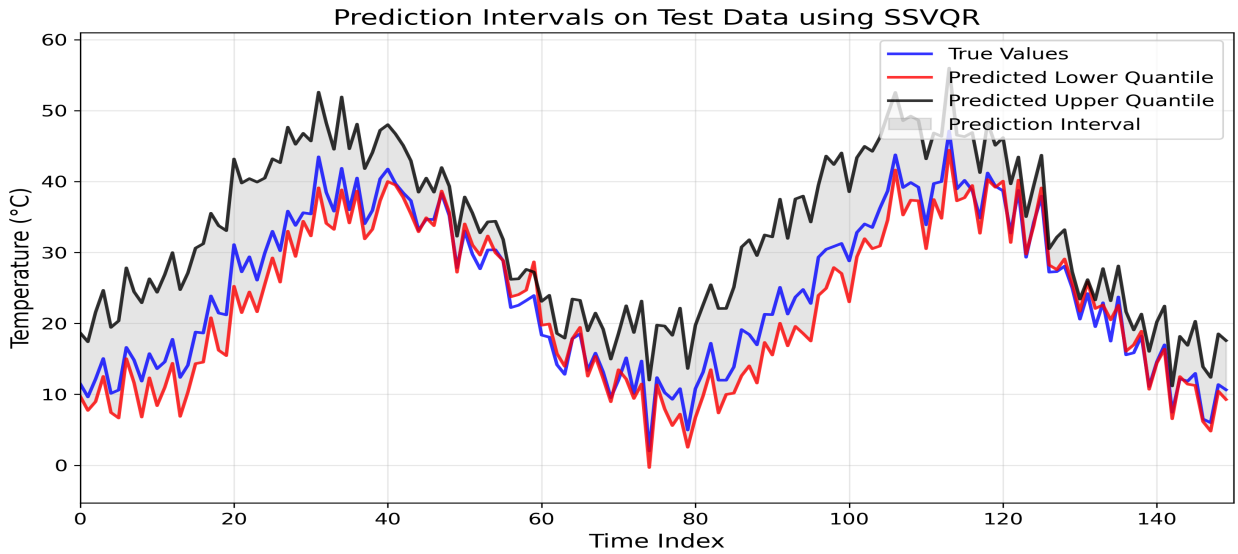
Table 13 presents the tuned neural architectures of the LSTM models, along with the number of weights to be optimized for each of the probabilistic forecasting models used. The LSTM based probabilistic forecasting models are more complex and requires the optimization of thousands of weights where as the SVQR based probabilistic models are much simpler architectures and could obtain similar quality of the PI as obtained by the LSTM based models. Figure (6) (a) and (b) shows the forecasting of the SSVQR and SVQR model on Temperature and Beer Production datasets respectively.

5.4 Numerical Results of SVM based Conformal Regression method

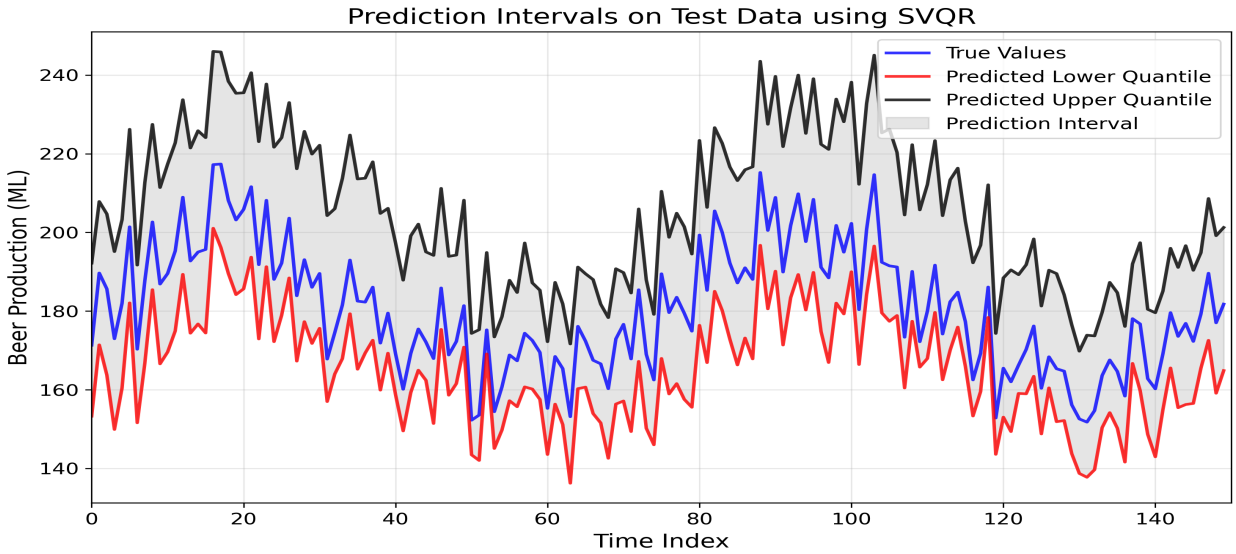
We need to empirically show that the SVM based CR model estimates are more stable and less uncertain than NN based CR estimate.

Baseline methods:- We compare performance of the SVM-based CR model (SVQR+CR) against the NN-based CQR model (CQR-NN) (Romano et al. (2019)) on several benchmark datasets under the split conformal setting, with the target coverage level $1 - \alpha = 0.90$.

Experimental Setup:- Both of the models SVQR+CR and CQR-NN were trained in Python. Both models were trained across 10 independent runs using fixed hyperparameter settings and identical splits of training



(a)



(b)

Figure 6: Probabilistic forecasting with SSVQR model on daily (a) Temperature and (b) Beer Production dataset

Dataset	Method	PICP	MPIW	Training Time	Sparsity
Female Births	SSVQR	0.93	28.00	1.12	61 %
	SVQR	**0.95**	**27.11**	**0.97**	0 %
	LS-SVR	0.95	37.70	3.60	0 %
	Quantile LSTM	0.95	28.20	118.00	–
	Tube LSTM	0.96	28.09	43.00	–
	QD LSTM	0.94	38.98	–	–
	DeepAR	0.94	29.8	55.0	0 %
Minimum Temp.	SSVQR	0.96	10.72	200.79	69 %
	SVQR	0.96	75.81	172.00	0 %
	LS-SVR	0.95	**10.59**	**0.53**	0 %
	Quantile LSTM	0.95	24.82	1135.00	–
	Tube LSTM	0.94	15.56	447.00	–
	QD LSTM	0.79	5.94	–	–
	DeepAR	0.90	12.79	58.0	0 %
Beer Prod.	SSVQR	0.96	78.21	1.77	70 %
	SVQR	**0.95**	**75.73**	**0.78**	0 %
	LS-SVR	0.96	79.48	0.29	0 %
	Quantile LSTM	0.94	134.80	132.80	–
	Tube LSTM	0.95	42.91	89.60	–
	QD LSTM	0.96	159.71	–	–
	DeepAR	0.76	12.43	62.0	0 %

Table 12: Performance comparison of SVM and deep learning based probabilistic forecasting methods on benchmark datasets.

Dataset	Model	Architecture	Weights	W. Size
Female Births	SVQR	Kernel	256	10
	SSVQR	Kernel	256	15
	LS-SVR	Kernel	256	20
	Quantile LSTM	LSTM [100]	30 K	25
	Tube LSTM	LSTM [100]	30 K	25
	QD LSTM	LSTM [100]	30 K	25
	DeepAR	LSTM [40,40]	13 K	28
Minimum Temp.	SVQR	Kernel	2 556	12
	SSVQR	Kernel	2 556	18
	LS-SVR	Kernel	2 556	22
	Quantile LSTM	LSTM [16,8]	32 K	28
	Tube LSTM	LSTM [16,8]	32 K	28
	QD LSTM	LSTM [16,8]	32 K	28
	DeepAR	LSTM [40,40]	13 K	56
Beer Prod.	SVQR	Kernel	325	8
	SSVQR	Kernel	325	14
	LS-SVR	Kernel	325	18
	Quantile LSTM	LSTM [64,32]	29 K	24
	Tube LSTM	LSTM [64,32]	29 K	24
	QD LSTM	LSTM [64,32]	29 K	24
	DeepAR	LSTM [40,40]	13 K	48

Table 13: Comparison of the complexity of used SVM and deep learning based probabilistic forecasting models on benchmark datasets. LSTM [16,8] means that the used LSTM model has two hidden layer containing 16 and 8 neurons respectively.

and calibration sets. For the experiments, each dataset was partitioned into a training set (60%), a calibration set (20%), and a test set (20%). For the SVQR+CR model, we used a Radial Basis Function (RBF) kernel with parameter $\gamma = 1.0/n_{\text{features}}$, a regularization parameter $C = 10.0$, for targeting the 0.05^{th} and 0.95^{th} quantile function. For the CQR-NN model, the architecture consists of a dense layer with 200 neurons and ReLU activation, followed by a linear output layer. The model was trained for 200 epochs using the Adam optimizer with a learning rate of 0.01 and a batch size of 40, minimizing the pinball loss.

Analysis:- The numerical results in Table 14 yield several insights. First, SVQR+CR and CQR-NN achieved the target coverage in 3 out of 5 datasets. Moreover, SVQR+CR yielded lower MPIW in 3 out of 5 cases. Most notably, the standard deviations of PICP and MPIW across the 10 runs were zero for SVQR+CR, indicating perfectly stable predictions. In contrast, CQR-NN exhibited significant fluctuations. This is

Dataset	Method	Performance			Std. Dev. (10 runs)		
		PICP (%)	MPIW	Time (s)	PICP	MPIW	Time (s)
Boston	CQR-NN	94.12	**2.19**	2.65	(1.14)	(0.12)	(0.23)
	SVQR+CP	**95.10**	2.24	**0.44**	(0.00)	(0.00)	(0.03)
Energy	CQR-NN	87.66	1.13	3.20	(1.12)	(0.03)	(0.31)
	SVQR+CP	**88.96**	**1.05**	**0.95**	(0.00)	(0.00)	(0.05)
Concrete	CQR-NN	92.58	19.62	1.89	(1.02)	(0.71)	(0.17)
	SVQR+CP	**91.35**	**18.74**	**0.38**	(0.00)	(0.00)	(0.01)
Yatch	CQR-NN	**91.69**	**2.43**	1.44	(1.49)	(0.35)	(0.12)
	SVQR+CP	90.82	2.87	**0.22**	(0.00)	(0.00)	(0.01)
Servo	CQR-NN	88.39	0.73	1.15	(2.21)	(0.10)	(0.09)
	SVQR+CP	**89.74**	**0.68**	**0.16**	(0.00)	(0.00)	(0.01)

Table 14: Comprehensive comparison of CQR-NN and SVQR+CP methods across benchmark datasets.

because, due to the non-convex nature of their optimization landscape, NN models often converge to different local minima across different training trials, even when the training data and hyperparameter settings remain fixed. Another key advantage of SVQR+CR is its reduced training time compared to CQR-NN, making it a more efficient and stable choice for conformal regression task.

6 Conclusions and Future Work

This paper presents a comprehensive roadmap for exploring UQ methods within the SVM regression framework, emphasizing its practical advantages in real-world applications. Due to the inherent stability and absence of parameter uncertainty in SVMs, the resulting PIs exhibit significantly lower overall uncertainty compared to those generated by NN. This makes SVMs a more reliable choice when accurate and stable PIs are critical for decision-making. Moreover, in probabilistic forecasting tasks, SVM based PI models can achieve comparable or even superior forecast quality while maintaining simpler model structures and requiring less training time. For high-dimensional data, we show that incorporating feature selection before training substantially improves both the PI quality and computational efficiency. In the context of conformal prediction for regression, our findings indicate that SVM models produce more stable and less uncertain prediction sets than NNs, making them a better alternative in such settings.

We develop a feature selection algorithm for PI estimation under the assumption that the bounds are linear functions of the input features. However, extending this approach to handle non-linear dependencies remains an important direction for future work, particularly in both NN and SVM-based models. Additionally, we show that SVM-based probabilistic forecasting models offer a compelling alternative to complex deep learning architectures by significantly reducing model complexity through the optimization of fewer parameters on batch datasets. Despite these advantages, they are not well-suited for dynamic or online data scenarios. To address this limitation, developing incremental or online SVM-based probabilistic forecasting models presents a promising avenue for future research. In this work, we have focused exclusively on the SVM regression model, a similar UQ analysis can be extended to SVM classification models in future research.

References

Pritam Anand, Reshma Rastogi, and Suresh Chandra. A new asymmetric -insensitive pinball loss function based support vector quantile regression model. *Applied Soft Computing*, 94:106473, 2020.

Pritam Anand, Tathagata Bandyopadhyay, and Suresh Chandra. Tube loss: A novel approach for prediction interval estimation and probabilistic forecasting. *arXiv preprint arXiv:2412.06853*, 2024.

Christopher M Bishop. Mixture density networks. *Neural Computing & Applications*, 1994.

Christopher M Bishop. Neural networks for pattern recognition. *Clarendon Press google schola*, 2:223–228, 1995.

Alex J Cannon. Quantile regression neural networks: Implementation in r and application to precipitation downscaling. *Computers & geosciences*, 37(9):1277–1284, 2011.

Qiang Cheng, Jale Tezcan, and Jie Cheng. Confidence and prediction intervals for semiparametric mixed-effect least squares support vector machine. *Pattern Recognition Letters*, 40:88–95, 2014.

Kris De Brabanter, Jos De Brabanter, Johan AK Suykens, and Bart De Moor. Approximate confidence and prediction intervals for least squares support vector regression. *IEEE Transactions on Neural Networks*, 22(1):110–120, 2010.

Richard D De VlEAUX, Jennifer Schumi, Jason Schweinsberg, and Lyle H Ungar. Prediction intervals for neural networks via nonlinear regression. *Technometrics*, 40(4):273–282, 1998.

Yaoyao He, Yang Qin, Shuo Wang, Xu Wang, and Chao Wang. Electricity consumption probability density forecasting method based on lasso-quantile regression neural network. *Applied energy*, 233:565–575, 2019.

JT Gene Hwang and A Adam Ding. Prediction intervals for artificial neural networks. *Journal of the American Statistical Association*, 92(438):748–757, 1997.

James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks*, volume 4, pp. 1942–1948. ieee, 1995.

Abbas Khosravi, Saeid Nahavandi, Doug Creighton, and Amir F Atiya. Lower upper bound estimation method for construction of neural network-based prediction intervals. *IEEE transactions on neural networks*, 22(3):337–346, 2010.

Roger Koenker and Gilbert Bassett Jr. Regression quantiles. *Econometrica: journal of the Econometric Society*, pp. 33–50, 1978.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Philippe Lauret, Mathieu David, and Hugo TC Pedro. Probabilistic solar forecasting using quantile regression models. *energies*, 10(10):1591, 2017.

Youjuan Li and Ji Zhu. L1-Norm Quantile Regression. *Journal of Computational and Graphical Statistics*, 17(1):163–185, 2008. doi: 10.1198/106186008X289155.

David JC MacKay. The evidence framework applied to classification networks. *Neural computation*, 4(5): 720–736, 1992.

Olvi L Mangasarian, Kristin P Bennett, and Emilio Parrado-Hernández. Exact 1-norm support vector machines via unconstrained convex differentiable minimization. *Journal of Machine Learning Research*, 7 (7), 2006.

James Mercer. Xvi. functions of positive and negative type, and their connection the theory of integral equations. *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character*, 209(441-458):415–446, 1909.

David A Nix and Andreas S Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 ieee international conference on neural networks (ICNN'94)*, volume 1, pp. 55–60. IEEE, 1994.

Harris Papadopoulos. Inductive conformal prediction: Theory and application to neural networks. In *Tools in artificial intelligence*. Citeseer, 2008.

Harris Papadopoulos, Kostas Proedrou, Volodya Vovk, and Alex Gammerman. Inductive confidence machines for regression. In *Machine learning: ECML 2002: 13th European conference on machine learning Helsinki, Finland, August 19–23, 2002 proceedings 13*, pp. 345–356. Springer, 2002.

Olivier C Pasche and Sebastian Engelke. Neural networks for extreme quantile regression with an application to forecasting of flood risk. *The Annals of Applied Statistics*, 18(4):2818–2839, 2024.

Tim Pearce, Alexandra Brintrup, Mohamed Zaki, and Andy Neely. High-quality prediction intervals for deep learning: A distribution-free, ensembled approach. In *International conference on machine learning*, pp. 4075–4084. PMLR, 2018.

Reshma Rastogi, Aman Pal, and Suresh Chandra. Generalized pinball loss svms. *Neurocomputing*, 322: 151–165, 2018.

Yaniv Romano, Evan Patterson, and Emmanuel Candes. Conformalized quantile regression. *Advances in neural information processing systems*, 32, 2019.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, 36(3):1181–1191, 2020.

Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. A generalized representer theorem. In *International conference on computational learning theory*, pp. 416–426. Springer, 2001.

George Seber and George AF Seber. Nonlinear regression models. *The Linear Model and Hypothesis: A General Unifying Theory*, pp. 117–128, 2015.

Nitin Anand Shrivastava, Abbas Khosravi, and Bijaya Ketan Panigrahi. Prediction interval estimation for electricity price and demand using support vector machines. In *2014 International Joint Conference on Neural Networks (IJCNN)*, pp. 3995–4002. IEEE, 2014.

Nitin Anand Shrivastava, Abbas Khosravi, and Bijaya Ketan Panigrahi. Prediction interval estimation of electricity prices using pso-tuned support vector machines. *IEEE Transactions on Industrial Informatics*, 11(2):322–331, 2015.

Johan AK Suykens, Jos De Brabanter, Lukas Lukas, and Joos Vandewalle. Weighted least squares support vector machines: robustness and sparse approximation. *Neurocomputing*, 48(1-4):85–105, 2002.

Ichiro Takeuchi, Quoc V Le, Timothy D Sears, Alexander J Smola, and Chris Williams. Nonparametric quantile estimation. *Journal of machine learning research*, 7(7), 2006.

Mohammad Tanveer, Sweta Sharma, Reshma Rastogi, and Pritam Anand. Sparse support vector machine with pinball loss. *Transactions on Emerging Telecommunications Technologies*, 32(2):e3820, 2021.

James W Taylor. A quantile regression neural network approach to estimating the conditional density of multiperiod returns. *Journal of forecasting*, 19(4):299–311, 2000.

Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic learning in a random world*, volume 29. Springer, 2005.

Volodya Vovk, Alexander Gammerman, and Craig Saunders. Machine-learning applications of algorithmic randomness. 1999.

Can Wan, Jin Lin, Jianhui Wang, Yonghua Song, and Zhao Yang Dong. Direct quantile regression for nonparametric probabilistic forecasting of wind power generation. *IEEE Transactions on Power Systems*, 32(4):2767–2778, 2016.

Ya-Fen Ye, Jie Wang, and Wei-Jie Chen. Nonlinear feature selection for support vector quantile regression. *Neural Networks*, 185:107136, 2025.

Hao Zhang, Yongqian Liu, Jie Yan, Shuang Han, Li Li, and Quan Long. Improved deep mixture density network for regional wind power probabilistic forecasting. *IEEE Transactions on Power Systems*, 35(4): 2549–2560, 2020a.

Wenjie Zhang, Hao Quan, and Dipti Srinivasan. An improved quantile regression neural network for probabilistic load forecasting. *IEEE Transactions on Smart Grid*, 10(4):4425–4434, 2018.

Wenjie Zhang, Hao Quan, Oktoviano Gandhi, Ram Rajagopal, Chin-Woo Tan, and Dipti Srinivasan. Improving probabilistic load forecasting using quantile regression nn with skip connections. *IEEE Transactions on Smart Grid*, 11(6):5442–5450, 2020b.