Support Vector Machines : A more certain estimate of uncertainty

Anonymous authors
Paper under double-blind review

Abstract

This paper explores Support Vector Machine (SVM) potential from the lens of uncertainty quantification (UQ), developed for regression and forecasting tasks. Unlike NN-based approaches commonly used for UQ estimation, SVM-based methods are more stable, sparse, interpretable, and offer well-defined optimal solutions. However, there is only a limited literature addressing 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 than, those produced by modern complex deep learning and neural network models, particularity for small and moderate scale datasets.

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 applications, the prediction of the regression model may not be perfectly accurate due to the 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 could involve a significant degree of uncertainty. In such cases, quantifying these uncertainties is crucial for making effective decisions.

The Prediction Interval (PI) estimation is the main 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 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), which quantifies the width of the PI tube.

The PI estimation models explore the different characteristics of the conditional distribution $Y|X$, rather than 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 $f_q(x)$, that should be defined as the infimum over all functions satisfying $P(y \leq f(x)|x) = q$. For time-series data,

estimating the PI for future observations using an auto-regressive approach is referred to as probabilistic forecasting.

A well-calibrated High Quality (HQ) PI guarantees the target coverage level t only asymptotically. However, it may fail to achieve it on finite test samples. In real-world decision-making, especially in high-stakes applications, finite test samples coverage guarantees 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.

In the last two decades, the literature has witnessed the evolution of impressive UQ approaches, the majority of which are built upon Neural Network (NN) architectures and can primarily be divided into two main categories. A popular class of UQ methods, developed for regression task, 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(CDF). Some important of them are Bayesian method MacKay (1992); Bishop (1995), Delta method De VlEAUX et al. (1998); Hwang & Ding (1997); Seber & Seber (2015) and Mean Variation Estimation (MVE) method Nix & Weigend (1994). Some of the recent NN architectures for the probabilistic forecasting task with distribution assumptions are Mixture Density Network Bishop (1994); Zhang et al. (2020a), Deep Auto-regressive Network (Deep AR) Salinas et al. (2020).

The other class of UQ approaches for regression task believe in estimating the pair of quantile functions $(f_q(x), f_{1+q-\alpha}(x))$ in a distribution-free setting without imposing any assumptions regarding the distribution of $Y|X$. For estimation of the qth quantile function, $f_q(x)$, most of them minimize the pinball loss function (Koenker & Bassett Jr (1978)). 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. The most prominent among these are the Lower-Upper-Bound Estimation (LUBE) NN , the Quality-Driven (QD) loss NN Pearce et al. (2018) and the Tube-loss NN Anand et al. (2024).

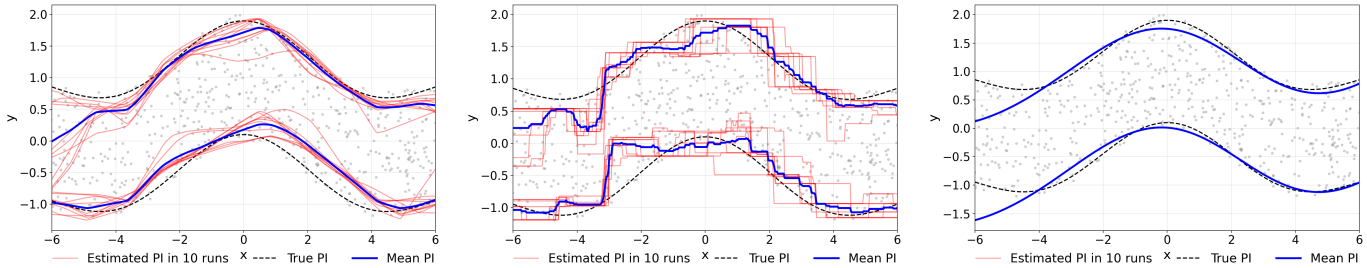
Beyond NN-based models, there is a class of literature that uses Random Forest (RF) for UQ tasks. Meinshausen analyzes the suitability of the RF for the quantile regression in his work Meinshausen (2006), while Lu Lu (2021) later established their advantages for PI estimation through a unified error-estimation framework. Most of the recent conformal regression literature also prefers to use the Quantile Random Forest (QRF) for obtaining the conformal prediction set. Some important of them are Romano et al. (2019),Vasiloudis et al. (2019),Seedat et al. (2023) and Luo & Zhou (2025).

In regression-based UQ, estimating the bounding quantile functions of the PI, $(f_q(x), f_{1+q-\alpha}(x))$ is a core task and is implemented by training either NN or RF-based models in recent literature. However, the quantile estimates obtained by them 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.

Further, the model uncertainty may be observed to arise from two major sources in UQ task in regression setting. First, variations in the learned model parameters across different training sets, and second, variations in the learned parameters across multiple training trials on a fixed training set. For a given finite training sample T', we term the latter source of uncertainty as *parameter uncertainty*, that has been extensively studied and used in recent UQ literature as *model uncertainty*. The ensemble-based approach is natural method for capturing the parameter uncertainty. Lakshminarayanan et al. (2017) have demonstrated that it is the most effective strategy for capturing model uncertainty in NN and deep learning models. The central idea is to train the same model multiple times over given training set T' using the same hyper-parameter configuration but different random initializations for quantifying the model uncertainty through

the variability observed in the resulting learned parameters or predictions. Furthermore, Pearce et al. (2018) aggregate model uncertainty captured via ensemble NN with the estimated PIs to quantify the overall uncertainty.

While performing the UQ estimation using the NN, deep learning or RF models, accurately capturing model uncertainty becomes an important task in addition to the data uncertainty. It is because that all of them exhibit significant parameter uncertainty. The objective functions of NN and deep learning models are typically non-convex. This causes the learned solution to oscillate among different local minima across training trials, even under the same hyper-parameter setting and a fixed training set T', thereby inducing significant parameter uncertainty, particularly when the training set is small. Similar to the NN, RF models also show significant variation in their estimate across training trails as they rely upon the bootstrap sampling and random feature selection at each split.



(a) Quantile NN PI (σ_{model} =0.26) (b) Quantile RF PI (σ_{model} =0.33) (c) Quantile SVM PI (σ_{model} =0)

Figure 1: The UQ task was to estimate the PI with $1-\alpha = 0.90$ with fixed training set. Unlike SVM, RF and NN based PI model induces the significant variations in estimated PI across 10 independent training runs. This variation can be quantified using the $\sigma^2_{model} = \sigma^2_{0.05} + \sigma^2_{0.95}$, where $\sigma^2_q = \sum_{i=1}^{k}\sum_{j=1}^{10}(f_q^{(j)}(x_i) - \bar{f}_q(x_i))^2$, $f_q^{(j)}(x_i)$ and $\bar{f}_q(x_i)$ denoting the qth quantile estimate for i^{th} test point at j^{th} run and their mean over all runs respectively upon k samples. More details are at Section 4.1.

However, in contrast to NN, deep learning and RF models, SVMs do not suffer from such parameter uncertainty, as they obtain the global optimal solution that remains invariant across different training trails and independent of initialization under a fixed hyper-parameter setting. It makes the SVMs less uncertain, more stable and trustworthy than their NN and RF counterparts. Beyond global optimality, SVMs also provide simple, interpretable, and often sparse solutions, together with an explicit and principled mechanism for regularization control. Despite these notable advantages, SVMs have remained surprisingly under-explored from the perspective of UQ. In this paper, we re-examine SVMs through the lens of contemporary UQ methodologies and highlights their intrinsic strengths in obtaining stable, more certain and reliable UQ estimate along with the additional benefits of obtaining the sparse and more interpretable solution. We identify the following distinctive and compelling properties of SVM models that strongly support their promotion for UQ regression tasks, especially under small-sample settings.

(a) In PI estimation task, SVMs fundamentally differ from RF and NN models by exhibiting zero parameter uncertainty, due to their convergence to a unique global optimal solution. This property enables SVMs to produce more stable and less uncertain PIs than their RF and NN counterparts, especially in small-sample learning scenarios (See Figure 1). We also show that the NN and RF models suffer from pronounced parameter uncertainty as the sample size decreases, which contributes substantially to the overall predictive uncertainty.

(b) SVM models can produce sparse solutions for the lower and upper quantile functions, required to construct the PI. The sparsity in the SVM solution, not only simplifies the learned quantile functions, enhancing interpretability, but more importantly, it enables the effective feature selection in PI estimation task. We highlight the significance of the feature selection in PI estimation and

show that the use of SVM models can successfully eliminate a significant percentage of irrelevant features, while simultaneously improving PI quality and complexity, particularly in high-dimensional learning settings.

(c) The adaption of the SVM models in the Conformal Regression (CR) problem can lead to more stable and certain prediction sets. Surprisingly, the potential of SVM models for deriving stable CR estimates remains largely unexplored in the literature.

(d) SVM models may obtain the high-quality probabilistic forecast comparable to the recent deep probabilistic forecasting models which often exhibit significant model uncertainty with larger numbers of parameters, particularly for small and moderate scale time-series datasets.

The remainder of this paper is structured as follows. Section 2 describes how SVMs can be used to estimate stable and less certain UQ estimate. Section 3 introduces sparse SVM models for PI estimation and demonstrates their use for feature selection in PI estimation tasks. Also, it extends the SVM for obtaining the stable conformal prediction sets. Section 4 empirically validates our claims regarding the strengths of SVM-based models for UQ tasks. Section 5 provides a detailed discussion from a broader perspective, analyzing the advantages and limitations of SVM-based UQ models in comparison with their NN and RF counterparts. Section 6 concludes the paper and outlines future directions for the adoption of SVM-based UQ methods across diverse application domains. Appendix A presents a detailed comparative analysis of existing SVM-based UQ methods, along with their strengths and limitations to provide a clearer roadmap for UQ researchers interested in SVM-based methods.

2 UQ in SVM regression

In this section, we outline SVM methods relevant to UQ estimation techniques in regression tasks.

2.1 Quantile Regression and Pinball loss

In distribution-free setting, for a given quantile $q \in (0, 1)$, the quantile value is estimated by minimizing the pinball loss function 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 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 $\min_{f \in F} \sum_{i=1}^{m} \rho_q(y - f(x_i))$. Takeuchi 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

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

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 mean of all b_k.

2.3 PI estimation in SVM

Given the target 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 this pair of quantile functions for some $0 \leq q \leq \alpha$. We detail the algorithm for PI estimation through SVQR in Algorithm 1.

Unlike the NN and RF generated PI, the estimated PI $[f_{\bar{q}}, f_{1+\bar{q}-\alpha}]$ using (1) remains unique and stable for a fixed choice of C and T'. This is due to the nature of the QPP (4), which always attains a unique global optimal solution.

Algorithm 1 PI estimation through SVQR

procedure PI THROUGH SVQR($T', 1-\alpha$)
  Choose $\bar{q} \in (0, 1-\alpha)$
  for each $q \in \{\bar{q}, 1+\bar{q}-\alpha\}$ **do**
    Solve QPP (4) for an appropriate value of C.
    Obtain the solution (α^*, β^*) and estimate $f_q(x)$.
  return $(f_{\bar{q}}, f_{1+\bar{q}-\alpha})$

3 Sparse PI estimation in SVM

The sparse solution is a yet another promising feature offered by the SVM model. A sparse solution vector in SVM offers significant advantages. It reduces the complexity of the model without compromising its predictive ability and hence expected to own better generalization ability on unseen test points. More importantly, it can help to eliminate the irrelevant features, making the learning task more simple and effective.

3.1 Sparse Support Support Vector Quantile Regression

The Sparse Support Support Vector Quantile Regression (SSVQR) can obtain the sparse solution for the quantile estimate. The SSVQR model minimizes the pinball loss function with L_1- norm regularization. It seeks the solution of the following problem

$$\min_{(w,b)} \frac{1}{2}||w||_1 + C\sum_{i=1}^{m} \rho_q(y_i - (w^T\phi(x_i) + b)), \tag{7}$$

where $C \geq 0$ is the user defined parameter for trading of the regularization against the empirical loss. The solution to problem (7) 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 representer theorem Schölkopf et al. (2001) 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 (7) can be written as

$$\min_{(u,b)} \frac{1}{2}||u||_1 + C\sum_{i=1}^{m}(q\xi_i + (1-q)\xi_i^*),$$

subject to,

$$\xi_i = \max\Big(y_i - \Big(\sum_{j=1}^{m} k(x_j, x_i)u_j + b\Big), 0\Big), i = 1, 2, ...m,$$

$$\xi_i^* = \max\Big(\Big(\sum_{j=1}^{m} k(x_j, x_i)u_j + b\Big) - y_i, 0\Big), i = 1, 2, ...m,$$

that can be simplified as

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

which can be written as

$$\begin{aligned}
&\min_{(u,b,\xi,\xi^*)} \frac{1}{2}||u||_1 + C\sum_{i=1}^{m}(q\xi_i + (1-q)\xi_i^*) \\
&\text{subject to,} \\
&\quad y_i - \Big(\sum_{j=1}^{m} k(x_j, x_i)u_j + b\Big) \leq \xi_i, \\
&\quad \Big(\sum_{j=1}^{m} k(x_j, x_i)u_j + b\Big) - y_i \leq \xi_i^*, \\
&\quad \xi_i, \xi_i^* \geq 0, \ i = \ 1, 2, ..m.
\end{aligned} \tag{8}$$

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 (8) can be expressed as

$$\begin{aligned}
&\min_{(r,p,b,\xi,\xi^*)} \frac{1}{2}\sum_{i=1}^{m}(r_i + p_i) + C\sum_{i=1}^{m}(q\xi_i + (1-q)\xi_i^*) \\
&\text{subject to,} \\
&\quad y_i - \Big(\sum_{j=1}^{m} k(x_j, x_i)(r_j - p_j) + b\Big) \leq \xi_i, \\
&\quad \Big(\sum_{j=1}^{m} k(x_j, x_i)(r_j - p_j) + b\Big) - y_i \leq \xi_i^*, \\
&\quad \xi_i, \xi_i^*, r_i, p_i \geq 0, \ i = \ 1, 2, ..m.
\end{aligned} \tag{9}$$

The above problem (9) 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 (9) determines the estimate of the qth quantile function using

$$f_q(x) = \sum_{i=1}^{m}(r_i^* - p_i^*)k(x_i, x) + b. \tag{10}$$

The asymptotical properties of the SSVQR model remains immune in view of the Lemma 3 of (Takeuchi et al. (2006)) which holds for a wide range of regularization schemes used in pinball-loss minimization within the SVM framework.

3.2 PI estimation through SSVQR

The PI estimation through SSVQR is straightforward on the line of SVQR. The quantile bound functions $f_{\bar{q}}(x)$ and $f_{1+\bar{q}-\alpha}(x)$ should be estimated by solving a pair of LPP problems (9) with $q = \bar{q}$ and $q = 1+\bar{q}-\alpha$ respectively. The solution of LPP problem (9) is sparse that makes the estimated quantile functions $f_{\bar{q}}(x)$ and $f_{1+\bar{q}-\alpha}(x)$ more simple that may enhance their interpretablity and generalization ability .

3.3 Feature selection in PI estimation through Sparse SVM

Now, we address the significance of the feature selection in UQ regression task that has also been unexplored in the UQ literature. Similar to other machine learning tasks, the PI estimation in high-dimensional settings also presents several challenges. The increased dimensionality not only increases 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.

Algorithm 2 Feature selection through SSVQR

procedure :- FEATURE SELECTION THROUGH SSVQR$(T, 1-\alpha, tol)$
  Choose some $\bar{q} \in (0, 1-\alpha)$
  for each $q \in \{\bar{q}, (1+\bar{q}-\alpha)\}$ **do**
    Consider linear kernel $k(x_i, x_j) = x_i^T x_j$ at LPP (9).
    Solve the LPP (9) 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 $I_{1+\bar{q}-\alpha} = \{i :, |w_{1+\bar{q}-\alpha}(i)| \leq tol\}$
  Compute $I = I_{\bar{q}} \cap I_{1+\bar{q}-\alpha}$ and *Feature Set* $= \{1, 2, ..., m\} - I$
  return (*Feature Set*)

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. $f_{\bar{q}}(x) = w_{\bar{q}}^T x + b_{\bar{q}}$ $f_{1+\bar{q}-\alpha}(x) = w_{1+\bar{q}-\alpha}^T x + b_{1+\bar{q}-\alpha}$.

3.4 Conformal Regression in SVM

Further, we extend the SVM based PI model in CR setting for obtaining the finite sample test set guarantees. In split CR setting, we detail the SVM based CR algorithm using the non-conformity score of Romano et al. (2019) as follows.

Algorithm 3 CR through SVQR

procedure CR THROUGH SVQR$(T,\ 1-\alpha)$
  Split T' into training I_1 and calibration I_2
  Choose $\bar{q} \in (0, 1-\alpha)$
  for $q \in \{\bar{q},\ 1+\bar{q}-\alpha\}$ **do**
    Train SVQR by solving QPP (4) on I_1 to obtain f_q
  $E_i \leftarrow \max\{f_{\bar{q}}(x_i) - y_i,\ y_i - f_{1+\bar{q}-\alpha}(x_i)\},\ x_i \in I_2$
  $Q \leftarrow (1-\alpha)(1+|I_2|^{-1})$-quantile of $\{E_i\}$
  return $[\, f_{\bar{q}}(x) - Q,\ f_{1+\bar{q}-\alpha}(x) + Q \,]$

The prediction set produced by Algorithm 3 remains stable across training trials without compromising its quality, whereas RF and NN models produce CR prediction sets that exhibit significant variability across different training runs even under the same hyper-parameter setting.

4 Experimental Results

In this section, we present the numerical results to verify our claims regarding the usefulness of the SVM in UQ regression tasks.

Method	Sum of RMSEs	σ_{model}
NN Quantile	0.353	0.132
QRF Quantile	0.264	0.123
SVM Quantile	**0.266**	**0.000**

Table 1: Comparison of data and model uncertainty across PI methods. Sum of RMSEs quantifies the cumulative deviation of the mean of estimated quantile functions from the quantile bounds of the actual PI.

4.1 SVMs are less uncertain for PI estimation than RFs and NNs

We simply generate samples using the $y_i = \frac{sin(x_i)}{x_i} + \epsilon_i$, where $\epsilon_i \sim \text{Uniform}(-1, 1)$. The task was to estimate the PI with target coverage 0.90. Due to presence of a symmetric noise distribution, we estimate the 0.05^{th} quantile function $f_{0.05}(x)$ and 0.95^{th} quantile function $f_{0.95}(x)$ for least possible PI width. The optimal $f_{0.05}(x)$ and $f_{0.95}(x)$ quantile functions were obtained from inverting the Uniform Cumulative Distribution Function and have been labeled with true PI as black-dotted line in Figure 1. We have considered 100 training points and 600 test points.

Figure 1 shows the PI estimated by the Quantile NN, Quantile RF and Quantile SVM models in 10 independent runs on test set with optimal tunned parameters. Unlike SVM, RF and NN based PI models induces the significant variations in estimated PI across 10 independent training runs. The model uncertainty σ^2_{model} is quantified as $\sigma^2_{0.05} + \sigma^2_{0.95}$, where $\sigma^2_q = \sum_{i=1}^{k} \sum_{j=1}^{10} (f_q^{(j)}(x_i) - \bar{f}_q(x_i))^2$, $f_q^{(j)}(x_i)$ and $\bar{f}_q(x_i)$ denoting the qth quantile estimate for i^{th} test point at j^{th} run and their mean over all runs respectively. The data uncertainty should be quantified by computing the quality of the PI, constructed by the mean of upper and lower quantile functions, estimated across 10 different runs respectively. For this, we compute the deviation of the mean estimate for q^{th} quantile function, $\bar{f}_q(x_i)$, from actual target quantile functions for $q = 0.05$ and 0.95 (black-dotted line in Figure 1) in terms of RMSEs and collect them as Sum of RMSEs to quantify the quality of the estimated PI $[\bar{f}_{0.05}(x_i), \bar{f}_{0.975}(x_i)]$. Apart from this, we have also reported the PICP and MPIW to assess the quality of estimated PI. Table 1 shows that SVM based PI model obtains better estimate for data and model uncertainty. A similar experiment with a training set of size 50 is reported in Appendix B.1 showing that the advantages of the SVM grow significant as the sample size decreases.

4.2 Feature selection is important in PI estimation task

Next, we apply our Sparse SVM model for feature selection in PI estimation with target coverage $1-\alpha = 0.95$ and linear kernel following the Algorithm 2.

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.

Datasets:- we use five popular real-world benchmark datasets namely Spambase (4601×58) (Hopkins & Suermondt (1999)), Student Performance (395×16) (Cortez (2008)) , Boston Housing (505×14) (Harrison Jr & Rubinfeld (1978)), UCI-secom (1568×591) (McCann & Johnston (2008)) and MADELON (2000×500) (Guyon (2004)) for the feature selection task using our proposed algorithm.

Following Algorithm 2 , we perform feature selection using the SSVQR PI model for $\bar{q} = 0.025$ and present the results in Table 2. Since Algorithm 2 solves the problem (9) with linear kernel twice, we need to tune only the parameter C for each $\bar{q}$ which was selected from the grid of $\{2^i : i = -15, ..., 15\}$.

On average, Algorithm 2 could reduce the 69% of features on considered datasets and reduce the complexity of the PI estimation task significantly, while maintaining the quality of the PI, as measured by PICP and MPIW. The training time of the PI estimating task improves around 53% after dropping irrelevant features through Algorithm 3. Also, it leads to the reduction of 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

Dataset	Dim.	Before FS			After FS			%Red.
		PICP	MPIW	Time	PICP	MPIW	Time	Feature.
Spambase	(4601,58)	0.966	0.933	60	0.965	0.934	52	46%
Student Perf.	(395,16)	0.886	6.843	0.48	0.886	6.843	0.42	73%
Boston Housing	(505,14)	0.941	23.06	0.81	0.941	23.06	0.70	38%
UCI-SECOM	(1568,591)	0.876	1.615	16	0.920	1.668	7	91%
MADELON	(2000,500)	0.710	2.001	58	0.938	2.002	11	99%

Table 2: Performance before and after feature selection using Algorithm 2. Time is overall runtime in seconds. FS: Feature Selection

in the quality of the estimated PI in terms of its coverage. Table 3 lists the dropped features by Algorithm 2 in case of different datasets.

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 3: Features dropped by the Algorithm 2. On average, it drops the 69% of features on considered datasets and reduce the complexity of the PI estimation task significantly.

4.3 SVM can obtain more stable conformal prediction sets

To show the stability of the SVM based CR model detailed in Algorithm 2 against the NN-based CQR model (CQR-NN) Romano et al. (2019) and RF based CR model (QRF+CP) Romano et al. (2019), we consider several benchmark datasets with target coverage level $1 - \alpha = 0.90$. Each dataset was partitioned into a training set (60%), a calibration set (20%), and a test set (20%) and all of considered models were trained across 10 independent runs using fixed hyperparameter settings and identical splits of training and calibration sets. All models were first trained to estimate the 0.05^{th} and 0.95^{th} quantile functions, after which nonconformity scores used in (Romano et al. (2019)) were computed for conformal prediction.

We have tuned the hyperparameters of each machine learning model to obtain a high-quality conformal set. For the SVM based CR model (SVQR+CR) model, we used the RBF kernel with parameter $\gamma = 1.0/n_{\text{features}}$, a regularization parameter $C = 10.0$, for targeting the quantile functions. For the NN based CR (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.

The results in Table 4 show that the standard deviations of PICP and MPIW across 10 runs are zero for the SVM+CP model, indicating perfectly stable prediction sets. In contrast, CQR-NN and Quantile Random Forest based CR model (QRF+CP) exhibit noticeable variability across runs, reflecting substantial model uncertainty. Moreover, the data uncertainty captured by SVM+CP, as measured by mean PICP and MPIW, is comparable to or better than that of the NN- and RF-based conformal models.

4.4 SVM potential for probabilistic forecasting

We compare the quality of the probabilistic forecasting obtained by SVM models against the recently developed complex deep probabilistic forecasting architectures upon popular time-series datasets namely Female

Data	Meth.	PICP	MPIW	Time
Boston	CQR-NN	94.12±1.14	2.19±0.12	2.65±0.23
	SVQR+CP	95.10±**0.00**	2.24± **0.00**	0.44±0.03
	QRF+CP	93.14±1.57	12.86±0.61	3.07±0.63
Energy	CQR-NN	87.66±1.12	1.13±0.03	3.20±0.31
	SVQR+CP	88.96±**0.00**	1.05± **0.00**	0.95±0.05
	QRF+CP	92.21±2.05	0.91±0.02	2.46±0.35
Concrete	CQR-NN	92.58±1.02	19.62±0.71	1.89±0.17
	SVQR+CP	91.35±**0.00**	18.74± **0.00**	0.38±0.01
	QRF+CP	93.78±1.66	25.26±0.64	2.95±0.32
Yacht	CQR-NN	91.69±1.49	{2.43±0.35	1.44±0.12
	SVQR+CP	90.82±**0.00**	2.87± **0.00**	0.22±0.01
	QRF+CP	92.58±5.16	4.74±1.43	1.75±0.09
Servo	CQR-NN	88.39±2.21	0.73±0.10	1.15±0.09
	SVQR+CP	89.74± **0.00**	0.68±**0.00**	0.16±0.01
	QRF+CP	95.30±2.35	2.19±0.35	1.72±0.26

Table 4: Comparison of CQR-NN, SVQR+CP, and QRF+CP across benchmark datasets. Values are reported as mean ± std over 10 runs. The SVM-based CR model (SVQR+CP) produces stable prediction sets with zero standard deviation, whereas the NN-based CR model (CQR- NN) and Quantile Random Forest based CR model (QRF+CP) exhibit variability across training runs. Time (sec.):- Training time in seconds.

Births (365 × 1) (datamarket.com), Minimum Temperature (3651 × 1) (machinelearningmastery.com) and Beer Production (464 × 1) (Australian (1996)). 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 Quantile based deep learning architectures for probabilistic forecasting developed in distribution-free setting, including Quantile-based LSTM, Quantile-based GRU, Quantile-based TCN and Quantile-based Transformers. The SVQR and SSVQR models are fundamentally quantile-based probabilistic forecasting approaches; therefore, when comparing them with deep learning counterparts, we should only consider quantile-based deep learning methods. However, for the sake of extensive comparisons, we also report the performance of deep probabilistic forecasting models that adopt different probabilistic forecasting approaches, including the Tube Loss LSTM (Anand et al. (2024)) and Quality-Driven (QD) Loss LSTM models (Pearce et al. (2018)), as well as the DeepAR model (Salinas et al. (2020)). The QD loss (Pearce et al. (2018)) is the improved version of the LUBE model (Khosravi et al. (2010)) can be used minimized with the gradient descent method in deep learning architecture.

All models were trained to obtain probabilistic forecasts at a target calibration level of $1-\alpha = 0.95$ and were repeated five times under identical hyperparameter settings after careful parameter tuning. Quantile-based models estimate the 0.025 and 0.975 quantiles of the predictive distribution.

SVM methods require 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, Quantile-based GRU, Quantile-based TCN, Quantile-based Transformers, Tube Loss LSTM, Deep AR and 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. A detailed description of tuned hyper-parameters, architectures of the tuned deep neural networks, and probabilistic forecasting results is provided in Appendix A.3.

Data (Size)	Method	PICP	MPIW	σ^2	#W	Sparsity (%)
FB	SSVQR	0.93	28.00	0	256	61
	SVQR	**0.95**	**27.11**	**0**	256	0
	Quantile-LSTM	0.95	28.20	1.52	30K	0
	Quantile-GRU	0.95	46.52	0.39	65K	0
	Quantile-TCN	0.96	57.00	5.54	120K	0
	Quantile-Transformer	0.95	59.00	17.34	80K	0
	Tube-LSTM	0.96	28.09	0.34	30K	0
	QD-LSTM	0.94	38.98	2.50	30K	0
	DeepAR	0.94	29.80	0.14	13K	0
MnTemp	SSVQR	**0.96**	**10.72**	**0**	2556	69
	SVQR	0.96	75.81	0	2556	0
	Quantile-LSTM	0.95	24.82	0.14	32K	0
	Quantile-GRU	0.94	16.53	0.19	65K	0
	Quantile-TCN	0.95	16.84	0.18	120K	0
	Quantile-Transformer	0.94	17.13	0.25	80K	0
	Tube-LSTM	0.94	15.56	0.28	32K	0
	QD-LSTM	0.79	5.94	0.14	32K	0
	DeepAR	0.90	12.79	0.11	13K	0
BP	SSVQR	0.96	78.21	0	325	70
	SVQR	**0.95**	**75.73**	**0**	325	0
	Quantile-LSTM	0.94	134.80	6.43	29K	0
	Quantile-GRU	0.94	139.88	89.54	65K	0
	Quantile-TCN	0.95	110.32	14.98	120K	0
	Quantile-Transformer	0.94	124.90	18.03	80K	0
	Tube-LSTM	0.95	42.91	10.61	29K	0
	QD-LSTM	0.96	159.71	60.02	29K	0
	DeepAR	0.76	12.43	154.65	13K	0

Table 5: Comparison of probabilistic forecasting methods. PICP and MPIW quantify data uncertainty, while σ^2 reflects model uncertainty across training runs. Sparsity denotes the percentage of zero coefficients in solution weight. The best model is the one that achieves the lowest MPIW while satisfying the target coverage level of 0.95. FB: Female Births (365×1), MnTemp: Minimum Temperature (3651×1), BP: Beer Production (464×1). #W denotes the number of trainable weights. Tube-LSTM Anand et al. (2024), QD-LSTM Pearce et al. (2018) and DeepAR Salinas et al. (2020) SVQR: Support Vector Quantile Regression, SSVQR: Sparse Support Vector Quantile Regression

Table 5 presents the numerical results. From a data uncertainty perspective, the SVM-based models achieve PI quality, measured by PICP and MPIW, that is comparable to or better than that of deep learning architectures, despite using significantly fewer trainable weights. Additionally , deep learning models exhibit noticeable model uncertainty, as indicated by non-zero values of σ^2. Furthermore, SVM-based UQ models are relatively simpler and require the adjustment of significantly fewer parameters compared to complex deep learning models. Overall, these results suggest that the SVM-based approach provides a stable, effective and promising solution for probabilistic forecasting when both data and model uncertainty are considered.

5 Broader Perspectives

SVM models seem to be a better choice for UQ in regression tasks, including PI estimation, conformal regression, and probabilistic forecasting, relative to NN, RF, and deep learning models. It is because of the inherent nature of their solution that remains stable and certain across different runs. On the other hand, NN and deep learning models seem to exhibit significant model uncertainty and appear inferior to the

SVM models despite their better expressive ability to model complex relationships. At the same time, SVM models are known to produce more interpretable, optimal, and sparse solutions.

However, these adornments of the SVM model produce the rainbow effect that vanishes when the day grows more sunny. The advantages of the SVM model over NN and deep learning models diminish with the increase in the size of the dataset. This is because the model uncertainty produced by NN and deep learning models reduces almost exponentially with the growth of the training sample size (see Figure 2). For large-scale datasets, NN and deep learning models show very little model uncertainty that remains manageable in view of their capability to model complex relationships and extract rich temporal features in the case of forecasting. Therefore, a significant advantage of SVM-based models for UQ tasks remains evident, particularly when working with small- to moderate-scale datasets.

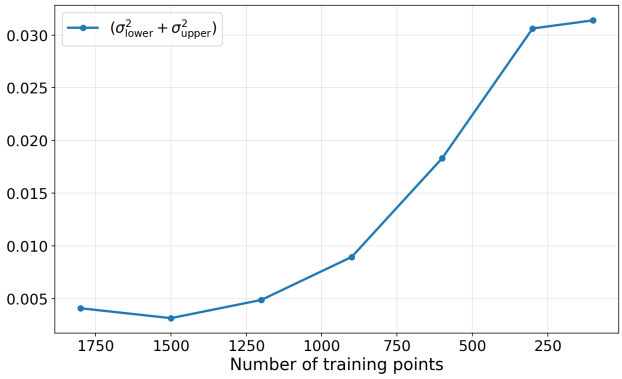


Figure 2: Plot of model uncertainty σ^2 of NN model computed in 10 independent runs upon the toy dataset simulated at Section 4.1.

Another perspective can be examined through the lens of model uncertainty. The idea of model uncertainty varies substantially across application domains and machine learning tasks. Compared to Bayesian and MC-drop out Gal & Ghahramani (2016) approaches, ensemble techniques provide a simpler and more effective mechanism for quantifying model uncertainty in practice.

Within ensemble methods, model uncertainty is typically quantified by assessing the variability in predictions across multiple training runs, which can arise from two sources: (i) repeated training on the same dataset with different random initializations under identical hyperparameter settings, and (ii) training on different bootstrap resamples of the training data. The ensemble approach of Lakshminarayanan et al. (2017) adopts the former strategy to quantify the uncertainty of the model and Pearce et al. (2018) further extends this idea to estimate the model uncertainty for the PI estimation problem. We have also followed the same protocol in our experiments. But, even when additional variability is introduced by training on multiple bootstrap datasets, the overall uncertainty exhibited by complex NN and deep learning models must remain higher than that of simple SVM models, solving the convex optimization problems, particularly on small- and moderate-scale datasets.

6 Implications and Future Directions

We encourage the adoption of SVM-based UQ methods across a wide range of application domains, particularly in small- and moderate-scale data regimes where stability and reliability are critical. SVMs provide stable, optimal, and reproducible solutions, making their UQ estimates especially valuable in medical, biomedical, and clinical applications, where datasets are often limited, feature dimensionality is high, and trustworthy uncertainty estimates are more important than marginal gains in point prediction accuracy.

In UQ regression tasks, particularly for high-dimensional tabular data, careful attention to feature selection is essential, as it not only reduces model complexity and improves interpretability but also enhances the quality of PIs by mitigating spurious sources of uncertainty.

While deep learning models can be effective for probabilistic forecasting on large-scale datasets, their deployment in data-limited settings often leads to over-parameterization, heightened model uncertainty, and unstable solutions. In such cases, we recommend evaluating the solution quality of SVM-based models before training large and complex deep architectures.

To provide a clearer roadmap for UQ researchers interested in SVM-based methods, Appendix A presents a detailed comparative analysis of existing SVM-based UQ methods, along with their strengths and limitations.

References

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.

Beer Production Australian. Beer production australian. `https://www.kaggle.com/datasets/sergiomora823/monthly-beer-production`, 1996. Accessed: 10-01-2024.

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.

Paulo Cortez. Student Performance. UCI Machine Learning Repository, 2008. DOI: https://doi.org/10.24432/C5TG7T.

datamarket.com. Daily total female births in california, 1959. `https://www.kaggle.com/datasets/dougcresswell/daily-total-female-births-in-california-1959`. Accessed: 10-01-2024.

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.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning*, pp. 1050–1059, 2016.

Isabelle Guyon. Madelon. UCI Machine Learning Repository, 2004. DOI: https://doi.org/10.24432/C5602H.

David Harrison Jr and Daniel L Rubinfeld. Hedonic housing prices and the demand for clean air. *Journal of environmental economics and management*, 5(1):81–102, 1978.

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.

Reeber Erik Forman George Hopkins, Mark and Jaap Suermondt. Spambase. UCI Machine Learning Repository, 1999. DOI: https://doi.org/10.24432/C53G6X.

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.

Bo Lu. A unified framework for random forest prediction error estimation. *Journal of Machine Learning Research*, 22(8):1–41, 2021.

Rui Luo and Zhixin Zhou. Conformal thresholded intervals for efficient regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 19216–19223. Association for the Advancement of Artificial Intelligence, 2025. doi: 10.1609/aaai.v39i18.76338. URL `https://ojs.aaai.org/index.php/AAAI/article/view/34115`.

machinelearningmastery.com. Daily minimum temperatures in melbourne. `https://www.kaggle.com/datasets/paulbrabban/daily-minimum-temperatures-in-melbourne`. Accessed: 10-01-2024.

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.

Michael McCann and Adrian Johnston. SECOM. UCI Machine Learning Repository, 2008. DOI: https://doi.org/10.24432/C54305.

Nicolai Meinshausen. Quantile regression forests. *Journal of Machine Learning Research*, 7(Jun):983–999, 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.

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.

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.

Nabeel Seedat, Alan Jeffares, Fergus Imrie, and Mihaela van der Schaar. Improving adaptive conformal prediction using self-supervised learning. In Francisco J. R. Ruiz, Jennifer G. Dy, and Jan-Willem van de Meent (eds.), *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 206, pp. 10160–10177. PMLR, 2023. URL `http://proceedings.mlr.press/v206/seedat23a.html`.

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.

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.

Theodore Vasiloudis, Gianmarco De Francisci Morales, and Henrik Boström. Quantifying uncertainty in online regression forests. *Journal of Machine Learning Research*, 20(155):1–35, 2019. URL `http://jmlr.org/papers/v20/19-006.html`.

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.

I-C Yeh. Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete research*, 28(12):1797–1808, 1998.

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.

A Comparative analysis of PI estimation method in SVM

In this section, we present a comparative analysis of all prediction interval (PI) estimation methods proposed in the SVM literature.

A.1 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.

$$\begin{aligned} &\min_{(w,b,\xi)} \frac{1}{2} w^T w + C \sum_{i=1}^{m} (\xi_i^2) \\ &\text{subject to,} \\ &\quad y_i - (w^T \phi(x) + b) = \xi_i, \ i = \ 1, 2, ..m. \end{aligned} \tag{11}$$

The solution of problem (11) can be obtained through the following system of equations

$$\begin{bmatrix} 0 & e^T \\ e & K(A, A^T) + \frac{2}{C} I \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ Y \end{bmatrix}, \tag{12}$$

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 (12), the LS-SVR estimate the regression function for a given $x \in \mathbb{R}^n$ using

$$f(x) = \sum_{i=1}^{m} k(x_i, x)\alpha_i + b. \tag{13}$$

A.2 PI estimation through LS-SVR

One of the naive PI estimation methods in SVM literature follows the normal assumption regarding the distribution of $Y|X$ and estimates its mean through (13) by training the LS-SVR model. The error distribution $\epsilon_i = y_i - f(x_i)$ follows a normal distribution with zero mean and variance σ. This variance can be estimated from the error computed on the 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 was proposed in (De Brabanter et al. (2010); Cheng et al. (2014)).

A.3 PI estimation through Tube loss in SVM

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 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 passes 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

$$\rho_{1-\alpha}^{r}(u_2, u_1) = \begin{cases} (1-\alpha)u_2, & \text{if} \quad u_2 > 0, \\ -\alpha u_2, & \text{if } u_2 \leq 0, u_1 \geq 0 \text{ and } \ ru_2 + (1-r)u_1 \geq 0, \\ \alpha u_1, & \text{if } u_2 \leq 0, u_1 \geq 0 \text{ and } \ ru_2 + (1-r)u_1 < 0, \\ -(1-\alpha)u_1, & \text{if } \ u_1 < 0, \end{cases} \tag{14}$$

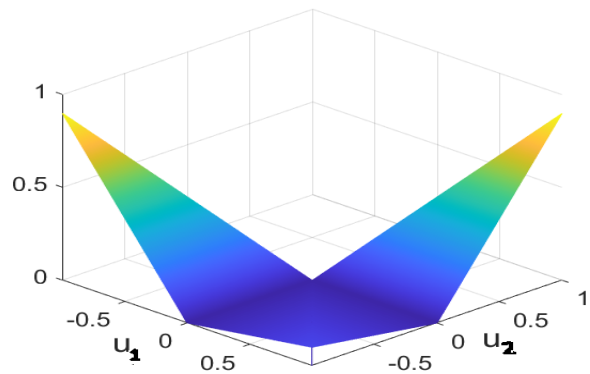


Figure 3: Tube loss function for $1-\alpha = 0.9$.

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

$$\mu_1(x) = \sum_{i=1}^{m} k(x_i, x)\alpha_i + b_1 \;\text{ and }\; \mu_2(x) = \sum_{i=1}^{m} k(x_i, x)\beta_i + b_2 \tag{15}$$

by minimizing the optimization problem

$$\begin{aligned}\min_{(\alpha,\beta,b_1,b_2)} J(\alpha,\beta,b_1,b_2) = \frac{\lambda}{2}(\alpha^T\alpha + \beta^T\beta) + \sum_{i=1}^{m} \rho^r_{1-\alpha}\big(y_i, \big(K(A^T, x_i)\alpha + b_1\big), \big(K(A^T, x_i)\beta + b_2\big)\ \big) \\ +\delta \sum_{i=1}^{m} \big|(K(A^T, x_i)(\alpha - \beta) + (b_1 - b_2)\big|,\end{aligned} \tag{16}$$

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 the gradient descent method can be found in (Anand et al. (2024)).

A.4 PI estimation through LUBE loss in SVM

The LUBE method (Khosravi et al. (2010)) was originally developed in the NN framework but, was extended to the SVM framework later in (Shrivastava et al. (2014; 2015)) for probabilistic forecasting of electric prices. For the given target confidence $(1-\alpha)$, and training set T', the LUBE SVM model seeks a pair of kernel generated functions of (15), $\mu_1(x)$ and $\mu_2(x)$, which are obtained by minimizing the following loss function

$$CWC = \frac{1}{R} MPIW\big(1 + \gamma\ (PICP)e^{-\eta(PICP-(1-\alpha))}\big). \tag{17}$$

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, PICP is the coverage of the estimated PI and computed using the $PICP = \frac{1}{m}\sum_{i=1}^{m} k_i$, where

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

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

Table 6: Comparisons of the different PI estimation models available in SVM framework

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.

The major problem with the LUBE cost function (17) 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 (17) 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. refined 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)).

A.5 Comparison of PI estimation SVM models

In Table 6, 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 estimates 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 asymptotically, 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 features 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 models maintain this guarantee by minimizing

the convex loss function in its optimization problems. The Tube loss problem (16) is non-convex and hence fails to guarantee the global optimal solution. Furthermore, the LUBE problem (17) 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 increase 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 are 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 lead 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.

A.6 Comparing the SVM based PI estimation methods through numerical experiments

We shall perform experiments to compare the SVM based (SVQR), Sparse SVM (SSVQR) PI and LS-SVR based PI estimation model on artificial and benchmark datasets.

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 $1-\alpha = 0.90$. To construct a PI with $1-\alpha$ confidence level, we seek the pair of quantile functions $(f_q(x), f_{1+q-\alpha}(x))$, 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.

However, to estimate the quality of the quantile function in the absence of information about the noise distribution, we can use the 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 qth 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 $\frac{1}{k}\#\{y_i : f_q(x_i) \leq y_i \leq$

$f_{1+q-\alpha}(x_i), i = 1, 2, ..k.\}$, where # denotes the cardinality of the set. Also, the MPIW can be computed by $\frac{1}{k}\sum_{i=1}^{k}|f_{1+q-\alpha}(x_i)\} - f_q(x_i)|$.

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.

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 6, PI estimation through SVQR and LS-SVR only obtains the global optimal solution but lacks 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 models 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 (9) 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 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 (9) 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.

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

	$(\bar{q},\ 1+\bar{q}-\alpha)$	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	0.6157
	(0.015, 0.965)	(1.7046, 1.4653)	(0%, 0%)	(0.0150, 0.9720)	0.957	0	2.8244	0.5844
	(0.02, 0.97)	(1.6519, 1.4799)	(0%, 0%)	(0.0180, 0.9720)	0.954	0	2.7855	0.6169
	(0.025, 0.975)	(1.5857, 1.5016)	(0%, 0%)	(0.0220, 0.9760)	**0.954**	0	**2.7379**	0.5546
	(0.03, 0.98)	**(1.5030, 1.5667)**	(0%, 0%)	(0.0250, 0.9800)	0.955	0	2.7212	**0.5150**
SSVQR	(0.01, 0.96)	(1.8060, 1.4418)	(15%, 18%)	(0.0110, 0.9680)	0.957	0	2.9056	0.4783
	(0.015, 0.965)	(1.7051, 1.4714)	(15%, 18%)	(0.0150, 0.9700)	0.955	0	2.8275	0.5485
	(0.02, 0.97)	(1.6497, 1.4861)	(15%, 18%)	0.0160, 0.9720	0.956	0	2.7939	**0.4712**
	(0.025, 0.975)	**(1.5517, 1.4978)**	(15%, 20%)	(0.0250, 0.9750)	**0.950**	0	**2.6995**	0.4928
	(0.03, 0.98)	(1.5133, 1.5604)	**(16%, 20%)**	(0.0290, 0.9800)	0.951	0	2.7288	0.6054

Table 7: Performance of the SVQR and SSVQR PI models on AD 1 dataset. Lw: Lower, Up: Upper, Spar: Sparsity, Time(s) : Training time in seconds. SSVQR PI models always obtain the sparse solution. Also, for $\bar{q} = 0.025$, SSVQR model obtains the best quality PI with MPIW = 2.70 and PICP = 0.95.

	$(\bar{q},\ 1+\bar{q}-\alpha)$	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	9.9780	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	8.3460	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 8: Performance of the SVQR and SSVQR PI models on AD 2 dataset. Lw: Lower, Up: Upper, Spar: Sparsity, Time(s) : Training time in seconds. SSVQR PI models always obtain the sparse solution. Also, for $\bar{q} = 0.02$, SSVQR model obtains the best quality PI with PCIP = 0.953 and MPIW = 8.78.

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 7-12. The leftmost column of these Tables list the different pairs of target quantiles $(\bar{q},\ 0.95+\bar{q})$, required for the PI estimation. 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 7-12 list these RMSE for different values of $\bar{q}$.

To effectively illustrate the comparative performance between the SVQR and SSVQR models across various artificial datasets, we normalize the MPIW values by dividing them by the variance of each dataset (AD1–

	$(\bar{q},\ 1+\bar{q}-\alpha)$	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	0.050	2.9000	0.4872
	(0.015, 0.965)	(1.7355, 1.6297)	(0%, 0%)	(0.0130, 0.9590)	**0.946**	0.004	**2.8870**	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)	0.940	0.010	2.7100	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)	0.940	0.010	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 9: Performance of the SVQR and SSVQR PI models on AD 3 dataset. Lw: Lower, Up: Upper, Spar: Sparsity, Time(s) : Training time in seconds. SSVQR PI models always obtain the sparse solution. SSVQR and SVQR model tends to obtain the similar performance.

	$(\bar{q},\ 1+\bar{q}-\alpha)$	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)	0.940	0.010	4.1802	0.4745
	(0.02, 0.97)	(2.4939, 2.2455)	(20%, 20%)	(0.0190, 0.9610)	**0.942**	0.008	**4.1830**	0.4818
	(0.025, 0.975)	**(2.3617, 2.3037)**	**(40%, 20%)**	(0.0240, 0.9640)	0.940	0.010	4.1002	**0.4662**
	(0.03, 0.98)	(2.2689, 2.4147)	(40%, 20%)	(0.0290, 0.9700)	0.941	0.009	4.1210	0.4884

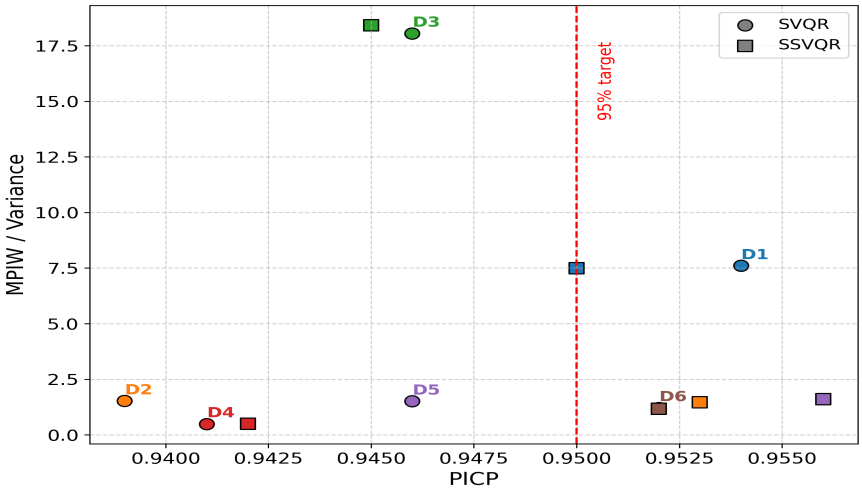
Table 10: Performance of the SVQR and SSVQR PI models on AD 4 dataset. Lw: Lower, Up: Upper, Spar: Sparsity, Time(s) : Training time in seconds. SSVQR PI models always obtain the sparse solution. SSVQR and SVQR model tends to obtain the similar performance.

	$(\bar{q},\ 1+\bar{q}-\alpha)$	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	7.9430	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**	0.00	**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 11: Performance of the SVQR and SSVQR PI models on AD 5 dataset. Lw: Lower, Up: Upper, Spar: Sparsity, Time(s) : Training time in seconds. SSVQR PI models always obtain the sparse solution. For $\bar{q} = 0.01$, SSVQR PI model obtains the best quality PI with PICP =0.956 and MPIW = 8.53.

	$(\bar{q},\ 1+\bar{q}-\alpha)$	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 12: Performance of the SVQR and SSVQR PI models on AD 6 dataset. Lw: Lower, Up: Upper, Spar: Sparsity, Time(s):Training time in seconds. SSVQR PI models always obtain the sparse solution. For $\bar{q} = 0.01$, SSVQR PI model obtains the best quality PI with PICP =0.956 and MPIW = 8.53.



(a)

Figure 4: Comparison of the quality of PIs estimated by the SVQR and SSVQR models. For a high-quality PI, the PICP should be greater than or equal to the target value of $(1 - \alpha) = 0.95$, while maintaining the smallest possible MPIW values. Among the six datasets (AD1–AD6), the SSVQR-based PI model successfully achieves the target PICP of 0.95 on four datasets, whereas the SVQR-based PI model fails to meet the target PICP on four datasets.

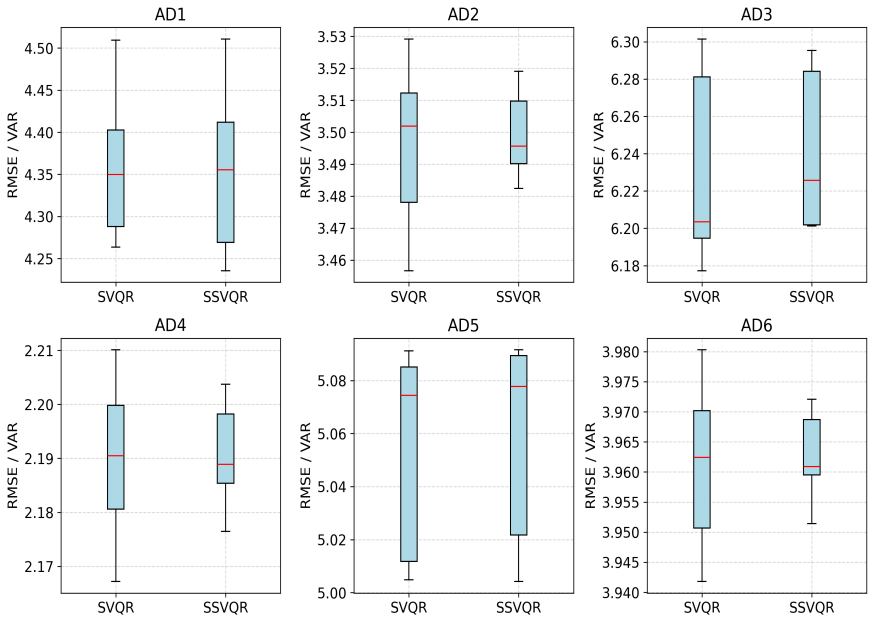
AD6). Additionally, we compute the aggregated RMSE as the mean of RMSE (Lp) and RMSE (Up), which represent the RMSE values corresponding to the lower and upper quantile bounds of the PI, respectively. Further, we also normalize them by dividing the variance present in each datasets for the comparative study.

Figure 4 shows the comparison of the quality of PIs estimated by the SVQR and SSVQR models. For a high-quality PI, the PICP should be greater than or equal to the target value of $(1 - \alpha) = 0.95$, while maintaining the smallest possible MPIW values. Among the six datasets (AD1-AD6), the SSVQR-based PI model successfully achieves the target PICP of 0.95 on four datasets, where as the SVQR-based PI model fails to meet the target PICP on four datasets. Figure 5 illustrates the comparisons of the normalized aggregated RMSE obtained by the SVQR and SSVQR models. 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 results in improved PI quality. However, the major advantage of using the SSVQR model over SVQR model is its ability to obtain the sparse solution vector. Figure 6(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.

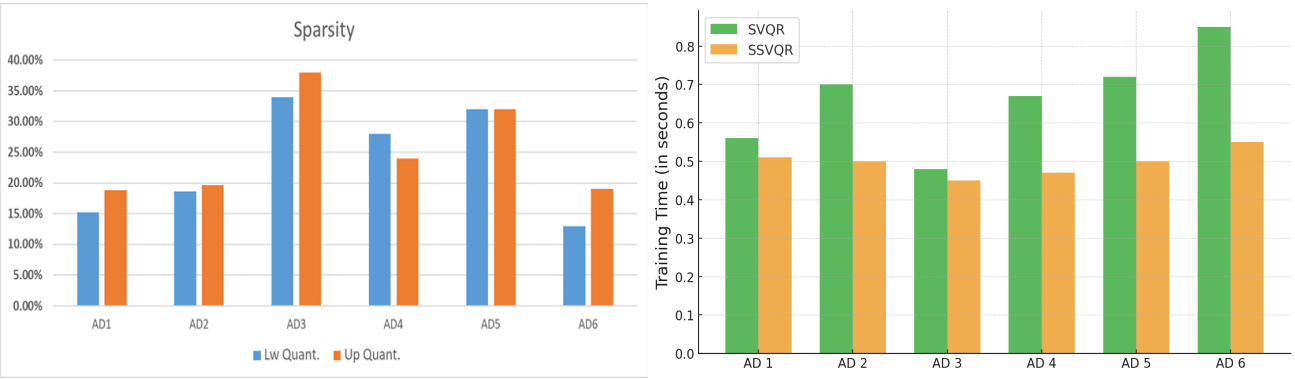
We compare the overall average training time (in seconds) taken by the SVQR PI and SSVQR PI models for the PI estimation task across different artificial datasets in Figure 6(b). It shows that the SSVQR requires fewer seconds train the PI model than the SVQR model. We have solved the QPPs of the SVQR 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 7-12 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 7. On AD2 dataset, the MPIW values of the PI obtained by both models increase with $\bar{q}$. It is evident from Table 8 that this increase is 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



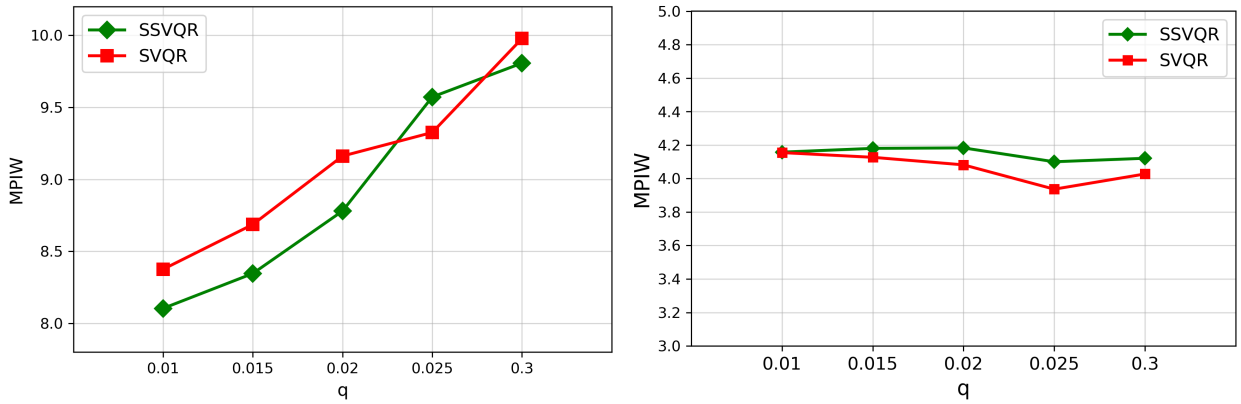
(a)

Figure 5: Comparison of the quality of the quantile functions estimated by the SVQR and SSVQR models is performed using the aggregated RMSE metric. The aggregated RMSE is computed as the mean of RMSE(Lw) and RMSE(up) which represent the RMSE values for the lower and upper quantile bounds of the PI, respectively. These RMSE values are then normalized by dividing by the variance of each dataset.



(a) (b)

Figure 6: (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.



(a) (b)

Figure 7: Plot of the MPIW values obtained by the SVQR and SSVQR PI models as a function of $\bar{q}$ for (a) the AD2 dataset and (b) the AD4 dataset. The AD2 dataset contains asymmetric noise drawn from a χ^2 distribution, resulting in a higher concentration of data points in the lower region of the input–target space. Consequently, the MPIW increases as the PI tube shifts upward with larger values of $\bar{q}$. In contrast, the AD4 dataset contains noise from a normal distribution; therefore, the centered PI obtained with the $\bar{q} = 0.025$ yields the narrower PI.

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 7(b) shows that at $\bar{q} = 0.025$, 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 (Harrison Jr & Rubinfeld (1978)) and Concrete (Yeh (1998)) 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 13 and 14 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 13: Comparison of different SVM PI estimation methods on Boston Housing. Lw: Lower, Up: Upper, Spar: Sparsity, Time(s) : Training time in seconds.

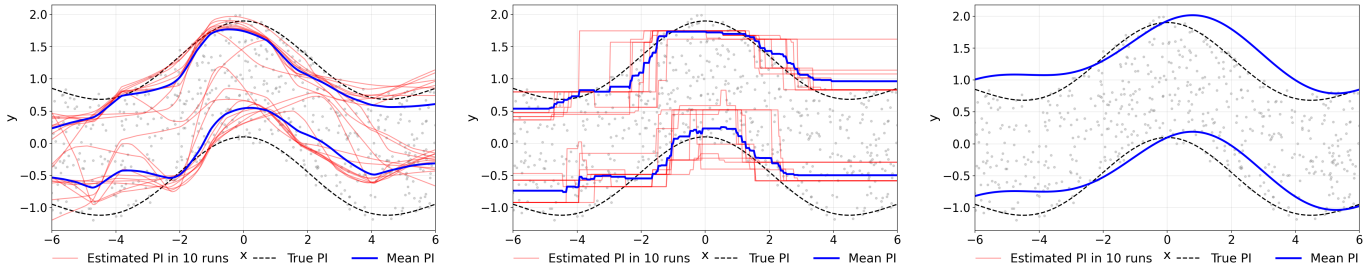
	q	Spar (Lw, Up)	CP (Lw, Up)	PICP	PICE	MPIW	Time (s)
SVQR	(0.025, 0.925)	(0%, 0%)	(0.0219, 0.8997)	0.8777	0.0723	32.5492	2.6141
	(0.05, 0.95)	(0%, 0%)	(0.0408, 0.9436)	**0.9028**	**0.0472**	**28.6541**	2.8405
	(0.075, 0.975)	(0%, 0%)	(0.0690, 0.9561)	0.8871	0.0629	32.1635	**2.5197**
SSVQR	(0.025, 0.925)	(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 14: Comparison of different SVM PI estimation methods on Concrete dataset.Lw: Lower, Up: Upper, Spar: Sparsity, Time(s) : Training time in seconds.

B Additional numerical results and experimental details

B.1 Model uncertainty across RF, NN and SVM

We repeat the experiment presented in Figure 1 using 50 training samples and 600 test samples under the same experimental settings, and report the results in Figure 8. Table 15 compares the performance of the SVM Quantile, NN Quantile, and RF Quantile methods. The SVM model is trained using an RBF kernel. As the training sample size decreases, the model uncertainty associated with NN and RF methods increases substantially, making the advantages of SVM-based approaches increasingly evident. The optimal neural architecture selected for the experiment was a Quantile Neural Network, trained with the pinball loss, using a single hidden layer with 64 neurons. For the random forest baseline, we employed a Quantile Random Forest (QRF) with 30 trees per ensemble.



(a) Quantile NN PI (σ_{model} =0.36) (b) Quantile RF PI (σ_{model} =0.34) (c) Quantile SVM PI (σ_{model} =0)

Figure 8: The UQ task was to estiamte the PI with $1-\alpha = 0.90$ with fixed training set of size 50 and 600 test points. Unlike SVM, RF and NN based PI model induces the significant variations in estimated PI across 10 independent training runs. This variation can quantified using the $\sigma^2_{model} = \sigma^2_{0.05} + \sigma^2_{0.95}$, where $\sigma^2_q = \sum_{i=1}^{k}\sum_{j=1}^{10}(f_q^{(j)}(x_i) - \bar{f}_q(x_i))^2$, $f_q^{(j)}(x_i)$ and $\bar{f}_q(x_i)$ denoting the qth quantile estimate for i^{th} test point at j^{th} run and their mean over all runs respectively. More details are at Section 4.1.

Method	**Sum of RMSEs**	PICP	(σ_{mode^2l})
NN Quantile	0.353	0.587	0.132
QRF Quantile	0.264	0.755	0.123
SVM Quantile	**0.266**	**0.848**	**0.000**

Table 15: Quantitative comparison of PI estimation methods. RMSE measures deviation from true interval bounds. PICP: Prediction Interval Coverage Probability. Model uncertainty σ^2_{model} quantifies run-to-run variability across 10 independent runs.

Dataset	Model	Architecture	Number of weights	Window 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
	Quantile GRU	GRU [128]	65 K	12
	Quantile TCN	TCN [32,64,64]	120 K	12
	Quantile Transformer	Transformer [64,4,2]	80 K	12
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
	Quantile GRU	GRU [128]	65 K	12
	Quantile TCN	TCN [32,64,64]	120 K	12
	Quantile Transformer	Transformer [64,4,2]	80 K	12
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
	QuantileGRU	GRU [128]	65 K	12
	Quantile TCN	TCN [32,64,64]	120 K	12
	Quantile Transformer	Transformer [64,4,2]	80 K	12

Table 16: 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. While the SVM-based model delivers competitive, stable performance with zero model uncertainty only with a few hundred parameters, deep learning models require optimizing thousands of parameters.

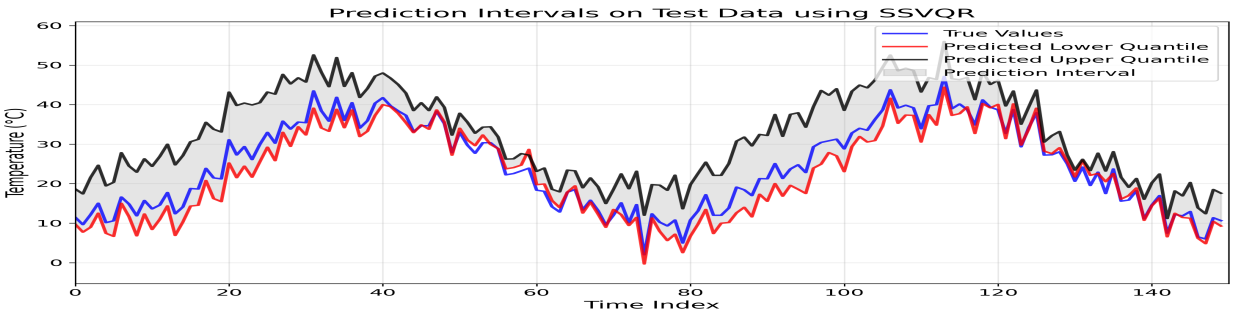
B.2 Details about the probabilistic forecasting experiments of Section 4.4

Datasets:- We consider three popular time-series datasets namely Female Births (365×1) (datamarket.com), Minimum Temperature (3651×1) (machinelearningmastery.com) and Beer Production (464×1) (Australian (1996)). 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.

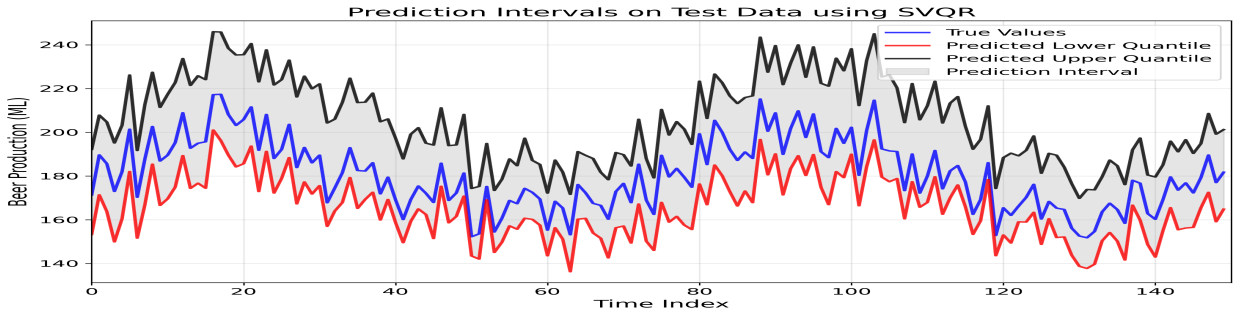
Baseline Methods:- We also train several recent and widely adopted Quantile based deep learning architectures for probabilistic forecasting developed in distribution-free setting, including Quantile-based LSTM, Quantile-based GRU, Quantile-based TCN and Quantile-based Transformers. The SVQR and SSVQR models are fundamentally quantile-based probabilistic forecasting approaches; therefore, when comparing them with deep learning counterparts, we should only consider quantile-based deep learning methods. However, for the sake of extensive comparisons, we also report the performance of deep probabilistic forecasting models that adopt different probabilistic forecasting approaches, including the Tube Loss LSTM and Quality-Driven (QD) Loss LSTM models (Pearce et al. (2018)), as well as the DeepAR model (Salinas et al. (2020)). 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.

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, Quantile-based GRU, Quantile-based TCN, Quantile-based Transformers, Tube Loss LSTM, Deep AR and 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.



(a)



(b)

Figure 9: Probabilistic forecasting with SSVQR model on daily (a) Temperature and (b) Beer Production dataset