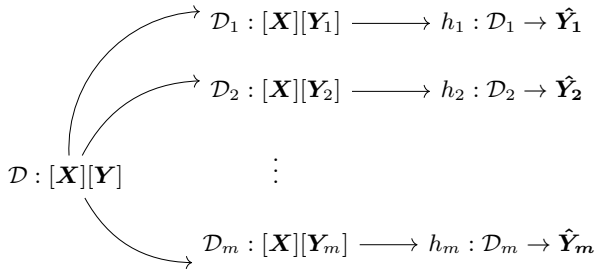


SVR Flow Diagram. Firstly, the multi-target dataset is divided into m ST datasets, $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_m$. Then m models, $h_1, h_2, \ldots, h_m$, are independently trained for each ST dataset.

Multi-Target Support Vector Regression (SVR)

Input: Training dataset $\mathcal{D}$
Output: ST models $h_j, j = 1, \ldots, m$
1: **for** $j = 1$ to m **do**
2: $\quad \mathcal{D}_j = \{\boldsymbol{X}, \boldsymbol{Y}_j\}$ ▷ Get ST data
3: $\quad h_j : \boldsymbol{X} \rightarrow \mathbb{R}$ ▷ Build ST model for the j^{th} target
4: **end for**

Build Chained Model

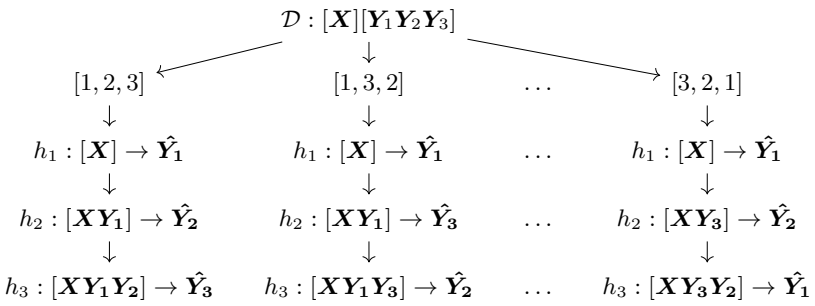
Input: Training dataset $\mathcal{D}$, random chain $\boldsymbol{C}$
Output: A chained model $h_j, j = \{1, \ldots, m\}$
1: $\mathcal{D}_1 = \{\boldsymbol{X}, \boldsymbol{Y}_{\boldsymbol{C}_1}\}$ ▷ Initialize first dataset
2: **for** $j = 1$ to m **do** ▷ For each target in chain $\boldsymbol{C}$
3: $\quad h_j : \mathcal{D}_j \rightarrow \mathbb{R}$ ▷ Train model on appended dataset
4: $\quad$ **if** $j < m$ **then**
5: $\quad\quad \mathcal{D}_{j+1} = \{\mathcal{D}_j, \boldsymbol{Y}_{\boldsymbol{C}_j}\}$ ▷ Append new target in chain to dataset
6: $\quad$ **end if**
7: **end for**

$$\min_{\boldsymbol{w},b,\boldsymbol{\xi}} \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_I \xi_I, \qquad \max_{\boldsymbol{\alpha}} \sum_I \alpha_I - \frac{1}{2}\sum_I \sum_{K\in I} \alpha_I \alpha_K Y_I Y_K \mathcal{K}\left(\boldsymbol{x}_{s_I}, \boldsymbol{x}_{s_K}\right)$$

$$\text{s.t. } Y_I(\langle \boldsymbol{w}, \boldsymbol{x}_{s_I}\rangle + b) \geq 1 - \xi_I,\ \forall I \in \{1, \ldots, n\}, \qquad \text{s.t.} \sum_I \alpha_I Y_I = 0,$$

$$\xi_I \geq 0,\ \forall I \in \{1, \ldots, n\}, \qquad 0 \leq \alpha_I \leq C,\ \forall I \in \{1, \ldots, n\},$$

$$s_I = \underset{i\in I}{\operatorname{argmax}}(\langle \boldsymbol{w}, \boldsymbol{x}_i\rangle + b),\ \forall I \in \{1, \ldots, n\}. \qquad s_I = \underset{i\in I}{\operatorname{argmax}}(\boldsymbol{o}_I),\ \forall I \in \{1, \ldots, n\}.$$

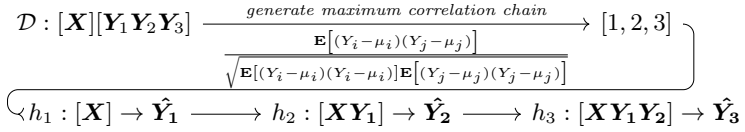


SVRRC Flow Diagram on a dataset with three targets. SVRRC first builds the six random chains of the target's indices (three examples are shown). It then constructs a chained model by proceeding recursively over the chain, building a model, and appending the current target to the input space to predict the next target in the chain.

Multi-Target SVR with Random-Chains (SVRRC)

Input: Training dataset $\mathcal{D}$, c random chains $\mathcal{C}$
Output: An ensemble of chained models $h_{\mathcal{C}}$

1: **for each** $\boldsymbol{C} \in \mathcal{C}$ **do** ▷ For each random chain
2: $h_{\boldsymbol{C}} \leftarrow$ `buildChainedModel`$(\mathcal{D}, \boldsymbol{C})$ ▷ Build a chained model for chain $\boldsymbol{C}$
3: **end for**



SVRCC Flow Diagram on a sample dataset with three targets. SVRCC first finds the direction of maximum correlation among the targets and uses that order as the only chain. It then constructs the chained model, as done in SVRRC.

Multi-Target SVR with max-Correlation Chain (SVRCC)

1: $\mathbf{P} = corrcoef(\boldsymbol{Y})$ ▷ Find correlation coefficient matrix for target variables
2: $\boldsymbol{C} = \sum_{i=1}^{n} \mathbf{P}_{ij}, \forall j = 1, \ldots, m$ ▷ Sum rows of the correlation matrix
3: $\boldsymbol{C} =$ `sort` $(\boldsymbol{C}, \textbf{decreasing})$ ▷ Sort sums in decreasing order
4: $h_{\boldsymbol{C}} =$ `buildChainedModel`$(\mathcal{D}, \boldsymbol{C})$ ▷ Build a max-correlation chained model

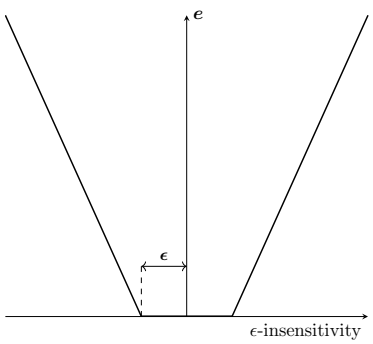


Figure 1: Vapnik's ϵ-insensitivity loss function.

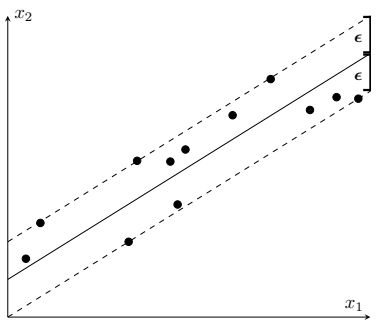


Figure 2: Linear support vector regression example solution on a toy 2D dataset.

$$\min_{(\boldsymbol{w},b)\in\mathcal{H}_o\times\mathbb{R}} R \;=\; \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{n} L(y_i,\, o_{(w,b)}(\boldsymbol{x}_i)) \tag{1}$$

$$L(y_i,\, o_{(w,b)}(\boldsymbol{x}_i)) = \max\left\{0, 1 - y_i o_{(w,b)}(\boldsymbol{x}_i)\right\} \tag{2}$$

$$\min_{(\boldsymbol{w},b)\in\mathcal{H}_o\times\mathbb{R}} R \;=\; \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{n} (|y_i - o_{(w,b)}(\boldsymbol{x_i})|_\epsilon) \tag{3}$$

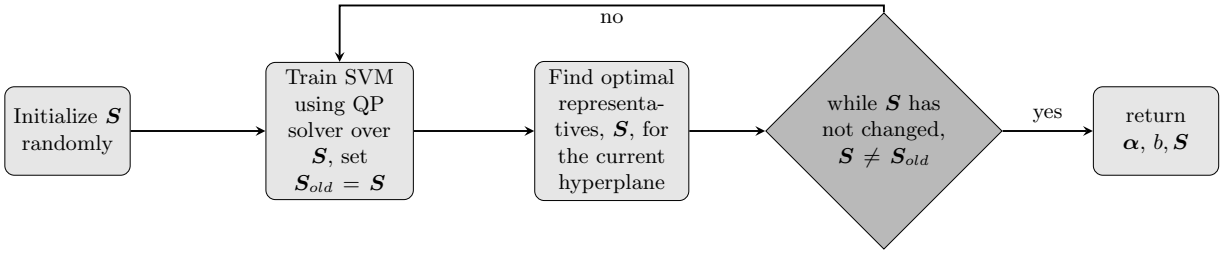
$$L(y_i,\, o_{(w,b)}(\boldsymbol{x}_i)) = \begin{cases} 0 & if |y_i - o_{(w,b)}(\boldsymbol{x_i})| \leq \epsilon \\ |y_i - o_{(w,b)}(\boldsymbol{x_i})| - \epsilon & \text{otherwise.} \end{cases} \tag{4}$$

Average Relative Root Mean Square Error (aRRMSE) for MT regressors

Datasets	MORF	ST	MTS	MTSC	RC	ERC	ERCC	SVR	SVRRC	SVRCC
Slump	0.6939	0.6886	0.6690	0.6938	0.7019	0.7022	0.6886	0.5765	**0.5545**	0.5560
Polymer	0.6159	0.5971	0.5778	0.6493	0.6270	0.6544	0.6131	0.5573	0.5253	**0.5116**
Andro	0.5097	0.5979	0.5155	0.5633	0.5924	0.5885	0.5666	0.4856	0.4651	**0.4455**
EDM	0.7337	0.7442	0.7413	0.7446	0.7449	0.7452	0.7443	0.7058	0.7070	**0.6978**
Solar Flare 1	1.3046	1.1357	1.1168	1.0758	0.9951	1.0457	1.0887	0.9917	0.9455	**0.9320**
Jura	0.5969	0.5874	0.5906	0.5892	0.5910	0.5896	0.5880	0.5952	**0.5764**	0.5885
Enb	0.1210	0.1165	0.1231	0.1211	0.1268	0.1250	0.1139	0.0977	0.0910	**0.0899**
Solar Flare 2	1.4167	1.1503	**0.9483**	1.0840	1.0092	1.0522	1.0928	1.0385	1.0253	1.0298
Wisconsin Cancer	0.9413	0.9314	0.9308	0.9336	**0.9305**	0.9313	0.9323	0.9555	0.9483	0.9427
California Housing	0.6611	0.6447	0.6974	0.6630	0.7131	0.6690	0.6146	0.6130	0.5945	**0.5852**
Stock	0.1653	0.1844	0.1787	0.1803	0.1802	0.1789	0.1752	0.1364	**0.1337**	0.1388
SCPF	0.8273	0.8348	0.8436	0.8308	0.8263	0.8105	0.8290	0.8164	0.8037	**0.8013**
Puma8NH	0.7858	0.8142	0.8118	0.8311	0.8199	0.8205	0.8207	**0.7655**	0.7744	0.7676
Friedman	0.9394	0.9214	0.9231	0.9210	0.9231	0.9209	0.9204	0.9218	0.9208	**0.9196**
Puma32H	0.9406	**0.8713**	0.8727	0.8791	0.8752	0.8729	0.8740	0.9364	0.9367	0.9319
Water Quality	**0.8994**	0.9085	0.9109	0.9093	0.9121	0.9097	0.9057	0.9343	0.9310	0.9045
M5SPEC	0.5910	0.5523	0.5974	0.5671	0.5552	0.5542	0.5558	0.2951	0.2935	**0.2925**
MP5SPEC	0.5522	0.5120	0.5683	0.5133	0.5145	0.5143	0.5119	0.2484	**0.2323**	0.2358
MP6SPEC	0.5553	0.5152	0.5686	0.5119	0.5198	0.5187	0.5109	0.2850	0.2669	**0.2623**
ATP7d	0.5563	0.5308	**0.5141**	0.5142	0.5558	0.5397	0.5182	0.5455	0.5371	0.5342
OES97	0.5490	0.5230	0.5229	0.5217	0.5239	0.5237	0.5222	0.4641	**0.4618**	0.4635
Osales	0.7596	0.7471	**0.7086**	0.7268	0.8318	0.7258	0.7101	0.7924	0.7924	0.7811
ATP1d	0.4173	0.3732	0.3733	0.3712	0.3790	**0.3696**	0.3721	0.3773	0.3707	0.3775
OES10	0.4518	0.4174	0.4176	0.4171	0.4178	0.4180	0.4166	0.3570	0.3555	**0.3538**
Average	0.6910	0.6625	0.6551	0.6589	0.6611	0.6575	0.6536	0.6039	0.5935	**0.5893**
Ranks	7.5000	5.7708	5.9375	6.1667	7.4375	6.3750	4.9792	4.7708	3.2708	**2.7917**

Run Time (seconds) for MT regressors

Datasets	MORF	ST	MTS	MTSC	RC	ERC	ERCC	SVR	SVRRC	SVRCC
Slump	38.1	2.6	9.9	15.9	1.8	11.1	50.5	**0.6**	1.9	0.7
Polymer	7.6	2.7	9.1	15.5	1.9	14.9	80.5	**0.5**	2.6	**0.5**
Andro	25.7	4.4	15.0	34.2	3.4	33.2	197.9	**1.1**	6.2	**1.1**
EDM	24.8	2.8	9.4	18.1	2.1	5.8	19.0	**0.9**	1.0	**0.9**
Solar Flare 1	34.1	3.5	13.6	26.7	2.7	17.7	86.9	**2.3**	9.3	2.6
Jura	64.3	7.9	31.8	74.3	6.4	43.5	254.2	**4.7**	18.7	5.3
Enb	71.4	6.6	26.1	63.6	**5.4**	15.6	69.6	11.3	17.7	15.9
Solar Flare 2	55.4	7.4	30.7	68.0	**6.3**	42.9	241.5	9.4	53.5	15.6
Wisconsin Cancer	51.4	6.1	21.9	53.7	4.9	14.8	61.6	**2.0**	2.4	**2.0**
California Housing	93.0	9.7	34.8	75.9	**8.2**	21.3	102.0	15.8	25.2	23.6
Stock	93.7	11.7	46.8	96.7	**11.0**	75.4	427.3	18.5	90.5	26.3
SCPF	66.3	19.3	65.9	176.3	**15.0**	104.2	734.2	32.8	162.8	48.8
Puma8NH	130.4	29.7	106.7	288.6	**27.9**	201.6	1227.7	94.1	516.6	177.1
Friedman	79.5	27.0	81.2	258.3	25.0	273.7	2871.6	**12.3**	322.3	18.8
Puma32H	93.9	68.1	181.0	635.0	87.7	667.9	6087.0	**32.2**	1018.7	53.1
Water Quality	108.4	**93.1**	262.1	912.3	127.2	925.4	10993.3	110.2	2567.9	189.5
M5SPEC	89.8	68.9	166.3	604.6	73.7	262.3	3132.1	**39.2**	546.7	45.1
MP5SPEC	84.5	94.6	221.2	888.3	91.5	557.0	6864.1	**49.3**	1132.1	58.4
MP6SPEC	90.3	93.4	212.6	871.0	89.1	557.6	6761.3	**47.2**	1227.1	58.5
ATP7d	**70.5**	262.6	452.1	2319.8	242.1	1779.2	24373.8	80.0	1897.4	136.5
OES97	**83.4**	485.3	1146.6	4928.9	499.8	5315.0	58072.1	148.2	3759.1	342.6
Osales	**92.0**	1094.8	2340.7	8322.2	986.5	11361.2	122265.3	437.0	4830.1	843.6
ATP1d	**70.7**	272.9	476.5	2568.9	261.9	2138.9	26768.9	95.0	2127.8	174.4
OES10	**90.0**	738.9	1633.6	6682.9	688.5	7150.8	83533.1	229.1	5419.4	577.1
Average	71.2	142.2	316.5	1250.0	136.2	1316.3	14803.2	**61.4**	1073.2	117.4
Ranks	5.5	3.71	6.0	8.29	3.0	7.08	9.92	**1.88**	6.71	2.92



A summary of the steps performed by MIRSVM. The representatives are first randomly initialized and continuously updated according to the current hyperplane. Upon completion, the model is returned along with the optimal bag-representatives.

Multi-Instance Representative SVM (MIRSVM)

Input: Training dataset $\mathcal{D}$, SVM Parameters C and σ
Output: SVM model parameters $\boldsymbol{\alpha}$ and b, Bag Representative IDs $\boldsymbol{S}$

for $I \in \{1, \ldots, n\}$ **do**
  $\boldsymbol{S}_I \leftarrow \text{rand}\left(|\mathcal{B}_I|, 1, 1\right)$ ▷ Assign each bag a random instance
end for
while $\boldsymbol{S} \neq \boldsymbol{S}_{old}$ **do**
  $\boldsymbol{S}_{old} \leftarrow \boldsymbol{S}$
  $\boldsymbol{X}_S \leftarrow \boldsymbol{X}(\boldsymbol{S})$, $\boldsymbol{Y}_S \leftarrow \boldsymbol{Y}(\boldsymbol{S})$ ▷ Initialize the representative dataset
  $\boldsymbol{G} \leftarrow (\boldsymbol{Y}_S \times \boldsymbol{Y}_S) \cdot \mathcal{K}\left(\boldsymbol{X}_S, \boldsymbol{X}_S, \sigma\right)$ ▷ Build Gram matrix
  $\boldsymbol{\alpha} \leftarrow \text{quadprog}\left(\boldsymbol{G}, -\mathbf{1}^n, \boldsymbol{Y}_S, \mathbf{0}^n, \mathbf{0}^n, \boldsymbol{C}^n\right)$ ▷ Solve QP Problem
  $\boldsymbol{sv} \leftarrow \text{find}\left(0 < \boldsymbol{\alpha} \leq C\right)$ ▷ Get the support vector indices
  $n_{\boldsymbol{sv}} \leftarrow \text{count}\left(0 < \boldsymbol{\alpha} \leq C\right)$ ▷ Get the number of support vectors
  $b \leftarrow \frac{1}{n_{\boldsymbol{sv}}} \sum_{i=1}^{n_{\boldsymbol{sv}}} \left(\boldsymbol{Y}_{\boldsymbol{sv}} - \boldsymbol{G}_{\boldsymbol{sv}} * (\boldsymbol{\alpha}_{\boldsymbol{sv}} \cdot \boldsymbol{Y}_{\boldsymbol{sv}})\right)$ ▷ Calculate the bias term
  for $I \in \{1, \ldots, n\}$ **do**
    $\boldsymbol{G}_I \leftarrow (Y_I \times \boldsymbol{Y}_S) \cdot \mathcal{K}\left(\mathcal{B}_I, \boldsymbol{X}_S, \sigma\right)$
    $\boldsymbol{S}_I \leftarrow \text{argmax}_{i \in I}\left(\boldsymbol{G}_I * \boldsymbol{\alpha} + b\right)$ ▷ Select optimal bag-representatives
  end for
end while