$$
\begin{tikzcd}
 & \mathcal{D}_1 : [\boldsymbol{X}][\boldsymbol{Y}_1] \arrow[r] & h_1 : \mathcal{D}_1 \to \boldsymbol{\hat{Y}_1} \\
 & \mathcal{D}_2 : [\boldsymbol{X}][\boldsymbol{Y}_2] \arrow[r] & h_2 : \mathcal{D}_2 \to \boldsymbol{\hat{Y}_2} \\
\mathcal{D} : [\boldsymbol{X}][\boldsymbol{Y}] \arrow[uur, bend left] \arrow[ur, bend left] \arrow[dr, bend right] & \vdots & \\
 & \mathcal{D}_m : [\boldsymbol{X}][\boldsymbol{Y}_m] \arrow[r] & h_m : \mathcal{D}_m \to \boldsymbol{\hat{Y}_m}
\end{tikzcd}
$$

Multi-Target Support Vector Regression (SVR)

Input: Training dataset $\mathcal{D}$
Output: ST models $h_j, j = 1, \ldots, m$

```
for j = 1 to m do
    D_j = {X, Y_j}                      ▷ Get ST data
    h_j : X → ℝ                         ▷ Build ST model for the j^th target
end for
```

Build Chained Model

Input: Training dataset $\mathcal{D}$, random chain $\boldsymbol{C}$
Output: A chained model $h_j, j = \{1, \ldots, m\}$

```
D_1 = {X, Y_{C_1}}                      ▷ Initialize first dataset
for j = 1 to m do                       ▷ For each target in chain C
    h_j : D_j → ℝ                       ▷ Train model on appended dataset
    if j < m then
        D_{j+1} = {D_j, Y_{C_j}}        ▷ Append new target in chain to dataset
    end if
end for
```

$$
\begin{array}{cccc}
 & \mathcal{D} : [\boldsymbol{X}][\boldsymbol{Y}_1\boldsymbol{Y}_2\boldsymbol{Y}_3] & & \\
\swarrow & \downarrow & & \searrow \\
[1,2,3] & [1,3,2] & \ldots & [3,2,1] \\
\downarrow & \downarrow & & \downarrow \\
h_1 : [\boldsymbol{X}] \to \boldsymbol{\hat{Y}_1} & h_1 : [\boldsymbol{X}] \to \boldsymbol{\hat{Y}_1} & \ldots & h_1 : [\boldsymbol{X}] \to \boldsymbol{\hat{Y}_1} \\
\downarrow & \downarrow & & \downarrow \\
h_2 : [\boldsymbol{X}\boldsymbol{Y_1}] \to \boldsymbol{\hat{Y}_2} & h_2 : [\boldsymbol{X}\boldsymbol{Y_1}] \to \boldsymbol{\hat{Y}_3} & \ldots & h_2 : [\boldsymbol{X}\boldsymbol{Y_3}] \to \boldsymbol{\hat{Y}_2} \\
\downarrow & \downarrow & & \downarrow \\
h_3 : [\boldsymbol{X}\boldsymbol{Y_1}\boldsymbol{Y_2}] \to \boldsymbol{\hat{Y}_3} & h_3 : [\boldsymbol{X}\boldsymbol{Y_1}\boldsymbol{Y_3}] \to \boldsymbol{\hat{Y}_2} & \ldots & h_3 : [\boldsymbol{X}\boldsymbol{Y_3}\boldsymbol{Y_2}] \to \boldsymbol{\hat{Y}_1}
\end{array}
$$

Multi-Target SVR with Random-Chains (SVRRC)

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

```
for each C ∈ 𝒞 do                            ▷ For each random chain
    h_C ← buildChainedModel(D, C)            ▷ Build a chained model for chain C
end for
```

$$\mathcal{D} : [\boldsymbol{X}][\boldsymbol{Y}_1\boldsymbol{Y}_2\boldsymbol{Y}_3] \xrightarrow[\frac{\mathbf{E}[(Y_i-\mu_i)(Y_j-\mu_j)]}{\sqrt{\mathbf{E}[(Y_i-\mu_i)(Y_i-\mu_i)]\mathbf{E}[(Y_j-\mu_j)(Y_j-\mu_j)]}}]{\textit{generate maximum correlation chain}} [1,2,3]$$

$$\hookrightarrow h_1 : [\boldsymbol{X}] \to \boldsymbol{\hat{Y}_1} \longrightarrow h_2 : [\boldsymbol{X}\boldsymbol{Y_1}] \to \boldsymbol{\hat{Y}_2} \longrightarrow h_3 : [\boldsymbol{X}\boldsymbol{Y_1}\boldsymbol{Y_2}] \to \boldsymbol{\hat{Y}_3}$$

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

```
P = corrcoef(Y)                      ▷ Find correlation coefficient matrix for target variables
C = Σ_{i=1}^{n} P_ij, ∀j = 1, ..., m ▷ Sum rows of the correlation matrix
C = sort (C, decreasing)             ▷ Sort sums in decreasing order
h_C = buildChainedModel(D, C)        ▷ Build a max-correlation chained model
```

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

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

$$
\begin{aligned}
&\min_{\boldsymbol{w},b,\boldsymbol{\xi}} \; \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_I \xi_I, \\
&\text{s.t. } Y_I(\langle \boldsymbol{w}, \boldsymbol{x}_{s_I}\rangle + b) \geq 1 - \xi_I, \; \forall I \in \{1,\ldots,n\}, \\
&\qquad \xi_I \geq 0, \; \forall I \in \{1,\ldots,n\}, \\
&\qquad s_I = \operatorname*{argmax}_{i\in I}(\langle \boldsymbol{w},\, \boldsymbol{x}_i\rangle + b), \; \forall I \in \{1,\ldots,n\}.
\end{aligned}
$$

$$
\begin{aligned}
&\max_{\boldsymbol{\alpha}} \sum_I \alpha_I - \frac{1}{2}\sum_I \sum_{K\in I} \alpha_I \alpha_K Y_I Y_K \mathcal{K}\left(\boldsymbol{x}_{s_I}, \boldsymbol{x}_{s_K}\right) \\
&\text{s.t. } \sum_I \alpha_I Y_I = 0, \\
&\qquad 0 \leq \alpha_I \leq C, \; \forall I \in \{1,\ldots,n\}, \\
&\qquad s_I = \operatorname*{argmax}_{i\in I}(\boldsymbol{o}_I), \; \forall I \in \{1,\ldots,n\}.
\end{aligned}
$$

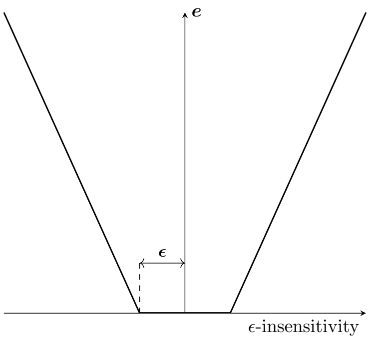


Figure 1: Vapnik's ϵ-insensitivity loss function.

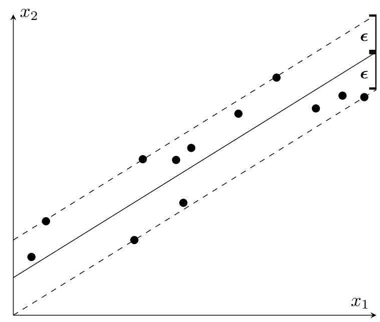
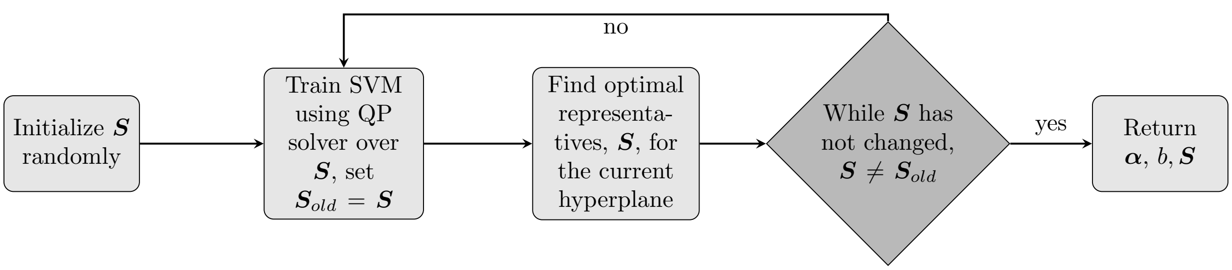


Figure 2: Linear support vector regression example solution on a toy 2D dataset.



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 $\boldsymbol{S}$

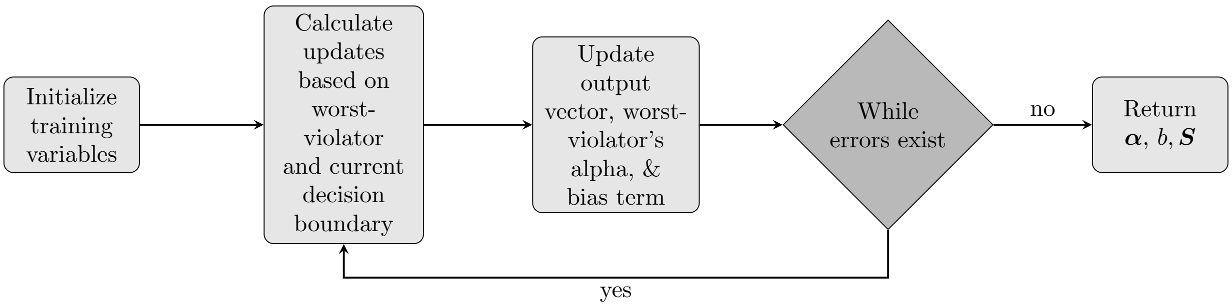
- **for** $I \in \{1,\ldots,n\}$ **do**
  - $\boldsymbol{S}_I \leftarrow \text{rand}\left(|\mathcal{B}_I|, 1, 1\right)$ ▷ Assign each bag a random instance
- **end for**
- **while** $\boldsymbol{S} \neq \boldsymbol{S}_{old}$ **do**
  - $\boldsymbol{S}_{old} \leftarrow \boldsymbol{S}$
  - $\boldsymbol{X}_S \leftarrow \boldsymbol{X}(\boldsymbol{S})$, $\boldsymbol{Y}_S \leftarrow \boldsymbol{Y}(\boldsymbol{S})$ ▷ 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**

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

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

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

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



OnLine Learning Algorithm using Worst-Violators (OLLAWV)

Input: $\mathcal{D}$, C, γ, β, M
Output: $\boldsymbol{\alpha}$, b, $\boldsymbol{S}$

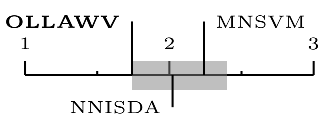
$\boldsymbol{\alpha} \leftarrow \mathbf{0}$, $b \leftarrow 0$, $\boldsymbol{S} \leftarrow \mathbf{0}$ ▷ Initialize OLLAWV model parameters
$\boldsymbol{o} \leftarrow \mathbf{0}$, $t \leftarrow 0$ ▷ Initialize the output vector and iteration counter
$wv \leftarrow 0$, $yo \leftarrow y_{wv} * \boldsymbol{o}_{wv}$ ▷ Initialize hinge loss error and worst-violator index
while $yo < M$ **do**
  $t \leftarrow t + 1$
  $\eta \leftarrow 2/\sqrt{t}$ ▷ Learning rate
  $\Lambda \leftarrow \eta * C * y_{wv}$ ▷ Calculate hinge loss update
  $B \leftarrow (\Lambda * \beta)/n$ ▷ Calculate bias update
  $\boldsymbol{o} \leftarrow \boldsymbol{o} + \Lambda * \mathcal{K}(\boldsymbol{x}_{\neg \boldsymbol{S}}, \boldsymbol{x}_{wv}, \gamma) + B$ ▷ Update output vector
  $\boldsymbol{\alpha}_{wv} \leftarrow \boldsymbol{\alpha}_{wv} + \Lambda$ ▷ Update worst-violator's alpha value
  $b \leftarrow b + B$ ▷ Update bias term
  $\boldsymbol{S}_t \leftarrow wv$ ▷ Save index of worst-violator
  $[yo, wv] \leftarrow \min_{wv \in \{\neg \boldsymbol{S}\}} \{y_{wv} \cdot o_{wv}\}$ ▷ Find the worst-violator
end while

Classification Datasets

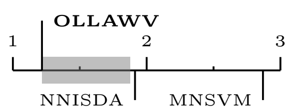
Dataset	# Samples	# Attributes	# Classes
small datasets			
iris	150	4	3
teach	151	5	3
wine	178	13	3
cancer	198	32	2
sonar	208	60	2
glass	214	9	6
vote	232	16	2
heart	270	13	2
dermatology	366	33	6
prokaryotic	997	20	3
eukaryotic	2,427	20	4
medium datasets			
optdigits	5,620	64	10
satimage	6,435	36	6
usps	9,298	256	10
pendigits	10,992	16	10
reuters	11,069	8,315	2
letter	20,000	16	26
large datasets			
adult	48,842	123	2
w3a	49,749	300	2
shuttle	58,000	7	7
web (w8a)	64,700	300	2
ijcnn1	141,691	22	2
intrusion	5,209,460	127	2

Comparison of OLLAWV vs. NNISDA and MNSVM

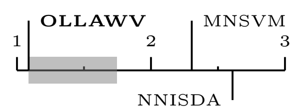
Dataset	Accuracy (%)			Run Time (s)			Support Vectors (%)		
	OLLAWV	NNISDA	MNSVM	OLLAWV	NNISDA	MNSVM	OLLAWV	NNISDA	MNSVM
small datasets									
iris	**97.33**	94.00	96.67	**0.05**	0.27	3.57	**13.50**	40.20	29.80
teach	52.32	52.31	**52.95**	**0.12**	0.44	8.85	**69.19**	99.80	87.40
wine	**98.87**	96.60	96.60	**0.28**	0.43	4.84	**15.02**	44.40	48.60
cancer	80.36	**81.86**	81.38	**0.49**	0.85	4.46	**42.79**	83.80	89.60
sonar	**92.32**	89.48	87.57	**0.59**	0.98	3.03	**31.26**	73.00	66.00
glass	**72.41**	67.81	69.30	**0.46**	1.01	11.94	**62.84**	90.80	87.60
vote	**96.54**	96.11	93.99	**0.26**	0.46	1.49	**13.36**	33.20	34.00
heart	82.22	**83.33**	**83.33**	**0.50**	0.91	6.45	**37.69**	73.00	82.00
dermatology	97.82	**98.36**	**98.36**	**1.62**	2.47	11.68	**36.94**	59.00	59.80
prokaryotic	88.96	88.86	**88.97**	**6.09**	10.64	50.86	**29.01**	51.20	49.00
eukaryotic	77.38	79.56	**81.21**	61.95	**49.16**	342.76	**54.11**	76.40	72.60
medium datasets									
optdigits	99.11	99.29	**99.31**	**411**	528	787	**28.64**	31.60	30.60
satimage	91.66	**92.39**	92.35	1,334	**687**	1,094	**20.72**	45.00	44.80
usps	97.49	98.05	**98.24**	10,214	**5,245**	7,777	**11.22**	29.40	28.00
pendigits	99.56	**99.62**	99.61	**723**	909	1,500	**10.27**	17.60	16.60
reuters	98.03	**98.08**	97.99	**954**	1,368	1,657	**8.770**	18.20	18.60
letter	96.99	99.11	**99.13**	**5,259**	12,009	26,551	**43.56**	57.60	56.60
large datasets									
adult	84.75	85.07	**85.13**	**21,025**	72,552	123,067	**34.66**	56.00	56.60
w3a	**98.86**	98.82	98.82	**6,532**	15,951	24,562	**3.270**	14.60	12.40
shuttle	99.77	99.83	**99.87**	**2,833**	7,420	45,062	**2.010**	6.00	16.40
web	98.94	**99.00**	99.00	**12,067**	30,583	38,040	**4.320**	13.20	10.80
ijcnn1	98.31	99.34	**99.41**	**162,587**	296,917	370,144	16.36	11.00	**7.600**
intrusion	**99.77**	99.67	99.66	**2,402,804**	4,646,810	3,772,113	**0.780**	2.000	1.700
Average	**91.29**	91.15	91.25	**114,209**	221,350	191,861	**25.66**	44.65	43.79
Ranks	**1.739**	2.022	2.239	**1.217**	1.913	2.869	**1.087**	2.609	2.304



Bonferroni-Dunn test for Accuracy



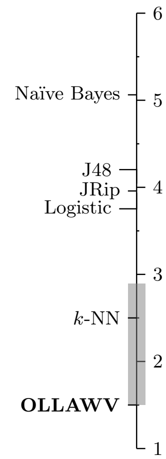
Bonferroni-Dunn test for Run Time

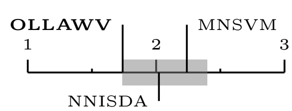


Bonferroni-Dunn test for % Support Vectors

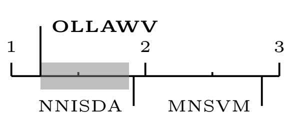
Accuracy (%) for Non-SVM Methods vs. OLLAWV

Dataset	OLLAWV	k-NN	J48	JRip	Naïve Bayes	Logistic
small datasets						
iris	**97.33 ± 1.49**	96.00 ± 3.65	94.00 ± 2.79	90.67 ± 4.35	96.00 ± 2.79	97.33 ± 2.79
teach	52.32 ± 3.46	**59.64 ± 2.89**	49.72 ± 7.58	56.75 ± 9.60	53.75 ± 6.46	51.77 ± 6.68
wine	**98.87 ± 1.54**	97.73 ± 3.72	90.43 ± 5.83	93.24 ± 3.27	96.60 ± 3.14	96.05 ± 2.58
cancer	**80.36 ± 5.80**	77.32 ± 6.93	73.81 ± 8.57	73.78 ± 5.81	67.73 ± 5.07	77.32 ± 7.78
sonar	**92.32 ± 3.11**	88.99 ± 4.59	76.16 ± 10.6	75.18 ± 6.77	73.69 ± 7.65	75.18 ± 7.31
glass	**72.41 ± 2.28**	67.73 ± 5.91	65.06 ± 5.51	65.59 ± 9.66	49.46 ± 5.19	62.04 ± 5.75
vote	**96.54 ± 1.87**	92.26 ± 3.19	95.70 ± 2.12	96.54 ± 2.45	92.24 ± 3.24	93.54 ± 2.59
heart	82.22 ± 2.93	79.63 ± 5.71	78.52 ± 2.81	80.74 ± 4.06	**84.44 ± 4.46**	83.33 ± 3.93
dermatology	**97.82 ± 0.05**	96.18 ± 1.78	94.52 ± 2.21	91.27 ± 5.08	97.28 ± 1.64	96.98 ± 2.28
prokaryotic	**88.96 ± 2.14**	87.96 ± 3.01	78.54 ± 1.62	79.13 ± 2.78	62.38 ± 3.54	87.57 ± 2.56
eukaryotic	77.38 ± 1.96	**81.42 ± 2.06**	65.27 ± 2.92	66.42 ± 3.47	39.27 ± 3.43	69.55 ± 1.34
medium datasets						
optdigits	**99.11 ± 0.38**	98.74 ± 0.39	90.87 ± 1.09	91.28 ± 0.40	92.42 ± 0.75	95.05 ± 0.91
satimage	**91.66 ± 0.80**	90.38 ± 0.72	85.64 ± 1.21	85.33 ± 0.77	85.41 ± 0.92	88.14 ± 1.11
usps	**97.49 ± 0.22**	97.04 ± 0.47	88.73 ± 0.46	89.20 ± 1.00	79.45 ± 0.59	91.88 ± 0.65
pendigits	**99.56 ± 0.12**	99.33 ± 0.17	96.24 ± 0.31	96.34 ± 0.41	88.34 ± 0.65	95.59 ± 0.18
reuters	**98.03 ± 0.22**	97.15 ± 0.43	96.90 ± 0.32	97.18 ± 0.44	93.52 ± 0.02	69.54 ± 0.28
letter	**96.99 ± 0.21**	95.71 ± 0.19	87.34 ± 0.68	87.02 ± 0.66	74.12 ± 0.97	77.45 ± 0.16
large datasets						
adult	**84.75 ± 0.26**	83.85 ± 0.28	84.38 ± 0.28	83.73 ± 0.17	80.57 ± 0.09	82.46 ± 0.14
w3a	**98.86 ± 0.04**	98.60 ± 0.06	98.71 ± 0.05	98.41 ± 0.10	96.71 ± 0.20	98.61 ± 0.12
shuttle	99.77 ± 0.03	99.93 ± 0.03	**99.97 ± 0.02**	99.96 ± 0.02	98.57 ± 0.24	96.83 ± 0.12
web	**98.94 ± 0.05**	98.89 ± 0.06	98.79 ± 0.09	98.50 ± 0.13	96.71 ± 0.21	98.70 ± 0.08
ijcnn1	98.31 ± 0.07	**98.48 ± 0.04**	98.40 ± 0.09	98.11 ± 0.10	90.69 ± 0.26	92.29 ± 0.16
intrusion	**99.77 ± 0.02**	88.20 ± 1.06	58.01 ± 26.6	87.66 ± 3.79	49.75 ± 30.7	65.15 ± 15.7
Average	**91.29 ± 1.26**	90.05 ± 2.06	84.60 ± 3.64	86.18 ± 2.84	79.96 ± 3.58	84.45 ± 2.83
Ranks	**1.500**	2.500	4.041	3.958	5.063	3.938

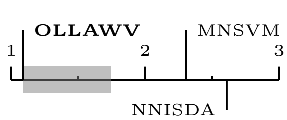




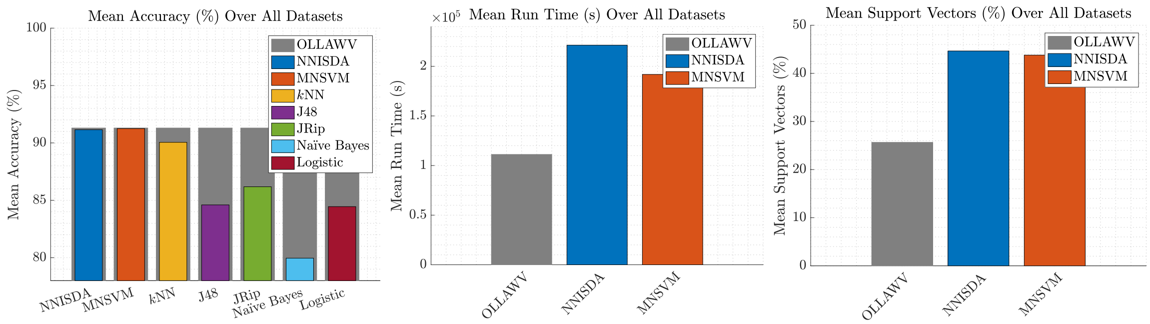
Accuracy (%)



Run Time (s)



Support Vectors (%)



$$C \in \{4^n\}, \quad n = \{-2, \ldots, 5\} \tag{5a}$$
$$\gamma \in \{4^n\}, \quad n = \{-5, \ldots, 2\} \tag{5b}$$

Algorithm Hyperparameters

Algorithm	Parameters
SVM	Penalty: $C \in \{4^n\}, n = \{-2, \ldots, 5\}$ RBF Kernel: $\gamma \in \{4^n\}, n = \{-5, \ldots, 2\}$
k-NN	Number of neighbors: $k \in \{1,\ 3,\ 5,\ 7\}$
J48	Pruning: {True, False} Pruning Confidence: {0.1, 0.25, 0.5}
JRip	Pruning: {True, False}
Naïve Bayes	Use kernel estimation: {True, False}
Logistic	Log-likelihood: $\left\{1\mathrm{e}^{-7},\ 1\mathrm{e}^{-8},\ 1\mathrm{e}^{-9}\right\}$

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

$$\min_{(\boldsymbol{w},b)\in\mathcal{H}_o\times\mathbb{R}} R \;=\; \frac{1}{2}||\boldsymbol{w}||^2 + CL(y_i,\, o_{(w,b)}(\boldsymbol{x}_i)).$$

$$\min_{\boldsymbol{w}\in\mathcal{H}_o\times\mathbb{R}} R \;=\; \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{n} \max\left\{0, 1 - y_i o_{(w)}(\boldsymbol{x}_i)\right\}$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta\frac{\partial R}{\partial \boldsymbol{w}}$$

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \eta\begin{cases} Cy_i\boldsymbol{x_i} - \boldsymbol{w} & y_i o_i < 1 \\ -\boldsymbol{w} & \text{otherwise}\end{cases}$$

$$\sum_{i=1}^{n}\alpha_i\phi(\boldsymbol{x}_i) \leftarrow \sum_{i=1}^{n}\alpha_i\phi(\boldsymbol{x}_i) + \eta\begin{cases} Cy_i\phi(\boldsymbol{x_i}) - \sum_{i=1}^{n}\alpha_i\phi(\boldsymbol{x}_i) & y_i o_i < 1 \\ -\sum_{i=1}^{n}\alpha_i\phi(\boldsymbol{x}_i) & \text{otherwise}\end{cases}$$

$$\forall i : \alpha_i\phi(\boldsymbol{x}_i) \leftarrow \alpha_i\phi(\boldsymbol{x}_i) + \eta\begin{cases} (Cy_i\phi(\boldsymbol{x_i}) - \alpha_i\phi(\boldsymbol{x}_i)) & y_i o_i < 1 \\ (-\alpha_i\phi(\boldsymbol{x}_i)) & \text{otherwise}\end{cases}$$

$$\forall i : \alpha_i \leftarrow \alpha_i + \eta\begin{cases} (Cy_i - \alpha_i) & y_i o_i < 1 \\ (-\alpha_i) & \text{otherwise}\end{cases}$$

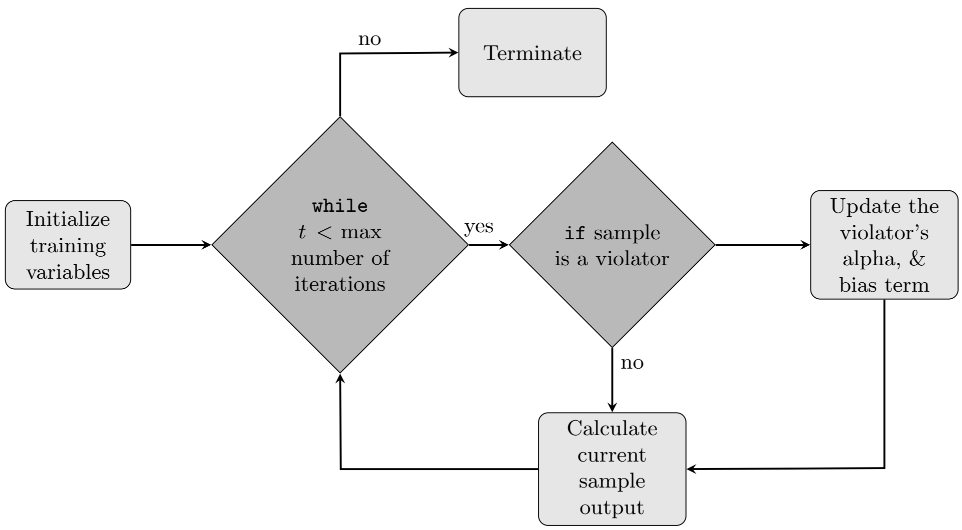
$$\Lambda \leftarrow Cy_i$$
$$\alpha_i \leftarrow \alpha_i + \eta\Lambda$$
$$b \leftarrow b + \eta\Lambda$$

$$\forall i : b \leftarrow b + \eta\frac{Cy_i}{n}$$

$$\boldsymbol{o} \leftarrow \boldsymbol{o} + \Lambda * \mathcal{K}\left(\boldsymbol{x}_{\neg\boldsymbol{S}},\, \boldsymbol{x}_{wv}, \gamma\right) + B \tag{6}$$

Comparison of OLLAWV vs. OLLA-L2

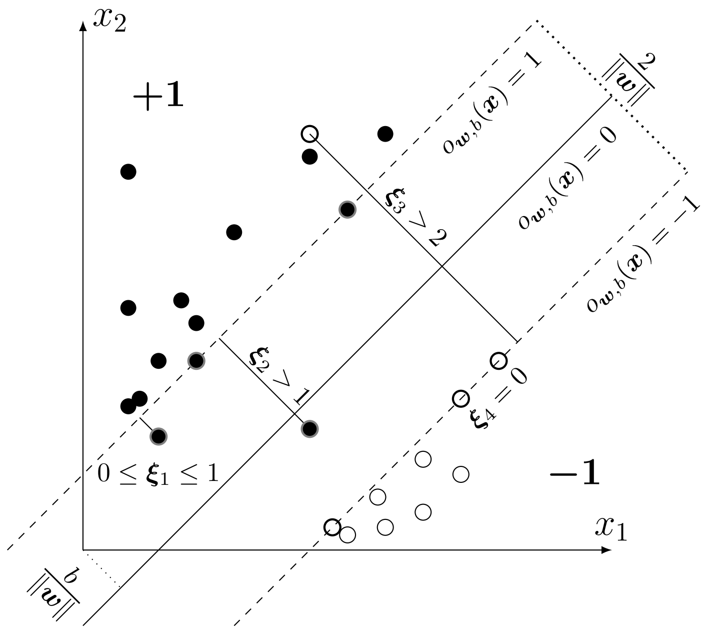
Dataset	Accuracy (%)		Run Time (s)	
	OLLA-L2	OLLAWV	OLLA-L2	OLLAWV
RBFNoDrift	93.07	**94.21**	**0.0238**	0.0329
HyperplaneSlow	87.40	**90.09**	**0.0261**	0.0353
HyperplaneFaster	87.40	**89.51**	**0.0256**	0.0263
STAGGERGeneratorF1	**100.0**	**100.0**	0.0034	**0.0021**
HyperplaneFaster02	87.41	**89.49**	**0.0257**	0.0268
MixedGeneratorBT	92.45	**98.00**	**0.0108**	0.0205
MixedGeneratorBF	92.55	**98.03**	**0.0107**	0.0299
SineGeneratorF1BF	97.37	**97.79**	**0.0091**	0.0122
SineGeneratorF2BF	97.37	**97.79**	**0.0091**	0.0121
STAGGERGeneratorF1BF	**100.0**	**100.0**	0.0035	**0.0021**
STAGGERGeneratorF2BF	**100.0**	**100.0**	0.0039	**0.0022**
HyperplaneFasterAN0	87.40	**89.51**	**0.0255**	0.0263
HyperplaneFasterAN5	87.29	**89.29**	**0.0258**	0.0264
SEASuddenAN0	84.01	**87.80**	0.0494	**0.0208**
SEASuddenAN05	83.69	**87.53**	0.0494	**0.0284**
Average	91.83	**93.94**	**0.0201**	0.0203
Rank	1.900	**1.100**	**1.3333**	1.6667



OnLine Learning Algorithm - List 2 (OLLA-L2)

Input: $\boldsymbol{X}, \boldsymbol{Y}, \beta, n, e$
Output: $\boldsymbol{\alpha}$, b, $\boldsymbol{S}$

```
α ← 0, b ← 0, S ← 0, o ← 0, i ← 0          ▷ Initialize model and algorithm parameters
for t = 1, ..., n ∗ e do
    η ← 2/√t                                ▷ Learning rate in function of time
    if y_i o_i ≤ 1 then                     ▷ Check if current sample is a violator
        Calculate Λ and P                   ▷ Calculate update parameters
        S ← [S ∪ i]                         ▷ Save index of current violator
        α_i ← α_i + (Λ − P)                 ▷ Update violator's alpha value
        b ← b + (Λ − P)β                    ▷ Update bias term
    end if
    i ← i + 1                               ▷ Get new sample
    if i = n then                           ▷ If the sample index exceeds the number of samples
        i = 0                               ▷ Reset sample index
    end if
    o_i ← K(x_i, x_S) α_S + b               ▷ Calculate the new sample's output value
end for
```



x_2
$+1$
$o_{w,b}(\boldsymbol{x}) = 1$
$\frac{2}{\|\boldsymbol{w}\|}$
$o_{w,b}(\boldsymbol{x}) = 0$
$o_{w,b}(\boldsymbol{x}) = -1$
$\boldsymbol{\xi}_3 > 2$
$\boldsymbol{\xi}_2 > 1$
$\boldsymbol{\xi}_4 = 0$
$0 \leq \boldsymbol{\xi}_1 \leq 1$
-1
x_1
$\frac{b}{\|\boldsymbol{w}\|}$