


---

### Multi-Target Support Vector Regression (SVR)

---

**Input:** Training dataset  $\mathcal{D}$

**Output:** ST models  $h_j, j = 1, \dots, m$

- 1: **for**  $j = 1$  to  $m$  **do**
  - 2:      $\mathcal{D}_j = \{\mathbf{X}, \mathbf{Y}_j\}$  ▷ Get ST data
  - 3:      $h_j : \mathbf{X} \rightarrow \mathbb{R}$  ▷ Build ST model for the  $j^{th}$  target
  - 4: **end for**
- 

---

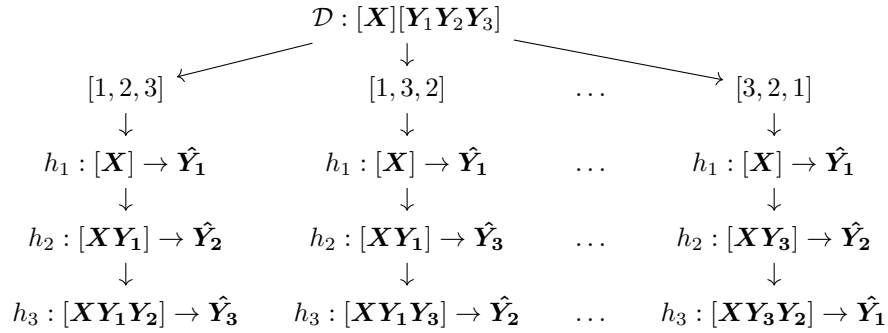
### Build Chained Model

---

**Input:** Training dataset  $\mathcal{D}$ , random chain  $\mathcal{C}$

**Output:** A chained model  $h_j, j = \{1, \dots, m\}$

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




---

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

$$\begin{array}{c}
\mathcal{D} : [\mathbf{X}][\mathbf{Y}_1 \mathbf{Y}_2 \mathbf{Y}_3] \xrightarrow[\frac{\mathbf{E}[(Y_i - \mu_i)(Y_j - \mu_j)]}{\sqrt{\mathbf{E}[(Y_i - \mu_i)(Y_i - \mu_i)]\mathbf{E}[(Y_j - \mu_j)(Y_j - \mu_j)]}}]{\text{generate maximum correlation chain}} [1, 2, 3] \\
\left. \vphantom{\frac{\mathbf{E}[(Y_i - \mu_i)(Y_j - \mu_j)]}{\sqrt{\mathbf{E}[(Y_i - \mu_i)(Y_i - \mu_i)]\mathbf{E}[(Y_j - \mu_j)(Y_j - \mu_j)]}}} \right\} \\
h_1 : [\mathbf{X}] \rightarrow \hat{\mathbf{Y}}_1 \longrightarrow h_2 : [\mathbf{X} \mathbf{Y}_1] \rightarrow \hat{\mathbf{Y}}_2 \longrightarrow h_3 : [\mathbf{X} \mathbf{Y}_1 \mathbf{Y}_2] \rightarrow \hat{\mathbf{Y}}_3
\end{array}$$

---

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

---

- |   |  |
|---|--|
| 1: $\mathbf{P} = \text{corrcoef}(\mathbf{Y})$                           | ▷ Find correlation coefficient matrix for target variables |
| 2: $\mathbf{C} = \sum_{i=1}^n \mathbf{P}_{ij}, \forall j = 1, \dots, m$ | ▷ Sum rows of the correlation matrix                       |
| 3: $\mathbf{C} = \text{sort}(\mathbf{C}, \text{decreasing})$            | ▷ Sort sums in decreasing order                            |
| 4: $h_C = \text{buildChainedModel}(\mathcal{D}, \mathbf{C})$            | ▷ 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	<b>0.5545</b>	0.5560
Polymer	0.6159	0.5971	0.5778	0.6493	0.6270	0.6544	0.6131	0.5573	0.5253	<b>0.5116</b>
Andro	0.5097	0.5979	0.5155	0.5633	0.5924	0.5885	0.5666	0.4856	0.4651	<b>0.4455</b>
EDM	0.7337	0.7442	0.7413	0.7446	0.7449	0.7452	0.7443	0.7058	0.7070	<b>0.6978</b>
Solar Flare 1	1.3046	1.1357	1.1168	1.0758	0.9951	1.0457	1.0887	0.9917	0.9455	<b>0.9320</b>
Jura	0.5969	0.5874	0.5906	0.5892	0.5910	0.5896	0.5880	0.5952	<b>0.5764</b>	0.5885
Enb	0.1210	0.1165	0.1231	0.1211	0.1268	0.1250	0.1139	0.0977	0.0910	<b>0.0899</b>
Solar Flare 2	1.4167	1.1503	<b>0.9483</b>	1.0840	1.0092	1.0522	1.0928	1.0385	1.0253	1.0298
Wisconsin Cancer	0.9413	0.9314	0.9308	0.9336	<b>0.9305</b>	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	<b>0.5852</b>
Stock	0.1653	0.1844	0.1787	0.1803	0.1802	0.1789	0.1752	0.1364	<b>0.1337</b>	0.1388
SCPF	0.8273	0.8348	0.8436	0.8308	0.8263	0.8105	0.8290	0.8164	0.8037	<b>0.8013</b>
Puma8NH	0.7858	0.8142	0.8118	0.8311	0.8199	0.8205	0.8207	<b>0.7655</b>	0.7744	0.7676
Friedman	0.9394	0.9214	0.9231	0.9210	0.9231	0.9209	0.9204	0.9218	0.9208	<b>0.9196</b>
Puma32H	0.9406	<b>0.8713</b>	0.8727	0.8791	0.8752	0.8729	0.8740	0.9364	0.9367	0.9319
Water Quality	<b>0.8994</b>	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	<b>0.2925</b>
MP5SPEC	0.5522	0.5120	0.5683	0.5133	0.5145	0.5143	0.5119	0.2484	<b>0.2323</b>	0.2358
MP6SPEC	0.5553	0.5152	0.5686	0.5119	0.5198	0.5187	0.5109	0.2850	0.2669	<b>0.2623</b>
ATP7d	0.5563	0.5308	<b>0.5141</b>	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	<b>0.4618</b>	0.4635
Osales	0.7596	0.7471	<b>0.7086</b>	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	<b>0.3696</b>	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	<b>0.3538</b>
Average	0.6910	0.6625	0.6551	0.6589	0.6611	0.6575	0.6536	0.6039	0.5935	<b>0.5893</b>
Ranks	7.5000	5.7708	5.9375	6.1667	7.4375	6.3750	4.9792	4.7708	3.2708	<b>2.7917</b>

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	<b>0.6</b>	1.9	0.7
Polymer	7.6	2.7	9.1	15.5	1.9	14.9	80.5	<b>0.5</b>	2.6	<b>0.5</b>
Andro	25.7	4.4	15.0	34.2	3.4	33.2	197.9	<b>1.1</b>	6.2	<b>1.1</b>
EDM	24.8	2.8	9.4	18.1	2.1	5.8	19.0	<b>0.9</b>	1.0	<b>0.9</b>
Solar Flare 1	34.1	3.5	13.6	26.7	2.7	17.7	86.9	<b>2.3</b>	9.3	2.6
Jura	64.3	7.9	31.8	74.3	6.4	43.5	254.2	<b>4.7</b>	18.7	5.3
Enb	71.4	6.6	26.1	63.6	<b>5.4</b>	15.6	69.6	11.3	17.7	15.9
Solar Flare 2	55.4	7.4	30.7	68.0	<b>6.3</b>	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	<b>2.0</b>	2.4	<b>2.0</b>
California Housing	93.0	9.7	34.8	75.9	<b>8.2</b>	21.3	102.0	15.8	25.2	23.6
Stock	93.7	11.7	46.8	96.7	<b>11.0</b>	75.4	427.3	18.5	90.5	26.3
SCPF	66.3	19.3	65.9	176.3	<b>15.0</b>	104.2	734.2	32.8	162.8	48.8
Puma8NH	130.4	29.7	106.7	288.6	<b>27.9</b>	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	<b>12.3</b>	322.3	18.8
Puma32H	93.9	68.1	181.0	635.0	87.7	667.9	6087.0	<b>32.2</b>	1018.7	53.1
Water Quality	108.4	<b>93.1</b>	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	<b>39.2</b>	546.7	45.1
MP5SPEC	84.5	94.6	221.2	888.3	91.5	557.0	6864.1	<b>49.3</b>	1132.1	58.4
MP6SPEC	90.3	93.4	212.6	871.0	89.1	557.6	6761.3	<b>47.2</b>	1227.1	58.5
ATP7d	<b>70.5</b>	262.6	452.1	2319.8	242.1	1779.2	24373.8	80.0	1897.4	136.5
OES97	<b>83.4</b>	485.3	1146.6	4928.9	499.8	5315.0	58072.1	148.2	3759.1	342.6
Osales	<b>92.0</b>	1094.8	2340.7	8322.2	986.5	11361.2	122265.3	437.0	4830.1	843.6
ATP1d	<b>70.7</b>	272.9	476.5	2568.9	261.9	2138.9	26768.9	95.0	2127.8	174.4
OES10	<b>90.0</b>	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	<b>61.4</b>	1073.2	117.4
Ranks	5.5	3.71	6.0	8.29	3.0	7.08	9.92	<b>1.88</b>	6.71	2.92

$$\begin{aligned}
\min_{\mathbf{w}, b, \xi} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_I \xi_I, & \max_{\alpha} \quad & \sum_I \alpha_I - \frac{1}{2} \sum_I \sum_{K \in I} \alpha_I \alpha_K Y_I Y_K \mathcal{K}(\mathbf{x}_{s_I}, \mathbf{x}_{s_K}) \\
\text{s.t.} \quad & Y_I (\langle \mathbf{w}, \mathbf{x}_{s_I} \rangle + b) \geq 1 - \xi_I, \forall I \in \{1, \dots, n\}, & \text{s.t.} \quad & \sum_I \alpha_I Y_I = 0, \\
& \xi_I \geq 0, \forall I \in \{1, \dots, n\}, & & 0 \leq \alpha_I \leq C, \forall I \in \{1, \dots, n\}, \\
& s_I = \operatorname{argmax}_{i \in I} (\langle \mathbf{w}, \mathbf{x}_i \rangle + b), \forall I \in \{1, \dots, n\}. & & s_I = \operatorname{argmax}_{i \in I} (\mathbf{o}_I), \forall I \in \{1, \dots, n\}.
\end{aligned}$$

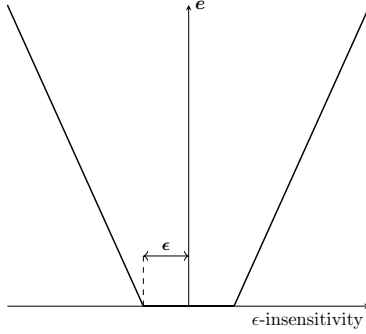


Figure 1: Vapnik's  $\epsilon$ -insensitivity loss function.

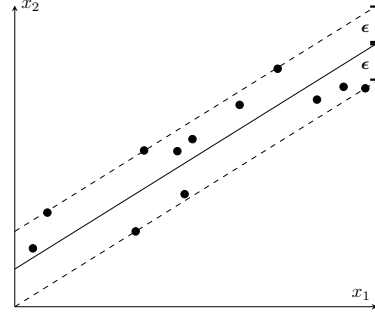
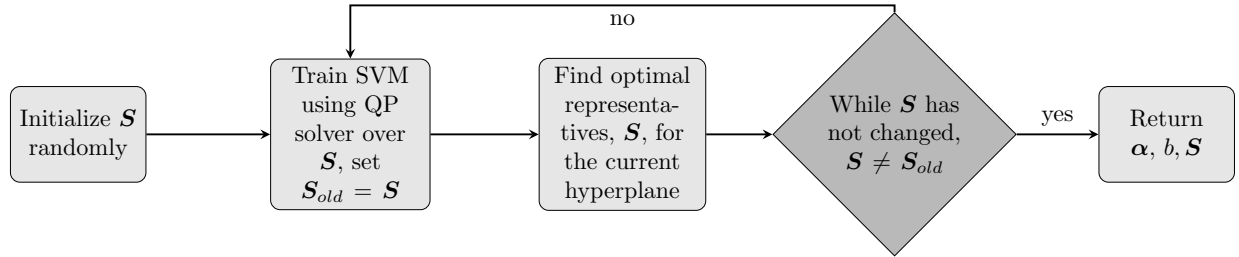


Figure 2: Linear support vector regression example solution on a toy 2D dataset.




---

#### Multi-Instance Representative SVM (MIRSVM)

---

**Input:** Training dataset  $\mathcal{D}$ , SVM Parameters  $C$  and  $\sigma$

**Output:** SVM model parameters  $\alpha$  and  $b$ , Bag Representative IDs  $\mathbf{S}$

```

1: for  $I \in \{1, \dots, n\}$  do
2:    $\mathbf{S}_I \leftarrow \text{rand}(|\mathcal{B}_I|, 1, 1)$  ▷ Assign each bag a random instance
3: end for
4: while  $\mathbf{S} \neq \mathbf{S}_{old}$  do
5:    $\mathbf{S}_{old} \leftarrow \mathbf{S}$ 
6:    $\mathbf{X}_S \leftarrow \mathbf{X}(\mathbf{S}), \mathbf{Y}_S \leftarrow \mathbf{Y}(\mathbf{S})$  ▷ Initialize the representative dataset
7:    $\mathbf{G} \leftarrow (\mathbf{Y}_S \times \mathbf{Y}_S) \cdot \mathcal{K}(\mathbf{X}_S, \mathbf{X}_S, \sigma)$  ▷ Build Gram matrix
8:    $\alpha \leftarrow \text{quadprog}(\mathbf{G}, -\mathbf{1}^n, \mathbf{Y}_S, \mathbf{0}^n, \mathbf{0}^n, C^n)$  ▷ Solve QP Problem
9:    $\mathbf{sv} \leftarrow \text{find}(0 < \alpha \leq C)$  ▷ Get the support vector indices
10:   $n_{sv} \leftarrow \text{count}(0 < \alpha \leq C)$  ▷ Get the number of support vectors
11:   $b \leftarrow \frac{1}{n_{sv}} \sum_{i=1}^{n_{sv}} (\mathbf{Y}_{sv} - \mathbf{G}_{sv} * (\alpha_{sv} \cdot \mathbf{Y}_{sv}))$  ▷ Calculate the bias term
12:  for  $I \in \{1, \dots, n\}$  do
13:     $\mathbf{G}_I \leftarrow (\mathbf{Y}_I \times \mathbf{Y}_S) \cdot \mathcal{K}(\mathcal{B}_I, \mathbf{X}_S, \sigma)$ 
14:     $\mathbf{S}_I \leftarrow \operatorname{argmax}_{i \in I} (\mathbf{G}_I * \alpha + b)$  ▷ Select optimal bag-representatives
15:  end for
16: end while

```

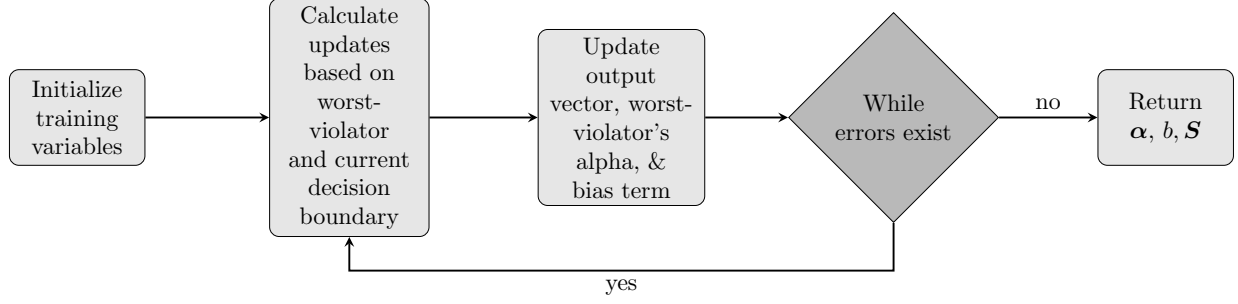
---

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

$$L(y_i, o_{(w,b)}(\mathbf{x}_i)) = \max \{0, 1 - y_i o_{(w,b)}(\mathbf{x}_i)\} \quad (2)$$

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

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




---

### OnLine Learning Algorithm using Worst-Violators (OLLAWV)

---

**Input:**  $\mathcal{D}, C, \gamma, \beta, M$

**Output:**  $\alpha, b, \mathcal{S}$

```

1:  $\alpha \leftarrow \mathbf{0}, b \leftarrow 0, \mathcal{S} \leftarrow \mathbf{0}$                                 ▷ Initialize OLLAWV model parameters
2:  $\mathbf{o} \leftarrow \mathbf{0}, t \leftarrow 0$                                           ▷ Initialize the output vector and iteration counter
3:  $wv \leftarrow 0, yo \leftarrow y_{wv} * \mathbf{o}_{wv}$                                 ▷ Initialize hinge loss error and worst-violator index
4: while  $yo < M$  do
5:    $t \leftarrow t + 1$ 
6:    $\eta \leftarrow 2/\sqrt{t}$                                                   ▷ Learning rate
7:
8:    $\Lambda \leftarrow \eta * C * y_{wv}$                                           ▷ Calculate hinge loss update
9:    $B \leftarrow (\Lambda * \beta) / n$                                           ▷ Calculate bias update
10:   $\mathbf{o} \leftarrow \mathbf{o} + \Lambda * \mathcal{K}(\mathbf{x}_{\neg \mathcal{S}}, \mathbf{x}_{wv}, \gamma) + B$           ▷ Update output vector
11:   $\alpha_{wv} \leftarrow \alpha_{wv} + \Lambda$                                     ▷ Update worst-violator's alpha value
12:   $b \leftarrow b + B$                                                         ▷ Update bias term
13:
14:   $\mathcal{S}_t \leftarrow wv$                                                     ▷ Save index of worst-violator
15:   $[yo, wv] \leftarrow \min_{wv \in \{\neg \mathcal{S}\}} \{y_{wv} \cdot \mathbf{o}_{wv}\}$           ▷ Find the worst-violator
16: end while
  
```

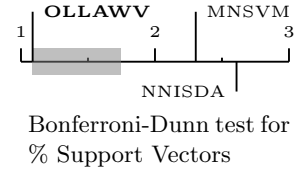
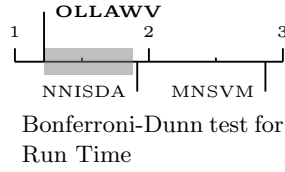
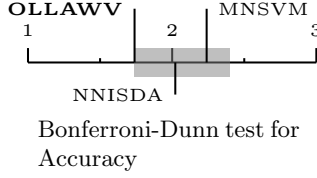
---

Classification Datasets

Dataset	# Samples	# Attributes	# Classes
<i><b>small datasets</b></i>			
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
<i><b>medium datasets</b></i>			
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
<i><b>large datasets</b></i>			
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

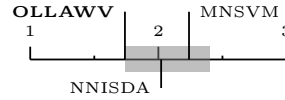
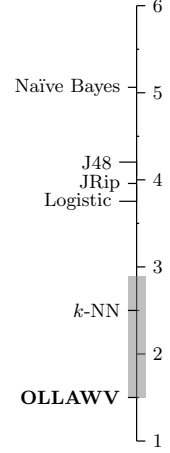
Comparison of OLLAWV vs. NNISDA and MNSVM

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

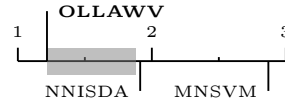


Accuracy (%) for Non-SVM Methods vs. OLLAWV

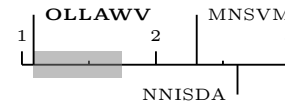
Dataset	OLLAWV	k-NN	J48	JRip	Naïve Bayes	Logistic
<i>small datasets</i>						
iris	<b>97.33 ± 1.49</b>	96.00 ± 3.65	94.00 ± 2.79	90.67 ± 4.35	96.00 ± 2.79	97.33 ± 2.79
teach	52.32 ± 3.46	<b>59.64 ± 2.89</b>	49.72 ± 7.58	56.75 ± 9.60	53.75 ± 6.46	51.77 ± 6.68
wine	<b>98.87 ± 1.54</b>	97.73 ± 3.72	90.43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



Accuracy (%)

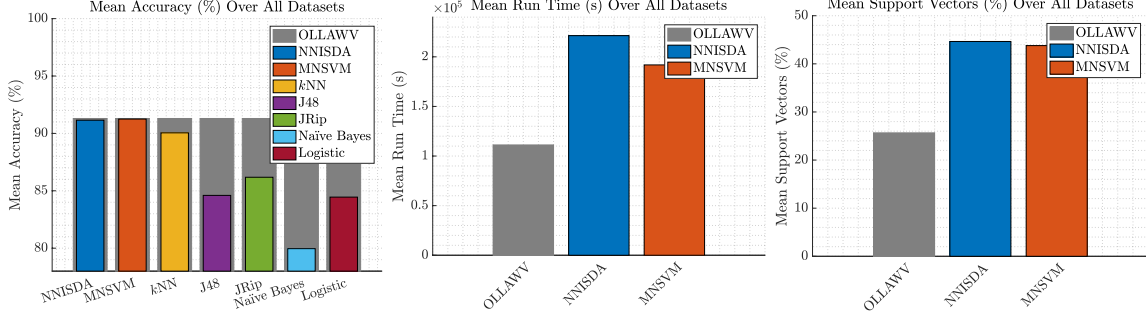


Run Time (s)



Support Vectors (%)





$$C \in \{4^n\}, \quad n = \{-2, \dots, 5\} \quad (5a)$$

$$\gamma \in \{4^n\}, \quad n = \{-5, \dots, 2\} \quad (5b)$$

#### Algorithm Hyperparameters

Algorithm	Parameters
SVM	Penalty: $C \in \{4^n\}, n = \{-2, \dots, 5\}$ RBF Kernel: $\gamma \in \{4^n\}, n = \{-5, \dots, 2\}$
$k$ -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: $\{1e^{-7}, 1e^{-8}, 1e^{-9}\}$

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

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

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

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

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

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

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

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

$$\Lambda \leftarrow Cy_i$$

$$\alpha_i \leftarrow \alpha_i + \eta \Lambda$$

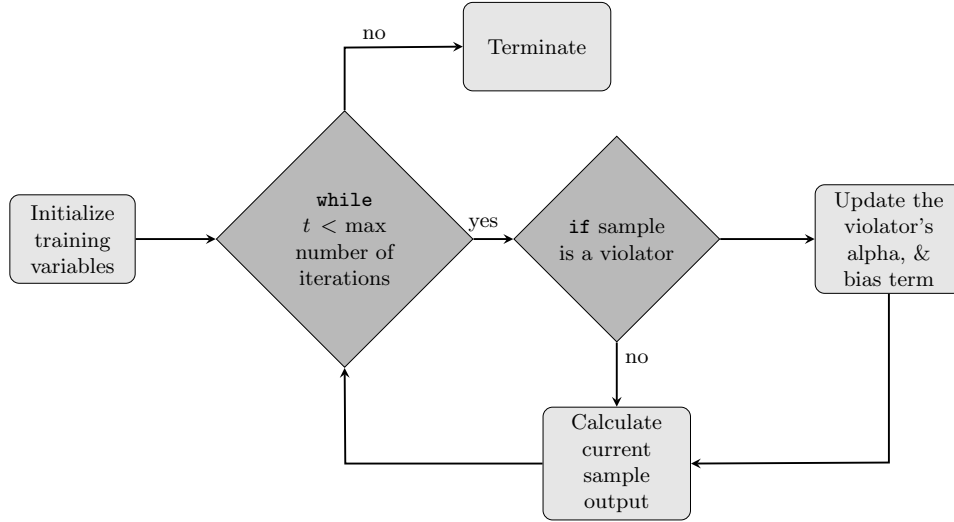
$$b \leftarrow b + \eta \Lambda$$

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

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

Comparison of OLLAWV vs. OLLA-L2

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




---

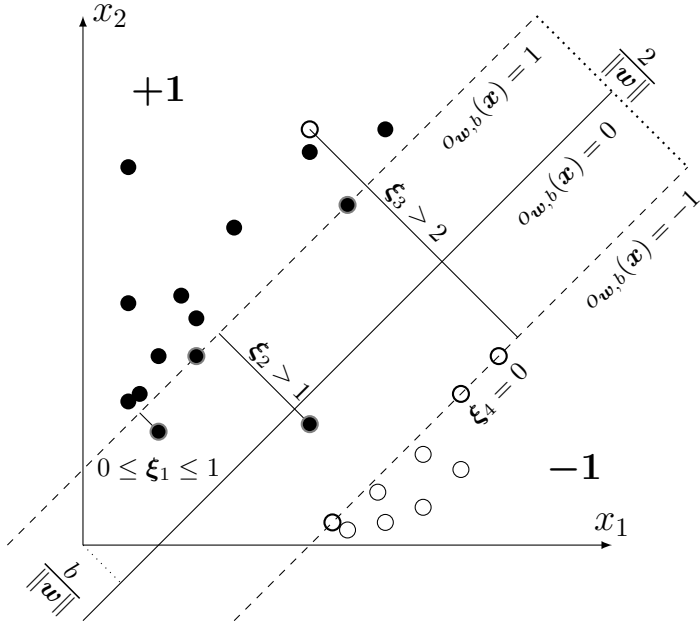
#### OnLine Learning Algorithm - List 2 (OLLA-L2)

---

**Input:**  $X, Y, \beta, n, e$

**Output:**  $\alpha, b, S$

- |  |   |
|--|---|
| <pre> 1: <math>\alpha \leftarrow \mathbf{0}, b \leftarrow 0, S \leftarrow \mathbf{0}, o \leftarrow \mathbf{0}, i \leftarrow 0</math> 2: <b>for</b> <math>t = 1, \dots, n * e</math> <b>do</b> 3:   <math>\eta \leftarrow 2/\sqrt{t}</math> 4:   <b>if</b> <math>y_i o_i \leq 1</math> <b>then</b> 5:     Calculate <math>\Lambda</math> and <math>P</math> 6:     <math>S \leftarrow [S \cup i]</math> 7:     <math>\alpha_i \leftarrow \alpha_i + (\Lambda - P)</math> 8:     <math>b \leftarrow b + (\Lambda - P)\beta</math> 9:   <b>end if</b> 10:  <math>i \leftarrow i + 1</math> 11:  <b>if</b> <math>i = n</math> <b>then</b> 12:    <math>i = 0</math> 13:  <b>end if</b> 14:  <math>o_i \leftarrow K(x_i, x_S) \alpha_S + b</math> 15: <b>end for</b> </pre> | <ul style="list-style-type: none"> <li>▷ Initialize model and algorithm parameters</li> <li>▷ Learning rate in function of time</li> <li>▷ Check if current sample is a violator             <ul style="list-style-type: none"> <li>▷ Calculate update parameters</li> <li>▷ Save index of current violator</li> <li>▷ Update violator's alpha value</li> <li>▷ Update bias term</li> </ul> </li> <li>▷ Get new sample</li> <li>▷ If the sample index exceeds the number of samples             <ul style="list-style-type: none"> <li>▷ Reset sample index</li> </ul> </li> <li>▷ Calculate the new sample's output value</li> </ul> |
|--|---|
-



---

OnLine Learning Algorithm - List 2 (OLLA-L2)

---

**Input:**  $\mathbf{X}, \mathbf{Y}, \beta, n, e$

**Output:**  $\alpha, b, \mathbf{S}$

```
1:  $\alpha \leftarrow \mathbf{0}, b \leftarrow 0, \mathbf{S} \leftarrow \mathbf{0}, \mathbf{o} \leftarrow \mathbf{0}, i \leftarrow 0$  ▷ Initialize model and algorithm parameters
2: for  $t = 1, \dots, n * e$  do
3:    $\eta \leftarrow 2/\sqrt{t}$  ▷ Learning rate in function of time
4:   if  $y_i o_i \leq 1$  then ▷ Check if current sample is a violator
5:     Calculate  $\Lambda$  and  $P$  ▷ Calculate update parameters
6:      $\mathbf{S} \leftarrow [\mathbf{S} \cup i]$  ▷ Save index of current violator
7:      $\alpha_i \leftarrow \alpha_i + (\Lambda - P)$  ▷ Update violator's alpha value
8:      $b \leftarrow b + (\Lambda - P)\beta$  ▷ Update bias term
9:   end if
10:   $i \leftarrow i + 1$  ▷ Get new sample
11:  if  $i = n$  then ▷ If the sample index exceeds the number of samples
12:     $i = 0$  ▷ Reset sample index
13:  end if
14:   $o_i \leftarrow \mathbf{K}(x_i, x_{\mathbf{S}}) \alpha_{\mathbf{S}} + b$  ▷ Calculate the new sample's output value
15: end for
```

---

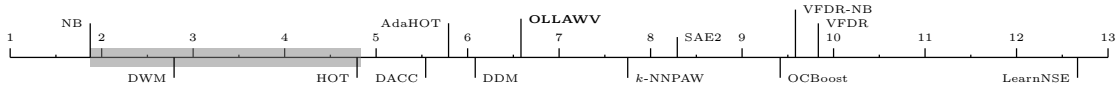
Accuracy (%) for Data Stream Classifiers

Dataset	OLLAWV	HOT	AdaHOT	NB	k-NNPAW	DDM	VFDR	VFDR-NB	SAE2	LearnNSE	DWM	DACC	OCBoost
CovType	<b>90.28</b>	85.34	86.22	60.04	87.89	59.56	60.32	75.58	76.21	69.97	71.96	61.70	71.29
Census	93.76	94.70	<b>94.74</b>	87.02	93.65	91.85	93.65	84.06	90.13	84.14	91.40	90.37	93.47
Shuttle	<b>99.67</b>	98.18	98.52	90.02	99.26	98.49	88.40	96.06	90.24	93.79	89.91	92.03	74.21
RBFNoDrift	<b>94.21</b>	92.94	92.96	71.99	93.75	92.48	77.53	81.71	89.16	70.28	70.43	65.01	92.08
LEDNoDrift	73.83	73.85	73.84	<b>73.94</b>	65.83	73.64	41.16	73.75	67.60	67.84	71.15	48.27	17.44
HyperplaneSlow	<b>90.09</b>	82.10	82.42	77.69	84.03	81.57	68.88	85.19	82.67	86.20	88.06	80.66	85.78
HyperplaneFaster	<b>89.51</b>	82.72	85.34	77.23	84.27	84.33	78.63	85.18	83.01	86.50	86.76	81.15	87.80
RBFGradualRecurring	98.41	94.63	94.44	58.33	<b>98.43</b>	93.47	60.08	86.16	88.48	72.87	74.96	61.91	49.62
RBFBlips	<b>99.07</b>	95.67	95.60	60.83	98.94	94.92	66.90	88.35	89.04	77.53	79.98	68.27	47.46
WaveformGenerator	83.94	82.99	<b>84.14</b>	80.41	80.13	83.61	63.88	75.84	80.18	80.19	78.39	73.59	55.05
STAGGERGeneratorF1	<b>100.0</b>	99.99	99.99	<b>100.0</b>	<b>100.0</b>	<b>100.0</b>	99.91	<b>100.0</b>	95.04	89.82	<b>100.0</b>	99.96	<b>100.0</b>
HyperplaneFaster02	<b>89.49</b>	82.77	85.36	77.25	84.27	87.64	78.89	85.11	83.04	86.51	86.76	81.23	87.59
RBFGradualRecurringv2	<b>97.18</b>	93.29	93.00	57.47	95.73	93.19	57.96	80.71	84.45	62.41	63.79	49.42	48.88
MixedGeneratorBT	98.00	99.11	<b>99.32</b>	91.93	97.67	99.11	83.12	93.28	93.16	90.91	91.19	89.16	98.98
MixedGeneratorBF	98.03	99.18	<b>99.36</b>	92.04	97.59	99.20	89.96	94.30	93.41	90.76	91.46	88.61	98.94
RandomRBFGeneratorC4A25	<b>99.12</b>	97.43	97.14	81.59	98.69	96.89	72.33	89.03	90.61	78.23	79.67	63.02	52.16
RandomRBFGeneratorC4A50	<b>99.74</b>	99.16	99.14	91.90	99.17	98.99	80.63	95.63	92.00	86.03	90.45	73.78	50.66
SineGeneratorF1BF	97.79	<b>99.75</b>	99.73	93.55	95.54	99.66	94.83	95.90	94.51	92.55	93.30	92.20	99.51
SineGeneratorF2BF	97.79	<b>99.75</b>	99.73	93.55	95.41	99.66	95.26	96.10	94.57	92.55	93.34	92.20	99.48
STAGGERGeneratorF1BF	<b>100.0</b>	99.99	99.99	<b>100.0</b>	<b>100.0</b>	<b>100.0</b>	99.91	<b>100.0</b>	95.04	89.82	<b>100.0</b>	99.96	<b>100.0</b>
STAGGERGeneratorF2BF	<b>100.0</b>	99.98	99.98	<b>100.0</b>	<b>100.0</b>	99.98	99.87	<b>100.0</b>	95.02	44.41	<b>100.0</b>	<b>100.0</b>	0.61
HyperplaneFasterAN5	<b>89.29</b>	82.69	85.24	87.38	84.19	88.74	79.14	84.89	82.92	86.41	86.67	81.12	87.38
SEASuddenAN0	87.80	84.92	85.18	88.23	87.22	<b>88.97</b>	81.56	85.17	85.11	85.77	86.93	83.73	88.23
SEASuddenAN05	87.53	84.57	84.82	87.53	86.96	<b>88.28</b>	81.55	85.11	84.57	85.63	86.54	83.53	87.53
Average	<b>93.94</b>	91.90	92.34	82.50	92.03	91.43	78.93	88.21	87.51	81.30	85.55	79.20	73.92
Rank	<b>2.52</b>	5.42	4.54	8.19	4.8542	4.63	10.96	6.77	8.46	9.04	7.27	10.90	7.46



Training Time (seconds) for Data Stream Classifiers

Dataset	OLLAWV	HOT	AdaHOT	NB	k-NNPAW	DDM	VFDR	VFDR-NB	SAE2	LearnNSE	DWM	DACC	OCBoost
CovType	0.0451	0.0765	0.0909	<b>0.0009</b>	0.0387	0.0577	0.0371	0.0445	0.1121	6.2806	0.0419	0.0357	0.0984
Census	0.0168	0.1040	0.1162	<b>0.0007</b>	0.0386	0.0345	0.0735	0.0458	0.0363	0.9368	0.0156	0.0175	0.0543
Shuttle	0.0103	0.0276	0.0298	<b>0.0004</b>	0.0386	0.0069	0.0108	0.0069	0.0217	0.2692	0.0104	0.0186	0.0581
RBFNoDrift	0.0329	0.0174	0.0247	<b>0.0002</b>	0.0383	0.0828	0.4267	0.3172	0.0491	3.4029	0.0104	0.0137	0.0402
LEDNoDrift	0.0697	0.0373	0.0649	<b>0.0003</b>	0.0389	0.0245	0.0097	0.0082	0.0694	5.4734	0.0175	0.0290	0.0547
HyperplaneSlow	0.0353	0.0094	0.0142	<b>0.0002</b>	0.0389	0.1405	0.1871	0.3006	0.0473	3.3734	0.0091	0.0142	0.0389
HyperplaneFaster	0.0263	0.0295	0.0416	<b>0.0002</b>	0.0384	0.1231	0.3016	0.3784	0.0424	3.2991	0.0098	0.0143	0.0376
RBFGradualRecurring	0.0456	0.0316	0.0369	<b>0.0003</b>	0.0389	0.0391	1.1284	0.9295	0.1004	11.7980	0.0335	0.0474	0.1070
RBFBlips	0.0396	0.0243	0.0299	<b>0.0003</b>	0.0384	0.0278	0.7587	0.8342	0.0948	11.9934	0.0342	0.0473	0.1050
WaveformGenerator	0.0394	0.0751	0.0965	<b>0.0006</b>	0.0390	0.1847	0.9670	0.9271	0.1715	17.9675	0.0511	0.0743	0.1318
STAGGERGeneratorF1	0.0021	0.0006	0.0007	<b>0.0001</b>	0.0389	0.0005	0.0013	0.0007	0.0018	0.5517	0.0003	0.0020	0.0113
HyperplaneFaster02	0.0268	0.0302	0.0465	<b>0.0002</b>	0.0388	0.0260	0.3192	0.3010	0.0456	3.2893	0.0095	0.0141	0.0372
RBFGradualRecurringv2	0.0545	0.0169	0.0203	<b>0.0003</b>	0.0383	0.0859	0.7622	0.7634	0.0552	12.0529	0.0367	0.0464	0.1053
MixedGeneratorBT	0.0205	0.0027	0.0026	<b>0.0001</b>	0.0336	0.0071	1.7640	1.5534	0.0086	0.9761	0.0024	0.0052	0.0204
MixedGeneratorBF	0.0299	0.0024	0.0024	<b>0.0001</b>	0.0334	0.0065	1.1035	1.0924	0.0093	0.8971	0.0024	0.0058	0.0209
RandomRBFGeneratorC4A25	0.0200	0.0441	0.0407	<b>0.0003</b>	0.0358	0.0730	1.7831	1.6927	0.1245	13.9876	0.0395	0.0719	0.1580
RandomRBFGeneratorC4A50	0.0157	0.0333	0.0337	<b>0.0007</b>	0.0350	0.0909	2.8802	2.5559	0.1917	27.8456	0.0752	0.1432	0.2974
SineGeneratorF1BF	0.0122	0.0056	0.0056	<b>0.0001</b>	0.0334	0.0073	1.9301	2.9169	0.0127	1.3768	0.0034	0.0071	0.0235
SineGeneratorF2BF	0.0121	0.0051	0.0053	<b>0.0001</b>	0.0314	0.0077	5.0598	5.9477	0.0117	1.3996	0.0027	0.0074	0.0236
STAGGERGeneratorF1BF	0.0021	0.0006	0.0006	<b>0.0001</b>	0.0323	0.0005	0.0006	0.0005	0.0015	0.6175	0.0003	0.0022	0.0143
STAGGERGeneratorF2BF	0.0022	0.0010	0.0009	<b>0.0001</b>	0.0340	0.0006	0.0005	0.0005	0.0017	0.6752	0.0003	0.0020	0.0139
HyperplaneFasterAN5	0.0264	0.0200	0.0200	0.0440	0.0269	0.0115	0.2298	0.2417	0.0312	2.5014	<b>0.0080</b>	0.0147	0.0740
SEASuddenAN0	0.0208	0.0051	0.0052	0.0268	0.0312	0.0099	0.0370	0.0373	0.0176	1.3545	<b>0.0032</b>	0.0059	0.0335
SEASuddenAN05	0.0284	0.0055	0.0055	0.0257	0.0309	0.0095	0.0321	0.0334	0.0167	1.3365	<b>0.0032</b>	0.0061	0.0326
Average	0.0264	0.0252	0.0307	<b>0.0043</b>	0.0359	0.0441	0.8252	0.8721	0.0531	5.5690	0.0175	0.0269	0.0663
Rank	6.5833	4.7917	5.7917	<b>1.8750</b>	7.7500	6.0833	9.8333	9.5833	8.2917	12.6667	2.7917	5.5417	9.4167



Algorithms			
Algorithm Description			
HOT	Hoeffding Option Tree		
AdaHOT	Adaptive Hoeffding Option Tree		
NB	Naïve Bayes		
$k$ -NNPAW	$k$ -NN with Probabilistic Adaptive Windows		
DDM	Drift Detection Method with HOT		
VFDR	Very Fast Decision Rules		
VFDR-NB	VFDR with Naïve Bayes		
SAE2	Social Adaptive Ensemble 2		
Learn.NSE	Learn++ for Non-Stationary Environments		
DWM	Dynamic Weighted Majority		
DACC	Dynamic Adaptation to Concept Changes		
OCBoost	Online Coordinate Boosting		

Base Streamed Datasets & Generators			
Dataset	# Samples	# Attributes	# Classes
<b><i>Static</i></b>			
Shuttle	57,999	10	7
Census	299,284	42	2
CovType	581,012	55	7
<b><i>Generators</i></b>			
RandomRBFGenerator	1,000,000	10	2
LEDGenerator	1,000,000	2	10
HyperplaneGenerator	1,000,000	10	2
WaveformGenerator	1,000,000	40	3
STAGGERGenerator	1,000,000	3	2
MixedGenerator	1,000,000	4	2
SineGenerator	1,000,000	2	2
SEAGenerator	1,000,000	2	2



