Novos métodos nebulosos para seleção de variáveis preditivas ou classificativas

Sumário

Lista de Anotações

1: **PlatEMO** e outros solvers
2: Hiperparâmetros
3: Métrica APE
4: Selecionar n variáveis e quais
5: **PLS VIP** online (todos)
6: Quantos θ's — com loop já era
7: Idle — fator de esquecimento
8: Zero variáveis
9: Métrica de overfitting
10: **Python** FBeM e eFGP, online
11: **Comparar** scores, rankings
12: **Correlações** entrada $\times$ entrada
13: Séries temporais
14: **Fórmulas Yamakawa** Batelada
15: Stepwise — formulação
16: Probabilidade
17: **Fórmulas Pearson** Batelada
18: Implementar **GA** = multiobjetivo
19: Full scan ou já era, ou evolucionário
20: Agrupar erros
21: Normalizar contribuições $\times$ constSlope
??: Ordenar para minimizar
22: Incerteza
23: Fuzzy
24: Adaptar **contribuições** $\times$ regressores
25: xit, xft Yamakawa
26: fwd e bwd já era. Big data.
27: Procure por ***
28: nVariáveis arbitrário
29: 200 variáveis $\neq$ classificação
30: Resultados por tipo: Vídeos

1. Resumo

Começamos pelo NFN Yamakawa e os problemas sugeridos na disciplina Sistemas Nebulosos.
constNFuncPertinencia = 10;
constMaxEpocas = 20;
Os problemas escolhidos para testar foram: mgdata, dynamic, Death Valley e o problema da seção 7.
O mgdata.dat é um dos problemas Mackey-Glass.
Utilizamos os dados disponíveis para download no site de D. Leite
sites.google.com/site/danfl. Modificamos os fontes ali disponíveis para contribuir no estado da arte em seleção de variáveis.

O artigo [9] utilizou apenas o método stepwise.
O artigo [10] utilizou stepwise, forward, backward e propôs o método rβ.AIC.
O artigo [11] tem várias metodologias, mais uma proposta (PLS.importância). Futuras comparações incluem 5.13. The GA and SR-GA methods.
No problema Dynamic, ao longo de várias execuções, os números aleatórios devem ser sempre os mesmos.
O APE foi retirado por oscilar muito, trocamos pelo RMS de validação.
Implementamos o nfn-afs com o nome de Finc (teste F via inclusão de uma variável).
A função anfis_yamakawa passou a reter o erro mínimo dentre as épocas.

Nota 1. De forma multiobjetivo, evolucionária, população.
Sabemos por [SPECTRAL] que PCA e SPEC estão agrupados pelo J. O mesmo para os SLEPs.
Temos o surrogate. E os solvers do Python? E online?
*** Se é otimização, eu defino todos os problemas e os solvers.
No PlatEMO, havia inteiros. Mas é evolucionário.
Há os determinísticos. E o SQP $\times$ gradiente?

Nota 2. automatizar números mágicos

Há um randperm (treino, validação, "seleção de linhas") nas tuplas do deathvalley. (Também é o mesmo em todas as execuções.) Isso faz com que cada teste seja de uma forma distinta.

Nota 3. Métricas: erros de treinamento, APE $= y$;

Nota 4. selecionar n variáveis e quais;
o mais certo é começar de 6 variáveis (aleatórias? todas?
nchoosek$(12,6) == 924$) e ir pra lá $= 12$ e pra cá $= 1$.

Nota 5. Implementar algoritmo PLS VIP online. Vide equação 1.

Nota 6. No momento, temos as opções de todos os θ's, único θ e um máximo, retirando RMSEs mínimos.

Nota 7. Está alta a quantidade de θ's. A ideia é armazenar $N(\ell_k)$, quantas são as amostras, para o AIC; e também idle(ℓ_k) que permitiria excluir os regressores não utilizados.

Nota 8. Testar problema yConstante: Em todos os métodos, o número de variáveis pode ser reduzido a zero. Que tal concatenar uma coluna constante e igual a 1?

Nota 9. métrica de overfitting$(+)$ e underfitting$(-)$: Como se mede um overfitting? Qual é a estatística que a gente olha para verificar se o segundo grau é melhor que o primeiro? Se o terceiro é melhor que o segundo...?

Nota 10. Traduzir para Python. E quando a fonte é o youtube? PCA; perda de informação; selectkbest, selectpercentile, fregression, fclassif; métrica MAE; randomforestregression; normalizar; dados desbalanceados.

1.1 Precedência — De Algoritmo para Função Matemática

Seja $f : A \to B$; $\#A = 2^n$; $N = \{1, \cdots, n\}$; $A = P(N)$.
O algoritmo dá pesos (defina função peso) às seleções: subconjuntos de N.
Dentre A, escolher qual é a melhor (defina melhor).
Sejam a,$b \subset N$; $a \prec b \Leftrightarrow a$ é melhor que b.
Adicionar é uma operação matemática.
$S_a = \{x_i\}$
$S_c = S_a \cup \{x_c\}$
$S_{c_1} \prec S_{c_2} \succ S_{c_3}$
Excluir é uma operação matemática.
$S_0 = \{x_i\}$
$S_e = S_0 - \{x_e\}$
$S_{e_1} \prec S_{e_2} \succ S_{e_3}$
Parar no slope é uma operação matemática.
Pare quando for suficientemente boa. (Defina a flag boa.)

Além disso, temos as opções de otimizar alguma função objetivo com ou sem transformada. Esse é o caso da seção 2.13.

Nota 11. O PLS VIP está agrupado no assunto scores, rankings a comparar.

1.2 Quadrado

Segundo grau: elipsoides, paraboloides, hiperboloides.
Exemplo: XOR. Acrescente a coluna xy, porque x^2 e y^2 são os mesmos.

```
x y xy (x + y - 2xy)
0 0 0        0
0 1 0        1
1 0 0        1
1 1 1        0
```

Logo, os pesos são $w = (1, 1, -2)$.

1.2.1 Correlação

Seja $\tau \in [-20,20]$. Calcular a correlação entre (x_i, y). Se, em $-15 \leq \tau \leq 15$, tudo estiver dentro da faixa de confiança, retirar x_i.

Por exemplo, no problema Dynamic, $y(k) = x_4(k) = x_1(k+1) = x_5(k+1) = x_2(k+2) = x_3(k+3)$. Entendemos entrada $\times$ saída, mas não entrada $\times$ entrada. Em geral, todas as entradas estarão correlacionadas entre si.

Nota 12. Correlações entre entradas: combinação duas a duas.

Nota 13. Séries temporais.

1.2.2 Dependência linear

Pivoteando a original, sabemos se o sistema é indeterminado.
Pivoteando a transposta, sabemos se as colunas são dependentes.

2. Cada Algoritmo — Batelada

2.1 (0) Contribuição Yamakawa

2.1.1 Explicação

Módulo da contribuição percentual (pesos w) maior que:
constContribGlobal $= 0.70$; // dividido por nVariaveis; $0.6 \leq$ usual ≤ 0.8

Nota 14. Fórmulas em batelada.

2.1.2 Algoritmo

```
0.1) Comece por todas as variáveis.
     Treine.
     Queremos min(contribuição(w)) > constante.
     Enquanto isso for falso, retire a que menos contribui.

0.2) Queremos theta/max(theta) > constante.
     Retire todas as outras colunas, uma vez só (sem loop).
```

2.1.3 Formulação

0.1) $S_0 = N$;
for $i = 0, \cdots, n$:
Peso $c(S_i) =$ contribuição; $c_i < c_j \Leftrightarrow S_i \prec S_j$;
$[c_e, e] = \min\{c\}$;
exit when flagBoa $= \left(c_e > k = \dfrac{0.70}{n} \right)$;
$S_{i+1} = S_i - \{x_e\}$;
end;
return S_i.

. . . . . .

0.2) $S_0 = N$;
Valor $v(n) = \dfrac{|\theta_n|}{\max\{|\theta_j|\}}$;
$x_e \in S_e \Leftrightarrow v(e) < k_\theta = 0.1$;
$x_f \in S_f \Leftrightarrow v(f) > k_\theta$;
return $S_f = S_0 - S_e$.

2.1.4 Resultado

2.2 (1) Stepwise

2.2.1 Explicação

A função já vem implementada: stepwiselm(xt,ydt);
Temos stepwise em Python.

Nota 15. formulação matemática

2.2.2 Algoritmo

```
1) ? stepwise em andamento
```

2.2.3 Formulação

(1, 2) utilizam \R2020a\toolbox\stats\stats\sequentialfs.m que foi modificado para sequentialFeatureSelection.

A fonte é o File Exchange da Mathworks.

2.2.4 Resultado

2.3 (2) Forward

2.3.1 Explicação

Vide nota 26.

Dividimos em 10 e 90%; máximo de iterações = nVariaveis; função de perda; dividimos pelo máximo;

constFwd = 0.75; // 75% do máximo

2.3.2 Algoritmo

```
2) Ordene asc, adicionando pela função loss;
   Nesta ordenação, procure quando é que loss pela primeira vez é menor que
   constante = 75% do máximo.
```

2.3.3 Formulação

Parte 1: $S_0 = \emptyset$;

Peso $p(S_i) = \text{loss}(i)$; $p_i < p_j \Leftrightarrow S_i \prec S_j$;

$[p_c, c] = \min\{p\}$;

$S_{i+1} = S_i \cup \{x_c\}$;

Parte 2: Mesmo peso.

$i = 1$;

for $i = 2, \cdots, n$:

exit when flagBoa = $(p_i \geq k = 0.75 \max\{p_j\})$;

end;

return S_i.

2.3.4 Resultado

2.4 (3) Backward

2.4.1 Explicação

Tudo idêntico ao anterior, exceto que o critério de parada não é pelo máximo, é 100% do mínimo.

2.4.2 Algoritmo

```
3) Ordene desc, retirando pela função loss;
   Nesta ordenação, minimize loss.
```

2.4.3 Formulação

Parte 1: $S_N = N$;

Peso $p(S_i) = \text{loss}(i)$; $p_i < p_j \Leftrightarrow S_i \prec S_j$;

$[p_e, e] = \min\{p\}$;

$S_{i-1} = S_i - \{x_e\}$;

Parte 2: Mesmo peso.

$[p_f, f] = \min\{p(S_i)\}$;
return S_f.

. . . . . .

4) Parte 1: $S_0 = \emptyset$;
Peso $p(S_i) = \text{AIC}(\xi_i)$; $p_i < p_j \Leftrightarrow S_i \prec S_j$;
$[p_c, c] = \min\{p\}$;
$S_{i+1} = S_i \cup \{x_c\}$;
Parte 2: Peso $q(S_i) = p(S_i) - p(S_{i-1})$; $q_i < k_{slope} = -0.1 \Leftrightarrow S_i \prec S_{i-1}$;
$i = 1$;
for $i = 2, \cdots, n$:
exit when flagBoa $= (q_i \geq k)$;
end;
return S_i.

2.4.4 Resultado

2.5 (4) AIC

2.5.1 Explicação

Se temos [1,3,5], os candidatos são 2,4,6$\cdots$12. Calcular o AIC de todos os candidatos e obter o mínimo = final. Monitoramos enquanto diminui, até a hora em que $\frac{\Delta y}{\Delta x} > \text{constSlope} = -0.1$. De vez em quando é negativo. Módulo aqui é pior.

2.5.2 Algoritmo

```
4) Ordene asc, adicionando AIC mínimo;
   Nesta ordenação, procure o AIC de baixa inclinação negativa.
```

2.5.3 Formulação

2.5.4 Resultado

2.6 (5) Finc

2.6.1 Explicação

Tudo idêntico ao anterior, exceto que, ao invés de procurar o vértice, tomamos o mínimo dentre os Finc finais.

Cálculo de RSSa, RSSc, sa, sc, pa, pc. Estava sempre negativo, por isso utilizamos o módulo mínimo.

2.6.2 Algoritmo

```
5) Ordene asc, adicionando Finc mínimo;
   Nesta ordenação, minimize Finc.
```

2.6.3 Formulação

Parte 1: $S_0 = \emptyset$;
Peso $p(S_i) = F_{inc}(i)$; $p_i < p_j \Leftrightarrow S_i \prec S_j$;
$[p_c, c] = \min\{p\}$;
$S_{i+1} = S_i \cup \{x_c\}$;
Parte 2: Mesmo peso.
$[p_f, f] = \min\{p(S_i)\}$;
return S_f.

2.6.4 Resultado

2.7 (6) RMSE

2.7.1 Explicação

Se temos [1,3,5], os candidatos são 2,4,6$\cdots$12. Calcular o RMSE definido em [11] de todos os candidatos e obter o mínimo = final. Monitoramos enquanto diminui, até a hora em que $\frac{\Delta y}{\Delta x} >$ constSlope $= -0.1$.

2.7.2 Algoritmo

```
6) Ordene asc, adicionando RMSE mínimo;
   Nesta ordenação, procure o RMSE de baixa inclinação negativa.
```

2.7.3 Formulação

Parte 1: $S_0 = \emptyset$;
  Peso $p(S_i) = \sqrt{\xi^\top \xi}$; $p_i < p_j \Leftrightarrow S_i \prec S_j$;
  $[p_c, c] = \min\{p\}$;
  $S_{i+1} = S_i \cup \{x_c\}$;
Parte 2: Peso $q(S_i) = p(S_i) - p(S_{i-1})$; $q_i < k_{slope} = -0.1 \Leftrightarrow S_i \prec S_{i-1}$;
  $i = 1$;
  for $i = 2, \cdots, n$:
    exit when flagBoa $= (q_i \geq k)$;
  end;
return S_i.

2.7.4 Resultado

2.8 (7) Fexc

2.8.1 Explicação

Os candidatos são a retirada de cada elemento. Calcule o Fexc de todos os candidatos e obtenha o máximo = final.

O artigo diz para calcular o F e o pValue. Este não achamos. O F só ficou aceitável ao dividir pelo maior: o atual, ao invés do candidato. $(c = a - 1)$.

Nota 16. Qual é o algoritmo com melhores probabilidades (de que)?

Tomamos o máximo dentre os Fexc finais.

2.8.2 Algoritmo

```
7) Ordene desc, retirando Fexc máximo;
   Nesta ordenação, maximize Fexc.
```

2.8.3 Formulação

Parte 1: $S_N = N$;
  Peso $p(S_i) = F_{exc}(i)$; $p_i > p_j \Leftrightarrow S_i \prec S_j$;
  $[p_e, e] = \max\{p\}$;
  $S_{i-1} = S_i - \{x_e\}$;
Parte 2: Mesmo peso.
  $[p_f, f] = \max\{p(S_i)\}$;
return S_f.

2.8.4 Resultado

2.9 (8) FexcLoss

2.9.1 Explicação

A função loss precisa de treinamento e validação. Dentre todos os candidatos a deletar, tomamos o loss mínimo. Quando todas as diferenças forem ≥ 0, retornar.

Ficou aceitável com $50/50 =$ treino/validação, ao contrário dos $90/10$ do forward e backward.

2.9.2 Algoritmo

```
8) Ordene desc, retirando pela função loss mínima;
   Nesta ordenação, procure quando é que, pela primeira vez,
   Delta loss >= 0 = constante.
```

2.9.3 Formulação

$g(S_i) = \text{loss}(i)$;
Peso $p(S_i) = g(S_i) - g(S_{i-1})$; $p_i < k = 0 \Leftrightarrow S_i \prec S_{i-1}$;
$i = 1$;
for $i = 2, \cdots, n$:
    exit when flagBoa $= (p_i \geq k)$;
end;
return S_i.

2.9.4 Resultado

2.10 (9) rβ.AIC

2.10.1 Explicação

Retirar variáveis pelo $\theta =$ vetor de regressores de módulo máximo. Ao final, procurar vértice do AIC.

2.10.2 Algoritmo

```
9) Ordene desc, retirando theta mínimo;
   Nesta ordenação, procure o AIC de baixa inclinação negativa.
```

2.10.3 Formulação

Parte 1: $S_N = N$;
Peso $p(S_i) = |\theta_i|$; $p_i < p_j \Leftrightarrow S_i \prec S_j$;
    $[p_e, e] = \min\{p\}$;
    $S_{i-1} = S_i - \{x_e\}$;
Parte 2: $g(S_i) = \text{AIC}(\xi_i)$;
    Peso $q(S_i) = g(S_i) - g(S_{i-1})$; $q_i < k_{slope} = -0.1 \Leftrightarrow S_i \prec S_{i-1}$;
    $i = 1$;
    for $i = 2, \cdots, n$:
        exit when flagBoa $= (q_i \geq k)$;
    end;
return S_i.

2.10.4 Resultado

2.11 (10) rβ.Finc

2.11.1 Explicação

Retirar variáveis como no anterior. Ao final, Finc mínimo.

2.11.2 Algoritmo

```
10) Ordene desc, retirando theta mínimo;
    Nesta ordenação, minimize J = RSSa/RSSc
```

2.11.3 Formulação

Parte 1: $S_0 = N$;
Peso $p(S_i) = |\theta_i|$; $p_i < p_j \Leftrightarrow S_i \prec S_j$;
$[p_e, e] = \min\{p\}$;
$S_{i+1} = S_i - \{x_e\}$;
Parte 2: Peso $q(S_n) = F_{inc}(n)$; $q_i < q_j \Leftrightarrow S_i \prec S_j$;
$[q_f, f] = \min\{q\}$;
return S_f.

2.11.4 Resultado

2.12 (11) rβ.RMSE

2.12.1 Explicação

Retirar variáveis pelo θ = vetor de regressores de módulo máximo. Ao final, procurar vértice do RMSE.

2.12.2 Algoritmo

```
11) Ordene desc, retirando theta mínimo;
    Nesta ordenação, procure o RMSE de baixa inclinação negativa.
```

2.12.3 Formulação

Parte 1: $S_N = N$;
Peso $p(S_i) = |\theta_i|$; $p_i < p_j \Leftrightarrow S_i \prec S_j$;
$[p_e, e] = \min\{p\}$;
$S_{i-1} = S_i - \{x_e\}$;
Parte 2: $g(S_i) = \sqrt{\xi^\top \xi}$;
Peso $q(S_i) = g(S_i) - g(S_{i-1})$; $q_i < k_{slope} = -0.1 \Leftrightarrow S_i \prec S_{i-1}$;
$i = 1$;
for $i = 2, \cdots, n$:
exit when flagBoa $= (q_i \geq k)$;
end;
return S_i.

2.12.4 Resultado

2.13 (12) PLS.VIP

2.13.1 Explicação

Retire as colunas cujo vip ≤ 1 = constante, uma vez só (sem loop). Vide equação 1.

2.13.2 Algoritmo

```
12) Queremos vip > constante.
    Retire todas as outras colunas, uma vez só (sem loop).
```

2.13.3 Formulação

$S_0 = N$;
Valor $v(n) = \text{vip}(n)$;
$x_e \in S_e \Leftrightarrow v(e) \leq k = 1$;
$x_f \in S_f \Leftrightarrow v(f) > k$;
return $S_f = S_0 - S_e$.

2.13.4 Resultado

2.14 (13) PLS.β

2.14.1 Explicação

Retire as colunas cujo $\dfrac{|\theta|}{\max|\theta|} <$ constante, uma vez só (sem loop).

2.14.2 Algoritmo

```
13) Queremos beta/max(beta) > constante.
    Retire todas as outras colunas, uma vez só (sem loop).
```

2.14.3 Formulação

$S_0 = N$;
Valor $v(n) = \dfrac{|\theta_n|}{\max\{|\theta_j|\}}$;
$x_e \in S_e \Leftrightarrow v(e) < k_\theta = 0.1$;
$x_f \in S_f \Leftrightarrow v(f) > k_\theta$;
return $S_f = S_0 - S_e$.

2.14.4 Resultado

2.15 (14) PLS.AIC

2.15.1 Explicação

Ordene desc, retirando v_w mínimo.
Nesta ordenação, procure o AIC de baixa inclinação negativa.

2.15.2 Algoritmo

```
14) Ordene desc, retirando v_w mínimo.
    Nesta ordenação, procure o AIC de baixa inclinação negativa.
```

2.15.3 Formulação

2.15.4 Resultado

2.16 (15) PLS.Finc

2.16.1 Explicação

Ordene desc, retirando v_w mínimo.
Nesta ordenação, minimize Finc.

2.16.2 Algoritmo

```
15) Ordene desc, retirando v_w mínimo.
    Nesta ordenação, minimize Finc.
```

2.16.3 Formulação

2.16.4 Resultado

2.17 (16) PLS.RMSE

2.17.1 Explicação

Ordene desc, retirando v_w mínimo.
Nesta ordenação, procure o RMSE de baixa inclinação negativa.

2.17.2 Algoritmo

```
16) Ordene desc, retirando v_w mínimo.
    Nesta ordenação, procure o RMSE de baixa inclinação negativa.
```

2.17.3 Formulação

2.17.4 Resultado

2.18 (17) Triangular.AIC

2.18.1 Explicação

Dada a matriz $\Psi = [x, y]$, esta rotina acha Q de forma que $Q^\top \Psi = V$ é triangular superior.

err é um vetor de valores que contêm as taxas de redução de erro de cada um dos regressores escolhidos. Vide nota **??**.

for $j = 1, \cdots, n$:
  Seja $J = \{j, \cdots, end\}$.
  for k = $j, \cdots, n$:
    $$\text{err}(k) = \max \left\{ c = \frac{(A_{J;k}^\top y_J)^2}{A_{J;k}^\top A_{J;k} y^\top y} \right\};$$
  end;
  $t = A_{all;j_m}$; // column of regressor with greatest err
  $A_{all;j_m} = A_{all;j}$;
  $A_{all;j} = t$;
  $x = [A]_{J;j}$;
  $n_x = \text{length}(x)$;
  $u = \|x\|$;
  $v = x$;
  if $u \neq 0$
    $b = x_1 + \text{sgn}(x_1) \cdot u$;
    $v_1 = 1$;
    $$v_{2,\cdots,n_x} = \frac{v_{2,\cdots,n_x}}{b}$$
  end
  $a = [A]_{J;J}$;
  $$b = -\frac{2}{v^\top v};$$
  $w = ba^\top v$;
  $a = a + vw^\top$;
  $[A]_{J;J} = a$;
end;

Substituímos a ordenação do err pela do AIC.

2.18.2 Algoritmo

```
17) Ordene desc, pelo err;
    Nesta ordenação, procure o AIC de baixa inclinação negativa.
```

2.18.3 Formulação

Parte 1: $S_N = N$;

Peso $p(S_k) = \text{err}(k) = \max\left\{c = \dfrac{(A_{J;k}^{\top} y_J)^2}{A_{J;k}^{\top} A_{J;k} y^{\top} y}\right\}$; $p_i > p_j \Leftrightarrow S_i \prec S_j$;

$[p_e, e] = \max\{p\}$;

$S_{i-1} = S_i - \{x_e\}$;

Parte 2: $g(S_i) = \text{AIC}(\xi_i)$;

Peso $q(S_i) = g(S_i) - g(S_{i-1})$; $q_i < k_{slope} = -0.1 \Leftrightarrow S_i \prec S_{i-1}$;

$i = 1$;

for $i = 2, \cdots, n$:

exit when flagBoa $= (q_i \geq k)$;

end;

return S_i.

2.18.4 Resultado

2.19 (18) Triangular.Finc

2.19.1 Explicação

O mesmo que i anterior, mas substituímos o AIC pelo Finc. Vide nota **??**.

2.19.2 Algoritmo

```
18) Ordene desc, pelo err;
    Nesta ordenação, minimize Finc.
```

2.19.3 Formulação

Parte 1: $S_N = N$;

Peso $p(S_k) = \text{err}(k) = \max\left\{c = \dfrac{(A_{J;k}^{\top} y_J)^2}{A_{J;k}^{\top} A_{J;k} y^{\top} y}\right\}$; $p_i > p_j \Leftrightarrow S_i \prec S_j$;

$[p_e, e] = \max\{p\}$;

$S_{i-1} = S_i - \{x_e\}$;

Parte 2: Peso $q(S_n) = F_{inc}(n)$; $q_i < q_j \Leftrightarrow S_i \prec S_j$;

$[q_f, f] = \min\{q\}$;

return S_f.

2.19.4 Resultado

2.20 (19) Triangular.RMSE

2.20.1 Explicação

O mesmo que o anterior, mas substituímos o Finc pelo RMSE. Vide nota **??**.

2.20.2 Algoritmo

```
19) Ordene desc, pelo err;
    Nesta ordenação, procure o RMSE de baixa inclinação negativa.
```

2.20.3 Formulação

Parte 1: $S_N = N$;

Peso $p(S_k) = \text{err}(k) = \max\left\{ c = \dfrac{(A_{J;k}^\top y_J)^2}{A_{J;k}^\top A_{J;k} y^\top y} \right\}$; $p_i > p_j \Leftrightarrow S_i \prec S_j$;

$[p_e, e] = \max\{p\}$;

$S_{i-1} = S_i - \{x_e\}$;

Parte 2: $g(S_i) = \sqrt{\xi^\top \xi}$;

Peso $q(S_i) = g(S_i) - g(S_{i-1})$; $q_i < k_{slope} = -0.1 \Leftrightarrow S_i \prec S_{i-1}$;

$i = 1$;

for $i = 2, \cdots, n$:

exit when flagBoa $= (q_i \geq k)$;

end;

return S_i.

2.20.4 Resultado

2.21 (20) Correlações de Pearson

2.21.1 Explicação

Nota 17. Fórmulas em batelada.

2.21.2 Algoritmo

```
?
```

2.21.3 Formulação

2.21.4 Resultado

2.22 GA

2.22.1 Explicação

Nota 18. implementar

Vide seção 6.6.

2.22.2 Algoritmo

```
?
```

2.22.3 Formulação

2.22.4 Resultado

2.23 (21) Lasso — MatLab

2.23.1 Explicação

Least absolute shrinkage and selection operator. Vide [PLS].

2.23.2 Algoritmo

```
?
lambda = 0.002;
B = lasso(xt,ydt,'Lambda',lambda);
```

Com este λ, foram removidas $6/12$ colunas em Death Valley.

2.23.3 Formulação

2.23.4 Resultado

2.24 (24) Contribuição θ MQ

2.24.1 Explicação

Vide Nota 24.

Módulo da contribuição percentual (regressores, pseudoinversa, offline) maior que:

constContribGlobal = 0.70; // dividido por nVariaveis; $0.6 \leq$ usual ≤ 0.8

A única com flagEMQ = false.

2.24.2 Algoritmo

```
24) Comece por todas as variáveis.
    Calcule theta pela pseudoinversa.
    Queremos min(contribuição(theta)) > constante.
    Enquanto isso for falso, retire a que menos contribui.
```

2.24.3 Formulação ***

$S_0 = N$;

for $i = 0, \cdots, n$:

Peso $c(S_i) = \dfrac{|\theta_i|}{\sum_j |\theta_j|}$; $c_i < c_j \Leftrightarrow S_i \prec S_j$;

$[c_e, e] = \min\{c\}$;

exit when flagBoa $= \left(c_e > k = \dfrac{0.70}{n} \right)$;

$S_{i+1} = S_i - \{x_e\}$;

end;

return S_i.

2.24.4 Resultado

2.25 (27) Full Scan Fwd Yamakawa

2.25.1 Explicação

Cada candidato evolui como um array bidimensional para frente e para trás.

Iniciar com 12 opções de 1 variável. Acrescentar uma variável a cada opção. Se o AIC aumentar, não será mais opção. Andar para trás: Retirar variáveis pelo θ = vetor de regressores de módulo menor que:

constTheta = 0.10; // 10% do máximo [Hiperparâmetro]

Enquanto isso, eliminar repetições. Se, depois disso, a quantidade de opções estabilizar, retornar.

Escolher $1 \leq n \leq 12$ variáveis dentre 12.

Dentre todas de $x = 1$ variável, escolher o mínimo $= f(x) = y$.

Em seguida, temos duas curvas (x,y), o Finc e o AIC.

(Isso não funcionou porque um não é vizinho do outro.)

Portanto, tomamos o mínimo erro de treinamento dentre todas as possibilidades. Isso deveria ser lento por serem 2^{12}. Acontece que revelou-se rápido. Com doze variáveis, terminou com 200 possibilidades.

Nota 19. Ou o full scan já era, ou precisamos de um equivalente. Por exemplo, evolucionário.

2.25.2 Algoritmo

```
27) Ordene asc, todas as possibilidades em que o AIC(n+1) >= AIC(n);
    Em todas as possibilidades, queremos theta maior ou igual que
    constante = 10% do máximo.
    Treine.
    Minimize o erro de treinamento, dentre todas as possibilidades.
```

2.25.3 Formulação

2.25.4 Resultado

2.26 (28) Full Scan Fwd Max

2.26.1 Explicação

O mesmo que o anterior, sem treinamento NFN Yamakawa.

2.26.2 Algoritmo

```
28) Ordene asc, todas as possibilidades em que o AIC(n+1) >= AIC(n);
    Em todas as possibilidades, queremos theta maior ou igual que
    constante = 10% do máximo.
    Minimize o SQE, dentre todas as possibilidades.
```

2.26.3 Formulação

2.26.4 Resultado

2.27 (29) Full Scan Fwd Sum

2.27.1 Explicação

2.27.2 Algoritmo

```
29) Ordene asc, todas as possibilidades em que o AIC(n+1) >= AIC(n);
    Em todas as possibilidades, queremos contribuição(theta) maior ou igual que
    constante = 0.70/n.
    Minimize o SQE, dentre todas as possibilidades.
```

2.27.3 Formulação

2.27.4 Resultado

2.28 (30) Full Scan Fwd Cumsum

2.28.1 Explicação

2.28.2 Algoritmo

```
30) Ordene asc, todas as possibilidades em que o AIC(n+1) >= AIC(n);
    Em todas as possibilidades, queremos contribuição acumulada menor ou igual que
    constante = 0.90.
    Minimize o SQE, dentre todas as possibilidades.
```

2.28.3 Formulação

2.28.4 Resultado

2.29 (31) Full Scan Bwd

2.29.1 Explicação

As possibilidades não se agrupam muito como no caso anterior.

Nota 20. Podemos agrupar os erros via $|e_i - e_j| < 10^{-5} \Rightarrow e_i \prec e_j \prec e_i$.

Vide seção 1.1.

2.29.2 Algoritmo

```
?
```

2.29.3 Formulação

2.29.4 Resultado

2.30 (32) Forest Importance

2.30.1 Explicação

2.30.2 Algoritmo

```
?
t = templateTree('NumVariablesToSample','all',...
'PredictorSelection','interaction-curvature','Surrogate','on');
Mdl = fitrensemble(xt,ydt,'Method','Bag','NumLearningCycles',200, ...
'Learners',t);
[impGain,predAssociation] = predictorImportance(Mdl);
```

2.30.3 Formulação

2.30.4 Resultado

2.31 (33) Permutation Importance

2.31.1 Explicação

2.31.2 Algoritmo

```
?
t = templateTree('NumVariablesToSample','all',...
'PredictorSelection','interaction-curvature','Surrogate','on');
Mdl = fitrensemble(xt,ydt,'Method','Bag','NumLearningCycles',200, ...
'Learners',t);
impOOB = oobPermutedPredictorImportance(Mdl);
```

2.31.3 Formulação

2.31.4 Resultado

3. Cada Algoritmo — Otimização × Solver

3.1 (34) Surrogate

Implementamos o surrogate com função de similaridade em um problema simples 5000×4, havia matriz 5000×5000. Depois com função de resíduo, que não ficou boa.

3.2 SLEP

3.2.1 (35) fusedLeastR

3.2.2 (36) nnLeastR

3.2.3 (37) nnLeastC

3.2.4 (38) LeastR0

3.2.5 (39) LeastR1

3.2.6 (40) LeastR2

3.2.7 (41) LeastC0

3.2.8 (42) LeastC1

3.2.9 (43) mtLeastR0

3.2.10 (44) mtLeastR1

3.2.11 (45) mtLeastR2

3.2.12 (46) mtLeastC0

3.2.13 (47) mtLeastC1

3.2.14 (48) mcLeastR0

3.2.15 (49) mcLeastR1

3.2.16 (50) mcLeastR2

3.2.17 (51) mcLeastC0

3.2.18 (52) mcLeastC1

3.2.19 (53) glLeastR0

3.2.20 (54) glLeastR1

3.2.21 (55) glLeastR2

3.2.22 (56) overlappingLeastR

3.2.23 (57) sgLeastR

3.2.24 (58) mcsgLeastR

3.2.25 (59) matPrimal

3.2.26 (60) matDual

3.2.27 (61) accelgradmtl

3.2.28 (62) accelgradmlr

3.2.29 (63) accelgradmc

3.2.30 (64) treemtLeastR0

3.2.31 (65) treemtLeastR1

3.2.32 (66) treemcLeastR0

3.2.33 (67) treemcLeastR1

3.2.34 (68) treeLeastR0

3.2.35 (69) treeLeastR1

70 a 192 reservados.

4. Python

4.1 193) SelectKBest

Exemplo 1.1

```
selector2 = SelectKBest(f_classif, k=ktmp)
clf_selected = make_pipeline(selector2, MinMaxScaler(), LinearSVC())
clf_selected.fit(X_train, y_train)
svm_weights_selected = np.abs(clf_selected[-1].coef_).sum(axis=0)
```

4.2 194) f classif × mutual info classif

Exemplo 1.2

```
selector2 = SelectKBest(mutual_info_classif, k=ktmp)
```

4.3 195) SVC e RFE

Exemplo 1.3

```
svc = SVC(kernel="linear", C=1)
rfe = RFE(estimator=svc, n_features_to_select=1, step=1)
rfe.fit(X_train, y_train)
ranking = rfe.ranking_ #.reshape(digits.images[0].shape)
ranking = ranking[-ktmp:];
X_train = X_train[:,ranking-1];
X_test = X_test[:,ranking-1];
```

4.4 196) PCA e NMF

Exemplo 2

4.5 197) Lasso

Exemplo 4
Vide [PLS].

```
lasso = LassoCV(cv=skf, random_state=42).fit(X, y)
lr = LogisticRegression(C=10, class_weight='balanced', max_iter=10000,
     random_state=42)
preds = cross_val_predict(lr, X[:, np.where(lasso.coef_!=0)[0]],
        y, cv=skf)
```

4.6 198) Random Forest Classifier × Select From Model

Exemplo 5.1

```
rf = RandomForestClassifier(n_estimators = 100, class_weight='balanced',
     random_state=42)
rf.fit(X_train, y_train)
importances = rf.feature_importances_
sfm = SelectFromModel(rf, threshold=0.06)
sfm.fit(X_train, y_train)
X_important_train = sfm.transform(X_train)
X_important_test = sfm.transform(X_test)
```

4.7 199) XGB Classifier

Exemplo 5.2

```
rf = XGBClassifier(n_estimators = 100, random_state=42)
```

4.8 200) Permutation Importance

Exemplo 6

```
perm = PermutationImportance(rf, random_state=42).fit(X_test, y_test)
skf = StratifiedKFold(n_splits=10)
rf = RandomForestClassifier(n_estimators = 100, class_weight='balanced',
     random_state=42)
preds= cross_val_predict(rf, X[:, np.where(abs(perm.feature_importances_)>=
       0.008)[0]], y, cv=skf)
```

4.9 Regression, Similarity based

O padrão é supervisionado.

4.9.1 201) SPEC, Unsupervised

Zhao, Zheng and Liu, Huan. "Spectral Feature Selection for Supervised and Unsupervised Learning." ICML 2007.

4.9.2 202) Fisher Score

[2,3] He, Xiaofei et al. "Laplacian Score for Feature Selection." NIPS 2005.
Duda, Richard et al. "Pattern classification." John Wiley & Sons, 2012.

4.9.3 203) Laplacian Score, Unsupervised

Quando fazemos 2 classes, clf.score retorna negativo.
Vide import Pipeline.

4.9.4 204) Relief F

Robnik-Sikonja, Marko et al. "Theoretical and empirical analysis of relieff and rrelieff." Machine Learning 2003.
Zhao, Zheng et al. "On Similarity Preserving Feature Selection." TKDE 2013.
5 Classes

4.9.5 205) Trace Ratio

Feiping Nie et al. "Trace Ratio Criterion for Feature Selection." AAAI 2008.

4.10 Classification, Wrapper

4.10.1 206) Decision Tree Backward

[6:9] Guyon, I. and Elisseeff, A., 2003. An introduction to variable and feature selection. Journal of machine learning research, 3(Mar), pp.1157-1182.
5 Classes
Here are some references for feature selection algorithms based on decision trees:
Quinlan, J. R. (1993). C4.5: Programs for machine learning. Morgan Kaufmann Publishers.
Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). Classification and regression trees. CRC press.
Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. Artificial Intelligence, 97(1-2), 273-324.
Liu, H., & Motoda, H. (Eds.). (1998). Feature extraction, construction and selection: A data mining perspective. Springer Science & Business Media.
Tin Kam Ho. (1998). The random subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(8), 832-844.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. Annals of Statistics, 1189-1232.

Zhang, Z., & Ma, Y. (2012). Feature selection using decision tree and genetic algorithm for microarray data classification. Journal of Medical Systems, 36(3), 1813-1820.

Liu, X. Y., & Wu, J. (2018). Feature selection based on decision tree for high-dimensional data. Journal of Applied Mathematics, 2018, 1-9.

4.10.2 207) Decision Tree Forward

5 Classes

4.10.3 208) SVM Backward

5 Classes

Here are some references for feature selection algorithms based on SVM:

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. Machine Learning, 46(1-3), 389-422.

Liu, J., Ding, X., & Yue, X. (2011). An effective SVM feature selection method for image retrieval. Expert Systems with Applications, 38(8), 10603-10611.

Sun, Y., Kamel, M. S., Wong, A. K., & Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. Pattern Recognition, 40(12), 3358-3378.

Wang, Y., Li, W., & Zhang, J. (2018). An efficient feature selection algorithm for SVM classification using hybrid genetic algorithm and mutual information. Information Sciences, 427, 1-18.

Huang, L., Liu, J., Yang, M., & Chen, L. (2017). Feature selection based on support vector machine and information entropy for Chinese text classification. Journal of Intelligent Information Systems, 49(3), 495-508.

Wang, J., Cao, L., Lu, J., & Zhang, L. (2019). Feature selection with a novel support vector data description. Neurocomputing, 339, 20-27.

Zheng, S., Zhao, Y., & Sun, Y. (2020). A novel feature selection method based on support vector machines and the max-relevance min-redundancy criterion. IEEE Access, 8, 16608-16616.

Liu, J., Chen, Y., & Zhao, J. (2021). An efficient recursive feature elimination algorithm for support vector machine classification. Pattern Recognition Letters, 148, 81-89.

4.10.4 209) SVM Forward

5 Classes

4.11 Regression, Structure

4.11.1 210) Graph FS

[10,11,12] Tang, J., Alelyani, S. and Liu, H., 2014. Feature selection for classification: A review. Data classification: Algorithms and applications, p.37.

This function implement the graph structural feature selection algorithm GOSCAR.

There are several research papers and articles that discuss graph structural feature selection algorithms. Here are a few references that you may find helpful:

Yanjie Fu, Jie Tang, Jing Zhang, and Zi Yang. "Graph-based feature selection for supervised learning."In Proceedings of the 27th AAAI Conference on Artificial Intelligence, pages 171-177, 2013.

Pengfei Jiao, Hao Peng, Senzhang Wang, and Xiangjie Kong. "Feature selection for graph-based semi-supervised learning."In Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM), pages 385-394, 2018.

Hao Wang, Wenchao Yu, and Jian Pei. "Graph-based feature selection: a review."ACM Computing Surveys (CSUR), 50(6), 2018.

Shuo Chen, Junjie Wu, Di Jin, and Xueqi Cheng. "Fast and scalable graph-based feature selection."In Proceedings of the 2019 SIAM International Conference on Data Mining (SDM), pages 390-398, 2019.

These papers provide insights into various aspects of graph structural feature selection algorithms, including their theoretical foundations, computational efficiency, and applications in different domains.

4.11.2 211) Group FS

This function implements supervised sparse group feature selection with least square loss.

[11,12] Liu, Jun, et al. "Moreau-Yosida Regularization for Grouped Tree Structure Learning." NIPS. 2010.

[11,12] Liu, Jun, et al. "SLEP: Sparse Learning with Efficient Projections." http://www.public.asu.edu/ jye02/S 2009.

4.11.3 212) Tree FS

This function implements tree structured group lasso regularization with least square loss.

4.12 Regression, Streaming

4.12.1 213) Alpha Investing

This function implements streamwise feature selection (SFS) algorithm alpha_investing for binary regression or univariate regression.

Zhou, Jing et al. "Streaming Feature Selection using Alpha-investing." KDD 2006.

4.13 Regression, Statistical Based

4.13.1 214) CFS

This function uses a correlation based heuristic to evaluate the worth of features which is called CFS.

Zhao, Zheng et al. "Advancing Feature Selection Research - ASU Feature Selection Repository" 2010.

Hall, M.A. and Smith, L.A., 1999, May. Feature selection for machine learning: comparing a correlation-based filter approach to the wrapper. In FLAIRS conference (Vol. 1999, pp. 235-239).

4.13.2 215) Chi Square

Liu, H. and Setiono, R., 1995, November. Chi2: Feature selection and discretization of numeric attributes. In Proceedings of 7th IEEE international conference on tools with artificial intelligence (pp. 388-391). IEEE.

Existing method for classification in scikit-learn.

5 Classes

4.13.3 216) F Score

Wright, S., 1965. The interpretation of population structure by F-statistics with special regard to systems of mating. Evolution, pp.395-420.

Existing method for classification in scikit-learn.

f_classif

4.13.4 217) Gini Index

5 Classes

Here are some references for Gini index feature selection:

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). Classification and regression trees. Chapman and Hall/CRC.

Zhang, H. (2004). The optimality of naive Bayes. Proceedings of the 17th International FLAIRS Conference, 562-567.

Peng, H., Long, F., & Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(8), 1226-1238.

Hall, M. A. (1998). Correlation-based feature selection for discrete and numeric class machine learning. Proceedings of the 4th International Conference on Machine Learning, 359-367.

Deng, X., & Huang, J. Z. (2012). Feature selection with a regularized gini index for gene expression data analysis. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 9(5), 1510-1521.

Brown, G., Pocock, A., Zhao, M. J., & Luján, M. (2012). Conditional likelihood maximisation: a unifying framework for information theoretic feature selection. Journal of Machine Learning Research, 13, 27-66.

Chen, J., Wang, X., & Xia, Y. (2018). A hybrid feature selection method based on Gini index and mutual information. Expert Systems with Applications, 98, 96-108.

Anwar, S., Sohail, A., & Javed, M. Y. (2019). An efficient feature selection algorithm based on Gini index and chaos theory for microarray gene expression data classification. Applied Soft Computing, 78, 475-487.

4.13.5 218) Low Variance, Unsupervised

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., 2011. Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12, pp.2825-2830.

Existing method in scikit-learn.

VarianceThreshold.

4.13.6 219) T Score

John C. Davis - Statistics and Data Analysis in Geology (3rd edition)-Wiley (2002)

This function calculates t_score for each feature.

2 Classes

4.14 Regression, Sparse Learning Based

4.14.1 220) LL L21

[20,21] Liu, Jun, et al. "Multi-Task Feature Learning Via Efficient l2,1-Norm Minimization." UAI. 2009.

2 Classes

4.14.2 221) LS L21

4.14.3 222) MCFS, Unsupervised

Cai, Deng et al. "Unsupervised Feature Selection for Multi-Cluster Data." KDD 2010.

4.14.4 223) NDFS, Unsupervised

Li, Zechao, et al. "Unsupervised Feature Selection Using Nonnegative Spectral Analysis." AAAI. 2012.

4.14.5 224) RFS

Nie, Feiping et al. “Efficient and Robust Feature Selection via Joint l2,1-Norms Minimization” NIPS 2010.

4.14.6 225) UDFS, Unsupervised

Yang, Yi et al. “l2,1-Norm Regularized Discriminative Feature Selection for Unsupervised Learning.” AAAI 2012.

4.15 Regression, Information Theoretical Based

4.15.1 226) CIFE

This function implements the CIFE feature selection.

[17,26,27,28,31:35] Brown, Gavin et al. “Conditional Likelihood Maximisation: A Unifying Framework for Information Theoretic Feature Selection.” JMLR 2012.

Lin, D. and Tang, X., 2006. Conditional infomax learning: An integrated framework for feature extraction and fusion. In Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9 (pp. 68-82). Springer Berlin Heidelberg.

4.15.2 227) CMIM

This function implements the CMIM feature selection.

Fleuret, F., 2004. Fast binary feature selection with conditional mutual information. Journal of Machine learning research, 5(9).

4.15.3 228) DISR

This function implement the DISR feature selection.

Meyer, P.E., Schretter, C. and Bontempi, G., 2008. Information-theoretic feature selection in microarray data using variable complementarity. IEEE Journal of Selected Topics in Signal Processing, 2(3), pp.261-274.

4.15.4 229) FCBF

This function implements Fast Correlation Based Filter algorithm.

Yu, Lei and Liu, Huan. “Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution.” ICML 2003.

4.15.5 230) ICAP

This function implements the ICAP feature selection.

El Akadi, A., El Ouardighi, A. and Aboutajdine, D., 2008. A powerful feature selection approach based on mutual information. International Journal of Computer Science and Network Security, 8(4), p.116.

4.15.6 231) JMI

This function implements the JMI feature selection.

Yang, H. and Moody, J., 1999. Data visualization and feature selection: New algorithms for nongaussian data. Advances in neural information processing systems, 12.

4.15.7 232) LCSI

This function implements the basic scoring criteria for linear combination of shannon information term.

Está sem referência.

4.15.8 233) MIFS

This function implements the MIFS feature selection.

Battiti, R., 1994. Using mutual information for selecting features in supervised neural net learning. IEEE Transactions on neural networks, 5(4), pp.537-550.

4.15.9 234) MIM

This function implements the MIM feature selection.

David D. Lewis. 1992. Feature Selection and Feature Extraction for Text Categorization. In Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992.

4.15.10 235) MRMR

This function implements the MRMR feature selection.

Peng, H., Long, F. and Ding, C., 2005. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on pattern analysis and machine intelligence, 27(8), pp.1226-1238.

5. Cada Algoritmo — Online

5.1 (22) Contribuição θ Online

5.1.1 Explicação

Inicie com todas as variáveis.

Para cada linha:

Procure em vartheta se o subconjunto atual já estava armazenado.

Se não estiver, inicialize com zero os regressores, a matriz de condicionamento P, e o RSE; e execute uma iteração do RMQ.

Caso contrário, evolua com aqueles regressores.

Armazene todos os thetas. A cada momento em que todas as contribuições são maiores que o mínimo, marque o θ como saída.

Ao final, percorra as saídas de $i = 1,2,\cdots,n$ variáveis e escolha o mínimo em função de i.

A seguir, escolha o mínimo dentre os i's.

Tentamos com inclinação constSlope, mas estavam todas na faixa de 10^{-10}.

Nota 21. Normalizar contribuições, depois constSlope.

5.1.2 Algoritmo

```
22) Comece por todas as variáveis.
    Calcule theta pelo Recursivo MQ.
    Queremos min(contribuição(theta)) > constante.
    Enquanto isso for falso, retire a que menos contribui.
```

5.1.3 Formulação

5.1.4 Resultado

5.2 fsContribSemLoop — (23) Contribuição θ Sem Loop

O algoritmo EMQ calculava a pseudoinversa recursivamente. Retiramos menores que $\frac{0.70}{n}$ de uma vez só.

5.2.1 Explicação

É um θ só para todas as variáveis. A cada linha, variam as contribuições maiores que constContrib (mas poderia ser também constTheta, trocando a soma no denominador pelo máximo).

Retorne o subconjunto da última linha.

5.2.2 Algoritmo

```
23) Comece por todas as variáveis.
    Calcule theta pelo Recursivo MQ.
    Queremos min(contribuição(theta)) > constante, de uma vez só (sem loop).
```

5.2.3 Formulação

5.2.4 Resultado

5.3 fsContribSemLoopMax

O mesmo que o anterior, mas retiramos 10% do máximo.

5.3.1 Explicação

5.3.2 Algoritmo

```
?
```

5.3.3 Formulação

5.3.4 Resultado

5.4 fsContribSemLoopCumsum

5.4.1 Explicação

5.4.2 Algoritmo

```
?
```

5.4.3 Formulação

5.4.4 Resultado

5.5 fsContribYamakawa — (25) Contribuição Yamakawa Online Sem Loop

5.5.1 Explicação

5.5.2 Algoritmo

```
25) ?
```

5.5.3 Formulação

5.5.4 Resultado

5.6 fsContribYamakawaCumsum

5.6.1 Explicação

5.6.2 Algoritmo

```
?
```

5.6.3 Formulação

5.6.4 Resultado

5.7 fsContrib

5.7.1 Explicação

5.7.2 Algoritmo

```
?
```

5.7.3 Formulação

5.7.4 Resultado

5.8 fsPCA — (26) PCA Online

5.8.1 Explicação

Vide equação 19.

5.8.2 Algoritmo

```
26) ?
```

5.8.3 Formulação

5.8.4 Resultado

5.9 fsPLSVIP

5.9.1 Explicação

5.9.2 Algoritmo

```
?
```

5.9.3 Formulação

5.9.4 Resultado

6. Comparação entre Formulações Matemáticas

6.1 Grau de Diferença

Qual é a diferença entre (0.2, 9, 10, 11) $\theta \geq$ constante; (4, 9, 14, 17) $\Delta AIC \geq$ constante; (6, 11, 16, 19) $\Delta RMSE \geq$ constante; e (5, 10, 15, 18) Finc mínimo?

(7) Fexc máximo;

(22, 23, 24) Contribuição $\geq$ constante.

6.2 Funções [Não] Lineares

meanAPE $\times$ maxAPE $\times$ validError: dependem de y_s e, portanto, do método NFN Yamakawa.

O Algoritmo $= 0$, parte 1, sem treinamento, originou o Algoritmo $= 22$.

O Algoritmo $= 0$, parte 2, se assemelha com o rβ.AIC, o rβ.Finc e o rβ.RMSE. (O assuntão são regressores e resíduos.)

Entre estes, o primeiro calcula RSS $= \sum \xi_i^2$ e o segundo calcula duas vezes e divide: $\frac{RSS_a}{RSS_c}$. No RMSE, $y_s = \Psi \cdot \theta$ é um protótipo de treinamento **linear**. É a raiz quadrada do RSS. O θ é uma função de (x,y). O ξ também. AIC e RMSE são duas funções de ξ, que devemos escolher.

Em concordância com os resultados, iremos estabelecer quando um cálculo é melhor que o outro.

6.3 Otimização

(7) O Fexc maximiza $J = \dfrac{\sum \xi_1^i}{\sum \xi_2^j}$.

O Finc minimiza $J = \dfrac{\sum \xi_1^i}{\sum \xi_2^j}$. Os outros minimizam o que?

. . . . . .

- **Complexidade** — o que muda se for online?

Temos os algoritmos recursivos de "Estimação de Parâmetros" para evoluir: MQR, MQRE, VI, VIM, A.Est., KF, EKF, UKF.

No "Filtro de Kalman", os θ's são substituídos por média e covariância. Denota-se $(\hat{\theta}_k, P_k)$. Dessa forma podemos comparar $\theta_A \pm P_A$ com $\theta_B \pm P_B$, isto é, as gaussianas.

*** A média e a covariância de Pearson e PCA bem que podia emular um Kalman.

. . . . . .

- **Intervalos**

Nota 22. Análise da incerteza do θ, da contribuição, do AIC, do RMSE, do Finc.

Implementamos incerteza no anfis e no evalfis. Existem os pesos $[w_-, w_+]$; as saídas $[y_{s-}, y_{s+}]$; a função de pertinência é $[\mu - \varepsilon, \mu + \varepsilon]$, $\varepsilon = 0.05$; isso implica em incerteza nas contribuições e nos erros de treinamento e de validação. O nome disso é: fuzzy tipo 2.

Nota 23. Fuzzificar e defuzzificar. $X = t \Leftrightarrow X \in C_t$, conjunto fuzzy.

. . . . . .

Deprecated

Inicialmente, buscávamos uma fórmula fechada para vip : $\mathbb{R}^n \times \mathbb{R}^{hn} \times \mathbb{R}^{n^2} \to \mathbb{R}^n$:

$$x_1(h \times n) = X_{train} \tag{1}$$
$$y_1(h \times 1) = Y_{train} \tag{2}$$
$$a_1(1 \times n) = \frac{y_1^\top x_1}{y_1^\top y_1} \tag{3}$$
$$c_1(n \times 1) = \frac{a_1^\top}{||a_1||} \tag{4}$$
$$d_k(h \times 1) = x_k c_1 \,;\, k \geq 1 \tag{5}$$
$$e_k(1 \times n) = \frac{d_k^\top x_k}{d_k^\top d_k} \tag{6}$$
$$t_k(h \times 1) = d_k ||e_k|| \tag{7}$$
$$w_k(n \times 1) = c_k ||e_k|| \tag{8}$$
$$p_k(n \times 1) = \frac{e_k^\top}{||e_k||} \tag{9}$$
$$b_k = \frac{y_k^\top t_k}{t_k^\top t_k} \in \mathbb{R} \tag{10}$$
$$y_{k+1}(h \times 1) = y_k - b_k t_k \tag{11}$$
$$x_{k+1} = x_k - \underbrace{t_k}_{h \times 1} \underbrace{b_k^\top}_{1 \times n} \tag{12}$$
$$k = k+1 \in \{2, 3, \cdots, n\} \text{ and go to } d_{k+1} \tag{13}$$
$$T(h \times n) = [t_1, t_2, \cdots, t_n] \tag{14}$$
$$\xi(h \times 1) = Y_{train} - T b^\top \tag{15}$$
$$P(n \times n) = [p_1, p_2, \cdots, p_n] \tag{16}$$
$$W(n \times n) = [w_1, w_2, \cdots, w_n] \tag{17}$$
$$V(1 \times n) = f(b, T, W) \tag{18}$$

Já temos um esboço de PCA online:

$$M = \overline{x_t}\,; M^{k+1} \leftarrow \frac{kM^k + x_{k+1}}{k+1} \tag{19}$$

$$x_L = x_t - M\,; a_{13} = \sum x_1 x_3 \mapsto \sum (x_1+b)(x_3+b) = a_{13} + b\sum x_1 + b\sum x_3 + kb^2 \tag{20}$$

$$S = \ \mathrm{cov}(x_L)\,; S_{13} = \overline{x_1 x_3} - m_1 m_3 \mapsto \frac{(S^k + m_1^k m_3^k)k + x_1^{k+1}x_3^{k+1}}{k+1} - m_1^{k+1}m_3^{k+1} \tag{21}$$

$$V,D \leftarrow (VDV^{-1} = S) \text{ — este é o passo-chave} \tag{22}$$

$$d = \ \mathrm{diag}(D) \tag{23}$$

$$\lambda = \frac{d}{\sum d_j} \tag{24}$$

$$m = \arg\min\{\ \mathrm{cumsum}(\lambda) \geq 0.9\} \tag{25}$$

$$W = V_{:,1:m} \tag{26}$$

$$X_{train} = x_L W \text{ ; cada linha está em um espaço. Funciona?} \tag{27}$$

$$X_{valid} = (x_v - M)W \tag{28}$$

6.4 Comparações entre Algoritmos

Imprimimos os algoritmos em ordem descendente de erros com suas incertezas.

Pela seção 8, execução 1, os melhores algoritmos são:

Death Valley:

21 (full scan Yamakawa) $\prec$ 10 (beta Finc) $\prec$ 19 (contribuição theta), 20 $\prec$ 11 (beta RMSE), 6 (RMSE), 16 (Triangular RMSE) $\prec$ 22 (full scan) $\prec$ 15 (triangular finc) $\prec$ 0 (contribuição Yamakawa) $\prec$ 1 (stepwise) $\prec$ 12 (pls vip) $\prec$ 8 (FexcLoss) $\prec$ 7 (Fexc) $\prec$ salto para 3 (Bwd) $\prec$ 5 (Finc) $\prec$ 2 (Fwd) $\prec$ 4 (AIC), 9 (beta AIC), 14 (triangular AIC).

O que há de bom na contribuição theta?
Por que os três RMSEs são parecidos?
Por que os três Fincs são bem diferentes?
Por que os dois Fexcs andam juntos?
Por que os três AICs juntos são os piores?

A contribuição theta tem a menor incerteza e minimiza a quantidade de colunas.

Compensa utilizar o full scan (pinv) ao invés do full scan Yamakawa.

Os dois Yamakawas estão próximos, apesar de as contribuições terem maior quantidade de colunas.

Todos os outros podem ser descartados.

15 variáveis arbitrárias:

Consideramos melhor os 19 (contribuição theta), 20, com 3 colunas; a seguir, o 0 (contribuição Yamakawa) com 12 colunas; a seguir os full scans.

Invertemos a flag EMQ e geramos a execução 2.

Interpretando o DeathValley, vemos o 19 e o 20 depois de um salto. O 0 e o 21 são os ótimos, a seguir o 22. Sendo que o 0 minimiza incerteza.

Pelo problema 4, vemos que entre o MQ e o EMQ, este tem erro de treinamento ligeiramente maior, com incerteza 6.57 vezes menor.

Ainda com a flag EMQ invertida, geramos a execução 3.

Interpretando o DeathValley, o ótimo é o 0, e apareceram os novos 15, 16 e 13 (PLS.Finc, PLS.RMSE e PLS.beta), equivalentes ao 0, com 8, 10, 12 e 12 colunas, incertezas baixas. A seguir, os full scans, e no final, as contribuições theta e MQ.

No problema 4, o 22 (contribuição theta) e o 23 (MQ) ficaram ótimos, a seguir o 14 (PLS.AIC), o 13 (PLS.beta) e o 0.

Com a flag EMQ de volta em zero na maioria, geramos a execução 4.

Interpretando o DeathValley, o ótimo é o 15 (PLS.Finc), a seguir o 16 (PLS.RMSE) e o 13 (PLS.beta), o 0 e os full scans.

No problema 4, o 22 (contribuição theta) e o 23 (MQ) ficaram ótimos, a seguir o 0. Os 14, 15, 16 estão depois de um salto.

Na execução 5, estão incluídas as Contribuições θ Online com loop e sem loop.

Na execução 6, estão incluídos o Yamakawa Online Sem Loop e o full scan fwd sum. Normalizamos os dados.

Com o objetivo de melhorar o MNIST, implementamos as funções Quadrado + PCA e Pivotear, e geramos a execução 7.

Dessa forma, conseguimos analisar se as linhas são linearmente dependentes com o quadrado.

6.5 Online

Criamos um metodoContribOnline. Testamos nos bancos "eOGS Classification" = EyeState; "eOGS Prediction" = Parkinsons; "FBeM BoxJenkins"; "FBeM Global40"; "FBeM Mackey Glass". Vide seção 7 — Plataforma de Testes.

A partir do momento que funcionou comparavelmente, implementamos o mesmo método nos códigos granulares: eGNN, FBeM, IBeM e eOGS.

D. Leite normaliza em [0,1]. Na hora de classificar, trocamos as classes 0 e 1 por 0.3333 e 0.6666.

Uma classe zero torna os regressores identicamente nulos e gera divisão por zero.

Antes as colunas eram $(h, :)$. Agora são (h, cols).

Sobrecarregamos o n, que passou a variar a cada linha. E todas as matrizes anteriores, consideramos nas iterações apenas as primeiras $n(h)$ colunas.

Na primeira iteração de h no FBeM, todas as variáveis são inicializadas com o número máximo de colunas.

Não nos preocupamos com mais de uma saída, por isso a saída é um vetor.

Ao executar a pseudoinversa $X \setminus c$, foi necessário retirar linhas iguais.

Ao fazer teste com 14900 linhas, a média foi de 1 segundo por linha.

A seguir, reestruturamos para fsContrib e fsContribSemLoop. E fizemos uma versão inicial para o MNIST. Embaralhamos as linhas, porém a acurácia ainda era baixa.

Pelo segundo algoritmo, o tempo de execução é cem vezes menor. Testamos os classificadores em uma versão reduzida do banco: 13017 linhas com as imagens "1", "5", "6", "7".

Algumas observações: $[0,1] \mapsto [1/3,2/3]$;

Se máximo = minímo, normalizar para 0.5;

D. Leite normaliza x_i de acordo com $\max(y) - \min(y)$.

O RMSE estava retornando NaN, porque havia muitos erros de execução em $A \setminus b$, por isso implementamos uma função pivotear, que elimina linhas de sistema indeterminado.

Quando procuramos pos = (PointInG == 0), há um bug que não encontra. Nesse caso, fazemos pos = nLinhas. Isso acontece no momento de combinar grânulos e quando o algoritmo armazena o número da linha em PointInG, que resultou em overflow no máximo de 2011 colunas.

A especificidade estava calculando logaritmo de zero, que trocamos para o próprio zero. Isso aconteceu também no cálculo de $[\underline{\ell},\overline{\ell}]$.

A saída prevista também retornava NaN, por divisão por soma escalar = 0. Deliberamos só dividir pela soma se não for nula.

O próximo teste, já com as melhorias acima, foi trocar a soma pelo máximo. Conforme o Algoritmo 23.

Conclusão: o NFN Yamakawa tem suas contribuições, analogamente cada algoritmo evolutivo as variáveis contribuem de outra forma.

Nota 24. Assim, deveríamos adaptar a contribuição ao classificador/regressor, por exemplo, via o RMSE do eGNN, que teve menor tempo de execução.

Temos várias funções dentro do assunto contribuição. Comparar com J e otimização. Vide seção 2.24.

Tentamos também a "feature selection com loop" com 13017 linhas e máximo de 700 θ's (eliminação por RSE mínimo), mas desistimos após 8 horas de execução.

Desejávamos que "com loop" funcionasse com variáveis globais; concluímos que só é possível sem loop.

Implementamos $\pm 5\%$ no min/máx IBeM (MNIST), FBeM (MNIST, box jenkins, global40, mackey glass), egnn (MNIST). A acurácia aumenta.

Implementamos o PCA.

Retiramos as colunas sem correlação entrada/saída, sobraram 532 colunas, por isso não houve memória para executar o quadrado das colunas e em seguida o PCA.

Implementamos a Contribuição Yamakawa online.

Nota 25. Faltam o xit, xft variando com as linhas.

Implementamos quadrados de 532 colunas, a fim de excluir uma coluna por ser o quadrado da outra, do mesmo jeito que se exclui dependência linear: $X_2 = aX_1^2 + bX_1 + c$.

Em batelada, testamos o quadrado, depois o PCA.

Implementamos o PCA evolutivo. Bom nos problemas tradicionais. 1.5 segundos por linha no MNIST: ruim.

Implementamos o quadrado evolutivo. Bom nos problemas tradicionais. Overflow de memória no MNIST.

Implementamos fsContribSemLoopCumsum.

Implementamos fsContribYamakawaCumsum.

Baixamos os bancos Parkinson's Disease Classification Data Set
e QSAR oral toxicity Data Set.

Pivotear, que antes retornava infinito, fica melhor com pinv.

A normalização do classificador depende do número de classes.

Esse número é 2: ELM/MLP são para classificação em $Y = \{1, -1\}$.

MNIST é classificação em 4 ou 10 classes. Conclusão: devemos nos preocupar apenas com classificação binária. A Tabela 1 mostra que apenas assim conseguimos acurácia maior que 30%.

Implementamos fsPearson. Funciona no MNIST com PCA. (Imaginemos executar uma fs, depois outra, como funções compostas.)

Corrigimos PCA := PCA2. Implementamos (30) Full Scan Fwd Cumsum. Não compensa fazer vários fsContribs/Sum/Max/Cumsum.

	$FBeM_1$	$FBeM_5$	$IBeM_1$	$IBeM_5$	$eGNN_1$	$eGNN_5$	$eOGS_1$	$eOGS_5$
	$FBeM_6$	$FBeM_7$	$IBeM_6$	$IBeM_7$	$eGNN_6$	$eGNN_7$	$eOGS_6$	$eOGS_7$
Acurácia	71.7907	72.5282	69.0328	74.9866	70.8595	59.2994	59.8679	67.3888
	75.7087	74.2952	82.1003	55.5658	68.1263	70.8458	63.7244	63.0022
Acertos	9345	9441	8986	9761	9223	7719	7793	8772
	9855	9671	10687	7233	8868	9222	8295	8201
Erros	3672	3576	4031	3256	3794	5298	5224	4245
	3162	3346	2330	5784	4149	3795	4722	4816
RMSE	0.1475	0.1470	0.1629	0.1507	0.1921	0.2549	0.2116	0.1910
	0.1375	0.1408	0.1181	0.1970	0.1917	0.2018	0.2019	0.2032
NDEI/GEI	1.0317	1.1276	1.1389	1.1557	1.3432	1.9554	0.7773	0.7414
	1.0130	1.0092	0.8696	1.4123	1.4120	1.4471	0.7662	0.7335
Especifidade	$\sim$		$\sim$		$\sim$		0.5079	0.3948
	$\sim$		$\sim$		$\sim$		0.4814	0.4310
Tempo (s)	225.1478	201.0479	472.9245	465.2704	747.0297	883.4884	907.9107	738.3306
	224.9299	182.5960	949.2061	512.6340	939.1768	666.0422	608.9714	551.7321
Regras	2	3	5	2	10	5	8	10
	2	1	2	3	3	4	8	15

Tabela 1 — Algoritmo 23 — Soma — PCA — 2 a 2 Classes

6.6 Próximos passos

Encontramos implementação Espectral em Python. Fizemos funcionar muitos algoritmos que vieram juntos. Desaguamos em problemas de otimização.

Executamos algoritmos tradicionais de mercado em Python. Procuramos análogos em MatLab. Tudo até o momento é em batelada.

Na bibliografia do github scikit-feature em Python, encontramos o github SLEP em MatLab compilando C. Essa otimização SLEP é rápida. Conseguimos selecionar variáveis de várias maneiras no banco MNIST.

O Yamakawa estava com muito custo computacional, por isso reduzimos de 20 para 1 época. Retiramos também o passo 0.2, em que calcular os regressores θ estava muito lento, para o caso de 13017×784.

Temos stepwise no Python. Temos várias funções dentro do assunto fwd e bwd.

Nota 26. Preocupar com big data.

Implementamos muitas incertezas nos dados. (Tanto Matlab — SLEP Inc's, por exemplo — quanto Python.)

Implementamos SQP (Matlab). Lento.

Verificamos que o SPEC é lento, mas há tantas formas de fazer e agrupar que deve ser possível acelerar/melhorar. O problema certo, com o solver certo, é uma melhoria. Por exemplo, o gradiente, acelerado ou não, poderia no mínimo ter uma busca linear. Porém, é claro que isso altera a simplicidade computacional.

A seguinte ideia funcionou (resta saber por quê): um score por linha $X(h,:)$. Depois o somatório dos scores. Implementamos em Python o (21) LS L21 online. Idem para (17) gini index online 2A2. Este não funcionou com uma linha por vez, mas funcionou de duas em duas linhas. Analisando os fontes, verificamos divisão dos dados em duas partes. O mesmo aconteceu para (15) chi square online 2A2; (18) low variance online 2A2; (2) fisher score online 2A2. Neste, para evitar divisão por zero, adicionamos a seguinte linha:

```
lap_score[np.where(lap_score == 0)] = 1
```

O mesmo para (16) f score online 2A2. Alteramos a seguinte parte do código:

```
dfbn = np.max([n_classes - 1,1])
dfwn = n_samples - n_classes
msb = ssbn / float(dfbn)
if dfwn == 0:
    msw = sswn
else:
    msw = sswn / float(dfwn)
constant_features_idx = np.where(msw == 0.0)[0]
if np.nonzero(msb)[0].size != msb.size and constant_features_idx.size:
    warnings.warn("Features %s are constant." % constant_features_idx, UserWarning)
msw[np.where(msw == 0)] = 1
f = msb / msw
```

No (29) FCBF online 7A7, havia entropias zero no denominador. Como elas vêm de um histograma, é necessária uma quantidade mínima de linhas, que no caso do MNIST, chegou a funcionar algumas vezes com 7, nunca com 6. Retiramos a baixa acurácia com a seguinte alteração:

```
for i in range(n_features):
    f = X[:, i]
    t1[i, 0] = i
    t1[i, 1] = su_calculation(f, y)
return t1[:,1]
```

Dessa forma, implementamos o SLEP online no Matlab, que de tão lento, só se aproveitou o OSLEP fusedLeastR, 1 ou 50 iterações (sendo que o original são 500), que executou em 30 ou 97 segundos. Adicionando incerteza a este, o tempo de execução passou para 1859 segundos.

Em Matlab, implementamos o (178) pls vip online, via somatório de scores. Idem para (179) pls beta online. Passamos também aos casos fsonline e MNIST. A seguir, adicionamos as incertezas:

fs2ContribSemLoopCumsum;
fs2ContribSemLoopMax;
fs2ContribSemLoop;
fs2ContribYamakawa;
fs2ContribYamakawaCumsum;
fs2Pearson;
fs2PLSVIP;
fs2PLSbeta.

Implementamos a acurácia de classificação em Matlab. Definitivamente, é preferível classificar no Python; o Yamakawa não é um bom classificador.

Utilizamos o chat.openai.com.

*Here are some references for **evolutionary** feature selection:*

Yang, S., & Moody, J. (1999). Data-driven evolutionary algorithms for solving mixed-integer optimization problems. IEEE Transactions on Evolutionary Computation, 3(4), 325-348.

Yu, L., Liu, H., & Motoda, H. (2004). Selecting relevant features for classification using an evolutionary algorithm. In Proceedings of the 3rd international conference on machine learning and cybernetics (Vol. 2, pp. 861-866).

Yang, C., Wang, X., & Zhang, L. (2010). An evolutionary feature selection algorithm based on mRMR criterion. In 2010 IEEE international conference on bioinformatics and biomedicine (pp. 243-246).

Luo, X., Wang, Y., & Chen, X. (2014). A novel feature selection algorithm using a genetic algorithm and support vector machine for diagnosis of erythemato-squamous diseases. Computational and mathematical methods in medicine, 2014.

Dash, M., & Liu, H. (1997). Feature selection for classification. Intelligent data analysis, 1(3), 131-156.

These references cover different aspects of evolutionary feature selection, including the use of different optimization algorithms, the evaluation of feature relevance, and the application of evolutionary feature selection in real-world problems.

*Yes, there are **fuzzy** methods for feature selection. Fuzzy feature selection methods use fuzzy logic and fuzzy sets to deal with uncertain and imprecise data, and to model the vague relationships between features and the target variable. Fuzzy feature selection methods can handle noisy or incomplete data, and can capture non-linear and non-monotonic relationships between features and the target variable.*

Some examples of fuzzy feature selection methods include:

Fuzzy rough sets: Fuzzy rough sets combine fuzzy logic and rough set theory to handle uncertainty and imprecision in data. Fuzzy rough sets can be used to identify the most relevant and discriminative features for classification or regression problems.

Fuzzy clustering-based feature selection: Fuzzy clustering-based feature selection uses fuzzy clustering algorithms to group similar features and reduce redundancy in the feature space. Fuzzy clustering-based feature selection can also identify the most representative and discriminative features for classification or regression problems.

Fuzzy support vector machines: Fuzzy support vector machines use fuzzy logic and fuzzy sets to model the uncertainty and vagueness in the data and the decision boundaries. Fuzzy support vector machines can also be used for feature selection by ranking the importance of the input features based on their contribution to the decision boundary.

Fuzzy decision trees: Fuzzy decision trees use fuzzy logic and fuzzy sets to model the uncertainty and vagueness in the data and the decision boundaries. Fuzzy decision trees can also be used for feature selection by ranking the importance of the input features based on their contribution to the decision rules.

Fuzzy feature selection methods have been applied in various fields, including pattern recognition, image processing, bioinformatics, and data mining.

*Here are some references for **fuzzy** feature selection:*

Derrac, J., García-Serrano, A., & Herrera, F. (2015). Fuzzy feature selection: a review and comparative analysis. Artificial Intelligence Review, 43(4), 519-535.

Geng, X., Zhang, Y., & Jin, Y. (2011). Fuzzy feature selection based on fuzzy rough sets for high-dimensional data. Pattern Recognition, 44(10-11), 2432-2447.

Huang, W., & Sun, C. (2010). A fuzzy clustering-based feature selection algorithm for support vector regression. Expert Systems with Applications, 37(3), 2159-2166.

Li, J., & Yang, J. B. (2011). Fuzzy clustering-based feature selection for support vector machines. Knowledge-Based Systems, 24(7), 1062-1071.

Liu, Y., Li, X., & Hu, Q. (2018). Fuzzy decision tree-based feature selection for high-dimensional data. Soft Computing, 22(12), 3947-3961.

Yu, X., Li, L., Li, Q., & Li, Y. (2019). A fuzzy rough set-based feature selection method for gene expression data. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 16(6), 1917-1926.

These references cover different aspects of fuzzy feature selection, including the use of different fuzzy logic and fuzzy set techniques, the evaluation of feature relevance, and the application of fuzzy feature selection in real-world problems.

*Here are some references for feature selection using **fuzzy support vector machines**:*

Jiao, L., Xue, B., Zhang, M., & Zhang, J. (2018). Feature selection via fuzzy support vector machines with mutual information. Knowledge-Based Systems, 141, 1-9.

Jiao, L., Xue, B., Zhang, M., & Zhang, J. (2018). Feature selection using fuzzy support vector machines with Gini index. Information Sciences, 465, 1-16.

Li, J., Li, Y., & Yang, J. B. (2011). Fuzzy clustering-based feature selection for support vector machines. Knowledge-Based Systems, 24(7), 1062-1071.

Li, Q., Zhang, J., & Jin, Y. (2012). Feature selection for support vector machines with a fuzzy-rough hybridization approach. IEEE Transactions on Fuzzy Systems, 20(6), 1096-1108.

Riaz, S., Qamar, U., Raza, S. A., & Siddiqui, S. A. (2018). Fuzzy entropy-based feature selection for support vector machine. Journal of Ambient Intelligence and Humanized Computing, 9(4), 1295-1304.

These references cover different aspects of feature selection using fuzzy support vector machines, including the use of different fuzzy logic and fuzzy set techniques, the evaluation of feature relevance, and the application of fuzzy feature selection in real-world problems.

*Here are some references for feature selection using **fuzzy decision trees**:*

Abawajy, J. H., & Kelarev, A. V. (2004). Feature selection with fuzzy decision trees. Journal of Mathematical Modelling and Algorithms, 3(3), 269-280.

Chakraborty, D., & Kundu, M. K. (2016). Fuzzy decision tree based feature selection for high-dimensional data classification. Applied Soft Computing, 49, 1023-1033.

Ghalmane, Z., & Bennani, S. (2015). Fuzzy decision trees-based feature selection for improved anomaly detection in computer networks. International Journal of Network Security, 17(5), 514-526.

Liu, Y., Li, X., & Hu, Q. (2018). Fuzzy decision tree-based feature selection for high-dimensional data. Soft Computing, 22(12), 3947-3961.

Yu, Y., & Zhou, J. (2013). Fuzzy decision tree-based feature selection for intrusion detection. Journal of Ambient Intelligence and Humanized Computing, 4(5), 567-575.

These references cover different aspects of feature selection using fuzzy decision trees, including the use of different fuzzy logic and fuzzy set techniques, the evaluation of feature relevance, and the application of fuzzy feature selection in real-world problems.

*Here are some references on **online** feature selection:*

Fan, W., & Lv, J. (2010). A selective overview of feature selection in machine learning. Data mining and knowledge discovery, 21(1), 191-202.

Hsu, C. W., Chang, C. C., & Lin, C. J. (2008). A practical guide to support vector classification. National Taiwan University, Taipei, Taiwan.

Liu, H., Motoda, H., & Yu, L. (2004). Feature selection for knowledge discovery and data mining. Springer.

Peng, H., Long, F., & Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on pattern analysis and machine intelligence, 27(8), 1226-1238.

? Wang, L., Chen, K., & Zhang, X. (2018). An overview of online feature selection for big data analytics. IEEE Access, 6, 73239-73251.

? Zhao, Y., & Liu, H. (2015). Online feature selection for big data analytics. Journal of Big Data, 2(1), 1-25.

These references provide an overview of different techniques for feature selection, including online feature selection, and their applications in machine learning and data mining.

*Here are some references that provide information on feature selection for **streaming** data:*

Katakis, I., Tsoumakas, G., & Vlahavas, I. (2009). Online feature selection for classification of streaming data. In Proceedings of the 9th International Conference on Data Mining (pp. 611-620).

Peng, Y., & Huang, S. J. (2013). Online feature selection for classification of high-speed data streams. Knowledge-Based Systems, 52, 46-54.

Tsai, C. F., Hsu, C. T., & Yang, C. H. (2016). A hybrid feature selection approach for data streams. Information Sciences, 367-368, 702-715.

Krawczyk, B., & Woźniak, M. (2017). An overview of feature selection methods in online feature selection with streaming data. Expert Systems with Applications, 69, 273-282.

Kumar, R., Gupta, V., & Agrawal, R. (2018). A review of feature selection techniques for streaming data. In Advances in Intelligent Systems and Computing (pp. 261-274). Springer.

Yang, W., Xie, F., & Xie, L. (2021). Online feature selection for classification of high-speed data streams: A survey. Information Sciences, 563, 273-292.

***Adaptive** feature selection is a widely researched topic, and there are several references available on the subject. Here are some of the most important ones:*

"Adaptive Feature Selection: Computationally Efficient Online Sparse Linear Regression under RIP" by M. Fardad, A. Khina, and G. J. Pappas, IEEE Transactions on Signal Processing, Vol. 60, No. 3, pp. 1362-1373, March 2012.

"Adaptive Feature Selection for High-Dimensional Data" by Y. Zhu, H. Zou, and T. Hastie, Journal of Machine Learning Research, Vol. 8, pp. 1027-1061, 2007.

"Adaptive Feature Selection Using Ensemble Methods" by T. C. Luong, D. D. Nguyen, and T. V. Nguyen, Expert Systems with Applications, Vol. 42, No. 6, pp. 3132-3143, 2015.

"Adaptive Feature Selection for Text Classification" by H. Jiang, Y. Bai, and Y. Li, Neurocomputing, Vol. 151, Part 2, pp. 1265-1272, 2015.

"Adaptive Feature Selection for Time-Series Classification" by T. Guo, J. Peng, and H. Sun, Knowledge-Based Systems, Vol. 160, pp. 68-79, 2018.

E o google scholar?

*** Exemplo de comparação: MQ e EMQ dá na mesma.

*** Exemplo de resultado: fsSemLoop, alguns são equivalentes.

*** Exemplo de conclusão: com mais ou menos 5% nas entradas e saída, a acurácia aumenta.

Nota 27. Procure por ***.

*** Pegar a contribuição do Yamakawa e comparar com a de Pearson, matematicamente.

Não lineares é o principal.

Uma não tem nada a ver com a outra.

Começar pelo x^2 da prova de sistemas nebulosos.

*** Se é 10×30000, então é física. Queremos dizer que a gravidade não depende da temperatura.

*** Temos incerteza no score Yamakawa. Queremos incertezas internas em todos os outros. Vide nota 22.

*** Como D. Leite faz com xit e xft? Vide nota 25.

*** Sobre autovetores:

[SPECTRAL] p. 141: PARPACK e ScaLAPACK.

O autor se preocupa com p processadores em rede, enquanto preocupamo-nos com online.

Na versão MatLab de PCA, há outras opções de cálculo que não diagonalizam. Melhores ou piores?

*** Granular a contribuição das características com o tempo h.

7. Plataforma de Testes

Nota 28. resultados para nVariaveis arbitrário.

Nota 29. nrg e outros problemas no repositório: https://archive.ics.uci.edu/ml/datasets.

Nota 30. resultados distintos por tipo de problema: classificação, regressão, imagem, vídeo

Criamos um problema "ArbitrNVar", de forma análoga ao exemplo de [JANG] $y = (1 + x_1^{0.5} + x_2^{-1} + x_3^{-1.5})^2$, com $y = \sum x_i^{a_i}$, em que os expoentes são fixos em $[-1.5,1.5]$, e os x_i de treinamento e validação são fixos em $[0,1]$.

Temos resultados para 1000 linhas de 25 variáveis. Preocupamo-nos também com zero variáveis.

Criamos um problema de classificação em que a coluna y é igual a randi$(c,n,1)$, em que o número de classes é $c \in \{2,5\}$.

Testamos nos bancos "eOGS Classification" = EyeState; "eOGS Prediction" = Parkinsons; "FBeM BoxJenkins"; "FBeM Global40"; "FBeM Mackey Glass".

Testamos no MNIST.

8. Resultados

Comparamos 0 = contribuição yamakawa; 1 = stepwise; 2 = forward; 3 = backward; 4 = AIC; 5 = Finc ; 6 = RMSE ; 7 = Fexc ; 8 = FexcLoss ; 9 = rβ.AIC ; 10 = rβ.Finc ; 11 = rβ.RMSE; 12 = PLS.VIP ; 14 = Triangular.AIC ; 15 = Triangular.Finc ; 16 = Triangular.RMSE ; 19 = Contribuição θ ; 20 = Contribuição EMQ ; 21 = full scan fwd Yamakawa ; 22 = full scan fwd.

Yamakawa geralmente é o melhor com hiperparâmetro 0.70.

Concluímos que o full scan fwd Yamakawa é o ótimo com constTheta $= 0$.

Mas que haveremos de aumentar o hiperparâmetro com n.

Com doze variáveis e 0.10 continua sendo o ótimo.

Nada impede de, em produção, executar este e o Yamakawa em paralelo.

A saída foi:

8.1 Execução 1

```
// Problema = mgdata, Execução 1
vip =

    0.9620    0.9757    1.0103    1.0496

// full scan 21: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 6, antes = 10
ell = 3, K = 6, antes = 13
ell = 4, K = 6, antes = 14
Elapsed time is 1.766301 seconds.
// full scan 22: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 6, antes = 10
ell = 3, K = 6, antes = 13
ell = 4, K = 6, antes = 14
Elapsed time is 0.955911 seconds.
// Erros de treinamento e validação 1:
( 6)           0.3187           0.0029  ( 6)           0.6776           0.0099
(11)           0.3187           0.0029  (11)           0.6776           0.0099
(16)           0.3187           0.0029  (16)           0.6776           0.0099
(19)           0.3187           0.0029  (19)           0.6776           0.0099
```

```
(20)           0.3187           0.0029  (20)           0.6776           0.0099
(12)           0.0849           0.0019  (12)           0.1856           0.0000
( 9)           0.0598           0.0015  ( 4)           0.1042           0.0016
( 4)           0.0598           0.0015  (14)           0.1042           0.0016
(14)           0.0598           0.0015  (15)           0.1042           0.0016
(15)           0.0598           0.0015  ( 9)           0.1042           0.0016
( 2)           0.0317           0.0007  ( 2)           0.0709           0.0005
( 5)           0.0317           0.0007  ( 5)           0.0709           0.0005
( 3)           0.0168           0.0005  ( 3)           0.0351           0.0008
( 0)           0.0125           0.0003  ( 0)           0.0221           0.0002
( 1)           0.0125           0.0003  ( 1)           0.0221           0.0002
( 8)           0.0125           0.0003  ( 8)           0.0221           0.0002
(10)           0.0125           0.0003  (10)           0.0221           0.0002
(21)           0.0125           0.0003  (21)           0.0221           0.0002
(22)           0.0125           0.0003  (22)           0.0221           0.0002
( 7)           0.0114           0.0004  ( 7)           0.0185           0.0002
(13)           0.0000           0.0000  (13)           0.0000           0.0000
(17)           0.0000           0.0000  (17)           0.0000           0.0000
(18)           0.0000           0.0000  (18)           0.0000           0.0000
// Colunas utilizadas 1:

k =

     4     3     2     1
     4     3     2     1
     2     1     0     0
     4     2     1     0
     4     3     1     0
     2     1     0     0
     4     0     0     0
     3     2     1     0
     4     3     2     1
     4     3     1     0
     4     3     2     1
     4     0     0     0
     4     3     0     0
     0     0     0     0
     4     3     1     0
     4     3     1     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     4     0     0     0
     4     3     2     1
     4     3     2     1

// Problema = Dynamic, Execução 1
vip =

    0.0192    0.0344    0.0350    2.2353    0.0249

// full scan 21: tic
ell = 1, K = 5, antes = 5
ell = 2, K = 5, antes = 11
ell = 3, K = 3, antes = 15
ell = 4, K = 2, antes = 16
ell = 5, K = 2, antes = 16
Elapsed time is 1.640558 seconds.
// full scan 22: tic
```

```
ell = 1, K = 5, antes = 5
ell = 2, K = 5, antes = 11
ell = 3, K = 3, antes = 15
ell = 4, K = 2, antes = 16
ell = 5, K = 2, antes = 16
Elapsed time is 0.651692 seconds.
// Erros de treinamento e validação 2:
( 5)           0.6413           0.0020  ( 5)           0.6440           0.0002
( 7)           0.4920           0.0006  ( 7)           0.4754           0.0047
( 0)           0.0547           0.0002  ( 0)           0.0511           0.0019
( 4)           0.0547           0.0002  ( 4)           0.0511           0.0019
( 6)           0.0547           0.0002  ( 6)           0.0511           0.0019
( 9)           0.0547           0.0002  ( 9)           0.0511           0.0019
(11)           0.0547           0.0002  (11)           0.0511           0.0019
(12)           0.0547           0.0002  (12)           0.0511           0.0019
(14)           0.0547           0.0002  (14)           0.0511           0.0019
(16)           0.0547           0.0002  (16)           0.0511           0.0019
(19)           0.0547           0.0002  (19)           0.0511           0.0019
(20)           0.0547           0.0002  (20)           0.0511           0.0019
(21)           0.0547           0.0002  (21)           0.0511           0.0019
(22)           0.0547           0.0002  (22)           0.0511           0.0019
( 2)           0.0536           0.0003  ( 8)           0.0483           0.0016
( 8)           0.0465           0.0003  ( 2)           0.0422           0.0042
( 1)           0.0435           0.0004  ( 1)           0.0416           0.0009
( 3)           0.0424           0.0001  (15)           0.0409           0.0004
(15)           0.0413           0.0002  ( 3)           0.0402           0.0004
(10)           0.0411           0.0002  (10)           0.0398           0.0005
(13)           0.0000           0.0000  (13)           0.0000           0.0000
(17)           0.0000           0.0000  (17)           0.0000           0.0000
(18)           0.0000           0.0000  (18)           0.0000           0.0000
// Colunas utilizadas 2:

k =

     4     0     0     0     0
     4     2     0     0     0
     4     3     0     0     0
     5     4     3     2     1
     4     0     0     0     0
     5     0     0     0     0
     4     0     0     0     0
     5     3     2     1     0
     5     4     3     2     0
     4     0     0     0     0
     5     4     2     1     0
     4     0     0     0     0
     4     0     0     0     0
```

```
     0     0     0     0     0
     4     0     0     0     0
     4     3     2     1     0
     4     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0

// Problema = DeathValley, Execução 1
vip =

  Columns 1 through 10

    1.0538    1.0455    1.0246    0.9960    0.9697    0.9513    0.9462    0.9548    0.9734    1.0015

  Columns 11 through 12

    1.0272    1.0474

// full scan 21: tic
ell = 1, K = 12, antes = 12
ell = 2, K = 49, antes = 73
ell = 3, K = 132, antes = 193
ell = 4, K = 233, antes = 409
ell = 5, K = 255, antes = 662
ell = 6, K = 243, antes = 795
ell = 7, K = 208, antes = 831
ell = 8, K = 186, antes = 834
ell = 9, K = 180, antes = 835
ell = 10, K = 179, antes = 835
ell = 11, K = 179, antes = 835
Elapsed time is 86.613007 seconds.
// full scan 22: tic
ell = 1, K = 12, antes = 12
ell = 2, K = 49, antes = 73
ell = 3, K = 132, antes = 193
ell = 4, K = 233, antes = 409
```

```
ell = 5, K = 255, antes = 662
ell = 6, K = 243, antes = 795
ell = 7, K = 208, antes = 831
ell = 8, K = 186, antes = 834
ell = 9, K = 180, antes = 835
ell = 10, K = 179, antes = 835
ell = 11, K = 179, antes = 835
Elapsed time is 22.709356 seconds.
// Erros de treinamento e validação 3:
 1 ( 4)          69.2252          0.1375  ( 2)          73.7292          3.3944
 1 ( 9)          69.2252          0.1375  ( 4)          71.2872          4.1529
 1 (14)          69.2252          0.1375  ( 9)          71.2872          4.1529
 3 ( 2)          60.8121          0.0070  (14)          71.2872          4.1529
 2 ( 5)          52.6799          0.2012  ( 3)          66.2396          1.5627
 5 ( 3)          51.8737          0.2304  (10)          59.0886          0.2545
 3 ( 7)          46.6962          0.6256  (15)          58.7454          1.1075
 8 ( 8)          46.6859          0.4298  ( 7)          57.4482          0.3815
 6 (12)          46.4454          0.0055  ( 1)          56.4892          1.0088
 7 ( 1)          45.2313          0.1366  ( 5)          56.0782          2.3622
 9 ( 0)          45.2020          0.2532  ( 8)          55.7659          0.9556
11 (15)          45.1749          0.3892  (22)          53.9994          1.0535
 5 (22)          44.9361          0.1743  ( 0)          53.0812          0.6347
 4 (11)          44.8848          0.1325  ( 6)          48.1100          0.2764
 4 ( 6)          44.8848          0.1325  (16)          48.1100          0.2764
 4 (16)          44.8848          0.1325  (11)          48.1100          0.2764
 3 (19)          44.6588          0.0272  (12)          47.0035          0.8293
 3 (20)          44.6588          0.0272  (19)          46.4504          1.0914
12 (10)          43.8568          0.4496  (20)          46.4504          1.0914
 5 (21)          42.7533          0.4990  (21)          46.3362          0.1184
 0 (13)           0.0000          0.0000  (13)           0.0000          0.0000
 0 (17)           0.0000          0.0000  (17)           0.0000          0.0000
 0 (18)           0.0000          0.0000  (18)           0.0000          0.0000
// Colunas utilizadas 3:

k =

    12    11    10     9     8     5     3     2     1     0     0     0
    12    11     9     7     4     2     1     0     0     0     0     0
    10     7     1     0     0     0     0     0     0     0     0     0
    11     9     6     4     1     0     0     0     0     0     0     0
```

```
     1     0     0     0     0     0     0     0     0     0     0     0
     7     6     0     0     0     0     0     0     0     0     0     0
    12     9     2     1     0     0     0     0     0     0     0     0
     8     7     6     0     0     0     0     0     0     0     0     0
    12    11    10     8     7     5     4     3     0     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     7     6     5     4     3     2     1
    12     9     2     1     0     0     0     0     0     0     0     0
    12    11    10     3     2     1     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     7     5     4     3     2     1     0
    12     9     2     1     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
    12     2     1     0     0     0     0     0     0     0     0     0
    12     2     1     0     0     0     0     0     0     0     0     0
    12     8     7     6     3     0     0     0     0     0     0     0
    12     9     8     5     2     0     0     0     0     0     0     0

// Problema = ArbitrNVar, Execução 1
vip =

  Columns 1 through 10

    0.9837    1.0345    1.1273    0.9494    1.0329    1.0119    1.1470    0.9133    0.9599    0.9924

  Columns 11 through 15

    1.0670    0.6734    1.0264    0.9954    1.0034

// full scan 21: tic
ell = 1, K = 15, antes = 15
ell = 2, K = 92, antes = 106
ell = 3, K = 269, antes = 355
ell = 4, K = 415, antes = 686
ell = 5, K = 578, antes = 1037
ell = 6, K = 608, antes = 1203
ell = 7, K = 597, antes = 1216
ell = 8, K = 596, antes = 1216
```

```
ell = 9, K = 596, antes = 1216
Elapsed time is 318.002198 seconds.
// full scan 22: tic
ell = 1, K = 15, antes = 15
ell = 2, K = 92, antes = 106
ell = 3, K = 269, antes = 355
ell = 4, K = 415, antes = 686
ell = 5, K = 578, antes = 1037
ell = 6, K = 608, antes = 1203
ell = 7, K = 597, antes = 1216
ell = 8, K = 596, antes = 1216
ell = 9, K = 596, antes = 1216
Elapsed time is 58.235780 seconds.
// Erros de treinamento e validação 4:
 1 ( 1)         245.4633         34.9802  ( 7)         90.2613          1.9506
 2 ( 2)         205.5005         12.5821  ( 3)         88.0515          2.3688
 8 (12)         177.9301          0.6779  ( 8)         84.4699          1.5230
14 ( 7)         177.5313          0.7324  ( 5)         83.9036          1.3577
 5 ( 6)         170.8485          3.9284  (10)         83.1619          1.5125
 5 (16)         170.8485          3.9284  (15)         82.7260          4.9263
 5 (11)         170.8485          3.9284  ( 0)         80.3805          1.8929
 5 (22)         170.1945          2.7819  (12)         62.9494          5.9414
 1 ( 4)         169.9335          2.4327  (22)         54.7064          4.5079
 1 ( 9)         169.9335          2.4327  (11)         51.4147          3.8438
 1 (14)         169.9335          2.4327  ( 6)         51.4147          3.8438
 3 (19)         168.6862          2.1328  (16)         51.4147          3.8438
 3 (20)         168.6862          2.1328  (21)         39.7322          4.3752
11 ( 5)         167.3301          3.2487  ( 1)         29.8402          5.8771
14 ( 3)         162.8998          3.0316  (19)         25.3323          0.4160
15 (10)         161.8816          2.7244  (20)         25.3323          0.4160
12 ( 0)         161.7587          2.2309  ( 2)         23.7938          1.5155
14 (15)         161.5492          2.9749  ( 4)         22.1463          0.1864
14 ( 8)         161.4395          2.8820  ( 9)         22.1463          0.1864
 4 (21)         160.7270          3.6282  (14)         22.1463          0.1864
 0 (13)           0.0000          0.0000  (13)          0.0000          0.0000
 0 (17)           0.0000          0.0000  (17)          0.0000          0.0000
 0 (18)           0.0000          0.0000  (18)          0.0000          0.0000
// Colunas utilizadas 4:

k =
```

```
    15    14    13    12    10     9     7     6     5     4     3     1     0     0     0
    12     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    12     4     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     7     6     5     4     3     2     1     0
     7     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    10     9     8     6     5     4     1     0     0     0     0
    12    11     7     3     2     0     0     0     0     0     0     0     0     0     0
    15    14    13    11    10     9     8     7     6     5     4     3     2     1     0
    15    14    12    11    10     9     8     7     6     5     4     3     2     1     0
     7     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     3     2     1
    12    11     7     3     2     0     0     0     0     0     0     0     0     0     0
    15    13    11     7     6     5     3     2     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     7     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11     9     8     7     6     5     4     3     2     1     0
    12    11     7     3     2     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    12     7     3     0     0     0     0     0     0     0     0     0     0     0     0
    12     7     3     0     0     0     0     0     0     0     0     0     0     0     0
    15    12     5     3     0     0     0     0     0     0     0     0     0     0     0
    12     7     3     2     1     0     0     0     0     0     0     0     0     0     0

// Problema = Classificacao 2, Execução 1
vip =

    0.9535    1.0602    0.9963    0.9871

// full scan 21: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.379447 seconds.
// full scan 22: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
```

```
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.191800 seconds.
// Erros de treinamento e validação 5:
( 7)           0.8887           0.0642  ( 2)           1.0791           0.1260
( 2)           0.7720           0.0179  ( 4)           1.0442           0.0505
( 4)           0.7465           0.0257  ( 9)           1.0442           0.0505
( 9)           0.7465           0.0257  (14)           1.0442           0.0505
(14)           0.7465           0.0257  ( 6)           0.9448           0.1229
( 3)           0.7043           0.0079  (11)           0.9448           0.1229
(15)           0.7043           0.0079  (12)           0.9448           0.1229
( 6)           0.6974           0.0200  (16)           0.9448           0.1229
(11)           0.6974           0.0200  ( 0)           0.8326           0.0267
(12)           0.6974           0.0200  (10)           0.8326           0.0267
(16)           0.6974           0.0200  (19)           0.8326           0.0267
( 5)           0.6750           0.0009  (20)           0.8326           0.0267
( 8)           0.6750           0.0009  (21)           0.8326           0.0267
( 0)           0.5994           0.0072  (22)           0.8326           0.0267
(10)           0.5994           0.0072  ( 3)           0.7884           0.0416
(19)           0.5994           0.0072  (15)           0.7884           0.0416
(20)           0.5994           0.0072  ( 5)           0.7221           0.0699
(21)           0.5994           0.0072  ( 8)           0.7221           0.0699
(22)           0.5994           0.0072  ( 7)           0.5533           0.0726
( 1)           0.0000           0.0000  ( 1)           0.5077           0.0000
(13)           0.0000           0.0000  (13)           0.0000           0.0000
(17)           0.0000           0.0000  (17)           0.0000           0.0000
(18)           0.0000           0.0000  (18)           0.0000           0.0000
// Colunas utilizadas 5:

k =

     4     3     2     1
     0     0     0     0
     3     1     0     0
     4     3     2     0
     3     2     0     0
     4     3     1     0
     2     0     0     0
     1     0     0     0
     4     3     1     0
     3     2     0     0
     4     3     2     1
     2     0     0     0
     2     0     0     0
     0     0     0     0
     3     2     0     0
     4     3     2     0
     2     0     0     0
     0     0     0     0
     0     0     0     0
     4     3     2     1
     4     3     2     1
     4     3     2     1
     4     3     2     1

// Problema = Classificacao 5, Execução 1
vip =

    1.0045    0.9666    1.0422    0.9851
```

```
// full scan 21: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 14
Elapsed time is 0.301158 seconds.
// full scan 22: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 14
Elapsed time is 0.110268 seconds.
// Erros de treinamento e validação 6:
( 4)           1.7655           0.0349  ( 4)           2.1671           0.0760
( 9)           1.7655           0.0349  ( 9)           2.1671           0.0760
(11)           1.7655           0.0349  (11)           2.1671           0.0760
(14)           1.7655           0.0349  (14)           2.1671           0.0760
(19)           1.7655           0.0349  (19)           2.1671           0.0760
(20)           1.7655           0.0349  (20)           2.1671           0.0760
( 6)           1.6708           0.0039  (12)           1.8629           0.0475
(16)           1.6708           0.0039  ( 2)           1.7578           0.0016
( 7)           1.6578           0.0038  ( 3)           1.7578           0.0016
( 8)           1.6578           0.0038  (22)           1.7578           0.0016
( 5)           1.6578           0.0038  ( 6)           1.6449           0.0936
(12)           1.6532           0.0363  (16)           1.6449           0.0936
( 2)           1.6469           0.0208  (15)           1.5857           0.0460
( 3)           1.6469           0.0208  (21)           1.5857           0.0460
(22)           1.6469           0.0208  ( 5)           1.5606           0.0360
(15)           1.5678           0.0009  ( 7)           1.5606           0.0360
(21)           1.5678           0.0009  ( 8)           1.5606           0.0360
( 0)           1.5458           0.0281  ( 0)           1.5458           0.0169
(10)           1.5458           0.0281  (10)           1.5458           0.0169
( 1)           0.0000           0.0000  ( 1)           1.4958           0.0000
(13)           0.0000           0.0000  (13)           0.0000           0.0000
(17)           0.0000           0.0000  (17)           0.0000           0.0000
(18)           0.0000           0.0000  (18)           0.0000           0.0000
// Colunas utilizadas 6:

k =

     4     3     2     1
     0     0     0     0
     4     3     1     0
     4     3     1     0
     3     0     0     0
     4     2     1     0
     3     2     0     0
     4     2     1     0
     4     2     1     0
     3     0     0     0
     4     3     2     1
     3     0     0     0
     3     1     0     0
     0     0     0     0
     3     0     0     0
     3     2     1     0
     3     2     0     0
     0     0     0     0
     0     0     0     0
     3     0     0     0
```

3	0	0	0
3	2	1	0
4	3	1	0

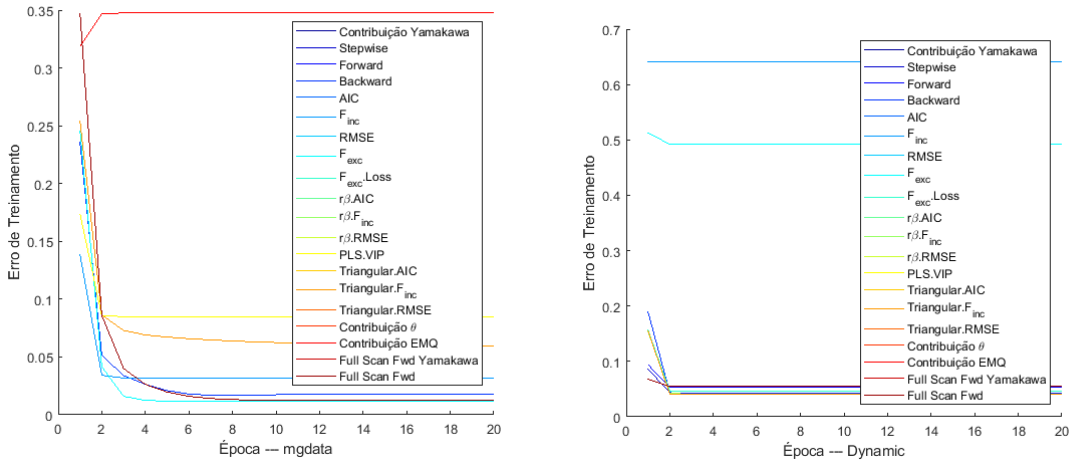

Figura 1: (1) mgdata (2) Dynamic

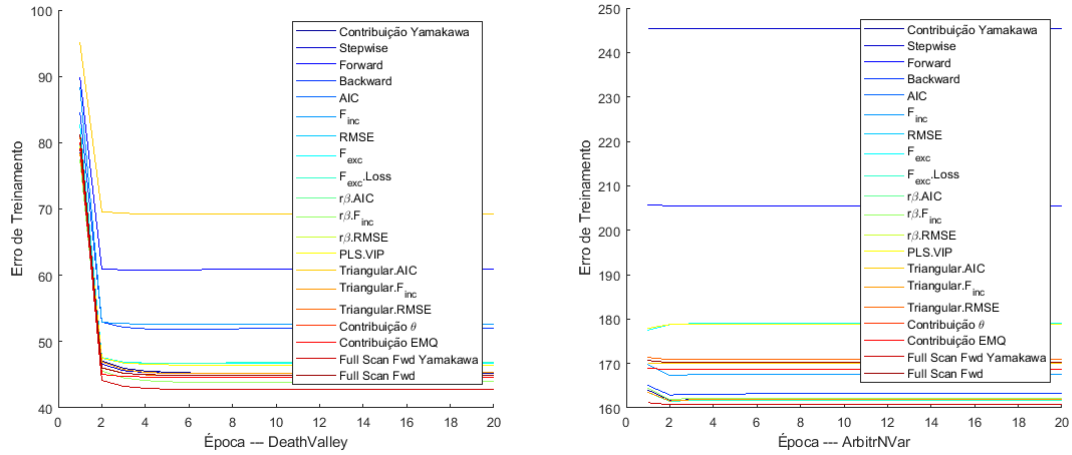

Figura 2: (1) DeathValley (2) ArbitrNVar 15

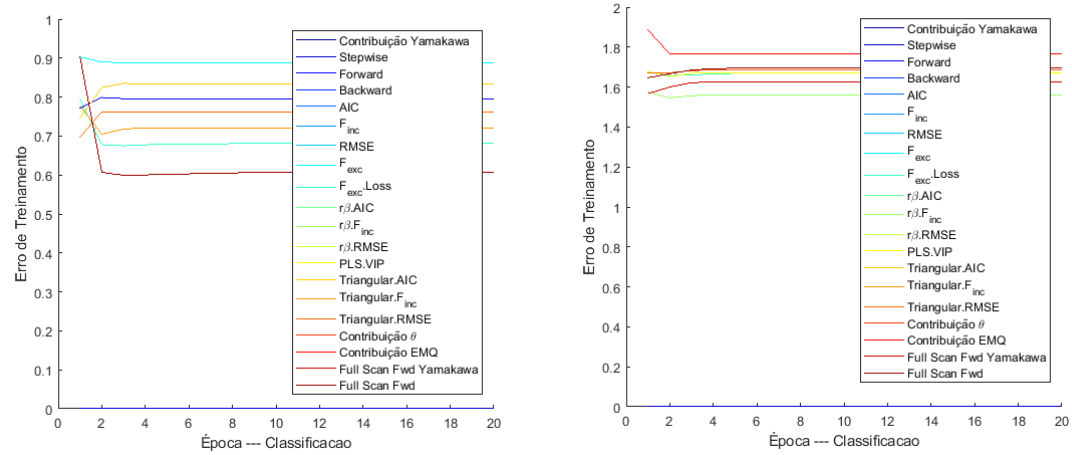

Figura 3: (1) Classificação Binária (2) 5 Classes

8.2 Execução 2

Aqui a flagEMQ é verdadeira sempre, exceto no Algoritmo $= 20$.

```
// Problema = mgdata, Execução 1
vip =

    0.9620    0.9757    1.0103    1.0496

// full scan 21: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 6, antes = 10
ell = 3, K = 6, antes = 13
ell = 4, K = 6, antes = 14
Elapsed time is 1.868714 seconds.
// full scan 22: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 6, antes = 10
ell = 3, K = 6, antes = 13
ell = 4, K = 6, antes = 14
Elapsed time is 1.064685 seconds.
// Erros de treinamento e validação 1:
 1 ( 6)           0.3187           0.0029  ( 6)           0.6776           0.0099
 1 (11)           0.3187           0.0029  (11)           0.6776           0.0099
 1 (16)           0.3187           0.0029  (16)           0.6776           0.0099
 1 (19)           0.3187           0.0029  (19)           0.6776           0.0099
 1 (20)           0.3187           0.0029  (20)           0.6776           0.0099
 2 (12)           0.0849           0.0019  (12)           0.1856           0.0000
 3 ( 9)           0.0598           0.0015  ( 4)           0.1042           0.0016
 3 ( 4)           0.0598           0.0015  (14)           0.1042           0.0016
 3 (14)           0.0598           0.0015  (15)           0.1042           0.0016
 3 (15)           0.0598           0.0015  ( 9)           0.1042           0.0016
 2 ( 2)           0.0317           0.0007  ( 2)           0.0709           0.0005
 2 ( 5)           0.0317           0.0007  ( 5)           0.0709           0.0005
 3 ( 3)           0.0168           0.0005  ( 3)           0.0351           0.0008
 4 ( 0)           0.0125           0.0003  ( 0)           0.0221           0.0002
 4 ( 1)           0.0125           0.0003  ( 1)           0.0221           0.0002
 4 ( 8)           0.0125           0.0003  ( 8)           0.0221           0.0002
 4 (10)           0.0125           0.0003  (10)           0.0221           0.0002
 4 (21)           0.0125           0.0003  (21)           0.0221           0.0002
 4 (22)           0.0125           0.0003  (22)           0.0221           0.0002
 3 ( 7)           0.0114           0.0004  ( 7)           0.0185           0.0002
 0 (13)           0.0000           0.0000  (13)           0.0000           0.0000
 0 (17)           0.0000           0.0000  (17)           0.0000           0.0000
 0 (18)           0.0000           0.0000  (18)           0.0000           0.0000
// Colunas utilizadas 1:

k =

     4     3     2     1
     4     3     2     1
     2     1     0     0
     4     2     1     0
     4     3     1     0
     2     1     0     0
     4     0     0     0
     3     2     1     0
     4     3     2     1
     4     3     1     0
     4     3     2     1
     4     0     0     0
     4     3     0     0
     0     0     0     0
     4     3     1     0
```

```
     4     3     1     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     4     0     0     0
     4     3     2     1
     4     3     2     1

// Problema = Dynamic, Execução 1
vip =

    0.0397    0.0163    0.0065    2.2346    0.0668

// full scan 21: tic
ell = 1, K = 5, antes = 5
ell = 2, K = 5, antes = 12
ell = 3, K = 5, antes = 12
Elapsed time is 5.117027 seconds.
// full scan 22: tic
ell = 1, K = 5, antes = 5
ell = 2, K = 5, antes = 12
ell = 3, K = 5, antes = 12
Elapsed time is 1.558072 seconds.
// Erros de treinamento e validação 2:
 4 ( 7)           0.4640           0.0014  ( 7)           0.4202           0.0056
 1 ( 5)           0.4539           0.0011  ( 5)           0.3997           0.0131
 3 (10)           0.0383           0.0000  ( 3)           0.0344           0.0002
 5 ( 3)           0.0379           0.0000  (10)           0.0343           0.0008
 2 ( 1)           0.0376           0.0001  ( 0)           0.0338           0.0009
 3 (15)           0.0375           0.0001  ( 4)           0.0338           0.0009
 4 ( 8)           0.0375           0.0000  ( 6)           0.0338           0.0009
 2 ( 2)           0.0370           0.0000  ( 9)           0.0338           0.0009
 1 ( 0)           0.0369           0.0000  (11)           0.0338           0.0009
 1 ( 4)           0.0369           0.0000  (12)           0.0338           0.0009
 1 ( 6)           0.0369           0.0000  (14)           0.0338           0.0009
 1 ( 9)           0.0369           0.0000  (16)           0.0338           0.0009
 1 (11)           0.0369           0.0000  (19)           0.0338           0.0009
 1 (12)           0.0369           0.0000  (20)           0.0338           0.0009
 1 (14)           0.0369           0.0000  (21)           0.0338           0.0009
 1 (16)           0.0369           0.0000  (22)           0.0338           0.0009
 1 (19)           0.0369           0.0000  ( 2)           0.0335           0.0000
 1 (20)           0.0369           0.0000  ( 8)           0.0328           0.0000
 1 (21)           0.0369           0.0000  (15)           0.0312           0.0002
 1 (22)           0.0369           0.0000  ( 1)           0.0311           0.0001
 0 (13)           0.0000           0.0000  (13)           0.0000           0.0000
 0 (17)           0.0000           0.0000  (17)           0.0000           0.0000
 0 (18)           0.0000           0.0000  (18)           0.0000           0.0000
// Colunas utilizadas 2:

k =

     4     0     0     0     0
     4     2     0     0     0
     4     3     0     0     0
     5     4     3     2     1
     4     0     0     0     0
     3     0     0     0     0
     4     0     0     0     0
     5     3     2     1     0
```

```
     5     4     3     2     0
     4     0     0     0     0
     5     4     1     0     0
     4     0     0     0     0
     4     0     0     0     0
     0     0     0     0     0
     4     0     0     0     0
     5     4     2     0     0
     4     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0

// Problema = DeathValley, Execução 1
vip =

  Columns 1 through 10

    1.0495    1.0437    1.0253    0.9992    0.9760    0.9588    0.9503    0.9585    0.9733    0.9964

  Columns 11 through 12

    1.0214    1.0405

// full scan 21: tic
ell = 1, K = 12, antes = 12
ell = 2, K = 40, antes = 64
ell = 3, K = 97, antes = 156
ell = 4, K = 173, antes = 318
ell = 5, K = 190, antes = 500
ell = 6, K = 189, antes = 612
ell = 7, K = 163, antes = 637
ell = 8, K = 145, antes = 640
ell = 9, K = 138, antes = 640
ell = 10, K = 137, antes = 640
ell = 11, K = 137, antes = 640
Elapsed time is 85.036309 seconds.
```

```
// full scan 22: tic
ell = 1, K = 12, antes = 12
ell = 2, K = 40, antes = 64
ell = 3, K = 97, antes = 156
ell = 4, K = 173, antes = 318
ell = 5, K = 190, antes = 500
ell = 6, K = 189, antes = 612
ell = 7, K = 163, antes = 637
ell = 8, K = 145, antes = 640
ell = 9, K = 138, antes = 640
ell = 10, K = 137, antes = 640
ell = 11, K = 137, antes = 640
Elapsed time is 35.572371 seconds.
// Erros de treinamento e validação 3:
 1 ( 2)          68.2743          0.3288  ( 2)          77.6270          2.7636
 1 ( 4)          68.2743          0.3288  ( 4)          77.6270          2.7636
 1 ( 9)          68.2743          0.3288  ( 9)          77.6270          2.7636
 1 (14)          68.2743          0.3288  (14)          77.6270          2.7636
 1 (19)          68.2743          0.3288  (19)          77.6270          2.7636
 1 (20)          68.2743          0.3288  (20)          77.6270          2.7636
 2 ( 5)          55.9940          0.4287  ( 5)          67.2582          2.9740
 5 ( 6)          49.3460          0.1806  ( 8)          65.4490          1.6978
 5 (16)          49.3460          0.1806  ( 3)          63.1930          0.3731
 3 (22)          49.2262          0.4758  ( 1)          60.4691          0.3089
 4 (11)          48.3375          0.3676  ( 7)          59.1051          0.3455
 3 ( 7)          47.1271          0.3445  (11)          58.9751          2.1062
 5 (12)          42.6447          0.0039  (15)          57.8250          1.6293
 8 ( 3)          41.6194          0.2089  ( 0)          57.7436          0.4084
10 ( 8)          41.3421          0.0793  ( 6)          57.5933          0.7088
 9 ( 0)          40.9969          0.0049  (16)          57.5933          0.7088
 9 ( 1)          40.2796          0.0867  (10)          57.5753          1.4983
11 (10)          39.4455          0.3055  (22)          52.5857          0.6790
11 (15)          38.6642          0.1933  (21)          52.0680          0.2444
 6 (21)          38.4611          0.0301  (12)          50.0154          0.5093
 0 (13)           0.0000          0.0000  (13)           0.0000          0.0000
 0 (17)           0.0000          0.0000  (17)           0.0000          0.0000
 0 (18)           0.0000          0.0000  (18)           0.0000          0.0000
// Colunas utilizadas 3:

k =
```

```
    12    11    10     7     6     4     3     2     1     0     0     0
    12    11    10     8     7     6     3     2     1     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
    12     8     7     5     4     3     2     1     0     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
     8     7     0     0     0     0     0     0     0     0     0     0
    12    10     4     2     1     0     0     0     0     0     0     0
     8     7     6     0     0     0     0     0     0     0     0     0
    12    10     8     7     6     5     4     3     2     1     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     7     6     4     3     2     1     0
    12    10     2     1     0     0     0     0     0     0     0     0
    12    11     3     2     1     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
    12    11    10     9     7     6     5     4     3     2     1     0
    12    10     4     2     1     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
    12    11     9     7     5     3     0     0     0     0     0     0
    12    10     2     0     0     0     0     0     0     0     0     0

// Problema = ArbitrNVar, Execução 1
vip =

  Columns 1 through 10

    0.9401    0.9402    0.5895    0.9702    0.9402    0.9017    1.1035    0.8182    1.0292    1.1473

  Columns 11 through 15

    1.1609    0.9775    1.0899    1.1295    1.1031

// full scan 21: tic
ell = 1, K = 15, antes = 15
ell = 2, K = 94, antes = 110
ell = 3, K = 268, antes = 400
```

```
ell = 4, K = 406, antes = 842
ell = 5, K = 529, antes = 1242
ell = 6, K = 590, antes = 1554
ell = 7, K = 587, antes = 1763
ell = 8, K = 541, antes = 1885
ell = 9, K = 507, antes = 1911
ell = 10, K = 498, antes = 1917
ell = 11, K = 498, antes = 1917
Elapsed time is 392.057926 seconds.
// full scan 22: tic
ell = 1, K = 15, antes = 15
ell = 2, K = 94, antes = 110
ell = 3, K = 268, antes = 400
ell = 4, K = 406, antes = 842
ell = 5, K = 529, antes = 1242
ell = 6, K = 590, antes = 1554
ell = 7, K = 587, antes = 1763
ell = 8, K = 541, antes = 1885
ell = 9, K = 507, antes = 1911
ell = 10, K = 498, antes = 1917
ell = 11, K = 498, antes = 1917
Elapsed time is 171.432098 seconds.
// Erros de treinamento e validação 4:
 7 (12)         404.8792          1.3563  (12)         473.3218          7.2019
14 ( 7)         404.1523          0.3863  ( 3)         406.0301          0.2135
 9 ( 8)         371.4383          3.2851  ( 8)         386.1273          5.3332
 9 ( 5)         367.4645          1.1845  ( 7)         385.0153         11.4480
 1 ( 4)         366.8722          9.1386  ( 5)         382.8309         12.7441
 1 ( 9)         366.8722          9.1386  (20)         379.6559          0.8821
 1 (14)         366.8722          9.1386  ( 0)         377.3858          8.6074
14 (15)         363.3413          2.3271  (19)         371.6085         14.6627
15 (10)         363.2612          1.7099  (22)         371.6085         14.6627
12 ( 0)         358.1534          3.8199  ( 2)         370.3506         19.7920
 9 ( 6)         354.6724          1.5015  ( 6)         359.0414          4.9162
 9 (16)         354.6724          1.5015  (11)         359.0414          4.9162
 9 (11)         354.6724          1.5015  (16)         359.0414          4.9162
 5 ( 3)         350.6869          2.9947  (10)         354.8758          1.2404
 7 (19)         349.2677          0.7424  (15)         350.3733          4.8786
 7 (22)         349.2677          0.7424  ( 1)         346.9991         45.2942
 7 (20)         344.5074          4.8810  (21)         278.2440         16.1779
```

```
 2 ( 1)          330.9073           9.2043  ( 4)          113.9404          25.4831
 3 ( 2)          327.3164           7.3517  ( 9)          113.9404          25.4831
 2 (21)          315.4428           7.9300  (14)          113.9404          25.4831
 0 (13)            0.0000           0.0000  (13)            0.0000           0.0000
 0 (17)            0.0000           0.0000  (17)            0.0000           0.0000
 0 (18)            0.0000           0.0000  (18)            0.0000           0.0000
// Colunas utilizadas 4:

k =

    15    14    13    12    11    10     9     8     7     6     4     3     0     0     0
     8     3     0     0     0     0     0     0     0     0     0     0     0     0     0
    10     8     3     0     0     0     0     0     0     0     0     0     0     0     0
    14    10     9     3     2     0     0     0     0     0     0     0     0     0     0
    10     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    12     9     8     6     5     4     3     2     1     0     0     0     0     0     0
    15    14    13    11    10     8     7     6     3     0     0     0     0     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     2     1     0
    15    14    12    11     8     7     4     3     2     0     0     0     0     0     0
    10     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     3     2     1
    15    14    13    11    10     8     7     6     3     0     0     0     0     0     0
    15    14    13    11    10     9     7     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    10     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     3     2     0
    15    14    13    11    10     8     7     6     3     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    11    10     8     3     0     0     0     0     0     0     0     0
    14    13    11    10     8     7     3     0     0     0     0     0     0     0     0
    15     3     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    11    10     8     3     0     0     0     0     0     0     0     0

// Problema = Classificacao 2, Execução 1
vip =

    1.0623    0.9911    0.9811    0.9627
```

```
// full scan 21: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.348039 seconds.
// full scan 22: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.207388 seconds.
// Erros de treinamento e validação 5:
 1 ( 7)           0.9210           0.0023  ( 7)           0.9859           0.0588
 3 ( 9)           0.7115           0.0136  ( 0)           0.9043           0.0158
 3 (19)           0.7115           0.0136  (10)           0.9043           0.0158
 3 (20)           0.7115           0.0136  (22)           0.9043           0.0158
 3 ( 5)           0.6898           0.0025  ( 3)           0.8501           0.0380
 1 ( 3)           0.6840           0.0053  (12)           0.8501           0.0380
 1 (12)           0.6840           0.0053  ( 2)           0.8473           0.0189
 2 ( 6)           0.6752           0.0100  ( 4)           0.8473           0.0189
 2 (16)           0.6752           0.0100  ( 8)           0.8473           0.0189
 4 ( 0)           0.6575           0.0200  (14)           0.8473           0.0189
 4 (10)           0.6575           0.0200  (15)           0.8473           0.0189
 4 (22)           0.6575           0.0200  (21)           0.8473           0.0189
 2 (11)           0.6355           0.0014  ( 5)           0.7794           0.0237
 3 ( 2)           0.6202           0.0044  ( 9)           0.7155           0.0164
 3 ( 4)           0.6202           0.0044  (19)           0.7155           0.0164
 3 ( 8)           0.6202           0.0044  (20)           0.7155           0.0164
 3 (14)           0.6202           0.0044  ( 6)           0.7064           0.0593
 3 (15)           0.6202           0.0044  (16)           0.7064           0.0593
 3 (21)           0.6202           0.0044  (11)           0.6594           0.0292
 0 ( 1)           0.0000           0.0000  ( 1)           0.5020           0.0000
 0 (13)           0.0000           0.0000  (13)           0.0000           0.0000
 0 (17)           0.0000           0.0000  (17)           0.0000           0.0000
 0 (18)           0.0000           0.0000  (18)           0.0000           0.0000
// Colunas utilizadas 5:

k =

     4     3     2     1
     0     0     0     0
     4     3     1     0
     1     0     0     0
     4     3     1     0
     3     2     1     0
     4     1     0     0
     2     0     0     0
     4     3     1     0
     4     3     2     0
     4     3     2     1
     4     3     0     0
     1     0     0     0
     0     0     0     0
     4     3     1     0
     4     3     1     0
     4     1     0     0
     0     0     0     0
     0     0     0     0
     4     3     2     0
```

```
     4     3     2     0
     4     3     1     0
     4     3     2     1

// Problema = Classificacao 5, Execução 1
vip =

    1.0848    0.9883    0.9579    0.9638

// full scan 21: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.335234 seconds.
// full scan 22: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.191632 seconds.
// Erros de treinamento e validação 6:
 1 ( 7)           1.9835           0.1015  ( 4)           2.3038           0.1343
 2 ( 3)           1.9054           0.0015  ( 6)           2.3038           0.1343
 2 ( 2)           1.8748           0.0473  (14)           2.3038           0.1343
 1 (12)           1.7899           0.0019  (15)           2.3038           0.1343
 3 ( 8)           1.7517           0.0022  (16)           2.3038           0.1343
 3 ( 4)           1.7453           0.0119  ( 9)           2.3038           0.1343
 3 ( 6)           1.7453           0.0119  (11)           2.3038           0.1343
 3 (14)           1.7453           0.0119  (19)           2.3038           0.1343
 3 (15)           1.7453           0.0119  (20)           2.3038           0.1343
 3 (16)           1.7453           0.0119  (21)           2.3038           0.1343
 3 ( 9)           1.7453           0.0119  (12)           2.1812           0.1869
 3 (11)           1.7453           0.0119  ( 7)           2.1517           0.1140
 3 (19)           1.7453           0.0119  ( 5)           1.9933           0.0991
 3 (20)           1.7453           0.0119  ( 0)           1.9933           0.0991
 3 (21)           1.7453           0.0119  (10)           1.9933           0.0991
 4 ( 0)           1.7139           0.0236  (22)           1.9933           0.0991
 4 ( 5)           1.7139           0.0236  ( 3)           1.9219           0.1582
 4 (10)           1.7139           0.0236  ( 2)           1.7161           0.2214
 4 (22)           1.7139           0.0236  ( 8)           1.7029           0.0534
 0 ( 1)           0.0000           0.0000  ( 1)           1.2001           0.0000
 0 (13)           0.0000           0.0000  (13)           0.0000           0.0000
 0 (17)           0.0000           0.0000  (17)           0.0000           0.0000
 0 (18)           0.0000           0.0000  (18)           0.0000           0.0000
// Colunas utilizadas 6:

k =

     4     3     2     1
     0     0     0     0
     4     3     0     0
     4     1     0     0
     4     2     1     0
     4     3     2     1
     4     2     1     0
     3     0     0     0
     4     3     1     0
     4     2     1     0
     4     3     2     1
```

4	2	1	0
1	0	0	0
0	0	0	0
4	2	1	0
4	2	1	0
4	2	1	0
0	0	0	0
0	0	0	0
4	2	1	0
4	2	1	0
4	2	1	0
4	3	2	1

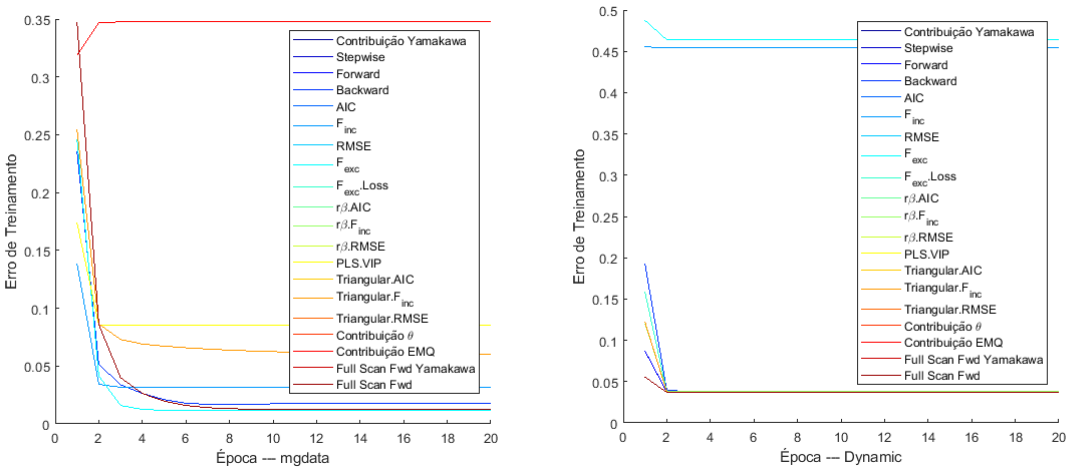

Figura 4: (1) mgdata (2) Dynamic

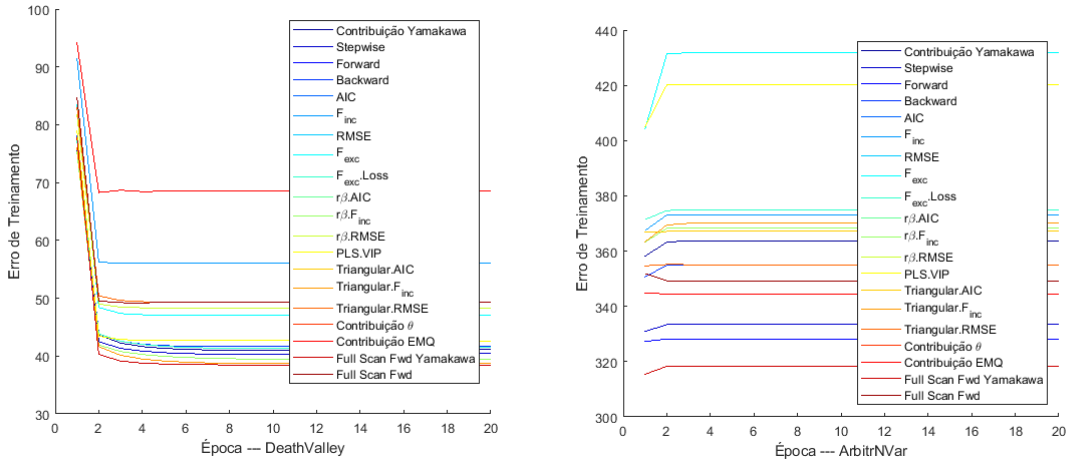

Figura 5: (1) DeathValley (2) ArbitrNVar 15

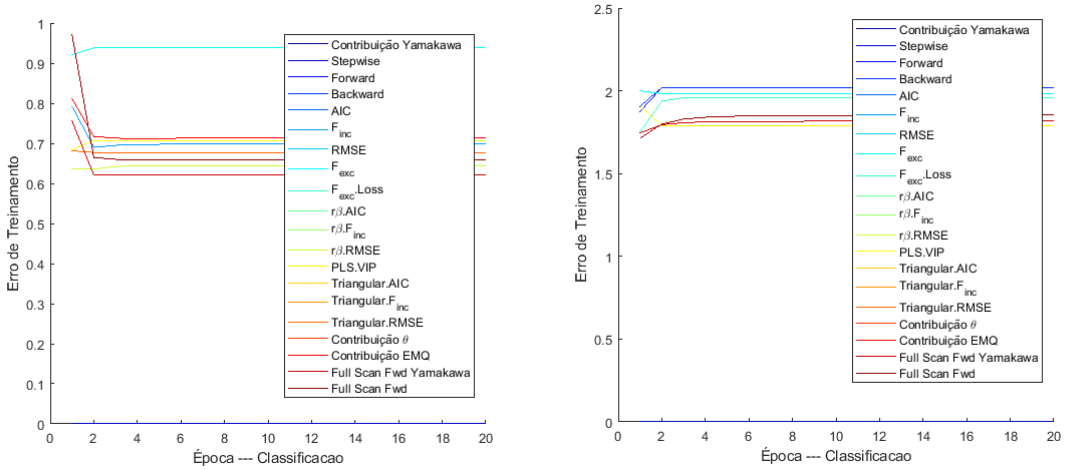

Figura 6: (1) Classificação Binária (2) 5 Classes

8.3 Execução 3

Quatro algoritmos PLS foram acrescentados.

Comparamos 0 = contribuição yamakawa; 1 = stepwise; 2 = forward; 3 = backward; 4 = AIC; 5 = Finc ; 6 = RMSE ; 7 = Fexc ; 8 = FexcLoss ; 9 = rβ.AIC ; 10 = rβ.Finc ; 11 = rβ.RMSE; 12 = PLS.VIP ; 13 = PLS.β ; 14 = PLS.AIC; 15 = PLS.Finc ; 16 = PLS.RMSE ; 17 = Triangular.AIC ; 18 = Triangular.Finc ; 19 = Triangular.RMSE ; 22 = Contribuição θ ; 23 = Contribuição MQ ; 24 = full scan fwd Yamakawa ; 25 = full scan fwd.

Aqui a flagEMQ é verdadeira sempre, exceto no Algoritmo $=23$.

```
// Problema = mgdata, Execução 1
vip =

    0.9620    0.9757    1.0103    1.0496

// full scan 24: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 6, antes = 10
ell = 3, K = 6, antes = 13
ell = 4, K = 6, antes = 14
Elapsed time is 1.814979 seconds.
// full scan 25: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 6, antes = 10
ell = 3, K = 6, antes = 13
ell = 4, K = 6, antes = 14
Elapsed time is 1.057037 seconds.
// Erros de treinamento e validação 1:
 1 ( 6)           0.3187           0.0029  ( 6)           0.6776           0.0099
 1 (11)           0.3187           0.0029  (11)           0.6776           0.0099
 1 (14)           0.3187           0.0029  (14)           0.6776           0.0099
 1 (15)           0.3187           0.0029  (15)           0.6776           0.0099
 1 (16)           0.3187           0.0029  (16)           0.6776           0.0099
 1 (19)           0.3187           0.0029  (19)           0.6776           0.0099
 1 (22)           0.3187           0.0029  (22)           0.6776           0.0099
 1 (23)           0.3187           0.0029  (23)           0.6776           0.0099
 2 (12)           0.0849           0.0019  (12)           0.1856           0.0000
 3 ( 9)           0.0598           0.0015  ( 4)           0.1042           0.0016
 3 ( 4)           0.0598           0.0015  (17)           0.1042           0.0016
 3 (17)           0.0598           0.0015  (18)           0.1042           0.0016
 3 (18)           0.0598           0.0015  ( 9)           0.1042           0.0016
```

```
 2 ( 2)           0.0317           0.0007  ( 2)           0.0709           0.0005
 2 ( 5)           0.0317           0.0007  ( 5)           0.0709           0.0005
 3 ( 8)           0.0238           0.0010  ( 8)           0.0412           0.0004
 4 ( 0)           0.0125           0.0003  ( 0)           0.0221           0.0002
 4 ( 1)           0.0125           0.0003  ( 1)           0.0221           0.0002
 4 ( 3)           0.0125           0.0003  ( 3)           0.0221           0.0002
 4 (10)           0.0125           0.0003  (10)           0.0221           0.0002
 4 (13)           0.0125           0.0003  (13)           0.0221           0.0002
 4 (24)           0.0125           0.0003  (24)           0.0221           0.0002
 4 (25)           0.0125           0.0003  (25)           0.0221           0.0002
 3 ( 7)           0.0114           0.0004  ( 7)           0.0185           0.0002
 0 (20)           0.0000           0.0000  (20)           0.0000           0.0000
 0 (21)           0.0000           0.0000  (21)           0.0000           0.0000
// Colunas utilizadas 1:

k =

     4     3     2     1
     4     3     2     1
     2     1     0     0
     4     3     2     1
     4     3     1     0
     2     1     0     0
     4     0     0     0
     3     2     1     0
     4     3     2     0
     4     3     1     0
     4     3     2     1
     4     0     0     0
     4     3     0     0
     4     3     2     1
     4     0     0     0
     4     0     0     0
     4     0     0     0
     4     3     1     0
     4     3     1     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     4     0     0     0
     4     3     2     1
     4     3     2     1

// Problema = Dynamic, Execução 1
vip =

    0.0223    0.0050    0.0042    2.2357    0.0345

// full scan 24: tic
ell = 1, K = 5, antes = 5
ell = 2, K = 5, antes = 11
ell = 3, K = 4, antes = 14
ell = 4, K = 5, antes = 14
ell = 5, K = 4, antes = 14
Elapsed time is 4.672466 seconds.
// full scan 25: tic
ell = 1, K = 5, antes = 5
ell = 2, K = 5, antes = 11
ell = 3, K = 4, antes = 14
```

```
ell = 4, K = 5, antes = 14
ell = 5, K = 4, antes = 14
Elapsed time is 2.178771 seconds.
// Erros de treinamento e validação 2:
 1 ( 5)           0.6497           0.0016  ( 5)           0.5701           0.0009
 1 (14)           0.5298           0.0012  (14)           0.5681           0.0313
 4 ( 7)           0.4646           0.0004  ( 7)           0.4884           0.0090
 3 (15)           0.0435           0.0001  ( 2)           0.0519           0.0021
 2 ( 2)           0.0434           0.0000  (15)           0.0479           0.0026
 2 (16)           0.0418           0.0001  (16)           0.0418           0.0010
 4 ( 3)           0.0399           0.0000  ( 0)           0.0385           0.0004
 1 ( 0)           0.0398           0.0000  ( 4)           0.0385           0.0004
 1 ( 4)           0.0398           0.0000  ( 6)           0.0385           0.0004
 1 ( 6)           0.0398           0.0000  ( 9)           0.0385           0.0004
 1 ( 9)           0.0398           0.0000  (11)           0.0385           0.0004
 1 (11)           0.0398           0.0000  (12)           0.0385           0.0004
 1 (12)           0.0398           0.0000  (13)           0.0385           0.0004
 1 (13)           0.0398           0.0000  (17)           0.0385           0.0004
 1 (17)           0.0398           0.0000  (19)           0.0385           0.0004
 1 (19)           0.0398           0.0000  (22)           0.0385           0.0004
 1 (22)           0.0398           0.0000  (23)           0.0385           0.0004
 1 (23)           0.0398           0.0000  (24)           0.0385           0.0004
 1 (24)           0.0398           0.0000  (25)           0.0385           0.0004
 1 (25)           0.0398           0.0000  ( 3)           0.0384           0.0020
 5 ( 8)           0.0387           0.0000  ( 8)           0.0364           0.0000
 2 (10)           0.0380           0.0000  (10)           0.0353           0.0011
 2 ( 1)           0.0372           0.0001  (18)           0.0323           0.0005
 2 (18)           0.0372           0.0001  ( 1)           0.0323           0.0005
 0 (20)           0.0000           0.0000  (20)           0.0000           0.0000
 0 (21)           0.0000           0.0000  (21)           0.0000           0.0000
// Colunas utilizadas 2:

k =

     4     0     0     0     0
     4     2     0     0     0
     4     3     0     0     0
     5     4     3     2     0
     4     0     0     0     0
     3     0     0     0     0
     4     0     0     0     0
     5     3     2     1     0
     5     4     3     2     1
     4     0     0     0     0
     4     1     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     5     0     0     0     0
     5     4     3     0     0
     5     4     0     0     0
     4     0     0     0     0
     4     2     0     0     0
     4     0     0     0     0
     0     0     0     0     0
```

```
     0     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0

// Problema = DeathValley, Execução 1
vip =

  Columns 1 through 9

    1.0495    1.0383    1.0163    0.9920    0.9685    0.9552    0.9519    0.9597    0.9798

  Columns 10 through 12

    1.0054    1.0299    1.0464

// full scan 24: tic
ell = 1, K = 12, antes = 12
ell = 2, K = 40, antes = 64
ell = 3, K = 92, antes = 150
ell = 4, K = 163, antes = 297
ell = 5, K = 194, antes = 474
ell = 6, K = 194, antes = 569
ell = 7, K = 170, antes = 592
ell = 8, K = 151, antes = 596
ell = 9, K = 143, antes = 596
ell = 10, K = 141, antes = 596
ell = 11, K = 140, antes = 596
ell = 12, K = 140, antes = 596
Elapsed time is 82.508792 seconds.
// full scan 25: tic
ell = 1, K = 12, antes = 12
ell = 2, K = 40, antes = 64
ell = 3, K = 92, antes = 150
ell = 4, K = 163, antes = 297
ell = 5, K = 194, antes = 474
ell = 6, K = 194, antes = 569
ell = 7, K = 170, antes = 592
ell = 8, K = 151, antes = 596
```

```
ell = 9, K = 143, antes = 596
ell = 10, K = 141, antes = 596
ell = 11, K = 140, antes = 596
ell = 12, K = 140, antes = 596
Elapsed time is 36.049906 seconds.
// Erros de treinamento e validação 3:
 1 (14)          123.3158           0.3679  (14)          123.1287           3.9888
 1 ( 2)           71.4776           0.5399  ( 5)           76.1442           0.2120
 1 ( 4)           71.4776           0.5399  ( 2)           69.2545           8.4286
 1 ( 9)           71.4776           0.5399  ( 4)           69.2545           8.4286
 1 (17)           71.4776           0.5399  ( 9)           69.2545           8.4286
 2 ( 5)           61.7667           0.3542  (17)           69.2545           8.4286
 3 (22)           50.0846           0.1241  ( 7)           68.8249           1.4181
 3 (23)           50.0846           0.1241  (10)           61.5358           0.4991
 4 ( 6)           49.1405           0.0170  (13)           61.5358           0.4991
 4 (19)           49.1405           0.0170  (16)           61.5358           0.4991
 4 (11)           49.1405           0.0170  (15)           61.2781           0.0581
 4 (25)           48.1483           0.1964  ( 3)           59.4608           0.0555
 3 ( 7)           48.1367           0.3670  (18)           58.2558           1.0087
 6 (12)           46.5700           0.0642  ( 0)           54.6112           1.0400
10 ( 3)           46.3948           0.0632  ( 8)           54.3003           0.9623
 9 ( 1)           44.9382           0.2562  ( 1)           53.2446           1.2648
 9 ( 8)           44.6911           0.0917  (24)           52.7255           1.5513
12 (10)           44.3767           0.0179  (12)           47.9542           0.9375
12 (13)           44.3767           0.0179  (25)           47.7618           1.8335
12 (16)           44.3767           0.0179  ( 6)           46.8883           2.0829
 8 ( 0)           43.9617           0.0288  (19)           46.8883           2.0829
11 (18)           43.8822           0.0196  (11)           46.8883           2.0829
10 (15)           43.7845           0.1084  (22)           45.8429           1.0666
 5 (24)           42.7979           0.0635  (23)           45.8429           1.0666
 0 (20)            0.0000           0.0000  (20)            0.0000           0.0000
 0 (21)            0.0000           0.0000  (21)            0.0000           0.0000
// Colunas utilizadas 3:

k =

    12    10     9     7     6     3     2     1     0     0     0     0
    12    11     9     8     7     6     3     2     1     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
    12    11     8     7     6     5     4     3     2     1     0     0
```

```
     1     0     0     0     0     0     0     0     0     0     0     0
     8     7     0     0     0     0     0     0     0     0     0     0
    12     9     2     1     0     0     0     0     0     0     0     0
     8     7     6     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     6     5     4     2     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     7     6     5     4     3     2     1
    12     9     2     1     0     0     0     0     0     0     0     0
    12    11    10     3     2     1     0     0     0     0     0     0
    12    11    10     9     8     7     6     5     4     3     2     1
    12     0     0     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     7     6     5     4     3     0     0
    12    11    10     9     8     7     6     5     4     3     2     1
     1     0     0     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     7     6     4     3     2     1     0
    12     9     2     1     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
    12     2     1     0     0     0     0     0     0     0     0     0
    12     2     1     0     0     0     0     0     0     0     0     0
    12    10     7     6     3     0     0     0     0     0     0     0
    12     9     8     2     0     0     0     0     0     0     0     0

// Problema = ArbitrNVar, Execução 1
vip =

  Columns 1 through 9

    1.0348    0.7196    0.9976    0.6274    1.1179    0.6218    0.9799    1.1580    0.7345

  Columns 10 through 15

    0.8968    1.1451    1.3955    0.9112    1.0892    1.2107

// full scan 24: tic
ell = 1, K = 15, antes = 15
ell = 2, K = 67, antes = 85
ell = 3, K = 140, antes = 211
ell = 4, K = 184, antes = 337
ell = 5, K = 190, antes = 408
```

```
ell = 6, K = 180, antes = 425
ell = 7, K = 177, antes = 428
ell = 8, K = 177, antes = 428
Elapsed time is 78.944217 seconds.
// full scan 25: tic
ell = 1, K = 15, antes = 15
ell = 2, K = 67, antes = 85
ell = 3, K = 140, antes = 211
ell = 4, K = 184, antes = 337
ell = 5, K = 190, antes = 408
ell = 6, K = 180, antes = 425
ell = 7, K = 177, antes = 428
ell = 8, K = 177, antes = 428
Elapsed time is 21.527466 seconds.
// Erros de treinamento e validação 4:
 1 ( 4)         891.5851          37.8129  ( 8)        3224.2064          13.5681
 1 ( 9)         891.5851          37.8129  (15)        3204.0502           7.3531
 1 (17)         891.5851          37.8129  (12)        3201.4556           4.6096
 2 (16)         880.2223          26.7201  (14)        3200.7770           0.1319
 5 ( 2)         870.0335          10.5799  (24)        3199.4983           1.1766
 5 ( 3)         852.6910           1.7292  ( 4)        3197.8894           0.9412
 7 (12)         823.1949           5.9570  ( 9)        3197.8894           0.9412
 7 (15)         812.3333           6.5236  (17)        3197.8894           0.9412
 4 ( 5)         803.7862          14.6813  (16)        3195.9956           1.6951
 5 ( 1)         793.0366          10.8386  ( 2)        3185.2451           5.2903
13 ( 0)         789.4013          15.8062  ( 5)        3178.9046          14.0258
14 ( 7)         786.5503          16.1556  (25)        3175.0893          17.0701
 5 (25)         783.5387           2.7272  ( 1)        3168.6826          24.3270
11 ( 8)         782.9160          11.3734  ( 3)        3164.0394          17.4311
14 (18)         782.2439          12.4705  (22)        3156.7095          25.2646
11 ( 6)         782.2277          13.0243  (23)        3156.7095          25.2646
11 (11)         782.2277          13.0243  ( 6)        3146.7407          26.3036
11 (19)         782.2277          13.0243  (11)        3146.7407          26.3036
15 (10)         775.1570          13.3594  (19)        3146.7407          26.3036
15 (13)         775.1570          13.3594  (18)        3138.2416          35.3005
 1 (14)         773.3489          30.4103  (10)        3134.9189          30.6668
 9 (22)         767.3643           8.1469  (13)        3134.9189          30.6668
 9 (23)         767.3643           8.1469  ( 7)        3130.1798          24.3081
 2 (24)         737.9084          13.8729  ( 0)        3123.7100          42.7450
 0 (20)           0.0000           0.0000  (20)           0.0000           0.0000
```

```
 0 (21)           0.0000           0.0000  (21)           0.0000           0.0000
// Colunas utilizadas 4:

k =

    15    14    13    12    11     9     8     7     6     5     4     3     1     0     0
    12     9     6     4     2     0     0     0     0     0     0     0     0     0     0
    14    10     7     2     1     0     0     0     0     0     0     0     0     0     0
    14    13     9     5     4     0     0     0     0     0     0     0     0     0     0
    12     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     9     6     4     2     0     0     0     0     0     0     0     0     0     0     0
    15    14    12    11     9     8     7     6     5     4     2     0     0     0     0
    15    14    13    11    10     9     8     7     6     5     4     3     2     1     0
    15    14    13    11    10     9     8     6     5     3     2     0     0     0     0
    12     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     3     2     1
    15    14    12    11     9     8     7     6     5     4     2     0     0     0     0
    15    14    12    11     8     5     1     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     3     2     1
    15     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     0     0     0     0     0     0     0     0
    15    14     0     0     0     0     0     0     0     0     0     0     0     0     0
    12     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     2     1     0
    15    14    12    11     9     8     7     6     5     4     2     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    12    11     9     8     6     5     4     2     0     0     0     0     0     0
    15    12    11     9     8     6     5     4     2     0     0     0     0     0     0
    15     5     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    12    11     9     4     0     0     0     0     0     0     0     0     0     0

// Problema = Classificacao 2, Execução 1
vip =

    0.9855    1.0206    0.9397    1.0508

// full scan 24: tic
ell = 1, K = 4, antes = 4
```

```
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.287643 seconds.
// full scan 25: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.201427 seconds.
// Erros de treinamento e validação 5:
 1 (14)           0.9287          0.0365  (24)           0.8614          0.0006
 1 (15)           0.9287          0.0365  ( 7)           0.8004          0.0069
 1 (16)           0.9287          0.0365  ( 8)           0.8004          0.0069
 3 (18)           0.8723          0.0109  (14)           0.7950          0.0115
 2 ( 4)           0.8721          0.0135  (15)           0.7950          0.0115
 2 ( 6)           0.8721          0.0135  (16)           0.7950          0.0115
 2 ( 9)           0.8721          0.0135  ( 2)           0.7904          0.0651
 2 (11)           0.8721          0.0135  ( 0)           0.7787          0.0712
 2 (12)           0.8721          0.0135  ( 5)           0.7787          0.0712
 2 (17)           0.8721          0.0135  (10)           0.7787          0.0712
 2 (19)           0.8721          0.0135  (13)           0.7787          0.0712
 2 (22)           0.8721          0.0135  (25)           0.7787          0.0712
 2 (23)           0.8721          0.0135  ( 3)           0.7605          0.0051
 3 ( 2)           0.7607          0.0048  ( 1)           0.5000          0.0000
 4 ( 0)           0.7558          0.0132  (18)           0.4955          0.0133
 4 ( 5)           0.7558          0.0132  ( 4)           0.3938          0.0476
 4 (10)           0.7558          0.0132  ( 6)           0.3938          0.0476
 4 (13)           0.7558          0.0132  (17)           0.3938          0.0476
 4 (25)           0.7558          0.0132  (19)           0.3938          0.0476
 3 ( 3)           0.7387          0.0120  ( 9)           0.3938          0.0476
 1 ( 7)           0.6816          0.0084  (11)           0.3938          0.0476
 1 ( 8)           0.6816          0.0084  (12)           0.3938          0.0476
 2 (24)           0.6757          0.0081  (22)           0.3938          0.0476
 0 ( 1)           0.0000          0.0000  (23)           0.3938          0.0476
 0 (20)           0.0000          0.0000  (20)           0.0000          0.0000
 0 (21)           0.0000          0.0000  (21)           0.0000          0.0000
// Colunas utilizadas 5:

k =

     4     3     2     1
     0     0     0     0
     3     2     1     0
     4     3     1     0
     4     2     0     0
     4     3     2     1
     4     2     0     0
     3     0     0     0
     3     0     0     0
     4     2     0     0
     4     3     2     1
     4     2     0     0
     4     2     0     0
     4     3     2     1
     4     0     0     0
     4     0     0     0
     4     0     0     0
     4     2     0     0
     4     2     1     0
```

```
     4     2     0     0
     0     0     0     0
     0     0     0     0
     4     2     0     0
     4     2     0     0
     4     3     0     0
     4     3     2     1

// Problema = Classificacao 5, Execução 1
vip =

    0.8067    1.0287    1.0493    1.0910

// full scan 24: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.374757 seconds.
// full scan 25: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.185555 seconds.
// Erros de treinamento e validação 6:
 1 ( 3)           2.2863           0.0296  ( 3)           2.9515           0.3755
 1 (14)           1.9472           0.0527  (14)           2.6240           0.0887
 1 (15)           1.9472           0.0527  (15)           2.6240           0.0887
 2 ( 4)           1.8798           0.0146  ( 4)           2.6024           0.0633
 2 ( 6)           1.8798           0.0146  ( 6)           2.6024           0.0633
 2 ( 9)           1.8798           0.0146  ( 9)           2.6024           0.0633
 2 (11)           1.8798           0.0146  (11)           2.6024           0.0633
 2 (16)           1.8798           0.0146  (16)           2.6024           0.0633
 2 (17)           1.8798           0.0146  (17)           2.6024           0.0633
 2 (19)           1.8798           0.0146  (19)           2.6024           0.0633
 2 (22)           1.8798           0.0146  (22)           2.6024           0.0633
 2 (23)           1.8798           0.0146  (23)           2.6024           0.0633
 3 (12)           1.7016           0.0411  (12)           1.8144           0.1014
 3 (18)           1.7016           0.0411  (18)           1.8144           0.1014
 2 ( 5)           1.6766           0.0283  ( 7)           1.6873           0.1446
 3 ( 7)           1.6761           0.0363  ( 8)           1.6873           0.1446
 3 ( 8)           1.6761           0.0363  ( 0)           1.6217           0.0338
 2 ( 2)           1.6554           0.0067  (10)           1.6217           0.0338
 4 ( 0)           1.5513           0.0028  (13)           1.6217           0.0338
 4 (10)           1.5513           0.0028  (24)           1.6217           0.0338
 4 (13)           1.5513           0.0028  (25)           1.6217           0.0338
 4 (24)           1.5513           0.0028  ( 2)           1.5948           0.0287
 4 (25)           1.5513           0.0028  ( 1)           1.5847           0.0000
 0 ( 1)           0.0000           0.0000  ( 5)           1.4510           0.1396
 0 (20)           0.0000           0.0000  (20)           0.0000           0.0000
 0 (21)           0.0000           0.0000  (21)           0.0000           0.0000
// Colunas utilizadas 6:

k =

     4     3     2     1
     0     0     0     0
     4     1     0     0
     3     0     0     0
```

4	3	0	0
2	1	0	0
4	3	0	0
3	2	1	0
3	2	1	0
4	3	0	0
4	3	2	1
4	3	0	0
4	3	2	0
4	3	2	1
4	0	0	0
4	0	0	0
4	3	0	0
4	3	0	0
4	3	2	0
4	3	0	0
0	0	0	0
0	0	0	0
4	3	0	0
4	3	0	0
4	3	2	1
4	3	2	1

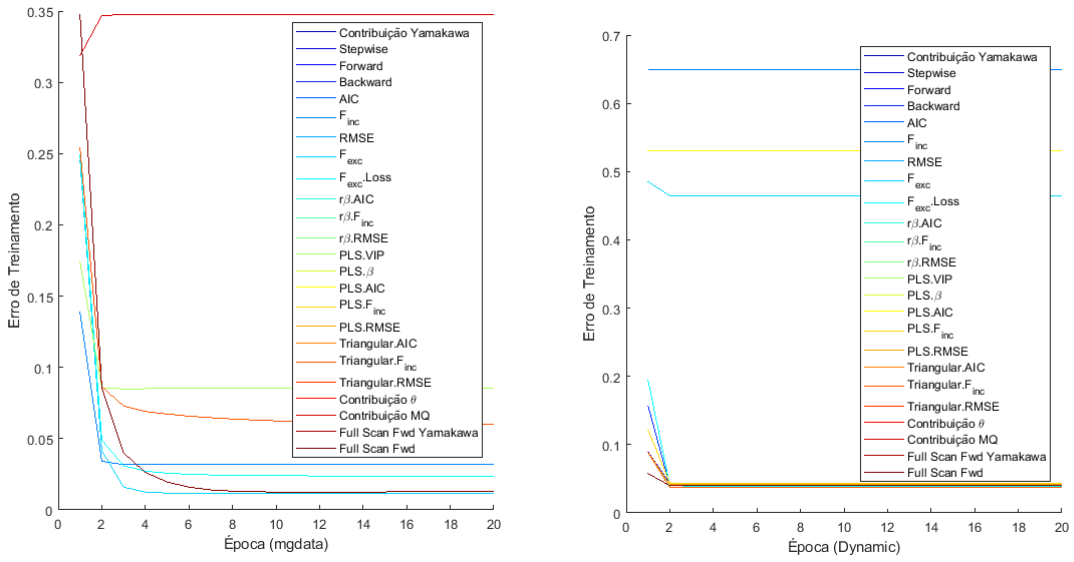

Figura 7: (1) mgdata (2) Dynamic

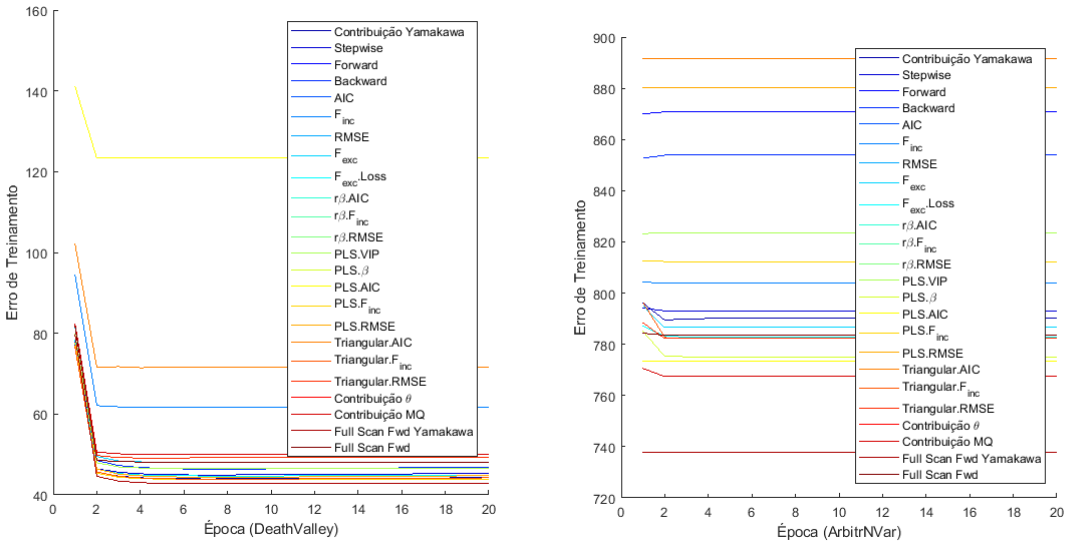

Figura 8: (1) DeathValley (2) ArbitrNVar 15

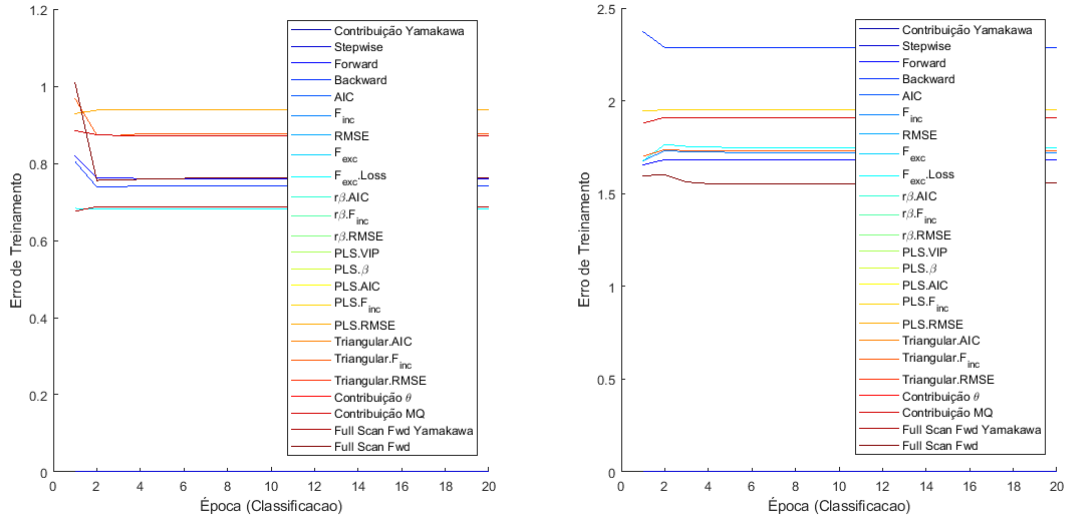

Figura 9: (1) Classificação Binária (2) 5 Classes

8.4 Execução 4

Aqui a flagEMQ é falsa sempre, exceto no Algoritmo = 23.

```
// Problema = mgdata, Execução 1
vip =

    0.9620    0.9757    1.0103    1.0496

// full scan 24: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 6, antes = 10
ell = 3, K = 6, antes = 13
ell = 4, K = 6, antes = 14
Elapsed time is 1.669426 seconds.
// full scan 25: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 6, antes = 10
ell = 3, K = 6, antes = 13
ell = 4, K = 6, antes = 14
```

```
Elapsed time is 0.921502 seconds.
// Erros de treinamento e validação 1:
 1 ( 6)           0.3187           0.0029  ( 6)           0.6776           0.0099
 1 (11)           0.3187           0.0029  (11)           0.6776           0.0099
 1 (14)           0.3187           0.0029  (14)           0.6776           0.0099
 1 (15)           0.3187           0.0029  (15)           0.6776           0.0099
 1 (16)           0.3187           0.0029  (16)           0.6776           0.0099
 1 (19)           0.3187           0.0029  (19)           0.6776           0.0099
 1 (22)           0.3187           0.0029  (22)           0.6776           0.0099
 1 (23)           0.3187           0.0029  (23)           0.6776           0.0099
 2 (12)           0.0849           0.0019  (12)           0.1856           0.0000
 3 ( 9)           0.0598           0.0015  ( 4)           0.1042           0.0016
 3 ( 4)           0.0598           0.0015  (17)           0.1042           0.0016
 3 (17)           0.0598           0.0015  (18)           0.1042           0.0016
 3 (18)           0.0598           0.0015  ( 9)           0.1042           0.0016
 2 ( 2)           0.0317           0.0007  ( 2)           0.0709           0.0005
 2 ( 5)           0.0317           0.0007  ( 5)           0.0709           0.0005
 3 ( 8)           0.0168           0.0005  ( 8)           0.0351           0.0008
 4 ( 0)           0.0125           0.0003  ( 0)           0.0221           0.0002
 4 ( 1)           0.0125           0.0003  ( 1)           0.0221           0.0002
 4 ( 3)           0.0125           0.0003  ( 3)           0.0221           0.0002
 4 (10)           0.0125           0.0003  (10)           0.0221           0.0002
 4 (13)           0.0125           0.0003  (13)           0.0221           0.0002
 4 (24)           0.0125           0.0003  (24)           0.0221           0.0002
 4 (25)           0.0125           0.0003  (25)           0.0221           0.0002
 3 ( 7)           0.0114           0.0004  ( 7)           0.0185           0.0002
 0 (20)           0.0000           0.0000  (20)           0.0000           0.0000
 0 (21)           0.0000           0.0000  (21)           0.0000           0.0000
// Colunas utilizadas 1:

k =

     4     3     2     1
     4     3     2     1
     2     1     0     0
     4     3     2     1
     4     3     1     0
     2     1     0     0
     4     0     0     0
     3     2     1     0
     4     2     1     0
     4     3     1     0
     4     3     2     1
     4     0     0     0
     4     3     0     0
     4     3     2     1
     4     0     0     0
     4     0     0     0
     4     0     0     0
     4     3     1     0
     4     3     1     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     4     0     0     0
     4     3     2     1
     4     3     2     1

// Problema = Dynamic, Execução 1
```

```
vip =

    0.0237    0.0380    0.0384    2.2350    0.0378

// full scan 24: tic
ell = 1, K = 5, antes = 5
ell = 2, K = 5, antes = 11
ell = 3, K = 3, antes = 15
ell = 4, K = 2, antes = 16
ell = 5, K = 2, antes = 16
Elapsed time is 1.654326 seconds.
// full scan 25: tic
ell = 1, K = 5, antes = 5
ell = 2, K = 5, antes = 11
ell = 3, K = 3, antes = 15
ell = 4, K = 2, antes = 16
ell = 5, K = 2, antes = 16
Elapsed time is 0.653106 seconds.
// Erros de treinamento e validação 2:
 1 (14)           0.4666           0.0004  (14)           0.4707           0.0077
 4 ( 7)           0.4414           0.0006  ( 7)           0.4289           0.0047
 2 ( 5)           0.4395           0.0006  ( 5)           0.4151           0.0137
 1 ( 0)           0.0462           0.0004  (16)           0.0437           0.0003
 1 ( 4)           0.0462           0.0004  (15)           0.0409           0.0000
 1 ( 6)           0.0462           0.0004  ( 0)           0.0393           0.0018
 1 ( 9)           0.0462           0.0004  ( 4)           0.0393           0.0018
 1 (11)           0.0462           0.0004  ( 6)           0.0393           0.0018
 1 (12)           0.0462           0.0004  ( 9)           0.0393           0.0018
 1 (13)           0.0462           0.0004  (11)           0.0393           0.0018
 1 (17)           0.0462           0.0004  (12)           0.0393           0.0018
 1 (19)           0.0462           0.0004  (13)           0.0393           0.0018
 1 (22)           0.0462           0.0004  (17)           0.0393           0.0018
 1 (23)           0.0462           0.0004  (19)           0.0393           0.0018
 1 (24)           0.0462           0.0004  (22)           0.0393           0.0018
 1 (25)           0.0462           0.0004  (23)           0.0393           0.0018
 2 (16)           0.0428           0.0001  (24)           0.0393           0.0018
 2 ( 2)           0.0419           0.0003  (25)           0.0393           0.0018
 3 (15)           0.0416           0.0003  (10)           0.0388           0.0005
 4 ( 3)           0.0390           0.0004  ( 3)           0.0385           0.0003
 4 ( 8)           0.0390           0.0004  ( 8)           0.0385           0.0003
 2 ( 1)           0.0387           0.0001  ( 1)           0.0385           0.0004
 3 (18)           0.0386           0.0004  (18)           0.0380           0.0004
 3 (10)           0.0376           0.0000  ( 2)           0.0370           0.0013
 0 (20)           0.0000           0.0000  (20)           0.0000           0.0000
 0 (21)           0.0000           0.0000  (21)           0.0000           0.0000
// Colunas utilizadas 2:

k =

     4     0     0     0     0
     4     2     0     0     0
     4     3     0     0     0
     5     4     3     2     0
     4     0     0     0     0
     5     1     0     0     0
     4     0     0     0     0
     5     3     2     1     0
```

```
     5     4     3     2     0
     4     0     0     0     0
     4     2     1     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     5     0     0     0     0
     5     4     3     0     0
     5     4     0     0     0
     4     0     0     0     0
     4     3     2     0     0
     4     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0

// Problema = DeathValley, Execução 1
vip =

  Columns 1 through 9

    1.0508    1.0436    1.0245    0.9966    0.9739    0.9576    0.9515    0.9568    0.9734

  Columns 10 through 12

    0.9960    1.0236    1.0443

// full scan 24: tic
ell = 1, K = 12, antes = 12
ell = 2, K = 49, antes = 73
ell = 3, K = 132, antes = 193
ell = 4, K = 238, antes = 409
ell = 5, K = 267, antes = 664
ell = 6, K = 254, antes = 797
ell = 7, K = 217, antes = 823
ell = 8, K = 197, antes = 827
ell = 9, K = 187, antes = 829
```

```
ell = 10, K = 184, antes = 829
ell = 11, K = 184, antes = 829
Elapsed time is 85.192755 seconds.
// full scan 25: tic
ell = 1, K = 12, antes = 12
ell = 2, K = 49, antes = 73
ell = 3, K = 132, antes = 193
ell = 4, K = 238, antes = 409
ell = 5, K = 267, antes = 664
ell = 6, K = 254, antes = 797
ell = 7, K = 217, antes = 823
ell = 8, K = 197, antes = 827
ell = 9, K = 187, antes = 829
ell = 10, K = 184, antes = 829
ell = 11, K = 184, antes = 829
Elapsed time is 22.439204 seconds.
// Erros de treinamento e validação 3:
 1 (14)         168.4009          1.9388  (14)         179.4844          19.9064
 1 ( 4)          74.3648          0.2830  ( 4)          69.7539          10.9025
 1 ( 9)          74.3648          0.2830  ( 9)          69.7539          10.9025
 1 (17)          74.3648          0.2830  (17)          69.7539          10.9025
 1 (23)          74.3648          0.2830  (23)          69.7539          10.9025
 2 ( 2)          74.2137          0.2862  ( 5)          64.1363           0.2832
 3 (22)          58.9829          0.2426  (18)          63.6624           1.2231
 5 (11)          57.8344          0.5102  ( 7)          61.4758           0.1027
 2 ( 5)          55.8091          0.3505  ( 8)          60.9444           0.6496
 4 ( 6)          55.5772          0.1859  (10)          60.9444           0.6496
 4 (19)          55.5772          0.1859  (13)          60.9444           0.6496
 3 ( 7)          50.0626          0.1979  (16)          60.9444           0.6496
 8 ( 0)          48.4286          0.4154  (15)          59.8733           0.9041
 5 (25)          46.7625          0.0506  ( 6)          59.7833           7.1770
 5 (12)          46.4891          0.2261  (19)          59.7833           7.1770
 5 ( 3)          45.1190          0.1635  (22)          58.8876           8.5550
 8 ( 1)          44.7921          0.0819  (24)          57.8178           1.4010
11 (18)          42.9309          0.0414  ( 2)          57.5225           0.6985
12 ( 8)          42.9262          0.0129  ( 1)          57.2526           0.5739
12 (10)          42.9262          0.0129  (25)          55.6451           1.2225
12 (13)          42.9262          0.0129  ( 0)          55.4831           1.6249
12 (16)          42.9262          0.0129  ( 3)          53.1334           0.0940
 6 (24)          42.3214          0.0611  (11)          52.9914           3.6257
```

```
10 (15)          42.2376          0.0509  (12)          42.3523          0.3473
 0 (20)           0.0000          0.0000  (20)           0.0000          0.0000
 0 (21)           0.0000          0.0000  (21)           0.0000          0.0000
// Colunas utilizadas 3:

k =

    12    10     9     7     4     3     2     1     0     0     0     0
    12    11    10     8     7     3     2     1     0     0     0     0
    10     1     0     0     0     0     0     0     0     0     0     0
    12     8     7     6     3     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
     7     6     0     0     0     0     0     0     0     0     0     0
    12     9     2     1     0     0     0     0     0     0     0     0
     8     7     6     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     7     6     5     4     3     2     1
     1     0     0     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     7     6     5     4     3     2     1
    12    10     4     2     1     0     0     0     0     0     0     0
    12    11     3     2     1     0     0     0     0     0     0     0
    12    11    10     9     8     7     6     5     4     3     2     1
    12     0     0     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     7     6     5     4     3     0     0
    12    11    10     9     8     7     6     5     4     3     2     1
     1     0     0     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     7     5     4     3     2     1     0
    12     9     2     1     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
    12     2     1     0     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
    12    11     8     7     5     3     0     0     0     0     0     0
    12     9     7     5     2     0     0     0     0     0     0     0

// Problema = ArbitrNVar, Execução 1
vip =

  Columns 1 through 9

    1.1105    1.4256    1.4799    1.0521    1.3214    0.5232    0.4049    0.2327    0.7400
```

```
  Columns 10 through 15

    1.2788    0.7080    0.7912    0.8315    1.0536    1.0429

// full scan 24: tic
ell = 1, K = 15, antes = 15
ell = 2, K = 24, antes = 32
ell = 3, K = 26, antes = 39
ell = 4, K = 25, antes = 41
ell = 5, K = 24, antes = 41
ell = 6, K = 24, antes = 41
Elapsed time is 4.668541 seconds.
// full scan 25: tic
ell = 1, K = 15, antes = 15
ell = 2, K = 24, antes = 32
ell = 3, K = 26, antes = 39
ell = 4, K = 25, antes = 41
ell = 5, K = 24, antes = 41
ell = 6, K = 24, antes = 41
Elapsed time is 1.597504 seconds.
// Erros de treinamento e validação 4:
 1 ( 1)        3543.8563         728.5121  ( 0)         606.8876         107.0801
 1 ( 5)        3543.8563         728.5121  ( 6)         593.9648          90.2573
 2 (16)        2615.8457          27.2670  (11)         593.9648          90.2573
 1 (14)        2576.2213         276.9942  (13)         593.9648          90.2573
13 (15)        2461.6570          12.2432  (19)         593.9648          90.2573
 4 (25)        2413.6054          18.9577  ( 8)         484.6791          46.2994
13 (10)        2396.0251          16.8088  (15)         459.6596          35.3943
13 ( 0)        2390.8092          10.8647  (10)         434.2722          60.3177
15 ( 6)        2362.8369           7.2068  (18)         431.1890          52.8747
15 (11)        2362.8369           7.2068  ( 7)         429.7381          48.8214
15 (13)        2362.8369           7.2068  (12)         173.1706          26.9337
15 (19)        2362.8369           7.2068  ( 2)          94.4923          43.1358
13 ( 8)        2362.0061           5.5378  (22)          78.2400          24.0545
14 ( 7)        2357.0846          12.7974  (23)          78.2400          24.0545
13 (18)        2343.7340           5.0830  (14)          67.0087           1.2738
 5 ( 2)        2338.4199           5.8860  (16)          65.7516           1.8486
 3 ( 3)        2324.5316          28.2363  (24)          65.6470           0.4284
 5 (22)        2311.7313          18.3428  ( 4)          64.1841           0.1031
```

```
 5 (23)        2311.7313          18.3428  ( 9)          64.1841          0.1031
 8 (12)        2241.0306          17.6812  (17)          64.1841          0.1031
 1 ( 4)        2174.0712          91.2147  ( 1)          52.6779          1.8808
 1 ( 9)        2174.0712          91.2147  ( 5)          52.6779          1.8808
 1 (17)        2174.0712          91.2147  ( 3)          50.3003          2.2362
 1 (24)        2107.7180          43.6727  (25)          48.4939          3.4732
 0 (20)           0.0000           0.0000  (20)           0.0000          0.0000
 0 (21)           0.0000           0.0000  (21)           0.0000          0.0000
// Colunas utilizadas 4:

k =

    15    14    13    12    11    10     9     8     6     5     4     2     1     0     0
     8     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     9     8     6     2     1     0     0     0     0     0     0     0     0     0     0
     8     3     1     0     0     0     0     0     0     0     0     0     0     0     0
     3     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     8     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     3     2     1
    15    14    13    12    11    10     9     7     6     5     4     3     2     1     0
    14    12    11    10     9     8     7     6     5     4     3     2     1     0     0
     3     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     8     7     6     5     4     3     2     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     3     2     1
    15    14    10     5     4     3     2     1     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     3     2     1
    15     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     3     0     0
    15    14     0     0     0     0     0     0     0     0     0     0     0     0     0
     3     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    14    13    12    11    10     8     7     6     5     4     3     2     1     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     3     2     1
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    10     8     7     3     2     0     0     0     0     0     0     0     0     0     0
    10     8     7     3     2     0     0     0     0     0     0     0     0     0     0
    10     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    10     8     7     3     0     0     0     0     0     0     0     0     0     0     0
```

```
// Problema = Classificacao 2, Execução 1
vip =

    0.9431    1.0042    1.0069    1.0432

// full scan 24: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.328374 seconds.
// full scan 25: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.175765 seconds.
// Erros de treinamento e validação 5:
 1 (14)           0.9196           0.0216  ( 5)           1.0461           0.0495
 1 (15)           0.9196           0.0216  ( 3)           1.0461           0.0495
 3 (12)           0.8257           0.0002  ( 0)           1.0355           0.0899
 3 (16)           0.8257           0.0002  (10)           1.0355           0.0899
 3 (18)           0.8257           0.0002  (13)           1.0355           0.0899
 3 (22)           0.8257           0.0002  (25)           1.0355           0.0899
 3 (23)           0.8257           0.0002  (12)           0.9977           0.0265
 1 ( 7)           0.8090           0.0009  (16)           0.9977           0.0265
 2 ( 4)           0.7596           0.0072  (18)           0.9977           0.0265
 2 ( 6)           0.7596           0.0072  (22)           0.9977           0.0265
 2 ( 9)           0.7596           0.0072  (23)           0.9977           0.0265
 2 (11)           0.7596           0.0072  ( 7)           0.9291           0.0274
 2 (17)           0.7596           0.0072  ( 4)           0.8107           0.0376
 2 (19)           0.7596           0.0072  ( 6)           0.8107           0.0376
 2 ( 8)           0.7448           0.0037  ( 9)           0.8107           0.0376
 4 ( 0)           0.7443           0.0086  (11)           0.8107           0.0376
 4 (10)           0.7443           0.0086  (17)           0.8107           0.0376
 4 (13)           0.7443           0.0086  (19)           0.8107           0.0376
 4 (25)           0.7443           0.0086  ( 2)           0.7903           0.0410
 3 ( 3)           0.7430           0.0027  (24)           0.7827           0.0442
 3 ( 5)           0.7430           0.0027  ( 8)           0.6998           0.0639
 3 (24)           0.7390           0.0017  (14)           0.5572           0.1268
 1 ( 2)           0.7060           0.0008  (15)           0.5572           0.1268
 0 ( 1)           0.0000           0.0000  ( 1)           0.5122           0.0000
 0 (20)           0.0000           0.0000  (20)           0.0000           0.0000
 0 (21)           0.0000           0.0000  (21)           0.0000           0.0000
// Colunas utilizadas 5:

k =

     4     3     2     1
     0     0     0     0
     3     0     0     0
     4     3     1     0
     3     2     0     0
     4     3     1     0
     3     2     0     0
     1     0     0     0
     4     1     0     0
     3     2     0     0
     4     3     2     1
     3     2     0     0
```

```
     4     3     2     0
     4     3     2     1
     4     0     0     0
     4     0     0     0
     4     3     2     0
     3     2     0     0
     4     3     2     0
     3     2     0     0
     0     0     0     0
     0     0     0     0
     4     3     2     0
     4     3     2     0
     4     2     1     0
     4     3     2     1

// Problema = Classificacao 5, Execução 1
vip =

    1.0280    1.0022    1.0273    0.9399

// full scan 24: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.276608 seconds.
// full scan 25: tic
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
Elapsed time is 0.157547 seconds.
// Erros de treinamento e validação 6:
 1 (14)           2.7105           0.1358  ( 2)           3.3303           0.3292
 1 (15)           2.7105           0.1358  (16)           3.3303           0.3292
 3 ( 2)           2.2690           0.0223  ( 8)           3.0234           0.1768
 3 (16)           2.2690           0.0223  (18)           2.9482           0.1087
 2 ( 8)           2.2550           0.0039  (24)           2.9482           0.1087
 1 ( 7)           2.2068           0.0204  ( 4)           2.7841           0.1116
 4 ( 0)           2.0886           0.0167  ( 6)           2.7841           0.1116
 4 ( 5)           2.0886           0.0167  ( 9)           2.7841           0.1116
 4 (10)           2.0886           0.0167  (11)           2.7841           0.1116
 4 (13)           2.0886           0.0167  (17)           2.7841           0.1116
 4 (22)           2.0886           0.0167  (19)           2.7841           0.1116
 4 (23)           2.0886           0.0167  ( 0)           2.7526           0.0600
 4 (25)           2.0886           0.0167  ( 5)           2.7526           0.0600
 3 ( 3)           2.0683           0.0205  (10)           2.7526           0.0600
 3 (12)           2.0683           0.0205  (13)           2.7526           0.0600
 2 ( 4)           2.0131           0.0254  (22)           2.7526           0.0600
 2 ( 6)           2.0131           0.0254  (23)           2.7526           0.0600
 2 ( 9)           2.0131           0.0254  (25)           2.7526           0.0600
 2 (11)           2.0131           0.0254  ( 3)           2.7293           0.0747
 2 (17)           2.0131           0.0254  (12)           2.7293           0.0747
 2 (19)           2.0131           0.0254  ( 7)           2.5899           0.1730
 3 (18)           1.9698           0.0075  (14)           1.8422           0.5208
 3 (24)           1.9698           0.0075  (15)           1.8422           0.5208
 0 ( 1)           0.0000           0.0000  ( 1)           1.6339           0.0000
 0 (20)           0.0000           0.0000  (20)           0.0000           0.0000
 0 (21)           0.0000           0.0000  (21)           0.0000           0.0000
// Colunas utilizadas 6:
```

```
k =

     4     3     2     1
     0     0     0     0
     4     3     2     0
     3     2     1     0
     4     1     0     0
     4     3     2     1
     4     1     0     0
     2     0     0     0
     3     2     0     0
     4     1     0     0
     4     3     2     1
     4     1     0     0
     3     2     1     0
     4     3     2     1
     4     0     0     0
     4     0     0     0
     4     3     2     0
     4     1     0     0
     4     2     1     0
     4     1     0     0
     0     0     0     0
     0     0     0     0
     4     3     2     1
     4     3     2     1
     4     2     1     0
     4     3     2     1
```

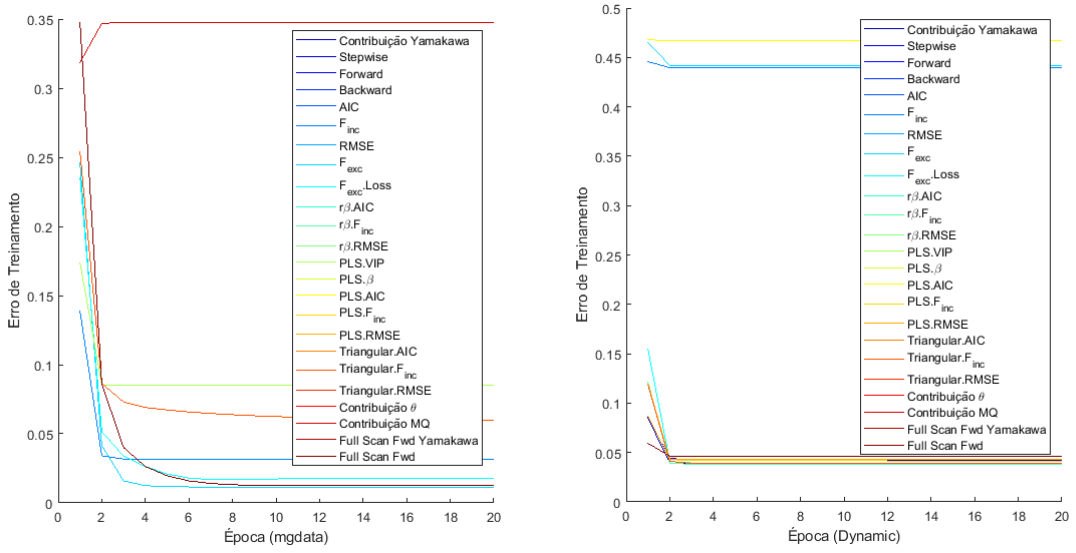

Figura 10: (1) mgdata (2) Dynamic

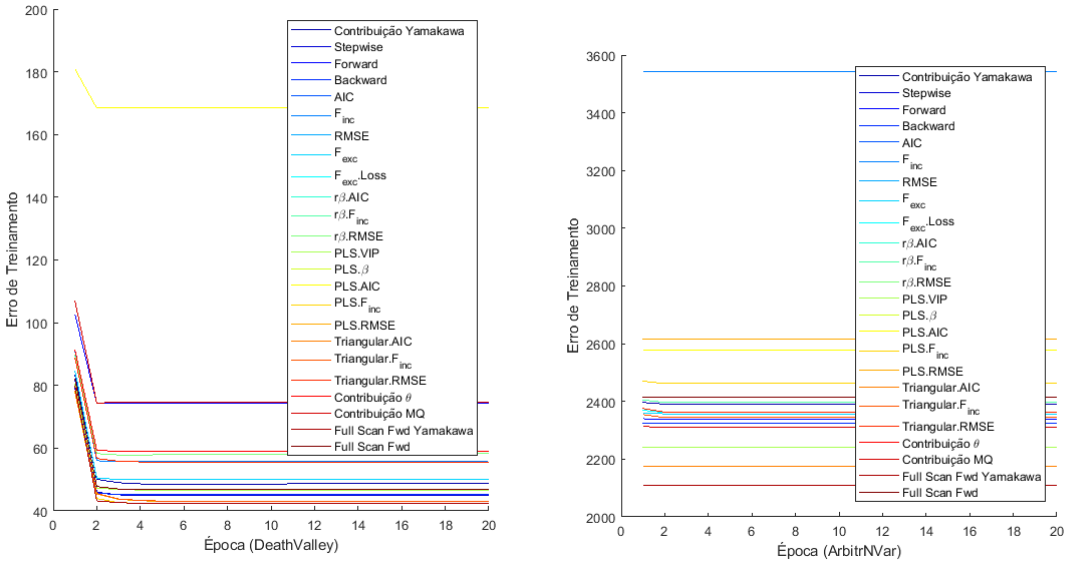

Figura 11: (1) DeathValley (2) ArbitrNVar 15

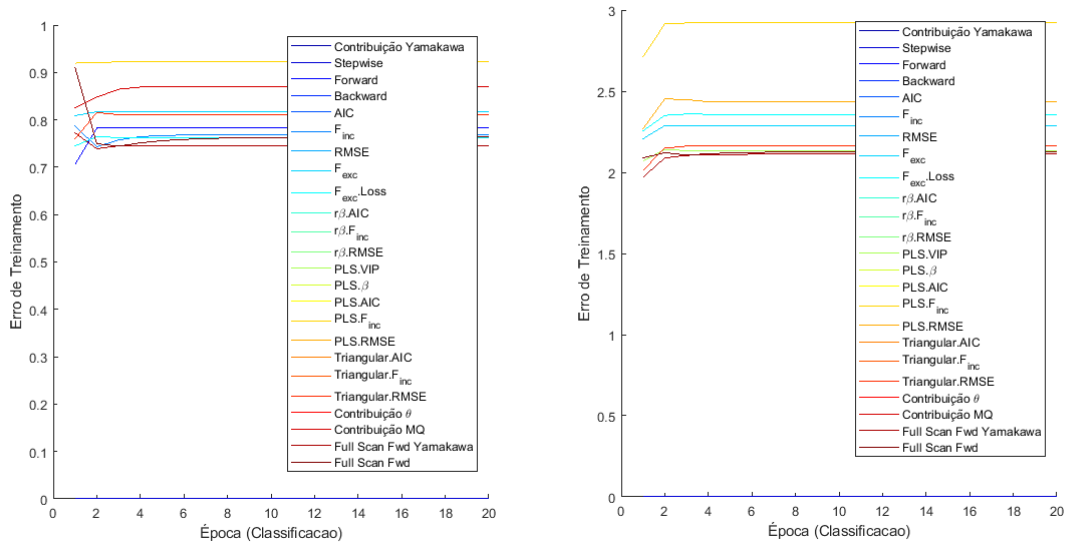

Figura 12: (1) Classificação Binária (2) 5 Classes

8.5 Execução 5

Dois algoritmos online foram acrescentados.

Comparamos 0 = contribuição yamakawa; 1 = stepwise; 2 = forward; 3 = backward; 4 = AIC; 5 = Finc ; 6 = RMSE ; 7 = Fexc ; 8 = FexcLoss ; 9 = rβ.AIC ; 10 = rβ.Finc ; 11 = rβ.RMSE; 12 = PLS.VIP ; 13 = PLS.β ; 14 = PLS.AIC; 15 = PLS.Finc ; 16 = PLS.RMSE ; 17 = Triangular.AIC ; 18 = Triangular.Finc ; 19 = Triangular.RMSE ; 22 = Contribuição θ Online ; 23 = Contribuição θ Sem Loop ; 24 = Contribuição θ MQ ; 25 = full scan fwd Yamakawa ; 26 = full scan fwd.

Aqui a flagEMQ é verdadeira sempre, exceto no Algoritmo $=$ 24.

```
// Problema = mgdata, Execução 1
vip =

    0.9620    0.9757    1.0103    1.0496

ell = 1, K = 4, antes = 4
ell = 2, K = 6, antes = 10
ell = 3, K = 6, antes = 13
ell = 4, K = 6, antes = 14
ell = 1, K = 4, antes = 4
ell = 2, K = 6, antes = 10
ell = 3, K = 6, antes = 13
ell = 4, K = 6, antes = 14
// Erros de treinamento e validação 1:
    0.1495 segundos  1 ( 6)          0.3187          0.0029  ( 6)          0.6776          0.0099
    0.0958 segundos  1 (11)          0.3187          0.0029  (11)          0.6776          0.0099
    0.1162 segundos  1 (14)          0.3187          0.0029  (14)          0.6776          0.0099
    0.1148 segundos  1 (15)          0.3187          0.0029  (15)          0.6776          0.0099
    0.1275 segundos  1 (16)          0.3187          0.0029  (16)          0.6776          0.0099
    0.0827 segundos  1 (19)          0.3187          0.0029  (19)          0.6776          0.0099
    0.0765 segundos  1 (24)          0.3187          0.0029  (24)          0.6776          0.0099
    0.4684 segundos  2 (22)          0.1037          0.0014  (22)          0.2012          0.0076
    0.2520 segundos  2 (12)          0.0849          0.0019  (12)          0.1856          0.0000
    0.2507 segundos  2 (23)          0.0849          0.0019  (23)          0.1856          0.0000
    0.5380 segundos  3 ( 9)          0.0598          0.0015  ( 4)          0.1042          0.0016
    0.5816 segundos  3 ( 4)          0.0598          0.0015  (17)          0.1042          0.0016
    0.5346 segundos  3 (17)          0.0598          0.0015  (18)          0.1042          0.0016
    0.5395 segundos  3 (18)          0.0598          0.0015  ( 9)          0.1042          0.0016
    2.0485 segundos  2 ( 2)          0.0317          0.0007  ( 2)          0.0709          0.0005
    0.3102 segundos  2 ( 5)          0.0317          0.0007  ( 5)          0.0709          0.0005
    0.7130 segundos  4 ( 0)          0.0125          0.0003  ( 0)          0.0221          0.0002
    3.7660 segundos  4 ( 1)          0.0125          0.0003  ( 1)          0.0221          0.0002
    1.7640 segundos  4 ( 3)          0.0125          0.0003  ( 3)          0.0221          0.0002
    1.1304 segundos  4 ( 8)          0.0125          0.0003  ( 8)          0.0221          0.0002
    0.9142 segundos  4 (10)          0.0125          0.0003  (10)          0.0221          0.0002
    0.8844 segundos  4 (13)          0.0125          0.0003  (13)          0.0221          0.0002
    1.9332 segundos  4 (25)          0.0125          0.0003  (25)          0.0221          0.0002
    1.0819 segundos  4 (26)          0.0125          0.0003  (26)          0.0221          0.0002
    0.5855 segundos  3 ( 7)          0.0114          0.0004  ( 7)          0.0185          0.0002
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
```

```
    0.0000 segundos  0 (21)           0.0000           0.0000  (21)           0.0000           0.0000
// Colunas utilizadas 1:

k =

     4     3     2     1
     4     3     2     1
     2     1     0     0
     4     3     2     1
     4     3     1     0
     2     1     0     0
     4     0     0     0
     3     2     1     0
     4     3     2     1
     4     3     1     0
     4     3     2     1
     4     0     0     0
     4     3     0     0
     4     3     2     1
     4     0     0     0
     4     0     0     0
     4     0     0     0
     4     3     1     0
     4     3     1     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     3     1     0     0
     4     3     0     0
     4     0     0     0
     4     3     2     1
     4     3     2     1

// Problema = Dynamic, Execução 1
vip =

    0.0230    0.0293    0.0071    2.2356    0.0293

ell = 1, K = 5, antes = 5
ell = 2, K = 5, antes = 12
```

```
ell = 3, K = 4, antes = 16
ell = 4, K = 4, antes = 16
ell = 1, K = 5, antes = 5
ell = 2, K = 5, antes = 12
ell = 3, K = 4, antes = 16
ell = 4, K = 4, antes = 16
// Erros de treinamento e validação 2:
    4.9252 segundos  2 (22)          0.4992          0.0015  (22)          0.5329          0.0075
    9.2301 segundos  4 ( 7)          0.4571          0.0006  ( 7)          0.4986          0.0066
    0.7899 segundos  1 (14)          0.4547          0.0006  (14)          0.4510          0.0022
    1.2814 segundos  1 ( 5)          0.4357          0.0028  ( 5)          0.4152          0.0147
    6.9067 segundos  1 ( 0)          0.0544          0.0001  ( 0)          0.0551          0.0049
    1.0805 segundos  1 ( 4)          0.0544          0.0001  ( 4)          0.0551          0.0049
    0.9582 segundos  1 ( 6)          0.0544          0.0001  ( 6)          0.0551          0.0049
    0.8507 segundos  1 ( 9)          0.0544          0.0001  ( 9)          0.0551          0.0049
    0.7948 segundos  1 (11)          0.0544          0.0001  (11)          0.0551          0.0049
    0.5560 segundos  1 (12)          0.0544          0.0001  (12)          0.0551          0.0049
    0.5744 segundos  1 (13)          0.0544          0.0001  (13)          0.0551          0.0049
    1.0299 segundos  1 (17)          0.0544          0.0001  (17)          0.0551          0.0049
    0.6713 segundos  1 (19)          0.0544          0.0001  (19)          0.0551          0.0049
    0.8133 segundos  1 (23)          0.0544          0.0001  (23)          0.0551          0.0049
    0.8859 segundos  1 (24)          0.0544          0.0001  (24)          0.0551          0.0049
    7.3958 segundos  1 (25)          0.0544          0.0001  (25)          0.0551          0.0049
    1.7733 segundos  1 (26)          0.0544          0.0001  (26)          0.0551          0.0049
    2.8366 segundos  2 ( 1)          0.0524          0.0003  ( 1)          0.0523          0.0018
    4.9475 segundos  3 (18)          0.0441          0.0000  (18)          0.0416          0.0004
    3.3335 segundos  2 (16)          0.0428          0.0000  ( 2)          0.0416          0.0035
    2.8852 segundos  2 (10)          0.0408          0.0001  ( 3)          0.0411          0.0012
   50.3126 segundos  4 ( 3)          0.0407          0.0001  ( 8)          0.0411          0.0012
   19.9796 segundos  4 ( 8)          0.0407          0.0001  (15)          0.0401          0.0018
   42.8510 segundos  2 ( 2)          0.0406          0.0000  (10)          0.0400          0.0003
    7.0472 segundos  3 (15)          0.0399          0.0000  (16)          0.0373          0.0006
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
// Colunas utilizadas 2:

k =

     4     0     0     0     0
     4     2     0     0     0
```

```
     4     3     0     0     0
     5     4     3     2     0
     4     0     0     0     0
     3     0     0     0     0
     4     0     0     0     0
     5     3     2     1     0
     5     4     3     2     0
     4     0     0     0     0
     4     1     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     5     0     0     0     0
     5     4     3     0     0
     5     4     0     0     0
     4     0     0     0     0
     5     4     2     0     0
     4     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     5     2     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0

// Problema = DeathValley, Execução 1
vip =

  Columns 1 through 11

    1.0499    1.0435    1.0242    0.9995    0.9754    0.9570    0.9507    0.9569    0.9733    0.9988    1.0223

  Column 12

    1.0413

ell = 1, K = 12, antes = 12
ell = 2, K = 41, antes = 60
ell = 3, K = 104, antes = 158
```

```
ell = 4, K = 187, antes = 335
ell = 5, K = 210, antes = 533
ell = 6, K = 201, antes = 641
ell = 7, K = 180, antes = 667
ell = 8, K = 164, antes = 672
ell = 9, K = 155, antes = 675
ell = 10, K = 152, antes = 675
ell = 11, K = 152, antes = 675
ell = 1, K = 12, antes = 12
ell = 2, K = 41, antes = 60
ell = 3, K = 104, antes = 158
ell = 4, K = 187, antes = 335
ell = 5, K = 210, antes = 533
ell = 6, K = 201, antes = 641
ell = 7, K = 180, antes = 667
ell = 8, K = 164, antes = 672
ell = 9, K = 155, antes = 675
ell = 10, K = 152, antes = 675
ell = 11, K = 152, antes = 675
// Erros de treinamento e validação 3:
    0.3833 segundos  1 (14)         124.8947          0.0546  (14)         105.1971          2.8512
    0.9655 segundos  1 ( 4)          56.8620          0.7770  ( 3)          75.7651          0.3254
    0.4566 segundos  1 ( 9)          56.8620          0.7770  ( 8)          71.0331          1.1082
    0.3751 segundos  1 (17)          56.8620          0.7770  (13)          69.2158          0.5286
    0.2856 segundos  1 (24)          56.8620          0.7770  (16)          69.2158          0.5286
    1.4566 segundos  2 ( 5)          54.7938          0.3004  ( 5)          69.0518          0.1308
   40.7260 segundos  3 ( 2)          47.2746          0.2957  (22)          68.6898          2.6379
  242.9809 segundos  8 ( 3)          46.4456          0.0698  (15)          65.8536          0.1475
    2.1985 segundos  3 ( 7)          44.9888          0.1669  ( 1)          64.9093          0.9217
    1.7733 segundos  4 (11)          44.0548          0.2347  ( 0)          64.1333          0.6959
    2.0688 segundos  4 ( 6)          43.9591          0.2460  (18)          61.7078          0.4285
    1.6780 segundos  4 (19)          43.9591          0.2460  ( 7)          59.2629          0.2501
    1.6032 segundos  4 (23)          42.9618          0.2518  (25)          57.4941          0.6121
   54.5429 segundos 10 (22)          42.4640          0.1014  (26)          54.4202          0.2910
    4.0364 segundos  9 ( 0)          42.2919          0.0589  ( 2)          51.7333          0.0413
    2.4464 segundos  5 (12)          42.2536          0.0343  (11)          50.6881          1.1709
    8.9890 segundos 10 (18)          42.2491          0.0199  ( 6)          50.0735          0.1767
   39.6826 segundos  5 (26)          41.9754          0.1113  (19)          50.0735          0.1767
    4.6466 segundos  7 (10)          41.9168          0.0622  (10)          49.9288          0.7114
    8.8461 segundos 10 (15)          41.5870          0.0698  ( 4)          47.4813          1.2300
```

```
   90.0316 segundos  7 (25)          41.3683          0.0550  ( 9)          47.4813          1.2300
   41.0643 segundos 11 ( 8)          41.0923          0.1386  (17)          47.4813          1.2300
   12.5991 segundos 12 (13)          41.0172          0.0556  (24)          47.4813          1.2300
   12.6848 segundos 12 (16)          41.0172          0.0556  (12)          43.1005          0.2300
   18.2818 segundos 10 ( 1)          40.7989          0.0653  (23)          42.5835          0.3384
    0.0000 segundos  0 (20)           0.0000          0.0000  (20)           0.0000          0.0000
    0.0000 segundos  0 (21)           0.0000          0.0000  (21)           0.0000          0.0000
// Colunas utilizadas 3:

k =

    12    11    10     9     7     4     3     2     1     0     0     0
    12    11    10     9     8     7     6     3     2     1     0     0
    11     8     1     0     0     0     0     0     0     0     0     0
    10     9     8     7     5     4     2     1     0     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
     8     7     0     0     0     0     0     0     0     0     0     0
    12     9     2     1     0     0     0     0     0     0     0     0
     8     7     6     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     7     6     5     4     2     1     0
     1     0     0     0     0     0     0     0     0     0     0     0
    12    10     6     4     3     2     1     0     0     0     0     0
    12    10     2     1     0     0     0     0     0     0     0     0
    12    11     3     2     1     0     0     0     0     0     0     0
    12    11    10     9     8     7     6     5     4     3     2     1
    12     0     0     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     7     6     5     4     3     0     0
    12    11    10     9     8     7     6     5     4     3     2     1
     1     0     0     0     0     0     0     0     0     0     0     0
    12    11    10     9     8     5     4     3     2     1     0     0
    12     9     2     1     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
    11    10     9     8     7     6     5     4     3     2     0     0
    12     3     2     1     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
    12    11     9     7     6     4     3     0     0     0     0     0
    12     8     7     4     2     0     0     0     0     0     0     0

// Problema = eOGS Classification, Execução 1
```

```
vip =

  Columns 1 through 11

    0.9995    0.9298    0.9902    0.9567    1.0079    1.0722    0.9457    1.0718    0.9756    0.9827    0.9761

  Columns 12 through 14

    0.9940    1.0704    1.0138

ell = 1, K = 14, antes = 14
ell = 2, K = 71, antes = 83
ell = 3, K = 196, antes = 268
ell = 4, K = 338, antes = 541
ell = 5, K = 392, antes = 721
ell = 6, K = 392, antes = 789
ell = 7, K = 385, antes = 808
ell = 8, K = 387, antes = 813
ell = 9, K = 387, antes = 813
ell = 1, K = 14, antes = 14
ell = 2, K = 71, antes = 83
ell = 3, K = 196, antes = 268
ell = 4, K = 338, antes = 541
ell = 5, K = 392, antes = 721
ell = 6, K = 392, antes = 789
ell = 7, K = 385, antes = 808
ell = 8, K = 387, antes = 813
ell = 9, K = 387, antes = 813
// Erros de treinamento e validação 4:
    9.9875 segundos  1 ( 4)          0.1066          0.0012  ( 9)          0.7688          0.0342
    9.9816 segundos  1 ( 6)          0.1066          0.0012  (11)          0.7688          0.0342
    3.9215 segundos  1 (14)          0.1066          0.0012  ( 4)          0.7343          0.0130
    3.8888 segundos  1 (16)          0.1066          0.0012  ( 6)          0.7343          0.0130
    3.6212 segundos  1 (17)          0.1066          0.0012  (14)          0.7343          0.0130
    3.4720 segundos  1 (19)          0.1066          0.0012  (16)          0.7343          0.0130
 1571.6594 segundos  4 ( 2)          0.0938          0.0002  (17)          0.7343          0.0130
  107.4540 segundos 10 ( 1)          0.0907          0.0004  (19)          0.7343          0.0130
  277.3877 segundos 14 ( 8)          0.0901          0.0000  (24)          0.6638          0.0268
  199.0826 segundos 14 (13)          0.0901          0.0000  ( 0)          0.6550          0.0085
  134.3145 segundos 11 (10)          0.0894          0.0001  (25)          0.6475          0.0306
```

```
    68.8179 segundos  8 (18)          0.0886          0.0000  (23)          0.6471          0.0055
   500.7017 segundos  5 (26)          0.0884          0.0004  (26)          0.6449          0.0111
  2313.6259 segundos 12 ( 3)          0.0882          0.0001  ( 2)          0.6434          0.0053
  3960.7912 segundos 11 (22)          0.0878          0.0008  (18)          0.6202          0.0085
   161.7940 segundos 12 ( 7)          0.0876          0.0001  ( 5)          0.6185          0.0085
    68.4123 segundos  8 (15)          0.0870          0.0007  (12)          0.6104          0.0000
    67.5909 segundos  7 ( 0)          0.0860          0.0003  (10)          0.6098          0.0016
    27.4659 segundos  5 (12)          0.0835          0.0011  (22)          0.5917          0.0026
     4.6243 segundos  1 ( 9)          0.0830          0.0006  ( 7)          0.5779          0.0071
     5.0710 segundos  1 (11)          0.0830          0.0006  ( 1)          0.5737          0.0009
    78.0845 segundos  8 ( 5)          0.0794          0.0001  ( 8)          0.5704          0.0107
    29.9593 segundos  5 (23)          0.0774          0.0000  (13)          0.5704          0.0107
     5.7794 segundos  2 (24)          0.0734          0.0005  ( 3)          0.5637          0.0106
  2102.1953 segundos  2 (25)          0.0727          0.0002  (15)          0.5307          0.0028
     0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
     0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
// Colunas utilizadas 4:

k =

    10     9     6     5     4     2     1     0     0     0     0     0     0     0
    14    12    11     8     6     5     4     3     2     1     0     0     0     0
     7     4     2     1     0     0     0     0     0     0     0     0     0     0
    14    13    12    11    10     8     7     6     5     4     2     1     0     0
    14     0     0     0     0     0     0     0     0     0     0     0     0     0
    11    10     9     8     7     6     4     2     0     0     0     0     0     0
    14     0     0     0     0     0     0     0     0     0     0     0     0     0
    13    12    11    10     9     8     7     6     5     4     3     1     0     0
    14    13    12    11    10     9     8     7     6     5     4     3     2     1
     4     0     0     0     0     0     0     0     0     0     0     0     0     0
    14    10     9     8     7     6     5     4     3     2     1     0     0     0
     4     0     0     0     0     0     0     0     0     0     0     0     0     0
    14    13     8     6     5     0     0     0     0     0     0     0     0     0
    14    13    12    11    10     9     8     7     6     5     4     3     2     1
    14     0     0     0     0     0     0     0     0     0     0     0     0     0
    14    13    12    11    10     9     8     7     0     0     0     0     0     0
    14     0     0     0     0     0     0     0     0     0     0     0     0     0
    14     0     0     0     0     0     0     0     0     0     0     0     0     0
    14    11     9     6     5     4     3     2     0     0     0     0     0     0
    14     0     0     0     0     0     0     0     0     0     0     0     0     0
```

```
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    13    11    10     9     8     6     5     4     3     2     1     0     0     0
    11     6     5     3     2     0     0     0     0     0     0     0     0     0
    12     6     0     0     0     0     0     0     0     0     0     0     0     0
     9     6     0     0     0     0     0     0     0     0     0     0     0     0
    10     5     3     2     1     0     0     0     0     0     0     0     0     0

// Problema = eOGS Prediction, Execução 1
vip =

    0.0000    0.0000    0.0000    2.4495    0.0001    0.0001

ell = 1, K = 6, antes = 6
ell = 2, K = 11, antes = 21
ell = 3, K = 11, antes = 27
ell = 4, K = 11, antes = 27
ell = 1, K = 6, antes = 6
ell = 2, K = 11, antes = 21
ell = 3, K = 11, antes = 27
ell = 4, K = 11, antes = 27
// Erros de treinamento e validação 5:
    1.4189 segundos  1 (14)           0.0732           0.0004  (14)           0.1293           0.0097
    1.5836 segundos  1 (16)           0.0732           0.0004  (16)           0.1293           0.0097
    1.8386 segundos  1 ( 6)           0.0674           0.0002  ( 6)           0.0989           0.0037
    1.5474 segundos  1 (19)           0.0674           0.0002  (19)           0.0989           0.0037
    2.6088 segundos  2 (15)           0.0643           0.0003  (15)           0.0919           0.0062
    4.4438 segundos  1 ( 0)           0.0607           0.0032  (12)           0.0831           0.0015
    1.4853 segundos  1 (11)           0.0607           0.0032  (13)           0.0831           0.0015
    1.5059 segundos  1 (23)           0.0607           0.0032  (22)           0.0831           0.0015
    1.3781 segundos  1 (24)           0.0607           0.0032  ( 7)           0.0710           0.0006
    4.0383 segundos  1 (26)           0.0607           0.0032  (25)           0.0625           0.0016
    1.1996 segundos  1 (12)           0.0583           0.0003  ( 3)           0.0587           0.0016
    1.2476 segundos  1 (13)           0.0583           0.0003  ( 2)           0.0582           0.0029
    5.5088 segundos  1 (22)           0.0583           0.0003  ( 8)           0.0582           0.0029
    2.3925 segundos  1 ( 7)           0.0568           0.0029  ( 1)           0.0579           0.0028
   14.7082 segundos  3 (25)           0.0505           0.0007  ( 5)           0.0579           0.0028
    2.9809 segundos  2 ( 4)           0.0483           0.0006  (10)           0.0576           0.0027
    2.9189 segundos  2 (17)           0.0483           0.0006  (18)           0.0561           0.0017
    7.6712 segundos  4 ( 9)           0.0472           0.0009  ( 9)           0.0553           0.0011
```

```
    64.2760 segundos  3 ( 3)          0.0471          0.0001  ( 0)          0.0527          0.0020
    70.1284 segundos  4 ( 2)          0.0466          0.0008  (11)          0.0527          0.0020
    29.4424 segundos  4 ( 8)          0.0466          0.0008  (23)          0.0527          0.0020
    11.1317 segundos  5 (10)          0.0463          0.0006  (24)          0.0527          0.0020
     7.8910 segundos  4 (18)          0.0459          0.0000  (26)          0.0527          0.0020
    19.3834 segundos  6 ( 1)          0.0453          0.0004  ( 4)          0.0520          0.0017
    16.2564 segundos  6 ( 5)          0.0453          0.0004  (17)          0.0520          0.0017
     0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
     0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
// Colunas utilizadas 5:

k =

     1     0     0     0     0     0
     6     5     4     3     2     1
     6     5     3     1     0     0
     6     4     1     0     0     0
     5     1     0     0     0     0
     6     5     4     3     2     1
     5     0     0     0     0     0
     3     0     0     0     0     0
     6     5     3     1     0     0
     5     3     2     1     0     0
     6     5     3     2     1     0
     1     0     0     0     0     0
     4     0     0     0     0     0
     4     0     0     0     0     0
     6     0     0     0     0     0
     6     5     0     0     0     0
     6     0     0     0     0     0
     5     1     0     0     0     0
     6     5     4     1     0     0
     5     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     4     0     0     0     0     0
     1     0     0     0     0     0
     1     0     0     0     0     0
     6     5     3     0     0     0
     1     0     0     0     0     0
```

```
// Problema = FBeM BoxJenkins, Execução 1
vip =

    1.1774    1.1620    1.0939    0.9710    0.3522

ell = 1, K = 5, antes = 5
ell = 2, K = 1, antes = 9
ell = 3, K = 1, antes = 9
ell = 1, K = 5, antes = 5
ell = 2, K = 1, antes = 9
ell = 3, K = 1, antes = 9
// Erros de treinamento e validação 6:
    0.4370 segundos  3 ( 5)          1.6779          0.0074  ( 5)          3.4918          0.0473
    0.9313 segundos  1 ( 0)          1.5655          0.0557  ( 7)          3.4177          0.0444
    0.1992 segundos  1 (14)          1.5655          0.0557  (12)          3.3530          0.0176
    0.2054 segundos  1 (15)          1.5655          0.0557  (22)          3.3196          0.0907
    0.2417 segundos  1 (23)          1.5655          0.0557  ( 0)          2.1136          0.2520
    0.1919 segundos  1 (24)          1.5655          0.0557  (14)          2.1136          0.2520
    0.2745 segundos  1 (25)          1.5655          0.0557  (15)          2.1136          0.2520
    0.2404 segundos  1 (26)          1.5655          0.0557  (23)          2.1136          0.2520
    0.5813 segundos  4 ( 7)          1.3988          0.0186  (24)          2.1136          0.2520
    0.3657 segundos  3 (12)          1.3217          0.0068  (25)          2.1136          0.2520
    0.3867 segundos  2 (22)          1.2398          0.0019  (26)          2.1136          0.2520
    0.3983 segundos  2 ( 9)          0.8327          0.0046  ( 2)          1.6534          0.0256
    0.2677 segundos  2 (11)          0.8327          0.0046  ( 3)          1.6534          0.0256
    0.2632 segundos  2 (16)          0.8327          0.0046  ( 9)          1.6414          0.0095
    0.3166 segundos  2 ( 4)          0.8268          0.0113  (11)          1.6414          0.0095
    0.3134 segundos  2 ( 6)          0.8268          0.0113  (16)          1.6414          0.0095
    0.3071 segundos  2 (17)          0.8268          0.0113  ( 4)          1.4969          0.0712
    0.2649 segundos  2 (19)          0.8268          0.0113  ( 6)          1.4969          0.0712
    1.1886 segundos  2 ( 2)          0.8024          0.0025  (17)          1.4969          0.0712
    2.8919 segundos  2 ( 3)          0.8024          0.0025  (19)          1.4969          0.0712
    0.9680 segundos  3 ( 8)          0.6884          0.0181  ( 8)          1.2890          0.0786
    0.5113 segundos  4 (18)          0.6487          0.0069  (18)          1.1755          0.0626
    0.9436 segundos  4 ( 1)          0.5861          0.0090  ( 1)          1.0066          0.0443
    0.7096 segundos  5 (10)          0.5615          0.0003  (10)          0.9908          0.0306
    0.6972 segundos  5 (13)          0.5615          0.0003  (13)          0.9908          0.0306
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
```

```
// Colunas utilizadas 6:

k =

     5     0     0     0     0
     5     4     2     1     0
     5     2     0     0     0
     5     2     0     0     0
     5     3     0     0     0
     4     3     2     0     0
     5     3     0     0     0
     4     3     2     1     0
     5     4     2     0     0
     5     4     0     0     0
     5     4     3     2     1
     5     4     0     0     0
     3     2     1     0     0
     5     4     3     2     1
     5     0     0     0     0
     5     0     0     0     0
     5     4     0     0     0
     5     3     0     0     0
     5     4     3     2     0
     5     3     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     2     1     0     0     0
     5     0     0     0     0
     5     0     0     0     0
     5     0     0     0     0
     5     0     0     0     0

// Problema = FBeM Global40, Execução 1
vip =

    0.9999    1.0001

ell = 1, K = 2, antes = 2
ell = 2, K = 1, antes = 3
ell = 1, K = 2, antes = 2
```

```
ell = 2, K = 1, antes = 3
// Erros de treinamento e validação 7:
    0.5138 segundos  1 ( 7)          1.6748          0.0046  ( 7)         11.2004          0.2650
    3.9247 segundos  1 ( 0)          1.4414          0.0130  ( 0)          8.9762          0.0093
    2.3554 segundos  1 ( 2)          1.4414          0.0130  ( 2)          8.9762          0.0093
    1.8516 segundos  1 ( 3)          1.4414          0.0130  ( 3)          8.9762          0.0093
    0.6703 segundos  1 ( 4)          1.4414          0.0130  ( 4)          8.9762          0.0093
    0.4848 segundos  1 ( 6)          1.4414          0.0130  ( 6)          8.9762          0.0093
    0.5134 segundos  1 ( 9)          1.4414          0.0130  ( 9)          8.9762          0.0093
    0.5609 segundos  1 (11)          1.4414          0.0130  (11)          8.9762          0.0093
    0.5115 segundos  1 (12)          1.4414          0.0130  (12)          8.9762          0.0093
    0.5299 segundos  1 (14)          1.4414          0.0130  (14)          8.9762          0.0093
    0.4899 segundos  1 (15)          1.4414          0.0130  (15)          8.9762          0.0093
    0.4742 segundos  1 (16)          1.4414          0.0130  (16)          8.9762          0.0093
    0.4714 segundos  1 (17)          1.4414          0.0130  (17)          8.9762          0.0093
    0.4897 segundos  1 (18)          1.4414          0.0130  (18)          8.9762          0.0093
    0.5112 segundos  1 (19)          1.4414          0.0130  (19)          8.9762          0.0093
    0.5141 segundos  1 (23)          1.4414          0.0130  (23)          8.9762          0.0093
    0.4547 segundos  1 (24)          1.4414          0.0130  (24)          8.9762          0.0093
    0.6600 segundos  1 (25)          1.4414          0.0130  (25)          8.9762          0.0093
    0.5178 segundos  1 (26)          1.4414          0.0130  (26)          8.9762          0.0093
    1.0004 segundos  2 ( 1)          1.3730          0.0120  ( 1)          8.6031          0.0909
    0.8439 segundos  2 ( 5)          1.3730          0.0120  ( 5)          8.6031          0.0909
    1.2199 segundos  2 ( 8)          1.3730          0.0120  ( 8)          8.6031          0.0909
    0.8436 segundos  2 (10)          1.3730          0.0120  (10)          8.6031          0.0909
    0.8484 segundos  2 (13)          1.3730          0.0120  (13)          8.6031          0.0909
    0.9727 segundos  2 (22)          1.3730          0.0120  (22)          8.6031          0.0909
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
// Colunas utilizadas 7:

k =

     2     0
     2     1
     2     0
     2     0
     2     0
     2     1
     2     0
```

```
     1     0
     2     1
     2     0
     2     1
     2     0
     2     0
     2     1
     2     0
     2     0
     2     0
     2     0
     2     0
     2     0
     0     0
     0     0
     2     1
     2     0
     2     0
     2     0
     2     0

// Problema = FBeM Mackey Glass, Execução 1
vip =

    0.9942    0.9981    1.0019    1.0058

ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
// Erros de treinamento e validação 8:
    1.4573 segundos  1 ( 6)          0.2526          0.0029  ( 6)          5.7418          0.0196
    1.2114 segundos  1 (11)          0.2526          0.0029  (11)          5.7418          0.0196
    1.0894 segundos  1 (14)          0.2526          0.0029  (14)          5.7418          0.0196
    1.0880 segundos  1 (15)          0.2526          0.0029  (15)          5.7418          0.0196
    1.0839 segundos  1 (16)          0.2526          0.0029  (16)          5.7418          0.0196
```

```
    1.0514 segundos  1 (19)          0.2526          0.0029  (19)          5.7418          0.0196
    4.2011 segundos  2 ( 7)          0.0180          0.0005  ( 7)          0.2243          0.0100
    3.4598 segundos  2 (23)          0.0174          0.0005  (18)          0.2046          0.0320
    3.2868 segundos  2 (12)          0.0166          0.0008  (23)          0.2003          0.0149
    3.3534 segundos  2 (24)          0.0166          0.0008  ( 0)          0.1929          0.0079
    9.3976 segundos  3 (22)          0.0154          0.0003  ( 2)          0.1903          0.0047
    8.5018 segundos  3 ( 0)          0.0143          0.0008  ( 3)          0.1903          0.0047
   97.1411 segundos  2 ( 2)          0.0143          0.0005  ( 8)          0.1903          0.0047
   95.1070 segundos  2 ( 3)          0.0143          0.0005  (22)          0.1891          0.0049
   35.1933 segundos  2 ( 8)          0.0143          0.0005  (25)          0.1875          0.0141
    7.5617 segundos  3 (18)          0.0142          0.0004  (12)          0.1836          0.0032
   13.7312 segundos  4 ( 4)          0.0141          0.0005  (24)          0.1836          0.0032
   12.9442 segundos  4 (17)          0.0141          0.0005  ( 1)          0.1788          0.0020
   16.3096 segundos  4 ( 1)          0.0141          0.0005  ( 5)          0.1788          0.0020
   14.0318 segundos  4 ( 5)          0.0141          0.0005  ( 9)          0.1788          0.0020
   13.0863 segundos  4 ( 9)          0.0141          0.0005  (10)          0.1788          0.0020
   13.3996 segundos  4 (10)          0.0141          0.0005  (13)          0.1788          0.0020
   12.7046 segundos  4 (13)          0.0141          0.0005  (26)          0.1788          0.0020
   15.0629 segundos  4 (26)          0.0141          0.0005  ( 4)          0.1788          0.0020
   21.6439 segundos  2 (25)          0.0135          0.0007  (17)          0.1788          0.0020
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
// Colunas utilizadas 8:

k =

     4     2     1     0
     4     3     2     1
     4     1     0     0
     4     1     0     0
     4     3     2     1
     4     3     2     1
     4     0     0     0
     2     1     0     0
     4     1     0     0
     4     3     2     1
     4     3     2     1
     4     0     0     0
     4     3     0     0
     4     3     2     1
```

```
     4     0     0     0
     4     0     0     0
     4     0     0     0
     4     3     2     1
     4     3     2     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     4     3     1     0
     3     2     0     0
     4     3     0     0
     4     2     0     0
     4     3     2     1

// Problema = ArbitrNVar, Execução 1
vip =

  Columns 1 through 11

    0.4086    1.0288    1.1668    0.8861    0.8516    1.1290    0.9245    0.9392    1.1044    1.0121    1.1445

  Columns 12 through 15

    1.0027    1.1962    0.9818    0.9681

ell = 1, K = 15, antes = 15
ell = 2, K = 65, antes = 82
ell = 3, K = 150, antes = 217
ell = 4, K = 345, antes = 498
ell = 5, K = 497, antes = 855
ell = 6, K = 481, antes = 1038
ell = 7, K = 454, antes = 1057
ell = 8, K = 454, antes = 1057
ell = 1, K = 15, antes = 15
ell = 2, K = 65, antes = 82
ell = 3, K = 150, antes = 217
ell = 4, K = 345, antes = 498
ell = 5, K = 497, antes = 855
ell = 6, K = 481, antes = 1038
ell = 7, K = 454, antes = 1057
```

```
ell = 8, K = 454, antes = 1057
// Erros de treinamento e validação 9:
    1.0083 segundos  1 ( 4)        1068.5073         146.3486  (16)       14097.1035          18.7068
    0.3482 segundos  1 ( 9)        1068.5073         146.3486  ( 4)       14091.8827           7.9317
    0.2319 segundos  1 (17)        1068.5073         146.3486  ( 9)       14091.8827           7.9317
    0.9738 segundos  3 (16)         958.2403          14.3153  (17)       14091.8827           7.9317
    0.2579 segundos  1 (14)         876.4185          28.4790  (14)       14084.1115           1.6756
    8.7961 segundos 10 (15)         852.3535           8.4297  (12)       14082.4212          21.0875
    5.5781 segundos  8 (12)         817.3315          16.8364  ( 1)       14015.4109          21.2220
   19.0637 segundos 14 ( 7)         799.2544          14.5127  (15)       13979.8464           4.7016
    7.1827 segundos  8 ( 5)         795.9379          10.6706  (25)       13971.5408          21.7314
    0.8074 segundos  1 ( 1)         785.4501          41.7561  ( 7)       13967.4546          11.2944
  481.4964 segundos 11 (22)         782.6989           4.6466  ( 2)       13967.0767          46.2922
   22.3899 segundos  3 ( 2)         776.1051          17.1509  (24)       13911.4029          26.3068
    1.4416 segundos  4 (24)         758.2394           0.6445  (26)       13877.1489          18.5687
    5.8261 segundos  7 (23)         751.4153           3.9315  ( 8)       13858.4170           9.4853
    6.4160 segundos  8 ( 6)         750.9764           3.3223  ( 3)       13809.6885           2.6574
    5.7824 segundos  8 (11)         750.9764           3.3223  (23)       13806.9761          36.8884
    5.6638 segundos  8 (19)         750.9764           3.3223  ( 0)       13793.2839          12.5979
   58.1589 segundos 10 ( 8)         747.3480           4.1682  ( 6)       13786.5743          27.6681
   74.7161 segundos  6 (26)         745.5362           1.0768  (11)       13786.5743          27.6681
  227.5175 segundos 12 ( 3)         742.3600           5.2817  (19)       13786.5743          27.6681
   17.1975 segundos 14 (18)         741.1738           0.7120  ( 5)       13777.4699          33.6177
   19.5752 segundos 15 (10)         739.2166           0.5476  (22)       13775.9796          24.7648
   19.3711 segundos 15 (13)         739.2166           0.5476  (18)       13773.6400           8.1864
    2.5051 segundos 13 ( 0)         733.0302           1.2157  (10)       13761.7150           2.9360
  265.4521 segundos  3 (25)         731.4876          10.9107  (13)       13761.7150           2.9360
    0.0000 segundos  0 (20)           0.0000           0.0000  (20)           0.0000           0.0000
    0.0000 segundos  0 (21)           0.0000           0.0000  (21)           0.0000           0.0000
// Colunas utilizadas 9:

k =

    15    13    11    10     9     8     7     6     5     4     3     2     1     0     0
     1     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    12     4     1     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     7     5     4     3     1     0     0     0
    13     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    14    12    10     8     7     5     4     1     0     0     0     0     0     0     0
    13    11     9     8     6     3     2     1     0     0     0     0     0     0     0
```

```
    15    14    13    12    11    10     9     8     7     6     5     4     3     2     0
    15    13    12    11    10     9     5     4     3     1     0     0     0     0     0
    13     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     3     2     1
    13    11     9     8     6     3     2     1     0     0     0     0     0     0     0
    13    12    11    10     9     6     3     2     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     8     7     6     5     4     3     2     1
    15     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11    10     9     8     7     6     0     0     0     0     0
    15    14    13     0     0     0     0     0     0     0     0     0     0     0     0
    13     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    11    10     9     8     7     6     5     4     3     2     1     0
    13    11     9     8     6     3     2     1     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    15    14    13    12    11     8     6     5     4     2     1     0     0     0     0
    13    11     9     8     6     3     1     0     0     0     0     0     0     0     0
    13     6     3     1     0     0     0     0     0     0     0     0     0     0     0
    15     3     1     0     0     0     0     0     0     0     0     0     0     0     0
    13    11     9     6     3     1     0     0     0     0     0     0     0     0     0

// Problema = Classificacao, Execução 1
vip =

    0.9439    1.0357    0.9099    1.0993

ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
// Erros de treinamento e validação 10:
    0.0898 segundos  3 (23)           0.7633           0.0182  ( 5)           1.2624           0.1251
    0.0995 segundos  3 ( 5)           0.7619           0.0144  (22)           1.0958           0.1473
    0.1963 segundos  1 ( 1)           0.7585           0.0231  (23)           1.0473           0.1015
    0.2213 segundos  1 ( 3)           0.7585           0.0231  ( 0)           0.9221           0.1074
    0.0179 segundos  1 ( 6)           0.7585           0.0231  (10)           0.9221           0.1074
```

```
    0.0194 segundos  1 (11)          0.7585          0.0231  (13)          0.9221          0.1074
    0.0204 segundos  1 (14)          0.7585          0.0231  (26)          0.9221          0.1074
    0.0239 segundos  1 (15)          0.7585          0.0231  ( 2)          0.9008          0.1230
    0.0211 segundos  1 (16)          0.7585          0.0231  (25)          0.9008          0.1230
    0.0156 segundos  1 (19)          0.7585          0.0231  ( 1)          0.8509          0.0058
    0.0490 segundos  2 ( 4)          0.7522          0.0089  ( 3)          0.8509          0.0058
    0.0474 segundos  2 ( 9)          0.7522          0.0089  ( 6)          0.8509          0.0058
    0.0452 segundos  2 (12)          0.7522          0.0089  (11)          0.8509          0.0058
    0.0460 segundos  2 (17)          0.7522          0.0089  (14)          0.8509          0.0058
    0.0427 segundos  2 (24)          0.7522          0.0089  (15)          0.8509          0.0058
    0.0302 segundos  1 ( 7)          0.7482          0.0381  (16)          0.8509          0.0058
    0.2217 segundos  3 ( 8)          0.7478          0.0069  (19)          0.8509          0.0058
    0.0949 segundos  3 (18)          0.7478          0.0069  ( 8)          0.7730          0.0281
    0.0899 segundos  4 ( 0)          0.7435          0.0245  (18)          0.7730          0.0281
    0.1611 segundos  4 (10)          0.7435          0.0245  ( 7)          0.7614          0.1247
    0.1530 segundos  4 (13)          0.7435          0.0245  ( 4)          0.6265          0.0331
    0.1843 segundos  4 (26)          0.7435          0.0245  (17)          0.6265          0.0331
    0.3414 segundos  3 ( 2)          0.7289          0.0043  ( 9)          0.6265          0.0331
    0.3245 segundos  3 (25)          0.7289          0.0043  (12)          0.6265          0.0331
    0.0609 segundos  2 (22)          0.7027          0.0046  (24)          0.6265          0.0331
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000

// Colunas utilizadas 10:

k =

     4     3     2     1
     4     0     0     0
     4     3     1     0
     4     0     0     0
     4     2     0     0
     3     2     1     0
     4     0     0     0
     1     0     0     0
     4     2     1     0
     4     2     0     0
     4     3     2     1
     4     0     0     0
     4     2     0     0
```

```
     4     3     2     1
     4     0     0     0
     4     0     0     0
     4     0     0     0
     4     2     0     0
     4     2     1     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     3     1     0     0
     4     3     2     0
     4     2     0     0
     4     3     1     0
     4     3     2     1

// Problema = Classificacao, Execução 1
vip =

    0.9579    1.0334    1.0540    0.9507

ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
// Erros de treinamento e validação 11:
    0.0232 segundos  1 ( 7)          2.5985          0.4011  ( 4)          2.4132          0.2318
    0.0981 segundos  3 ( 5)          2.0592          0.0537  ( 6)          2.4132          0.2318
    0.2932 segundos  3 ( 3)          2.0408          0.0114  ( 9)          2.4132          0.2318
    0.2210 segundos  3 ( 8)          2.0408          0.0114  (11)          2.4132          0.2318
    0.1518 segundos  1 ( 2)          1.9196          0.0033  (12)          2.4132          0.2318
    0.0149 segundos  1 (14)          1.9196          0.0033  (17)          2.4132          0.2318
    0.0154 segundos  1 (15)          1.9196          0.0033  (19)          2.4132          0.2318
    0.0475 segundos  2 ( 4)          1.9183          0.0202  ( 3)          2.1317          0.2694
    0.0462 segundos  2 ( 6)          1.9183          0.0202  ( 8)          2.1317          0.2694
    0.0450 segundos  2 ( 9)          1.9183          0.0202  ( 7)          1.9455          0.0125
    0.0450 segundos  2 (11)          1.9183          0.0202  (16)          1.9280          0.0723
```

```
    0.0412 segundos  2 (12)          1.9183          0.0202  (18)          1.9280          0.0723
    0.0441 segundos  2 (17)          1.9183          0.0202  ( 0)          1.8205          0.1103
    0.0449 segundos  2 (19)          1.9183          0.0202  (10)          1.8205          0.1103
    0.0266 segundos  1 (22)          1.9133          0.0480  (13)          1.8205          0.1103
    0.2649 segundos  2 (25)          1.8312          0.0141  (23)          1.8205          0.1103
    0.0855 segundos  4 ( 0)          1.8122          0.0157  (24)          1.8205          0.1103
    0.1671 segundos  4 (10)          1.8122          0.0157  (26)          1.8205          0.1103
    0.1561 segundos  4 (13)          1.8122          0.0157  ( 5)          1.6789          0.0249
    0.1539 segundos  4 (23)          1.8122          0.0157  (22)          1.6727          0.0216
    0.1501 segundos  4 (24)          1.8122          0.0157  ( 2)          1.3654          0.1192
    0.1838 segundos  4 (26)          1.8122          0.0157  (14)          1.3654          0.1192
    0.0929 segundos  3 (16)          1.8103          0.0068  (15)          1.3654          0.1192
    0.0911 segundos  3 (18)          1.8103          0.0068  (25)          1.3511          0.0686
    0.1166 segundos  0 ( 1)          0.0000          0.0000  ( 1)          1.2798          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
// Colunas utilizadas 11:

k =

     4     3     2     1
     0     0     0     0
     4     0     0     0
     3     2     1     0
     3     2     0     0
     4     3     1     0
     3     2     0     0
     1     0     0     0
     3     2     1     0
     3     2     0     0
     4     3     2     1
     3     2     0     0
     3     2     0     0
     4     3     2     1
     4     0     0     0
     4     0     0     0
     4     3     2     0
     3     2     0     0
     4     3     2     0
```

3	2	0	0
0	0	0	0
0	0	0	0
2	0	0	0
4	3	2	1
4	3	2	1
4	2	0	0
4	3	2	1

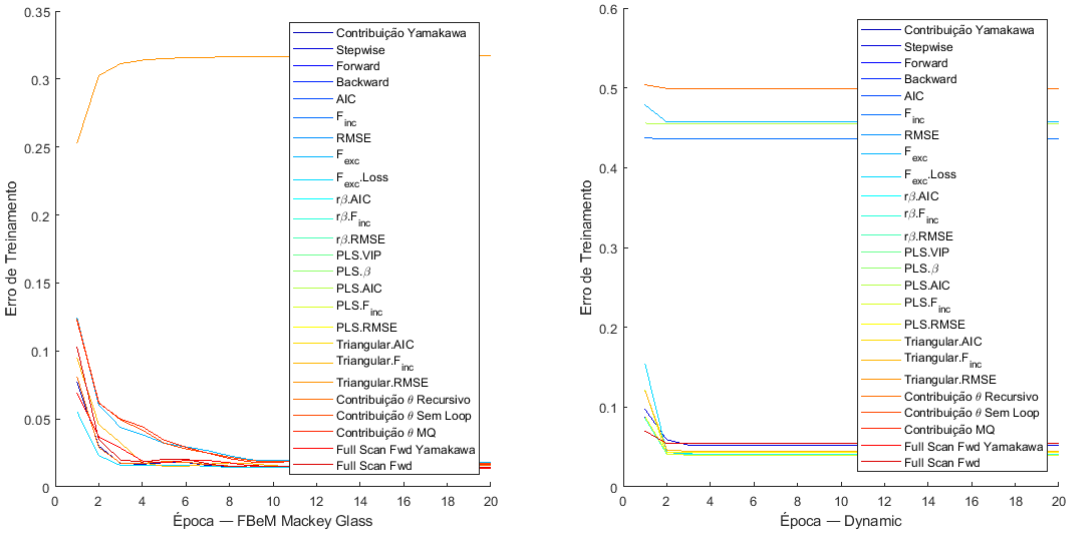

Figura 13: (1) FBeM Mackey Glass (2) Dynamic

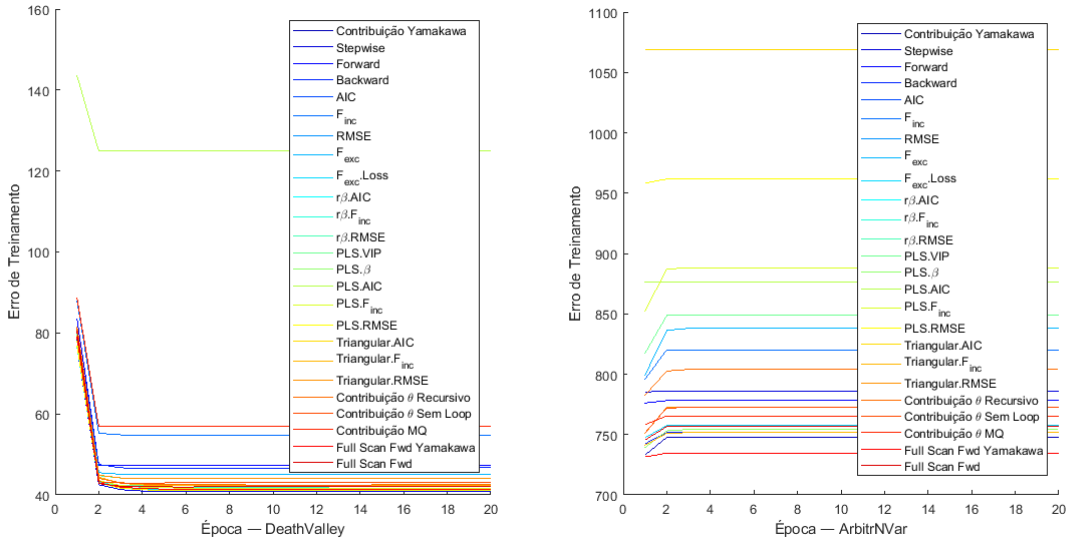

Figura 14: (1) DeathValley (2) ArbitrNVar 15

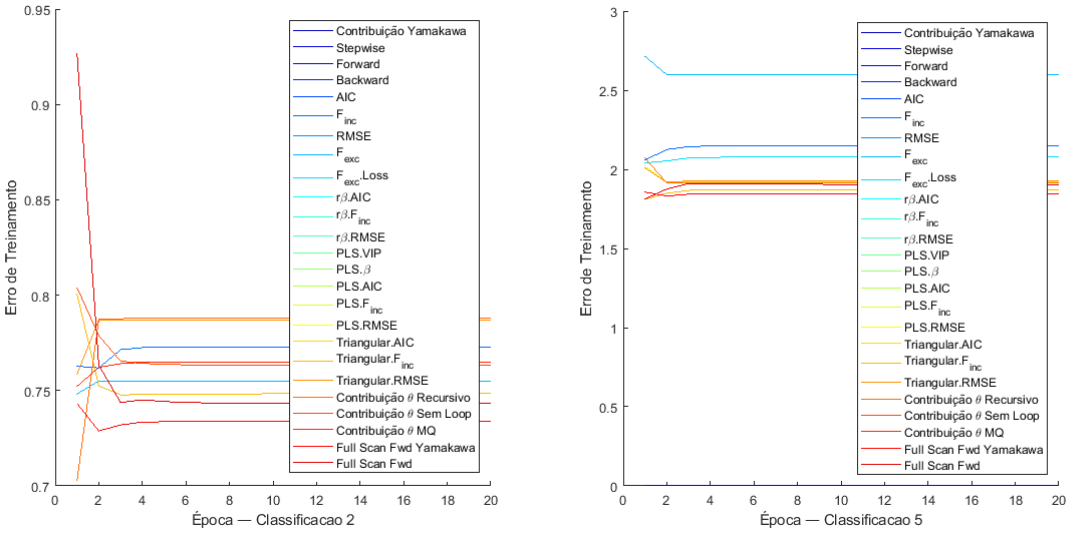

Figura 15: (1) Classificação Binária (2) 5 Classes

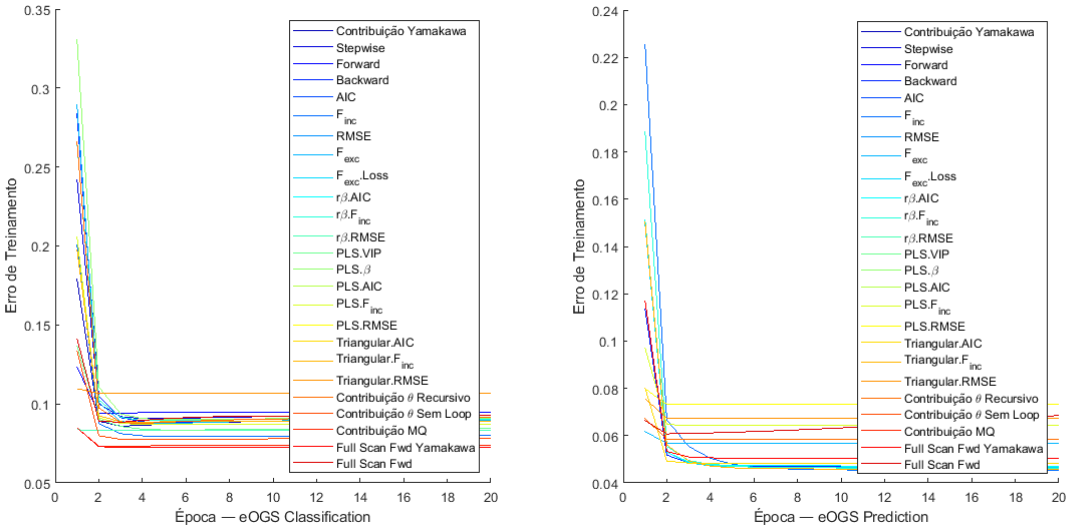

Figura 16: (1) EyeState (2) Parkinsons

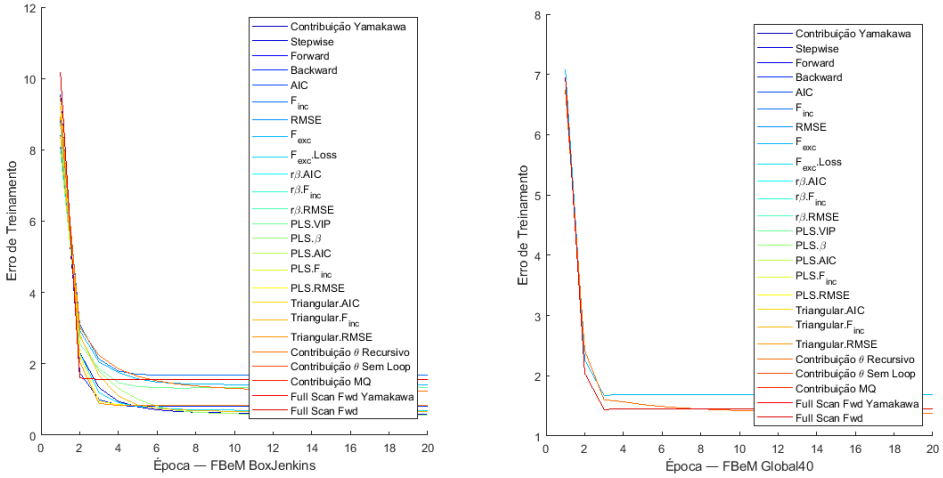

Figura 17: (1) BoxJenkins (2) Global40PxLast

8.6 Execução 6

Comparamos 0 = contribuição yamakawa; 6 = RMSE ; 11 = rβ.RMSE; 12 = PLS.VIP ; 16 = PLS.RMSE ; 23 = Contribuição θ Sem Loop ; 25 = Contribuição Yamakawa Online Sem Loop ; 27 = full scan fwd Yamakawa ; 28 = full scan fwd max ; 29 = full scan fwd sum.

Aqui a flagEMQ é sempre verdadeira, flagQuadrado e flagPCA são falsas.

Precisávamos de todos os resultados de novo, normalizados, mas o problema ArbitrNVar com 25 variáveis parece um caso de underfitting, porque, para ser rápido, somente com o quadrado/pca em verdadeiros.

```
// Problema = Dynamic, Quadrado = 0, Execução 1
vip =

    0.9977    1.1116    0.9223    1.0944    0.8488

ell = 1, K = 5, antes = 5
ell = 2, K = 5, antes = 12
ell = 3, K = 3, antes = 14
ell = 4, K = 3, antes = 14
ell = 1, K = 5, antes = 5
ell = 2, K = 5, antes = 12
ell = 3, K = 3, antes = 14
ell = 4, K = 3, antes = 14
ell = 1, K = 5, antes = 5
ell = 2, K = 4, antes = 12
ell = 3, K = 3, antes = 13
ell = 4, K = 3, antes = 13
// Erros de treinamento e validação 2:
    1.1639 segundos  1 ( 6)           0.0383          0.0002  ( 6)          0.0393          0.0011
    0.8566 segundos  1 (11)           0.0383          0.0002  (11)          0.0393          0.0011
    0.7337 segundos  1 (23)           0.0383          0.0002  (23)          0.0393          0.0011
    3.5369 segundos  1 (27)           0.0383          0.0002  (27)          0.0393          0.0011
    1.9457 segundos  1 (28)           0.0383          0.0002  (28)          0.0393          0.0011
    1.7896 segundos  1 (29)           0.0383          0.0002  (29)          0.0393          0.0011
    2.1183 segundos  2 (12)           0.0337          0.0001  (12)          0.0367          0.0002
    2.2853 segundos  2 (16)           0.0332          0.0000  (16)          0.0363          0.0021
    4.4110 segundos  3 ( 0)           0.0314          0.0001  (25)          0.0348          0.0018
   13.4472 segundos  5 (25)           0.0304          0.0001  ( 0)          0.0332          0.0014
    0.0000 segundos  0 ( 1)           0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)           0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)           0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)           0.0000          0.0000  ( 4)          0.0000          0.0000
```

```
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 2:

k =

     5     4     1     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     4     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     4     0     0     0     0
     4     2     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     5     4     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
```

```
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     4     0     0     0     0
     0     0     0     0     0
     5     4     3     2     1
     0     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0
     4     0     0     0     0

// Problema = DeathValley, Quadrado = 0, Execução 1
vip =

  Columns 1 through 9

    1.1697    1.1468    1.0779    0.9873    0.9003    0.8387    0.8175    0.8422    0.9072

  Columns 10 through 12

    0.9910    1.0779    1.1496

ell = 1, K = 12, antes = 12
ell = 2, K = 53, antes = 66
ell = 3, K = 141, antes = 199
ell = 4, K = 241, antes = 420
ell = 5, K = 325, antes = 663
ell = 6, K = 359, antes = 771
ell = 7, K = 364, antes = 796
ell = 8, K = 364, antes = 801
ell = 9, K = 364, antes = 801
ell = 1, K = 12, antes = 12
ell = 2, K = 53, antes = 66
ell = 3, K = 141, antes = 199
ell = 4, K = 241, antes = 420
ell = 5, K = 325, antes = 663
ell = 6, K = 359, antes = 771
ell = 7, K = 364, antes = 796
ell = 8, K = 364, antes = 801
ell = 9, K = 364, antes = 801
```

```
ell = 1, K = 12, antes = 12
ell = 2, K = 16, antes = 66
ell = 3, K = 16, antes = 75
ell = 4, K = 14, antes = 78
ell = 5, K = 13, antes = 78
ell = 6, K = 13, antes = 78
// Erros de treinamento e validação 3:
    0.7028 segundos  1 (11)          0.1588          0.0000  (11)          0.0953          0.0052
    0.4993 segundos  1 (16)          0.1588          0.0000  (16)          0.0953          0.0052
    0.9798 segundos  1 ( 6)          0.0646          0.0003  (25)          0.0691          0.0044
    9.5095 segundos 10 (25)          0.0593          0.0009  ( 0)          0.0470          0.0010
    2.8801 segundos  2 (29)          0.0581          0.0005  (28)          0.0440          0.0007
    2.7771 segundos  5 (12)          0.0566          0.0001  ( 6)          0.0415          0.0000
    4.3876 segundos  7 ( 0)          0.0555          0.0000  (27)          0.0389          0.0005
   78.7574 segundos  4 (28)          0.0554          0.0002  (12)          0.0385          0.0001
    1.1982 segundos  3 (23)          0.0553          0.0003  (29)          0.0369          0.0004
  207.1703 segundos  4 (27)          0.0536          0.0002  (23)          0.0356          0.0002
    0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 3:

k =
```

```
    12    11     8     7     3     2     1     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     1     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
    12     0     0     0     0     0     0     0     0     0     0     0
    12    11     3     2     1     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
    12     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
    12     2     1     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
    10     9     8     7     6     5     4     3     2     1     0     0
     0     0     0     0     0     0     0     0     0     0     0     0
    12     6     2     1     0     0     0     0     0     0     0     0
    12     7     2     1     0     0     0     0     0     0     0     0
    12     2     0     0     0     0     0     0     0     0     0     0

// Problema = eOGS Classification, Quadrado = 0, Execução 1
vip =

  Columns 1 through 9

    0.9642    1.1130    0.9954    0.9125    0.6437    0.7694    0.8627    0.8327    0.8785

  Columns 10 through 14
```

```
    1.0591    1.2430    1.1280    1.1819    1.2068

ell = 1, K = 14, antes = 14
ell = 2, K = 74, antes = 95
ell = 3, K = 195, antes = 310
ell = 4, K = 334, antes = 584
ell = 5, K = 382, antes = 799
ell = 6, K = 368, antes = 906
ell = 7, K = 321, antes = 942
ell = 8, K = 283, antes = 948
ell = 9, K = 269, antes = 950
ell = 10, K = 266, antes = 950
ell = 11, K = 266, antes = 950
ell = 1, K = 14, antes = 14
ell = 2, K = 74, antes = 95
ell = 3, K = 195, antes = 310
ell = 4, K = 334, antes = 584
ell = 5, K = 382, antes = 799
ell = 6, K = 368, antes = 906
ell = 7, K = 321, antes = 942
ell = 8, K = 283, antes = 948
ell = 9, K = 269, antes = 950
ell = 10, K = 266, antes = 950
ell = 11, K = 266, antes = 950
ell = 1, K = 14, antes = 14
ell = 2, K = 22, antes = 95
ell = 3, K = 21, antes = 131
ell = 4, K = 19, antes = 136
ell = 5, K = 19, antes = 136
// Erros de treinamento e validação 4:
    4.6542 segundos  1 (11)          0.0912          0.0013  ( 6)          0.4441          0.0079
    3.9020 segundos  1 (16)          0.0912          0.0013  (11)          0.4244          0.0110
  423.9417 segundos  2 (28)          0.0847          0.0000  (16)          0.4244          0.0110
   10.2712 segundos  1 ( 6)          0.0841          0.0005  (28)          0.4193          0.0167
   38.3673 segundos  6 (12)          0.0786          0.0004  (29)          0.3884          0.0187
  204.8977 segundos 14 (25)          0.0774          0.0000  (27)          0.3652          0.0065
   61.6363 segundos  9 ( 0)          0.0773          0.0004  (23)          0.3482          0.0047
   38.5668 segundos  2 (29)          0.0747          0.0000  (12)          0.3432          0.0041
   51.6051 segundos  7 (23)          0.0731          0.0002  (25)          0.3219          0.0055
```

```
 1557.9271 segundos  3 (27)          0.0691          0.0007  ( 0)          0.3172          0.0106
    0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 4:

k =

    14    13    12    11    10     8     5     4     1     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    13     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    14     0     0     0     0     0     0     0     0     0     0     0     0     0
    14    13    12    11    10     2     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
```

```
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    14     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    14    13    11    10     6     5     1     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    14    13    12    11    10     9     8     7     6     5     4     3     2     1
     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    12    11     7     0     0     0     0     0     0     0     0     0     0     0
    14     1     0     0     0     0     0     0     0     0     0     0     0     0
    13    10     0     0     0     0     0     0     0     0     0     0     0     0

// Problema = eOGS Prediction, Quadrado = 0, Execução 1
vip =

    0.1913    0.3945    0.1645    1.4519    1.4671    1.2330

ell = 1, K = 6, antes = 6
ell = 2, K = 16, antes = 21
ell = 3, K = 25, antes = 40
ell = 4, K = 27, antes = 51
ell = 5, K = 25, antes = 55
ell = 6, K = 24, antes = 55
ell = 1, K = 6, antes = 6
ell = 2, K = 16, antes = 21
ell = 3, K = 25, antes = 40
ell = 4, K = 27, antes = 51
ell = 5, K = 25, antes = 55
ell = 6, K = 24, antes = 55
ell = 1, K = 6, antes = 6
ell = 2, K = 6, antes = 21
ell = 3, K = 8, antes = 25
ell = 4, K = 8, antes = 29
ell = 5, K = 8, antes = 29
// Erros de treinamento e validação 5:
    1.4172 segundos  1 (16)           0.1158           0.0021  (16)           0.1089           0.0085
```

```
     1.7332 segundos  1 ( 6)          0.1073          0.0002  ( 6)          0.0840          0.0038
     4.5749 segundos  3 (12)          0.0748          0.0003  (12)          0.0666          0.0016
     2.8244 segundos  2 (11)          0.0684          0.0019  ( 0)          0.0460          0.0013
     5.3830 segundos  2 (29)          0.0684          0.0019  (28)          0.0460          0.0013
     4.7092 segundos  3 (23)          0.0674          0.0010  (11)          0.0448          0.0010
     5.4805 segundos  5 ( 0)          0.0664          0.0019  (29)          0.0448          0.0010
    16.3575 segundos  5 (28)          0.0664          0.0019  (27)          0.0444          0.0012
    16.4048 segundos  6 (25)          0.0614          0.0011  (23)          0.0435          0.0011
    44.5887 segundos  4 (27)          0.0600          0.0015  (25)          0.0422          0.0003
     0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
     0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
     0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
     0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
     0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
     0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
     0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
     0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
     0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
     0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
     0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
     0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
     0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
     0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
     0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
     0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
     0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
     0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
     0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
     0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 5:

k =

     6     5     3     2     1     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     5     0     0     0     0     0
```

```
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     5     1     0     0     0     0
     6     5     4     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     6     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     0     0     0     0     0     0
     5     2     1     0     0     0
     0     0     0     0     0     0
     6     5     4     3     2     1
     0     0     0     0     0     0
     6     4     3     1     0     0
     6     5     3     2     1     0
     5     1     0     0     0     0

// Problema = FBeM BoxJenkins, Quadrado = 0, Execução 1
vip =

    0.9242    0.9276    0.9379    0.9463    1.2289

ell = 1, K = 5, antes = 5
ell = 2, K = 10, antes = 14
ell = 3, K = 9, antes = 21
ell = 4, K = 7, antes = 22
ell = 5, K = 7, antes = 22
ell = 1, K = 5, antes = 5
ell = 2, K = 10, antes = 14
ell = 3, K = 9, antes = 21
ell = 4, K = 7, antes = 22
ell = 5, K = 7, antes = 22
ell = 1, K = 5, antes = 5
```

```
ell = 2, K = 5, antes = 14
ell = 3, K = 2, antes = 19
ell = 4, K = 1, antes = 19
ell = 5, K = 1, antes = 19
// Erros de treinamento e validação 6:
    0.2087 segundos  1 ( 6)          0.0745          0.0020  ( 6)          0.0711          0.0046
    0.2053 segundos  1 (11)          0.0745          0.0020  (11)          0.0711          0.0046
    0.1947 segundos  1 (12)          0.0745          0.0020  (12)          0.0711          0.0046
    0.2155 segundos  1 (16)          0.0745          0.0020  (16)          0.0711          0.0046
    0.2765 segundos  1 (29)          0.0745          0.0020  (29)          0.0711          0.0046
    0.2645 segundos  2 (23)          0.0461          0.0023  (23)          0.0641          0.0038
    0.3654 segundos  3 ( 0)          0.0369          0.0024  ( 0)          0.0486          0.0031
    0.8316 segundos  3 (27)          0.0369          0.0024  (27)          0.0486          0.0031
    0.4658 segundos  3 (28)          0.0369          0.0024  (28)          0.0486          0.0031
    0.7362 segundos  5 (25)          0.0346          0.0009  (25)          0.0483          0.0017
    0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 6:

k =

     5     3     1     0     0
```

```
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     5     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     5     0     0     0     0
     5     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     5     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     5     1     0     0     0
     0     0     0     0     0
     5     4     3     2     1
     0     0     0     0     0
     5     3     1     0     0
     5     3     1     0     0
     5     0     0     0     0

// Problema = FBeM Global40, Quadrado = 0, Execução 1
vip =

    0.9997    1.0003

ell = 1, K = 2, antes = 2
ell = 2, K = 2, antes = 2
ell = 1, K = 2, antes = 2
ell = 2, K = 2, antes = 2
ell = 1, K = 2, antes = 2
```

```
ell = 2, K = 2, antes = 2
// Erros de treinamento e validação 7:
    0.5392 segundos  1 (11)          0.0153          0.0000  (11)          0.0626          0.0013
    0.9623 segundos  2 ( 0)          0.0129          0.0000  ( 0)          0.0492          0.0011
    0.9377 segundos  2 (25)          0.0129          0.0000  (25)          0.0492          0.0011
    0.4864 segundos  1 ( 6)          0.0125          0.0001  ( 6)          0.0458          0.0017
    0.5064 segundos  1 (12)          0.0125          0.0001  (12)          0.0458          0.0017
    0.4739 segundos  1 (16)          0.0125          0.0001  (16)          0.0458          0.0017
    0.5040 segundos  1 (23)          0.0125          0.0001  (23)          0.0458          0.0017
    0.7749 segundos  1 (27)          0.0125          0.0001  (27)          0.0458          0.0017
    0.4960 segundos  1 (28)          0.0125          0.0001  (28)          0.0458          0.0017
    0.5006 segundos  1 (29)          0.0125          0.0001  (29)          0.0458          0.0017
    0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 7:

k =

     2     1
     0     0
     0     0
     0     0
```

```
     0     0
     0     0
     2     0
     0     0
     0     0
     0     0
     0     0
     1     0
     2     0
     0     0
     0     0
     0     0
     2     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     2     0
     0     0
     2     1
     0     0
     2     0
     2     0
     2     0

// Problema = FBeM Mackey Glass, Quadrado = 0, Execução 1
vip =

    0.9916    0.9972    1.0028    1.0083

ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
```

```
ell = 1, K = 4, antes = 4
ell = 2, K = 6, antes = 10
ell = 3, K = 6, antes = 14
ell = 4, K = 7, antes = 15
// Erros de treinamento e validação 8:
    1.3600 segundos  1 ( 6)          0.2024          0.0007  ( 6)          2.9094          0.0250
    1.2679 segundos  1 (11)          0.2024          0.0007  (11)          2.9094          0.0250
    1.1290 segundos  1 (16)          0.2024          0.0007  (16)          2.9094          0.0250
    5.4601 segundos  2 (29)          0.0136          0.0001  (23)          0.1028          0.0107
    3.4994 segundos  2 (12)          0.0130          0.0002  (29)          0.0990          0.0098
    7.8687 segundos  3 (23)          0.0112          0.0005  (27)          0.0966          0.0068
   14.5958 segundos  4 (25)          0.0111          0.0002  ( 0)          0.0947          0.0012
   15.6375 segundos  4 (28)          0.0111          0.0002  (12)          0.0929          0.0040
    9.6172 segundos  2 ( 0)          0.0111          0.0003  (25)          0.0915          0.0019
   22.6148 segundos  2 (27)          0.0107          0.0006  (28)          0.0915          0.0019
    0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 8:

k =

     4     1     0     0
```

```
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     4     3     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     3     2     0
     0     0     0     0
     4     3     2     1
     0     0     0     0
     4     2     0     0
     4     3     2     1
     3     2     0     0

// Problema = Classificacao 2, Quadrado = 0, Execução 1
vip =

    0.9367    1.0231    0.9737    1.0620

ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
ell = 1, K = 4, antes = 4
```

```
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
ell = 1, K = 4, antes = 4
ell = 2, K = 4, antes = 10
ell = 3, K = 2, antes = 14
ell = 4, K = 1, antes = 14
// Erros de treinamento e validação 10:
    0.0936 segundos  2 (12)          0.6738          0.0193  ( 0)          0.2243          0.0090
    0.0709 segundos  2 (23)          0.6738          0.0193  (25)          0.2243          0.0090
    0.5761 segundos  4 ( 0)          0.6020          0.0005  (28)          0.2243          0.0090
    0.3157 segundos  4 (25)          0.6020          0.0005  (27)          0.2174          0.0149
    0.2736 segundos  4 (28)          0.6020          0.0005  (12)          0.2016          0.0122
    0.2147 segundos  1 ( 6)          0.5765          0.0011  (23)          0.2016          0.0122
    0.0435 segundos  1 (11)          0.5765          0.0011  ( 6)          0.1733          0.0230
    0.0855 segundos  1 (16)          0.5765          0.0011  (11)          0.1733          0.0230
    0.0998 segundos  1 (29)          0.5765          0.0011  (16)          0.1733          0.0230
    0.3811 segundos  2 (27)          0.5654          0.0019  (29)          0.1733          0.0230
    0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 10:
```

```
k =

     4     3     2     1
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     4     2     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     2     0     0
     0     0     0     0
     4     3     2     1
     0     0     0     0
     4     3     0     0
     4     3     2     1
     4     0     0     0

// Problema = Classificacao 5, Quadrado = 0, Execução 1
vip =

    1.1011    1.0323    0.8583    0.9927

ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
```

```
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 10
ell = 3, K = 8, antes = 14
ell = 4, K = 8, antes = 15
ell = 1, K = 4, antes = 4
ell = 2, K = 6, antes = 10
ell = 3, K = 7, antes = 12
ell = 4, K = 7, antes = 12
// Erros de treinamento e validação 11:
    0.0409 segundos  1 (16)          0.4156          0.0066  (16)          0.1706          0.0089
    0.0720 segundos  1 ( 6)          0.3733          0.0080  ( 0)          0.1673          0.0172
    0.0406 segundos  1 (11)          0.3733          0.0080  (25)          0.1673          0.0172
    0.1048 segundos  3 (23)          0.3700          0.0060  (27)          0.1673          0.0172
    0.2622 segundos  3 (29)          0.3700          0.0060  (28)          0.1673          0.0172
    0.0914 segundos  2 (12)          0.3679          0.0012  ( 6)          0.1553          0.0007
    0.1570 segundos  4 ( 0)          0.3623          0.0007  (11)          0.1553          0.0007
    0.1974 segundos  4 (25)          0.3623          0.0007  (23)          0.1479          0.0068
    0.4917 segundos  4 (27)          0.3623          0.0007  (29)          0.1479          0.0068
    0.5228 segundos  4 (28)          0.3623          0.0007  (12)          0.1389          0.0039
    0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
```

```
    0.0000 segundos  0 (26)           0.0000           0.0000  (26)           0.0000           0.0000
// Colunas utilizadas 11:

k =

     4     3     2     1
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     1     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     1     0     0     0
     2     1     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     2     1     0
     0     0     0     0
     4     3     2     1
     0     0     0     0
     4     3     2     1
     4     3     2     1
     4     2     1     0
```

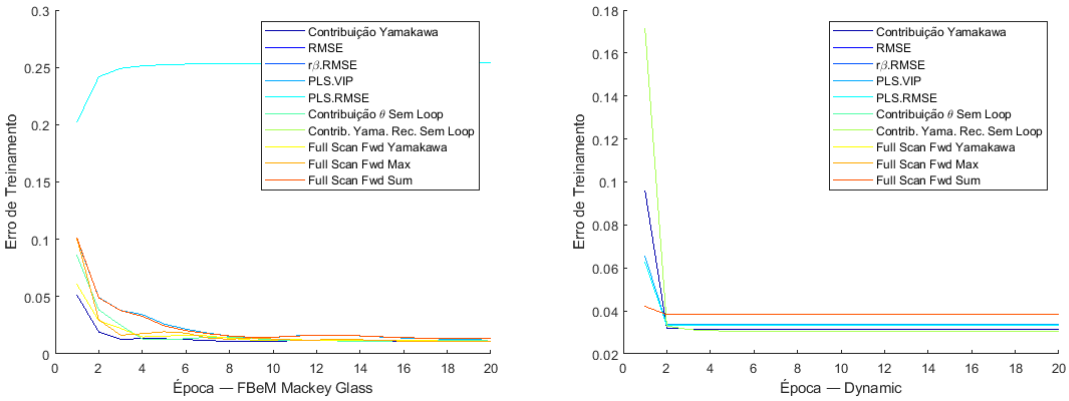

Figura 18: (1) FBeM Mackey Glass (2) Dynamic

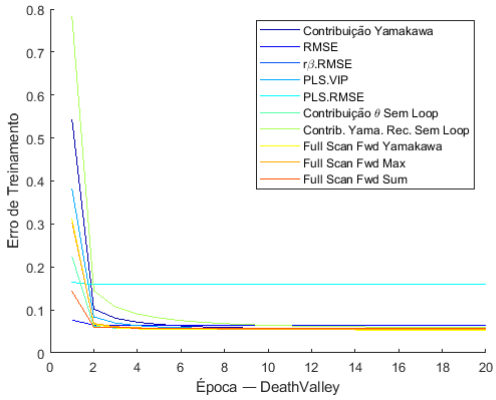

Figura 19: DeathValley

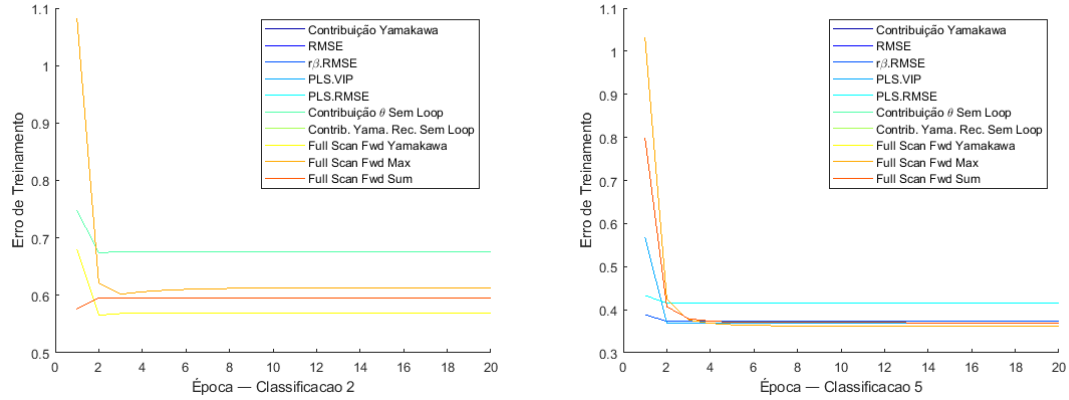

Figura 20: (1) Classificação Binária (2) 5 Classes

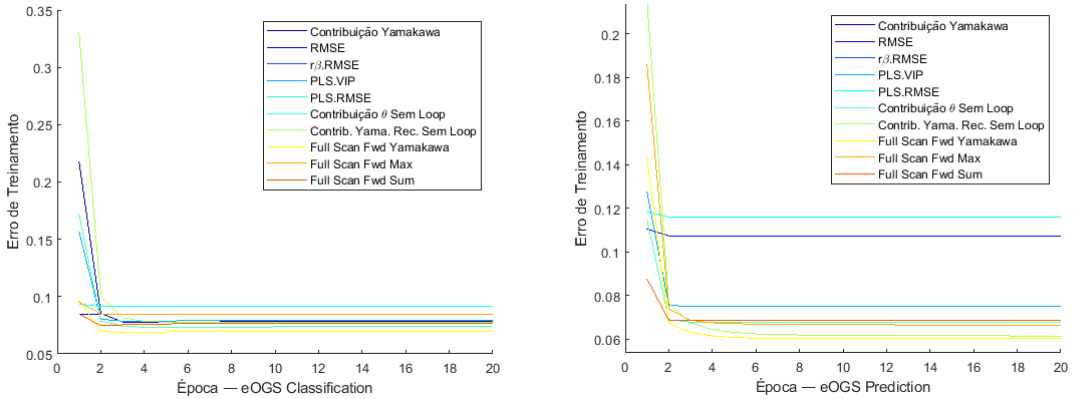

Figura 21: (1) EyeState (2) Parkinsons

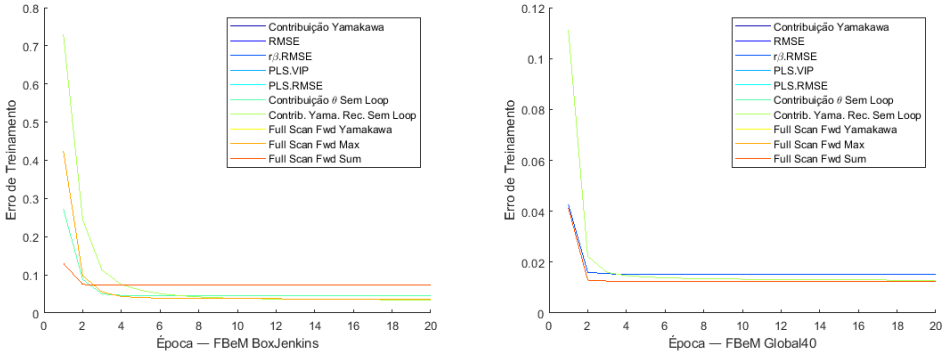

Figura 22: (1) BoxJenkins (2) Global40PxLast

8.7 Execução 7

Aqui a flagQuadrado e a flagPCA são verdadeiras.

```
// Problema = Dynamic, Quadrado = 1, Execução 1
vip =

    0.0622    0.1919    1.2340    2.8989    0.1563    0.0190    0.0838    0.0159    0.0245    0.0146

ell = 1, K = 10, antes = 10
```

```
ell = 2, K = 13, antes = 18
ell = 3, K = 14, antes = 26
ell = 4, K = 15, antes = 27
ell = 5, K = 15, antes = 27
ell = 1, K = 10, antes = 10
ell = 2, K = 13, antes = 18
ell = 3, K = 14, antes = 26
ell = 4, K = 15, antes = 27
ell = 5, K = 15, antes = 27
ell = 1, K = 10, antes = 10
ell = 2, K = 11, antes = 18
ell = 3, K = 10, antes = 24
ell = 4, K = 10, antes = 24
// Erros de treinamento e validação 2:
    1.1624 segundos  1 (16)          0.4462          0.0019  (16)          0.4339          0.0245
    2.6460 segundos  1 ( 6)          0.1026          0.0012  ( 6)          0.0677          0.0054
    1.3953 segundos  1 (11)          0.1026          0.0012  (11)          0.0677          0.0054
    5.3850 segundos  1 (29)          0.1026          0.0012  (29)          0.0677          0.0054
   11.3358 segundos  2 ( 0)          0.0419          0.0005  ( 0)          0.0384          0.0007
    2.1936 segundos  2 (12)          0.0419          0.0005  (12)          0.0384          0.0007
    2.3159 segundos  2 (23)          0.0419          0.0005  (23)          0.0384          0.0007
   24.0544 segundos  2 (27)          0.0419          0.0005  (27)          0.0384          0.0007
    9.5757 segundos  2 (28)          0.0419          0.0005  (28)          0.0384          0.0007
   41.2768 segundos  9 (25)          0.0306          0.0001  (25)          0.0319          0.0019
    0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
```

```
    0.0000 segundos  0 (21)           0.0000           0.0000  (21)           0.0000           0.0000
    0.0000 segundos  0 (22)           0.0000           0.0000  (22)           0.0000           0.0000
    0.0000 segundos  0 (24)           0.0000           0.0000  (24)           0.0000           0.0000
    0.0000 segundos  0 (26)           0.0000           0.0000  (26)           0.0000           0.0000
// Colunas utilizadas 2:

k =

     4     3     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     4     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     4     0     0     0     0     0     0     0     0     0
     4     3     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
    10     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     4     3     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
    10     9     8     7     5     4     3     2     1     0
     0     0     0     0     0     0     0     0     0     0
     4     3     0     0     0     0     0     0     0     0
     4     3     0     0     0     0     0     0     0     0
     4     0     0     0     0     0     0     0     0     0

// Problema = DeathValley, Quadrado = 1, Execução 1
```

```
vip =

    1.3635    1.0674    0.0406

ell = 1, K = 3, antes = 3
ell = 2, K = 3, antes = 4
ell = 3, K = 3, antes = 4
ell = 1, K = 3, antes = 3
ell = 2, K = 3, antes = 4
ell = 3, K = 3, antes = 4
ell = 1, K = 3, antes = 3
ell = 2, K = 3, antes = 4
ell = 3, K = 3, antes = 4
// Erros de treinamento e validação 3:
    0.7113 segundos  1 (11)          0.4469          0.0017  (11)          0.2602          0.0157
    0.4953 segundos  1 (16)          0.4469          0.0017  (16)          0.2602          0.0157
    0.6241 segundos  1 ( 6)          0.1858          0.0041  ( 6)          0.1054          0.0094
    3.3122 segundos  3 ( 0)          0.0562          0.0001  (12)          0.0474          0.0008
    1.2793 segundos  3 (25)          0.0562          0.0001  (23)          0.0474          0.0008
    0.9038 segundos  2 (12)          0.0545          0.0001  (27)          0.0474          0.0008
    1.3769 segundos  2 (23)          0.0545          0.0001  (28)          0.0474          0.0008
    1.1843 segundos  2 (27)          0.0545          0.0001  (29)          0.0474          0.0008
    0.7684 segundos  2 (28)          0.0545          0.0001  ( 0)          0.0450          0.0018
    0.7353 segundos  2 (29)          0.0545          0.0001  (25)          0.0450          0.0018
    0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
```

```
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 3:

k =

     3     2     1
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     1     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     3     0     0
     2     1     0
     0     0     0
     0     0     0
     0     0     0
     3     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     2     1     0
     0     0     0
     3     2     1
     0     0     0
     2     1     0
     2     1     0
     2     1     0

// Problema = eOGS Classification, Quadrado = 1, Execução 1
```

```
vip =

    1.0785    0.0505    1.3544

ell = 1, K = 3, antes = 3
ell = 2, K = 3, antes = 5
ell = 3, K = 3, antes = 5
ell = 1, K = 3, antes = 3
ell = 2, K = 3, antes = 5
ell = 3, K = 3, antes = 5
ell = 1, K = 3, antes = 3
ell = 2, K = 3, antes = 5
ell = 3, K = 3, antes = 5
// Erros de treinamento e validação 4:
    8.3384 segundos  2 (28)          0.0802          0.0004  ( 0)          0.3959          0.0065
    8.3196 segundos  2 (29)          0.0802          0.0004  (12)          0.3959          0.0065
    4.7486 segundos  1 ( 6)          0.0788          0.0012  (27)          0.3959          0.0065
    4.8842 segundos  1 (11)          0.0788          0.0012  (25)          0.3712          0.0012
    4.5726 segundos  1 (16)          0.0788          0.0012  (28)          0.3420          0.0048
    4.6401 segundos  1 (23)          0.0788          0.0012  (29)          0.3420          0.0048
   14.1388 segundos  3 (25)          0.0783          0.0008  ( 6)          0.3339          0.0077
   11.0883 segundos  2 ( 0)          0.0734          0.0015  (11)          0.3339          0.0077
    7.6421 segundos  2 (12)          0.0734          0.0015  (16)          0.3339          0.0077
   13.7854 segundos  2 (27)          0.0734          0.0015  (23)          0.3339          0.0077
    0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
```

```
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 4:

k =

     3     1     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     3     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     3     0     0
     3     1     0
     0     0     0
     0     0     0
     0     0     0
     3     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     0     0     0
     3     0     0
     0     0     0
     3     2     1
     0     0     0
     3     1     0
     3     2     0
     3     2     0

// Problema = eOGS Prediction, Quadrado = 1, Execução 1
```

```
vip =

    1.6078    1.0592    0.5374    0.0656

ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 9
ell = 3, K = 8, antes = 12
ell = 4, K = 8, antes = 13
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 9
ell = 3, K = 8, antes = 12
ell = 4, K = 8, antes = 13
ell = 1, K = 4, antes = 4
ell = 2, K = 7, antes = 9
ell = 3, K = 8, antes = 12
ell = 4, K = 8, antes = 13
// Erros de treinamento e validação 5:
    1.3972 segundos  1 (16)           0.0947          0.0007  (16)          0.0924          0.0013
    1.4662 segundos  1 (11)           0.0773          0.0000  (11)          0.0919          0.0024
    1.5094 segundos  1 ( 6)           0.0698          0.0003  ( 6)          0.0656          0.0033
    2.5804 segundos  2 (12)           0.0692          0.0001  (12)          0.0570          0.0009
    4.7324 segundos  3 (23)           0.0637          0.0001  (23)          0.0483          0.0000
    4.2775 segundos  4 ( 0)           0.0631          0.0001  ( 0)          0.0483          0.0002
    8.0172 segundos  4 (25)           0.0631          0.0001  (25)          0.0483          0.0002
   16.1974 segundos  4 (27)           0.0631          0.0001  (27)          0.0483          0.0002
    8.5750 segundos  4 (28)           0.0631          0.0001  (28)          0.0483          0.0002
    8.6258 segundos  4 (29)           0.0631          0.0001  (29)          0.0483          0.0002
    0.0000 segundos  0 ( 1)           0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)           0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)           0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)           0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)           0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)           0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)           0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)           0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)           0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)           0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)           0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)           0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)           0.0000          0.0000  (17)          0.0000          0.0000
```

```
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 5:

k =

     4     3     2     1
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     1     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     3     0     0     0
     2     1     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     3     2     1     0
     0     0     0     0
     4     3     2     1
     0     0     0     0
     4     3     2     1
     4     3     2     1
```

```
      4     3     2     1

// Problema = FBeM BoxJenkins, Quadrado = 1, Execução 1
vip =

    0.7937    1.7885    0.3636    0.1973

ell = 1, K = 4, antes = 4
ell = 2, K = 4, antes = 5
ell = 3, K = 4, antes = 6
ell = 4, K = 4, antes = 6
ell = 1, K = 4, antes = 4
ell = 2, K = 4, antes = 5
ell = 3, K = 4, antes = 6
ell = 4, K = 4, antes = 6
ell = 1, K = 4, antes = 4
ell = 2, K = 3, antes = 5
ell = 3, K = 3, antes = 5
// Erros de treinamento e validação 6:
    0.2127 segundos  1 (16)          0.1687          0.0005  (16)          0.1645          0.0085
    0.2094 segundos  1 ( 6)          0.0666          0.0007  (23)          0.0932          0.0258
    0.2060 segundos  1 (11)          0.0666          0.0007  (25)          0.0871          0.0313
    0.2118 segundos  1 (12)          0.0666          0.0007  ( 6)          0.0807          0.0025
    0.2437 segundos  1 (28)          0.0666          0.0007  (11)          0.0807          0.0025
    0.2240 segundos  1 (29)          0.0666          0.0007  (12)          0.0807          0.0025
    0.3758 segundos  3 (23)          0.0564          0.0003  (28)          0.0807          0.0025
    0.4053 segundos  3 ( 0)          0.0418          0.0008  (29)          0.0807          0.0025
    0.5647 segundos  3 (27)          0.0380          0.0012  (27)          0.0763          0.0349
    0.5512 segundos  4 (25)          0.0310          0.0005  ( 0)          0.0724          0.0042
    0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
```

```
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 6:

k =

     3     2     1     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     2     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     2     0     0     0
     2     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     3     2     0
     0     0     0     0
     4     3     2     1
     0     0     0     0
```

```
     4     2     1     0
     2     0     0     0
     2     0     0     0

// Problema = FBeM Global40, Quadrado = 1, Execução 1
vip =

     1

ell = 1, K = 1, antes = 1
ell = 1, K = 1, antes = 1
ell = 1, K = 1, antes = 1
// Erros de treinamento e validação 7:
    2.3876 segundos  1 ( 0)           0.0146          0.0000  (12)          0.1505          0.0000
    0.7692 segundos  1 ( 6)           0.0146          0.0000  ( 0)          0.0595          0.0003
    0.6850 segundos  1 (11)           0.0146          0.0000  ( 6)          0.0595          0.0003
    0.6104 segundos  1 (16)           0.0146          0.0000  (11)          0.0595          0.0003
    0.5017 segundos  1 (23)           0.0146          0.0000  (16)          0.0595          0.0003
    0.6823 segundos  1 (25)           0.0146          0.0000  (23)          0.0595          0.0003
    0.6723 segundos  1 (27)           0.0146          0.0000  (25)          0.0595          0.0003
    0.5076 segundos  1 (28)           0.0146          0.0000  (27)          0.0595          0.0003
    0.4658 segundos  1 (29)           0.0146          0.0000  (28)          0.0595          0.0003
    0.0000 segundos  0 ( 1)           0.0000          0.0000  (29)          0.0595          0.0003
    0.0000 segundos  0 ( 2)           0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 3)           0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 4)           0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 5)           0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 7)           0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 8)           0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 9)           0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 (10)           0.0000          0.0000  ( 9)          0.0000          0.0000
    0.3695 segundos  0 (12)           0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)           0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)           0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)           0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)           0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)           0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)           0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)           0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)           0.0000          0.0000  (21)          0.0000          0.0000
```

```
    0.0000 segundos  0 (22)           0.0000           0.0000  (22)           0.0000           0.0000
    0.0000 segundos  0 (24)           0.0000           0.0000  (24)           0.0000           0.0000
    0.0000 segundos  0 (26)           0.0000           0.0000  (26)           0.0000           0.0000
// Colunas utilizadas 7:

k =

     1     0     0     0     0     0     1     0     0     0     0     1     0     0     0     0
     1     0     0     0     0     0     0     1     0     1     0     1     1     1

// Problema = FBeM Mackey Glass, Quadrado = 1, Execução 1
vip =

     1

ell = 1, K = 1, antes = 1
ell = 1, K = 1, antes = 1
ell = 1, K = 1, antes = 1
// Erros de treinamento e validação 8:
    2.1008 segundos  1 ( 0)           0.2195           0.0016  ( 0)           2.9334           0.0022
    0.9918 segundos  1 ( 6)           0.2195           0.0016  ( 6)           2.9334           0.0022
    0.9926 segundos  1 (11)           0.2195           0.0016  (11)           2.9334           0.0022
    1.1076 segundos  1 (16)           0.2195           0.0016  (16)           2.9334           0.0022
    1.0392 segundos  1 (23)           0.2195           0.0016  (23)           2.9334           0.0022
    1.5724 segundos  1 (25)           0.2195           0.0016  (25)           2.9334           0.0022
    1.8629 segundos  1 (27)           0.2195           0.0016  (27)           2.9334           0.0022
    0.9815 segundos  1 (28)           0.2195           0.0016  (28)           2.9334           0.0022
    0.9795 segundos  1 (29)           0.2195           0.0016  (29)           2.9334           0.0022
    0.0000 segundos  0 ( 1)           0.0000           0.0000  (12)           0.1223           0.0000
    0.0000 segundos  0 ( 2)           0.0000           0.0000  ( 1)           0.0000           0.0000
    0.0000 segundos  0 ( 3)           0.0000           0.0000  ( 2)           0.0000           0.0000
    0.0000 segundos  0 ( 4)           0.0000           0.0000  ( 3)           0.0000           0.0000
    0.0000 segundos  0 ( 5)           0.0000           0.0000  ( 4)           0.0000           0.0000
    0.0000 segundos  0 ( 7)           0.0000           0.0000  ( 5)           0.0000           0.0000
    0.0000 segundos  0 ( 8)           0.0000           0.0000  ( 7)           0.0000           0.0000
    0.0000 segundos  0 ( 9)           0.0000           0.0000  ( 8)           0.0000           0.0000
    0.0000 segundos  0 (10)           0.0000           0.0000  ( 9)           0.0000           0.0000
    0.1289 segundos  0 (12)           0.0000           0.0000  (10)           0.0000           0.0000
    0.0000 segundos  0 (13)           0.0000           0.0000  (13)           0.0000           0.0000
    0.0000 segundos  0 (14)           0.0000           0.0000  (14)           0.0000           0.0000
```

```
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 8:

k =

     1     0     0     0     0     0     1     0     0     0     0     1     0     0     0     0
     1     0     0     0     0     0     0     1     0     1     0     1     1     1

// Problema = ArbitrNVar, Quadrado = 1, Execução 1
vip =

  Columns 1 through 9

    0.9424    1.1010    0.8294    0.5360    1.5813    1.2604    2.0908    1.4056    2.1908

  Columns 10 through 18

    0.1543    0.0110    0.3244    0.2926    0.7014    1.1901    1.5394    0.1961    0.8703

  Columns 19 through 27

    1.0849    1.6989    0.2537    0.4321    1.6829    0.4292    1.7079    0.1832    0.0634

  Columns 28 through 34

    0.0703    0.0252    0.0946    0.0153    0.0424    0.0758    0.3043

ell = 1, K = 34, antes = 34
ell = 2, K = 34, antes = 34
ell = 1, K = 34, antes = 34
ell = 2, K = 34, antes = 34
ell = 1, K = 34, antes = 34
```

```
ell = 2, K = 34, antes = 34
// Erros de treinamento e validação 9:
  109.4230 segundos 34 (25)           0.1674          0.0005  (25)          0.0748          0.0024
   25.3211 segundos 16 (23)           0.0578          0.0011  (23)          0.0090          0.0003
   30.6837 segundos 15 ( 0)           0.0555          0.0012  ( 0)          0.0085          0.0016
    8.9769 segundos  1 ( 6)           0.0505          0.0001  (12)          0.0048          0.0004
    2.2608 segundos  1 (16)           0.0505          0.0001  ( 6)          0.0036          0.0001
    4.8078 segundos  1 (28)           0.0505          0.0001  (16)          0.0036          0.0001
    4.7655 segundos  1 (29)           0.0505          0.0001  (28)          0.0036          0.0001
   14.9552 segundos 12 (12)           0.0472          0.0006  (29)          0.0036          0.0001
    2.7543 segundos  1 (11)           0.0401          0.0007  (27)          0.0025          0.0000
    8.2760 segundos  1 (27)           0.0391          0.0001  (11)          0.0024          0.0001
    0.0000 segundos  0 ( 1)           0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)           0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)           0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)           0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)           0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)           0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)           0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)           0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)           0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)           0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)           0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)           0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)           0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)           0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)           0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)           0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)           0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)           0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)           0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)           0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 9:

k =

  Columns 1 through 16

    34    33    32    31    30    28    27    26    25    24    23    22    14     8     3     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
```

```
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    34     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    32     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    25    23    20    19    16    15     9     8     7     6     5     2     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    34     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    34    33    32    30    28    27    26    25    23    20    19    16     9     8     7     5
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    34    33    32    31    30    29    28    27    26    25    24    23    22    21    20    19
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     9     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    34     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
    34     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0

Columns 17 through 32

     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
```

```
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
  18    17    16    15    14    13    12    11    10     9     8     7     6     5     4     3
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
   0     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0

Columns 33 through 34

   0     0
   0     0
   0     0
   0     0
   0     0
   0     0
   0     0
   0     0
   0     0
   0     0
   0     0
   0     0
   0     0
   0     0
   0     0
   0     0
```

```
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     0     0
     2     1
     0     0
     0     0
     0     0
     0     0

// Problema = Classificacao 2, Quadrado = 1, Execução 1
vip =

    1.4904    0.4472    1.2428    0.1852

ell = 1, K = 4, antes = 4
ell = 2, K = 4, antes = 4
ell = 1, K = 4, antes = 4
ell = 2, K = 4, antes = 4
ell = 1, K = 4, antes = 4
ell = 2, K = 4, antes = 4
// Erros de treinamento e validação 10:
    0.0601 segundos  2 (12)          0.7006          0.0129  ( 6)          0.3354          0.0013
    0.0520 segundos  2 (23)          0.7006          0.0129  (11)          0.3354          0.0013
    0.0896 segundos  3 ( 0)          0.6799          0.0110  (28)          0.3354          0.0013
    0.1813 segundos  4 (25)          0.6481          0.0011  (29)          0.3354          0.0013
    0.0329 segundos  1 (16)          0.5842          0.0161  (25)          0.2196          0.0109
    0.0951 segundos  1 (27)          0.5842          0.0161  ( 0)          0.2113          0.0054
    0.0290 segundos  1 ( 6)          0.5835          0.0128  (12)          0.2063          0.0049
    0.0273 segundos  1 (11)          0.5835          0.0128  (23)          0.2063          0.0049
    0.0245 segundos  1 (28)          0.5835          0.0128  (16)          0.1750          0.0072
    0.0269 segundos  1 (29)          0.5835          0.0128  (27)          0.1750          0.0072
    0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
```

```
    0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 10:

k =

     4     2     1     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     1     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     1     0     0     0
     3     1     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
```

```
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     3     1     0     0
     0     0     0     0
     4     3     2     1
     0     0     0     0
     4     0     0     0
     1     0     0     0
     1     0     0     0

// Problema = Classificacao 5, Quadrado = 1, Execução 1
vip =

    0.2434    0.9481    1.7440    0.0162

ell = 1, K = 4, antes = 4
ell = 2, K = 4, antes = 4
ell = 1, K = 4, antes = 4
ell = 2, K = 4, antes = 4
ell = 1, K = 4, antes = 4
ell = 2, K = 4, antes = 4
// Erros de treinamento e validação 11:
    0.0217 segundos  1 (16)          0.5813          0.0057  ( 6)          0.1606          0.0041
    0.0243 segundos  1 ( 6)          0.4520          0.0161  (11)          0.1606          0.0041
    0.0237 segundos  1 (11)          0.4520          0.0161  (12)          0.1606          0.0041
    0.0206 segundos  1 (12)          0.4520          0.0161  (28)          0.1606          0.0041
    0.0374 segundos  1 (28)          0.4520          0.0161  (29)          0.1606          0.0041
    0.0393 segundos  1 (29)          0.4520          0.0161  (16)          0.1479          0.0440
    0.1923 segundos  4 (25)          0.4040          0.0147  (25)          0.1447          0.0188
    0.1198 segundos  3 ( 0)          0.3709          0.0175  (27)          0.1286          0.0076
    0.0572 segundos  2 (23)          0.3640          0.0069  ( 0)          0.1285          0.0038
    0.0892 segundos  1 (27)          0.3630          0.0112  (23)          0.1252          0.0042
    0.0000 segundos  0 ( 1)          0.0000          0.0000  ( 1)          0.0000          0.0000
    0.0000 segundos  0 ( 2)          0.0000          0.0000  ( 2)          0.0000          0.0000
    0.0000 segundos  0 ( 3)          0.0000          0.0000  ( 3)          0.0000          0.0000
    0.0000 segundos  0 ( 4)          0.0000          0.0000  ( 4)          0.0000          0.0000
    0.0000 segundos  0 ( 5)          0.0000          0.0000  ( 5)          0.0000          0.0000
    0.0000 segundos  0 ( 7)          0.0000          0.0000  ( 7)          0.0000          0.0000
```

```
    0.0000 segundos  0 ( 8)          0.0000          0.0000  ( 8)          0.0000          0.0000
    0.0000 segundos  0 ( 9)          0.0000          0.0000  ( 9)          0.0000          0.0000
    0.0000 segundos  0 (10)          0.0000          0.0000  (10)          0.0000          0.0000
    0.0000 segundos  0 (13)          0.0000          0.0000  (13)          0.0000          0.0000
    0.0000 segundos  0 (14)          0.0000          0.0000  (14)          0.0000          0.0000
    0.0000 segundos  0 (15)          0.0000          0.0000  (15)          0.0000          0.0000
    0.0000 segundos  0 (17)          0.0000          0.0000  (17)          0.0000          0.0000
    0.0000 segundos  0 (18)          0.0000          0.0000  (18)          0.0000          0.0000
    0.0000 segundos  0 (19)          0.0000          0.0000  (19)          0.0000          0.0000
    0.0000 segundos  0 (20)          0.0000          0.0000  (20)          0.0000          0.0000
    0.0000 segundos  0 (21)          0.0000          0.0000  (21)          0.0000          0.0000
    0.0000 segundos  0 (22)          0.0000          0.0000  (22)          0.0000          0.0000
    0.0000 segundos  0 (24)          0.0000          0.0000  (24)          0.0000          0.0000
    0.0000 segundos  0 (26)          0.0000          0.0000  (26)          0.0000          0.0000
// Colunas utilizadas 11:

k =

     3     2     1     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     3     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     3     0     0     0
     3     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     4     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
```

0	0	0	0
0	0	0	0
0	0	0	0
3	2	0	0
0	0	0	0
4	3	2	1
0	0	0	0
2	0	0	0
3	0	0	0
3	0	0	0

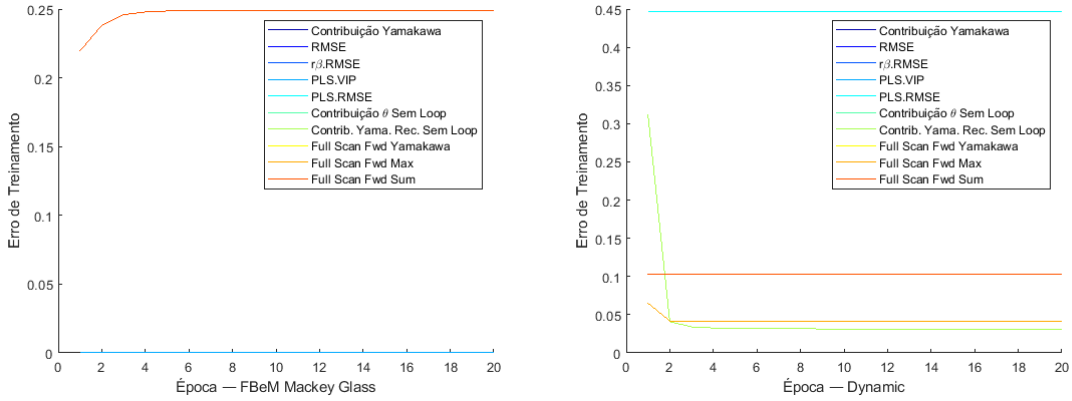

Figura 23: (1) FBeM Mackey Glass (2) Dynamic

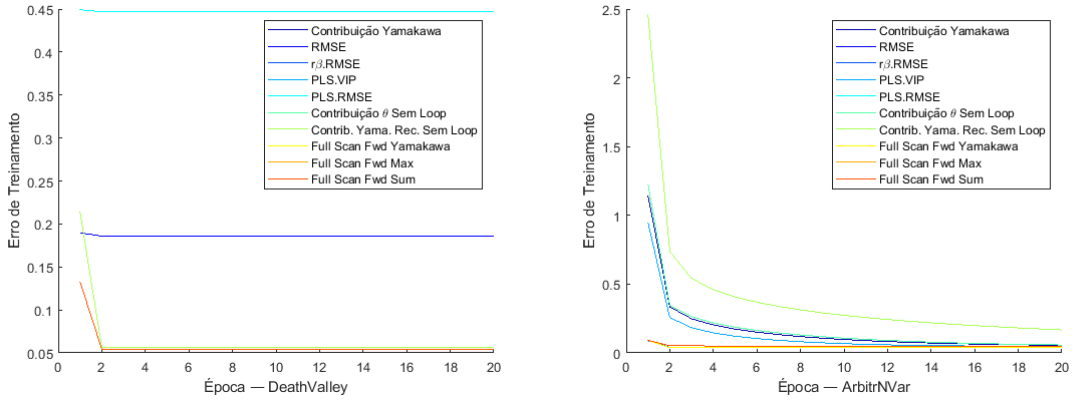

Figura 24: (1) DeathValley (2) ArbitrNVar 25

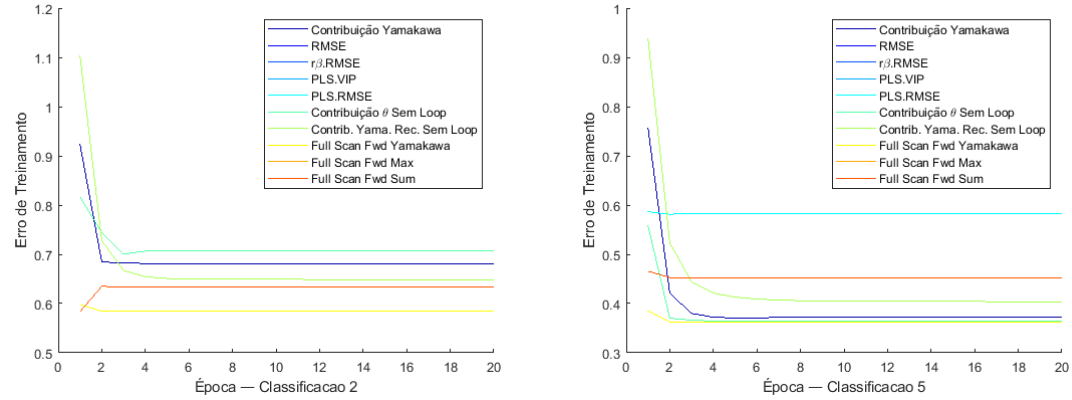

Figura 25: (1) Classificação Binária (2) 5 Classes

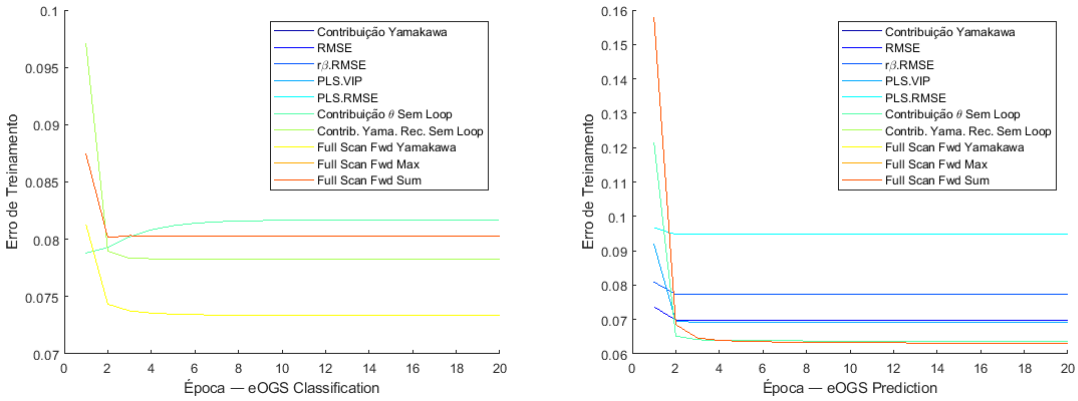

Figura 26: (1) EyeState (2) Parkinsons

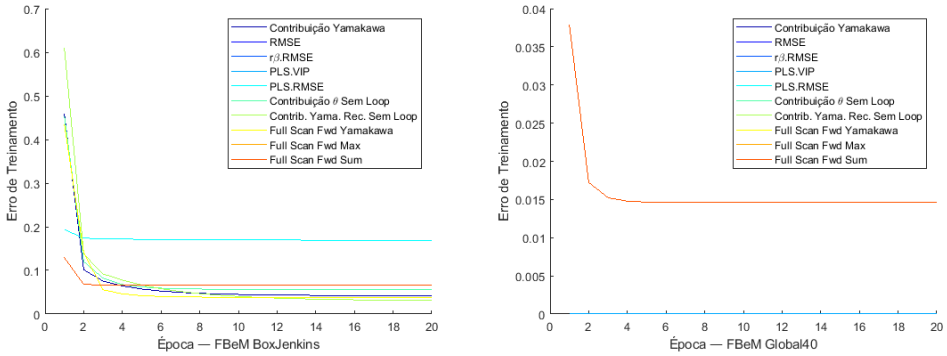

Figura 27: (1) BoxJenkins (2) Global40PxLast

Os fontes estão no Google Drive.

9. Referências

[JANG] Neuro-Fuzzy and Soft Computing, A Computational Approach to Learning and Machine Intelligence.

[9] Seleção de variáveis stepwise aplicadas em redes neurais artificiais para previsão de demanda de cargas elétricas. ALVES, M.; LOTUFO, A.; LOPES, M.

[10] Seleção das variáveis de processo mais relevantes para predição dos níveis de sucata em um processo do setor metal-mecânico. STEIN, M.; ANZANELLO, M.; CERVO, V.; KAHMANN, A.

[11] Um novo método para seleção de variáveis preditivas com base em índices de importância. ZIMMER, J.; ANZANELLO, M.

[PLS] CHONG, I.-G.; JUN, C.-H. Performance of some variable selection methods when multicollinearity is present. Chemometrics Intelligent Laboratory Systems, v. 78, p. 103-112, 2005.

[20] GAUCHI, J. P.; CHAGNON, P. Comparison of selection methods of explanatory variables in PLS regression with application to manufacturing process data. Chemometrics Intelligent Laboratory Systems, v. 58, p. 171-193, 2001.

[SPECTRAL] Zheng Alan Zhao, Huan Liu - Spectral Feature Selection for Data Mining - CRC Press, 2012.

Versão de 19/abril/2023* por Vinicius Claudino Ferraz.

*Fora da caridade não há salvação.