

1) Carregue o arquivo supermercado.csv no Weka:

- a) Quantas regras existem com suporte igual a 10% e confiança igual a 90% (default)?
Zero (nenhuma regra).
- b) Altere o suporte do item “a” para 3%. Quais foram as melhores regras de associação encontradas?

1. Laranja=S, Maca=S, Alho=S ==> Limao=S

2. Limao=S, Tomate=S, Alho=S ==> Kiwi=S

3. Kiwi=S, Cebola=S, Pepino=S, Alho=S ==> Maca=S

- c) Altere a confiança do item “a” para 50%. Quais foram as melhores regras de associação encontradas?

1. Laranja=S ==> Banana=S

2. Laranja=S, Limao=S ==> Banana=S

3. Maca=S, Limao=S ==> Banana=S

4. Banana=S, Maca=S ==> Limao=S

5. Limao=S, Alface=S ==> Banana=S

6. Limao=S ==> Banana=S

7. Banana=S, Limao=S ==> Alface=S

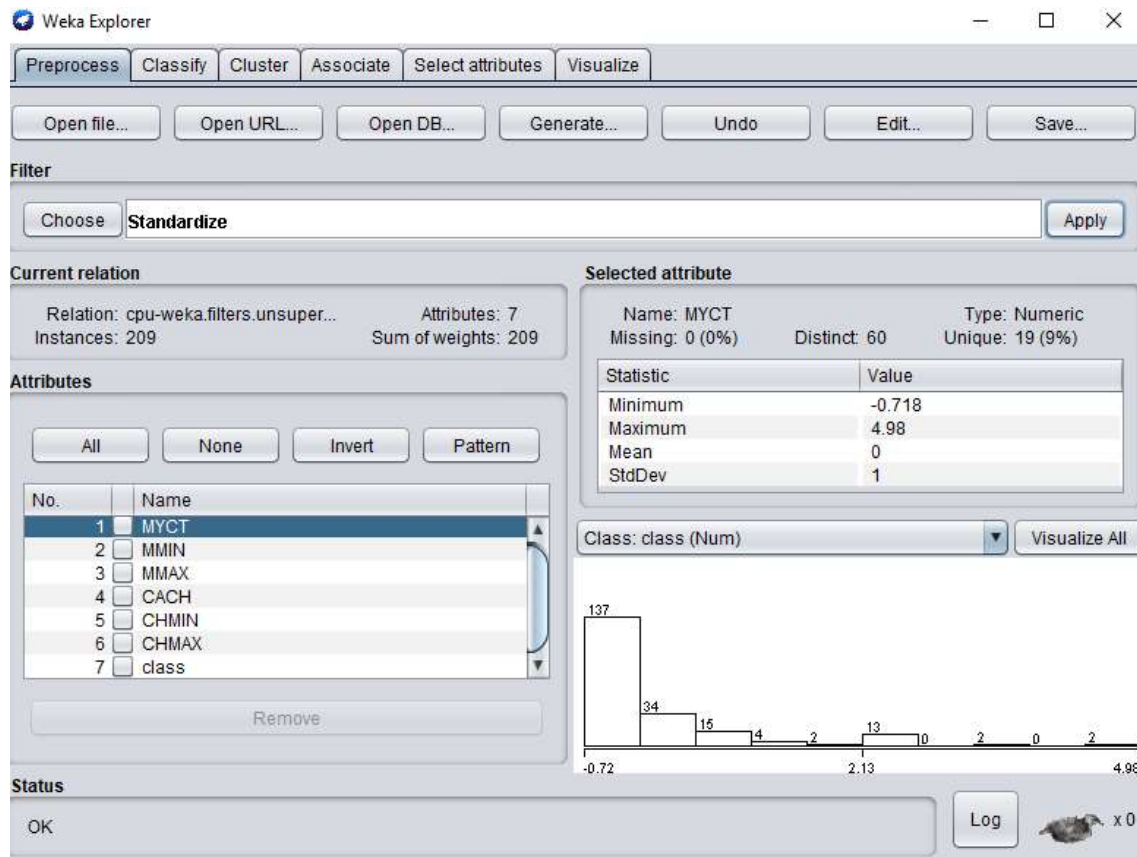
8. Tomate=S ==> Alface=S

9. Alface=S ==> Banana=S

10. Alho=S ==> Kiwi=S

2) Carregue o arquivo cpu.arff no Weka:

a) Normalize os atributos usando Z-score (observe que não é usando Min-max);



b) Explore o algoritmo k-means (SimpleKMeans) e calcule os clusters pré-definidos pelo algoritmo.

Clustered Instances

#0 171 (82%)

#1 38 (18%)

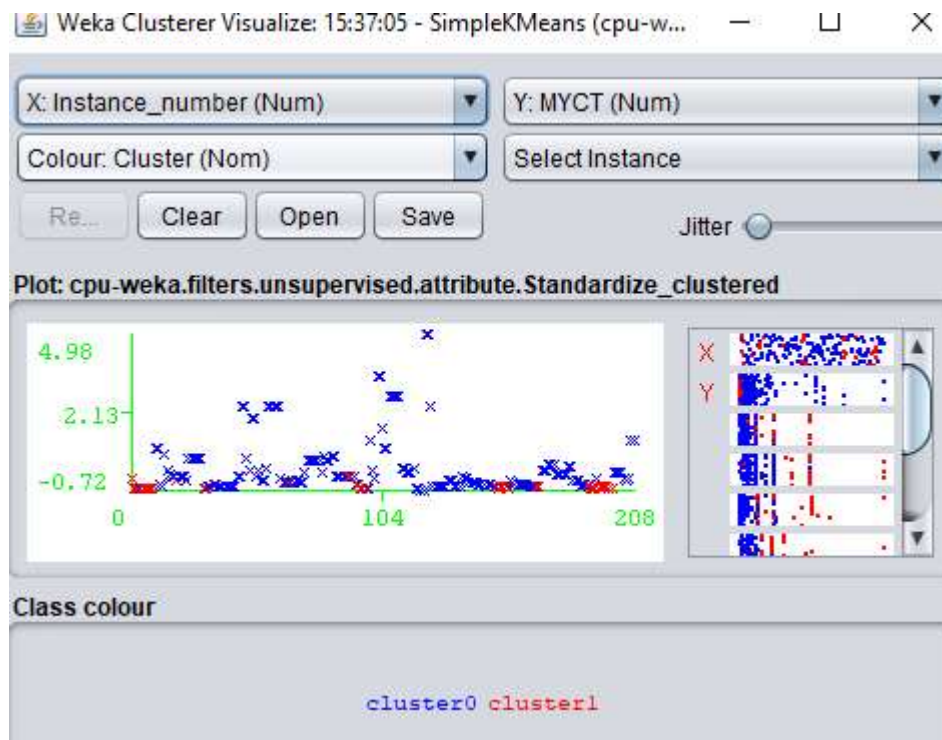
(Captura de tela com o modelo disponível na próxima página)

c) Mude a semente (seed) para o k-means e observe o comportamento do algoritmo. O que ocorreu? Apresente telas de associação dos clusters para demonstrar o que ocorreu.

Clustered Instances

#0 182 (87%)

#1 27 (13%)



=== Clustering model (full training set) ===

kMeans
 =====

Number of iterations: 12
 Within cluster sum of squared errors: 21.179615011158205

Initial starting points (random):

Cluster 0: 1.522218,-0.541408,-0.835381,-0.620392,-0.542608,-0.66422,16
 Cluster 1: -0.556449,0.291852,0.017383,0.167228,0.190931,-0.2411,113

Missing values globally replaced with mean/mode

Final cluster centroids:

Attribute	Full Data (209.0)	Cluster#	
		0 (171.0)	1 (38.0)
MYCT	0	0.1348	-0.6066
MMIN	0	-0.2957	1.3307
MMA	0	-0.3456	1.5551
CACH	0	-0.3311	1.4899
CHMIN	-0	-0.2921	1.3144
CHMAX	-0	-0.2272	1.0222
class	105.622	50.2632	354.7368

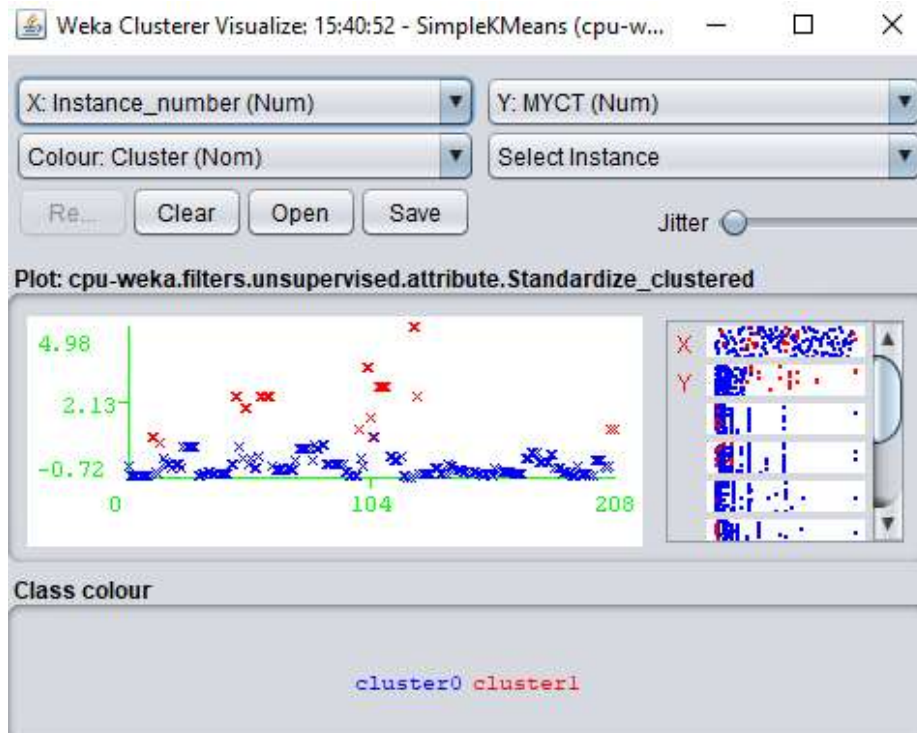
Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

0 171 (82%)
 1 38 (18%)

Seed original (10)



=== Clustering model (full training set) ===

kMeans
=====

Number of iterations: 8
Within cluster sum of squared errors: 27.746671928629375

Initial starting points (random):

Cluster 0: 0.369538,-0.541408,-0.62219,-0.620392,-0.542608,0.220486,45
Cluster 1: 4.98026,-0.541408,-0.835381,-0.620392,-0.689316,-0.702686,18

Missing values globally replaced with mean/mode

Final cluster centroids:

Attribute	Full Data (209.0)	Cluster#	
		0 (182.0)	1 (27.0)
MYCT	0	-0.3295	2.2209
MMIN	0	0.0835	-0.5629
MMA	0	0.1009	-0.6803
CACH	0	0.0845	-0.5693
CHMIN	-0	0.0829	-0.5589
CHMAX	-0	0.0926	-0.6243
class	105.622	118.033	21.963

Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

0 182 (87%)
1 27 (13%)

Seed alterada (56)

Na segunda execução, o resultado foi diferente pois os clusters foram inicializados de forma diferente:

“Seed is just a random numbers seed. Once seed is fixed, even a randomized algorithm behaves deterministically. KMeans is not deterministic, so if you want repeatable results - you fix a seed. However there is completely no relation between exact value of the seed and the results of KMeans clustering. ... the *only* purpose of seeding is to make sure that you get the exact same result when you run this code many times on the exact same data. KMeans is randomized, and so could lead to many different results if you simply run it many times - this way you can "force it" to be repeatable”

(<https://stackoverflow.com/questions/33973817/what-is-the-seed-in-wekas-simplekmeans-clusterer>)

- d) Quantos dos atributos gerados através da Análise de Componentes Principais (PCA) devem ser utilizados considerando uma cobertura de 90% da variância dos dados? Qual a composição destes em relação aos atributos originais?

The screenshot shows the Weka Attribute Evaluator window. The 'Choose' button is set to 'PrincipalComponents -R 0.9 -A 50'. The 'Search Method' is set to 'Ranker -T -1.7976931348623157E308 -N -1'. The 'Attribute Selection Mode' is set to 'Use full training set' with 'Folds' set to 10 and 'Seed' set to 1. The '(Num) class' dropdown is set to '(Num) class'. The 'Start' button is highlighted. The 'Attribute selection output' pane displays the following text:

```
=== Attribute Selection on all input data ===  
  
Search Method:  
Attribute ranking.  
  
Attribute Evaluator (unsupervised):  
Principal Components Attribute Transformer  
  
Correlation matrix  
1      -0.34 -0.38 -0.32 -0.3  -0.25  
-0.34   1      0.76 0.53 0.52 0.27  
-0.38   0.76   1      0.54 0.56 0.53  
-0.32   0.53   0.54   1      0.58 0.49  
-0.3    0.52   0.56   0.58   1      0.55  
-0.25   0.27   0.53   0.49   0.55   1
```

The 'Result list (right-click for options)' pane shows a single result: '21:22:34 - Ranker + PrincipalComponent'.

eigenvalue	proportion	cumulative	
3.35674	0.55946	0.55946	-0.469MMAX-0.435CHMIN-0.429CACH-0.427MMIN-0.374CHMAX+0.29 MYCT
0.82936	0.13823	0.69768	0.682MYCT+0.559CHMAX-0.333MMIN+0.275CHMIN+0.152CACH-0.114MMAX
0.73923	0.1232	0.82089	0.669MYCT+0.548MMIN-0.426CHMAX+0.264MMAX-0.03CHMIN+0.02 CACH
0.49632	0.08272	0.90361	0.714CACH-0.477MMAX-0.436CHMAX+0.255CHMIN-0.088MMIN-0.027MYCT

Eigenvectors

V1	V2	V3	V4	
0.29	0.6822	0.6686	-0.027	MYCT
-0.4274	-0.333	0.5477	-0.0882	MMIN
-0.4691	-0.1141	0.2643	-0.477	MMAX
-0.4286	0.1516	0.0199	0.7137	CACH
-0.4353	0.2746	-0.0302	0.2546	CHMIN
-0.3742	0.5588	-0.4264	-0.4358	CHMAX

Ranked attributes:

0.4405	1	-0.469MMAX-0.435CHMIN-0.429CACH-0.427MMIN-0.374CHMAX+0.29 MYCT
0.3023	2	0.682MYCT+0.559CHMAX-0.333MMIN+0.275CHMIN+0.152CACH-0.114MMAX
0.1791	3	0.669MYCT+0.548MMIN-0.426CHMAX+0.264MMAX-0.03CHMIN+0.02 CACH
0.0964	4	0.714CACH-0.477MMAX-0.436CHMAX+0.255CHMIN-0.088MMIN-0.027MYCT

Selected attributes: 1,2,3,4 : 4

Output da execução do PCA considerando uma cobertura de 90% da variância dos dados

Como percebemos pela imagem acima, devem ser utilizados 4 atributos. A composição dos 4 atributos em relação aos atributos originais é, como mostra a imagem:

Atributo 1:

-0.469*MMAX-0.435*CHMIN-0.429*CACH-0.427*MMIN-0.374*CHMAX+0.29*MYCT

Atributo 2:

0.682*MYCT+0.559*CHMAX-0.333*MMIN+0.275C*HMIN+0.152*CACH-0.114*MMAX

Atributo 3

0.669*MYCT+0.548*MMIN-0.426*CHMAX+0.264*MMAX-0.03*CHMIN+0.02*CACH

Atributo 4

0.714*CACH-0.477*MMAX-0.436*CHMAX+0.255*CHMIN-0.088*MMIN-0.027*MYCT

- 3) Carregue o arquivo iris-train.arff no Weka:
a) Normalize os atributos usando Min-max;

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Open file... Open URL... Open DB... Generate... Undo Edit... Save...

Filter

Choose **Normalize -S 1.0 -T 0.0** Apply

Current relation

Relation: iris-weka.filters.unsuperv... Attributes: 5
Instances: 120 Sum of weights: 120

Attributes

All None Invert Pattern

No.	Name
1	<input checked="" type="checkbox"/> sepalength
2	<input type="checkbox"/> sepalwidth
3	<input type="checkbox"/> petallength
4	<input type="checkbox"/> petalwidth
5	<input type="checkbox"/> class

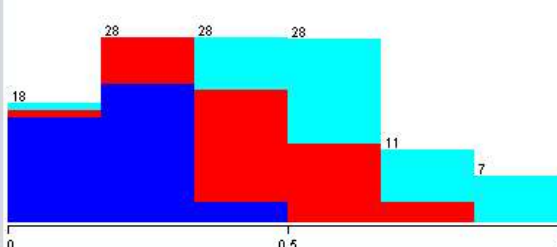
Remove

Selected attribute

Name: sepalength
Missing: 0 (0%) Distinct: 33 Type: Numeric
Unique: 7 (6%)

Statistic	Value
Minimum	0
Maximum	1
Mean	0.442
StdDev	0.235

Class: class (Nom) Visualize All

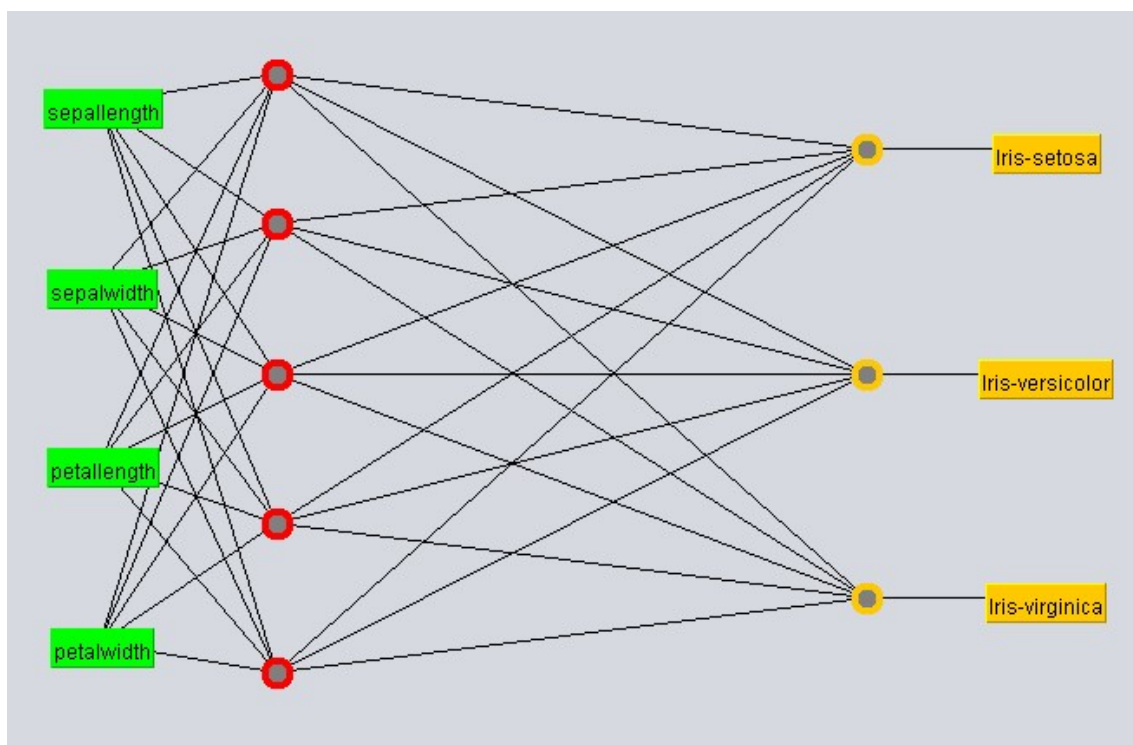


0 0.5 1

Status

OK Log x 0

- b) Treine uma RNA para classificação utilizando os dados no item a). A rede deverá ter 5 neurônios na camada escondida. Entregue como resultado a imagem da rede obtida no Weka e a matriz de confusão resultante. Treine utilizando a opção Cross-validation com valor padrão.



Time taken to build model: 0.11 seconds

=== Stratified cross-validation ===
 === Summary ===

Correctly Classified Instances	117	97.5	%
Incorrectly Classified Instances	3	2.5	%
Kappa statistic	0.9625		
Mean absolute error	0.0339		
Root mean squared error	0.1259		
Relative absolute error	7.6169	%	
Root relative squared error	26.714	%	
Total Number of Instances	120		

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1,000	0,000	1,000	1,000	1,000	1,000	1,000	1,000	Iris-setosa
	0,950	0,013	0,974	0,950	0,962	0,944	0,996	0,993	Iris-versicolor
	0,975	0,025	0,951	0,975	0,963	0,944	0,996	0,992	Iris-virginica
Weighted Avg.	0,975	0,013	0,975	0,975	0,975	0,963	0,997	0,995	

=== Confusion Matrix ===

a	b	c	<-- classified as
40	0	0	a = Iris-setosa
0	38	2	b = Iris-versicolor
0	1	39	c = Iris-virginica



- c) Submeta o arquivo iris-test.arff à rede treinada no item b) e avalie o resultado obtido na classificação apresentando a matriz de confusão.

```

=== Confusion Matrix ===

  a  b  c  <-- classified as
40  0  0 |  a = Iris-setosa
 0 38  2 |  b = Iris-versicolor
 0  1 39 |  c = Iris-virginica

=== Re-evaluation on test set ===

User supplied test set
Relation:      iris
Instances:     unknown (yet). Reading incrementally
Attributes:    5

=== Summary ===

Correctly Classified Instances      15          50      %
Incorrectly Classified Instances    15          50      %
Kappa statistic                     0.25
Mean absolute error                 0.3344
Root mean squared error             0.5369
Total Number of Instances          30


=== Detailed Accuracy By Class ===

                TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                0,500    0,000    1,000     0,500    0,667      0,632    1,000    1,000    Iris-setosa
                0,000    0,250    0,000     0,000    0,000      -0,316    0,000    0,206    Iris-versicolor
                1,000    0,500    0,500     1,000    0,667      0,500    0,500    0,331    Iris-virginica
Weighted Avg.   0,500    0,250    0,500     0,500    0,444      0,272    0,500    0,512

=== Confusion Matrix ===

  a  b  c  <-- classified as
 5  5  0 |  a = Iris-setosa
 0  0 10 |  b = Iris-versicolor
 0  0 10 |  c = Iris-virginica

```



- d) Reduza o número de épocas para 10 e faça um novo treinamento da rede. O que ocorreu?

O modelo resultante foi pior que o modelo original com 500 épocas. A rede treinada com cross validation passou a acertar apenas 83.33% (contra 97.5% de antes) e a aplicação dos dados do arquivo de teste resultou em um acerto de apenas 33.33% (contra 50% de antes). A matriz de confusão reflete essa piora no modelo, quando a compararmos com a do modelo antigo:

a	b	c	<-- classified as	a	b	c	<-- classified as
5	5	0	a = Iris-setosa	0	1	9	a = Iris-setosa
0	0	10	b = Iris-versicolor	0	0	10	b = Iris-versicolor
0	0	10	c = Iris-virginica	0	0	10	c = Iris-virginica

Modelo antigo

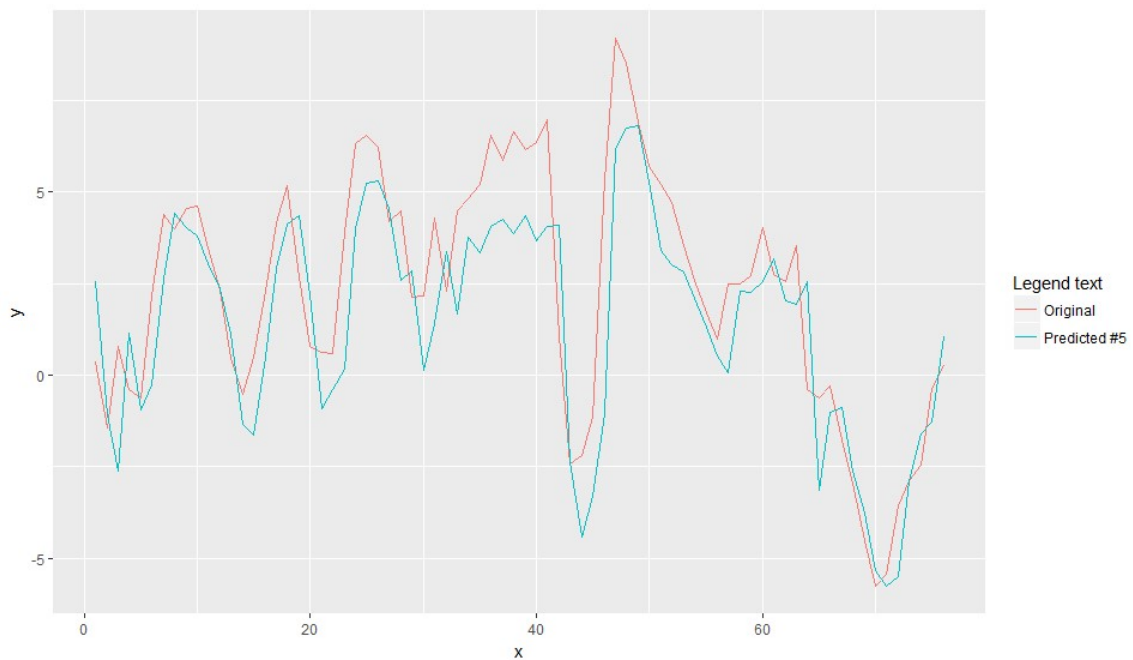
Modelo novo

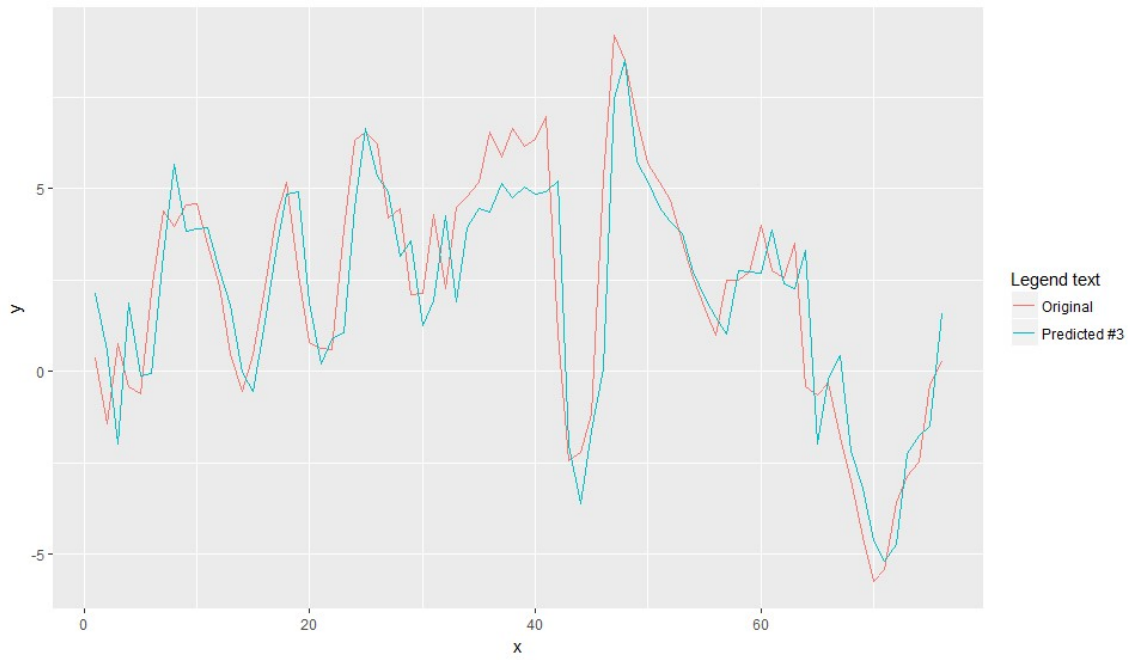
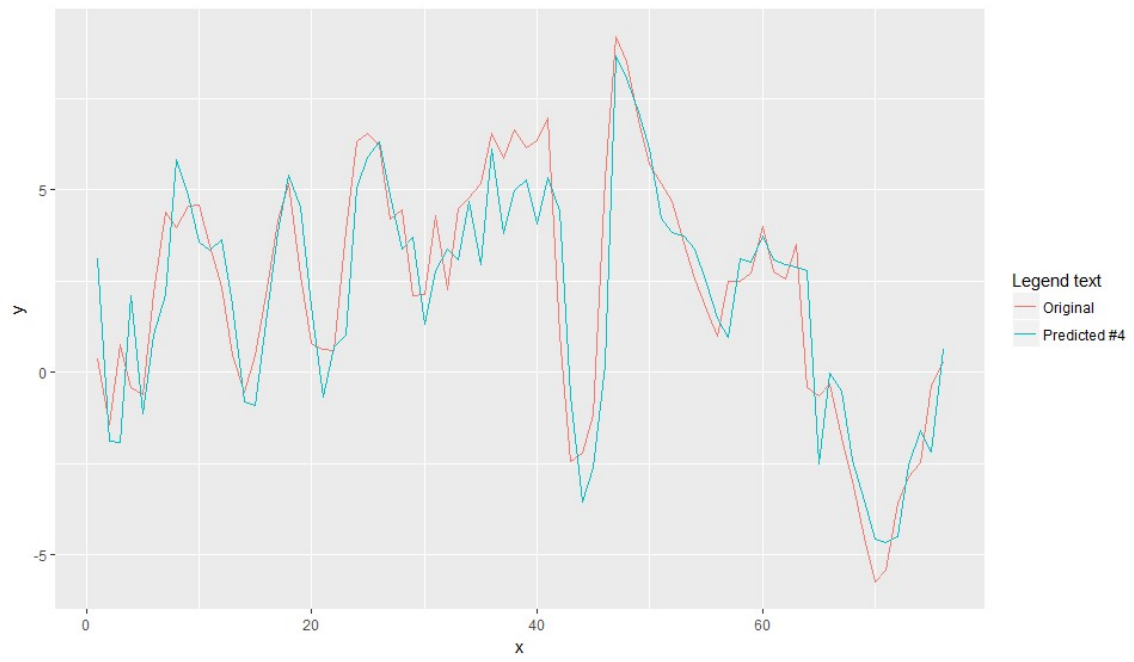
4) Utilizando o conjunto de dados PIBBrasil.csv e o pacote “RSNNS” do R, monte uma rede neural de Elman para prever o valor do PIB um passo à frente utilizando uma janela deslizante de tamanho n . O valor de n deverá ser avaliado entre os valores de 3 a 6. Varie também o número de neurônios na camada escondida. Utilize os 4 últimos valores do PIB para o conjunto de teste e os demais para o conjunto de treinamento.

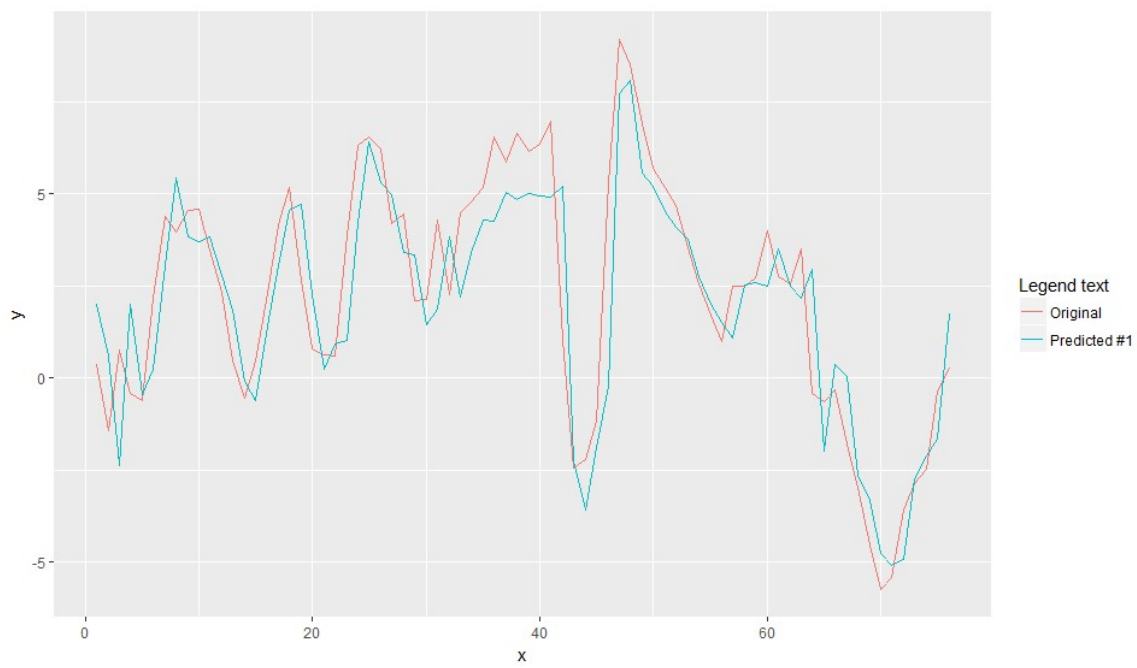
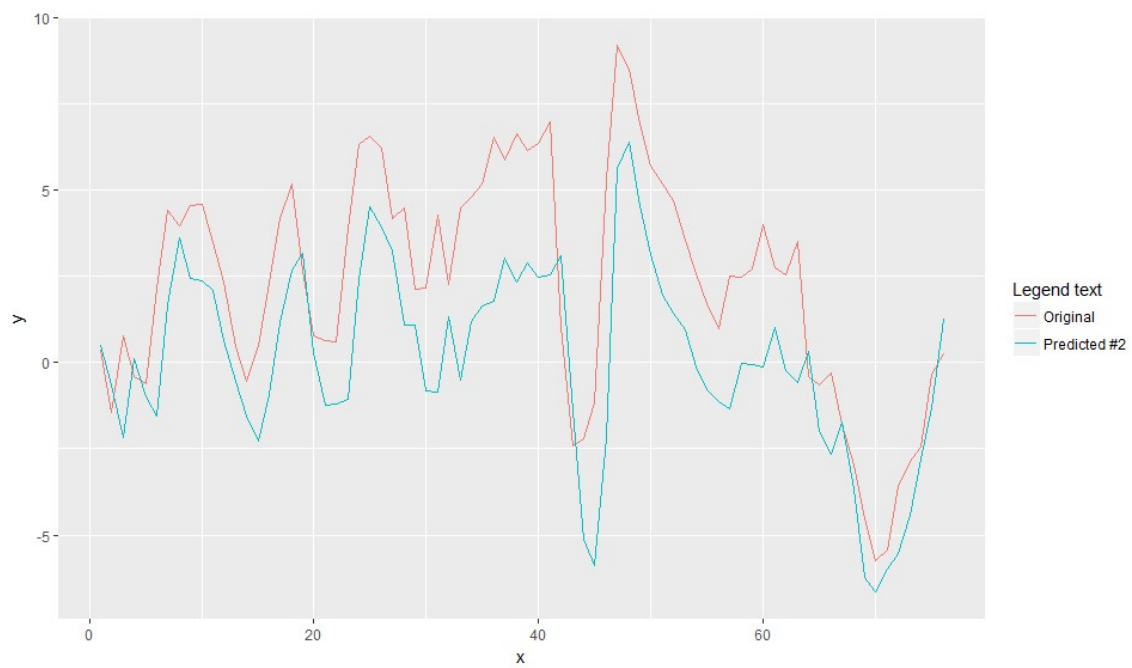
a) Como resultado, preencha a tabela abaixo com as cinco configurações (n versus número de neurônios na camada escondida) com menor valor de RMSE para o conjunto de teste.

Janela	Número de Neurônios	RMSE de Treinamento	RMSE de Teste
3	6	1.556645	1.040738
3	5	1.500314	1.052697
3	4	1.543523	1.091918
4	4	1.756668	1.093390
3	13	1.574219	1.096785

b) Plote um gráfico com a série original do PIB e com o resultado da previsão para as cinco configurações escolhidas.







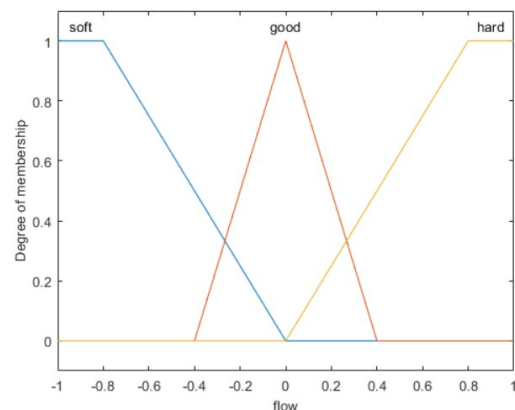
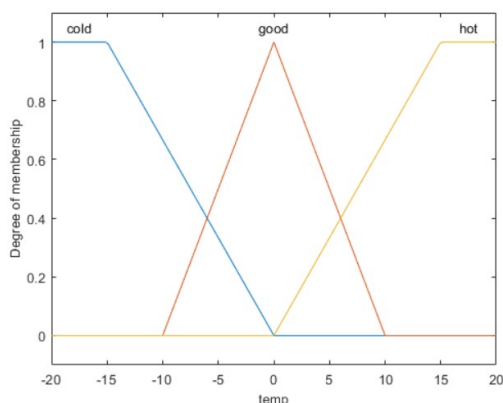
- 5) Você vai passar um mês em uma região selvagem e estará levando uma mochila, no entanto, o peso máximo que pode carregar é de 20kg. Você tem uma série de itens de sobrevivência disponíveis, cada um com seu próprio número de "pontos de sobrevivência", conforme tabela abaixo. Seu objetivo é maximizar o número de pontos de sobrevivência. Resolva o problema apresentando a lista de itens a serem selecionados usando Algoritmo Genético (pacote "genalg") no ambiente R.

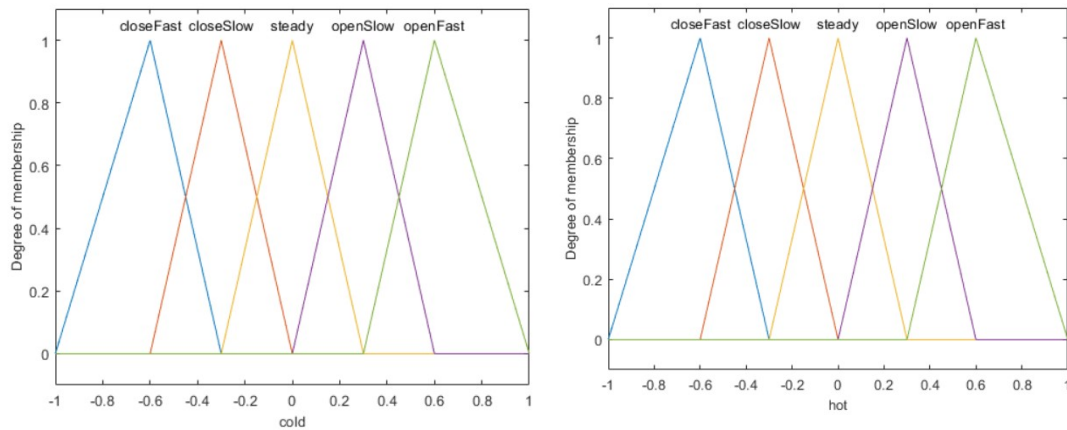
Item	Ponto de sobrevivência	Peso
A	10	1
B	20	5
C	15	10
D	2	1
E	30	7
F	10	5
G	30	1

```
> print(paste( melhor solucao encontrada: , paste(labeis,collapse = + ), score = , best %%% points, peso = ,
est %%% weights))
[1] "Melhor solucao encontrada: B+E+F+G score = 90 peso = 18"
>
```

Melhor solução encontrada: B+E+F+G (score = 90; peso = 18)

- 6) Implementar um SIF no Matlab usando o Fuzzy Logic Toolbox para controlar um chuveiro. As variáveis de entrada são a temperatura da água e o fluxo de água. As variáveis de saída são duas, a taxa na qual as válvulas de água fria e de água quente devem ser abertas ou fechadas. Os conjuntos fuzzy de entrada e saída bem como as regras estão apresentadas abaixo. Após implementado, avalie as saídas das válvulas para os pares de entrada [Temperatura Fluxo] = [-12 -0.6; -5 0.2; 0 0; 5 0.2; 12 0.6]. Deverá ser entregue o arquivo "fis" do Matlab com o nome chuveiro.fis e o resultado das simulações.





If (temp is cold) and (flow is soft) then (cold is openSlow)(hot is openFast) (1)
 If (temp is cold) and (flow is good) then (cold is closeSlow)(hot is openSlow) (1)
 If (temp is cold) and (flow is hard) then (cold is closeFast)(hot is closeSlow) (1)
 If (temp is good) and (flow is soft) then (cold is openSlow)(hot is openSlow) (1)
 If (temp is good) and (flow is good) then (cold is steady)(hot is steady) (1)
 If (temp is good) and (flow is hard) then (cold is closeSlow)(hot is closeSlow) (1)
 If (temp is hot) and (flow is soft) then (cold is openFast)(hot is openSlow) (1)
 If (temp is hot) and (flow is good) then (cold is openSlow)(hot is closeSlow) (1)
 If (temp is hot) and (flow is hard) then (cold is closeSlow)(hot is closeFast) (1)

Resultado:

Temp	Flow	Cold	Hot
-12	-0.6	0.3000	0.6350
-5	0.2	0.2689	0.0228
0	0	0.0000	0.0000
5	0.2	0.0228	0.2689
12	0.6	0.3000	0.6350