INSTITUTO POLITECNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
NOMBRE: Cerda García Gustavo
Materia: Inteligencia Artificial

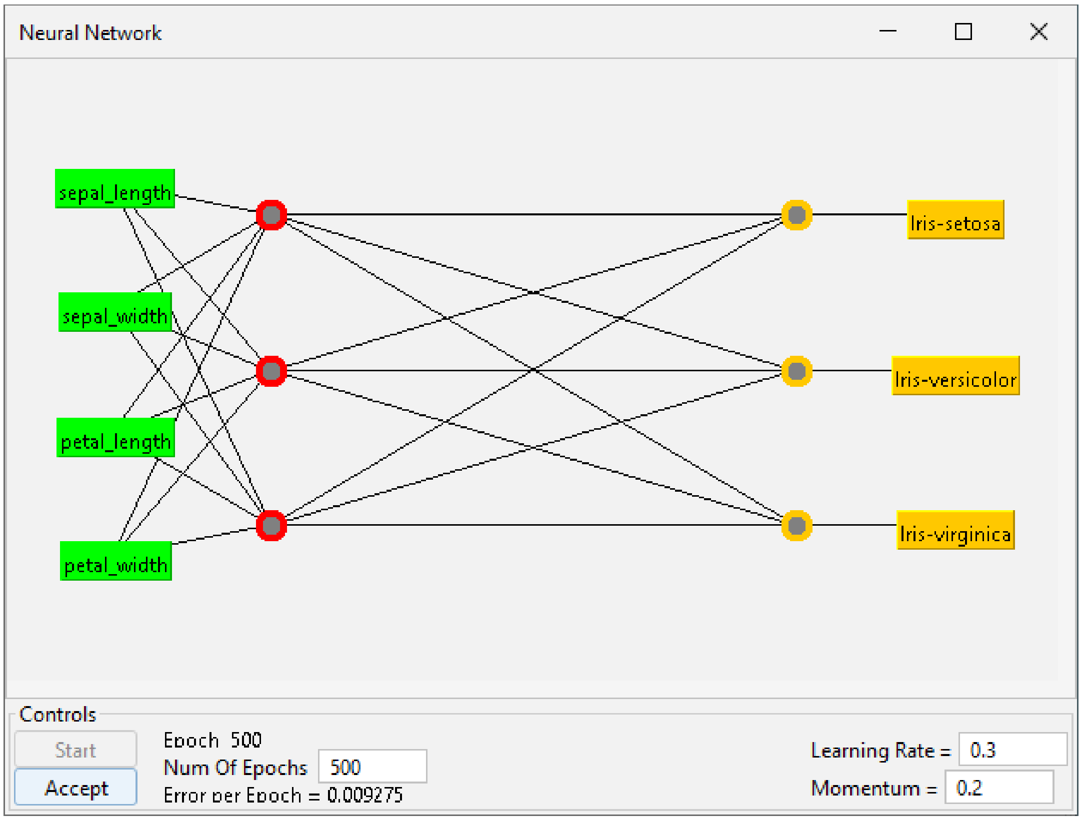
Ejercicio de Laboratorio 14. Perceptrón Multicapa y Red Neuronal RBF

DataSet	Perceptrón Multicapa		Red Neuronal RBF	
	Hold Out 70/30	10-Fold-Cross-Validation	Hold Out 70/30	10-Fold-Cross-Validation
IRIS	97.7778 %	97.3333 %	95.5556 %	95.3333 %
WINE	96.2264 %	97.191 %	60.3774 %	48.8764 %
BREAST CANCER WISCONSIN	97.6608 %	95.6063 %	94.152 %	93.6731 %

Perceptrón Multicapa

Hold-Out 70/30

IRIS



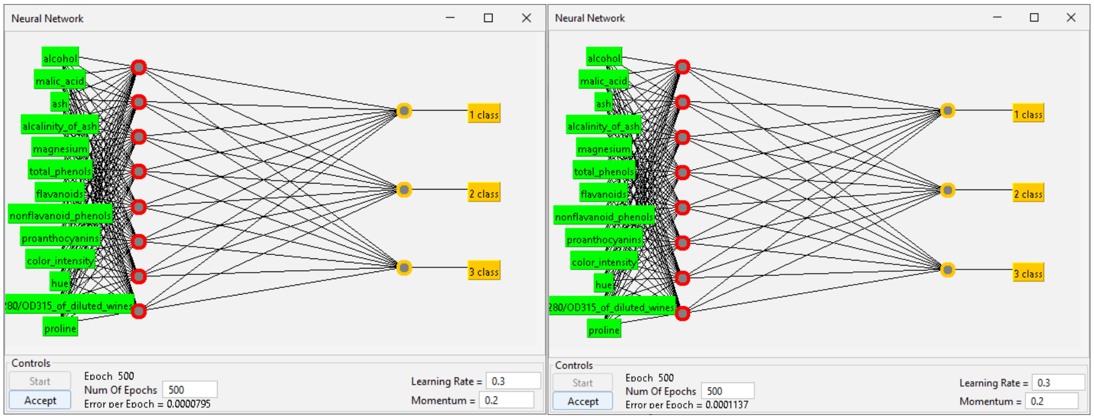
```
=== Classifier model (full training set) ===

Sigmoid Node 0
    Inputs    Weights
    Threshold    -3.5015971588434014
    Node 3    -1.005811085385995
    Node 4    9.07503844669134
    Node 5    -4.107780453339234
Sigmoid Node 1
    Inputs    Weights
    Threshold    1.0692845992273177
    Node 3    3.898873687789407
    Node 4    -9.768910360340266
    Node 5    -8.59913449315135
Sigmoid Node 2
    Inputs    Weights
    Threshold    -1.0071762383436476
    Node 3    -4.218406133827042
    Node 4    -3.626059686321116
    Node 5    8.805122981737854
Sigmoid Node 3
    Inputs    Weights
    Threshold    3.3824855566856726
    Attrib sepal_length    0.9099827458022287
    Attrib sepal_width    1.5675138827531245
    Attrib petal_length    -5.037338107319891
    Attrib petal_width    -4.915469682506093
Sigmoid Node 4
    Inputs    Weights
    Threshold    -3.3305735922918323
    Attrib sepal_length    -1.1116750023770101
    Attrib sepal_width    3.1250096866676538
    Attrib petal_length    -4.133137022912303
    Attrib petal_width    -4.079589727871457
Sigmoid Node 5
    Inputs    Weights
    Threshold    -7.496091023618097
    Attrib sepal_length    -1.2158878822058794
    Attrib sepal_width    -3.5332821317534946
    Attrib petal_length    8.401834252274107
    Attrib petal_width    9.460215580472836
Class Iris-setosa
    Input
    Node 0
Class Iris-versicolor
    Input
    Node 1
```

```
Class Iris-virginica
    Input
    Node 2


Time taken to build model: 0.07 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances          44               97.7778 %
Incorrectly Classified Instances         1                2.2222 %
Kappa statistic                          0.9666
Mean absolute error                      0.024
Root mean squared error                  0.1153
Relative absolute error                  5.3891 %
Root relative squared error             24.4455 %
Total Number of Instances               45
```

Matriz de Confusión

$$\begin{bmatrix} 14 & 0 & 0 \\ 0 & 16 & 0 \\ 0 & 1 & 14 \end{bmatrix}$$

Wine



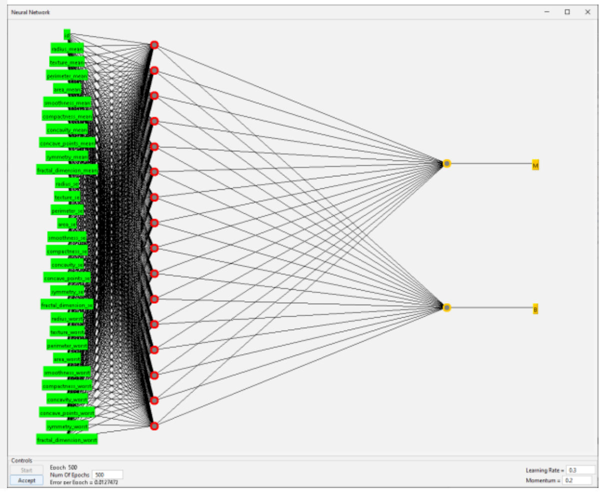

```
Class 1 class
    Input
    Node 0
Class 2 class
    Input
    Node 1
Class 3 class
    Input
    Node 2


Time taken to build model: 32.07 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances          51               96.2264 %
Incorrectly Classified Instances         2                3.7736 %
Kappa statistic                          0.9431
Mean absolute error                      0.0259
Root mean squared error                  0.1328
Relative absolute error                  5.7819 %
Root relative squared error             27.5972 %
Total Number of Instances               53
```

Matriz de Confusión

$$\begin{bmatrix} 19 & 0 & 0 \\ 0 & 14 & 2 \\ 0 & 0 & 18 \end{bmatrix}$$

Breast Cancer Wisconsin

```
Class M
    Input
    Node 0
Class B
    Input
    Node 1


Time taken to build model: 2.19 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances         167               97.6608 %
Incorrectly Classified Instances         4                2.3392 %
Kappa statistic                          0.9504
Mean absolute error                      0.0223
Root mean squared error                  0.1384
Relative absolute error                  4.7583 %
Root relative squared error             28.5122 %
Total Number of Instances              171
```



Matriz de Confusión

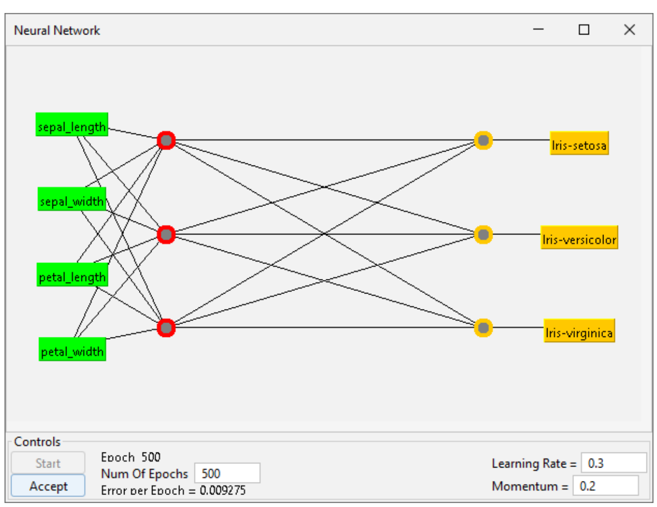
$$\begin{bmatrix} 63 & 2 \\ 2 & 104 \end{bmatrix}$$

10-Fold-Cross-Validation

IRIS

```
=== Classifier model (full training set) ===

Sigmoid Node 0
    Inputs    Weights
    Threshold    -3.5015971588434014
    Node 3    -1.005811085385995
    Node 4    9.07503844669134
    Node 5    -4.107780453339234
Sigmoid Node 1
    Inputs    Weights
    Threshold    1.0692845992273177
    Node 3    3.898873687789407
    Node 4    -9.768910360340266
    Node 5    -8.59913449315135
Sigmoid Node 2
    Inputs    Weights
    Threshold    -1.0071762383436476
    Node 3    -4.218406133827042
    Node 4    -3.626059686321116
    Node 5    8.805122981737854
Sigmoid Node 3
    Inputs    Weights
    Threshold    3.3824855566856726
    Attrib sepal_length    0.9099827458022287
    Attrib sepal_width    1.5675138827531245
    Attrib petal_length    -5.037338107319891
    Attrib petal_width    -4.915469682506093
Sigmoid Node 4
    Inputs    Weights
    Threshold    -3.3305735922918323
    Attrib sepal_length    -1.1116750023770101
    Attrib sepal_width    3.1250096866676538
    Attrib petal_length    -4.133137022912303
    Attrib petal_width    -4.079589727871457
Sigmoid Node 5
    Inputs    Weights
    Threshold    -7.496091023618097
    Attrib sepal_length    -1.2158878822058794
    Attrib sepal_width    -3.5332821317534946
    Attrib petal_length    8.401834252274107
    Attrib petal_width    9.460215580472836
```

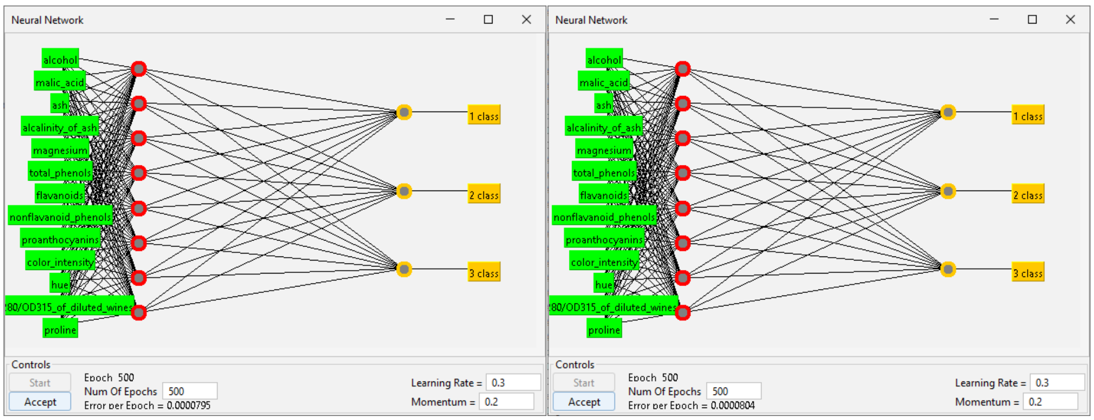
```
Class Iris-setosa
    Input
    Node 0
Class Iris-versicolor
    Input
    Node 1
Class Iris-virginica
    Input
    Node 2


Time taken to build model: 1.69 seconds

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

Correctly Classified Instances         146               97.3333 %
Incorrectly Classified Instances         4                2.6667 %
Kappa statistic                          0.96
Mean absolute error                      0.0327
Root mean squared error                  0.1291
Relative absolute error                  7.3555 %
Root relative squared error             27.3796 %
Total Number of Instances              150
```



Matriz de Confusión

$$\begin{bmatrix} 50 & 0 & 0 \\ 0 & 48 & 2 \\ 0 & 2 & 48 \end{bmatrix}$$

Wine

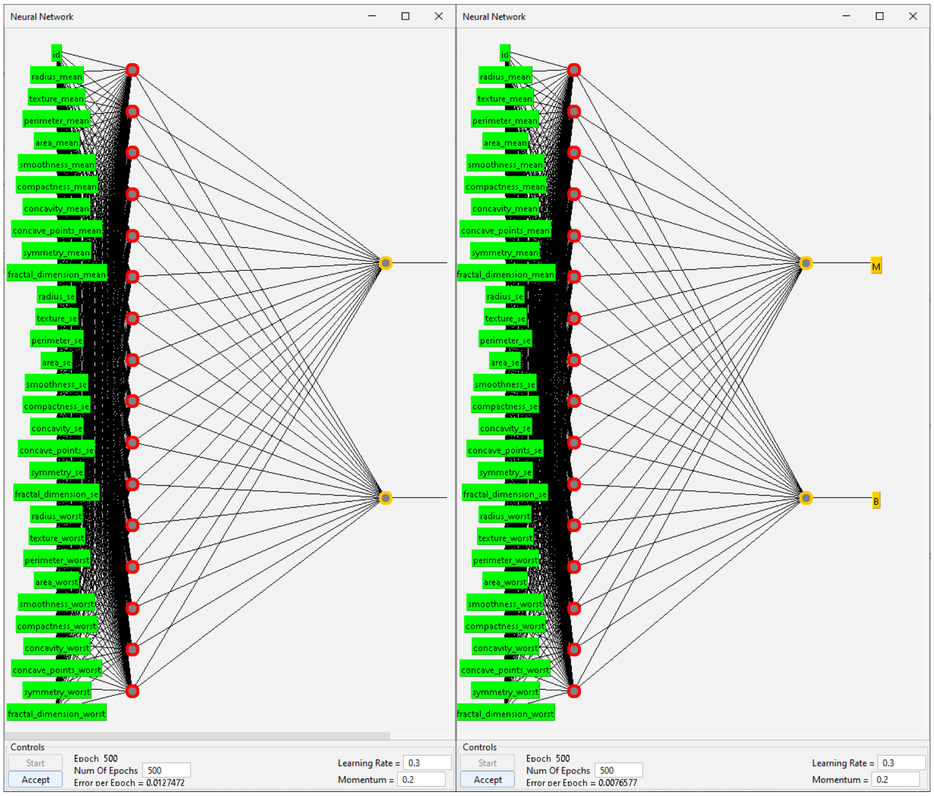



```
Sigmoid Node 11
    Inputs    Weights
    Threshold    0.0456187407994518
Class 1 class
    Input
    Node 0
Class 2 class
    Input
    Node 1
Class 3 class
    Input
    Node 2


Time taken to build model: 11.58 seconds

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

Correctly Classified Instances         173               97.191  %
Incorrectly Classified Instances         5                2.809  %
Kappa statistic                          0.9574
Mean absolute error                      0.0247
Root mean squared error                  0.1172
Relative absolute error                  5.6355 %
Root relative squared error             25.0058 %
Total Number of Instances              178
```

Matriz de Confusión

$$\begin{bmatrix} 59 & 0 & 0 \\ 2 & 67 & 2 \\ 0 & 1 & 47 \end{bmatrix}$$

Breast Cancer Wisconsin




```
Sigmoid Node 18
    Inputs    Weights
    Threshold    0.03691676807580688
Class M
    Input
    Node 0
Class B
    Input
    Node 1


Time taken to build model: 21.56 seconds

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

Correctly Classified Instances         544               95.6063 %
Incorrectly Classified Instances        25                4.3937 %
Kappa statistic                          0.9061
Mean absolute error                      0.042
Root mean squared error                  0.1901
Relative absolute error                  8.9755 %
Root relative squared error             39.318  %
Total Number of Instances              569
```

Matriz de Confusión

$$\begin{bmatrix} 200 & 12 \\ 13 & 344 \end{bmatrix}$$

Red Neuronal RBF

Hold-Out 70/30

IRIS

```
=== Classifier model (full training set) ===

Radial basis function network
(Logistic regression applied to K-means clusters as basis functions):

Logistic Regression with ridge parameter of 1.0E-8
Coefficients...
                               Class
Variable                 Iris-setosa      Iris-versicolor
==========================================================
pCluster_0_0                132.7289             -37.7972
pCluster_0_1                134.5741             -38.3226
pCluster_1_0                -70.3097              27.9147
pCluster_1_1                -63.3603              29.7589
pCluster_2_0                -67.1796               2.9573
pCluster_2_1                -74.4605              19.0193
Intercept                     1.3218             -21.8733


Odds Ratios...
                               Class
Variable                 Iris-setosa      Iris-versicolor
==========================================================
pCluster_0_0    4.399731892019819E57                    0
pCluster_0_1    2.784886924641302E58                    0
pCluster_1_0                       0  1.328036449082467E12
pCluster_1_1                       0  8.397223353462365E12
pCluster_2_0                       0              19.2459
pCluster_2_1                       0        181964100.5946


Time taken to build model: 0 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances          43               95.5556 %
Incorrectly Classified Instances         2                4.4444 %
Kappa statistic                          0.9331
Mean absolute error                      0.0302
Root mean squared error                  0.1722
Relative absolute error                  6.7897 %
Root relative squared error             36.4949 %
Total Number of Instances               45
```

Matriz de Confusión

$$\begin{bmatrix} 13 & 1 & 0 \\ 0 & 16 & 0 \\ 0 & 2 & 13 \end{bmatrix}$$

Wine

```
=== Classifier model (full training set) ===

Radial basis function network
(Logistic regression applied to K-means clusters as basis functions):

Logistic Regression with ridge parameter of 1.0E-8
Coefficients...
                  Class
Variable        1 class  2 class
================================
pCluster_0_0     0.1699    -1.45
pCluster_0_1    -0.1699     1.45
Intercept        0.2066  -0.4558


Odds Ratios...
                  Class
Variable        1 class  2 class
================================
pCluster_0_0     1.1852   0.2346
pCluster_0_1     0.8437   4.2629


Time taken to build model: 0.01 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances          32               60.3774 %
Incorrectly Classified Instances        21               39.6226 %
Kappa statistic                          0.4073
Mean absolute error                      0.3682
Root mean squared error                  0.4264
Relative absolute error                 82.3329 %
Root relative squared error             88.634  %
Total Number of Instances               53
```

Matriz de Confusión

$$\begin{bmatrix} 6 & 7 & 6 \\ 5 & 9 & 2 \\ 0 & 1 & 17 \end{bmatrix}$$

Breast Cancer Wisconsin

```
=== Classifier model (full training set) ===

Radial basis function network
(Logistic regression applied to K-means clusters as basis functions

Logistic Regression with ridge parameter of 1.0E-8
Coefficients...
                    Class
Variable                M
========================
pCluster_0_0       3.4492
pCluster_0_1       5.1992
pCluster_1_0      -4.1135
pCluster_1_1      -1.1999
Intercept         -0.9243


Odds Ratios...
                    Class
Variable                M
========================
pCluster_0_0      31.4762
pCluster_0_1     181.1227
pCluster_1_0       0.0164
pCluster_1_1       0.3012


Time taken to build model: 0.02 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances         161               94.152  %
Incorrectly Classified Instances        10                5.848  %
Kappa statistic                          0.8759
Mean absolute error                      0.1004
Root mean squared error                  0.2317
Relative absolute error                 21.4114 %
Root relative squared error             47.7196 %
Total Number of Instances              171
```

Matriz de Confusión

$$\begin{bmatrix} 60 & 5 \\ 5 & 101 \end{bmatrix}$$

10-Fold-Cross-Validation

IRIS

```
=== Classifier model (full training set) ===

Radial basis function network
(Logistic regression applied to K-means clusters as basis functions):

Logistic Regression with ridge parameter of 1.0E-8
Coefficients...
                               Class
Variable                 Iris-setosa      Iris-versicolor
=========================================================
pCluster_0_0                132.7289             -37.7972
pCluster_0_1                134.5741             -38.3226
pCluster_1_0                -70.3097              27.9147
pCluster_1_1                -63.3603              29.7589
pCluster_2_0                -67.1796               2.9573
pCluster_2_1                -74.4605              19.0193
Intercept                     1.3218             -21.8733


Odds Ratios...
                               Class
Variable                 Iris-setosa      Iris-versicolor
=========================================================
pCluster_0_0    4.399731892019819E57                    0
pCluster_0_1    2.784886924641302E58                    0
pCluster_1_0                       0  1.328036449082467E12
pCluster_1_1                       0  8.397223353462365E12
pCluster_2_0                       0              19.2459
pCluster_2_1                       0        181964100.5946


Time taken to build model: 0 seconds

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

Correctly Classified Instances         143               95.3333 %
Incorrectly Classified Instances         7                4.6667 %
Kappa statistic                          0.93
Mean absolute error                      0.034
Root mean squared error                  0.1585
Relative absolute error                  7.6407 %
Root relative squared error             33.6209 %
Total Number of Instances              150
```

Matriz de Confusión

$$\begin{bmatrix} 49 & 1 & 0 \\ 0 & 47 & 3 \\ 0 & 3 & 47 \end{bmatrix}$$

Wine

```
=== Classifier model (full training set) ===

Radial basis function network
(Logistic regression applied to K-means clusters as basis functions):

Logistic Regression with ridge parameter of 1.0E-8
Coefficients...
                 Class
Variable        1 class  2 class
================================
pCluster_0_0     0.1699    -1.45
pCluster_0_1    -0.1699     1.45
Intercept        0.2066  -0.4558


Odds Ratios...
                 Class
Variable        1 class  2 class
================================
pCluster_0_0     1.1852   0.2346
pCluster_0_1     0.8437   4.2629


Time taken to build model: 0.01 seconds

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

Correctly Classified Instances          87               48.8764 %
Incorrectly Classified Instances        91               51.1236 %
Kappa statistic                          0.1856
Mean absolute error                      0.3997
Root mean squared error                  0.448
Relative absolute error                 91.0416 %
Root relative squared error             95.6225 %
Total Number of Instances              178
```

Matriz de Confusión

$$\begin{bmatrix} 18 & 34 & 7 \\ 4 & 61 & 6 \\ 8 & 32 & 8 \end{bmatrix}$$

Breast Cancer Wisconsin

```
=== Classifier model (full training set) ===

Radial basis function network
(Logistic regression applied to K-means clusters as basis functions

Logistic Regression with ridge parameter of 1.0E-8
Coefficients...
                       Class
Variable                   M
============================
pCluster_0_0          3.4492
pCluster_0_1          5.1992
pCluster_1_0         -4.1135
pCluster_1_1         -1.1999
Intercept            -0.9243


Odds Ratios...
                       Class
Variable                   M
============================
pCluster_0_0         31.4762
pCluster_0_1        181.1227
pCluster_1_0          0.0164
pCluster_1_1          0.3012


Time taken to build model: 0.01 seconds

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

Correctly Classified Instances         533               93.6731 %
Incorrectly Classified Instances        36                6.3269 %
Kappa statistic                          0.8639
Mean absolute error                      0.0948
Root mean squared error                  0.2338
Relative absolute error                 20.2639 %
Root relative squared error             48.3492 %
Total Number of Instances              569
```

Matriz de Confusión

$$\begin{bmatrix} 191 & 21 \\ 15 & 342 \end{bmatrix}$$

Documentación del Notebook

Primero debemos importar las librerías que vamos a utilizar

```python
import tensorflow as tf #Importar la libreria de tensorflow

from tensorflow.keras import datasets, layers, models #Importar las librerias de keras para datasets, layers y models
import matplotlib.pyplot as plt #Importar la libreria de matplotlib para graficar
```

Seguimos descargando los datos de CIFAR10 y normalizamos los valores de los pixeles de las imágenes.

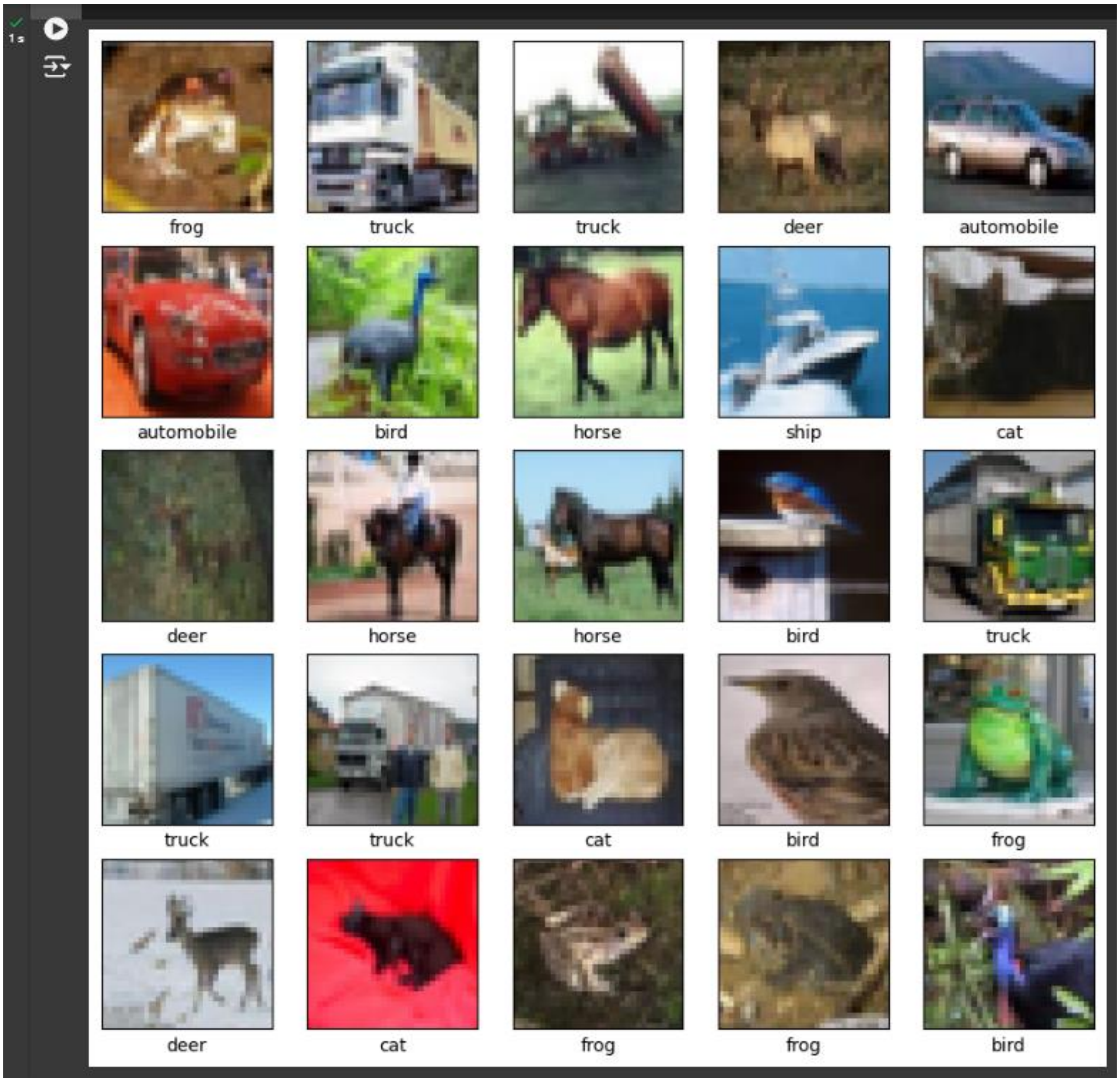
```python
(train_images, train_labels), (test_images, test_labels) = datasets.cifar10.load_data() #Cargar los datos de entrenamiento y prueba de CIFAR10

# Normalizar los valores de los pixeles de las imagenes entre 0 y 1
train_images, test_images = train_images / 255.0, test_images / 255.0
```

```python
(train_images, train_labels), (test_images, test_labels) = datasets.cifar10.load_data()

# Normalize pixel values to be between 0 and 1
train_images, test_images = train_images / 255.0, test_images / 255.0
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170498071/170498071 [==============================] - 18s 0us/step
```

Ahora vamos a mostrar las primeras 25 imágenes de CIFAR10, y definimos las clases y el tamaño de las figuras

```python
class_names = ['airplane', 'automobile', 'bird', 'cat', 'deer',
               'dog', 'frog', 'horse', 'ship', 'truck'] #Definir las clases de CIFAR10

plt.figure(figsize=(10,10)) #Definir el tamaño de la figura
#Graficar las primeras 25 imagenes de CIFAR10
for i in range(25):
    plt.subplot(5,5,i+1) #Definir la posicion de la imagen en la figura
    plt.xticks([]) #Quitar las marcas del eje x
    plt.yticks([]) #Quitar las marcas del eje y
    plt.grid(False) #Quitar la cuadricula
    plt.imshow(train_images[i]) #Mostrar la imagen
    # Definir el titulo de la imagen con la clase a la que pertenece
    plt.xlabel(class_names[train_labels[i][0]])
plt.show() #Mostrar la figura
```



En esta parte se define la arquitectura de la red neuronal convolucional.

```
model = models.Sequential() #Crear un modelo secuencial
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3))) #Agregar una capa convolucional
model.add(layers.MaxPooling2D((2, 2))) #Agregar una capa de pooling
model.add(layers.Conv2D(64, (3, 3), activation='relu')) #Agregar una capa convolucional
model.add(layers.MaxPooling2D((2, 2))) #Agregar una capa de pooling
model.add(layers.Conv2D(64, (3, 3), activation='relu')) #Agregar una capa convolucional
```

Y se muestra un resumen de la arquitectura de la red neuronal.

```
model.summary() #Mostrar un resumen de la arquitectura de la red neuronal
```

```
[5]  1 model.summary()

Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 30, 30, 32)        896

 max_pooling2d (MaxPooling2  (None, 15, 15, 32)        0
 D)

 conv2d_1 (Conv2D)           (None, 13, 13, 64)        18496

 max_pooling2d_1 (MaxPoolin  (None, 6, 6, 64)          0
 g2D)

 conv2d_2 (Conv2D)           (None, 4, 4, 64)          36928

=================================================================
Total params: 56320 (220.00 KB)
Trainable params: 56320 (220.00 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Se agregan las capas densas para la clasificación y se muestra un resumen de la arquitectura.

```
model.add(layers.Flatten()) #Agregar una capa de aplanado
model.add(layers.Dense(64, activation='relu')) #Agregar una capa densa
model.add(layers.Dense(10)) #Agregar una capa densa
```

```
[7]
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(10))
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 30, 30, 32)        896

 max_pooling2d (MaxPooling2  (None, 15, 15, 32)        0
 D)

 conv2d_1 (Conv2D)           (None, 13, 13, 64)        18496

 max_pooling2d_1 (MaxPoolin  (None, 6, 6, 64)          0
 g2D)

 conv2d_2 (Conv2D)           (None, 4, 4, 64)          36928

 flatten (Flatten)           (None, 1024)              0

 dense (Dense)               (None, 64)                65600

 dense_1 (Dense)             (None, 10)                650

=================================================================
Total params: 122570 (478.79 KB)
Trainable params: 122570 (478.79 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Ya llegamos a la parte donde compilamos y entrenamos el modelo, utilizaremos el optimizador 'Adam' ya que es uno de los mas utilizados en la actualidad como la función de entropía cruzada para la clasificación, y la métrica de precisión para evaluar el rendimiento del modelo. Tendremos 10 épocas.

```
model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy']) #Compilar el modelo


history = model.fit(train_images, train_labels, epochs=10,
                    validation_data=(test_images, test_labels)) #Entrenar el modelo
```

```
model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])

history = model.fit(train_images, train_labels, epochs=10,
                    validation_data=(test_images, test_labels))
```

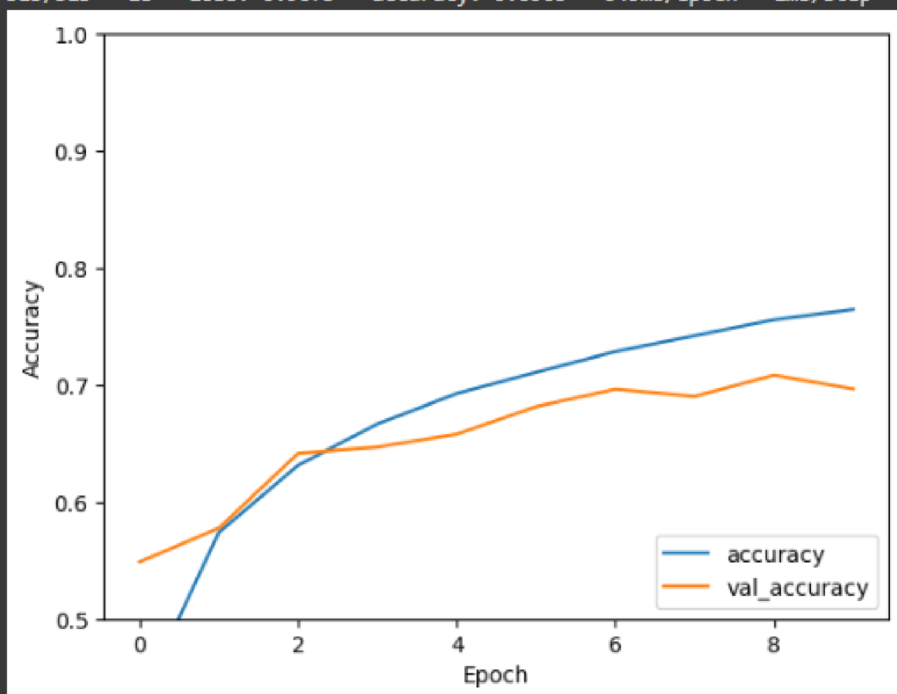
```
Epoch 1/10
1563/1563 [==============================] - 15s 6ms/step - loss: 1.5574 - accuracy: 0.4289 - val_loss: 1.2755 - val_accuracy: 0.5487
Epoch 2/10
1563/1563 [==============================] - 8s 5ms/step - loss: 1.1964 - accuracy: 0.5738 - val_loss: 1.1823 - val_accuracy: 0.5775
Epoch 3/10
1563/1563 [==============================] - 8s 5ms/step - loss: 1.0423 - accuracy: 0.6313 - val_loss: 1.0184 - val_accuracy: 0.6415
Epoch 4/10
1563/1563 [==============================] - 8s 5ms/step - loss: 0.9479 - accuracy: 0.6666 - val_loss: 1.0219 - val_accuracy: 0.6468
Epoch 5/10
1563/1563 [==============================] - 8s 5ms/step - loss: 0.8757 - accuracy: 0.6925 - val_loss: 0.9699 - val_accuracy: 0.6578
Epoch 6/10
1563/1563 [==============================] - 7s 5ms/step - loss: 0.8275 - accuracy: 0.7108 - val_loss: 0.9265 - val_accuracy: 0.6814
Epoch 7/10
1563/1563 [==============================] - 8s 5ms/step - loss: 0.7750 - accuracy: 0.7284 - val_loss: 0.8775 - val_accuracy: 0.6962
Epoch 8/10
1563/1563 [==============================] - 8s 5ms/step - loss: 0.7343 - accuracy: 0.7420 - val_loss: 0.9039 - val_accuracy: 0.6901
Epoch 9/10
1563/1563 [==============================] - 8s 5ms/step - loss: 0.6965 - accuracy: 0.7557 - val_loss: 0.8802 - val_accuracy: 0.7083
Epoch 10/10
1563/1563 [==============================] - 9s 6ms/step - loss: 0.6644 - accuracy: 0.7644 - val_loss: 0.9073 - val_accuracy: 0.6965
```

Ahora graficaremos la precisión del modelo en entrenamiento y en validación.

```
plt.plot(history.history['accuracy'], label='accuracy') #Graficar la precision en el entrenamiento
plt.plot(history.history['val_accuracy'], label = 'val_accuracy') #Graficar la precision en la validacion
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.ylim([0.5, 1])
plt.legend(loc='lower right')

test_loss, test_acc = model.evaluate(test_images,  test_labels, verbose=2) #Evaluar el modelo en el conjunto de
prueba
```

```python
plt.plot(history.history['accuracy'], label='accuracy')
plt.plot(history.history['val_accuracy'], label = 'val_accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.ylim([0.5, 1])
plt.legend(loc='lower right')

test_loss, test_acc = model.evaluate(test_images,  test_labels, verbose=2)
```

```
313/313 - 1s - loss: 0.9073 - accuracy: 0.6965 - 645ms/epoch - 2ms/step
```



Y para finalizar mostraremos la precisión del modelo.

```python
print(test_acc)
```

```
0.6965000033378601
```