



Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias de la Computación

Minería de Datos

Primavera 2022

1er Parcial

3 de Abril de 2022

| | |
|---------------------------------|-----------|
| BRYAN CASTILLO DE LOS SANTOS | 201904248 |
| TOMÁS GARCÍA GARCÍA | 201936947 |
| LUIS DAVID GONZALEZ AVENDAÑO | 201920252 |
| KARLA GUADALUPE RAMIREZ ALVAREZ | 201955422 |

1. Construir una Muestra(Conjunto de imágenes de incendios forestales a partir de una estrategia de muestreo propuesta para el problema).

La población de imágenes de las cuales partimos fue de 369 (250 px de ancho por 180 px de alto), este conjunto de imágenes en su totalidad son de incendio y no incendio forestal en bosque de coníferas, más específicamente esta es una población de imágenes de día.

MUESTREO ESTRATIFICADO

El muestreo estratificado es un tipo de muestreo probabilístico. Los individuos de toda la población se dividen en grupos o estratos. Cada elemento pertenece a un único estrato. La variable elegida para formar los estratos no debe permitir que un individuo o elemento de la población pertenezca a más de uno de ellos. En el caso de nuestra población dividiremos la población en 3 estratos, los 3 estratos serán humo, incendio y no incendió, la representación de estas particiones en la población se vería como la imagen a continuación. Véase imagen 1.

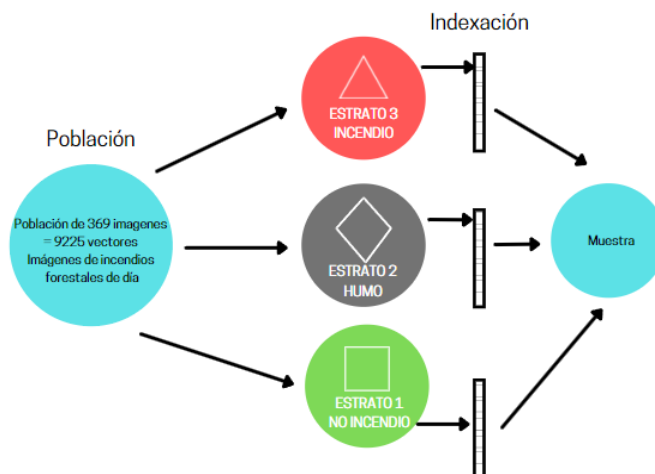


Imagen 1. Estratos.

La muestra se elige escogiendo en cada estrato un número representativo de individuos para ello se implementó un programa en python con el que cada imagen es particionada en 25 vectores es decir particiones. Cada partición se clasifica en alguno de los estratos mencionados. En el caso de la clase no incendio contamos con 2397 vectores, en la clase humo se cuenta con 2560, y por último en la clase incendio hay 2397.

Después del agrupamiento se procede a la indexación de instancias por clase esto utilizando un marco de muestreo aleatorio. Al término de la indexación se realiza un muestreo aleatorio simple por cada grupo. Véase imagen 2.

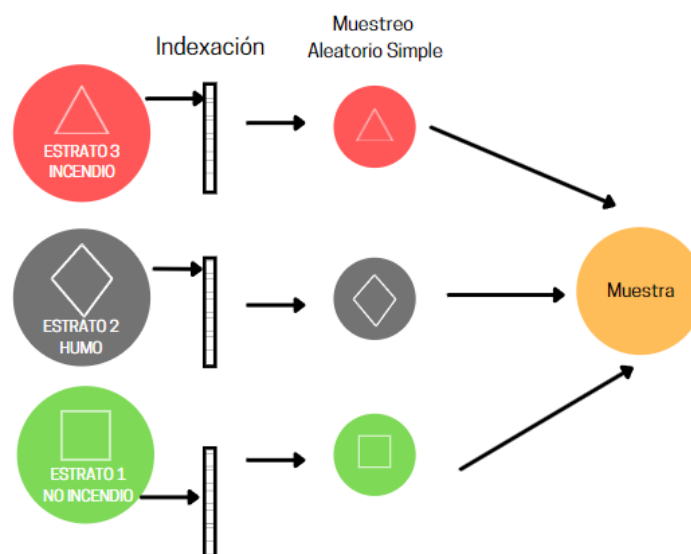


Imagen 2. Muestreo Aleatorio.

El muestreo aleatorio simple es una técnica de muestreo en la que todos los elementos que forman la población en este caso todos los que conforman el grupo y que por lo tanto están incluidos en el marco de indexación tienen idéntica probabilidad de ser seleccionados para la muestra.

Al aplicar el muestreo aleatorio utilizamos una herramienta web de la cual se seleccionaron un total de 1397 vectores de no incendio, 1397 de incendio y 1560 de humo. Al tener el número total de vectores por cada clase tenemos una muestra de 4354 instancias con el siguiente formato y la siguiente información relacionada. Véase imagen 3.

```
% 1. Title: Fire Detection
% 2. Relevant Information:
%
% This data is about texture and distribution of colors in images from
% fire in coniferous forest taken in day.
%
% This data set includes 1397 instances of the class "fire"
% and 1397 instances of "nofire" class, and 1560 instances of "smoke" class
% The instances are described by 12 attributes.
%..
%
% 3. Number of Instances: 4354
% |
% 4. Number of Attributes: 11 + the class attribute
%
% 5. For Each Attribute:
% 1. Average value of red color (numeric)
% 2. Average value of green color (numeric)
% 3. Average value of blue color (numeric)
% 4. Standard Deviation value of red color (numeric)
% 5. Standard Deviation value of green color (numeric)
% 6. Standard Deviation value of blue color (numeric)
% 7. Energy value of the picture converted to gray-scale (numeric)
% 8. Homogeneity value of the picture converted to gray-scale (numeric)
% 9. Contrast value of the picture converted to gray-scale (numeric)
% 10. Dissimilarity value of the picture converted to gray-scale (numeric)
% 11. Correlation value of the picture converted to gray-scale (numeric)
% 12. Class variable (fire, no_fire, smoke)
%
% 6. Missing Attribute Values: None
%
```

Imagen 3. Estructura.

2. Implementar un proceso de particionamiento para el etiquetado de zonas locales de una imagen con incendio forestal.

Para este punto se implementó un programa en python con el objetivo de particionar las imágenes iniciales en vectores(imágenes más pequeñas), así como clasificar cada partición en tres clases diferentes, cada partición se convierte en un vector dentro de un archivo el cual contiene aparte de su respectiva clase: no incendio, humo e incendio, el valor de siete atributos descriptivos, dos de ellos medidas de tendencia central y los otros 5 son medidas de textura.

El programa cuenta con una interfaz gráfica que se visualiza como la imagen a continuación. Véase imagen 4.

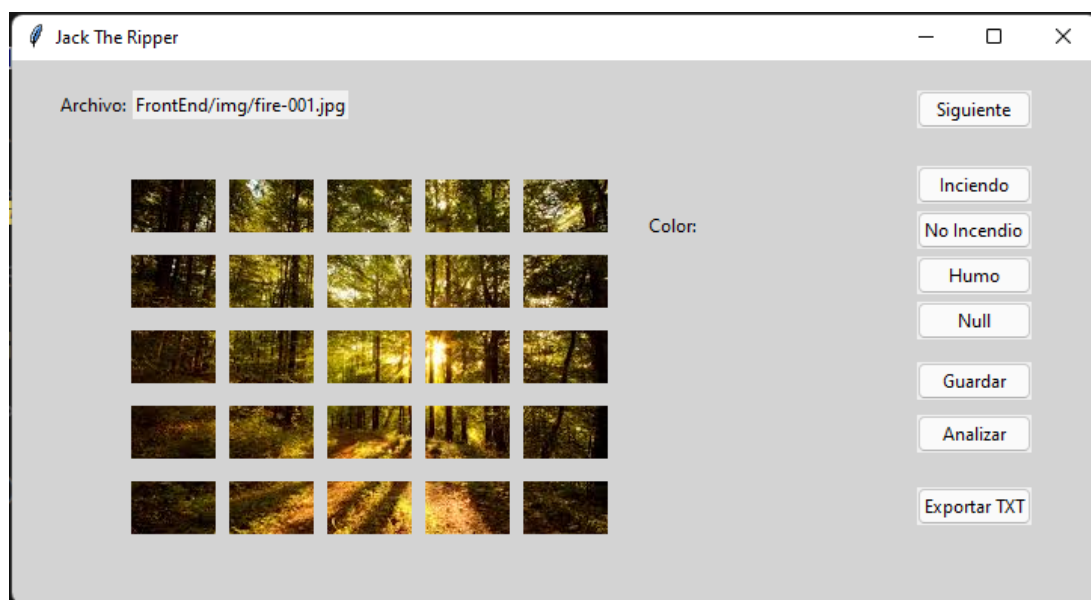
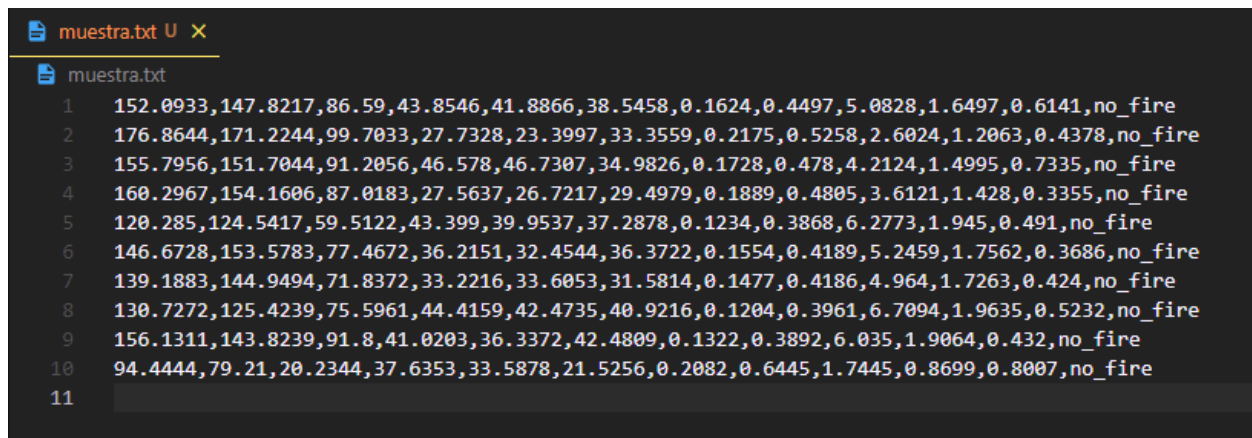


Imagen 4. Interfaz.

El software cuenta con diferentes funciones, la primera es, imágenes que son mostradas una por una en un formato donde podemos ver la imagen particionada en vectores de 50px x 36px. La interfaz cuenta también con 6 diferentes botones. el

primero ubicado en la esquina superior derecha llamado siguiente que sirve para recorrer las imágenes recolectadas. Los botones “Incendio”, “No Incendio” y “Humo” sirven para marcar las particiones de la imagen con su etiqueta correspondiente: incendio, no incendio y humo. El botón de Guardar se ocupa de cada categorización de imagen y guarda los valores para después seleccionarlos. El botón analizar sirve para cuando se recorran suficientes imágenes haga los cálculos de tendencia central y los de textura (la imagen convertida a escala de grises). Finalmente el botón de exportar sirve para exportar esos datos calculados anteriormente hacía un archivo de extensión “.txt”, de donde se tomarán los datos capturados para el archivo “.arff” de weka.



```
muestra.txt U X
muestra.txt
1 152.0933,147.8217,86.59,43.8546,41.8866,38.5458,0.1624,0.4497,5.0828,1.6497,0.6141,no_fire
2 176.8644,171.2244,99.7033,27.7328,23.3997,33.3559,0.2175,0.5258,2.6024,1.2063,0.4378,no_fire
3 155.7956,151.7044,91.2056,46.578,46.7307,34.9826,0.1728,0.478,4.2124,1.4995,0.7335,no_fire
4 160.2967,154.1606,87.0183,27.5637,26.7217,29.4979,0.1889,0.4805,3.6121,1.428,0.3355,no_fire
5 120.285,124.5417,59.5122,43.399,39.9537,37.2878,0.1234,0.3868,6.2773,1.945,0.491,no_fire
6 146.6728,153.5783,77.4672,36.2151,32.4544,36.3722,0.1554,0.4189,5.2459,1.7562,0.3686,no_fire
7 139.1883,144.9494,71.8372,33.2216,33.6053,31.5814,0.1477,0.4186,4.964,1.7263,0.424,no_fire
8 130.7272,125.4239,75.5961,44.4159,42.4735,40.9216,0.1204,0.3961,6.7094,1.9635,0.5232,no_fire
9 156.1311,143.8239,91.8,41.0203,36.3372,42.4809,0.1322,0.3892,6.035,1.9064,0.432,no_fire
10 94.4444,79.21,20.2344,37.6353,33.5878,21.5256,0.2082,0.6445,1.7445,0.8699,0.8007,no_fire
11
```

Imagen 5. Instancias.

3. Definir un conjunto de valores característicos a extraer de las imágenes locales a partir de una imagen de entrada.

La información contenida en una imagen multiespectral proporciona abundantes datos a quien las analiza, por ello se necesita recurrir a la estadística para cuantificar las diferentes características de la imagen. Todos los descriptores listados a continuación son parámetros fundamentales para caracterizar estadísticamente una imagen multiespectral.

La elección de la media como atributo clasificador se debe a que es el valor más importante cuando hay datos dispersos, en este caso es aplicado a una imagen en la que cada pixel tiene un número diferente de RGB por lo tanto existe valores dispersos y la media nos informa un valor representativo del conjunto de datos.

En el caso de la desviación estándar la elección de su uso va de la mano con la media ya que mide la distribución de datos en torno a la media.

Media

La media se utiliza para distribuciones normales de números, con una cantidad baja de valores atípicos. En este caso este descriptor nos ayuda a entender la tendencia central, en el caso de que existan distribuciones numéricas sesgadas

$$Media(X) + \bar{x} = \frac{\sum_{i=1}^N X_i}{N}$$

Siendo (X_1, X_2, \dots, X_N) cada dato en su serie

Desviación Estándar

Es alta cuando la desviación estándar de los píxeles dentro de la ventana es también alta. La desviación estándar es la medida de dispersión más común, que indica qué tan dispersos están los datos con respecto a la media. Mientras mayor sea la desviación estándar, mayor será la dispersión de los datos.

$$\sigma = \sqrt{\frac{\sum_i^n (X_i - \bar{X})^2}{N}}$$

Medidas de Textura

Las medidas de textura son conjuntos de métricas calculadas en el procesamiento de imágenes que están diseñadas para cuantificar la textura percibida en una imagen a escala de grises.

La textura de la imagen nos brinda información sobre la disposición espacial del color o las intensidades en una imagen o región de ella. Las texturas ayudan a la segmentación o clasificación de las imágenes. Para una segmentación más precisa, las características más útiles son la frecuencia espacial y un nivel de gris medio.

Para estos últimos 5 atributos clasificadores es necesaria la matriz de co-ocurrencia, la matriz de co-ocurrencia describe la frecuencia de un nivel de gris que aparece en una relación espacial específica con otro valor de gris, dentro del área de una ventana determinada. La matriz de co-ocurrencia es un resumen de la forma en que los valores de los píxeles ocurren al lado de otro valor en una pequeña ventana.

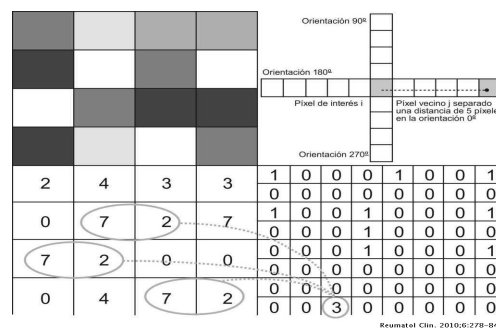


Imagen 6. Ejemplo Matriz de Concurrencia.

Homogeneidad

$$\frac{\sum_{i,j=0}^{N-1} P_{i,j}}{1 + (i-j)^2}$$

Siendo P_{ij} la probabilidad de co-ocurrencia de los valores de gris i y j , para una distancia dada. Esta variable cuantifica el grado de homogeneidad que presenta la imagen pero de una forma completamente distinta a la uniformidad. En la ecuación puede observarse que para el cálculo de la homogeneidad se tienen en cuenta las diferencias entre el nivel de gris de cada uno de los píxeles de la pareja; por lo que cuando en una imagen las diferencias en los niveles de gris de los píxeles son pequeñas, la uniformidad tenderá a ser mayor, mientras que si las diferencias en los píxeles son grandes, la uniformidad será menor.

La homogeneidad fue elegida como medida de textura ya que mide la uniformidad de los valores de los niveles de grises en la imagen dejándonos así con un valor importante con relación a valores comunes entre píxeles.

Contraste

Es lo opuesto a la homogeneidad, es decir es una medida de la variación local en una imagen. Esto quiere decir que mide la diferencia entre valores de píxeles. Tiene un valor alto cuando la región dentro de la escala de la ventana tiene un alto contraste.

$$\sum_{i,j=0}^{N-1} P_{i,j}(i-j)^2$$

Disimilaridad

Se da, en general, el nombre de distancia o disimilaridad entre dos píxeles i y j a una medida, indicada por $d(i,j)$, que mide el grado de semejanza, o a mejor decir de desemejanza, entre ambos píxeles, en relación a un cierto número de características cuantitativa y / o cualitativas. El valor de $d(i,j)$ es siempre un valor no negativo, y cuanto mayor sea este valor mayor será la diferencia entre los individuos i y j .

$$\sum_{i,j=0}^{N-1} P_{i,j} |i-j|$$

Entropía

La entropía es una medida estadística de aleatoriedad que se puede utilizar para caracterizar la textura de la imagen de entrada. Es alta cuando los elementos de la matriz de coocurrencia tienen relativamente valores iguales. Es baja cuando los elementos son cercanos a 0 o 1

$$\sum_{i,j=0}^{N-1} P_{i,j} \ln(P_{i,j})$$

Se asume que $0 \cdot \ln(0) = 0$

Puede interpretarse como el grado de desorden o caos (en el sentido físico del término) presente en la imagen. Tomará su máximo valor cuando todas las parejas de píxeles estén representadas en la imagen con la misma probabilidad, lo que de alguna forma significa el máximo desorden posible en la imagen.

Energía

La transformación de energía refleja la uniformidad de la distribución de grises de la imagen y el grosor de la textura. Si los valores de los elementos de la matriz de co-ocurrencia del nivel de grises son similares, la energía es pequeña, lo que significa que la textura es más fina; si algunos de los valores son grandes y otros valores son pequeños, el valor de energía es grande. Un valor energético elevado indica un patrón de textura más uniforme y regular.

$$Asm = \sum_i \sum_j P(i,j)^2$$

4. Realizar un análisis y argumentar que tipo de validación cruzada(que valor de K o valores) se usará en la experimentación considerando que se debe usar la mayor cantidad de cruces posibles, garantizando en todo momento la correcta medición de lo que sucede cada clase .

La validación cruzada es una técnica utilizada para la evaluación de resultados de un análisis estadístico. La validación cruzada k-fold significa que el conjunto de datos se divide en un número K. Divide el conjunto de datos en el punto en el que el conjunto de prueba utiliza cada partición. En este escenario, el método dividirá el conjunto de datos en ocho particiones. El modelo utiliza la primera partición en la primera iteración para probar el modelo. Utiliza los conjuntos de datos restantes para entrenar el modelo. La segunda partición ayuda a probar el conjunto de datos y el otro apoya el proceso de entrenamiento. El mismo proceso se repite hasta que el conjunto de pruebas utiliza las 8 particiones.

Usamos una validación cruzada de $k = 8$ debido a que nos dio buenos resultados con la combinación de que tuviera usando el algoritmo Multilayer Perceptron en el programa de Weka. Esto se escogió ya que queremos una buena base para hacer el análisis del algoritmo, tomando un buen número de instancias debido a que los datos son para predecir si tiene humo, fuego o no fuego, entonces podemos respaldarnos con un buen número de instancias, al igual teniendo en cuenta que en el árbol de clasificación no debe ser tan individual pues esto provoca que exista un overfitting, pero tampoco podemos dejar que el árbol sea demasiado general pues puede provocar un underfitting. Esto se evita con la validación cruzada, con $k = 8$ tenemos un buen número de instancias para entrenar y testear. Véase tabla 1.

| K = | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|------|---------|---------|---------|---------|---------|---------|---------|---------|
| 1397 | 698-699 | 465-466 | 349-350 | 279-280 | 232-233 | 199-200 | 174-175 | 155-156 |
| 1397 | 698-699 | 465-466 | 349-350 | 279-280 | 232-233 | 199-200 | 174-175 | 155-156 |
| 1560 | 780 | 520 | 390 | 312 | 260 | 222-223 | 195 | 173-174 |

Tabla 1. Validación Cruzada.

5. Definir la metodología a utilizar para el entrenamiento de la RNA.

Para el entrenamiento de la red neuronal utilizamos una red de tipo perceptrón multicapa. RNA supervisada ya que necesita conocer los valores esperados para cada una de las entradas presentadas a la red. Este es uno de los tipos de redes más comunes. Se basa en otra red más simple llamada perceptrón simple solo que el número de capas ocultas puede ser mayor o igual que una. Es una red unidireccional. La arquitectura típica de esta red es la siguiente. Véase imagen 7.

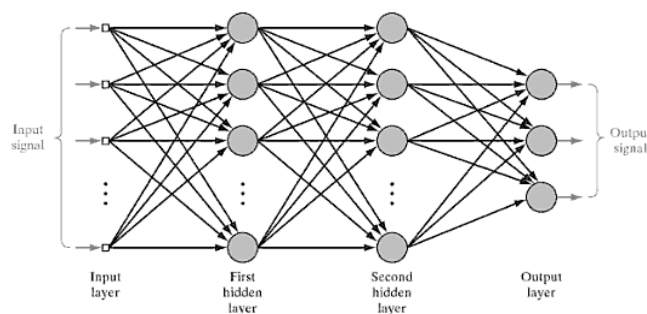


Imagen 7. Arquitectura gráfica de una red perceptron multicapas.

Las neuronas de la capa oculta usan como regla de propagación la suma ponderada de las entradas con los pesos sinápticos y sobre esa suma ponderada se aplica una función de transferencia de tipo sigmoide, que es acotada en respuesta. El aprendizaje que se suele usar en este tipo de redes recibe el nombre de retropropagación del error (backpropagation).

La fórmula de aprendizaje de una red tipo perceptrón es la siguiente:

$$a = f\left(\sum_{i=1}^s w_i p_i\right) \neq \text{hard lim } s\left(\sum_{i=1}^s w_i p_i\right)$$

Para cada entrada p , la salida es comparada con t : Si $a_i = t_i$, no se realizan ajustes Si $a_i \neq t_i$, se realizan ajustes en los pesos w . La siguiente tabla contiene la información de los datos de entrada para el entrenamiento de la red neuronal. Véase tabla 2.

| # | Neuronas | Capas | Épocas | L. Rate | Momentum |
|---|----------|-------|---------|---------|----------|
| 1 | 13 | 4 | 100-150 | 0.3 | 0.2 |
| 2 | 26 | 4 | 80-150 | 0.2 | 0.1 |
| 3 | 19 | 4 | 50-180 | 0.2 | 0.1 |

Tabla 2. Valores de Entrada.

Las RNA supervisadas necesitan conocer los valores esperados para que al momento de una entrada se pueda hacer uso de la función de transferencia que mejor se ajuste al problema.

A) El número de neuronas en el caso de la primera configuración partimos de un número equivalente al número total de atributos más 1 de la muestra, para la segunda configuración de 26 neuronas decidimos duplicar solo el número inicial para verificar si el porcentaje de error se reducía de forma significativa o decrecía. Por último en el caso de 19 neuronas fue resultado de una búsqueda entre diferentes valores entre el punto de partida y la duplicación de neuronas con valor de 26.

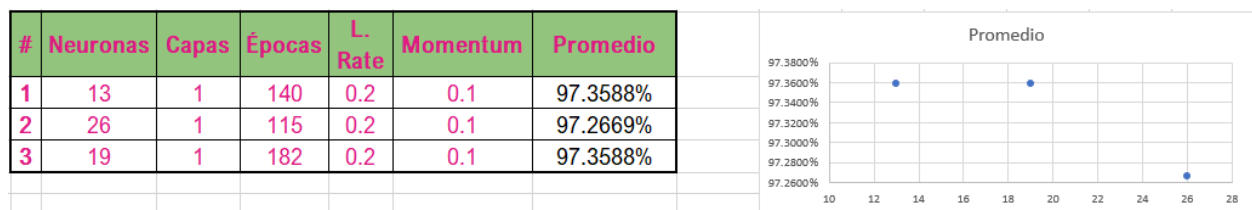
B) Las capas en las diferentes configuraciones son 4 ya que fue el valor donde era más fácil observar el movimiento de incremento o decremento del error dado que los datos se mantienen sin menos subidas y bajadas de forma inesperada.

C) Las épocas colocadas en la tabla (Véase tabla 1) son el rango de valores en los cuales fueron realizados los distintos experimentos detallados en el siguiente inciso de este documento.

D) Se escogió el learning rate de 0.3 y momentum de 0.2 debido a que son los ajustes “default” más recomendados en la minería de datos. En el caso de 0.2 de learning rate y 0.1 de momentum, fue debido a que se puede tener un cálculo aunque mas “lento” mas fino porque se hace con más procesos que hace que llegue a tener más precisión.

6. Realizar un proceso de búsqueda de parámetros que permitan encontrar la mejor clasificación que se pueda obtener con la RNA y la muestra a trabajar.

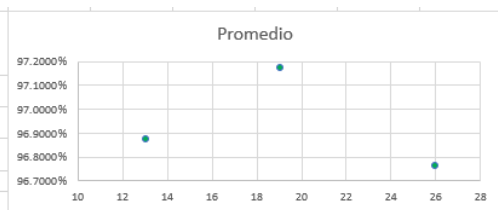
A) Para una capa los experimentos mostraban un alto porcentaje de aprendizaje por lo que decidimos probar con un valor más alto de neuronas para que la red no cayera en overfitting.



Gráfica 1. Número-Capas-1.

Se experimentó en los 3 experimentos con el número 1 de capas con 0.2 de learning rate y 0.1 de momentum, con variación en el número de épocas en cada caso donde fue resultante de un promedio alrededor del 97%.

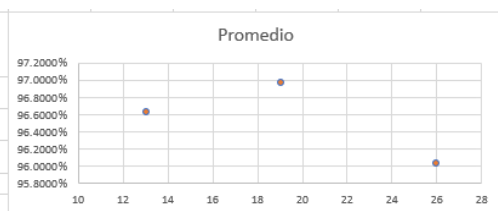
| # | Neuronas | Capas | Épocas | L. Rate | Momentum | Promedio |
|---|----------|-------|--------|---------|----------|----------|
| 1 | 13 | 2 | 140 | 0.2 | 0.1 | 96.8764% |
| 2 | 26 | 2 | 115 | 0.2 | 0.1 | 96.7616% |
| 3 | 19 | 2 | 182 | 0.2 | 0.1 | 97.1750% |



Gráfica 2. Número Capas-1.

Con los experimentos anteriores con misma configuración a excepción de un número de capas de 2 se obtuvo un promedio alrededor del 96% y 97%.

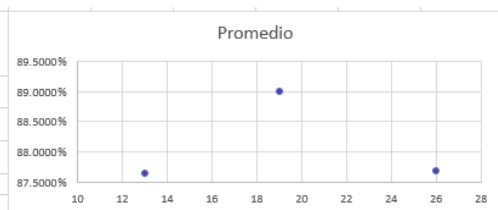
| # | Neuronas | Capas | Épocas | L. Rate | Momentum | Promedio |
|---|----------|-------|--------|---------|----------|----------|
| 1 | 13 | 3 | 140 | 0.2 | 0.1 | 96.6238% |
| 2 | 26 | 3 | 115 | 0.2 | 0.1 | 96.0323% |
| 3 | 19 | 3 | 182 | 0.2 | 0.1 | 96.9683% |



Gráfica 3. Número Capas-3.

Con 3 capas se obtuvo menor promedio que la configuraciones con menos capas, pues se llegó de promedio a un 96.0% como mínimo y un 96.9% como máximo.

| # | Neuronas | Capas | Épocas | L. Rate | Momentum | Promedio |
|---|----------|-------|--------|---------|----------|----------|
| 5 | 13 | 4 | 140 | 0.2 | 0.1 | 87.6435% |
| 4 | 26 | 4 | 115 | 0.2 | 0.1 | 87.6895% |
| 4 | 19 | 4 | 182 | 0.2 | 0.1 | 88.9986% |

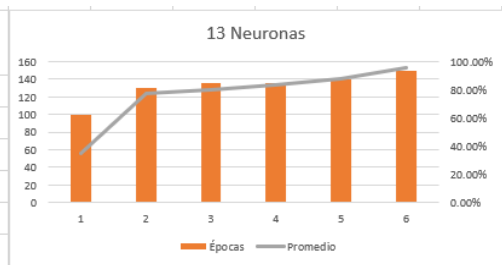


Gráfica 4. Número Capas-4.

Al llegar con el número de capas de 4, se obtuvo una diferencia más grande en resultados de promedio pues el máximo que se obtuvo fue 88.99% y el menor de 87.64%.

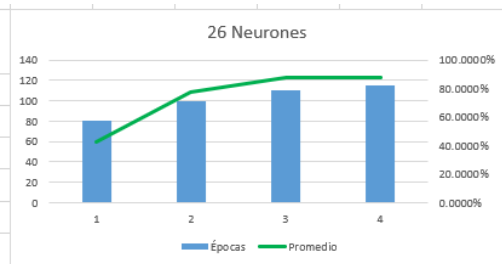
B) A continuación se presentan las gráficas de los porcentajes de acierto con relación al número de épocas y neuronas de las configuraciones listadas anteriormente.

| # | Neuronas | Capas | Épocas | L. Rate | Momentum | Promedio |
|---|----------|-------|--------|---------|----------|----------|
| 1 | 13 | 4 | 100 | 0.2 | 0.1 | 35.3698% |
| 2 | 13 | 4 | 130 | 0.2 | 0.1 | 77.8594% |
| 3 | 13 | 4 | 135 | 0.2 | 0.1 | 79.6968% |
| 4 | 13 | 4 | 135 | 0.2 | 0.1 | 83.5783% |
| 5 | 13 | 4 | 140 | 0.2 | 0.1 | 87.6435% |
| 6 | 13 | 4 | 150 | 0.2 | 0.1 | 95.7051% |



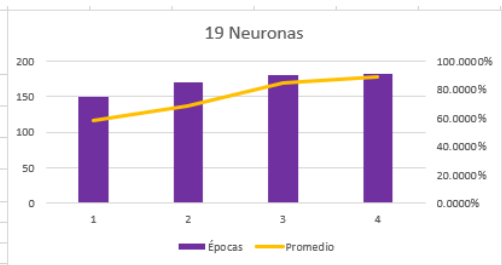
Gráfica 5. Resultados-13-Neuronas.

| # | Neuronas | Capas | Épocas | L. Rate | Momentum | Promedio |
|---|----------|-------|--------|---------|----------|----------|
| 1 | 26 | 4 | 80 | 0.2 | 0.1 | 42.7423% |
| 2 | 26 | 4 | 100 | 0.2 | 0.1 | 77.1704% |
| 3 | 26 | 4 | 110 | 0.2 | 0.1 | 87.4828% |
| 4 | 26 | 4 | 115 | 0.2 | 0.1 | 87.6895% |



Gráfica 6. Resultados-26-Neuronas.

| # | Neuronas | Capas | Épocas | L. Rate | Momentum | Promedio |
|---|----------|-------|--------|---------|----------|----------|
| 1 | 19 | 4 | 150 | 0.2 | 0.1 | 57.8089% |
| 3 | 19 | 4 | 170 | 0.2 | 0.1 | 68.6495% |
| 3 | 19 | 4 | 180 | 0.2 | 0.1 | 84.9340% |
| 4 | 19 | 4 | 182 | 0.2 | 0.1 | 88.9986% |



Gráfica 7. Resultados-19-Neuronas.

C) Al inicio que se usó un learning rate de 0.3 y un momentum de 0.2 pues son valores default y recomendados, al estar haciendo los experimentos notamos que al menos con estos 2 valores los valores se disparaban muy alto que hacía que cayera en sobreajuste a las pocas épocas, además que hacía más difícil de predecir en qué épocas daba esos grandes saltos; lo que se implementó fue bajar los valores de learning rate a 0.2 y el de momentum a 0.1 pues hicieron aun teniendo que agregar más épocas, que hubiera mejores resultados respecto a instancias clasificadas correctamente.

En las siguientes imágenes podemos observar la diferencia al utilizar estas dos diferentes configuraciones. (Véase imagen 8 y 9).

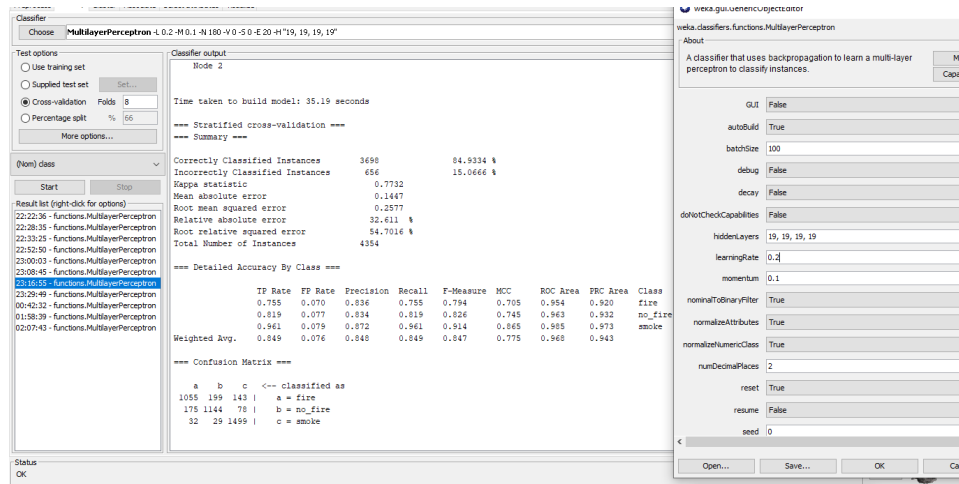


Imagen 8. Learning-Momentum 1.

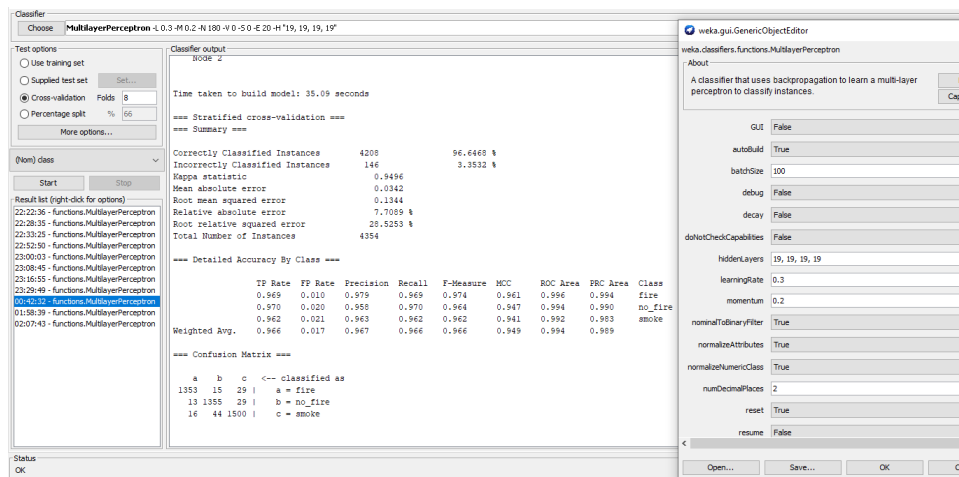


Imagen 9. Learning-Momentum 2.

7. Hacer un análisis de los resultados obtenidos, donde se explique la forma en la cual influyó la búsqueda de parámetros para obtener un mejor resultado en el proceso de clasificación.

El trabajo de búsqueda de parámetros fue fundamental para obtener resultados de clasificación más altos comenzando con el número de capas ya que al empezar los entrenamientos desde la 1ra capa hasta la 4ta capa nos dio resultados favorables y congruentes, a partir de la 5ta capa nos daba resultados inconsistentes que no podíamos tomar como válidos, es por ello que los experimentos se basaron en el rango de 1 a 4 capas, tomando como valor predilecto el 4. En el caso de las neuronas iniciamos con el valor de los atributos más un agregado, para este primer valor de referencia o inicial y a partir del buen comportamiento de esta configuración se llevaron a cabo más experimentos llegando a un número del doble de neuronas y escogiendo un punto entre ellos hasta encontrar una configuración favorable para que el nivel de instancias . Lo siguiente fue ajustar las épocas con las que se iba a tratar, primero se consideró un número base para después de ver el resultado aumentarlo o disminuirlo con respecto a este, ese número base era 50, en la mayoría de casos se tuvo que aumentar gradualmente de 10 en 10 hasta llegar a un número adecuado, en menor medida los casos de aumento era de un sola época esto dado que las diferencias eran más notorias por época donde los resultados estaban en un rango de 80% y 90% de clasificaciones correctas. Para finalizar el learning rate fue reducido de 0.3 a 0.2 para reducir el porcentaje de cambios con el que se actualizan los pesos en cada interacción dejando ver con mayor facilidad el aumento o decremento del valor del error. El momentum fue reducido de 0.2 a 0.1 ya que los valores de default del programa proporcionaban un movimiento bastante brusco del vector gradiente, impidiendo así un análisis más notable del porcentaje de error.