TAE Trabajo número 4

William aguilar Restrepo, Santiago Cardona

21 de marzo de 2019

Clasificación de dígitos.

Definiciones.

“La base de datos MNIST (base de datos modificada del Instituto Nacional de Estándares y Tecnología) es una gran base de datos de dígitos escritos a mano que se usa comúnmente para capacitar a varios sistemas de procesamiento de imágenes. La base de datos también se usa ampliamente para capacitación y pruebas en el campo del aprendizaje automático. Fue creado”mezclando de nuevo" las muestras de los conjuntos de datos originales de NIST. Los creadores sintieron que dado que el conjunto de datos de entrenamiento del NIST se tomó de los empleados de la Oficina del Censo de Estados Unidos, mientras que el conjunto de datos de prueba se tomó de American estudiantes de secundaria, no era adecuado para experimentos de aprendizaje automático. Además, las imágenes en blanco y negro del NIST se normalizaron para encajar en un cuadro delimitador de 28x28 píxeles y con antialias, lo que introdujo niveles de escala de grises." (***Wikipedia***)

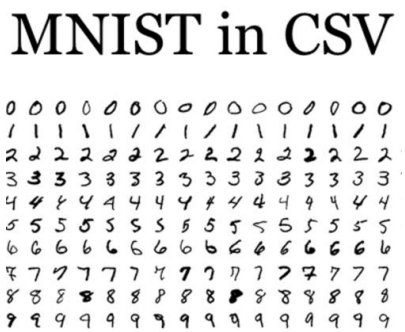


Imagen 1. *Datos mnist.*

“La base de datos MNIST contiene 60,000 imágenes de entrenamiento y 10,000 imágenes de prueba. La mitad del conjunto de entrenamiento y la mitad del conjunto de prueba se tomaron del conjunto de datos de entrenamiento del NIST, mientras que la otra mitad del conjunto de entrenamiento y la otra mitad del conjunto de prueba se tomaron del conjunto de datos de prueba del NIST.” (***Wikipedia***)

Imagen 2. *Datos mnist con solo en número 2.*

El ánalisis de datos **MNIST** es realizado por medio de **análisis supervisado**, este tipo de análisis comprende un areá de análisis sobre el **Machine Learning**, estos dos conceptos los explicamos a continuación:

Machine Learning:

"Machine Learning es una disciplina científica del ámbito de la Inteligencia Artificial que crea sistemas que aprenden automáticamente. Aprender en este contexto quiere decir identificar patrones complejos en millones de datos. La máquina que realmente aprende es un algoritmo que revisa los datos y es capaz de predecir comportamientos futuros." (***CleverData***) "Es un método de análisis de datos que automatiza la construcción de modelos analíticos." (***SAS***)

Aprendizaje supervisado

"El aprendizaje supervisado es un tipo de algoritmo de Machine Learning que emplea un conjunto de datos conocidos (el denominado conjunto de datos de entrenamiento) para realizar predicciones. El conjunto de datos de entrenamiento incluye datos de entrada y valores de respuesta. A partir de él, el algoritmo de aprendizaje supervisado busca crear un modelo que pueda realizar predicciones acerca de los valores de respuesta para un nuevo conjunto de datos. Con frecuencia se utiliza un conjunto de datos de prueba para validar el modelo. Si se utilizan conjuntos de datos de entrenamiento de mayor tamaño, a menudo se generan modelos cuya capacidad predictiva es mayor y que se pueden aplicar con buenos resultados sobre nuevos conjuntos de datos." (***MathWorks***)

Métodología

El problema a resolver para los datos MNIST es usar los diferentes métodos de aprendizaje supervisado usados en clase (Técnicas de Aprendizaje Estadístico en la Universidad Nacional de Colombia - Sede Medellín) que corresponden a los siguientes tipos de técnica:

1. Regresión logística multinomial.
2. Arboles de clasificación.

3. Bosques aleatorios.
4. Máquinas de soporte vectorial.
5. Redes neuronales.

Estos diferentes tipos de técnicas se abarcaran más adelante, con las diferentes configuraciones seleccionadas, ya que requieren parámetros que optimizan según la necesidad del método seleccionado.

Datos

Los datos son suministrados por el profesor Juan David Ospina Arango, obtenidas de **"THE MNIST DATABASE of handwritten digits"**, la pagina web comprende cierta información acerca de análisis de los datos MNIST usando diferentes métodos de Machine Learning, además de la base de datos, separada en 4 bases de datos, la primera contiene la base de datos de las variables regresoras para el entrenamiento, es decir, contiene 784 variables que corresponde al tamaño de la imagen de 28x28 con 60000 observaciones correspondiente a los números del código postal, la segunda base de datos contiene el número al cual está asociada las variables regresoras, en este caso se menciona que a las 784 variables asociadas para cada observación se encuentra una relación con una variable respuesta que corresponde al número del código postal, por este suceso de que hallan variables regresoras y una variable respuesta, se usa un método de aprendizaje supervisado, y las dos últimas bases de datos son iguales que las dos anteriormente mencionadas solo que con menos observaciones, en este caso 10000, y son diferentes observaciones que las dos primeras bases de datos, se debe a que las dos primeras bases de datos corresponden al **entrenamiento** y **validación**, y las dos últimas son para realizar la **prueba**. Y qué es entrenamiento, prueba y validación?

Entrenamiento, validación y prueba.

"Cuando realizamos modelos predictivos hay 3 conjuntos de datos fundamentales que todo dataminer debe manejar:

1. Muestra de Entrenamiento (TRAINING) : son los datos con los que se entrenan los modelos.

2. Muestra de Validación (VALIDATION) : selecciona el mejor de los modelos entrenados.

3. Muestra de Prueba (TEST) : Entrega el error real cometido con el modelo seleccionado." (***ANÁLISIS Y DECISIÓN***)

"Cuando tenemos suficientes datos, se puede subdividir los datos en estos tres conjuntos. Durante el proceso de seleción del mejor modelo, los modelos se ajustan a los datos de entrenamiento y el error de predicción para dichos modelos es obtenido mediante el uso de los datos de validación. Este error de predicción en los datos de validación se puede utilizar (es decir, el algoritmo los utiliza) para decidir cuándo dar por terminado el proceso de selección o para decidir cuáles son los efectos a incluir a

medida que avanza el proceso. Finalmente, una vez que termina el proceso y se tiene seleccionado el modelo, se pueden utilizar los datos de prueba para evaluar la manera en que el modelo seleccionado se generaliza para los datos que no jugaron ningún papel en la selección del mismo; En algunos casos es posible que se desee utilizar sólo los datos de entrenamiento y prueba." (***WEBMINING CONSULTORES***)

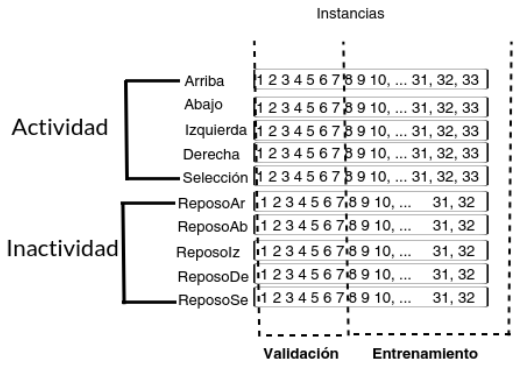


Imagen 3. *Ejemplo sobre entrenamiento y validación.*

Tasa de clasificación o análisis discriminante

No se ha mencionado lo que es una tasa declasificación, pero, es el método que se utiliza para hacer un análisis comparativo de los diferente métodos de aprendizaje o configuraciones utilizadas para el análisis, y qué es una tasa de clasificación?

"En el análisis discriminante estudiamos las técnicas de clasificación de sujetos en grupos ya definidos. Partimos de una muestra de N sujetos en los que se ha medido p variables cuantitativas independientes, que son las que se utilizarán para tomar la decisión en cuanto al grupo en el que se clasifica cada sujeto, mediante el modelo estimado a partir de los datos." (***Alceingenieria***)

Dificultades para el análisis.

1. El análisis descriptivo para las variables regresoras (pixeles) no es posible, ya que no da una idea básica de como se comportan los datos, a menos de que se reproduzca la imagen, es decir, dar uso a los valores de cada variable y asignarles un color de tal manera que sea posible ver la imagen del número que corresponde al resultado de la variable respuesta.

2. El problema computacional que requiere al recorrer los algoritmo ya que se cuenta con matrices de 60000x784 = 47.040.000 valores dentro de la matriz, haciendo que los problemas algorítmicos no tengan dificultades, como por ejemplo agregar muchas capas y muchos nodos a una red neuronal, hacer una tolerancia y un costo muy bajo a una máquina de soporte vectorial, muchos arboles a un bosque aleatorio, entre otros.

3. Durante los procesos computacionales no se permite usar el computador para realizar otras actividades, de realizar otra actividad los procesos

computacionales son aun más lentos y no solo eso, si no que adicionalmente el modelo tiene una peor precisión para aproximar a un buen modelo.

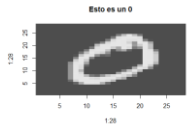
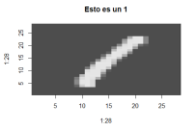
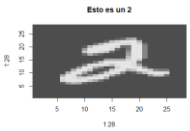
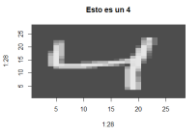
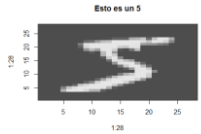
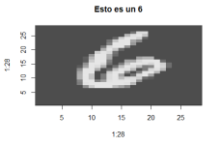
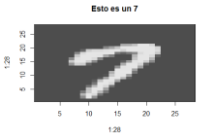
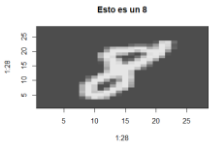
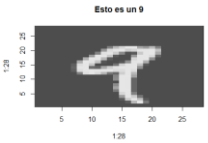
Facilidades en el análisis.

1. Las imagenes ya son transcritas a una matriz, lo que se convierte en solo un problema de saber cual es el mejor método que resuelva el problema mnist.

2. Las imagenes quedan de la misma dimensión para todas, es decir, las imagenes son de 28x28, de no ser así, los resultados para las dimensiones de la matriz se convierte en un dilema fundamental.

Análisis descriptivo.

Como se ha mencionado anteriormente, el análisis descriptivo es bastante complicado en relación a las variables regresoras, por lo que, es preferible abarcar el análisis con respecto a las imagenes que arrojan dichas variables, a continuación se muestran gráficamente el resultado de las imagenes que arroja las variables regresoras, el código es suministrado por el profesor Juan David Ospina Arango.

 ***Imagen 4.*** *Número 0 de mnist.*	 ***Imagen 5.*** *Número 1 de mnist.*	 ***Imagen 6.*** *Número 2 de mnist.*	 ***Imagen 7.*** *Número 3 de mnist.*	 ***Imagen 8.*** *Número 4 de mnist.*
 ***Imagen 9.*** *Número 5 de mnist.*	 ***Imagen 10.*** *Número 6 de mnist.*	 ***Imagen 11.*** *Número 7 de mnist.*	 ***Imagen 12.*** *Número 8 de mnist.*	 ***Imagen 13.*** *Número 9 de mnist.*

Es curioso observar la relación que tiene un pixel con la imagen, ya que el tono más oscuro de gris son las variables que contienen un valor de 0 en la observación, mientras que, un tono más claro representan números más altos, ya que son variables que otorgan información para observar el valor real del número que se muestra para la variable respuesta.

Clasificadores multiclase.

Para los diferentes clasificadores multiclase, se hace una breve introducción del método de aprendizaje supervisado, y los diferentes modelos que se usan. El resultado solo va a ser un modelo bajo ciertos paramétros y la conclusión del análisis computacional bajo una tasa de clasificación hecha con la base de datos test.

Regresión logística multinomial.

Introducción

Comenzando con la idea de la regresión logística simple, que corresponde a: "La Regresión Logística Simple, desarrollada por David Cox en 1958, es un método de regresión que permite estimar la probabilidad de una variable cualitativa binaria en función de una variable cuantitativa. Una de las principales aplicaciones de la regresión logística es la de clasificación binaria, en el que las observaciones se clasifican en un grupo u otro dependiendo del valor que tome la variable empleada como predictor." (***Rpubs***, *Joaquin AR*) Por lo que "la regresión logística multinomial generaliza el método de regresión logística simple para problemas multiclase, es decir, con más de dos posibles resultados discretos. Es decir, se trata de un modelo que se utiliza para predecir las probabilidades de los diferentes resultados posibles de una distribución categórica como variable dependiente, dado un conjunto de variables independientes (que pueden ser de valor real, valor binario, categórico-valorado, etc.)" (***Wikipedia***)

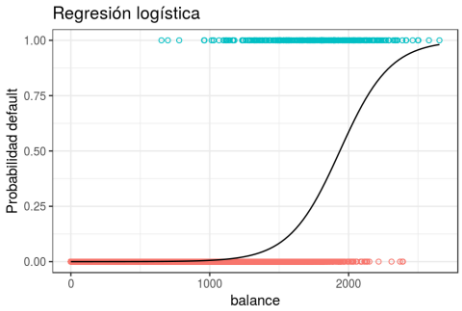


Imagen 14. *Regresión logística simple.*

Metodología implementada

Utilizamos la función *multinom* del paquete *nnet* para estimar un modelo de regresión logística multinomial.

Tomando la información en la base de datos del Train como se menciona anteriormente, bajo diferentes configuraciones, entre ellas, las cantidades de iteraciones que deba hacer la función logística con la red neuronal para un mayor aprendizaje a la regresión (*iter*), una tolerancia absoluta (*abstol*), es decir, una mayor precisión en los resultados, pero con un tiempo de computación más rápido si es más grande que el de la configuración por defecto que tiene la función, o más lento si la tolerancia es menor a la configuración por defecto, (*reltol*) actua de la misma manera

que el parámetro *abstol*, solo que *abstol* actua para detenerse cuando el criterio de ajuste está por debajo, y *reltol* si el optimizador no puede reducir el criterio de ajuste en un factor de al menos **1 - reltol**. Tener en cuenta que usar el parámetro *MaxNWts* establece los pesos para las observaciones dentro de las variables, lo que implica una solución al problema en tiempo de ejecución por el gran tamaño que tiene la matriz.

Resultados:

Los resultados bajo diferentes configuraciones son las siguientes:

Modelo	Iteraciones	Tolerancia absoluta	Tolerancia relativa	Tasa de clasificación
1	100	1.0e-4	1.0e-8	0.9032
2	200	1.0e-4	1.0e-8	0.9058
3	100	1.0e-8	1.0e-8	0.9032
4	200	1.0e-8	1.0e-8	0.9058
5	100	1.0e-8	1.0e-12	0.9032
6	100	1.0e-1	1.0e-1	0.7873
7	100	1.0e-2	1.0e-1	0.7873
8	100	1.0e-3	1.0e-1	0.7873
9	100	1.0e-4	1.0e-1	0.7873
10	100	1.0e-1	1.0e-2	0.845
11	100	1.0e-1	1.0e-3	**0.9328**
12	100	1.0e-1	1.0e-4	0.9305
13	100	1.0e-1	1.0e-5	0.9032
14	100	1.0e-2	1.0e-3	**0.9328**
15	100	1.0e-3	1.0e-3	**0.9328**
16	200	1.0e-4	1.0e-3	0.9058

[**Tabla 1.** Resultados Regresión logística.]

Según la tabla bajo diferentes criterios utilizados, el mejor modelo es aquel que que tiene 100 iteraciones, con una tolerancia relativa de 1.0e-3 (0.001), con una tolerancia absulta de tamaños inferiores a 1.0e-1 (0.1); observando la tabla de confusión para alguno de los 3 mejores modelos.

$Y_{original}/Y_{estimado}$	0	1	2	3	4	5	6	7	8
0	5733	1	22	13	14	41	38	10	44
1	1	6581	31	19	7	22	3	12	55
2	30	47	5453	90	56	19	55	57	129
3	19	18	121	5585	7	159	17	47	113
4	16	22	24	9	5499	5	50	14	35

5	60	23	45	159	51	4779	88	17	149
6	30	10	38	1	35	62	5712	5	23
7	16	22	60	30	45	6	3	5872	17
8	34	92	52	119	24	145	37	17	5274
9	27	22	10	66	128	31	2	135	43

[**Tabla 2.** Tabla de confusión del modelo 11.]

Árboles de clasificación o de decisión.

Introducción

"Los árboles de decisión son un método usado en distintas disciplinas como modelo de predicción. Estos son similares a diagramas de flujo, en los que llegamos a puntos en los que se toman decisiones de acuerdo a una regla." (***Rpubs***, *jboscomendoza*) "Son útiles para entender la estructura de un conjunto de datos. Sirven para resolver problemas tanto de clasificación (predecir una variable discreta, típicamente binaria) como de regresión (predecir una variable continua). Se trata de modelos excesivamente simples pero, y ahí reside fundamentalmente su interés, fácilmente interpretables." (***datanalytics***) Tienen "una representación visual de la manera en que se estructuran las decisiones que dan lugar a la clasificación. El árbol de clasificación lo componen una serie de nodos internos y externos así como arcos que unen los nodos. Los nodos externos se les conocen como hojas del árbol y se marcan con una clase o una distribución de probabilidad sobre las clases." (***DIEGO CALVO***)

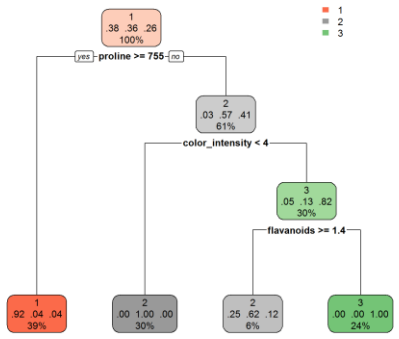


Imagen 15. *Ejemplo árbol de decisión.*

Metodología implementada

"Es importante saber que existen variadas implementaciones (librerías) de árboles de decisión en R como por ejemplo: *rpart*, *tree*, *party*, *ctree*, etc. Algunas se diferencias en las heurísticas (método para aumentar el conocimiento) utilizadas para el proceso de poda del árbol y otras manejan un componente probabilísto internamente." (***bookdown***)

Resultados:

Los resultados bajo diferentes configuraciones son las siguientes:

Modelo	Librería	Tasa de clasificación
1	rpart	0.6165
2	C5.0	**0.9696**
3	tree	0.2261

[**Tabla 3.** Resultados Árboles de clasificación.]

Según la tabla bajo diferentes criterios utilizados, el mejor modelo es aquel que proviene de la librería C5.0, posiblemente el error se debe enel código generado para predecir los valores; más sin embargo, la predicción hecha por la librería C5.0, es bastante buena, veamos la tabla de confusión

$Y_{original}/Y_{estimado}$	0	1	2	3	4	5	6	7	8
0	5844	2	9	12	8	12	17	1	15
1	0	6659	26	6	8	6	8	12	15
2	35	9	5772	23	29	16	24	18	27
3	12	15	54	5893	13	52	3	24	38
4	7	3	19	18	5649	17	28	10	24
5	30	9	10	74	14	5177	40	5	36
6	23	4	14	12	16	28	5807	0	12
7	17	7	18	23	23	14	4	6084	22
8	18	18	29	43	23	51	23	8	5607
9	14	5	11	31	77	32	4	52	44

[**Tabla 4.** Tabla de confusión del modelo 2.]

Bosques aleatorios.

Introducción

“El algoritmo de Random Forest surge como la agrupación de varios árboles de clasificación; básicamente selecciona de manera aleatoria una cantidad de variables con las cuales se construye cada uno de los árboles individuales, y se realizan predicciones con estas variables que posteriormente serán ponderadas a través del cálculo de la clase más votada de los árboles que se generaron, para finalmente hacer la predicción por Random Forest.” (***Universidad de Lima***)

“La idea esencial del bagging es promediar muchos modelos ruidosos pero aproximadamente imparciales, y por tanto reducir la variación. Los árboles son los candidatos ideales para el bagging, dado que ellos pueden registrar estructuras de interacción compleja en los datos, y si crecen suficientemente profundo, tienen

relativamente baja parcialidad. Producto de que los árboles son notoriamente ruidosos, ellos se benefician enormemente al promediar.

Cada árbol es construido usando el siguiente algoritmo:

1. Sea N el número de casos de prueba, M es el número de variables en el clasificador.

2. Sea m el número de variables de entrada a ser usado para determinar la decisión en un nodo dado; m debe ser mucho menor que M.

3. Elegir un conjunto de entrenamiento para este árbol y usar el resto de los casos de prueba para estimar el error.

4. Para cada nodo del árbol, elegir aleatoriamente m variables en las cuales basar la decisión. Calcular la mejor partición del conjunto de entrenamiento a partir de las m variables.

Para la predicción un nuevo caso es empujado hacia abajo por el árbol. Luego se le asigna la etiqueta del nodo terminal donde termina. Este proceso es iterado por todos los árboles en el ensamblado, y la etiqueta que obtenga la mayor cantidad de incidencias es reportada como la predicción." (***Wikipedia***)

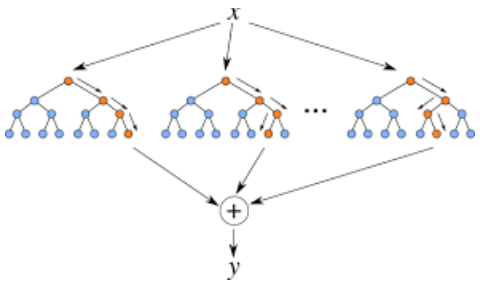


Imagen 16. *Ejemplo bosques aleatorios.*

"¿Qué es bagging? La agregación de bootstrap, también conocida como empaquetado, es un metaalgoritmo de aprendizaje automático diseñado para mejorar la estabilidad y precisión de algoritmos de aprendizaje automático usados en clasificación estadística y regresión. Además reduce la varianza y ayuda a evitar el sobreajuste. Aunque es usualmente aplicado a métodos de árboles de decisión, puede ser usado con cualquier tipo de método. El empaquetado es un caso especial del promediado de modelos. Dado un conjunto de entrenamiento estándar D de tamaño n, el empaquetado genera m nuevos conjuntos de entrenamiento $D_i D_i$, cada uno de tamaño n′, mediante muestreo uniforme y con reemplazo de D. En el caso del muestreo con reemplazo, algunas observaciones deben repetirse en $D_i D_i$. Si n′=n, entonces para un n grande el conjunto $D_i D_i$ se espera que tenga (1 - 1/e) (≈63.2%) ejemplos únicos de D, siendo el resto duplicados. Este tipo de muestra es conocido como muestra bootstrap. Los m modelos son aproximados usando las m muestras bootstrap y combinados promediando el resultado (para regresión) o votando (para clasificación)." (***Wikipedia***)

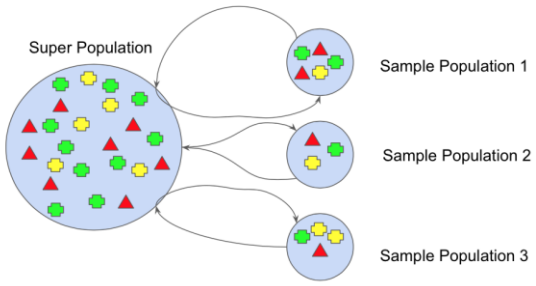


Imagen 17. *Idea sobre el bagging o boostrap.*

Metodología implementada

El paquete mejor implementado para bosques aleatorios en R es conocido como *randomForest* y tiene el mismo nombre para la función. Aunque la función contiene muchos parámetros de interés para un análisis más profundo, solo se tendrá en cuenta la cantidad de árboles para cada modelo para el bosque aleatorio, aunque se tratara de controlar ciertas situaciones para poder manter una buena replicabilidad a los resultados obtenidos. Por eso la función *set.seed()* proporciona una "semilla" es decir una cantidad de números aleatorios siempre que se llame el mismo número como parámetro de la función, y *ntree* el parámetro que usamos como configruación en el bosque aleatorio ya que significa la cantidad de árboles se usan para el bosque aleatorio.

Resultados:

Los resultados bajo diferentes configuraciones son las siguientes:

Modelo	Número de árboles	Tasa de clasificación
1	1	0.3042
2	5	0.7533
3	25	0.943
4	50	0.9593
5	100	0.9659
6	200	**0.9686**

[**Tabla 5.** Resultados de bosques aleatorios.]

Como podemos observar la tabla de las diferentes configuraciones que se usan según la cantidad de árboles, resulta ser lógico el hecho de que a mayor cantidad de árboles mejora su presición, pero la diferencia en tiempo de ejecución entre 50 árboles, 100 árboles y 200 árboles es bastante amplia, por lo que si se tiene un computador que no tiene el mejor procesador recomiendo usar el de 50 árboles.

Según la tabla bajo diferentes criterios utilizados,

$Y_{original}/Y_{estimado}$	0	1	2	3	4	5	6	7	8
0	5839	1	9	5	4	8	20	3	31
1	0	6649	34	15	13	1	6	10	9
2	23	9	5771	27	25	7	15	34	41
3	6	6	79	5837	1	70	3	44	57
4	9	9	12	0	5670	0	25	9	11
5	17	7	6	62	5	5214	41	9	38
6	25	9	4	0	8	34	5819	0	19
7	6	20	49	5	35	1	0	6066	13
8	13	30	34	44	17	36	26	6	5591
9	20	11	10	72	69	20	3	41	41

[**Tabla 6.** Tabla de confusión del modelo 6.]

Máquinas de soporte vectorial.

Introducción

"Una máquina de vectores de soporte (SVM) es un algoritmo de aprendizaje supervisado que se puede emplear para clasificación binaria o regresión. Las máquinas de vectores de soporte son muy populares en aplicaciones como el procesamiento del lenguaje natural, el habla, el reconocimiento de imágenes y la visión artificial.

Una máquina de vectores de soporte construye un hiperplano óptimo en forma de superficie de decisión, de modo que el margen de separación entre las dos clases en los datos se amplía al máximo. Los vectores de soporte hacen referencia a un pequeño subconjunto de las observaciones de entrenamiento que se utilizan como soporte para la ubicación óptima de la superficie de decisión.

Las máquinas de vectores de soporte pertenecen a una clase de algoritmos de Machine Learning denominados métodos kernel y también se conocen como máquinas kernel." (***MathWorks***)

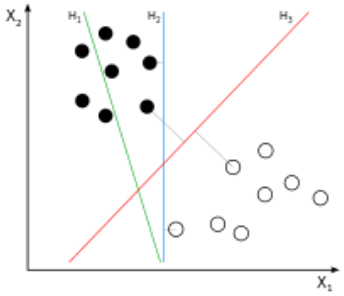


Imagen 18. *Ejemplo máquina de soporte vectorial.*

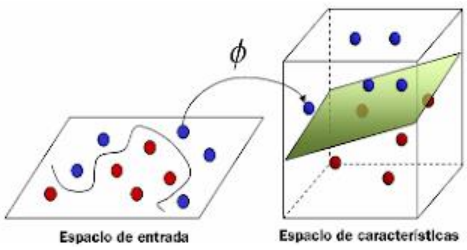


Imagen 19. *Ejemplo hiperplano de un svm.*

Metodología implementada

Para métodos usados de svm son de la librería *e1071*, contiene la función *svm* para métodos con máquina de soporte vectorial, estos procesos son de los más lentos junto a bosques aleatorios para árboles mayores o iguales a 200; la función svm tiene muchos parámetros para poder jugar con diferentes configuraciones, pero al ser lentos no es fácil obtener muchos modelos diferentes, por lo que la idea principal para ver diferentes máquinas de soporte vectorial es en el parámetro del *kernel* de la función svm puesto que dicho parámetro es el principal en el análisis del svm, contiene cuatro tipos de kernels principales para un modelo que son:

1. "lineal": $u'v$
2. "polynomial": $(\gamma u'v + coef0)^{degree}$
3. "radial basis": $e^{-\gamma|u-v|^2}$
4. "sigmoid": $tanh(\gamma u'v + coef0)$

Resultados:

Los resultados bajo diferentes configuraciones son las siguientes:

Modelo	Kernel	Tasa de clasificación
1	"linear"	**0.9709**
2	"polynomial"	0.56385
3	"radial basis"	0.9431
4	"sigmoid"	0.9292

[**Tabla 7.** Resultados máquina de soporte vectorial.]

Según la tabla bajo diferentes criterios utilizados, se obtiene un mejor modelo a partir de un kernel lineal, pero posiblemente se encuentre un error debido a que en el momento de realizar la prueba bajo un kernel polinmial el computador se utilizaba para otras actividades igual de pesadas que podrían causar un mal análisis en el código de R para el modelo.

$Y_{original}/Y_{estimado}$	0	1	2	3	4	5	6	7	8

0	5898	0	2	0	0	8	3	0	11
1	1	6708	11	2	0	0	0	3	15
2	11	15	5762	35	34	18	17	16	41
3	5	5	85	5825	0	114	2	15	48
4	4	4	9	0	5720	2	4	7	1
5	12	5	11	92	6	5204	32	2	41
6	8	1	11	1	5	26	5861	0	5
7	2	7	26	5	35	5	0	6073	6
8	13	47	38	64	8	93	14	4	5540
9	11	16	2	20	103	21	0	97	15

[**Tabla 8.** Tabla de confusión del modelo 1.]

Redes neuronales.

Introducción

"Las redes neuronales son modelos simples del funcionamiento del sistema nervioso. Las unidades básicas son las neuronas, que generalmente se organizan en **capas**.

Una red neuronal es un modelo simplificado que emula el modo en que el cerebro humano procesa la información: Funciona simultaneando un número elevado de unidades de procesamiento interconectadas que parecen versiones abstractas de neuronas.

Las unidades de procesamiento se organizan en capas. Hay tres partes normalmente en una red neuronal : una capa de entrada, con unidades que representan los campos de entrada; una o varias capas ocultas; y una capa de salida, con una unidad o unidades que representa el campo o los campos de destino. Las unidades se conectan con fuerzas de conexión variables (o ponderaciones). Los datos de entrada se presentan en la primera capa, y los valores se propagan desde cada neurona hasta cada neurona de la capa siguiente. al final, se envía un resultado desde la capa de salida.

La red aprende examinando los registros individuales, generando una predicción para cada registro y realizando ajustes a las ponderaciones cuando realiza una predicción incorrecta. Este proceso se repite muchas veces y la red sigue mejorando sus predicciones hasta haber alcanzado uno o varios criterios de parada.

Al principio, todas las ponderaciones son aleatorias y las respuestas que resultan de la red son, posiblemente, disparatadas. La red aprende a través del entrenamiento. Continuamente se presentan a la red ejemplos para los que se conoce el resultado, y las respuestas que proporciona se comparan con los resultados conocidos. La información procedente de esta comparación se pasa hacia atrás a través de la red, cambiando las ponderaciones gradualmente. A medida que progresa el entrenamiento, la red se va haciendo cada vez más precisa en la replicación de

resultados conocidos. Una vez entrenada, la red se puede aplicar a casos futuros en los que se desconoce el resultado." (***IBM***)

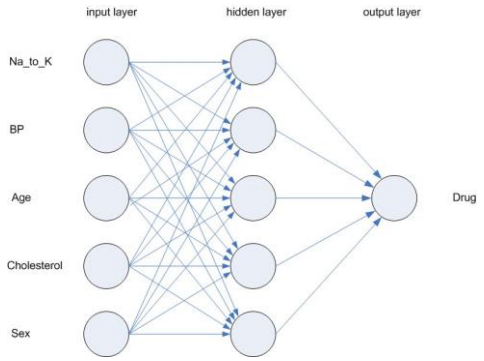


Imagen 20. *Ejemplo de una red neuronal.*

Metodología implementada

El método a simple vista parece ser complicada la primera vez en que se observa el código, pero, con una breve explicación de las funciones y parámetros, se entiende completamente y es fácil de volver a replicar, entonces:

La función de la red neuronal viene de *mxnet*, está diseñado completamente para resolver problemas con redes neuronal, como se menciona en la introducción como mínimo deben existir 3 capas en una red neuronal, en el código viene siendo como capa de entrada "data", como capas ocultas "fc" y un número asignado para darle un orden y la capa de salida como "softmax", las funciones mencionadas se debe a que van a estar completamente conectadas como se observa en la imagen y el tipo de activación que en este caso es "relu" (unidad lineal rectificada) "$f(x) = max(0, x)$, (…) con una aproximación suave del rectificador es una función analítica $f(x) = ln(1 + e^x)$" (***Wikipedia***). Lo interante surge para la configuración que se va a utilizar son con la cantidad de nodos, y la cantidad de capas a utilizar; el nodo va a tener como resultados $nodos = 2^x$, pero en la tabla solo se observara el valor de la x, es decir, si en la tabla aparece el número de 5, es porque se usaron $2^5 = 32$ nodos para cada capa oculta.

Resultados:

Los resultados bajo diferentes configuraciones son las siguientes:

Modelo	nodos	capas	Tasa de clasificación
1	1	4	0.9180
2	1	5	0.9180
3	1	6	0.9180
4	1	7	0.9180
5	2	4	0.9427
6	2	5	0.9610

7	2	6	0.9745
8	2	7	**0.9807**
9	3	4	0.9484
10	3	5	0.9662
11	3	6	0.9730
12	3	7	0.9730
13	4	4	0.9484
14	4	5	0.9662
15	4	6	0.9730
16	4	7	0.9730
17	5	4	0.9512
18	5	5	0.9605
19	5	6	0.9740
20	5	7	0.9788

[**Tabla 9.** Resultados red neuronal.]

Según la tabla bajo diferentes criterios utilizados, el mejor modelo es aquel que tiene 2 capas con $2^7 = 128$ nodos, con una presición de 98.07%, un valor casi exacto, más o menos de cada 100 cien cartas a lo sumo se equivoca en 2.

$Y_{original}/Y_{estimado}$	0	1	2	3	4	5	6	7	8
0	972	0	1	1	1	1	1	1	2
1	0	1127	1	1	0	1	2	1	2
2	2	1	1011	2	3	0	2	6	4
3	0	0	2	990	0	4	0	2	4
4	1	1	3	0	964	0	1	1	2
5	3	0	0	6	2	873	3	1	3
6	2	2	2	1	7	5	938	0	1
7	2	3	9	2	0	0	0	1005	1
8	3	0	5	3	5	2	2	3	947
9	1	2	0	7	7	6	1	4	1

[**Tabla 10.** Tabla de confusión del modelo 8.]

Conclusiones finales.

Sacando la tabla de los mejores modelos según su clasificación, podemos ver los siguiente:

Arquitectura para el modelo

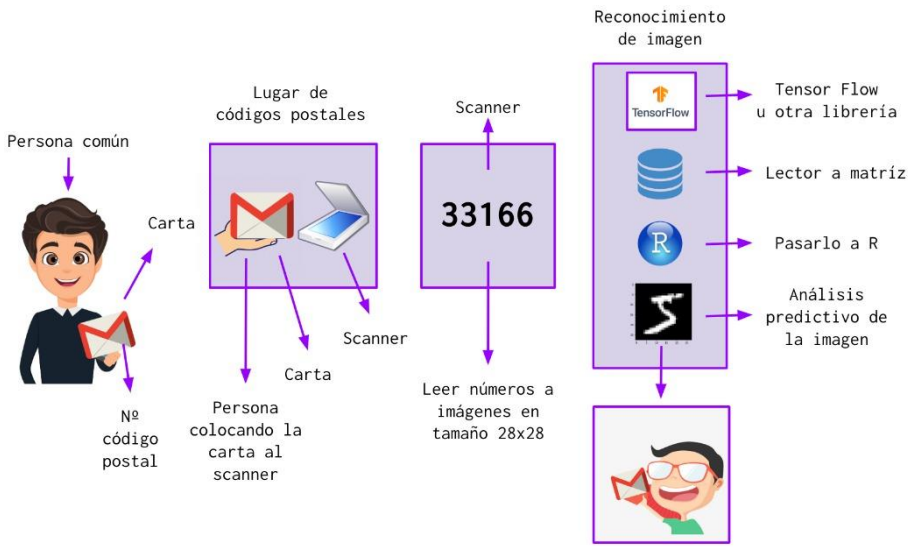


Imagen 21. *Arquitectura.*

El sistema de clasificación que se propone para la identificación de números de códigos postales, se basa desde la entrada misma de las imagenes hasta una clasificación automatizada.

El sistema consiste en:

1. El almacenamiento para cada número del código postal sea dentro de una franja delimitada.
2. La cara de los números queden frente a un detector de imagen.
3. Realizar la captura de la imagen a los diferentes campos donde se deben encontrar los números del código postal.
4. La conversión de la imagen a un tamaño de 28x28.
5. Utilizar aplicaciones que permitan hacer uso de un reconocimiento de imagen.
6. Decodificar las imagenes a una matriz o base de datos.
7. Mandar la base de datos a un software encargado a hacer análisis estadístico.
8. Al obtener los resultados, el software debe devolver los resultados a la base de datos que requiera los resultados, para así poder enviar la carta.

Repositorio para los códigos.

https://github.com/WilliamAR/Analisis_mnist.git

Referencias.

*__Base de datos MNIST__ http://yann.lecun.com/exdb/mnist/

Información de la base MNIST https://en.wikipedia.org/wiki/MNIST_database

Información sobre Machine Learning https://cleverdata.io/que-es-machine-learning-big-data/ https://www.sas.com/es_co/insights/analytics/machine-learning.html

Información de aprendizaje supervisado
https://es.mathworks.com/discovery/supervised-learning.html

Información acerca de entrenamiento, validación y prueba
https://analisisydecision.es/entrenamiento-validacion-y-test/

Información para tasa de clasificación
https://www.alceingenieria.net/bioestadistica/clasifica.pdf

Regresión logística https://rpubs.com/Joaquin_AR/229736

Árboles de decisión
https://rpubs.com/jboscomendoza/arboles_decision_clasificacion
https://www.datanalytics.com/libro_r/arboles-de-decision.html
http://www.diegocalvo.es/arboles-de-clasificacion-en-r/
https://bookdown.org/content/2031/arboles-de-decision-parte-ii.html

Bosques aleatorios
https://es.wikipedia.org/wiki/Random_forest#Definición_de_Random_forests
https://medium.com/coriers/how-to-develop-a-robust-algorithm-c38e08f32201

Boostrap https://es.wikipedia.org/wiki/Agregaci%C3%B3n_de_bootstrap

Máquinas de soporte vectorial https://es.mathworks.com/discovery/support-vector-machine.html http://compuinteligencia.blogspot.com/2009/06/maquinas-de-soporte-vectorial.html
https://es.wikipedia.org/wiki/M%C3%A1quinas_de_vectores_de_soporte

Redes neuronales
https://www.ibm.com/support/knowledgecenter/es/SS3RA7_sub/modeler_mainhelp_client_ddita/components/neuralnet/neuralnet_model.html

Adicionales (Relu) https://es.wikipedia.org/wiki/Rectificador_(redes_neuronales)