Aprendizaje de Máquina: Proyecto final

Fajardo Oroz Angel Farid
Givois Mendez Timothee Rene
Vilchis Domínguez Miguel Alonso

December 12, 2016

1 Introducción

El aprendizaje de máquina es una rama de la inteligencia artificial, la cual tiene como objetivo generar algoritmos que sean capaces de hacer que las computadoras aprendar ciertos patrones o características de los datos, sin la necesidad de especificar dichos patrones de forma explícita a la máquina, es decir, estos algoritmos de aprendizaje de máquina deben ser capaces de extraer ciertas características de los datos de manera autónoma e independiente.

De manera general, los algoritmos de aprendizaje de màquina suelen ser clasificados de dos formas: algoritmos de aprendizaje supervisado y algoritmos de aprendizaje no supervisado. Los algoritmos de aprendizaje superivsado son aquellos en donde el conjunto de datos de entrenamiento están clasificados previamente, es decir, tienen etiquetas que son correspondientes a determinadas funciones de las variables de entrada. Estos algoritmos son utilizados en problemas de regresión y clasificación. Por otra parte, los algoritmos de aprendizaje no supervisado tienen como finalidad descubrir la estructura o las relaciones entre los datos, por ejemplo, al generar *clusters*. Lo anterior se calcula a partir de la información que se puede extraer de los datos, sin necesidad de que éstos sean previamente etiquetados.

En este trabajo se utilizaron algoritmos de aprendizaje supervisado debido a que se necesita resolver un problema de clasificación y se tiene una base de datos que ha sido previamente etiquetada.

2 Algunos algoritmos de aprendizaje supervisado

2.1 Regresión logística

La regresión logística es un tipo de aprendizaje supervisado, la cual es utilizada para resolver problemas de clasificación. La característica de este tipo de regresión es que se describe por una función logística, la cual acota la salida de esta función a [0,1] de tal manera que el resultado se puede interpretar como una probabilidad de pertenencia a una categoría.

$$f(x) = \frac{1}{1 + exp^{-X\Theta}} \tag{1}$$

De la ecuación anterior la x representa a los datos de entrada, y las θ representan los pesos que ponderan a las entradas.

2.2 Redes neuronales

Las redes neuronales, también conocidas como redes de perceptrones, son un arreglo de elementos que en conjunto son capaces de aproximar cualquier función lineal o no lineal. Estos elementos se activan dependiendo de alguna función elegida, misma que puede ser lineal, logística, tangente hiperbólica, ReLu, etc.

Básicamente este algoritmo se basa en generar una arquitectura feedforward (las salidas de los elementos son las entradas de los elementos de capas de "neuronas" siguientes); y el error entre la predicción del algoritmo y el valor real objetivo, se calcula con base a un descenso de gradiente, y dicho error se propaga hacia las otras capas de neuronas para actualizar los pesos de sus entradas (*backpropagation*).

2.3 K vecinos cercanos

Esta técnica de aprendizaje supervisado se basa en clasificar o predecir un valor (regresión), dependiendo de un número k de vecinos cercanos.

Si es un problema de clasificación, la etiqueta de un elemento x se asigna dependiendo del mayor nùmero de etiquetas de los k vecinos cercanos. Por otro lado, si es un problema de regresión, el valor numérico que corresponde a dicho elemento en cuestión es igual al promedio del valor de los k vecinos cercanos.

2.4 Árboles de decisión

Los árboles de clasificación y regresión (CART) son un método utilizado para aproximar funciones o para etiquetar. Una característica muy importante de los árboles de decisión es que son fáciles de interpretar, porque su funcionamiento se basa fundamentalmente es tomar una decisión de acuerdo a un cierto umbral entre las variables.

La selección de los nodos es mediante la función de pureza. Las hojas son aquellos nodos que tienen solamente un tipo de clase. Los algoritmos CART utilizan diferentes funciones para determinar la pureza de cada variable. La función de pureza más utilizada es la de entropía.

La selección de atributos se desarrolla por medio de una función de ganancia, ecuación 2. Sea S la entropía general y S_A es la entropía de la variable A. El subconjunto seleccionado es el que tiene una menor entropía. La variable seleccionada es la que sólo pertenece a un a clase a partir de en un umbral.

$$G_{entropía}(S, A) = S - S_A \tag{2}$$

Otra medida de pureza es conocida como el coeficiente de Gini y está basado en un criterio de impureza que mide la divergencia de acuerdo a la distribución de probabilidad. Se define de acuerdo a la ecuación 3, donde p_i corresponde a la probabilidad del conjunto de clase i de S.

$$Gini(S) = 1 - \sum_{i_1}^{k} p_i^2 \tag{3}$$

Se considera que un conjunto es puro si cumple con $p_i = 1$ tal que $1 - 1^2 = 0$. La variable seleccionada es de acuerdo a la función de ganancia 4.

$$G_{Gini}(A, S) = Gini(S) - Gini(A, S) = Gini(S) - \sum_{i=1}^{n} \frac{|S_i|}{S} Gini(S_i) \tag{4}$$

2.5 Bosque aleatorio

El bosque aleatorio (RF) es un método ensamblado que contienen un conjunto de árboles. Este toma el promedio ponderado de la clasificación para determinar el resultado. Los árboles del RF se forman al seleccionar un subconjunto de variables y observaciones mediante el método de *Bagging*.

El término *Bagging* proviene de un muestreo con reemplazo conocido como *Bootstrap aggregating*. Este método reduce la varianza, ya que cada subconjunto es evaluado por un árbol.
Dado un conjunto de entrenamiento D de tamaño n, el *Bagging* genera m subconjuntos de entrenamiento de tamaño D_i con tamaño n'. En el caso de usar *Bagging* con reemplazo algunas observaciones son repetidas en el conjunto D_i. Si $n' = n$ se cumple, entonces para una n grande el conjunto D_i deberá tener al menos $(1 - 1/e)$ ejemplos únicos de D, siendo el resto duplicados.

La forma de implementar un RF a través de *Bagging* queda expresado de la siguiente manera: dado un conjunto de entrenamiento $X = x_1, \ldots, x_n$ y un conjunto de la función objetivo $Y = y_1, \ldots, y_n$. Se realiza muestreo aleatorio con reemplazo y cada muestra le corresponde a un árbol. El muestreo es para cada subconjunto de $b = 1, \ldots, B$.

1. La muestra con reemplazo de n entrena los elementos de X para y y es conocido como X_b y Y_b.

2. El entrenamiento del árbol (f_b) es realizado con X_b, Y_b

Una vez entrenados los árboles con cada una de las muestras, la predicción del RF es el promedio de la clasificación de cada árbol:

$$\hat{f} = \frac{1}{B}\sum_{b=1}^{B} \hat{f}_b(x') \tag{5}$$

donde B es el conjunto *boostrap* y f_b es un conjunto de entrenamiento de un árbol para realizar la clasificación.

3 Resultados obtenidos

Para el problema de clasificar a los clientes de Walmart, utilizamos un set de datos que tiene 7 variables:

1. Trip Type.- Representa el tipo de comprador (categoría del cliente)
2. Visit number.-Es un número único que representa a un sólo viaje de un solo cliente.
3. Week day.- Día de la semana del viaje.
4. UPG.- Código de barras del producto comprado.
5. Scan Count.- Número de artículos adquiridos (números negativos representan devolver productos)
6. Department Description.- Descripción de alto nivel del departamento del artículo.
7. Timeline Number.- Categoría más refinada de los productos.

De las variables anteriores, sólo tomamos en consideración el *weekday*, el *Scan count* y *Department description*. Con base a éstas variables, se implementaron los siguientes algoritmos (Python y R).

Algoritmo	**Accuracy**	Tiempo de entrenamiento	Parametros
AdaBoost Classifier	0.3	2 min	Estimadores=100
Bagging Classifier	0.348	5 min	Estimadores=50
DNN Classifier	0.33	1 hora	Hidden Layers= 100, 1000, 100
Gradient Boosting Classifier	0.35	30 min	
KNN Classifier	0.27	10 min	10 vecinos
MLP Classifier	0.28	15 min	learning rate = 1e-5 y un lfbgs para optimizar
Voting Classifier	0.348	12 min	Bagging, Random Forest y AdaBoost con pesos de 2, 2, 1 para los votos
Random Forest C5.0	0.3514	45 min	numero=10, repitedcv, 1 iteración

De la tabla anterior se aprecia que el modelo que alcanzó una mayor presición fue el Random Forest (C5.0) con un 35%.

4 Interpretación de resultados y conclusiones

Los modelos implementados mostraron una baja presición. Probablemente esto fue porque:

1. La calidad de datos que se encuentran en el set de entrenamiento y prueba no son muy descriptivos, o no se relacionan directamente con el tipo de etiquetas de los clientes.

2. Los modelos que prometían muy buenos resultados, como xgboost o SVD, n o han terminado de ejecutarse...

Por otra parte, el modelo de Random Forest ofrece resultados muy buenos. Un motivo de ello es el hecho que es un modelo de ensamble (de varios árboles de decisión).

"Garbage in, garbage out", es decir, cuando la calidad de los datos no es buena, es muy difícil que un modelo de aprendizaje de máquina ofrezca resultados buenos, por tal motivo, es necesario aplicar en estos casos técnicas de *feature engineering* con el objetivo de obtener variables valiosas para un modelo.