Implementación de algoritmos de Machine Learning utilizando Scala

Gutierrez Luna Yuridia Nayeli
Mosqueda Espinoza Adamari Antonia
Instituto Tecnológico de Tijuana
Tijuana, Baja California
adamari.mosqueda16@tectijuana.edu.mx
yuridia.gutierrez16@tectijuana.edu.mx

***Abstract*— Los algoritmos de Machine Learning permiten que una serie de operaciones funcionen de forma automática, incluso realizan predicciones para facilitar la toma de decisiones. En este proyecto se hizo la implementación de 4 algoritmos para realizar una comparación.**

I. INTRODUCCIÓN

A lo largo del semestre hemos visto muchos temas interesantes, claro que cada uno separado y con datos diferentes pero ahora es tiempo de poner a prueba alguno de los modelos que vimos. Utilizando una misma BD vamos a utilizar los modelos de SVM, Logistic Regression, Decision Treey y Multilayer Perceptron para ver la diferencia entre estos y dar una conclusion de cual de estos modelos es el mejor (para nosotros) según el rendimiento, el porcentaje de efectividad así como el tiempo de respuesta de cada uno de esto.

II. MARCO TEÓRICO

El aprendizaje automático (ML) es un subconjunto de la inteligencia artificial, que construye un modelo matemático basado en datos de muestra, conocido como "datos de entrenamiento", con el fin de hacer predicciones o decisiones sin estar programado explícitamente para realizar la tarea.

"Se dice que un programa de computadora aprende de la experiencia E con respecto a alguna clase de tareas T y medida de desempeño P si su desempeño en las tareas de T, medido por P, mejora con la experiencia E."

-Mitchell T.M

En el aprendizaje automático, las redes neuronales, las máquinas de vectores de soporte y la computación evolutiva, generalmente se nos proporciona un conjunto de entrenamiento y un conjunto de prueba. Por conjunto de entrenamiento, significará la unión del conjunto etiquetado y el conjunto de ejemplos sin etiquetar disponibles para los aprendices de máquina. En comparación, el conjunto de prueba consta de ejemplos nunca antes vistos.

A. Support Vector Machine

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.

El entrenamiento de una máquina de vectores de soporte consta de dos fases:

1) Transformar los predictores (datos de entrada) en un espacio de características altamente dimensional. En esta fase es suficiente con especificar el kernel; los datos nunca se transforman explícitamente al espacio de características. Este proceso se conoce comúnmente como el truco kernel.

2) Resolver un problema de optimización cuadrática que se ajuste a un hiperplano óptimo para clasificar las características transformadas en dos clases. El número de características transformadas está determinado por el número de vectores de soporte.

B. Decision Tree

Un Árbol de Decisión (o Árboles de Decisiones) es un método analítico que a través de una representación esquemática de las alternativas disponible facilita la toma de mejores decisiones, especialmente cuando existen riesgos, costos, beneficios y múltiples opciones. El nombre se deriva de la apariencia del modelo parecido a un árbol y su uso es amplio en el ámbito de la toma de decisiones bajo incertidumbre (Teoría de Decisiones) junto a otras herramientas como el Análisis del Punto de Equilibrio. Los árboles de decisión son especialmente útiles cuando:

- Las alternativas o cursos de acción están bien definidas (por ejemplo: aceptar o rechazar una propuesta, aumentar o no la capacidad de producción, construir o no una nueva bodega, etc.)
- Las incertidumbres pueden ser cuantificadas (por ejemplo: probabilidad de éxito de una campaña publicitaria, probable efecto en ventas, probabilidad de pasar de etapas, etc.)
- Los objetivos están claros (por ejemplo: aumentar las ventas, maximizar utilidades, minimizar costos, etc.)

C. Logistic Regression

La regresión logística es un método popular para predecir una respuesta categórica. Es un caso especial de modelos lineales generalizados que predice la probabilidad de los resultados. En spark.ml, la regresión logística se puede utilizar para predecir un resultado binario mediante la regresión logística binomial, o se puede utilizar para predecir un resultado multiclase mediante la regresión logística multinomial. Utilice el parámetro de familia para seleccionar entre estos dos algoritmos, o déjelo sin configurar y Spark deducirá la variante correcta.

La regresión logística multinomial se puede utilizar para la clasificación binaria estableciendo el parámetro de familia en "multinomial". Producirá dos conjuntos de coeficientes y dos intersecciones.

Al ajustar Logistic Regression Model sin interceptar en el conjunto de datos con una columna constante distinta de cero, Spark MLlib genera coeficientes cero para columnas constantes distintas de cero. Este comportamiento es el mismo que R glmnet pero diferente de LIBSVM.

D. Multilayer Perceptron Classifier

El clasificador de perceptrones multicapa (MLPC) es un clasificador basado en la red neuronal artificial feedforward. MLPC consta de múltiples capas de nodos. Cada capa está completamente conectada a la siguiente capa de la red. Los nodos de la capa de entrada representan los datos de entrada. Todos los demás nodos asignan entradas a salidas mediante una combinación lineal de las entradas con los pesos w del nodo y el sesgo b, aplicando una función de activación. Esto se puede escribir en forma de matriz para MLPC con capas K + 1 de la siguiente manera:

$y(x) = fK(...f2(wT2f1(wT1x + b1) + b2) ... + bK)$

Los nodos en las capas intermedias utilizan la función sigmoidea (logística):

$f(zi) = 11 + e - zi$

Los nodos en la capa de salida usan la función softmax:

$f(zi) = ezi \sum Nk = 1ezk$

El número de nodos N en la capa de salida corresponde al número de clases. MLPC emplea retropropagación para aprender el modelo. Usamos la función de pérdida logística para la optimización y L-BFGS como rutina de optimización.

III. IMPLEMENTACIÓN

Para este proyecto usamos Spark porque tiene muchos beneficios:

Velocidad: Spark puede ser 100 veces más rápido que Hadoop para el procesamiento de datos a gran escala al explotar la computación en memoria y otras optimizaciones.

Facilidad de uso: Opera en grandes conjuntos de datos, esto incluye una colección de más de 100 operadores para transformar datos.

Un motor unificado: Viene empaquetado con bibliotecas que incluyen soporte para consultas SQL, transmisión de datos, aprendizaje automático y procesamiento de gráficos.

Spark provee API para Python, Java y Scala, nosotras elegimos Scala porque es un lenguaje funcional que permite implementar el paradigma MapReduce de manera sencilla y rápida. Scala trabaja sobre la JVM, lo que nos permite disponer de las múltiples librerías creadas para Java.

IV.RESULTADOS

A. Support Vector Machine

Las primeras 5 pruebas arrojaron los mismos resultados, se hicieron en diferentes días en la misma computadora, las otras 5 pruebas se hicieron en una computadora distinta, los resultados fueron los mismos.

TABLA 1
Resultados obtenidos de SVM

	Coeficientes	Intercept
1	[-2.125897501491213E-6,-0.013517727458849872,7.514021888017163E-4,2.7022337506408964E-4,0.011177544540215354]	-1.084924165339881
2	[-2.125897501491213E-6,-0.013517727458849872,7.514021888017163E-4,2.7022337506408964E-4,0.011177544540215354]	-1.084924165339881
3	[-2.125897501491213E-6,-0.013517727458849872,7.514021888017163E-4,2.7022337506408964E-4,0.011177544540215354]	-1.084924165339881
4	[-2.125897501491213E-6,-0.013517727458849872,7.514021888017163E-4,2.7022337506408964E-4,0.011177544540215354]	-1.084924165339881
5	[-2.125897501491213E-6,-0.013517727458849872,7.514021888017163E-4,2.7022337506408964E-4,0.011177544540215354]	-1.084924165339881
6	[-2.125897501491213E-6,-0.013517727458849872,7.514021888017163E-4,2.7022337506408964E-4,0.011177544540215354]	-1.084924165339881
7	[-2.125897501491213E-6,-0.013517727458849872,7.514021888017163E-4,2.7022337506408964E-4,0.011177544540215354]	-1.084924165339881
8	[-2.125897501491213E-6,-0.013517727458849872,7.514021888017163E-4,2.7022337506408964E-4,0.011177544540215354]	-1.084924165339881
9	[-2.125897501491213E-6,-0.013517727458849872,7.514021888017163E-4,2.7022337506408964E-4,0.011177544540215354]	-1.084924165339881
10	[-2.125897501491213E-6,-0.013517727458849872,7.514021888017163E-4,2.7022337506408964E-4,0.011177544540215354]	-1.084924165339881

B. Decision Tree

Al igual que SVM se realizaron 5 iteraciones en una computadora y el resto en otra, en las primeras 5 pruebas la probabilidad de exactitud fue de 0.8907834778774071 (89.1%) mientras el error fue de 0.06896551724137934 (6.89%), y estos fueron los resultados de los árboles. En las otras 5 pruebas la probabilidad de exactitud fue de 0.8933757133569313 (89.34%) mientras el error fue de 0.10662428664306867 (10.66%)

C. Logistic Regression

Al realizar las 5 interacciones de este modelo nos arrojó los mismos resultados, lo único en lo que cambia es dependiendo del equipo que se utiliza para correr el modelo como podemos observar en una computadora da diferentes resultados a la otra.

TABLA 2
Resultados obtenidos de Logistic Regression

	Coeficientes	Intercept
1	[2.1953717210865443E-5,-0.0039087241148942735,0.0020207318126466336,0.0013950274211932889,0.0427408662344112 7]	-2.706584067945768
2	[2.1953717210865443E-5,-0.0039087241148942735,0.0020207318126466336,0.0013950274211932889,0.0427408662344112 7]	-2.706584067945768
3	[2.1953717210865443E-5,-0.0039087241148942735,0.0020207318126466336,0.0013950274211932889,0.0427408662344112 7]	-2.706584067945768
4	[2.1953717210865443E-5,-0.0039087241148942735,0.0020207318126466336,0.0013950274211932889,0.0427408662344112 7]	-2.706584067945768
5	[2.1953717210865443E-5,-0.0039087241148942735,0.0020207318126466336,0.0013950274211932889,0.0427408662344112 7]	-2.706584067945768
6	[2.1295060967543102E-5,-0.0032248638711286446,0.002004245563577638,0.0014137466827612378,0.0371939132948556]	-2.695580402935833
7	[2.1295060967543102E-5,-0.0032248638711286446,0.002004245563577638,0.0014137466827612378,0.0371939132948556]	-2.695580402935833
8	[2.1295060967543102E-5,-0.0032248638711286446,0.002004245563577638,0.0014137466827612378,0.0371939132948556]	-2.695580402935833
9	[2.1295060967543102E-5,-0.0032248638711286446,0.002004245563577638,0.0014137466827612378,0.0371939132948556]	-2.695580402935833
10	[2.1295060967543102E-5,-0.0032248638711286446,0.002004245563577638,0.0014137466827612378,0.0371939132948556]	-2.695580402935833

Teniendo dos porcentajes de efectividad en la predicción entre 88.49 % y 88.87 %.

D. Multilayer Perceptron Classifier.

Este solo nos arrojó el porcentaje de la exactitud, durante las 5 primeras pruebas la respuesta fue de 0.8835474819081377 (88.35%) mientras que en las otras pruebas fue de 0.8829225352112676 (88.29%)

V. CONCLUSIONES

Como hemos visto a lo largo del curso hay diferentes modelos para la predicción de datos, claro que cada uno tiene sus ventajas y desventajas, todo depende de los datos, lo que se requiera y el grado de exactitud que desees en las predicciones, así como podemos ver que también depende del equipo en el cuar realices el montado del modelo por que entre PC puede cambiar el porcentaje de efectividad. En el caso de nuestro team el modelo con mayor efectividad es el "Decision Tree" con un porcentaje muy poco mayor a los demás modelos por lo que este sería el adecuado para utilizar por nuestro team en cualquier implementación con nuestros PC.

REFERENCIAS

[1] GEO Tutoriales. (2016). Árbol de Decisión (Qué es y para qué sirve). 2020, de Gestión de Operaciones Sitio web: https://www.gestiondeoperaciones.net/procesos/arbol-de-decision/

[2] Ilabaca, S. (----). Apache Spark. Enero, 2021, de Analytics Sitio web: https://www.analytics10.com/que-es-apache-spark/

[3] MathWorks (----) Máquina de vectores de soporte (SVM). Recuperado de https://la.mathworks.com/discovery/support-vector-machine.html

[4] Spark. (----). Logistic regression. 2020, de Spark Sitio web: https://spark.apache.org/docs/2.4.7/ml-classification-regression.html#logistic-regression

[5] Zhang, X. D. (2020). Machine learning. In A Matrix Algebra Approach to Artificial Intelligence (pp. 223-440). Springer, Singapore.