Aplicación y Comparación de Clasificadores usando un dataset de sentimientos de Keras.

***Resumen* – Con este proyecto se pretende usar tres diferentes clasificadores usando el mismo dataset Keras de 25 000 reseñas de películas clasificadas por sentimientos negativos y positivos.**

Palabras Claves – clasificadores; movie review data set; SCIKit – learn; IMDB; Keras; matriz de confusión; métodos de evaluación; árbol de decisiones; algoritmos de ML (Naive Bayes; SVM y kNN).

I. INTRODUCCIÓN

Una de las tareas de la Minería de Datos es la **clasificación o la discriminación de datos**; asociándolos en parejas o en grupos predefinidos; aprendizaje supervisado. Esta tarea es encontrar modelos (funciones) que describen y distinguen clases o conceptos para futuras predicciones.

La aplicación de este concepto la podemos encontrar en y para la calificación de un crédito bancario, reconocimiento de imágenes y patrones, diagnósticos médicos, detección de fallos en aplicaciones industriales clasificar tendencias de mercados financieros,…

Algunos de los métodos utilizados son los siguientes:
- ✓ Análisis discriminante.
- ✓ Arboles de decisiones.
- ✓ Reglas de clasificación.
- ✓ Redes neuronales.

A. Tipos de Clasificadores.

1) Naive Bayes.

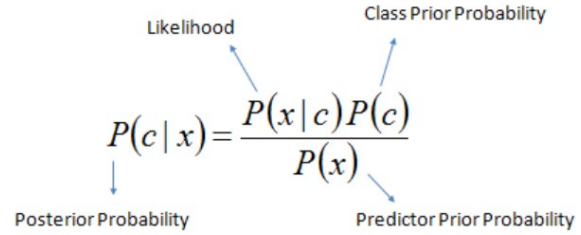
Es una técnica de clasificación basada en el Teorema de Bayes con un supuesto de independencia entre los predictores. En términos simples, un clasificador Naive Bayes asume que la presencia de una característica particular en una clase no está relacionada con la presencia de cualquier otra característica.

Por ejemplo, una fruta puede considerarse una manzana si es roja, redonda y de aproximadamente 3 pulgadas de diámetro. Incluso si estas características dependen unas de otras o de la existencia de las otras características, un Clasificador Naive Bayes consideraría que todas estas propiedades contribuyen independientemente a la probabilidad de que esta fruta sea una manzana.

El modelo Ingenuo Bayesiano es fácil de construir y particularmente útil para conjuntos de datos muy grandes. Junto con la simplicidad, se sabe que Naive Bayes supera incluso a los métodos de clasificación altamente sofisticados.

El teorema de Bayes proporciona una forma de calcular la probabilidad posterior P (c|x) a partir de P (c), P (x) y P (x| c).

De acuerdo con la siguiente ecuación:



$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Fig. 1 Teorema de Bayes.

Donde:

- P (c|x) es la probabilidad posterior de la clase (objetivo) dado el predictor (atributo).
- P (c) es la probabilidad previa de la clase.
- P (x|c) es la probabilidad del predictor de una clase determinada.
- P (x) es la probabilidad previa del predictor.

2) kNN (k- Nearest Neighbors).

Puede ser usado tanto para problemas de clasificación como de regresión. Sin embargo, es más ampliamente utilizado en problemas de clasificación en la industria. kNN es un algoritmo simple que almacena todos los casos disponibles y clasifica los nuevos casos por mayoría de votos de la variable k. El caso que se asigna a la clase es más común entre la variable k más cercana medida por una función de distancia.

Estas funciones de distancia pueden ser Euclidiana, Manhattan, Minkowski y Hamming. Las primeras tres funciones se utilizan para la función continua y la cuarta (Hamming) para las variables categóricas. Si k = 1, entonces el caso se asigna simplemente a la clase de k más cercana. A veces, elegir k resulta ser un desafío al realizar el modelado kNN.

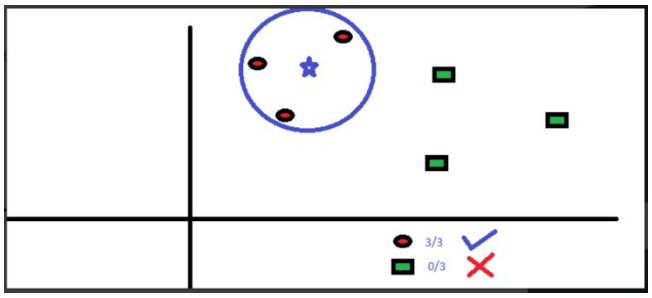


Fig. 2 Clasificador de kNN.

kNN puede aplicarse fácilmente a nuestras vidas reales. Si desea obtener información sobre una persona de la que no tiene información, ¡le gustaría conocer a sus amigos cercanos y los círculos que frecuenta y obtener acceso a su información!

Cosas a considerar antes de seleccionar kNN:

- kNN es computacionalmente caro.
- Las variables deben normalizarse, de lo contrario, las variables de rango superior pueden sesgarla.
- Funciona en la etapa de pre-procesamiento, antes de optar por kNN como valor atípico, eliminación de ruido.

3) SVM (Support Vector Machine)

En este método de clasificación, se traza cada elemento de datos como un punto en el espacio n-dimensional (donde n es el número de entidades que tiene) con el valor de cada entidad como el valor de una coordenada en particular.

Por ejemplo, si solo tuviéramos dos características como la altura y la longitud del cabello de un individuo, primero dibujaríamos estas dos variables en un espacio bidimensional donde cada punto tiene dos coordenadas (estas coordenadas se conocen como vectores de soporte)

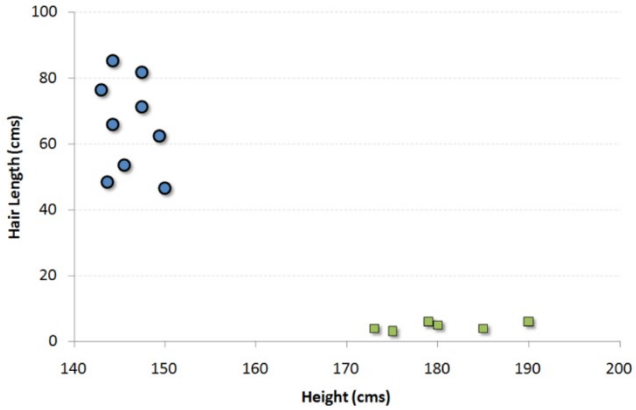


Fig. 3 Ejemplo Clasificador SVM.

Ahora, encontraremos una línea que divide los datos entre los dos grupos de datos clasificados de manera diferente. Esta será la línea tal que las distancias desde el punto más cercano en cada uno de los dos grupos estarán más alejadas.

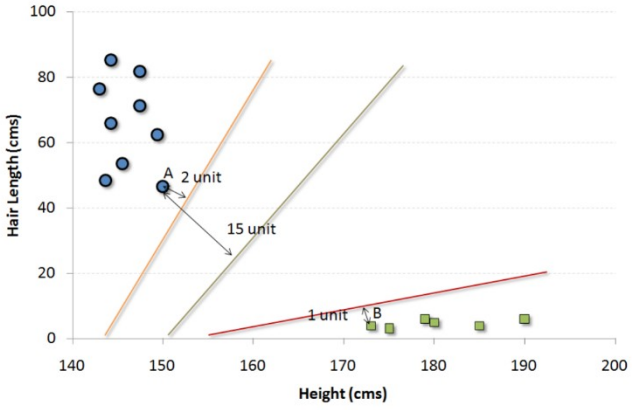


Fig. 4 Ejemplo Clasificador SVM.

En el ejemplo que se muestra arriba, la línea que divide los datos en dos grupos clasificados de manera diferente es la línea negra, ya que los dos puntos más cercanos son los más alejados de la línea. Esta línea es nuestro clasificador. Luego, dependiendo de dónde caigan los datos de prueba a ambos lados de la línea, esa es la clase en la que podemos clasificar los nuevos datos.

4) Decision tree

La motivación de este clasificador radica en el contexto de que se tienen varias clases de datos y dentro de esas clases existen subclases, por lo tanto, se puede seguir un camino para la selección de la clase de datos, que esta a su vez se relaciona con los datos que fueron utilizados al momento de entrenar el clasificador.

Diciéndolo de otro modo el árbol de clasificación, repetitivamente divide el área de búsqueda de los datos, en subpartes identificadas por un delimitador.

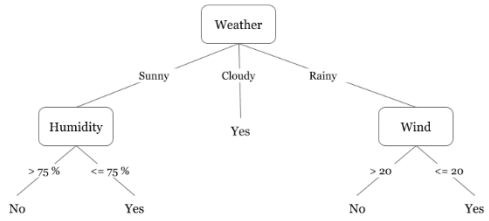


Fig. 5 Ejemplo árbol de decisión sobre el clima.

Aunque en estas subclases divididas por los delimitadores podemos contener impurezas, lo cual significa que datos que pertenecen a una subclase se encuentran también en otra. En este caso nos puede ayudar la entropía que mide la aleatoriedad de los elementos en las clases, lo cual ayuda a determinar el grado de impureza de los datos.

$$H = -\sum p(x) \log p(x)$$

Fig. 6 Entropía.

5) Random Forest

Este algoritmo de clasificación esta basado en los arboles de decisión. El algoritmo de bosque aleatorio genera un conjunto de arboles de decisión a los cuales entrena con un conjunto de datos aleatorios, una vez que cada árbol tiene la clasificación de las clases y subclases, se procede a realizar una votación para la decisión final de las clases. Esto ayuda a solventar el problema de sobre alimentación que pueden llegar a tener los arboles de clasificación.

Para la votación de los árboles de selección se puede dar mayor peso a las clases de los árboles con menor porcentaje de error y a su vez darles menor peso a los árboles con mayor porcentaje de error.

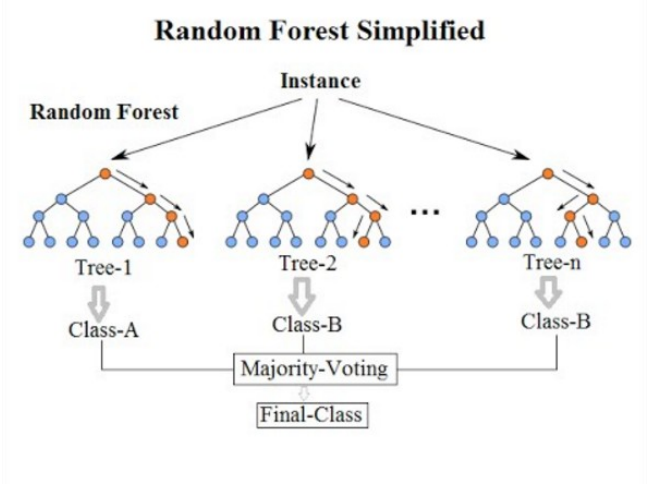


Fig. 7 Ejemplo simple de un bosque aleatorio.

Se puede comparar este algoritmo con el de vecinos cercanos dado que ambos usan un esquema en el cual se la da un peso a una entidad (vecino u árbol). Y en base a eso los algoritmos realizan sus predicciones.

II. IMPLEMENTACION

Para realizar este proyecto se consideraron los siguientes requerimientos:

A. Requerimientos.

1) Dataset de 25000 reseñas de películas (IMDB), etiquetas por sentimientos (positivos/negativos).

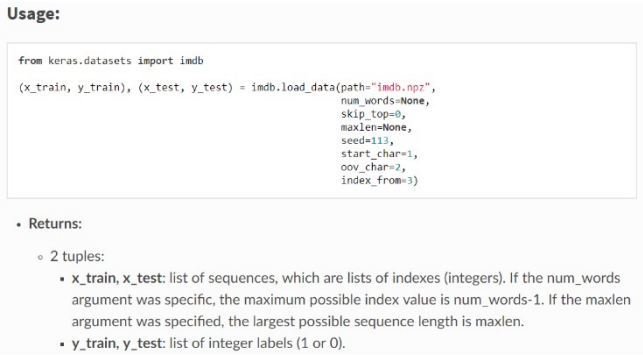


Fig. 8 Dataset Keras (IMDB).

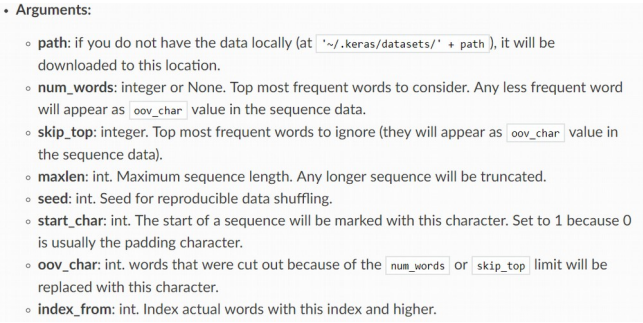


Fig. 9 Dataset Keras (IMDB).

2) Usar mínimo tres diferentes clasificadores, como; Naive Bayes, kNN (k- Nearest Neighbors) y SVM (Support Vector Machine).

classifier = NewClassifier()
classifier.fit(datax, datay)
classifier.predict(newdata)

3) Librerías:

numpy
matplotlib
scikit-learn

III. RESULTADOS

IV. CONCLUSIONES

V. REFERENCIAS

[1] Keras: The Python Deep Learning library, IMDB Movie reviews sentiment classification, https://keras.io/datasets/#imdb-movie-reviews-sentiment-classification

[2] Essentials of Machine Learning Algorithms (with Python and R Codes), https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/

[3] Machine Learning algorithms, Decision Tree, https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/ml-decision-tree/tutorial/

[4] Random Forest, https://en.wikipedia.org/wiki/Random_forest