Practica 2 - Ejercicio 1 - Clasificadores

- Ignacio David Vázquez Pérez
- 218292866

- (2022B) Sem. de Sol. de Problemas de Inteligencia Artificial II
- D05 (2022B)

Realizar una investigación sobre los siguientes métodos de clasificación en aprendizaje maquina:
• Regresión logística (Logistic Regression)
• K-Vecinos Cercanos (K-Nearest Neighbors)
• Maquinas Vector Soporte (Support Vector Machines)
• Naive Bayes

Debes identificar el funcionamiento de cada método y para que tipo de dataset es comúnmente usado. Entregar la investigación en un archivo PDF.

Introducción

Elegir un algoritmo de clasificación apropiado para una tarea de problema particular requiere experiencia; cada algoritmo tiene sus propias peculiaridades y se basa en ciertas suposiciones. Para reafirmar el teorema de David H. Wolpert, ningún clasificador único funciona mejor en todos los escenarios posibles (The Lack of A Priori Distinctions Between Learning Algorithms, Wolpert and David H, Neural Computation (1996).

En la práctica, siempre se recomienda comparar el rendimiento de distintos algoritmos de aprendizaje para seleccionar el mejor modelo para el problema en particular; estos pueden diferir en la cantidad de características o muestras, la cantidad de ruido en un conjunto de datos y si las clases son lineal mente separables o no.

El rendimiento de un clasificador (rendimiento computacional y poder predictivo) depende en gran medida de los datos subyacentes que están disponibles para el aprendizaje. Los cinco pasos principales que intervienen en el entrenamiento de un algoritmo de aprendizaje automático se pueden resumir de la siguiente manera:

1. Selección de funciones y recopilación de muestras de entrenamiento.
2. Elegir una métrica de rendimiento.
3. Elección de un clasificador y algoritmo de optimización.
4. Evaluar el desempeño del modelo.
5. Ajuste del algoritmo.

Regresión logística (Logistic Regression)

La regresión logística es un modelo de clasificación que es muy fácil de implementar pero funciona muy bien en clases linealmente separables.

La razón de posibilidades se puede escribir como `p / (1 - p)` donde p representa la probabilidad del evento positivo. El término evento positivo no necesariamente significa bueno, sino que se refiere al evento que queremos predecir, por ejemplo, la probabilidad de que un paciente tenga una determinada enfermedad; podemos pensar en el evento positivo como etiqueta de clase y = 1 . Luego podemos definir aún más la función logit, que es simplemente el logaritmo de la razón de probabilidades (logaritmo de probabilidades): `logit ( p ) = log p/( 1 − p )`.

La función logit toma valores de entrada en el rango de 0 a 1 y los transforma en valores en todo el rango de números reales, que podemos usar para expresar una relación lineal entre los valores de las características y las probabilidades logarítmicas:

$$logit\left(p\left(y=1|\boldsymbol{x}\right)\right)=w_0x_0+w_1x_1+\cdots+w_mx_m=\sum_{i=0}^{m}w_ix_i=\boldsymbol{w}^T\boldsymbol{x}$$

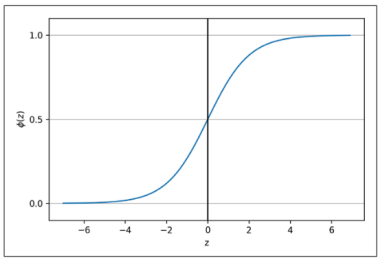
p(y=1|x) es la probabilidad condicional de que una determinada muestra pertenezca a la clase 1 dadas sus características x.

Es predecir la probabilidad de que cierta muestra que pertenezca a una clase en particular, que es la forma inversa de la función logit. También se denomina función logística sigmoidea, a veces simplemente abreviada como función sigmoidea debido a su característica forma de S:

$$\phi\left(z\right)=\frac{1}{1+e^{-z}}$$

Aquí z es la entrada neta, la combinación lineal de pesos y características de la muestra,

$$z=\boldsymbol{w}^T\boldsymbol{x}=w_0x_0+w_1x_1+\cdots+w_mx_m\ .$$



esta función sigmoidea toma valores de números reales como entrada y los transforma en valores en el rango [0, 1] con una intersección en φ (0) = 0.5 .

En la regresión logística, esta función de activación se convierte en la función sigmoidea.

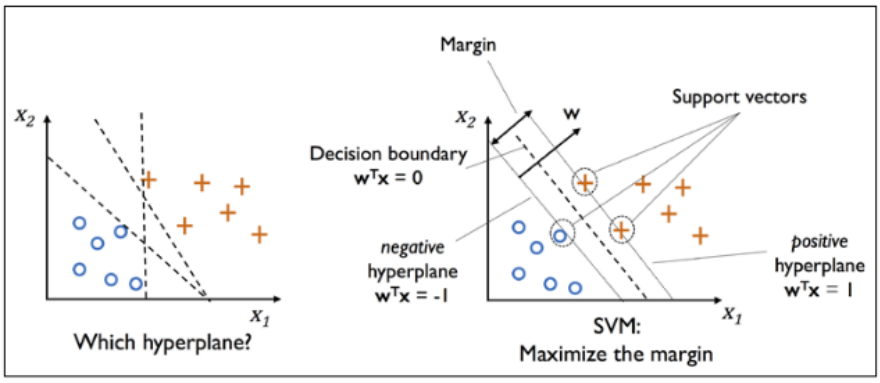
La salida de la función sigmoidea se interpreta entonces como la probabilidad de que una muestra particular pertenezca a la clase 1, `φ ( z ) = P ( y = 1| x ; w)`, dadas sus características **x** parametrizadas por los pesos **w**. La probabilidad pronosticada se puede convertir simplemente en un resultado binario a través de una función de umbral:

$$\hat{y} = \begin{cases} 1 & if\ z \geq 0.0 \\ 0 & otherwise \end{cases}$$

De hecho, hay muchas aplicaciones en las que no solo estamos interesados en las etiquetas de clase predichas, sino en las que la estimación de la probabilidad de pertenencia a la clase es particularmente útil (la salida de la función sigmoidea antes de aplicar la función de umbral). La regresión logística se usa en el pronóstico del tiempo, por ejemplo, no solo para predecir si lloverá en un día en particular, sino también para informar la probabilidad de lluvia.

Máquina de vectores de soporte (SVM)

Puede considerarse una extensión del perceptrón. Usando el algoritmo de perceptrón, minimizamos los errores de clasificación. Sin embargo, en SVM nuestro objetivo de optimización es maximizar el margen. El margen se define como la distancia entre el hiperplano de separación (límite de decisión) y las muestras de entrenamiento más próximas a este hiperplano, que son los denominados vectores soporte.



Intuición de margen máximo

La razón detrás de tener límites de decisión con márgenes grandes es que tienden a tener un error de generalización más bajo, mientras que los modelos con márgenes pequeños son más propensos al sobreajuste. La maximización del margen, aquellos hiperplanos positivos y negativos que son paralelos a la frontera de decisión, se pueden expresar de la siguiente manera:

$$\frac{\boldsymbol{w}^T\left(\boldsymbol{x}_{pos}-\boldsymbol{x}_{neg}\right)}{\|\boldsymbol{w}\|}=\frac{2}{\|\boldsymbol{w}\|}$$

El lado izquierdo de la ecuación anterior se puede interpretar como la distancia entre el hiperplano positivo y negativo, que es el llamado margen que queremos maximizar. Ahora, la función objetivo de la SVM se convierte en la maximización de este margen al maximizar `2/||w||` bajo la restricción de que las muestras se clasifiquen correctamente, lo que se puede escribir como:

$$w_0+\boldsymbol{w}^T\boldsymbol{x}^{(i)}\geq 1 \; if \; y^{(i)}=1$$

$$w_0+\boldsymbol{w}^T\boldsymbol{x}^{(i)}\leq -1 \; if \; y^{(i)}=-1$$

$$\text{for } i=1\ldots N$$

Aquí, `N` es el número de muestras en nuestro conjunto de datos. Estas dos ecuaciones básicamente dicen que todas las muestras negativas deben caer en un lado del hiperplano negativo, mientras que todas las muestras positivas deben caer detrás del hiperplano positivo, que también se puede escribir de manera más compacta de la siguiente manera:

$$y^{(i)}\left(w_0+\boldsymbol{w}^T\boldsymbol{x}^{(i)}\right)\geq 1 \; \forall_i$$

La motivación para introducir la variable de holgura ξ fue que las restricciones lineales deben relajarse para los datos separables de forma no lineal para permitir la convergencia de la optimización en presencia de clasificaciones erróneas, bajo la penalización de costo adecuada. La variable de holgura de valores positivos simplemente se agrega a las restricciones lineales:

$$w_0+\boldsymbol{w}^T\boldsymbol{x}^{(i)}\geq 1-\xi^{(i)} \quad if \quad y^{(i)}=1$$

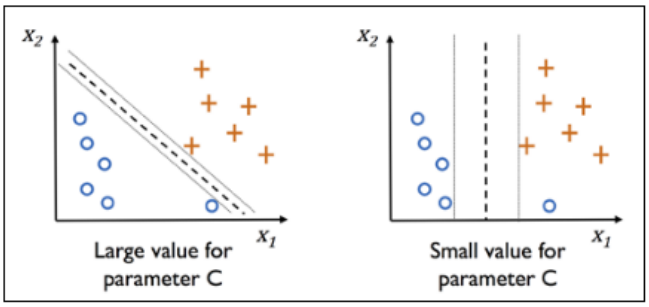
$$w_0+\boldsymbol{w}^T\boldsymbol{x}^{(i)}\leq -1+\xi^{(i)} \quad if \quad y^{(i)}=-1$$

$$\text{for } i=1\ldots N$$

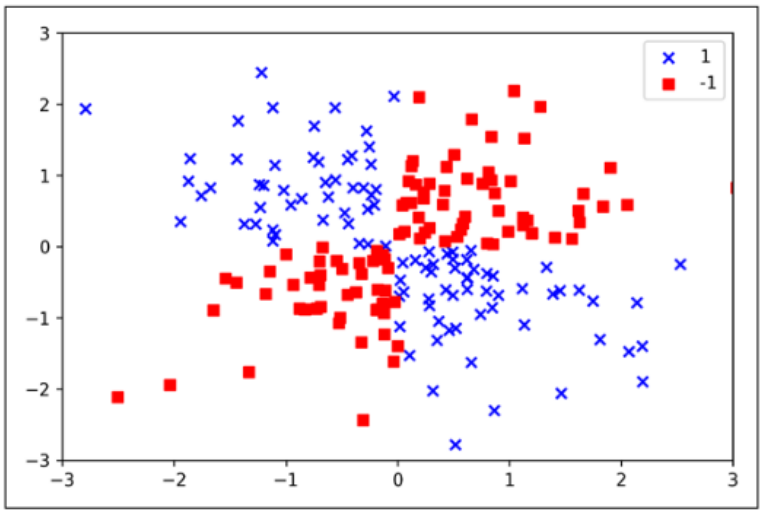
Aquí, N es el número de muestras en nuestro conjunto de datos. Entonces, el nuevo objetivo a minimizar (sujeto a las restricciones) se convierte en:

$$\frac{1}{2}\|\boldsymbol{w}\|^2 + C\left(\sum_i \xi^{(i)}\right)$$

A través de la variable C, podemos controlar la penalización por clasificación errónea. Los valores grandes de C corresponden a grandes penalizaciones de error, mientras que somos menos estrictos con los errores de clasificación si elegimos valores más pequeños para C . Luego podemos usar el parámetro C para controlar el ancho del margen y, por lo tanto, ajustar la compensación de sesgo-varianza.



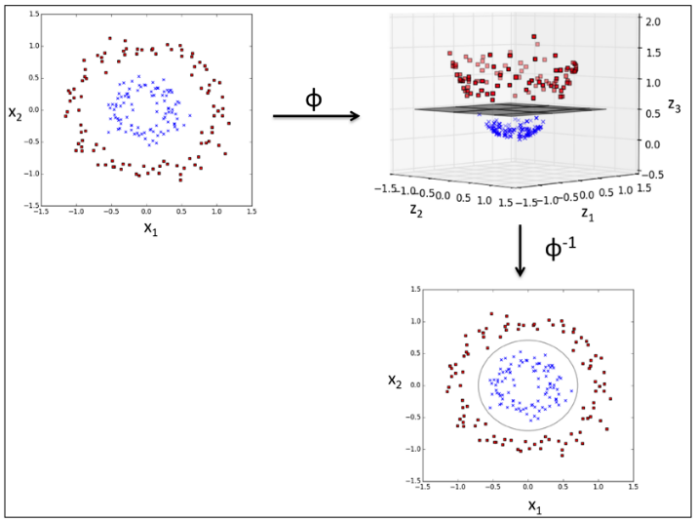
No podríamos separar muy bien las muestras de la clase positiva y negativa utilizando un hiperplano lineal como límite de decisión a través de la regresión logística lineal o el modelo SVM lineal.



La idea básica detrás de los métodos del kernel para manejar estos datos linealmente inseparables es crear combinaciones no lineales de las características originales para proyectarlas en un espacio de mayor dimensión a través de una función de mapeo φ donde se vuelve linealmente separable.

$$\phi\left(x_1, x_2\right) = \left(z_1, z_2, z_3\right) = \left(x_1, x_2, x_1^2 + x_2^2\right)$$

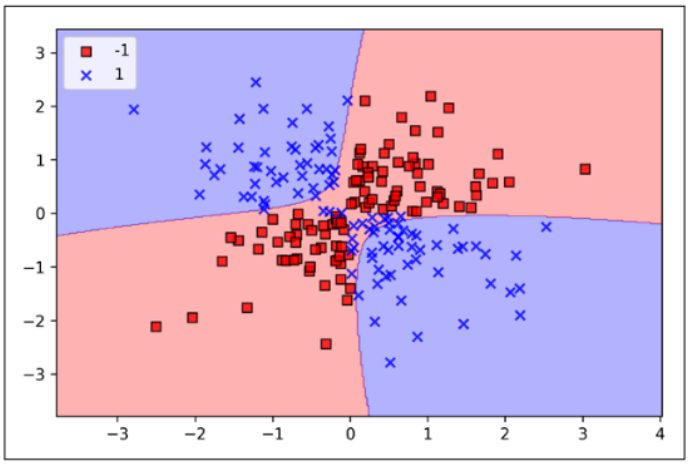
Esto nos permite separar las dos clases que se muestran en el gráfico a través de un hiperplano lineal que se convierte en un límite de decisión no lineal si lo proyectamos de nuevo en el espacio de características original:



Para resolver un problema no lineal usando una SVM, transformaríamos los datos de entrenamiento en un espacio de características de mayor dimensión a través de una función de mapeo φ y entrenaríamos un modelo SVM lineal para clasificar los datos en este nuevo espacio de características. Luego, podemos usar la misma función de mapeo φ para transformar datos nuevos e invisibles para clasificarlos usando el modelo SVM lineal.

El término kernel puede interpretarse como una función de similitud entre un par de muestras. El signo menos invierte la medida de la distancia en una puntuación de similitud y, debido al término exponencial, la puntuación de similitud resultante caerá en un rango entre 1 (para muestras exactamente similares) y 0 (para muestras muy diferentes).

Como podemos ver en el gráfico resultante, el núcleo SVM separa los datos XOR relativamente bien:



K-nearest neighbors

KNN es un ejemplo típico de un estudiante perezoso. Se llama perezoso no por su aparente simplicidad, sino porque no aprende una función discriminatoria de los datos de entrenamiento, sino que memoriza el conjunto de datos de entrenamiento.

Modelos paramétricos y no paramétricos

Los algoritmos de aprendizaje automático se pueden agrupar en modelos paramétricos y no paramétricos. Usando modelos paramétricos, estimamos parámetros del conjunto de datos de entrenamiento para aprender una función que puede clasificar nuevos puntos de datos sin requerir más el conjunto de datos de entrenamiento original.

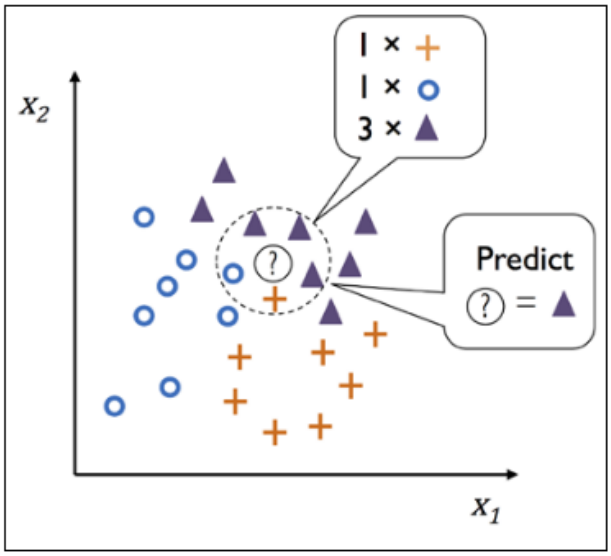
KNN pertenece a una subcategoría de modelos no paramétricos que se describe como aprendizaje basado en instancias. Los modelos basados en el aprendizaje basado en instancias se caracterizan por memorizar el conjunto de datos de entrenamiento, y el aprendizaje perezoso es un caso especial de aprendizaje basado en instancias que se asocia con ningún costo (cero) durante el proceso de aprendizaje.

El algoritmo KNN

Se puede resumir en los siguientes pasos:

1. Elija el número de k y una métrica de distancia.
2. Encuentra los k-vecinos más cercanos de la muestra que queremos clasificar.
3. Asignar la etiqueta de clase por mayoría de votos.

La siguiente figura ilustra cómo se asigna la etiqueta de clase de triángulo a un nuevo punto de datos (?) en función de la votación mayoritaria entre sus cinco vecinos más cercanos.



En función de la métrica de distancia elegida, el algoritmo KNN encuentra las k muestras en el conjunto de datos de entrenamiento que son más cercanas (más similares) al punto que queremos clasificar. La etiqueta de clase del nuevo punto de datos se determina luego por mayoría de votos entre sus k vecinos más cercanos.

La principal ventaja de este enfoque basado en la memoria es que el clasificador se adapta inmediatamente a medida que recopilamos nuevos datos de entrenamiento. Sin embargo, la desventaja es que la complejidad computacional para clasificar nuevas muestras crece linealmente con la cantidad de muestras en el conjunto de datos de entrenamiento en el peor de los casos, a menos que el conjunto de datos tenga muy pocas dimensiones (características) y el algoritmo se haya

implementado utilizandoestructuras de datos eficientes, tales como árboles-KD. La elección correcta de k es crucial para encontrar un buen equilibrio entre overfitting y underfitting. También debemos asegurarnos de elegir una métrica de distancia que sea adecuada para las características del conjunto de datos.

Es importante mencionar que KNN es muy susceptible al sobreajuste debido a la maldición de la dimensionalidad. La maldición de la dimensionalidad describe el fenómeno en el que el espacio de características se vuelve cada vez más escaso para un número creciente de dimensiones de un conjunto de datos de entrenamiento de tamaño fijo. Intuitivamente, podemos pensar que incluso los vecinos más cercanos están demasiado lejos en un espacio de alta dimensión para dar una buena estimación.

Clasificación ingenua de Bayes

Los clasificadores Naive Bayes son clasificadores lineales que son conocidos por ser simples pero muy eficientes. El modelo probabilístico de los clasificadores de Bayes ingenuos se basa en el teorema de Bayes, y el adjetivo ingenuo proviene de la suposición de que las características de un conjunto de datos son mutuamente independientes. En la práctica, la suposición de independencia a menudo se viola, pero los clasificadores ingenuos de Bayes todavía tienden a funcionar muy bien bajo esta suposición poco realista. Especialmente para tamaños de muestra pequeños, los clasificadores bayesianos ingenuos pueden superar a las alternativas más potentes.

El modelo de probabilidad que se formuló por Thomas Bayes (1701-1761) es bastante simple pero poderoso; se puede escribir en palabras simples de la siguiente manera:

```
probabilidad posterior = probabilidad condicional · probabilidad previa / evidencia
```

La probabilidad posterior, en el contexto de un problema de clasificación, se puede interpretar como: "¿Cuál es la probabilidad de que un objeto en particular pertenezca a la clase i dados los valores de sus características observadas?"

Un problema no lineal (B) sería un caso en el que los clasificadores lineales, como el ingenuo Bayes, no serían adecuados ya que las clases no son linealmente separables. En tal escenario, se deben preferir clasificadores no lineales (por ejemplo, clasificadores de vecinos más cercanos basados en instancias).

La notación general de la probabilidad posterior se puede escribir como:

$$P(\omega_j \mid \mathbf{x}_i) = \frac{P(\mathbf{x}_i \mid \omega_j) \cdot P(\omega_j)}{P(\mathbf{x}_i)}$$

- x_i sea el vector de características de la muestra i, $i \in \{1, 2, ..., n\}$,
- ω_j sea la notación de clase j, $j \in \{1, 2, ..., m\}$,

- y $P(x_i \mid \omega_j)$ la probabilidad de observar la muestra x_i dado que pertenece a la clase ω_j.

La función objetivo en la probabilidad ingenua de Bayes es maximizar la probabilidad posterior dados los datos de entrenamiento para formular la regla de decisión.

$$\text{predicted class label} \leftarrow \underset{j=1\ldots,m}{\arg\max}\, P(\omega_j \mid \mathbf{x}_i)$$

Una suposición adicional de los clasificadores de Bayes ingenuos es la independencia condicional de las características. Bajo esta suposición ingenua, las probabilidades condicionales de clase o (posibilidades) de las muestras se pueden estimar directamente a partir de los datos de entrenamiento en lugar de evaluar todas las posibilidades de **x**. Por lo tanto, dado un vector de características d-dimensional **x**, la probabilidad condicional de clase se puede calcular de la siguiente manera:

$$P(\mathbf{x} \mid \omega_j) = P(x_1 \mid \omega_j) \cdot P(x_2 \mid \omega_j) \cdot \ldots \cdot P(x_d \mid \omega_j) = \prod_{k=1}^{d} P(x_k \mid \omega_j)$$

Aquí, `P(x | ω<sub>j</sub>)` simplemente significa: "¿Qué tan probable es observar este patrón particular x dado que pertenece a la clase ωj?" Las probabilidades "individuales" para cada característica en el vector de características se pueden estimar a través de la estimación de máxima verosimilitud, que es simplemente una frecuencia en el caso de datos categóricos:

$$\hat{P}(x_i \mid \omega_j) = \frac{N_{x_i,\omega_j}}{N_{\omega_j}} \quad (i = (1, \ldots, d))$$

- $N_{xi,\omega j}$: Número de veces que la característica x_i aparece en muestras de la clase ωj .
- $N_{\omega j}$: Conteo total de todas las entidades en la clase ωj .

En la práctica, el supuesto de independencia condicional se viola a menudo, pero se sabe que los clasificadores de Bayes ingenuos funcionan bien en esos casos.

Al ser un "aprendiz entusiasta", se sabe que los clasificadores ingenuos de Bayes son relativamente rápidos en la clasificación de nuevas instancias. Los "estudiantes ansiosos" están aprendiendo algoritmos que aprenden un modelo de un conjunto de datos de entrenamiento tan pronto como los datos estén disponibles. Una vez que se aprende el modelo, no es necesario volver a evaluar los datos de entrenamiento para hacer una nueva predicción. En el caso de "estudiantes entusiastas", el paso computacionalmente más costoso es el paso de construcción del modelo, mientras que la clasificación de nuevas instancias es relativamente rápida.

Sin embargo, los "aprendices perezosos" memorizan y reevalúan el conjunto de datos de entrenamiento para predecir la etiqueta de clase de las nuevas instancias. La ventaja del aprendizaje

perezoso es que la fase de construcción del modelo (entrenamiento) es relativamente rápida. Por otro lado, la predicción real suele ser más lenta en comparación con los estudiantes ansiosos debido a la reevaluación de los datos de entrenamiento. Otra desventaja de los estudiantes perezosos es que los datos de entrenamiento deben conservarse, lo que también puede ser costoso en términos de espacio de almacenamiento.

Referencias

- Raschka, S. (2017). *Python Machine Learning, Second edition: Machine learning and deep learning with python, scikit-learn, and tensorflow*. Packt Publishing.
- Raschka, S. (2014). *Naive Bayes and Text Classification I Introduction and Theory*. https://arxiv.org/pdf/1410.5329v3.pdf