Título del trabajo (elegir uno original)

Hugo Borrego Angulo
Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla
Sevilla, España
Correos electrónicos UVUS y de contacto (si distinto)

Rafael Duque Colete
Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla
Sevilla, España
Correos electrónicos UVUS y de contacto (si distinto)

***Abstract*—En este trabajo se presenta una propuesta para la selección de características, basada en la utilización de dos algoritmos de búsqueda: *búsqueda secuencial hacia atrás* y *búsqueda secuencial hacia atrás mixta*. Estos algoritmos se han aplicado a dos conjuntos de datos referentes a: *titanic* y *breast cancer*.**

Una vez obtenidos los resultados y con ello las características seleccionadas, se han aplicado 2 algoritmos de entrenamiento: *DecisionTreeClassifier* y *Naive Bayes*. Con los que podemos evaluar las soluciones que nos proporcionan los algoritmos de búsqueda anteriormente mencionados.

***Index Terms*—Inteligencia Artificial, otras palabras clave...**

I. Introducción

En la actualidad, la selección de características es una técnica fundamental en el análisis de datos y el aprendizaje automático utilizado en el campo de la Inteligencia Artificial. La selección de características nos ayuda en el preprocesamiento de datos, ya que ayuda a reducir la dimensionalidad de los datos, mejorar la precisión de los modelos y facilitar la interpretación de los resultados.

Una vez que los datos han sido adecuadamente preprocesados, la selección de características nos ayudará a mejorar la eficiencia y la interpretabilidad de los modelos. Los algoritmos de búsqueda secuencial hacia atrás y búsqueda secuencial hacia atrás mixta son técnicas para la selección de estas características. Estos algoritmos eliminan iterativamente las características menos relevantes, con el objetivo de identificar un subconjunto óptimo de características que maximicen el rendimiento predictivo.

En este trabajo, se aplica un enfoque sistemático para analizar y comparar el rendimiento de dos modelos de clasificación. El Decision Tree Classifier y el Naive Bayes, en los conjuntos de datos del Titanic y Breast Cancer. Se evaluará cómo la selección de características mediante los algoritmos mencionados influye en el rendimiento de estos modelos.

Este trabajo tiene como objetivo principal explorar y demostrar la efectividad de diversas técnicas de preprocesamiento y selección de características en la mejora del rendimiento de modelos de clasificación. Los resultados obtenidos proporcionarán una comprensión más profunda de las mejores prácticas para el manejo y análisis de datos en contextos similares y destacarán la importancia de un preprocesamiento adecuado y una selección de características rigurosa en la construcción de modelos predictivos robustos.

Fig. 1. Ejemplo de un pie de figura. Imagen con derechos Creative Commons

II. Preliminares

Para comprender el trabajo realizado, es necesario conocer los conceptos básicos de la selección de características y los algoritmos, que ha sido utilizados en este trabajo.

A. Métodos empleados

Tratamiento de datos

- Normalización: Se ha aplicado la normalización a los datos que era necesario, con el objetivo de que todas las variables tengan la misma escala.
- Codificación de variables categóricas: Se ha aplicado la codificación de variables categóricas a los datos que era necesario, con el objetivo de que todas las variables sean numéricas.

Algoritmos de selección de características

- Búsqueda secuencial hacia atrás: Este algoritmo elimina iterativamente las características menos relevantes, con el objetivo de identificar un subconjunto óptimo de características que maximicen el rendimiento predictivo.
- Búsqueda secuencial hacia atrás mixta: Este algoritmo elimina iterativamente las características menos relevantes, y añade las características más relevantes, con el objetivo de identificar un subconjunto óptimo de características que maximicen el rendimiento predictivo.

Algoritmos de entrenamiento

- Decision Tree Classifier: Este algoritmo es un método de aprendizaje supervisado que se utiliza para la clasificación y regresión.
- Naive Bayes: Este algoritmo es un método de aprendizaje supervisado que se utiliza para la clasificación.

III. Metodología

A continuación, un ejemplo de uso de listas numeradas:

1) *Trabajos con dos alumnos:* poner nombre y apellidos completos de cada uno, y correos electrónicos de contacto (a ser posible de la Universidad de Sevilla). El orden de los alumnos se fijará por orden alfabético según los apellidos.

Para el desarrollo de este trabajo se han implementado los algoritmos de búsqueda secuencial hacia atrás y búsqueda secuencial hacia atrás mixta. Todo esto en el lenguaje de programación Python, utilizando las librerías de Scikit-learn y Pandas.

A. Funciones implementadas

Método Robust Evaluation: Para ambos métodos se ha usado la función *robust_evaluation* que hace uso del método *cross_val_score* de la librería Scikit-learn, esta función nos permitirá realizar evaluaciones mediante validación cruzada. Respecto a la función *robust_evaluation* está nos permitirá evaluar el rendimiento de los subconjuntos de variables predictoras seleccionadas, siendo necesario para su uso:

1) X: El conjunto de datos con todas las variables predictoras.
2) y: El conjunto de datos con la variable objetivo.
3) model: El modelo de clasificación que se va a evaluar.
4) N_Exp: El número de repeticiones del experimento por validación cruzada.
5) cV: El número de pliegues del conjunto de datos para la validación cruzada.

Devolviendo el promedio de las N_Exp evaluaciones realizadas por la función *cross_val_score*, usando como medida de rendimiento la tasa de acierto balanceada.

Búsqueda secuencial hacia atrás: Esta función nos proporcionará una tabla con los subconjuntos de variables predictoras, su rendimiento, y número de variables predictoras seleccionadas. Para su uso es necesario:

1) X: El conjunto de datos con todas las variables predictoras y la variable objetivo.
2) objective: El nombre de la variable objetivo.
3) model: El modelo de clasificación que se va a evaluar.
4) N_Exp: El número de repeticiones del experimento por validación cruzada.
5) cV: El número de pliegues del conjunto de datos para la validación cruzada.

Una vez sabemos que es necesario para su uso pasamos a la explicación de dicho algoritmo. En primer lugar, se inicializa el conjunto de variables predictoras con todas las variables predictoras, y otro con la variable respuesta. Además de una lista vacía que contendrá el resultado, y una solución actual que contendrá las variables predictoras seleccionadas.

A continuación, se realizará el siguiente proceso k veces, siendo k el número de variables predictoras menos.

1) Se eliminan variables de la solución actual una por una.
2) Se aplicará la función *robust_evaluation* sobre conjunto de variables predictoras actualizado.
3) Se guardará el mejor rendimiento de entre todos los subconjuntos de variables predictoras, además de la peor variable.
4) Se elimina de la solución actual la peor variable.
5) Se añade la solución actual el conjunto de variables que proporcionó el mejor rendimiento al resultado.
6) Se vuelve al inicio del bucle si este no ha sido ejecutado k veces.
7) Por último añadimos al resultado el rendimiento de el conjunto de variables predictoras completo.
8) Devolvemos un DataFrame con el resultado.

Búsqueda secuencial hacia atrás mixta: Esta función nos proporcionará un DataFrame con los subconjuntos el de variables predictoras, su rendimiento, y número de variables predictoras seleccionadas. Para su uso es necesario:

1) X: El conjunto de datos con todas las variables predictoras y la variable objetivo.
2) objective: El nombre de la variable objetivo.
3) model: El modelo de clasificación que se va a evaluar.
4) N_Exp: El número de repeticiones del experimento por validación cruzada.
5) cV: El número de pliegues del conjunto de datos para la validación cruzada.
6) M: Número de iteraciones una vez se hayan eliminado todas las variables predictoras, y además no se añade ninguna variable predictora.

Una vez sabemos que es necesario para su uso pasamos a la explicación de dicho algoritmo. En primer lugar, se inicializa el conjunto de variables predictoras con todas las variables predictoras, y otro con la variable respuesta. Además de una lista vacía que contendrá el resultado, y una solución actual que contendrá las variables predictoras seleccionadas.

Luego mientras se cumpla la condición de parada, la cuál será que el conjunto de variables predictoras haya sido eliminadas al menos 1 vez y una vez esto se cumpla el bucle haya tenido al menos M iteraciones, siempre y cuando no se añada ninguna variable predictora, se realizará el siguiente proceso:

1) Se eliminan variables de la solución actual una por una.
2) Se aplicará la función *robust_evaluation* sobre el conjunto de variables predictoras actualizado.
3) Se guardará el mejor rendimiento de entre todos los subconjuntos de variables predictoras, además de la peor variable.
4) Se elimina de la solución actual la peor variable y se añade a la lista de eliminadas.
5) Se añaden variables de la solución actual una por una, siempre que estas no estén ni en la lista de eliminados ni en la solución actual.
6) Se aplicará la función *robust_evaluation* sobre el conjunto de variables predictoras actualizado.
7) Se guardará el mejor rendimiento de entre todos los subconjuntos de variables predictoras si este supera el rendimiento ya calculado anteriormente, además de la mejor variable.
8) Por último se añade la solución actual el conjunto de variables que proporcionó el mejor rendimiento al resultado.
9) Devolvemos un DataFrame con el resultado.

Las figuras se deben mencionar en el texto, como la Fig. 1. También se pueden añadir ecuaciones, como la ecuación (1).

$$a + b = \gamma \tag{1}$$

Un ejemplo de pseudocódigo se puede observar en la Fig. 2.

* mergesort(V)

si V es unitario entonces
+ devolver V
- si no entonces
+ $V_1 \leftarrow$ primera mitad de V
$V_2 \leftarrow$ segunda mitad de V
$V_1 \leftarrow mergesort(V_1)$
$V_2 \leftarrow mergesort(V_2)$
devolver mezcla(V_1, V_2)
* mezcla(V_1, V_2)

$si V_1$ no tiene elementos entonces
+ devolver V_2
- si no si V_2 no tiene elementos entonces
+ devolver V_1
- si no entonces
+ $x_1 \leftarrow$ primer elemento de V_1
$x_2 \leftarrow$ primer elemento de V_2
si $x_1 \leq x_2$ entonces
+ $x \leftarrow x_1$
quitar el primer elemento de V_1
- si no entonces
+ $x \leftarrow x_2$
quitar el primer elemento de V_2
- $V \leftarrow mezcla(V_1, V_2)$
añadir x como primer elemento de V
devolver V

Fig. 2. Algoritmo de ordenación `MergeSort`

TABLE I
EJEMPLO DE TABLA

Encabezado 1	Encabezado 2	Encabezado 3
Celda 1	Celda 2	Celda 3
Celda 4	Celda 5	Celda 6

IV. RESULTADOS

En esta sección se detallarán tanto los experimentos realizados como los resultados conseguidos:

- Los experimentos realizados, indicando razonadamente la configuración empleada, qué se quiere determinar, y como se ha medido.
- Los resultados obtenidos en cada experimento, explicando en cada caso lo que se ha conseguido.
- Análisis de los resultados, haciendo comparativas y obteniendo conclusiones.

Se pueden hacer uso de tablas, como el ejemplo de la tabla I.

V. CONCLUSIONES

En esta sección se hace un resumen de los resultados obtenidos y se sacan conclusiones finales.

Además, es importante mencionar las limitaciones del trabajo realizado y las posibles líneas futuras de investigación.

AGRADECIMIENTOS

Agradecimientos (si los hay).

REFERENCES

[1] Primer artículo.
[2] Segundo artículo.
[3] Tercer artículo.