Minería de Datos

Práctica 4. Clasificación y selección de atributos

Ángel Ríos San Nicolás

19 de marzo de 2021

1. Carga el fichero de datos "credit_g.arff" en Weka.

2. Estudia los distintos atributos, su significado y su distribución. ¿Hay algún atributo que podría ser considerado "poco relevante"? (por ejemplo, porque la mayoría de las filas caigan en el mismo valor).

checking_status	Existencia y estado actual de la cuenta corriente
duration	Duración del crédito en meses
credit_history	Historial de créditos: sin créditos previos, impagados, etc.
purpose	Propósito: coche, equipamiento, reformas, formación, etc.
credit_amount	Cuantía del crédito
saving_status	Cuantía en cuentas de ahorro
employment	Antigüedad en el trabajo: desempleado, de 1 a 4 años, etc.
installment_commitment	Tasa de pago a plazos en porcentaje respecto al ingreso
personal_status	Género y estado civil
other_parties	Existencia de terceros: otros deudores, solicitantes, etc.
residence_since	Antigüedad en la residencia actual
property_magnitude	Propiedades: Inmobiliaria, hipoteca, vehículos, etc.
age	Edad
other_payment_plans	Otros modos de pago: Banco, bienes o ninguno
housing	Vivienda de alquier, propia o gratuita
existing_credits	Número de creditos existentes con el banco
job	Profesión: cualificado, no cualificado, autónomo, etc.
num_dependents	Número de dependientes del crédito
own_telephone	Propietario o no de teléfono propio
foreign_worker	Trabajador extranjero o no
class	Calidad del cliente respecto a la concesión de un crédito

   La mayoría de las instancias tienen el mismo valor en los atributos *other-parties* y *foreign worker* por lo que podríamos considerarlos poco relevantes. Podríamos también considerar poco relevante si el cliente tiene o no teléfono propio, es decir, el atributo *own_telephone*.

3. Realiza las siguientes tareas de clasificación con 10-cross-validation:

   a. Usando J48

   b. Usando NaïveBayes

   c. Usando Redes Bayesianas con TAN

   d. Usando 5-NN (Vecinos cercanos con $k = 5$.)

4. Una vez obtenidos los diferentes modelos del punto 3, responde:

   a. ¿Qué clasificador obtiene mejores resultados? ¿Cuál tiene un peor rendimiento?

      - J48

Accuracy: 70.5%

good	*bad*
588	112
183	117

Clase	Recall	Precision
good	0.840	0.763
bad	0.390	0.511

- NaïveBayes

Accuracy: 75.4%

good	*bad*
605	95
151	149

Clase	Recall	Precision
good	0.864	0.800
bad	0.497	0.611

- Redes Bayesianas con TAN

Accuracy: 74.5%

good	*bad*
599	101
154	146

Clase	Recall	Precision
good	0.856	0.795
bad	0.487	0.591

- 5-NN

Accuracy: 74.2%

good	*bad*
624	76
182	118

Clase	Recall	Precision
good	0.891	0.774
bad	0.393	0.608

De los clasificadores anteriores el que obtiene mejores resultados es NaïveBayes, es el método con el mayor Accuracy de los cuatro. Aunque tiene menor Recall que 5-NN para la clase *good*, su Precision es mayor en ambas clases. El clasificador que obtiene peor rendimiento es J48 porque sus medidas de evaluación son todas menores que las del resto de métodos.

b. ¿Qué diferencia hay entre el modelo de clasificación obtenido en el punto 3.b y el obtenido en el 3.c?

El modelo NaïveBayes es mejor que el método del árbol generador de peso máximo, clasifica mejor para los dos valores del atributo como se ve en la matriz de confusión y todas las medidas de evaluación son mayores en consecuencia.

5. Utilizando la técnica de selección de atributos CfsSubsetEval y 10-cross-validation, elimina los atributos que hayan sido seleccionados menos de un 70% de las veces. Repite el punto 3 usando solamente los atributos seleccionados para la clasificación. ¿Qué modelos mejoran y cuáles empeoran?

- J48

Accuracy: 70.5%

good	*bad*
615	85
210	90

Clase	Recall	Precision
good	0.879	0.745
bad	0.300	0.514

- NaïveBayes

Accuracy: 74.4%

good	*bad*
639	61
195	105

Clase	Recall	Precision
good	0.913	0.766
bad	0.350	0.633

- Redes Bayesianas con TAN

Accuracy: 72.5%

good	*bad*
631	69
206	94

Clase	Recall	Precision
good	0.901	0.754
bad	0.313	0.577

- 5-NN

Accuracy: 71.7%

good	*bad*
600	100
183	117

Clase	Recall	Precision
good	0.857	0.766
bad	0.390	0.539

El método J48 se comporta de manera similar, mantiene el mismo Accuracy, mejora clasificando la clase *good* tanto como empeora clasificando *bad*. Sabemos, como recoge la documentación del archivo de datos, que es más grave clasificar un cliente *bad* como *good* que lo contrario, con lo que podemos decir que el clasificador es algo peor.

Los métodos de NaïveBayes y de redes bayesianas con TAN tienen peor Accuracy, mejoran clasificando *good*, pero se equivocan mucho más clasificando *bad*, con lo que también empeora.

El método 5-NN empeora, tiene peor Accuracy y, al contrario que los anteriores, se equivoca más en la clasificación de ambas clases.

6. Repite el punto 5, pero usando ClassifierSubSetEval como método de selección de atributos, y usando como clasificador base el clasificador que luego se va a usar para generar el modelo de predicción. Prueba en cada caso eliminando los atributos que hayan sido seleccionado menos de un 70% o un 50%. ¿En qué casos se obtiene mejor o peor resultado? ¿Mejoran con respecto a los obtenidos en el apartado 5?

   - J48

     $\geq$ 70%

     Accuracy: 72.4%

good	*bad*
601	99
177	123

Clase	Recall	Precision
good	0.859	0.772
bad	0.410	0.554

     $\geq$ 50%

     Accuracy: 73%

good	*bad*
604	96
167	133

Clase	Recall	Precision
good	0.863	0.783
bad	0.443	0.581

     Obtenemos resultados algo mejores considerando los atributos que se seleccionan el 50% de las veces, mejora la matriz de confusión y, por lo tanto, todas las medidas de evaluación del clasificador.

   - NaïveBayes

     $\geq$ 70%

     Accuracy: 75%

good	*bad*
628	72
178	122

Clase	Recall	Precision
good	0.897	0.779
bad	0.407	0.629

     $\geq$ 50%

     Accuracy: 74.4%

good	*bad*
606	94
162	138

Clase	Recall	Precision
good	0.866	0.789
bad	0.460	0.595

     Si seleccionamos los atributos escogidos al menos un 50% de las veces, en este caso el clasificador clasifica más instancias como *bad*, con lo que mejora para esa clase, pero empeora clasificando *good*. Como resultado el Accuracy es ligeramente menor.

   - Redes Bayesianas con TAN

     $\geq$ 70%

     Accuracy: 74.5%

good	*bad*
604	96
159	141

Clase	Recall	Precision
good	0.863	0.792
bad	0.470	0.595

     $\geq$ 50%

     Accuracy: 74.7%

good	*bad*
607	93
160	140

Clase	Recall	Precision
good	0.867	0.791
bad	0.467	0.601

Al escoger los datos seleccionados un 50% de las veces, se clasifica mejor la clase *good*, pero peor la clase *bad*. La diferencia es sutil, tres instancias en el primer caso y una en el segundo, con lo que el Accuracy, aunque es mejor, varía muy poco.

- 5-NN

  $\geq$ 70%

  Accuracy: 76.2%

	good	*bad*
good	608	92
bad	146	154

Clase	Recall	Precision
good	0.869	0.806
bad	0.513	0.626

  $\geq$ 50%

  Accuracy: 73.7%

	good	*bad*
good	625	75
bad	188	112

Clase	Recall	Precision
good	0.893	0.769
bad	0.373	0.599

En el apartado anterior, los atributos fueron seleccionados de la misma manera independientemente del clasificador escogido. En este apartado, todos los clasificadores mejoran su Accuracy, lo que tiene sentido porque el proceso de selección de atributos se ha especializado en el clasificador que se ha usado después.

7. En base a lo observado, ¿qué tipo de clasificadores se ven más afectados por la selección de atributos? ¿por qué?

   El clasificador que se ve más afectado por la selección de atributos es 5-NN porque se basa en una distancia de los datos. Dadas dos instancias, se define una distancia a partir de sus atributos, pero, si eliminamos parte de estos, generalmente la distancia cambia completamente, con lo que también lo hacen los k vecinos más cercanos y el clasificador tiene un comportamiento muy diferente.

   J48, por el contrario, se ve poco afectado por la selección de atributos porque lleva implementada su propia selección.

   NaïveBayes supone por hipótesis que los atributos son independientes entre sí dado el valor de la clase, con lo que su clasificación se basará en los atributos que tengan mayor peso en la muestra. Redes bayesianas con TAN es una implementación de NaïveBayes con una búsqueda en un árbol generador de peso máximo. A la hora de seleccionar los atributos, también buscamos los que tienen mayor peso para la clasificación, así que tiene sentido que ambos clasificadores se vean relativamente poco afectados por la selección previa.