2016/2017

Ingeniería basada en el conocimiento

Opiniones sobre hoteles

Héctor Rodríguez Salgado & Marta Loriente Nieves
Máster en Ingeniería Informática – ESEI – Universidad de Vigo

Índice de contenido

Introducción 2

Exploración de datos 2

Análisis de predicción de atributos 2

Lazy learning 3

Decision trees 3

Rules 4

Clustering 5

K-Means 6

EM 7

Associate 7

Conclusión final 8

Índice de ilustraciones

Ilustración 1 - Dataset inicial 2

Ilustración 3 - IB1 3

Ilustración 4 - IB5 3

Ilustración 5 - J48 4

Ilustración 6 - JRIP 4

Ilustración 7: ZeroR 5

Ilustración 8 - PART 5

Ilustración 9 - SimpleKMeans 6

Ilustración 10 - SimpleKMeans 6

Ilustración 11: SimpleKMeans 6

Ilustración 12: EM 7

Ilustración 13: EM 7

Ilustración 14: A priori 8

Introducción

El objetivo principal del análisis de datos es la construcción y comparación de diferentes modelos de minería de datos, enfocando la discusión en sus resultados.

Se evaluarán un conjunto de datos siguiendo tres enfoques de análisis diferentes: modelos predictivos, modelos de agrupamiento y modelos de asociación.

Todos estos modelos se construirán utilizando la herramienta de minería de datos WEKA y sus algoritmos.

Exploración de datos

Para realizar los análisis se ha escogido un conjunto de datos de alrededor de 12.000 instancias que recogen las opiniones de los clientes que han visitado distintos hoteles. Dicha información se ha obtenido del buscador de hoteles *Tripadvisor*.

En un inicio, el *dataset* contenía la siguiente información:

```
<Author>KGBT
<Content>Wow, what charm! As a Travel Agent, I've stayed at quite a few hotel, but this is the only Historic
hotel so far...  I loved it! Had to go back for a personal stay. The decor is beautiful, the lobby furniture
fits the time period  is still comfy. The city view rooms are great - love the little balconies. Great
breakfast, nice people, great location - The Seattle Underground Tours is a 1/2 block away. I've aready sent
my folk there for a stay  have told others.
<Date>Dec 14, 2008
<No. Reader>-1
<No. Helpful>-1
<Overall>5
<Value>4
<Rooms>5
<Location>5
<Cleanliness>5
<Check in / front desk>4
<Service>-1
<Business service>4
```

Ilustración 1 - Dataset inicial

Para el análisis, únicamente se requería la opinión del cliente y su valoración. Por tanto, se ha extraído cada comentario junto con su valoración, obteniendo como resultado en cada instancia la opinión del cliente y su valoración del hotel.

Una vez realizado dicho proceso, se han escogido las 50 palabras que más se repiten en todas las opiniones para determinar si la opinión del cliente ha sido buena o mala hacia el hotel en el que se ha hospedado, ayudándose de la puntuación que le ha dado el cliente al hotel. Dichas palabras pasarán a ser los atributos de interés para el análisis.

A continuación, se muestra el listado de las palabras escogidas: “hotel, small, bed, parking, rooms, friendly, good, got, view, price, seattle, helpful, day, bathroom, area, like, excellent, night, comfortable, desk, didnt, breakfast, dont, little, city, best, recommend, street, hotel, place, floor, free, just, restaurant, staff, right, clean, great, room, nice, really, stay, lobby, stayed, hotels, service, nights, location, time, did, walk”.

Análisis de predicción de atributos

La primera tarea es determinar si es posible predecir si la opinión de un cliente es buena o mala en base a las palabras que contiene su opinión y su valoración. Para ello se han ejecutado diferentes algoritmos predictivos y se ha observado el resultado.

Lazy learning

El primer algoritmo que se ha decidido utilizar ha sido de tipo “lazy learning: (k-) nearest-neighbour”, conocido como IBK en WEKA. Se ha decidido realizar una predicción para cada atributo del dataset, variando el número de vecinos en cada análisis. La tendencia en cada uno de los análisis es que, a más vecinos, mejor es la predicción realizada, ya que el porcentaje de acierto es mayor.

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        7386               60.4221 %
Incorrectly Classified Instances      4838               39.5779 %
Kappa statistic                          0.2018
Mean absolute error                      0.4096
Root mean squared error                  0.569
Relative absolute error                 84.6253 %
Root relative squared error            115.664  %
Total Number of Instances            12224

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,598    0,392    0,516      0,598    0,554      0,204    0,607     0,479     0
               0,608    0,402    0,685      0,608    0,644      0,204    0,607     0,669     1
Weighted Avg.  0,604    0,398    0,615      0,604    0,607      0,204    0,607     0,591

=== Confusion Matrix ===

    a    b   <-- classified as
 3004 2017 |    a = 0
 2821 4382 |    b = 1
```

Ilustración 2 - IB1

```
=== Summary ===

Correctly Classified Instances        7926               64.8397 %
Incorrectly Classified Instances      4298               35.1603 %
Kappa statistic                          0.2653
Mean absolute error                      0.4114
Root mean squared error                  0.4732
Relative absolute error                 84.9893 %
Root relative squared error             96.1847 %
Total Number of Instances            12224

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,535    0,272    0,578      0,535    0,555      0,266    0,679     0,603     0
               0,728    0,465    0,692      0,728    0,709      0,266    0,679     0,712     1
Weighted Avg.  0,648    0,386    0,645      0,648    0,646      0,266    0,679     0,667

=== Confusion Matrix ===

    a    b   <-- classified as
 2684 2337 |    a = 0
 1961 5242 |    b = 1
```

Ilustración 3 - IB5

Como se puede observar en los resultados anteriores, a mayor número de vecinos, mayor es el porcentaje de acierto de este algoritmo. Aunque observando los aciertos obtenidos, se reduce considerablemente el número de instancias clasificadas correctamente para el tipo ‘b’ (valoraciones positivas sobre hoteles) y, en cambio, aumenta para el tipo ‘a’ (valoraciones negativas sobre hoteles). Esto indica que no siempre que se obtenga un porcentaje de acierto mayor, el análisis va a ser mejor, como ocurre en este caso, ya que dicho porcentaje, a efectos prácticos, sólo se refleja en las valoraciones positivas, no en las valoraciones negativas.

Decision trees

El concepto principal detrás del aprendizaje árbol de decisión es que, a partir de los datos de entrenamiento, se va a construir un modelo predictivo que se asigna a una estructura de árbol. El objetivo es lograr la clasificación perfecta con el número mínimo de la decisión, aunque no siempre es posible debido al ruido o inconsistencias en los datos.

Dentro de este tipo de algoritmos se ha decidido realizar el análisis con el tipo J48.

El resultado que se ha ofrecido utilizando un testeo de cross-validation de 10 folds es el siguiente:

```
=== Summary ===

Correctly Classified Instances        7711               63.0808 %
Incorrectly Classified Instances      4513               36.9192 %
Kappa statistic                          0.2311
Mean absolute error                      0.4113
Root mean squared error                  0.5518
Relative absolute error                 84.9696 %
Root relative squared error            112.1662 %
Total Number of Instances            12224

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,524    0,295    0,553      0,524    0,538      0,231    0,595     0,469     0
               0,705    0,476    0,680      0,705    0,692      0,231    0,595     0,648     1
Weighted Avg.  0,631    0,402    0,628      0,631    0,629      0,231    0,595     0,574

=== Confusion Matrix ===

    a    b   <-- classified as
 2630 2391 |    a = 0
 2122 5081 |    b = 1
```

Ilustración 4 - J48

Se toma como atributo clase las valoraciones siendo 0 las valoraciones negativas y 1 las positivas. Tal y como se ve en la ilustración anterior está cometiendo más error cuando clasifica las valoraciones negativas. Con ello se puede decir con certeza que este algoritmo no es bueno para realizar un estudio para este caso.

Rules

Dentro de este tipo de algoritmos el primero que se va a utilizar será JRIP cuyo objetivo es podar para cortar la reducción de errores, obteniendo como resultado en base al atributo *valuation* como clase del algoritmo:

```
=== Summary ===

Correctly Classified Instances        8305               67.9401 %
Incorrectly Classified Instances      3919               32.0599 %
Kappa statistic                          0.3049
Mean absolute error                      0.4183
Root mean squared error                  0.4627
Relative absolute error                 86.4136 %
Root relative squared error             94.0598 %
Total Number of Instances            12224

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,454    0,164    0,659      0,454    0,538      0,317    0,659     0,604     0
               0,836    0,546    0,687      0,836    0,755      0,317    0,659     0,684     1
Weighted Avg.  0,679    0,389    0,676      0,679    0,666      0,317    0,659     0,651

=== Confusion Matrix ===

    a    b   <-- classified as
 2280 2741 |    a = 0
 1178 6025 |    b = 1
```

Ilustración 5 - JRIP

Como el resto de algoritmos vistos hasta ahora se puede que se mantiene el tanto por ciento de instancias clasificadas correctamente, pero si se observa detenidamente la matriz de confusión se produce menos errores al predecir las valoraciones positivas como negativas. Por lo tanto, en base a esta última afirmación se puede decir que es hasta el momento el mejor algoritmo de predicción para este caso.

El siguiente algoritmo de este tipo será OneR manteniendo el atributo clase *valuation* obteniendo más o menos el mismo resultado que con el algoritmo JRIP. Sin embargo, utilizando el algoritmo ZeroR los resultados son bastante distintos como se puede ver en la siguiente imagen:

```
=== Summary ===

Correctly Classified Instances        7203               58.9251 %
Incorrectly Classified Instances      5021               41.0749 %
Kappa statistic                          0
Mean absolute error                      0.4841
Root mean squared error                  0.492
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances            12224

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,000    0,000    0,000      0,000    0,000      0,000    0,500     0,411     0
               1,000    1,000    0,589      1,000    0,742      0,000    0,500     0,589     1
Weighted Avg.  0,589    0,589    0,347      0,589    0,437      0,000    0,500     0,516

=== Confusion Matrix ===

    a    b   <-- classified as
    0 5021 |    a = 0
    0 7203 |    b = 1
```

Ilustración 6: ZeroR

A la hora de predecir las valoraciones negativas no tiene ninguna precisión, con lo cual este algoritmo queda completamente descartado.

A continuación, se muestra el resultado obtenido para el algoritmo PART:

```
=== Summary ===

Correctly Classified Instances        7595               62.1319 %
Incorrectly Classified Instances      4629               37.8681 %
Kappa statistic                          0.2188
Mean absolute error                      0.4017
Root mean squared error                  0.5826
Relative absolute error                 82.9828 %
Root relative squared error            118.4167 %
Total Number of Instances            12224

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,544    0,325    0,539      0,544    0,541      0,219    0,586     0,455     0
               0,675    0,456    0,680      0,675    0,678      0,219    0,586     0,655     1
Weighted Avg.  0,621    0,402    0,622      0,621    0,622      0,219    0,586     0,573

=== Confusion Matrix ===

    a    b   <-- classified as
 2730 2291 |    a = 0
 2338 4865 |    b = 1
```

Ilustración 7 - PART

Se mantienen los resultados medios obtenido con los primeros algoritmos aplicados, pero se sigue obteniendo un error al predecir las valoraciones positivas de la mitad de las instancias.

Con respecto a los árboles de decisión se puede afirmar que nos aporta peores resultados, con lo cual queda eliminado.

Conclusión: para el análisis de atributos el algoritmo que mejor se adapta es el **Rules** porque es el que obtiene mayor porcentaje de instancias clasificadas correctamente y los errores que se comenten en la predicción, tal y como se muestra en la matriz, son los menores. Por lo tanto, tiene mayor precisión a la hora de clasificar los atributos.

Clustering

El *clustering* o agrupamiento es la tarea de agrupar un conjunto de objetos de tal manera que los miembros del mismo grupo (llamado clúster) sean más similares, en algún sentido u otro. ES la tarea principal de la minería de datos exploratoria y es una técnica común en el análisis de datos estadísticos.

K-Means

Es un método de agrupamiento, que tiene como objetivo la participación de un conjunto de *n* observaciones en *k* grupos en el que cada observación pertenece al grupo cuyo valor medio es más cercano.

Analizamos con el número de clúster igual a 10. También ponemos el atributo displayStdDevs a True (atributo que nos mostrará información más detallada de los clústers).

Vamos a ejecutar el .arff con el modo de testeo *Classes to cluster evaluation* con el atributo *valuation* como clase.

A continuación, se explica paso a paso las partes que se han obtenido al ejecutar este dataset. La primera captura nos muestra el número de iteraciones junto con el error de la suma cuadrática, que se ha comprobado que cuánto mayor número de clúster es menor. La siguiente sección muestra qué puntos aleatorios se pusieron al inicio para saber si son iguales, distintos…

```
kMeans
======

Number of iterations: 5
Within cluster sum of squared errors: 125582.0

Initial starting points (random):

Cluster 0: 0,0,1,0,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,1,1,0,1,1,0,0,0,0,0,1,0,1,1,1,0,0,0,0,0,1,0
Cluster 1: 0,0,0,1,0,0,0,0,0,0,1,0,1,1,0,0,1,1,1,0,0,0,1,0,0,1,0,1,1,0,0,0,0,1,0,1,1,1,0,1,1,0,1,0,0,0,1,0,0,1
```

Ilustración 8 - SimpleKMeans

Se muestran ahora los clústers que se han realizado, y la aparición de las palabras en cada uno de ellos.

```
Final cluster centroids:
                           Cluster#
Attribute      Full Data           0           1
               (12224.0)    (8974.0)    (3250.0)
=============================================
small                  0           0           0
  0           9288.0 ( 75%)7286.0 ( 81%)2002.0 ( 61%)
  1           2936.0 ( 24%)1688.0 ( 18%)1248.0 ( 38%)

bed                    0           0           0
  0           8714.0 ( 71%)6815.0 ( 75%)1899.0 ( 58%)
  1           3510.0 ( 28%)2159.0 ( 24%)1351.0 ( 41%)
```

Ilustración 9 - SimpleKMeans

Tal y como se muestra en la ilustración anterior se han realizado 2 agrupaciones, con tantos miembros que se indica justo debajo del número del clúster. Seguidamente se toma cada una de las palabras del dataset y se detalla la aparición en cada clúster (valoraciones negativas y positivas). Este nivel de detalle es gracias al atributo mencionado anteriormente *displayStdDevs*.

Para finalizar se interpretan los clusters obtenidos:

```
=== Model and evaluation on training set ===

Clustered Instances

0       8974 ( 73%)
1       3250 ( 27%)


Class attribute: valuation
Classes to Clusters:

    0    1  <-- assigned to cluster
 4046  975 | 0
 4928 2275 | 1

Cluster 0 <-- 0
Cluster 1 <-- 1
```

Como se puede ver se ha obtenido el número de instancias para cada uno de los clústers. Es decir, 8974 ocurrencias se han producido en el clúster 0, valoraciones negativas, lo que es un 73% del total. El resto son valoraciones positivas para hoteles.

En la matriz de confusión se ve que se producen más fallos en la clasificación de valoraciones positivas, y el porcentaje de instancias incorrectamente asignadas es casi del 50%

Ilustración 10: SimpleKMeans

2 %

EM

El algoritmo de expectativa de maximización es un método iterativo para la búsqueda de la máxima verosimilitud, teniendo en cuenta variables latentes.

```
EM
==

Number of clusters: 2
Number of iterations performed: 27


                Cluster
Attribute             0          1
                 (0.23)     (0.77)
=================================
small
  0            2780.6486 6509.3514
  1              38.3994 2899.6006
  [total]       2819.048  9408.952
bed
  0            2738.5213 5977.4787
  1              80.5267 3431.4733
  [total]       2819.048  9408.952
```

Aplicamos este algoritmo con un número de clúster igual a 2. En este caso ya se pueden ver los resultados. En cada uno de estos clústers se puede observar la cantidad de instancias en cada uno de ellos. En el clúster 0 (valoraciones negativas) tiene 23% de las instancias y en el clúster 1 (valoraciones positivas) tienen un 77%.

Ilustración 112: EM

```
t ===

Clustered Instances

0       2813 ( 23%)
1       9411 ( 77%)


Log likelihood: -22.38216


Class attribute: valuation
Classes to Clusters:

    0    1  <-- assigned to cluster
 1605 3416 | 0
 1208 5995 | 1

Cluster 0 <-- 0
Cluster 1 <-- 1

Incorrectly clustered instances :       4624.0   37.8272 %
```

Ilustración 32: EM

De nuevo se pueden observar ambos clústers y el porcentaje junto al número de instancias asignados a cada uno. Más abajo se observa la matriz de confusión viendo que con respecto a las valoraciones negativas se producen más fallos que aciertos, mientras que con las valoraciones positivas la cantidad de instancias clasificadas erróneamente son menores.

El porcentaje de instancias clasificadas incorrectamente en los clústers es de un 37%.

Conclusión con respecto al clústering es mucho mejor el algoritmo EM debido a que comete menos errores y tiene un menor porcentaje de errores.

Associate

Este tipo de algoritmo consta de unas reglas de asociación y se utilizan para descubrir hecho que ocurren en común dentro de un determinado conjunto de datos.

En este apartado vamos a tener en cuenta el algoritmo *A priori*, el cual permite encontrar de forma eficiente "conjuntos de ítems frecuentes", que sirven de base para generar reglas de asociación.

```
Apriori
=======

Minimum support: 0.95 (11613 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 1

Generated sets of large itemsets:

Size of set of large itemsets L(1): 3

Size of set of large itemsets L(2): 3

Size of set of large itemsets L(3): 1

Best rules found:

 1. didnt=0 dont=0 12029 ==> seattle=0 11986    <conf:(1)> lift:(1) lev:(0) [5] conv:(1.1)
 2. dont=0 12115 ==> seattle=0 12070    <conf:(1)> lift:(1) lev:(0) [3] conv:(1.06)
 3. didnt=0 12113 ==> seattle=0 12067    <conf:(1)> lift:(1) lev:(0) [2] conv:(1.03)
 4. seattle=0 didnt=0 12067 ==> dont=0 11986    <conf:(0.99)> lift:(1) lev:(0) [26] conv:(1.31)
 5. didnt=0 12113 ==> dont=0 12029    <conf:(0.99)> lift:(1) lev:(0) [24] conv:(1.27)
 6. seattle=0 dont=0 12070 ==> didnt=0 11986    <conf:(0.99)> lift:(1) lev:(0) [25] conv:(1.29)
 7. dont=0 12115 ==> didnt=0 12029    <conf:(0.99)> lift:(1) lev:(0) [24] conv:(1.26)
 8. seattle=0 12175 ==> dont=0 12070    <conf:(0.99)> lift:(1) lev:(0) [3] conv:(1.02)
 9. seattle=0 12175 ==> didnt=0 12067    <conf:(0.99)> lift:(1) lev:(0) [2] conv:(1.01)
10. didnt=0 12113 ==> seattle=0 dont=0 11986    <conf:(0.99)> lift:(1) lev:(0) [25] conv:(1.19)
```

Ilustración 13: A priori

Como se puede comprobar en la ilustración 14 se han originado las 10 mejores reglas. Estas reglas se han basado en cuatro aspectos: confianza (confidence), elevación (lift), aplacamiento (leverage) y convicción (conviction).
Entonces la mejor regla tiene de confianza 1, lo cual significa que en todas aquellas instancias que no haya el atributo didn't y don't tampoco tendrán Seattle. Tiene de elevación 1, esto quiere decir que los conjuntos son totalmente independientes.
Se han realizado más análisis para intentar variar las reglas producidas, pero el resultado ha sido muy similar al mostrado anteriormente. Las palabras que el algoritmo ha escogido no son demasiado útiles para extraer alguna conclusión relevante.

Conclusión final

Tras realizar una valoración de todos los análisis realizados, se puede observar que, en la mayoría de ellos, el porcentaje de acierto de los algoritmos no alcanza niveles tan altos como se desearía. Esto se debe en gran parte a que la valoración proporcionada por el cliente no se corresponde del todo con el comentario que ha realizado hacia el hotel. Es decir, el cliente ha podido escribir un comentario bueno hacia ese hotel, pero ha dado una valoración demasiado baja. Con lo cual, no siempre se pueden asociar las palabras positivas con los comentarios con buena puntuación. Lo mismo ocurre en el caso contrario, que la valoración haya sido alta pero el comentario sea malo hacia el hotel.
A pesar de ello, se puede ver claramente que las valoraciones positivas hacia los hoteles superan con creces a las negativas, por tanto, se puede determinar que gran parte de los clientes que visitan un hotel de la web TripAdvisor han dado una valoración positiva del hotel.
Aunque se trata de un análisis general, ya que no se está tratando ningún hotel en concreto, se puede utilizar para ver que este buscador de hoteles contiene buenas referencias. Lo cual hace que, si un cliente realiza una búsqueda en TripAdvisor y se hospeda en un hotel de su lista, la próxima vez que necesite buscar otro hotel, posiblemente lo vuelva a hacer desde este buscador y no desde otro.