Introducción al Reconocimiento de Patrones 2015

Clasificación de Bosques por Información Cartográfica

Jimena Arruti, Pablo Massaferro

8 de diciembre de 2015

Índice

1. Introducción **1**

2. Análisis de datos **1**
2.1. Visualización de Datos 1
2.2. Análisis de relaciones entre características 2
2.3. Características binarias 3

3. Pre-procesamiento de datos **4**
3.1. Identificación y remoción de outliers 4
3.2. Balance de Clases 5
3.3. Definición de características 7

4. Selección y Extracción de características **7**
4.1. Criterios de selección 7
4.2. Análisis de Discriminantes Lineales (LDA) 8

5. Clasificación **10**
5.1. Primera aproximación 10
5.2. Optimización de IBK 10
5.3. Random Forest 12
5.3.1. Evaluación con características creadas 13
5.3.2. Evaluación con LDA 14
5.3.3. Evaluación de set de datos originales y LDA. 15
5.3.4. Optimización de parámetros 16

6. Combinación de Clasificadores **17**
6.1. Boosting 17
6.2. Cascada de Clasificadores 18
6.2.1. Primera etapa 18
6.2.2. Segunda etapa 19
6.2.3. Combinación de resultados 20

7. Desempeño del SRP sobre el conjunto de test **20**

8. Conclusiones **22**

9. Bibliografía **23**

Resumen

Se busca encontrar el mejor sistema de reconocimiento de patrones para un problema de clasificación de tipo de árboles en bosques nativos. Se hace énfasis en las diferentes etapas de análisis en un proyecto de este tipo: edición, pre-procesamiento de los datos, creación de características, selección y extracción de características, distintos tipos de clasificadores y cómo optimizarlos, y finalmente combinación de clasificadores. Se obtiene como mejor sistema, para los datos de entrenamiento, el uso del clasificador Random Forest utilizando además *AdaBoost*; aunque resulta ser un sistema sobreajustado a la hora de evaluar su desempeño para los datos de test.

1. Introducción

El problema presentado en este proyecto consiste en analizar datos sobre la clasificación del tipo mayoritario de árboles en bosques nativos estadounidenses. Concretamente, se analizarán cuatro áreas de bosques naturales ubicados al norte del estado de Colorado. Cada muestra consiste en un área de 30x30 metros.

Se diseñará un sistema de reconocimiento de patrones buscando maximizar el acierto en la clasificación de tipo de árbol de cobertura. El entrenamiento del sistema será realizado el utilizando los datos pertenecientes a la base de datos pública: `https://archive.ics.uci.edu/ml/datasets/Covertype`

El objetivo primario del presente trabajo es profundizar el aprendizaje en las herramientas y metodología a utilizar frente a este tipo de problemas, así como entender todas las etapas de trabajo que esto conlleva. El desempeño final del clasificador obtenido, según fue planteado en el curso, debe ser al menos de un 82 % de acierto sobre las muestras de entrenamiento.

2. Análisis de datos

2.1. Visualización de Datos

Inicialmente se tiene un set de datos de entrenamiento que consiste en 7612 muestras etiquetadas dentro de 7 clases posibles de bosques, donde cada muestra cuenta con 54 características.

La primera aproximación a los datos se da utilizando las herramientas de visualización de `Weka`. Observando los histogramas individuales de cada característica se tiene una primer idea, un tanto intuitiva, de qué características pueden llegar a tener más peso a la hora de realizar la clasificación, e incluso cuáles podrían llegar a ser descartadas por no aportar información o por mostrar un grado importante de correlación. Se distingue también el caso de las primeras 10 características (numéricas) de las restantes (binarias), por ser de diferente naturaleza.

Sobre las primeras, se observa entonces, que la característica que a priori parece ser más relevante es la denominada *Elevation*, la cual parece distinguir razonablemente entre las distintas clases.

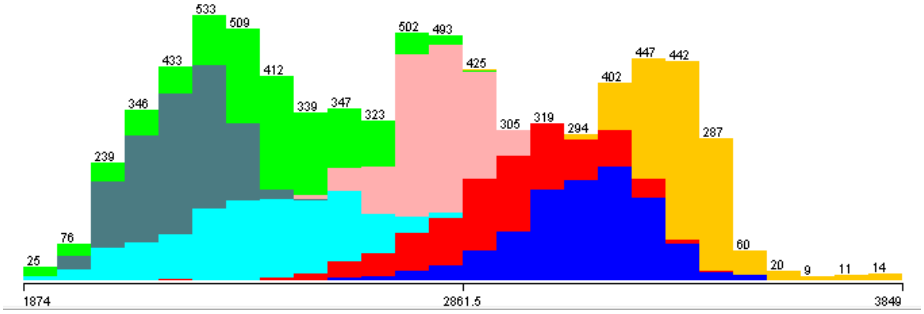


Figura 1: Histograma obtenido en Weka para la clasificación por característica de Elevación.

La mayoría de las características numéricas no muestran buena separabilidad entre clases si son vistas individualmente, como se puede apreciar, a modo de ejemplo, en la figura 2.

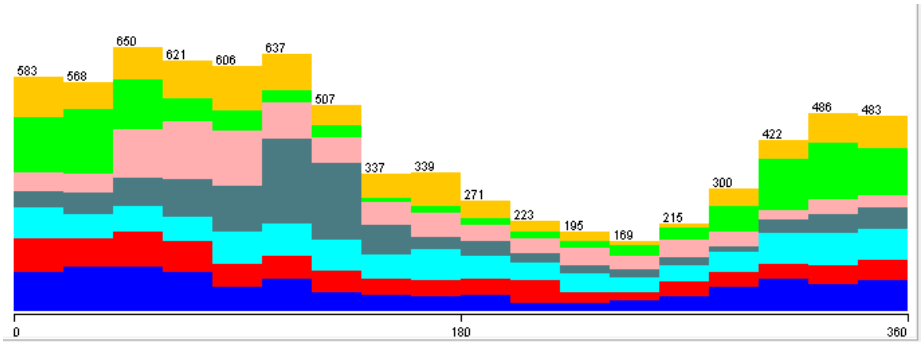


Figura 2: Histograma obtenido en Weka para la clasificación por característica Aspect.

2.2. Análisis de relaciones entre características

Por la descripción de las características, se puede intuir relaciones entre algunas de ellas. Distinguimos dos casos evidentes:

- *Horizontal_Distance_To_Hidrology* y *Vertical_Distance_To_Hidrology*. Si bien son variables independientes, podría observarse qué sucede al combinarlas y obtener otra medida de distancia a la fuente de agua más cercana, por ejemplo la distancia euclideana obtenida a partir de ellas.

- *Hillshade_9am, Hillshade_Noon y Hillshade_3pm.*

  El índice de sombra medido a distintas horas debería tener una correlación. Visualizando las características 2 a 2 se pueden identificar umbrales e incluso algunas muestras como outliers (figura 3).

  Además, se observan en gráficas de características 2 a 2 las dependencias

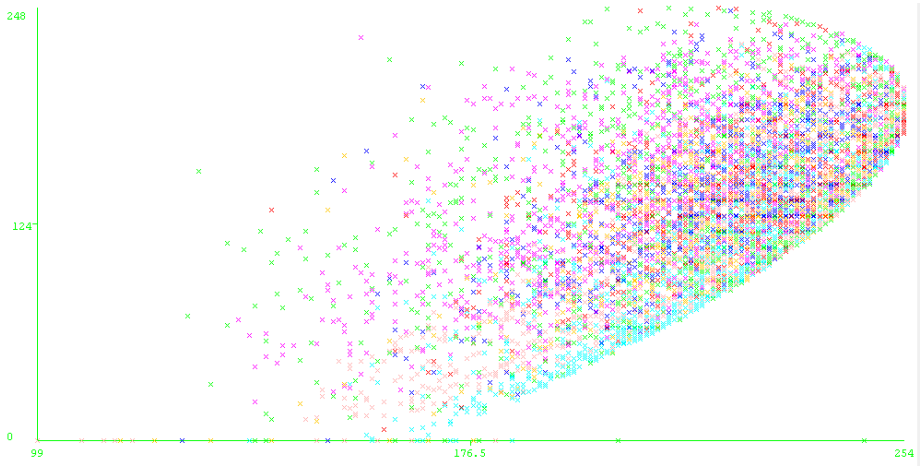
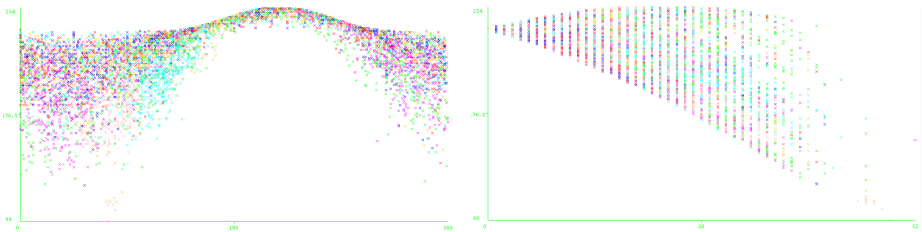


Figura 3: Característica de Indice de sombras a mediodía y 3pm.

entre el índice de sombra y las características *Aspect* y *Slope*, las cuales pueden verse en la figura 4.



(a) Visualización para las características Aspect (abscisas) y Hillshade_Noon (ordenadas)

(b) Visualización para las características Slope (abscisas) y Hillshade_Noon (ordenadas)

Figura 4: Visualización de características dos a dos.

Se identifican otras relaciones particulares, como pueden ser la relación entre las características *Elevation* y *Vertical_Distance_To_Hydrology*, lo cual se puede ver en la figura 5. Observando el eje de inclinación que presentan los datos, parece ser adecuado realizar extracción de características mediante LDA. Esto debería mejorar el desempeño al menos para los clasificadores de tipo árbol, al tomar un eje de coordenadas que ayude a mejorar la separabilidad entre determinadas clases.

2.3. Características binarias

Respecto a las características binarias, se distinguen dos casos:

- Las 4 características de tipos de área son simplemente descripciones de localización dentro del terreno estudiado, por lo cual podrían aportar información sobre si existen tipos de forestación con mayor presencia en determinadas zonas.

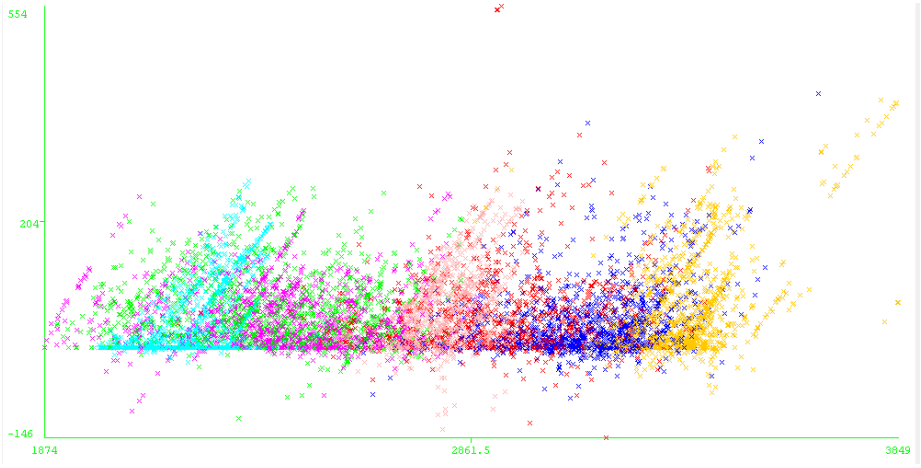


Figura 5: Visualización para las características Elevation (abscisas) y Vertical_Distance_To_Hydrology(ordenadas)

- Las 40 características correspondientes al tipo de suelo están definidas de un modo que parece no ser muy eficiente. Cada muestra puede tomar el valor 1 sólo en una de las 40 por lo que a priori ya se ve una fuerte correlación entre estas variables y una dimensionalidad del problema que no parece ser la óptima. Si se observa con detenimiento la descripción de cada tipo de suelo, se ve que hay palabras descriptivas que se repiten a lo largo de las distintas características. Con dicha información se podrían generar nuevas características para describir el tipo de suelo indicando pertenencia o no, o grados de pertenencia, a ciertos subgrupos como *rubbly, stony*, etc.

Por otro lado, es conveniente observar que las características a utilizar efectivamente aporten algún tipo de información. El caso extremo sucede con las características *Soil_type_7*, *Soil_type_8* y *Soil_type_15*, que siempre toman el valor 0, lo cual quiere decir que nunca se ven representadas en las muestras y no aportan información alguna.

3. Pre-procesamiento de datos

3.1. Identificación y remoción de outliers

Como se mencionó en la sección anterior, se detectó visualmente la presencia de datos que presentan valores muy diferentes al resto, incluso superando algunos umbrales que parecen ser condiciones intrínsecas del problema en cuestión. En este punto se decide utilizar un filtrado no supervisado que se basa en el comportamiento estadístico de los datos, descartando los casos extremos. Se utilizó el filtrado por rango de intercuartiles con un factor de 3 entre los cuartiles Q1 y Q3. De esta forma se detectaron 87 datos que fueron considerados outliers, por lo cual fueron removidos del set de datos.

3.2. Balance de Clases

Habiendo removido los outliers se analiza el desbalance de representatividad entre clases. Con la premisa de que los datos se deberían distribuir de forma equiprobable entre las clases, se busca disminuir el efecto que esto pueda tener en el desempeño del Sistema de Reconocimiento de Patrones (SRP). En la figura 6 se puede ver que el desbalance más notorio se da en la clase 2.

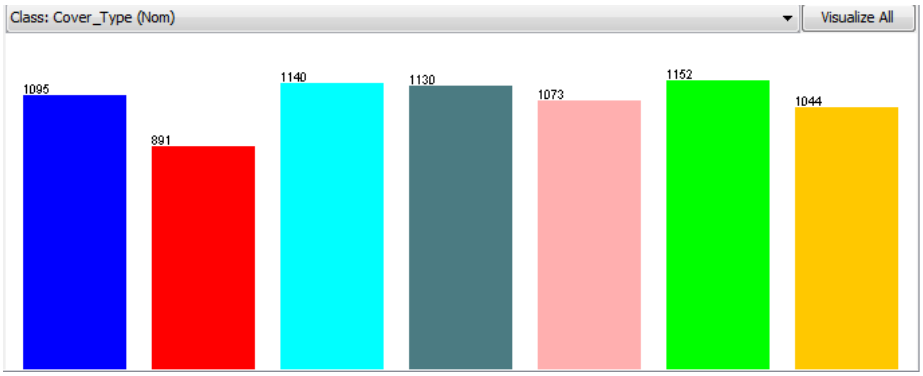


Figura 6: Distribución de datos de entrenamiento por clase

Se utiliza el método de sobre muestreo *Synthetic Minority Oversamplig Technique* (SMOTE) para generar nuevos datos y de esa forma balancear las clases. Se utilizó un 17.2 % como parámetro de instancias a crear, con 5 vecinos más cercanos, para igualar la clase minoritaria (clase 2) a la clase siguiente en cantidad de muestras (clase 7). El resultado se observa en la figura 7.

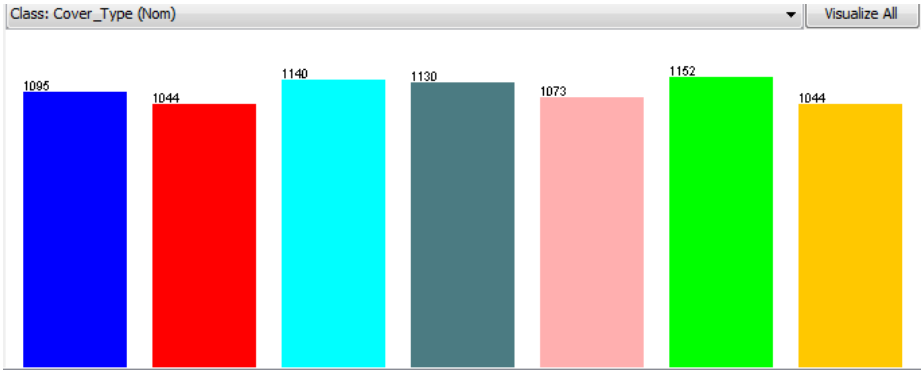


Figura 7: Distribución de datos de entrenamiento por clase Balanceados

La generación estadística de muestras no respeta los umbrales detectados en la sección 2. Al generar nuevos datos se generan nuevos outliers lo cual se puede visualizar en varias relaciones de características. A modo de ejemplo se muestran los nuevos outliers en la relación entre indices de sombra en la figura 8.

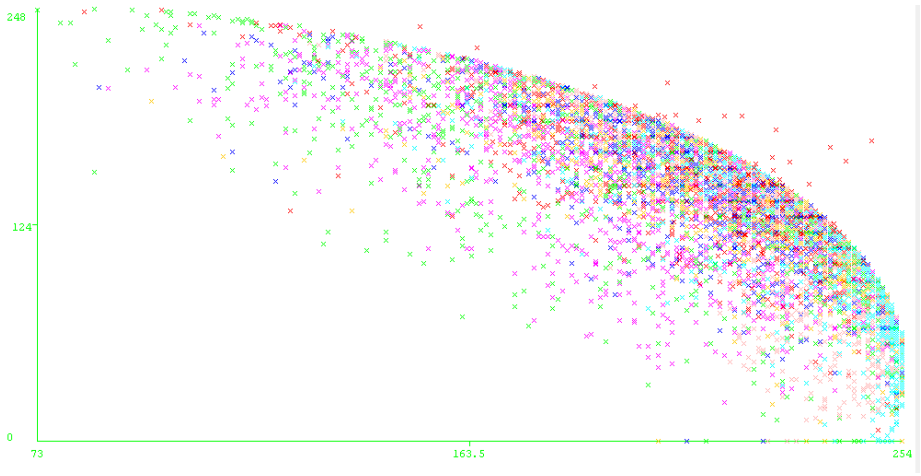


Figura 8: Nuevos outliers de la clase 2 en relación entre Hillshade_9am y Hillshade_3pm.

Debido a la dificultad en balancear las clases respetando todas las restricciones identificadas, se decide evaluar el impacto que puede tener este desbalance sobre el desempeño de algunos clasificadores que serán utilizados. Para ello se dividió el set de datos en dos nuevos sets, uno de entrenamiento con el 66 % de los datos y otro de test con el restante 34 %. Sobre el segundo conjunto se aplico SMOTE sobre 3 clases, hasta obtener un conjunto de datos con un mejor balance entre clases. Se utilizan los clasificadores *Vecino más cercano (1-NN)*, *Árbol C4.5* y *Naive Bayes* sobre el conjunto de datos completo con la opción split para test en 66 %. Esto se compara con el resultado de entrenar y evaluar usando el set de datos de test balanceado descrito anteriormente. Los resultados se presentan en la tabla 1.

Clasificador	Desempeño split (%)	Desempeño test balanceado (%)
1-NN (IBK)	76.1923	75.0942
C4.5 (J48)	74.1595	75.0942
NaiveBayes	67.2791	67.4830

Cuadro 1: Comparación de desempeño de clasificadores, con o sin balance de datos de test

En la tabla se ven diferentes comportamientos según el clasificador utilizado. En un caso aumenta, en otro disminuye y en otro se mantiene el desempeño. Ningún cambio parece ser muy significativo, ni los resultados concluyentes, por lo que se decide continuar sin agregar nuevos datos al set de entrenamiento. El efecto del desbalance detectado se comprobará posteriormente mediante el uso de los datos de test provistos por el cuerpo docente, los cuales sí presentan equiprobabilidad de clases.

3.3. Definición de características

En primer término son eliminadas las 3 características binarias que fueron identificadas previamente por no aportar información.

Habiendo analizado los datos, se plantea la creación de nuevas características que podrían eventualmente resultar de mayor utilidad. Se pretende, más adelante, trabajar con diferentes conjuntos de las mismas, con el fin de disminuir la dimensionalidad del problema así como para evaluar posibles mejoras en el desempeño de los clasificadores a utilizar posteriormente.

Se crea una nueva característica combinando todas las características binarias de tipos de suelo en una única variable nominal. Del mismo modo se crea una única variable para las variables binarias de áreas. Estas características se crean con el objetivo de disminuir la dimensionalidad.

Respecto a las distancias a cursos de aguas mencionadas en la sección 2.1, se crean características nuevas para evaluar su aporte de información en este problema. Estas son: distancia euclideana a la fuente de agua más cercana, y suma y producto de distancias horizontal y vertical a la fuente de agua más cercana.

Teniendo en cuenta la relación mencionada en la sección 2.2 entre los índices de sombras se crean tres nuevas características. Buscando representar un índice medio de la sombra en la zona muestreada se suman los tres indices de sombras. Se crean otras dos características representando las diferencias entre los índices de sombras a medida que avanza el día: sombra a las 12am menos el valor a las 9 am, e índice de sombra a las 3pm menos el valor a las 12am.

En suma, se tienen las características indicadas en el cuadro 2, las cuales posteriormente serán referenciadas con el número correspondiente.

4. Selección y Extracción de características

4.1. Criterios de selección

A lo largo de la resolución de problema se utilizó como criterio de selección de características una aproximación de filtrado; se midió la relevancia de las mismas mediante su ganancia de información. Este método es computacionalmente simple y permitió evaluar rápidamente la variación en desempeño al tomar diferentes subconjuntos de forma manual. En los casos en los que se exploró con clasificadores del tipo árbol la selección es intrínseca (embedding) como son el caso de J48 y *Random Forest* que se utiliza más adelante, por encargarse el propio árbol de elegir qué característica es conveniente utilizar para cada división.

Según el criterio de ganancia de información, las 59 características que han sido mencionadas se ordenan como se muestra en el cuadro 3.

Confirmando lo visto en la sección 2, la característica más relevante, tomando este criterio, es *Elevation* (1), seguida por las 2 variables creadas para combinar las características binarias (53 y 15). Dentro de las 10 más relevantes también se encuentran *Horizontal_Distance_To_Roadways*, *Horizontal_Distance_To_Hydrology*

1	Elevation
2	Aspect
3	Slope
4	Horizontal distance to hydrology
5	Vertical distance to hydrology
6	Horizontal distance to roadways
7	Hillshade 9am
8	Hillshade Noon
9	Hillshade 3pm
10	Horizontal distance to fire points
11-14	Wilderness area (4 variables binarias)
15	*Area (conjunción de binarias)*
16-52	Soil type (37 variables binarias)
53	*Soil (conjunción de binarias)*
54	*Distancia Euclideana a fuente de agua*
55	*Suma de distancias a fuente de agua*
56	*Multiplicación de distancias a fuente de agua*
57	*Suma de los valores de sombras a diferentes horas*
58	*Diferencia entre sombra de las 12am y las 9am*
59	*Diferencia entre sombra de las 3pm y las 12am*

Cuadro 2: Características a utilizar

y la nueva característica de distancia euclideana a fuente de agua (54).

Resulta de particular interés observar en qué posición se ubican las variables creadas, y su ubicación respecto a las variables a partir de las cuales fueron creadas.

Se observa, por ejemplo, que todas las características creadas a partir de los índices de sombra se ubican, con este criterio, antes que el índice de sombra al mediodía (lugar 28) y la sombra a las 3pm (lugar 22). La diferencia entre sombra de las 3pm y las 12am incluso queda rankeada por encima de todos los índices de sombras, siendo la que más información aporta.

Asimismo, para el caso de las distancias a fuente de agua, se observa que todas las variables creadas se ubican por encima de *Vertical_Distance_To_Hydrology* (lugar 18), pero por debajo de *Horizontal_Distance_To_Hydrology* (lugar 9), siendo la que más aporta información la distancia euclideana a la fuente de agua (lugar 10).

4.2. Análisis de Discriminantes Lineales (LDA)

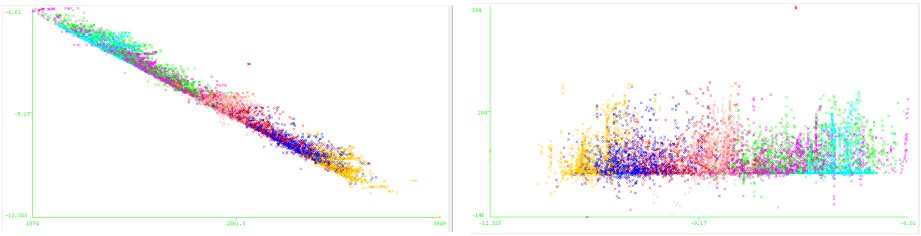
Como se observó en la sección 2.2, se buscó la extracción de una nueva característica en base a *Elevation* y *Vertical_Distance_To_Hydrology*, mediante LDA, a modo de lograr una mejor separabilidad entre clases. En este caso, se prioriza separar según la dirección que presentan los datos de las clases 1 y 2, pues, como se verá en secciones próximas, son las clases que más dificultades ocasionan, confundiéndose entre sí, a la hora del clasificado. Se obtiene entonces el hiperplano separador (en este caso, una recta) que mejor discrimina entre

1	Elevation
53	*Soil (conjunción de binarias)*
15	*Area (conjunción de binarias)*
14	Wilderness area 4
6	Horizontal distance to roadways
10	Horizontal distance to fire points
11	Wilderness area 1
23	Soil type 9
4	Horizontal distance to hydrology
54	*Distancia Euclideana a fuente de agua*
13	Wilderness area 3
55	*Suma de distancias a fuente de agua*
59	*Diferencia entre sombra de las 3pm y las 12am*
7	Hillshade 9am
18	Soil type 3
57	*Suma de los valores de sombras a diferentes horas*
56	*Multiplicación de distancias a fuente de agua*
5	Vertical distance to hydrology
50	Soil type 38
2	Aspect
51	Soil type 39
58	*Diferencia entre sombra de las 12am y las 9am*
9	Hillshade 3pm
3	Slope
41	Soil type 29
42	Soil type 30
52	Soil type 40
8	Hillshade Noon
35	Soil type 23
19	Soil type 4
12	Wilderness area 2
-	Restantes caracsterísticas del tipo Soil type

Cuadro 3: Características a utilizar ordenadas según su ganancia de información

ambas clases, y la nueva característica se conforma por la distancia ortogonal entre los datos originales y la recta separadora obtenida.

En la figura 9, se observa la visualización 2 a 2 de la nueva característica contra las características que fueron utilizadas para su creación. Se puede ver que la nueva variable presenta una notoria correlación con la característica *Elevation*. Al observarla junto con la característica *Vertical_Distance_To_Hydrology*, se aprecia que los datos se disponen de una forma que facilita la clasificación para algoritmos de tipo árbol.



(a) Visualización para las características Elevation (abscisas) y característica obtenida mediante LDA (ordenadas)

(b) Visualización para las características obtenida mediante LDA (abscisas) y Vertical_Distance_To_Hydrology. (ordenadas)

Figura 9: Visualización de características dos a dos, para observar los efectos de la extracción mediante LDA.

5. Clasificación

5.1. Primera aproximación

Se decide comenzar evaluando el desempeño de algunos clasificadores clásicos sin agregar nuevas características, utilizando un set de datos sin outliers y sin las características que evidentemente no aportan información. De este modo se obtiene un punto de partida contra el cual comparar el desempeño posteriormente. De aquí en más, salvo que se indique explícitamente lo contrario, todas las verificaciones con el conjunto de de datos de entrenamiento se realizaron con validación cruzada en 10 particiones, utilizando la semilla randómica para la partición que utiliza `Weka` por defecto.

Se probaron los algorimos: *Naive Bayes*, *árbol C4.5* (implementación J48 de `Weka`) y *vecinos más cercanos* (implementación IBK de `Weka`), se variaron los parámetros de cada clasificador buscando maximizar el desempeño. Se utilizaron, además, métodos automáticos de búsqueda de parámetros óptimos.

En el caso de *Naive Bayes* se utilizó con la aproximación por Gaussianas y por Kernel (ver cuadro 4), para *C4.5* se ajustó el valor de confianza (ver cuadro 5) y para IBK se ajustó el numero de vecinos utilizando la opción *CrossValidation*. Notar que para el caso de *IBK*, la opción *CrossValidation* arroja que el resultado óptimo se obtiene utilizando *Vecino más cercano 1-NN* (ver cuadro 6).

Naive Bayes	Desempeño (%)
Gaussiano	65.9668
Con Kernel	**69.3156**

Cuadro 4: Desempeño para Naive Bayes

5.2. Optimización de IBK

Se decide avanzar sobre *Vecinos más cercanos* para ver qué desempeño puede alcanzar este algoritmo en el problema en estudio. Para esto, se utilizan distintos sets de características, incluyendo como nuevas características las que

C4.5	Factor de Confianza	Desempeño (%)
	0,05	76,9037
	0.1	**76,9834**
	0.15	76,8106
	0.2	76,8904
	0.3	76,9037
	0.4	76,8771
	0.5	76,7841

Cuadro 5: Desempeño para árboles con algoritmo C4.5

IBK	Desempeño (%)
1-NN	**78.6844**

Cuadro 6: Desempeño para IBK

se mostraron en la sección 3.

Para un nuevo set de datos, cuya diferencia con el original radica en sustituir las características binarias por dos variables nominales (una para tipos de suelo y otra para áreas), se buscó el punto óptimo entre cantidad de vecinos y cantidad de características a utilizar por *IBK*. Utilizando el criterio de ganancia de información para rankear las características se obtuvieron los resultados mostrados en el cuadro 7.

IBK	Características consideradas	Desempeño (%)
1-NN	7	81.4352
1-NN	8	**81.8206**
1-NN	9	81.0897
1-NN	10	79.8538

Cuadro 7: Desempeño para IBK, variando cantidad de características a considerar, ordenadas por criterio de ganancia de información

La combinación de las características binarias en una sola nominal no tuvo ningún efecto sobre el desempeño del clasificador al evaluar utilizando todo el conjunto de características ya que internamente, al ser nominales, `Weka` las trabaja como diferentes variables binarias. Sin embargo, utilizando las nuevas variables combinadas, el efecto de tomar sólo las 8 características que más información aportan sí fue significativo en el desempeño de IBK.

En la matriz de confusión de la figura 10 podemos ver cómo este algoritmo, para este set de datos, comete la mayor cantidad de errores entre las clases 1 y 2, y entre las clases 3 y 6.

```
=== Confusion Matrix ===

   a    b    c    d    e    f    g   <-- classified as
 809  169    2    0   29    6   80 |    a = 1
 203  494   21    0  120   40   13 |    b = 2
   0   15  839   62   19  205    0 |    c = 3
   0    0   25 1083    0   22    0 |    d = 4
  13   42   11    0  999    8    0 |    e = 5
   5   17  157   38   11  924    0 |    f = 6
  31    4    0    0    0    0 1009 |    g = 7
```

Figura 10: Matriz de Confusión 1-NN

5.3. Random Forest

Con el objetivo de alcanzar nuevos desempeños se decide utilizar un algoritmo de clasificación más complejo, como es el caso de *Random Forest*.

Con las características provistas originalmente, se probó el desempeño de este algoritmo utilizando los parámetros por defecto (100 arboles de a 6 características) obteniendo resultados notoriamente superiores a los anteriores (ver cuadro 8). Además, en la figura 11, puede observarse la matriz de confusión obtenida al aplicar el clasificador. Puede verse, que para este clasificador las confusiones mayores siguen dándose entre las mismas clases que se mencionaron anteriormente: 1 y 2, y, 3 y 6.

Random Forest	Desempeño (%)
100 árboles, 6 características	**83.6944**

Cuadro 8: Desempeño para Random Forest, características originales

```
=== Confusion Matrix ===

   a    b    c    d    e    f    g   <-- classified as
 851  123    2    0   37    9   84 |    a = 1
 219  525   28    0   93   37    8 |    b = 2
   0    5  875   75   14  172    0 |    c = 3
   0    0   14 1101    0   15    0 |    d = 4
   4   28   16    0 1025   14    0 |    e = 5
   2    5  143   40   10  953    0 |    f = 6
  40    3    0    0    1    0 1041 |    g = 7
```

Figura 11: Matriz de Confusión para el algoritmo Random Forest

5.3.1. Evaluación con características creadas

Se observa qué sucede si se utiliza el clasificador considerando la totalidad de las características (originales y creadas), mencionadas en la sección 3.3. En el cuadro 9, se muestra el desempeño obtenido.

Random Forest	Desempeño (%)
100T, 6C	81.7807

Cuadro 9: Desempeño para Random Forest, para la totalidad de las características

Se observa que el desempeño del clasificador disminuye notoriamente al utilizar la totalidad de las características disponibles, por lo tanto se busca, para los parámetros por defecto del clasificador, encontrar qué conjunto de características contribuye a lograr una mejor clasificación.

Se prueba qué sucede al eliminar las variables binarias del espacio de características (cuadro 10). Se observa una mejoría respecto al desempeño obtenido utilizando todas las características, pero no se supera el obtenido con las características originales.

Random Forest	Desempeño (%)
100T, 6C	83.402

Cuadro 10: Desempeño para Random Forest, quitando las características binarias

Luego se utilizan las características binarias originales y no las nominales creadas, obteniéndose el desempeño mostrado en el cuadro 11.

Random Forest	Desempeño (%)
100 árboles, 6 características	82.1661

Cuadro 11: Desempeño para Random Forest, quitando las características nominales de agrupación de binarias

Por ser este resultado peor que el anterior, y por ser las características binarias y las nominales creadas a partir de éstas, de cierto modo redundantes, se prosigue utilizando sólo las características nominales creadas.

Para este subconjunto de características (sin las variables binarias), se intenta establecer para qué conjunto de características se obtiene el mejor desempeño.

Se comienza, ordenando las características en un ranking por el criterio de ganancia de información, y se utilizan diferentes números de características.

Debe tenerse en cuenta que el ranking de ganancia de información no evalúa cómo trabajan los características en conjunto, sino individualmente; existen correlaciones entre características que pueden ser determinantes en el desempeño final. Aún así, con este método de selección computacionalmente sencillo, se obtienen mejorías notorias en el desempeño (ver cuadro 12).

Características consideradas	Desempeño (%)
5	80.5581
6	82.4850
7	82.8704
8	83.0432
9	83.3488
10	83.5615
11	83.4153
12	83.8007
13	83.6146
14	83.7874
15	83.7475
16	**83.8937**
17	83.6146

Cuadro 12: Desempeño para Random Forest, con parámetros por defecto, variando cantidad de características a considerar, ordenadas por criterio de ganancia de información

De todos modos, siguiendo con la línea anterior de pensamiento (en la cual se eliminaron las variables binarias, por ser redundantes), se analizan las posibles redundancias que existen entre las características que fueron creadas y las que existían anteriormente. Las distancias se obtuvieron a partir de dos medidas de distancia independientes, por lo cual sería razonable elegir las dos características de distancia que aporten más información, esto es, quedarse con *Horizontal_Distance_To_Hydrology* y la distancia euclideana a fuentes de agua, descartando las demás. Análogamente, para el caso de índices de sombra, se debería trabajar con la diferencia entre sombra de las 3pm y las 12am, *Hillshade_9am* y la suma de los valores de sombras a diferentes horas.

Para el set de características el desempeño es el reflejado en el cuadro 13.

RandomForest	Desempeño (%)
100 árboles, 6 características	83.3621

Cuadro 13: Desempeño para Random Forest, con características no redundantes

Se concluye que el criterio tomado para descartar características por correlaciones deducidas a partir de su creación no es bueno. Otros métodos de selección que evalúen el desempeño de conjuntos de características serían más apropiados.

5.3.2. Evaluación con LDA

Se quiere ahora analizar qué efecto tiene considerar la característica creada por extracción mediante LDA, a partir de las características *Elevation* y *Vertical_Distance_To_Hydrology*. En primera instancia se observa el desmpeño del clasificador para la totalidad de características, incluyendo LDA (cuadro 14).

Quitando las variables binarias, a modo de repetir el procedimiento de la parte anterior, se obtiene lo reflejado en el cuadro 15.

RandomForest	Desempeño (%)
100 árboles, 6 características	83.2425

Cuadro 14: Desempeño para Random Forest, agregando la totalidad de las variables creadas

Random Forest	Desempeño (%)
100 árboles, 6 características	**83.8538**

Cuadro 15: Desempeño para Random Forest, quitando las características binarias

Luego, se repite el razonamiento de filtrado por ganancia de información, considerando el total de las características menos las binarias. Es pertinente aclarar que en el nuevo ranking de ganancia de información, la característica creada mediante LDA se ubica en el segundo lugar, luego de *Elevation*, lo cual tiene sentido ya que ambas variables presentan una correlación notoria, como se analizó previamente. Los resultados, según la cantidad de características elegidas para clasificar, se muestran en el cuadro 16.

Características consideradas	Desempeño (%)
5	80.0000
6	81.8770
7	83.4419
8	83.4817
9	83.5748
10	83.9336
11	83.7874
12	83.8671
13	83.9203
14	83.8937
15	84.1860
16	**84.2525**
17	84.1728
18	84.1329
19	84.0399

Cuadro 16: Desempeño para Random Forest, con parámetros por defecto, variando cantidad de características a considerar, ordenadas por criterio de ganancia de información

5.3.3. Evaluación de set de datos originales y LDA.

Dado que la característica obtenida mediante LDA aparece como muy importante en cuanto a aporte de información, se planteó el uso del set de características originales agregando solamente la obtenida mediante la extracción por LDA. Para los parámetros por defecto de *Random Forest*, se obtiene lo visto en el cuadro 17. Se considera el resultado como muy prometedor, teniendo un mejor desempeño que en todos los casos anteriores, y no habiendo hecho selección

de características ni optimizado parámetros del clasificador aún.

RandomForest	Desempeño (%)
100 árboles, 6 características	**84.2924**

Cuadro 17: Desempeño para Random Forest, con características originales y la creada a mediante LDA

Se realizó selección de características, utilizando el rankeo de ganancia de información, y eliminando las tres características numéricas menos relevantes. Esto puede verse en los cuadros 18, 19 y 20.

Random Forest	Desempeño (%)
100 árboles, 6 características	**84.7851**

Cuadro 18: Desempeño para Random Forest, con características originales y la creada a mediante LDA, quitando Hillshade_Noon.

Random Forest	Desempeño (%)
100 árboles, 6 caracerísticas	84.6512

Cuadro 19: Desempeño para Random Forest, con características originales y la creada a mediante LDA, quitando Hillshade_Noon y Slope.

Random Forest	Desempeño (%)
100 árboles, 6 caracerísticas	**84.8638**

Cuadro 20: Desempeño para Random Forest, con características originales y la creada a mediante LDA, quitando Hillshade_Noon, Slope y Hillshade_3pm.

Se observa como, sin optimizar los parámetros del clasificador, ya se obtiene un resultado mucho mejor que el inicial, considerando el caso óptimo el mostrado en la figura 20. Por esta razón, para las partes siguientes se utilizó el conjunto de características original, quitando *Hillshade_Noon*, *Slope* y *Hillshade_3pm*, y agregando la nueva variable creada mediante LDA.

5.3.4. Optimización de parámetros

Se utilizó el clasificador junto con la función meta *gridSearch* de `Weka`, a modo de obtener los parámetros para un desempeño óptimo, variando los parámetros de cantidad de árboles y cantidad de características a utilizar al crearlos.

A modo de disminuir el costo computancional que impone el uso de *grid-Search*, se utilizó la opción de realizar la búsqueda en la grilla considerando el 10 % de las muestras.

Se realizó en primera instancia una búsqueda entre 90 y 140 árboles (con paso 10), y entre 3 y 7 características (con paso 1); esto resultó en un óptimo de

140 árboles y 5 características. Al observar que el óptimo en cuanto a cantidad de árboles coincidió con el límite impuesto, se realizó una segunda corrida, esta vez con los límites entre 130 y 190 árboles, y 4 y 9 características; esto arrojó como ideal el uso de 150 árboles y 5 características.

En el cuadro 21 se muestran los desempeños para distintos parámetros del clasificador.

Random Forest	Desempeño (%)
140 árboles, 5 características	85.1163
150 árboles, 5 características	**85.1960**
170 árboles, 5 características	85.1429
140 árboles, 4 características	85.1030
150 árboles, 4 características	85.1429
170 árboles, 4 características	85.1030

Cuadro 21: Desempeño para Random Forest, utilizando el conjunto de características considerado óptimo, y variando parámetros del clasificador

Se considera entonces, que los parámetros óptimos hallados para el clasificador son 150 árboles y 5 características.

6. Combinación de Clasificadores

En esta sección se exploran algunas alternativas para mejorar los resultados obtenidos con *Random Forest*. Se utilizan diferentes criterios de combinación con el objetivo de mejorar la clasificación en las zonas donde se comenten mayor cantidad de errores. En primer lugar se busca ponderar los errores cometidos por *Random Forest* a través del método *Boosting* en sucesivas iteraciones, y en segundo lugar se hará un abordaje estudiando las matrices de confusión para generar una cascada de clasificadores.

6.1. Boosting

Con el objetivo de lograr superar el desempeño de clasificador se utilizará en este punto la implementación del método de *boosting* de `Weka` *AdaBoostM1*. Siendo que la evaluación es hecha sobre particiones del conjunto de entrenamiento y que en este punto se ponderan los errores cometidos en dicho conjunto, es probable que se pueda dar un sobreajuste si nuevos datos no se comportan de igual manera. En el cuadro 22 se muestra el resultado de aplicar boosting a *Random Forest* con los parámetros óptimos obtenidos en la parte anterior.

Se observa que no necesariamente los parámetros óptimos para *Random Forest* coinciden con los óptimos para el clasificador con *AdaBoost*. Se realiza una nueva búsqueda de parámetros óptimos, siendo el mejor resultado el del cuadro 23.

AdaBoostM1	Desempeño (%)
Random Forest 150 árboles 5 características	85.2625

Cuadro 22: Desempeño de AdaBoostM1 sobre clasificador optimizado

AdaBoostM1	Desempeño (%)
Random Forest 172 árboles 4 características	**85.4751**

Cuadro 23: Desempeño de AdaBoostM1 sobre clasificador optimizado

6.2. Cascada de Clasificadores

Luego de analizar las matrices de confusión y ver que los diferentes algoritmos utilizados a lo largo del proyecto comenten un alto porcentaje de errores entre las clases 1-2 y 3-6, se descarta la posibilidad de combinar por *Voting* y se decide probar una clasificación en cascada.

Se arma un nuevo set de datos en Matlab combinando dichas clases y teniendo entonces sólo 5 etiquetas para todos los datos. Luego de clasificar este primer set se crean 2 sets de datos más, con los datos bien clasificados de las clases combinadas 1-2 y 3-6. Estos nuevos sets son clasificados nuevamente contrastando la clasificación obtenida contra sus etiquetas originales.

6.2.1. Primera etapa

Los resultados de la primer clasificación sobre 5 clases pueden verse en la tabla 24.

RandomForest	Desempeño (%)
172 árboles, 4 características	93.515 %

Cuadro 24: Desempeño de Random Forest,sobre set de 5 clases

```
=== Confusion Matrix ===

    a    b    c    d    e   <-- classified as
 1738   78    0   97   73 |    a = 1
   10 2204   64   14    0 |    b = 3
    0   40 1090    0    0 |    c = 4
   35   28    0 1010    0 |    d = 5
   49    0    0    0  995 |    e = 7
```

Figura 12: Matriz de Confusión de Random Forest sobre 5 clases

En la matriz de confusión de la figura 12 se ve como las nuevas clases a y b tienen más patrones debido a que son resultado de agrupar dos clases originales.

Los porcentajes de acierto en clasificación dentro de las nuevas clases 1-2 y 3-6 se ven en el cuadro 25, así como la cantidad de muestras acertadas y erradas.

Clases	Desempeño (%)	Cant. Aciertos	Cant. Errores
1-2	87.513	1738	248
3-6	96.161	2204	88

Cuadro 25: Desempeño para clases combinadas

En la figura 13, se muestran los datos, pertenecientes a las 7 clases, con las predicciones obtenidas al clasificar en 5 clases. A modo de ejemplo, en rojo y azul se ven las clases 1 y 2, y se puede ver cómo aparecen patrones de clase 1 en la clase 7 (5ta columna) y patrones de la clase 2 en las clases 3 y 6.(2da y cuarta columnas de la figura).

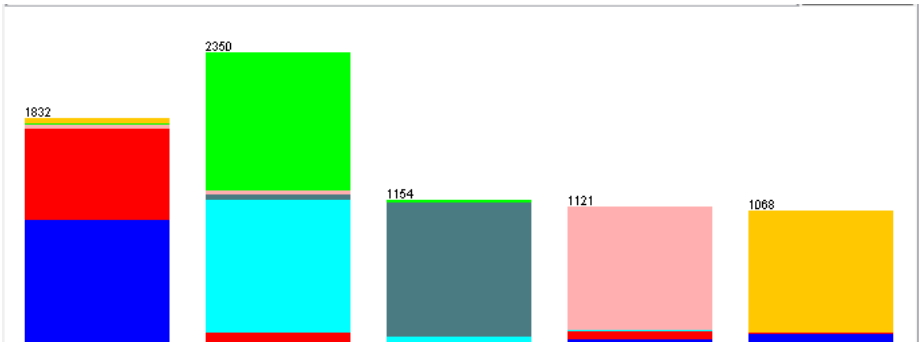


Figura 13: Distribución de datos clasificados en 5 clases, con el color de su clase original

6.2.2. Segunda etapa

Se analizan las características para el sub-set de los patrones de las clases 1 y 2 bien clasificados en la etapa anterior, detectándose 10 variables binarias sin aporte de información, las cuales no son tenidas en cuenta.
Sobre dicho sub-set se vuelve a correr el clasificador *Random Forest* con optimización de parámetros mediante *gridSearch* obteniendo el resultado del cuadro 26.

Random Forest	Desempeño (%)
200 árboles, 3 características	79.4591

Cuadro 26: Desempeño de Random Forest,sobre set de clases 1 y 2 bien clasificados en etapa 1

De forma análoga se analizan las características para el sub-set de los patrones de las clases 3 y 6 bien clasificados en la etapa anterior, y se detectan 20 variables binarias sin aporte de información, las cuales son removidas. Sobre el nuevo sub-set se corre el clasificador *Random Forest* con optimización de

```
=== Confusion Matrix ===

   a   b   <-- classified as
 847 150 |   a = 1
 207 534 |   b = 2
```

Figura 14: Matriz de confusión para clases 1 y 2

parámetros mediante *gridSearch* obteniendo el resultado del cuadro 27.

Random Forest	Desempeño (%)
180 árboles, 3 características	87.069

Cuadro 27: Desempeño de Random Forest,sobre set de clases 3 y 6 bien clasificados en etapa 1

```
=== Confusion Matrix ===

   a   b   <-- classified as
 928 152 |   a = 3
 133 991 |   b = 6
```

Figura 15: Matriz de confusión para clases 1 y 2

6.2.3. Combinación de resultados

El resultado final de la clasificación está dado por la cascada de aciertos según se muestra en el cuadro 28. Allí se puede ver que el resultado obtenido es prácticamente el mismo que el que se obtuvo sin combinación de clasificadores. Esto se podría deber al hecho de no haber encontrado un clasificador que separe mejor entre clases conflictivas para los sub-sets creados. Cabe mencionar que, sin eliminar las características binarias que no aportaban información en estos casos, el resultado era exactamente el mismo, lo cual en principio no era esperable por el hecho de utilizar diferentes parámetros.

7. Desempeño del SRP sobre el conjunto de test

El set de datos de Test cuenta con 7049 patrones distribuidos de forma equiprobable entre las 7 clases, con 1007 patrones por clase de árbol. El archivo *Recpat_2015_Test.arff* fue modificado para eliminar las características descartadas por el modelo tomado anteriormente y para agregar la característica nueva

Clases	Etapa 1 %	Cant. Aciertos	Etapa 2 %	Cant. Aciertos
1	87.513	1738	84.9548	847
2	clase 1		72.0648	534
3	96.161	2204	86.2037	931
4	96.4601	1090		(1090)
5	94.1286	1010		(1010)
6	clase 3		87.9004	988
7	95.3065	995		(995)
Resultado final	95.515	7037	**84.9834**	6395

Cuadro 28: Desempeño por clases de cascada de Random Forest (total de patrones 7525)

a partir de extracción mediate LDA.

Al evaluar el sistema de reconocimiento de patrones obtenido previamente con el conjunto de datos de test se ve un claro deterioro respecto al desempeño obtenido con *CrossValidation* en el conjunto de entrenamiento (cuadro 29).

AdaBoostM1	Desempeño con datos de Test
Random Forest 172 árboles 4 características	**78.6495**

Cuadro 29: Desempeño de SRP sobre datos de Test

```
=== Confusion Matrix ===

   a   b   c   d   e   f   g   <-- classified as
 844  61   0   0   1   0 101 |   a = 1
 430 493  15   0  42  13  14 |   b = 2
   6  28 731  34  12 196   0 |   c = 3
   0   2  72 898   0  35   0 |   d = 4
 113  88  12   0 780   9   5 |   e = 5
  27  25 112  24   5 814   0 |   f = 6
  22   1   0   0   0   0 984 |   g = 7
```

Figura 16: Matriz de confusión de SRP sobre datos de test

Claramente hay un sobreajuste a los datos de entrenamiento. Para ver si esto se debe al *Boosting*, se evalúa nuevamente sin *AdaBoost* (cuadro 30).

Random Forest	Desempeño con datos de Test (%)
172 árboles 4 características	**77.0038**

Cuadro 30: Desempeño de SRP sobre datos de Test sin Boosting

Al decrecer el desempeño sin *Boosting* se podría inferir que el efecto de sobreajuste no se debe al uso de *Adaboost*.

Es probable también que al hacer el análisis del impacto del desbalance en la sección 3.2, se haya subestimado el efecto que podría llegar a generar luego de que se ajustaran todos los parámetros de SRP. Para ver esta influencia se realiza un balance de datos utilizando el método SMOTE (ya mencionado en la sección 3.2) sobre la clase 2 con un 20 %, dejando el set de datos de entrenamiento más balanceado. El resultado obtenido es algo mejor, como se muestra en el cuadro 31 y la matriz de confusión en la figura 17.

```
=== Confusion Matrix ===

   a   b   c   d   e   f   g   <-- classified as
 828  81   0   0   2   0  96 |   a = 1
 356 571  12   0  43   9  16 |   b = 2
   3  43 754  28  16 163   0 |   c = 3
   0   0  94 882   0  31   0 |   d = 4
  78 109  10   0 798   3   9 |   e = 5
  13  43 151  19  19 762   0 |   f = 6
  20   1   0   0   0   0 986 |   g = 7
```

Figura 17: Matriz de confusión sobre datos de test con entrenamiento semi-balanceado

AdaBoostM1	Desempeño con datos de Test (%)
Random Forest 172 árboles 4 características	**79.1744**

Cuadro 31: Desempeño de SRP sobre datos de Test con datos de entrenamiento semi-balanceado

8. Conclusiones

Luego del estudio realizado, se encuentra que *Random Forest* con *AdaBoost* fue el mejor sistema de clasificación, considerando los datos de entrenamiento.

Resultó evidente la utilidad de seguir una metodología coherente a lo largo del problema, siendo fundamentales las etapas de análisis de datos y preprocesamiento de los mismos.

A su vez, una de las mejoras importates en el desempeño del clasificador elegido se obtuvo gracias a la extracción de una nueva característica mediante LDA, no resultando tan útiles otras características creadas. Otras mejoras se atribuyen a la selección de caracteristicas mediante rankeo y a la optimización de parámetros, demostrando que *gridSearch* es una herramienta fundamental.

Si bien el método de selección de características utilizado fue útil, éste no asegura que se haya seleccionado el mejor grupo de características, dejando lu-

gar a mejoras en ese aspecto.

Se exploraron métodos de combinación de clasificadores en cascada, sin lograr aumentar el desempeño del sistema. Esto se podría atribuir a que no se exploraron otros clasificadores que podrían distinguir mejor entre clases conflictivas. Al utilizar combinación por método de *boosting* se logró una leve mejoría sobre el desempeño en el conjunto de entrenamiento.

Finalmente, al evaluar el sistema obtenido con el conjunto de test, se obtiene un desempeño sensiblemente peor. Se hace evidente el sobreajuste del modelo a los datos de entrenamiento. Se comprueba que el sobreajuste no se debe al uso del *boosting*, y que el haber usado un conjunto de entrenamiento desbalanceado tiene un efecto negativo en el modelado del sistema.

9. Bibliografía

[1] Richard O. Duda, Peter E. Hart, David G. Stork, "Pattern Classification, 2nd Edition", 2000. ISBN: 978-0-471-05669-0.

[2] Ludmila I. Kuncheva, "Combining Pattern Classifiers, Methods and algorithms", 2004, ISBN: 0-471-21078-1

[3] Christopher M. Bishop, "Pattern Recognition and Machine Learning", 2006, ISBN-13: 978-0387-31073-2.