Práctica 5

Minería de datos

D. Ramirez Ambrosi

7 de diciembre de 2014

Índice

1. Módulos implementados

He implementado dos de los tres módulos diseñados al inicio de la práctica. Los módulos en cuestión son la generación de predicciones y la unión de clasificadores.

1.1. Módulo de generación de predicciones

Este módulo se ha ampliado en funcionalidad respecto al diseño inicial.

Inicialmente, dado un modelo y un test compatible, se encargaba de obtener un fichero en el que la primera línea contiene los valores posibles de la clase y en las posteriores, para cada instancia, contenía la probabilidad de pertenencia a cada una de las clases posibles. El aspecto de este fichero es el siguiente:

```
V1;V2
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
0.0019117847876553326;0.9980882152123447
...
```

La funcionalidad añadida consiste en generar un fichero extra. Este fichero contiene para cada instancia la clase predicha por el modelo cargado. El aspecto de este fichero es el siguiente:

```
V2
V2
V2
V2
V2
V2
V2
V2
V2
V2
V2
V2
V2
...
```

1.1.1. Implementación

La implementación de esta funcionalidad se ha realizado en la clase *GeneracionPredicciones.java* del módulo *pack.datamining.modules.main* y *PrediccionProbabilidad.java* del *módulo pack.datamining.modules.evaluation.*

1.1.2. Pruebas

Se ha comprobado el funcionamiento en los siguientes casos:

- Se ha creado un modelo oneR con la gui de Weka con los datos de entrenamiento sin preprocesar. El programa genera correctamente ambos ficheros de predicciones.
- Se ha creado un modelo j48 con la gui de Weka, habiendo eliminado previamente varios atributos al azar del fichero de entrenamiento. El programa genera correctamente ambos ficheros de predicciones proporcionando el test con todos los atributos.
- Se ha utilizado un modelo SVM con kernel polinomial de la carpeta de modelos del repositorio. El programa falló al proporcionar un test no preprocesado. En este caso resulta necesario usar un test preparado con el preproceso aplicado al conjunto de entrenamiento.

Junto al ejecutable se proporciona un caso de prueba correcto.

1.1.3. Observaciones

A la hora de cargar los ficheros .model, es necesario usar la misma versión de librerías que las usadas para guardar los modelos. De esta forma, las siguientes pruebas fallaron:

- Carga de un modelo creado con la gui de Weka 3.6, usando la versión 3.7 a la hora de programar.
- Carga del modelo SVM con kernel polinomial. En este caso se descubrió que entre compañeros usábamos versiones distintas de la librería LibSVM y se corrigió.

Por lo tanto se concluye que este módulo solo funcionará con los modelos creados con la versión 3.7 de Weka y la versión 1.6 de LibSVM.

1.2. Módulo de unión de clasificadores

Implementado de acuerdo a las especificaciones realizadas inicialmente.

Tal y como se especifica en el readme correspondiente, el módulo precisa un fichero de configuración cuyo contenido tiene el siguiente aspecto:

```
/home/david/Escritorio/test.e2e.wOOV.obfuscated-pred-prob.txt;0.5
/home/david/Escritorio/test.e2e.wOOV.obfuscated-pred-prob-j48.txt;0.5
```

1.2.1. Implementación

La implementación del módulo se ha llevado a cabo en la clase *UnionClasificadoresPV.java* del módulo *pack.datamining.modules.main.*

El código precisa de refactoring para modularizarlo y extraer diversos métodos.

1.2.2. Situaciones excepcionales

A la hora de programar el módulo se han tenido en cuenta las siguientes situaciones excepcionales que impiden el correcto funcionamiento del programa y fuerzan su cierre. En caso de fallo, se procura informar dónde se ha hallado para facilitar al usuario la corrección. Además de las siguientes, también podrían darse situaciones que no se han tenido en cuenta a la hora de programar.

- Fichero de configuración no encontrado.
- Uno de los ficheros de predicciones no se encuentra.
- Las ponderaciones no están expresadas en formato *double*.
- Las ponderaciones asignadas a los modelos no suman 1.
- Los ficheros de predicciones especificados contienen diferente número de clases.
- Los ficheros de predicciones especificados contienen clases (etiquetas) diferentes.
- Alguna de las líneas de los ficheros de predicciones contiene un número de elementos diferente al número de clases del problema.
- Alguno de los valores del fichero de predicciones no está expresado en *double*.
- Alguno de los valores del fichero de predicciones no es una probabilidad.
- El programa se llama sin parámetros.

1.2.3. Pruebas

Proporcionando un archivo de configuración correcto preparado para la unión de dos ficheros de predicciones correctos, el programa genera los dos ficheros esperados.

Con pequeñas variaciones en los ficheros de prueba, se ha comprobado el funcionamiento de algunas de las excepciones.

Junto al ejecutable se proporciona un caso de prueba correcto.

1.2.4. Capacidades

La combinación de este módulo con el anterior permite preparar la predicción de test usando el método de combinación en paralelo por votación. Además, la forma en que está programado permite trabajar con problemas en que la clase es capaz de tomar más de 2 valores. Se ha procurado no sobreajustar el módulo al problema de dos clases dado.

Otra posibilidad con este módulo es realizar varias uniones de clasificadores y volver a unir las predicciones resultantes gracias a que uno de los ficheros de resultados es igual a los ficheros de entrada.

1.3. Módulo de estimación de parámetros para la unión

Finalmente no ha llegado a implementarse por falta de dedicación de tiempo a esta práctica.

2. Repositorio de trabajo

El repositorio es accesible mediante la siguiente dirección:

https://github.com/spolex/datamining-modules.git

En él se encuentra el código fuente de todos los módulos desarrollados en grupo, así como la *javaDoc* de las clases implementadas y los ejecutables. En esta carpeta de ejecutables se encuentran los *jar* y sus correspondientes *readme*. En la carpeta correspondiente a cada uno de los integrantes se encuentran también los informes individuales de la práctica.

En la carpeta *modelos* se encuentran algunos de los modelo óptimos creados tras los barridos serializados en ficheros *.model* y sus evaluaciones no-honesta, hold-out y 10-Fold CV.