Informe del progreso de TFG, fase 2 (2017.10.26-2017.11.16)

Siguiendo el hilo del progreso, se han de solucionar algunos problemas que se observaron en la cita anterior, abajo se presentan éstos con las respuestas:

1. **Remplazo de los valores "NaN" por 5.0, no es razonable.**

   Se ha eliminado este reemplazo, como no es razonable. Pero estos valores de NaN dan problemas a la hora de predecir con el "RandomForestRegressor", porque no puede tratarlos. El "RandomForestRegressor" es para la predicción de calificaciones, que no es nuestro objetivo, sirve como una comparativa respecto al caso de predicción de Ranking. Como no es tan importante, dejamos esta cuestión para más adelante.

   En el caso de Ranking, la solución es directa. A la hora de convertir las calificaciones en los valores del ranking, las que son NaNs se convertirán en los pésimos rankings. Por ejemplo, sea [asig1: 8.0, asig2: NaN, asig3: 4.5, asig4: NaN, asig5: 7.8], la conversión da este resultado: [asig1: 1, asig2: 4, asig3: 3, asig4: 5, asig5: 2]. Así, todas las asignaturas tienen un valor, pero el problema es que cuando hay varias asignaturas con el valor NaN, no se sabe cuál es mejor.

2. **La función usada para calcular la puntuación del "10-fold cross validation" podría no ser correcta y hay que revisarla.**

   He estudiado este problema (mirando la documentación de sklearn), y he visto que se puede obtener la puntuación del cv mediante tres vías:

   a. cross_val_score(clf, X, y, cv=10): es muy directo, ya que al pasarle los parámetros necesarios, te da la puntuación. El cv=10 indica que se harán 10 validaciones, por tanto, la porción de test-set es de un décimo en cada validación; cuando cv=integer, por defecto se usa el KFold como el iterador del cross validation. En cuanto al "loss-function", por defecto esta función emplea el método "score" del estimador "clf".
   b. kf = KFold(n_splits=10, random_state=0):
   c. cross_validate: hace lo mismo que la "cross_val_score", pero permite espeficar múltiples métricas para la evaluación y además del test_score, retorna también el training_scores, fit_times y scores_times.

   Las tres vías dan el mismo valor de test_score (test accuracy) y provienen del mismo paquete (sklearn.model_selection, sustitución del sklearn.cross_validation en la versión 0.20).

3. **La medida de validación usada para cada predicción podría no ser adecuada, hay que buscar una adecuada de forma que los resultados sean comparables.**

Anteriormente, para calcular la puntuación en los dos casos (RandomForestRegressor, RandomForestClassifier) se utilizaba principalmente la “metrics.accuracy_score”, y daba una puntuación muy baja, lo dejamos para más adelante.

4. **Hay que averiguar si el RF testea de forma independiente o conjunta las clases existentes. Si es el primer caso, diríamos que no se está clasificando bien, ya que se espera que entre las clases predichas haya una relación.**

   Lo dejamos para más adelante.

5. **Se observa que cuando se clasifica por el ranking, se da un resultado en que aparecen clases repetidas, por eso se puede decir que no es una clasificación estricta, la que necesitamos. Para ello la tutora propuso hacer un recomendador, el cual puede realizar la clasificación estricta, también solucionaría el problema 4.**

   Este problema es la tarea principal de esta fase, que consiste en la implementación de un recomendador basado en usuarios. La tarea se divide en siguientes parts:

   - Estudiar el código relacionado con el recomendador basado en usuarios de Sergi y adaptarlo en el propio código.
   - Una vez tenemos ya definida la clase “RecUserBased” con su estructura (los atributos y los métodos necesarios), se ha implementado un algoritmo que hace la predicción sobre el ranking de asignaturas. El hilo de proceso de la recomendación es: ……..

     Y por último se ha definido un método (loss function) para la evaluación de precisión del recomendador.
   - Se ha evaluado de varias maneras:
     - train_test_split: separa las muestras en training_set y testing_set en proporción de forma aleatoria.

DataFrame: id_assig

id_alumne	CDDV	ESAL	GELI	GRAF	MNU1	CIDV	GEPR	HIMA	MMSD	TOPO
259	2	0	5	1	4	2	2	2	4	0
361	0	7	3	2	2	1	1	5	5	0
34	5	5	0	0	1	1	6	1	5	0
660	0	2	5	0	2	8	5	2	2	4
326	2	4	5	1	2	1	4	1	4	0

Mean absolute error entre y_pred e y_test (y_true) de las cinco primera filas

```
refactored classes and functio
different from that of this mo
  "This module will be removed
Margin= 0
errors= 24.177777777777777
Margin= 1
errors= 23.77777777777778
Margin= 2
errors= 23.91111111111111
Margin= 3
errors= 23.91111111111111
Margin= 4
errors= 23.91111111111111
Margin= 5
errors= 23.91111111111111
Margin= 6
errors= 23.91111111111111
Margin= 7
errors= 23.91111111111111
Margin= 8
errors= 23.91111111111111
Margin= 9
errors= 23.91111111111111
final
```

Errores de la predicción, k=5 y diferentes valores de margin.

También se ha evaluado con las diferentes combinaciones de k (1-9) y margin (0-9), y la conclusión es:
Mejor puntuación: [23.78: 1,5; 6,6; 9,6]
Peor puntuación: 26.3: 0,2
El resto están entre 24 y 26

- cross validation con KFold=5: a diferencia de lo anterior, que hace 1 sola validación, esta hace 5 validaciones; el KFold reparte los índices para el training y testing de forma no aleatoria.

```
>>> list_errors
[26.0, 22.5, 23.90909090909091, 22.954545454545453, 25.227272727272727]
>>> mean(list_errors)
24.118181818181817
```

Lista de errores de cada validación y el valor promedio.

Notas:

1. Algunos detalles:
   a. Repetición de entrenamientos. Tanto si se ha entrenado la muestra o no, no afectará a model_selection.cross_val_score, el resultado saldrá lo mismo.
   b. Modificación del dataframe original:
      - train_test_split? No afecta, se supone que esta función hace una copia de X e y.

- La conversión (qual2rangking): hacer la copia antes de la conversión.
- Método “fit” de RecomUserBased: No afecta, ya que no se van a modificar los valores de X_train e y_train.

c. Se hallan valores de NaN en el resultado de predicción.
d. Se ha forzado la clasificación estricta.

2. Posibles mejoras:
   a. Reemplazo de NaN's por recomendación antes de la predicción? SI
   b. Predecir si un alumno va a matricularse en una asignatura? NO

3. Tareas para la siguiente cita:
   a. Revisar los problemas relacionados con el RandomForest
   b. Mejora del algoritmo de recomendación (modificación o implementación de otro)
   c. Clasificación de suspensos vía recomendación. Si te da tiempo, sino no pasa nada…