Universidad de San Andrés Noviembre de 2022

Big Data, Aprendizaje y Minería de Datos

# Trabajo Práctico N°4: Regularización aplicada a la EPH

Profesor: Walter Sosa Escudero
Asistentes: Belén Michel Torino y Matías Gómez Seeber

**Fecha de entrega:** Martes 15/11 a las 23:59.
**Contenidos:** Se profundizará el ejercicio de predicción de los hogares bajo la línea de pobreza para aquellos individuos que no reportan sus ingresos. Incluiremos los nuevos métodos vistos en clase para clasificar a estos individuos.

## Modalidad de Entrega

- Al finalizar el trabajo práctico deben hacer un último `commit` en su repositorio de GitHub con el mensaje "Entrega final del tp".
- Asegúrense de haber creado una carpeta llamada TP4. Su reporte (pdf) y código (jupyter notebook) deben estar dentro de esa carpeta.
- También deben completar el link de su repositorio -para que pueda ser clonado y corregido- en esta google sheet
- La última versión en el repositorio es la que será evaluada. Por lo que es importante que:
  - No completen la google sheet hasta no haber terminado y estar seguros de que han hecho el `commit` y `push` a la versión final que quieren entregar. Debido a que se pueden tomar hasta 3 días de extensión a lo largo del curso, no corregiremos sus tareas hasta no ver el link en la google sheet.
  - No hagan nuevos `push` después de haber entregado su versión final. Esto generaría confusión acerca de qué versión es la que quieren que se les corrija.

## Reglas de formato y presentación

- El trabajo se debe entregar como Jupyter Notebook. Este debe contener un boton para esconder los bloques de código. De esta forma, al corregir podremos optar por ver tando la versión con código como la versión 'limpia' donde solo esten sus bloques de texto con respuestas, graficos y tablas. La versión 'limpia' debe tener una extensión máxima de 10 páginas al imprimirse en PDF. **Para ello deben asegurarse de eliminar todos los prints innecesarios.**
- Todos los miembros del equipo deben haber hecho al menos un `commit` durante la producción del TP para asegurar que todos hayan aportado a su resolución.
- Se espera una buena redacción en la resolución del práctico.

## Parte I: Análisis de la base de hogares y cálculo de pobreza

Ahora que ya se han familiarizado con la Encuesta Permanente de Hogares (EPH) asegúrense de estar calculando niveles de pobreza a nivel individual y a nivel de hogar similares a los que calcula el INDEC para la misma región. Además, asegúrense de estar realizando una limpieza de la base con mayor dedicación y entendimiento de sus variables.

1. Descarguen la base de microdatos de la EPH correspondiente al primer trimestre de 2022 (la base de hogares se llama `usu_hogar_T122.xls`). Importen los datos de la encuesta de hogar y al igual que en el TP1 conserven sólo las observaciones que corresponden a los aglomerados de Ciudad Autónoma de Buenos Aires o del Gran Buenos Aires.
2. Unan la tabla de la encuesta individual con la de la encuesta de hogar. **Asegúrense de estar usando las variables** `CODUSU` **y** `NRO_HOGAR`**.**
3. Limpien la base de datos tomando criterios que hagan sentido, tanto para el tratamiento de los valores faltantes, de los *outliers*, como así también decidan qué variables categóricas y strings usarán y transfórmenlas de forma que haga sentido para los ejercicios siguientes. Justifiquen sus decisiones.
4. **Construyan variables (mínimo 2) que no estén en la base pero que sean relevantes a la hora de predecir individuos bajo la línea de pobreza (ej. proporción de mujeres (o niños) en el hogar, ¿su cónyuge trabaja?)**
5. **Sean creativos y presenten un gráfico (que no sea de barras) para describir la interacción o correlación entre dos o más variables.**
6. Construyan la columna `adulto_equiv` y la columna `ad_equiv_hogar` y luego dividan la base en dos dataframes donde: uno conserve las personas que reportaron ITF (llamada `respondieron`) y la otra conserve las personas que **no** reportaron ITF (llamada `norespondieron`). Además, agreguen a la base `respondieron` una columna llamada `ingreso_necesario` que sea el producto de la canasta básica por `ad_equiv_hogar`.
7. Agreguen a `respondieron` una columna llamada `pobre` que tome valor 1 si

el ITF es menor al `ingreso_necesario` que necesita esa familia, y 0 en caso contrario.

8. Para calcular la tasa de hogares bajo la linea de pobreza utilicen una sola observación por hogar y sumen el ponderador `PONDIH` que permite expandir la muestra de la EPH al total de la población que representa. ¿Cuál es la tasa de hogares bajo la línea de pobreza para el Gran Buenos Aires? ¿Lograron que se asemeje al % que reporta el INDEC?

## Parte II: Construcción de funciones

El objetivo de esta parte del trabajo es **revisar y mejorar** el código que escribieron en la parte II del TP3. Deben buscar que sea flexible y esté modularizado (en funciones bien documentadas con docstrings). De esta forma evitarán repetir código y podrán utilizarlo en distintos escenarios (como por ejemplo la Parte III de este TP y sus proyectos personales a futuro).

1. Escriban una función, llamada `evalua_metodo`, que reciba como argumentos un modelo y los datos de entrenamiento y prueba (`X_train, y_train, X_test, y_test`). La función debe ajustar el modelo con los datos de entrenamiento y calcular las métricas que considere necesarias para esta problemática **(de mínima debe reportar falsos positivos, falsos negativos, verdaderos positivos, verdaderos negativos, AUC, *accuracy* y *precision* de cada método)**. El output de la función debe ser una colección con las métricas evaluadas.
2. Escriban una función, llamada `cross_validation`, que realice validación cruzada con $k$ iteraciones ($k$-fold CV), llamando a la función del inciso anterior en cada una, pero para las $k$ distintas particiones. La función debe recibir como argumentos el modelo, el valor de $k$ y un dataset (es decir, sólo `X` e `y`[1]). Pueden ayudarse con la función KFold para generar las particiones necesarias.
3. Escriban una función, llamada `evalua_config` que reciba una lista de configuraciones de hiperparámetros (los distintos valores a probar como hiperparámetros podrían codificarse en diccionarios de Python) y utilizando la función `cross_validation` obtenga el error[2] promedio para cada configuración. Finalmente, la función debe devolver la configuración que genere menor error. **Asegúrense de que esta función sirva para cualquier hiperparámetro que quieran elegir por *cross validation* para cualquier modelo.**
4. Escriban una función, llamada `evalua_multiples_metodos` que les permita implementar los métodos que se enumeran a continuación. **Esta función debe utilizar su función `evalua_config` para optimizar los parámetros que ustedes decidan (de mínima deben optimizar el $K$ -cantidad de**

[1] **cuando usen esta función en la parte III del Tp asegurense de pasar X_train e y_train para no utilizar las observaciones de test en esta instancia de validación**

[2] utilicen la medición de error que prefieran

**vecinos- para el modelo de KNN).** Finalmente, el output de la función debe ser una tabla donde las columnas sean las métricas que hayan evaluado (las que hayan incluido en la función `evalua_metodo`) y las filas sean los modelos (con su configuración de hiperparámetros asociada) que hayan corrido[3]:

- Regresión logística
- Análisis de discriminante lineal
- KNN
- **Árbol de decisión**
- **Support vector machines (SVM)**
- **Bagging**
- **Random Forests**
- **Boosting**

## Parte III: Clasificación y Regularización

El objetivo de esta parte del trabajo es nuevamente intentar predecir si una persona es o no pobre utilizando datos distintos al ingreso, dado que muchos hogares son reacios a responder cuánto ganan.

1. Eliminen de ambas bases (`respondieron`, `norespondieron`) todas las variables relacionadas a ingresos (en el archivo `codigos_eph.pdf` ver las categorías: ingresos de la ocupación principal de los asalariados, ingresos de la ocupación principal, ingresos de otras ocupaciones, ingreso total individual, ingresos no laborales, ingreso total familiar, ingreso per cápita familiar). Elimine también las columnas `adulto_equiv`, `ad_equiv_hogar` e `ingreso_necesario`. Establezcan a `pobre` como su variable dependiente (vector y). El resto de las variables serán las variables independientes (matriz X).
2. Corran la función `evalua_multiples_metodos` con la base `respondieron`. **Asegurense de estar utilizando su función de** `evalua_config` **para optimizar algunos hiperparámetros (de mínima el K en el modelo KNN).**
3. ¿Cuál de todos los métodos evaluados predice mejor? ¿Con qué hiperparámetros? Justifiquen utilizando las medidas de precisión que conoce.
4. **¿Lograron mejorar sus predicciones respecto al TP3[4]?**
5. Con el método que seleccionaron, predigan qué personas son pobres dentro de la base `norespondieron`. ¿Qué proporción de los hogares son pobres en esa submuestra?

[3] **Pista:** Para los modelos que elijan optimizar sus hiperparámetros (ej. KNN), observen que deberán correr la función `evalua_metodo` dos veces. Una para optimizar los hiperparámetros (con el set de datos para train, que será dividido nuevamente en train y validación) y otra para obtener las métricas con el hiperparámetro óptimo (con un set de datos para train y otro para test)

[4] Si crearon variables relevantes en el ejercicio 4 parte I, se esperaría que sus modelos mejoren su predicción