Trabajo práctico 4: Regularización aplicada a la EPH

Integrantes: Catalina Banfi, Matías Lima, Santiago López

Lo primero que hacemos es crear un botón que permita esconder todas las celdas que sean código y mostrar todo lo demás si se lo desea. La solución no es original, sino que la obtuvimos de una provista por el usuario de Stack Overflow Eric Shepherd en el siguiente link: https://stackoverflow.com/a/53136940 (https://stackoverflow.com/a/53136940)

Show code

También sumamos el siguiente código para ocultar las warnings, entendiendo que las mismas son útiles pero molestas en el caso de una presentación cuando ya se entendió el motivo de las mismas. Por este motivo, también dejamos comentada una línea que permite limitar la cantidad de advertencias por acción a 1.

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

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.

Out[6]:

	CODUSU	ANO4	TRIMESTRE	NRO_HOGAR	COMPONENTE	H15	REGION
0	TQRMNOSUPHKKPQCDEIJAH00780151	2022	1	1	1	1	1
1	TQRMNOSUPHKKPQCDEIJAH00780151	2022	1	1	2	1	1
2	TQRMNOSUPHKKPQCDEIJAH00780151	2022	1	1	3	1	1
3	TQRMNOPQQHKMRLCDEIJAH00780169	2022	1	1	1	1	1
4	TQRMNOSXRHJMTRCDEIJAH00693084	2022	1	1	1	1	1
...	...	...	...	...	...	...	...
6701	TQRMNOTTSHKNLSCDEIIAD00780102	2022	1	1	2	1	1
6702	TQRMNOPRPHMNMLCDEIIAD00701192	2022	1	1	1	1	1
6703	TQRMNOPRPHMNMLCDEIIAD00701192	2022	1	1	2	1	1
6704	TQRMNOSRWHKMLHCDEIIAD00780103	2022	1	1	1	1	1

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.

Con el primer código lo que queremos es ver qué columnas tienen una gran cantidad de valores faltantes, para darnos una idea de cuál sería una cantidad justa para eliminar de la base. En base a eso, decidimos que columnas con más de un 50% de valores faltantes sean descartadas. Luego, lo que usamos es una función de numpy, .isna, que nos permite setear con condiciones qué columnas se van a eliminar.

```
Amount of missing values in -
CODUSU : 0%
ANO4 : 0%
TRIMESTRE : 0%
NRO_HOGAR : 0%
COMPONENTE : 0%
H15 : 0%
REGION : 0%
MAS_500 : 0%
AGLOMERADO : 0%
PONDERA : 0%
CH03 : 0%
CH04 : 0%
CH05 : 0%
CH06 : 0%
CH07 : 0%
CH08 : 0%
CH09 : 0%
CH10 : 0%
CH11 : 0%
```

Una vez hecha la limpieza, nos concentramos en transformar variables categóricas que creemos que son importantes (no observamos ningún string como importante por eso no hacemos nada con respecto a eso). Seleccionamos las variables que mencionamos como comentario a continuación porque creemos que son importantes para el análisis de la pobreza. Estas son: el tipo de vivienda, el material de los pisos, de dónde consigue agua el hogar, la fuente de agua, si tiene baño, dónde está el baño, si la vivienda está ubicada en la

proximidad de un basural, si la vivienda está ubicada en una villa de emergencia, el régimen de tenencia, el combustible utilizado para cocinar, el origen de los ingresos de los habitantes de la vivienda (salarios por trabajo, jubilaciones o pensiones, o subsidios o ayuda social), el decil de ingreso en el que se encuentra el hogar, si el individuo sabe leer y escribir, si asiste a un establecimiento educativo, qué tipo de establecimiento es, y, por último, cuál es el nivel educativo que cursa o el más alto que alcanzó.

Out[11]:

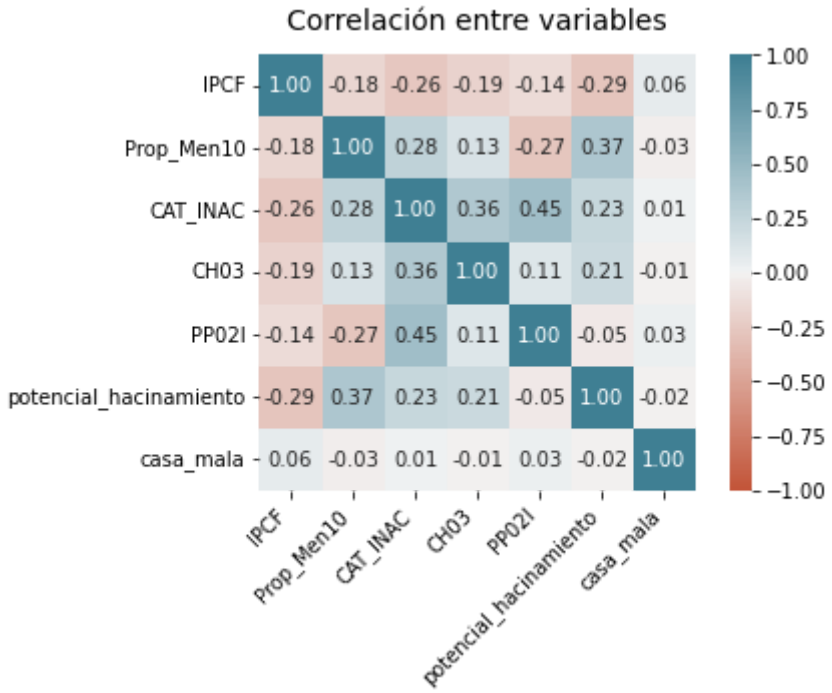
	CODUSU	ANO4	TRIMESTRE	NRO_HOGAR	COMPONENTE
0	TQRMNOSUPHKKPQCDEIJAH00780151	2022	1	1	2
1	TQRMNOPQQHKMRLCDEIJAH00780169	2022	1	1	1
2	TQRMNOSXRHJMTRCDEIJAH00693084	2022	1	1	1
3	TQRMNOSXRHJMTRCDEIJAH00693084	2022	1	1	3
4	TQRMNOSXXHKOKMCDEIJAH00780170	2022	1	1	1
...	...	...	...	...	...
5291	TQRMNOTTRHJMLMCDEIIAD00718251	2022	1	1	1
5292	TQRMNOPRPHMNMLCDEIIAD00701192	2022	1	1	1
5293	TQRMNOPRPHMNMLCDEIIAD00701192	2022	1	1	2
5294	TQRMNOSRWHKMLUCDEIIAD00780103	2022	1	1	1
5295	TQRMNOSRWHKMLUCDEIIAD00780103	2022	1	1	2

5294 rows × 135 columns

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?)

La primera variable que construimos muestra la proporción de menores de 10 años que hay en el hogar (Prop_Men10). Consideramos que es una variable importante a la hora de predecir si un hogar es pobre ya que, como sabemos, la pobreza infantil es una realidad que afecta a nuestro pais. La segunda variable que creamos toma en consideración variables descriptivas de los materiales con los cuales están hechos los hogares, y nos informa con un 1 cuando un hogar está construido con materiales de baja calidad (casa_mala). Por éltimo, creamos una tercera variable binaria que indica con un 1 cuando el hogar está sobrepoblado (potencial_hacinamiento).

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?

```
El 33.73 % de hogares es pobre.
```

Como puede verse arriba, la tasa de pobreza que identificamos mediante los ingresos familiares declarados es del 33,73%, mientras que el informe elaborado por el INDEC para el mismo período identifica al 28,2% de los hogares del GBA como pobres. Esta diferencia es congruente con una posible hipótesis de subreporte de ingresos en la EPH, lo que nos haría estimar una mayor cantidad de pobres de la real si tomamos solo la variable ITF.

Parte II: Construcción de funciones

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). 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 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 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: Regresión logística, Análisis de discriminante lineal, KNN, Árbol de decisión, Support vector machines (SVM), Bagging, Random Forests, Boosting.

Parte III: Clasificación y Regularización

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).

Out[36]:

	Metodo	Accuracy	ECM	AP	AUC	Verdadero 0	Falso 1	Falso 0	Verdadero 1	N
0	LDA	0.786382	0.213618	0.528108	0.707773	472	40	120	117	
1	Logit (l1)	0.787717	0.212283	0.530467	0.708750	473	39	120	117	
2	Logit (l2)	0.785047	0.214953	0.525780	0.706796	471	41	120	117	
3	KNN	0.787717	0.212283	0.538457	0.733679	451	61	98	139	
4	Tree	0.809079	0.190921	0.572369	0.727774	486	26	117	120	
5	Bagg	0.805073	0.194927	0.565560	0.740708	469	43	103	134	
6	Rand_For	0.811749	0.188251	0.578771	0.751257	469	43	98	139	
7	Grad_Boost	0.838451	0.161549	0.632850	0.776454	484	28	93	144	
8	SVC	0.805073	0.194927	0.567976	0.750907	460	52	94	143	

3. ¿Cuál de todos los métodos evaluados predice mejor? ¿Con qué hiperparámetros? Justifiquen utilizando las medidas de precisión que conoce.

Basandonos en la tabla presentada en el inciso anterior, el método que minimiza el ECM y tiene la mejor precisión es Gradient Boosting (con un máximo de siete particiones y 40 estimaciones). En comparación con el resto de los métodos, esta variable de Boosting también cuenta con el área más grande debajo de la curva ROC.

Si la idea es identificar hogares pobres para planificar políticas públicas, el objetivo pasaría a ser el de identificar la mayor cantidad posible de verdaderos positivos, ya que esto daría un mayor alcance dentro de la población objetivo. Teniendo en cuenta esto, Gradient Boosting sigue siendo el mejor método para predecir, ya que cuenta con 144 verdaderos positivos (seguido de cerca por SVC, con 143 y un alfa optimizado mediante KFold CV en un valor de 0.005).

4. ¿Lograron mejorar sus predicciones respecto al TP3?

En el TP anterior, con el metodo LDA predecíamos 112 verdaderos positivos. Como mencionamos en el inciso 3, la cantidad de verdaderos postivos predichos por Boosting es de 144. Por lo tanto hay una notable diferencia en la capacidad predictiva respecto del trabajo previamente entregado. Incluso, si comparamos el método que

predecía con un menor ECM en el TP3, también podemos ver una mejora. Para el trabajo previo, Logit (con λ = 0.00001) predecía con un ECM igual a 0.217623. En este caso, Boosting arroja un ECM igual a 0.161549.

Es importante aclarar que, además de agregar las variables nuevas que creímos relevantes y de incorporar metodos a nuestra función de evaluación, realizamos ajustes en las funciones respecto a las entregadas en el código anterior. Por esta razón, la mejora en todos los métodos bajo evaluación respnde a varios factores, pero Gradient Boosting con las especificaciones mencionadas supera en todas las métricas al resto, tanto en minimización del error, precisión y clasificación.

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?

```
En base al método de Gradient Boosting, la cantidad predicha de hogares pobr
es para la submuestra es de 172
La proporción de hogares pobres en la submuestra es del 22.96 %.
```

Tal como muestra el *print* de la celda superior, estimar la proporción de hogares pobres en nuestra submuestra mediante Gradient Boosting nos permite identificar que un 22.96% de los hogares son pobres (correspondiéndose esto con un total de 172). Esto es una mejora contra el TP3, donde nuestra predicción mediante el método LDA era del 20.29%, ya que ahora nos acercamos más al valor publicado por INDEC para este período (28.2%), como indicábamos en el ejercicio 8 de la parte I.