Universidad de
SanAndrés

Professor: Noelia Romero

Trabajo Práctico 4

Pacheco - Paganini - Starobinski

Fecha: 26/11/2023

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

Inciso 1

Ver código

Inciso 2

Ver código

Inciso 3

Para la limpieza de datos, se tomaron en cuenta diferentes factores. En primer lugar, se eliminaron las variables que tienen más de un 50 % de missings. En segundo lugar, se eliminaron observaciones anómalas como edades e ingresos negativos. En tercer lugar, se buscan outliers en algunas variables mediante la construcción de una función, pero no se eliminan observaciones porque las variables que contienen outliers son aquellas que contienen información de los ingresos de los individuos y estas variables se eliminarán posteriormente para la predicción de la pobreza. Además, dado que la distribución de los ingresos es asimétrica, los datos atípicos encontrados mediante la función que se definió pueden no decir mucho. Finalmente, se convierten las variables categóricas a variables dummy. Es importante mencionar también que se decidió no eliminar las observaciones de las variables que contienen la categoría "Ns/Nr"debido a que se quiere replicar más de cerca al INDEC. Asimismo, consideramos también que no responder a una pregunta puede tener cierta significancia social por lo que incluir estas observaciones puede no conllevar a algún error en nuestra predicción de la pobreza.

Inciso 4

Se generaron 3 variables que no se encuentran en la base de datos de la EPH. La primera es la proporción de menores de edad en el hogar (menores de 18). Esta se define como la división entre la cantidad de menores y la cantidad de miembros del hogar. La segunda es una variable dummy que toma el valor de 1 a nivel hogar si el cónyuge trabaja. La tercera y

última variable que se construyó es una variable dummy a nivel hogar que toma el valor de 1 si el jefe del hogar tiene primario incompleto y 0 por el contrario.

Inciso 5

En la figura 1, se presenta un gráfico de dispersión que muestra la relación entre la proporción de menores de edad en el hogar (variable construida a partir de la cantidad de menores de 18 años y el número de miembros del hogar) y el Ingreso Per Cápita Familiar. Podemos observar una corrrelación negativa que puede ser explicada por el hecho de que mientras más grande sea la proporción de menores de edad más grande puede ser la cantidad de miembros inactivos; es decir, una mayor proporción indica que existen más menores de edad en el hogar que al momento de realizarse la encuesta no trabajan debido a su edad o porque se encuentran estudiando y, por tanto, IPCF es menor. Este resultado puede sugerir que esta proporción es un predictor de la pobreza, ya que un menor IPCF puede generar que el ingreso familiar no sea suficiente para cubrir las necesidades básicas de los miembros del hogar.

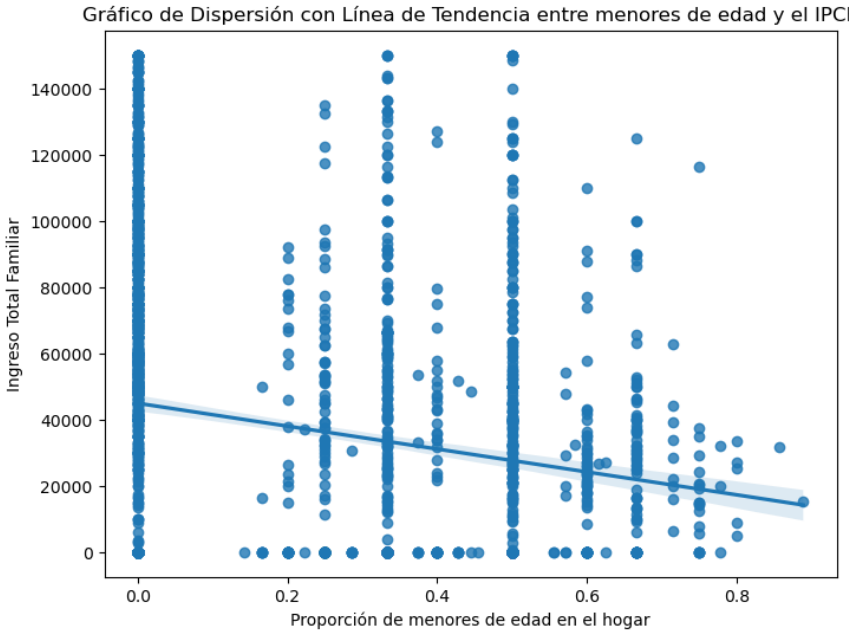

Figura 1

Inciso 6

Ver código

Inciso 7

La tasa de hogares que se encuentra por debajo de la línea de pobreza que obtuvimos es 28.32 %. Este resultado se asemeja con lo reportado por el INDEC que encuentra que el 30.3 % de hogares están en situación de pobreza. Cabe mencionar que esta incongruencia con la INDEC y nuestro resultado puede deberse a la diferencia temporal del análisis, puesto que el informe reporta resultados para el primer semestre mientras que nosotros utilzamos datos trimestrales.

Parte II: Construcción de funciones

Las funciones creadas diferen en muy poco con las del TP3. Las diferencias son: a) *evalua_metodo* ahora arroja como *output* los verdaderos positivos, verdaderos negativos, falso positivos y falsos negativos, y b) *evalua_config* y *evalua_metodo* han sido adaptadas para incorporar también para permitir la optimización de hiperparámetros para un árbol de decisión y tres metodologías basadas en árboles (*Bagging* con árboles, *Random Forests* y *Boosting*), así como la evaluación de los resultados predictivos de metodologías.

Inciso 1

Se crea primeramente la función textitevalua_metodo, que ajusta un modelo previamente configurado (lo recibe como argumento) utilizando datos de entrenamiento y genera métricas de desempeño a partir de una muestra de prueba. Se tienen en cuenta ponderaciones en caso de que se tenga una variable con esa información y el método empleado sea de regresión logística (que admite *sample weights*). La decisión de requerir la previa configuración del modelo a ajustar responde a cuestiones de compatibilidad con las funciones de los siguientes incisos, así como a la consideración de que la utilidad de la *evalua_metodo* no se ve aumentada si, en lugar de configurar el modelo antes de emplearla, se incluyen los detalles de configuración como argumentos.

Las métricas obtenidas incluyen las requeridas por la consigna actual y la anterior: la matriz de confusión (tanto la matriz en sí como la representación gráfica como imagen), el gráfico de la curva de ROC, verdaderos positivos, verdaderos negativos, falsos positivos, falsos negativos, el AUC (*Area Under the Curve*), *Accuracy* (porcentaje de datos correctamente clasificados) y el Valor Predictivo Positivo (VPP, a veces llamado *Precision*, el porcentaje de individuos correctamente clasificados dentro de los clasificados como pobres). Además de estas, se eligió computar el Error Cuadrático Medio (ECM), que será utilizado como medida del error de

referencia en el TP, junto al Valor Predictivo Negativo (VPN, el porcentaje de individuos correctamente clasificados dentro de los clasificados como no pobres). Estas últimas dos métricas aportan información más desagregada que *Accuracy*, y pueden ser relevantes en algunos contextos en que se quiere penalizar de diferente forma a los errores de predicción en función de la categoría predicha.

La función de pérdida vista en clase para tareas de clasificación es del tipo $L(Y, \hat{Y}) = 1(Y \neq \hat{Y})$, por lo que podría parecer inapropiado utilizar el ECM, que habitualmente se justifica en base a una función de pérdida $L(Y, \hat{Y}) = (Y - \hat{Y})^2$, más apropiada para contextos de regresión. De hecho, existen razones teóricas para evitar el uso de ECM cuando se explican variables binarias. Sin embargo, estas razones aplican para cuando se define a $\hat{Y}$ como la probabilidad predicha de pertenecer a una categoría. Además, al definir $\hat{Y}$ se define como la categoría predicha (es decir, no se define como la probabilidad predicha sino como la categoría a la que asigna a una observación a partir de esa probabilidad, utilizando algún método como el clasificador de Bayes), las dos funciones de pérdida mencionadas son equivalentes, y el porcentaje de errores equivale al ECM. Una consecuencia de esto es que, **en el contexto específico de este TP**, se cumple que $ECM = 1 - Accuracy$.

Podría utilizarse otra nomenclatura para este porcentaje de errores, o directamente hacer mención a la *Accuracy*, puesto que contiene la misma información. Sin embargo, por conveniencia (en vistas de lo construido para la consigna anterior) **se optará por utilizar al ECM como medida principal del error de un modelo**, y retener la nomenclatura de "ECM". De todas formas, las funciones de los incisos siguientes están construidas de manera flexible, a efectos de facilitar la utilización de otras medidas de error que puedan construirse a partir de los *outputs* de *evalua_metodo*.

Inciso 2

Se construye la función *cross_validation*, que genera K particiones de una muestra (utilizando la función *KFold*) y utiliza la metodología de *K-fold cross-validation* para generar un error de predicción promedio. En vista de lo establecido en el inciso anterior, la medida de error a generar para cada submuestra en este TP estará dado por el ECM. Sin embargo, el código permite que se utilicen otras medidas de desempeño, a saber, el AUC, el VPP y el VPN. Estos guarismos registran aciertos, no errores, en el ejercicio de predicción, por lo que, cuando se utilice alguna métrica A arrojada por *evalua_metodo* diferente del ECM, la medida de error estará dada por $E = 1 - A$ (todas las métricas escalares arrojadas por *evalua_metodo* reflejan porcentajes, y se sitúan en el intervalo $[0, 1]$). Es posible utilizar *Accuracy*, también, pero en ese caso la medida de error arrojada por la función será equivalente al ECM, habida cuenta de lo discutido en el Inciso 1. No se permite utilizar

En la tutorial vimos que es buena práctica estandarizar los datos antes de implementar técnicas de regularización. Si bien regresiones como la lineal o la logística no son afectadas por la estandarización, sí son positivamente afectadas las técnicas de regularización como LASSO y Ridge como se vio en clase, así como también el método de K vecinos más cercanos. La función construida estandariza las variables de la muestra de entrenamiento y de prueba, utilizando *StandardScaler*, y contiene una opción que permite desactivar la estandarización.

Se requiere que la estandarización no implique la contaminar los datos de la muestra de prueba con información de la muestra de entrenamiento. Es por esto que la estandarización se aplica por separado a ambas muestras en cada partición: si la función o el *loop* de particiones recibieran una muestra ya estandarizada, no podrían evitar esta contaminación.

Inciso 3

A continuación se genera la función *evalua_config*, que utiliza los insumos necesarios para la función *cross_validation*, junto con listas de hiperparámetros, y arroja la combinación de hiperparámetros óptima dentro de las opciones consideradas (en el sentido de minimizar una medida de error escogida) y el error asociado a esa combinación. La función se construye para los métodos de regresión logística, K vecinos más cercanos, árbol de decisió y tres métodos basados en árboles (*Bagging* con árboles, *Random Forests* y *AdaBoost*).

En el primer caso, los hiperparámetros a elegir son λ (*regularization strength*) y el llamado *l1 ratio* para la metodología de regularización de *Elastic Nets*, que determina la combinación a emplear de los coeficientes λ_1 y λ_2, correspondientes a la penalización $l1$ (al estilo de LASSO) y $l2$ (al modo de Ridge), respectivamente. Se fijó el *solver* de la clase de *scikit-learn LogisticRegression* en en *saga*, puesto que este es el único que admite LASSO, Ridge y Elastic Nets simultáneamente. Incluir al hiperparámetro *l1_ratio* permite subsumir a las regularizaciones LASSO y Ridge como casos particulares de *Elastic Nets*, puesto que LASSO se corresponde con el caso en que *l1_ratio=1* y Ridge al caso en que *l1_ratio=0* (como puede verse en la documentación). Incluir los valores 1 y 0 en el la lista de insumos *elastic_list* permite considerar a ambos tipos de regularización, junto con un conjunto de combinaciones diferentes de ambas en un marco de *Elastic Nets*, dentro del conjunto de opciones para la regularización que se comparan en el proceso de optimización de *evalua_config*. En caso de quererse evaluar diferentes opciones de λ únicamente para LASSO o Ridge, puede fijarse *elastic_list* como una lista con un único elemento igual a 1 o a 0, respectivamente.

Si el método utilizado es el de K vecinos más cercanos, se optimiza en función de un único hiperparámetro: la cantidad (K) de vecinos. La lista *neighbor_list* consiste de un conjunto de valores posibles para la cantidad de vecinos a tomar para el ajuste del modelo.

A diferencia del TP3, la definición actual de esta función admite llamar a *cross_validation* con la opción de estandarización dessactivada, puesto que ahora es posible utilizarla para optimizar metodologías basadas en árboles, que no requieren estandarización.

Inciso 4

Por último, se crea *evalua_multiples_metodos*, una función que toma como insumos los necesarios para *evalua_config* pero reemplazando al método por una lista de métodos, y necesitando de una partición previa de la muestra en datos de entrenamiento y prueba, a diferencia de sus contrapartes de los Incisos 2 y 3 (estas funciones se aplican sobre toda la muestra proporcionada, pero la función *evalua_multiples_metodos* solamente les insuma los datos de entrenamiento). La salida de la función es una tabla, que detalla, para cada método listado, los hiperparámetros óptimos dentro de las opciones provistas de acuedo con la función *evalua_config* (en caso de ser aplicable) y el subconjunto de las medidas de desempeño construidas en el Inciso 1 que son escalares. Si bien la tabla muestra todas esas medidas de desempeño, la función continúa necesitando como insumo una medida de error específica, al igual que las dos anteriores, puesto que es en función de esa métrica que se optimizará la elección de hiperparámetros. Esta optimización se realiza en base a la función del inciso anterior y, por tanto, solamente es posible para los dos métodos allí considerados, pero la función *evalua_multiples_metodos* permite de todas formas la inclusión de cualquier otro método que pueda emplearse con la biblioteca *scikit-learn* (entre ellos el de análisis discriminante lineal).

Es posible desactivar la opción de estandarizar los datos. Su activación implica tanto llamar a *evalua_config* con estandarización activada, en el paso de elegir los hiperparámetros óptimos, como estandarizar los datos antes de realizar el ajuste final. Esto garantiza consistencia entre el tipo de datos utilizados para estos dos pasos. La estandarización no se hace antes del *loop* que recorre todas las metodologías evaluadas, porque esto implicaría que el proceso de validación cruzada se realizara en un contexto en que todas las posibles particiones están contaminadas de información del resto de la muestra.

Clasificación y regularización

Inciso 1

Se eliminaron las variables requeridas por la consigna, junto con variables que permanecen constantes en todas las muestras (indicadora de aglomeraciones urbanas de más de 500.000 habitantes, año, trimestre, región geográfica) y las variables identificadoras, puesto que refie-

ren a aspectos de la encuesta, y no de los hogares encuestados. No resultó apropiado eliminar estas variables en la sección de limpieza de datos, puesto que era necesario utilizar algunas de ellas para tareas de *merging* de bases e identificación de hogares pobres (se había evitado eliminar en aquel punto todas las que se pertenecían a la categoría de "IDENTIFICACIÓN" en la EPH). Se excluyeron, asimismo, las variables ponderadoras, a excepción de de *PONDIH*. Según detalla el documento de diseño de la encuesta, esta variable es la adecuada para el trataminento del ingreso per cápita familiar, por lo que parece relevante para un TP en el que se estudia a la población clasificada como pobre de acuerdo a una metodología de línea de pobreza.

Se corrigió, respecto del TP3, la eliminación de las variables de ingreso. Algunas variables habían quedado sin eliminar en aquella instancia (tales como *GDECINDR* o *DECOCUR*); en esta instancia se eliminaron todas las variables relacionadas a ingreso.

La eliminación de las variables requeridas por la consigna se hizo para ambas bases. Las supresiones adicionales se hicieron solamente para *respondieron*, debido a que algunas de las variables a descartar serán utilizadas en el Inciso 5 para trabajar con las predicciones hechas para *norespondieron*.

Luego de esto, se separó la base *respondieron* en tres *dataframes*, dos de ellos de una sola columna (correspondientes a la variable dependiente y y a una variable *ponds_resp* de ponderadores) y otro que corresponde al conjunto X de variables explicativas.

Inciso 2

Luego de partir los tres *dataframes* en muestras de entrenamiento y prueba, se corrió la función *evalua_multiples_metodos* para obtener los resultados de predecir en la muestra de prueba luego de ajustar en la muestra de entrenamiento con cada metodología considerada. Los métodos utilizados fueron: regresión logística (en adelante RL), K vecinos más cercanos (KVC), análisis discriminante lineal (ADL), árbol de decisión (AD), *Bagging* con árboles de decisión (BAD), *Random Forests* (*RF*) y *Boosting* con árboles utilizando la técnica de *AdaBoost* (AB). Para todas las metodologías excepto ADL, se realizó una optimización dentro de un conjunto (previamente definido) de posibles hiperparámetros, eligiendo el hiperparámetro (o la combinación de hiperparámetros) que mejores resultados obtenía en un procedimiento de *10-fold cross-validation* en la muestra de entrenamiento.

Para la RL se eligieron todas las combinaciones de hiperparámetros que fueron consideradas en el TP3, tanto en el inciso 2 de la Parte III como en el 5. Esto implicó tomar, para λ, un barrido en $\lambda = 10^n$ con $n \in \{-5, \ldots, 5\}$ junto a $\lambda = 5$, y para *l_ratio*, simplementre tres valores (0;0,5 y 1) que correspondían a tres tipos de regularización: LASSO, Ridge y Elastic

Nets (fijando para esta última un *l_ratio* de 0,5). Al igual que en el TP3, no se experimentó con diferentes hiperparámetros para ADL y para KVC se consideraron cantidades de vecinos entre 1 y 10. A su vez, para el AD se optimizó el número de profundidad máxima (máximo número de particiones entre nodo inicial y nodos terminales) entre un conjunto que incluía todos los múltiplos de 10 iguales o menos a la cantidad total de variables explicativas (si esta cantidad es inferior a 20 se considerarían todos los número naturales iguales o menors que ella), dejando el mínimo de observaciones por nodo terminal en su valor por defecto de 2. El hiperparámetro a optimizar para BAD y *RF* fue la cantidad de árboles a promediar, con los demás hiperparámetros (proporción de los datos a utilizar para las muestras construidas con *bootstrapping*, en el caso de *RF* la proporción de regresores a utilizar en cada partición, la medida de impureza, etc.) fijados en su número por defecto. Se optimizó AB con el número máximo de árboles a construir secuencualmente. El conjunto de posibles hiperparámetros para los últimos tres fueron los múltipos de 10 entre 10 y 100.

Inciso 3

La Tabla 1 resume los resultados de la corrida de *evalua_multiples_metodos* llevada adelante en el Inciso 2, y sugieren que la mejor predicción en el conjunto de prueba corresponde al método de *Bagging*, con una cantidad de árboles o muestras de entrenamiento (creadas a partir de *bootstrap*) igual a 80. Su ECM (7,8 %) es el menor de entre todas las metodologías, lo que necesariamente implica que su *Accuracy* (92,8 %) es la mayor, dado que, como se discutió en la Parte II, es igual a $1 - ECM$ en este contexto específico y con la definición de ECM que se ha tomado. También superó a sus contrapartes para las otros métodos su AUC (91,0 %), y arrojó la máxima cantidad de verdaderos positivos (397) y la menor cantidad de falsos negativos (59). El único modelo comparable resultó ser el *Random Forest*, con un número óptimo de árboles igual a 90, que obtuvo una mayor *precision* (92,8 % contra 90,8 %), una mayor cantidad de verdaderos negativos (771 contra 759) y una menor cantidad de falsos positivos (28 contra 40).

Método	Configuración	Matriz de confusión	Verdaderos positivos	Verdaderos negativos	Falsos positivos	Falsos negativos	AUC	Accuracy	Precision/VPP	VPN	ECM
Logistic Regression	Regularización EN (l1_ratio = 0.5), Lambda =...	[[673, 126], [146, 310]]	310	673	126	146	0,761064	0,783267	0,711009	0,821734	0,216733
KNeighbors Classifier	(Vecinos = 1)	[[728, 71], [104, 352]]	352	728	71	104	0,841534	0,860558	0,832151	0,875	0,139442
Linear Discriminant Analysis	—	[[725, 74], [132, 324]]	324	725	74	132	0,808955	0,835857	0,81407	0,845974	0,164143
Decision Tree Classifier	(Profundidad = 20)	[[703, 96], [67, 389]]	389	703	96	67	0,86646	0,87012	0,802062	0,912987	0,12988
Bagging Classifier	(B de Bagging = 80)	[[759, 40], [59, 397]]	397	759	40	59	0,910276	0,921116	0,908467	0,927873	0,078884
Random Forest	(B de Random Forest = 90)	[[771, 28], [95, 361]]	361	771	28	95	0,878311	0,901992	0,928021	0,8903	0,098008
Ada Boost	(B de Random Forest = 50)	[[712, 87], [138, 318]]	318	712	87	138	0,794241	0,820717	0,785185	0,837647	0,179283

Tabla 1

Los dos ajustes tuvieron un desempñeo similar, pero el de *Random Forests* parece haber sido algo más conservador para predecir que individuos son pobres (más verdaderos negativos y menos falsos positivos, a costa de menos verdaderos positivos y más falsos negativos). En un contexto en que se prefiere este conservadurismo, porque se quiere penalizar más las predicciones afirmativas y erróneas de pobreza, sería preferible este método. En este Trabajo Practico, sin embargo, será preferido el método de *Bagging*, porque tiene mejor ECM (medida que se toma como preferida, dado que se pretenden generar predicciones precisas de la pobreza sin penalizar distintos tipos de errores de diferente forma) y mayor área bajo la curva ROC.

Inciso 4

En el TP3, la eliminación de las variables de ingreso no había sido completa, y esto aumentó la capacidad predictiva de los diferentes métodos afuera de la muestra. El resultado de que ajustar un modelo de ADL arrojaba un ECM de 4,4 %, superador de todas las metodologías utilizadas en el presente Trabajo Práctico, debe ser visto en ese contexto. Reducen la comparabilidad, también, las modificaciones en la limpieza de datos que alteraron la base *respondieron* respecto del Trabajo Práctico anterior.

Para obtener una predicción "del TP3" comparable a las del TP4, se volvió a correr la función *evalua_multiples_metodos* solamente para las metodologías consideradas en el TP3, repitiendo las mismas listas de hiperparámetros que en el inciso 2 (recordar que habían sido definidas como para abarcar todas las combinaciones de hiperparámetros con las que se había experimentado en el TP3), con las observaciones utilizadas en el TP4 y eliminando las variables creadas en el inciso 4 de la Parte I.

Se obtienen ECM's de 22,2 % para RL (con regularización de *Elastic Nets* y $\lambda = 10^{-4}$), 13,5 % para KVC (con 1 vecino) y 16,7 % para ADL. Los resultados son claramente peores que los del método de *Bagging* evaluado en el inciso 2, pero la comparación relevante es con los mismos métodos. En la Tabla 1, puede verse que los ECM's para RL, KVC y ADL son 21,7 %, 13,9 % y 16,4 % (los hiperparámetros elegidos fueron los mismos, con excepción de λ para RL en el inciso 2, que tomó el valor de 0,1). Puede concluirse que las variables creadas permitieron una predicción levemente mejor para RL y ADL; sorprendentemente, fue levemente *peor* en el caso de KCV, de acuerdo con el criterio del ECM. Puede corroborarse que conclusiones similares se desprenden de cualquier otro criterio, con excepción del Valor Predictivo Negativo, que es apenas superior para KVC ajustado en el inciso 2.

Inciso 5

Primeramente, se creó una nueva base para la predicción, llamada *noresp_pred*, igual a *norespondieron* pero excluyendo las variables que no pertenecen al *dataframe* X de variables explicativas en la muestra *respondieron*. No se habían eliminado antes (y no se eliminaron, incluso en este inciso, para las base *norespondieron* original), porque dos de ellas (*CODUSU* y *NRO_HOGAR*) son necesarias para construir el porcentaje de hogares pobres según la predicción realizada en este inciso. Luego de esto se creó una versión estandarizada de X, llamada X_stand, para realizar el ajuste. Esto puede parecer innecesario, dado que estandarizar los datos no mejora el ajuste para la metodología de *Boosting* que se utilizó, pero se intentó mantener todo en igualdad de condiciones (al máximo grado posible) con respecto a la evaluación de los diferentes métodos llevada adelante en el inciso 2, y en esa instancia se había estandarizado los datos por la presencia de otras metodologías que sí lo requerían. Trabajar con una muestra estandarizada para el ajuste requiere estandarizar también las variables explicativas en la muestra de predicción, por lo que se creó también un dataframe llamado *noresp_pred_stand* con datos estandarizados.

De esta manera, se llevó adelante una predicción de los individuos pobres en *noresp_pred_stand* (equivalente a una predicción análoga para *norespondieron*). **Se obtuvo el resultado de que, de los 2959 individuos provenientes de hogares que no respondieron su ingreso familiar, la predicción arroja que 1081 son pobres**, un 36,5 %.

Luego de esto se construyó la cantidad de hogares pobres compatible con la predicción realizada. Pese a que se importó la base de hogares, la unión con la base de individous generó datos a nivel individual, lo que implica que el ajuste y la predicción se hacen a nivel de individuos. Esto posibilitó la predicción anterior de cantidad de personas pobres, pero trae aparejado un problema: con la metodología de identificación de la pobreza previamente vista, una persona era pobre si y solo si otra persona del mismo hogar lo era, de manera que no había discrepancias posibles entre la identificación de hogares pobres y personas pobres. Si bien es esperable que la probabilidad de que se prediga una sola categoría de pobreza (pobre o no pobre) para todos los individuos de un solo hogar sea razonablemente alta, dado cualquier hogar en particular, también es probable que esto no haya sucedido para algunos hogares. En vistas de esto, para computar el porcentaje de hogares pobres en la muestra, primeramente se computó la proporción de pobres predichos por hogar, y luego se clasificó al hogar como pobre si se había predicho que la mitad o más de sus miembros no eran.

Se obtuvo, de esta manera, que un 29,6 % de los hogares que no respondieron su ingreso familiar son pobres.

Este guarismo es satisfactorio, puesto que es levemente mayor a la tasa de 28,3 % obtenida en el inciso 7 de la Parte I. Debe tenerse en cuenta, sin embargo, que las dos números no son plenamente comparables, puesto que el número calculado en la Parte I es la **tasa de pobreza**.

Para la base *norespondieron*, no fue posible calcular una tasa de pobreza extrapolable a la población en general, habida cuenta de que la variable ponderadora *PONDIH*, construida justamente en base a los ingresos declarados, les asigna peso nulo. Para poder realizar correctamente la comparación, se computa al porcentaje de hogares, sin ponderar, que son pobres en la muestra de los que sí respondieron. El resultado es un 25,3 %. El modelo predice que el porcentaje de pobres es moderada, pero claramente mayor en la muestra de los que no respondieron, algo que parece esperable.