MissForest

Introducción

El presente trabajo es una descripción del paper *"MissForest - nonparametric missing value imputation for mixed-type data"*. El método descripto en el mencionado artículo ofrece una solución al problema de estimar los valores faltantes que suelen aparecer en las bases de datos. En un contexto de Machine Learning, nuestra base de datos es la principal fuente de información (junto quizás con algún conocimiento previo sobre la distribución de los datos) que nos ayudará a definir cuál es la clase/valor, de las nuevas instancias desconocidas. Pero como vimos en la materia, nuestras bases de datos pueden no estar completas, y por otro lado, los algoritmos que solemos utilizar para generar nuestros estimadores, no soportan celdas vacías. Por lo tanto es indispensable que antes de entrenar a nuestro modelo estimemos aquellos valores faltantes.

Cómo funciona el algoritmo

Consideramos una matriz **X** con *n* filas y *p* columnas.
Dada una columna Xs, consideramos la siguiente división de la matriz:

1. Las celdas no vacías en la columna X_s: Ys-obs.
2. Las celdas vacías en la columna X_s: Ys-miss.
3. Todas las columnas salvo X_s, considerando las filas que intersectan con Ys-obs: Xs-obs.
4. Todas las columnas salvo X_s, considerando las filas que intersectan con Ys-miss: Xs-miss.

En la siguiente tabla ilustramos estas cuatro regiones para una columna X_s.

X1	X2	X3	...	...	...	Xs	...	...	...	...	...	Xp
3	Blue	3			False	1,2	9			1	Cold	7,2
7		2	7	Yes		4,5	4	9	2	0	Cold	8,1
3	Blue		3	Yes	True			6		1	Hot	2,5
		7	2		True		7		0	0		9,3
7	Brown	3	7	No	True	8,3		9		1	Hot	10,1
7	Brown	7		No			9	10	4	0	Hot	8,0

Ys-obs, Ys-miss, Xs-obs, Xs-miss.

1. Ordenamos las p columnas de menor a mayor en base a la cantidad de celdas vacías que hay en cada columna.
2. Estimamos todos los valores faltantes en base a algún método simple de estimación, (por ejemplo el promedio de la columna, o la categoría que más aparece en esa columna para valores discretos).
3. Mientras no se cumpla el *criterio de parada*:
    a. Guardamos en X_{old} la última matriz estimada.

b. Para cada columna s, (ordenadas según el paso 1):
    i. Entrenar un Random Forest con Ys-obs, Xs-obs.
    ii. Predecir Ys-miss en base a Xs-miss.
    iii. Guardar en *Xnew* la nueva matriz., en base a lo estimado para Ys-miss
c. Fin para.
d. Verificar el criterio de parada.

4. Fin mientras.
5. Retornar *Xnew.*

En cuanto al criterio de parada, este consiste en mirar si el cambio o la diferencia entre *Xnew* y *Xold*, de acuerdo a una medida dada para variables continuas ΔN y otra para las categóricas ΔF, es mayor que en la iteración anterior. Cuando esto ocurre para ambas medidas entonces se asume que los valores predichos ya no son buenos, o al menos no son mejores que los que se habían predicho previamente.

Otros métodos

El paper compara missForest con otros métodos, separando en tres casos: datos con variables continuas solamente, datos con solo variables categóricas y por último conjuntos de datos con variables de ambos tipos. Para el primer caso compara contra KNNImpute y MissPaLasso. KNNImpute sigue la misma idea que KNN, estimar los valores faltantes en base a las K instancias más cercanas. MissPALAsso intenta minimizar el error de lasso[1].

Para los casos de variables categóricas y para los de variables mixtas se comparó contra KNNImpute y MICE. Este último se adapta a ambos tipos de variables pero opera bajo ciertas asunciones sobre los datos[2], dado que asume que los datos faltantes son Missing At Random (MAR), lo cual significa que la probabilidad de que un valor esté faltante depende solo de los valores observados y no de los faltantes. Implementar MICE cuando los datos no son MAR puede resultar en resultados sesgados.

Para KNNImpute con variables categóricas se recurrió al método de Dummy Variables visto en clase. Se agregan primero las columnas nuevas, se corre KNNImpute estimando los valores faltantes, y después se reconvierte la matriz a la forma original.

Experimentos

En cada ensayo, a los datasets completos se le eliminaron datos totalmente al azar, separando en tres casos: 10%, 20%, y 30% de datos eliminados. Se evaluó el desempeño midiendo la diferencia entre los datasets imputados y los originales. Para las variables continuas se usó “normalized root mean squared error” (ecuación 1). Para las variables categóricas se consideró la proporción de casos mal imputados sobre el total de casos faltantes.

$$\mathrm{NRMSE} = \sqrt{\frac{\mathrm{mean}\left((\mathbf{X}^{\mathrm{true}} - \mathbf{X}^{\mathrm{imp}})^2\right)}{\mathrm{var}\left(\mathbf{X}^{\mathrm{true}}\right)}},$$

Ecuación 1. Cálculo de Normalized Root Mean Squared Error. X^{true}: matriz de valores reales original, X^{imp}: matriz con valores imputados.

[1] https://es.wikipedia.org/wiki/LASSO_(estad%C3%ADstica)

[2] *Más información en* https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3074241/#R30

Se comparó además el error estimado por medio del método *Out Of Bag* propia del Random Forest. Se vio que esta estimación fue muy precisa difiriendo poco del error NRMSE, tanto para variables categóricas como para las continuas. Aunque se observó que este método tiende a subestimar el error a medida que la cantidad de valores faltantes aumenta.

Resultados de las experimentos.

Los resultados arrojados fueron especialmente buenos para los casos de conjuntos de datos con variables mixtas y solo categóricas, los experimentos mostraron que missForest redujo, en algunos casos, hasta un 60% el NRMSE más que los otros métodos. Para el caso de valores continuas, en algunas ocasiones missForest redujo el error un 25% más que KNNImpute, aunque en algunos casos MissPaLasso obtuvo un error menor que missForest.

Con respecto a la eficiencia computacional, *missForest* mostró ser mayor que MICE y MissPALasso, aunque KNNImpute fue por lejos el más performante. Pero el paper rescata que en comparación con KNNImute, *missForest* no requiere una previa estandarización de los datos, ni el trabajo adicional de codificar las variables dummy, ni tampoco implementar grid search/cross validation para tunear hiperparámetros.[3]

Conclusiones generales

La principal ventaja del método es que no necesita un trato especial para variables discretas o continuas. El algoritmo trata a los dos tipos de variables por igual y no necesita preparación previa. Este hecho hace que missForest pueda captar la correlación que pudiera haber en un mismo data set entre variables de diferente tipo. MICE funciona bien con ambos tipos de variables pero en condiciones en las que se cumple las asunciones mencionadas previamente, mientras que *missForest* no asume ninguna distribución sobre los datos, y por lo tanto puede estimar bien para cualquier distribución desconocida. Es notable que para el caso de variables discretas la performance de *missForest* fue mucho mejor que la de MICE. Por otro lado, MissPaLasso tiene la desventaja de ser un algoritmo de exclusivo uso para variables continuas. Por último, al utilizar el estimador de error *OOB* no es necesario separar datos para validación y podemos entonces utilizar un mayor conjunto de datos para entrenar al modelo.

[3] Este punto es en realidad discutible, dado que el paper fija en $\sqrt{p}$ la cantidad de variables que de manera random se seleccionan en cada árbol, y en 100 la cantidad de árboles de cada *forest*. Estos dos valores son en realidad hiperparámetros del algoritmo y se los podría ajustar con grid search/cross validation.