

- 1 - Consideraciones previas
- 2 - Objetivo
- 3 - Obtención de modelos
- 4 - Conclusiones

Pump It Up. Data Mining The Water Table

Code ▾

Miguel Benayas Penas

Septiembre 2021

1 - Consideraciones previas

Siguiendo lo que marca el archivo "00 Guia_de_Estudio_Aplicaciones_BigData_en_Empresa.pdf", voy a asumir que estoy trabajando como Data Scientist en una empresa X y que el jefe del departamento me ha pedido un primer informe para predecir el estado de los acuíferos, en base a los inputs proporcionados en el sitio web.

Por tanto, en lo que resta de archivo, se asume que el destinatario está familiarizado con la Ciencia de Datos. Esto desemboca que la naturaleza del documento deba de ser de carácter técnico.

En aras a una mayor legibilidad, otro objetivo de este módulo del máster, solo se mostrarán los outputs de los dos mejores modelos junto con un representante de cross validation con caret. Los otros modelos (con su respectivo código) se mencionarán y se pasará rápido a las conclusiones. Esto evita al lector perderse en un maremágnum de tablas y gráficos.

Como nota final, todo este trabajo lo he realizado yo solo sin haber consultado nada de la solución proporcionada por el ganador del concurso, allá en julio. Por tanto, toda parecido - si hubiera - es fruto de la casualidad.

Una vez esbozado la motivación de este archivo, comenzamos.

2 - Objetivo

Primera aproximación en la resolución de un problema de clasificación sobre el estado de las bombas en acuíferos (waterpoints) mediante un método iterativo. La variable objetivo presenta tres categorías, es decir, se trata de una clasificación multinomial.

Al tratarse de una clasificación no binaria, se usará accuracy como métrica de bondad en vez de ROC.

El problema consta de dos conjuntos. Hay un train con el que se entrenarán y validarán los modelos y un conjunto test con el que se calificará a los mejores modelos candidatos. Como aclaración, se entenderá por modelo a la combinación de 1) conjunto de variables inputs, 2) sus transformaciones, 3) algoritmo con su configuración particular de hiperparámetros y 4) técnica de remuestreo usada.

La tipología de este problema condiciona los algoritmos a evaluar. Tres algoritmos han sido empleados para esta primera toma de contacto con el problema: random forest (en su instancia ranger) y boosting (gbm y xgboost).

Debido a las limitaciones de fecha límite y capacidad computacional, no se podrá hacer una búsqueda exhaustiva de modelos. Es por tanto que el presente documento se basa en el paradigma de ramificación y poda a lo largo de las iteraciones.

3 - Obtención de modelos

Se cargan los paquetes a usar y se empieza con la primera iteración.

Code

3.1 - Iteración 1

En esta primera iteración no se prioriza el optimizar sino el explorar y limpiar el dataset, para así ganar una primera intuición de qué variables y sus transformaciones pueden ser más influyentes.

3.1.1 - Data Loading

Se cargan las todas variables del training set, junto con las variables input del test.

Code

3.1.2 - Data Cleaning

Mínima limpieza, en sucesivos modelos se harán cambios más profundos, así se reduce la posibilidad de sesgar todos los futuros modelos, basados en este conjunto de variables.

Se convierten los ficheros importados en data frames. Se comprueba si hay observaciones duplicadas. Se comprueba que, en efecto, hay el mismo número de columnas en los dataset que el estipulado en la descripción del problema. Se elimina las columnas id al no aportar información.

Code

```
## Número de duplicados en los predictores: 0
```

Code

```
## Número de duplicados en las etiquetas: 0
```

[Code](#)

```
## El número de predictores es de: 40
```

[Code](#)

Comprobación de si los datos están correctamente formateados.

[Code](#)

```

## 'data.frame': 59400 obs. of 40 variables:
## $ amount_tsh : num 6000 0 25 0 0 20 0 0 0 0 ...
## $ date_recorded : IDate, format: "2011-03-14" "2013-03-06" ...
## $ funder : chr "Roman" "Grumeti" "Lottery Club" "Unicef" ...
## $ gps_height : int 1390 1399 686 263 0 0 0 0 0 ...
## $ installer : chr "Roman" "GRUMETI" "World vision" "UNICEF" ...
## $ longitude : num 34.9 34.7 37.5 38.5 31.1 ...
## $ latitude : num -9.86 -2.15 -3.82 -11.16 -1.83 ...
## $ wpt_name : chr "none" "Zahanati" "Kwa Mahundi" "Zahanati Ya Na nyumbu" ...
## $ num_private : int 0 0 0 0 0 0 0 0 0 ...
## $ basin : chr "Lake Nyasa" "Lake Victoria" "Pangani" "Ruvuma / Southern Coast" ...
## $ subvillage : chr "Mnyusi B" "Nyamara" "Majengo" "Mahakamani" ...
## $ region : chr "Iringa" "Mara" "Manyara" "Mtwara" ...
## $ region_code : int 11 20 21 90 18 4 17 17 14 18 ...
## $ district_code : int 5 2 4 63 1 8 3 3 6 1 ...
## $ lga : chr "Ludewa" "Serengeti" "Simanjiro" "Nanyumbu" ...
## $ ward : chr "Mundindi" "Natta" "Ngorika" "Nanyumbu" ...
## $ population : int 109 280 250 58 0 1 0 0 0 0 ...
## $ public_meeting : logi TRUE NA TRUE TRUE TRUE TRUE ...
## $ recorded_by : chr "GeoData Consultants Ltd" ...
## $ scheme_management : chr "VWC" "Other" "VWC" "VWC" ...
## $ scheme_name : chr "Roman" "" "Nyumba ya mungu pipe scheme" "" ...
## $ permit : logi FALSE TRUE TRUE TRUE TRUE TRUE ...
## $ construction_year : int 1999 2010 2009 1986 0 2009 0 0 0 0 ...
## $ extraction_type : chr "gravity" "gravity" "gravity" "submersible" ...
## $ extraction_type_group: chr "gravity" "gravity" "gravity" "submersible" ...
## $ extraction_type_class: chr "gravity" "gravity" "gravity" "submersible" ...
## $ management : chr "vwc" "wug" "vwc" "vwc" ...
## $ management_group : chr "user-group" "user-group" "user-group" "user-group" ...
## $ payment : chr "pay annually" "never pay" "pay per bucket" "ne ver pay" ...
## $ payment_type : chr "annually" "never pay" "per bucket" "never pay" ...
## $ water_quality : chr "soft" "soft" "soft" ...
## $ quality_group : chr "good" "good" "good" "good" ...
## $ quantity : chr "enough" "insufficient" "enough" "dry" ...
## $ quantity_group : chr "enough" "insufficient" "enough" "dry" ...
## $ source : chr "spring" "rainwater harvesting" "dam" "machine dbh" ...
## $ source_type : chr "spring" "rainwater harvesting" "dam" "borehol e" ...
## $ source_class : chr "groundwater" "surface" "surface" "groundwater" ...
## $ waterpoint_type : chr "communal standpipe" "communal standpipe" "communal standpipe multiple" "communal standpipe multiple" ...
## $ waterpoint_type_group: chr "communal standpipe" "communal standpipe" "communal standpipe" "communal standpipe" ...

```

```
## $ status_group
```

```
: chr "functional" "functional" "functional" "non functional" ...
```

Se efectúan copias del train y test, datMod y predict_x, sobre los que se aplicarán los cambios. Se eliminan las variables originales de las que se derivan las transformadas para reducir el riesgo de alta correlación/asociación de variables y con ello el sobreajuste.

La variable date_recorded ha sido correctamente asignada con el tipo Date. Aunque se intuye que tiene demasiadas categorías, y habrá que transformala.

Tres variables han sido incorrectamente asignadas como tipo int.

Dos variables han sido interpretadas como valores lógicos, se procederá a cambiarlas por caracteres.

Se cambia el tipo de la variable objetivo (status_group) por factor.

Code

```

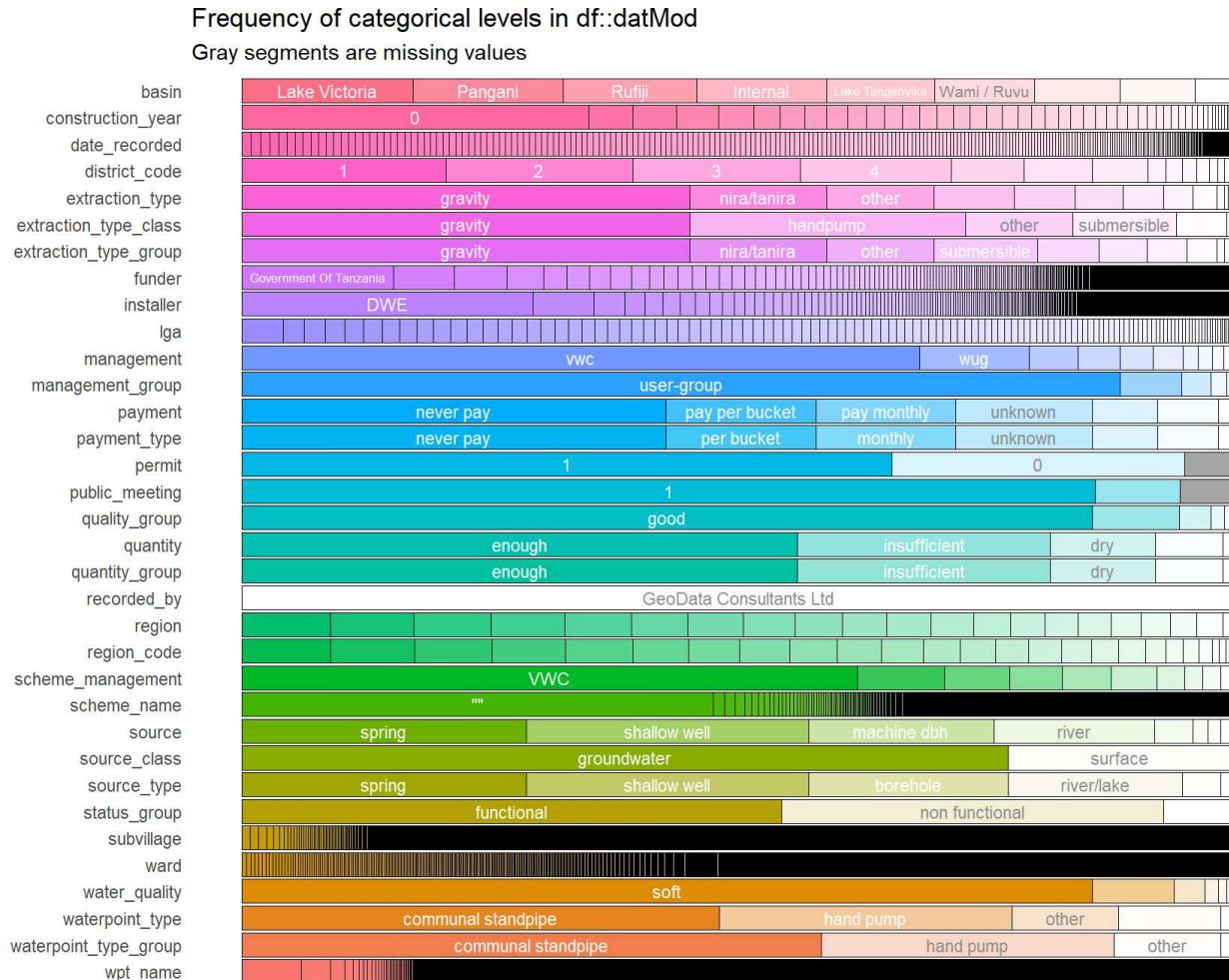
## 'data.frame': 59400 obs. of 40 variables:
## $ amount_tsh : num 6000 0 25 0 0 20 0 0 0 0 ...
## $ date_recorded : IDate, format: "2011-03-14" "2013-03-06" ...
## $ funder : chr "Roman" "Grumeti" "Lottery Club" "Unicef" ...
## $ gps_height : int 1390 1399 686 263 0 0 0 0 0 ...
## $ installer : chr "Roman" "GRUMETI" "World vision" "UNICEF" ...
## $ longitude : num 34.9 34.7 37.5 38.5 31.1 ...
## $ latitude : num -9.86 -2.15 -3.82 -11.16 -1.83 ...
## $ wpt_name : chr "none" "Zahanati" "Kwa Mahundi" "Zahanati Ya Na nyumbu" ...
## $ num_private : int 0 0 0 0 0 0 0 0 0 ...
## $ basin : chr "Lake Nyasa" "Lake Victoria" "Pangani" "Ruvuma / Southern Coast" ...
## $ subvillage : chr "Mnyusi B" "Nyamara" "Majengo" "Mahakamani" ...
## $ region : chr "Iringa" "Mara" "Manyara" "Mtwara" ...
## $ region_code : chr "11" "20" "21" "90" ...
## $ district_code : chr "5" "2" "4" "63" ...
## $ lga : chr "Ludewa" "Serengeti" "Simanjiro" "Nanyumbu" ...
## $ ward : chr "Mundindi" "Natta" "Ngorika" "Nanyumbu" ...
## $ population : int 109 280 250 58 0 1 0 0 0 ...
## $ public_meeting : chr "1" NA "1" "1" ...
## $ recorded_by : chr "GeoData Consultants Ltd" ...
## $ scheme_management : chr "VWC" "Other" "VWC" "VWC" ...
## $ scheme_name : chr "Roman" "" "Nyumba ya mungu pipe scheme" "" ...
## $ permit : chr "0" "1" "1" "1" ...
## $ construction_year : chr "1999" "2010" "2009" "1986" ...
## $ extraction_type : chr "gravity" "gravity" "gravity" "submersible" ...
## $ extraction_type_group: chr "gravity" "gravity" "gravity" "submersible" ...
## $ extraction_type_class: chr "gravity" "gravity" "gravity" "submersible" ...
## $ management : chr "vwc" "wug" "vwc" "vwc" ...
## $ management_group : chr "user-group" "user-group" "user-group" "user-group" ...
## $ payment : chr "pay annually" "never pay" "pay per bucket" "never pay" ...
## $ payment_type : chr "annually" "never pay" "per bucket" "never pay" ...
## $ water_quality : chr "soft" "soft" "soft" ...
## $ quality_group : chr "good" "good" "good" "good" ...
## $ quantity : chr "enough" "insufficient" "enough" "dry" ...
## $ quantity_group : chr "enough" "insufficient" "enough" "dry" ...
## $ source : chr "spring" "rainwater harvesting" "dam" "machine dbh" ...
## $ source_type : chr "spring" "rainwater harvesting" "dam" "borehole" ...
## $ source_class : chr "groundwater" "surface" "surface" "groundwater" ...
## $ waterpoint_type : chr "communal standpipe" "communal standpipe" "communal standpipe multiple" "communal standpipe multiple" ...
## $ waterpoint_type_group: chr "communal standpipe" "communal standpipe" "communal standpipe" "communal standpipe" ...

```

```
## $ status_group : Factor w/ 3 levels "functional","functional needs re
pair",...: 1 1 1 3 1 1 3 3 3 1 ...
```

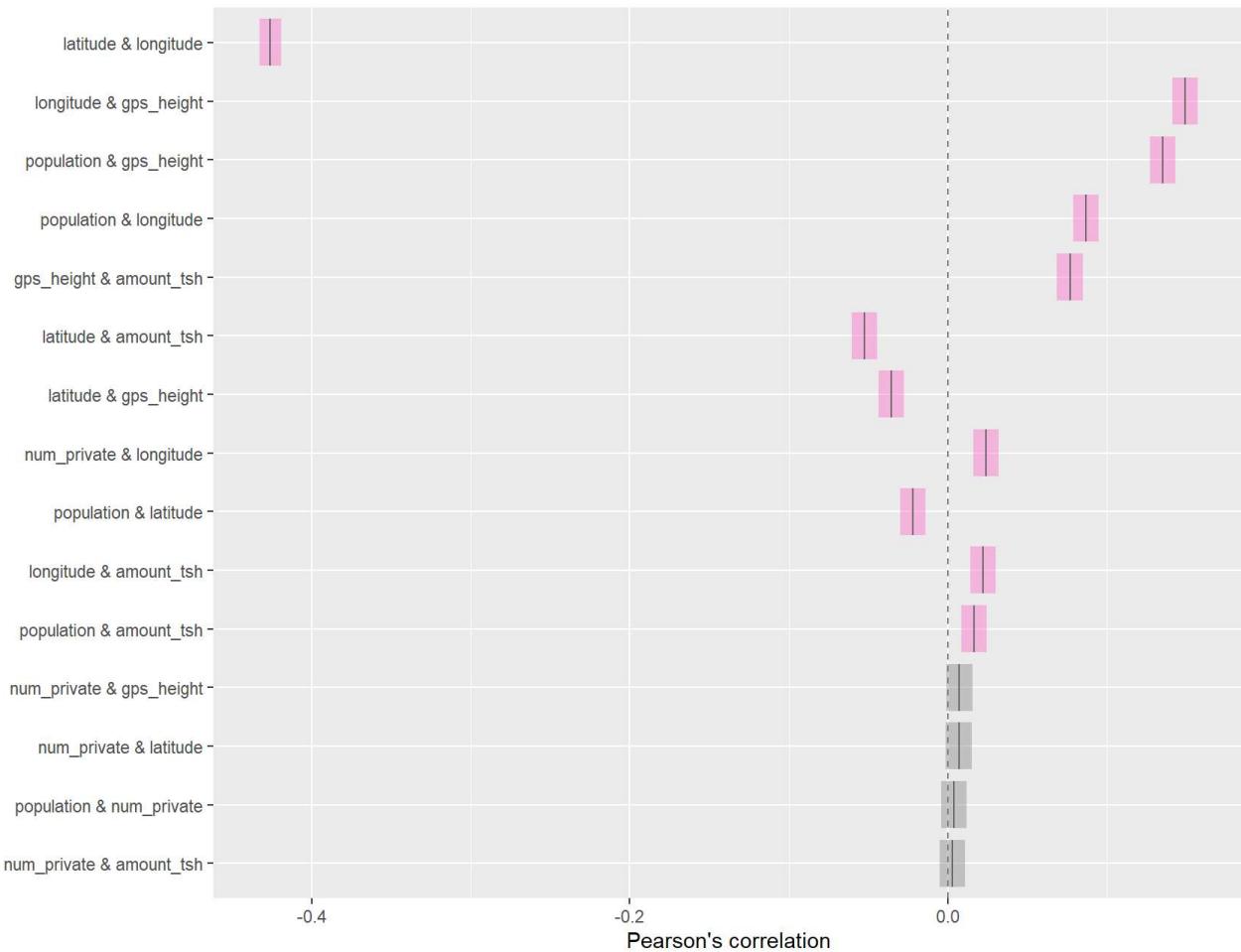
3.1.3 - EDA

[Code](#)



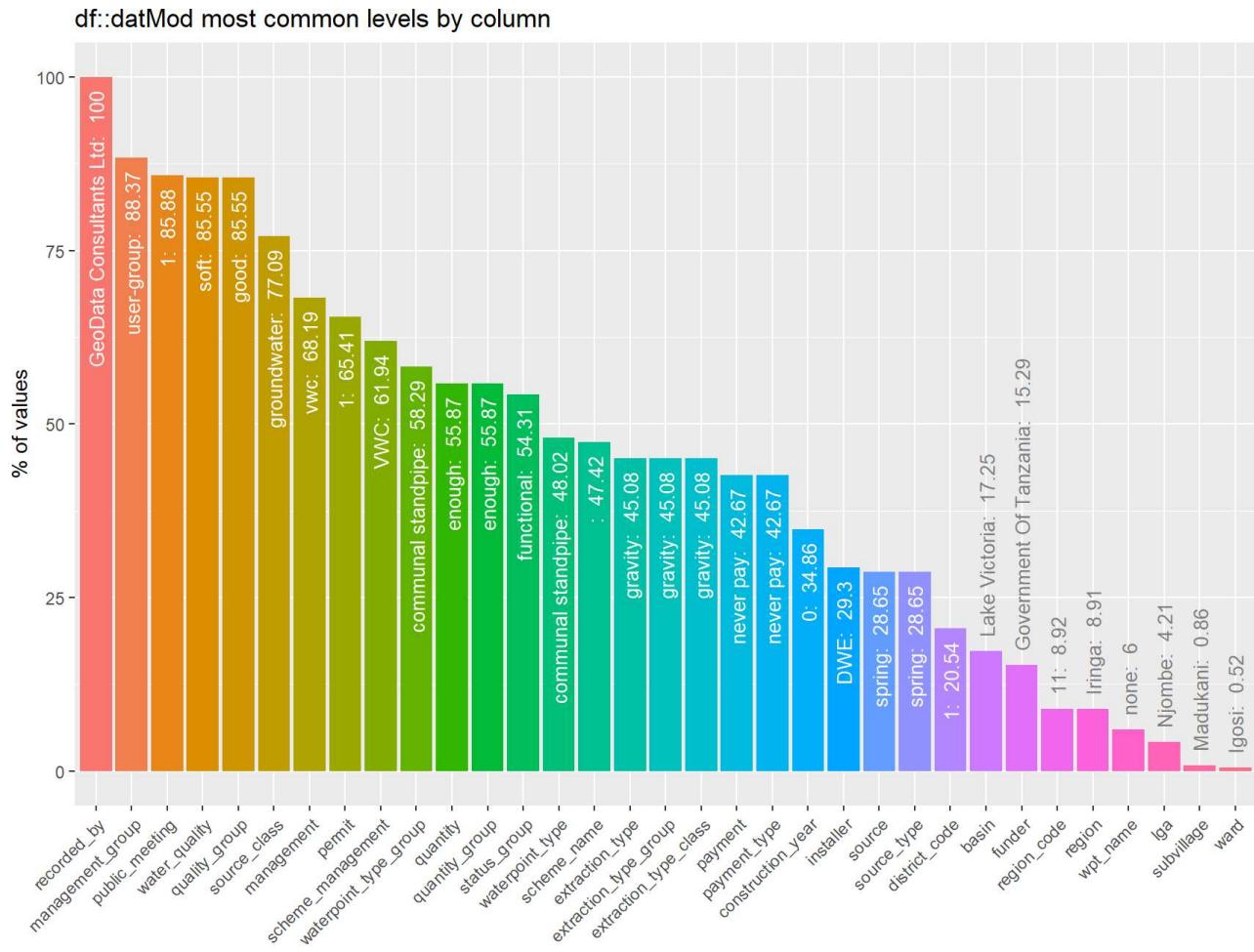
[Code](#)

Correlation of columns in df::datMod



Code

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

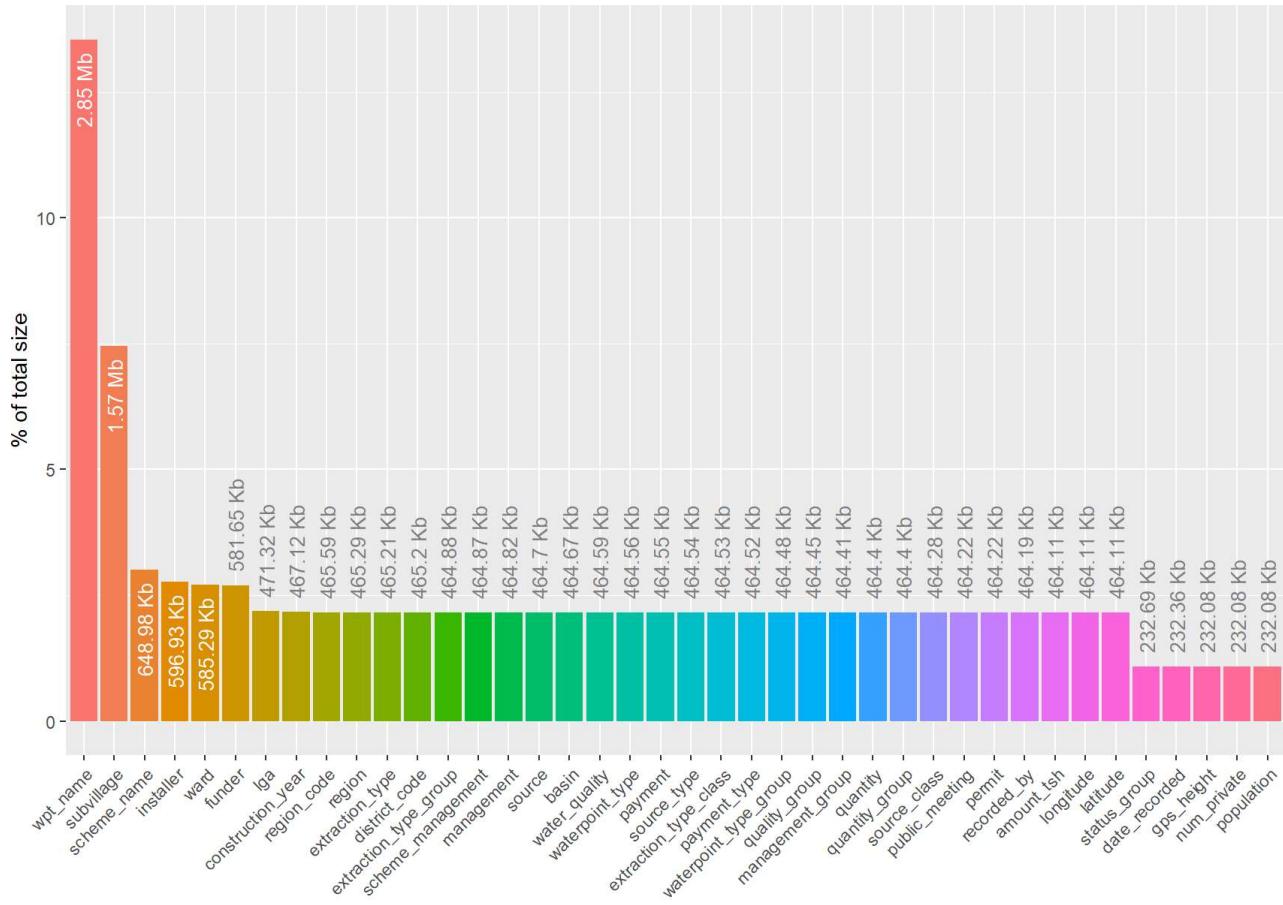


Code

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> = ## "none")` instead.
```

Column sizes in df::datMod

df::datMod has 40 columns, 59400 rows & total size of 21.09 Mb

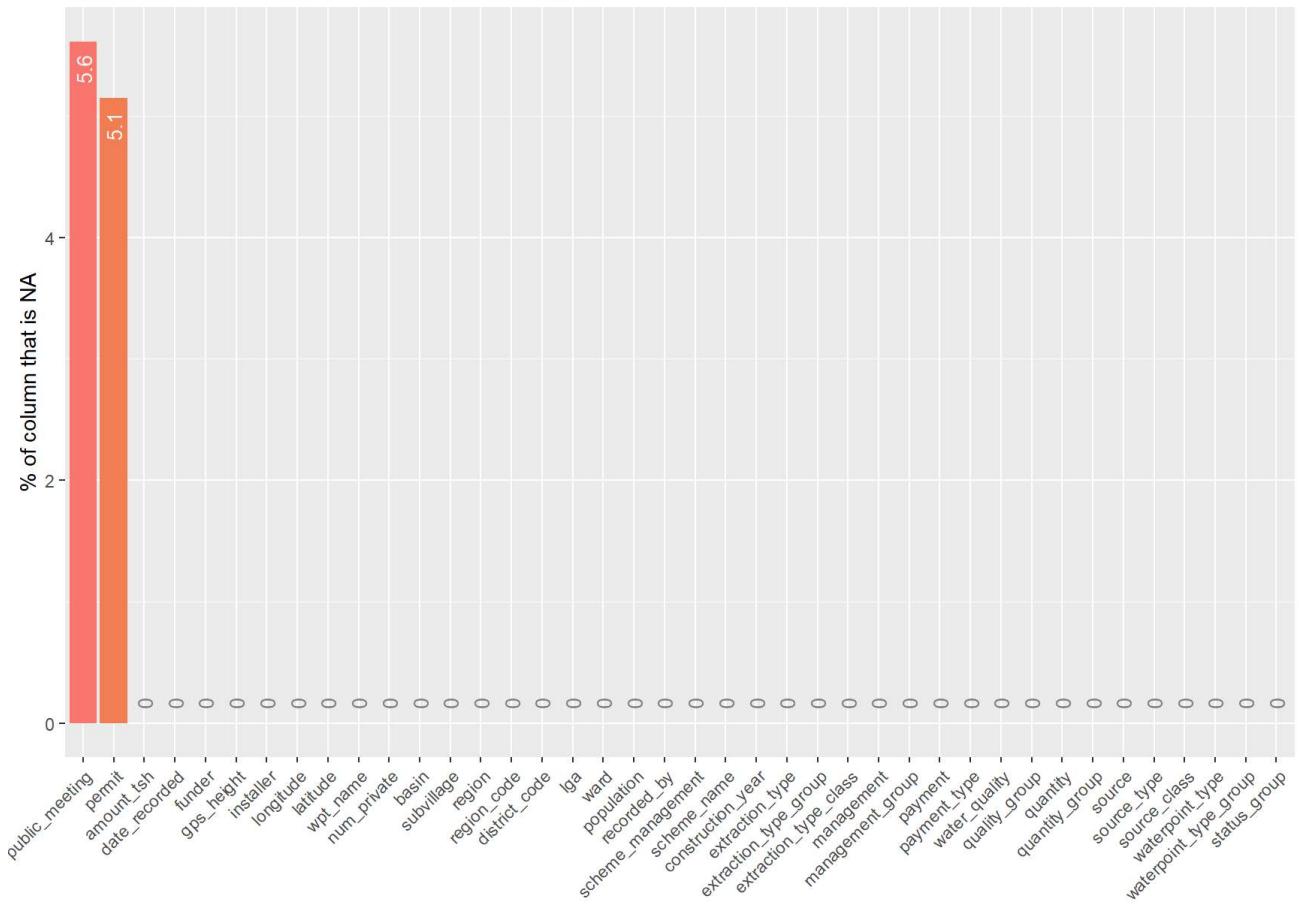


Code

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> = ## "none")` instead.
```

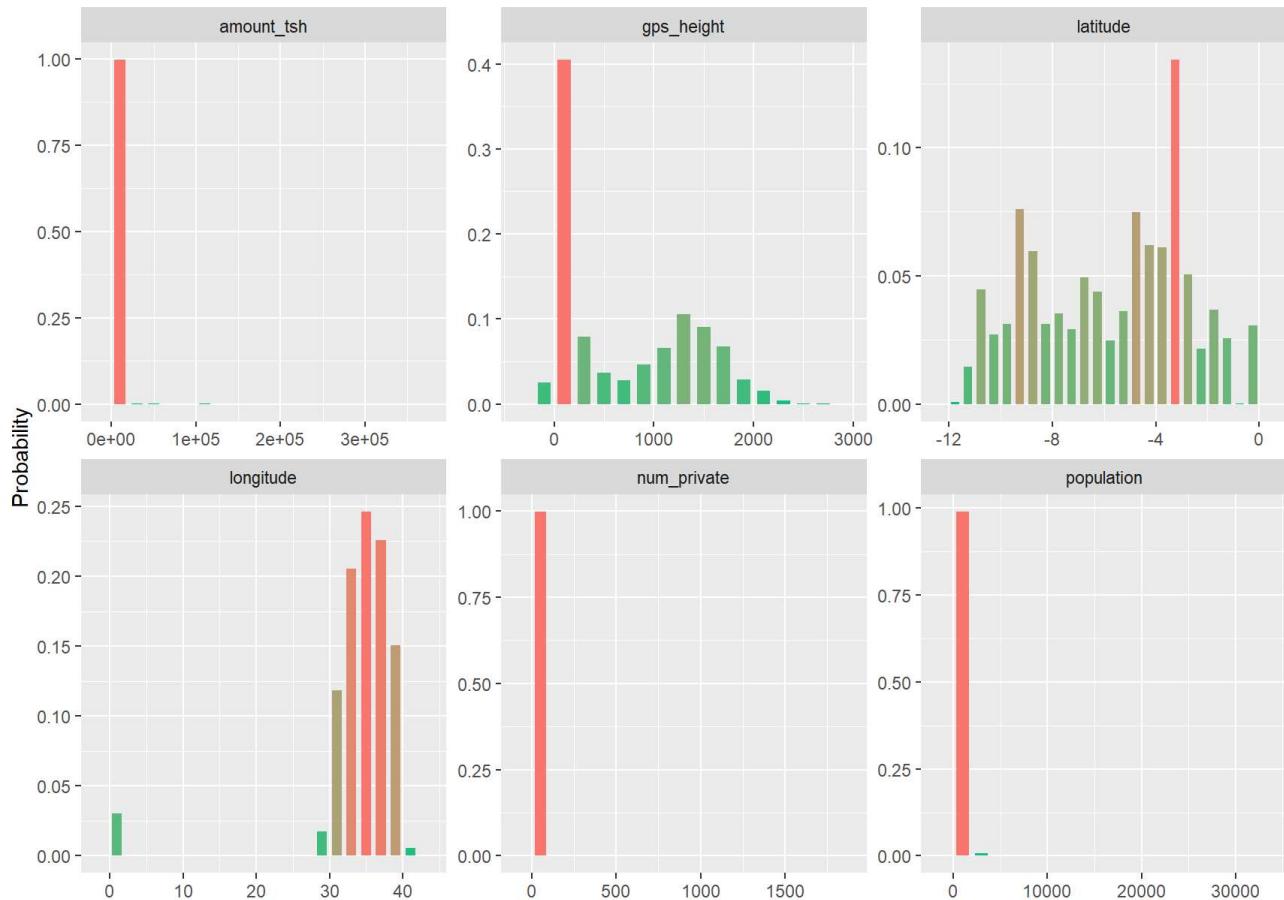
Prevalence of NAs in df::datMod

df::datMod has 40 columns, of which 2 have missing values

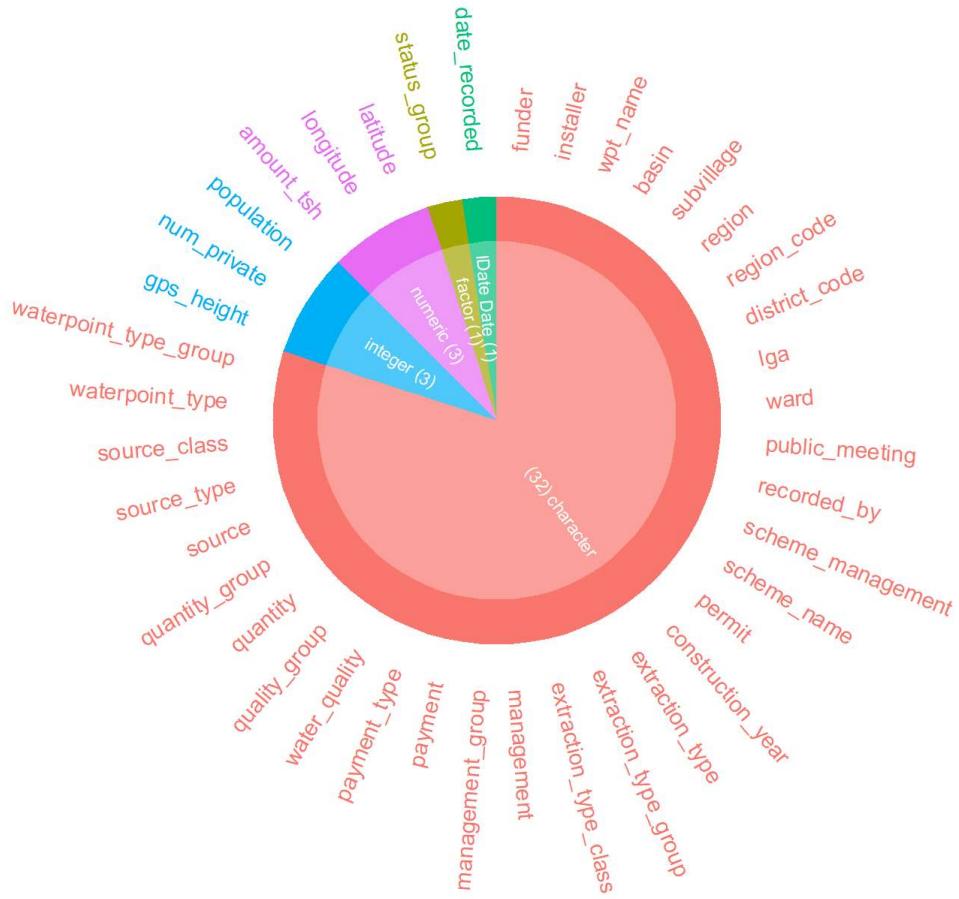


Code

Histograms of numeric columns in df::datMod



Code



Las clases están desbalanceadas en la variable objetivo, stratifiedKfolds es recomendable para reducir el overfitting.

[Code](#)

Presencia de missings en dos variables , permit y public_meeting, ambas variables categóricas. Se procede a su recategorización por 'Desconocido', al tener una frecuencia mayor de 5%.

[Code](#)

De este primer EDA se extraen las siguientes conclusiones:

- Predominio de variables categóricas. Por tanto, es posible que modelos basados en árboles den accuracies altas, dado el tipo de superficie de predicción que presentan.
- Variables cuantitativas no tienen unas correlaciones con valores absolutos por encima de 0.95, reduciendo así el riesgo de overfitting. A simple vista, no parecen tener valores fuera de rango.
- Algunas variables tienen frecuencias muy desbalanceadas o pequeñas que pueden dar lugar a overfitting y altos costes computacionales.
- La variable population figura como int, así que viene redondeada en el data set, omitiendo efectos de segundo orden (decimales).

El EDA sirve además para realizar los siguientes cambios:

- La variable numprivate es llamativa al carecer de definición y tener casi todos los valores nulos, como indica el barplot. Se prescindirá al no aportar información. Lo mismo ocurre para la variable recorded_by.

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
0	58643	98.7	98.7
1	73	0.1	0.1
2	23	0.0	0.0
3	27	0.0	0.0
4	20	0.0	0.0
5	46	0.1	0.1
6	81	0.1	0.1
7	26	0.0	0.0
8	46	0.1	0.1
9	4	0.0	0.0

1-10 of 65 rows [Previous](#) **1** [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [Next](#)

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
GeoData Consultants Ltd	59400	100	100
1 row			

[Code](#)

- Se evalúa si amount_tsh y population presentan menos de 10 categorías. De ser así, se cambiaría su tipo a categórica. Se analizan sus valores.
 - Variable amount_tsh: ¿Es plausible que un 70% de los acuíferos estén secos? Sí, el hecho de haber acuíferos registrados desde los años 60 junto con la desertización climática más el aumento de población mundial podrían dar lugar a esta situación. Por tanto, no se transformará esta variable para este primer EDA.
 - Variable population: ¿Puede estar deshabitada la zona próxima en un 36% de los acuíferos? Sí, por lo que en consonancia con la política de mínimas transformaciones para esta primera iteración, no se modifica esta variable.
- Se confirma que longitude tiene más de 10 valores, lo cual el histograma del EDA daba lugar a dudas.

- Por otro lado, el valor 0 para la construction_year, sí que parece estar mal. Se asume como missing, cuya frecuencia es 34.9%. Se formará la categoría 'Desconocido' para estos datos faltantes.

[Code](#)[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
Desconocido	20709	34.9	34.9
2010	2645	4.5	4.5
2008	2613	4.4	4.4
2009	2533	4.3	4.3
2000	2091	3.5	3.5
2007	1587	2.7	2.7
2006	1471	2.5	2.5
2003	1286	2.2	2.2
2011	1256	2.1	2.1
2004	1123	1.9	1.9
1-10 of 55 rows	Previous	1 2 3 4 5 6 Next	

[Code](#)

3.1.4 - Feature engineering

Lo mínimo indispensable en esta primera optimización. Además, al emplear modelos basados en árboles, no se estandarizarán las variables numéricas.

La variable date_recorded presenta multitud de categorías que 1) aumentan tiempo de computación y 2) incrementan el riesgo de overfitting. Se crearán nuevas variables a partir de ella y se luego se eliminará.

[Code](#)

Hay otras variables categóricas que presentan muchas categorías, como se puede observar en el EDA y en la siguiente tabla. Se les aplicará una codificación (transformación) lumping a aquellas con mayor número de categorías.

[Code](#)

	x
wpt_name	37400
subvillage	19288
scheme_name	2697
installer	2146
ward	2092

	x
funder	1898
Iga	125
fe_construction_year	55
region_code	27
region	21
district_code	20
extraction_type	18
scheme_management	13
extraction_type_group	13
management	12
source	10
basin	9
water_quality	8
extraction_type_class	7
payment	7
payment_type	7
source_type	7
waterpoint_type	7
quality_group	6
waterpoint_type_group	6
management_group	5
quantity	5
quantity_group	5
source_class	3
fe_permit	3
fe_public_meeting	3

Se hace lumping a las 6 primeras variables, las cuales presentan miles de categorías. Para esas 6 variables, se agrupan aquellas categorías con menos de un 5% de frecuencia. Si no hay ninguna categoría mayor de 5%, se elimina la variable. Obviamente todas estas transformaciones que se aplican al train se deben de efectuarse también en el test.

La variable wpt_name pasa a tener dos categorías, 'named' o 'none'. Se comprueba que ambos datasets tienen frecuencias muy parecidas, indicando (de forma no suficiente) que pueden venir ambos de una misma muestra.

Code

	n <dbl>	% <dbl>	val% <dbl>
none	3563	6.0	6.0
Shuleni	1748	2.9	2.9
Zahanati	830	1.4	1.4
Msikitini	535	0.9	0.9
Kanisani	323	0.5	0.5
Bombani	271	0.5	0.5
Sokoni	260	0.4	0.4
Ofisini	254	0.4	0.4
School	208	0.4	0.4
Shule Ya Msingi	199	0.3	0.3
1-10 of 10,000 rows	Previous	1 2 3 4 5 6 ... 1000	Next

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
named	55837	94	94
none	3563	6	6
2 rows			

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
named	13973	94.1	94.1
none	877	5.9	5.9
2 rows			

Se elimina la categoría subvillage por no tener ninguna frecuencia superior al 5%.

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
Madukani	508	0.9	0.9
Shuleni	506	0.9	0.9
Majengo	502	0.8	0.8
Kati	373	0.6	0.6
	371	0.6	0.6
Mtakuja	262	0.4	0.4
Sokoni	232	0.4	0.4
M	187	0.3	0.3
Muongano	172	0.3	0.3
Mbuyuni	164	0.3	0.3
1-10 of 10,000 rows			
Previous 1 2 3 4 5 6 ... 1000 Next			

[Code](#)

Se pasó por alto el gran número de missings en scheme_name. Al no suponer más del 50% se renombrarán como 'Desconocido' y a las demás se agruparán como 'Conocido'. De nuevo, training y set tienen frecuencias similares tras renombrar.

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
	28166	47.4	47.4
K	682	1.1	1.1
None	644	1.1	1.1
Borehole	546	0.9	0.9
Chalinze wate	405	0.7	0.7
M	400	0.7	0.7
DANIDA	379	0.6	0.6
Government	320	0.5	0.5
Ngana water supplied scheme	270	0.5	0.5
wanging'ombe water supply s	261	0.4	0.4

1-10 of 2,697 rows

Previous **1** 2 3 4 5 6 ... 270 Next[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
Conocido	31234	52.6	52.6
Desconocido	28166	47.4	47.4
2 rows			

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
Conocido	7758	52.2	52.2
Desconocido	7092	47.8	47.8
2 rows			

Agrupamos en 'Otros' en la variable installer y renombramos los missings no detectados anteriormente al pasar de 5%.

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
DWE	17402	29.3	29.3
	3655	6.2	6.2
Government	1825	3.1	3.1
RWE	1206	2.0	2.0

	n <dbl>	% <dbl>	val% <dbl>
Commu	1060	1.8	1.8
DANIDA	1050	1.8	1.8
KKKT	898	1.5	1.5
Hesawa	840	1.4	1.4
0	777	1.3	1.3
TCRS	707	1.2	1.2

1-10 of 2,146 rows Previous **1** 2 3 4 5 6 ... 215 Next

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
Otros	38343	64.6	64.6
DWE	17402	29.3	29.3
Desconocido	3655	6.2	6.2

3 rows

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
Otros	9624	64.8	64.8
DWE	4349	29.3	29.3
Desconocido	877	5.9	5.9

3 rows

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
Igosi	307	0.5	0.5
Imalinyi	252	0.4	0.4
Siha Kati	232	0.4	0.4
Mdandu	231	0.4	0.4
Nduruma	217	0.4	0.4
Kitunda	203	0.3	0.3
Mishamo	203	0.3	0.3

	n <dbl>	% <dbl>	val% <dbl>
Msindo	201	0.3	0.3
Chalinze	196	0.3	0.3
Maji ya Chai	190	0.3	0.3
1-10 of 2,092 rows	Previous 1 2 3 4 5 6 ... 210 Next		

[Code](#)

Se modifica la variable funder. De nuevo, training y test son muy parecidos en las proporciones. Como se dijo anteriormente, todas estas similitudes evidenciadas en las frecuencias son condiciones necesarias, que no suficientes, para afirmar que training y test provienen de la misma población.

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
Government Of Tanzania	9084	15.3	15.3
	3635	6.1	6.1
Danida	3114	5.2	5.2
Hesawa	2202	3.7	3.7
Rwssp	1374	2.3	2.3
World Bank	1349	2.3	2.3
Kkkt	1287	2.2	2.2
World Vision	1246	2.1	2.1
Unicef	1057	1.8	1.8
Tasaf	877	1.5	1.5
1-10 of 1,898 rows	Previous 1 2 3 4 5 6 ... 190 Next		

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
Otros	43567	73.3	73.3
Government Of Tanzania	9084	15.3	15.3
Desconocido	3635	6.1	6.1
Danida	3114	5.2	5.2
4 rows			

[Code](#)

	n <dbl>	% <dbl>	val% <dbl>
Otros	10973	73.9	73.9
Government Of Tanzania	2215	14.9	14.9
Desconocido	869	5.9	5.9
Danida	793	5.3	5.3
4 rows			

Code

3.1.5 - Modelos

Se construyen modelos a partir de los inputs modificados en esta primera iteración, datMod, a evaluar en el igualmente transformado conjunto test predict_x. Los algoritmos a analizar son solo dos, ranger (random forest) y gbm (boosting) por restricciones con la fecha de entrega.

Se divide datMod en un conjunto train y otro validation, siendo las proporciones 80% y 20% respectivamente tras un shuffle previo. Los modelos se entrena con train y se evalúan con validation. Posteriormente, se probarán con el conjunto test aquellos modelos que presenten una accuracy más alta en validación.

Dividir el train en dos subconjuntos tiene la desventaja de que se entrena con una población menor pero se consigue tener una muestra completamente independiente para evaluar antes de subir las predicciones al sitio web, donde solo se permite 3 soluciones al día. De no haber tal restricción no se habría establecido el conjunto validación. No obstante, dado al número alto de muestras proporcionados, quitar un 20% no tendría que suponer una fuerte penalización en cuanto a accuracy.

Debido a las restricciones de deadline y capacidad computacional, se evalúan dos técnicas de remuestreo, ninguna o 'none' (un solo fold) y stratifiedKfolds. El none sirve para estimar los tiempos de computación requeridos, lo cual condiciona el número de folds y repeticiones asumible. Por otro lado, al haber variables con muchas categorías, podría reducir el overfitting también en este problema en particular. No se aplica grid search en los hiperparámetros.

Uso de la función de caret createDataPartition para hacer el remuestreo tipo stratifiedKfolds (5 folds) con shuffle.

En esta primera iteración no se hacen repeticiones con ningún modelo.

Se empieza con un ranger, con parámetros con valores o bien por defecto o bien 'comunes', al ser usados en códigos de profesores del máster en Data Science que el autor está cursando. No hay grid search paramétrico en esta primera iteración para ningún algoritmo.

En aras de mantener una cierta consistencia, los valores de parámetros se mantendrán iguales para aquellos compartidos por los algoritmos. Obviamente, dicha consistencia no es perfecta al tratarse de distintos algoritmos. Por ejemplo, num.trees = 100 tanto para ranger como gbm.

Code

```
## Loading required package: foreach
```

[Code](#)

```
## Loading required package: iterators
```

[Code](#)

```
## Loading required package: parallel
```

[Code](#)

```
## 23.5 sec elapsed
```

Siendo conservadores, ha tardado unos 30 segundos en ejecución. Esto da una idea de cómo de exhaustivos podemos ser ante futuras evaluaciones con cv (stratified) y grid search.

```
## Ranger result
##
## Call:
##   ranger(status_group ~ ., data = my_train, num.trees = 100, importance = "impurity",
##          write.forest = TRUE, min.node.size = 1, splitrule = "gini",      verbose =
##          TRUE, classification = TRUE)
##
## Type:                      Classification
## Number of trees:            100
## Sample size:                47522
## Number of independent variables: 38
## Mtry:                       6
## Target node size:           1
## Variable importance mode:   impurity
## Splitrule:                  gini
## OOB prediction error:       19.02 %
```

[Code](#)

```
## Ranger result
##
## Call:
##   ranger(status_group ~ ., data = my_train, num.trees = 100, importance = "impurity",
##          write.forest = TRUE, min.node.size = 1, splitrule = "gini",      verbose =
##          TRUE, classification = TRUE)
##
## Type:                      Classification
## Number of trees:           100
## Sample size:                47522
## Number of independent variables: 38
## Mtry:                      6
## Target node size:          1
## Variable importance mode:  impurity
## Splitrule:                  gini
## OOB prediction error:      19.02 %
```

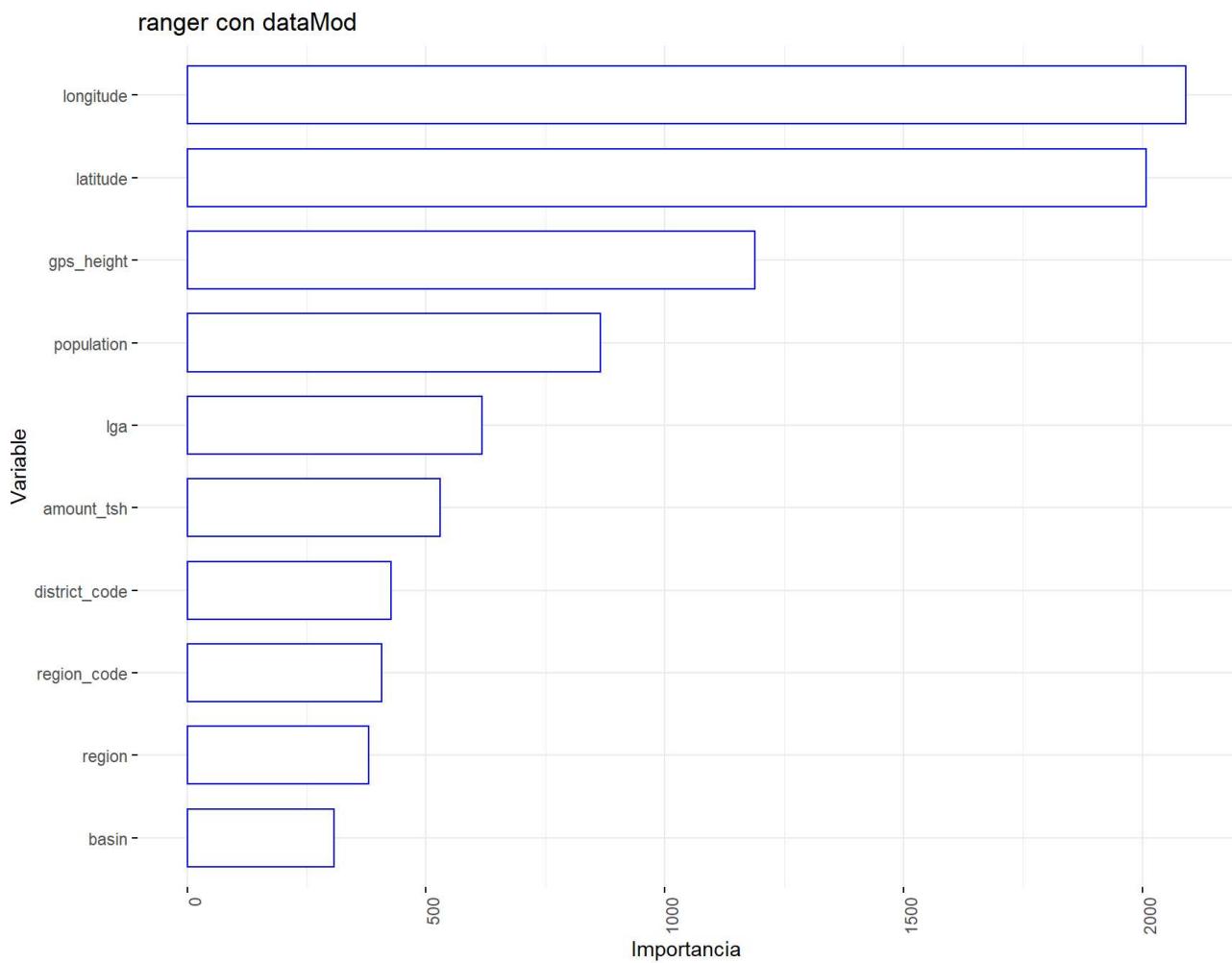
[Code](#)

	Length	Class	Mode
## predictions	47522	factor	numeric
## num.trees	1	-none-	numeric
## num.independent.variables	1	-none-	numeric
## mtry	1	-none-	numeric
## min.node.size	1	-none-	numeric
## variable.importance	38	-none-	numeric
## prediction.error	1	-none-	numeric
## forest	9	ranger.forest	list
## confusion.matrix	9	table	numeric
## splitrule	1	-none-	character
## treetype	1	-none-	character
## call	10	-none-	call
## importance.mode	1	-none-	character
## num.samples	1	-none-	numeric
## replace	1	-none-	logical

[Code](#)

El OOB prediction error sale 19.02 %. Esto da una aproximación de lo que se puede esperar al predecir nuevos resultados, accuracies de 80%. A continuación se muestra el barplot con la importancia de variables y la accuracy del validation test.

[Code](#)



Code

```

## Confusion Matrix and Statistics
##
##                                     Reference
## Prediction                      functional  functional needs repair non functional
##   functional                      5774                  126          551
##   functional needs repair        442                   295          126
##   non functional                 941                   67          3556
##
## Overall Statistics
##
##                               Accuracy : 0.8103
##                               95% CI : (0.8032, 0.8173)
##   No Information Rate : 0.6025
##   P-Value [Acc > NIR] : < 2.2e-16
##
##                               Kappa : 0.644
##
## McNemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                                     Class: functional Class: functional needs repair
## Sensitivity                      0.8068                  0.60451
## Specificity                      0.8566                  0.95013
## Pos Pred Value                   0.8951                  0.34183
## Neg Pred Value                   0.7452                  0.98248
## Prevalence                       0.6025                  0.04108
## Detection Rate                  0.4861                  0.02484
## Detection Prevalence            0.5431                  0.07266
## Balanced Accuracy                0.8317                  0.77732
##
##                                     Class: non functional
## Sensitivity                      0.8401
## Specificity                      0.8681
## Pos Pred Value                   0.7791
## Neg Pred Value                   0.9074
## Prevalence                       0.3564
## Detection Rate                  0.2994
## Detection Prevalence            0.3842
## Balanced Accuracy                0.8541

```

Una accuracy en la validación por encima del 80% (0.8103) resulta prometedor, en vista a los resultados obtenidos por los demás participantes en el concurso. **La puntuación obtenida al subir sus predicción al sitio web es de 0.8156.**

Code

A continuación se aplica stratifiedkfold para ranger, así como se evalúa gbm con los dos mismos remuestreos, de nuevo sin grid search. Es decir, se aplica al GBM un training control none y stratifiedkfold (5 particiones) con shuffle previo. Las accuracies obtenidas en validación son más bajas: 0.7586, 0.7246 y 0.6789 respectivamente. Por otra parte los errores OOB también son más bajos, sobre el 18% en vez del 19.02% del primer modelo.

Parece que los modelos sufren de sobreajuste al bajar la accuracy en la validación y el OOB en el entrenamiento, al usar GBM y una técnica de remuestreo más avanzada que la simple partición train-validation. Hay que recordar que aún quedan muchas variables categóricas con categorías con bajas frecuencias. Se aplicará lumping de nuevo en las variables categóricas en la iteración 2, para ver si se puede paliar este problema de overfitting al usar kfolds.

GBM presenta tiempos de ejecución mayores que ranger, aunque del mismo orden un x3. Por ejemplo GBM con stratified tardó 30 minutos en terminar. Las dos simulaciones de GBM dieron peores accuracies que sus homólogos en ranger.

[Code](#)

```
## 192.02 sec elapsed
```

[Code](#)

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   f   fnr   nf
##           f  5966    15  470
##           fnr  668    65 130
##           nf  1571    13 2980
##
## Overall Statistics
##
##                 Accuracy : 0.7586
##                   95% CI : (0.7508, 0.7663)
##       No Information Rate : 0.6908
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                 Kappa : 0.5253
##
## McNemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                         Class: f Class: fnr Class: nf
## Sensitivity          0.7271  0.698925  0.8324
## Specificity          0.8680  0.932287  0.8091
## Pos Pred Value       0.9248  0.075319  0.6529
## Neg Pred Value       0.5874  0.997458  0.9180
## Prevalence           0.6908  0.007830  0.3014
## Detection Rate       0.5023  0.005472  0.2509
## Detection Prevalence 0.5431  0.072655  0.3842
## Balanced Accuracy    0.7975  0.815606  0.8208
```

[Code](#)

```

## ranger variable importance
##
##   only 20 most important variables shown (out of 398)
##
##                                     Overall
## quantityenough                    100.00
## extraction_type_groupother       99.34
## quantity_groupenough              90.72
## longitude                         85.88
## waterpoint_typeother              81.72
## latitude                          81.43
## waterpoint_type_groupother       72.93
## amount_tsh                        67.49
## extraction_typeother              67.22
## gps_height                        65.19
## extraction_type_classother        59.97
## payment_type_never pay            56.38
## waterpoint_typecommunal standpipe 44.63
## population                        41.82
## fe_dianum                         41.08
## fe_mes                            37.86
## extraction_type_classhandpump    34.78
## waterpoint_typehand pump          34.51
## extraction_typegravity            30.69
## fe_funderGovernment Of Tanzania  28.78

```

[Code](#)[Code](#)[Code](#)[Code](#)[Code](#)[Code](#)

3.2 - Iteración 2

La primera iteración consistió en lanzar modelos con las mínimas transformaciones posibles, para evitar excesivos overfitting y tiempos de computación. En la iteración 2 se realizan tres transformaciones extra al conjunto de variables input:

- Cambiar algunas variables derivadas de la fecha a categóricas.
- Tratamiento de los outliers en las variables numéricas.
- Lumping en algunas variables categóricas que aún tengan categorías infrarepresentadas .

Se aplican unos límites menos agresivos que en la primera iteración, dado que el número de categorías ya ha sido reducido en la iteración 1 y el número de observaciones es bastante grande. Dicho límite es 2% tanto para recategorizar como para considerar un valor como outlier.

Lo primero, se transforman algunas variables derivadas de la fecha a categóricas, ya que se presuponen valores discretos. El sufijo '2' que se añade para renombrar estas variables representa la iteración donde estas se han transformado.

Los conjuntos train/test derivados de esta segunda iteración pasarán a llamarse datMod2 y predict_x2 respectivamente.

Code

Code

3.2.1 - Variables numéricas

Las variables amount_tsh y population presentan algunos valores muy altos, observando la mediana y la media del conjunto train. Se comprueba si se tratan de outliers o de 'datos grandes', en función del porcentaje sobre el train.

Esta comprobación también se extiende, con menor riesgo de presencia de outliers, al resto de variables numéricas.

Code

Se crean unos boxplots estandarizados con el valor máximo, para ayudar a visualizar la proporción de posibles outliers.

Code

Las variables amount_tsh y population presentan una porcentaje amplio de datos grandes, mayor del 5%. Por tanto, no pueden borrarse todos directamente. Sin embargo, viendo los boxplots estandarizados, se aprecia que hay unas observaciones muy grandes que podrían perjudicar los modelos.

Por tanto, se va a evaluar cómo quedan los boxplots poniendo como NAs aquellos valores en el percentil 98% e imputando con las medianas de las variables en el train. Estos boxplots están normalizados con los valores máximos del conjunto resultante de la iteración 1, datMod.

Como se mencionó anteriormente, se aplican los valores medianos del datMod2 a predict_x2 de cara a la imputación.

Code

Se aprecia como los datos están más compactos, comparando la escala del eje y con respecto a los boxplots anteriores.

3.2.2 - Variables categóricas

Antes de aplicar las siguientes transformaciones, se guarda el datMod2 con el tratamiento de outliers ya realizado en una nueva variable (datMod2_num) para la iteración 3. De esta forma, se puede comparar el efecto de eliminar outliers de forma independiente.

Se realiza un lumping, agrupando aquellas variables que presenten categorías con frecuencias inferiores a 2%. En este proceso se excluye la variable día del mes, fe_dianum2.

[Code](#)

Comprobación rápida del train y test, ambos conjuntos contienen el mismo número de columnas y similares proporciones. Como se esperaba, el gráfico de abajo muestra menos categorías que en la iteración 1. Es decir, está menos 'segmentado'.

[Code](#)

3.2.3 - Modelos

Mismos algoritmos, hiperparámetros y remuestreos que la iteración 1. Solo cambian las transformaciones añadidas al train y test. Los resultados son peores en general, la accuracy en validación cae en 3 de los 4 modelos respecto a sus análogos de la iteración 1. Hay una mejora de unos tres puntos porcentuales en el caso de ranger con stratifiedKfolds.

- Ranger sin remuestreo: 0.8133
- Ranger y stratifiedKfolds: 0.79
- GBM sin remuestreo: 0.7089
- GBM y stratifiedKfolds: 0.6741

[Code](#)

[Code](#)

[Code](#)

[Code](#)

[Code](#)

3.3 - Iteración 3

Se aprovecha una copia intermedia de los conjuntos train y test de la iteración 2, datMod2_num, para evaluar resultados sin haber tratado las variables categóricas en la iteración anterior. De nuevo, mismos algoritmos, hiperparámetros y técnicas de remuestreo que en las dos iteraciones anteriores, lo único que cambian son las transformaciones realizadas a las variables numéricas originales.

Las accuracies salen con valores intermedios respecto a los obtenidos en las iteraciones 1 y 2. Esto, mirando en retrospectiva, tiene sentido al ser una etapa intermedia entre las transformaciones aplicadas en las iteraciones 1 y 2.

- Ranger sin remuestreo: 0.8063

- Ranger y stratifiedKfolds: 0.7627
- GBM sin remuestreo: 0.7246
- GBM y stratifiedKfolds: 0.6739

[Code](#)[Code](#)[Code](#)[Code](#)[Code](#)

3.4 - Análisis de los resultados parciales y feature engineering

Los tres conjuntos — evaluados con los mismos algoritmos, valores de los hiperparámetros y técnicas de remuestreo — dan lugar a resultados similares en términos de accuracy.

En el caso de stratifiedKfolds, el esfuerzo en reducir categorías en la iteración 2 ha dado una mejor accuracy en la validación respecto a la iteración 1 en ranger, aunque ha decrecido ligeramente para GBM. No obstante, el mejor resultado ha sido con el primer modelo en la iteración 1, ranger sin remuestreo.

Las variables más relevantes, aquellas que han sido usadas más veces, son por este orden:

- quantity
- waterpoint_type
- extraction_type
- latitude
- longitude

Las tres primeras son categóricas y las dos últimas son numéricas. Crear nuevas variables que puedan repercutir en una mejora en la accuracy no es un proceso trivial y requiere, siempre que sea posible, tiempo de una persona experta en el problema a evaluar. En este caso, se aplicará una transformación comúnmente usada para datos geográficos $fe_lonlat = \sqrt{latitude^2 + longitude^2}$ con la que se ampliarán los conjuntos.

Otras transformaciones usuales son recategorizar las variables tipo fecha por estaciones, etapas del día (mañana, tarde, noche) etc. No se usarán para las siguientes iteraciones al no haber figurado como relevantes en los barplots.

3.5 - Iteración 4

Aprovechando que al final se ha contado con el suficiente tiempo. Se evalúan los conjuntos de las iteraciones 1, 2 y 3 con xgboost solo con stratifiedKfolds. Las accuracies obtenidas son algo mejores que gbm pero inferiores a ranger - 0.6901, 0.6808 y 0.6801 respectivamente -

precisando del doble de tiempo en converger con los parámetros usados que no tienen por qué ser los óptimos. De nuevo, los valores de los hiperparámetros replican sus semejantes en los otros dos algoritmos, por ejemplo, `min_child_weight=c(1)` y `eta=c(0.01)`

En base a los resultados obtenidos de accuracy y tiempo computacional, teniendo siempre en cuenta las limitaciones por dealine y capacidad de computación, se procederá a evaluar modelos usando exclusivamente ranger.

[Code](#)[Code](#)[Code](#)

3.6 - Iteración 5

Se realizan dos grid searches con el conjunto de la tercera iteración, `datMod2_num`, ampliado con la variable anteriormente mencionada `fe_lonlat`. Los parámetros a los que se le ha aplicado el mallado son `mtry`, `min.node.size`, `num.trees` y número de folds. Su rango de valores incluyen los originales de la iteración 1 y otros superiores - por ejemplo, `min.node.size = c(fit_r1$min.node.size,10,20)` - para ver si se puede reducir el overfitting. La técnica de remuestreo vuelve a ser `stratifiedKfolds`, el segundo remuestreo se hace con repetición.

Se usa este conjunto en vez el de la iteración 1, `datMod`, al haber dado mejores resultados con remuestreo para ranger.

Se consiguen mejores valores de accuracy en la validación así como mayores valores de errores OOB. 0.8003/18.73 y 0.8041/18.85 respectivamente. Esto indica que se ha mejorado el problema de overfitting. Sin embargo, se sigue logrando una menor accuracy en la validación que usando ranger sin remuestro para la iteración 1, cuyo valor fue recordemos de 0.8103.

La mejor configuración es `nodesize = 1` con los valores más altos de `mtry` y `num.trees` proporcionados, 20 y 300 respectivamente, La variable `lonlat` se preserva al haber sido relevante.

Viendo los resultados obtenidos con `datMod2_num`, se procede a hacer lo mismo en la siguiente iteración con el conjunto de la iteración 2 - nombrado como `datMod2`, que comprendía tratamiento de outliers más un segundo lumping - el cual ha dado los mejores resultados ha dado en validación cruzada.

[Code](#)[Code](#)[Code](#)[Code](#)

3.7 - Iteración 6

Grid search del segundo conjunto, que presentaba un tratamiento de outliers en algunas variables numéricas así como un lumping extra para algunas variables categóricas. En este caso se toma como punto de partida los valores altos de mtry y número de arboles de la iteración anterior. Uso de stratifiedKfold con repetición. La accuracy de validación es de 0.8073, todavía inferior al mejor modelo obtenido. El mtry óptimo ha sido un valor central del mallado por lo que no se sigue explorando con valores superiores.

[Code](#)
[Code](#)

3.8 - Iteración 7

Viendo que las accuracies obtenidas siguen siendo más bajas al modelo óptimo y teniendo en cuenta el deadline cercano, se procede a probar a cambiar los hiperparámetros del mejor modelo, ranger sin remuestreo con el conjunto datMod, con aquellos valores de los hiperparámetros obtenidos de la iteración 6.

Se prueban dos conjuntos de variables del original de la iteración 1 (datMod) ampliado con fe_lonlat y con el datMod prístino. Las accuracies obtenidas son similares al óptimo pero algo por debajo, 0.8001 y 0.8043 respectivamente.

[Code](#)
[Code](#)

3.9 - Iteración 8

Hasta ahora el mejor valor de la accuracy ha sido con el primer modelo de la iteración 1. A modo de concluir este primer análisis del problema, se procede a hacer una optimización local (valores de los hiperparámetros cercanos a ese óptimo).

Se logra una accuracy muy similar en el conjunto de validación 0.8099, **que da lugar a una pequeña mejora el conjunto test consiguiendo un 0.8173.**

[Code](#)

El siguiente es el mejor modelo obtenido.

[Code](#)

```

## Confusion Matrix and Statistics
##
##                                     Reference
## Prediction                      functional  functional needs repair non functional
##   functional                      5795                  125          531
##   functional needs repair        445                   295          123
##   non functional                 973                   61          3530
##
## Overall Statistics
##
##                               Accuracy : 0.8099
##                               95% CI : (0.8027, 0.8169)
##   No Information Rate : 0.6073
##   P-Value [Acc > NIR] : < 2.2e-16
##
##                               Kappa : 0.6426
##
## McNemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                                     Class: functional Class: functional needs repair
## Sensitivity                      0.8034                  0.61331
## Specificity                      0.8594                  0.95016
## Pos Pred Value                   0.8983                  0.34183
## Neg Pred Value                   0.7387                  0.98311
## Prevalence                        0.6073                  0.04050
## Detection Rate                   0.4879                  0.02484
## Detection Prevalence            0.5431                  0.07266
## Balanced Accuracy                0.8314                  0.78173
##
##                                     Class: non functional
## Sensitivity                      0.8437
## Specificity                      0.8656
## Pos Pred Value                   0.7734
## Neg Pred Value                   0.9106
## Prevalence                        0.3522
## Detection Rate                   0.2972
## Detection Prevalence            0.3842
## Balanced Accuracy                0.8546

```

Code

3.10 - Resultados

A continuación se muestran las puntuaciones logradas en el concurso, situando al modelo óptimo encontrado cerca del top 10%.

SUBMISSIONS

Score	Submitted by	Timestamp
0.8156	miguel_benayas	2021-09-23 09:42:53 UTC
0.8133	miguel_benayas	2021-09-25 14:38:19 UTC
0.0000	miguel_benayas	2021-09-27 09:49:27 UTC
0.8134	miguel_benayas	2021-09-27 09:54:58 UTC
0.8173	miguel_benayas	2021-09-27 17:47:38 UTC

Resultados obtenidos en conjunto test

Submissions

BEST	CURRENT RANK	# COMPETITORS	SUBS. MADE
0.8173	1577	12507	0 of 3

Clasificación en el concurso

4 - Conclusiones

Se ha seguido un esquema iterativo atendiendo al paradigma de ramificación y poda, debido a las restricciones de tiempo y capacidad de computación.

En la primera iteración se presenta el conjunto transformado, datMod, que servirá como base al de las iteraciones 2 y 3. En estas tres iteraciones se evalúan random forest (en su variante implementación ranger) y gbm, con parámetros ‘típicos’ sin hacer grid search. Se concluye que ranger da lugar a mejores resultados más rápido para las configuraciones evaluadas en este problema.

En la iteración 4 se evalúa xgboost y se vuelve a concluir que ranger presenta mejores resultados más rápido. Por lo que, por motivos de tiempo, sólo se hacen modelos con algoritmo ranger.

Con estos resultados parciales, se analiza la importancia de las variables y se crea una nueva fe_lonlat. Se hace un grid search no fino en las iteraciones 5 y 6 para los conjuntos de las iteraciones 2 y 3, ya que presentan mejores accuracies que la iteración 1 con remuestreo. Las accuracies mejoran pero no superan al modelo de ranger sin remuestreo de la iteración 1.

Viendo que el modelo óptimo sigue siendo el ranger sin remuestreo de la iteración 1, y teniendo en cuenta el cercano deadline, se vuelve a ese modelo en la iteración 7, probando con los valores de los hiperparámetros obtenidos en las iteraciones 5 y 6. Los resultados en al accuracy siguen por debajo del mejor modelo.

Finalmente, se prueba una optimización local de este mejor modelo, mejorando la predicción del test unas centésimas en la iteración 8.

Dos conclusiones se extraen de este primer análisis del problema.

- Importancia del feature engineering. Con las transformaciones de variables adecuadas, algoritmos no tan punteros como el random forest - xgboost es el dominante en Kaggle - pueden lograr buenos resultados, como se observa en los resultados de esta clasificación multinomial. Unas transformaciones resultarán beneficiosas para ciertos algoritmos y perniciosas para otros.
- A veces menos es más. En este problema con propensión al overfitting por el alto número categorías una simple partición train-validation ha logrado mejores resultados que el stratifiedKfolds.

Estas dos conclusiones evidencian - de forma no robusta - la no trivialidad de la Ciencia de Datos y como los métodos más usados no siempre tienen por qué ser los óptimos.