19.- Feature Transforms_CU_53_02_spi_v_01

June 13, 2023

#

CU53_impacto de las políticas de inversión en sanidad, infraestructuras y promoción turística en el SPI

Citizenlab Data Science Methodology > III - Feature Engineering Domain *** > # 19.- Feature Transforms

Feature Transform (Polynomial Features Transform) is the process to create new features by raising existing features to an exponent, in order to see if they improve model performance, when the input features interact in unexpected and often nonlinear ways.

0.1 Tasks

Feature Transforms

- Perform Polynomial Features Transform
- Evaluate a KNN model

0.2 Consideraciones casos CitizenLab programados en R

- Algunas de las tareas de este proceso se han realizado en los notebooks del proceso 05 Data Collection porque eran necesarias para las tareas ETL. En esos casos, en este notebook se referencia al notebook del proceso 05 correspondiente
- Otras tareas típicas de este proceso se realizan en los notebooks del dominio IV al ser más eficiente realizarlas en el propio pipeline de modelización.
- Por tanto en los notebooks de este proceso de manera general se incluyen las comprobaciones necesarias, y comentarios si procede
- Las tareas del proceso se van a aplicar solo a los archivos que forman parte del despliegue, ya que hay muchos archivos intermedios que no procede pasar por este proceso
- El nombre de archivo del notebook hace referencia al nombre de archivo del proceso 05 al que se aplica este proceso, por eso pueden no ser correlativa la numeración
- Las comprobaciones se van a realizar teniendo en cuenta que el lenguaje utilizado en el despliegue de este caso es R

0.3 File

- Input File: CU_53_14_02_spi
- Output File: No aplica

0.3.1 Encoding

Con la siguiente expresión se evitan problemas con el encoding al ejecutar el notebook. Es posible que deba ser eliminada o adaptada a la máquina en la que se ejecute el código.

[1]:
```
Sys.setlocale(category = "LC_ALL", locale = "es_ES.UTF-8")
```

'LC_CTYPE=es_ES.UTF-8;LC_NUMERIC=C;LC_TIME=es_ES.UTF-8;LC_COLLATE=es_ES.UTF-8;LC_MONETARY=es_ES.UTF-8;LC_MESSAGES=en_US.UTF-8;LC_PAPER=es_ES.UTF-8;LC_NAME=C;LC_ADDRESS=C;LC_TELEPHONE=C;LC_MEASUREMENT 8;LC_IDENTIFICATION=C'

0.4 Settings

0.4.1 Libraries to use

[9]:
```
library(readr)
library(dplyr)
library(tidyr)
library(forcats)
library(lubridate)
library(caret)
```

```
Loading required package: ggplot2

Loading required package: lattice
```

0.4.2 Paths

[3]:
```
iPath <- "Data/Input/"
oPath <- "Data/Output/"
```

0.5 Data Load

OPCION A: Seleccionar fichero en ventana para mayor comodidad

Data load using the {tcltk} package. Ucomment the line if using this option

[4]:
```
# file_data <- tcltk::tk_choose.files(multi = FALSE)
```

OPCION B: Especificar el nombre de archivo

[4]:
```
iFile <- "CU_53_14_02_spi.csv"
file_data <- paste0(iPath, iFile)

if(file.exists(file_data)){
    cat("Se leerán datos del archivo: ", file_data)
} else{
    warning("Cuidado: el archivo no existe.")
```

```
}
```

```
Se leerán datos del archivo:  Data/Input/CU_53_14_02_spi.csv
```

Data file to dataframe Usar la función adecuada según el formato de entrada (xlsx, csv, json, …)

[5]:
```
data <- read_csv(file_data)
```

```
Rows: 2028 Columns: 18
  Column specification



Delimiter: ","
dbl (17): rank_score_spi, score_spi, score_bhn, score_fow, score_opp,
score_…
lgl  (1): is_train

 Use `spec()` to retrieve the full column specification for this
data.
 Specify the column types or set `show_col_types = FALSE` to quiet
this message.
```

Estructura de los datos:

[6]:
```
data |> glimpse()
```

```
Rows: 2,028
Columns: 18
$ rank_score_spi <dbl> 80, 97, 46, 84, 99, 150, 74, 105, 36,
143, 154, 69, 168…
$ score_spi      <dbl> 0.234430921, -0.247745795,
0.644506738, -0.070067671, -…
$ score_bhn      <dbl> 0.4097479, 0.1290857, 0.5753443,
0.4274030, 0.3293843, …
$ score_fow      <dbl> 0.22131225, -0.67087093, 0.55485637,
-0.04224433, -0.26…
$ score_opp      <dbl> 0.040287945, -0.176082184,
0.684177595, -0.557195503, -…
$ score_nbmc     <dbl> 0.4417846, -0.4611703, 0.4195220,
0.2610630, 0.5105377,…
$ score_ws       <dbl> 0.5398626, 0.3861578, 0.6209430,
0.1056095, 0.1274964, …
$ score_sh       <dbl> 0.6722671, 0.1921862, 0.7589734,
0.5545286, 0.6229812, …
$ score_ps       <dbl> -0.451618611, 0.297686264,
0.192315395, 0.822032832, -0…
```

```
$ score_abk      <dbl> 0.038575928, -1.291936532,
0.767026841, -0.404764773, -…
$ score_aic      <dbl> 0.65139291, -1.02544160, 0.37750377,
-0.38712186, 0.831…
$ score_hw       <dbl> -0.17460539, 0.46862381, 0.28376095,
0.31816759, -0.652…
$ score_eq       <dbl> 0.145913492, -0.124265238,
0.496988695, 0.680773448, -0…
$ score_pr       <dbl> 0.49893581, 0.02236525, 0.89103387,
-0.40632266, -0.458…
$ score_pfc      <dbl> -0.17705492, 0.12172033, 0.31379064,
-0.42914696, -0.28…
$ score_incl     <dbl> -0.412651603, 0.380048297,
0.997036708, 0.009655802, -0…
$ score_aae      <dbl> 0.13726735, -1.14969465, 0.14456184,
-1.20857192, -0.38…
$ is_train       <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE, TRUE, T…
```

Muestra de los primeros datos:

[7]:
```
data |> slice_head(n = 5)
```

A spec_tbl_df: 5 × 18

	rank_score_spi <dbl>	score_spi <dbl>	score_bhn <dbl>	score_fow <dbl>	score_opp <dbl>	score_nbmc <dbl>	s
	80	0.23443092	0.4097479	0.22131225	0.04028795	0.4417846	0
	97	-0.24774579	0.1290857	-0.67087093	-0.17608218	-0.4611703	0
	46	0.64450674	0.5753443	0.55485637	0.68417760	0.4195220	0
	84	-0.07006767	0.4274030	-0.04224433	-0.55719550	0.2610630	0
	99	-0.16212549	0.3293843	-0.26860033	-0.50440793	0.5105377	0

0.6 Polynomial Features Transform

0.6.1 Evaluating a KNN model

[8]:
```
train_set <- subset(data[data$is_train == TRUE, ], select = -is_train)
train_set <- select_if(train_set, is.numeric)
test_set <- subset(data[data$is_train == FALSE, ], select = -is_train)
test_set <- select_if(test_set, is.numeric)
```

Evaluating without tranform

[12]:
```
# Entrenar un modelo KNN sin transformación
knn_fit <- train(rank_score_spi ~ ., data = train_set, method = "knn",
                 trControl = trainControl(method = "cv", number = 10))

# Evaluar el modelo KNN sin transformación
knn_pred <- predict(knn_fit, newdata = test_set[,-1])
postResample(knn_pred, test_set$rank_score_spi)
```

RMSE 2.70180552342683 **Rsquared** 0.996926515917185 **MAE** 1.85975308641975

Evaluating Polynomial Features Tranform

[19]:
```
library(recipes)

# Definir una receta para la transformación
rec <- recipe(rank_score_spi ~ ., data = train_set) %>%
  step_poly(all_predictors(), degree = 2)

# Preparar los datos con la receta
prep_rec <- prep(rec)
data_poly <- bake(prep_rec, new_data = train_set)

# Entrenar un modelo KNN con transformación de características polinomiales
knn_fit_poly <- train(rank_score_spi ~ ., data = data_poly, method = "knn",
                      trControl = trainControl(method = "cv", number = 10))

# Evaluar el modelo KNN con transformación de características polinomiales
knn_pred_poly <- predict(knn_fit_poly, newdata = data_poly)
postResample(knn_pred_poly, test_set$rank_score_spi)
```

RMSE <NA> **Rsquared** 0.00244718169130076 **MAE** <NA>

Exploring varying degree in the polynomial transform. Range of degree in the polynomial transform

[10]:
```
# Range of degree in the polynomial transform
deg_i=1
deg_f=3
```

Operation

[21]:
```
# Definir el rango de grados
deg_i <- 1
deg_f <- 10

# Inicializar una lista para almacenar los modelos y sus métricas de rendimiento
models <- list()
performance <- data.frame()

# Bucle para entrenar un modelo para cada grado en el rango
for (deg in deg_i:deg_f) {

  # Definir una receta para la transformación
  rec <- recipe(rank_score_spi ~ ., data = train_set) %>%
    step_poly(all_predictors(), degree = deg)

  # Preparar los datos con la receta
```

```r
  prep_rec <- prep(rec)
  data_poly <- bake(prep_rec, new_data = train_set)

  # Entrenar un modelo KNN
  knn_fit <- train(rank_score_spi ~ ., data = data_poly, method = "knn",
                   trControl = trainControl(method = "cv", number = 10),
                   tuneLength = 10)

  # Almacenar el modelo en la lista de modelos
  models[[paste0("knn_poly_", deg)]] <- knn_fit

  # Evaluar el modelo y almacenar su rendimiento
  knn_pred <- predict(knn_fit, newdata = data_poly)
  perf <- postResample(knn_pred, data$rank_score_spi)

  # Almacenar las métricas de rendimiento en el dataframe
  performance <- rbind(performance, cbind(Degree = deg, t(perf)))
}

# Ver las métricas de rendimiento de todos los modelos
print(performance)
```

```
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
```

```
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
Warning message in pred - obs:
"longer object length is not a multiple of shorter object length"
```

	Degree	RMSE	Rsquared	MAE
1	1	31.37243	NA	12.43634
2	2	31.33104	NA	12.42022
3	3	31.36896	NA	12.45079
4	4	31.38685	NA	12.44970
5	5	31.36487	NA	12.45030
6	6	31.36820	NA	12.43825
7	7	31.37252	NA	12.45641
8	8	31.34772	NA	12.44269
9	9	31.34350	NA	12.46504
10	10	31.32901	NA	12.47512

0.6.2 Polynomial Features Transform

Select features to Normalizate

[ ]:

Operation

[ ]:

[11]:
```
# histograms of the variables
```

Select Data to concatenate to transformed data

[12]:
```
# Data to concatenate to transformed data
```

Operation

[13]:
```
# Generate new data set
```

0.7 Data Save

- Solo si se han hecho cambios
- No aplica

Identificamos los datos a guardar

[14]:
```
data_to_save <- data
```

Estructura de nombre de archivos:

- Código del caso de uso, por ejemplo “CU_04”
- Número del proceso que lo genera, por ejemplo ”_06”.
- Resto del nombre del archivo de entrada
- Extensión del archivo

Ejemplo: ”CU_04_06_01_01_zonasgeo.json, primer fichero que se genera en la tarea 01 del proceso 05 (Data Collection) para el caso de uso 04 (vacunas) y que se ha transformado en el proceso 06

Importante mantener los guiones bajos antes de proceso, tarea, archivo y nombre

0.7.1 Proceso 19

[22]:
```
caso <- "CU_53"
proceso <- '_19'
tarea <- "_02"
archivo <- ""
proper <- "_spi"
extension <- ".csv"
```

OPCION A: Uso del paquete “tcltk” para mayor comodidad

- Buscar carpeta, escribir nombre de archivo SIN extensión (se especifica en el código)
- Especificar sufijo2 si es necesario
- Cambiar datos por datos_xx si es necesario

[16]:
```
# file_save <- paste0(caso, proceso, tarea, tcltk::tkgetSaveFile(), proper,␣
 ↪extension)
# path_out <- paste0(oPath, file_save)
# write_csv(data_to_save_xxxx, path_out)

# cat('File saved as: ')
# path_out
```

OPCION B: Especificar el nombre de archivo

- Los ficheros de salida del proceso van siempre a Data/Output/.

[17]:
```
# file_save <- paste0(caso, proceso, tarea, archivo, proper, extension)
# path_out <- paste0(oPath, file_save)
```

```
# write_csv(data_to_save_xxxx, path_out)

# cat('File saved as: ')
# path_out
```

Copia del fichero a Input Si el archivo se va a usar en otros notebooks, copiar a la carpeta Input

[18]:
```
# path_in <- paste0(iPath, file_save)
# file.copy(path_out, path_in, overwrite = TRUE)
```

0.8 REPORT

A continuación se realizará un informe de las acciones realizadas

0.9 Main Actions Carried Out

- Si eran necesarias se han realizado en el proceso 05 por cuestiones de eficiencia
- Las transformaciones no implican mejoras, por tanto no se generan nuevos datos
- O bien se hacen en el dominio IV o V para integrar en el pipeline de modelización

0.10 Main Conclusions

- Los datos están listos para la modelización y despliegue

0.11 CODE TO DEPLOY (PILOT)

A continuación se incluirá el código que deba ser llevado a despliegue para producción, dado que se entiende efectúa operaciones necesarias sobre los datos en la ejecución del prototipo

Description

- No hay nada que desplegar en el piloto, ya que estos datos son estáticos o en todo caso cambian con muy poca frecuencia, altamente improbable durante el proyecto.

CODE

[ ]:
```

```