

Kobe Bryant Shot Selection

Proyecto Kaggle

Laura Rodríguez Navas

Introducción

Este trabajo lleva a cabo un proyecto completo de Ciencia de Datos donde vamos a analizar, transformar, modelar y evaluar un conjunto de datos de *Kaggle*. Concretamente, para este trabajo se ha usado un conjunto de datos que describe los aciertos y los fallos de lanzamientos a canasta del jugador de baloncesto Kobe Bryant durante los 20 años de su carrera en la NBA (<https://www.kaggle.com/c/kobe-bryant-shot-selection/data/>).

El conjunto de datos contiene 30697 instancias y un gran número de variables explicativas (11 discretas y 14 numéricas). Estas 25 variables (incluyendo clase a predecir “*shot_made_flag*”) se centran en la descripción cualitativa y cuantitativa de multitud de aspectos de cada uno de los lanzamientos de Kobe Bryant.

```
## 'data.frame':   30697 obs. of  25 variables:
## $ action_type      : Factor w/ 57 levels "Alley Oop Dunk Shot",...: 27 27 27 27 6 ..
## $ combined_shot_type: Factor w/ 6 levels "Bank Shot","Dunk",...: 4 4 4 4 2 4 5 4 4 ..
## $ game_event_id    : int   10 12 35 43 155 244 251 254 265 294 ...
## $ game_id          : int   20000012 20000012 20000012 20000012 20000012 20000012 2..
## $ lat              : num   34 34 33.9 33.9 34 ...
## $ loc_x            : int   167 -157 -101 138 0 -145 0 1 -65 -33 ...
## $ loc_y            : int   72 0 135 175 0 -11 0 28 108 125 ...
## $ lon              : num  -118 -118 -118 -118 -118 ...
## $ minutes_remaining: int   10 10 7 6 6 9 8 8 6 3 ...
## $ period           : int    1 1 1 1 2 3 3 3 3 3 ...
## $ playoffs         : int    0 0 0 0 0 0 0 0 0 0 ...
## $ season           : Factor w/ 20 levels "1996-97","1997-98",...: 5 5 5 5 5 5 5 5 ..
## $ seconds_remaining: int   27 22 45 52 19 32 52 5 12 36 ...
## $ shot_distance    : int   18 15 16 22 0 14 0 2 12 12 ...
## $ shot_made_flag    : int   NA 0 1 0 1 0 1 NA 1 0 ...
## $ shot_type        : Factor w/ 2 levels "2PT Field Goal",...: 1 1 1 1 1 1 1 1 1 ..
## $ shot_zone_area    : Factor w/ 6 levels "Back Court(BC)",...: 6 4 3 5 2 4 2 2 4 2 ..
## $ shot_zone_basic   : Factor w/ 7 levels "Above the Break 3",...: 5 5 5 5 6 5 6 6 3..
## $ shot_zone_range   : Factor w/ 5 levels "16-24 ft.", "24+ ft.",...: 1 3 1 1 5 3 5 5..
## $ team_id          : int  1610612747 1610612747 1610612747 1610612747 1610612747 ..
## $ team_name         : Factor w/ 1 level "Los Angeles Lakers": 1 1 1 1 1 1 1 1 1 ..
## $ game_date        : Factor w/ 1559 levels "1996-11-03","1996-11-05",...: 311 311 ..
## $ matchup          : Factor w/ 74 levels "LAL @ ATL","LAL @ BKN",...: 29 29 29 29 ..
## $ opponent         : Factor w/ 33 levels "ATL","BKN","BOS",...: 26 26 26 26 26 26 ..
## $ shot_id          : int    1 2 3 4 5 6 7 8 9 10 ...
```

La tarea de este trabajo es predecir si los lanzamientos a canastas de Kobe Bryant entraron o no en el aro, es decir, los lanzamientos acertados (atributo “*shot_made_flag*”). Del conjunto de datos se han eliminado 5000 valores de este atributo (representados como valores faltantes). Estos datos estarán en el conjunto de evaluación (test) sobre el cual se realizará la predicción.

Una vez descargados los datos, es necesario dividir el conjunto de datos en un conjunto de datos de entrenamiento y un conjunto de datos de evaluación (test). Para ello, primero analizamos la existencia de esos valores faltantes nombrados anteriormente (atributo “*shot_made_flag*”), que como hemos comentado serán los valores que tendremos que predecir y que estarán en el conjunto de datos de test pero no en el conjunto de datos de entrenamiento.

A continuación, podemos observar este proceso.

```
train <- data[!is.na(data$shot_made_flag), ]  
any(is.na(train))
```

```
## [1] FALSE
```

```
## int [1:25697] 0 1 0 1 0 1 1 0 0 1 ...
```

```
test <- data[is.na(data$shot_made_flag), ]  
any(is.na(test))
```

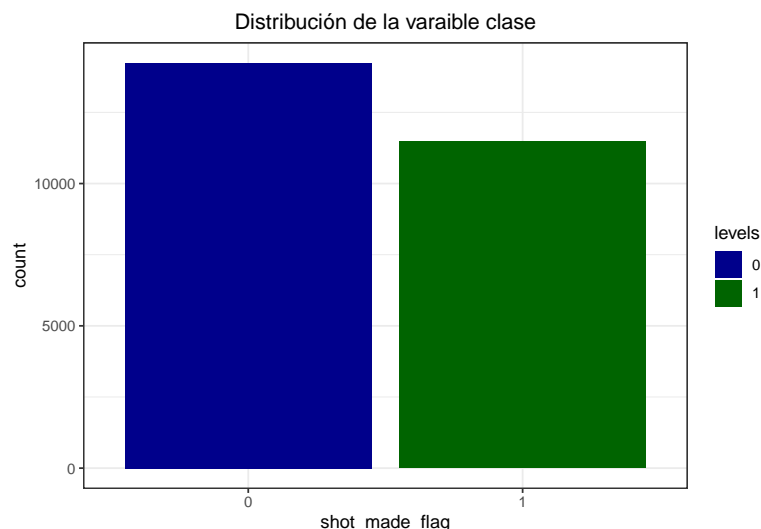
```
## [1] TRUE
```

```
## int [1:5000] NA NA NA NA NA NA NA NA NA NA ...
```

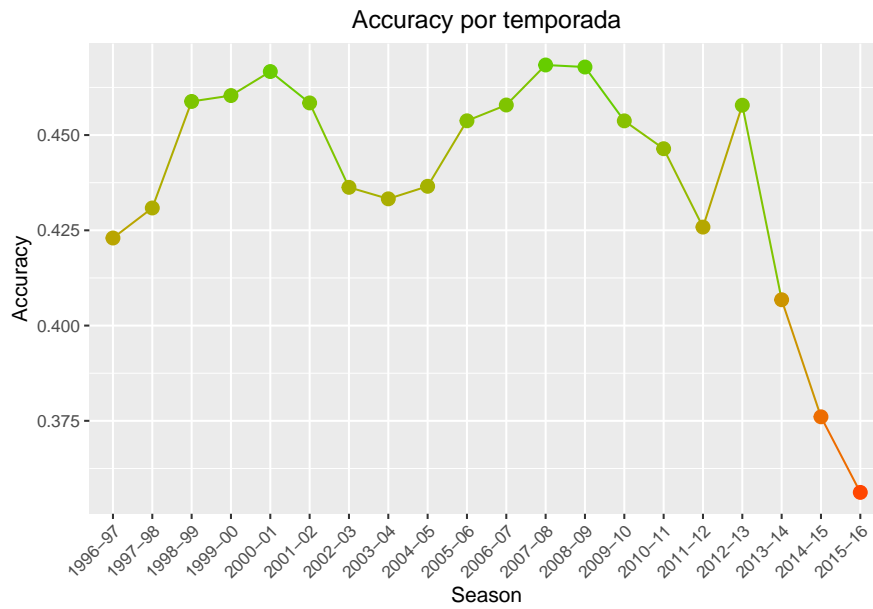
Una vez dividido el conjunto de los datos, es necesario realizar un análisis del conjunto de datos, así como un proceso de exploración, transformación y limpieza de estos datos con el objetivo de resaltar información útil para la fase de modelado. Este análisis nos permitirá controlar la presencia de valores fuera de rango, una idea inicial de la forma que tienen los datos, etc. así como las relaciones entre los distintos atributos. Aunque para sintetizar el análisis realizado solo se comentará aquello que se ha considerado más interesante durante éste.

Exploración de datos

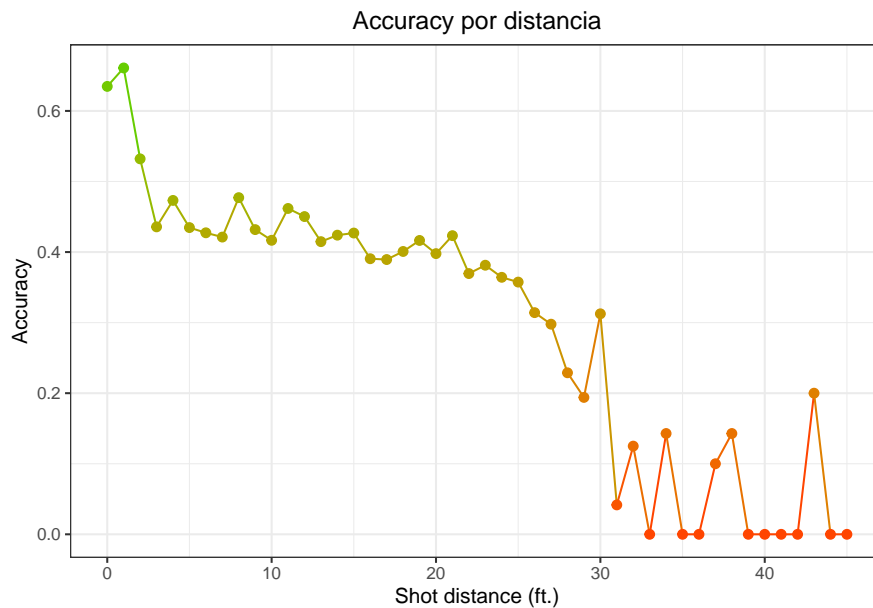
Empezamos analizando visualmente la variable de clase a predecir (atributo “*shot_made_flag*”), que es binaria y que se distribuye de manera bastante equitativa. Vemos que el número de canastas que no entraron en el aro es superior al número de canastas que sí que entraron. Así que, intentaremos averiguar si esto puede estar relacionado con la gran lesión que tuvo Kobe Bryant durante la temporada 2013-14.



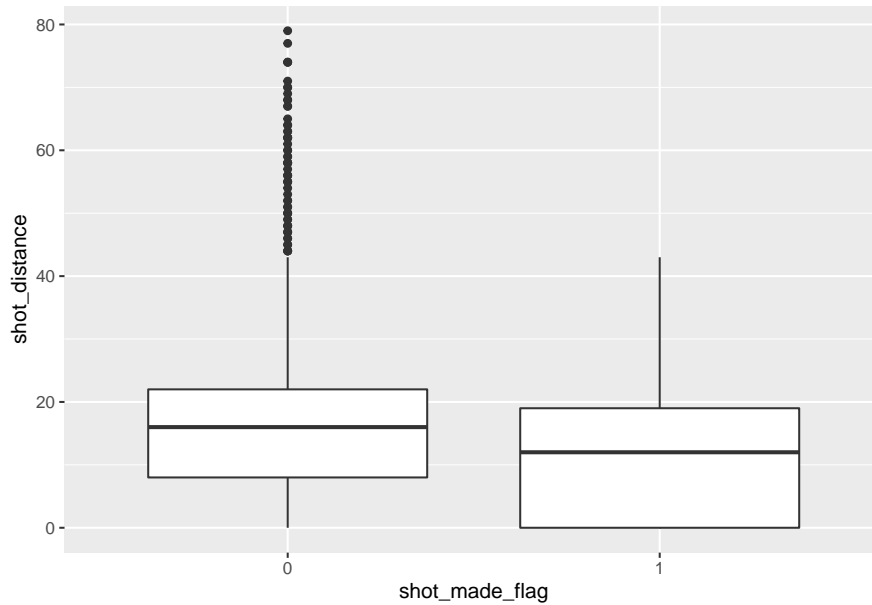
A continuación, analizamos visualmente la precisión de los lanzamientos realizados por temporada (atributo “*season*”), y vemos que a partir de la temporada 2013-14 la precisión de los lanzamientos baja drásticamente. ¿Así que, una gran cantidad de los lanzamientos que no entraron en el aro están correlacionados a la gran lesión que tuvo en la temporada 2013-14? Podría ser.



En el siguiente gráfico analizamos visualmente la precisión de los lanzamientos respecto a la distancia de tiro (atributo “*shot_distance*”), porqué en diferentes exploraciones de datos que han realizado otros usuarios en *Kaggle*, se ha podido observar que el atributo “*shot_distance*” contiene valores fuera de rango que podríamos eliminar en el apartado de limpieza de datos y que nos sería muy beneficioso para reducir los fallos durante la predicción en base a la distancia de los lanzamientos. Los valores fuera de rango podrían encontrarse a partir de los lanzamientos realizados a más de 30 (ft.) ya que la precisión de estos baja drásticamente a partir de este punto.



Y en siguiente grafico podemos observar la existencia de esos valores fuera de rango que estábamos buscando, en el conjunto de datos de entrenamiento. Y que valores exactamente se encuentran fuera de rango son a partir de 40 (ft.). Así que, en el apartado de limpieza de datos serán eliminados los valores del atributo *shot_distance* superiores a 40 (ft.).



Anteriormente, hemos analizado visualmente la precisión de los lanzamientos por temporada, los lanzamientos respecto a la distancia y hemos encontrado valores fuera de rango en el conjunto de datos que tendremos que eliminar si queremos realizar una buena predicción de la variable clase “*shot_made_flag*”.

Al trabajar con un conjunto de datos real, debemos tener en cuenta el hecho de que algunos datos pueden faltar o estar dañados, por lo tanto, es crucial realizar los procesos de transformación y limpieza del conjunto de datos para obtener un buen ajuste del modelo y una mejor capacidad predictiva.

Transformación de datos

La primera transformación que tenemos que realizar es la categorización del atributo “*shot_made_flag*”, porque es la variable clase a predecir e inicialmente es de tipo entero.

```
## int [1:25697] 0 1 0 1 0 1 1 0 0 1 ...
## Factor w/ 2 levels "0","1": 1 2 1 2 1 2 2 1 1 2 ...
```

Si nos fijamos en los atributos “*minutes_remaining*” y “*seconds_remaining*” podríamos realizar la segunda transformación sobre el conjunto de datos ya que la información que contienen la podríamos combinar, los minutos del atributo “*minutes_remaining*” los podríamos convertir a segundos y sumar-los con los segundos del atributo “*seconds_remaining*”. La nueva información combinada la guardaríamos en un nuevo atributo “*time_remaining*”. El proceso se muestra a continuación.

```
train$time_remaining <- train$minutes_remaining * 60 + train$seconds_remaining
test$time_remaining <- test$minutes_remaining * 60 + test$seconds_remaining
```

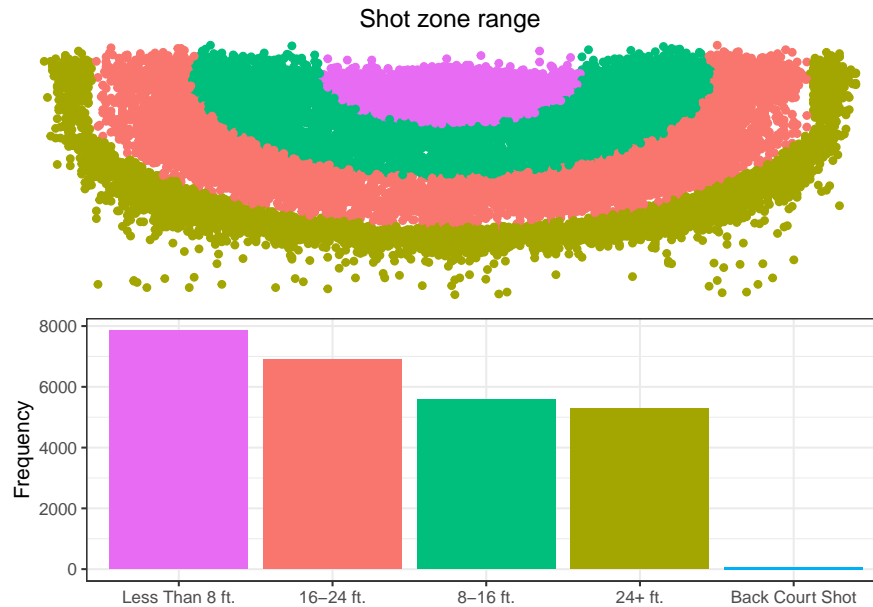
El siguiente paso es normalizar los atributos “*time_remaining*” y “*shot_distance*”. También se hace al final con los valores de la variable clase “*shot_made_flag*”, ya que vamos a predecir probabilidades y se tiene que hacer la normalización antes del modelado.

Sabemos que vamos a predecir probabilidades por el fichero de ejemplo que nos facilita Kaggle.

```
## shot_id shot_made_flag
## 1      1             0.5
## 2      8             0.5
## 3     17             0.5
## 4     20             0.5
## 5     33             0.5
```

```
normalize <- function (target) {
  (target - min(target))/(max(target) - min(target))
}
train$shot_distance <- normalize(train$shot_distance)
test$shot_distance <- normalize(test$shot_distance)
train$time_remaining <- normalize(train$time_remaining)
test$time_remaining <- normalize(test$time_remaining)
```

Solo normalizamos los atributos “*time_remaining*” y “*shot_distance*” porque queremos investigar la relación entre los aciertos y los fallos con la distancia de los lanzamientos, durante el tiempo. Como lo sugiere el gráfico siguiente: cuando la distancia se hace mayor, la frecuencia de lanzamiento disminuye.



Todas las operaciones de transformación realizadas se han aplicado sobre el conjunto de datos de entrenamiento y el conjunto de datos de test.

Limpieza de datos

Durante la limpieza de datos vamos a eliminar los valores fuera de rango que hemos encontrado durante la exploración del conjunto de datos y eliminaremos los atributos que hemos creído independientes al modelado.

Comenzamos eliminando los valores fuera de rango, que como hemos visto anteriormente una buena opción es descartar los lanzamientos por distancia superiores a 40 (ft.). El proceso se muestra a continuación.

```
train$shot_distance[train$shot_distance > 40] <- 40
test$shot_distance[test$shot_distance > 40] <- 40
```

Y creemos que las siguientes columnas pueden descartarse, con independencia de la variable clase.

- **game_event_id.** Independiente al modelado.
- **game_id.** Independiente al modelado.
- **loc_x.** Correlacionada con lat.
- **loc_y.** Correlacionada con lon.
- **lat.** Correlacionada con loc_x.
- **lon.** Correlacionada con loc_y.
- **shot_zone_area.** Independiente al modelado.

- **shot_zone_basic**. Independiente al modelado.
- **shot_zone_range**. Independiente al modelado.
- **team_id**. Siempre es el mismo número.
- **team_name**. Siempre es el mismo valor: *LA Lakers*.
- **game_date**. Independiente al modelado.
- **matchup**. Los atributos *oponent* y *matchup* contienen básicamente la misma información. Nos quedamos solo con el atributo *oponent*.
- **minutes_remaining**. Hemos combinado sus valores en una nueva columna (*“time_remaining”*) que contiene la misma información.
- **seconds_remaining**. Hemos combinado sus valores en una nueva columna (*“time_remaining”*) que contiene la misma información.

#Después de la limpieza los conjuntos de datos de entrenamiento y de test quedaron:

Todas las operaciones de limpieza realizadas se han aplicado sobre el conjunto de datos de entrenamiento y el conjunto de datos de test.

Modelado

Una vez se ha realizado la exploración, la transformación y la limpieza de los datos, pasamos a la fase del modelado. Crearemos dos nuevos conjuntos de datos sobre los conjuntos de datos de entrenamiento y el conjunto de datos de test con los atributos (*“time_remaining”* y *“shot_distance”*) y la variable de clase a predecir (*“shot_made_flag”*), que usaremos para hacer la predicción.

Los nuevos conjuntos de datos de entrenamiento y de test son:

```
## 'data.frame': 25697 obs. of 3 variables:
## $ shot_distance : num 0.19 0.203 0.278 0 0.177 ...
## $ time_remaining: num 0.871 0.651 0.577 0.531 0.801 ...
## $ shot_made_flag: Factor w/ 2 levels "0","1": 1 2 1 2 1 2 2 1 1 2 ...

## shot_distance time_remaining shot_made_flag
## 1 0.1898734 0.8711485 0
## 2 0.2025316 0.6512605 1
## 3 0.2784810 0.5770308 0
## 4 0.0000000 0.5308123 1
## 5 0.1772152 0.8011204 0

## 'data.frame': 5000 obs. of 3 variables:
## $ shot_distance : num 0.2951 0.0328 0 0 0.2787 ...
## $ time_remaining: num 0.8831 0.6831 0.00141 0.90986 0.9662 ...
## $ shot_made_flag: int NA NA NA NA NA NA NA NA NA NA ...

## shot_distance time_remaining shot_made_flag
## 1 0.29508197 0.883098592 NA
## 2 0.03278689 0.683098592 NA
## 3 0.00000000 0.001408451 NA
## 4 0.00000000 0.909859155 NA
## 5 0.27868852 0.966197183 NA
```

Se ha escogido un modelo de regresión logística ya que el resultado de la predicción será numérico y porque la variable a predecir es binaria. La regresión logística generalizada, es un método de regresión que permite estimar la probabilidad de una variable cualitativa binaria, como la variable a predecir *“shot_made_flag”*, en función de alguna variable cuantitativa. Es importante tener en cuenta que, aunque la regresión logística permite clasificar, se trata de un modelo de regresión. Que además nos permite calcular la probabilidad de las variables dependientes para a cada una de las dos categorías en función del valor que adquiera la variable independiente.

Ejecutamos el modelado y obtenemos su resultado.

```
model <- glm(shot_made_flag~., data=train_dat, family = binomial(link = "logit"))
summary(model)
```

```
##
## Call:
## glm(formula = shot_made_flag ~ ., family = binomial(link = "logit"),
##      data = train_dat)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.3683  -1.0523  -0.8707   1.2016   1.8815
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    0.30442    0.03055   9.966 < 2e-16 ***
## shot_distance  -3.46800    0.11131 -31.157 < 2e-16 ***
## time_remaining  0.13464    0.04404   3.057  0.00223 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 35325  on 25696  degrees of freedom
## Residual deviance: 34286  on 25694  degrees of freedom
## AIC: 34292
##
## Number of Fisher Scoring iterations: 4
```

En la tabla coeficientes, *shot_distance* es un atributo altamente significativo, con un p-valor bajísimo, lo que demuestra que será un atributo muy útil para predecir la variable (*shot_made_flag*).


Resultado

Finalmente, realizamos la predicción sobre el conjunto de datos de test y guardamos el resultado en un fichero *.CSV" para la presentación del resultado en Kaggle. A continuación, vemos los primeros valores del resultado obtenido para tener una idea de como es el fichero que se ha subido.

```
submission <- read.csv("glm.csv")
head(submission, 10)
```

```
##      shot_id shot_made_flag
## 1          1      0.5253336
## 2          8      0.9240430
## 3         17      0.9199584
## 4         20      0.9928629
## 5         33      0.5574973
## 6         34      0.4785334
## 7         35      0.9459042
## 8         36      0.9351108
## 9         37      0.9481853
## 10        38      0.5428182
```

Resultado en Kaggle



Kobe Bryant Shot Selection

Which shots did Kobe sink?
1,117 teams · 4 years ago

[Overview](#) [Data](#) [Notebooks](#) [Discussion](#) [Leaderboard](#) [Rules](#) [Team](#) [My Submissions](#) [Late Submission](#)

Your most recent submission

Name	Submitted	Wait time	Execution time	Score
glm.csv	a few seconds to go	0 seconds	0 seconds	0.86488

Complete

[Jump to your position on the leaderboard](#)

Make a submission for [CDAA_LauraRodriguez](#)

La implementación que se ha utilizado se encuentra en el repositorio de *GitHub*: <https://github.com/lrodrin/masterAI/tree/master/A3/Proyecto%20Kaggle>

Conclusiones

El resultado en Kaggle parece que es bueno. Realizar una técnica de regresión logística ha dado buenos resultados, aunque hay resultados mucho mejores en Kaggle con la aplicación de otros modelos. Un dato curioso es que muchos de los usuarios de Kaggle utilizaron el modelo XGBoost. Me pareció más entendedor y interesante utilizar un modelo de regresión logística.

Además de los buenos resultados en Kaggle y del análisis del conjunto de datos que me ayudó con la elección del modelo, la decisión de escoger el modelo de regresión logística también se basó en el hecho de que aún no había trabajado con una técnica de regresión durante la asignatura, así que me dio la posibilidad de investigar y aprender. Sobre todo, al hacer la evaluación del modelo. Es un modelo muy difícil evaluar. El trabajo se podría mejorar en este punto.

Otra de las cosas que consideré complicada y que me costó mucho fue la selección de variables para la predicción. Había muchas opciones, al final seleccioné la que me pareció más simple y que reducía más el conjunto de datos con la selección de variables. Ya que como hemos visto en la asignatura la selección de variables es un proceso que también es crucial para una buena predicción.

Que el trabajo se haya desarrollado en Kaggle le ha dado una motivación extra.

Citas para fuentes usadas

- Notebook en Kaggle de xvivancos (<https://www.kaggle.com/xvivancos/kobe-bryant-shot-selection/>).
- Notebook en Kaggle de khozzy (<https://www.kaggle.com/khozzy/kobe-shots-show-me-your-best-model/>).
- Notebook en Kaggle de dixhom (<https://www.kaggle.com/dixhom/data-analysis-for-beginners/>).