

# 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 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, pero no en el conjunto de datos de entrenamiento.

Una vez descargados los datos de *Kaggle*, es necesario dividir el conjunto de datos en un conjunto de datos de entrenamiento y un conjunto de datos de evaluación (test). Para ello, y como hemos comentado anteriormente, el conjunto de datos de entrenamiento no tendrá los valores faltantes del atributo “*shot\_made\_flag*”, pero sí que los tendrá el conjunto de datos de test, que son los valores que tendremos que predecir.

```
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))
```

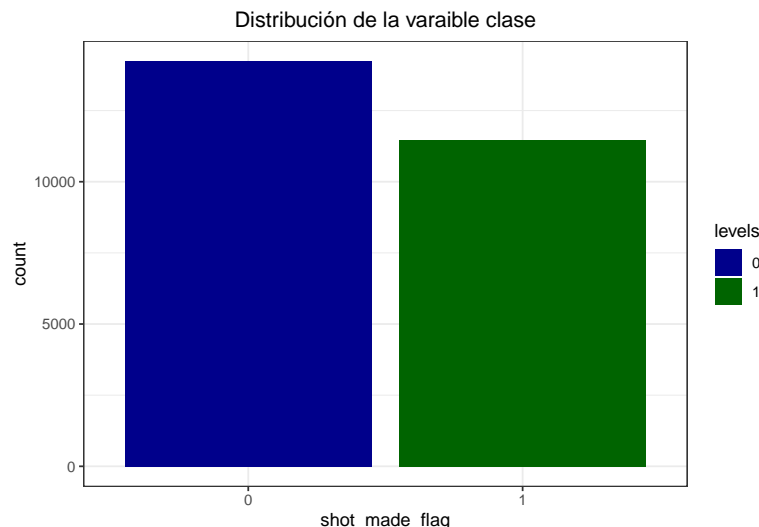
```
## [1] TRUE
```

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

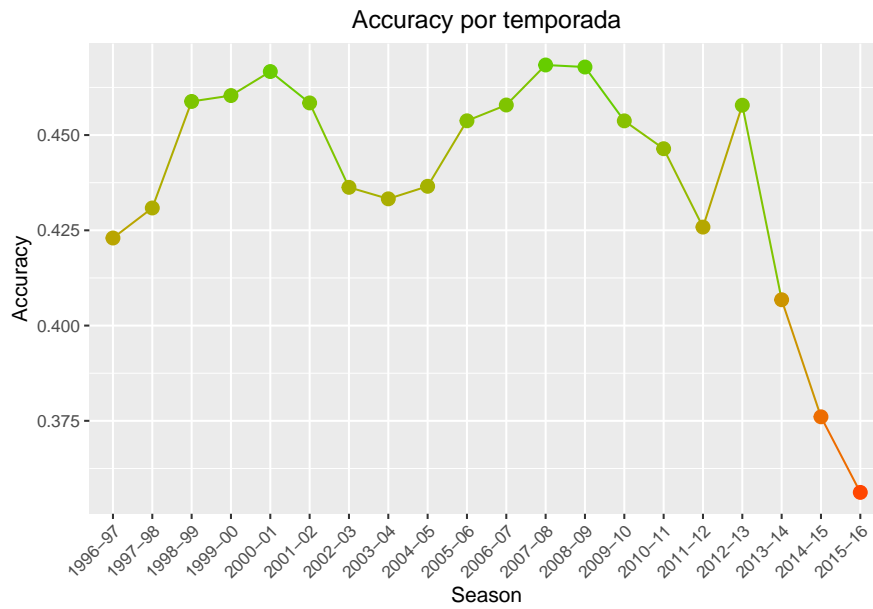
Una vez dividido el conjunto de los datos, es necesario realizar un análisis de los datos, así como un proceso de exploración, transformación y limpieza de estos datos, con el objetivo de resaltar información útil para la fase de modelado. Este análisis nos permitirá controlar la presencia de valores 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 su realización.

## Exploración, Limpieza y Transformación

Primero, empezamos con la exploración de algunos datos, analizando visualmente la variable de clase a predecir (atributo “*shot\_made\_flag*”), que observamos que es binaria y que se distribuye de manera bastante equitativa. Con un poco más de profundidad vemos que el número de canastas falladas es superior al número de canastas con aciertos. Así que, intentaremos averiguar si esto puede estar relacionado con la gran lesión que tuvo Kobe Bryant durante la temporada 2013-14.

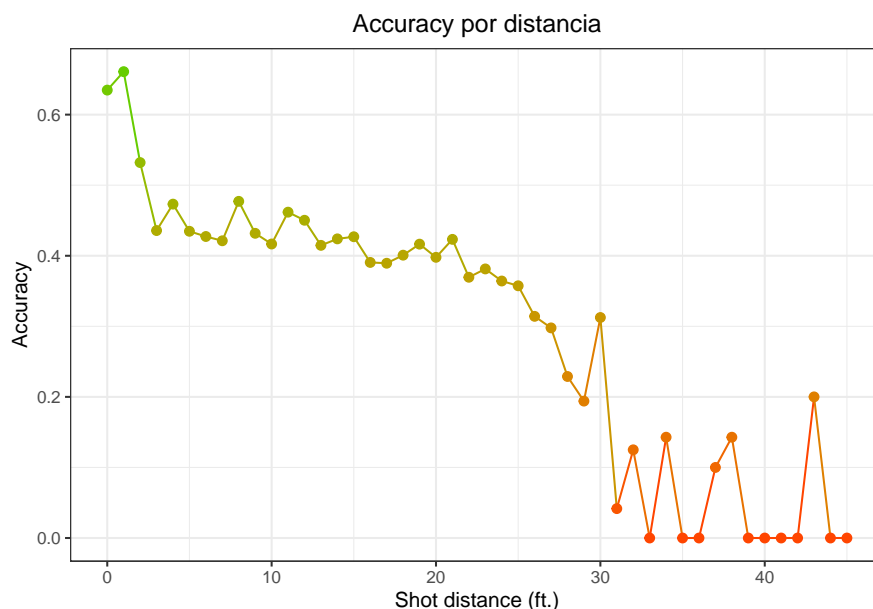


Después, analizamos visualmente la precisión de los lanzamientos realizados por temporada (atributo “*season*”) y vemos que a partir de la temporada 2013-14 la precisión de los lanzamientos empieza a bajar drásticamente. ¿Así que, una gran cantidad de los lanzamientos fallados podrían estar relacionados con la gran lesión que tuvo Kobe Bryant a principios de la temporada 2013-14 y que le afectó en su rendimiento durante las siguientes temporadas? Parece que sí.

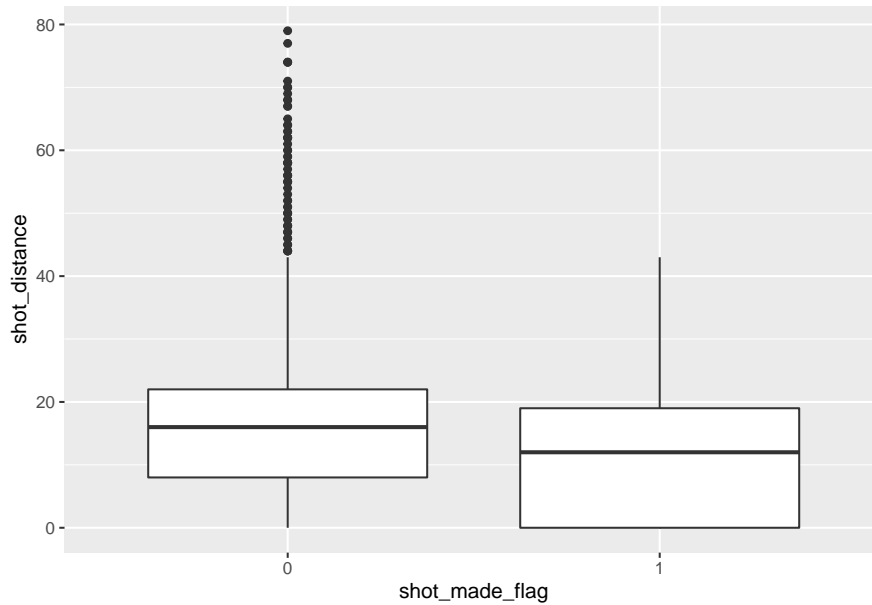


A continuación, en el siguiente gráfico, analizamos visualmente la precisión de los lanzamientos respecto a la distancia del tiro de estos (atributo “*shot\_distance*”), porque nos fijamos que en diferentes exploraciones de datos que han realizado otros usuarios de *Kaggle*, se puede observar que el atributo “*shot\_distance*” parece ser una buena variable con la que realizar la predicción juntamente con la variable de clase “*shot\_made\_flag*”, y que además contiene valores fuera de rango que podríamos eliminar, cosa que sería muy beneficioso para reducir los fallos durante la predicción.

Concretamente, si nos fijamos en el gráfico los valores fuera de rango podrían encontrarse a partir de los lanzamientos realizados a más de 30 (ft.) ya que la precisión baja drásticamente a partir de ese punto. En el siguiente gráfico se analiza con más profundidad.



En el siguiente grafico comprobamos la existencia de esos valores fuera de rango que estábamos buscando. Los lanzamientos fallados que se realizan a una distancia superior a 40 (ft.) se considerarán valores fuera de rango y serán eliminados más adelante.



Hasta aquí, hemos realizado una pequeña exploración de los datos donde hemos analizado visualmente la precisión de los lanzamientos por temporada, los lanzamientos respecto a la distancia de estos y hemos encontrado valores fuera de rango en el conjunto de datos, que tendremos que eliminar si queremos realizar una buena predicción con la variable clase “*shot\_made\_flag*”.

Al trabajar con un conjunto de datos real y con muchos atributos, 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 sobre el conjunto de datos para obtener un buen ajuste del modelo y una mejor capacidad predictiva.

## Limpieza

Durante el proceso de limpieza sobre el conjunto de datos, primero eliminaremos los atributos que consideramos independientes al modelado y después eliminaremos los valores fuera de rango detectados durante el proceso de exploración.

Los atributos que pueden descartarse del conjunto de datos son:

- **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. Solo nos quedaremos con el atributo *oponente*.

Los valores fuera de rango que pueden descartarse, que ya hemos visto que es una buena opción, son los lanzamientos realizados a una distancia superior 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
```

El conjunto de datos de entrenamiento y de test después de la limpieza de datos:

```
## 'data.frame': 25697 obs. of 12 variables:
## $ action_type : Factor w/ 57 levels "Alley Oop Dunk Shot",...: 27 27 27 6 27 ..
## $ combined_shot_type: Factor w/ 6 levels "Bank Shot","Dunk",...: 4 4 4 2 4 5 4 4 4 ..
## $ minutes_remaining : int 10 7 6 6 9 8 6 3 1 11 ...
## $ period : int 1 1 1 2 3 3 3 3 1 ...
## $ 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 22 45 52 19 32 52 12 36 56 0 ...
## $ shot_distance : num 15 16 22 0 14 0 12 12 25 17 ...
## $ shot_made_flag : int 0 1 0 1 0 1 1 0 0 1 ...
## $ shot_type : Factor w/ 2 levels "2PT Field Goal",...: 1 1 1 1 1 1 1 1 2 1 ..
## $ opponent : Factor w/ 33 levels "ATL","BKN","BOS",...: 26 26 26 26 26 26 ..
## $ shot_id : int 2 3 4 5 6 7 9 10 11 12 ...

## 'data.frame': 5000 obs. of 12 variables:
## $ action_type : Factor w/ 57 levels "Alley Oop Dunk Shot",...: 27 27 13 13 27..
## $ combined_shot_type: Factor w/ 6 levels "Bank Shot","Dunk",...: 4 4 5 5 4 4 5 5 ..
## $ minutes_remaining : int 10 8 0 10 11 10 7 5 4 5 ...
## $ period : int 1 3 1 3 1 1 1 1 2 ...
## $ 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 5 1 46 26 58 33 58 9 33 ...
## $ shot_distance : num 18 2 0 0 17 20 1 1 0 16 ...
## $ shot_made_flag : int NA NA NA NA NA NA NA NA NA NA ...
## $ shot_type : Factor w/ 2 levels "2PT Field Goal",...: 1 1 1 1 1 1 1 1 1 1 ..
## $ opponent : Factor w/ 33 levels "ATL","BKN","BOS",...: 26 26 31 31 32 32 ..
## $ shot_id : int 1 8 17 20 33 34 35 36 37 38 ...
```

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

## Transformación

La primera de las transformaciones 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 ...
```

Ahora vamos a crear un nuevo atributo (“*time\_remaining*”) a partir de los atributos “*minutes\_remaining*” y “*seconds\_remaining*”. Porque si nos fijamos, la información que contienen estos dos atributos, la podríamos combinar. Los minutos del atributo “*minutes\_remaining*” los podríamos pasar a segundos y sumar-los a los segundos del atributo “*seconds\_remaining*”. Y la nueva información combinada la guardamos como nuevo atributo “*time\_remaining*”.

Este proceso se muestra a continuación y al final los atributos “*minutes\_remaining*” y “*seconds\_remaining*” se eliminarán ya que contendrán información repetida en “*time\_remaining*”.

```
train$time_remaining <- train$minutes_remaining * 60 + train$seconds_remaining
test$time_remaining <- test$minutes_remaining * 60 + test$seconds_remaining
train <- subset(train, select = -c(minutes_remaining, seconds_remaining))
test <- subset(test, select = -c(minutes_remaining, seconds_remaining))
```

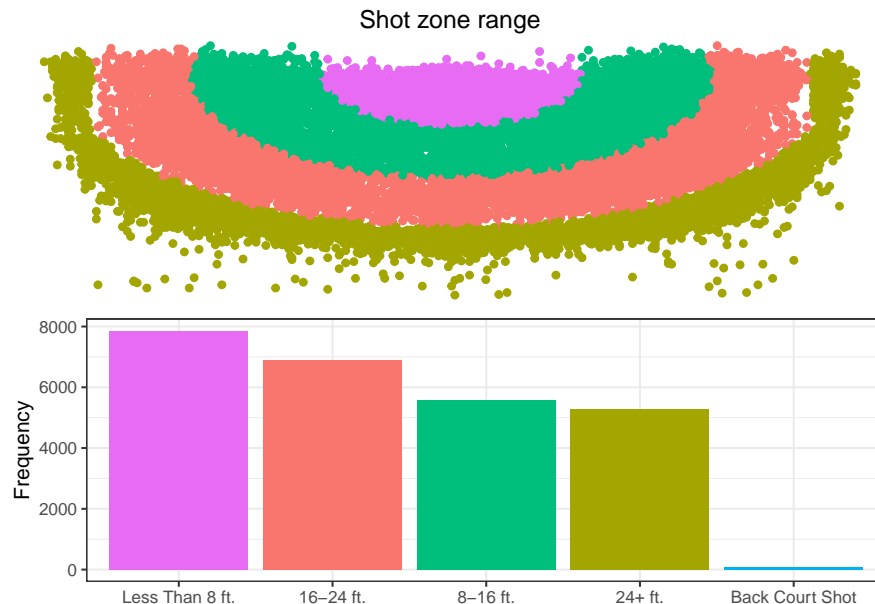
La siguiente transformación es normalizar los atributos “*time\_remaining*” y “*shot\_distance*” que se utilizarán para la predicción. También se hará al final con los valores de la variable clase “*shot\_made\_flag*”, ya que vamos a predecir probabilidades. Sabemos que tenemos que predecir probabilidades por el fichero de ejemplo que nos facilita *Kaggle*, que mostramos a continuación.

```
##   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
```

Se procede con la normalización de los atributos “*time\_remaining*” y “*shot\_distance*”.

```
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 se ha decidido investigar para la predicción la relación entre los lanzamientos acertados o no con la distancia de estos que se realizaron durante más tiempo. Para tomar la decisión nos ayudamos del gráfico siguiente que nos sugiere que cuando la distancia del lanzamiento es mayor, la frecuencia de lanzamientos disminuye.



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

## Modelado y Evaluación

Una vez se ha realizado la exploración, la transformación y la limpieza sobre el conjunto de datos, pasamos a la fase del modelado y evaluación del modelo. Para ello, primero crearemos dos nuevos conjuntos de datos sobre los conjuntos de datos de entrenamiento y de test, solo con los atributos (“*time\_remaining*” y “*shot\_distance*”) y la variable de clase a predecir (“*shot\_made\_flag*”).

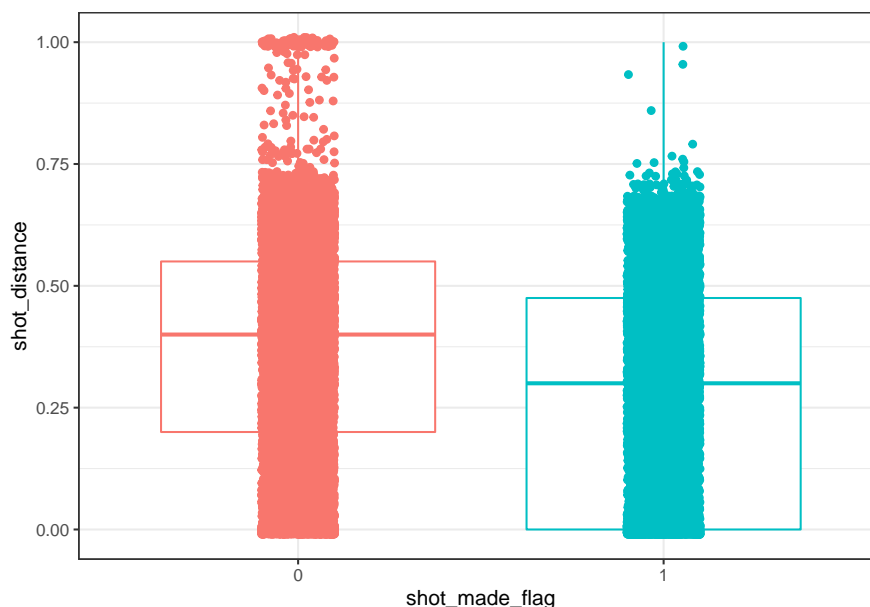
Que como se ha comentado anteriormente, solo seleccionamos estos atributos porque queremos investigar la relación entre los lanzamientos acertados o no con la distancia de estos que se realizaron durante más tiempo. Los nuevos conjuntos de datos de entrenamiento y de test después de hacer la selección de variables son:

```
##      shot_distance time_remaining shot_made_flag
## 16424      0.550      1.0000000            0
## 20392      0.550      1.0000000            1
## 20411      0.000      1.0000000            1
## 2963       0.200      0.9985994            0
## 4636       0.525      0.9985994            0

##      shot_distance time_remaining shot_made_flag
## 539          0.000      1.0000000           NA
## 18           0.175      0.9985915           NA
## 2601         0.425      0.9985915           NA
## 4065         0.000      0.9985915           NA
## 4979         0.000      0.9985915           NA
```

Para comprobar que ha sido una buena decisión, se han realizado todas las observaciones posibles de todos los atributos del nuevo conjunto de datos de entrenamiento con la variable clase “*shot\_made\_flag*” para encontrar diferencias entre las observaciones. Y hemos observado que parecen existir diferencias en las observaciones entre “*shot\_distance*” y “*shot\_made\_flag*”. Así que, la variable “*shot\_distance*” seguramente será un buen predictor.

En el siguiente gráfico observamos las diferencias encontradas entre las observaciones de “*shot\_distance*” y “*shot\_made\_flag*”.



Por lo que respecta al modelo utilizado, el modelo que se ha escogido es un modelo de regresión logística (binominal) ya que la respuesta esperada en la predicción debe ser numérica, y además sabemos que la variable a predecir es categórica de 2 niveles y binaria.

La regresión logística (binominal), 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 nos permite clasificar, se trata de un modelo de regresión. Que además puede calcular la probabilidad de la variable dependiente para a cada una de las dos categorías en función del valor que adquieran la variables independientes.

Generamos el modelo de regresión logística (binominal).

```
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
## -1.3696  -1.0519  -0.8694   1.2015   1.8258
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    0.30269    0.03058   9.899 < 2e-16 ***
## shot_distance -1.76008    0.05657 -31.111 < 2e-16 ***
## time_remaining  0.13940    0.04401   3.167  0.00154 **
## ---
## 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: 34298  on 25694  degrees of freedom
## AIC: 34304
##
## Number of Fisher Scoring iterations: 4
```

A la hora de evaluar la validez y la calidad del modelo se usan las métricas *Accuracy* y *LogLoss* (score utilizado en *Kaggle* como evaluación). Aunque existe una evaluación propia para este tipo de modelos, que no se ha desarrollado en este trabajo ya que nos interesa la métrica *LogLoss* porque el modelo se evaluará en *Kaggle*. Además, para una buena clasificación y comparación de los diferentes modelos que se han probado antes de escoger el modelo final, la métrica *Accuracy*\* es adecuada para este caso.

Para calcular los valores de *Accuracy* se va a utilizar un intervalo de 0.5. Es decir, si la probabilidad de que la variable “*shot\_made\_flag*” adquiera el valor 1 (acierto) es superior a 0.5, se le asigna a este nivel, si es menor a 0.5 se le asigna al nivel 0 (no acierto).

```
newdata <- data.frame(train_dat[,-3])
pred <- predict(model, newdata, type = 'response')

newdf <- data.frame(shot_id=train$shot_id, shot_made_flag=pred)
normalize <- function(target) {
  (target - min(target))/(max(target) - min(target))
}
newdf$shot_made_flag <- normalize(newdf$shot_made_flag)
preds_th <- ifelse(as.numeric(pred) > 0.5,1,0)

# confusion matrix
cm <- table(newdf$shot_made_flag, preds_th)
accuracy <- (cm[1,1] + cm[2,2]) / (cm[1,1] + cm[2,2] + cm[1,2] + cm[2,1])
accuracy

## [1] 0.9294118
```



```
loglogg <- logLoss(train_dat$shot_made_flag, pred)
loglogg
```

```
## [1] 0.8921015
```

Con este modelo obtenemos un Accuracy de 0.9294118 y un valor de *LogLoss* de 0.8921015, lo cual parece muy bueno para realizar la predicción con él.

## Resultado

Una vez generado y evaluado el modelo se realiza la predicción para estimar los datos restantes no empleados como datos de entrenamiento, y se normaliza la variable a predecir.


```
newdata <- data.frame(test_dat[, -3])
pred <- predict(model, newdata, type = "response")
submission <- data.frame(shot_id=test$shot_id, shot_made_flag=pred)
normalize <- function(target) {
  (target - min(target))/(max(target) - min(target))
}
submission$shot_made_flag <- normalize(submission$shot_made_flag)
```

Finalmente se guarda el resultado de la predicción en un fichero *.CSV* para la presentación en *Kaggle*. Vemos los primeros valores de la predicción obtenida para tener una idea de como es el fichero que se ha subido.

```
write.csv(submission, "glm.csv", row.names = FALSE)
submission <- read.csv("glm.csv")
head(submission, 5)
```

```
##   shot_id shot_made_flag
## 1      1      1.0000000
## 2      8      0.8205792
## 3     17      0.5601230
## 4     20      0.9998886
## 5     33      0.9998886
```

## 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](#)

## Conclusiones

El modelo creado para predecir la probabilidad de los lanzamientos acertados que nos planteaba este trabajo ha dado un valor de *LogLoss* (score) de 0.86488 en *Kaggle*. Este valor de *LogLoss* es un poco inferior al valor de *LogLoss* calculado sobre el conjunto de datos de entrenamiento. Así que la predicción ha ido mejor de lo esperado.

Realizar una técnica de regresión logística (binominal) ha dado un buen resultado, aunque hay resultados mucho mejores en *Kaggle*, con la aplicación de otros modelos. Un dato curioso es que muchos de los usuarios de *Kaggle* utilizan el modelo *XGBoost*. En mi opinión el modelo de regresión logística (binominal) me pareció más interesante, ya que, a parte de dar muy buenos resultados, pocos usuarios de *Kaggle* lo han utilizado para realizar sus predicciones, y teóricamente es un buen candidato para este problema en concreto por sus características.

Además de los buenos resultados que da el modelo de regresión logística (binominal) en *Kaggle*, la decisión de escoger este tipo de modelo también se basó en el hecho de que personalmente aún no había trabajado con una técnica de regresión logística durante la asignatura, así que me pareció muy interesante la posibilidad que me brindaba para investigar y aprender un poco más allá de los conocimientos adquiridos durante la asignatura y complementarlos.

Del resultado de la investigación del modelo de regresión logística he podido observar la gran dificultad de la evaluación de este tipo de modelos. Se evalúan de diferente manera a lo que se ha practicado durante la asignatura (a partir de desviaciones, coeficientes, diferentes métricas como *p-value*, etc.), un proceso de evaluación mucho más complicado, que en este trabajo no ha sido necesario desarrollar.

Finalmente he de comentar que durante este trabajo también ha sido muy importante el proceso de normalización estadística. Sin este proceso las distribuciones de probabilidad finales no se pueden conseguir.

## 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/>.
- Mi repositorio personal en *GitHub* (lrodrin) <https://github.com/lrodrin/masterAI/tree/master/A3/Proyecto%20Kaggle>