

Kobe Bryant Shot Selection

Laura Rodríguez Navas

Introducción

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

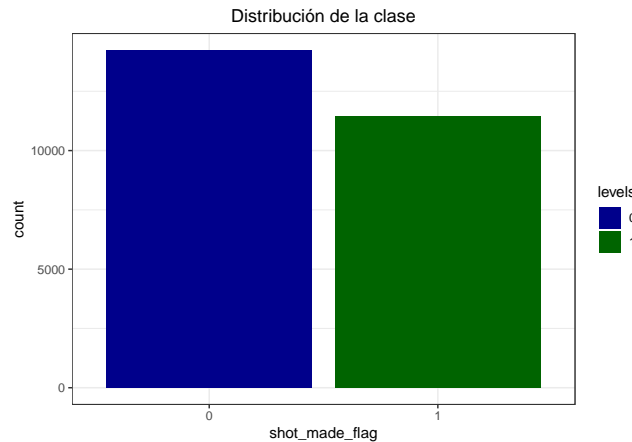
El conjunto de datos contiene 30697 (25697 + 5000) instancias y un gran número de variables explicativas (11 discretas y 14 numéricas). Estas 25 variables (incluyendo clase a predecir “*shot_made_flag*”) se centran en la descripción cualitativa y cuantitativa de multitud de aspectos de cada uno de los 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 ...
```

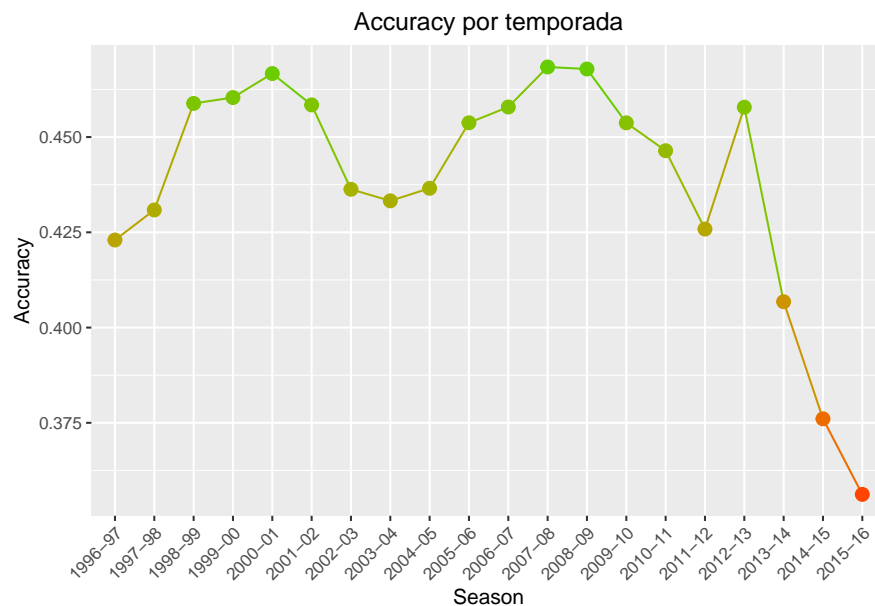
La tarea del proyecto es predecir si los lanzamientos a canastas de Kobe Bryant entraron o no en el aro (atributo “*shot_made_flag*”), es decir, los lanzamientos acertados. Del conjunto de datos se han eliminado 5000 valores de este atributo (representados como valores faltantes en el conjunto de datos).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

Exploración de datos

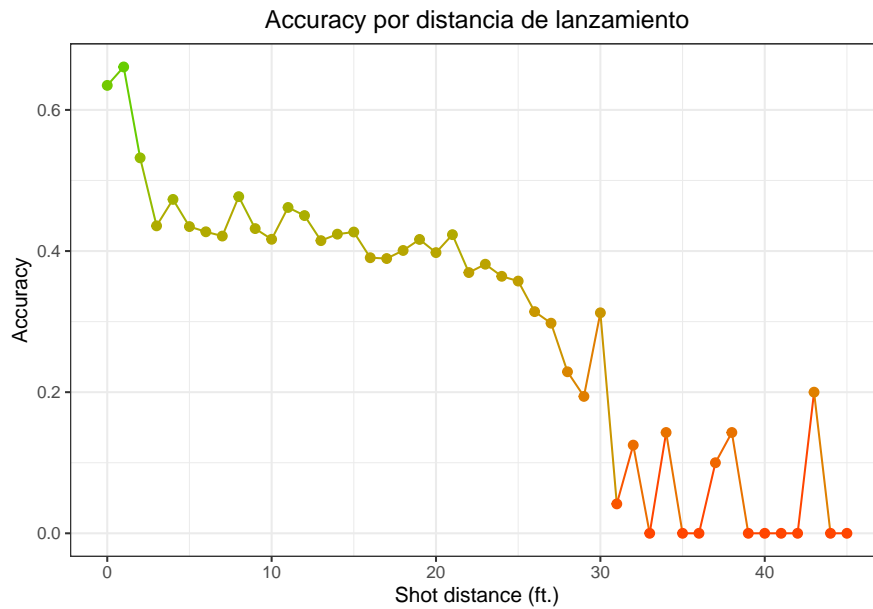
Empezamos analizando visualmente la variable de clase a predecir (atributo “*shot_made_flag*”). Podemos ver que la variable de clase se distribuye de manera bastante equitativa. No se realizará ninguna acción para tratar con el conjunto de datos desequilibrado.



A continuación, analizamos visualmente la precisión de los lanzamientos por temporada (atributo “*season*”). Observamos que a partir de la temporada 2013-2014 la precisión de los lanzamientos baja drásticamente. Así que, exploraremos las correlaciones de este período de tiempo con los atributos del conjunto de datos y la variable de clase a predecir.



En el siguiente gráfico vemos visualmente que la precisión de los lanzamientos por distancia (atributo “*shot_distance*”) se correlaciona con la precisión de los lanzamientos por temporada. Es más, los lanzamientos realizados a más de 30 (ft.) los podríamos considerar como valores inusuales. En el apartado de selección de variables concretaremos que valores son con más exactitud y en el apartado de limpieza de datos serán eliminados.



Selección de variables

Dividiremos el conjunto de datos en un conjunto de datos de entrenamiento y un conjunto de datos de evaluación (test). Para ello primero tenemos que analizar si existen valores faltantes y observamos que sí que existen valores faltantes dentro de la variable de clase a predecir (*“shot_made_flag”*). Concretamente, esos valores son los que tendremos que predecir en la evaluación.

En este proyecto consideramos que para dividir el conjunto de datos en un conjunto de datos de entrenamiento y un conjunto de datos de test para realizar una mejor predicción, tenemos que crear un conjunto de datos de entrenamiento sin valores faltantes y un conjunto de datos de test con los valores faltantes que tenemos que queremos predecir.

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

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

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

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

Anteriormente, hemos analizado visualmente la precisión de los lanzamientos por temporada con los lanzamientos por distancia y hemos encontrado valores inusuales en el conjunto de datos que tendremos que eliminar si queremos realizar una buena predicción de la variable de clase *“shot_made_flag”*. Ahora, veremos que valores inusuales se encuentran primero en el conjunto de datos de entrenamiento y después en el conjunto de datos de test.

```
## [1] 47 48 62 70 60 56 55 51 68 47 64 55 50 68 50 60 54 62 48 74 70 61 65 59 63
```

```
## [26] 49 55 46 74 58 64 51 56 56 69 47 50 49 57 58 53 58 59 48 67 62 71 52 77 63
```

```
## [51] 79 74 62 46 58 52 67
```

```
## [1] 49 49 51 59 52 61 56 54 52 53 54
```

Los valores inusuales los podemos directamente correlacionar con la falta de acierto de los lanzamientos superiores a 46 (ft.) y 49 (ft.).

```
## [1] 46
```

```
## [1] 49
```

En el apartado de limpieza de datos seran eliminados.

Transformación de datos

La primera transformación que realizamos es la binarización de la variable clase “*shot_made_flag*”.

También juntamos la columnas minutes y seconds en una tercera columna time. Y después eliminamos las columnas antiguas. minutes_remaining y seconds_remaining parecen ser un par, así que combinémoslos.

```
train$shot_made_flag <- as.factor(train$shot_made_flag)
test$shot_made_flag <- as.factor(test$shot_made_flag)

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

Normalizamos, las columns que utilizaremos para la predicción.

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

Limpieza de datos

Asumimos la independencia de cada lanzamiento, por lo tanto, las siguientes columnas pueden descartarse.

- **game_event_id.** Independiente al análisis.
- **game_id.** Independiente al análisis.
- **loc_x.**
- **loc_y.**
- **lat.** Correlacionada con loc_x.
- **lon.** Correlacionada con loc_y.
- **shot_zone_area.**
- **shot_zone_basic.**
- **shot_zone_range.**
- **team_id.** Siempre es el mismo número.
- **team_name.** Siempre es el mismo valor: *LA Lakers*.
- **game_date.**
- **matchup.** oponent y matchup son básicamente la misma información. Solo se necesita oponente.
- **minutes_remaining.**
- **seconds_remaining.**

```
## 'data.frame': 25697 obs. of 11 variables:
## $ action_type : Factor w/ 57 levels "Alley Oop Dunk Shot",...: 27 27 27 6 27 28 27 42 27 27 ..
## $ combined_shot_type: Factor w/ 6 levels "Bank Shot","Dunk",...: 4 4 4 2 4 5 4 4 4 4 ...
## $ 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 5 5 ...
## $ shot_distance : num 0.19 0.203 0.278 0 0.177 ...
## $ shot_made_flag : Factor w/ 2 levels "0","1": 1 2 1 2 1 2 2 1 1 2 ...
## $ 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 26 26 26 31 ...
```

```
## $ shot_id          : int  2 3 4 5 6 7 9 10 11 12 ...
## $ time_remaining   : num  0.871 0.651 0.577 0.531 0.801 ...

## 'data.frame':      5000 obs. of  11 variables:
## $ action_type      : Factor w/ 57 levels "Alley Oop Dunk Shot",...: 27 27 13 13 27 27 28 28 35 27 .
## $ combined_shot_type: Factor w/ 6 levels "Bank Shot","Dunk",...: 4 4 5 5 4 4 5 5 4 ...
## $ period           : int   1 3 1 3 1 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 5 ...
## $ shot_distance    : num   0.2951 0.0328 0 0 0.2787 ...
## $ shot_made_flag    : Factor w/ 0 levels: 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 ...
## $ opponent         : Factor w/ 33 levels "ATL","BKN","BOS",...: 26 26 31 31 32 32 32 32 32 ...
## $ shot_id          : int   1 8 17 20 33 34 35 36 37 38 ...
## $ time_remaining    : num   0.8831 0.6831 0.00141 0.90986 0.9662 ...
```

Los valores inusuales observados anteriormente también pueden descartarse de los conjuntos de entrenamiento y test, además obserando esos valores, una buena opción es descartar los lanzamientos por distancia superiores a 40 (ft.).

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

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

Modelado

Una vez se ha realizado el análisis sobre los datos, incluyendo la limpieza, transformación y generación de nuevas variables interesantes para el estudio, pasamos a la fase del modelado.

Solo nos quedamos con shot_distance, time_remaining y shot_made_flag. separar los datos para el entrenamiento y la sumisión

Ahora separemos los datos.

```
##   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
## 6    0.0000000      0.7450980             1

##   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>
## 6    0.32786885      0.926760563          <NA>
```

Provamos diferentes modelos vistos en la asignatura, però ninguno de ellos nos dio tan buen resultado como GLM. Una tabla de el valor de accuracy de los modelos que he probado.

```
table <- matrix(c(0.5972292, 0.6083590, 0.6099156, 0.6085924, 0.7203565, 0.5879675, 0.6076585, 0.804335),
               ncol=1,byrow=TRUE)
colnames(table) <- c("Accuracy")
rownames(table) <- c("LDA", "Naive Bayes", "Decision Tree", "Neural Network", "Nearest Neighbour",
                    "SVM (linear kernel)", "Multilayer Perceptron", "Random Forest")
```

```

table <- as.table(table)
table

##              Accuracy
## LDA              0.5972292
## Naive Bayes      0.6083590
## Decision Tree    0.6099156
## Neural Network   0.6085924
## Nearest Neighbour 0.7203565
## SVM (linear kernel) 0.5879675
## Multilayer Perceptron 0.6076585
## Random Forest    0.8043351

GLM es bla bla,

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

anova(model)

## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: shot_made_flag
##
## Terms added sequentially (first to last)
##
##
##              Df Deviance Resid. Df Resid. Dev
## NULL              25696      35325
## shot_distance    1  1030.22      25695      34295
## time_remaining   1    9.35      25694      34286

# show accuracy by train data
newdata <- data.frame(train_dat[,-3])
pred <- predict(model, newdata, type = 'response')

newdf <- data.frame(shot_id=train$shot_id, shot_made_flag=pred)
# predict generates a vector of probabilities that we threshold at 0.5
newdf$shot_made_flag <- normalize(newdf$shot_made_flag)
preds_th <- ifelse(as.numeric(pred) > 0.5,1,0)

# matriz de confusion y accuracy
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])

```

Matriz de confusión y accuracy

Evaluación

Hacer datos de envío

Shot_made_flag predicho se escribe en un archivo csv.

Construyendo un modelo final


Usemos los parámetros que acabamos de obtener para el modelo final y la predicción.

```
#model predict the test data
newdata <- data.frame(test_dat[,-3])
pred <- predict(model, newdata)
submission <- data.frame(shot_id=test$shot_id, shot_made_flag=pred)
submission$shot_made_flag <- normalize(submission$shot_made_flag)

write.csv(submission, "glm.csv", row.names = FALSE)
```

Conclusiones

Resultado de 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 [CDAALauraRodriguez](#)

Bibliografía