Predicción de la ERA (Earned Run Average) de pithcers de la MLB

Douglas Torres, carnet: 11-11027

Mauricio Salerno, carnet: 12-10627

El siguiente trabajo explica y analiza un modelo de regresión que busca predecir la efectividad anual de un pitcher, usando como variables ciertas estadísticas que se tengan de este pitcher en la temporada anterior a la del año que se quiere predecir.

Key Words: Machine Learning, regresión lineal, TensorFlow, Medium Square Error

1 INTRODUCCIÓN

La motivación para realizar este proyecto fue, además de un interés genuino por el tema, lo mucho que se está usando el análisis de datos y predicción de comportamiento de jugadores en distintos deportes, y particularmente en la MLB (Major League Baseball). Todos recuerdan la película Moneyball, donde se ataca el problema de armar un equipo como una optimización de las estadísticas del equipo usando el capital disponible.

Esta nueva visión estadística del deporte ya está bastante institucionalizada, aunque todavía es importante hacer otro tipo de consideraciones, porque no todos los puntos en los que influye un jugador pueden reflejarse en números . **¿Por qué béisbol?** Además del interés particular que tenemos por este deporte, al ser un deporte que se juega por turnos, donde cada jugada es registrada y tiene un fin bien definido, se presta mucho más para este tipo de análisis que deportes como el fútbol o el baloncesto. Además, la MLB es una liga con una historia enorme, que se remonta a finales de 1800, por lo que la cantidad de datos recopilados sobre la liga es bastante grande, que facilitaría la recolección de instancias para el entrenamiento.

El trabajo tocará los siguientes temas: recolección y elección de los datos, análisis de los datos (consideraciones personales, datos más relevantes), distintos

modelos (selección de atributos), selección y explicación de la herramienta de software, análisis de resultados, y, para terminar, una breve conclusión.

2 RECOLECCIÓN DE DATOS

Como se mencionó brevemente en la introducción, hay una gran cantidad de información recopilada a lo largo de toda la historia de la liga, por lo que se pensó que no sería mayor problema descargarla en algún formato que facilitara la lectura desde algún ambiente de desarrollo. Esto no fue del todo cierto, hubo varias y variadas dificultades a la hora conseguir los datos que se querían para el desarrollo de este trabajo.

Primero, se intentó obtener los datos directamente de la página oficial de la liga (www.mlb.com) , pero hubo un par de problemas. A pesar de tener la información más completa sobre las grandes ligas (como era de esperarse), había ciertas restricciones en cuanto a la descarga de los datos, que, para resolver, había que afiliarse (pagar).

Luego de haber fallado con el sitio oficial, se siguieron buscando opciones, y nos encontramos con una librería de python llamada mlbgame, que permitía obtener, de manera bastante sencilla e inmediata, información y estadísticas de la liga. El problema fue que mlbgames está hecho para hacer un análisis más “en vivo” de lo que está pasando en la liga.

Se puede obtener información en tiempo real de los juegos de la liga, además de poder consultar juegos archivados. Esta librería no nos convenía porque no se podía consultar estadísticas anuales por jugadores, sino lo acontecido en los diversos juegos. A pesar de esto, se intentó obtener los promedios anuales de pitchers tomando la información de los distintos juegos, y procesándolo para obtener lo deseado. El problema fue que, de nuevo, nos topamos con un límite de consultas.

Finalmente, nos encontramos un sitio que se dedicaba a la recolección de datos e información de las grandes ligas. Este sitio es una iniciativa de Sean Lahman, que pretende poner al alcance de todos las estadísticas de la liga. En ella, se encontraban muchísima información y estadísticas de jugadores, equipos, temporadas, etc. En particular, tenía la información estadística anual de cada jugado de la liga desde 1872, que era justo lo que estábamos buscando. Además de ser gratuita y bastante completa, ofrecía los datos en diversos formatos, uno de ellos siendo un script que generaba una base de datos en un servidor mysql con todos los datos de los jugadores que estábamos buscando. Esta opción

de inmediato nos pareció la mejor, por varias razones, siendo estas dos las más importantes: 1)Las facilidades que ofrecen los manejadores de bases de datos para darle diversos formatos a la salida de las consultas. 2)Generar un modelo era tan sencillo como hacer una consulta, facilitando la creación de diversos modelos para comparar los resultados y elegir el mejor.

La base de datos tenía 27 tablas distintas, dentro de las que nos interesaban más dos o tres tablas: La tabla “Master”, que guardaba información relacionada a cada jugador de la liga; La tabla “Pitching”, que guardaba las estadísticas anuales de cada pitcher de la liga; La tabla “PitchingPost”, que guardaba estadísticas anuales de cada pitcher que haya participado en la postemporada de ese año (estadísticas calculados usando tomando en cuenta sólo lo acontecido en postemporada); La tabla “Batting”, porque, como la MLB ha ido cambiando, pensamos que el promedio de bateo de la liga influía en la efectividad de todos los pitchers.

3 ANÁLISIS DE LOS DATOS, CREACIÓN DE MODELOS.

Antes de pasar al análisis de los datos, se describirá brevemente el esquema de la tabla Pitching.

playerID Player ID code: clave primaria base de datos.

yearID Año: año en que se recopilan estas estadísticas

stint Orden de aparición.

teamID Equipo: equipo en el que juega el pitcher.

lgID Liga: liga en la que juega.

W Ganados: juegos ganados.

L Perdidos: juegos perdidos.

G Jugados: juegos jugados.

GS Juegos iniciados.

CG Juegos completos.

SHO No-hitters: juegos completos en los que no permite hits..

SV Juegos Salvados.

IPOuts Outs pichados .

H Hits permitidos.

ER Carreras limpias.

HR Homeruns.

BB Boletos.

SO Ponches.

BAOpp AVG oponente.

ERA Efectividad.

IBB Boletos intencionales.

WP Wild Pitches.

Se tomaron los atributos que se consideran los más relevantes de un pitcher para formar el primer modelo, que serían: efectividad, juegos ganados y perdidos, juegos en los que participa, innings lanzados, hits permitidos, carreras limpias, ponches, boletos y promedio de bateo de los bateadores que enfrentó.

Intentando refinar este modelo, se pensó que quizá no importaba demasiado el número de juegos ganados o perdidos, sino la proporción entre jugados y ganados y perdidos. Tampoco era un valor fiel al desempeño del pitcher el número de hits permitidos, y se decide eliminar este atributo. Algo parecido se hace con ponches y boletas, donde se toma el promedio por cada nueve innings. Con estas consideraciones se crea un nuevo modelo.

Como se quiere predecir la efectividad, que es el promedio del número de carreras limpias permitidas por un pitcher cada nueve innings, y esta es una estadística que se considera independiente de la defensa y ofensiva del equipo, se pensó que usar sólo atributos que fueran independientes de la defensa y ofensiva sería lo mejor para predecir este valor, por los que se elimina la proporción de ganados y perdidos. Estos tres modelos serán los que se entrenarán y, entre ellos, se compararán los resultados.

TENSORFLOW

Para el cumplir el objetivo previamente mencionado, se decidió utilizar la librería open source enfocada en desarrollo de programas de machine learning llamada TensorFlow, desarrollada por Google como una edición más robusta, portable, fácil de usar y open source de su primer software del mismo estilo, llamado “DistBelief”. TensorFlow fue publicado como software libre desde noviembre del 2015 y desde entonces ha sido usado para distintos

modelos de machine learning, dado que presta facilidades y optimizaciones a una gran cantidad de algoritmos que van desde descenso del gradiente hasta redes neuronales y deep learning.

El cómputo realizado por TensorFlow se expresa en un grafo computacional donde se pueden ver las conexiones entre los diferentes estados pertenecientes al modeloy la interacción que tienen estos componentes en el algoritmo implementado en esta librería.
La unidad más elemental y central de ésta librería es el “Tensor”, es la forma en que se representan las operaciones sobre los datos y permite el cálculo de las mismas en una “session” de TensorFlow. Dicha representación consiste en valores dentro de arreglos de distintas dimensiones, dependiendo de los datos y las necesidades del problema. Los tensores pueden ser constantes, variables, y en el API TF.Learn se pueden crear tensores de valores nominales, valores reales, buckets (para valores continuos), combinación de atributos (se relacionan atributos específicos entre sí cuando hay relaciones distintas para clases distintas).

Una ventaja sobre el uso de tensorflow es que se puede encontrar en uso en muchos lugares, especialmente en Google dónde TensorFlow es usado para el reconocimiento de imágenes, texto, entre otros.

En el presente trabajo se usó la versión 1.0 de dicho software.

RESULTADOS Y ANÁLISIS

A continuación se presentarán los resultados de las evaluaciones de los distintos modelos previamente entrenados. Todos los modelos fueron entrenados con el 80% de los datos, y se evaluaron utilizando el 20% restante. La medida que se usa como métrica de performance para todos los modelos se usó el error cuadrático medio (MSE), y el número de iteraciones fue de 100000.

Modelo 1.

Recordando los atributos usados para el primer modelo: efectividad, juegos ganados y perdidos, juegos en los que participa, innings lanzados, hits permitidos, carreras limpias, ponches, boletos y promedio de bateo de los bateadores que enfrentó. Este modelo obtuvo un MSE de: 1.4825717210769653. Se puede tomar la raíz cuadrada del error cuadrático

medio, conocido como RMSE (root mean square error) RMSD (root mean square deviation), que es al MSE lo que es la desviación estándar a la varianza, obtenemos RMSD: 1.2176090181486687, lo que nos dice que, en promedio, se aleja en 1,2 el valor predictivo del valor real. En béisbol, una diferencia de 1 en efectividades entre dos pitchers, aunque tiene relevancia, no es demasiado significativa. Además, revisando algunos valores predictivos y comparando con los valores reales, se puede apreciar que, en general, las predicciones se acercan bastante al valor real, con predicciones ocasionales bastante alejadas del valor real, y es esto lo que aumenta el RMSD.

Se presentan a continuación algunos resultados, y la curva de corvengencia:

Jugador: rincoju01 | Año: 2005 | Prediccion: 3.234666347503662 | Verdadero: 2.45
Jugador: fosteal01 | Año: 1976 | Prediccion: 4.131134986877441 | Verdadero: 3.22
Jugador: guidrro01 | Año: 1981 | Prediccion: 4.010429859161377 | Verdadero: 2.76
Jugador: farrejo03 | Año: 1990 | Prediccion: 4.062338352203369 | Verdadero: 4.28
Jugador: valvejo01 | Año: 2010 | Prediccion: 3.604670763015747 | Verdadero: 3.0
Jugador: vogelry01 | Año: 2005 | Prediccion: 4.931672096252441 | Verdadero: 4.43
Jugador: benesan01 | Año: 1993 | Prediccion: 3.649414539337158 | Verdadero: 3.78
Jugador: lawrebr01 | Año: 1958 | Prediccion: 4.005692005157471 | Verdadero: 4.13
Jugador: assenpa01 | Año: 1989 | Prediccion: 3.493619680404663 | Verdadero: 3.59
Jugador: krausle02 | Año: 1968 | Prediccion: 3.9801666736602783 | Verdadero: 3.11
Jugador: sanfoja02 | Año: 1960 | Prediccion: 3.7891476154327393 | Verdadero: 3.82

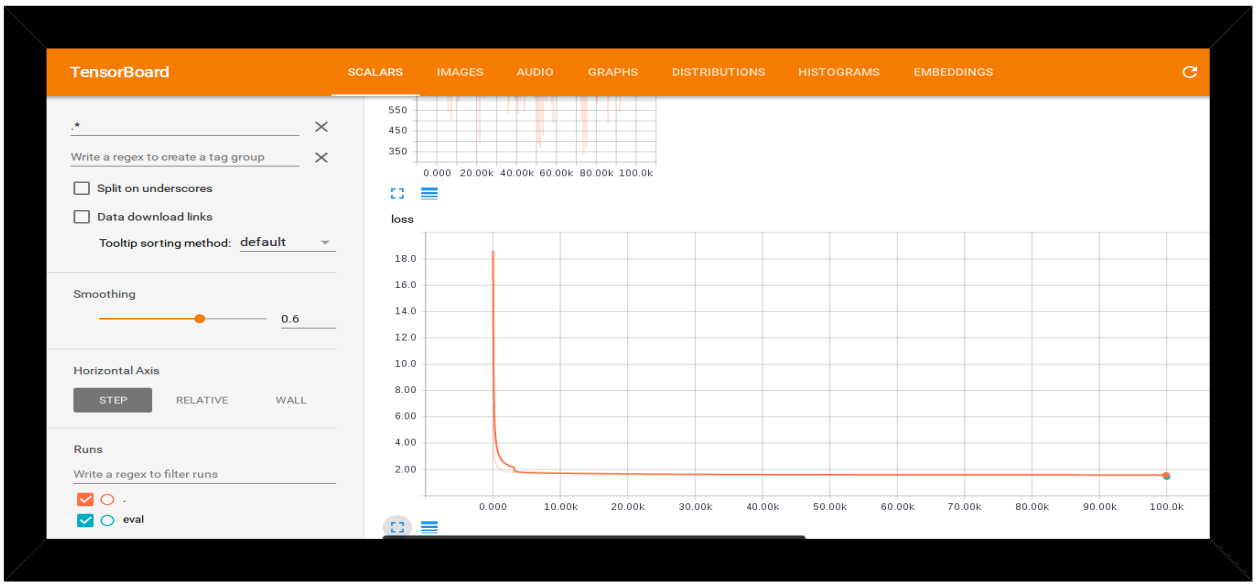

Fig. 1. **Curva de costos del modelo 1**

Modelo 2.

En este modelo, se consideró que lo relevante no era, por ejemplo, el número de hits permitido por un pitcher en un temporada, sino el promedio de hits permitidos cada nueve innings, porque un pitcher con más lanzamientos podría tener más hits y más carreras anotadas, pero menor (mejor) efectividad. También se tomó la proporción de juegos ganados y perdidos en lugar de la cantidad. Se obtuvo un MSE de 1.564926266670227, un valor bastante parecido al modelo anterior, y un RMSE de 1.25, que está apenas a 0.04 de lo obtenido en el modelo pasado, por lo que no se considera que haya una diferencia significativa entre los modelos. Algunas instancias con su valor predictivo y real, y curva de convergencia:

Jugador: 2 | Año: 3 | Prediccion: 4.376896858215332 | Verdadero: 6.99
Jugador: 3 | Año: 1 | Prediccion: 4.190258979797363 | Verdadero: 3.88
Jugador: 4 | Año: 10 | Prediccion: 3.971588134765625 | Verdadero: 3.87
Jugador: 4 | Año: 4 | Prediccion: 3.9393012523651123 | Verdadero: 5.84
Jugador: 9 | Año: 12 | Prediccion: 4.291379928588867 | Verdadero: 3.91
Jugador: 3 | Año: 3 | Prediccion: 4.375844955444336 | Verdadero: 4.78
Jugador: 9 | Año: 6 | Prediccion: 4.07598876953125 | Verdadero: 4.1
Jugador: 8 | Año: 13 | Prediccion: 4.352788925170898 | Verdadero: 3.44
Jugador: 1 | Año: 3 | Prediccion: 3.494799852371216 | Verdadero: 3.65

Fig. 2. **Curva de costos del modelo 2**

Modelo 3.

Para el tercer modelo, que intentaba usar estadísticas que no dependieran de la defensa, se tomó el modelo anterior, menos los atributos relacionados con cantidad de juegos ganados o perdidos. Se obtuvo un MSE de 4.658642292022705, y un RMSD de 2.15. Contrario a lo que se pensaba, no tomar en cuenta características que no dependían únicamente del pitcher empeoraron el modelo predictivo. Se piensa que esto se debe a la falta de otras variables importantes independientes del resto del equipo que no se tenía en el conjunto de datos que se usó para entrenar. Algunos valores predictivos y sus correspondientes valores reales y curva de convergencia:

Jugador: frierer01 | Año: 2013 | Prediccion: 2.0738842487335205 | Verdadero: 3.8
Jugador: mcgloly01 | Año: 1974 | Prediccion: 1.6951546669006348 | Verdadero: 2.69
Jugador: moralfr01 | Año: 2011 | Prediccion: 1.9868805408477783 | Verdadero: 3.62
Jugador: agostju01 | Año: 1986 | Prediccion: 1.988057017326355 | Verdadero: 8.85
Jugador: warddu01 | Año: 1992 | Prediccion: 2.9685556888580322 | Verdadero: 1.95
Jugador: waitsri01 | Año: 1976 | Prediccion: 1.9806206226348877 | Verdadero: 4.0
Jugador: garbege01 | Año: 1974 | Prediccion: 4.013171672821045 | Verdadero: 4.82

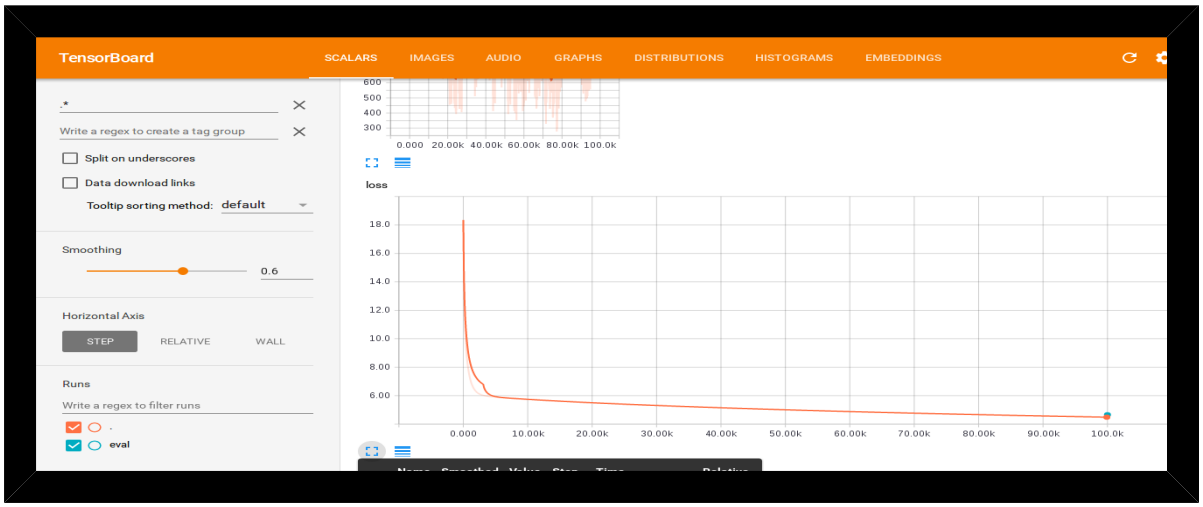

Fig. 3. **Curva de costos del modelo 3**

4 CONCLUSIONES

Se podría decir que se obtuvieron resultados favorables, se construyeron modelos que predicen con bastante precisión la efectividad (ERA) de un pitcher de grandes ligas (MLB), usando un algoritmo de regresión implementado en librería para python3 TensorFlow.

Como recomendaciones, usar alguna de las estadísticas más avanzadas que lleva la MLB sobre los pitcher, como el porcentaje de “outs” áereos o terrestres -que no se usaron en este trabajo porque las datos que se conseguían de manera gratuita no los incluían-, las que, varios estudios indican que se relacionan fuertemente con la efectividad, podría mejorar el modelo predictivo.

BIBLIOGRAFÍA

- Página web oficial de TensorFlow: https://www.tensorflow.org (Visitada por última vez el 31/03/2017)
- Tutorial en la página web oficial de TensorFlow: https://www.tensorflow.org/get_started/get_started (Visitada por última vez el 31/03/2017)
- Google Research Blog - Primer año de TensorFlow como open source: https://research.googleblog.com/2016/11/celebrating-tensorflows-first-year.html (Visitada por última vez el 31/03/2017)
- Google Research Blog - Presentación de TensorFlow: https://research.googleblog.com/2015/11/tensorflow-googles-latest-machine_9.html (Visitada por última vez el 31/03/2017)