Análisis con Weka the un dataset sobre la Premier League

Luis Domínguez López - MUSS

Índice

1 - Objetivos de la investigación

El objetivo del trabajo consistió en utilizar weka para tratar de obtener información de una dataset de resultados de la liga inglesa de fútbol. Como es normal surgieron algunos contratiempos que obligaron a que tuviese que crear un nuevo dataset, para ello y después de investigar sobre el tema decidí realizar el dataset con python y de compartirlo en Kraggle con el fin de seguir experimentando. Una vez arreglado el dataset con weka realice experimentos aplicando los distintos algoritmos y realizando limpieza de datos desde el propio programa.

2- Selección de los datos

Después Buscar distintos dataset seleccione dos datasets uno sobre Pokemon go y otro sobre la Premier League. El problema es que el de pokemon Go tenía 290000 instancias y tratando de reducir instancias se me congelaba el programa. Entonces seleccione este Premier League, en el tenemos 9664 instancias.

Contexto

El CSV contiene información desde la temporada 1993/1994 a la 2017/2018. Contiene los siguientes campos :

- Equipo local
- Equipo visitante
- Temporada
- FTHG (final time home goals) (Goles al final del partido del local)
- FTAG (final time away goals) (Goles al final del partido del visitante)
- FTR (full time result) (winner of the match (H for home team, A for away team, D for draw))
- HTHG (half time home goals) (Goles al medio del partido del local)
- HTAG (half time away goals) (Goles al medio del partido del visitante)
- HTR (half time result) (Resultado en la primera mitad)
- Div : División donde se ha jugado el partido (Borrado puesto que no nos aporta nada solo hay una división)

También adicionalmente tienen un csv en el que centralizan los resultados por club, con más información sobre tarjetas amarillas o penaltis concedidos. Sería interesante después de analizar estos datos poder analizar este segundo csv para comprobar si hay relación con los resultados y buscar nuevos resultados con más campos a tener en cuenta.

Después de que en un primer momento trabajase con estos datos decidimos que los resultados que se explican más adelante eran demasiado cortos y no aportan demasiada información. Por tanto en busqueda de un dataset que tuviese mas información me encontré con una cantidad de blogs sobre el tema, en el que todos buscaban con escaso éxito un dataset que tuviera más información sobre cada partido. Entonces encontré el trabajo de english-premier-league-match-data en el explica que ha conseguido extraer los datos de la página premierleague, el problema que encontramos aquí es que sus datos están separados en distintos ficheros , en json y con una estructura peculiar, probablemente debido a como ha extraído los datos.

Teniendo esto en cuenta me dispuse a realizar un programa en Python que me permitiera unificar los datos de forma que pudiese tratarlos con Weka este programa se puede encontrar en el repositorio domlopluis94/DataMining_MUSS/DataminingPL , en el básicamente realizó la lectura de todos los ficheros y organizó los datos de forma que obtenemos como resultado un CSV y JSON con toda la información agrupada.

Aplicando esta técnica obtenemos un CSV con los siguientes campos :

- FIELD1 : id usado para partido
- season
- home_team_id
- away_team_id
- home_team_name
- away_team_name
- date_string
- half_time_score
- full_time_score
- full_time_result : H wins local team, A wins away team, D for draw
- home_att_goal_low_left
- home_won_contest
- home_possession_percentage
- home_total_throws
- home_att_miss_high_left
- home_blocked_scoring_att
- home_total_scoring_att
- home_att_sv_low_left

- home_total_tackle
- home_att_miss_high_right
- home_aerial_won
- home_att_miss_right
- home_att_sv_low_centre
- home_aerial_lost
- home_accurate_pass
- home_total_pass
- home_won_corners
- home_shot_off_target
- home_ontarget_scoring_att
- home_goals
- home_att_miss_left
- home_fk_foul_lost
- home_att_sv_low_right
- home_att_goal_low_centre
- away_att_goal_low_left
- away_won_contest
- away_possession_percentage
- away_total_throws
- away_att_miss_high_left
- away_blocked_scoring_att
- away_total_scoring_att
- away_att_sv_low_left
- away_total_tackle
- away_att_miss_high_right
- away_aerial_won
- away_att_miss_right
- away_att_sv_low_centre
- away_aerial_lost
- away_accurate_pass
- away_total_pass
- away_won_corners
- away_shot_off_target
- away_ontarget_scoring_att
- away_goals
- away_att_miss_left
- away_fk_foul_lost
- away_att_sv_low_right
- Away_att_goal_low_centre.

Ademas despues de probarlo en Weka y comprobar su correcto funcionamiento , lo subi a Weka para su libre disposición y que se puedan realizar pruebas con este contenido. Disponible en Kaggle con una usabilidad del 8.8 y con 1309 instancias puesto que estos datos contemplan desde la temporada 2014/2015 a 2017/2018.

Objetivos

El objetivo es extraer información interesante de los datos tratando de obtener predicciones ¿Cómo afecta el ir ganando al descanso siendo visitante ?¿Siendo local ?… En general la idea es obtener conocimiento sobre las relaciones goles/victoria - Local/Visitante, que permitan realizar predicciones.

Así a primera vista se aprecia que el equipo local suele ser el que consigue la victoria y que lo menos probable es que empaten.

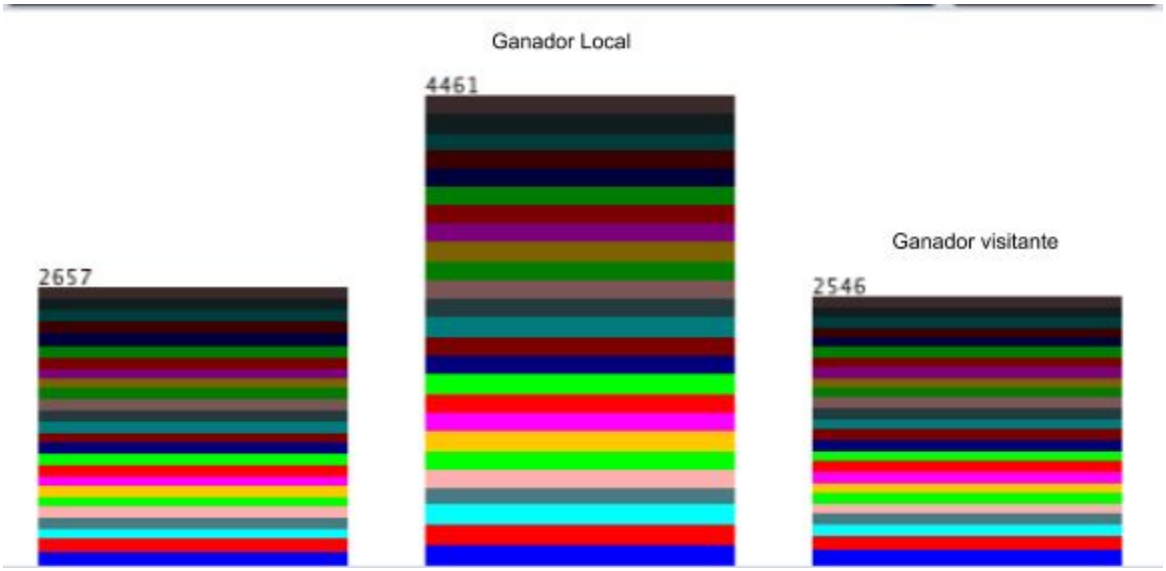


Imagen: gráfica extraída de WEKA

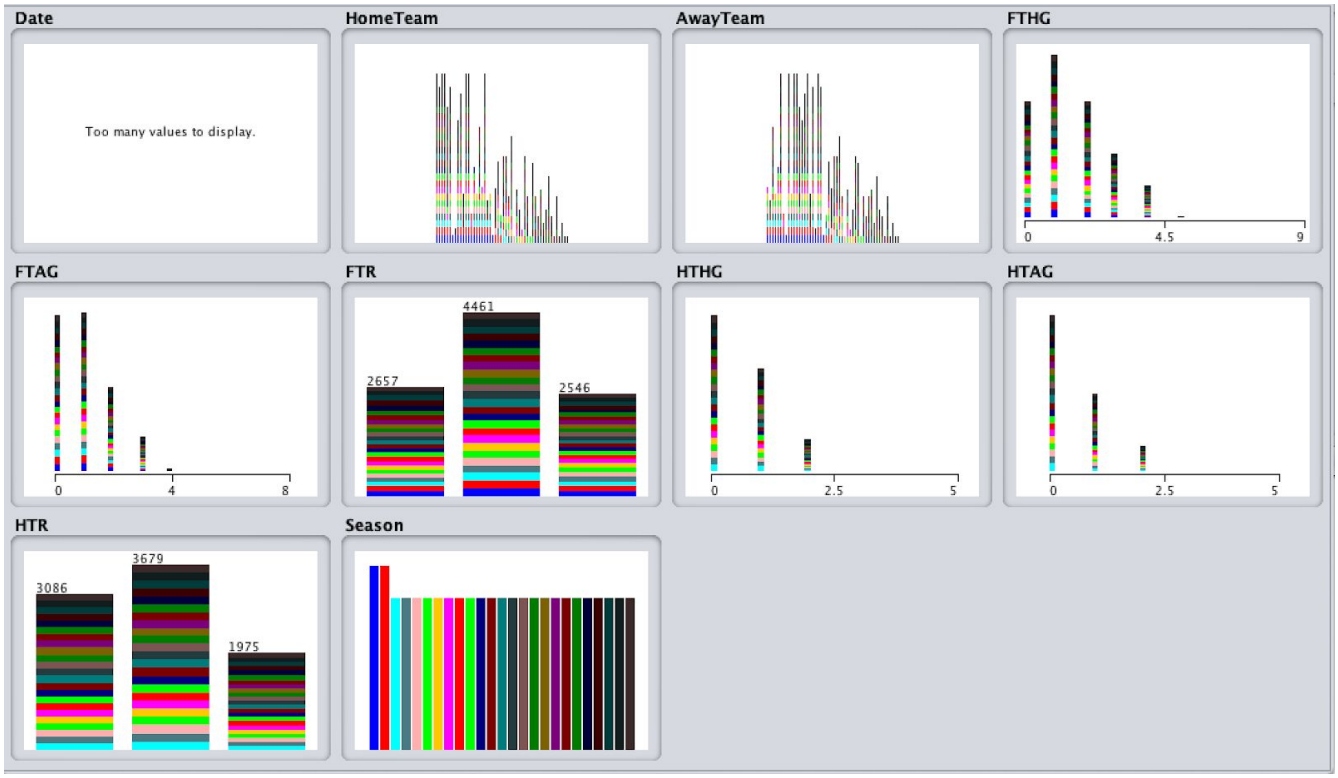


Imagen: Muestra de todos los gráficos resultantes a cada campo.

El resto de resultados se aprecia que lo normal es que tengamos menos de 3 goles como visitantes y como locales y parece que es más normal meter goles como local, normal si tenemos en cuenta que hay más victorias como local.

Ahora que tenemos un segundo data set más completo vamos a actualizar el contenido de este pequeño estudio

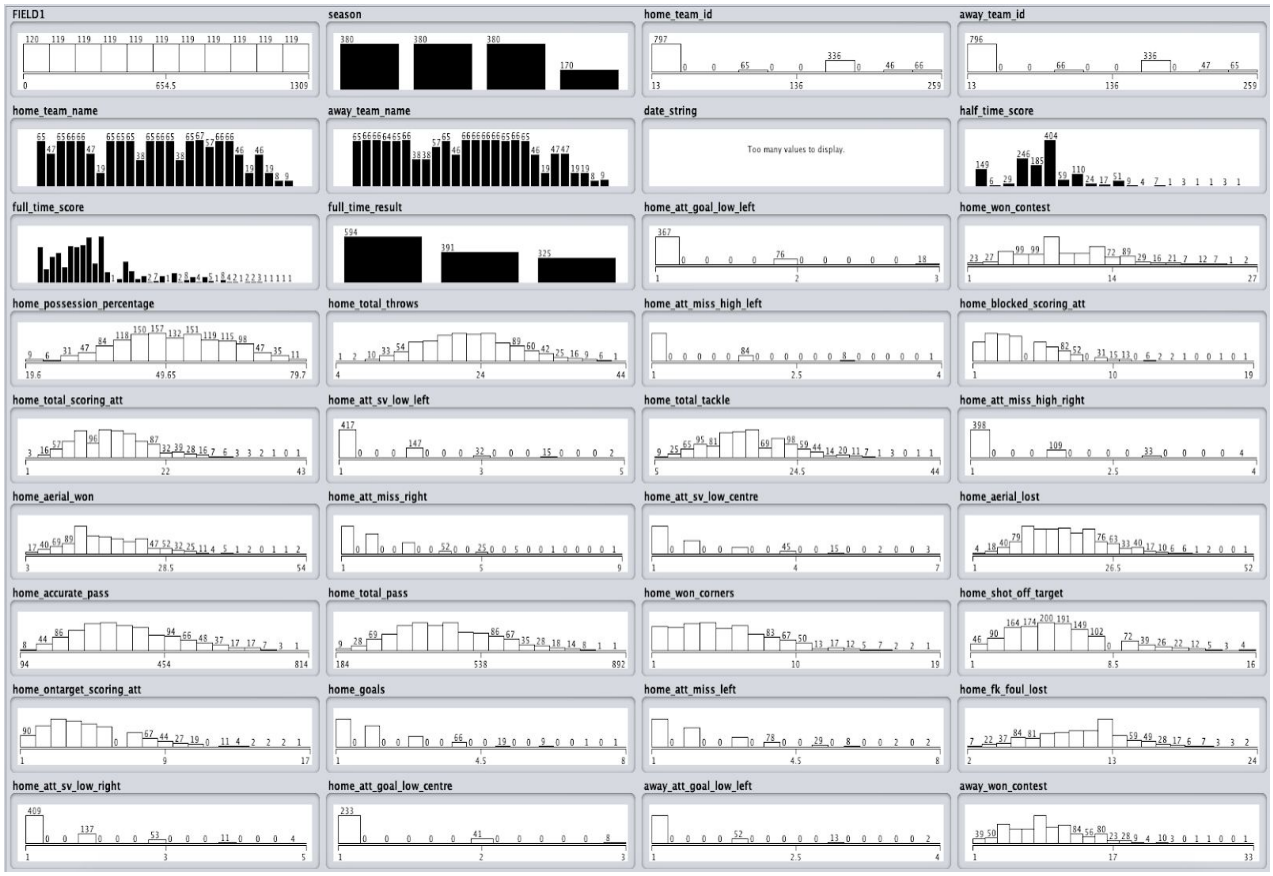


Imagen: Muestra de todos los gráficos resultantes a cada campo.

Como se aprecia en la tabla tenemos nuevos campos como el número disparos de fallidos, el número de paradas, todos los pases, centros …

Gracias a subirlo a Kaggle con el bot de creación de kernel automatico hemos obtenido la matriz de confusión de los datos creados.

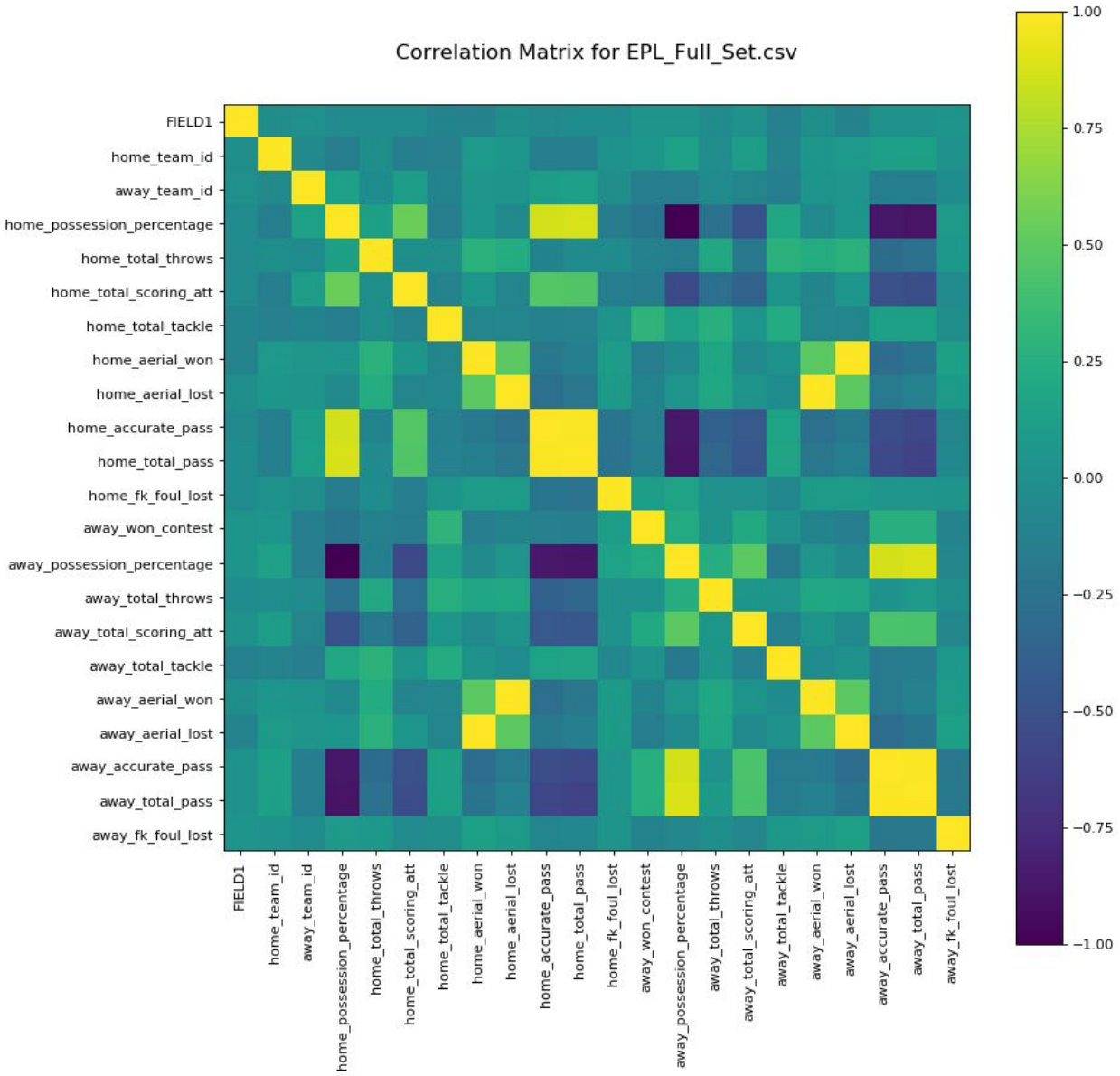


Imagen: Matriz de confusión de los datos creados sobre la Premier League

Y luego también la matriz de dispersión y densidad:

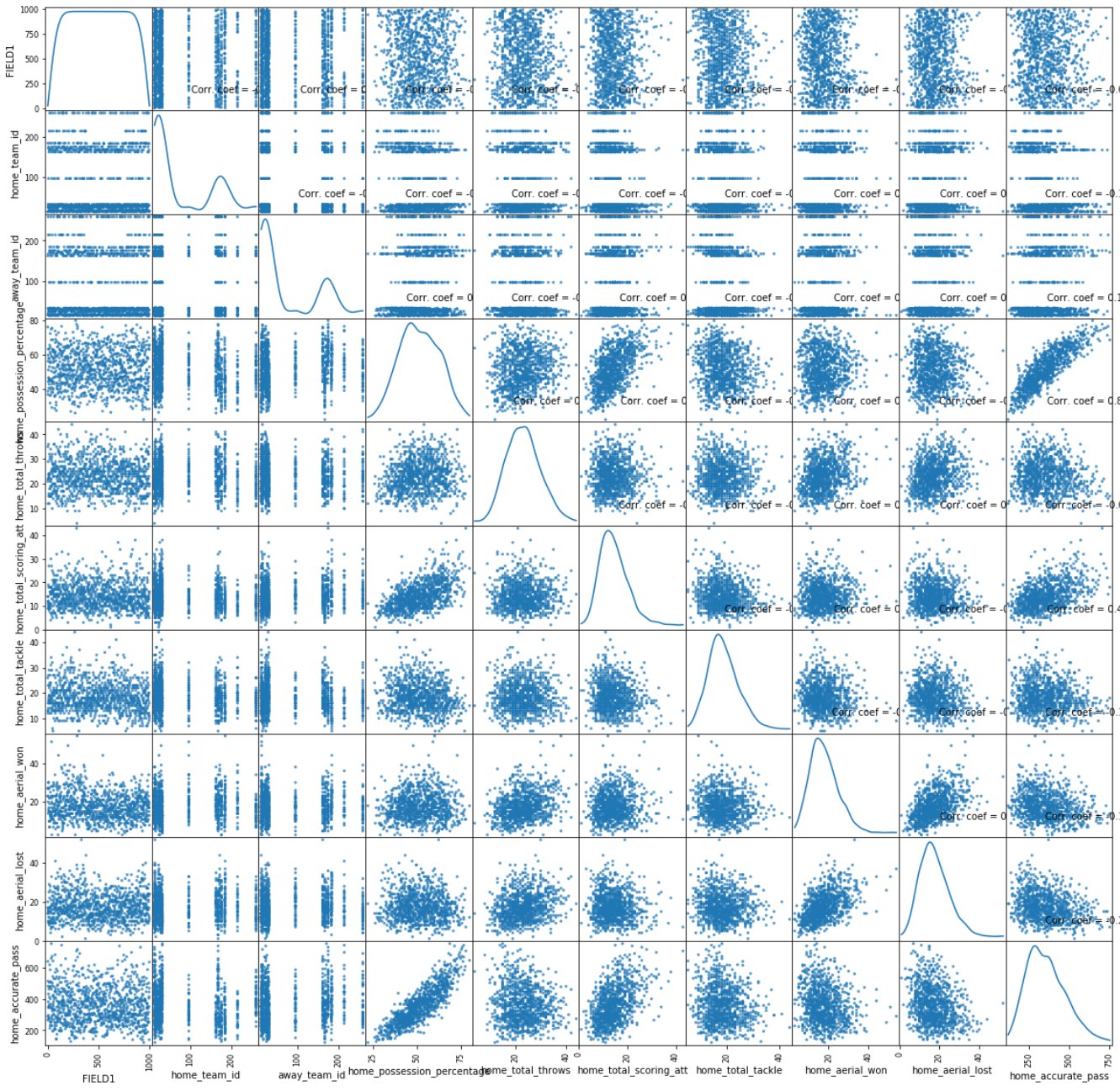


Imagen: Matriz de dispersión y densidad

3- Creación y agrupación de los datos

Entrando en la creación de los datos y con el fin de clarificar lo máximo posible los resultados en los dos dataset creados decidí seguir la misma metodología de trabajo. Recordar que el codigo esta disponible en DataMining_MUSS_Main.py

Primero realice una inspección de los datos en crudo en el caso del primer dataset decidí que todos los partidos tenían una datos parecidos y los unifique realizando además algunas funciones que me permiten crear nuevos campos como seleccionar los goles del descanso por separado o el ganador del partido y no solo el resultado final.

Para el segundo dataset al utilizar a los jugadores la dificultad incrementó de forma muy considerada puesto que estos no comportarte casi ninguna estadística, ni siquiera los que juegan en las mismas posiciones. Para solucionar esto cree un objeto Player que será único para todos los jugadores y que por cada partido agruparon los resultados de cada jugador, de esa manera pese a que muchos jugadores tenían estadísticas vacías pude terminar de agrupar a todos los jugadores. Por supuesto el tiempo de ejecución de este segundo proceso superó con creces el del primer dataset con un tiempo entorno a los 15 minutos de ejecución.

Sobre el formato de salida, el proceso que seguí fue sacarlo como json y despues pasarlo a csv con una librería de python, además también hay conversores online de json a csv que pese a no aceptar fichero tan grande podrían ser útiles.

Después como Weka tiene su conversor de CSV a ARFF solo con pasarlo por el programa realiza la conversión sin mucho problema.

4- Aplicación de clasificadores

Aplicación al primer DataSet

Lo primero que quiero realizar como introducción es aplicar las clasificaciones vistas en clase y tratar de ir mejorando los resultados.

Por ejemplo primer caso, aplicamos el J48 sobre el resultado final del partido

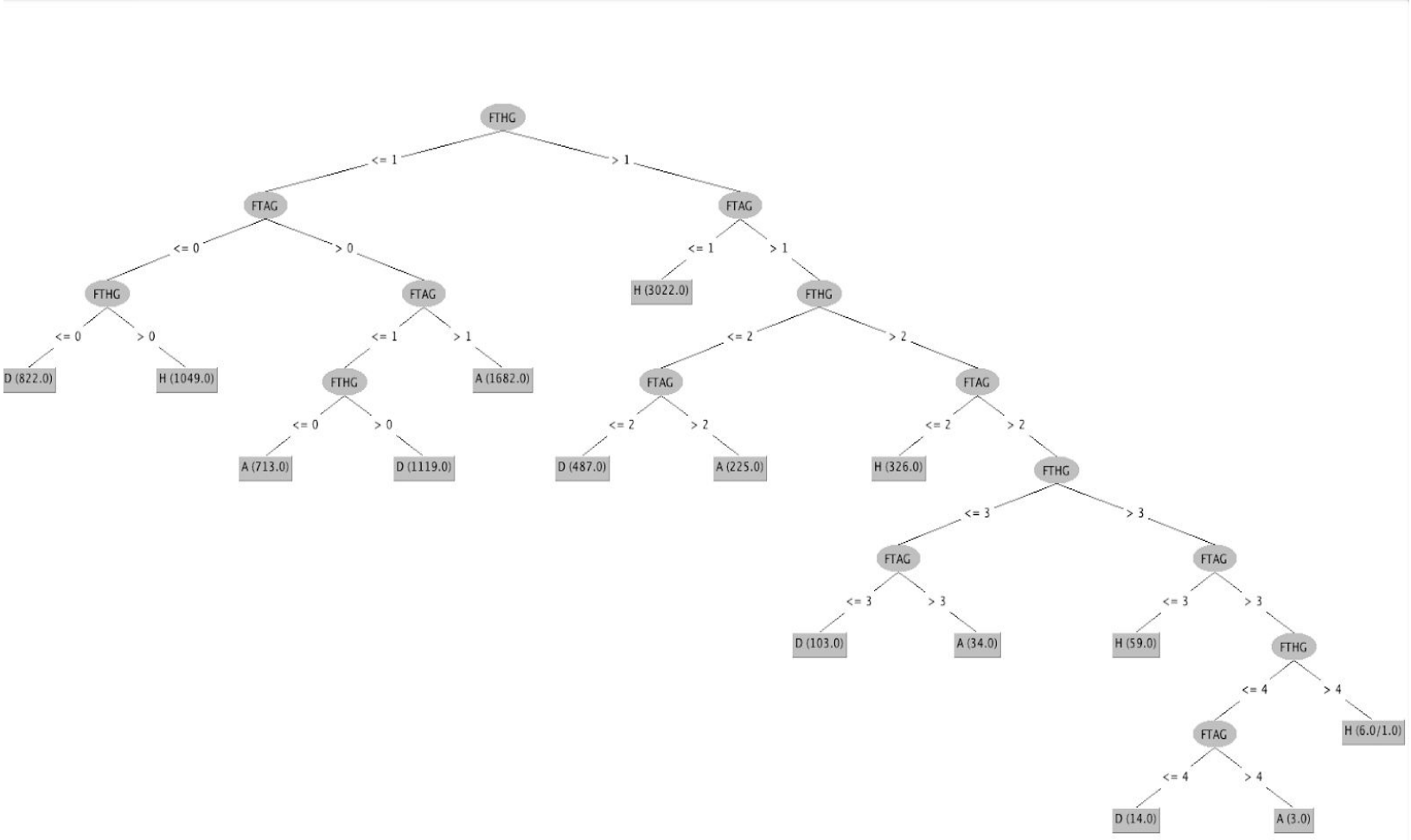


Imagen: Resultado obtenido de aplicar J48 con cross-validation 10

Tenemos un porcentaje de instancias correctamente instanciadas del 99.9793 %. Sobre el resultado se puede ver que por una parte nos dice conocimientos básicos, como que si el equipo local mete 1 o 0, el visitante 0 , gana el local y 1 empate. Son los conceptos básicos del juego.

Si por Ejemplo realizamos el mismo experimento pero para el resultado a mitad del partido:

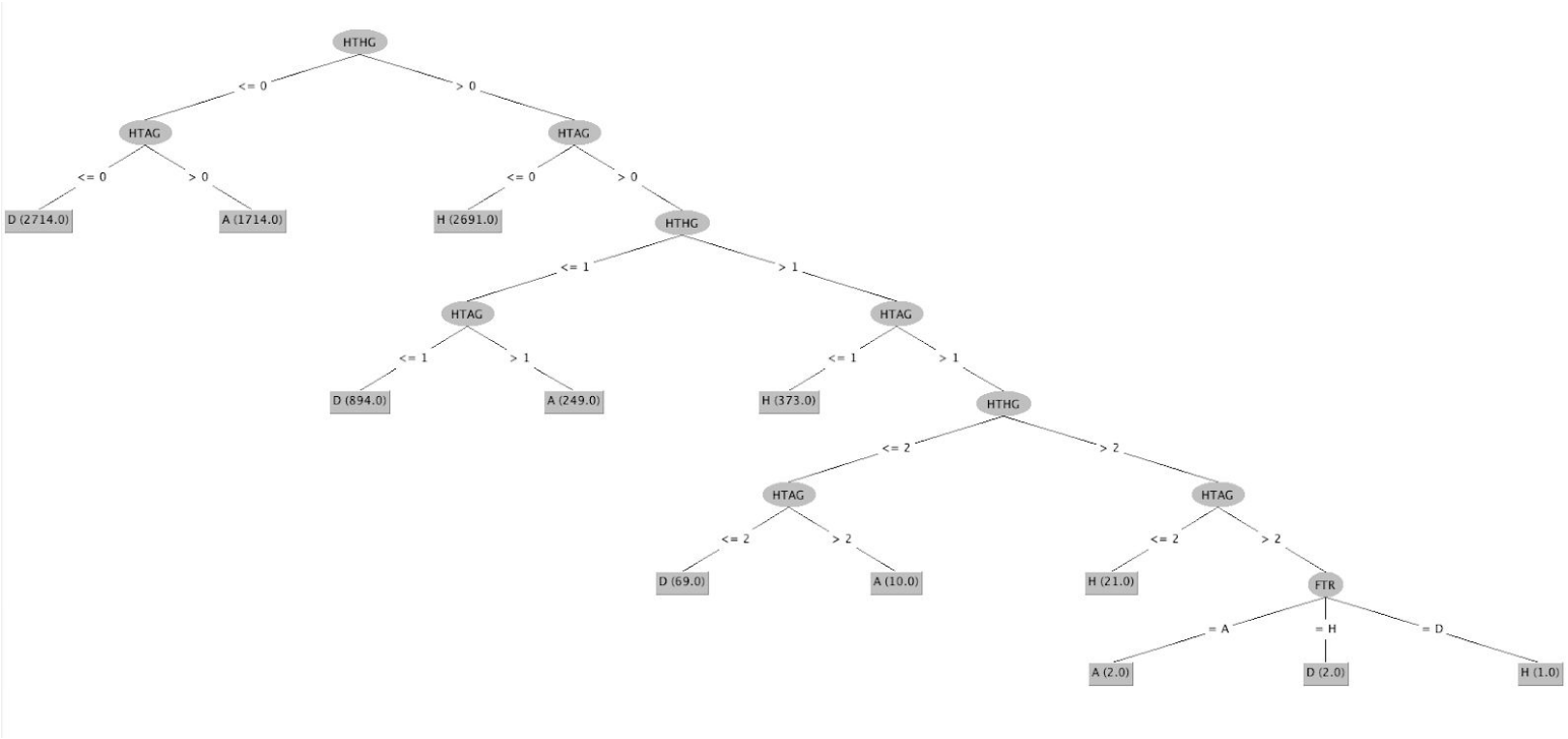


Imagen : Resultado obtenido de aplicar J48 con cross-validation 10

Podemos apreciar que es prácticamente igual, con una diferencia muy significativa

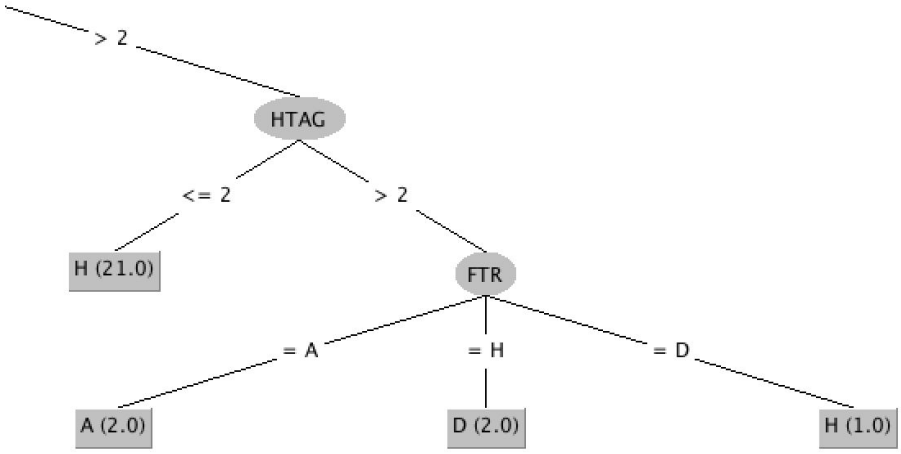


Imagen:Resultado obtenido de aplicar J48 con cross-validation 10

Lo curioso es que en el caso de que los equipos fuesen con más de 2 goles cada uno al descanso el resultado de la primera mitad según el resultado final sería tal que , si ha perdido el local iría perdiendo al descanso, si por el contrario ganó el partido iba empatado al descanso, y si empataron el local iría ganando al descanso.

He tratado de aplicar también el Random Forest o el Random Tree, pero no he conseguido ningun resultado que aportase algo, o incluso se me ha caido weka tratando de crear el modelo.

Es interesante aplicar el j48 sobre la temporada y ver cómo afecta a cada equipo con sus enfrentamientos y las probabilidades de cada enfrentamiento:

```
HomeTeam = Arsenal
|   AwayTeam = Coventry
|   |   FTAG <= 0
|   |   |   HTHG <= 0: 1996-97 (2.0/1.0)
|   |   |   HTHG > 0: 1997-98 (2.0/1.0)
|   |   FTAG > 0
|   |   |   FTHG <= 1: 1993-94 (2.0/1.0)
|   |   |   FTHG > 1: 1994-95 (2.0/1.0)
|   AwayTeam = QPR
|   |   FTHG <= 1: 1993-94 (4.0/3.0)
|   |   FTHG > 1: 1995-96 (2.0/1.0)
|   AwayTeam = Blackburn
|   |   FTR = A: 1997-98 (2.0/1.0)
|   |   FTR = H
|   |   |   HTAG <= 0
|   |   |   |   FTHG <= 2: 1998-99 (3.67/2.67)
|   |   |   |   FTHG > 2: 2004-05 (3.0/2.0)
|   |   |   HTAG > 0: 2006-07 (3.33/2.33)
|   |   FTR = D
|   |   |   FTHG <= 0: 1994-95 (3.0/2.0)
|   |   |   FTHG > 0: 1996-97 (2.0/1.0)
```

Imagen: Ejemplo de los resultados obtenidos

Es imposible apreciar estos resultados en el árbol puesto que realiza este análisis para cada equipo y todos los equipos de la competición, pero la verdad necesito depurar el análisis para intentar sacar mejores datos de estos.

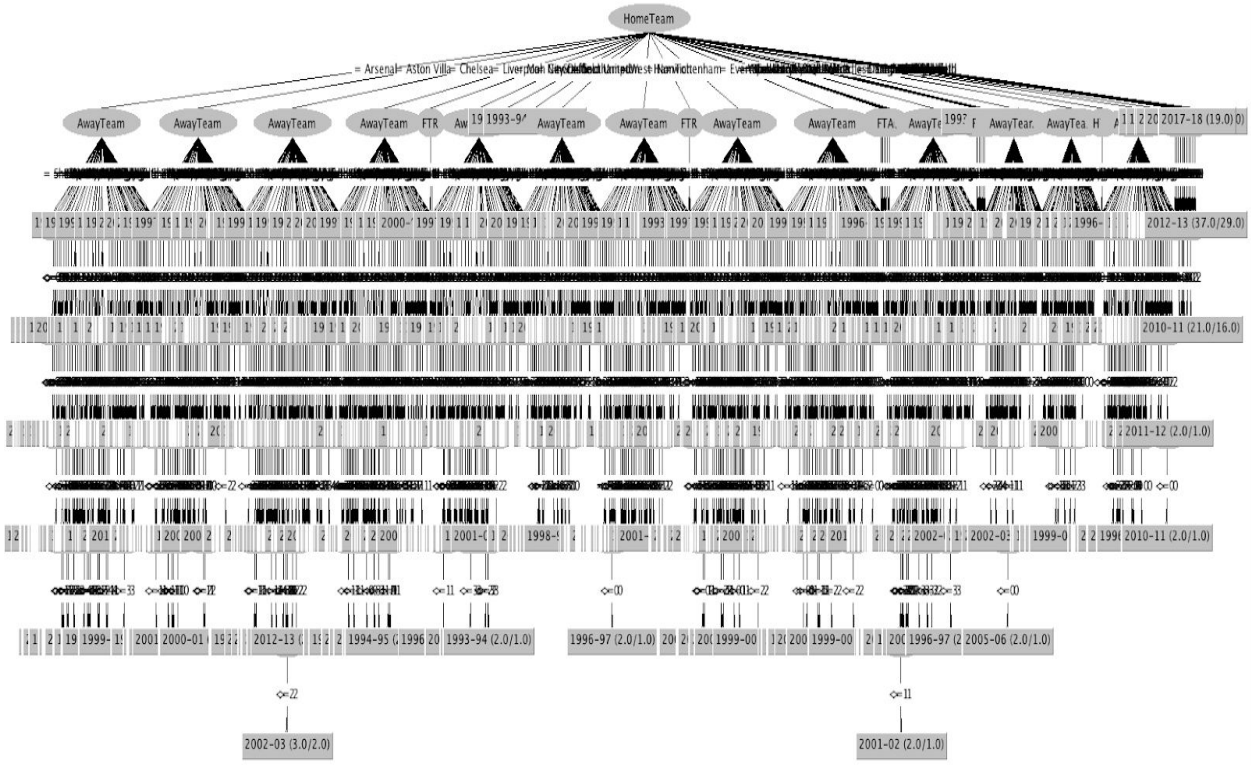


Imagen:Resultado obtenido de aplicar J48 con cross-validation 10

Aplicación al segundo DataSet

Lo primero que he hecho es depurar los datos y quitar los datos no relevantes como el id de cada partido y las fechas. Y a continuación he aplicado los algoritmos de clasificación sobre los datos nuevos, con los siguientes resultados :

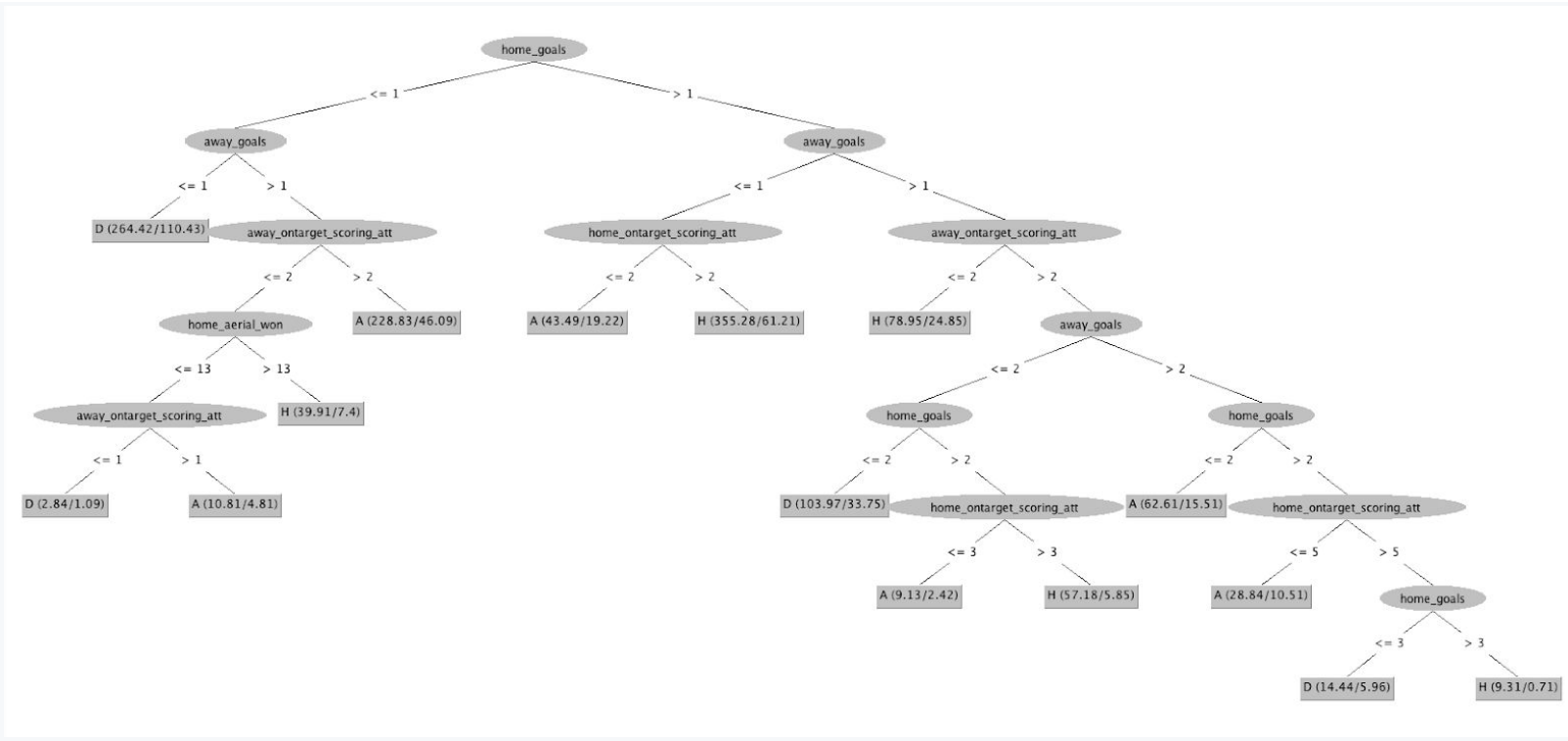


Imagen:Resultado obtenido de aplicar J48 con cross-validation 10 con un 78% de corrección de instancias

En este resultado vemos que lo que siempre encuentra relacionado es el número de goles, pero en este caso encontramos también la media de goles de cada equipo cuando juega fuera y su media de ganancias en el juego aéreo.

Como por ejemplo vemos que si el local mete 1 o menos goles , podemos ver como dependiendo del número de balones aéreos <13 ganando el visitante o empatando o >13 puede ganar el local.

También he realizado más pruebas quitando directamente los goles y buscando relaciones en los otros atributos.

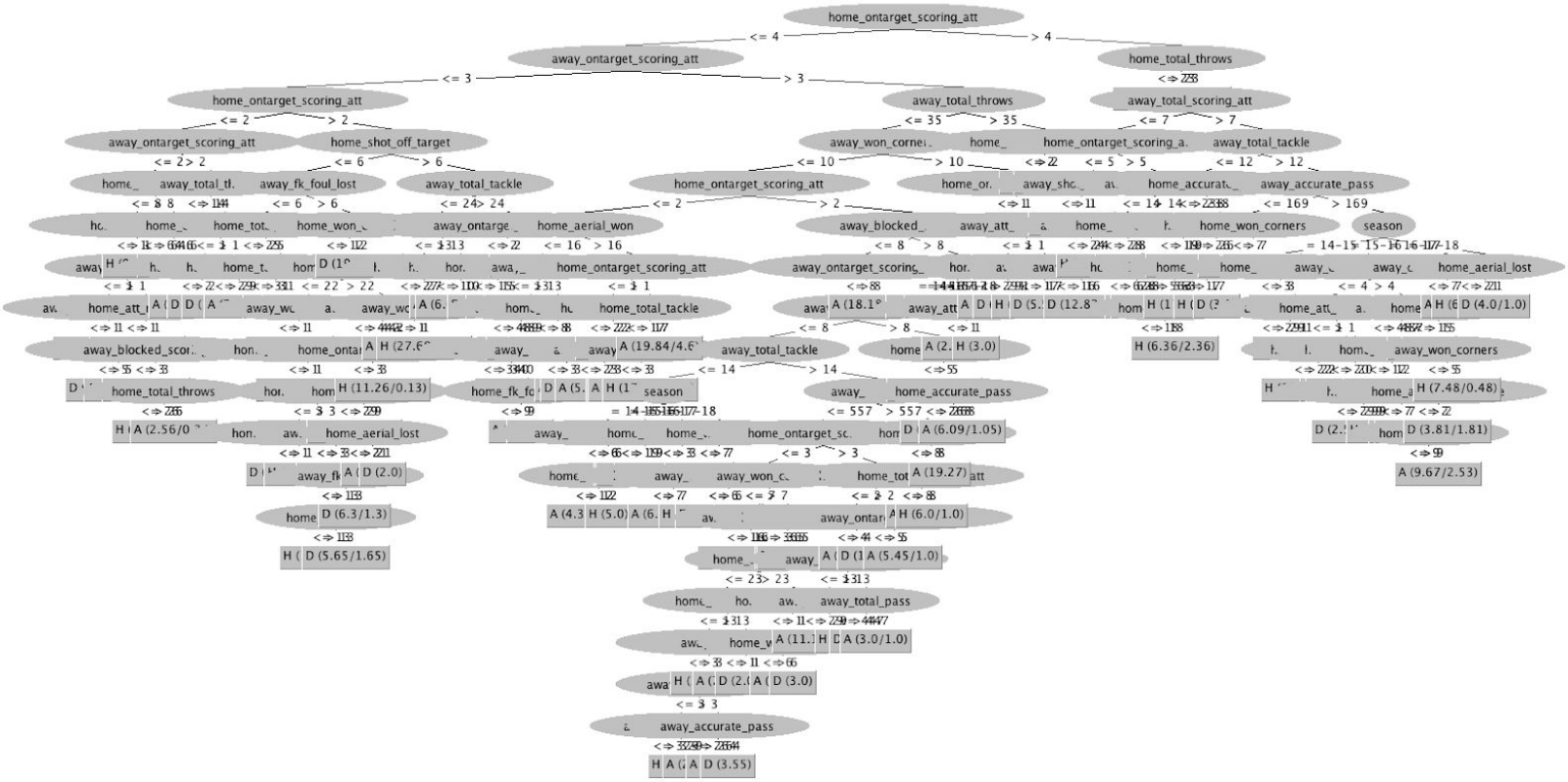


Imagen:Resultado obtenido de aplicar J48 con cross-validation 10 con un 52,33% de corrección de instancias

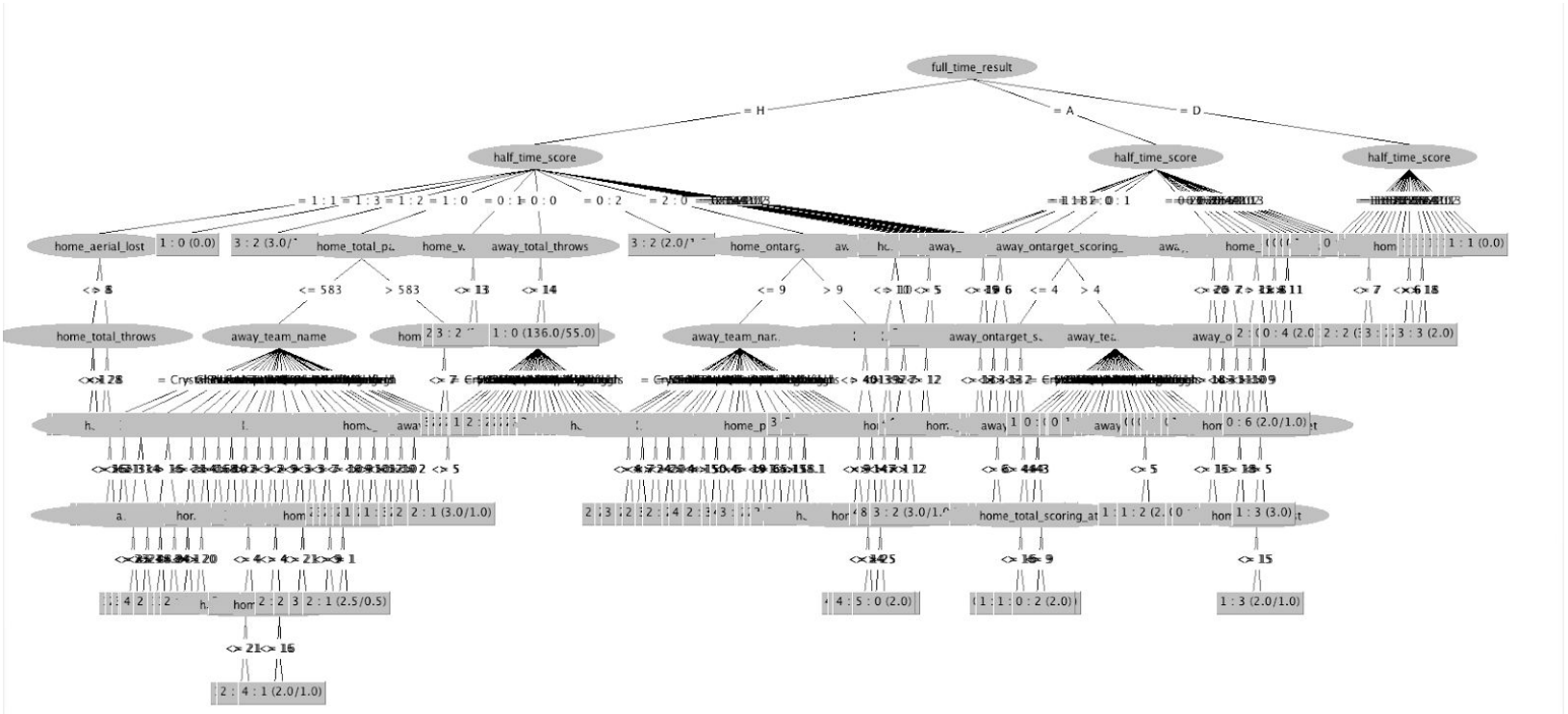
Nos encontramos con nuevos resultados que lamentablemente tienen instancias correctamente clasificadas el 52,33% osea que hemos bajado considerablemente, pero podemos ver como ha encontrado nuevos caminos como el siguiente :

En el que podemos ver como los corners o los errores dependiendo de la banda afectan al resultado final.

1. home_ontarget_scoring_att <= 4
2. | away_ontarget_scoring_att <= 3
3. | | home_ontarget_scoring_att <= 2
4. | | | away_ontarget_scoring_att <= 2
5. | | | | home_won_corners <= 8

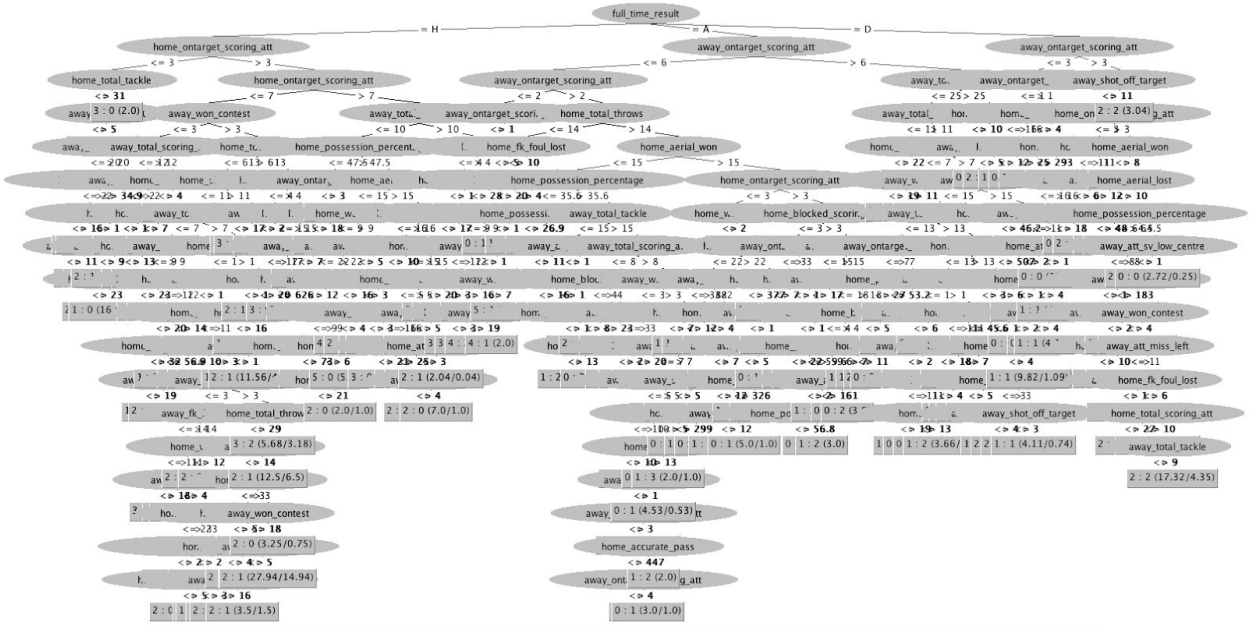
6. | | | | | home_att_miss_high_left <= 1
7. | | | | | | away_att_miss_left <= 1
8. | | | | | | | away_att_miss_high_left <= 1: D (15.32/8.83)
9. | | | | | | | away_att_miss_high_left > 1
10. | | | | | | | | away_shot_off_target <= 5: D (2.87/1.09)
11. | | | | | | | | away_shot_off_target > 5: A (2.46/0.14)
12. | | | | | | away_att_miss_left > 1
13. | | | | | | | home_att_miss_right <= 1: H (10.05/2.44)

Durante la última semana trate de encontrar relaciones al resultado final que no estuvieran relacionado con los goles totales, lo primero que encontré fue como afecta el resultado durante el descanso en el resultado final :



Lo interesante es que tenemos un 41% de clasificación pero tenemos una idea de cómo puede afectar el resultado del descanso al resultado final. Es curioso ver que una vez pasado el resultado al descanso para a ser importante las pérdidas totales de balones y también como separa en función del nombre del equipo se podrían estudiar las tendencias de cada equipo ante estos resultados en el descanso.

Si por el contrario terminamos de quitar los goles el resultado se dispersa bajando al 20% de clasificación :

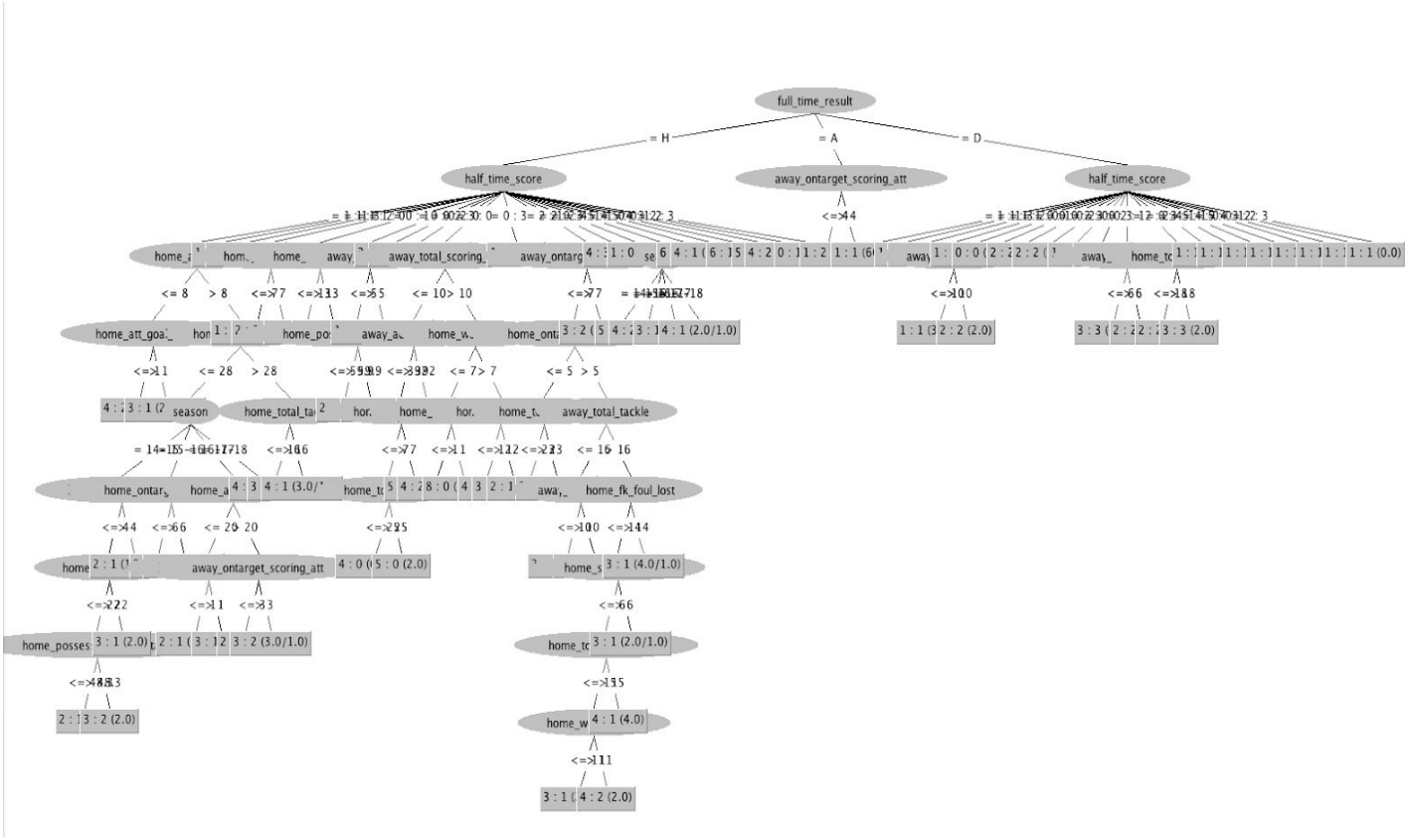


Como vemos lo primero será el atributo de goles, pero luego entran en juego el número de entradas agresivas y el número de balones recuperados. Cosa que tiene sentido puesto que a mejor defensa menos goles encajados, aunque lo que también se puede ver es que en casos de nº entradas muy altas puede significar un asedio constante por parte del equipo rival.

Con el dataset de jugadores no fui capaz de obtener ningún resultado , el tener datos tan dispares no permitió a los algoritmos encontrar resultados con un mínimo de correlación

5- Evaluación de los resultados finales

Después de la evaluación de todos los resultados uno de los resultados más interesantes fue el que se muestra a continuación :



Como se puede ver es interesante porque permite en función del resultado del descanso predecir el resultado final , este resultado tiene una corrección del 45% con casi 600 instancias correctas esto quiere decir que de 1310 instancias podríamos apostando por partido tener un beneficio positivo. Adentrando en el grafo además encontramos predicciones en resultado que sorprende sobre todo en el caso de predecir remontadas , como el resultado del descanso siendo 1:2 puede terminar 3:2, es cierto que el resultado de corrección no es excesivamente alto pero teniendo en cuenta el carácter del campo son resultado que podrían ser de gran ayuda a la hora de realizar una apuesta. Desde un punto de vista más personal , es posible que estos resultado no tengan que ser tomados como único punto para realizar una apuesta deportiva puesto que el tener en cuenta

los anteriores resultados o cambios en la plantilla durante el mercado de fichajes puedan influir mucho en el resultado, pero puede ser útil como punto de inflexión entre una decisión o otra.

Sobre el resto del trabajo la realización de los datasets con python me ha resultado interesante y subirlos a kaggle y tener soporte de personas que han podido utilizar el dataset y me han recomendado técnicas para mejorar los resultados futuros ha sido muy interesante. Además de aprender a mejorar desde el punto de inicio como relacionar datos y predecir datos que no son interesantes y que luego en weka tenía que descartar por no descartar durante la creación del dataset.