Predicción de resultados de partidos

Deportes electrónicos y un poco de contexto

League of legends forma parte de aquellos videojuegos catalogados como deportes electrónicos.

Son competiciones de videojuegos que se han convertido en eventos de gran popularidad.

Se caracterizan por …

1. Ser juegos juegos jugados por miles y millones de jugadores
2. Poseen ligas y torneos al igual que muchos de los deportes convencionales
3. Jugadores profesionales son remunerados como un trabajo normal
4. Poseen competiciones internacionales que son transmitidas a través de internet y visualizadas por decenas de miles en simultáneo.

Otros deportes electrónicos conocidos

Bases del juego

1. Se enfrentan 2 equipos (Azul y Rojo)
2. Cada equipo tiene 5 integrantes, cada uno ocupa una posición, por tanto cada partido está compuesto por 10 jugadores.
3. Cada integrante del equipo elegirá un personaje (entre más de 120) acorde a su posición
4. Cada personaje tiene habilidades que sirven para eliminar a los personajes enemigos, y de esa forma poder derribar las torres
5. Quien destruya el nexo (la torre más importante en el mapa) se corona como ganador del partido.
6. No existen los empates

Mapa explicativo

Obtención del dataset

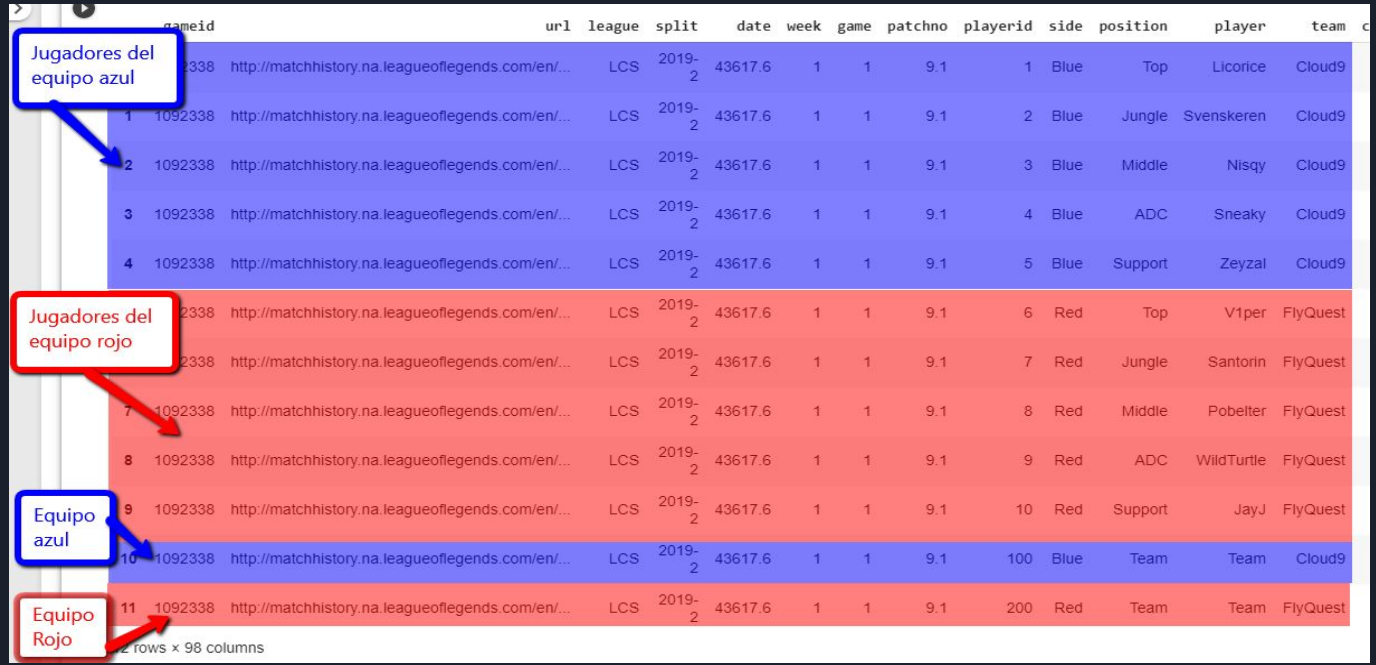
Los datasets requeridos para hacer el análisis fueron obtenidos de http://oracleselixir.com/match-data/ , y cuentan con su correspondiente diccionario de datos explicando cada variable del dataset.

Estos 2 dataset contienen la información de los torneos de verano y primavera para cada una de las siguientes ligas

- CBLoL = Brasil
- LCS = Norteamerica
- LEC = Europa
- LCK = Corea
- LMS = Taiwan , HongKong y Macao

Estructura básica del dataset

El dataset contiene para cada partido, información del desempeño de cada jugador y además información de cada equipo participante del encuentro, en total son 12 registros por partido



	gameid	url	league	split	date	week	game	patchno	playerid	side	position	player	team	c
0	1092338	http://matchhistory.na.leagueoflegends.com/en/...	LCS	2019-2	43617.6	1	1	9.1	1	Blue	Top	Licorice	Cloud9	
1	1092338	http://matchhistory.na.leagueoflegends.com/en/...	LCS	2019-2	43617.6	1	1	9.1	2	Blue	Jungle	Svenskeren	Cloud9	
2	1092338	http://matchhistory.na.leagueoflegends.com/en/...	LCS	2019-2	43617.6	1	1	9.1	3	Blue	Middle	Nisqy	Cloud9	
3	1092338	http://matchhistory.na.leagueoflegends.com/en/...	LCS	2019-2	43617.6	1	1	9.1	4	Blue	ADC	Sneaky	Cloud9	
4	1092338	http://matchhistory.na.leagueoflegends.com/en/...	LCS	2019-2	43617.6	1	1	9.1	5	Blue	Support	Zeyzal	Cloud9	
5	1092338	http://matchhistory.na.leagueoflegends.com/en/...	LCS	2019-2	43617.6	1	1	9.1	6	Red	Top	V1per	FlyQuest	
6	1092338	http://matchhistory.na.leagueoflegends.com/en/...	LCS	2019-2	43617.6	1	1	9.1	7	Red	Jungle	Santorin	FlyQuest	
7	1092338	http://matchhistory.na.leagueoflegends.com/en/...	LCS	2019-2	43617.6	1	1	9.1	8	Red	Middle	Pobelter	FlyQuest	
8	1092338	http://matchhistory.na.leagueoflegends.com/en/...	LCS	2019-2	43617.6	1	1	9.1	9	Red	ADC	WildTurtle	FlyQuest	
9	1092338	http://matchhistory.na.leagueoflegends.com/en/...	LCS	2019-2	43617.6	1	1	9.1	10	Red	Support	JayJ	FlyQuest	
10	1092338	http://matchhistory.na.leagueoflegends.com/en/...	LCS	2019-2	43617.6	1	1	9.1	100	Blue	Team	Team	Cloud9	
11	1092338	http://matchhistory.na.leagueoflegends.com/en/...	LCS	2019-2	43617.6	1	1	9.1	200	Red	Team	Team	FlyQuest	

2 rows × 98 columns

Reflexiones sobre cómo encarar el dataset

Si bien mi idea era predecir el resultado de un partido, yo podía realizar dos tipos de predicciones, una con todos los datos de un partido puntual, determinar si el equipo ganó o perdió, una ventaja de conocer el juego y ver el dataset me llevó a la conclusión que no sería tan enriquecedor, ya que hay variables predictoras que son muy determinantes a la hora de establecer si se ganó o no un partido ya teniendo la información previamente.

Por tal motivo decidí que sería más desafiante calcular métricas de jugadores y equipos antes de llegar a disputar un partido y que el entrenamiento del modelo se realice con esa información. Esto me permitiría por ejemplo echar un resultado de algún partido que verdaderamente no ocurrió, por ejemplo para imputarlo en un sitio de apuestas.

Reflexiones sobre cómo encarar el dataset

Como comenté antes decidí tomar métricas tanto de los jugadores como de los equipos para dejar en el dataset final .

Este dataset final va a contener solamente un registro y el resultado de la clase a predecir, que es si ganó o perdió el equipo rojo.

Dataframes Matches

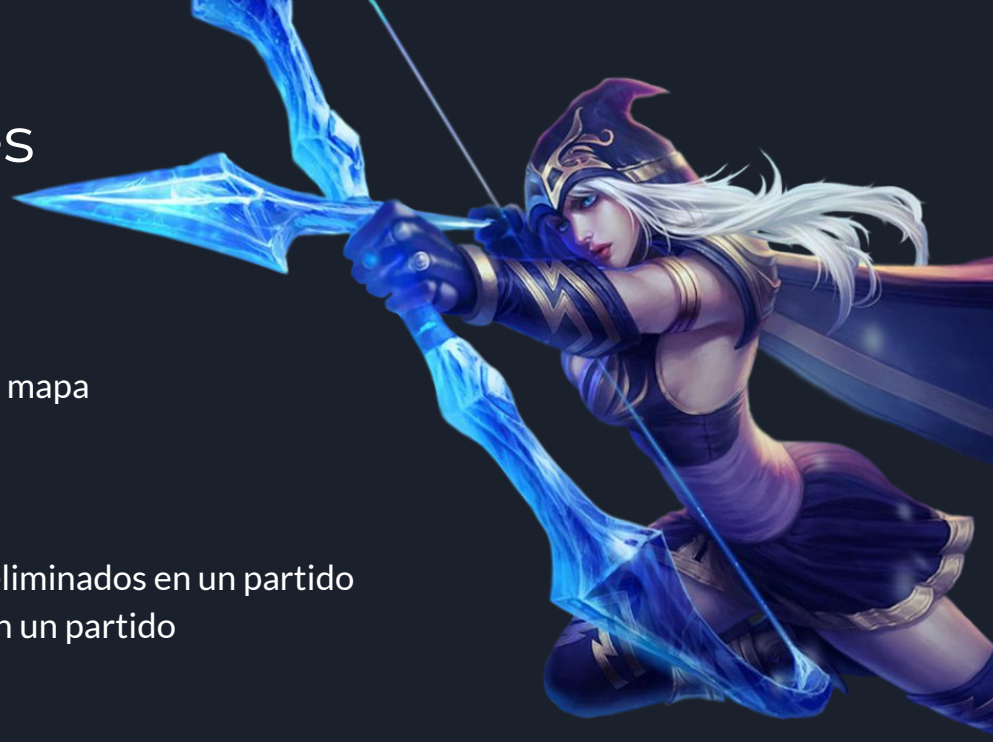
Para poder calcular estadísticas sobre jugadores al llegar a un partido decidí crear un dataframe de que filtre los registros relativos a partidos per se.

Para las estadísticas de los mismos realicé un promedio acumulativo de las siguientes métricas *)

Metricas por minuto : Mención especial, considero que son mediciones valiosas al ser datos por minuto ya que los partidos en sí pueden tener distinta duración, y esto pone a la métrica del partido en igualdad de condiciones.

- earnedgpm: Oro acumulado por minuto
- cspm: Creep score por minuto (monstruos neutrales)
- dmgtochampsperminute: Daño a un enemigo por minuto

Dataframes Matches



Métricas normales:

- wards : Elemento que da visión en el mapa
- fb: Primera eliminación
- fd: Primer objetivo neutral
- ft: Primera torre derrivada
- teamdragkills: Objetivos neutrales eliminados en un partido
- teamtowerkills: Torres eliminadas en un partido

Dataframes Players

Para poder calcular estadísticas sobre jugadores al llegar a un partido decidí crear un dataframe de que filtre los registros relativos a jugadores.

Para las estadísticas de los mismos realicé una suma acumulativa de las siguientes métricas **)

- k : Cuando un jugador elimina a un enemigo
- d: Cuando un jugador muere en manos de un enemigo
- a: Cuando un jugador asiste a otro en la eliminación de un enemigo
- kda: Es una métrica utilizada para medir el desempeño del jugador y parte de un calculo basado en las métricas anteriores

Reducción de registros a 1 por dataframe

Las métricas antes mencionadas fueron agrupadas por gameid, (id del partido) para poder reducirlas a 1 registro. ***)

Una vez esto realizado el siguiente paso fue mergear usando como clave gameid, para dejar los dos datasets en 1 solo y reducidos a 1 registro por juego

```
dfMixed.columns
```

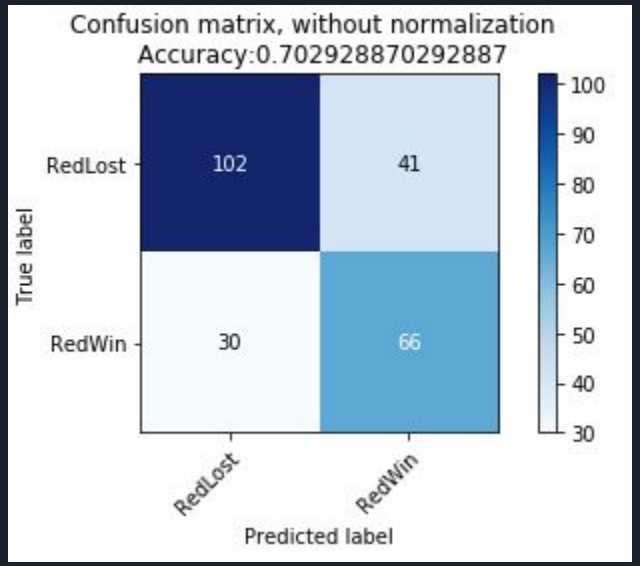
```
Index(['Redearnedgpmcmv', 'Redwardscmv', 'Red_fb_avgcumulative',
       'Red_fd_avgcumulative', 'Red_teamdragkills_avgcumulative',
       'Red_cspm_avgcumulative', 'Red_dmgtochampsperminute_avgcumulative',
       'Red_teamtowerkills_avgcumulative', 'Blueearnedgpmcmv', 'Bluewardscmv',
       'Blue_fb_avgcumulative', 'Blue_fd_avgcumulative',
       'Blue_teamdragkills_avgcumulative', 'Blue_cspm_avgcumulative',
       'Blue_dmgtochampsperminute_avgcumulative',
       'Blue_teamtowerkills_avgcumulative', 'RedWin', 'RedTop_kda',
       'RedJungle_kda', 'RedMid_kda', 'RedAdc_kda', 'RedSupport_kda',
       'BlueTop_kda', 'BlueJungle_kda', 'BlueMid_kda', 'BlueAdc_kda',
       'BlueSupport_kda'],
      dtype='object')
```

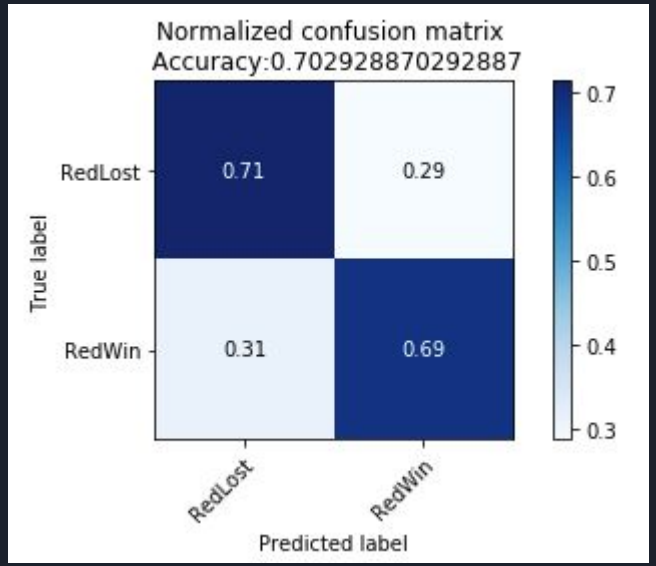
Entrenamiento de modelos en base a distintos algoritmos

Siguientes pasos

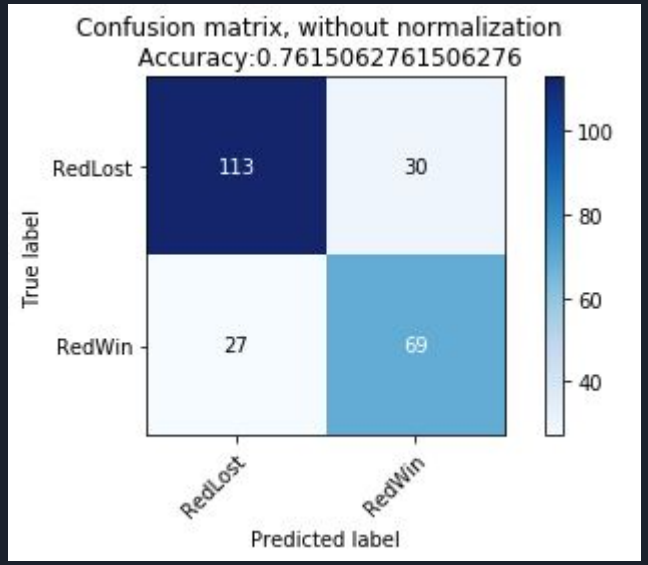
- Realice un split tomando una muestra de 80% para entrenar y un 20% para validación
- Sometí a la muestra de prueba a un entrenamiento con los siguientes algoritmos de ML
  - Decision tree Classifier
  - KNeighbors Classifier
  - SVC
  - Random Forest

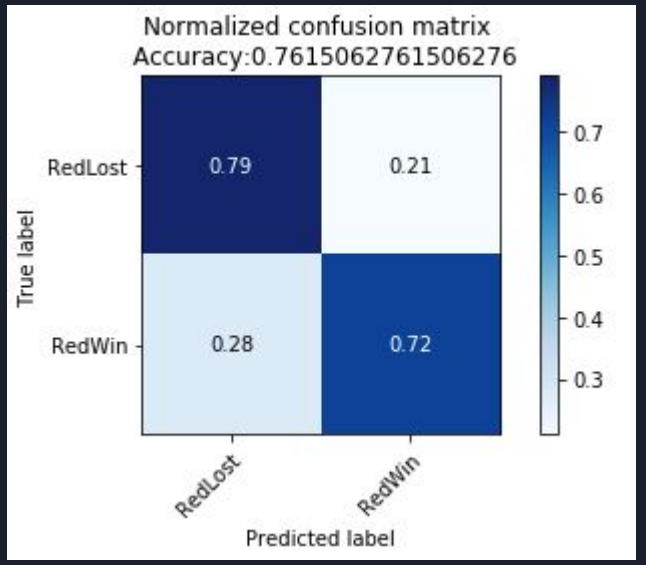
Decision Tree Classifier



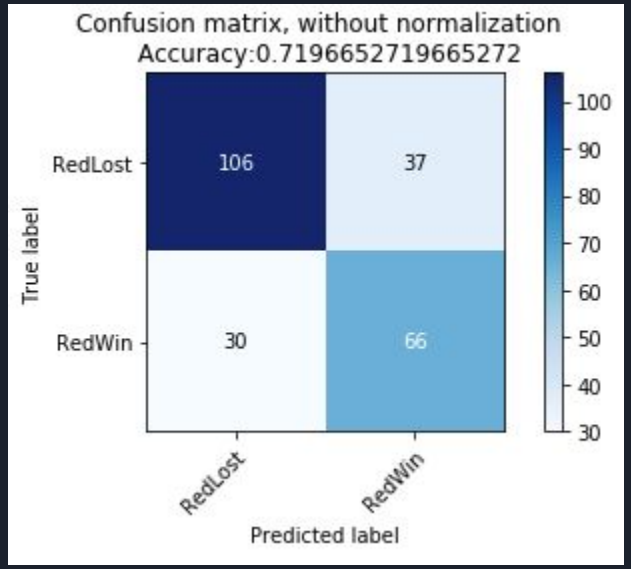


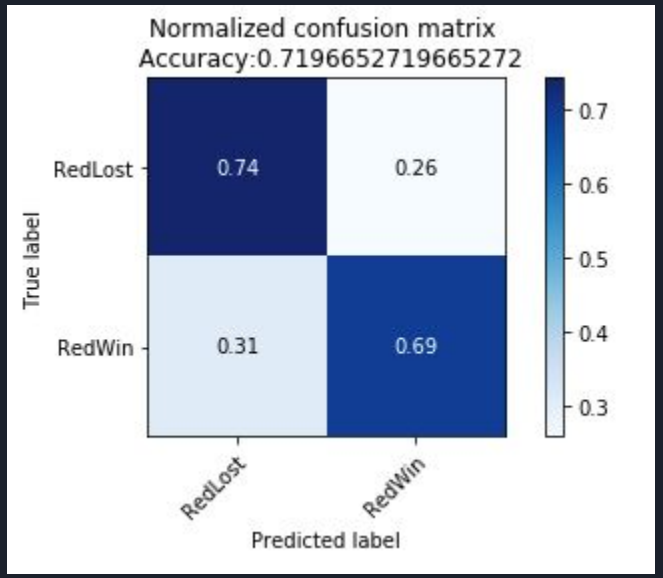
Random Forest



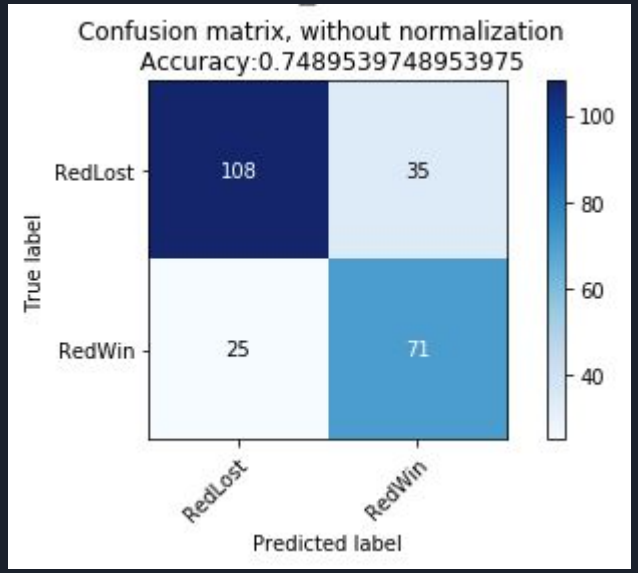


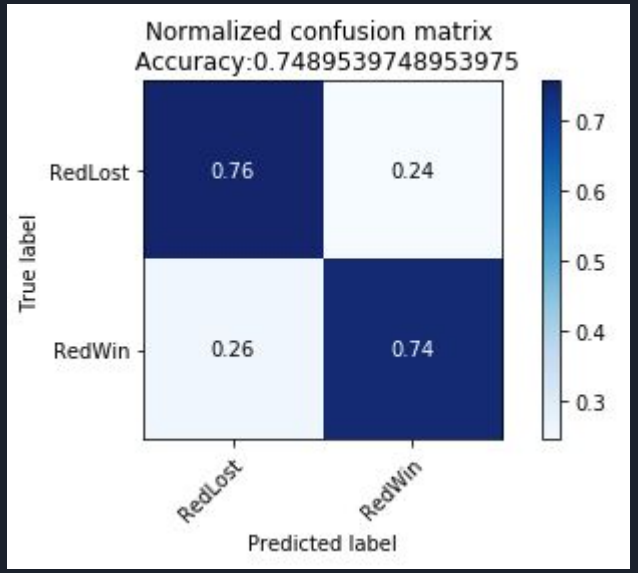
Kneighbors


Confusion matrix, without normalization
Accuracy:0.7196652719665272
RedLost
RedWin
106
37
30
66
True label
RedLost
RedWin
Predicted label
100
90
80
70
60
50
40
30


Normalized confusion matrix
Accuracy:0.7196652719665272
RedLost
RedWin
0.74
0.26
0.31
0.69
True label
RedLost
RedWin
Predicted label
0.7
0.6
0.5
0.4
0.3

Support Vector Classifier





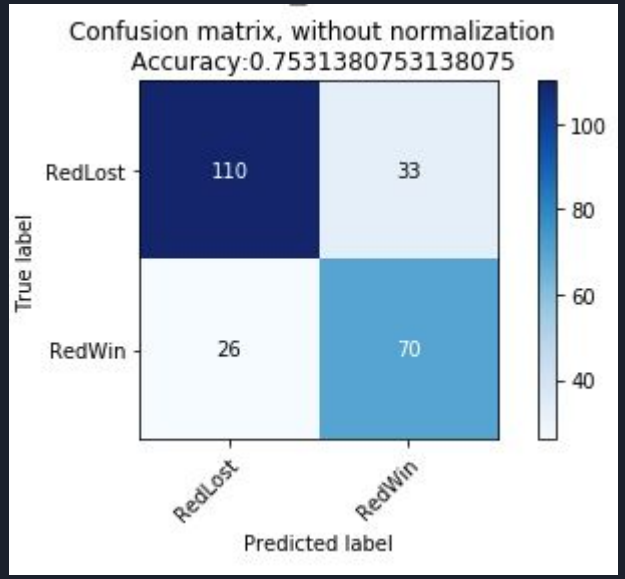
Ensamble

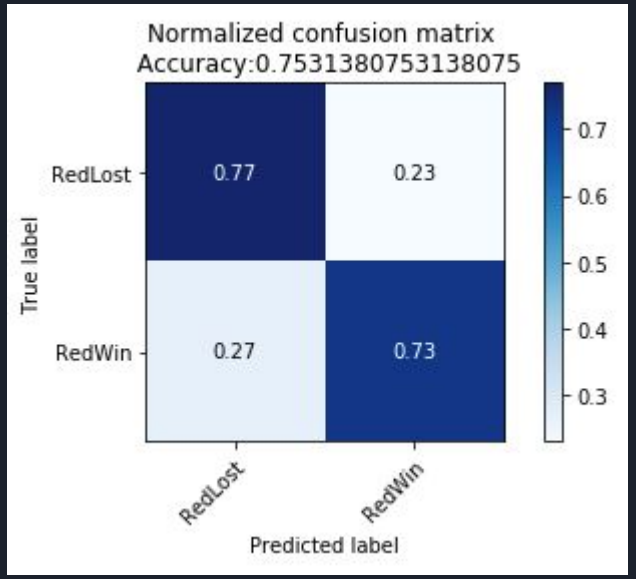
El siguiente paso que tomé fue realizar seleccionar los algoritmos que dieron los 3 mejores scores de presición y realizar una votación con una función custom.

1. RandomForest: 0.7615062761506276 - y_pred_rfc
2. Svc : 0.7489539748953975 - y_pred_svc
3. KNeighbors: 0.7196652719665272 - y_pred_kn
4. TreeClassifier: 0.702928870292887 - y_pred_tree

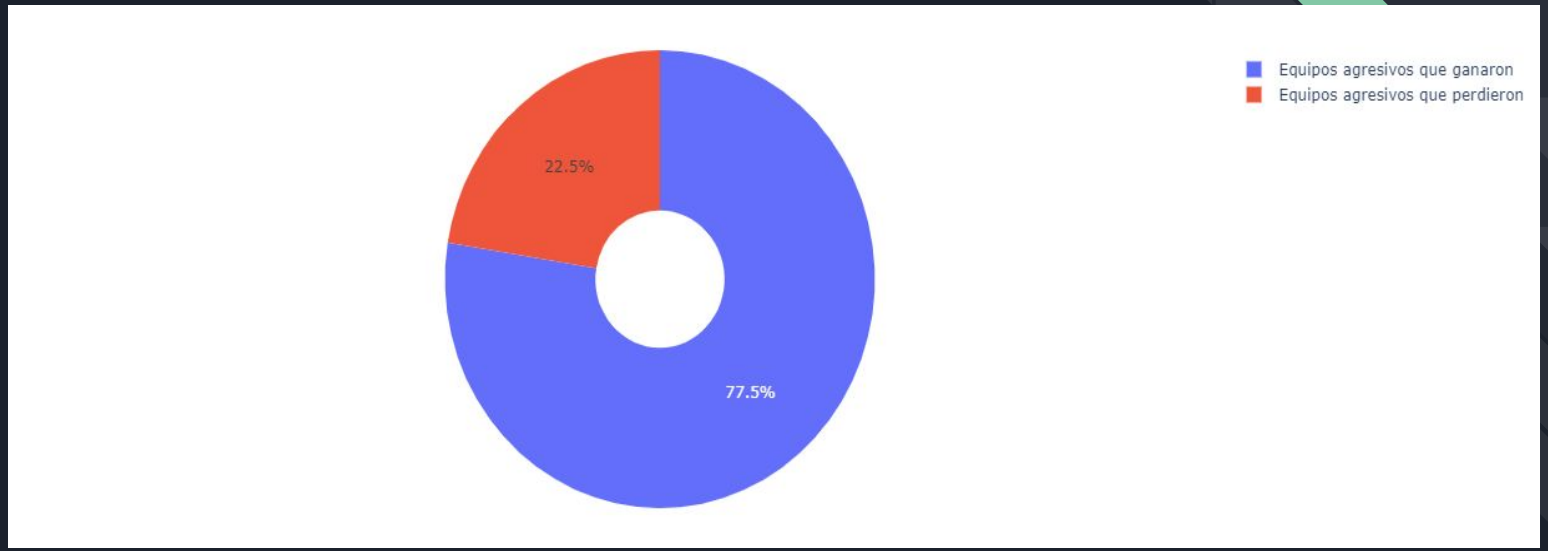
El ensamble dió menos score que el RandomForest el cual fue el que tuvo el mayor score, es decir un 76% de precisión

Ensamble de modelos

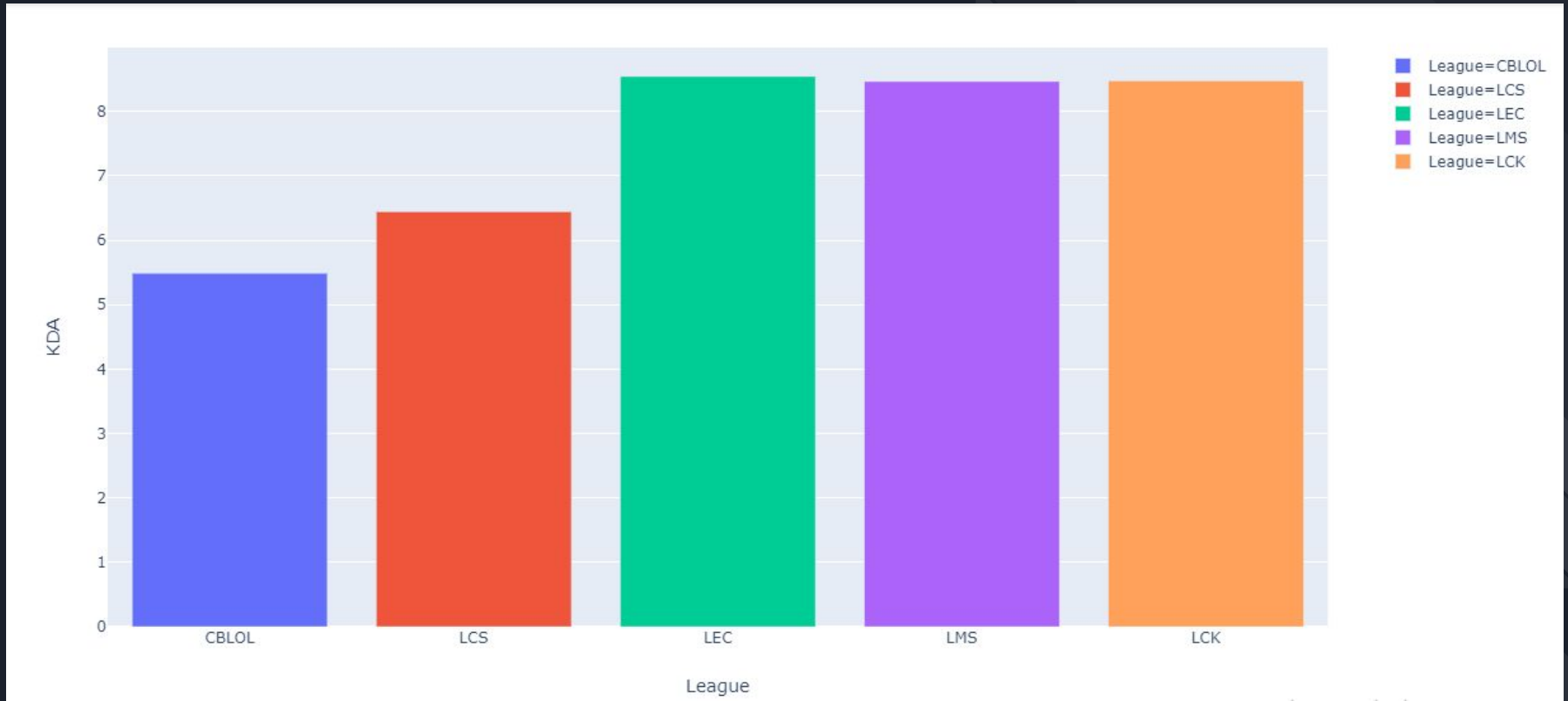




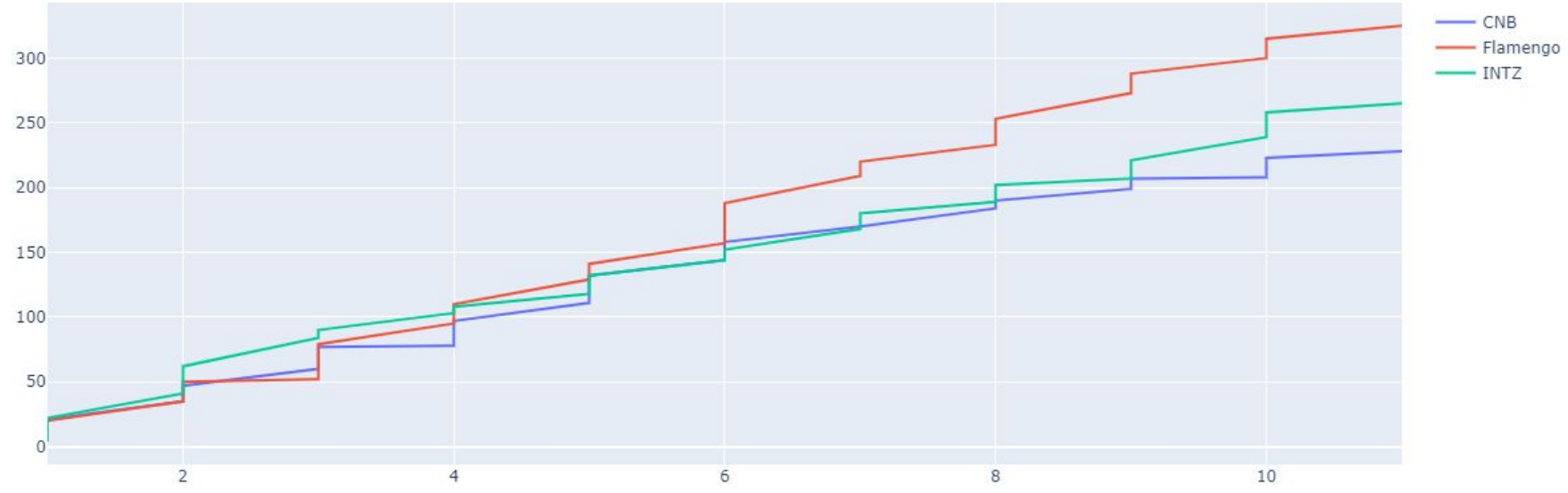
¿ Cuál es la probabilidad de ganar aplicando una estrategia efectiva en el juego temprano ?



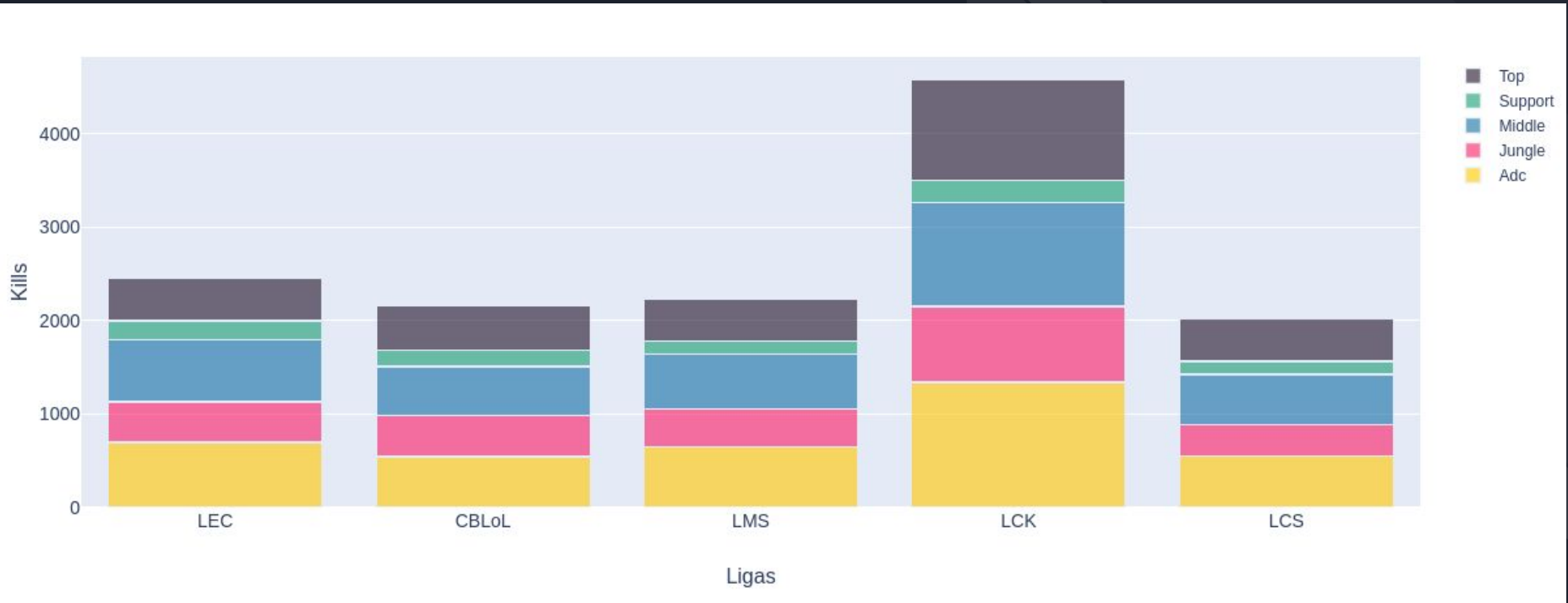
¿ Qué liga obtuvo el jugador con mayor kda al momento de llegar el último partido de cada equipo ?



Kills acumuladas por equipo a través de las semanas



¿Cómo se reparten las muertes a enemigos por rol en cada liga?



https://qrgo.page.link/iKcaG

Muchas gracias!