Reporte Final: Uso de biblioteca de aprendizaje máquina para la implementación de una solución

Héctor San Román Caraza - A01422876

***Resumen*—Usaremos una biblioteca de aprendizaje de máquina aplicada a una base de datos con el propósito de predecir los resultados en una base de datos. La base de datos que utilizaremos será un conjunto de los registros de la *Serie A* a través de distintas temporadas. Implementaremos un modelo de Random Forest, después revisaremos sus resultados y resaltaremos los hallazgos.**

I. Introducción

El fútbol es un deporte conocido de forma mundial. Tiene ligas en todas partes del mundo y sus aficionados llenan cada rincón conocido y por conocer. La pregunta aquí es, ¿a partir de datos históricos será posible predecir la posición de un equipo en la tabla general de su liga? En este trabajo inspeccionaremos los datos de la Serie A, desde la temporada 09-10 hasta la 18-19, buscando predecir las posiciones de los equipos. Para esto llevaremos a cabo un modelo de clasificación Random Forest. Para probar la significancia del modelo haremos pruebas de sesgo, de varianza y del nivel de ajuste.

II. Preparación de los datos

Primero realizamos la importación de los datos. Dado que las distintas temporadas estaban en distintos archivos tuvimos que hacer un concatenado de estos archivos. Un problema con esto es que las fechas venían en distinto formato por lo que tuvimos que poner un estándar dentro de estas; aunque lo que usamos finalmente fue solo el año del torneo en cuanto a la fecha.

Una vez con nuestra base de datos ya creada y conjunta, tenemos que echarle un vistazo:

	Div	Date	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR	...	BbAvAHH	BbMxAHA	BbAvAHA	PSH	PSD	PSA	PSCH	PSCD
0	I1	2009-08-22	Bologna	Fiorentina	1.0	1.0	1	1.0	0.0	3	...	2.38	1.60	1.54	NaN	NaN	NaN	NaN	NaN
1	I1	2009-08-22	Siena	Milan	1.0	2.0	0	1.0	1.0	1	...	1.85	2.08	2.02	NaN	NaN	NaN	NaN	NaN
2	I1	2009-08-23	Catania	Sampdoria	1.0	2.0	0	1.0	1.0	1	...	1.76	2.12	2.04	NaN	NaN	NaN	NaN	NaN
3	I1	2009-08-23	Genoa	Roma	3.0	2.0	3	0.0	0.0	1	...	1.67	2.30	2.13	NaN	NaN	NaN	NaN	NaN
4	I1	2009-08-23	Inter	Bari	1.0	1.0	1	0.0	0.0	1	...	1.79	2.15	2.05	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3800	I1	2019-05-26	Inter	Empoli	2.0	1.0	3	0.0	0.0	1	...	2.05	1.85	1.81	1.39	5.35	7.81	1.27	6.36
3801	I1	2019-05-26	Roma	Parma	2.0	1.0	3	1.0	0.0	3	...	1.85	2.10	2.01	1.20	7.50	14.07	1.17	8.59
3802	I1	2019-05-26	Sampdoria	Juventus	2.0	0.0	3	0.0	0.0	1	...	1.96	1.95	1.90	3.92	3.98	1.93	3.06	3.55
3803	I1	2019-05-26	Spal	Milan	2.0	3.0	0	1.0	2.0	0	...	2.02	1.89	1.84	6.25	4.51	1.54	5.41	4.30
3804	I1	2019-05-26	Torino	Lazio	3.0	1.0	3	0.0	0.0	1	...	2.03	1.88	1.84	2.34	3.76	3.01	2.36	3.56

3805 rows × 77 columns

Figura 1. Base de datos primer vistazo

A primera vista vemos que existen datos faltantes. También podemos ver que tenemos 3805 filas y 77 columnas. Una vez que nos deshagamos de estos datos faltantes y seleccionemos solo las columnas que nos son útiles veremos de nuevo la forma de nuestro dataset.

	Season	Date	AwayTeam	HomeTeam	AS	HS	AST	HST	HTAG	HTHG	...	FTHG	FTR	AC	HC	AF	HF	AY	HY	AR	HR
0	2009.0	2009-08-22	Fiorentina	Bologna	17.0	6.0	5.0	4.0	0.0	1.0	...	1.0	1	9.0	4.0	20.0	17.0	1.0	1.0	0.0	0.0
1	2009.0	2009-08-22	Milan	Siena	13.0	15.0	5.0	4.0	1.0	1.0	...	1.0	0	8.0	5.0	11.0	22.0	1.0	2.0	0.0	0.0
2	2009.0	2009-08-23	Sampdoria	Catania	8.0	8.0	4.0	4.0	1.0	1.0	...	1.0	0	4.0	11.0	13.0	29.0	2.0	4.0	0.0	1.0
3	2009.0	2009-08-23	Roma	Genoa	11.0	14.0	6.0	7.0	0.0	0.0	...	3.0	3	5.0	8.0	13.0	20.0	2.0	5.0	0.0	0.0
4	2009.0	2009-08-23	Bari	Inter	13.0	20.0	4.0	3.0	0.0	0.0	...	1.0	1	1.0	10.0	14.0	20.0	2.0	3.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3792	2019.0	2019-05-26	Empoli	Inter	9.0	20.0	5.0	15.0	0.0	0.0	...	2.0	3	2.0	8.0	9.0	11.0	2.0	4.0	1.0	1.0
3793	2019.0	2019-05-26	Parma	Roma	9.0	16.0	5.0	8.0	0.0	1.0	...	2.0	3	8.0	13.0	8.0	13.0	1.0	2.0	0.0	0.0
3794	2019.0	2019-05-26	Juventus	Sampdoria	6.0	10.0	1.0	3.0	0.0	0.0	...	2.0	3	6.0	7.0	12.0	6.0	2.0	0.0	0.0	0.0
3795	2019.0	2019-05-26	Milan	Spal	16.0	7.0	8.0	4.0	2.0	1.0	...	2.0	0	4.0	8.0	13.0	17.0	2.0	2.0	0.0	0.0
3796	2019.0	2019-05-26	Lazio	Torino	9.0	9.0	4.0	7.0	0.0	0.0	...	3.0	3	5.0	5.0	10.0	9.0	1.0	0.0	0.0	0.0

3797 rows × 22 columns

Figura 2. Base de datos segundo vistazo

Vemos que nuestros datos se han reducido a 3797 filas y 22 columnas. Aquí se hizo la selección de las columnas más importantes (en mi opinión) las cuales cuentan con el número de goles de local y de visitante, la diferencia de goles, tiros de esquina de local y visitante, tiros a portería de local y visitante, etc.

III. Creación del modelo

III-A. *Datos de entrenamiento y prueba*

Antes de hacer nuestro modelo, necesitamos datos de entrenamiento y datos para hacer nuestra prueba del modelo. Para esto haremos uso de la biblioteca sklearn y su función para hacer esta separación de los datos

```
from sklearn.model_selection import train_test_split
from sklearn import preprocessing

y = df["FTR"]
X = df.drop(["Season", "Date", "HomeTeam", "AwayTeam", "FTR"], axis=1)
print(len(X))

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
scaler = preprocessing.StandardScaler().fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

Figura 3. Creación datos de entrenamiento y prueba

III-B. *El Modelo*

Nuestro modelo será un modelo de clasificación Random Forest. Este tipo de modelo consiste en varios árboles de decisión. Se utiliza la aleatoreidad en conjunto con las predicciones de los demás árboles para mejorar el resultado que cualquier árbol de decisión individual podría brindarte. En nuestra primer corrida realizaremos un modelo simple para el cual haremos una búsqueda de los mejores hyperparámetros más adelante.

```
rand_forest = RandomForestClassifier()
rand_forest.fit(X_train, y_train)
y_pred = rand_forest.predict(X_test)
```

Figura 4. Creación del modelo

III-C. *Resultados*

Una vez implementado el modelo podemos ver los resultados:

```
Accuracy:  0.9921052631578947
Peor Accuracy:  0.2841717145114564
Cohens Score (aleatoreidad):  0.9876819678596609
```

Figura 5. Resultados del modelo.

Vemos que la precisión del modelo es muy buena. La peor precisión que este ha presentado es 0.28. El score de Cohen nos habla sobre la aleatoreidad de los datos y su confianza, entre más cerca esté el número a 1 sabemos que hay más confianza en los resultados.

IV. Mejora del modelo y pruebas

IV-A. *Hyperparámetros*

Para mejorar nuestro modelo utilice la función de GridSearchCV, la cual es parte de Sklearn. Esta función buscará la mejor selección de parámetros de un grupo de valores de prueba que se le da. Nos ahorra la prueba y error y automatiza el proceso. El método de implementación fue el siguiente:

```
from sklearn.metrics import make_scorer
from sklearn.model_selection import GridSearchCV

hyperparam_grid = {'n_estimators': [3, 100, 1000],
                   'max_features': [0.05, 0.5, 0.95],
                   'max_depth': [10, 50, 100, None]}

grid_scorer = make_scorer(cohen_kappa_score)
rand_forest = GridSearchCV(RandomForestClassifier(), hyperparam_grid, cv= 5,
```

Figura 6. Uso de GridsearchCV.

Una vez ejecutados los comandos, nuestro modelo nos arroja los mejores parámetros. Estos parámetros fueron los siguientes:

- max_depth: 100
- max_features : 0.95
- n_estimators: 100

IV-B. *Sesgo*

Para revisar el sesgo revisaremos la importancia de las variables para realizar las decisiones dentro de nuestro modelo.

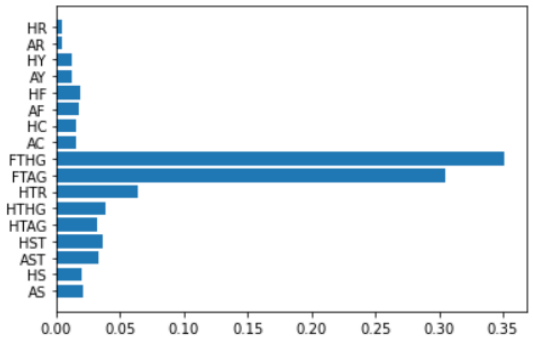

Figura 7. Resultados del modelo.

Vemos que existe menos sesgo para las variables de visitante. Estas parecen tener menor importancia para el modelo que las variables en casa.

V. Conclusión

Ahora imprimamos los resultados gráficos al imprimir ambas tablas, la tabla real de la liga y la tabla a partir de las predicciones:

	HomeTeam	FTR	FTR_Pred
0	Juventus	73	80
1	Napoli	68	81
2	Milan	61	78
3	Torino	57	70
4	Roma	54	66
5	Inter	53	74
6	Lazio	50	70
7	Atalanta	50	69
8	Sampdoria	50	66
9	Bologna	44	61
10	Genoa	42	66
11	Cagliari	41	64
12	Fiorentina	38	64
13	Sassuolo	36	67
14	Spal	35	62
15	Udinese	34	54
16	Empoli	30	39
17	Chievo	25	51
18	Parma	22	37
19	Crotone	13	22
20	Benevento	13	15
21	Verona	9	12
22	Frosinone	9	24

	HomeTeam	FTR	FTR_Pred
0	Napoli	68	81
1	Juventus	73	80
2	Milan	61	78
3	Inter	53	74
4	Torino	57	70
5	Lazio	50	70
6	Atalanta	50	69
7	Sassuolo	36	67
8	Roma	54	66
9	Sampdoria	50	66
10	Genoa	42	66
11	Fiorentina	38	64
12	Cagliari	41	64
13	Spal	35	62
14	Bologna	44	61
15	Udinese	34	54
16	Chievo	25	51
17	Empoli	30	39
18	Parma	22	37
19	Frosinone	9	24
20	Crotone	13	22
21	Benevento	13	15
22	Verona	9	12

Figura 8. Resultados del modelo.

Del lado derecho de la tabla tenemos los resultados de las predicciones, mientras que del lado izquierdo vemos los resultados reales en un determinado año. Vemos que la tabla ha cambiado un poco de lugar en algunas posiciones; sin embargo, los primeros 10 lugares se mantienen de entre los mejores 10 en las predicciones. Esto nos enseña que a pesar de no tener una predicción perfecta de las posiciones, nuestro modelo ha logrado hacer una clasificación lo suficientemente buena como para darnos una idea.

El fútbol, como todos los deportes, es un juego cambiante año con año. Las variables cambian cada temporada, se realizan nuevos fichajes, se cambian estrategias, incluso los equipos pueden no ser los mismos. El fin de este análisis era explorar si es posible hacer una clasificación a partir de datos históricos. Vemos que es posible hacer la clasificación que nos de una idea del resultado de la liga en un determinado año, mientras se cuente con la información adecuada. Sin embargo, este tipo de predicciones no se recomienda hacer si se busca un resultado exacto.