PROJECTE FINAL

Anàlisi i predicció del rendiment en l'IRONMAN 70.3 mitjançant Machine Learning

Júlia Casadevall

ÍNDEX

1. Introducció

L'IRONMAN

- Consta de 3,86 km de natació, 180,25 km de ciclisme i 42,2 km de cursa.
- Es va establir a Hawaii l'any 1978
- Els tres esports (natació, ciclisme i córrer) es completen en una sessió, per tant, no hi ha descans.

L'**IRONMAN 70.3** és una variant que ha esdevingut molt popular.

- Es conegut com el "**half IRONMAN**" perquè es realitza a la meitat de la distància.
- Els participants completen **1,9 km de natació**, **90 km de ciclisme** i **21,1 km de cursa a peu.**

2. Objectiu General

Analitzar i predir mitjançant Machine Learning el temps final d'un atleta en una proba IRONMAN 7.3 a partir de les seves dades personals.

Objectius Específics

1. Conduir un Anàlisi Exploratori de Dades (EDA) per aprofundir en el comportament del conjunt de dades.

2. Emprar algoritmes de Machine Learning per establir models que descriguin la interacció entre les variables i anticipar el rendiment dels esportistes, avaluant la precisió i utilitat d'aquests models.

3. Explorar com les variables "Edat", "Gènere" i "Temps de Transició" poden contribuir en les prediccions del model.

3. Conjunt de Dades

- El dataset utilitzat és una versió pre-processada de les dades originals descarregades del lloc web oficial de l'IRONMAN (www.ironman.com).

- Link: https://www.kaggle.com/datasets/aiaiaidavid/ironman-703-race-data-between-2004-and-2020/data

- Inclou 840.075 registres d'Ironman 70.3 PRO i de triatletes recreatius (grups d'edat) entre 2004 i 2020, amb el seu gènere, país d'origen, grup d'edat (no per PRO) i ubicació i any de la competició.

- Consta de 840.075 files i 13 columnes

3. Conjunt de Dades: Estructura

Gender	AgeGroup	AgeBand	Country	CountryISO2	EventYear	EventLocation	SwimTime	Transition1Time	BikeTime	Transition2Time	RunTime	FinishTime
M	40-44	40	Andorra	AD	2019	IRONMAN 70.3 South American Championship Bueno...	1679	119	9107	95	5515	16514
M	45-49	45	Andorra	AD	2019	IRONMAN 70.3 South American Championship Bueno...	2070	177	9160	132	6070	17609
M	45-49	45	Andorra	AD	2020	IRONMAN 70.3 Bariloche	1667	161	9891	122	5190	17031
M	45-49	45	Andorra	AD	2019	IRONMAN 70.3 World Championship	1750	183	10363	160	5071	17527
M	40-44	40	Andorra	AD	2019	IRONMAN 70.3 World Championship	2063	182	10065	142	5556	18008

3. Conjunt de Dades: Variables

- **Gènere:** El gènere del participant amb els valors 'M' (Masculí) i 'F' (Dona).
- **AgeGroup:** Grup d'edat dels participants.
- **AgeBand:** Representació numèrica del grup d'edat. Per exemple, "40" correspon al grup d'edat "40-44".
- **País:** el país d'origen del participant.
- **CountryISO2:** codi estandarditzat de dues lletres per a cada país.
- **EventYear:** l'any en què va tenir lloc l'esdeveniment.
- **EventLocation:** La ubicació o el nom de l'esdeveniment.
- **SwimTime:** El temps que triga el participant a la part de natació de l'esdeveniment, mesurat en segons.
- **Transition1Time:** el temps trigat durant la primera transició entre la natació i el ciclisme, mesurat en segons.
- **BikeTime:** El temps que triga el participant a la part de ciclisme de l'esdeveniment, mesurat en segons.
- **Transition2Time:** el temps trigat durant la segona transició entre anar en bicicleta i córrer, mesurat en segons.
- **Temps d'execució:** El temps que triga el participant a la part de carrera de l'esdeveniment, mesurat en alguna unitat de temps.
- **FinishTime:** El temps total que triga el participant a acabar tot l'esdeveniment, des de l'inici de la natació fins al final de la carrera, mesurat en segons.

4. Tractament de Dades

- Eliminar columnes innecessàries `df = df.drop('CountryISO2', axis=1)`

	Gender	AgeGroup	AgeBand	Country	EventYear	EventLocation	SwimTime	Transition1Time	BikeTime	Transition2Time	RunTime	FinishTime
0	M	40-44	40	Andorra	2019	IRONMAN 70.3 South American Championship Bueno...	1679	119	9107	95	5515	16514
1	M	45-49	45	Andorra	2019	IRONMAN 70.3 South American Championship Bueno...	2070	177	9160	132	6070	17609
2	M	45-49	45	Andorra	2020	IRONMAN 70.3 Bariloche	1667	161	9891	122	5190	17031
3	M	45-49	45	Andorra	2019	IRONMAN 70.3 World Championship	1750	183	10363	160	5071	17527
4	M	40-44	40	Andorra	2019	IRONMAN 70.3 World Championship	2063	182	10065	142	5556	18008

5. Exploratory Data Analysis (EDA)

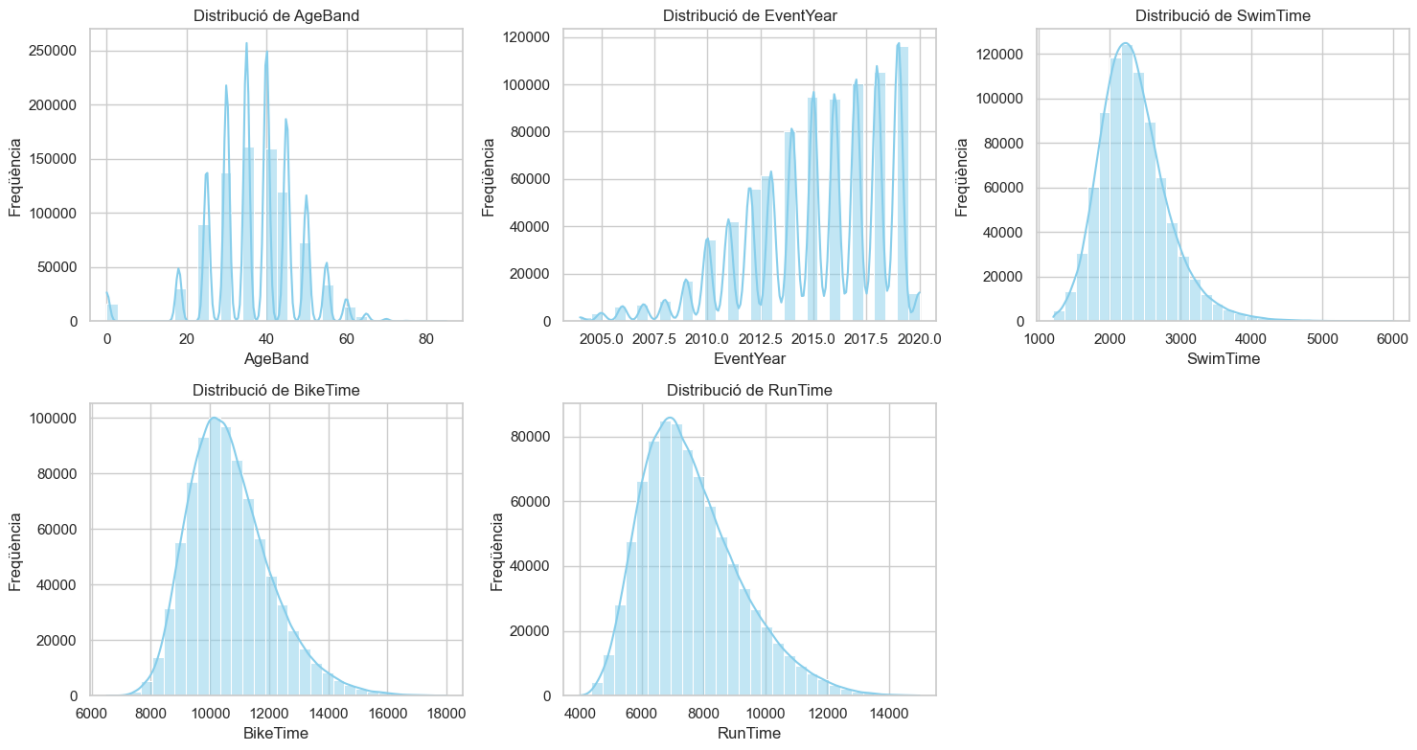
Analisis descriptiu

```
df.describe(include='all').round(2)
```

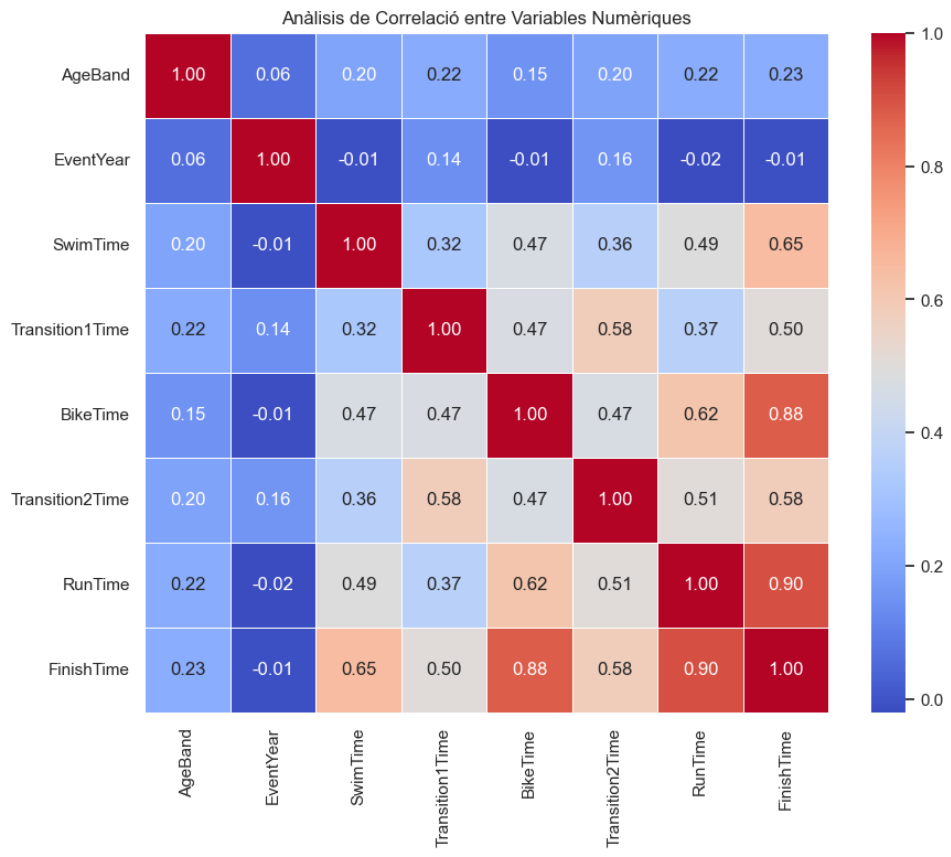
	Gender	AgeGroup	AgeBand	Country	EventYear	EventLocation	SwimTime	Transition1Time	BikeTime	Transition2Time	RunTime	FinishTime
count	840075	840075	840075.00	840075	840075.00	840075	840075.00	840075.00	840075.00	840075.00	840075.00	840075.00
unique	2	15	NaN	240	NaN	195	NaN	NaN	NaN	NaN	NaN	NaN
top	M	35-39	NaN	United States	NaN	IRONMAN 70.3 World Championship	NaN	NaN	NaN	NaN	NaN	NaN
freq	635680	160918	NaN	332037	NaN	26123	NaN	NaN	NaN	NaN	NaN	NaN
mean	NaN	NaN	36.90	NaN	2015.11	NaN	2340.25	286.68	10675.31	219.79	7606.97	21129.01
std	NaN	NaN	10.84	NaN	3.18	NaN	486.45	98.42	1397.95	94.24	1588.10	3076.23
min	NaN	NaN	0.00	NaN	2004.00	NaN	1201.00	46.00	6511.00	46.00	4002.00	13004.00
25%	NaN	NaN	30.00	NaN	2013.00	NaN	2010.00	211.00	9682.00	147.00	6440.00	18919.00
50%	NaN	NaN	35.00	NaN	2016.00	NaN	2286.00	280.00	10503.00	205.00	7362.00	20839.00
75%	NaN	NaN	45.00	NaN	2018.00	NaN	2604.00	359.00	11483.00	279.00	8544.00	23052.00
max	NaN	NaN	85.00	NaN	2020.00	NaN	5997.00	499.00	17993.00	499.00	14999.00	36529.00

AgeBand: 00 correspon a la categoria PRO. Inclou tots els grups d’edat.

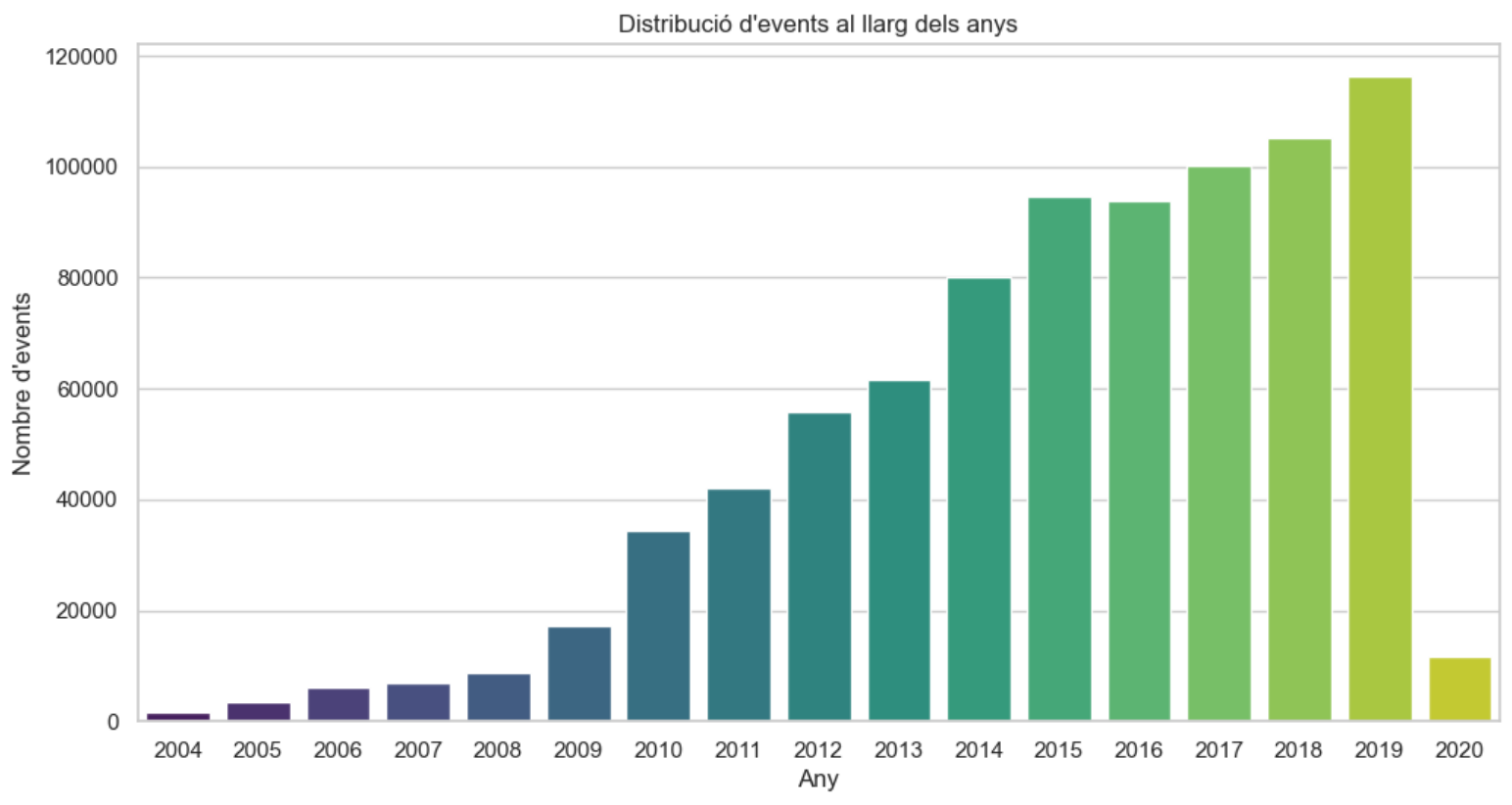
5. EDA: Distribució de les variables



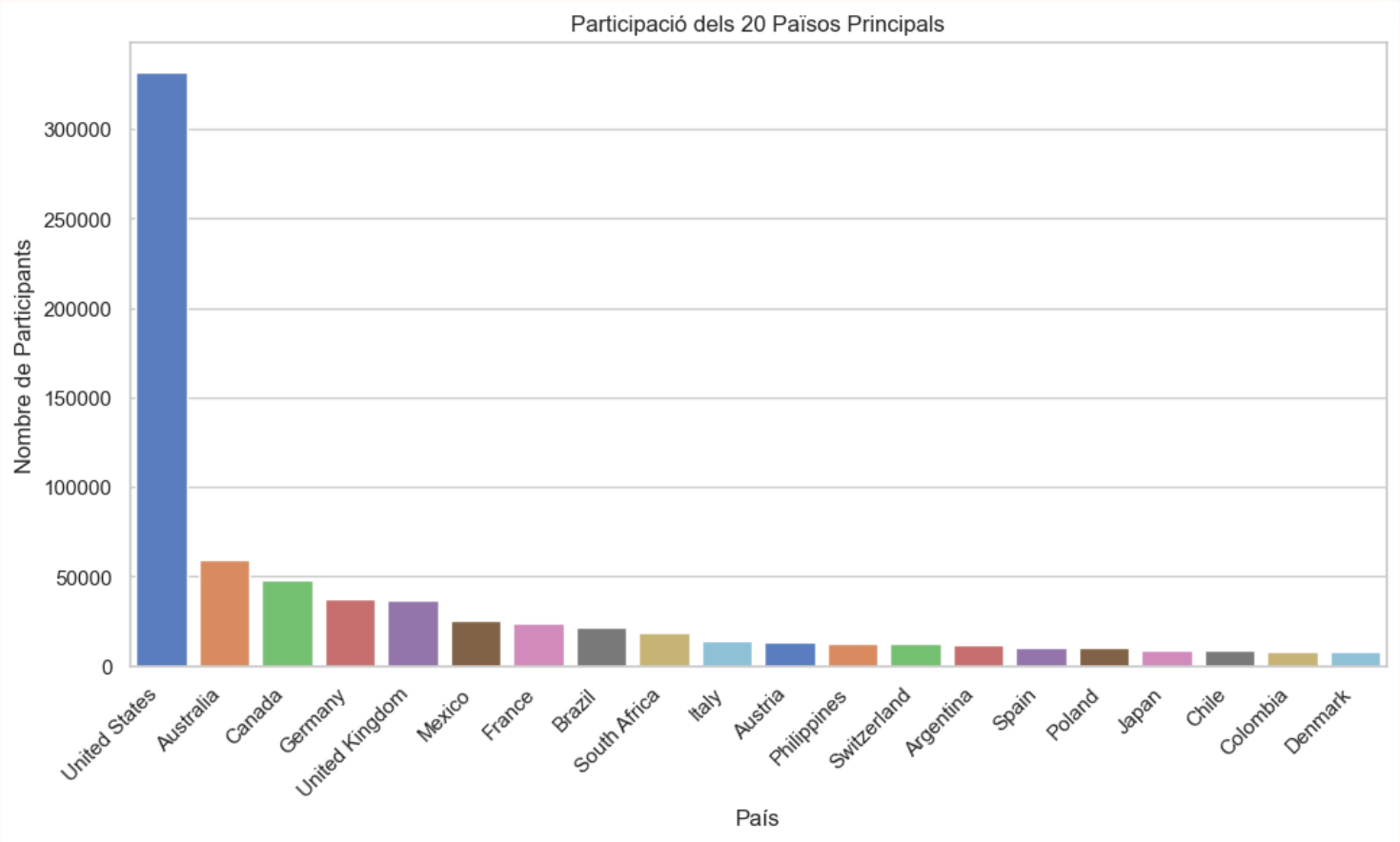
5. EDA: Correlació entre Variables



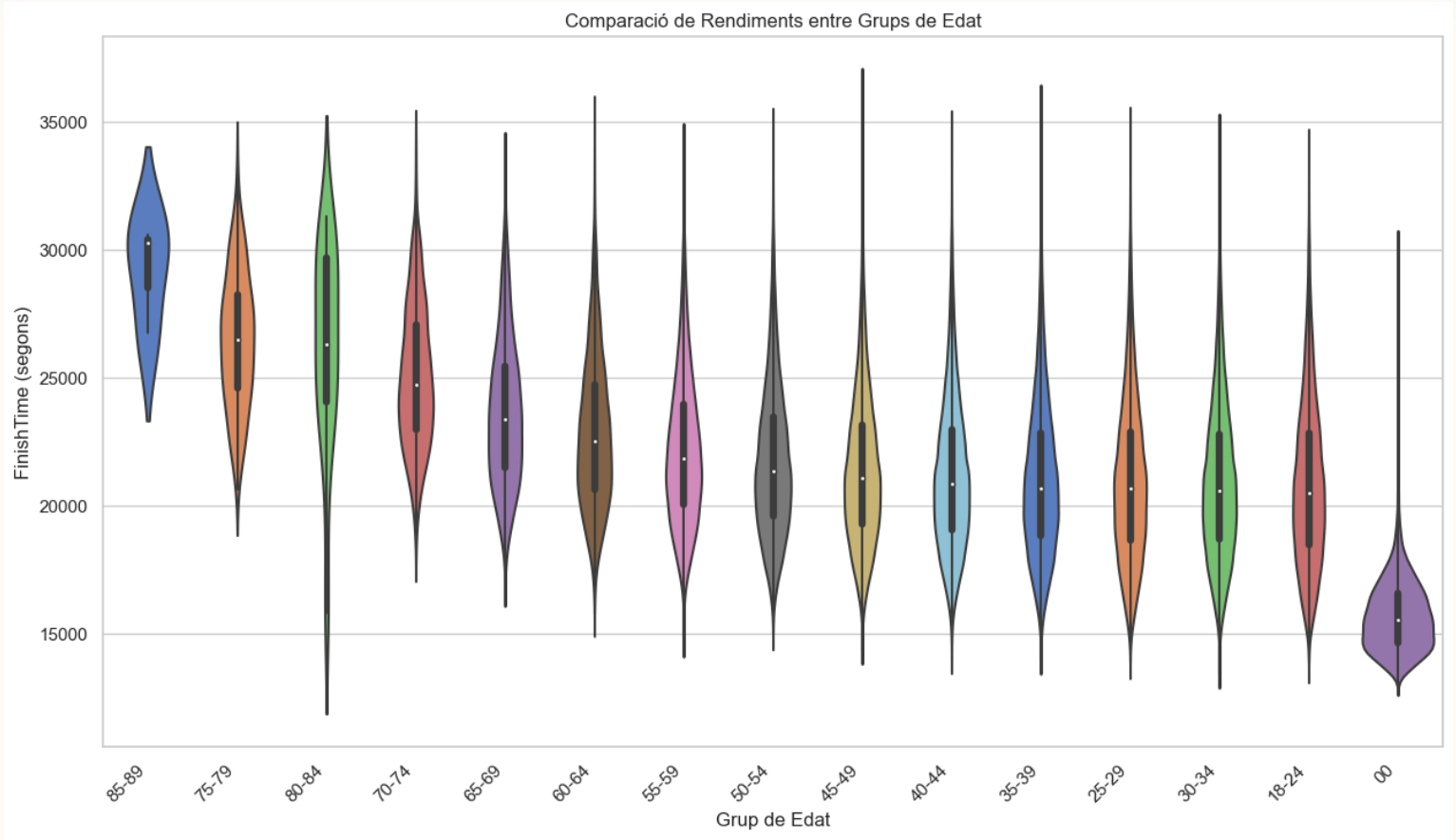
5. EDA: Anàlisi de Tendències Temporals



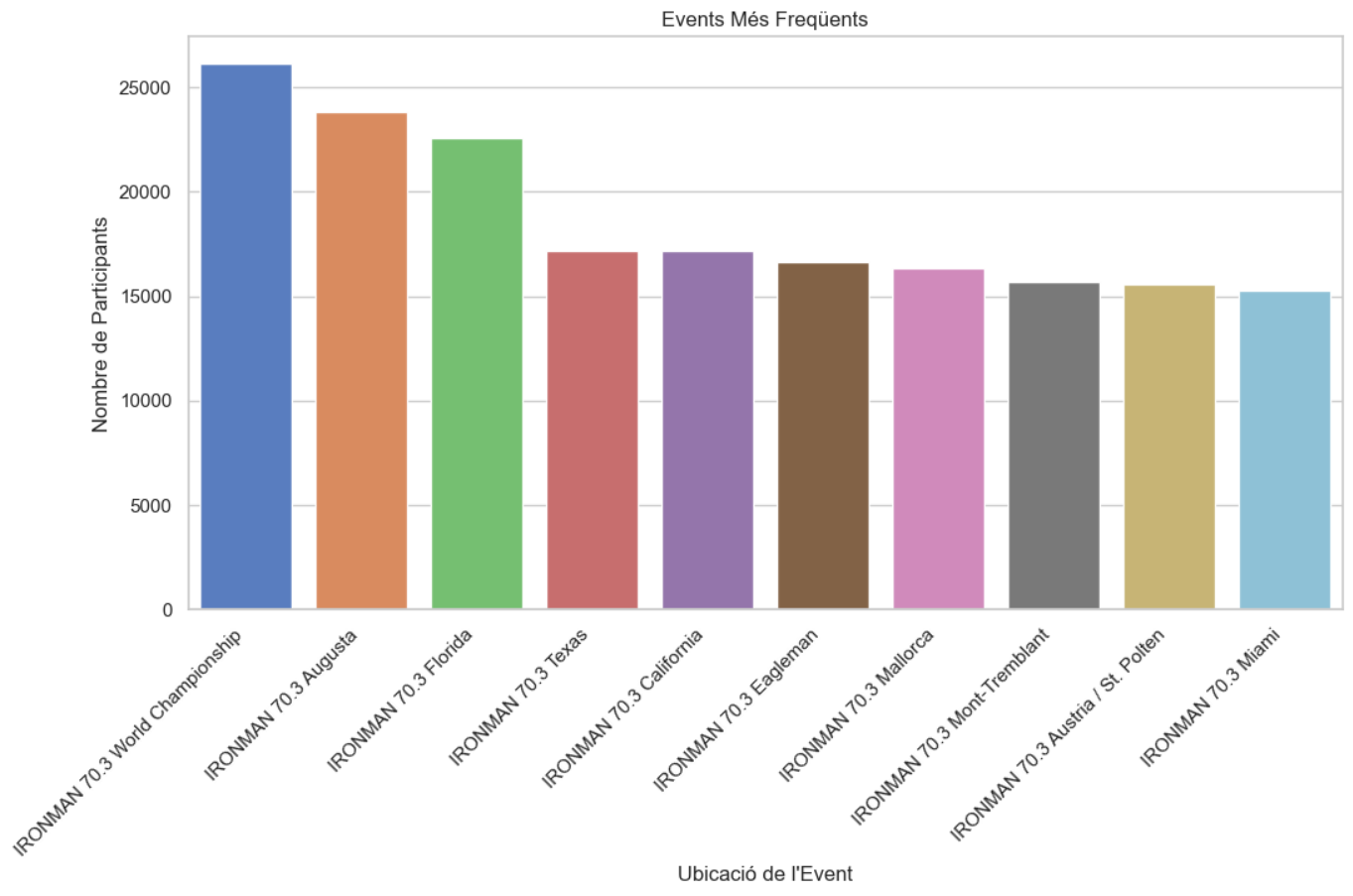
5. EDA: Comparació entre Països



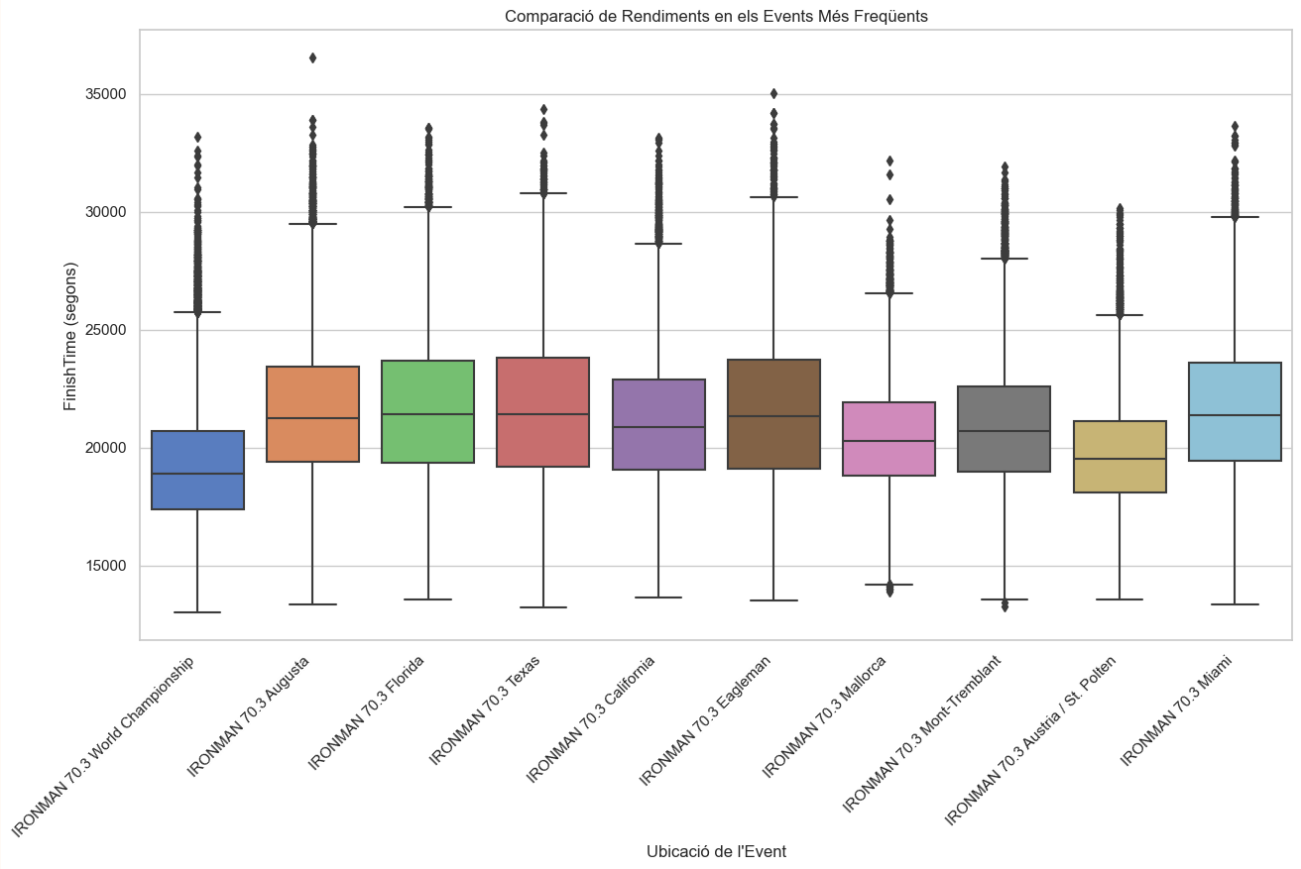
5. EDA: Rendiment per grups d'edat



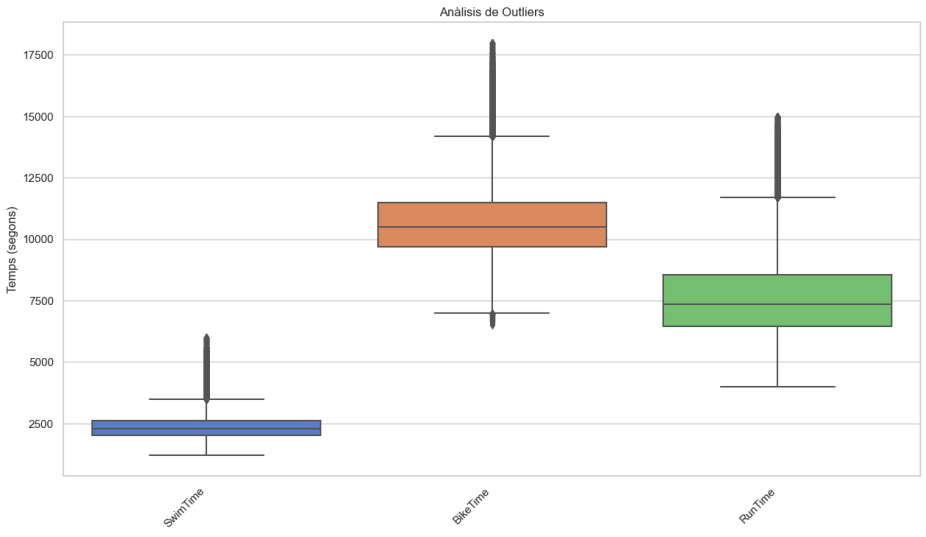
5. EDA: Events més freqüents

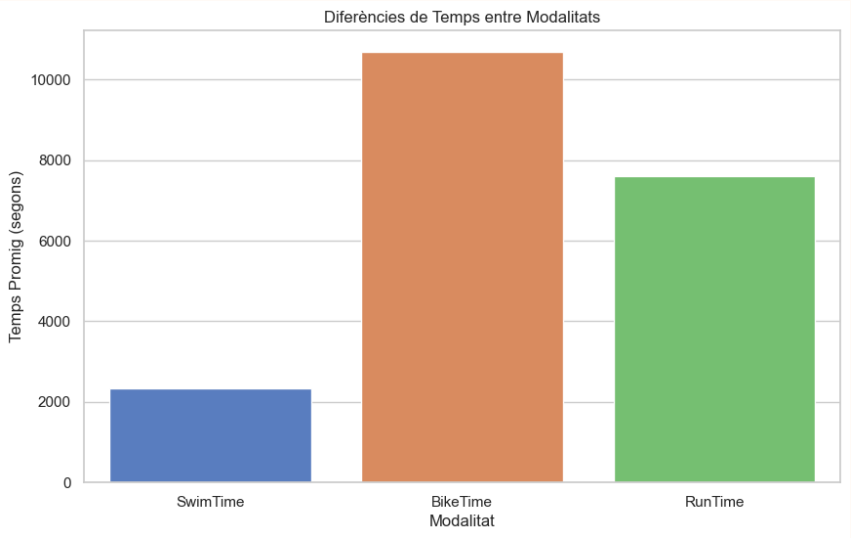


5. EDA: Rendiment i esdeveniments més freqüents

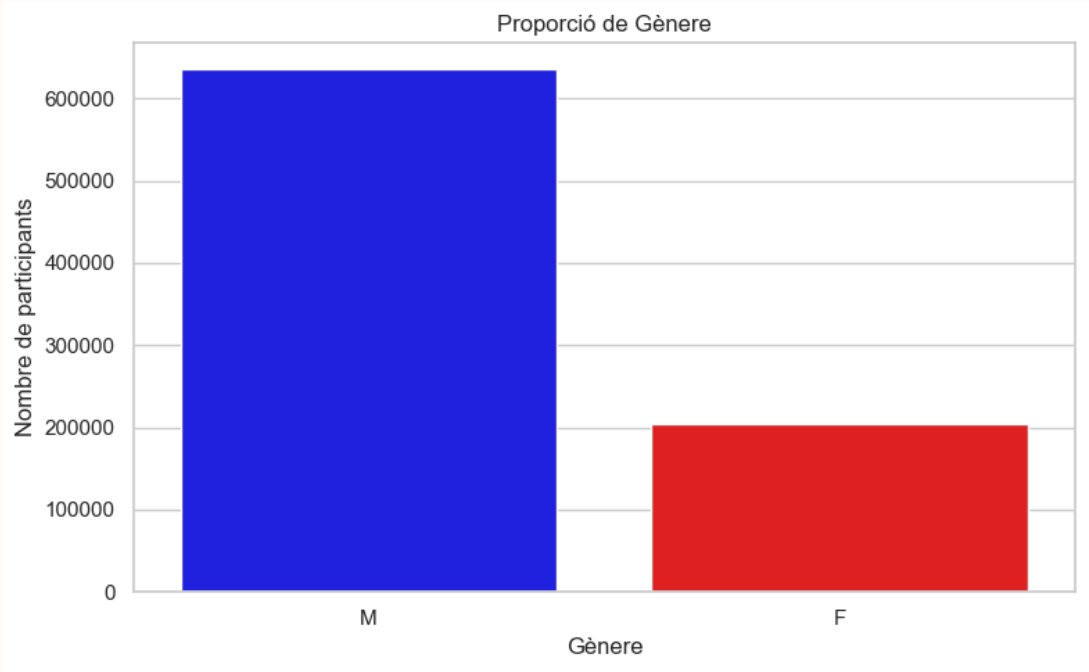


5. EDA: Diferències de temps entre modalitats

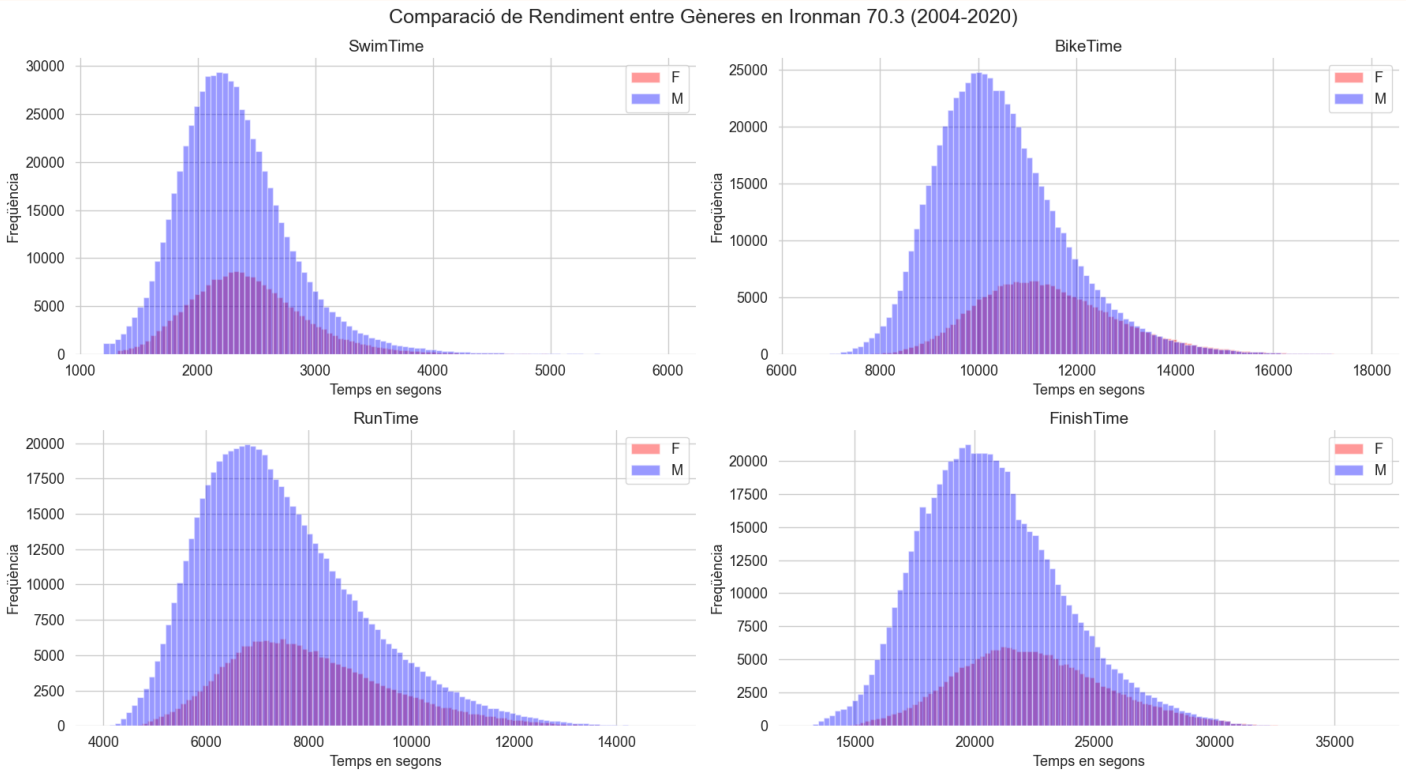




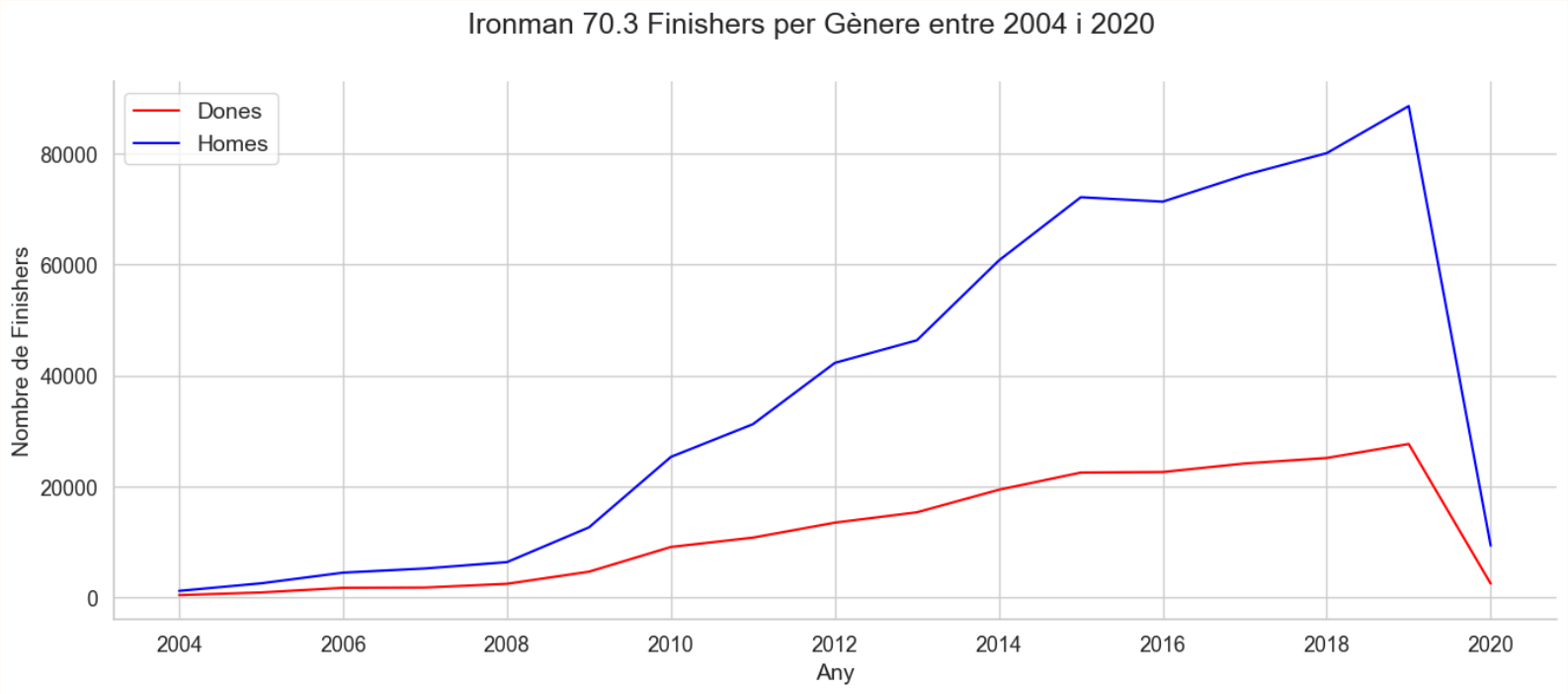
5. EDA: Proporció de Gènere



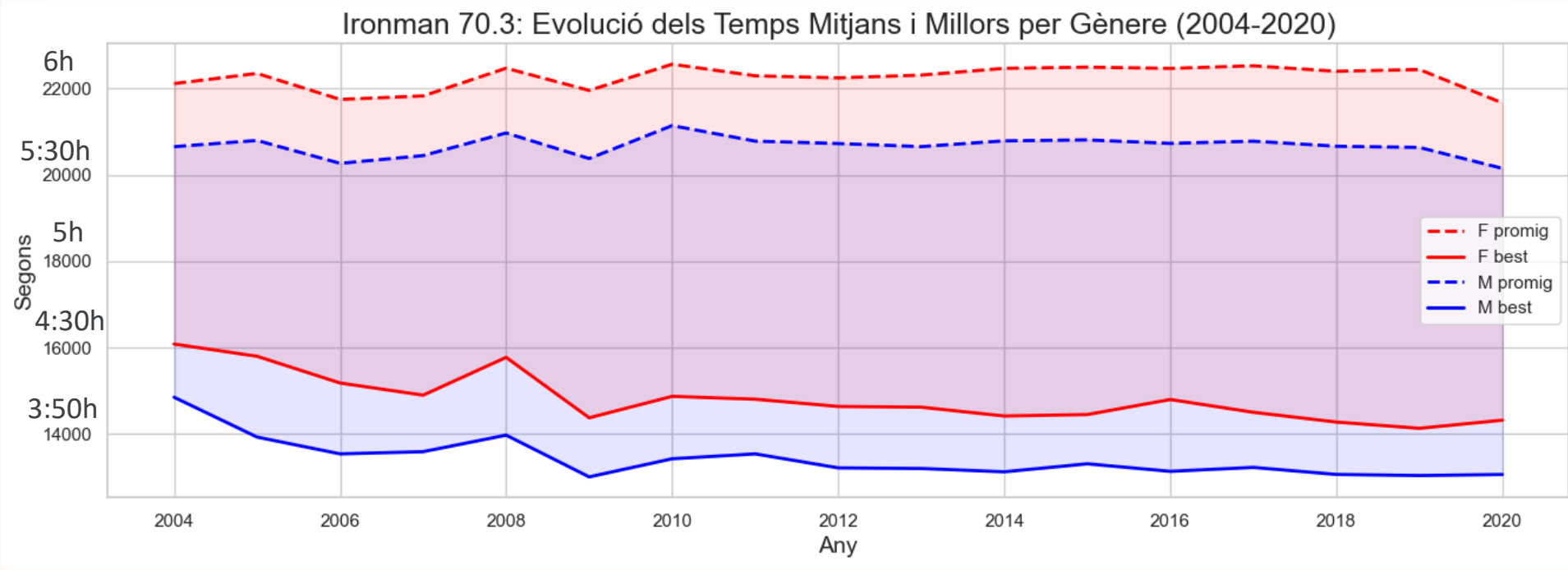
5. EDA: Rendiment en funció del Gènere



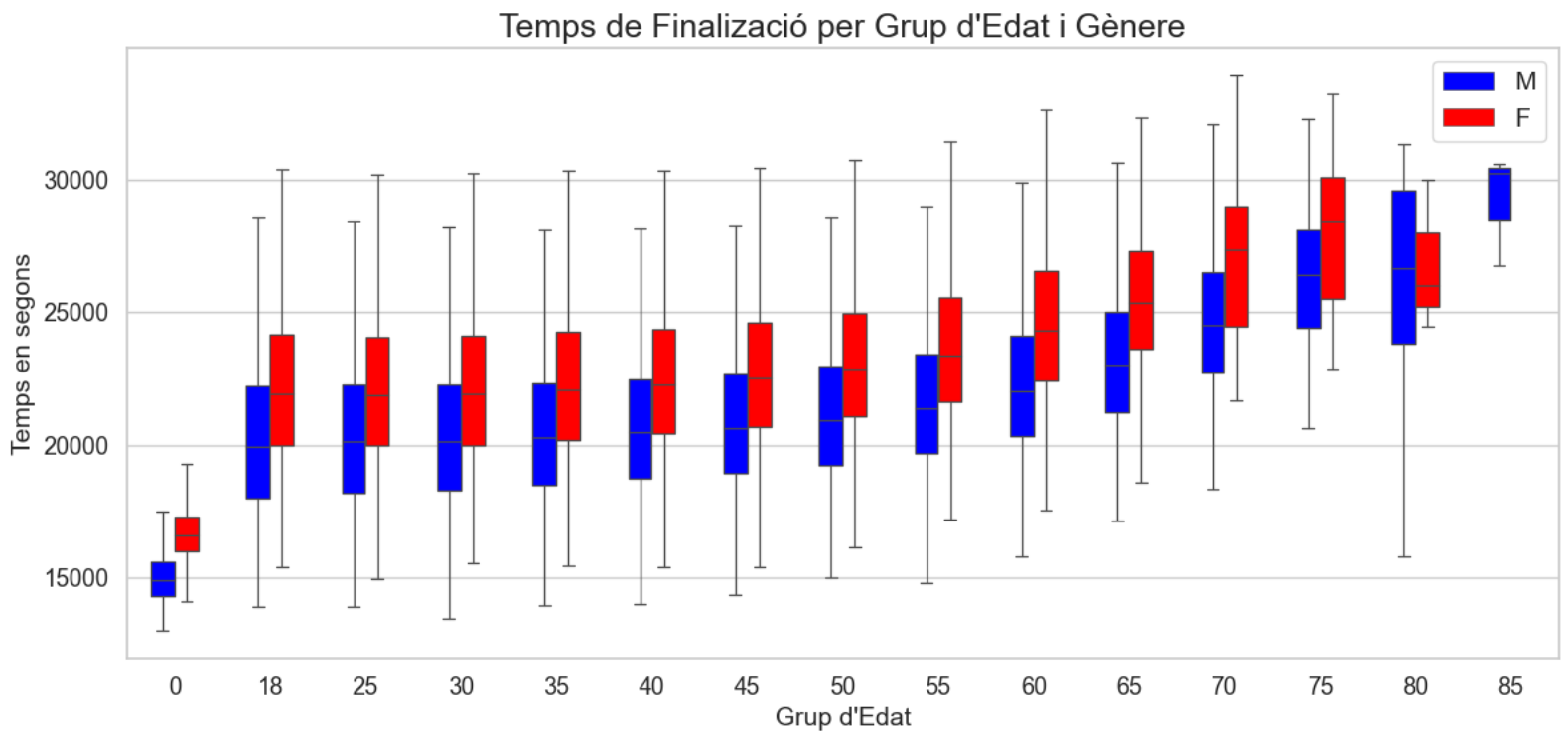
5. EDA: Rendiment en funció del Gènere



5. EDA: Rendiment en funció del Gènere



5. EDA: Rendiment en funció del Grup d'Edat i Gènere



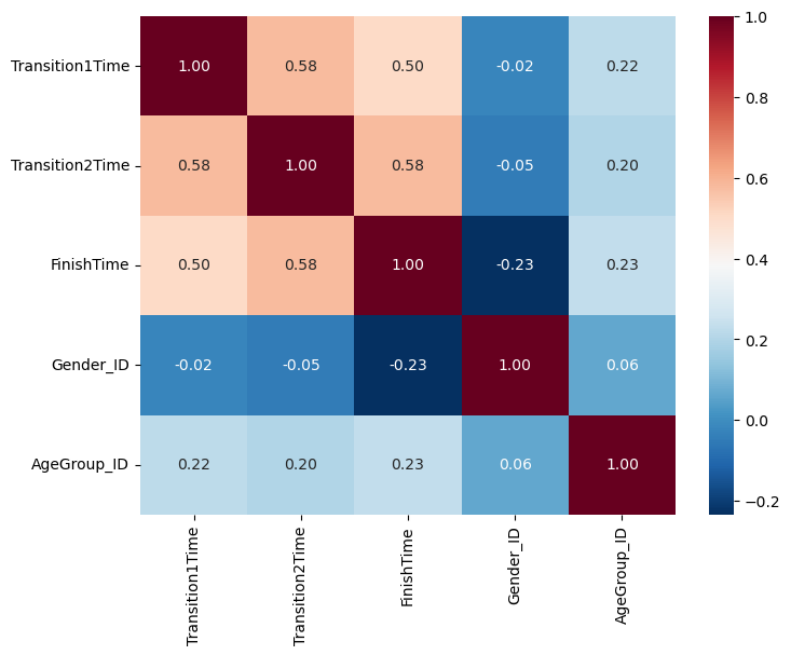
6. Entrenament i Avaluació de Models

```
columnas_a_eliminar = ['AgeBand','SwimTime','EventYear', 'BikeTime', 'RunTime','EventLocation', 'Country']
df_xg = df.drop(columnas_a_eliminar, axis=1)
```

```
from sklearn.preprocessing import LabelEncoder
labelEncoder = LabelEncoder()
df_xg['Gender_ID'] = labelEncoder.fit_transform(df_xg['Gender'])
```

```
def first2(s):
    return s[:2]

df_xg['AgeGroup_ID'] = df_xg['AgeGroup'].apply(first2).astype(int)
df_xg.sample(5)
```

	Gender	AgeGroup	Transition1Time	Transition2Time	FinishTime	Gender_ID	AgeGroup_ID
717415	F	35-39	221	151	18064	0	35
554201	M	35-39	367	129	24834	1	35
460674	M	30-34	222	222	19340	1	30
111016	M	40-44	272	103	17750	1	40
337986	F	45-49	492	295	26573	0	45



6. Entrenament i Avaluació de Models

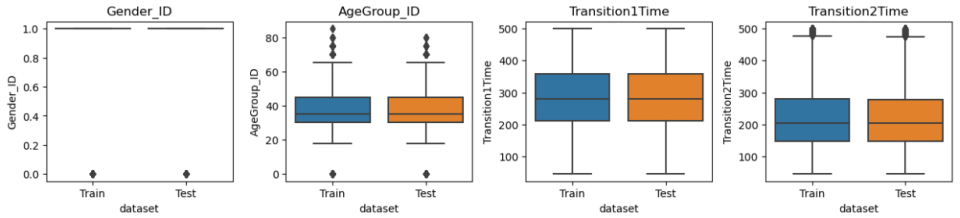
Divide data in train and test subsets

```
target = 'FinishTime'
predictors = ['Gender_ID', 'AgeGroup_ID', 'Transition1Time', 'Transition2Time' ]

X =  df_xg[predictors]
y = df_xg[target]
```

```
from sklearn.model_selection import train_test_split

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

# Show the results of the split
print("Training set has {} samples.".format(X_train.shape[0]))
print("Testing set has {} samples.".format(X_test.shape[0]))
```

```
Training set has 630056 samples.
Testing set has 210019 samples.
```

```
import matplotlib.pyplot as plt
import seaborn as sns
# Boxplots per comparar les distribucions:
plt.figure(figsize=(16, 8))
for i, feature in enumerate(X_train.columns):
    plt.subplot(3, 5, i + 1)
    sns.boxplot(x='dataset', y=feature, data=pd.concat([X_train.assign(dataset='Train'), X_test.assign(dataset='Test')]))
    plt.title(feature)

plt.tight_layout()
plt.show()
```



Test de U de Mann-Whitney

- **Hipòtesi Nul·la (H0):** La distribució de les dues mostres és la mateixa
- **Hipòtesi Alternativa (H1):** Hi ha diferències significatives entre les dues mostres

```
from scipy.stats import mannwhitneyu

for variable in X_train.columns:
    statistic, p_value = mannwhitneyu(X_train[variable], X_test[variable])

    print(f"Variable: {variable}")
    print(f"Estadístic U: {statistic}")
    print(f"p-valor: {p_value}")

    if p_value < 0.05:
        print("Hi ha diferències significatives entre les dues mostres.\n")
    else:
        print("La distribució entre les dues mostres és la mateixa.\n")
```

```
Variable: Gender_ID
Estadístic U: 66071216892.0
p-valor: 0.2050508083460314
La distribució entre les dues mostres és la mateixa.

Variable: AgeGroup_ID
Estadístic U: 66223796159.0
p-valor: 0.5150267604187935
La distribució entre les dues mostres és la mateixa.

Variable: Transition1Time
Estadístic U: 66181727982.5
p-valor: 0.8365017032815301
La distribució entre les dues mostres és la mateixa.

Variable: Transition2Time
Estadístic U: 66241239997.0
p-valor: 0.4095422372545702
La distribució entre les dues mostres és la mateixa.
```

6. Entrenament i Avaluació de Models

Totes les variables, menys Transition1Time, tenen outliers.

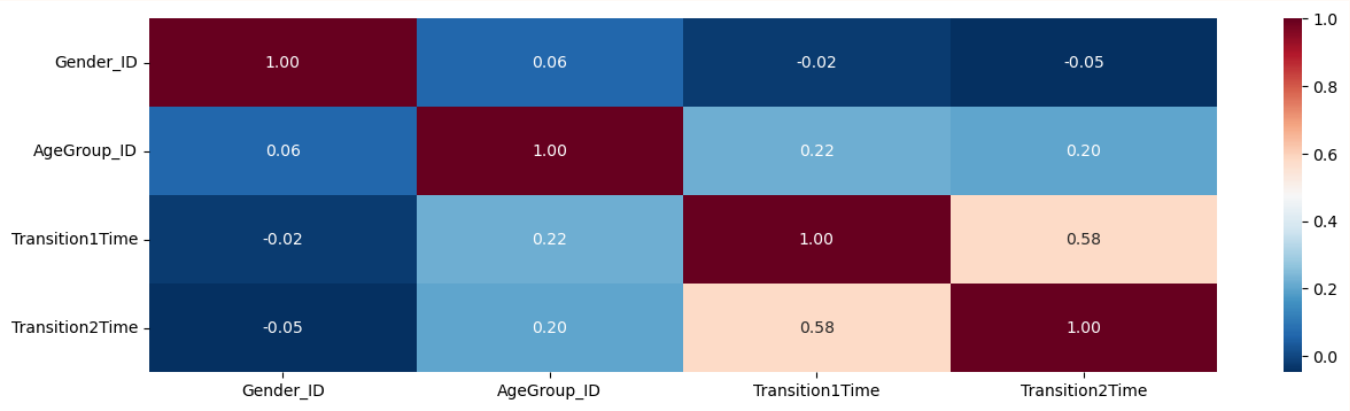
Un cop sé quines variables tenen outliers i quines no, faré el Robust Scale a totes les variables del training set amb outliers (Gender, AgeGroup, Transition2Time). La transformació es fa en el training set perque representa que el test set no el conec. Pero abans faig una copia per tenir els valors de X_train originals (sense transformacions).

```
X_train_transformat = X_train.copy()
```

```
from sklearn import preprocessing
X_train_transformat[["Gender_ID","Transition2Time","AgeGroup_ID"]]=preprocessing.RobustScaler().fit_transform(X_train_transformat
```

Després d'elminar els outliers amb l'ús del robust scaler,transformaré la variable restant amb el MinMaxScaler.

```
X_train_transformat[['Transition1Time']]=preprocessing.MinMaxScaler().fit_transform(X_train_transformat[['Transition1Time']])
X_train_transformat.head(10)
```



6. Entrenament i Avaluació de Models

Linear Regression

```
]: from sklearn.linear_model import LinearRegression
   #Creo un model de regressió de Regressio lineal
   RL = LinearRegression()
   #entreno el model
   RL.fit(X_train, y_train)
   #faig les prediccions del conjunt de proba (x_test)
   lr_y_test_pred = RL.predict(X_test)
```

```
]: from sklearn.metrics import r2_score
   r2_rl = r2_score(y_test, lr_y_test_pred)
   print('El coeficient de determinació és:', r2_rl)
```

```
El coeficient de determinació és: 0.43003719642766847
```

```
]: import matplotlib.pyplot as plt

   plt.scatter(y_test, lr_y_test_pred, color='Blue')
   plt.plot(y_test, y_test, color='red', linestyle='--')
   plt.xlabel('Valors reals (y_test)')
   plt.ylabel('Prediccions (lr_y_test_pred)')
   plt.title('Regressió Lineal')

   plt.text(plt.xlim()[0], plt.ylim()[1], f'$R^2={r2_rl:.2f}$', fontsize=12, verticalalignment='top', horizontalalignment='left')
```


Linear Regression

]: from sklearn.linear_model import LinearRegression
#Creo un model de regressió de Regressio lineal
RL = LinearRegression()
#entreno el model
RL.fit(X_train, y_train)
#faig les prediccions del conjunt de proba (x_test)
lr_y_test_pred = RL.predict(X_test)

]: from sklearn.metrics import r2_score
r2_rl = r2_score(y_test, lr_y_test_pred)
print('El coeficient de determinació és:', r2_rl)

El coeficient de determinació és: 0.43003719642766847

]: import matplotlib.pyplot as plt

plt.scatter(y_test, lr_y_test_pred, color='Blue')
plt.plot(y_test, y_test, color='red', linestyle='--')
plt.xlabel('Valors reals (y_test)')
plt.ylabel('Prediccions (lr_y_test_pred)')
plt.title('Regressió Lineal')

plt.text(plt.xlim()[0], plt.ylim()[1], f'$R^2={r2_rl:.2f}$', fontsize=12, verticalalignment='top', horizontalalignment='left')

6. Entrenament i Avaluació de Models

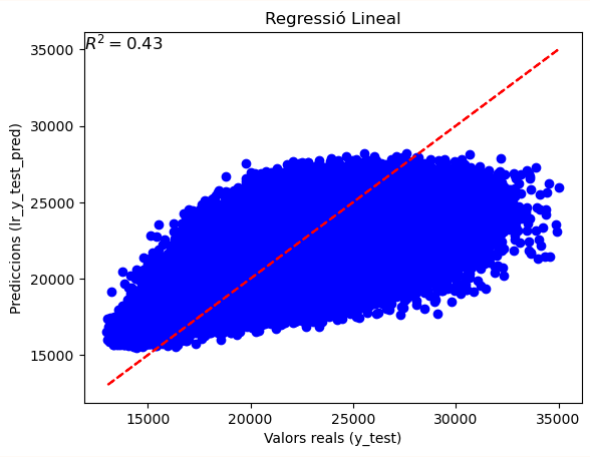
Linear Regression

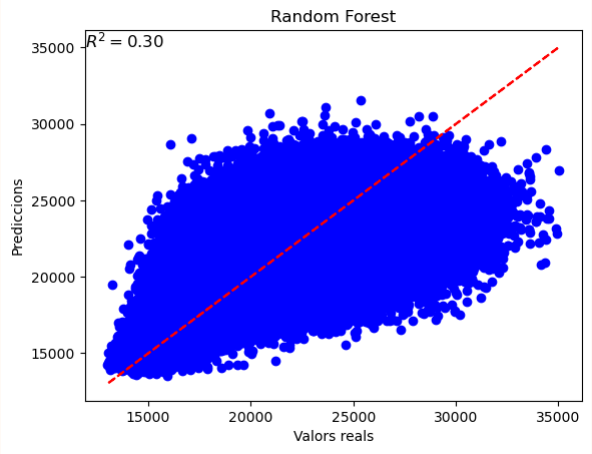
```
from sklearn.linear_model import LinearRegression
#Creo un model de regressió de Regressio Lineal
RL = LinearRegression()
#entreno el model
RL.fit(X_train, y_train)
#faig les prediccions del conjunt de proba (x_test)
lr_y_test_pred = RL.predict(X_test)
```

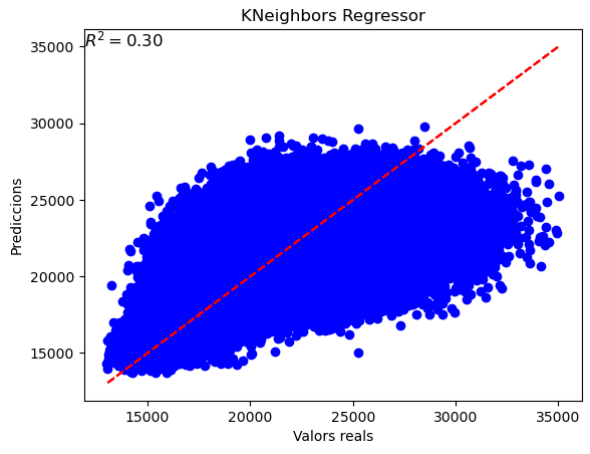
```
from sklearn.metrics import r2_score
r2_rl = r2_score(y_test, lr_y_test_pred)
print('El coeficient de determinació és:', r2_rl)
```

```
El coeficient de determinació és: 0.43003719642766847
```

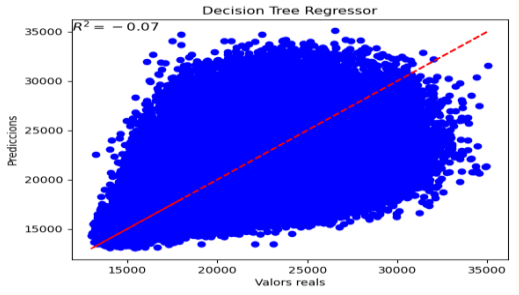
```
import matplotlib.pyplot as plt

plt.scatter(y_test, lr_y_test_pred, color='Blue')
plt.plot(y_test, y_test, color='red', linestyle='--')
plt.xlabel('Valors reals (y_test)')
plt.ylabel('Prediccions (lr_y_test_pred)')
plt.title('Regressió Lineal')

plt.text(plt.xlim()[0], plt.ylim()[1], f'$R^2={r2_rl:.2f}$', fontsize=12, verticalalignment='top', horizontalalignment='left')
```







Comparació de MSE



7. Millora del model i Avaluació

	Model	R2	MSE
0	Regressió Lineal	0.43	5375503.73
1	RandomForest	0.30	658193.95
2	KNR	0.30	6571962.17
3	DTR	-0.07	10091154.85

El millor model sembla ser la Regressió Lineal (RL) amb un Mean Squared Error (MSE) de 5375503.73 (en segons) i un Coeficient de Determinació (R^2) de 0.43. Per millorar encara més el seu rendiment, ajustaré els hiperparàmetres, és a dir, trobar la combinació òptima d'aquests.

Primer, hem de identificar quins hiperparametres del model es poden ajustar, especificar els intervals o els valors possibles per a cada un, entrenar el model amb diferents combinacions i avaluar el seu rendiment utilitzant mètriques rellevants.

```
# Hiperparametres
RL.get_params()
```

```
{'copy_X': True, 'fit_intercept': True, 'n_jobs': None, 'positive': False}
```

```
from sklearn.model_selection import GridSearchCV, cross_val_score
RL_parametres = {'fit_intercept': [True, False],'n_jobs': [None, 1]}

RL1 = LinearRegression()
#configuro el grid search amb el model Linear Regression i els parametres seleccionats
GS_RL = GridSearchCV(RL1, RL_parametres)

#Busqueda en quadricula utilitzant com a dades X i y per seleccionar la millor combinació de paràmetres
GS_RL.fit(X, y)
```

```
#Quins son els millors hiperparàmetres
print("Millors hiperparàmetres de Linear Regression:", GS_RL.best_params_)
```

```
Millors hiperparàmetres de Linear Regression: {'fit_intercept': True, 'n_jobs': None}
```

7. Millora del model i Avaluació

```
Mean Squared Error (en segons): 5375452.5
R2 score: 0.43
```

La millora no es significativa amb els millors hiperparametres.

8. Predicció Final

Cross Validation

```
from sklearn.model_selection import cross_val_score
```

```
Cross_Val_RL_GS = cross_val_score(RL_best_model,X.values, y.values.ravel())
Cross_Val_RL_GS
```

```
array([0.45592766, 0.29720638, 0.36472917, 0.43243767, 0.41666366])
```

El primer fold es el que te el rendiment mes alt, despres el segon es molt baix, els altres es mantenen en la mateixa linia,no tant baixos pero si que amb un rediment baix en context general.

```
# Seleccionar les columnes
selected_columns = ['Gender_ID','AgeGroup_ID', "Transition1Time","Transition2Time"]

# Crear nou DataFrame amb les columnes seleccionades
df_selected = df_xg[selected_columns]
df_selected
```

	Gender_ID	AgeGroup_ID	Transition1Time	Transition2Time
0	1	40	119	95
1	1	45	177	132
2	1	45	161	122
3	1	45	183	160
4	1	40	182	142
...	...	...	...	...
840070	1	50	261	160
840071	1	40	352	265
840072	0	30	357	332
840073	0	35	193	233
840074	0	30	244	273

Gènere: Home
Edat: 50 anys
Transició 1 en 4:13 min
Transició 2 en 3:18 min

(5h 43min i 36seg)

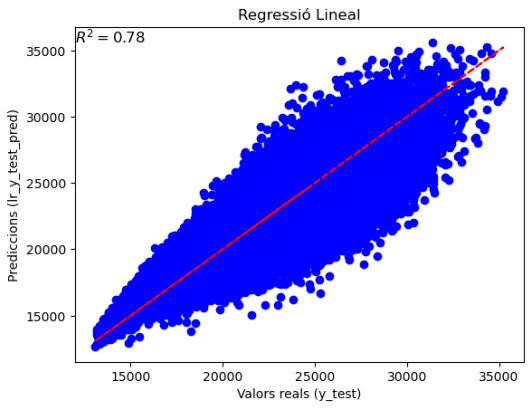
Segona Predicció

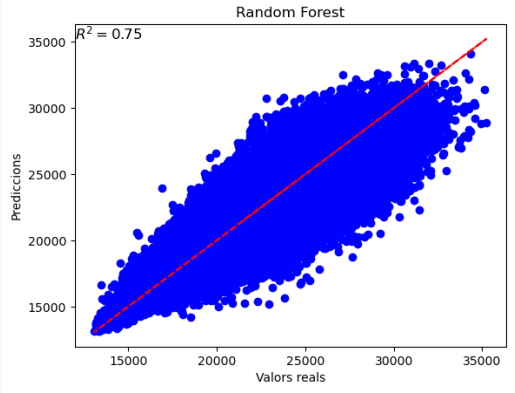
Gènere: Home
Edat: 20 anys
Transició 1 en 4:13 min
Transició 2 en 3:18 min

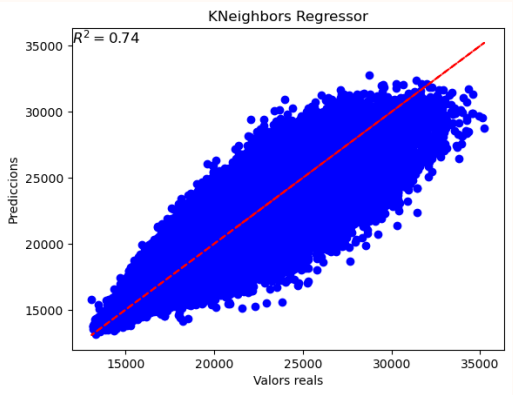
```
# Predicció
input_data = pd.DataFrame([['1', '50', '253', '198']])

pred = RL_best_model.predict(input_data)


print(f'The predicted Finish Time is {pred} segons.')
```

```
The predicted Finish Time is [20616.21512147] segons.
```

6. Creació d'un model més precís







Divide data in train and test subsets

```
target = 'FinishTime'
predictors = ['Gender_ID', 'BikeTime', 'AgeGroup_ID']

X =  df_xg[predictors]
y = df_xg[target]
```

	Model	R2	MSE
0	Regressió Lineal	0.78	2085404.45
1	RandomForest	0.75	2328078.32
2	KNR	0.74	2428209.88
3	DTR	0.74	2478626.00

6. Creació d'un model més precís

Mateix procediment amb el model de Regresssió Lineal:

1. GridSearch per mirar quin son els millors hiperparàmetres
2. Cross Validation per avaluar el rendiment general del model.
3. Predicció

```
El coeficient de determinació de Linear Regression és: 0.7780594371687855
```

Cross Validation

```
from sklearn.model_selection import cross_val_score
```

```
Cross_Val_RL_GS = cross_val_score(RL_best_model,X.values, y.values.ravel())
Cross_Val_RL_GS
```

```
array([0.75483383, 0.72783327, 0.78955819, 0.79732307, 0.80927274])
```

Els primers dos folds son els que presenten un menor rendiment en canvi els altres, tenen tenen més rendiment.

```
print(round(Cross_Val_RL_GS.mean(),3))
```

```
0.776
```

Home de 50 anys
Bike Time: 02:46:40h (32.40 km/h)

```
input_data = pd.DataFrame([['1', '10000', '50']])

pred = RL_best_model.predict(input_data)
# Convertir segons a timedelta
predicted_time_delta = timedelta(seconds=int(pred[0]))

# Formatejar la cadena al format hh:mm:ss
formatted_time = str(predicted_time_delta)

print(f'The predicted Finish Time is {formatted_time}.')
```

```
The predicted Finish Time is 5:37:19.
```

Dona de 20 anys
Bike Time: 02:13:40h (40 km/h)

```
input_data = pd.DataFrame([['0', '10000', '20']])
```

```
The predicted Finish Time is 5:22:07.
```

Home de 20 anys
Bike Time: 02:13:40h (40 km/h)

```
input_data = pd.DataFrame([['1', '8000', '20']])
```

```
The predicted Finish Time is 4:21:29.
```

9. Conclusions

1. El conjunt de dades **manca d'especificitat**, limitant-ne la predicció precisa del rendiment en les curses IRONMAN 70.3.
2. L**'edat i el gènere no són determinants clars del rendiment** en les competicions.
3. Importància de **recopilar dades més detallades** i rellevants per millorar la qualitat de les prediccions com ara:
    - Composició corporal (percentatge de greix, múscul i aigua)
    - Pes
    - VO2 max
    - Hores d'entrenament setmanals
    - Qualitat del descans i nutrició

Gràcies!