Annex 1

```
import pandas as pd
qfasei = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\qfaseini.xlsx')
qf= qfasei[qfasei['GRUP_CLASSE']!='CONV']
qf = qf.dropna()
print((len(qfasei)-len(qf))/len(qfasei))
qf = qf[qf['CODI_PROGRAMA']==752]
qf = qf.drop('CODI_PROGRAMA',1)
qf = qf.drop(['NOTA_NUM_DEF','NOTA_PROF'], 1)
qf.loc[qf['CODI_UPC_UD']==240011,'CODI_UPC_UD']= 'Algebra Lineal'
qf.loc[qf['CODI_UPC_UD']==240015,'CODI_UPC_UD']= 'Fonaments Informatica'
qf.loc[qf['CODI_UPC_UD']==240014,'CODI_UPC_UD']= 'Quimica I'
qf.loc[qf['CODI_UPC_UD']==240012,'CODI_UPC_UD']= 'Calcul I'
qf.loc[qf['CODI_UPC_UD']==240013,'CODI_UPC_UD']= 'Mecanica Fonamental'
qf.loc[qf['CODI_UPC_UD']==240021,'CODI_UPC_UD']= 'Geometria'
qf.loc[qf['CODI_UPC_UD']==240022,'CODI_UPC_UD']= 'Calcul II  '
qf.loc[qf['CODI_UPC_UD']==240024,'CODI_UPC_UD']= 'Qui-mica II'
qf.loc[qf['CODI_UPC_UD']==240023,'CODI_UPC_UD']= 'Termodinamica Fonamental'
qf.loc[qf['CODI_UPC_UD']==240025,'CODI_UPC_UD']= 'Expressio Grafica'
qf.loc[qf['SUPERA']=='S','SUPERA']= 1 #passem a binari, 1 si True 0 si False
qf.loc[qf['SUPERA']=='N','SUPERA']= 0
qf = qf.drop(['CREDITS','SUPERA'],1)
qf['CONVOCATORIA'] = qf.duplicated(['CODI_EXPEDIENT','CODI_UPC_UD'])

dadespreinscr = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\dadespersnomespreins.xlsx')
dadespreinscr                                                                    =
dadespreinscr.drop(['ANY_ACCES','SEXE','CP_FAMILIAR','TIPUS_ACCES','CP_CENTRE_
SEC'],1)
dadespreinscr = dadespreinscr.pivot_table(dadespreinscr, index = ['CODI_EXPEDIENT'])


Primeraconvocatoria = qf.loc[qf['CONVOCATORIA']==False]
Primeraconvocatoria = Primeraconvocatoria.drop('CONVOCATORIA', 1)
Primeraconvocatoria = Primeraconvocatoria.pivot_table(Primeraconvocatoria, index =
['CODI_EXPEDIENT','CODI_UPC_UD'])
Primeraconvocatoria = Primeraconvocatoria.drop(['CURS','QUAD'],1)
Primeraconvocatoria = Primeraconvocatoria.rename(columns = {'NOTA_NUM_AVAL' :
'NOTA_PRIMERA_CONVOCATORIA'})
Pc = Primeraconvocatoria.unstack(1)

qf['CONVOCATORIA2'] = qf.duplicated(['CODI_EXPEDIENT','CODI_UPC_UD'], keep = 'last')
Ultimaconvocatoria = qf.loc[qf['CONVOCATORIA2']==False]
Ultimaconvocatoria = Ultimaconvocatoria.drop(['CONVOCATORIA2','CONVOCATORIA'],1)
Ultimaconvocatoria = Ultimaconvocatoria.pivot_table(Ultimaconvocatoria, index =
['CODI_EXPEDIENT','CODI_UPC_UD'])
```

```
Ultimaconvocatoria = Ultimaconvocatoria.drop(['CURS','QUAD'],1)
Ultimaconvocatoria = Ultimaconvocatoria.rename(columns = {'NOTA_NUM_AVAL' : 'NOTA_ULTIMA_CONVOCATORIA'})
Uc = Ultimaconvocatoria.unstack(1)

qf1 = qf.loc[:,['CODI_EXPEDIENT','CODI_UPC_UD','CONVOCATORIA','NOTA_NUM_AVAL']]
qf1.loc[qf1['CONVOCATORIA'] == True, 'CONVOCATORIA'] = 1
qf1.loc[qf1['CONVOCATORIA'] == False, 'CONVOCATORIA'] = 0
qf1 = pd.pivot_table(qf1, index =['CODI_EXPEDIENT','CODI_UPC_UD',]) #fa mitjana de totes les convocatories
qf4 = qf1.copy()
qf4['CONVOCATORIA']= 1 / (1 - qf4['CONVOCATORIA'])
qf4 = qf4.rename(columns = {'NOTA_NUM_AVAL' : 'NOTA_MITJANA'})

notamitjana = qf4.unstack(1)

general = pd.merge(notamitjana, dadespreinscr, left_index=True, right_index=True)
general = pd.merge(general, Uc, left_index=True, right_index=True)
general = pd.merge(general, Pc, left_index=True, right_index=True)
general2 = general.dropna()

qfasenoi = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\qfasenoini.xlsx')
qfasenoi = qfasenoi[qfasenoi['CODI_PROGRAMA']==752] #sel·leccionem nomes els que estan fent enginyeria industrial
qfasenoi = qfasenoi.drop('CODI_PROGRAMA',1) #Tots són d'eng, ind. aquesta columna no aporta info.
qfasenoi = qfasenoi.drop(['NOTA_NUM_DEF','NOTA_PROF','CREDITS','CURS','QUAD','GRUP_CLASSE', 'NOTA_NUM_AVAL'], 1)
qfasenoi.loc[qfasenoi['CODI_UPC_UD']=='240031','CODI_UPC_UD']= 'Electromagnetisme'
qfasenoi.loc[qfasenoi['CODI_UPC_UD']=='240131','CODI_UPC_UD']= 'Equacions Diferencials'
qfasenoi.loc[qfasenoi['CODI_UPC_UD']=='240132','CODI_UPC_UD']= 'Informatica'
qfasenoi.loc[qfasenoi['CODI_UPC_UD']=='240033','CODI_UPC_UD']= 'Materials'
qfasenoi.loc[qfasenoi['CODI_UPC_UD']=='240133','CODI_UPC_UD']= 'Mecanica'
qfasenoi.loc[qfasenoi['CODI_UPC_UD']=='240032','CODI_UPC_UD']= 'Metodes Numerics'
qfasenoi.loc[qfasenoi['CODI_UPC_UD']=='240043','CODI_UPC_UD']= 'Dinamica de Sistemes'
qfasenoi.loc[qfasenoi['CODI_UPC_UD']=='240041','CODI_UPC_UD']= 'Economia i Empresa'
qfasenoi.loc[qfasenoi['CODI_UPC_UD']=='240042','CODI_UPC_UD']= 'Estadistica'
qfasenoi.loc[qfasenoi['CODI_UPC_UD']=='240044','CODI_UPC_UD']= 'Projecte I'
qfasenoi.loc[qfasenoi['CODI_UPC_UD']=='240141','CODI_UPC_UD']= 'Teoria de Maquines i Mecanismes'
qfasenoi.loc[qfasenoi['SUPERA']=='S','SUPERA']= 1 #passem a binari, 1 si True 0 si False
qfasenoi.loc[qfasenoi['SUPERA']=='N','SUPERA']= 0
qfasenoi = qfasenoi.dropna()
```

```
qfasenoiQ3 = qfasenoi.loc[qfasenoi['CODI_UPC_UD'].isin(['Electromagnetisme', 'Equacions Diferencials','Informatica','Materials','Mecanica','Metodes Numerics'])]
qfasenoiQ3['CONVOCATORIA'] = qfasenoiQ3.duplicated(['CODI_EXPEDIENT','CODI_UPC_UD'])
qfasenoiQ3_1conv = qfasenoiQ3.loc[qfasenoiQ3['CONVOCATORIA']==False]['CODI_EXPEDIENT','CODI_UPC_UD'])
qfasenoiQ3_1conv = qfasenoiQ3_1conv.unstack(1)

generalperpredirQ3 = pd.merge(general2, qfasenoiQ3_1conv, left_index=True, right_index=True)
generalperpredirQ3 = generalperpredirQ3.dropna()
generalperpredirQ3.to_csv('generalperpredirQ3')
generalperpredirQ3.to_excel('generalperpredirQ3.xlsx')
```

Annex 2

Electromagnetisme

Max_Depth

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,41:42]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)


max_depths = np.linspace(1, 20, 20, endpoint=True)

R_electromagnetisme=[]
P_electromagnetisme=[]
F_electromagnetisme=[]
A_electromagnetisme=[]


for max_depth in max_depths:
   dt = DecisionTreeClassifier(max_depth=max_depth, random_state=1)
   dt.fit(X_train, y_train)
   y_pred = dt.predict(X_test)

   R_electromagnetisme.append(recall_score(y_test,y_pred, pos_label=0))
   P_electromagnetisme.append(precision_score(y_test,y_pred, pos_label=0))
   F_electromagnetisme.append(f1_score(y_test,y_pred, pos_label=0))

   A_electromagnetisme.append(accuracy_score(y_test,y_pred))
```

```
plt.plot(max_depths,R_electromagnetisme, 'r', label = 'R')
plt.plot(max_depths,P_electromagnetisme, 'b', label = 'P')
plt.plot(max_depths,F_electromagnetisme, 'y', label = 'F')
plt.plot(max_depths,A_electromagnetisme, 'k', label = 'A')
plt.legend()
plt.show()
```

```
Max_depth_electromagnetisme_millor_A = {max_depths[A_electromagnetisme.index(max(A_electromagnetisme))]:max(A_electromagnetisme)}
Max_depth_electromagnetisme_millor_F = {max_depths[F_electromagnetisme.index(max(F_electromagnetisme))]:max(F_electromagnetisme)}
Max_depth_electromagnetisme_millor_P = {max_depths[P_electromagnetisme.index(max(P_electromagnetisme))]:max(P_electromagnetisme)}
Max_depth_electromagnetisme_millor_R = {max_depths[R_electromagnetisme.index(max(R_electromagnetisme))]:max(R_electromagnetisme)}
```

Min_Sample_Split

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,41:42]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)


min_samples_splits = np.linspace(0.01, 0.6, 60, endpoint=True)

R_electromagnetisme=[]
P_electromagnetisme=[]
```

```
F_electromagnetisme=[]
A_electromagnetisme=[]


for min_samples_split in min_samples_splits:
  dt = DecisionTreeClassifier(min_samples_split=min_samples_split, random_state=1)
  dt.fit(X_train, y_train)
  y_pred = dt.predict(X_test)

  R_electromagnetisme.append(recall_score(y_test,y_pred, pos_label=0))
  P_electromagnetisme.append(precision_score(y_test,y_pred, pos_label=0))
  F_electromagnetisme.append(f1_score(y_test,y_pred, pos_label=0))

  A_electromagnetisme.append(accuracy_score(y_test,y_pred))



plt.plot(min_samples_splits,R_electromagnetisme, 'r', label = 'R')
plt.plot(min_samples_splits,P_electromagnetisme, 'b', label = 'P')
plt.plot(min_samples_splits,F_electromagnetisme, 'y', label = 'F')
plt.plot(min_samples_splits,A_electromagnetisme, 'k', label = 'A')
plt.legend()
plt.show()

min_samples_splits_electromagnetisme_millor_A =
{min_samples_splits[A_electromagnetisme.index(max(A_electromagnetisme))]:max(A_electr
omagnetisme)}
min_samples_splits_electromagnetisme_millor_F =
{min_samples_splits[F_electromagnetisme.index(max(F_electromagnetisme))]:max(F_electr
omagnetisme)}
min_samples_splits_electromagnetisme_millor_P =
{min_samples_splits[P_electromagnetisme.index(max(P_electromagnetisme))]:max(P_electr
omagnetisme)}
min_samples_splits_electromagnetisme_millor_R =
{min_samples_splits[R_electromagnetisme.index(max(R_electromagnetisme))]:max(R_elect
romagnetisme)}
```

Best Tree

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
from sklearn.model_selection import GridSearchCV


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,41:42]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)

parameters = {'max_depth' : [10,11,12,13,14,15], 'min_samples_split' : [0.02,0.04,0.06,0.08,0.1]}
clf_tree = DecisionTreeClassifier(random_state = 1)
clf=GridSearchCV(clf_tree,parameters,scoring = 'f1')
clf.fit(X_train, y_train)
clf.best_estimator_.fit(X_train, y_train)

y_pred = clf.best_estimator_.predict(X_test)

R_electromagnetisme = recall_score(y_test,y_pred, pos_label=0)
P_electromagnetisme = precision_score(y_test,y_pred, pos_label=0)
F_electromagnetisme = f1_score(y_test,y_pred, pos_label=0)


A_electromagnetisme = accuracy_score(y_test,y_pred)
```

Equació Diferencial

Max_Depth

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,41:]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)


max_depths = np.linspace(1, 30, 30, endpoint=True)

R_eqdif=[]
P_eqdif=[]
F_eqdif=[]
A_eqdif=[]


for max_depth in max_depths:
   dt = DecisionTreeClassifier(max_depth=max_depth, random_state=1)
   dt.fit(X_train, y_train)
   y_pred = dt.predict(X_test)

   R_eqdif.append(recall_score(y_test.iloc[:,1],y_pred[:,1], pos_label=0))
   P_eqdif.append(precision_score(y_test.iloc[:,1],y_pred[:,1], pos_label=0))
   F_eqdif.append(f1_score(y_test.iloc[:,1],y_pred[:,1], pos_label=0))

   A_eqdif.append(accuracy_score(y_test.iloc[:,1],y_pred[:,1]))



plt.plot(max_depths,R_eqdif, 'r', label = 'R')
plt.plot(max_depths,P_eqdif, 'b', label = 'P')
plt.plot(max_depths,F_eqdif, 'y', label = 'F')
```

```
plt.plot(max_depths,A_eqdif, 'k', label = 'A')
plt.legend()
plt.show()


Max_depth_eqdif_millor_A = {max_depths[A_eqdif.index(max(A_eqdif))]:max(A_eqdif)}
Max_depth_eqdif_millor_F = {max_depths[F_eqdif.index(max(F_eqdif))]:max(F_eqdif)}
Max_depth_eqdif_millor_P = {max_depths[P_eqdif.index(max(P_eqdif))]:max(P_eqdif)}
Max_depth_eqdif_millor_R = {max_depths[R_eqdif.index(max(R_eqdif))]:max(R_eqdif)}
```

Min_Sample_Split

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,42:43]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)


min_samples_splits = np.linspace(0.01, 0.6, 60, endpoint=True)

R_eqdif=[]
P_eqdif=[]
F_eqdif=[]
A_eqdif=[]


for min_samples_split in min_samples_splits:
  dt = DecisionTreeClassifier(min_samples_split=min_samples_split)
  dt.fit(X_train, y_train)
  y_pred = dt.predict(X_test)

  R_eqdif.append(recall_score(y_test,y_pred, pos_label=0))
  P_eqdif.append(precision_score(y_test,y_pred, pos_label=0))
```

```
    F_eqdif.append(f1_score(y_test,y_pred, pos_label=0))

    A_eqdif.append(accuracy_score(y_test,y_pred))

plt.plot(min_samples_splits,R_eqdif, 'r', label = 'R')
plt.plot(min_samples_splits,P_eqdif, 'b', label = 'P')
plt.plot(min_samples_splits,F_eqdif, 'y', label = 'F')
plt.plot(min_samples_splits,A_eqdif, 'k', label = 'A')
plt.legend()
plt.show()

min_samples_splits_eqdif_millor_A =
{min_samples_splits[A_eqdif.index(max(A_eqdif))]:max(A_eqdif)}
min_samples_splits_eqdif_millor_F =
{min_samples_splits[F_eqdif.index(max(F_eqdif))]:max(F_eqdif)}
min_samples_splits_eqdif_millor_P =
{min_samples_splits[P_eqdif.index(max(P_eqdif))]:max(P_eqdif)}
min_samples_splits_eqdif_millor_R =
{min_samples_splits[R_eqdif.index(max(R_eqdif))]:max(R_eqdif)}
```

Best Tree

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
from sklearn.model_selection import GridSearchCV


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:42]
y =qf.iloc[:,42:43]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)

parameters = {'max_depth' : [10,11,12,13,14,15], 'min_samples_split' :
[0.01,0.02,0.03,0.04,0.05]}
clf_tree = DecisionTreeClassifier(random_state = 1)
clf=GridSearchCV(clf_tree,parameters,scoring = 'f1')
clf.fit(X_train, y_train)
clf.best_estimator_.fit(X_train, y_train)
```

```
y_pred = clf.best_estimator_.predict(X_test)

R_eqdif = recall_score(y_test,y_pred, pos_label=0)
P_eqdif = precision_score(y_test,y_pred, pos_label=0)
F_eqdif = f1_score(y_test,y_pred, pos_label=0)


A_eqdif = accuracy_score(y_test,y_pred)
```

Informàtica

Max_Depth

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,43:44]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)


max_depths = np.linspace(1, 25, 25, endpoint=True)

R_info=[]
P_info=[]
F_info=[]
A_info=[]

for max_depth in max_depths:
   dt = DecisionTreeClassifier(max_depth=max_depth, random_state=1)
   dt.fit(X_train, y_train)
   y_pred = dt.predict(X_test)

   R_info.append(recall_score(y_test,y_pred, pos_label=0))
   P_info.append(precision_score(y_test,y_pred, pos_label=0))
   F_info.append(f1_score(y_test,y_pred, pos_label=0))

   A_info.append(accuracy_score(y_test,y_pred))



plt.plot(max_depths,R_info, 'r', label = 'R')
plt.plot(max_depths,P_info, 'b', label = 'P')
plt.plot(max_depths,F_info, 'y', label = 'F')
plt.plot(max_depths,A_info, 'k', label = 'A')
```

```
plt.legend()
plt.show()


Max_depth__millor_A = {max_depths[A_info.index(max(A_info))]:max(A_info)}
Max_depth_info_millor_F = {max_depths[F_info.index(max(F_info))]:max(F_info)}
Max_depth_info_millor_P = {max_depths[P_info.index(max(P_info))]:max(P_info)}
Max_depth_info_millor_R = {max_depths[R_info.index(max(R_info))]:max(R_info)}
```

Min_Sample_Split

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,43:44]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)


min_samples_splits = np.linspace(0.01, 0.18, 18, endpoint=True)

R_info=[]
P_info=[]
F_info=[]
A_info=[]


for min_samples_split in min_samples_splits:
  dt = DecisionTreeClassifier(min_samples_split=min_samples_split)
  dt.fit(X_train, y_train)
  y_pred = dt.predict(X_test)

  R_info.append(recall_score(y_test,y_pred, pos_label=0))
  P_info.append(precision_score(y_test,y_pred, pos_label=0))
```

```
    F_info.append(f1_score(y_test,y_pred, pos_label=0))

    A_info.append(accuracy_score(y_test,y_pred))



plt.plot(min_samples_splits,R_info, 'r', label = 'R')
plt.plot(min_samples_splits,P_info, 'b', label = 'P')
plt.plot(min_samples_splits,F_info, 'y', label = 'F')
plt.plot(min_samples_splits,A_info, 'k', label = 'A')
plt.legend()
plt.show()

min_samples_splits_info_millor_A =
{min_samples_splits[A_info.index(max(A_info))]:max(A_info)}
min_samples_splits_info_millor_F =
{min_samples_splits[F_info.index(max(F_info))]:max(F_info)}
min_samples_splits_info_millor_P =
{min_samples_splits[P_info.index(max(P_info))]:max(P_info)}
min_samples_splits_info_millor_R =
{min_samples_splits[R_info.index(max(R_info))]:max(R_info)}
```

Best Tree

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
from sklearn.model_selection import GridSearchCV


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,43:44]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)

parameters = {'max_depth' : [20,21,22,23,24,25], 'min_samples_split' :
[0.03,0.04,0.05,0.06,0.07,0.08]
}
clf_tree = DecisionTreeClassifier(random_state = 1)
clf=GridSearchCV(clf_tree,parameters,scoring = 'f1')
clf.fit(X_train, y_train)
```

```
clf.best_estimator_.fit(X_train, y_train)

y_pred = clf.best_estimator_.predict(X_test)

R_electromagnetisme = recall_score(y_test,y_pred, pos_label=0)
P_electromagnetisme = precision_score(y_test,y_pred, pos_label=0)
F_electromagnetisme = f1_score(y_test,y_pred, pos_label=0)


A_electromagnetisme = accuracy_score(y_test,y_pred)
```

Materials

Max_Depth

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,44:45]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)


max_depths = np.linspace(1, 30, 30, endpoint=True)

R_materials=[]
P_materials=[]
F_materials=[]
A_materials=[]


for max_depth in max_depths:
   dt = DecisionTreeClassifier(max_depth=max_depth, random_state=1)
   dt.fit(X_train, y_train)
```

```
    y_pred = dt.predict(X_test)

    R_materials.append(recall_score(y_test,y_pred, pos_label=0))
    P_materials.append(precision_score(y_test,y_pred, pos_label=0))
    F_materials.append(f1_score(y_test,y_pred, pos_label=0))

    A_materials.append(accuracy_score(y_test,y_pred))



plt.plot(max_depths,R_materials, 'r', label = 'R')
plt.plot(max_depths,P_materials, 'b', label = 'P')
plt.plot(max_depths,F_materials, 'y', label = 'F')
plt.plot(max_depths,A_materials, 'k', label = 'A')
plt.legend()
plt.show()


Max_depth_materials_millor_A                                                    =
{max_depths[A_materials.index(max(A_materials))]:max(A_materials)}
Max_depth_materials_millor_F                                                    =
{max_depths[F_materials.index(max(F_materials))]:max(F_materials)}
Max_depth_materials_millor_P                                                    =
{max_depths[P_materials.index(max(P_materials))]:max(P_materials)}
Max_depth_materials_millor_R                                                    =
{max_depths[R_materials.index(max(R_materials))]:max(R_materials)}
```

Min_Sample_Split

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,44:45]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)
```

```
min_samples_splits = np.linspace(0.01, 0.4, 10, endpoint=True)

R_materials=[]
P_materials=[]
F_materials=[]
A_materials=[]


for min_samples_split in min_samples_splits:
  dt = DecisionTreeClassifier(min_samples_split=min_samples_split)
  dt.fit(X_train, y_train)
  y_pred = dt.predict(X_test)

  R_materials.append(recall_score(y_test,y_pred, pos_label=0))
  P_materials.append(precision_score(y_test,y_pred, pos_label=0))
  F_materials.append(f1_score(y_test,y_pred, pos_label=0))

  A_materials.append(accuracy_score(y_test,y_pred))



plt.plot(min_samples_splits,R_materials, 'r', label = 'R')
plt.plot(min_samples_splits,P_materials, 'b', label = 'P')
plt.plot(min_samples_splits,F_materials, 'y', label = 'F')
plt.plot(min_samples_splits,A_materials, 'k', label = 'A')
plt.legend()
plt.show()

min_samples_splits_materials_millor_A =
{min_samples_splits[A_materials.index(max(A_materials))]:max(A_materials)}
min_samples_splits_materials_millor_F =
{min_samples_splits[F_materials.index(max(F_materials))]:max(F_materials)}
min_samples_splits_materials_millor_P =
{min_samples_splits[P_materials.index(max(P_materials))]:max(P_materials)}
min_samples_splits_materials_millor_R =
{min_samples_splits[R_materials.index(max(R_materials))]:max(R_materials)}
```

Best Tree

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
from sklearn.model_selection import GridSearchCV
```

```
qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,44:45]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)

parameters = {'max_depth' : [15,16,17,18,19,20], 'min_samples_split' : [0.01,0.02,0.03,0.04,0.05,0.06]}
clf_tree = DecisionTreeClassifier(random_state = 1)
clf=GridSearchCV(clf_tree,parameters,scoring = 'f1')
clf.fit(X_train, y_train)
clf.best_estimator_.fit(X_train, y_train)

y_pred = clf.best_estimator_.predict(X_test)

R_electromagnetisme = recall_score(y_test,y_pred, pos_label=0)
P_electromagnetisme = precision_score(y_test,y_pred, pos_label=0)
F_electromagnetisme = f1_score(y_test,y_pred, pos_label=0)


A_electromagnetisme = accuracy_score(y_test,y_pred)
```

Mecànica

Max_Depth

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,45:46]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)
```

```
max_depths = np.linspace(1, 30, 30, endpoint=True)

R_mecanica=[]
P_mecanica=[]
F_mecanica=[]
A_mecanica=[]


for max_depth in max_depths:
   dt = DecisionTreeClassifier(max_depth=max_depth, random_state=1)
   dt.fit(X_train, y_train)
   y_pred = dt.predict(X_test)

   R_mecanica.append(recall_score(y_test,y_pred, pos_label=0))
   P_mecanica.append(precision_score(y_test,y_pred, pos_label=0))
   F_mecanica.append(f1_score(y_test,y_pred, pos_label=0))

   A_mecanica.append(accuracy_score(y_test,y_pred))



plt.plot(max_depths,R_mecanica, 'r', label = 'R')
plt.plot(max_depths,P_mecanica, 'b', label = 'P')
plt.plot(max_depths,F_mecanica, 'y', label = 'F')
plt.plot(max_depths,A_mecanica, 'k', label = 'A')
plt.legend()
plt.show()



Max_depth_mecanica_millor_A                                                    =
{max_depths[A_mecanica.index(max(A_mecanica))]:max(A_mecanica)}
Max_depth_mecanica_millor_F                                                    =
{max_depths[F_mecanica.index(max(F_mecanica))]:max(F_mecanica)}
Max_depth_mecanica_millor_P                                                    =
{max_depths[P_mecanica.index(max(P_mecanica))]:max(P_mecanica)}
Max_depth_mecanica_millor_R                                                    =
{max_depths[R_mecanica.index(max(R_mecanica))]:max(R_mecanica)}
```

Min_Sample_Split

```
import pandas as pd
from sklearn.model_selection import train_test_split
```

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,45:46]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)


min_samples_splits = np.linspace(0.01, 0.7, 70, endpoint=True)

R_mecanica=[]
P_mecanica=[]
F_mecanica=[]
A_mecanica=[]


for min_samples_split in min_samples_splits:
  dt = DecisionTreeClassifier(min_samples_split=min_samples_split)
  dt.fit(X_train, y_train)
  y_pred = dt.predict(X_test)

  R_mecanica.append(recall_score(y_test,y_pred, pos_label=0))
  P_mecanica.append(precision_score(y_test,y_pred, pos_label=0))
  F_mecanica.append(f1_score(y_test,y_pred, pos_label=0))

  A_mecanica.append(accuracy_score(y_test,y_pred))



plt.plot(min_samples_splits,R_mecanica, 'r', label = 'R')
plt.plot(min_samples_splits,P_mecanica, 'b', label = 'P')
plt.plot(min_samples_splits,F_mecanica, 'y', label = 'F')
plt.plot(min_samples_splits,A_mecanica, 'k', label = 'A')
plt.legend()
plt.show()

min_samples_splits_mecanica_millor_A                                                    =
{min_samples_splits[A_mecanica.index(max(A_mecanica))]:max(A_mecanica)}
```

```
min_samples_splits_mecanica_millor_F = {min_samples_splits[F_mecanica.index(max(F_mecanica))]:max(F_mecanica)}
min_samples_splits_mecanica_millor_P = {min_samples_splits[P_mecanica.index(max(P_mecanica))]:max(P_mecanica)}
min_samples_splits_mecanica_millor_R = {min_samples_splits[R_mecanica.index(max(R_mecanica))]:max(R_mecanica)}
```

Best Tree

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
from sklearn.model_selection import GridSearchCV


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,45:46]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)

parameters = {'max_depth' : [18,19,20,21,22,23], 'min_samples_split' : [0.05,0.06,0.07,0.08,0.09,0.1]}
clf_tree = DecisionTreeClassifier(random_state = 1)
clf=GridSearchCV(clf_tree,parameters,scoring = 'f1')
clf.fit(X_train, y_train)
clf.best_estimator_.fit(X_train, y_train)

y_pred = clf.best_estimator_.predict(X_test)

R_electromagnetisme = recall_score(y_test,y_pred, pos_label=0)
P_electromagnetisme = precision_score(y_test,y_pred, pos_label=0)
F_electromagnetisme = f1_score(y_test,y_pred, pos_label=0)


A_electromagnetisme = accuracy_score(y_test,y_pred)
```

Mètodes Numèrics

Max_Depth

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,46:47]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)


max_depths = np.linspace(1, 30, 30, endpoint=True)

R_metodesnum=[]
P_metodesnum=[]
F_metodesnum=[]
A_metodesnum=[]


for max_depth in max_depths:
   dt = DecisionTreeClassifier(max_depth=max_depth, random_state=1)
   dt.fit(X_train, y_train)
   y_pred = dt.predict(X_test)

   R_metodesnum.append(recall_score(y_test,y_pred, pos_label=0))
   P_metodesnum.append(precision_score(y_test,y_pred, pos_label=0))
   F_metodesnum.append(f1_score(y_test,y_pred, pos_label=0))

   A_metodesnum.append(accuracy_score(y_test,y_pred))



plt.plot(max_depths,R_metodesnum, 'r', label = 'R')
plt.plot(max_depths,P_metodesnum, 'b', label = 'P')
```

```
plt.plot(max_depths,F_metodesnum, 'y', label = 'F')
plt.plot(max_depths,A_metodesnum, 'k', label = 'A')
plt.legend()
plt.show()


Max_depth_metodesnum_millor_A =
{max_depths[A_metodesnum.index(max(A_metodesnum))]:max(A_metodesnum)}
Max_depth_metodesnum_millor_F =
{max_depths[F_metodesnum.index(max(F_metodesnum))]:max(F_metodesnum)}
Max_depth_metodesnum_millor_P =
{max_depths[P_metodesnum.index(max(P_metodesnum))]:max(P_metodesnum)}
Max_depth_metodesnum_millor_R =
{max_depths[R_metodesnum.index(max(R_metodesnum))]:max(R_metodesnum)}
```

Min_Sample_Split

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,46:47]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)


min_samples_splits = np.linspace(0.01,0.15,15, endpoint=True)

R_metodesnum=[]
P_metodesnum=[]
F_metodesnum=[]
A_metodesnum=[]


for min_samples_split in min_samples_splits:
   dt = DecisionTreeClassifier(min_samples_split=min_samples_split)
```

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

    R_metodesnum.append(recall_score(y_test,y_pred, pos_label=0))
    P_metodesnum.append(precision_score(y_test,y_pred, pos_label=0))
    F_metodesnum.append(f1_score(y_test,y_pred, pos_label=0))

    A_metodesnum.append(accuracy_score(y_test,y_pred))



plt.plot(min_samples_splits,R_metodesnum, 'r', label = 'R')
plt.plot(min_samples_splits,P_metodesnum, 'b', label = 'P')
plt.plot(min_samples_splits,F_metodesnum, 'y', label = 'F')
plt.plot(min_samples_splits,A_metodesnum, 'k', label = 'A')
plt.legend()
plt.show()

min_samples_splits_metodesnum_millor_A                                                                    =
{min_samples_splits[A_metodesnum.index(max(A_metodesnum))]:max(A_metodesnum)}
min_samples_splits_metodesnum_millor_F                                                                    =
{min_samples_splits[F_metodesnum.index(max(F_metodesnum))]:max(F_metodesnum)}
min_samples_splits_metodesnum_millor_P                                                                    =
{min_samples_splits[P_metodesnum.index(max(P_metodesnum))]:max(P_metodesnum)}
min_samples_splits_metodesnum_millor_R                                                                    =
{min_samples_splits[R_metodesnum.index(max(R_metodesnum))]:max(R_metodesnum)}
```

Best Tree

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
from sklearn.model_selection import GridSearchCV


qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,46:47]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=24601)
```

```
parameters = {'max_depth' : [13,14,15,16, 17, 18], 'min_samples_split' :[0.01,0.015,0.02,0.025,0.03]}
clf_tree = DecisionTreeClassifier(random_state = 1)
clf=GridSearchCV(clf_tree,parameters,scoring = 'f1')
clf.fit(X_train, y_train)
clf.best_estimator_.fit(X_train, y_train)

y_pred = clf.best_estimator_.predict(X_test)

R_electromagnetisme = recall_score(y_test,y_pred, pos_label=0)
P_electromagnetisme = precision_score(y_test,y_pred, pos_label=0)
F_electromagnetisme = f1_score(y_test,y_pred, pos_label=0)


A_electromagnetisme = accuracy_score(y_test,y_pred)
```

Annex 3

Electromagnetimse

Max_Depth

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,41:42]
```

```
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_electromagnetisme_Max_depth=[]
P_electromagnetisme_Max_depth=[]
F_electromagnetisme_Max_depth=[]
A_electromagnetisme_Max_depth=[]

Max_depths = range(1,40)

for Max_depth in Max_depths:
    rfc = RandomForestClassifier(max_depth=Max_depth, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_electromagnetisme_Max_depth.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_electromagnetisme_Max_depth.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_electromagnetisme_Max_depth.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_electromagnetisme_Max_depth.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(Max_depths,R_electromagnetisme_Max_depth, 'r', label = 'R')
plt.plot(Max_depths,P_electromagnetisme_Max_depth, 'b', label = 'P')
plt.plot(Max_depths,F_electromagnetisme_Max_depth, 'y', label = 'F')
plt.plot(Max_depths,A_electromagnetisme_Max_depth, 'k', label = 'A')
plt.legend()
plt.show()


Max_depth_electromagnetisme_millor_A = {Max_depths[A_electromagnetisme_Max_depth.index(max(A_electromagnetisme_Max_depth))]:max(A_electromagnetisme_Max_depth)}
Max_depth_electromagnetisme_millor_F = {Max_depths[F_electromagnetisme_Max_depth.index(max(F_electromagnetisme_Max_depth))]:max(F_electromagnetisme_Max_depth)}
Max_depth_electromagnetisme_millor_P = {Max_depths[P_electromagnetisme_Max_depth.index(max(P_electromagnetisme_Max_depth))]:max(P_electromagnetisme_Max_depth)}
Max_depth_electromagnetisme_millor_R = {Max_depths[R_electromagnetisme_Max_depth.index(max(R_electromagnetisme_Max_depth))]:max(R_electromagnetisme_Max_depth)}
```

Min_Sample_Split

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,41:42]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3,
random_state=24601)

R_electromagnetisme_min_samples_splits=[]
P_electromagnetisme_min_samples_splits=[]
F_electromagnetisme_min_samples_splits=[]
A_electromagnetisme_min_samples_splits=[]

min_samples_splits = np.linspace(0.01, 0.25, 25, endpoint=True)

for min_samples_split in min_samples_splits:
    rfc = RandomForestClassifier(min_samples_split=min_samples_split,
random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_electromagnetisme_min_samples_splits.append(recall_score(y_test.iloc[:,0],y_pred,
pos_label = 0))

P_electromagnetisme_min_samples_splits.append(precision_score(y_test.iloc[:,0],y_pred,
pos_label = 0))
    F_electromagnetisme_min_samples_splits.append(f1_score(y_test.iloc[:,0],y_pred,
pos_label = 0))


A_electromagnetisme_min_samples_splits.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(min_samples_splits,R_electromagnetisme_min_samples_splits, 'r', label = 'R')
plt.plot(min_samples_splits,P_electromagnetisme_min_samples_splits, 'b', label = 'P')
plt.plot(min_samples_splits,F_electromagnetisme_min_samples_splits, 'y', label = 'F')
```

```
plt.plot(min_samples_splits,A_electromagnetisme_min_samples_splits, 'k', label = 'A')
plt.legend()
plt.show()
```

```
min_samples_splits_electromagnetisme_millor_A = {min_samples_splits[A_electromagnetisme_min_samples_splits.index(max(A_electromagnetisme_min_samples_splits))]:max(A_electromagnetisme_min_samples_splits)}
min_samples_splits_electromagnetisme_millor_F = {min_samples_splits[F_electromagnetisme_min_samples_splits.index(max(F_electromagnetisme_min_samples_splits))]:max(F_electromagnetisme_min_samples_splits)}
min_samples_splits_electromagnetisme_millor_P = {min_samples_splits[P_electromagnetisme_min_samples_splits.index(max(P_electromagnetisme_min_samples_splits))]:max(P_electromagnetisme_min_samples_splits)}
min_samples_splits_electromagnetisme_millor_R = {min_samples_splits[R_electromagnetisme_min_samples_splits.index(max(R_electromagnetisme_min_samples_splits))]:max(R_electromagnetisme_min_samples_splits)}
```

Max_Features

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,41:42]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_electromagnetisme_max_features=[]
P_electromagnetisme_max_features=[]
F_electromagnetisme_max_features=[]
A_electromagnetisme_max_features=[]

max_features = list(range(1,42))

for max_feature in max_features:
```

```
    rfc = RandomForestClassifier(max_features=max_feature, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_electromagnetisme_max_features.append(recall_score(y_test.iloc[:,0],y_pred,
pos_label = 0))
    P_electromagnetisme_max_features.append(precision_score(y_test.iloc[:,0],y_pred,
pos_label = 0))
    F_electromagnetisme_max_features.append(f1_score(y_test.iloc[:,0],y_pred,  pos_label =
0))

    A_electromagnetisme_max_features.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(max_features,R_electromagnetisme_max_features, 'r', label = 'R')
plt.plot(max_features,P_electromagnetisme_max_features, 'b', label = 'P')
plt.plot(max_features,F_electromagnetisme_max_features, 'y', label = 'F')
plt.plot(max_features,A_electromagnetisme_max_features, 'k', label = 'A')
plt.legend()
plt.show()



max_features_electromagnetisme_millor_A                                                =
{max_features[A_electromagnetisme_max_features.index(max(A_electromagnetisme_max_
features))]:max(A_electromagnetisme_max_features)}
max_features_electromagnetisme_millor_F                                                =
{max_features[F_electromagnetisme_max_features.index(max(F_electromagnetisme_max_f
eatures))]:max(F_electromagnetisme_max_features)}
max_features_electromagnetisme_millor_P                                                =
{max_features[P_electromagnetisme_max_features.index(max(P_electromagnetisme_max_
features))]:max(P_electromagnetisme_max_features)}
max_features_electromagnetisme_millor_R                                                =
{max_features[R_electromagnetisme_max_features.index(max(R_electromagnetisme_max_
features))]:max(R_electromagnetisme_max_features)}
```

N_Estimators

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
```

```
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,41:42]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, stratify=y, random_state=24601)

R_electromagnetisme_segonsEstimators=[]
P_electromagnetisme_segonsEstimators=[]
F_electromagnetisme_segonsEstimators=[]
A_electromagnetisme_segonsEstimators=[]

n_estimators = range(1,150)

for estimator in n_estimators:
    rfc = RandomForestClassifier(n_estimators=estimator, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_electromagnetisme_segonsEstimators.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_electromagnetisme_segonsEstimators.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_electromagnetisme_segonsEstimators.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_electromagnetisme_segonsEstimators.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(n_estimators,R_electromagnetisme_segonsEstimators, 'r', label = 'R')
plt.plot(n_estimators,P_electromagnetisme_segonsEstimators, 'b', label = 'P')
plt.plot(n_estimators,F_electromagnetisme_segonsEstimators, 'y', label = 'F')
plt.plot(n_estimators,A_electromagnetisme_segonsEstimators, 'k', label = 'A')
plt.legend()
plt.show()


Estimators_electromagnetisme_millor_A = {n_estimators[A_electromagnetisme_segonsEstimators.index(max(A_electromagnetisme_segonsEstimators))]:max(A_electromagnetisme_segonsEstimators)}
Estimators_electromagnetisme_millor_F = {n_estimators[F_electromagnetisme_segonsEstimators.index(max(F_electromagnetisme_segonsEstimators))]:max(F_electromagnetisme_segonsEstimators)}
Estimators_electromagnetisme_millor_P = {n_estimators[P_electromagnetisme_segonsEstimators.index(max(P_electromagnetisme_segonsEstimators))]:max(P_electromagnetisme_segonsEstimators)}
```

```
Estimators_electromagnetisme_millor_R = {n_estimators[R_electromagnetisme_segonsEstimators.index(max(R_electromagnetisme_segonsEstimators))]:max(R_electromagnetisme_segonsEstimators)}
```

Best Random Forest

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,41:42]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.4, random_state=24601)


rfc = RandomForestClassifier(n_estimators=8 , max_depth=23, min_samples_split=0.04, max_features= 23,random_state=525600)
rfc = rfc.fit(X_train, y_train)
y_pred = rfc.predict(X_test)

M_elect = confusion_matrix(y_test.iloc[:,0],y_pred)

R_elect = recall_score(y_test.iloc[:,0],y_pred, pos_label = 0)
P_elect = precision_score(y_test.iloc[:,0],y_pred, pos_label = 0)

F_elect = f1_score(y_test.iloc[:,0],y_pred, pos_label = 0)

A_elect = accuracy_score(y_test.iloc[:,0],y_pred)
```

Equacions Diferencials

Max_Depth

```
import pandas as pd
from sklearn.model_selection import train_test_split
```

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,42:43]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3,
random_state=24601)

R_eqdif_Max_depth=[]
P_eqdif_Max_depth=[]
F_eqdif_Max_depth=[]
A_eqdif_Max_depth=[]

Max_depths = range(1,25)

for Max_depth in Max_depths:
    rfc = RandomForestClassifier(max_depth=Max_depth, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_eqdif_Max_depth.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_eqdif_Max_depth.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_eqdif_Max_depth.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_eqdif_Max_depth.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(Max_depths,R_eqdif_Max_depth, 'r', label = 'R')
plt.plot(Max_depths,P_eqdif_Max_depth, 'b', label = 'P')
plt.plot(Max_depths,F_eqdif_Max_depth, 'y', label = 'F')
plt.plot(Max_depths,A_eqdif_Max_depth, 'k', label = 'A')
plt.legend()
plt.show()


Max_depth_eqdif_millor_A =
{Max_depths[A_eqdif_Max_depth.index(max(A_eqdif_Max_depth))]:max(A_eqdif_Max_dept
h)}
Max_depth_eqdif_millor_F =
{Max_depths[F_eqdif_Max_depth.index(max(F_eqdif_Max_depth))]:max(F_eqdif_Max_dept
h)}
```

```
Max_depth_eqdif_millor_P = {Max_depths[P_eqdif_Max_depth.index(max(P_eqdif_Max_depth))]:max(P_eqdif_Max_depth)}
Max_depth_eqdif_millor_R = {Max_depths[R_eqdif_Max_depth.index(max(R_eqdif_Max_depth))]:max(R_eqdif_Max_depth)}
```

Min_Sample_Split

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,42:43]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_eqdif_min_samples_splits=[]
P_eqdif_min_samples_splits=[]
F_eqdif_min_samples_splits=[]
A_eqdif_min_samples_splits=[]

min_samples_splits = np.linspace(0.005, 0.05, 10, endpoint=True)

for min_samples_split in min_samples_splits:
    rfc = RandomForestClassifier(min_samples_split=min_samples_split, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_eqdif_min_samples_splits.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_eqdif_min_samples_splits.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_eqdif_min_samples_splits.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_eqdif_min_samples_splits.append(accuracy_score(y_test.iloc[:,0],y_pred))
```

```
plt.plot(min_samples_splits,R_eqdif_min_samples_splits, 'r', label = 'R')
plt.plot(min_samples_splits,P_eqdif_min_samples_splits, 'b', label = 'P')
plt.plot(min_samples_splits,F_eqdif_min_samples_splits, 'y', label = 'F')
plt.plot(min_samples_splits,A_eqdif_min_samples_splits, 'k', label = 'A')
plt.legend()
plt.show()
```

```
min_samples_splits_eqdif_millor_A = {min_samples_splits[A_eqdif_min_samples_splits.index(max(A_eqdif_min_samples_splits))]:max(A_eqdif_min_samples_splits)}
min_samples_splits_eqdif_millor_F = {min_samples_splits[F_eqdif_min_samples_splits.index(max(F_eqdif_min_samples_splits))]:max(F_eqdif_min_samples_splits)}
min_samples_splits_eqdif_millor_P = {min_samples_splits[P_eqdif_min_samples_splits.index(max(P_eqdif_min_samples_splits))]:max(P_eqdif_min_samples_splits)}
min_samples_splits_eqdif_millor_R = {min_samples_splits[R_eqdif_min_samples_splits.index(max(R_eqdif_min_samples_splits))]:max(R_eqdif_min_samples_splits)}
```

Max_Features

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,42:43]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_eqdif_max_features=[]
P_eqdif_max_features=[]
F_eqdif_max_features=[]
A_eqdif_max_features=[]

max_features = list(range(1,42))
```

```
for max_feature in max_features:
    rfc = RandomForestClassifier(max_features=max_feature, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_eqdif_max_features.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_eqdif_max_features.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_eqdif_max_features.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_eqdif_max_features.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(max_features,R_eqdif_max_features, 'r', label = 'R')
plt.plot(max_features,P_eqdif_max_features, 'b', label = 'P')
plt.plot(max_features,F_eqdif_max_features, 'y', label = 'F')
plt.plot(max_features,A_eqdif_max_features, 'k', label = 'A')
plt.legend()
plt.show()



max_features_eqdif_millor_A =
{max_features[A_eqdif_max_features.index(max(A_eqdif_max_features))]:max(A_eqdif_ma
x_features)}
max_features_eqdif_millor_F =
{max_features[F_eqdif_max_features.index(max(F_eqdif_max_features))]:max(F_eqdif_max
_features)}
max_features_eqdif_millor_P =
{max_features[P_eqdif_max_features.index(max(P_eqdif_max_features))]:max(P_eqdif_ma
x_features)}
max_features_eqdif_millor_R =
{max_features[R_eqdif_max_features.index(max(R_eqdif_max_features))]:max(R_eqdif_ma
x_features)}
```

N_Estimators

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt
```

```
qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,42:43]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, stratify=y, random_state=24601)

R_eqdif_segonsEstimators=[]
P_eqdif_segonsEstimators=[]
F_eqdif_segonsEstimators=[]
A_eqdif_segonsEstimators=[]

n_estimators = range(1,150)

for estimator in n_estimators:
    rfc = RandomForestClassifier(n_estimators=estimator, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_eqdif_segonsEstimators.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_eqdif_segonsEstimators.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_eqdif_segonsEstimators.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_eqdif_segonsEstimators.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(n_estimators,R_eqdif_segonsEstimators, 'r', label = 'R')
plt.plot(n_estimators,P_eqdif_segonsEstimators, 'b', label = 'P')
plt.plot(n_estimators,F_eqdif_segonsEstimators, 'y', label = 'F')
plt.plot(n_estimators,A_eqdif_segonsEstimators, 'k', label = 'A')
plt.legend()
plt.show()



Estimators_eqdif_millor_A =
{n_estimators[A_eqdif_segonsEstimators.index(max(A_eqdif_segonsEstimators))]:max(A_eqdif_segonsEstimators)}
Estimators_eqdif_millor_F =
{n_estimators[F_eqdif_segonsEstimators.index(max(F_eqdif_segonsEstimators))]:max(F_eqdif_segonsEstimators)}
Estimators_eqdif_millor_P =
{n_estimators[P_eqdif_segonsEstimators.index(max(P_eqdif_segonsEstimators))]:max(P_eqdif_segonsEstimators)}
Estimators_eqdif_millor_R =
{n_estimators[R_eqdif_segonsEstimators.index(max(R_eqdif_segonsEstimators))]:max(R_eqdif_segonsEstimators)}
```

Best Random Forest

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,42:43]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)


rfc = RandomForestClassifier(n_estimators=1, max_depth=22, min_samples_split=0.005, max_features=21, random_state=525600)
rfc = rfc.fit(X_train, y_train)
y_pred = rfc.predict(X_test)

M_eqdif = confusion_matrix(y_test.iloc[:,0],y_pred)
R_eqdif = recall_score(y_test.iloc[:,0],y_pred, pos_label = 0)
P_eqdif = precision_score(y_test.iloc[:,0],y_pred, pos_label = 0)

F_eqdif = f1_score(y_test.iloc[:,0],y_pred, pos_label = 0)

A_eqdif = accuracy_score(y_test.iloc[:,0],y_pred)
```

Informàtica

Max_Depth

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
```

```
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,43:44]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_info_Max_depth=[]
P_info_Max_depth=[]
F_info_Max_depth=[]
A_info_Max_depth=[]

Max_depths = range(1,25)

for Max_depth in Max_depths:
    rfc = RandomForestClassifier(max_depth=Max_depth, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_info_Max_depth.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_info_Max_depth.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_info_Max_depth.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_info_Max_depth.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(Max_depths,R_info_Max_depth, 'r', label = 'R')
plt.plot(Max_depths,P_info_Max_depth, 'b', label = 'P')
plt.plot(Max_depths,F_info_Max_depth, 'y', label = 'F')
plt.plot(Max_depths,A_info_Max_depth, 'k', label = 'A')
plt.legend()
plt.show()


Max_depth_info_millor_A = {Max_depths[A_info_Max_depth.index(max(A_info_Max_depth))]:max(A_info_Max_depth)}
Max_depth_info_millor_F = {Max_depths[F_info_Max_depth.index(max(F_info_Max_depth))]:max(F_info_Max_depth)}
Max_depth_info_millor_P = {Max_depths[P_info_Max_depth.index(max(P_info_Max_depth))]:max(P_info_Max_depth)}
Max_depth_info_millor_R = {Max_depths[R_info_Max_depth.index(max(R_info_Max_depth))]:max(R_info_Max_depth)}
```

Min_Sample_Split

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,43:44]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3,
random_state=24601)

R_info_min_samples_splits=[]
P_info_min_samples_splits=[]
F_info_min_samples_splits=[]
A_info_min_samples_splits=[]

min_samples_splits = np.linspace(0.001, 0.1, 100, endpoint=True)

for min_samples_split in min_samples_splits:
    rfc = RandomForestClassifier(min_samples_split=min_samples_split,
random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_info_min_samples_splits.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_info_min_samples_splits.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_info_min_samples_splits.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_info_min_samples_splits.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(min_samples_splits,R_info_min_samples_splits, 'r', label = 'R')
plt.plot(min_samples_splits,P_info_min_samples_splits, 'b', label = 'P')
plt.plot(min_samples_splits,F_info_min_samples_splits, 'y', label = 'F')
plt.plot(min_samples_splits,A_info_min_samples_splits, 'k', label = 'A')
plt.legend()
plt.show()
```

```
min_samples_splits_info_millor_A = {min_samples_splits[A_info_min_samples_splits.index(max(A_info_min_samples_splits))]:max(A_info_min_samples_splits)}
min_samples_splits_info_millor_F = {min_samples_splits[F_info_min_samples_splits.index(max(F_info_min_samples_splits))]:max(F_info_min_samples_splits)}
min_samples_splits__millor_P = {min_samples_splits[P_info_min_samples_splits.index(max(P_info_min_samples_splits))]:max(P_info_min_samples_splits)}
min_samples_splits_info_millor_R = {min_samples_splits[R_info_min_samples_splits.index(max(R_info_min_samples_splits))]:max(R_info_min_samples_splits)}
```

Max_Features

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,43:44]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_info_max_features=[]
P_info_max_features=[]
F_info_max_features=[]
A_info_max_features=[]

max_features = list(range(1,42))

for max_feature in max_features:
    rfc = RandomForestClassifier(max_features=max_feature, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_info_max_features.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_info_max_features.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_info_max_features.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_info_max_features.append(accuracy_score(y_test.iloc[:,0],y_pred))
```

```
plt.plot(max_features,R_info_max_features, 'r', label = 'R')
plt.plot(max_features,P_info_max_features, 'b', label = 'P')
plt.plot(max_features,F_info_max_features, 'y', label = 'F')
plt.plot(max_features,A_info_max_features, 'k', label = 'A')
plt.legend()
plt.show()


max_features_info_millor_A = {max_features[A_info_max_features.index(max(A_info_max_features))]:max(A_info_max_features)}
max_features_info_millor_F = {max_features[F_info_max_features.index(max(F_info_max_features))]:max(F_info_max_features)}
max_features_info_millor_P = {max_features[P_info_max_features.index(max(P_info_max_features))]:max(P_info_max_features)}
max_features_info_millor_R = {max_features[R_info_max_features.index(max(R_info_max_features))]:max(R_info_max_features)}
```

N_Estimators

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,43:44]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, stratify=y, random_state=24601)

R_info_segonsEstimators=[]
P_info_segonsEstimators=[]
F_info_segonsEstimators=[]
A_info_segonsEstimators=[]
```

```
n_estimators = range(60,100)

for estimator in n_estimators:
    rfc = RandomForestClassifier(n_estimators=estimator, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_info_segonsEstimators.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_info_segonsEstimators.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_info_segonsEstimators.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_info_segonsEstimators.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(n_estimators,R_info_segonsEstimators, 'r', label = 'R')
plt.plot(n_estimators,P_info_segonsEstimators, 'b', label = 'P')
plt.plot(n_estimators,F_info_segonsEstimators, 'y', label = 'F')
plt.plot(n_estimators,A_info_segonsEstimators, 'k', label = 'A')
plt.legend()
plt.show()



Estimators_info_millor_A =
{n_estimators[A_info_segonsEstimators.index(max(A_info_segonsEstimators))]:max(A_info
_segonsEstimators)}
Estimators_info_millor_F =
{n_estimators[F_info_segonsEstimators.index(max(F_info_segonsEstimators))]:max(F_info_
segonsEstimators)}
Estimators_info_millor_P =
{n_estimators[P_info_segonsEstimators.index(max(P_info_segonsEstimators))]:max(P_info
_segonsEstimators)}
Estimators_info_millor_R =
{n_estimators[R_info_segonsEstimators.index(max(R_info_segonsEstimators))]:max(R_info
_segonsEstimators)}
```

Best Random Forest

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
```

```
qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,43:44]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)



rfc = RandomForestClassifier(n_estimators=86, max_depth=13, min_samples_split=0.014,max_features=36, random_state=525600)
rfc = rfc.fit(X_train, y_train)
y_pred = rfc.predict(X_test)

M_info = confusion_matrix(y_test.iloc[:,0],y_pred)

R_info = recall_score(y_test.iloc[:,0],y_pred, pos_label = 0)
P_info = precision_score(y_test.iloc[:,0],y_pred, pos_label = 0)
F_info = f1_score(y_test.iloc[:,0],y_pred, pos_label = 0)


A_info = accuracy_score(y_test.iloc[:,0],y_pred)
```

Materials

Max_Depth

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,44:45]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_materials_Max_depth=[]
```

```
P_materials_Max_depth=[]
F_materials_Max_depth=[]
A_materials_Max_depth=[]

Max_depths = range(1,26)

for Max_depth in Max_depths:
    rfc = RandomForestClassifier(max_depth=Max_depth, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_materials_Max_depth.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_materials_Max_depth.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_materials_Max_depth.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_materials_Max_depth.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(Max_depths,R_materials_Max_depth, 'r', label = 'R')
plt.plot(Max_depths,P_materials_Max_depth, 'b', label = 'P')
plt.plot(Max_depths,F_materials_Max_depth, 'y', label = 'F')
plt.plot(Max_depths,A_materials_Max_depth, 'k', label = 'A')
plt.legend()
plt.show()


Max_depth_materials_millor_A = {Max_depths[A_materials_Max_depth.index(max(A_materials_Max_depth))]:max(A_materials_Max_depth)}
Max_depth_materials_millor_F = {Max_depths[F_materials_Max_depth.index(max(F_materials_Max_depth))]:max(F_materials_Max_depth)}
Max_depth_materials_millor_P = {Max_depths[P_materials_Max_depth.index(max(P_materials_Max_depth))]:max(P_materials_Max_depth)}
Max_depth_materials_millor_R = {Max_depths[R_materials_Max_depth.index(max(R_materials_Max_depth))]:max(R_materials_Max_depth)}
```

Min_Sample_Split

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
```

```
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,44:45]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_materials_min_samples_splits=[]
P_materials_min_samples_splits=[]
F_materials_min_samples_splits=[]
A_materials_min_samples_splits=[]

min_samples_splits = np.linspace(0.0015, 0.15, 100, endpoint=True)

for min_samples_split in min_samples_splits:
    rfc = RandomForestClassifier(min_samples_split=min_samples_split, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_materials_min_samples_splits.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_materials_min_samples_splits.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_materials_min_samples_splits.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_materials_min_samples_splits.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(min_samples_splits,R_materials_min_samples_splits, 'r', label = 'R')
plt.plot(min_samples_splits,P_materials_min_samples_splits, 'b', label = 'P')
plt.plot(min_samples_splits,F_materials_min_samples_splits, 'y', label = 'F')
plt.plot(min_samples_splits,A_materials_min_samples_splits, 'k', label = 'A')
plt.legend()
plt.show()



min_samples_splits_materials_millor_A = {min_samples_splits[A_materials_min_samples_splits.index(max(A_materials_min_samples_splits))]:max(A_materials_min_samples_splits)}
min_samples_splits_materials_millor_F = {min_samples_splits[F_materials_min_samples_splits.index(max(F_materials_min_samples_splits))]:max(F_materials_min_samples_splits)}
```

```
min_samples_splits_materials_millor_P = {min_samples_splits[P_materials_min_samples_splits.index(max(P_materials_min_samples_splits))]:max(P_materials_min_samples_splits)}
min_samples_splits_materials_millor_R = {min_samples_splits[R_materials_min_samples_splits.index(max(R_materials_min_samples_splits))]:max(R_materials_min_samples_splits)}
```

Max_Features

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,44:45]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_materials_max_features=[]
P_materials_max_features=[]
F_materials_max_features=[]
A_materials_max_features=[]

max_features = list(range(15,38))

for max_feature in max_features:
    rfc = RandomForestClassifier(max_features=max_feature, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_materials_max_features.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_materials_max_features.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_materials_max_features.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_materials_max_features.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(max_features,R_materials_max_features, 'r', label = 'R')
```

```
plt.plot(max_features,P_materials_max_features, 'b', label = 'P')
plt.plot(max_features,F_materials_max_features, 'y', label = 'F')
plt.plot(max_features,A_materials_max_features, 'k', label = 'A')
plt.legend()
plt.show()
```

```
max_features_materials_millor_A = {max_features[A_materials_max_features.index(max(A_materials_max_features))]:max(A_materials_max_features)}
max_features_materials_millor_F = {max_features[F_materials_max_features.index(max(F_materials_max_features))]:max(F_materials_max_features)}
max_features_materials_millor_P = {max_features[P_materials_max_features.index(max(P_materials_max_features))]:max(P_materials_max_features)}
max_features_materials_millor_R = {max_features[R_materials_max_features.index(max(R_materials_max_features))]:max(R_materials_max_features)}
```

N_Estimators

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,44:45]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, stratify=y, random_state=24601)

R_materials_segonsEstimators=[]
P_materials_segonsEstimators=[]
F_materials_segonsEstimators=[]
A_materials_segonsEstimators=[]

n_estimators = range(1,50)
```

```
for estimator in n_estimators:
    rfc = RandomForestClassifier(n_estimators=estimator, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_materials_segonsEstimators.append(recall_score(y_test.iloc[:,0],y_pred,  pos_label  =
0))
    P_materials_segonsEstimators.append(precision_score(y_test.iloc[:,0],y_pred,  pos_label
= 0))
    F_materials_segonsEstimators.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_materials_segonsEstimators.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(n_estimators,R_materials_segonsEstimators, 'r', label = 'R')
plt.plot(n_estimators,P_materials_segonsEstimators, 'b', label = 'P')
plt.plot(n_estimators,F_materials_segonsEstimators, 'y', label = 'F')
plt.plot(n_estimators,A_materials_segonsEstimators, 'k', label = 'A')
plt.legend()
plt.show()



Estimators_materials_millor_A                                                    =
{n_estimators[A_materials_segonsEstimators.index(max(A_materials_segonsEstimators))]:
max(A_materials_segonsEstimators)}
Estimators_materials_millor_F                                                    =
{n_estimators[F_materials_segonsEstimators.index(max(F_materials_segonsEstimators))]:
max(F_materials_segonsEstimators)}
Estimators_materials_millor_P                                                    =
{n_estimators[P_materials_segonsEstimators.index(max(P_materials_segonsEstimators))]:
max(P_materials_segonsEstimators)}
Estimators_materials_millor_R                                                    =
{n_estimators[R_materials_segonsEstimators.index(max(R_materials_segonsEstimators))]:
max(R_materials_segonsEstimators)}
```

Best Random Tree

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
```

```
qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,44:45]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)


rfc = RandomForestClassifier(n_estimators=2, max_depth=23, min_samples_split=0.0015, max_features=35, random_state=525600)
rfc = rfc.fit(X_train, y_train)
y_pred = rfc.predict(X_test)

M_materials = confusion_matrix(y_test.iloc[:,0],y_pred)

R_materials = recall_score(y_test.iloc[:,0],y_pred, pos_label = 0)
P_materials = precision_score(y_test.iloc[:,0],y_pred, pos_label = 0)
F_materials = f1_score(y_test.iloc[:,0],y_pred, pos_label = 0)

A_materials = accuracy_score(y_test.iloc[:,0],y_pred)
```

Mecànica

Max_Depth

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,45:46]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_mecanica_Max_depth=[]
P_mecanica_Max_depth=[]
F_mecanica_Max_depth=[]
```

```
A_mecanica_Max_depth=[]


Max_depths = range(1,35)

for Max_depth in Max_depths:
    rfc = RandomForestClassifier(max_depth=Max_depth, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_mecanica_Max_depth.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_mecanica_Max_depth.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_mecanica_Max_depth.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_mecanica_Max_depth.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(Max_depths,R_mecanica_Max_depth, 'r', label = 'R')
plt.plot(Max_depths,P_mecanica_Max_depth, 'b', label = 'P')
plt.plot(Max_depths,F_mecanica_Max_depth, 'y', label = 'F')
plt.plot(Max_depths,A_mecanica_Max_depth, 'k', label = 'A')
plt.legend()
plt.show()


Max_depth_mecanica_millor_A = {Max_depths[A_mecanica_Max_depth.index(max(A_mecanica_Max_depth))]:max(A_mecanica_Max_depth)}
Max_depth_mecanica_millor_F = {Max_depths[F_mecanica_Max_depth.index(max(F_mecanica_Max_depth))]:max(F_mecanica_Max_depth)}
Max_depth_mecanica_millor_P = {Max_depths[P_mecanica_Max_depth.index(max(P_mecanica_Max_depth))]:max(P_mecanica_Max_depth)}
Max_depth_mecanica_millor_R = {Max_depths[R_mecanica_Max_depth.index(max(R_mecanica_Max_depth))]:max(R_mecanica_Max_depth)}
```

Min_Sample_Split

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
```

```
from sklearn.metrics import f1_score
import numpy as np
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,45:46]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_mecanica_min_samples_splits=[]
P_mecanica_min_samples_splits=[]
F_mecanica_min_samples_splits=[]
A_mecanica_min_samples_splits=[]

min_samples_splits = np.linspace(0.004, 0.4, 100, endpoint=True)

for min_samples_split in min_samples_splits:
    rfc = RandomForestClassifier(min_samples_split=min_samples_split, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_mecanica_min_samples_splits.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_mecanica_min_samples_splits.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_mecanica_min_samples_splits.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_mecanica_min_samples_splits.append(accuracy_score(y_test.iloc[:,0],y_pred))


plt.plot(min_samples_splits,R_mecanica_min_samples_splits, 'r', label = 'R')
plt.plot(min_samples_splits,P_mecanica_min_samples_splits, 'b', label = 'P')
plt.plot(min_samples_splits,F_mecanica_min_samples_splits, 'y', label = 'F')
plt.plot(min_samples_splits,A_mecanica_min_samples_splits, 'k', label = 'A')
plt.legend()
plt.show()


min_samples_splits_mecanica_millor_A = {min_samples_splits[A_mecanica_min_samples_splits.index(max(A_mecanica_min_samples_splits))]:max(A_mecanica_min_samples_splits)}
min_samples_splits_mecanica_millor_F = {min_samples_splits[F_mecanica_min_samples_splits.index(max(F_mecanica_min_samples_splits))]:max(F_mecanica_min_samples_splits)}
```

```
min_samples_splits_mecanica_millor_P = {min_samples_splits[P_mecanica_min_samples_splits.index(max(P_mecanica_min_samples_splits))]:max(P_mecanica_min_samples_splits)}
min_samples_splits_mecanica_millor_R = {min_samples_splits[R_mecanica_min_samples_splits.index(max(R_mecanica_min_samples_splits))]:max(R_mecanica_min_samples_splits)}
```

Max_Features

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,45:46]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_mecanica_max_features=[]
P_mecanica_max_features=[]
F_mecanica_max_features=[]
A_mecanica_max_features=[]

max_features = list(range(1,42))

for max_feature in max_features:
    rfc = RandomForestClassifier(max_features=max_feature, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_mecanica_max_features.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_mecanica_max_features.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_mecanica_max_features.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_mecanica_max_features.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(max_features,R_mecanica_max_features, 'r', label = 'R')
plt.plot(max_features,P_mecanica_max_features, 'b', label = 'P')
```

```
plt.plot(max_features,F_mecanica_max_features, 'y', label = 'F')
plt.plot(max_features,A_mecanica_max_features, 'k', label = 'A')
plt.legend()
plt.show()



max_features_mecanica_millor_A = {max_features[A_mecanica_max_features.index(max(A_mecanica_max_features))]:max(A_mecanica_max_features)}
max_features_mecanica_millor_F = {max_features[F_mecanica_max_features.index(max(F_mecanica_max_features))]:max(F_mecanica_max_features)}
max_features_mecanica_millor_P = {max_features[P_mecanica_max_features.index(max(P_mecanica_max_features))]:max(P_mecanica_max_features)}
max_features_mecanica_millor_R = {max_features[R_mecanica_max_features.index(max(R_mecanica_max_features))]:max(R_mecanica_max_features)}
```

N_Estimators

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,45:46]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, stratify=y, random_state=24601)

R_mecanica_segonsEstimators=[]
P_mecanica_segonsEstimators=[]
F_mecanica_segonsEstimators=[]
A_mecanica_segonsEstimators=[]

n_estimators = range(1,150,2)

for estimator in n_estimators:
    rfc = RandomForestClassifier(n_estimators=estimator, random_state=525600)
```

```
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_mecanica_segonsEstimators.append(recall_score(y_test.iloc[:,0],y_pred,  pos_label  =
0))
    P_mecanica_segonsEstimators.append(precision_score(y_test.iloc[:,0],y_pred,  pos_label
= 0))
    F_mecanica_segonsEstimators.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_mecanica_segonsEstimators.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(n_estimators,R_mecanica_segonsEstimators, 'r', label = 'R')
plt.plot(n_estimators,P_mecanica_segonsEstimators, 'b', label = 'P')
plt.plot(n_estimators,F_mecanica_segonsEstimators, 'y', label = 'F')
plt.plot(n_estimators,A_mecanica_segonsEstimators, 'k', label = 'A')
plt.legend()
plt.show()



Estimators_mecanica_millor_A                                                    =
{n_estimators[A_mecanica_segonsEstimators.index(max(A_mecanica_segonsEstimators))]:
max(A_mecanica_segonsEstimators)}
Estimators_mecanica_millor_F                                                    =
{n_estimators[F_mecanica_segonsEstimators.index(max(F_mecanica_segonsEstimators))]:
max(F_mecanica_segonsEstimators)}
Estimators_mecanica_millor_P                                                    =
{n_estimators[P_mecanica_segonsEstimators.index(max(P_mecanica_segonsEstimators))]:
max(P_mecanica_segonsEstimators)}
Estimators_mecanica_millor_R                                                    =
{n_estimators[R_mecanica_segonsEstimators.index(max(R_mecanica_segonsEstimators))]:
max(R_mecanica_segonsEstimators)}
```

Best Random Tree

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
```

```
X = qf.iloc[:,:41]
y =qf.iloc[:,45:46]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)



rfc = RandomForestClassifier(n_estimators=48, max_depth=22, min_samples_split=0.008, max_features=29, random_state=525600)
rfc = rfc.fit(X_train, y_train)
y_pred = rfc.predict(X_test)

M_mecanica = confusion_matrix(y_test.iloc[:,0],y_pred)

R_mecanica = recall_score(y_test.iloc[:,0],y_pred, pos_label = 0)
P_mecanica = precision_score(y_test.iloc[:,0],y_pred, pos_label = 0)
F_mecanica = f1_score(y_test.iloc[:,0],y_pred, pos_label = 0)


A_mecanica= accuracy_score(y_test.iloc[:,0],y_pred)
```

Mètodes Numèrics

Max_Depth

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,46:47]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_MetNum_Max_depth=[]
P_MetNum_Max_depth=[]
F_MetNum_Max_depth=[]
A_MetNum_Max_depth=[]
```

```
Max_depths = range(1,30)

for Max_depth in Max_depths:
    rfc = RandomForestClassifier(max_depth=Max_depth, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_MetNum_Max_depth.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_MetNum_Max_depth.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_MetNum_Max_depth.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_MetNum_Max_depth.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(Max_depths,R_MetNum_Max_depth, 'r', label = 'R')
plt.plot(Max_depths,P_MetNum_Max_depth, 'b', label = 'P')
plt.plot(Max_depths,F_MetNum_Max_depth, 'y', label = 'F')
plt.plot(Max_depths,A_MetNum_Max_depth, 'k', label = 'A')
plt.legend()
plt.show()


Max_depth_MetNum_millor_A = {Max_depths[A_MetNum_Max_depth.index(max(A_MetNum_Max_depth))]:max(A_MetNum_Max_depth)}
Max_depth_MetNum_millor_F = {Max_depths[F_MetNum_Max_depth.index(max(F_MetNum_Max_depth))]:max(F_MetNum_Max_depth)}
Max_depth_MetNum_millor_P = {Max_depths[P_MetNum_Max_depth.index(max(P_MetNum_Max_depth))]:max(P_MetNum_Max_depth)}
Max_depth_MetNum_millor_R = {Max_depths[R_MetNum_Max_depth.index(max(R_MetNum_Max_depth))]:max(R_MetNum_Max_depth)}
```

Min_Sample_Split

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import numpy as np
```

```
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,46:47]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_MetNum_min_samples_splits=[]
P_MetNum_min_samples_splits=[]
F_MetNum_min_samples_splits=[]
A_MetNum_min_samples_splits=[]

min_samples_splits = np.linspace(0.001, 0.03, 30, endpoint=True)

for min_samples_split in min_samples_splits:
    rfc = RandomForestClassifier(min_samples_split=min_samples_split, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_MetNum_min_samples_splits.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_MetNum_min_samples_splits.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_MetNum_min_samples_splits.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_MetNum_min_samples_splits.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(min_samples_splits,R_MetNum_min_samples_splits, 'r', label = 'R')
plt.plot(min_samples_splits,P_MetNum_min_samples_splits, 'b', label = 'P')
plt.plot(min_samples_splits,F_MetNum_min_samples_splits, 'y', label = 'F')
plt.plot(min_samples_splits,A_MetNum_min_samples_splits, 'k', label = 'A')
plt.legend()
plt.show()



min_samples_splits_MetNum_millor_A = {min_samples_splits[A_MetNum_min_samples_splits.index(max(A_MetNum_min_samples_splits))]:max(A_MetNum_min_samples_splits)}
min_samples_splits_MetNum_millor_F = {min_samples_splits[F_MetNum_min_samples_splits.index(max(F_MetNum_min_samples_splits))]:max(F_MetNum_min_samples_splits)}
```

```
min_samples_splits_MetNum_millor_P = {min_samples_splits[P_MetNum_min_samples_splits.index(max(P_MetNum_min_samples_splits))]:max(P_MetNum_min_samples_splits)}
min_samples_splits_MetNum_millor_R = {min_samples_splits[R_MetNum_min_samples_splits.index(max(R_MetNum_min_samples_splits))]:max(R_MetNum_min_samples_splits)}
```

Max_Features

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,46:47]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)

R_MetNum_max_features=[]
P_MetNum_max_features=[]
F_MetNum_max_features=[]
A_MetNum_max_features=[]

max_features = list(range(1,42))

for max_feature in max_features:
    rfc = RandomForestClassifier(max_features=max_feature, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_MetNum_max_features.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_MetNum_max_features.append(precision_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    F_MetNum_max_features.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_MetNum_max_features.append(accuracy_score(y_test.iloc[:,0],y_pred))



plt.plot(max_features,R_MetNum_max_features, 'r', label = 'R')
plt.plot(max_features,P_MetNum_max_features, 'b', label = 'P')
plt.plot(max_features,F_MetNum_max_features, 'y', label = 'F')
```

```
plt.plot(max_features,A_MetNum_max_features, 'k', label = 'A')
plt.legend()
plt.show()



max_features_MetNum_millor_A = {max_features[A_MetNum_max_features.index(max(A_MetNum_max_features))]:max(A_MetNum_max_features)}
max_features_MetNum_millor_F = {max_features[F_MetNum_max_features.index(max(F_MetNum_max_features))]:max(F_MetNum_max_features)}
max_features_MetNum_millor_P = {max_features[P_MetNum_max_features.index(max(P_MetNum_max_features))]:max(P_MetNum_max_features)}
max_features_MetNum_millor_R = {max_features[R_MetNum_max_features.index(max(R_MetNum_max_features))]:max(R_MetNum_max_features)}
```

N_Estimators

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
import matplotlib.pyplot as plt

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
X = qf.iloc[:,:41]
y =qf.iloc[:,46:47]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, stratify=y, random_state=24601)

R_MetNum_segonsEstimators=[]
P_MetNum_segonsEstimators=[]
F_MetNum_segonsEstimators=[]
A_MetNum_segonsEstimators=[]

n_estimators = range(1,150)

for estimator in n_estimators:
```

```
    rfc = RandomForestClassifier(n_estimators=estimator, random_state=525600)
    rfc = rfc.fit(X_train, y_train)
    y_pred = rfc.predict(X_test)
    R_MetNum_segonsEstimators.append(recall_score(y_test.iloc[:,0],y_pred, pos_label = 0))
    P_MetNum_segonsEstimators.append(precision_score(y_test.iloc[:,0],y_pred, pos_label =
0))
    F_MetNum_segonsEstimators.append(f1_score(y_test.iloc[:,0],y_pred, pos_label = 0))

    A_MetNum_segonsEstimators.append(accuracy_score(y_test.iloc[:,0],y_pred))


plt.plot(n_estimators,R_MetNum_segonsEstimators, 'r', label = 'R')
plt.plot(n_estimators,P_MetNum_segonsEstimators, 'b', label = 'P')
plt.plot(n_estimators,F_MetNum_segonsEstimators, 'y', label = 'F')
plt.plot(n_estimators,A_MetNum_segonsEstimators, 'k', label = 'A')
plt.legend()
plt.show()


Estimators_MetNum_millor_A                                                    =
{n_estimators[A_MetNum_segonsEstimators.index(max(A_MetNum_segonsEstimators))]:m
ax(A_MetNum_segonsEstimators)}
Estimators_MetNum_millor_F                                                    =
{n_estimators[F_MetNum_segonsEstimators.index(max(F_MetNum_segonsEstimators))]:m
ax(F_MetNum_segonsEstimators)}
Estimators_MetNum_millor_P                                                    =
{n_estimators[P_MetNum_segonsEstimators.index(max(P_MetNum_segonsEstimators))]:m
ax(P_MetNum_segonsEstimators)}
Estimators_MetNum_millor_R                                                    =
{n_estimators[R_MetNum_segonsEstimators.index(max(R_MetNum_segonsEstimators))]:m
ax(R_MetNum_segonsEstimators)}
```

Best Random Tree

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score

qf = pd.read_excel(r'C:\Users\Núria\Desktop\TFG\generalperpredirQ3.xlsx')
qf.set_index('CODI_EXPEDIENT', inplace=True)
```

```
X = qf.iloc[:,:41]
y =qf.iloc[:,46:47]
X_train, X_test, y_train, y_test = train_test_split(X,y, stratify=y, test_size=0.3, random_state=24601)


rfc = RandomForestClassifier(n_estimators=6, max_depth=14, min_samples_split=0.001, max_features=37, random_state=525600)
rfc = rfc.fit(X_train, y_train)
y_pred = rfc.predict(X_test)

M_MetNum = confusion_matrix(y_test.iloc[:,0],y_pred)

R_MetNum = recall_score(y_test.iloc[:,0],y_pred, pos_label = 0)
P_MetNum = precision_score(y_test.iloc[:,0],y_pred, pos_label = 0)
F_MetNum = f1_score(y_test.iloc[:,0],y_pred, pos_label = 0)


A_MetNum = accuracy_score(y_test.iloc[:,0],y_pred)
```