Predicció de l'èxit o abandonament acadèmic dels estudiants

Mariona Farré Tapias

Índex:

01
Descripció de les dades originals

02
Preprocessament de les dades

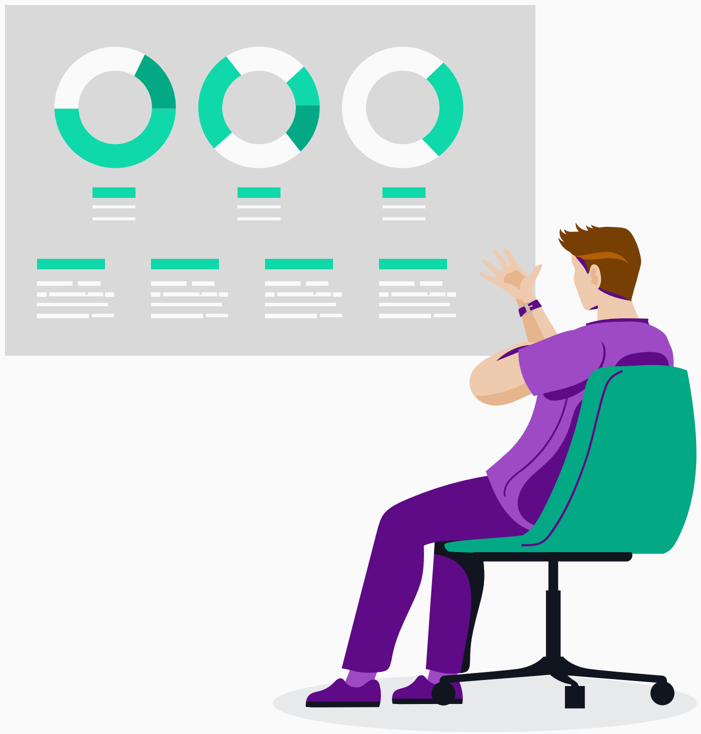
03
Data Mining Models

04
Machine Learning methods

05
Conclusions

01

Descripció de les dades originals



Dades originals

Extretes pàgina web: UCI Machine Learning Repository
Títol: **Predict students' dropout and academic success** - predicció de l'èxit o abandonament acadèmic dels estudiants.
Url:

	A	B	C	D	E	F	G	H	I	J
1	Marital status	Application mode	Application order	Course	Daytime/evening attendance	Previous qualification	Previous qualification (grade)	Nacionality	Mother's qualification	Father's qualification
2	1	17	5	171	1	1	122.0	1	19	12
3	1	15	1	9254	1	1	160.0	1	1	3
4	1	1	5	9070	1	1	122.0	1	37	37
5	1	17	2	9773	1	1	122.0	1	38	37
6	2	39	1	8014	0	1	100.0	1	37	38

Dataset de 4424 exemples amb 36 atributs + l'objectiu

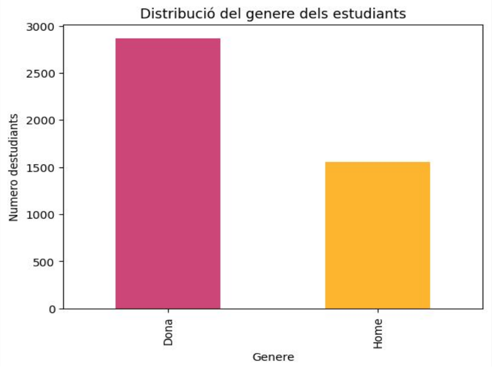
Dades originals

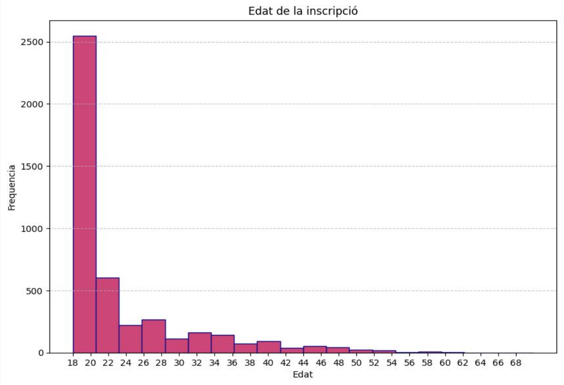
Atributs:

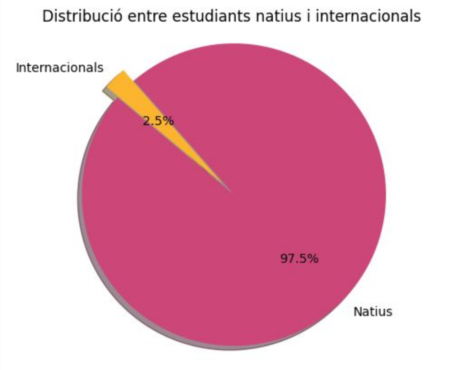
- Marital Status (Estat Civil)
- Application Mode (Mode de Sol·licitud)
- Application Order (Ordre de Sol·licitud)
- Course (Curs-Grau universitari)
- Daytime/Evening Attendance (Assistència Dia/Vespre)
- Previous Qualification (Qualificació Prèvia)
- Previous Qualification (Grade) (Qualificació Prèvia - Nota)
- Nationality (Nacionalitat)
- Mother's Qualification (Qualificació de la Mare)
- Father's Qualification (Qualificació del Pare)
- Mother's Occupation (Ocupació de la Mare)
- Father's Occupation (Ocupació del Pare)
- Admission Grade (Nota d'Admissió)
- Displaced (Desplaçat)
- Educational Special Needs (Necessitats Educatives Especials)
- Debtor (Deutor)
- Tuition Fees Up to Date (Taxes Acadèmiques pagades)
- Gender (Gènere)
- Scholarship Holder (Becari)
- Age at Enrollment (Edat a l'Inscripció)
- International (Internacional)
- Curricular units 1st sem (credited) (Unitats Curriculars 1r semestre - Acreditades)
- Curricular units 1st sem (enrolled) (Unitats Curriculars 1r semestre - Matriculades)
- Curricular units 1st sem (evaluations) (Unitats Curriculars 1r semestre - Avaluacions)
- Curricular units 1st sem (approved) (Unitats Curriculars 1r semestre - Aprovades)
- Curricular units 1st sem (grade) (Unitats Curriculars 1r semestre - Nota)
- Curricular units 1st sem (without evaluations) (Unitats Curriculars 1r semestre - Sense Avaluacions)
- Curricular units 2nd sem (credited) (Unitats Curriculars 2n semestre - Acreditades)
- Curricular units 2nd sem (enrolled) (Unitats Curriculars 2n semestre - Matriculades)
- Curricular units 2nd sem (evaluations) (Unitats Curriculars 2n semestre - Avaluacions)
- Curricular units 2nd sem (approved) (Unitats Curriculars 2n semestre - Aprovades)
- Curricular units 2nd sem (grade) (Unitats Curriculars 2n semestre - Nota)
- Curricular units 2nd sem (without evaluations) (Unitats Curriculars 2n semestre - Sense Avaluacions)
- Unemployment Rate (Taxa d'Atur)
- Inflation Rate (Taxa d'Inflació)
- GDP (PIB)
- Target (Objectiu)

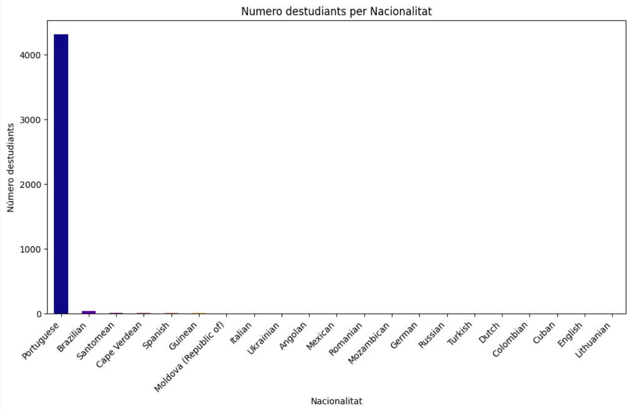
Dades originals

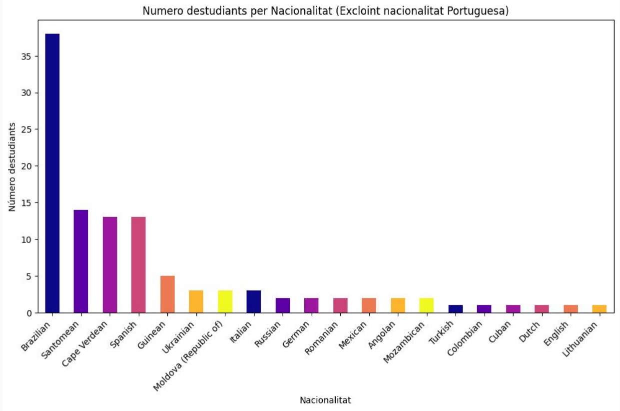
Representació gràfica:





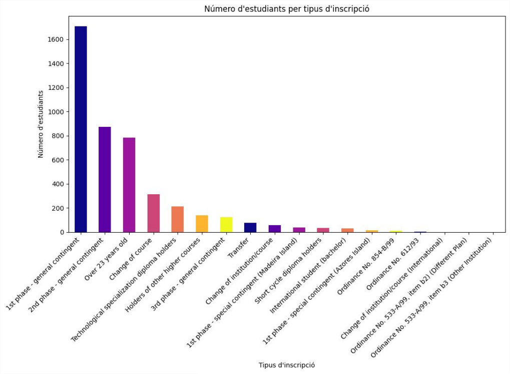


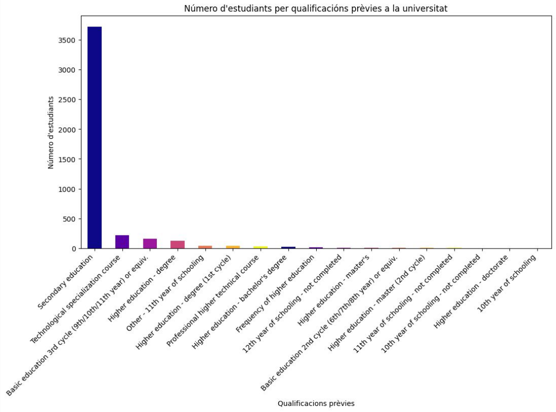


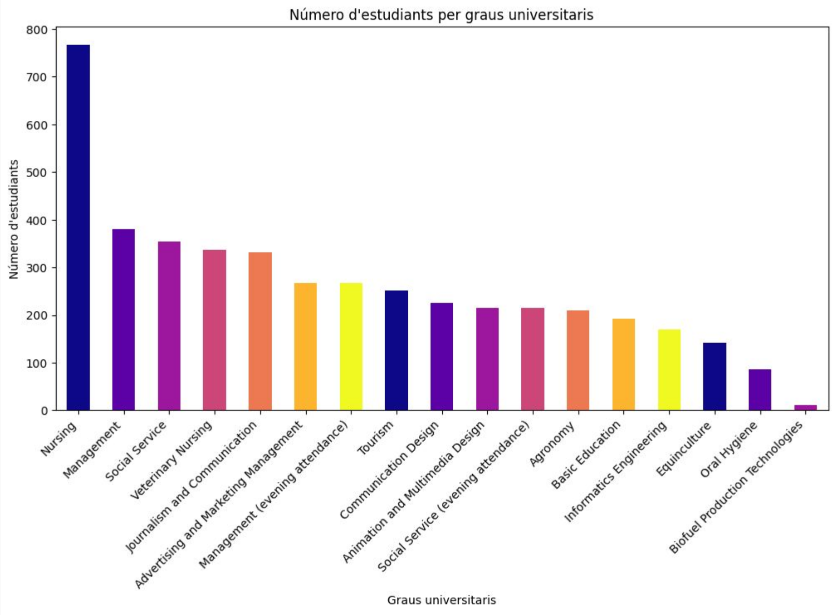


Dades originals

Representació gràfica:

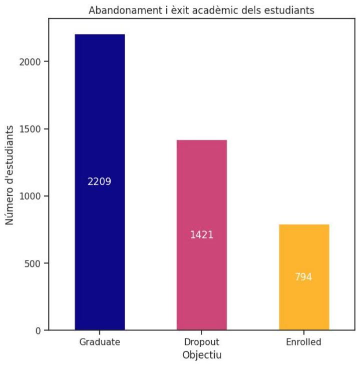


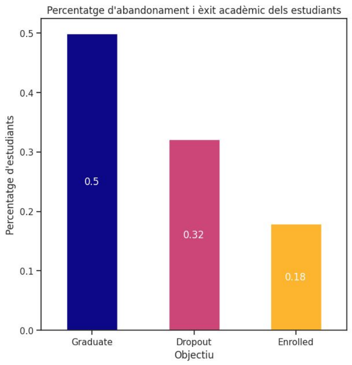


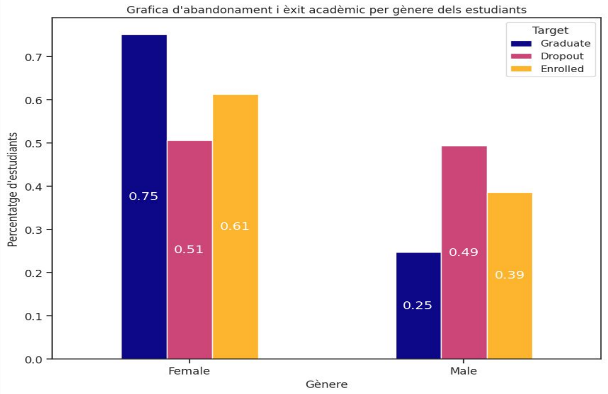


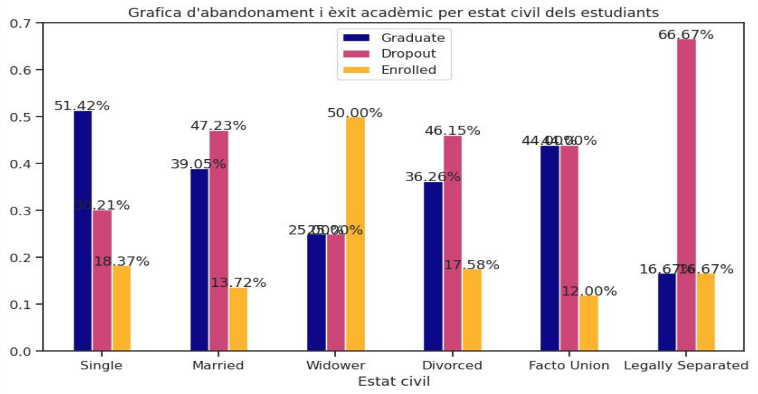
Dades originals

Representació gràfica amb l'objectiu:

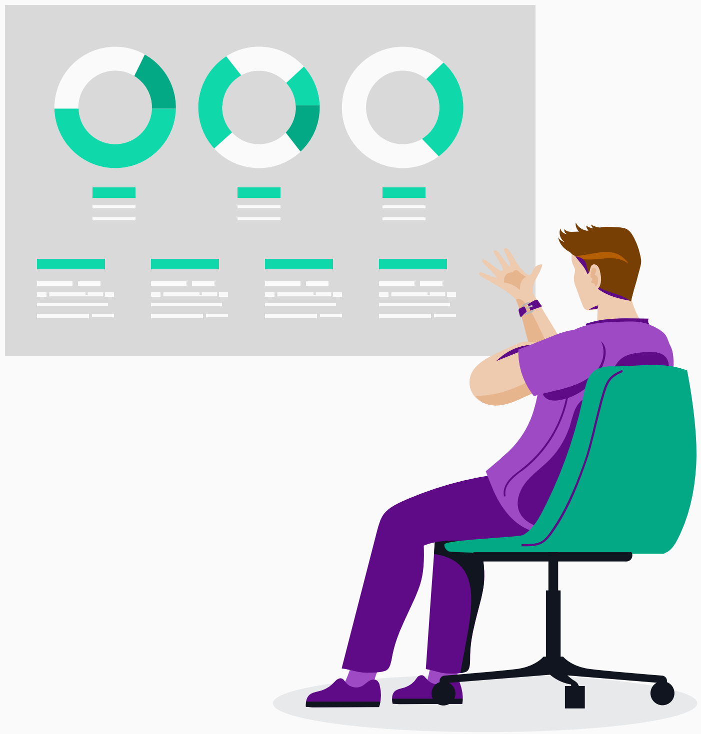








02
Preprocessament de les dades



Preprocessament de les dades

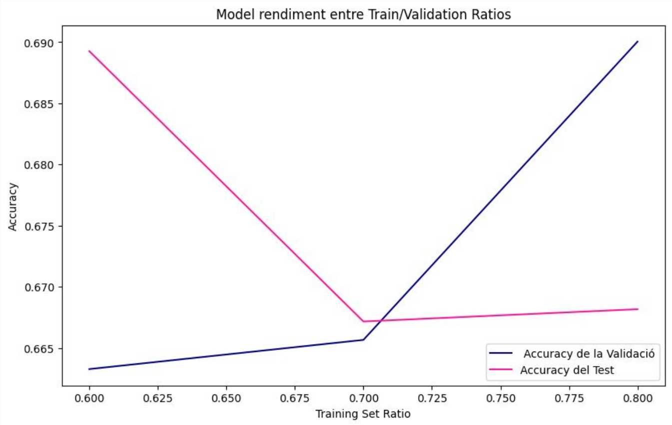
Valors Buits:

```
Valors buits dels atributs:
0

Valors buits del objectiu:
0
```

Files Duplicades:

```
Numero de files duplicades:
Marital status                                   0.0
Application mode                                 0.0
Application order                                0.0
Course                                           0.0
Daytime/evening attendance\t                     0.0
Previous qualification                           0.0
Previous qualification (grade)                   0.0
Nacionality                                      0.0
Mother's qualification                           0.0
Father's qualification                           0.0
Mother's occupation                              0.0
Father's occupation                              0.0
Admission grade                                  0.0
Displaced                                        0.0
Educational special needs                        0.0
Debtor                                           0.0
Tuition fees up to date                          0.0
Gender                                           0.0
Scholarship holder                               0.0
Age at enrollment                                0.0
International                                    0.0
Curricular units 1st sem (credited)              0.0
Curricular units 1st sem (enrolled)              0.0
Curricular units 1st sem (evaluations)           0.0
Curricular units 1st sem (approved)              0.0
Curricular units 1st sem (grade)                 0.0
Curricular units 1st sem (without evaluations)   0.0
Curricular units 2nd sem (credited)              0.0
Curricular units 2nd sem (enrolled)              0.0
Curricular units 2nd sem (evaluations)           0.0
Curricular units 2nd sem (approved)              0.0
Curricular units 2nd sem (grade)                 0.0
Curricular units 2nd sem (without evaluations)   0.0
Unemployment rate                                0.0
Inflation rate                                   0.0
GDP                                              0.0
Target                                           0.0
```

Diferents divisions del dataset:



Quedar: **80% entrenament 20% test**

Pàgina web:

Are there recommended data splits?
The dataset was used, in our project, with a data split of 80% for training and 20% for test.

Objectiu: Categòric a numeral:

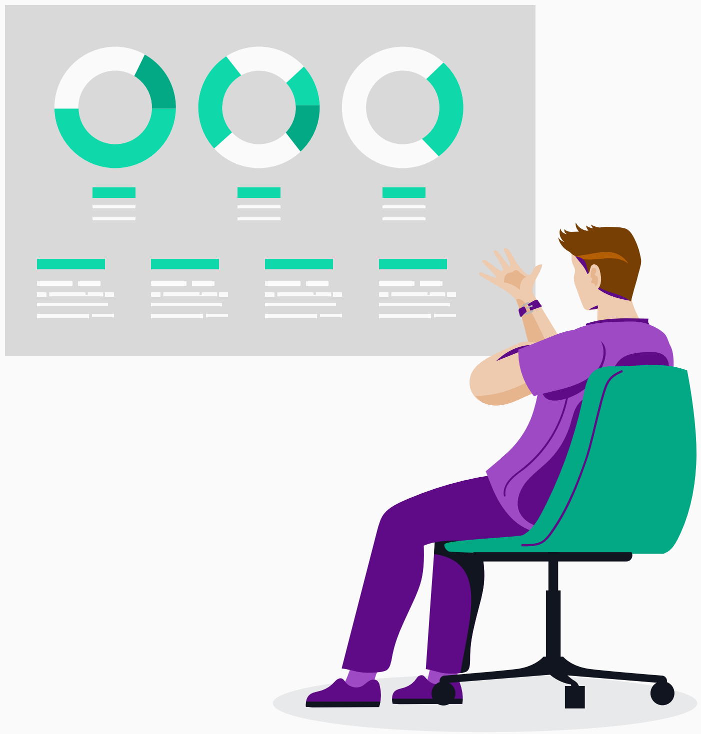
```
Abans treure estudiants que actualment esta estudien:
Graduate    2209
Dropout     1421
Enrolled     794
Name: Target, dtype: int64
Estudiants totals 4424

Despres de treure els estudiants que actualment estudien:
Graduate    2209
Dropout     1421
Name: Target, dtype: int64
Estudiants totals 3630

----------
Canviar el target d'un atribut categoric a numeral:
1    2209
0    1421
Name: Target, dtype: int64
Estudiants totals 3630

--> 1 son els estudiants que s'han graduat i 0 els que no ho han fet
```

Pàgina web:

Was there any data preprocessing performed?
We performed a rigorous data preprocessing to handle data from anomalies, unexplainable outliers, and missing values.

Has Missing Values?
No

03
Data Mining Models



Data Mining Models: Single Fold CV

Dividir una sola vegada dataset: Model **RandomForestClassifer**
Optimitzar amb **GridSearchCV** : trobar màxima profunditat

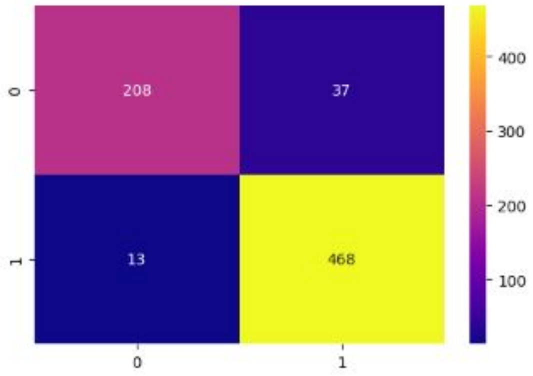
```
#Grid busqueda per cassificador validation
grid_search = GridSearchCV(RandomForestClassifier(n_estimators=100, random_state=20), param_grid, cv=5)
grid_search.fit(X_train, y_train)
```

```
Millor  max_depth trobada: 15
```

```
#Classificadors Random Forest Classifier
rf = RandomForestClassifier(n_estimators=50,max_depth=grid_search.best_params_['max_depth'],
,min_samples_split=4, min_samples_leaf=2, random_state=20)
```

```
INFORME MODEL SINGLE FOLD CV

              precision    recall  f1-score   support

           0       0.94      0.85      0.89       245
           1       0.93      0.97      0.95       481

    accuracy                           0.93       726
   macro avg       0.93      0.91      0.92       726
weighted avg       0.93      0.93      0.93       726
```

MATRIU DE CONFUSIÓ




```
RESULTATS ACCURACY
Accuracy:  0.931129476584022
Training Accuracy (entrenat): 0.9755509641873278
Testing Accuracy (test): 0.931129476584022
```

Data Mining Models: K-Fold CV

Dividir una k vegades dataset: Model **KNeighborsClassifier**
Optimitzar amb **GridSearchCV** : trobar valor millor de k

```
#Trobar millor parametre n_neighbors pel model de kfold cross validation
param_grid = {'n_neighbors': range(1, 30)}
knn = KNeighborsClassifier()
grid_search = GridSearchCV(knn, param_grid, cv=5)
```

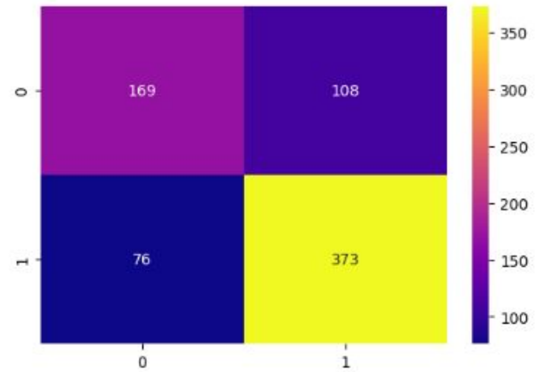
```
Millor número n_neighors: 8
```

```
# K-Fold Cross-Validation
for train_index, test_index in kf.split(X_train_full, y_train_full):
    X_train, X_val = X_train_full.iloc[train_index], X_train_full.iloc[test_index]
    y_train, y_val = y_train_full.iloc[train_index], y_train_full.iloc[test_index]

    #Model amb el nombre n_neighbors millor trobat
    model = KNeighborsClassifier(n_neighbors=grid_search.best_params_['n_neighbors'])

    model.fit(X_train, y_train)
    y_pred = model.predict(X_val) #Prediccio per cada fold

    # Calculate metrics
    accuracies.append(accuracy_score(y_val, y_pred))
    precisions.append(precision_score(y_val, y_pred, average='weighted'))
    recalls.append(recall_score(y_val, y_pred, average='weighted'))
    f1_scores.append(f1_score(y_val, y_pred, average='weighted'))
```

INFORME MODEL K FOLD CV

	precision	recall	f1-score	support
0	0.69	0.61	0.65	277
1	0.78	0.83	0.80	449
accuracy			0.75	726
macro avg	0.73	0.72	0.72	726
weighted avg	0.74	0.75	0.74	726

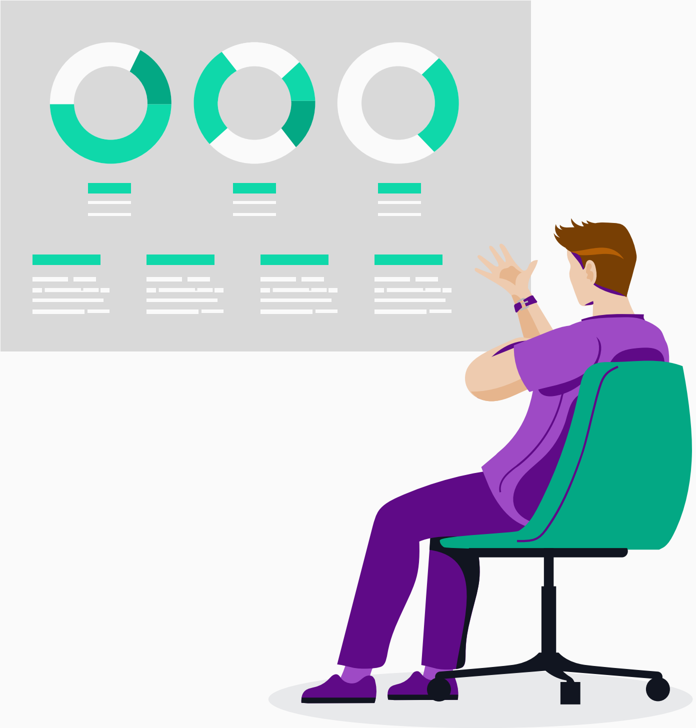
MATRIU DE CONFUSIÓ



MITJANES DE CADA FOLD
Average Accuracy: 0.7623995489346548
Average Precision: 0.7597992879252166
Average Recall: 0.7623995489346548
Average F1 Score: 0.7589954175837326

04

Machine Learning methods



Machine Learning methods : Naïve Bayes

Model **Naïve Bayes**

Optimitzar amb **GridSearchCV** : trobar millor valor var_smothing i el threshold

```
param_grid_nb = {'var_smoothing': np.logspace(0, -9, num=100)}

#Stratified K-Fold per cross validation
cv = StratifiedKFold(n_splits=10, random_state=42, shuffle=True)

# Model Gaussian Naive Bayes
gnb = GaussianNB()

grid_search_nb = GridSearchCV(estimator=gnb, param_grid=param_grid_nb, cv=cv, scoring='accuracy')
```

```
Millor var_smoothing: {'var_smoothing': 1e-08} Accuracy: 0.8421487603305785
```

```
gnb = GaussianNB(var_smoothing=grid_search_nb.best_params_['var_smoothing'])
gnb.fit(X_train, y_train)

# Prediccions per la CLASSE 1
probs = gnb.predict_proba(X_test)[:, 1]

# Define thresholds to search for the optimal threshold
thresholds = np.linspace(0, 1, 101)

best_threshold = 0
best_f1_score = 0

#Buscar millor threshold
for threshold in thresholds:
    preds = filterp(threshold, probs)
    f1 = f1_score(y_test, preds, pos_label=1)
    if f1 > best_f1_score:
        best_f1_score = f1
        best_threshold = threshold

print(f"Best threshold: {best_threshold} with F1-score: {best_f1_score}")
```

```
Best threshold: 0.98 with F1-score: 0.8974943052391801
```

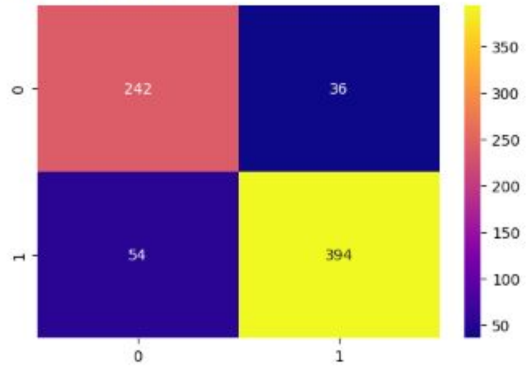
Predicció valors optimitzats:

```
#Millor threshold trobat, aplicar en prediccions
final_preds = filterp(best_threshold, probs)
```

INFORME MODEL NAIVE BAYES

	precision	recall	f1-score	support
0	0.82	0.87	0.84	278
1	0.92	0.88	0.90	448
accuracy			0.88	726
macro avg	0.87	0.87	0.87	726
weighted avg	0.88	0.88	0.88	726

MATRIU DE CONFUSIÓ



RESULTATS ACCURACY
```
Accuracy:  0.8760330578512396
Training Accuracy (entrenat): 0.8440082644628l
Testing Accuracy (test): 0.8553719008264463
```

Machine Learning methods : KNN

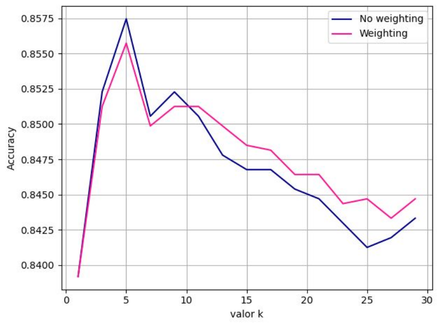
Model **K-nearest Neighbors**

Optimitzar amb **GridSearchCV** : trobar millor valor n_neighbors i el weight

```
accuracies = []
for k in range(1,10):
    knn = KNeighborsClassifier(n_neighbors=k)
    cv_scores = cross_val_score(knn, X=X_train, y=y_train, cv=10)
    accuracies.append(np.mean(cv_scores))
    print(f"Accuracy {k} neighbours: {np.mean(cv_scores)}")
```

```
Accuracy 1 neighbours: 0.8391918473752813
Accuracy 2 neighbours: 0.8106067069558005
Accuracy 3 neighbours: 0.8522810759568669
Accuracy 4 neighbours: 0.847121696883517
Accuracy 5 neighbours: 0.8574534897499705
Accuracy 6 neighbours: 0.860543903306079
Accuracy 7 neighbours: 0.8505545680767865
Accuracy 8 neighbours: 0.8591669629102974
Accuracy 9 neighbours: 0.8522751510842518
```

Representació k amb el pes:



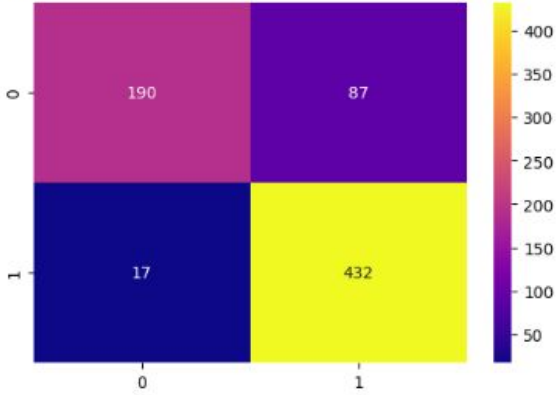

```
knc = nb.KNeighborsClassifier()
clf = GridSearchCV(knc, param_grid=params,cv=10,n_jobs=-1)
```

```
Millors Parametres: {'n_neighbors': 5, 'weights': 'uniform'} Accuracy= 0.8574534897499705
```

INFORME MODEL KNN

	precision	recall	f1-score	support
0	0.92	0.69	0.79	277
1	0.83	0.96	0.89	449
accuracy			0.86	726
macro avg	0.88	0.82	0.84	726
weighted avg	0.86	0.86	0.85	726

MATRIU DE CONFUSIÓ



RESULTATS ACCURACY

Accuracy 0.8567493112947658

Machine Learning methods : Decision Tree

Model **Decision Trees**

Optimitzar amb **GridSearchCV** : trobar millors paràmetres

```
param_grid = {
    'criterion': ['entropy', 'gini'],
    'max_depth': [None,2, 5, 10, 20, 30],
    'min_impurity_decrease': [0, 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5],
    'min_samples_split': [2, 5, 10, 15, 20, 25],
    'min_samples_leaf': [1, 2, 4],
}
```

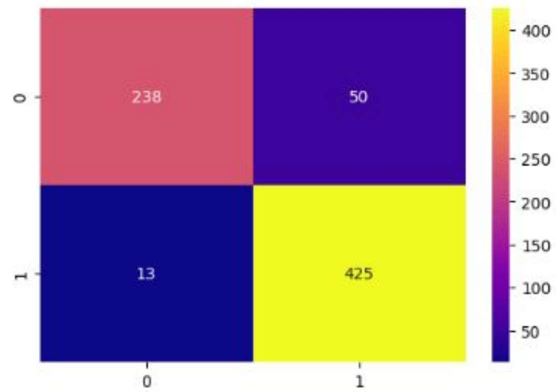
```
Millors Parameters: {'criterion': 'entropy', 'max_depth': 5, 'min_impurity_decrease': 0, 'min_samples_leaf': 1,
 'min_samples_split': 25}  Accuracy: 0.8922232773458365
```

Representació de l'arbre:

```
best_tree = grid_search.best_estimator_ #millors parametres calculats
best_tree.fit(X_train, y_train)

plt.figure(figsize=(200, 100))
tree.plot_tree(best_tree, filled=True, rounded=True, feature_names=list(Xn.columns.values))
plt.title("Arbre de Decisió ")
plt.show()
```

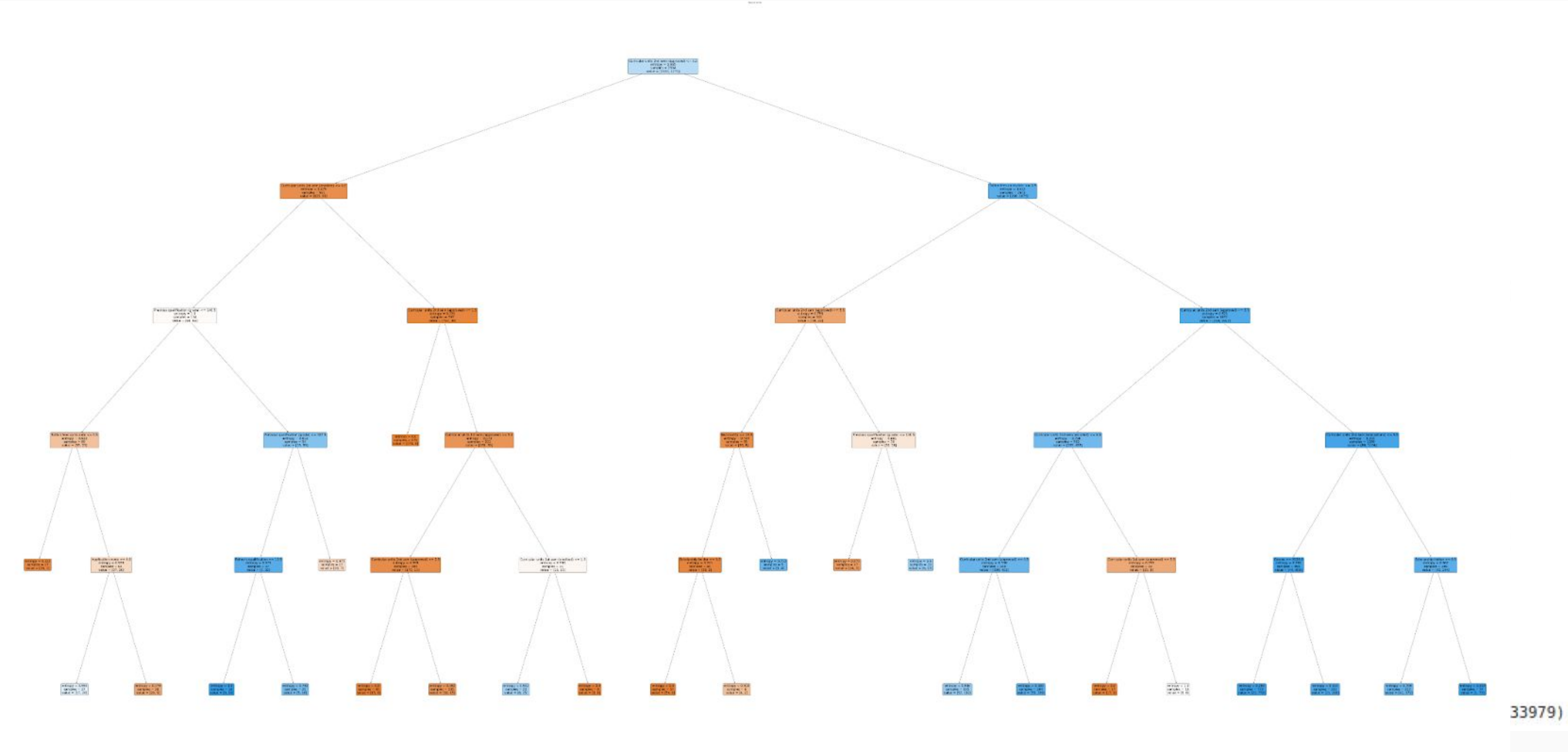
```
INFORME MODEL DECISION TREE

              precision    recall  f1-score   support

           0       0.95      0.83      0.88       288
           1       0.89      0.97      0.93       438

    accuracy                           0.91       726
   macro avg       0.92      0.90      0.91       726
weighted avg       0.92      0.91      0.91       726
```

MATRIU DE CONFUSIÓ




```
RESULTATS MODEL
Accuracy  0.9132231404958677
Interval of confidence: (0.8906489244766083, 0.9320104152633979)
```

Machine Learning methods : Decision Tree



Machine Learning methods : SVM

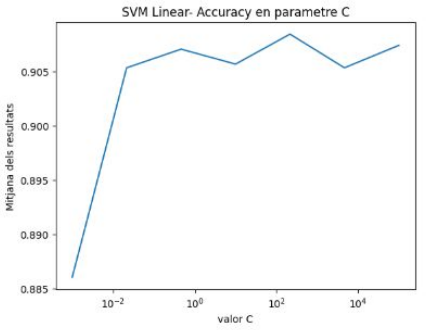
Model **SVM**

Optimitzar amb **GridSearchCV** : trobar millors paràmetres depenen del kerenel

SVM lineal:

Trobar el valor de C

```
#Trobar cs amb grid search
Cs = np.logspace(-3, 5, num=7, base=10.0)
param_grid_linear = {'C': Cs, 'kernel': ['linear']}
grid_search_linear = GridSearchCV(SVC(), param_grid_linear, cv=10)
grid_search_linear.fit(X_train, y_train)
```

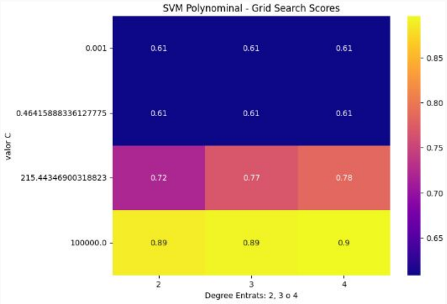


Millor C 10^2 a accuracy 90%

SVM polynomial:

Trobar el valor de C i degree

```
#Trobar cs amb grid search + degree
Cs = np.logspace(-3, 5, num=4, base=10.0)
param_grid_poly = {'C': Cs, 'kernel': ['poly'], 'degree': [2, 3,4], 'gamma': ['scale']}
grid_search_poly = GridSearchCV(SVC(), param_grid_poly, cv=3, n_jobs=-1, verbose=1)
```

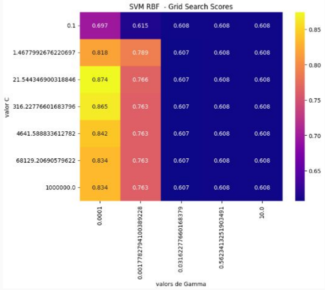


Millor C 100000 a qualsevol degree a accuracy 89%

SVM rbf:

Trobar el valor de C i gammas

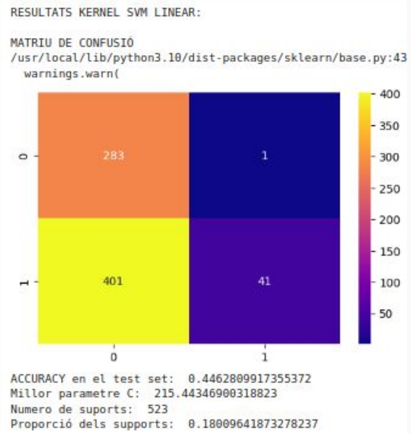
```
#Trobar cs + gammas amb gridsearch
gammas = np.logspace(-4, 1, num=5, base=10.0)
Cs = np.logspace(-1, 6, num=7, base=10.0)

param_grid_rbf = {'C': Cs, 'gamma': gammas, 'kernel': ['rbf']}
grid_search_rbf = GridSearchCV(SVC(), param_grid_rbf, cv=5)
```



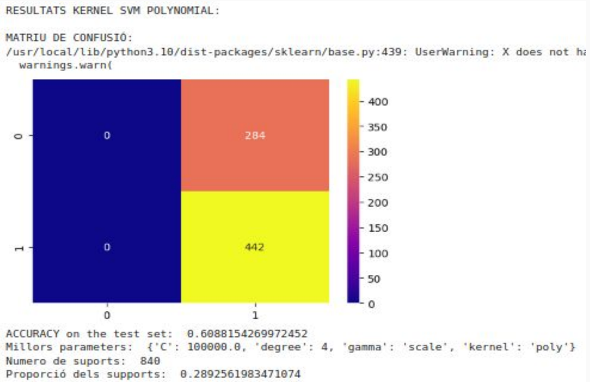
Millor C a 21 i gamma a 0,0001 a accuracy 87%

Machine Learning methods : SVM

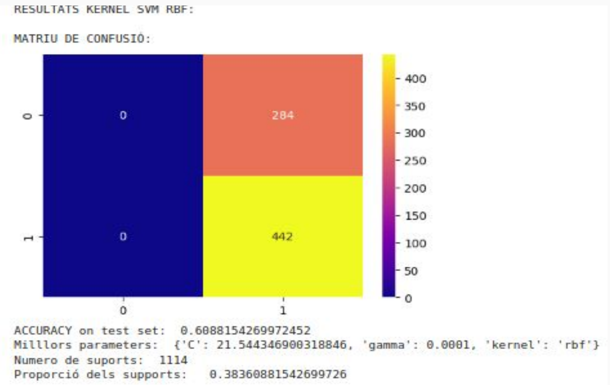
SVM lineal:



SVM polynomial:



SVM rbf:



- **Accuracy dels kernels:**
  - Linear: 44,6%
  - Poly: 60%
  - Rbf: 60%
- **Paràmetres per C:**
  - Linear: 215.44
  - Poly: 100000.0
  - Rbf: 21.54
- **Número de suports / Proporció dels suports :**
  - Linear: 523 0,180
  - Poly: 840 0,289
  - Rbf: 1114 0,383

Millors Resultats Kernel SVM:

polynomial

Machine Learning methods : Meta Learning algorithm

Classificadors per els algorismes:

Naïve Bayes:

```
cv = 50
meta_nbl = GaussianNB()
scores = cross_val_score(meta_nbl, X, y, cv=cv, scoring='accuracy')
print("Accuracy: %0.3f de Naive Bayes" % scores.mean())
```

```
Accuracy: 0.839 de Naive Bayes
```

kNN:

```
params = {'n_neighbors': list(range(1, 30, 2)), 'weights': ('distance', 'uniform')}
knc = KNeighborsClassifier()
clf_knn = GridSearchCV(knc, param_grid=params, cv=cv, n_jobs=-1)
clf_knn.fit(X, y)
print("Millors parametres per KNN =", clf_knn.best_params_, "Accuracy =", clf_knn.best_score_)

#Utilitzar els parametres trobats per knn
meta_knn1 = KNeighborsClassifier(n_neighbors=clf_knn.best_params_['n_neighbors'], weights=clf_knn.best_params_['weights'])
scores = cross_val_score(meta_knn1, X, y, cv=cv, scoring='accuracy')
print("Accuracy: %0.3f  de KNN " % scores.mean())
```

```
Millors parametres per KNN = {'n_neighbors': 5, 'weights': 'distance'} Accuracy = 0.7710692541856925
Accuracy: 0.771  de KNN
```

Decision Tree:

```
meta_dtl = DecisionTreeClassifier(criterion='entropy')
scores = cross_val_score(meta_dtl, X, y, cv=cv, scoring='accuracy')
print("Accuracy: %0.3f de Decision Tree" % scores.mean())
```

```
Accuracy: 0.862 de Decision Tree
```

Majority Voting:

Hard Voting:

```
Accuracy: 0.771 [Majority Voting: Hard voting]
```

Soft Voting:

```
Accuracy: 0.840 [Weighted Voting: Soft voting]
```

Millor majority voting: **soft voting** a **84%**

Machine Learning methods : Meta Learning algorithm

Bagging:

Bagging Classifier amb DecisionTreeClassifier

```
RESULTATS DECISION TREE BAGGING - ESTIMADORS:
 Accuracy: 0.8509550989345509   - num estimadors: 1
 Accuracy: 0.820955098934551    - num estimadors: 2
 Accuracy: 0.8889954337899542   - num estimadors: 5
 Accuracy: 0.8936757990867581   - num estimadors: 10
 Accuracy: 0.9000114155251141   - num estimadors: 20
 Accuracy: 0.900818112633181    - num estimadors: 50
 Accuracy: 0.9021917808219178   - num estimadors: 100
 Accuracy: 0.9038432267884322   - num estimadors: 200

 Millor num estimadors 200 amb un Accuracy de: 0.9038432267884322
```

Bagging Classifier amb DecisionTreeClassifier amb max features

```
RESULTATS DECISION TREE BAGGING - ESTIMADORS i MAX FEATURES:
Accuracy: 0.7814726027397261   - num estimadors: 1
Accuracy: 0.7651369863013697   - num estimadors: 2
Accuracy: 0.8589459665144596   - num estimadors: 5
Accuracy: 0.8806887366818874   - num estimadors: 10
Accuracy: 0.8928919330289194   - num estimadors: 20
Accuracy: 0.9068797564687975   - num estimadors: 50
Accuracy: 0.9052283105022831   - num estimadors: 100
Accuracy: 0.9049619482496195   - num estimadors: 200

Millor num estimadors 50 amb un Accuracy de: 0.9068797564687975  amb maxim features: 0.35
```

Millor bagging: **max característiques** amb **90,6%**

RandomForest i Extra Trees

Random Forest

```
RESULTATS RANDOM FOREST:
 Accuracy: 0.8360881542699724   - num arbres: 1
 Accuracy: 0.8201101928374657   - num arbres: 2
 Accuracy: 0.8820936639118457   - num arbres: 5
 Accuracy: 0.8914600550964187   - num arbres: 10
 Accuracy: 0.9024793388429752   - num arbres: 20
 Accuracy: 0.9068870523415977   - num arbres: 50
 Accuracy: 0.9049586776859504   - num arbres: 100
 Accuracy: 0.9055096418732782   - num arbres: 200

 Millor num d'arbre 50 amb un Accuracy de: 0.9068870523415977
```

Extra Trees:

```
RESULTATS EXTRA TREES:
 Accuracy: 0.8286501377410469   - num arbres: 1
 Accuracy: 0.8074380165289256   - num arbres: 2
 Accuracy: 0.8887052341597796   - num arbres: 5
 Accuracy: 0.8922865013774105   - num arbres: 10
 Accuracy: 0.9079889807162533   - num arbres: 20
 Accuracy: 0.9088154269972453   - num arbres: 50
 Accuracy: 0.9115702479338843   - num arbres: 100
 Accuracy: 0.9115702479338843   - num arbres: 200

 Millor num d'arbre 100 amb un Accuracy de: 0.9115702479338843
```

Millor arbre: **extra Trees** amb **91.1%**

Machine Learning methods : Meta Learning algorithm AdaBoost:

Adaboost

```
RESULTATS ADABOOST:
 Accuracy: 0.8680365296803652   - num estimadors: 1
 Accuracy: 0.8875875190258752   - num estimadors: 2
 Accuracy: 0.8906126331811264   - num estimadors: 5
 Accuracy: 0.8958523592085236   - num estimadors: 10
 Accuracy: 0.8988812785388127   - num estimadors: 20
 Accuracy: 0.8999809741248096   - num estimadors: 50
 Accuracy: 0.9052283105022831   - num estimadors: 100
 Accuracy: 0.9038546423135463   - num estimadors: 200

 Millor num d'estimadors 100 amb un Accuracy de: 0.9052283105022831
```

Adaboost amb maxima profunditat

```
RESULTATS ADABOOST - MAX DEPTH:
 Accuracy: 0.8977701674277018   - num estimadors: 1
 Accuracy: 0.892001522070015    - num estimadors: 2
 Accuracy: 0.8840182648401828   - num estimadors: 5
 Accuracy: 0.8727054794520547   - num estimadors: 10
 Accuracy: 0.861974885844749    - num estimadors: 20
 Accuracy: 0.8724010654490105   - num estimadors: 50
 Accuracy: 0.8856773211567733   - num estimadors: 100
 Accuracy: 0.898595890410959    - num estimadors: 200

 Millor num d'estimadors 200 amb un Accuracy de: 0.898595890410959 amb màxima profunditat:  5
```

Millor adaboost , **no máxima profunditat** amb **90,6%**

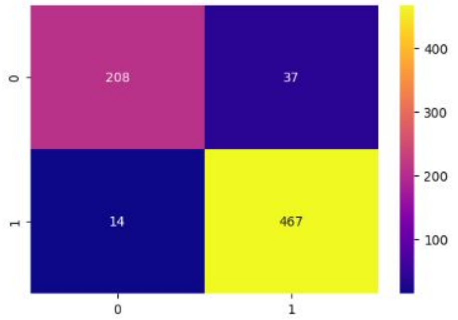
Millors Resultats meta Learning

Millor: **Extra Trees Classifier** amb **91.1%**

INFORME MODEL EXTRA TREES

	precision	recall	f1-score	support
0	0.94	0.85	0.89	245
1	0.93	0.97	0.95	481
accuracy			0.93	726
macro avg	0.93	0.91	0.92	726
weighted avg	0.93	0.93	0.93	726

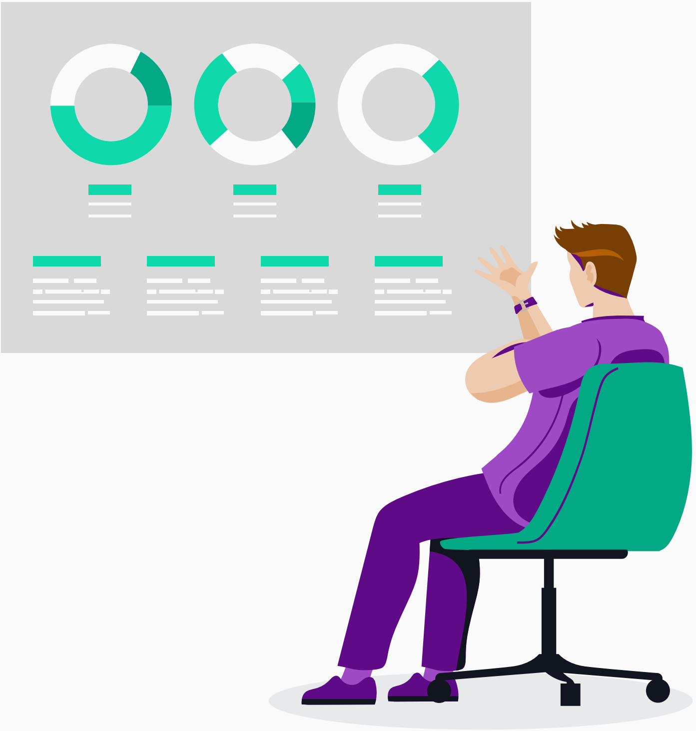
MATRIU DE CONFUSIÓ



RESULTATS MODEL
Accuracy: 0.9297520661157025

05

Conclusions



Conclusions:

Cross Validation	Accuracy	%
Single-Fold CV	0.931129476584022	93,1%
K-Fold CV	0.7623995489346548	76,2%

-millors models però mètode no representatiu
-pitjor resultat no generalitza bé

Model/Classificador	Accuracy	%
NaÏve Bayes	0.8760330578512396	87,6%
KNN	0.8567493112947658	85,6%
Decision Tree	0.9132231404958677	91,3%
SVM (polinominal)	0.8705234159779615	87,0%
Meta-learning (Extra Trees)	0.9297520661157025	92,2%

-intermedia, suposició independencia atributs correcte en el dataset
- intermedia, dades compactes,agrupació natural
- millors models, divisons clares facil de seguir categories
- intermig, relacions dades no sigui polinominal, igualment millor kernel svm
- millors models. decision tree millorats amb més diversitat

Conclusions:

Model/Classificador		precision	recall	f1-score	support
Naïve Bayes	0	0.82	0.87	0.84	278
	1	0.92	0.88	0.90	448
	accuracy			0.88	726
KNN	0	0.92	0.69	0.79	277
	1	0.83	0.96	0.89	449
	accuracy			0.86	726
Decision Tree	0	0.95	0.83	0.88	288
	1	0.89	0.97	0.93	438
	accuracy			0.91	726
SVM (polinominal)	0	0.81	0.81	0.81	245
	1	0.90	0.90	0.90	481
	accuracy			0.87	726
Meta-learning (Extra Trees)	0	0.94	0.85	0.89	245
	1	0.93	0.97	0.95	481
	accuracy			0.93	726

- equilibrat precision/recall bon model, inconvenient amb possibles independencies de atributs

-alta precisió predicció classe 0, dificultats classes menys representades

- alta precisió i recall, millor classe 1, bones relacions entre característiques

-rendiment equilibrat efectiu per separar les diferents classes, sobretot si els paràmetres s'optimitzen

- millor rendiment alta precisió i alt recall en les dues classes, ajuden les millores de meta learning

Millors mètode:

DECISION TREES i EXTRA TREES

Preguntes?

Moltes gràcies