Mitocaru Irina

343C1

Analiza problemei

Contents

[Introducere 2](#_Toc35723067)

[Analiza setului de date 2](#_Toc35723068)

[1. Dimensiune 2](#_Toc35723069)

[2. Clase 2](#_Toc35723070)

[3. Atribute 2](#_Toc35723071)

[Analiza clasificatorilor 3](#_Toc35723072)

[K-means 3](#_Toc35723073)

[1. Baseline 3](#_Toc35723074)

[2. Imbunatatiri 3](#_Toc35723075)

[3. Grafice intra-algoritmi 3](#_Toc35723076)

[Random Forest Classifier 3](#_Toc35723077)

[1. Baseline 3](#_Toc35723078)

[2. Imbunatatiri 3](#_Toc35723079)

[3. Grafice intra-algoritmi 4](#_Toc35723080)

[XGBoost Classifier 5](#_Toc35723081)

[1. Baseline 5](#_Toc35723082)

[2. Imbunatatiri 5](#_Toc35723083)

[3. Grafice intra-algoritmi 6](#_Toc35723084)

[SVM (SVC) Classifier 7](#_Toc35723085)

[1. Baseline 7](#_Toc35723086)

[2. Imbunatatiri 7](#_Toc35723087)

[3. Grafice intra-algoritmi 7](#_Toc35723088)

[Grafice inter-algoritmi 8](#_Toc35723089)

[1. Acuratete 8](#_Toc35723090)

[2. Precizie 8](#_Toc35723091)

[3. Recall 9](#_Toc35723092)

[4. F1 9](#_Toc35723093)

[5. Matricea de confuziune 10](#_Toc35723094)

# Introducere

Acest document ilustreaza atat analiza amanuntita a setului de date cu care se lucreaza, cat si analiza 4 modele folosite la clasificarea unui set de date dupa genul muzical.

# Analiza setului de date

### Dimensiune

Setul de date e impartit in 2000 de exemple pentru setul de invatare si 400 de exemple pentru setul de testare si validare. In total sunt 2400 de exemple.

### Clase

Facand o analiza a distributiei claselor, s-a observat ca toate 4 (Hip-Hop, Electronic, Folk si Rock) sunt impartite in mod egal, fiecare avand 600 de exemple, lucru util deoarece se evita fenomenul de overfitting.

### Atribute

#### Echonest.csv

* audio\_features: 8 atribute, valori numerice si fara NaN
* social\_features: 5 atribute, valori numerice si fara NaN
* ranks: 5 atribute, valori numerice si NaN
* metadata: 7 atribute, valori numerice, categorice si NaN
* temporal\_features: 224 atribute, valori numerice si fara NaN

#### Features.csv

* chroma\_cens
* chroma\_cqt
* chroma\_stft
* mfcc
* rmse
* spectral\_bandwidth
* spectral\_centroid
* spectral\_contrast
* spectral\_rolloff
* tonnetz
* zcr

Toate sunt in combinatie cu: 'kurtosis', 'max', 'mean', 'min', 'skew' si 'std', avand atat valori numerice, categorice, cat si NaN.

# Analiza clasificatorilor

## K-means

### Baseline

### Imbunatatiri

#### Hiperparametri

#### Preprocesare

#### Atribute

### Grafice intra-algoritmi

## Random Forest Classifier

### Baseline

RandomForestClassifier(random\_state=0)

### Imbunatatiri

#### Hiperparametri

RandomForestClassifier(n\_jobs=-1, n\_estimators=200, max\_depth=depth, random\_state=0)

S-au incercat urmatorii hiper-parametri:

**n\_estimators**:

* (100, 500): acuratete cu 3% mai mare
* >= 500: acuratete la metrica f1\_score cu 2% mai mare, iar la restul cu 3% mai mare

Astfel, s-a ales valoarea 200.

**max\_depth**:

* None: acurarete la fel ca la baseline
* > ¾ din numarul de atribute: acuratete cu 1% mai mare
* < ¾ din numarul de atribute: acuratete mai slaba ca la baseline
* = ¾ din numarul de atribute: acuratete cu 4% mai mare

S-a observat ca o combinatie de numar de estimatori mic (100) cu o limita de adancime mica (1/2 din numarul de atribute) rezulta in performante proaste, deoarece algoritmul nu ia in considerare toate atributele (sau mai multe atribute) care aduc un plus acuratetei.

S-a ales max\_depth = 3/4 din numarul de atribute.

**criterion**:

* ‘gini’: acurarete la fel ca la baseline
* ‘entropy’: acuratete mai slaba ca la baseline

S-a ales ‘gini’.

**n\_jobs**:

S-a ales -1 pentru a folosi tot CPU-ul la maxim, daca este nevoie.

**min\_samples\_split & min\_samples\_leaf:**

* >2: nu avem imbunatatiri

S-a ales valoarea default: 1.

**max\_features:**

* int: nu avem imbunatatiri
* alte optiuni mai putin default: performante mai proaste

S-a ales default: auto.

**max\_leaf\_nodes & min\_impurity\_decrease & min\_impurity\_split & bootstrap & cc\_alpha:**

* alte optiuni mai putin default: performante mai proaste

S-a ales default: None.

**random\_state:**

* S-a ales valoarea 0 pentru a face modelul determinist.

#### Preprocesare

depth = int(len(x\_train.columns) \* 3 / 4)

x\_train = SelectKBest(f\_classif, k=depth).fit\_transform(x\_train, y\_train)

Pentru preprocesarea setului de invatare, s-a incercat:

* Standardizare: fara rezultat
* Normalizare: fara rezultat
* SelectKBest: singura metoda care a adus un plus de 2-3% la perfomanta. Si aici s-a ales un numar de atribute = ¾ din numarul total

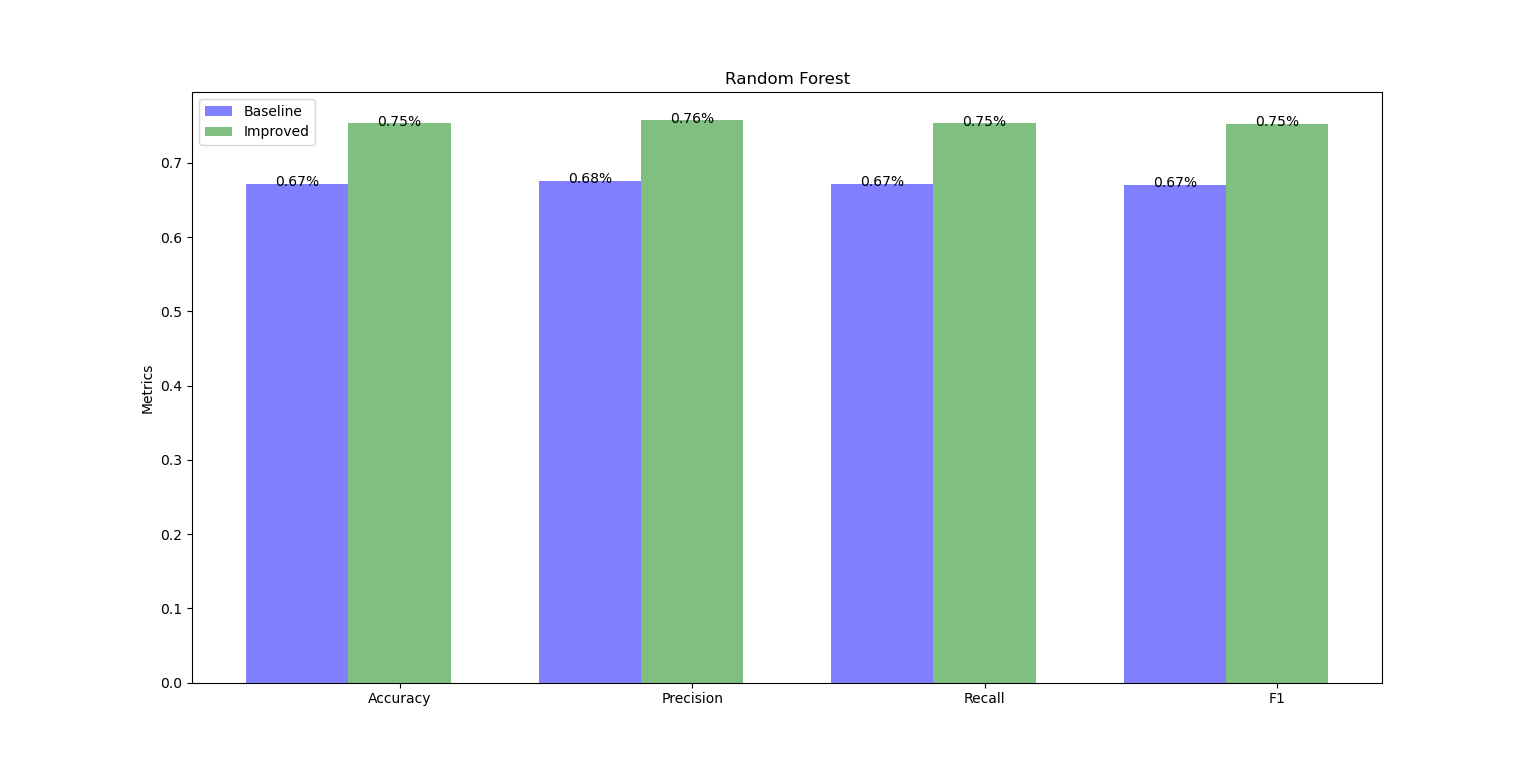
#### Atribute

S-a incercat combinatia dintre audio\_features cu fiecare din celelalte din categoria Echonest si cateva din Features. Singura combinatie care a adus un plus performantei a fost:

‘audio\_features’ cu ‘temporal\_features’

### Grafice intra-algoritmi

* Acuratete: 0.67% baseline, 0.75% varianta imbunatatita
* Precizie: 0.68% baseline, 0.76% varianta imbunatatita
* Recall: 0.67% baseline, 0.75% varianta imbunatatita
* F1: 0.67% baseline, 0.75% varianta imbunatatita



## XGBoost Classifier

### Baseline

XGBClassifier(random\_state=0)

### Imbunatatiri

#### Hiperparametri

XGBClassifier(n\_jobs=-1, random\_state=0, subsample=0.9, objective='multi:softmax', num\_class=4, learning\_rate=0.1)

S-au incercat urmatorii hiper-parametri:

**learning\_rate**:

* = 0.1: performanta imbunatatita cu 7%
* > 0.1: performanta imbunatatita cu 5%
* < 0.1: performanta intre baseline si baseline + 5%

S-a ales 0.1.

**max\_depth:**

* None: acurarete la fel ca la baseline
* > ¾ din numarul de atribute: acurarete la fel ca la baseline
* < ¾ din numarul de atribute: acuratete mai slaba ca la baseline
* = ¾ din numarul de atribute: acurarete la fel ca la baseline

S-a ales default: None.

**booster:**

* ’gbtree’: acurarete la fel ca la baseline
* ’gblinear’: acuratete mai slaba ca la baseline
* ’dart’: acurarete la fel ca la baseline

S-a ales default: None

**num\_parallel\_tree:**

S-au incercat mai multe valori, toate crescand complexitatea temporala si neaducand un plus de performanta. S-a ales default: None

**n\_estimators:**

* >= 100: acurarete la fel ca la baseline

S-a ales default: 100

**importance\_type:**

S-au incercat mai multe valori, fara rezultat. S-a ales default: None

**subsampling:**

* 1: acurarete la fel ca la baseline
* < 0.9: acuratete cu 1-2% mai mare
* = 0.9: acuratete cu 2% mai mare

S-a ales 0.9

**objective:**

S-a ales valoarea 'multi:softmax' deoarece modelul foloseste by default binary:logistic, care se foloseste la clasificator binar.

**num\_class:**

S-a ales valoarea 4 deoarece acesta este numarul de clase a setului nostru de date. Hiper-parametrul se foloseste in combinatie cu ’objective’.

#### Preprocesare

Pentru preprocesarea setului de invatare, s-a incercat:

* Standardizare: fara rezultat
* Normalizare: fara rezultat
* SelectKBest: fara rezultat

Astfel, nu s-a efectuat nicio preprocesare a setului de invatare pentru acest model.

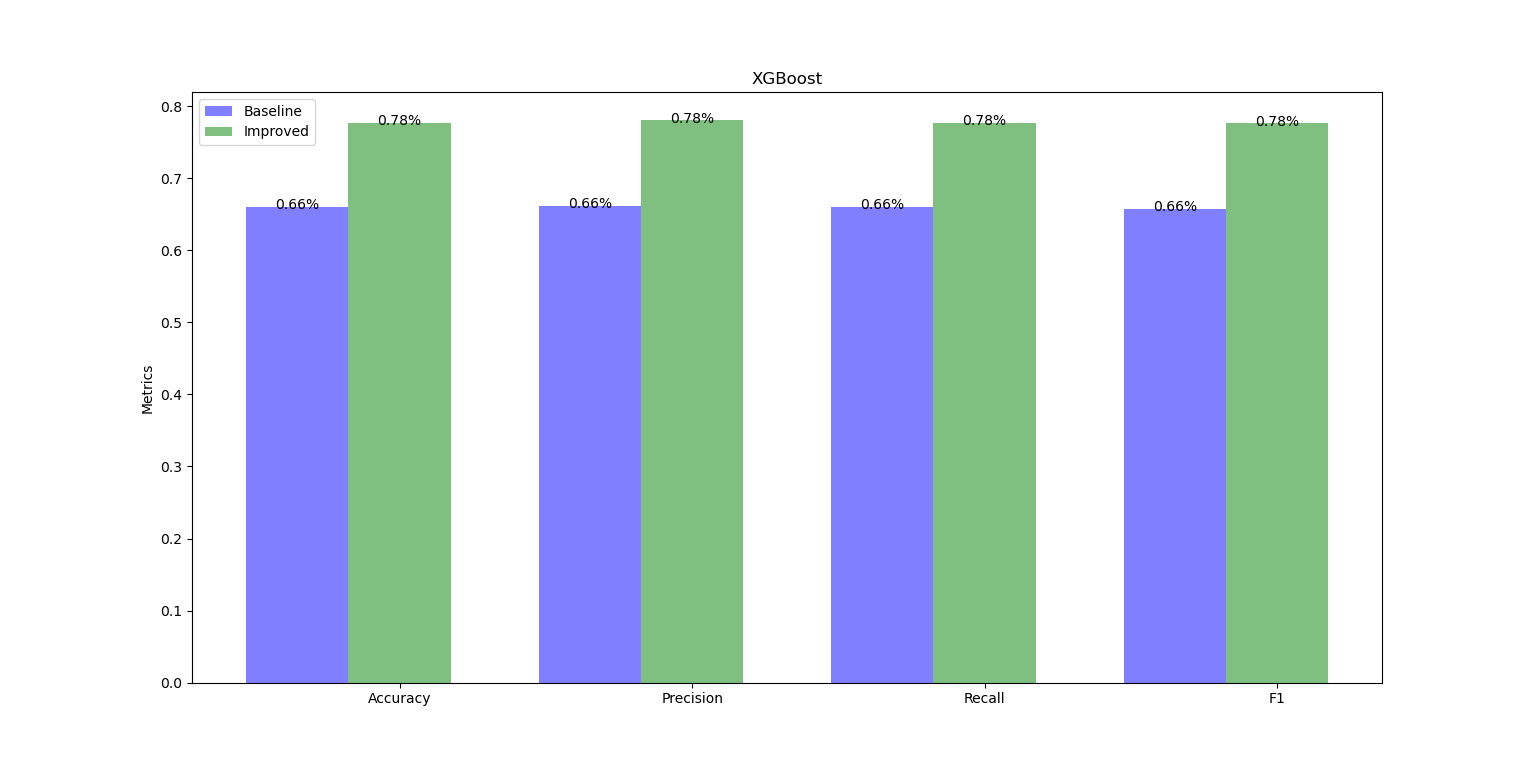
#### Atribute

S-a incercat combinatia dintre audio\_features cu fiecare din celelalte din categoria Echonest si cateva din Features. Singura combinatie care a adus un plus performantei a fost:

‘audio\_features’ cu ‘temporal\_features’

### Grafice intra-algoritmi

* Acuratete: 0.66% baseline, 0.78% varianta imbunatatita
* Precizie: 0.66% baseline, 0.78% varianta imbunatatita
* Recall: 0.66% baseline, 0.78% varianta imbunatatita
* F1: 0.66% baseline, 0.78% varianta imbunatatita



## SVM (SVC) Classifier

### Baseline

SVC(random\_state=0)

### Imbunatatiri

#### Hiperparametri

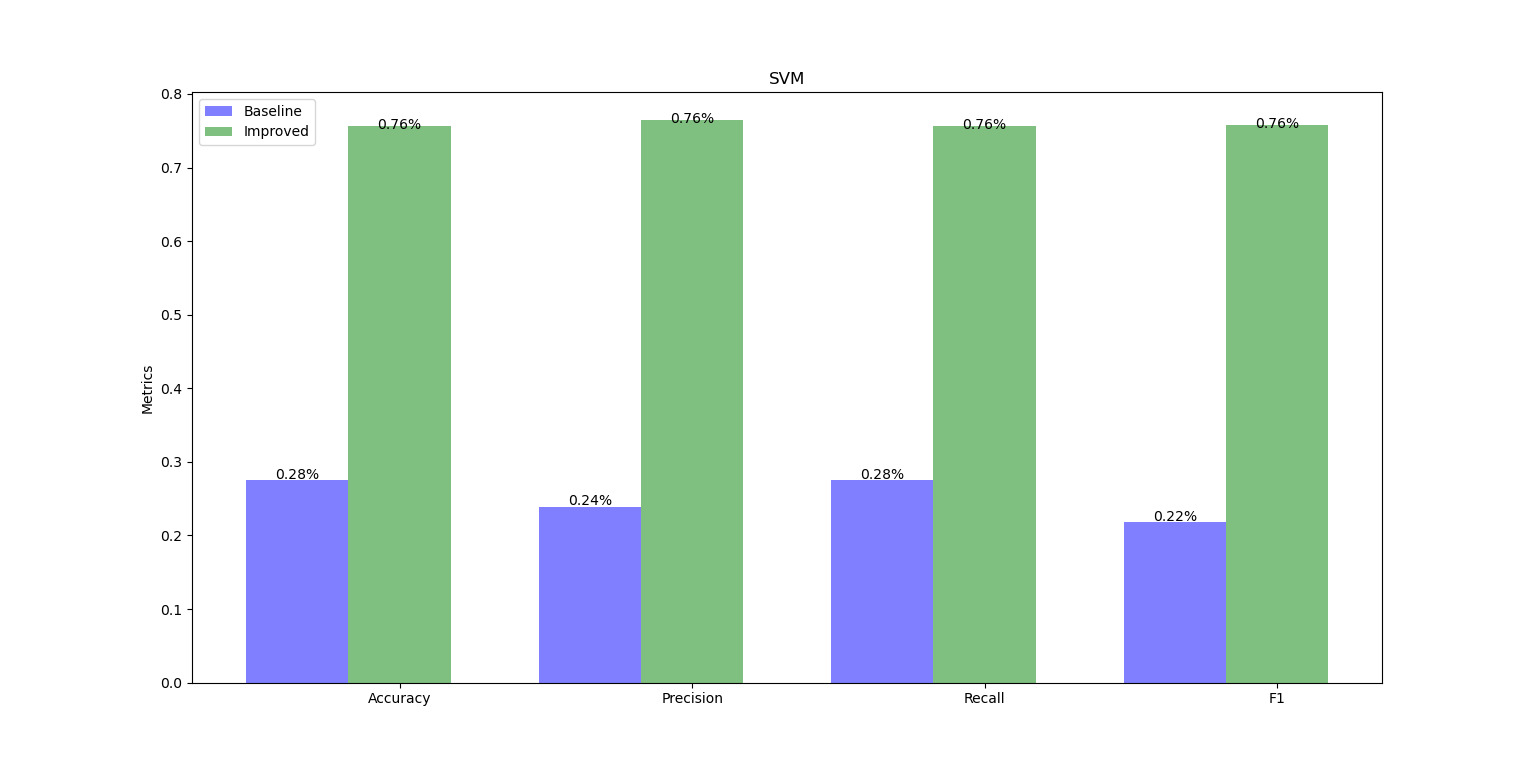
SVC(random\_state=0, decision\_function\_shape='ovo', kernel='rbf')

#### Preprocesare

#### Atribute

### Grafice intra-algoritmi

1. Acuratete: 0.28% baseline, 0.76% varianta imbunatatita
2. Precizie: 0.24% baseline, 0.76% varianta imbunatatita
3. Recall: 0.28% baseline, 0.76% varianta imbunatatita
4. F1: 0.22% baseline, 0.76% varianta imbunatatita

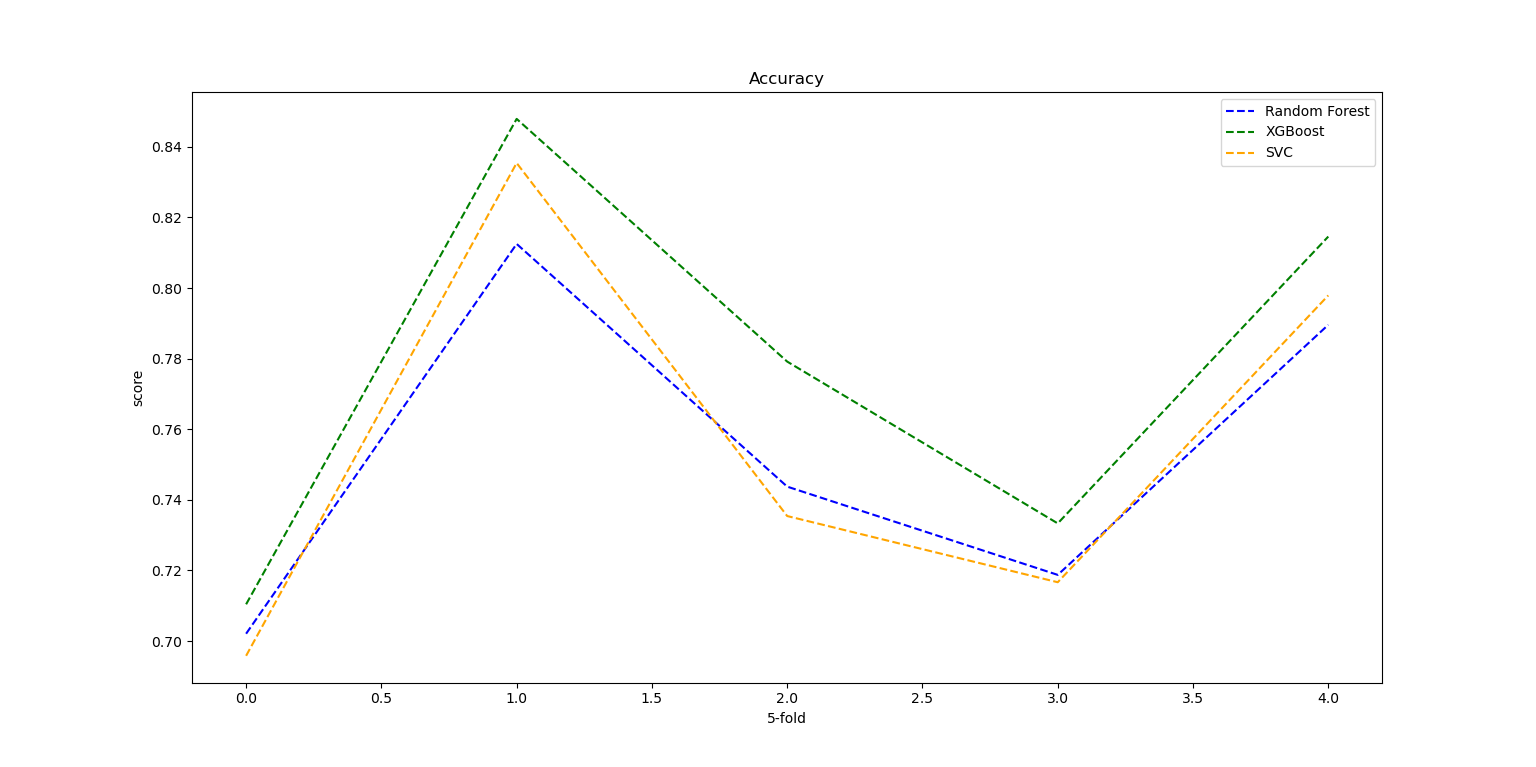


# Grafice inter-algoritmi

Pentru evaluare, s-a folosit 5-Fold Cross Validation si functia cross\_validate cu metricile de mai jos pe fiecare model in parte. Asupra vectorilor rezultati s-a aplicat average pentru a calcula procentul.

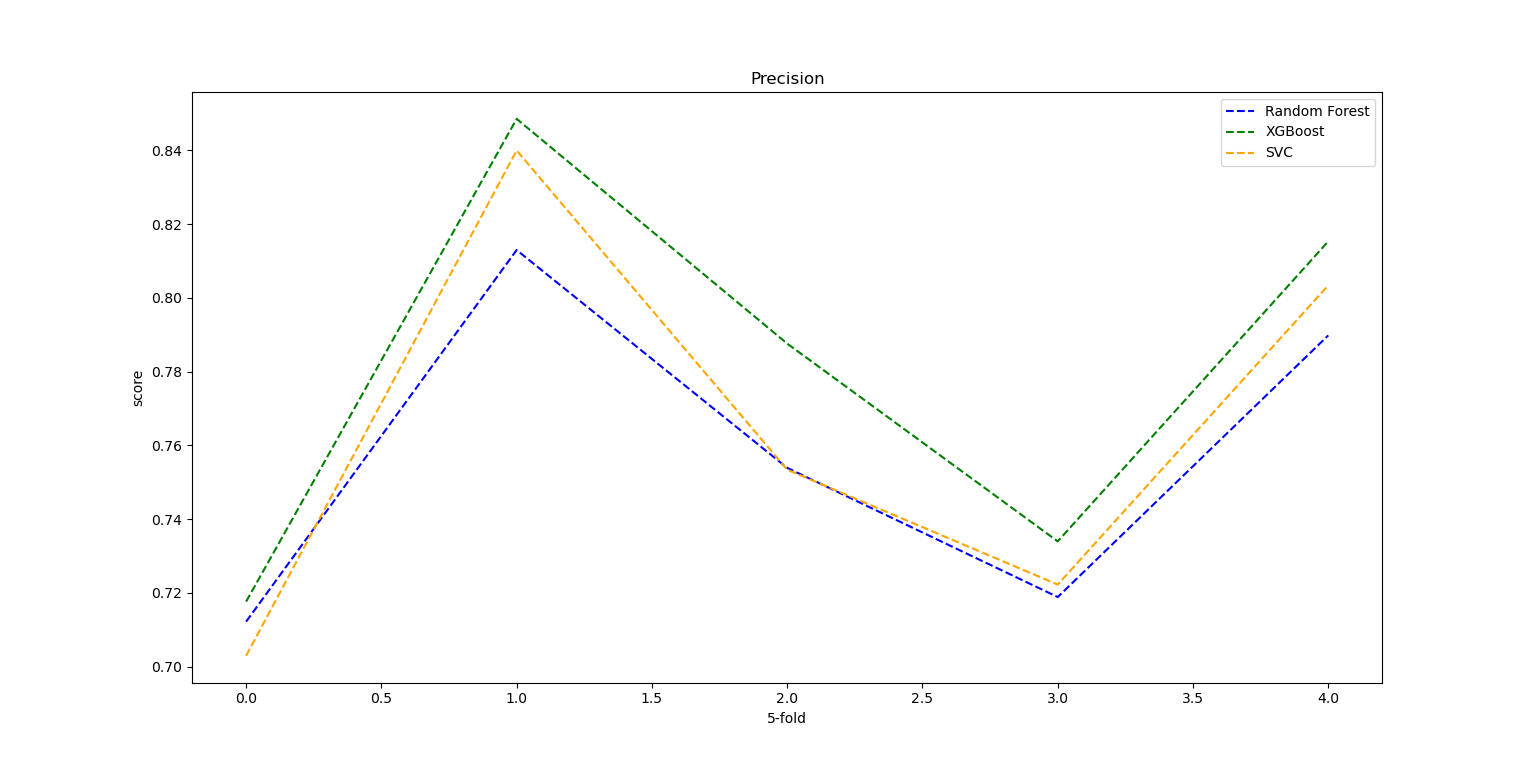
### Acuratete

* Random Forest: 0.75%
* XGBoost: 0.78%
* SVM: 0.76%



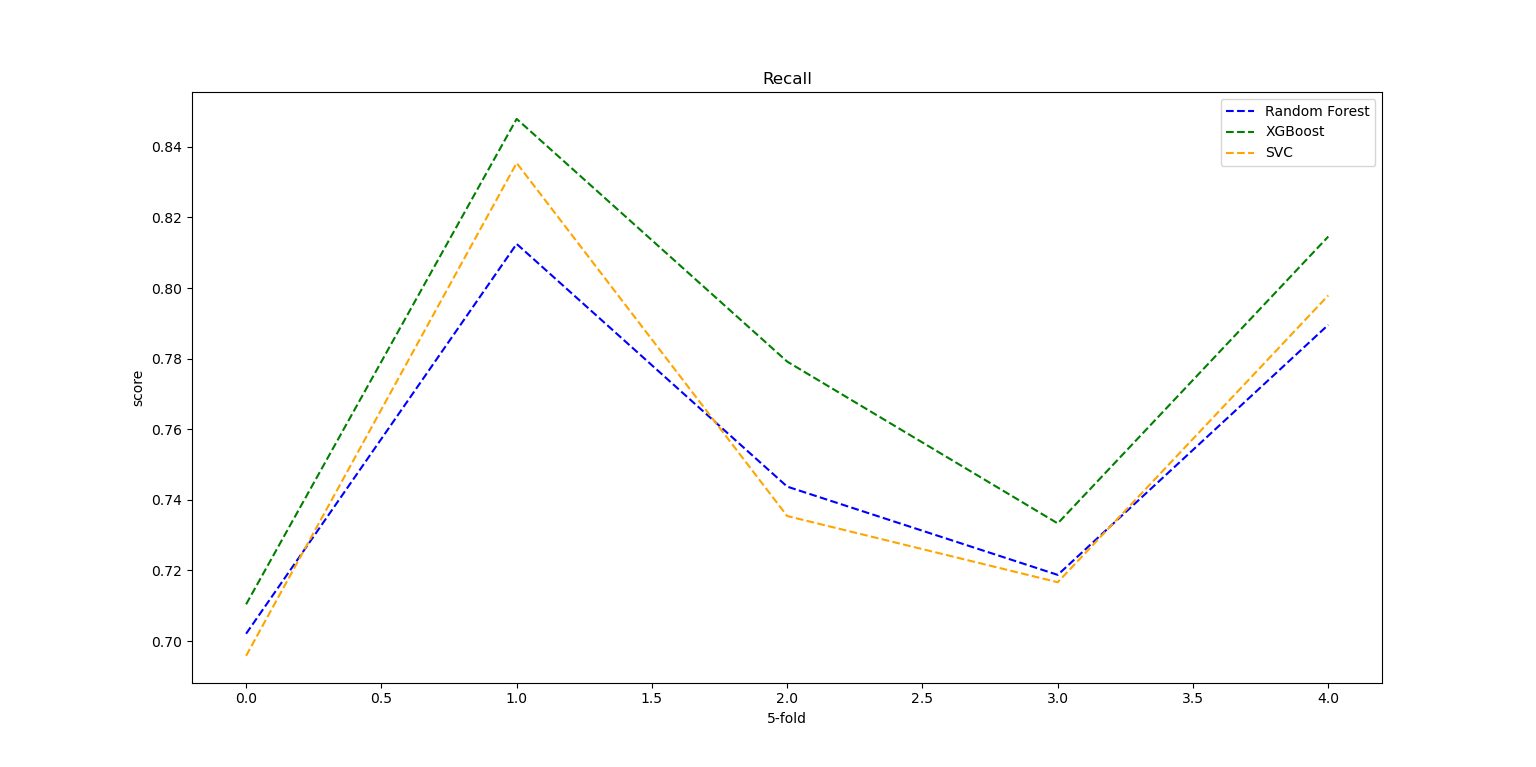
### Precizie

* Random Forest: 0.76%
* XGBoost: 0.78%
* SVM: 0.76%



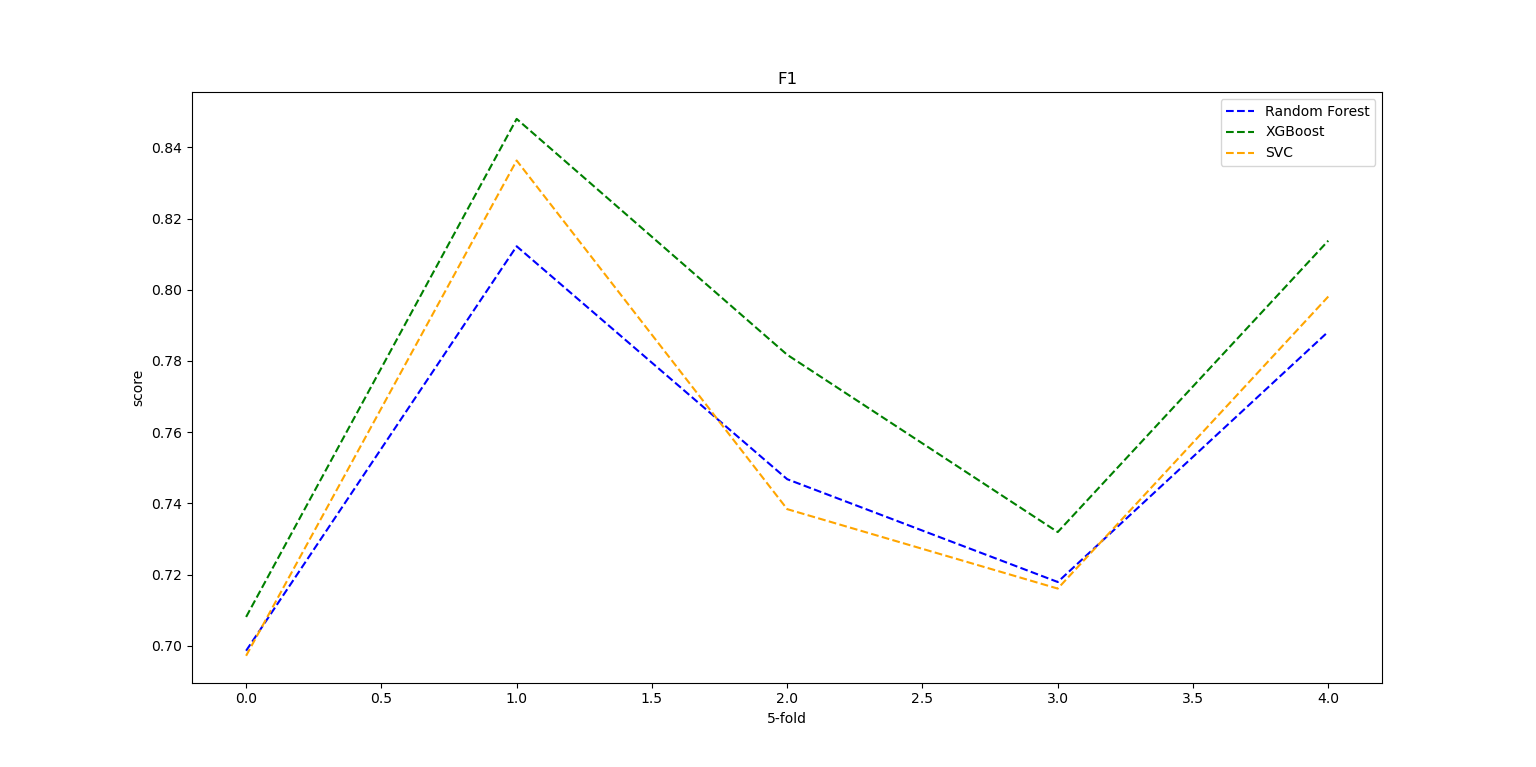
### Recall

* Random Forest: 0.75%
* XGBoost: 0.78%
* SVM: 0.76%



### F1

* Random Forest: 0.75%
* XGBoost: 0.78%
* SVM: 0.76%



### Matricea de confuziune