

Random Forest

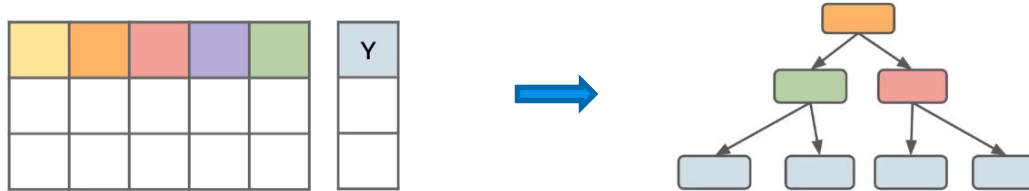
Ce ne așteaptă?

1. **Esența algoritmului Random Forest**
2. **Numărul arborilor și al caracteristicilor**
3. **Bootstrapping**
4. **Eroarea Out-of-Bag**
5. **Instrumente Scikit-Learn**

1. Esența algoritmului Random Forest

■ Neajunsurile algoritmului Decision Trees

- Lipsa garanției utilizării tuturor caracteristicilor



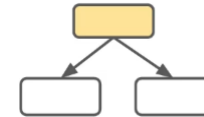
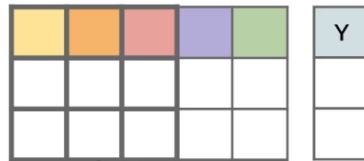
- Influența exagerată a caracteristicii de bază asupra structurii arborelui



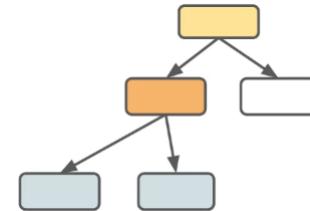
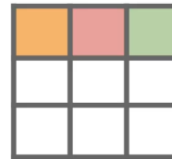
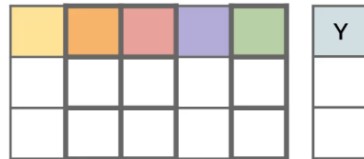
- Probabilitate înaltă de apariție a efectului overfitting

- **Random Forest utilizează doar un subset aliator format din câteva caracteristici pentru fiecare procedură de divizare**

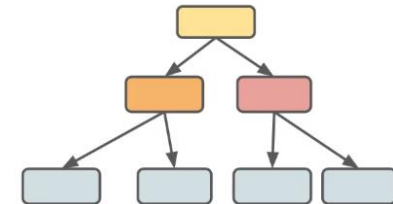
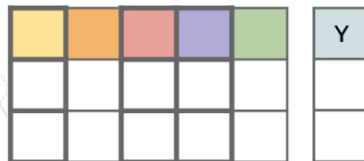
Prima etapă de divizare



A doua etapă de divizare

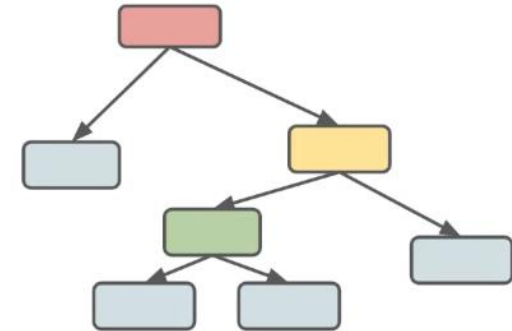
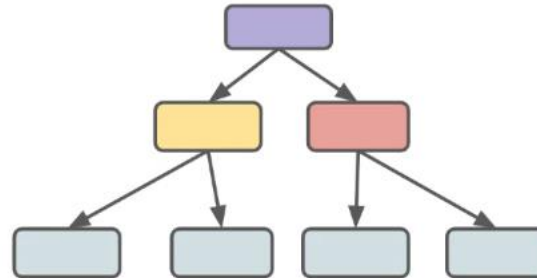
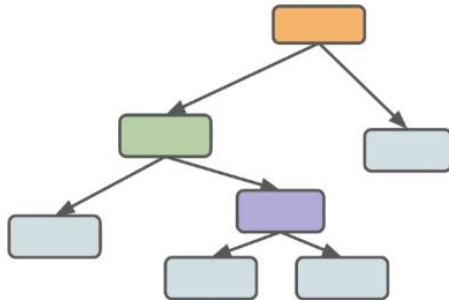
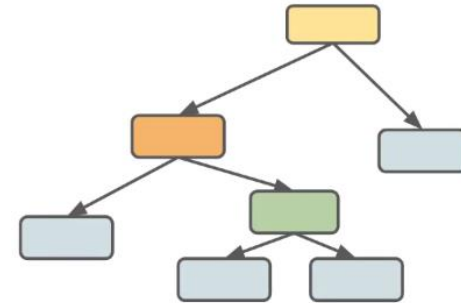
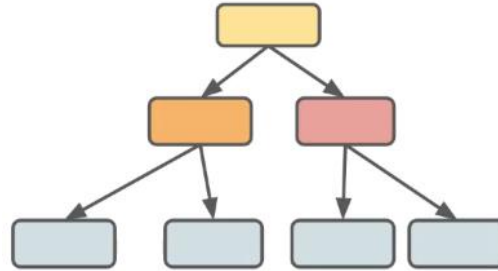


A doua etapă de divizare

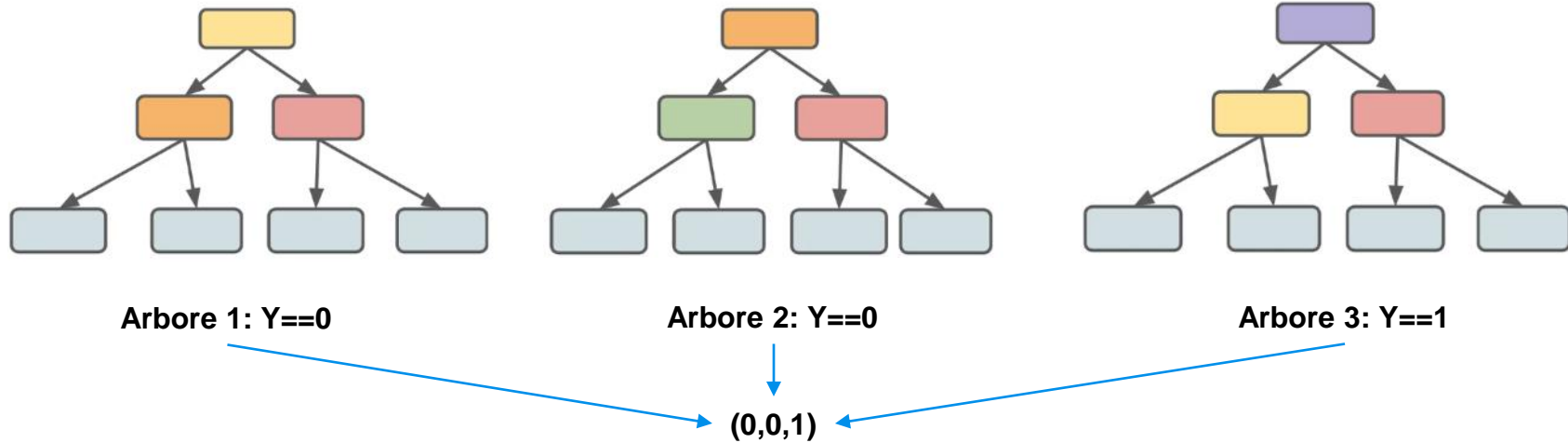


- **Random Forest permite crearea mai multor arbori după acest principiu**

Y

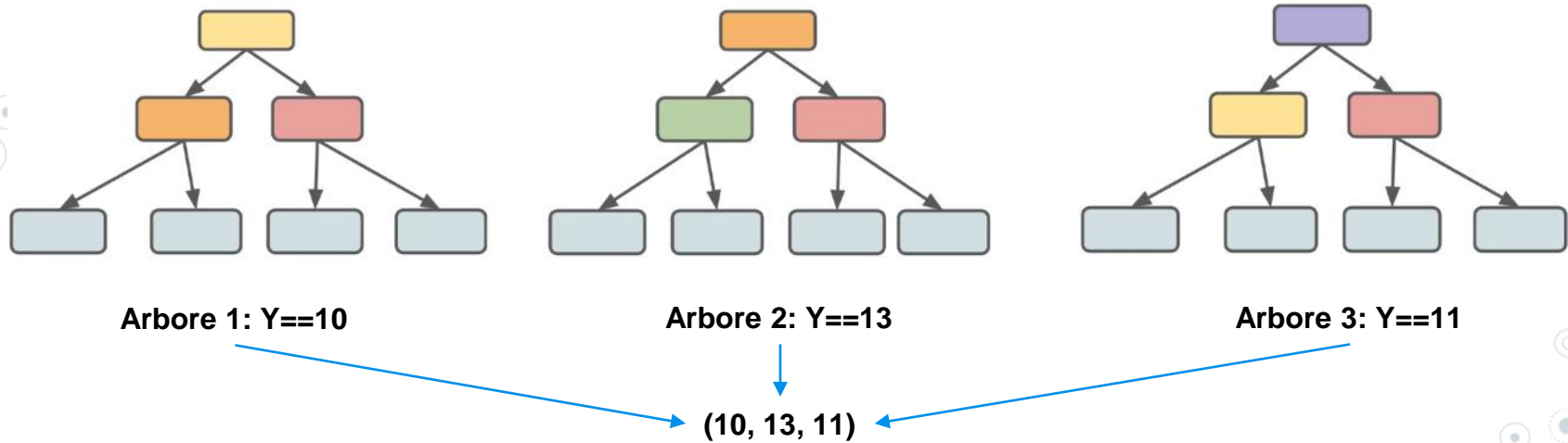


- Realizarea predicției în Random Forest în cazul sarcinilor de clasificare



- În cazul sarcinilor de clasificare valoarea predicție se va considera valoarea majoritară în rezultatele tuturor arborilor $Y==0$

■ Realizarea predicției în Random Forest în cazul sarcinilor de regresie



- În cazul sarcinilor de regresie valoarea predicție se va considera valoarea medie a rezultatelor tuturor arborilor $Y==11,3$

2. Numărul arborilor și al caracteristicilor

- Numărul arborilor ce vor fi considerați în algoritmul Random Forest poate fi nelimitat fără a crea apariția efectului overfitting
- Însă de la un număr oarecare se va constata:
 - Noii arbori nu mai conțin informație nouă deoarece aceștia sunt corelați cu cei existenți
 - Noii arbori reprezintă copii ai arborilor existenți
- Numărul optim al arborilor variază între 64-128 în funcție de numărul de caracteristici
- Pentru selectarea numărului optim de arbori se va utiliza unul din mecanismele:
 - Cross Validation
 - Metoda 'cotului' (dependența erorii de numărul arborilor)

- Numărul caracteristicilor va determina câte caracteristici ale setului se vor include în subsetul selectat aliator
- Pentru sarcini de clasificare numărul optim de caracteristici determinat empiric se consideră \sqrt{N} , unde N numărul total
- Pentru sarcini de regresie numărul optim de caracteristici determinat empiric se consideră $\frac{N}{3}$, unde N numărul total
- Un alt număr optim al caracteristicilor ar putea fi $\log_2(N + 1)$, unde N numărul total

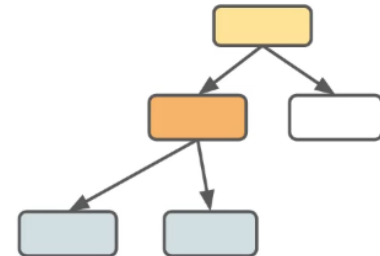
3. Bootstrapping

- Bootstrapping este procedura de selectare aleatoare în procesul de divizare nu doar a caracteristicilor dar și a datelor
- Bootstrapping presupune că unele date pot să nu participe în procesul de divizare, iar unele pot participa de mai multe ori
- Bootstrapping exclude efectul formării arborilor corelați sau a arborilor copii

					Y
0					
1					
2					
3					
4					
5					
6					

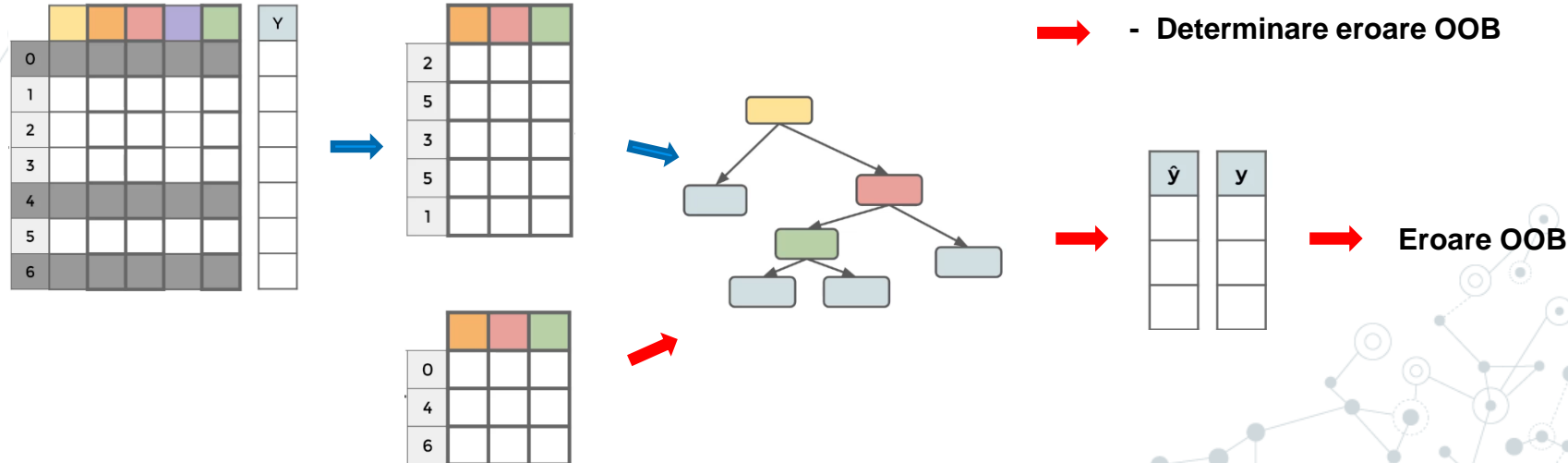


2			
5			
3			
5			
1			



4. Eroarea Out-of-Bag

- Eroarea Out-of-Bag (OOB) este un parametru suplimentar de apreciere a performanțelor modelului ce include bootstrapping
- Eroarea Out-of-Bag (OOB) se obține prin comparația valorilor reale cu valorile predicție pe datele ce au fost excluse prin procedura de bootstrapping



5. Instrumente Scikit-Learn

- **Random Forest pentru sarcini de clasificare**

- **Se importa clasa algoritmului RandomForestClassifier**

- `from sklearn.ensemble import RandomForestClassifier`

- **Se creează un model cu fixarea valorilor hiper-parametrilor:**

- **criterion, max_depth, max_leaf_node, min_impurity_decrease** – identic algoritmului Decision Trees
 - **n_estimators** - numărul de arbori ai algoritmului (valori posibile = int, implicit =100)
 - **max_features** - numărul maxim al caracteristicilor unui subset (valori posibile =int, {"auto", "sqrt", "log2"}, implicit="auto")
 - **bootstrap** – includerea procedurii bootstrapping (valori posibile = bool, implicit=True)
 - **oob_score** – includerea determinării erorii OOB (valori posibile = bool, implicit=False)

- `model = DecisionTreeClassifier(n_estimators =100, max_features
="sqrt", bootstrap=True, oob_score = True)`

- Se realizează trainingul modelului pe datele de training
`model.fit(X_train, y_train)`
- Se vizualizează importanța fiecărei caracteristici
`model.feature_importances_`
- Se realizează predicția pe datele de test
`y_pred = model.predict(X_test)`
- Se realizează predicția pe datele de test cu afișarea probabilităților
`y_pred = model.predict_proba(X_test)`
- Se vizualizează eroarea OOB
`model.oob_score_`

- **Random Forest pentru sarcini de regresie**

- **Se importa clasa algoritmului RandomForestRegressor**

```
from sklearn.ensemble import RandomForestRegressor
```

- **Se creează modelul :**

```
model = RandomForestRegressor()
```

- **Se realizează trainingul modelului pe datele de training**

```
model.fit(X_train, y_train)
```

- **Se realizează predicția pe datele de test**

```
y_pred = model.predict(X_test)
```