# Maximizarea profitului turistic prin ierarhizarea categoriilor de activități

**Ioana Astefanoae**

17 ianuarie 2025

## Cuprins



## 1 Introducere

Scopul acestei lucrări este să prezinte o metodă clară de **maximizare a veniturilor** (`Revenue`) în domeniul turismului, bazată pe analiza unui *set de date* de la adresa:

`https://www.kaggle.com/datasets/umeradnaan/tourism-dataset`

Setul de date conține, pentru mai multe țări și intervale de timp, **informații despre**:

- **Country** – țara (există 7 țări distincte în total);
- **Year** – anul (ex. 2010–2019 sau alt interval);
- **Category** – tipul de activitate turistică (Nature, Historical, Cultural, Beach, Adventure, Urban);

- **Visitors** – numărul de turiști care au participat la activitatea respectivă;
- **Revenue** – venitul total (brut) obținut.

În context, ne dorim să **descoperim** care dintre categoriile turistice are potențialul de a genera cel mai mare profit într-o țară *fixată la momentul deciziei*. Rezolvarea implică **predicția valorii** `Revenue` și **rangarea (sortarea descrescătoare)** a categoriilor. Pentru a îndeplini acest obiectiv, vom folosi un **model de tip Random Forest** (regresie).

# 2 Analiza setului de date și pași de preprocesare

## 2.1 Curățarea și pregătirea datelor

Pentru a asigura **coerența** și **calitatea** modelelor de Învățare Automată, am aplicat următorii pași:

1. **Eliminarea valorilor lipsă** (`NaN`): dacă unele rânduri au valori necunoscute la `Visitors` sau `Revenue`, le excludem, asumând că aceste cazuri sunt puține și nu afectează major setul.
2. **Codificarea variabilelor categorice**:
   - `Country` se transformă în coduri de tip *One-Hot*, pentru fiecare țară (de ex. `Country_India`, `Country_Brazil` etc.).
   - `Category` se transformă la rândul ei în coduri `Category_Nature`, `Category_Beach` ș.a.m.d.
3. **Împărțirea setului în train/test**: folosim o proporție de **80%** pentru antrenare și **20%** pentru testare. Astfel, validăm corect performanța modelului pe date *nevăzute* anterior.
4. **Crearea variabilei-țintă**: ne concentrăm pe `Revenue` ca indicator de profit, deși, la nevoie, putem examina și `Revenue / Visitors` ca indice de profitabilitate pe turist.

## 2.2 Argumente pentru metoda Random Forest

Ne-am orientat spre un model **Random Forest Regressor** din motivele:

- **Robustețe** față de zgomotul din date: un singur arbore de decizie poate supraînvăța datele (overfitting), dar Random Forest, prin combinarea mai multor arbori, reduce varianța și îmbunătățește generalizarea.

- **Capacitate de a surprinde relații non-liniare**: `Revenue` depinde nu doar de `Visitors`, ci și de `Country` (diferențe geografice, demografice), `Category` (unele activități pot fi mult mai profitabile) și `Year` (posibile trenduri în timp). Arborii de decizie nu cer relații strict liniare, fiind eficienți în astfel de situații.
- **Implementare directă și rapidă**: librăria `scikit-learn` oferă o clasă `RandomForestRegressor` ușor de folosit, facilitând antrenarea și testarea pe un set standard de date.

După ce modelul este antrenat să *prezică* `Revenue` pentru (Country, Year, Category, ...), îl vom folosi pentru a **evalua potențialul** fiecărei categorii turistice în cadrul *unei țări anume*, având astfel un mod direct de **maximizare** a veniturilor preconizate.

# 3 Pași concreți de implementare

## 3.1 Fluxul rezolvării

Odată pregătite datele (curățate și codificate), fluxul de implementare este:

1. **Definirea setului de antrenare**: `X` conține variabilele-predictor (de ex. `Year`, `Visitors`, plus `Country_X` și `Category_X`), iar `y` = `Revenue`.
2. **Antrenarea modelului RandomForestRegressor**: folosim un număr specific de arbori (ex. 100), stabilim `random_state` pentru reproducibilitate.
3. **Testarea pe setul de test** (20% date): calculăm *RMSE* (Root Mean Squared Error) și *MAE* (Mean Absolute Error). Cu cât sunt mai mici, cu atât modelul e mai bun.
4. **Generarea ierarhiei (ranking)**: creăm 6 instanțe de input, fiecare având **Category** diferită (Nature, Historical etc.), dar aceeași țară (`Country_X`). Modelul prezice `Revenue` pe fiecare, noi sortăm descrescător și aflăm care categorie ocupă locul 1, 2, 3 etc.

## 3.2 Exemplu de interpretare a ierarhiei

Dacă, de pildă, la testare observăm că **Beach** conduce la cel mai mare *Revenue* prezis, atunci un antreprenor care deschide un hotel în acea țară se poate **concentra** pe dezvoltarea activităților de plajă (sporturi acvatice, turism litoral etc.), știind că acestea vor aduce **venituri superioare** față de alte categorii (de ex. Cultural sau Historical).

# 4 Evaluare și concluzii

## 4.1 Metrici de performanță

Pentru a *valida* modelul, măsurăm:

- **RMSE** (Root Mean Squared Error): cu cât mai mic, cu atât modelul a reprodus mai precis veniturile reale.
- **MAE** (Mean Absolute Error): oferă o imagine clară asupra erorii medii în termeni monetari.

Valori bune pentru aceste metrici indică un model **credibil**, care poate **ghida** deciziile investiționale.

## 4.2 Concluzie finală

Astfel, **Random Forest** se arată o metodă puternică pentru **maximizarea profitului turistic**, deoarece:

1. Poate manipula relativ ușor date eterogene (Date despre țări, categorii, număr de vizitatori etc.).
2. Reduce supra-antrenarea și are *o performanță* solidă, validată de RMSE și MAE rezonabile pe setul de test.
3. Permite *rangarea* categoriilor turistice pentru **orice țară** interesată, generând un *scor numeric* (venitul prezis) ce se poate ordona descrescător.

   **Codul complet Python**, care implementează toți pașii descriși (inclusiv antrenarea *RandomForestRegressor* și generarea ranking-ului), se găsește într-un fișier separat, `cod_random_forest.py`. În acesta, fiecare linie importantă este comentată succint, pentru a clarifica funcționalitatea.