# 🛒 Shopping Cart Abandonment Prediction

This machine learning project predicts whether an online shopper will abandon their cart based on behavioral data like time spent on site, pages visited, and interaction with discounts.

## 🎯 Objective

To build a Supervised Machine Learning model that classifies whether a user is likely to abandon their shopping cart using a Decision Tree Classifier.

## 🧾 Dataset Description

This is a synthetic dataset created for demonstration purposes.

| Feature Name | Description | Type |
| --- | --- | --- |
| Time\_Spent\_on\_Site\_Min | Time user spent browsing (in minutes) | Numeric |
| Pages\_Visited | Number of product pages the user viewed | Numeric |
| Added\_to\_Cart | Whether the user added items to cart (1=yes, 0=no) | Binary |
| Discount\_Viewed | Whether the user viewed a discount (1=yes, 0=no) | Binary |
| Abandoned | Target variable: Did the user abandon the cart? | Categorical (Yes/No) |

## 🛠️ Technologies Used

* Python 3.x
* Pandas
* Scikit-learn
* Matplotlib

## 🧠 Model

* Algorithm: DecisionTreeClassifier
* Criterion: 'entropy' (uses information gain)
* Encoder: LabelEncoder for binary target variable
* Train/Test Split: (Optional for learning, full data used here)

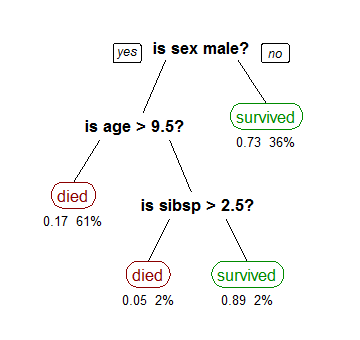
## 🧾 Step-by-Step Process

1. Import libraries
2. Create a synthetic dataset with features and labels
3. Encode the target variable (Yes = 1, No = 0)
4. Train the Decision Tree Classifier
5. Evaluate model performance
6. Visualize the decision tree

## 📈 Results

* Training Accuracy: 100% (due to small dataset)
* Evaluation: classification\_report shows perfect scores on small data
* Interpretation: Time spent on site, adding items to cart, and discount viewing are key features in predicting abandonment

## 📊 Decision Tree Visualization

  
(Use matplotlib to generate your actual tree from the code.)

## 🚀 How to Run the Code

1. Clone or download this repository
2. Install dependencies:

bash

CopyEdit

pip install pandas scikit-learn matplotlib

1. Run the script in a Jupyter Notebook or Python IDE

## 🔮 Future Improvements

* Use a larger real-world dataset (e.g., from e-commerce analytics)
* Apply a train-test split or cross-validation
* Add additional features like user demographics or session ID
* Deploy using Flask/Streamlit as a web app

## 📁 License

This project is open-source under the MIT License.

Step-by-step breakdown of the Shopping Cart Abandonment Prediction project using a Decision Tree Classifier — great for documentation, learning, or presentation.

## 🛒 Step-by-Step: Shopping Cart Abandonment Prediction

### ✅ Step 1: **Import Required Libraries**

python

CopyEdit

import pandas as pd

from sklearn.preprocessing import LabelEncoder

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score, classification\_report

from sklearn import tree

import matplotlib.pyplot as plt

### ✅ Step 2: **Create the Dataset**

You can use this manually created dataset for initial experimentation.

python

CopyEdit

data = {

'Time\_Spent\_on\_Site\_Min': [5, 12, 20, 35, 60, 50, 15, 25, 40, 30, 55, 10, 18, 33, 45],

'Pages\_Visited': [2, 4, 5, 10, 12, 9, 3, 6, 11, 7, 13, 2, 5, 8, 10],

'Added\_to\_Cart': [0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1],

'Discount\_Viewed': [0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1],

'Abandoned': ['Yes', 'No', 'No', 'No', 'No', 'Yes', 'Yes', 'No', 'Yes', 'No', 'No', 'Yes', 'No', 'No', 'No']

}

df = pd.DataFrame(data)

### ✅ Step 3: **Encode the Target Column**

Convert Abandoned values to numeric using LabelEncoder.

python

CopyEdit

le = LabelEncoder()

df['Abandoned'] = le.fit\_transform(df['Abandoned']) # Yes = 1, No = 0

### ✅ Step 4: **Split Features and Target**

Separate inputs (X) and output (y).

python

CopyEdit

X = df.drop('Abandoned', axis=1)

y = df['Abandoned']

### ✅ Step 5: **Train the Decision Tree Classifier**

Use the Decision Tree model with entropy as the criterion.

python

CopyEdit

model = DecisionTreeClassifier(criterion='entropy', random\_state=0)

model.fit(X, y)

### ✅ Step 6: **Make Predictions and Evaluate**

Predict on the same dataset and calculate performance metrics.

python

CopyEdit

y\_pred = model.predict(X)

accuracy = accuracy\_score(y, y\_pred)

print("Accuracy:", accuracy)

report = classification\_report(y, y\_pred, target\_names=['No', 'Yes'])

print("Classification Report:\n", report)

### ✅ Step 7: **Visualize the Decision Tree**

Use matplotlib and sklearn.tree to draw the tree.

python

CopyEdit

plt.figure(figsize=(12, 8))

tree.plot\_tree(model, feature\_names=X.columns, class\_names=['No', 'Yes'], filled=True)

plt.title("Decision Tree - Shopping Cart Abandonment Prediction")

plt.show()

## 📝 Summary

| Step | Description |
| --- | --- |
| 1 | Load required libraries |
| 2 | Create or load dataset |
| 3 | Encode the target label (Yes/No) |
| 4 | Split into features and target |
| 5 | Train the decision tree classifier |
| 6 | Evaluate model accuracy and generate report |
| 7 | Visualize the decision tree |