# 🎮 Gaming Addiction Risk Prediction

This machine learning project uses a Decision Tree Classifier to predict whether an individual is at high or low risk of gaming addiction based on lifestyle and behavioral factors.

## 📌 Objective

To build a supervised learning model that predicts gaming addiction risk level (High or Low) using features like gaming hours, sleep patterns, social activity, and academic performance.

## 🧾 Dataset

This project uses a synthetic dataset with the following features:

| Feature Name | Description | Type |
| --- | --- | --- |
| Hours\_Gaming\_Per\_Week | Number of hours spent gaming in a week | Numeric |
| Sleep\_Hours | Average daily sleep hours | Numeric |
| Social\_Activity\_Score | Social engagement score (1-10) | Numeric |
| Academic\_Performance\_Score | Academic/work performance score (1-10) | Numeric |
| Risk | Target variable (High or Low) | Categorical |

## 🛠️ Technologies Used

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

## 📈 Model

A Decision Tree Classifier is used with:

* criterion='entropy' – uses information gain to split nodes
* Model trained on 15 data points
* Encodes target labels (High = 1, Low = 0)

## 🔍 Code Overview

1. Data Preparation
   * Create a pandas DataFrame from hardcoded sample data
   * Encode target labels using LabelEncoder
2. Model Training
   * Features: Gaming hours, sleep, social activity, academic performance
   * Target: Addiction risk level (High/Low)
   * Model: DecisionTreeClassifier
3. Model Evaluation
   * Training accuracy: 100% (overfitting likely due to small dataset)
   * Evaluated with classification\_report
4. Visualization
   * Tree plotted using matplotlib and sklearn.tree.plot\_tree

## 📊 Output

* Accuracy: 100% on training data (due to small dataset)
* Classification Report: Perfect precision and recall on both classes
* Tree Visualization:

Shows clear decision paths based on input features like gaming hours, sleep, etc.

## 📌 How to Run

1. Clone or download this project
2. Install requirements:

bash

CopyEdit

pip install pandas scikit-learn matplotlib

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

## 🚀 Future Improvements

* Use real-world survey data
* Introduce train-test split to avoid overfitting
* Add user input or API for real-time prediction
* Deploy as a web app with Streamlit or Flask

## 📁 License

This project is open-source under the MIT License.

## 🎯 Step-by-Step Guide: Gaming Addiction Risk 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**

Manually create a synthetic dataset simulating gaming habits and risk factors.

python

CopyEdit

data = {

'Hours\_Gaming\_Per\_Week': [5, 10, 25, 40, 35, 50, 20, 15, 60, 55, 12, 8, 30, 45, 22],

'Sleep\_Hours': [8, 7, 6, 5, 5, 4, 7, 6, 4, 4, 7, 8, 6, 5, 7],

'Social\_Activity\_Score': [8, 7, 4, 2, 3, 1, 5, 6, 1, 2, 7, 8, 4, 2, 5],

'Academic\_Performance\_Score': [9, 8, 6, 3, 4, 2, 6, 7, 2, 3, 8, 9, 5, 3, 6],

'Risk': ['Low', 'Low', 'High', 'High', 'High', 'High', 'Low', 'Low', 'High', 'High', 'Low', 'Low', 'High', 'High', 'Low']

}

df = pd.DataFrame(data)

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

Convert the Risk column from string (Low/High) to numeric (0/1).

python

CopyEdit

le = LabelEncoder()

df['Risk'] = le.fit\_transform(df['Risk']) # Low = 0, High = 1

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

Separate input features and output labels.

python

CopyEdit

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

y = df['Risk']

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

Create and fit a decision tree model on the entire dataset.

python

CopyEdit

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

model.fit(X, y)

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

Evaluate performance on the same dataset (note: overfitting likely).

python

CopyEdit

y\_pred = model.predict(X)

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

print("Accuracy:", accuracy)

report = classification\_report(y, y\_pred, target\_names=['Low', 'High'])

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

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

Use matplotlib to show the learned tree visually.

python

CopyEdit

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

tree.plot\_tree(model, feature\_names=X.columns, class\_names=['Low', 'High'], filled=True)

plt.title("Decision Tree - Gaming Addiction Risk Prediction")

plt.show()

### 🧠 Summary

* Trained on 15 samples
* Achieved 100% accuracy (due to small size)
* Decision tree clearly shows how features like hours gaming, social activity, and sleep impact risk