MUSHROOM-HUNTER

مقدمه:

پروژه‌ی Mushroom Classification با هدف تشخیص قارچ‌های سمی از قارچ‌های خوراکی انجام شده است.

قارچ‌ها انواع مختلفی دارند و بعضی از آن‌ها سمی هستند و مصرفشان خطرناک یا حتی مرگبار است. بنابراین تشخیص سریع و دقیق قارچ‌های سمی اهمیت زیادی دارد.

در این پروژه، با استفاده از داده‌های مربوط به ویژگی‌های ظاهری قارچ‌ها و الگوریتم‌های یادگیری ماشین (Machine Learning)، مدلی ساخته شد که می‌تواند با دقت بالا تعیین کند یک قارچ خوراکی است یا سمی.

هدف اصلی پروژه علاوه بر پیش‌بینی درست، حساس کردن مدل نسبت به قارچ‌های سمی بوده است تا هیچ قارچ سمی از دست نرود. این کار با تنظیم مناسب Threshold و استفاده از الگوریتم Random Forest انجام شد.

داده ها و کتابخانه ها:

در این پروژه، برای پردازش داده‌ها و ساخت مدل از کتابخانه‌های Python استفاده شد:

- pandas برای مدیریت داده‌ها و عملیات روی DataFrame
- numpy برای محاسبات عددی
- seaborn و matplotlib برای مصورسازی داده‌ها
- scikit-learn برای پیش‌پردازش داده‌ها، تقسیم داده‌ها، مدل‌سازی و ارزیابی

داده‌ها از فایل mushrooms.csv بارگذاری شدند و شامل ویژگی‌های مختلف ظاهری هر قارچ است. با دستور زیر داده‌ها خوانده و بررسی اولیه انجام شد.

```
import numpy as np
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
train_csv = pd.read_csv("train.csv")
```

تحلیل داده ها:

پس از بارگذاری داده‌ها، بررسی اولیه شامل مشاهده ستون‌ها و مقادیر یکتای هر ویژگی انجام شد تا با محتوای هر ستون آشنا شویم. این کار کمک می‌کند بفهمیم هر ویژگی چند دسته مختلف دارد و چه نوع مقادیری در آن قرار گرفته است

```
for col in train_csv.columns:
    print(col, train_csv[col].unique())
```

با اجرای این کد، نام هر ستون و مقادیر یکتای آن نمایش داده شد.

این مرحله کمک کرد تا ویژگی‌های دسته‌ای (categorical) شناسایی شوند و برای مرحله‌ی بعد یعنی پیش‌پردازش داده‌ها (Preprocessing) آماده شوند.

پیش پردازش داده ها:

دیتاست قارچ‌ها شامل ویژگی‌های دسته‌ای (categorical) بود، یعنی مقادیر هر ستون به صورت متن یا برچسب بودند. مدل‌های یادگیری ماشین نمی‌توانند مستقیماً با مقادیر متنی کار کنند، بنابراین لازم بود همه‌ی ستون‌ها به مقادیر عددی تبدیل شوند.

برای این کار از LabelEncoder از کتابخانه‌ی scikit-learn استفاده شد

```
from sklearn.preprocessing import LabelEncoder

encoder = LabelEncoder()
for col in train_csv.columns:
    train_csv[col] = encoder.fit_transform(train_csv[col])
```

با اجرای این کد، هر ستون دسته‌ای به عدد تبدیل شد و داده‌ها برای مرحله‌ی مدل‌سازی آماده گردید.

پیش پردازش ستون هدف:

ستون هدف (class) نشان‌دهنده‌ی نوع قارچ است:

- e → خوراکی (edible)
- p → سمی (poisonous)

برای اینکه مدل یادگیری ماشین بتواند پیش‌بینی انجام دهد، لازم بود مقادیر متنی این ستون به عدد تبدیل شود. برای این کار از LabelEncoder استفاده شد:

```
encoder = LabelEncoder()
y = encoder.fit_transform(train_csv["class"])
print(encoder.classes_)
```

با اجرای این کد، خروجی ['e' 'p'] نشان داد که:

- مقدار e به عدد 0 تبدیل شده (خوراکی)
- مقدار p به عدد 1 تبدیل شده (سمی)

این کار باعث شد ستون هدف آماده‌ی استفاده در مدل‌سازی گردد.

آماده سازی ویژگی ها و ستون هدف:

پس از پیش‌پردازش داده‌ها، لازم بود ستون‌ها به دو بخش ویژگی‌ها (Features) و ستون هدف (Target) تقسیم شوند تا برای مدل‌سازی آماده گردند.

- X شامل تمام ستون‌ها به جز ستون هدف (class) می‌شود و به عنوان ورودی مدل استفاده می‌شود.
- y همان ستون هدف (class) است که برچسب خوراکی یا سمی بودن قارچ‌ها را مشخص می‌کند.

کد انجام این کار به صورت زیر بود:

```
X = train_csv.drop("class", axis=1)
y = train_csv["class"]
```

تقسیم داده به آموزش و تست:

برای آموزش مدل و ارزیابی عملکرد آن، داده‌ها به دو بخش جدا تقسیم شدند:

- مجموعه آموزش (Training Set): برای یادگیری مدل استفاده می‌شود.
- مجموعه تست (Testing Set): برای ارزیابی عملکرد مدل روی داده‌های دیده‌نشده استفاده می‌شود.

در این پروژه، ۲۰٪ از داده‌ها به مجموعه تست اختصاص داده شد و برای اطمینان از تکرارپذیری، مقدار random_state=42 تعیین گردید.

کد مربوطه به صورت زیر است:

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42)
```

با اجرای این مرحله، داده‌ها آماده‌ی ساخت مدل Random Forest شدند.

مدل سازی و تنظیم آستانه:

برای پیش‌بینی نوع قارچ (خوراکی یا سمی)، از RandomForestClassifier استفاده شد. این الگوریتم بر اساس مجموعه‌ای از درختان تصمیم (Decision Trees) ساخته شده و برای داده‌های دسته‌ای و پیچیده بسیار مناسب است.

ویژگی‌های مهم در آموزش مدل:

- n_estimators=200 → تعداد درخت‌ها
- class_weight="balanced" → حساس کردن مدل نسبت به کلاس سمی
- random_state=42 → تکرارپذیری نتایج

پس از آموزش مدل، برای رسیدن به Recall برابر با ۱۰۰٪ برای کلاس سمی، Threshold پیش‌بینی کاهش یافت. این کار باعث شد مدل حساس‌تر نسبت به قارچ‌های سمی شود و هیچ نمونه سمی از دست نرود.

کد مربوطه به صورت زیر است:

```
model = RandomForestClassifier(
    n_estimators=200,
    max_depth=None,
    random_state=42,
    class_weight="balanced"
)
model.fit(X_train, y_train)

y_prob = model.predict_proba(X_test)[:,1]
y_pred_thresh = (y_prob > 0.3).astype(int)

print(classification_report(y_test, y_pred_thresh))
```

- y_prob = model.predict_proba(X_test)[:,1] → احتمال هر نمونه برای کلاس سمی
- y_pred_thresh = (y_prob > 0.3) → اعمال Threshold 0.3 برای حساس کردن مدل

با این روش، مدل توانست تمام قارچ‌های سمی را شناسایی کند و Recall کلاس سمی به ۱۰۰٪ نزدیک شد.

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       697
           1       1.00      1.00      1.00       603

    accuracy                           1.00      1300
   macro avg       1.00      1.00      1.00      1300

                                weighted avg       1.00      1.00      1.00      1300
```

ارزیابی مدل:

برای ارزیابی عملکرد مدل، به ویژه حساسیت آن نسبت به قارچ‌های سمی، از معیار Recall استفاده شد.

- Recall میزان نمونه‌های مثبت واقعی (سمی) است که به درستی توسط مدل شناسایی شده‌اند.

هدف پروژه، جلوگیری از از دست دادن هیچ قارچ سمی بود، بنابراین Recall کلاس سمی معیار اصلی ارزیابی بود.

کد محاسبه Recall کلاس سمی:

```
from sklearn.metrics import recall_score

y_prob = model.predict_proba(X_test)[:, 1]
y_pred_thresh = (y_prob > 0.3).astype(int)

recall_poisonous = recall_score(y_test, y_pred_thresh, pos_label=1)

print("Recall (Poisonous class):", recall_poisonous)
```

```
Recall (Poisonous class): 1.0
```

pos_label=1 → کلاس سمی به عنوان کلاس مثبت تعیین شد.

با تنظیم Threshold و استفاده از کلاس وزن‌دهی، Recall کلاس سمی به ۱۰۰٪ نزدیک شد.

این مرحله تضمین می‌کند که مدل هیچ قارچ سمی را از دست ندهد و پروژه هدف اصلی خود را محقق کرده است.