ابتدا کتابخانه‌های مورد نیاز کارمان را در ابتدا برنامه import می‌کنیم اما بهتر است از کتابخانه‌ها و توابعی نیاز داریم را به صورت به خصوص import کنیم:
مثلا:
import sklearn که به این صورت اصلا نباید انجام داد بخاطر اینکه RAM را اشغال می‌کند
from sklearn.preprocessing import LabelEncoder نحوه درست

از کتابخانه pandas برای کار با دیتاست استفاده می‌شود. دیتاست ما در اینجا titanic است که باید متادیتا نیز در کنار آن ارسال شود. متادیتا شامل توضیحات ستون‌ها و مقادیر آن‌ها است تا درک درستی از داده داشته باشیم.
استفاده از from sklearn.preprocessing import LabelEncoder برای پیش‌پردازش داده‌ها استفاده می‌شود برای زمانی که می‌خواهیم داده‌های دسته‌بندیمان را به عددی تبدیل کنیم.
استفاده از from sklearn.model_selection import train_test_split برای تقسیم‌بندی داده‌ها به دو نوع آموزشی و آزمایشی که به صورت تصادفی و درهم‌ریخته این کار را انجام می‌دهد به همین منظور از random_state استفاده می‌کنیم تا داده‌هایی که به صورت تصادفی جدا می‌شود در هر بار تکرار، یکسان باشد تا بتوان به درستی مدلمان را ارزیابی کنیم. همچنین نسبت داده‌های آموزشی به آزمایشی را میتوان مشخص کرد: train_size=0.8 و ما برای این تقسیم‌بندی باید بردار ویژگی X و بردار هدف y را به آن دهیم.
بردار ویژگی X در این برنامه برابر است با ستون یک به بعد.
بردار هدف y در این برنامه برابر است با ستون صفرم که ستون زنده بودن را نمایش می‌دهد.
استفاده از from sklearn.ensemble import RandomForestClassifier برای آموزش داده‌ها (train) مدل است که از روش‌های Ensemble که خود دو روش Bagging و Boosting دارد استفاده می‌کنیم. در اینجا از جنگل تصادفی استفاده شده است که زیر مجموعه روش‌های Bagging است. و در دل این الگوریتم می‌توان چندین درخت تصمیم را آموزش داد که هر کدام به صورت جداگانه یاد میگیردند و به همین دلیل به آن‌ها Bagging می‌گوییم اما در روش‌های Boosting به این صورت است که تجارب آموخته شده روی مدل‌های دیگر تاثیر می‌گذارد و خطا نیز کمتر می‌شود.
استفاده از from sklearn.metrics import classification_report, confusion_matrix برای ارزیابی عملکرد مدل است که classification_report به ما مقادیر Accuracy, Recall, Precision, F1 Score را می‌دهد به اضافه تعدادی پارامتر ارزیابی دیگر و confusion_matrix به ماتریس درهم‌ریختگی را می‌دهد که می‌توان اطلاعات مفیدی بدست آورد و باید در نظر داشت که این معیارهای ارزیابی دو مقدار y_pred و y_true را از ما می‌گیرند.

داده‌های خود را که با استفاده از کتابخانه pandas می‌خوانیم و داده‌ها به صورت DataFrame که ساختمان‌داده کتابخانه Pandas تبدیل می‌شود. باید درک درست و خوبی از داده‌ها بدست بیاوریم و به همین دلیل از روش‌های آماری گوناگون استفاده می‌کنیم همانند واریانس داده‌های عددی و کمترین و بیشتری و ... .
در این مرحله پیش‌پردازش است باید ستون‌هایی که ارزشی برای مدل ندارد را حذف کرد و همچنین داده‌های null را یا حذف کرد یا با مقادیر درست جای گذاری کرد و همچنین داده‌های طبقه‌بندی را به عددی تبدیل کرد. که تمام این موارد انجام شده است و در کد توضیح داده شده است.

بعد از مرحله پیش پردازش باید داده‌ها را به numpy تبدیل کنیم و دو مجموعه آموزشی و آزمایشی درست کنیم که هر کدام نیز شامل بردار ویژگی X و بردار هدف y است. الان داده‌ها آماده آموزش هستند.

یک نمونه از آن مدل یادگیری میسازیم و پارامترها و هایپرپارامتر را مشخص میکنیم.
rfc = RandomForestClassifier(n_estimators=100) که در اینجا n_estimators یک هایپرپارامتر است که تعداد درخت‌های تصمیمی که قرار است ساخته شود را بیان می‌کند و این دست محقق یا برنامه‌نویس است که با تجربه بدست می‌اید و عدد مشخی ندارد.
مدل ساخته شده را با داده‌های آموزشی آموزش می‌دهیم که کمتر از چندین ثانیه این یادگیری انجام می‌شود. :) قدرت پایتون

حال وقت آن رسیده که مدل را تست کنیم و داده‌های تست را به آن می‌هیم و به y_pred می‌دهد که با کمک معیارهای ارزیابی، y_pred را با y_true می‌سنجیم.
که در اینجا مدل در مرحله آموزش توانسته است به دقت میاگین 95 درصد برسد اما در مرحله آزمایش به دقت میانگین 83 درصد. که می‌توان گفت بیش‌برازش (overfit) اتفاق افتاده است و باید این مورد را رفع کرد.