

# ML Model Deployment (Local)

Week 4: Deployment on Flask



by Xiyuan Wu

# Table of contents

- [Introduction](#)
- [Files Needed](#)
- [Deploy Model](#)
- [Example Output](#)

# Introduction

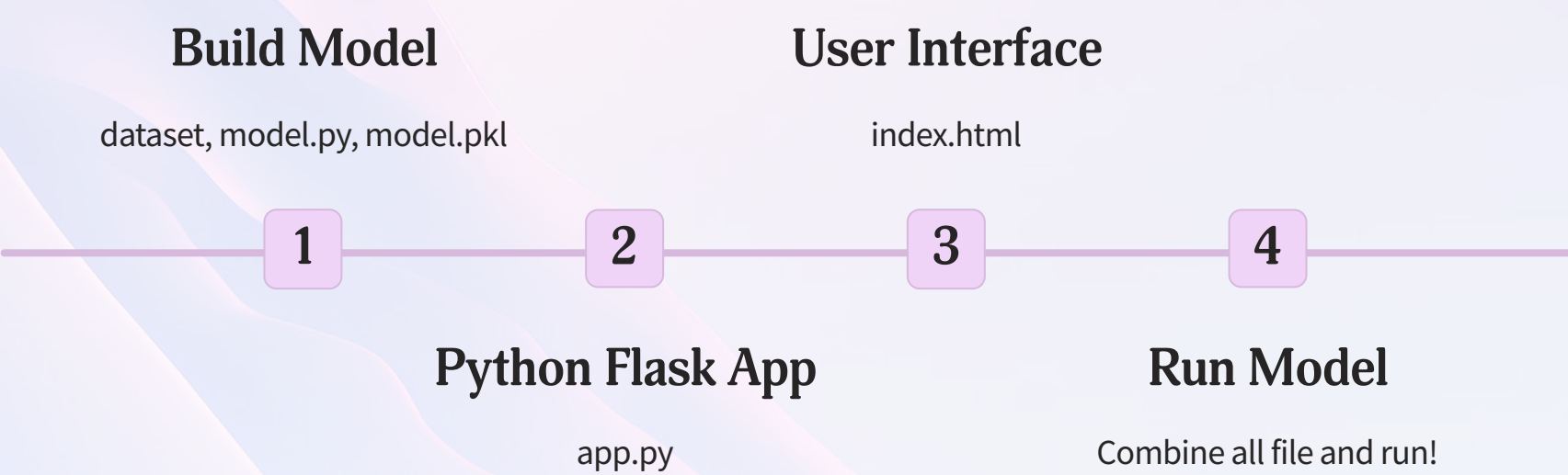
## What is Modeling & Deploying Model?

Modeling in the context of machine learning is the process of training an algorithm on a dataset to recognize patterns and relationships, resulting in a model that can make predictions or decisions.

Deploying a model refers to the integration of this trained model into a production environment where it can process real-time data and provide outputs for practical applications and decision-making.

So, In this project, we are going to deploy a machine learning model using Flask.

In order to deploy an ML model, we need 5 files, followed by 4 steps:




I will use one of the projects I have done before — Salifort Motors Workforce Analysis — to complete this model deployment.

## Background


Salifort Motors is a fictional French-based alternative energy vehicle manufacturer. Its global workforce of over 100,000 employees research, design, construct, validate, and distribute electric, solar, algae, and hydrogen-based vehicles. Salifort’s end-to-end vertical integration model has made it a global leader at the intersection of alternative energy and automobiles.

As a data specialist working for Salifort Motors, you have received the results of a recent employee survey. The senior leadership team has tasked you with analyzing the data to come up with ideas for how to increase employee retention. To help with this, they would like you to design a model that predicts whether an employee will leave the company based on their department, number of projects, average monthly hours, and any other data points you deem helpful.

To recap, I have already completed a full analysis and finished building the model beforehand. Feel free to check it out here:

 [github.com](#)

**Salifort Motors Workforce Analysis: Employee Turnover Forecast**



## Goal

In this project, I built a model that predicts whether an employee will leave the company or not. As its final form, we can type in employee information on a web interface, such as salary, rating, number of years in the company, etc., and we can get the result - whether an employee will leave or not!

# Files Needed

## Model.py/Model.pkl

The `model.py` file typically contains the code that defines and trains a machine learning model, specifying the algorithm, the training process, hyperparameters, and the data it will learn from. After training, the resulting model—which encapsulates what has been learned—is often saved to a file, commonly using the `.pkl` format for Python's pickle module, resulting in a file like `model.pkl`. This file can then be loaded in other scripts or systems to make predictions without needing to retrain the model, facilitating model deployment in various environments.

## App.py

The `app.py` file in a Flask project orchestrates the application's server-side operations, including defining routes that handle web requests, integrating with the `model.pkl` to process data and return predictions, and rendering the appropriate HTML templates to the user. It acts as the central script where the Flask instance is created, routes are defined, and the application is run, allowing for the deployment of the machine learning model as a web service.

## Index.html

The `index.html` file is the front-end component that users interact with, usually serving as the landing page or the main interface of a web application. It is structured with HTML and often styled with CSS and enhanced with JavaScript. In the context of a Flask application, `index.html` is typically where the user inputs data into a form, which is then submitted to the Flask backend via the `app.py` routes. The form submission interacts with the deployed model, and `index.html` would display the results or predictions from the model back to the user.

Unfortunately, you will have to code `app.py` and `index.html` on your own, and the content of these files will vary completely depending on the model you are trying to deploy.



# Deploy Model

After we have all the files prepared, we can follow the steps below:

## 1 Open VSCode

You can use any IDE you want; in this example, I use VS Code. Don't forget install Flask by using `pip install flask`!

## 2 Create Folder

Create a folder with any name you prefer; I named mine `ML_Model_Deployment`. After creating it, put all 5 files into this folder. Then, within the `ML_Model_Deployment` folder, create another folder called `templates`. Place only the `index.html` file under the `templates` folder, while the rest of the 4 files will be kept under the `ML_Model_Deployment` folder.

## 3 Run

After we have everything set up, we can run our file! Note that the process is not as easy as you might think. When you try to run it for the first time, you will encounter all kinds of issues that you need to solve. After you address these issues, it should run successfully and open a new page in the browser.

## 4 Ger result

After we open the new page, we can type in our information, and it will start predicting!

# Example Output

## Type Infomation

Employee Retention Prediction Form

Satisfaction Level:

0.86

Last Performance Rating:

0.75

Number of Projects:

4

Average Monthly Hours:

236

Years at Company:

4

Had Work Accident:

no

Promoted in Last 5 Years:

no

Salary Level:

medium

Department:

hr

Predict

## Result

Employee Retention Prediction Form

Satisfaction Level:

Last Performance Rating:

Number of Projects:

Average Monthly Hours:

Years at Company:

Had Work Accident:

no

Promoted in Last 5 Years:

no

Salary Level:

low

Department:

RandD

Predict

The employee is likely to stay.

Next week, this model will be deployed to the cloud instead of local PCs. Come check it out later if you are interesting!