

Model Deployment

From Model to Reality: Deploying & Monitoring ML Models

Subtitle: What happens after you've trained your final model?

The Journey of a Machine Learning Model

1. **Problem Definition** → **Data Collection**
2. **Data Cleaning** → **Exploratory Analysis**
3. **Model Training** → **Model Evaluation** ✓
4. ➡ **MODEL DEPLOYMENT** (*This is what we're talking about!*)
5. **Monitoring & Maintenance**

Key Message: Training a good model is only half the battle. Its real value is realized when it's used to make decisions in the real world.

What is Model Deployment?

Deployment is the process of making your trained model available to others so they can use it to make predictions on new data.

- It's about **integrating the model into an existing production environment**.
- The model is no longer a static file on your computer; it becomes a **live prediction service**.

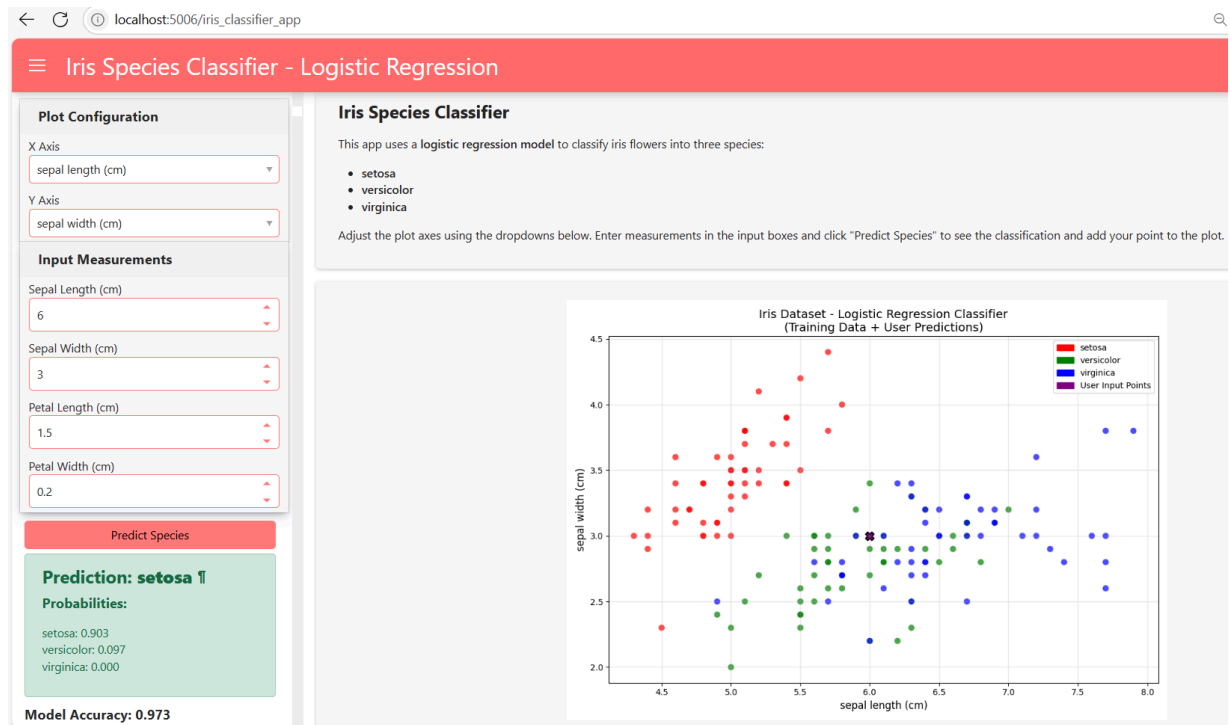
A Common and Powerful Example: The Interactive Web Application

- Wrap your model in a web framework (e.g., Flask, FastAPI, Django).
 - Create a simple website where users can input their own data.
 - The website sends the data to your model running on a server.
 - The model returns a prediction, and the website displays the result.
-

Demo: A Simple ML Web App

 **Iris Flower Classifier**

```
In [3]: from IPython.display import Image, display
# Display image from file path with specific width/height
display(Image(filename=\\
"C:/Users/hwang/Dropbox/stat766Fall2021/Module4_model_deployment/iris_classifie
width=750, height=490))
```



Please enter your measurements: Sepal Length: [5.1] Sepal Width: [3.5] Petal Length: [1.4] Petal Width: [0.2]

[Predict Species]

Prediction: setosa

How it works: User Input → Web App (UI) → API Endpoint → Your ML Model → Prediction → Web App → User

Your Job Isn't Over: The Need for Monitoring

Deploying the model is not a "set it and forget it" task. The real world changes!

Why Monitor? A model that performs well today might become inaccurate and harmful tomorrow. We must ensure it continues to perform as expected.

What to Monitor After Deployment

1. Model Performance (Accuracy, F1-score, etc.)

- **How?** If you can get true labels for some of the live data (e.g., user feedback, later verification), you can calculate performance metrics.
- **Alert:** A significant drop in accuracy signals a problem.

2. Data Drift (Covariate Shift)

- **What?** The distribution of the *input data* changes from the data you trained on.
- **Example:** Your model was trained on summer user behavior, but now it's winter. The inputs look different.
- **How to Monitor?** Compare statistics (mean, standard deviation) of live features vs. training features.

3. Concept Drift

- **What?** The relationship between the input data and the target variable changes.
 - **Example:** A model predicts credit risk based on economic patterns. After a major economic crash, the old patterns no longer hold. The concept of "risk" has changed.
 - **How to Monitor?** Track performance metrics; a drop may indicate concept drift.
-


4. Data Staleness & Degradation

- **What?** The world evolves, and your training data becomes less representative over time. Your model gets "stale."
- **Solution:** Periodically **retrain your model** on newer data to keep it relevant. This is a continuous cycle.

5. System Health & Traffic

- **Latency:** How long does it take to get a prediction? (Is it still fast?)
 - **Throughput:** How many predictions can it serve per second? (Can it handle load?)
 - **Uptime:** Is the service available 24/7?
 - **Error Rates:** How often does the server crash or return an error?
-

The MLOps Lifecycle: It's a Loop!

- **Data Collection** → **Data Preparation**
- **Model Training** → **Model Evaluation**
- **Model Deployment** → **Live Monitoring**  (back to Data Collection)

Retraining is key! Monitoring tells you **when** it's time to go back to the start and improve your model with new data.

Key Takeaways

- 🚀 **Deployment** makes your model useful by putting it in the hands of users (e.g., through a web app).
 - 👁️ **Monitoring** is critical. You must watch for **model decay** caused by:
 - **Data Drift** (input data changes)
 - **Concept Drift** (the real-world relationship changes)
 - **Data Staleness** (model becomes outdated)
 - 🔄 **ML is a continuous process**, not a one-time project. Models often need to be retrained and redeployed.
-