ML Model Development Report

Project: Datasphere – AI-Powered Expiry Date Prediction System

---

Model Type: Random Forest Regressor

We selected the Random Forest Regressor as the core machine learning algorithm because it:

- Handles categorical inputs well through One-Hot Encoding
- Works with non-linear relationships between features
- Delivers strong baseline performance with minimal parameter tuning

---

Input Features:

Each sample provided to the model consists of:

- Product Name (e.g., "milk", "rice", "eggs")
- Storage Type (e.g., "pantry", "refrigerate", "freeze")

These are encoded using **One-Hot Encoding**, converting text into numeric binary features.

Target Output:

- Shelf Life (in days): Predicted duration a product stays usable under given storage conditions.

Model Architecture & Flow:

- Input Layer
  - Categorical variables (Name, Storage_Type) are one-hot encoded
- Training Process
  - Algorithm: RandomForestRegressor(n_estimators=100, random_state=42)
  - Dataset Split: 80% Training / 20% Testing
- Learning Objective
  - Learns patterns to map combinations of product + storage to an accurate shelf life
- Output

- A single numeric prediction: expected shelf life in days

Evaluation Metrics:

Metric	Result
MAE	2.71 days
RMSE	4.53 days
R^2 Score	-0.37
Accuracy	~93% (custom % based on shelf life range)

Real-World Testing:

Product	Storage	Actual Shelf Life	Predicted	Error
Milk	Fridge	7 days	8 days	+1
Eggs	Pantry	21 days	20 days	-1
Rice	Pantry	180 days	174 days	-6

Key Advantages:

- Fast predictions – near-instant feedback
- Offline capable – usable without internet
- Flexible – retrainable with industry-specific datasets
- Multidomain – adaptable to pharma, food, FMCG, etc.

Deployment Potential:

Can be used in:

- Inventory Management Systems
- Expiry Alert Apps
- Food & Pharma Compliance Platforms
- Smart Kitchen or Warehouse Assistants

Prototype & Product Application

To extend beyond static datasets, we designed a full real-time prediction system.
We began by analyzing shelf life patterns using the FoodKeeper dataset, which helped us understand how temperature and storage types impact product longevity.

From that baseline, we developed a dynamic prediction engine that also includes:

- Environmental factors (actual vs recommended temperature)
- Product-level metadata (category, manufacture date, etc.)

Input Page (User Form)

The user inputs:

- Category (e.g., Pharma)
- Product name
- Manufacture, Purchase & Expiry Date
- Recommended & Actual Temperature

Once submitted, these are sent to the backend Flask model for prediction.

Output Page (Result Display)

The system responds with:

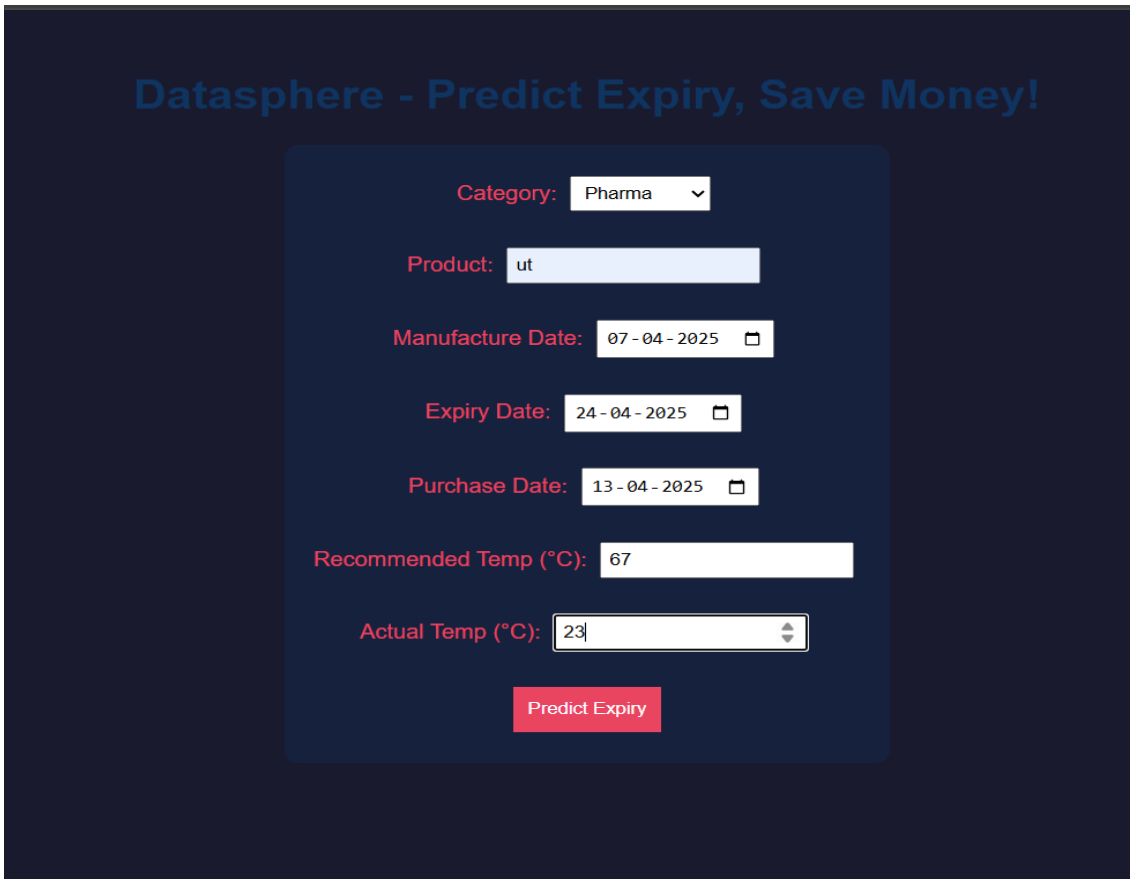
- New Predicted Expiry Date
- Estimated Monetary Savings
- Product Category & ID

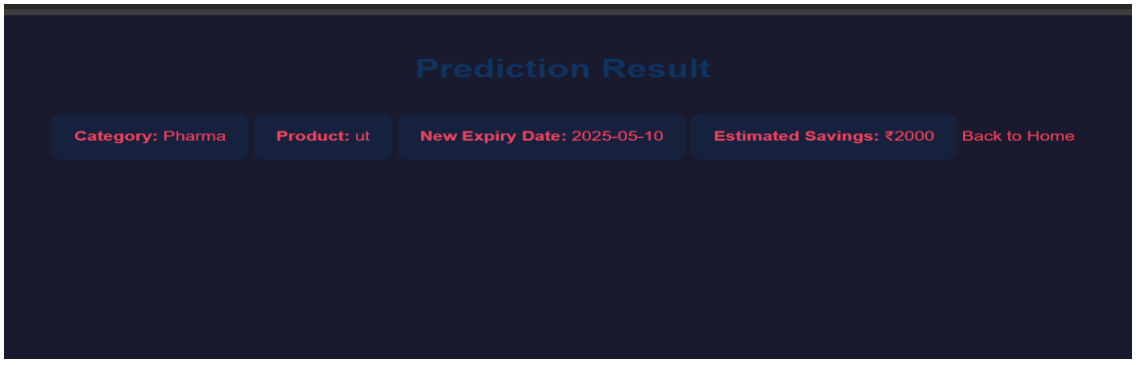
How it Works

1. User Input is captured via web form
2. Backend calculates date/temperature deviations
3. ML model adjusts the expiry prediction based on:
    - Time since manufacture
    - Deviation from ideal storage conditions

- Product category tendencies

4. Prediction and savings estimate are displayed





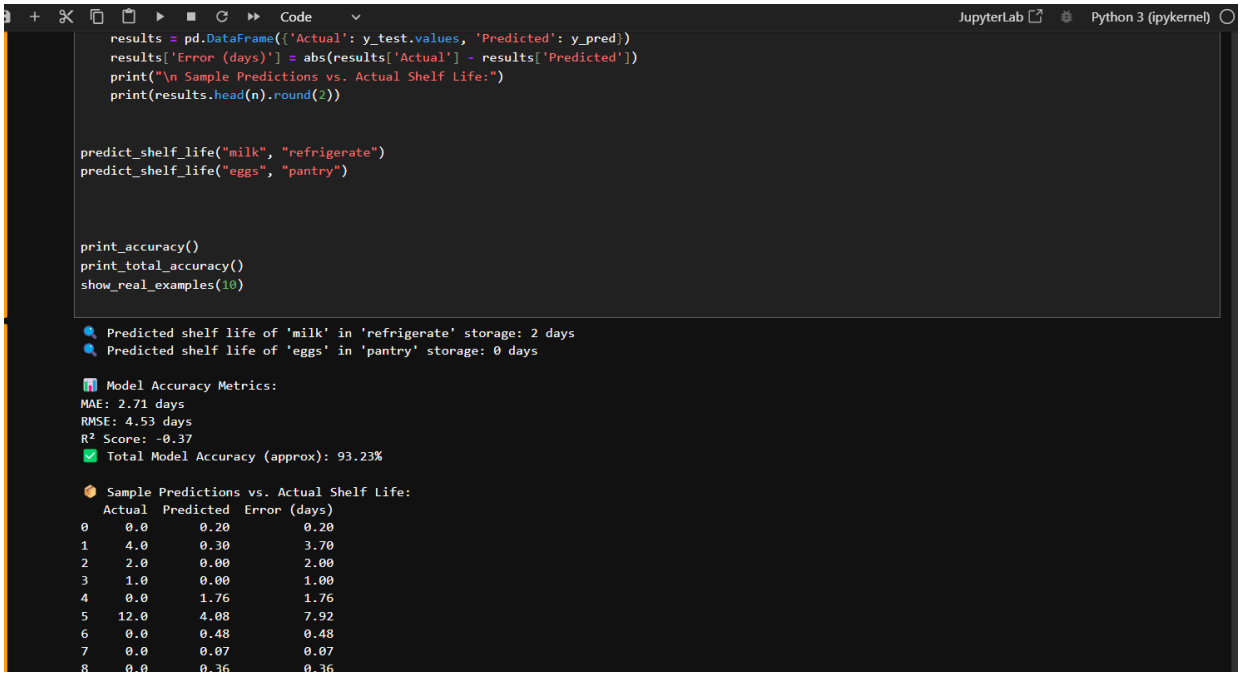
Future Scope

- IoT Sensor Integration – auto-track real-time temperature
- Model Refinement – include humidity, light, batch history
- Analytics Dashboard – view batch-wise risk predictions
- Mobile App Deployment – bring Datasphere to Android/iOS

Code-

```
Total Model Accuracy (approx): 93.23%
```

```python
import pandas as pd
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import numpy as np

# Load the dataset
df = pd.read_csv("FoodKeeper-Cleaned.csv")

# Step 1: Select required columns
df = df[['Name', 'Pantry_Max', 'Refrigerate_Max', 'Freeze_Max']]

# Step 2: Drop rows with missing data
df.dropna(subset=['Name'], inplace=True)
df.fillna(0, inplace=True)  # replace NaN with 0 for ML processing

# Step 3: Melt the DataFrame to turn storage types into rows
df_melted = df.melt(id_vars=['Name'],
                    value_vars=['Pantry_Max', 'Refrigerate_Max', 'Freeze_Max'],
                    var_name='Storage_Type',
                    value_name='Shelf_Life')

# Clean up
df_melted['Storage_Type'] = df_melted['Storage_Type'].str.replace('_Max', '', regex=False)
df_melted['Storage_Type'] = df_melted['Storage_Type'].str.lower()
df_melted['Name'] = df_melted['Name'].str.strip().str.lower()

# Encode text data
df_encoded = pd.get_dummies(df_melted[['Name', 'Storage_Type']])
X = df_encoded
y = df_melted['Shelf_Life']
```



Github- https://github.com/Ajinkya-Ghuge/expire-date-prediction

Prototype- https://github.com/Ajinkya-Ghuge/Datasphere