AI-Powered Concrete Strength Prediction & Supplier Recommendation System

🧭 Project Overview

The **Ai Powered Concrete Strength** is a machine learning–driven web application that predicts the **compressive strength of concrete** based on its composition and age.

In addition to prediction, the system provides **supplier recommendations** for concrete materials based on cost, reliability, and delivery time — helping engineers and construction companies make informed decisions.

The complete system is built using **FastAPI**, **scikit-learn**, and **MongoDB Atlas**, and is containerized using **Docker** for scalable deployment.

🎯 Project Objectives

#	Objective	Status
1	Gather and preprocess a reliable concrete strength dataset	☑ Achieved
2	Train a machine learning regression model to predict concrete compressive strength	☑ Achieved
3	Evaluate model accuracy and optimize performance	☑ Achieved (90%)
4	Integrate supplier recommendation logic	☑ Achieved
5	Build a full-stack FastAPI-based web interface for prediction and supplier selection	☑ Achieved
6	Connect application with a cloud database (MongoDB Atlas)	☑ Achieved
7	Dockerize the entire application for deployment	☑ Achieved
8	Push final Docker image to Docker Hub for cloud deployment	☑ Achieved

---

📊 Dataset Description

Source

The dataset was gathered from **kaggle**, primarily based on the , a well-known dataset in regression analysis and materials prediction.

Dataset File

`concrete_data.csv`

Features

Column	Description
`cement`	Amount of cement (kg/m^3)
`slag`	Blast furnace slag (kg/m^3)
`flyash`	Fly ash (kg/m^3)
`water`	Water content (kg/m^3)
`superplasticizer`	Chemical admixture (kg/m^3)
`coarseaggregate`	Coarse aggregate (kg/m^3)
`fineaggregate`	Fine aggregate (kg/m^3)
`age`	Age of concrete sample (days)
`csMPa`	Compressive strength (MPa) – Target variable

🧠 Machine Learning Model

Algorithm

The project used a **Random Forest Regressor** (from scikit-learn) to predict compressive strength.

Workflow

1. Loaded and cleaned data using `pandas` and `numpy`
2. Normalized feature values
3. Split into **80% training** and **20% testing** data
4. Trained the model using **RandomForestRegressor**
5. Evaluated with **R^2 score** and **MSE**

Results

- **Accuracy (R^2 Score):** `0.95` → **95%**
- **Error Rate:** Very low
- **Interpretation:** The model accurately captures relationships between ingredients and compressive strength.

🏗 Application Features

1. **Concrete Strength Prediction**
   - User enters values for materials and concrete age.
   - Model predicts compressive strength in MPa.
2. **Supplier Recommendations**
   - Suggests top suppliers ranked by:
     - Total cost

        - Delivery time
        - Reliability score
    - Supplier data stored in MongoDB Atlas.

3. **Prediction History**

    - Each prediction (inputs + result) saved in the database.

4. **Responsive Web Interface**

    - Built using **FastAPI** and **Jinja2** templates.
    - User-friendly forms and data tables.

🗄️ Database Configuration

- **Database:** MongoDB Atlas (Cloud-hosted)
- **Collections:**
    - `predictions` → stores user inputs and predicted results
    - `suppliers` → supplier details (cost, reliability, delivery)

Environment Variables

```
MONGODB_URI=mongodb+srv://tabarakkhan:****@cluster0.rmellx6.mongodb.net/?
appName=Cluster0
MONGODB_DB=zynex_db
MODEL_PATH=models/concrete_strength_model.pkl
SUPPLIERS_CSV=data/suppliers.csv
```

Category	Technology
Frontend	HTML, CSS, Jinja2
Backend	FastAPI
Machine Learning	scikit-learn, pandas, numpy
Database	MongoDB Atlas
Deployment	Docker
Container Registry	Docker Hub
Language	Python 3.10

Dockerization & Pushing

Dockerfile

```
FROM python:3.10-slim
WORKDIR /app
COPY . .
RUN pip install --no-cache-dir -r requirements.txt
EXPOSE 8000
CMD ["uvicorn", "app.main:app", "--host", "0.0.0.0", "--port", "8000"]
```

Docker Commands

```
docker build -t zynex-ai-app .
docker tag zynex-ai-app tabarakallah/zynex-ai-app:latest
docker push tabarakallah/zynex-ai-app:latest
```

Achievement Summary

Task	Status	Description
Data Gathering	☑	Collected and cleaned dataset
Model Training	☑	Trained Random Forest model
Model Evaluation	☑	Achieved 95% accuracy
Web Development	☑	Built FastAPI frontend/backend
Database Integration	☑	Linked MongoDB Atlas
Supplier System	☑	Implemented supplier ranking logic
Dockerization	☑	Built and containerized app
Cloud Registry	☑	Uploaded to Docker Hub

DockerHub Image Link:https://hub.docker.com/r/tabarakallah/zynex-ai-app

Conclusion

The Ai Powered Concrete Strength Insight Platform successfully achieved all project goals:

Trained an accurate machine learning model for predicting concrete strength (95% accuracy)

Developed a full-stack web interface

Integrated a supplier recommendation engine

Connected with a real MongoDB Atlas cloud database

Containerized and deployed the application using Docker

This demonstrates a complete end-to-end AI system that can be easily deployed in production or scaled for industry use.

Future Improvements

1. Deploy to a free hosting provider (Render, Railway, Hugging Face Spaces)
2. Add user authentication for suppliers and engineers
3. Create analytics dashboards
4. Extend model with deep learning for more robust predictions

---

Author

Tabarak Khan AI & Machine Learning Developer Docker Hub – tabarakallah