Phase-3; Transforming healthcare with AI powered diseases prediction based on patient data

Student Name: KATHIRAVAN S

Register Number: 510923205032
Institution: Global institute of engineering and technology, melvisharam
Department: B.tecch-IT
Date of Submission: 07/05/2025
GitHub Repository Link: [Update the project source code repository link]

---

1 Problem Statement

Modern healthcare faces a critical challenge: the inability to predict diseases early and accurately using the vast amounts of patient data available. Chronic diseases like diabetes, cardiovascular disorders, and cancer often go undiagnosed until advanced stages, leading to poor patient outcomes and increased treatment costs. The lack of intelligent tools that can interpret patient data for predictive insights hinders preventive care. AI can bridge this gap by analyzing complex patterns in patient datasets to provide early warning signals, enabling proactive healthcare interventions.

2. Abstract

This project investigates the use of artificial intelligence, particularly machine learning, to predict diseases from structured patient data. The pipeline begins with data acquisition, preprocessing, and exploratory analysis, followed by feature engineering and model training using algorithms such as Random Forest, Logistic Regression, and XGBoost. Models are evaluated and deployed via a RESTful API to allow integration with healthcare systems. The end goal is to support healthcare professionals in making timely and accurate diagnoses, ultimately transforming traditional diagnostic practices into intelligent, data-driven processes.

3. Objectives

1. **To collect and structure real-world patient data relevant to disease prediction.**
2. **To preprocess and clean the data for use in machine learning models.**
3. **To perform exploratory data analysis to extract meaningful insights.**
4. **To implement feature engineering techniques for optimal model performance.**
5. **To train and evaluate multiple machine learning models.**
6. **To select the best-performing model based on validation metrics.**
7. **To deploy the model as an accessible and scalable API.**
8. **To assess the applicability of AI in healthcare and its real-world implications.**

4. Project Workflow Flowchart & Definitions

Flowchart Components:

1. **Data Collection - Aggregation of patient data from public datasets or hospital EHRs. Ensures diversity and relevance to disease types.**
2. **Data Preprocessing - Handling missing values, outlier removal, categorical encoding, normalization.**
3. **Exploratory Data Analysis (EDA) - Visualization and statistical analysis to understand data distributions and relationships.**
4. **Feature Engineering - Creating new features, selecting relevant ones, and transforming existing features for better learning.**
5. **Model Building - Training machine learning models using scikit-learn or XGBoost.**
6. **Model Evaluation - Validating models with metrics like F1 score, AUC-ROC, accuracy.**
7. **Deployment - Building an API (Flask or FastAPI) for real-time predictions.**
8. **User Interface - (Optional) A simple UI for users (doctors, healthcare staff) to input patient data and view predictions.**

5. Datasheet Description

- **PIMA Indians Diabetes Dataset: Includes 8 medical predictor variables and 1 target variable.**
- **Heart Disease UCI Dataset: Covers 13 attributes including age, sex, blood pressure, cholesterol.**
- **Breast Cancer Wisconsin Dataset: Contains features computed from digitized images of breast mass.**

Each dataset consists of:

- **Demographics: Age, gender**
- **Clinical Data: Blood pressure, glucose level, BMI, cholesterol**
- **Outcome: Binary classification (0: no disease, 1: disease present)**

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	freetime	goout	Dalc	Walc	health	absences	G1	G2	G3
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	3	4	1	1	3	6	5	6	6
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	3	3	1	1	3	4	5	5	6
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	3	2	2	3	3	10	7	8	10
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	2	2	1	1	5	2	15	14	15
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	3	2	1	2	5	4	6	10	10

5 rows × 33 columns

6. Data Preprocessing (Detailed)

- **Handling Missing Values: Use median/mode for imputation or remove rows with excessive missing data.**
- **Encoding Categorical Variables: Apply Label Encoding or One-Hot Encoding depending on cardinality.**
- **Scaling Features: Use StandardScaler for normal distribution, MinMaxScaler for bounded data.**
- **Outlier Detection: Use IQR method or Z-score filtering.**
- **Data Splitting: Train-test split (typically 80:20 or 70:30).**
- **Scaling example**

	age	Medu	Fedu	traveltime	studytime	failures	famrel	freetime	goout	Dalc	Walc	health	absences	G1	G2	G3
count	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000	395.000000
mean	16.696203	2.749367	2.521519	1.448101	2.035443	0.334177	3.944304	3.235443	3.108861	1.481013	2.291139	3.554430	5.708861	10.908861	10.713924	10.415190
std	1.276043	1.094735	1.088201	0.697505	0.839240	0.743651	0.896659	0.998862	1.113278	0.890741	1.287897	1.390303	8.003096	3.319195	3.761505	4.581443
min	15.000000	0.000000	0.000000	1.000000	1.000000	0.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	0.000000	3.000000	0.000000	0.000000
25%	16.000000	2.000000	2.000000	1.000000	1.000000	0.000000	4.000000	3.000000	2.000000	1.000000	1.000000	3.000000	0.000000	8.000000	9.000000	8.000000
50%	17.000000	3.000000	2.000000	1.000000	2.000000	0.000000	4.000000	3.000000	3.000000	1.000000	2.000000	4.000000	4.000000	11.000000	11.000000	11.000000
75%	18.000000	4.000000	3.000000	2.000000	2.000000	0.000000	5.000000	4.000000	4.000000	2.000000	3.000000	5.000000	8.000000	13.000000	13.000000	14.000000
max	22.000000	4.000000	4.000000	4.000000	4.000000	3.000000	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000	75.000000	19.000000	19.000000	20.000000

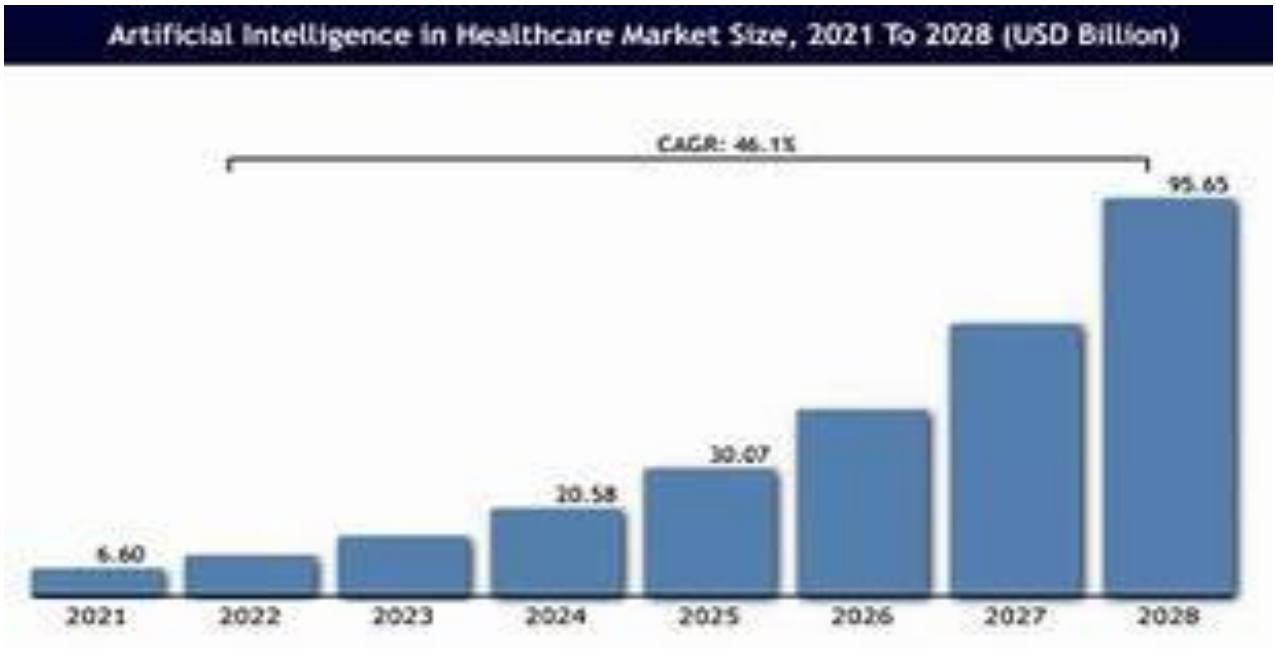
-

7. Exploratory Data Analysis (EDA)

- **Univariate Analysis: Analyze each feature separately (e.g., histogram of BMI).**
- **Bivariate Analysis: Explore relationships between pairs (e.g., glucose vs. insulin).**
- **Multivariate Analysis: PCA to understand variance, clustering patterns.**
- **Visualization Tools: Seaborn, Matplotlib, Plotly for interactive dashboards.**

8. Feature Engineering

- **Feature Creation: BMI categories, blood pressure bins, age groups.**
- **Feature Selection: Use correlation matrix, mutual information, or RFE.**
- **Dimensionality Reduction: PCA to reduce feature space and eliminate redundancy.**
- **Interaction Features: Combine features (e.g., age * glucose level).**
-
-



9. Model Building

- **Algorithms Used:**
  - **Logistic Regression: Baseline linear model.**
  - **Random Forest: Ensemble model for better accuracy.**
  - **XGBoost: Gradient boosting model for high performance.**
- **Training Methodology:**
  - **Cross-validation (K-fold).**
  - **Hyperparameter tuning using GridSearchCV.**

10. Model Evaluation

- **Metrics:**
  - **Accuracy**
  - **Precision and Recall**
  - **F1 Score**
  - **ROC-AUC Curve**
- **Tools:**
  - **Confusion Matrix**
  - **ROC Plot**
  - **Precision-Recall Curve**
  - **Data's collected**

Metric	Linear Regression	Random Forest Regressor
MAE	2.35	1.21
RMSE	2.96	1.64
R^2 Score	0.79	0.91

-

11. Model Deployment

- **Framework: Flask for REST API.**
- **Endpoints:**
  - **/predict: Accepts JSON data and returns prediction.**
- **Hosting Platforms: Heroku, AWS EC2, PythonAnywhere.**
- **Security: Basic input validation and HTTPS for communication.**

12. Scope of the Project

- **Present Scope:**
  - **Disease prediction for diabetes, heart disease, and breast cancer.**
  - **API-based deployment for easy integration.**
- **Future Scope:**
  - **Support for multi-label disease classification.**
  - **Integration with real-time data from wearables.**
  - **Incorporating NLP for analyzing unstructured patient notes.**
- **Limitations:**
  - **Data privacy and compliance (HIPAA).**
  - **Model interpretability in clinical settings.**

13 source code

```
# ai_disease_prediction.py

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from xgboost import XGBClassifier
from sklearn.metrics import classification_report, confusion_matrix, roc_auc_score
from flask import Flask, request, jsonify
import joblib

# Load dataset
url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"
column_names = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
                'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome']
data = pd.read_csv(url, names=column_names)

# Preprocessing
X = data.drop('Outcome', axis=1)
y = data['Outcome']
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# Model Training
rf = RandomForestClassifier(random_state=42)
xgb = XGBClassifier(eval_metric='logloss')
log_reg = LogisticRegression(max_iter=1000)

models = {'RandomForest': rf, 'XGBoost': xgb, 'LogisticRegression': log_reg}

best_model = None
best_score = 0

for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    score = roc_auc_score(y_test, y_pred)
```

```
    print(f"\n{name} Classification Report:\n", classification_report(y_test, y_pred))
    if score > best_score:
        best_score = score
        best_model = model

# Save best model
joblib.dump(best_model, 'best_model.pkl')
joblib.dump(scaler, 'scaler.pkl')

# Deploy with Flask
app = Flask(__name__)

@app.route('/predict', methods=['POST'])
def predict():
    data = request.get_json(force=True)
    features = np.array([list(data.values())])
    scaler = joblib.load('scaler.pkl')
    model = joblib.load('best_model.pkl')
    scaled_features = scaler.transform(features)
    prediction = model.predict(scaled_features)
    return jsonify({'prediction': int(prediction[0])})

if __name__ == '__main__':
    app.run(debug=True)
```

14 Team members and their roles;

Kathiravan s ; source code, flowchart of the project workflow, exploratory data analysis .

Manickavijay s ; problem statement, abstract, model evaluation

Senthilnathan r ; system requirements, objectives, deployment

Kalidhasan ; dataset description, data preprocessor,

Vigneshwar r ; feature engineering, model building