AI Health Predictor - Complete Project Documentation

Table of Contents

Project Overview

AI Health Predictor is a comprehensive machine learning-based web application designed for early disease detection and risk assessment. The system utilizes advanced machine learning algorithms to predict the likelihood of three major diseases: Diabetes, Heart Disease, and Parkinson's Disease.

Project Highlights

- **Multi-Disease Prediction**: Single platform for predicting multiple diseases
- **High Accuracy Models**: Achieves 78-95% prediction accuracy
- **User-Friendly Interface**: Built with Streamlit for easy interaction
- **Real-Time Predictions**: Instant risk assessment based on health parameters
- **Educational Purpose**: Demonstrates practical ML applications in healthcare

Technical Architecture

Technology Stack

Backend Technologies:

- Python 3.8+
- scikit-learn (Machine Learning)
- pandas (Data Processing)
- numpy (Numerical Computing)
- pickle (Model Serialization)

Frontend Framework:

- Streamlit (Web Application Framework)

Machine Learning Algorithms:

- Support Vector Machine (SVM)
- Logistic Regression
- Random Forest Classifier

System Architecture

The application follows a modular architecture with the following components:

1. **Data Input Layer**: User interface for collecting health parameters
2. **Preprocessing Layer**: Data validation and normalization
3. **Model Layer**: Trained ML models for disease prediction
4. **Presentation Layer**: Results display with recommendations

Disease Prediction Models

1. Diabetes Prediction Model

Algorithm: Support Vector Machine (SVM)
Accuracy: 78-81%
Dataset: PIMA Indians Diabetes Database

Input Features (8 parameters):

- Number of Pregnancies (0-17)
- Glucose Level (mg/dL)
- Blood Pressure (mm Hg)
- Skin Thickness (mm)
- Insulin Level (mu U/ml)
- Body Mass Index (BMI)
- Diabetes Pedigree Function
- Age (years)

Model Training Process:

```
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler

# Standardize features<a></a>
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_train)

# Train SVM model<a></a>
model = SVC(kernel='linear', C=1.0, random_state=42)
model.fit(X_scaled, y_train)
```

Risk Assessment Criteria:

- Glucose > 140 mg/dL: High risk indicator
- BMI > 30: Obesity-related risk
- Age > 45: Age-related risk factor
- Diabetes Pedigree > 0.5: Genetic predisposition
- Blood Pressure > 80 mm Hg: Hypertension indicator

2. Heart Disease Prediction Model

Algorithm: Logistic Regression
Accuracy: 85%
Dataset: UCI Heart Disease Dataset

Input Features (11 parameters):

- Age (29-77 years)
- Sex (Male/Female)
- Chest Pain Type (0-3)
- Resting Blood Pressure (mm Hg)
- Cholesterol (mg/dL)
- Fasting Blood Sugar (>120 mg/dl)
- Resting ECG Results (0-2)
- Maximum Heart Rate
- Exercise Induced Angina
- ST Depression (Oldpeak)
- ST Slope (0-2)

Model Training Process:

```
from sklearn.linear_model import LogisticRegression

# Train Logistic Regression model<a></a>
model = LogisticRegression(max_iter=1000, random_state=42)
model.fit(X_train_scaled, y_train)
```

Risk Assessment Criteria:

- Age > 55: Increased cardiovascular risk
- Cholesterol > 240 mg/dL: High cholesterol
- Max Heart Rate < 100: Poor cardiac response
- Chest Pain Type 3 (Asymptomatic): Critical indicator
- Oldpeak > 2: Significant ST depression

3. Parkinson's Disease Prediction Model

Algorithm: Support Vector Machine (SVM)
Accuracy: 87-95%
Dataset: UCI Parkinson's Dataset

Input Features (13 vocal parameters):

- MDVP:Fo(Hz) - Average vocal frequency

- MDVP:Fhi(Hz) - Maximum frequency
- MDVP:Flo(Hz) - Minimum frequency
- MDVP:Jitter(%) - Frequency variation
- MDVP:Shimmer - Amplitude variation
- NHR - Noise to harmonics ratio
- HNR - Harmonics to noise ratio
- RPDE - Recurrence period density
- DFA - Detrended fluctuation analysis
- Spread1, Spread2 - Nonlinear measures
- D2 - Correlation dimension
- PPE - Pitch period entropy

Model Training Process:

```
from sklearn.svm import SVC

# Train SVM with RBF kernel<a></a>
model = SVC(kernel='rbf', C=1.0, gamma='scale', random_state=42)
model.fit(X_train_scaled, y_train)
```

Risk Assessment Criteria:

- Jitter > 0.01: Voice instability
- Shimmer > 0.05: Amplitude irregularity
- NHR > 0.05: Increased noise ratio
- HNR < 15: Reduced harmonics
- RPDE > 0.6: Complexity increase

Project Structure

File Organization

```
ai-health-predictor/
├── app.py                      # Main Streamlit application
├── train_models.py             # Model training script
├── requirements.txt            # Python dependencies
├── models/
│   ├── diabetes_model.pkl      # Trained diabetes model
│   ├── heart_model.pkl         # Trained heart model
│   └── parkinsons_model.pkl    # Trained Parkinson's model
├── scalers/
│   ├── scaler_diabetes.pkl     # Feature scaler
│   ├── scaler_heart.pkl        # Feature scaler
│   └── scaler_parkinsons.pkl   # Feature scaler
├── data/
```

```
│   ├── diabetes.csv            # Training dataset
│   ├── heart.csv               # Training dataset
│   └── parkinsons.csv          # Training dataset
└── README.md                   # Project documentation
```

Installation and Setup

Prerequisites

- Python 3.8 or higher
- pip (Python package manager)
- Git (for version control)

Step 1: Clone or Download Project

```
git clone https://github.com/yourusername/ai-health-predictor.git
cd ai-health-predictor
```

Step 2: Create Virtual Environment

```
# Windows<a></a>
python -m venv venv
venv\\Scripts\\activate

# macOS/Linux<a></a>
python3 -m venv venv
source venv/bin/activate
```

Step 3: Install Dependencies

```
pip install -r requirements.txt
```

Step 4: Download Datasets

Diabetes Dataset:

- Source: Kaggle PIMA Indians Diabetes Database
- URL: https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database

Heart Disease Dataset:

- Source: UCI Machine Learning Repository
- URL: https://archive.ics.uci.edu/ml/datasets/heart+Disease

Parkinson's Dataset:

- Source: UCI Machine Learning Repository
- URL: https://archive.ics.uci.edu/ml/datasets/parkinsons

Step 5: Train Models (Optional)

```
python train_models.py
```

Step 6: Run Application

```
streamlit run app.py
```

The application will open in your default browser at `http://localhost:8501`

Deployment Guide

Deploy on Streamlit Cloud (Free)

Step 1: Prepare Repository

- Create GitHub repository
- Push all project files
- Include requirements.txt

Step 2: Deploy

- Visit: https://share.streamlit.io/
- Sign in with GitHub
- Click "New app"
- Select repository and branch
- Set main file: app.py
- Click "Deploy"

Step 3: Access Application

- Application URL: `https://yourapp.streamlit.app/`
- Share with users

Deploy on Heroku

```
# Create Procfile<a></a>
echo "web: streamlit run app.py --server.port=$PORT" &gt; Procfile

# Create setup.sh<a></a>
cat &gt; setup.sh &lt;&lt; EOF
mkdir -p ~/.streamlit/
```

```
echo "[server]
headless = true
port = $PORT
enableCORS = false
" &gt; ~/.streamlit/config.toml
EOF

# Deploy<a></a>
heroku create your-app-name
git push heroku main
```

Application Features

1. Home Page

- Welcome screen with project overview
- Disease prediction cards
- Navigation instructions
- Important disclaimer

2. Disease Prediction Pages

- Interactive input forms
- Real-time validation
- Instant predictions
- Color-coded risk assessment
- Personalized recommendations

3. About Page

- Project information
- Model details
- Dataset sources
- Technology stack
- Usage instructions

User Guide

How to Use the Application

Step 1: Navigate

- Use sidebar menu to select disease prediction

Step 2: Enter Parameters

- Fill in all required health parameters
- Refer to tooltips for guidance
- Use default values as reference

Step 3: Get Prediction

- Click prediction button
- Wait for analysis (1-2 seconds)
- Review results and recommendations

Step 4: Interpret Results

- Green box: Low risk
- Red box: High risk
- Follow provided recommendations
- Consult healthcare professionals

Model Performance Metrics

Diabetes Model

- **Accuracy**: 78-81%
- **Precision**: 76.8-81.0%
- **Recall**: 75.9-80.2%
- **F1-Score**: 76.1-81.2%

Heart Disease Model

- **Accuracy**: 85%
- **Precision**: 84.5%
- **Recall**: 83.7%
- **F1-Score**: 84.1%

Parkinson's Model

- **Accuracy**: 87-95%
- **Precision**: 82.0-95.0%
- **Recall**: 81.5-94.5%
- **F1-Score**: 81.5-94.7%

Data Preprocessing

Feature Scaling

All models use StandardScaler for feature normalization:

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Benefits:

- Improved model convergence
- Better performance with SVM and Logistic Regression
- Consistent feature importance

Data Splitting

Train-test split ratio: 80-20

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

Important Considerations

Medical Disclaimer

⚠ **Critical Information:**

1. **Not Medical Advice**: This application is for educational purposes only
2. **Consult Professionals**: Always seek advice from qualified healthcare providers
3. **No Diagnosis**: Predictions are not medical diagnoses
4. **Emergency**: For medical emergencies, call emergency services immediately
5. **Data Privacy**: Application does not store personal health information

Limitations

Model Limitations:

- Based on historical data patterns
- May not capture all medical complexities

- Cannot replace clinical judgment
- Accuracy varies with input quality

Technical Limitations:

- Requires accurate parameter input
- Limited to three disease categories
- Simplified risk assessment
- No personalized treatment plans

Future Enhancements

Planned Features

1. **Additional Diseases**
    - Cancer prediction
    - Kidney disease
    - Liver disease
    - Respiratory disorders
2. **Advanced ML Models**
    - Deep learning integration
    - Ensemble methods
    - Neural networks
3. **User Features**
    - User accounts
    - Prediction history
    - Progress tracking
    - Report generation
4. **Integration**
    - EHR system integration
    - Wearable device data
    - Lab result import
    - API for mobile apps
5. **Visualization**
    - Interactive charts
    - Risk trend analysis
    - Comparative analytics
    - 3D visualizations

Troubleshooting

Common Issues

Issue 1: Module Not Found

```
Solution: pip install -r requirements.txt
```

Issue 2: Port Already in Use

```
Solution: streamlit run app.py --server.port=8502
```

Issue 3: Model Files Missing

```
Solution: Run train_models.py to generate model files
```

Issue 4: Streamlit Not Starting

```
Solution: Check Python version (3.8+)
Reinstall: pip install --upgrade streamlit
```

Contributing

How to Contribute

1. Fork the repository
2. Create feature branch
3. Make improvements
4. Test thoroughly
5. Submit pull request

Areas for Contribution

- Additional disease models
- Improved UI/UX
- Better accuracy
- Documentation
- Bug fixes

References

Datasets

[1] PIMA Indians Diabetes Database. UCI Machine Learning Repository.
[2] Heart Disease Dataset. UCI Machine Learning Repository.
[3] Parkinson's Disease Dataset. UCI Machine Learning Repository.

Research Papers

[4] Detrano, R., et al. "International application of a new probability algorithm for the diagnosis of coronary artery disease." American Journal of Cardiology, 1989.

[5] Little, M. A., et al. "Exploiting Nonlinear Recurrence and Fractal Scaling Properties for Voice Disorder Detection." BioMedical Engineering OnLine, 2007.

Technology Documentation

[6] scikit-learn Documentation. https://scikit-learn.org/
[7] Streamlit Documentation. https://docs.streamlit.io/
[8] Python Documentation. https://docs.python.org/

Conclusion

AI Health Predictor demonstrates the potential of machine learning in healthcare by providing accessible, accurate disease risk assessments. This educational project showcases how AI can support early disease detection while emphasizing the importance of professional medical consultation.

Key Takeaways

✅ Machine learning can achieve high accuracy in disease prediction
✅ Multiple diseases can be assessed in a single platform
✅ User-friendly interfaces make ML accessible to non-technical users
✅ AI tools should complement, not replace, medical professionals
✅ Continuous improvement and validation are essential

Contact and Support

For questions, suggestions, or issues:

- GitHub Issues: Report bugs and request features
- Email: support@aihealthpredictor.com
- Documentation: Visit project wiki

Built with Python, scikit-learn, and Streamlit