📈 LSTM Stock Price Forecasting Tool

A comprehensive deep learning solution for stock price prediction using LSTM neural networks, with real-time data fetching from Yahoo Finance and an interactive Dash dashboard for visualization.

🌟 Features

- **LSTM Deep Learning Model**: Multi-layer LSTM architecture for time series forecasting
- **Real-time Data**: Automatic data fetching from Yahoo Finance API
- **Interactive Dashboard**: Beautiful Dash web interface for visualization
- **Comprehensive Metrics**: MSE, RMSE, MAE, and MAPE evaluation
- **Future Forecasting**: Predict stock prices days/weeks ahead
- **Model Persistence**: Save and load trained models
- **Robust Error Handling**: Comprehensive validation and error management

🏗️ Architecture

LSTM Model Structure

- **Layer 1**: LSTM (50 units) with return sequences + Dropout (0.2)
- **Layer 2**: LSTM (50 units) with return sequences + Dropout (0.2)
- **Layer 3**: LSTM (50 units) + Dropout (0.2)
- **Layer 4**: Dense (25 units)
- **Output**: Dense (1 unit) - Price prediction

Key Components

- **Data Pipeline**: Fetch → Normalize → Sequence Creation → Train/Test Split
- **Training**: Early stopping, learning rate reduction, validation monitoring
- **Forecasting**: Iterative prediction with sequence updating

📋 Prerequisites

- Python 3.8 or higher
- pip package manager
- Internet connection (for data fetching)

🚀 Installation

1. Clone or Download the Project

```bash
# Create project directory
mkdir lstm-stock-forecasting
cd lstm-stock-forecasting
```

2. Install Dependencies

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

Dependencies:

- `numpy`: Numerical computations
- `pandas`: Data manipulation
- `yfinance`: Yahoo Finance data API
- `scikit-learn`: Data preprocessing and metrics
- `tensorflow`: Deep learning framework
- `dash`: Interactive web dashboard
- `plotly`: Visualization library

📁 Project Structure

```
lstm-stock-forecasting/
│
├── main.py              # Core LSTM model implementation
├── dashboard.py         # Interactive Dash dashboard
├── requirements.txt     # Python dependencies
├── README.md            # This file
│
├── [Generated Files]
├── *.h5                 # Saved model files
├── *_scaler.pkl         # Saved scaler objects
└── predictions.pkl      # Saved prediction results
```

🎯 Usage

Method 1: Command Line Training

Train a model directly from the command line:

```bash
python main.py
```

This will:

1. Fetch historical data for AAPL (default)
2. Train the LSTM model
3. Evaluate performance
4. Generate 30-day forecast
5. Save model and predictions

Customize the training:

Edit the configuration in `main.py`:

```python
# Configuration
TICKER = 'AAPL'          # Change stock ticker
START_DATE = '2020-01-01' # Change start date
SEQUENCE_LENGTH = 60     # Lookback window
EPOCHS = 50              # Training epochs
BATCH_SIZE = 32          # Batch size
```

Method 2: Interactive Dashboard

Launch the web dashboard for interactive training and visualization:

```bash
python dashboard.py
```

Then open your browser and navigate to:

```
http://localhost:8050
```

Dashboard Features:

- Enter any stock ticker (AAPL, GOOGL, TSLA, etc.)
- Set forecast horizon (1-90 days)
- Train models with one click
- View real-time training progress
- Interactive charts with zoom and hover
- Performance metrics display

📊 Dashboard Guide

Main Components

1. **Control Panel**
   - Stock Ticker Input: Enter any valid stock symbol
   - Forecast Days: Choose prediction horizon
   - Train & Predict Button: Start model training
2. **Metrics Cards**
   - MSE (Mean Squared Error)
   - RMSE (Root Mean Squared Error)
   - MAE (Mean Absolute Error)
   - MAPE (Mean Absolute Percentage Error)
3. **Historical Price Chart**
   - Full historical price data
   - Interactive zoom and pan
   - Hover for exact values
4. **Forecast Chart**
   - Last 180 days of history
   - Test set predictions
   - Future forecast visualization
   - Color-coded for clarity
5. **Performance Chart**
   - Actual vs Predicted comparison
   - Model accuracy visualization
   - Test set evaluation

🔧 Advanced Usage

Custom Model Configuration

Modify model hyperparameters:

```python
from main import StockPredictor

# Initialize with custom parameters
predictor = StockPredictor(
    sequence_length=90,    # Use 90 days lookback
    lstm_units=100,        # More LSTM units
    dropout_rate=0.3       # Higher dropout
)
```

Multiple Stock Analysis

```python
tickers = ['AAPL', 'GOOGL', 'MSFT', 'TSLA']

for ticker in tickers:
    predictor = StockPredictor()
    data = predictor.fetch_data(ticker, '2020-01-01', '2024-01-01')
    X_train, y_train, X_test, y_test, _ = predictor.prepare_data(data)
    predictor.train(X_train, y_train, X_test, y_test)
    predictor.save_model(f'{ticker}_model.h5')
```

Load Existing Model

```python
from main import StockPredictor

predictor = StockPredictor()
predictor.load_model('AAPL_model.h5')

# Make predictions on new data
predictions = predictor.predict(X_new)
```

📈 Interpretation Guide

Metrics Explained

- **MSE (Mean Squared Error)**: Average squared difference between predicted and actual. Lower is better.
- **RMSE (Root MSE)**: Square root of MSE, in same units as price. Easier to interpret.
- **MAE (Mean Absolute Error)**: Average absolute difference. Shows typical prediction error.
- **MAPE (Mean Absolute Percentage Error)**: Percentage error. Good for comparing across different price ranges.

Model Performance

- **MAPE < 5%**: Excellent performance
- **MAPE 5-10%**: Good performance
- **MAPE 10-20%**: Acceptable performance
- **MAPE > 20%**: Consider retraining or adjusting hyperparameters

🛠️ Troubleshooting

Common Issues

1. Installation Errors

```bash
# Update pip first
pip install --upgrade pip

# Install with specific versions
pip install tensorflow==2.13.0
```

2. Yahoo Finance Data Issues

```python
# Check ticker is valid
import yfinance as yf
stock = yf.Ticker("AAPL")
print(stock.info)
```

3. Memory Issues

```python
```

```
# Reduce batch size
predictor.train(X_train, y_train, X_test, y_test, batch_size=16)

# Reduce sequence length
predictor = StockPredictor(sequence_length=30)
```

4. Dashboard Not Loading

```bash
# Check port availability
lsof -i :8050

# Use different port
app.run_server(debug=True, port=8051)
```

GPU Support (Optional)

For faster training with GPU:

```bash
# Install TensorFlow with GPU support
pip install tensorflow-gpu==2.13.0

# Verify GPU availability
python -c "import tensorflow as tf; print(tf.config.list_physical_devices('GPU'))"
```

📚 Understanding LSTM for Stock Prediction

Why LSTM?

- **Memory**: LSTMs can remember long-term patterns
- **Sequential Data**: Perfect for time series
- **Non-linearity**: Captures complex price movements
- **Gate Mechanism**: Learns what to remember and forget

Model Training Process

1. **Data Preparation**: Normalize prices to 0-1 range
2. **Sequence Creation**: Create sliding windows of historical data
3. **Training**: Learn patterns from past to predict future
4. **Validation**: Test on unseen data

5. **Forecasting**: Iteratively predict future values

Limitations

⚠️ **Important Disclaimers:**

- Stock markets are influenced by many unpredictable factors
- Past performance doesn't guarantee future results
- This tool is for educational purposes only
- Always do thorough research before making investment decisions
- Consider consulting financial advisors for investment advice

🎓 Educational Use Cases

- **Machine Learning Education**: Learn LSTM implementation
- **Financial Analysis**: Understand price patterns
- **Portfolio Management**: Analyze multiple stocks
- **Algorithm Trading Research**: Develop trading strategies
- **Data Science Projects**: Time series forecasting practice

🔮 Future Enhancements

Potential improvements:

- [ ] Multiple feature inputs (volume, indicators, sentiment)
- [ ] Attention mechanisms
- [ ] Ensemble models
- [ ] Real-time streaming predictions
- [ ] Portfolio optimization tools
- [ ] Technical indicators integration
- [ ] News sentiment analysis
- [ ] Backtesting framework

📄 License

This project is provided as-is for educational purposes. Feel free to modify and extend for your needs.

🤝 Contributing

Contributions welcome! Feel free to:

- Report bugs
- Suggest features
- Submit pull requests
- Improve documentation

📞 Support

For issues or questions:

1. Check the troubleshooting section
2. Review the code comments
3. Consult TensorFlow/Dash documentation

🌟 Acknowledgments

- **TensorFlow**: Deep learning framework
- **Yahoo Finance**: Free financial data API
- **Dash by Plotly**: Interactive visualization
- **Scikit-learn**: Machine learning utilities

Happy Forecasting! 📊 🚀

Remember: This tool is for educational and research purposes. Always conduct thorough analysis and consult professionals before making investment decisions.