TPOT AutoML Flask Application Documentation

1 Introduction

This document provides detailed documentation for the TPOT AutoML Flask Application. This Flask application offers a web interface for automatically training machine learning models using the TPOT library on user-uploaded CSV datasets.

2 Features

- **File Upload:** Users can upload CSV files to be used as datasets for model training.
- **Feature Specification:** Users can specify which columns to use as features and target in the dataset.
- **Model Training:** The application trains a model using TPOT, an automated machine learning tool.
- **Model Evaluation:** It displays the accuracy of the trained model.
- **Model Download:** Users can download the trained model for future use.

3 Requirements

The application requires the following Python libraries:

- Flask
- Pandas
- NumPy
- scikit-learn
- TPOT
- Werkzeug

4 Setup and Installation

4.1 Environment Setup

It is recommended to use a virtual environment. Create and activate it as follows:

```
python -m venv venv
source venv/bin/activate
```

4.2 Install Dependencies

Install the required Python libraries using pip:

```
pip install flask pandas numpy scikit-learn tpot werkzeug
```

4.3 Running the Application

Run the application using Flask:

```
export FLASK_APP=app.py
flask run
```

Access the web interface at `http://127.0.0.1:5000/`.

5 Application Structure

5.1 app.py

- **Initialization:** Sets up the Flask application and configuration.
- **Routes:**
  - `/`: Main page for file upload.
  - `/upload`: Handles the uploading of CSV files.
  - `/train`: Processes the uploaded file and trains a model using TPOT.
  - `/results`: Displays the results, including model accuracy.
  - `/download_model`: Allows downloading of the trained model.

5.2 Utility Functions

- `allowed_file(filename)`: Checks if the uploaded file is in an allowed format (CSV).
- `preprocess_data(X_train, X_test)`: Preprocesses the training and test data, applying feature scaling and encoding.

6 Security and Limitations

- The application uses Flask's built-in server, which is not suitable for production.
- File uploads should be handled carefully to avoid security risks. The application currently only allows CSV files.
- The secret key for Flask should be set to a secure, random value in production.

7 Further Development

- Enhance security for file uploads and data handling.
- Implement progress tracking for model training.
- Consider deploying with a production-ready server like Gunicorn.