AutoML Streamlit App - Project Documentation

🔍 AutoML Streamlit App

An interactive Streamlit application for automated machine learning (AutoML). Upload your dataset, run EDA, handle class imbalance, select classification or regression tasks, and evaluate models — all in a few clicks.

🚀 Features

- Upload CSV dataset
- Automatic task detection (classification vs regression)
- Interactive EDA using: Sweetviz and YData Profiling
- Preprocessing pipeline (missing values, encoding, scaling)
- Imbalanced dataset handling (SMOTE, undersampling)
- Auto model selection & training
- Performance evaluation (accuracy, precision, recall, F1, confusion matrix)

AutoML Streamlit code.

- **Imports**
- These load the necessary Python packages:
  - **Streamlit**: for web UI
  - **Pandas, Numpy**: data handling
  - **Matplotlib, Seaborn**: data visualization
  - **Sweetviz, ydata_profiling**: for automated EDA
  - **Scikit-learn**: for preprocessing, ML algorithms, and evaluation
  - **Statsmodels**: for statistical regression (OLS)
  - **Imbalanced-learn (SMOTE, RandomUnderSampler)**: to handle imbalanced data
  - **XGBoost**: for gradient boosting classification/regression

---

- **data_collection()**
  - Lets user upload a file in the Streamlit app.
  - Reads it into a DataFrame (df) and shows the first 10 rows.

- **Feature_selection(df)**
  - User selects input features (X) and target variable (y) using a multiselect and dropdown.

- **data_understanding(df)**
  - Displays data shape, types, missing values, and basic stats.
  - Creates:
    - Pairplot
    - Boxplot
    - Heatmap of correlations
  - Generates a **ydata-profiling report** (HTML rendered inside Streamlit).

- **data_preprocessing(df)**
  - **Missing Value Imputation**:
    - Uses different strategies based on column type (mean, mode, or interpolate).
  - **Outlier Detection & Treatment**:
    - Uses IQR method (excluding binary data).
    - Outliers replaced with the median.

- **data_preparation(df)**
  - **Label encoding** for categorical columns using pd.factorize.
  - **Datetime parsing** (only if all columns are datetime).
  - **Normality check (D'Agostino test)**:
    - If non-Gaussian → MinMaxScaler
    - If Gaussian → StandardScaler

- **check_class_balance(Target)**
  - Checks for imbalance using ratio of majority to minority class count.

- **perform_classification(...)**
  - Trains 6 classifiers.
  - Evaluates using Accuracy, Precision, Recall, F1-Score.
  - Displays metrics in a table.
  - Identifies and stores the **best model** in st.session_state.

- **perform_regression(...)**
  - Trains 9 regressors + Polynomial + OLS (if 1 feature).
  - Evaluates with **R^2** and **RMSE** using cross-validation.
  - Returns results and best model based on R^2.

- **main()**
  - Coordinates everything:
    - Uploads and processes data
    - Feature/target selection
    - Data EDA, preprocessing, and scaling
    - Detects task type:
      - **Classification**: if target is categorical or 0/1
        - Uses SMOTE or undersampling if imbalanced
        - Trains, evaluates, and predicts with best model
      - **Regression**: if target is continuous
        - Trains, evaluates, and predicts using best regressor

---

Tech Stack

- Python
- Streamlit
- Scikit-learn
- Imbalanced-learn
- Sweetviz
- YData Profiling

Contributing

Feel free to open issues or pull requests.

License

MIT License