One Sentence to Uncover the Past & Predict the Future - User Guide

Project Background

This tool was developed to bridge the gap between marketing teams and data insights by eliminating technical barriers. It transforms natural language questions into SQL queries, visualizations, and predictions, enabling marketers to make data-driven decisions without relying on data science teams.

The project combines state-of-the-art NLP with advanced analytics to create a self-service platform that delivers both historical analysis and predictive capabilities.

System Requirements

- **Python Version**: 3.8 or higher
- **Core Dependencies**:
  - PySpark 3.0+
  - LightGBM
  - Flask
  - Pandas
  - NumPy
  - Matplotlib/Seaborn
  - OpenAI API (for NLP capabilities)
- **Hardware Requirements**:
  - Minimum 8GB RAM recommended
  - 2GB free disk space
- **Supported Operating Systems**:
  - Windows 10/11
  - macOS 10.15+
  - Ubuntu 20.04+ or other Linux distributions

Installation and Setup

Step 1: Clone the Repository

```
git clone https://github.com/bbehaz/BDA-Trends
cd ctr-predictor-module
```

Step 2: Set Up a Virtual Environment

For Windows:

```
python -m venv venv
```

```
venv\Scripts\activate
```

For macOS/Linux:

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

Step 3: Install Dependencies

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

Step 4: Configure OpenAI API (for NLP Features)

In the same `config.py` file, set your OpenAI API key:

```
# OpenAI API Configuration
api_key = 'your_openai_api_key'
OPENAI_MODEL = 'gpt-4-mini'  # Options: gpt-3.5-turbo, gpt-4
```

Step 5: Model Configuration

Choose between using the pre-trained model or training your own:

Option A: Use Pre-trained Model

Place the provided `model.txt` file in the `ctr-predictor-module-main/model/` directory.

Option B: Train a New Model

See the "Training a New Model" section below for detailed instructions.

Usage Instructions

Method 1: Web Interface

After running the jupyter notebook:

1. Open your web browser and navigate to `http://localhost:5000`
2. You'll see the main dashboard with a query input box
3. Enter your natural language query (e.g., "How does CTR vary by day of the week?")
4. Click "Analyze" to process your query
5. View the results in the form of visualizations and data tables

Web Interface Features:

- Input natural language queries
- Select from visualization types (bar charts, line graphs, heatmaps, etc.)
- Export results as CSV, Excel, or PNG files
- Save queries for future use
- View query history
- Set up scheduled reports

Training a New Model

How To Run the Pipeline

1. **Install requirements** (if not already):

pip install pandas lightgbm duckdb scikit-learn

2. **Run the test query pipeline**:

python test_main_pipeline.py

This runs the full end-to-end pipeline:
SQL → Feature creation → CTR prediction

How To Customize

- To try a different SQL query, open `test_main_pipeline.py` and modify the SQL string.
- To retrain the model using the full Avazu dataset:
  - Unzip the full `train.csv`

Run the following command:
python train_avazu_model.py

- This will update:
  - `model/ctr_model.txt`
  - `utils/ctr_encoding_map.json`

How It Works Under the Hood

- **`feature_creator.py`**:
  Adds time features, CTR encodings, hashed interactions. Handles missing fields using fallback defaults.
- **`predict_ctr.py`**:
  Loads the trained LightGBM model and makes predictions using selected numeric features.
- **`main_pipeline.py`**:
  Takes a SQL query string, runs it on the Avazu subset, and returns predicted CTRs for matching rows.
- **`test_main_pipeline.py`**:
  Test script that simulates an NLP-generated SQL input.

Integration Notes

- You can call `run_sql_to_ctr_predictions(sql_query)` from any upstream component (e.g., a UI, an NLP handler, etc.).
- The module is easy to plug into a dashboard or log predictions into DuckDB.
- Designed to be **modular**, **robust**, and compatible with an **NLP** → **SQL** → **Prediction** pipeline.

System Features

Natural Language Query Capabilities

The system can process various types of natural language queries:

Time-Based Analysis

- "How does CTR vary by hour of day?"
- "Show me CTR trends over the past month"
- "Compare weekday vs weekend CTR"

Device Comparison

- "Compare CTR between Android and iOS devices"
- "Which device type has the highest CTR?"
- "Show CTR by device model"

Placement Analysis

- "Which ad placements perform best?"
- "Compare top vs bottom banner performance"
- "Show CTR heatmap by page position"

Multi-Dimensional Analysis

- "Analyze mobile CTR by hour of day"
- "Compare weekend performance across device types"
- "Show CTR by country and device type"

Predictive Analysis

- "Predict CTR for next week"
- "Forecast campaign performance for the upcoming weekend"
- "Which day next week will likely have the highest CTR?"

Visualization Capabilities

The system automatically generates appropriate visualizations:

1. **Time Series Plots**
2. **Comparative Charts**
3. **Distribution Plots**