Phase 2 Report: Data Preparation for Predicting Advertisement Demand on Avito Platform

Artur Zagitov, Ildar Zalialiev, Artem Nazarov

Contents

1 Chapter 3: Data Preparation **2**
1.1 Select Data . . . 2
1.1.1 Data Selection Rationale . . . 2
1.1.2 Data Inclusion . . . 2
1.2 Clean Data . . . 2
1.2.1 Handling Missing Values . . . 2
1.2.2 Category Collapsing . . . 2
1.3 Construct Data . . . 2
1.3.1 Derived Attributes . . . 2
1.3.2 Text Feature Construction . . . 3
1.3.3 Time Feature Extraction . . . 3
1.4 Standardize Data . . . 3
1.4.1 Normalization and Encoding . . . 3
1.5 Automated Workflows and Pipelines . . . 3
1.5.1 Apache Airflow Data Extraction Workflow . . . 3
1.5.2 ZenML Pipeline for Data Preparation . . . 4

1 Chapter 3: Data Preparation

Data preparation is a crucial step in any machine learning project as it involves constructing the final dataset from the raw data. This phase includes various tasks such as data selection, cleaning, construction, and standardization to ensure that the data fed into the machine learning pipelines is optimized for better modeling performance.

1.1 Select Data

1.1.1 Data Selection Rationale

Given the project's constraints and objectives, we decided to use a subset of 10,000 rows sampled from the initial dataset. This decision was based on the balance between computational efficiency and maintaining enough data variety for model training.

Columns Excluded

- **'image':** Excluded due to the lack of resources and time to process image data effectively.
- **'item_id':** Removed as it is a unique identifier with no predictive power regarding ad demand.
- **'user_id':** Omitted because there are not other specific user information that could be used to make this feature relevant to predicting the demand.

1.1.2 Data Inclusion

The columns retained are the ones that are essential for predicting advertisement demand, including ad descriptions, prices, categorical data about the ad's category, and user type and price.

1.2 Clean Data

1.2.1 Handling Missing Values

Missing data was addressed using following strategies to improve the dataset quality:

- **'param_1', 'param_2', 'param_3':** Imputed with 'missing' to treat missing values as a separate category.
- **'description':** Missing descriptions were filled with placeholder text 'No description.' to maintain uniformity.
- **'price':** Missing prices were imputed with the median price of the respective category to reflect typical values.
- **'image_top_1':** Missing entries were assigned a new category by using the maximum value+1.

1.2.2 Category Collapsing

For columns with numerous unique categories, we reduced complexity by retaining the top 10 categories and grouping all others into an 'Other' category. This was done for:

- **'category_name', 'city', 'param_1', 'param_2', 'param_3':** This approach was chosen to reduce the dimensionality and avoid sparse data issues post one-hot encoding. We ensured that top 10 categories encompassed more then 50% of all entries in the dataset.

1.3 Construct Data

1.3.1 Derived Attributes

New attributes created include:

- **'description_length', 'title_length', 'param_length':** These features were engineered based on exploratory data analysis insights to capture the potential impact of content length on ad demand.

1.3.2 Text Feature Construction

To handle the text data efficiently:

- **Text Vectorization:** ‘description‘ and ‘title‘ were transformed using TfIdf vectorization with 128 features for each of two columns.
- **Dimensionality Reduction:** The number of features from the TfIdf vectorization was reduced using PCA, where ‘description‘ was brought down to 16 features and ‘title‘ to 8 features. This was essential to capture the most relevant information while keeping the number of dimensions manageable.

1.3.3 Time Feature Extraction

Given the dataset's time-related attributes, we extracted and encoded the following features:

- **‘day_of_week‘ and ‘day_of_month‘:** From the ‘activation_date‘, we extracted the day of the week and the day of the month to capture temporal trends and cycles in advertisement activity and demand. We choose only these features, as our EDA showed, that all items in the dataset were posted on the same year and within only two different months.
- **Sine-Cosine Encoding:** We applied sine-cosine transformations to these features to maintain their cyclical nature.

1.4 Standardize Data

1.4.1 Normalization and Encoding

All categorical features were one-hot encoded to transform them into a format suitable for the model. Numerical features were scaled using Standard Scaler to normalize their distribution.

Uniform Data Schema

To ensure consistency across the data preparation and modeling phases, we:

- Trained and saved all encoders and scalers on a separate dataset of 100,000 non-overlapping samples with train and test sets to prevent data leakage.
- Used these trained components for subsequent data transformations.
- Ensured all features are in a specific order by sorting columns alphabetically, which is important, since input layer of the neural network expects features to be in the same order every time.

1.5 Automated Workflows and Pipelines

1.5.1 Apache Airflow Data Extraction Workflow

An automated workflow, implemented in Apache Airflow (‘data_extract_dag.py‘), performs four tasks for data extraction and versioning.

1. **Data Sampling:** Extracts a new data sample.
2. **Data Validation:** Validates the sample using Great Expectations to ensure data quality.
3. **Data Versioning:** Versions the data sample using DVC and logs the version in ‘./configs/datasets.yaml‘.
4. **Data Loading:** Loads the sample to the DVC-managed data store.

1.5.2 ZenML Pipeline for Data Preparation

The ETL pipeline (‘transform_data.py‘) utilizes ZenML for executing four data processing tasks. This setup ensures that the data transformations and validations are consistently applied, and the resulting features are properly versioned and stored.

1. **Data Extraction:** Data is extracted from the DVC store based on the version provided.
2. **Data Transformation:** Applies preprocessing to transform the data sample into features suitable for machine learning models, using pipeline specified above.
3. **Feature Validation:** New expectations are created and validated using Great Expectations to ensure the quality of the features.
4. **Feature Loading and Versioning:** Features are loaded into the ZenML artifact store with versioning.

A second Apache Airflow DAG (‘data_prepare_dag.py‘) manages the integration of the data extraction and preparation pipelines.