Foundations of Machine Learning

Hackathon Report

G Sai Keerthi - CS22BTECH11024

Firstly about libraries:

In our code, we used **Pandas** for data manipulation, **NumPy** for numerical operations. **Scikit-learn** provided preprocessing tools and evaluation metrics. **XGBoost** was the main library for training, chosen for its high performance.

Our code is structured in 3 sections which are feature classification, data preprocessing and model training.

1. **Classifying features:** In this section, we categorize the dataset's features. Our approach involves the 'categorize_features' function which processes the data and classifies features into 4 primary types: numeric, categorical, binary, and features that need to be dropped.
    - **Numeric features**: These are discrete or continuous discrete columns. These columns are very important for our data and help in determining appropriate scaling and imputation strategies.
    - **Categorical features**: Columns representing types or classes are treated as categorical features. We have specified some columns as categorical features manually in the code which are given in the last lines of description. These include 'TypeOfIrrigationSystem','CropFieldConfiguration', 'FarmClassification', 'HarvestProcessingType', 'LandUsageType', 'FieldZoneLevel', 'FieldConstructionType'.
    - **Binary features**: If any feature has 2 or less than 2 values being represented then they are classified as binary features and we are appending them to binary_features column.
    - **Features to drop**: Some columns, such as UID and RawLocationId, may not contribute meaningful information for training. Such columns are stored and dropped later.

This initial classification provides a better,comprehensive understanding of the dataset,which enables the implementation of specific preprocessing procedures.

2. **Data Preprocessing:** This is an essential step in maintaining data consistency. Preprocessing calculates missing values and performs other duties such as imputation, feature scaling, and encoding.
    - **Handling missing values**: For each feature, we are checking the missing percentage of values. If there are missing values more than 90 percent, then we are dropping the feature columns.
    - **Imputation**: Numeric features with missing values are imputed using the median of the column because the median is less sensitive to outliers compared to the mean. Categorical features having missing values are filled with the mode as this helps in maintaining the integrity of data.
    - **Feature scaling**: Numeric columns are scaled and standardized to have a mean of 0 and a standard deviation of 1, which ensures that large-scale features do not impact the training process.
    - **Encoding**: Categorical features are one-hot encoded which converts each category into a separate binary column.Binary features are also encoded to maintain consistency throughout the data. This further helps for algorithms that cannot handle categorical instances.

3. **Training a model**: For the given dataset we can use models like Neural Networks, Decision tree, Random Forest, SVMs with Kernel methods, and Ensemble methods. But we chose XGBoost as the core algorithm due to its strong performance over the above mentioned models.

    What we did ?
    - After preprocessing, we have initialized XGBoost with a set of carefully chosen hyperparameters which can fit for our dataset. We have set n_estimators to be 500, learning_rate to 0.1, max_depth to be 8 to avoid overfitting. Lower learning rate with a high number of trees gives us better performance.
    - We have also trained and observed many models like Decision tree, random forest with varying hyper parameters, Neural Network and XGBoost (Ensemble method).
    - But as we observed, XGBoost gave us the highest score ,which seemed to have generalized the dataset the most among all the above models that we have trained.
    - Finally,we have decided on training a XGBoost classifier as it is a combination of many models and has the scope of generalizing the data better on real-world data where there are many outliers in general.

Observations:

Dropping columns with missing values resulted in significant speed benefits. The columns were removed to improve accuracy to 90%, up from 70% when they were present.

Learning rate and n_estimators have impact on model performance. Low learning_rates like 0.01, 0.05 having high estimators around 500 gave good results for our preprocessed data, compared to estimators with 100.