1. Data Upload and Summary

- **File Upload**: Users can upload a dataset in CSV format.
- **Dataset Summary**: After uploading, the application displays a quick summary of the dataset, including its shape (number of rows and columns), descriptive statistics (mean, median, etc.), and information about missing values.
- **Purpose**: This step helps users understand their data and its quality before diving into any machine learning tasks.

2. Data Visualization

- **Correlation Heatmap**: Visualizes correlations between numerical columns. Correlation helps identify relationships between variables, which can inform feature selection.
- **Value Counts**: Displays frequency counts for categorical variables to identify imbalances in the data.
- **Scatter Plots**: Useful for understanding the relationship between two numerical columns.

---

3. Data Cleaning

- **Options Provided**:
  - Drop rows/columns with missing values.
  - Remove duplicate rows.
  - Drop specific columns as selected by the user.
- **Purpose**: Ensuring data quality by removing noise and irrelevant information, which improves model performance.

---

4. Data Preprocessing

This section includes techniques that prepare the data for training machine learning models.

a. Handling Class Imbalance

In classification problems, class imbalance occurs when one class significantly outnumbers the other(s). This can lead to biased models. The application provides three options:

- **Over Sampling**: Uses RandomOverSampler to duplicate minority class samples to balance the classes.
- **Under Sampling**: Uses RandomUnderSampler to randomly remove samples from the majority class to balance the classes.
- **Combined Sampling**: Uses SMOTEENN (Synthetic Minority Over-sampling Technique + Edited Nearest Neighbors) to both oversample the minority class and clean up the oversampled dataset.

b. Principal Component Analysis (PCA)

- **PCA** is a dimensionality reduction technique that transforms the features into a smaller set of uncorrelated components, capturing as much variance as possible.
- **Purpose**: Reduces the complexity of the dataset, speeding up training and reducing overfitting.

---

5. Model Training

This is the core part of the application where users can train machine learning models.

a. Classification vs. Regression vs. Clustering

- **Classification**: Predicts discrete labels (e.g., spam vs. not spam).
- **Regression**: Predicts continuous values (e.g., house prices).
- **Clustering**: Groups data into clusters based on similarity (e.g., customer segmentation).

b. Algorithms Available

- **Logistic/Linear Regression**:
    - **Logistic Regression** is used for binary or multiclass classification.
    - **Linear Regression** is used for predicting continuous numerical values.
- **Random Forest**:
    - An **ensemble learning** method using multiple decision trees for classification and regression.
    - Robust against overfitting and handles missing data well.
- **XGBoost**:
    - An optimized **gradient boosting** algorithm that is highly efficient and often used in competitions.
- **CatBoost**:
    - A gradient boosting method optimized for categorical features.
    - Often outperforms other boosting algorithms when dealing with categorical data.
- **Support Vector Machines (SVM)**:
    - **SVC** (Support Vector Classifier) for classification.
    - **SVR** (Support Vector Regressor) for regression.
    - Effective for high-dimensional data and non-linear problems.
- **K-Nearest Neighbors (KNN)**:
    - A simple algorithm that classifies based on the majority class of the **k-nearest neighbors**.
    - Works well for smaller datasets but can be slow for large datasets.

- **Clustering**:
  - **K-Means** groups data into a predefined number of clusters.
  - **Hierarchical Clustering** groups data into a hierarchy (tree-like structure) without specifying the number of clusters beforehand.

c. Hyperparameters

Users can customize hyperparameters for each algorithm, such as:

- Maximum iterations for logistic/linear regression.
- Number of estimators for Random Forest, XGBoost, and CatBoost.
- Kernel type and regularization (C) for SVM.
- Number of neighbors for KNN.
- Number of clusters and linkage type for clustering algorithms.

d. Training:

After configuring the models and their hyperparameters, the selected models are trained using the uploaded dataset. The dataset is split into training and test sets using train_test_split.

---

6. Model Evaluation

Once models are trained, they are evaluated based on the problem type:

a. For Classification Models:

- **Accuracy**: Percentage of correct predictions.
- **Precision**: Correct positive predictions out of total predicted positives.
- **Recall (Sensitivity)**: Correct positive predictions out of actual positives.
- **F1 Score**: Harmonic mean of precision and recall.
- **Confusion Matrix**: Shows True Positives, True Negatives, False Positives, and False Negatives.

b. For Regression Models:

- **Mean Absolute Error (MAE)**: Average of absolute differences between predicted and actual values.
- **Root Mean Squared Error (RMSE)**: Square root of the average squared differences between predicted and actual values.
- **R2 Score**: Measures the proportion of variance explained by the model.

c. For Clustering Models:

- **Silhouette Score**: Measures how similar a sample is to its own cluster compared to other clusters.

---

7. Model Download

- Users can download trained models using the **pickle** library.
- This allows users to save their models for future use or deployment in production systems.