Predict Backpack Prices

Juan Esteban Rodriguez Camacho
Edgar Julian Roldan Rojas
Daniel Esteban Camacho Ospina

Problem Context

Goal: Predict Prices Given Various Attributes

Evaluation: RMSE

Regression Model to estimate price of backpacks from dataset

Identify interactions, sensitivity and chaos

Quality guidelines: Scalability and mantainability

System Properties

1

Homeostasis

In the relationship between the model and market changes that are not in the dataset.

2

Adaptability

In the way the system can be updated to handle new market conditions.

3

Resilience

In the robustness of the system against defective data, outliers or noise.

4

Emergency

In complex relationships between simple variables such as brand, material and size.

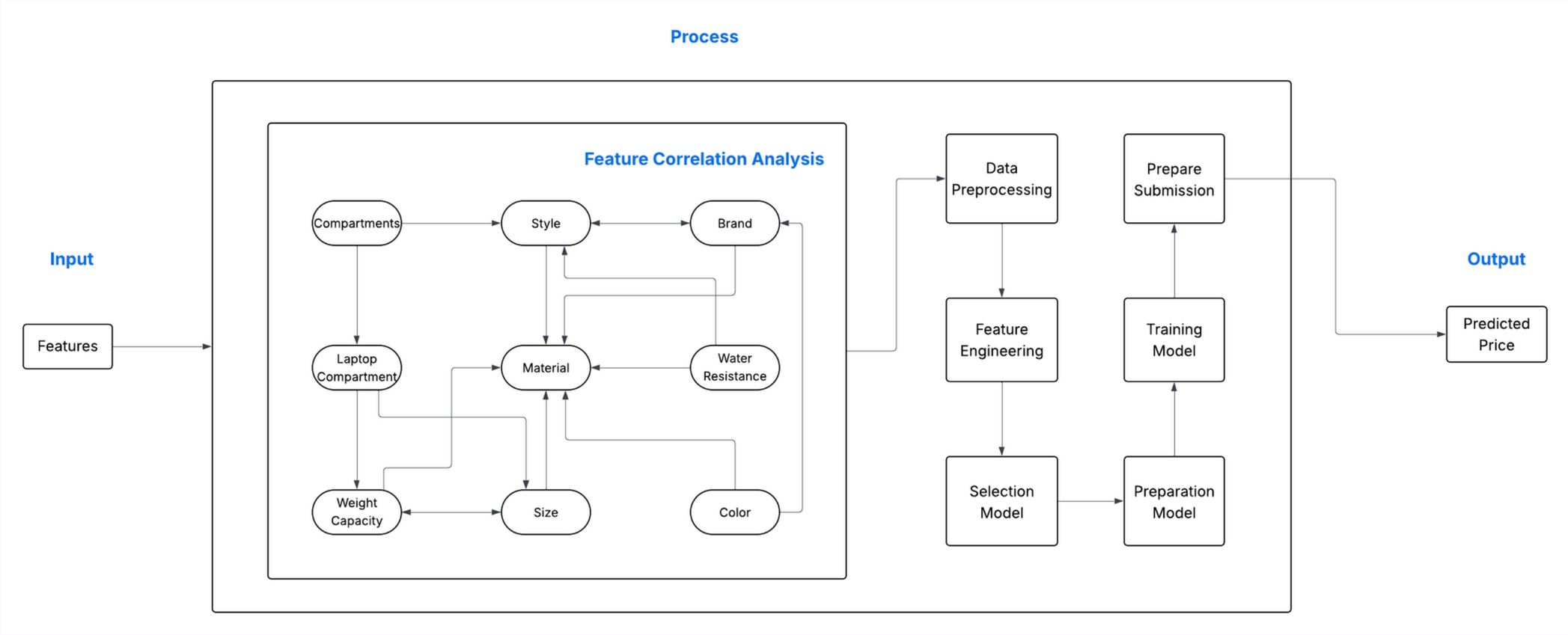
Sensitivity Analysis

1. **Changes to a feature generate changes in the output.**

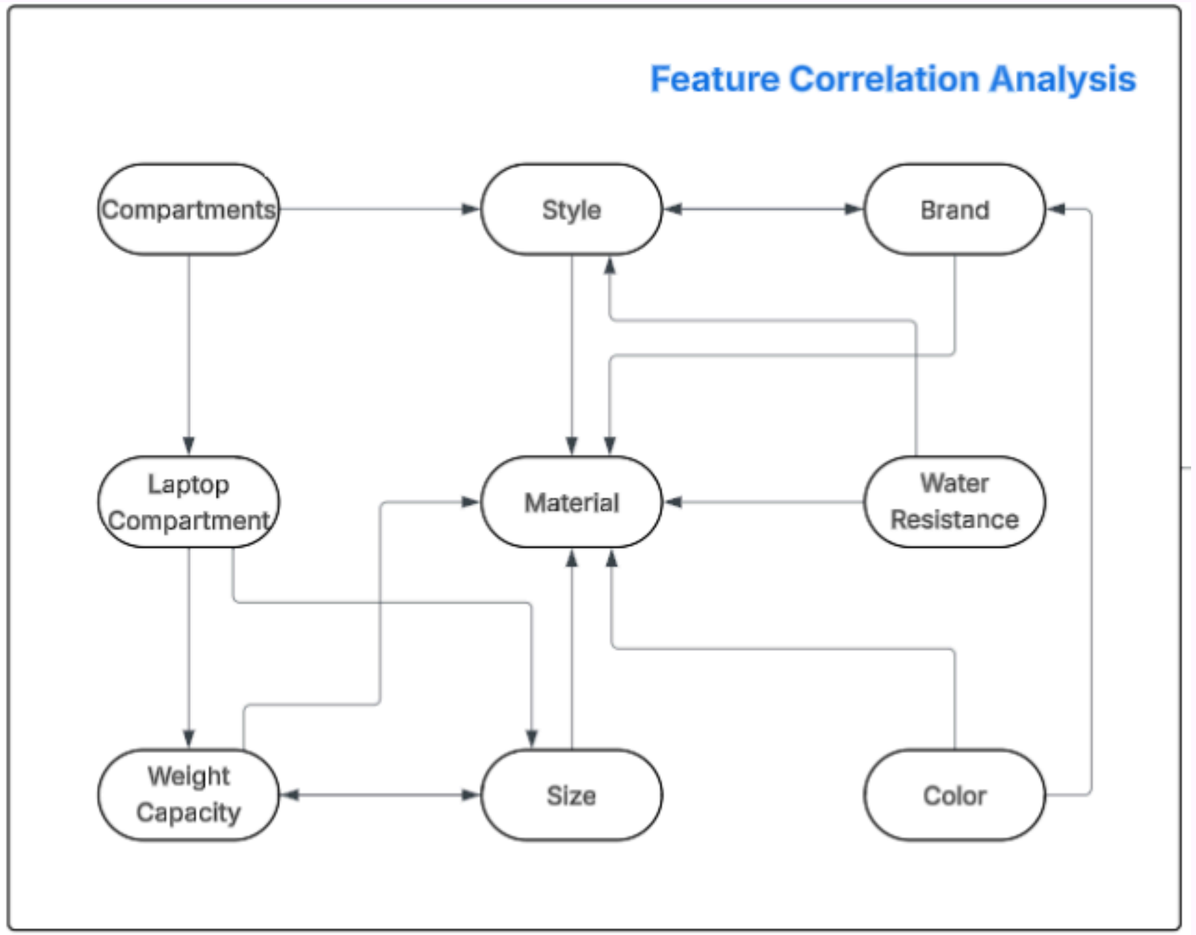
2. **Hyperparameter adjustment**

3. **Comparison between models**

Proposed Solution



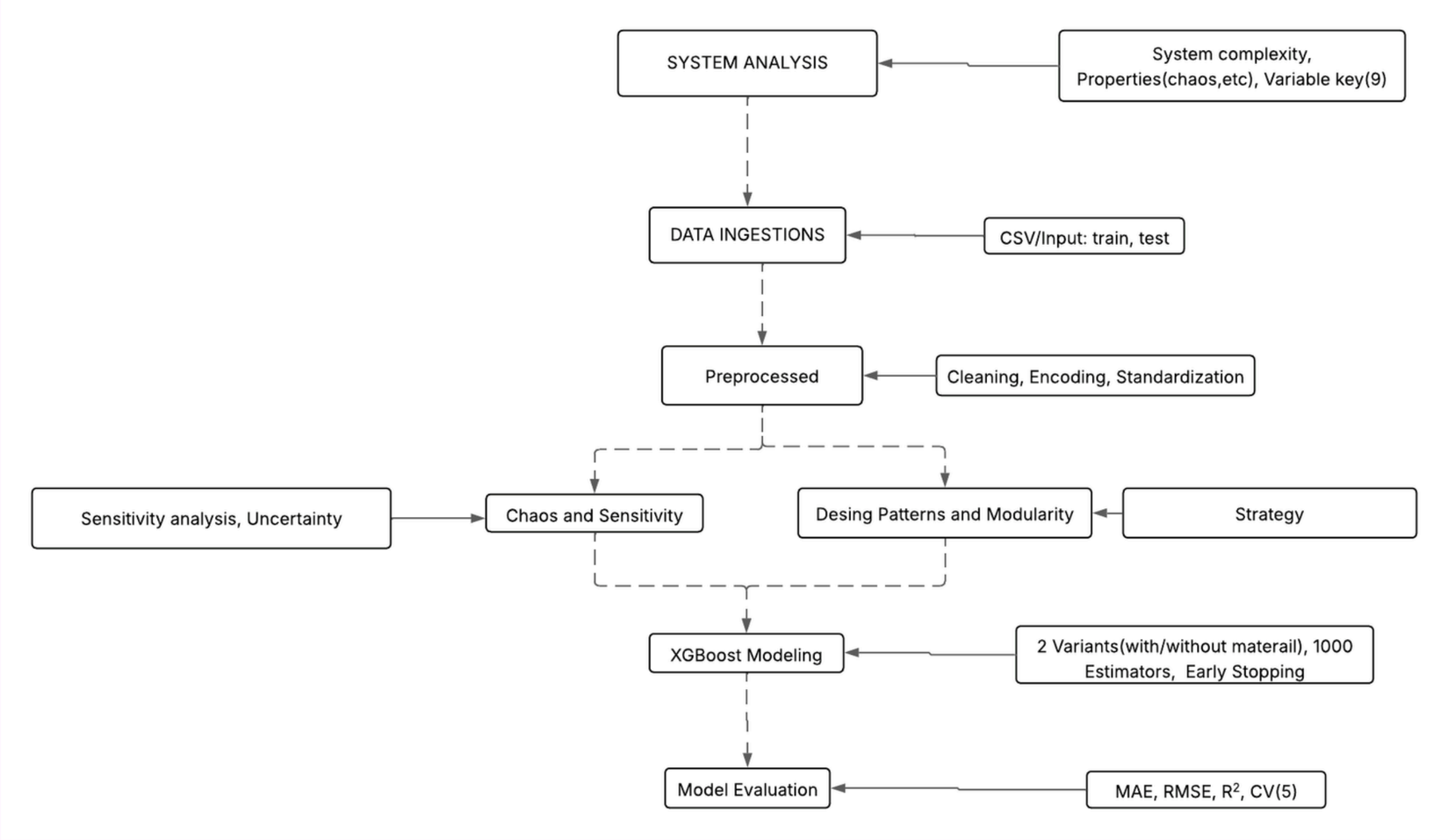
Feature Analysis



Material feature exhibits strong correlation

Dependency pattern in the model

High Level Architecture

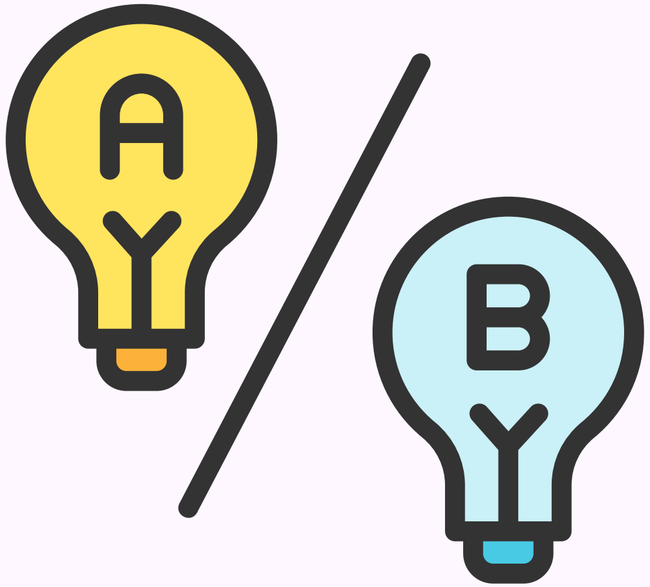


Solution Analysis

1. **Adopt XGBoost**

2. **Implements a dual-model architecture**

3. **Integrates a robust preprocessing pipeline**



Dataset Preparation

```python
def preprocess_data(df, test):
    # Handle missing values and encode categorical variables
    test['Size'] = test['Size'].fillna(test['Size'].mode()[0])
    df.dropna(inplace=True)
    size_order = ['Small', 'Medium', 'Large']
    encoder = OrdinalEncoder(categories=[size_order])
    df['Size'] = encoder.fit_transform(df[['Size']])
    test['Size'] = encoder.transform(test[['Size']])
    categorical_cols = ['Brand', 'Laptop Compartment', 'Waterproof',
     'Style', 'Color']
    df = pd.get_dummies(df, columns=categorical_cols)
    test = pd.get_dummies(test, columns=categorical_cols)
    return df, test
```

Dual Model Implementation

```python
def prepare_data(self, df, test):
    # Prepare data including the 'Material' column
    df = pd.get_dummies(df, columns=['Material'])
    test = pd.get_dummies(test, columns=['Material'])
    X_train = df.drop(['id', 'Price'], axis=1)
    Y_train = df['Price']
    X_valid = df.drop(['id', 'Price'], axis=1)
    Y_valid = df['Price']
    X_test = test.drop(['id'], axis=1)
    return X_train, Y_train, X_valid, Y_valid, X_test
```

```python
def prepare_data(self, df, test):
    # Prepare data excluding the 'Material' column
    X_train = df.drop(['id', 'Price', 'Material'], axis=1)
    Y_train = df['Price']
    X_valid = df.drop(['id', 'Price', 'Material'], axis=1)
    Y_valid = df['Price']
    X_test = test.drop(['id', 'Material'], axis=1)
    return X_train, Y_train, X_valid, Y_valid, X_test
```

Results

submission.csv Complete (after deadline) · 1h ago	**39.02916**	**39.23195**	☐
submission.csv Complete (after deadline) · 2mo ago	**39.02674**	**39.23126**	☐

Thank You