Predicting Building Damage from Earthquakes

Certainly, here's a high-level overview of loading and preprocessing a dataset for an earthquake prediction model:

1. **Data Collection**
    - Gather earthquake-related data from reliable sources, such as seismometers, geological surveys, or earthquake databases.

2. **Data Cleaning**

- Remove duplicates and irrelevant data.

- Handle missing values through imputation or removal.

- Check for outliers and anomalies in the data.

3. **Feature Selection/Engineering**

- Identify relevant features such as seismic activity, location, depth, and time.

- Create additional features if necessary, like earthquake magnitude bins or time-based features.

4. **Data Transformation**

- Normalize or scale numerical features to ensure they have similar scales.

- Encode categorical variables using techniques like one-hot encoding or label encoding.

5. **Temporal Data Handling:**

- If working with time-series data, ensure the correct handling of time-related features.

- Consider techniques like rolling statistics or windowed data for time-based analysis.

6. **Data Splitting**:

- Split the dataset into training, validation, and test sets. The typical split might be 70% for training, 15% for validation, and 15% for testing.

7. **Data Augmentation (optional):**

- To enhance the dataset, you can use techniques like data resampling, time series augmentation, or adding synthetic data.

8. **Handling Class Imbalance**

(if applicable):

- In earthquake prediction, you might encounter class imbalance (few earthquakes compared to non-earthquake instances). Use techniques like oversampling, undersampling, or synthetic data generation to address this.

9. **Data Pipeline**:

- Create a data preprocessing pipeline to automate these steps and ensure consistency.

10. **Target Variable Definition**:

- Define the target variable, which could be binary (earthquake or no earthquake) or a regression task predicting earthquake magnitude.

11. **Data Normalization**:

- Consider using techniques like Z-score normalization for numerical features to center them around a mean of 0 and a standard deviation of 1.

12. **Model-Specific Preprocessing**

- Tailor the preprocessing to the machine learning model you plan to use. For example, convolutional neural networks (CNNs) for image data may require specific preprocessing steps.

The specific steps and techniques you employ will depend on the nature of your dataset, the problem you're trying to solve, and the machine learning algorithms you plan to use. Consistent and well-thought-out data preprocessing is crucial for the success of your earthquake prediction model.

Simplified example program

```
# Import necessary libraries
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

# Step 1: Load the dataset
data = pd.read_csv("earthquake_data.csv")

# Step 2: Data Cleaning (if needed)
# Remove missing values or handle outliers here

# Step 3: Feature Selection (assuming 'magnitude', 'depth', and 'latitude' as features)
features = data[['magnitude', 'depth', 'latitude']]

# Step 4: Data Splitting
X = features
y = data['earthquake_label']  # Assuming 'earthquake_label' is the target variable
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 5: Feature Scaling
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Step 6: Model Training
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

Step 7: Model Evaluation

y_pred = model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)

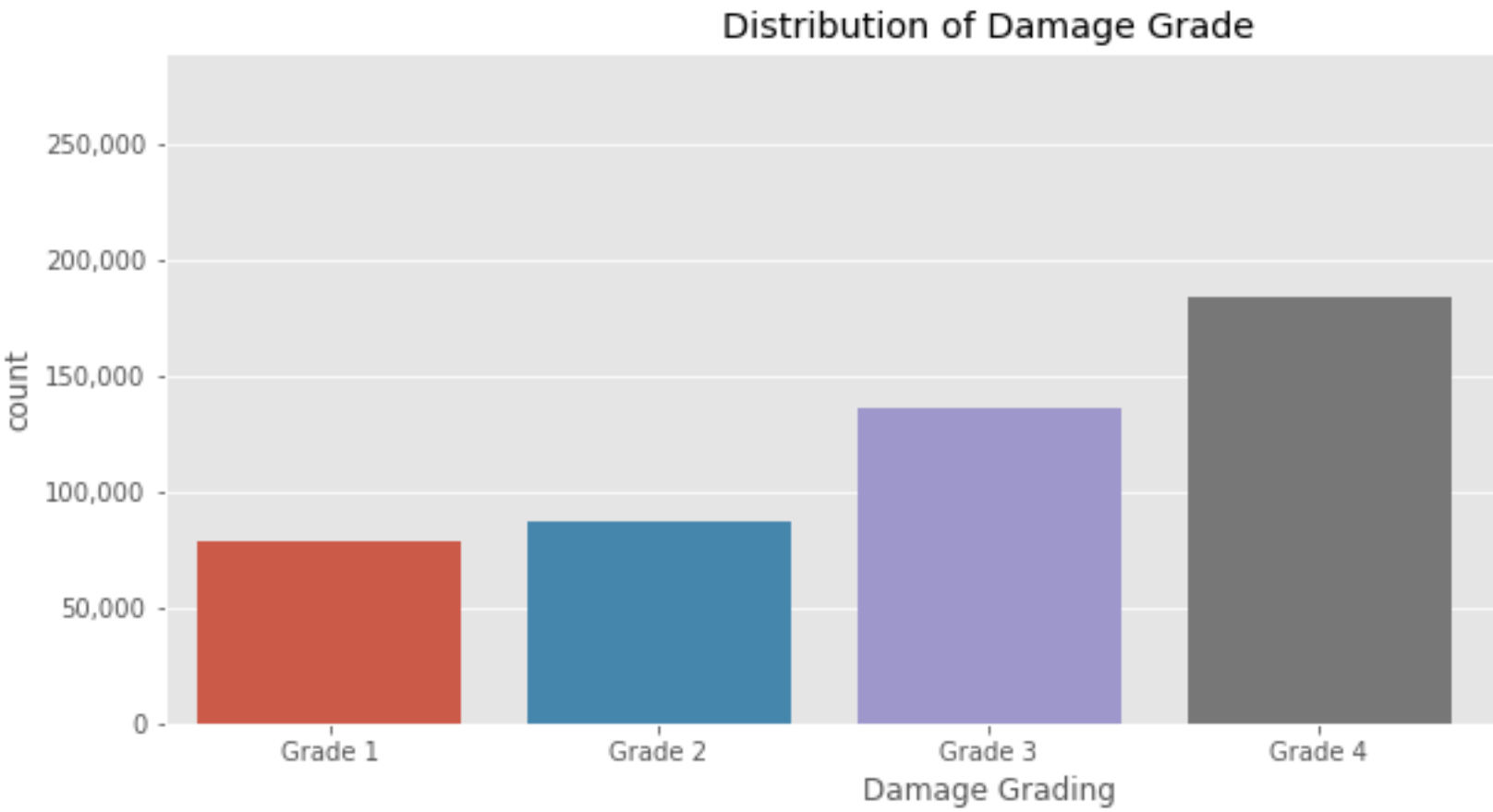
print(f"Accuracy: {accuracy}")

You can further fine-tune hyperparameters and perform more comprehensive preprocessing based on your dataset and requirements.

```
# Convert data types to categorical
df_stru = df_stru.astype({'district_id': 'object', 'vdcmun_id': 'object',
'ward_id': 'object'})
```

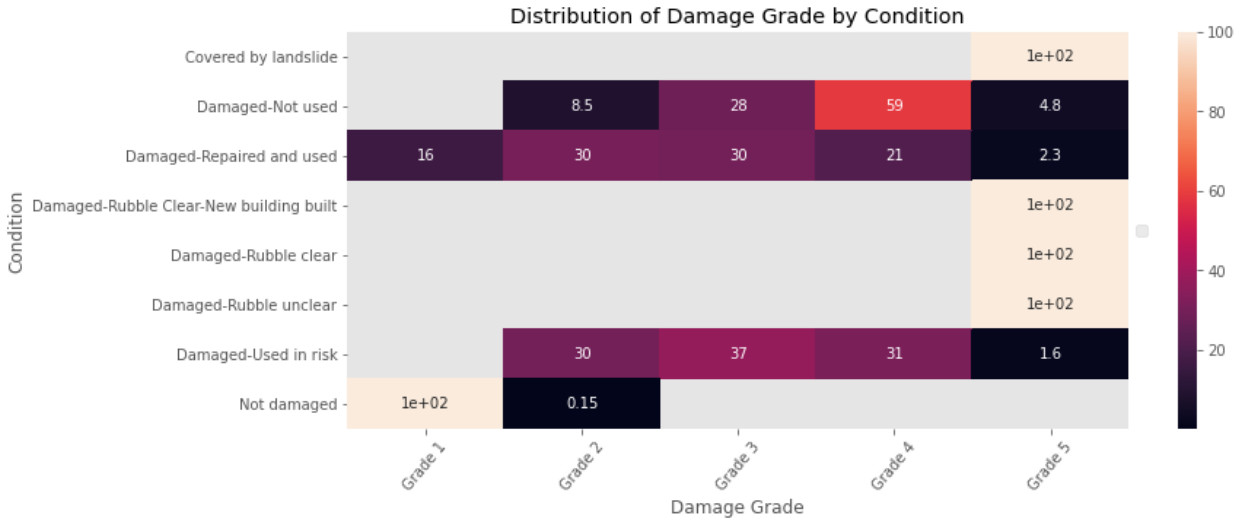
Exploratory Analysis

To start we look at the distribution of the target variable, we can see the occurence of each grade increases with the classification. Such that Grade 5 occurs the most frequently in the dataset while Grade 1 appears the least frequently.




```
Mud mortar-Stone/Brick    628705
Bamboo/Timber              57472
Cement-Stone/Brick         39245
RC                         32120
Other                       4552
```

```
Damaged-Not used                           207968
Damaged-Rubble unclear                     125650
Damaged-Used in risk                       123843
Damaged-Repaired and used                  107791
Damaged-Rubble clear                       102191
Not damaged                                 61139
Damaged-Rubble Clear-New building built     33130
Covered by landslide                          382
Name: condition_post_eq, dtype: int64
```


Distribution of Damage Grade by Condition
Covered by landslide
Damaged-Not used
Damaged-Repaired and used
Damaged-Rubble Clear-New building built
Damaged-Rubble clear
Damaged-Rubble unclear
Damaged-Used in risk
Not damaged
Condition
1e+02
8.5
28
59
4.8
16
30
30
21
2.3
1e+02
1e+02
1e+02
30
37
31
1.6
1e+02
0.15
Grade 1
Grade 2
Grade 3
Grade 4
Grade 5
Damage Grade
100
80
60
40
20