# **Team Members**

* Finhas Demissie
* Selam Habtewolde
* Beali Girma

# **Data Preparation/Feature Engineering**

**1. Overview**

This phase is critical in building a predictive model for landslide susceptibility, as it involves transforming raw data into a form suitable for machine learning algorithms. Proper feature engineering ensures that models can accurately capture the patterns in the data.

**2. Data Collection**

The dataset used for this project was sourced from **Kaggle**, containing relevant features such as elevation, rainfall, land cover, and more, which are crucial for predicting landslide risks. The dataset was loaded into the environment using Google Colab, and basic exploratory analysis was conducted.

*from google.colab import drive*

*drive.mount('/content/drive')*

*import pandas as pd*

*df = pd.read\_csv('/content/drive/MyDrive/Complete-data.csv')*

*df.head()*

**3. Data Cleaning**

Missing values were handled using the following approach:

* Missing values were identified using .isnull().sum() to count them.
* Basic imputation strategies were applied to ensure completeness.

*df.isnull().sum()*

**4. Exploratory Data Analysis (EDA)**

Exploratory data analysis (EDA) was performed to understand the distribution and relationships between variables. The key findings include the following:

* A correlation heatmap was used to identify relationships between different features.
* A histogram displayed the distribution of raw and normalized data.

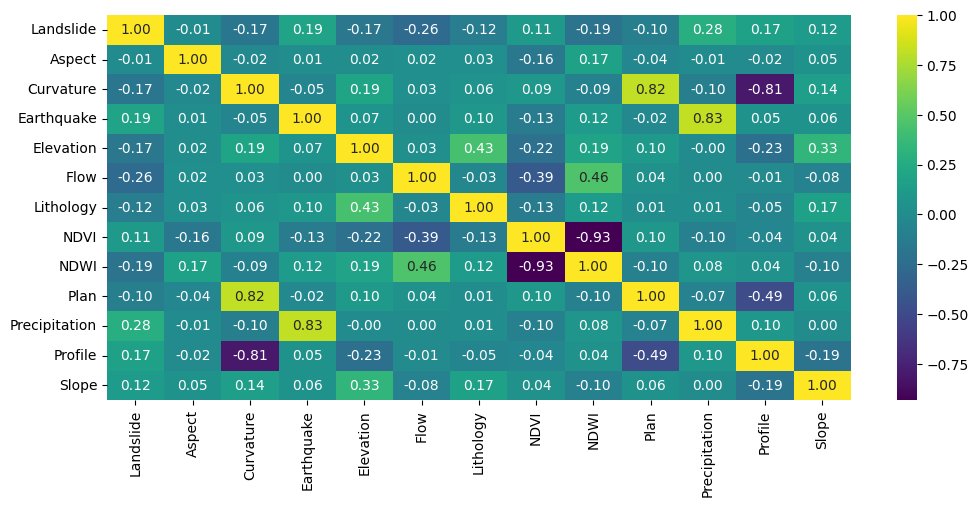
*import seaborn as sns*

*import matplotlib.pyplot as plt*

*plt.figure(figsize=(12, 5))*

*sns.heatmap(df.corr(), annot=True, cmap='viridis', fmt='.2f')*

*plt.show()*

**

**Key Visualizations**

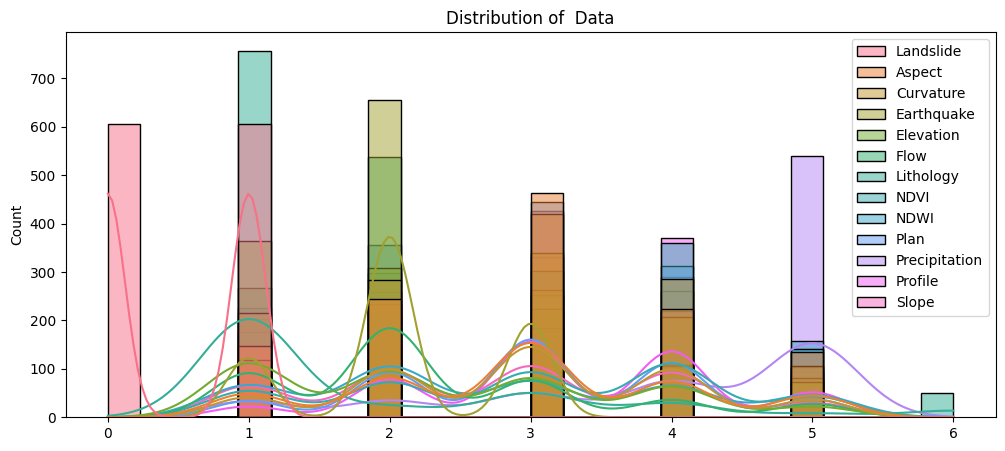
* Correlation Heatmap: This shows the relationship between key features such as elevation, rainfall, and land cover.
* Data Distribution: Visualizing the distribution of raw and scaled data highlights any skewness in features that may affect model performance.

*plt.figure(figsize=(12, 5))*

*sns.histplot(df, kde=True)*

*plt.title('Distribution of Data')*

*plt.show()*

**

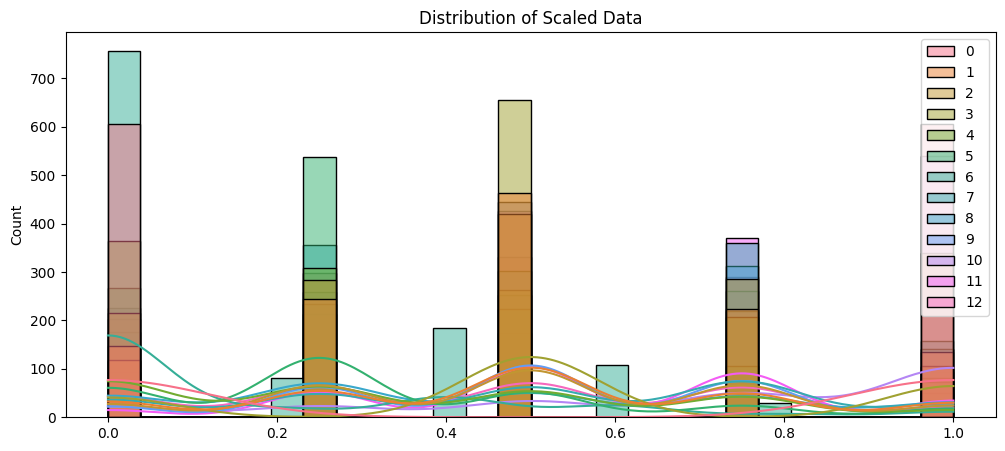
**5. Feature Engineering**

Feature scaling using **MinMaxScaler** was applied to normalize the dataset. This ensures that all features have the same scale, which is essential for distance-based algorithms.

*from sklearn.preprocessing import MinMaxScaler*

*scaler = MinMaxScaler()*

*df\_scaled = scaler.fit\_transform(df)*

**

# **Model Exploration**

**1. Model Selection**

Multiple machine learning models were considered for predicting landslide susceptibility, including:

* Logistic Regression
* Decision Tree Classifier
* Random Forest Classifier
* Support Vector Machine (SVM)
* K-Nearest Neighbors (KNN)

Each model has unique strengths and weaknesses, but ensemble models such as Random Forest generally provide higher accuracy due to their ability to reduce overfitting.

**2. Model Training**

Each model was trained on the prepared dataset. Hyperparameters were kept at their default values for the initial run, and models were evaluated using **accuracy** as the primary metric.

*from sklearn.linear\_model import LogisticRegression*

*from sklearn.tree import DecisionTreeClassifier*

*from sklearn.ensemble import RandomForestClassifier*

*from sklearn.svm import SVC*

*from sklearn.neighbors import KNeighborsClassifier*

*models = {*

*'Logistic Regression': LogisticRegression(),*

*'Decision Tree': DecisionTreeClassifier(),*

*'Random Forest': RandomForestClassifier(),*

*'Support Vector Machine': SVC(),*

*'K-Nearest Neighbors': KNeighborsClassifier()*

*}*

*for name, model in models.items():*

*model.fit(X\_train, y\_train)*

*y\_pred = model.predict(X\_test)*

*accuracy = accuracy\_score(y\_test, y\_pred)*

*print(f'{name}: Accuracy = {accuracy}')*

**3. Model Evaluation**

The accuracy of each model was visualized to compare their performance. The **Random Forest Classifier** was expected to perform the best due to its ensemble nature and ability to handle non-linear relationships in the data.

*import matplotlib.pyplot as plt*

*accuracies = {name: accuracy\_score(y\_test, model.predict(X\_test)) for name, model in models.items()}*

*plt.figure(figsize=(10, 6))*

*plt.bar(accuracies.keys(), accuracies.values())*

*plt.xlabel('Models')*

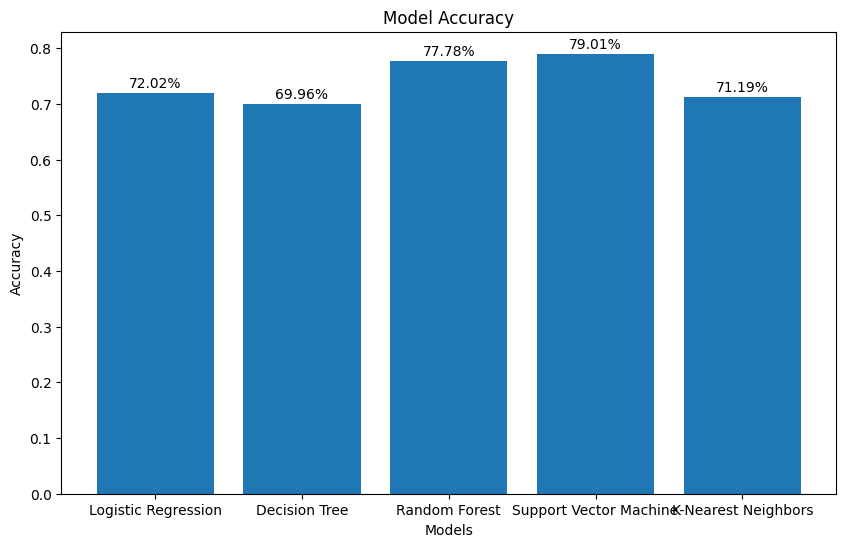
*plt.ylabel('Accuracy')*

*plt.title('Model Accuracy')*

*for i, v in enumerate(accuracies.values()):*

*plt.text(i, v + 0.01, f'{v:.2%}', ha='center')*

*plt.show()*

**

**Key Findings**

* **Random Forest** is likely the most suitable model due to its accuracy and ability to handle complex data patterns.
* **Logistic Regression** and **Decision Trees** provide a simpler interpretation but may lack precision for landslide susceptibility predictions.