Crowd Sourced Mapping

Abstract

This data-set is primarily designed to derive training data from crowd-sourced polygons, enabling the automated classification of the complete satellite image. Our challenge is to develop a machine learning model for classifying land cover based on geographical data. This report details the analysis of a dataset in the context of Crowdsourced Mapping using Multivariate Classification. The focus is on classifying various entities based on environmental or geographical features, primarily using 'max_ndvi' and other temporal data. The project involves in-depth statistical analysis and machine learning techniques to extract meaningful insights from the dataset, which comprises 10,545 entries with 29 features.

Introduction

The project explores multivariate classification in a geographical or environmental context. The primary aim is to classify different entities based on a comprehensive set of features, with a significant focus on temporal data analysis. The application of this study is critical in understanding and predicting environmental patterns and changes. Land, an essential natural resource, changes significantly as a result of human activity; this process is known as "land change." This phrase refers to any modification of the surface of the earth, including changes in land usage and cover. Land use refers to the ways in which humans use these lands, such as parks or agriculture, whereas land cover refers to the physical and biological elements that cover the land, such as vegetation and urban infrastructure.

Environmental management requires a thorough understanding of and vigilance over vegetation cover. The production of food and shelter, among other necessities, has led to human-induced modifications that have a considerable impact on environmental issues including pollution and climate change. Geoscience now relies heavily on machine learning, especially on techniques like logistic regression, neural networks and others to categorize vegetation cover. Crowdsourced georeferenced polygons and Landsat satellite images are two sources of detailed data that are used in this classification. In order to manage the effects of land change and maintain resource output, the essay concentrates on these cutting-edge machine learning techniques.

Dataset Description

DATA SOURCE: The data is taken from UCI Machine Learning repository. Johnson,Brian. (2016), Crowdsourced Mapping. Url link: https://archive.ics.uci.edu/dataset/400/crowdsourced+mapping

BACKGROUND

The dataset we used is Crowdsourced data from OpenStreetMap, which was utilized in a research specifically designed for the automated classification of satellite imagery into many land cover categories in the Laguna de Bay region of the Philippines. This dataset combines data from two main geographical sources: georeferenced polygons with land cover annotations gathered through crowdsourcing, and Landsat satellite imagery from 2014 to 2015. These crowdsourced polygons represent a small subset of the image area and are primarily used to gather training data that is necessary for categorizing the greater image region. Climate and environment are its subject areas, and it has multivariate dataset features. This dataset is focused on automating the classification of satellite images into distinct land cover classes. The classification includes categories such as impervious surfaces, farms, forests, grasslands, orchards, and water bodies

Columns	**Description**
class	The target variable to classify the land cover class.
max_ndvi	Maximum "Normalized Difference Vegetation Index" (NDVI) value
20140101_N - 20150720_N	Values of NDVI between January 2014 and July 2015

Normalized Difference Vegetation Index (NDVI) is a widely used metric for quantifying vegetation health and density. It is calculated using the following formula:

$$\textbf{NDVI} = (\text{NIR} - \text{RED}) / (\text{NIR} + \text{RED})$$

where NIR is reflectance of near-infrared light and RED is the reflectance of red light by the vegetation

The dataset contains 10,545 entries and 29 columns. Each entry represents an entity with various features, including 'max_ndvi' and other temporal data points. The data is divided into training and testing sets, indicating a supervised learning approach.

Exploratory Data Analysis (EDA)

DATA EXPLORATION

Data Inspection: Basic methods like .info(), .shape, and .head() are used for initial data inspection, providing insights into data types, the number of records, and a preview of the dataset.

```
[45]  train_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10545 entries, 0 to 10544
Data columns (total 29 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   class       10545 non-null  object
 1   max_ndvi    10545 non-null  float64
 2   20150720_N  10545 non-null  float64
 3   20150602_N  10545 non-null  float64
 4   20150517_N  10545 non-null  float64
 5   20150501_N  10545 non-null  float64
 6   20150415_N  10545 non-null  float64
 7   20150330_N  10545 non-null  float64
 8   20150314_N  10545 non-null  float64
 9   20150226_N  10545 non-null  float64
 10  20150210_N  10545 non-null  float64
 11  20150125_N  10545 non-null  float64
 12  20150109_N  10545 non-null  float64
 13  20141117_N  10545 non-null  float64
 14  20141101_N  10545 non-null  float64
 15  20141016_N  10545 non-null  float64
 16  20140930_N  10545 non-null  float64
 17  20140813_N  10545 non-null  float64
 18  20140626_N  10545 non-null  float64
 19  20140610_N  10545 non-null  float64
 20  20140525_N  10545 non-null  float64
 21  20140509_N  10545 non-null  float64
 22  20140423_N  10545 non-null  float64
 23  20140407_N  10545 non-null  float64
 24  20140322_N  10545 non-null  float64
 25  20140218_N  10545 non-null  float64
 26  20140202_N  10545 non-null  float64
 27  20140117_N  10545 non-null  float64
 28  20140101_N  10545 non-null  float64
dtypes: float64(28), object(1)
memory usage: 2.3+ MB
```

```
[46]  train_data.shape

(10545, 29)
```

```
train_data.head(10)
```

	class	max_ndvi	20150720_N	20150602_N	20150517_N	20150501_N	20150415_N	20150330_N	20150314_N	20150226_N	...
0	water	997.904	637.5950	658.668	-1882.030	-1924.360	997.904	-1739.99000	630.087	-1628.240	...
1	water	914.198	634.2400	593.705	-1625.790	-1672.320	914.198	-692.38600	707.626	-1670.590	...
2	water	3800.810	1671.3400	1206.880	449.735	1071.210	546.371	1077.84000	214.564	849.599	...
3	water	952.178	58.0174	-1599.160	210.714	-1052.630	578.807	-1564.63000	-858.390	729.790	...
4	water	1232.120	72.5180	-1220.880	380.436	-1256.930	515.805	-1413.18000	-802.942	683.254	...
5	forest	7091.960	5102.9000	6996.710	201.956	6130.950	6439.300	6818.67000	523.379	593.067	...
6	water	6423.920	1585.3100	2891.640	756.563	2978.580	3215.560	5033.86000	5049.720	5520.140	...
7	water	2455.480	1136.4400	-761.046	205.408	1647.830	1935.800	-44.56840	2158.980	-1367.920	...
8	water	2631.760	1116.8600	2631.760	-408.147	1685.700	1046.670	-7.58804	1435.990	-1145.830	...
9	water	3192.460	1485.7700	-223.142	727.773	180.491	1779.890	2613.97000	1869.390	-333.333	...

10 rows × 29 columns

```
[48]  train_data.columns

Index(['class', 'max_ndvi', '20150720_N', '20150602_N', '20150517_N',
       '20150501_N', '20150415_N', '20150330_N', '20150314_N', '20150226_N',
       '20150210_N', '20150125_N', '20150109_N', '20141117_N', '20141101_N',
       '20141016_N', '20140930_N', '20140813_N', '20140626_N', '20140610_N',
       '20140525_N', '20140509_N', '20140423_N', '20140407_N', '20140322_N',
       '20140218_N', '20140202_N', '20140117_N', '20140101_N'],
      dtype='object')
```

We have only one categorical value 'class' (target variable). Rest all are numeric columns (features)

Data Description: train_data.describe(include="all") gives a statistical summary of the data, including count, mean, standard deviation, and distribution of values across each column.

```
1 train_data.describe(include="all")
```

	class	max_ndvi	20150720_N	20150602_N	20150517_N	20150501_N	20150415_N	20150330_N	20150314_N	20150226_N	...
count	10545	10545.000000	10545.000000	10545.000000	10545.000000	10545.000000	10545.000000	10545.000000	10545.000000	10545.000000	...
unique	6	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
top	forest	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
freq	7431	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
mean	NaN	7282.721268	5713.832981	4777.434284	4352.914883	5077.372030	2871.423540	4898.348680	3338.303406	4902.600296	...
std	NaN	1603.782784	2283.945491	2735.244614	2870.619613	2512.162084	2675.074079	2578.318759	2421.309390	2691.397266	...
min	NaN	563.444000	-433.735000	-1781.790000	-2939.740000	-3536.540000	-1815.630000	-5992.080000	-1677.600000	-2624.640000	...
25%	NaN	7285.310000	4027.570000	2060.600000	1446.940000	2984.370000	526.911000	2456.310000	1017.710000	2321.550000	...
50%	NaN	7886.260000	6737.730000	5270.020000	4394.340000	5584.070000	1584.970000	5638.400000	2872.980000	5672.730000	...
75%	NaN	8121.780000	7589.020000	7484.110000	7317.950000	7440.210000	5460.080000	7245.040000	5516.610000	7395.610000	...
max	NaN	8650.500000	8377.720000	8566.420000	8650.500000	8516.100000	8267.120000	8499.330000	8001.700000	8452.380000	...

	20140610_N	20140525_N	20140509_N	20140423_N	20140407_N	20140322_N	20140218_N	20140202_N	20140117_N	20140101_N
count	10545.000000	10545.000000	10545.000000	10545.000000	10545.000000	10545.000000	10545.000000	10545.000000	10545.000000	10545.000000
unique	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
top	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
freq	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
mean	4787.492858	3640.367446	3027.313647	3022.054677	2041.609136	2691.604363	2058.300423	6109.309315	2563.511596	2558.926018
std	2745.333581	2298.281052	2054.223951	2176.307289	2020.499263	2408.279935	2212.018257	1944.613487	2336.052498	2413.851082
min	-3765.860000	-1043.160000	-4869.010000	-1505.780000	-1445.370000	-4354.630000	-232.292000	-6807.550000	-2139.860000	-4145.250000
25%	2003.930000	1392.390000	1405.020000	1010.180000	429.881000	766.451000	494.858000	5646.670000	689.922000	685.680000
50%	5266.930000	3596.680000	2671.400000	2619.180000	1245.900000	1511.180000	931.713000	6862.060000	1506.570000	1458.870000
75%	7549.430000	5817.750000	4174.010000	4837.610000	3016.520000	4508.510000	2950.880000	7378.020000	4208.730000	4112.550000
max	8489.970000	7981.820000	8445.410000	7919.070000	8206.780000	8235.400000	8247.630000	8410.330000	8418.230000	8502.020000

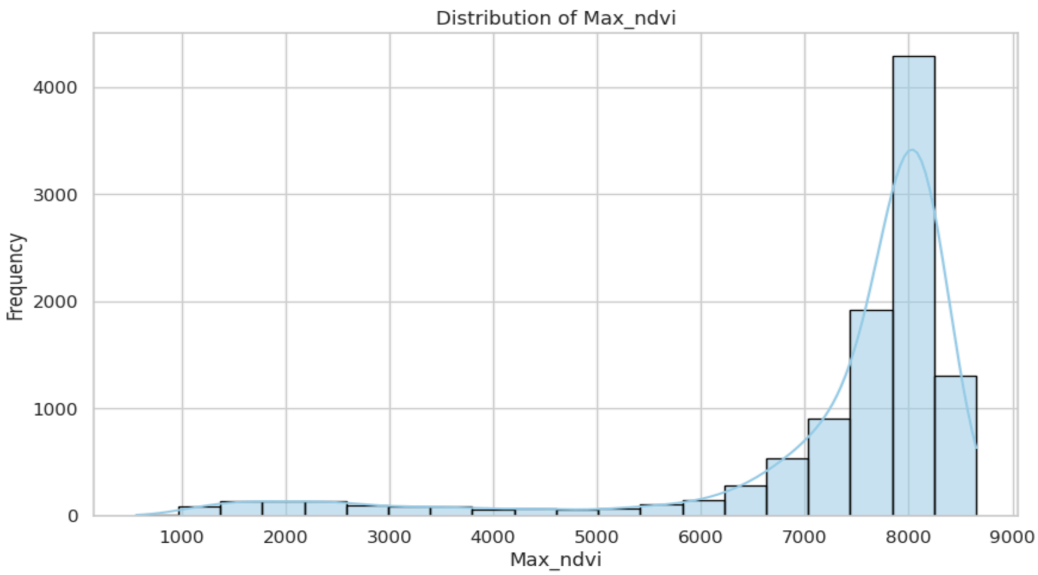
11 rows × 29 columns

There are no missing values in the dataset.

Target Variable Analysis: The target variable 'class' is identified as categorical, with the rest being numeric features. The unique classes in 'class' are identified and counted. Total number of unique in class are six (6) and they are 'water', 'forest', 'impervious', 'farm', 'grass' and 'orchard'

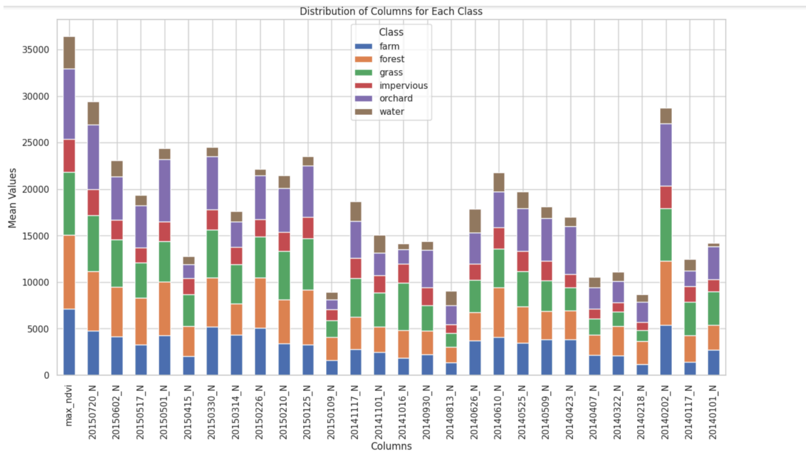
VISUALIZATION

Distribution of 'max_ndvi': A histogram with Kernel Density Estimate (KDE) plot is created to visualize the distribution of 'max_ndvi', a key feature in the dataset.



Max_ndvi values are skewed to the right and peak at 7000-8000, indicating dense vegetation in places with many observations.

Class Means: The mean values for each class are calculated and visualized, providing an insight into how different features vary across classes.



FEATURE ENGINEERING AND SELECTION

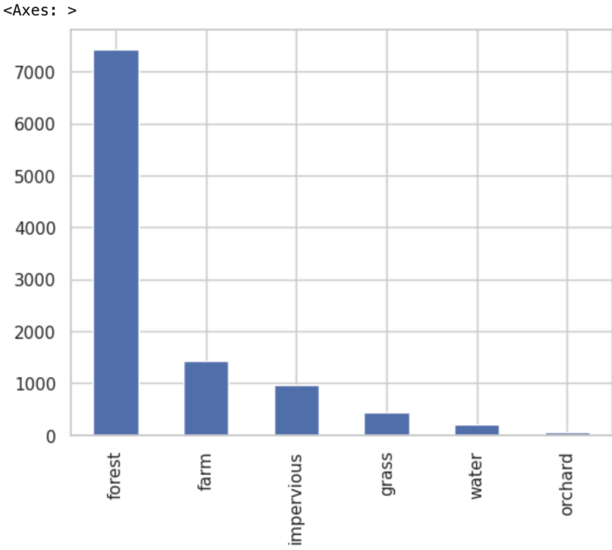
Data Preprocessing: The dataset is preprocessed with min-max scaling applied to the features. This normalization step is crucial for many machine learning algorithms to perform effectively.

$$x_{\text{normalized}} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

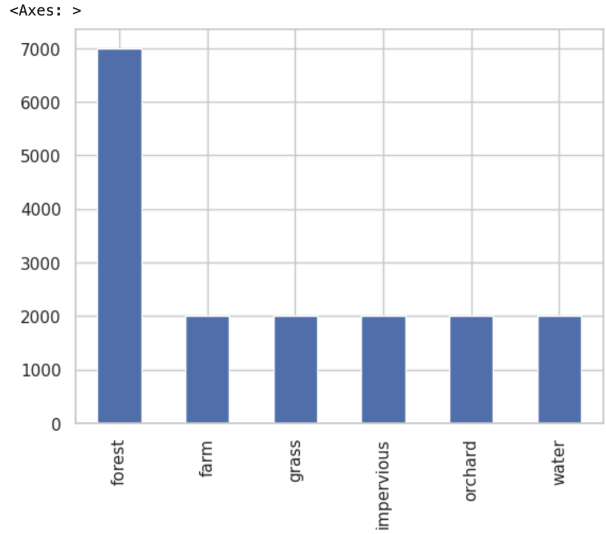
The min-max normalization will set all feature values between 0 and 1. This method works well when the parameters have multiple units or scales and when using algorithms like k-nearest neighbors and neural networks that assume the data is on the same scale.

Handling Class Imbalance: Techniques like SMOTE and RandomUnderSampler are employed to balance the dataset, indicating awareness of class imbalance issues.

<Axes: >



<Axes: >



In response to the issue of data imbalance, we employed SMOTE oversampling, which produced an augmented dataset comprising approximately 45,000 data points, which is three times the dimensions of the initial set. We elected to prioritize enhancing the representation of minority classes while simultaneously preserving the original dataset, as opposed to employing a broad oversampling technique that might overshadow the original data. The implementation of this particular oversampling methodology resulted in improved generalization and enhanced predictive capabilities.

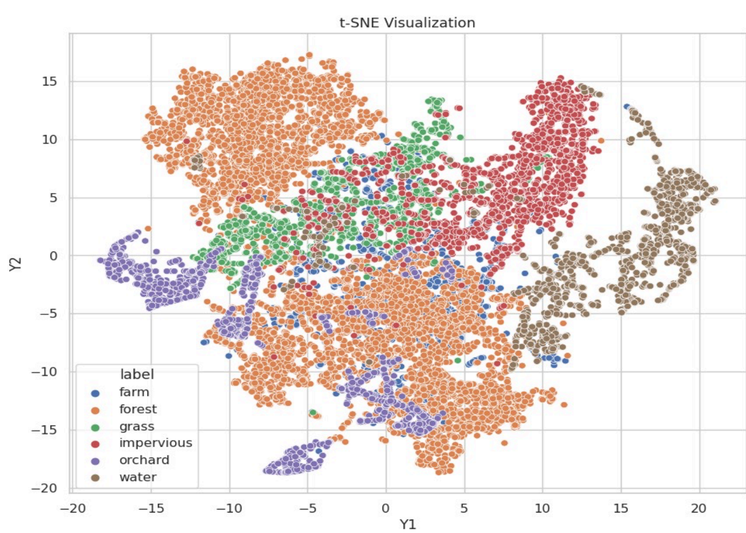
Dimensionality Reduction

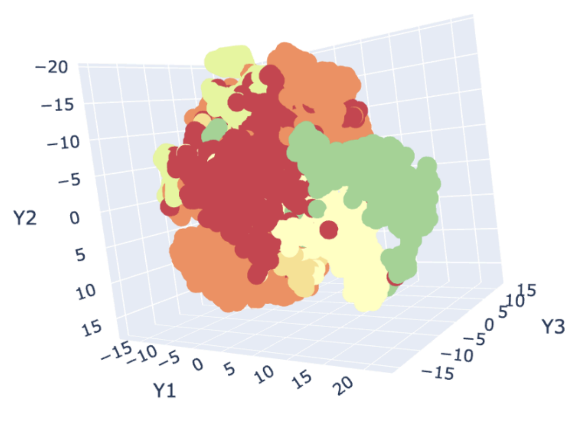
t-SNE: t-Distributed Stochastic Neighbor Embedding (t-SNE) is used to reduce the dimensions of the dataset for visualization, aiding in understanding the data structure and class separability.

Similarity Computation in High-Dimensional Space: t-SNE calculates high-dimensional instance similarities. Similarity is measured using Gaussian distributions centered on each data point. T-SNE creates probability distribution over pairs of data points by measuring similarity.

Similarity Computation in Low-Dimensional Space: t-SNE then transfers high-dimensional data to a 2D or 3D space for display and computes point similarity. Low-dimensional similarities are modeled with t-distributions.

Minimization of the Kullback-Leibler (KL) Divergence: T-SNE aims to minimize high-dimensional and low-dimensional distribution divergence with descending gradients. KL divergence indicates how a probability distribution contradicts an expected one.



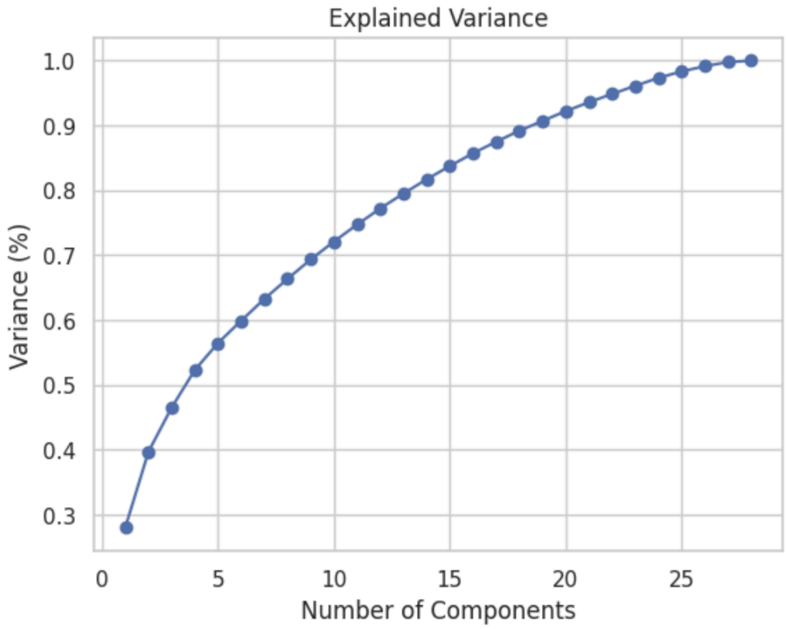


Lower-dimensional coordinate adjustment via t-SNE minimizes divergence. We opt for incorporating elevated perplexity values as an alternative to expanding the iteration count, aiming to streamline the execution time.

We're using something called t-SNE to simplify complex patterns in data. But, unfortunately, it's tough to clearly see the different groups in the data. We've been tweaking some hyperparameters to improve this, but there are still a lot of areas where the groups overlap.

PCA (Principal Component Analysis): PCA is implemented for dimensionality reduction, and the results are visualized using scatter plots. This step is crucial for understanding the inherent structure of the data and reducing computational complexity.

Covariance Matrix: The next step is to ascertain the interrelationships among the variables in the dataset. The variance measures the spread of the dataset around the mean, while the covariance assesses the joint variability of two variables.
Eigenvalues and Eigenvectors Determination: This phase involves computing the eigenvalues and eigenvectors of the covariance matrix, which are for identifying the principal components.
Selection of Principal Components: The following step is to select the first *k* eigenvectors, which will form the new *k*-dimensional space. Overall, Custom PCA implementation is provided to analyze the variance captured by different components, aiding in feature selection and understanding data variance.



Following the analysis, it was found that the dataset's variance is evenly spread across all variables, with no one big component showing a large portion of the variance. The fact that the variables' variance is spread out evenly suggests that there aren't any dominant features in the sample. This means that each variable adds about the same amount to the overall variability of the dataset.

Methods and Results

Logistic Regression

Our investigation commenced with the application of logistic regression, an algorithm adept at predicting the probability of multiclass outcomes. We selected this model for its proficiency in handling binary data that exhibits linear separability and independence, and is free from outliers.

Model Performance

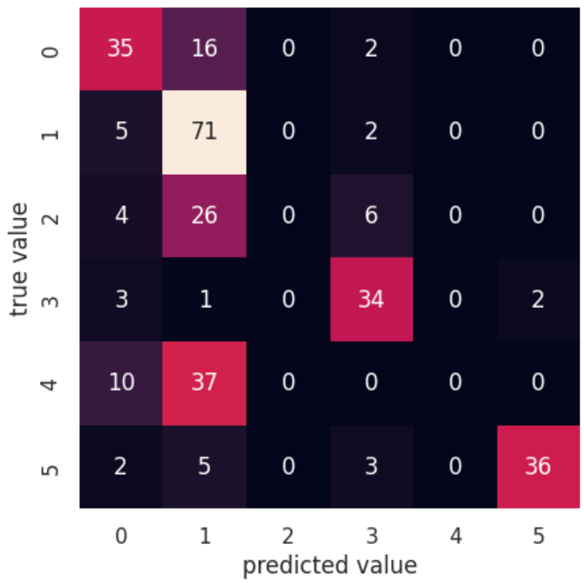
```
100%|██████████| 10000/10000 [00:50<00:00, 196.31it/s]

Accuracy : 70.5
Precision : 0.705
Recall : 1.0
Time:  51.01 seconds
```

Neural Network

Neural networks, inspired by the structure and function of the human brain, are a form of machine learning algorithm composed of layers of interconnected nodes, or neurons. These layers include the input layer, one or more hidden layers, and an output layer. They are particularly adept at deciphering intricate patterns and non-linear relationships within data sets. In the training phase, neural networks iteratively adjust the connections' weights to reduce the errors in predictions. They incorporate activation functions to add non-linear properties to the model, with the sigmoid function often being used in the output layer for binary classification tasks. While capable of processing diverse types of data, neural networks typically require substantial data and computational power. A common challenge is overfitting, which can be mitigated through strategies like regularization and the implementation of dropout layers

The results would detail the outcomes of the applied models, including performance metrics and insights derived from the analysis. This would involve a thorough examination of the classification results and their implications in the context of Crowdsourced Mapping.

```
Accuracy: 0.587
Recall (macro average): 0.534
Recall (weighted average): 0.587
```



Bias and Variance Tradeoff

To optimize the balance between bias and variance in our model development, we employed several strategies:

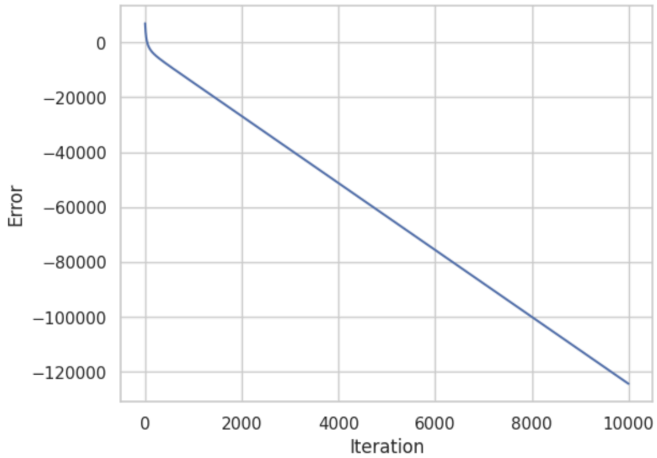
<u>Implementation of a Validation Set</u>: We partitioned our dataset into three segments: 70% for training, 15% for testing, and 15% for validation. This approach enabled us to assess our models using the validation set while keeping the test data separate and untouched. By comparing various models on the validation data, we identified the most accurate model for predictions on the test data.

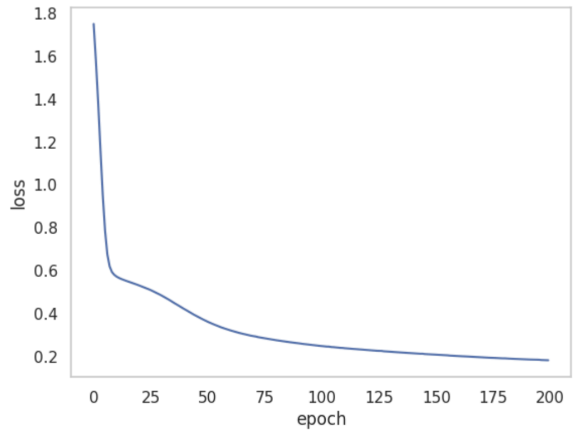
<u>Random Data Shuffling:</u> To avoid training bias towards any particular label category, we randomized the dataset before splitting it into training and testing subsets. This ensured a

balanced representation of data during training, reducing the likelihood of model bias and enhancing its generalization capabilities.

Hyperparameter Optimization: Through extensive experimentation with hyperparameters – such as learning rate and tolerance in logistic regression, and the number of neurons, batch size, and epochs in neural networks – we fine-tuned our models. Adjusting these parameters significantly improved model performance and reduced error rates.

- For instance, in our logistic regression implementation, we measured performance over time, this resulted in an accuracy of 70.5%, precision of 0.705, recall of 1.0, and a total runtime of 76.97 seconds.





- In contrast, our neural network model, structured with multiple dense layers which yielded an accuracy of 53.7%, with macro and weighted recall averages of 47.8% and 53.7%, respectively.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 28)                812

 dense_1 (Dense)             (None, 512)               14848

 dense_2 (Dense)             (None, 256)               131328

 dense_3 (Dense)             (None, 128)               32896

 dense_4 (Dense)             (None, 6)                 774

=================================================================
Total params: 180658 (705.70 KB)
Trainable params: 180658 (705.70 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

By incorporating these machine learning classification models, the equilibrium between variance and bias is significantly enhanced, hence playing a pivotal role in optimizing model performance. Variance, in this particular context, pertains to the model's susceptibility to changes in the training data, whereas bias denotes the inclination of the model to make erroneous assumptions about the data.

Discussion

The objective of the research is to create machine learning models that can accurately categorize land cover based on environmental or geographical characteristics, with a specific focus on utilizing the 'max_ndvi' and other temporal data. The dataset utilized consists of 10,545 records with 29 characteristics. The study investigated sophisticated machine learning methodologies to categorize land cover based on data obtained via crowdsourced mapping. The primary techniques employed were logistic regression and neural networks, with an emphasis on addressing issues such as class imbalance, bias and variance tradeoff, and dimensionality reduction.

The logistic regression model demonstrated high performance, exhibiting remarkable accuracy and precision. In contrast, the neural network model, although it had lesser accuracy, yielded valuable insights into intricate data patterns. The study highlighted the importance of data preparation, feature engineering, and the utilization of advanced algorithms to accurately analyze and categorize environmental data. The findings and knowledge acquired from this study are essential for comprehending environmental patterns and fluctuations, showcasing the capacity of machine learning in geospatial analysis.