GeoGenIE: Geographic Genetic Inference Engine

Geolocation predicting from SNPs using deep learning

Drs. Bradley T. Martin, Tyler K. Chafin, Zach D. Zbinden, Marlis R. Douglas, and Michael E. Douglas

Version: 1.0.4

2025-04-10

Contents

List of Figures

GeoGenIE: Geolocation Predictions from SNPs using Deep Learning

Introduction

GeoGenIE (Geographic-Genetic Inference Engine) is a comprehensive software tool designed to predict geographic coordinates (longitude and latitude) from genetic SNP data. GeoGenIE utilizes deep learning models and offers several advanced features such as automated model parameter searches (via Optuna (Akiba et al. 2019)), outlier detection to remove translocated individuals (based on the GGOutlieR method (Chang et al. 2023)), handling of imbalanced sampling using a custom oversampling algorithm adapted from SMOTE (Lemaître, Nogueira, and Aridas 2017), and weighting the loss function by inverse sampling densities. The software is user-friendly and provides extensive visualizations and metrics for evaluating predictions, making it a robust and accurate solution for geographic-genetic inference.

GeoGenIE is designed for researchers in population genetics and molecular ecology, providing a powerful tool to infer geographic origins of individuals based on their genetic data. The software employs state-of-the-art deep learning techniques to handle complex genetic data and generate precise geographic predictions. However, researchers often face challenges due to imbalanced sampling, where certain geographic regions are overrepresented while others are underrepresented in the data. GeoGenIE addresses these challenges through several innovative algorithms.

GeoGenIE incorporates an outlier detection algorithm adapted from GGOutlieR to identify and remove individuals that have been translocated. This is crucial for ensuring the accuracy

of geographic predictions by eliminating samples that could introduce bias. Additionally, GeoGenIE implements a weighted loss function using PyTorch, where inverse sample weights are used to focus the loss function more heavily on areas with lower sample densities. This approach helps in balancing the influence of samples from different regions, improving the robustness of the model.

Furthermore, GeoGenIE employs a regression-based synthetic oversampling method adapted from SMOTE. This method uses a genotype interpolation algorithm based on Mendelian inheritance to generate synthetic samples in underrepresented regions, thereby balancing the dataset. These advanced algorithms collectively enable GeoGenIE to provide reliable geographic predictions even in the presence of sampling imbalances.

Deep Learning Model Architecture

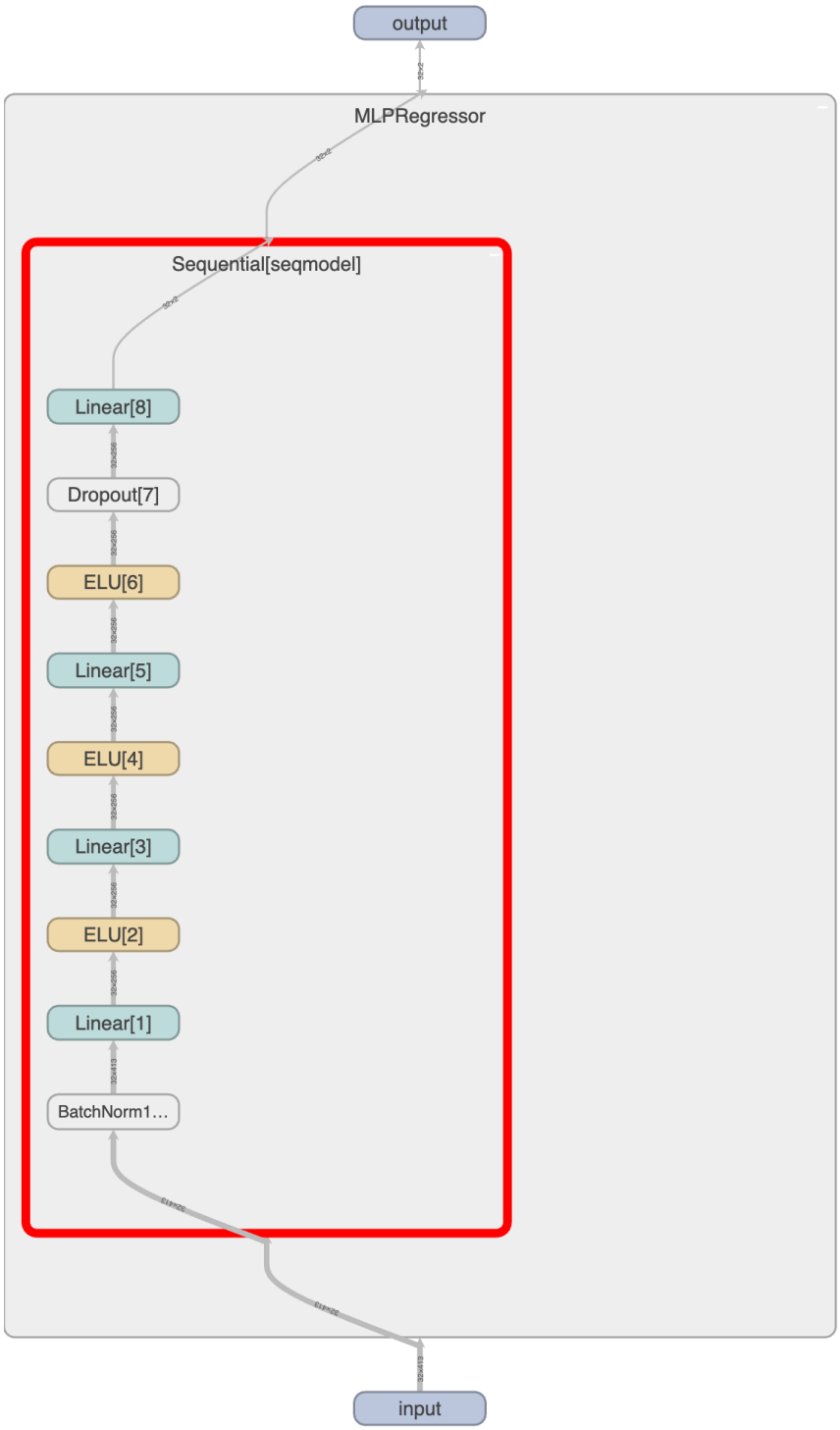

Figure 1: GeoGenIE model architecture diagram.

GeoGenIE was written in PyTorch. Below is the deep learning model architecture (Figure 1), as adapted from the original Locator architecture (Battey, Ralph, and Kern 2020). GeoGenIE allows lots of flexibility in the architecture, with each hidden layer either being constant or reduced by a factor with the `--factor` option. The model also includes batch normalization and dropout layers to reduce overfitting and facilitate better cross-batch training.

Installation

To install GeoGenIE, it is recommended to use a virtual environment or conda environment. From the root project directory, enter the following command:

```
pip install GeoGenIE
```

Dependencies

The following packages will be installed when running `pip install GeoGenIE`:

Here is the list of dependencies:

- python >= 3.11,<3.13
- geopandas
- geopy
- imblearn
- jenkspy
- kaleido
- kneed
- matplotlib
- numba
- numpy
- optuna
- pandas
- plotly
- pynndescent
- pykrige
- pysam
- pyyaml
- requests
- scikit-learn
- scipy
- seaborn
- statsmodels
- torch
- xgboost

Usage

Running GeoGenIE

GeoGenIE can be run with individual command-line arguments, but using a YAML config file is recommended. See `config_files/config.yaml` for a template YAML file. Assuming GeoGenIE is installed in your environment, you can run it like this:

```
geogenie --config config_files/config.yaml
```

Command-line Options

You can see all the command-line options by running the help flag:

```
geogenie -h
```

Note: If you do not want to use the configuration file, you can specify each argument individually on the command line. For example:

```
geogenie --vcf <path/to/vcf_file.vcf.gz> --sample_data
   <path/to/coordinates_file.tsv> <other arguments>
```

We do recommend using the configuration file, however, as it enables reproducible runs and also promotes ease-of-use when performing multiple runs.

Configuration File

You can set all the options for input files, model parameters, etc., in the `config_files/config.yaml` file. Using a configuration file allows tracking of parameters across multiple runs and ensures better reproducibility.

- Python `None` values are represented by `null` (without quotes).
- Python `True` values are represented by `true` (all lowercase, no quotes).
- Python `False` values are represented by `false` (all lowercase, no quotes).

You can also leave comments with `# my_comment`. The arguments can be in any order in the `config_files/config.yaml` file.

Running the Software

```
geogenie --config config_files/config.yaml
```

Required Input Files

Input Argument	Description
vcf	VCF file containing SNP data.
sample_data	CSV or TSV file with per-sample coordinates. Columns: "sampleID", "x", "y". Set unknown coordinates to "nan"

Input Argument	Description
known_coords_file	File with known coordinates for all samples. For per-sample bootstrap plots. Can be same as sample_data.

Algorithms to Mitigate Sampling Imbalance

GeoGenIE employs several advanced algorithms to accommodate sampling imbalances, ensuring robust and accurate geographic predictions:

Feature	Description
detect_outliers	Remove individuals deviating from expected geographic and/or genetic patterns.
use_weighted	Use inverse sample weights, focusing loss function on areas with lower densities.
oversample_method	Oversamples by generating synthetic samples in underrepresented regions.

GeoGenIE Features and Settings

Data Input and Preprocessing

GeoGenIE supports various options for data input and preprocessing:

Option	Description	Default	Importance
min_mac	Filters out SNPs with a minor allele count below the specified threshold.	2	**High**
max_SNPs	Limits the number of SNPs used in the analysis to reduce computational load.	None (Use all SNPs)	**Medium**

Model Configuration

Configure the deep learning model with the following options:

Option	Description	Default	Importance
dropout_prop	Dropout rate to reduce overfitting.	0.25	**High**
nlayers	Number of hidden layers in the neural network.	10	**Medium**
width	Number of neurons per hidden layer.	256	**Medium**
criterion	Loss function. Options: 'rmse', 'huber', 'drms'.	"rmse"	**Medium**
load_best_params	Reuse the best parameters from a previous Optuna search	None	**Medium**

Option	Description	Default	Importance
dtype	PyTorch data type. Options: 'float32' or 'float64'.	"float32"	**Medium**

Model Configuration Tips

- **dropout_prop**: Adjust higher to reduce overfitting. Lower if underfitting.
- **criterion**: We recommend starting with the default "rmse" criterion, and then if you get poor performance, try "huber" next.
- **load_best_params**: Load the best params from the JSON file saved when running the Optuna grid search.
- **dtype**: Only "float32" is supported if using a GPU.
- **nlayers** and **width**: More layers or higher widths can learn more complex models, but use caution; setting too high can lead to overfiting.

Training Parameters

Define training parameters:

Option	Description	Default	Importance
max_epochs	Maximum number of training cycles.	5000	**High**
learning_rate	Step size used to update model weights.	0.001	**High**
train_split	Proportion of the dataset used for training.	0.8	**High**
val_split	Proportion of the dataset used for validation.	0.2	**High**
batch_size	Samples processed before updating model weights.	32	**Medium**
early_stop_patience	Epochs with no improvement before early stopping.	48	**Medium**
l2_reg	Used to penalize large weights, reducing overfitting.	0.0	**Medium**
do_bootstrap	Enable bootstrapping to estimate confidence intervals.	False	**Medium**
nboots	Number of bootstrap replicates.	100	**Medium**

Training Parameter Tips

- **max_epochs**: Set this high and let early stopping take effect.
- **train_split** and **val_split**: Ensure these sum to 1.0.
- **batch_size**: Larger values can lead to more training stability, but consumes more memory.

- **do_bootstrap**: Use this to estimate confidence intervals for predictions and evaluations.

Geographic Density Sampler

Configure the geographic density sampler:

Option	Description	Default	Importance
use_weighted	Weights samples by inverse density during training.	"none"	**High**
oversample_method	Generates synthetic samples in underrepresented regions.	"none"	**High**
oversample_neighbors	Number of nearest neighbors with synthetic samples.	5	**Medium**
use_kmeans	Use KMeans clustering for calculating inverse weights.	False	**High**
use_kde	Use Kernel Density Estimation to calculate inverse weights.	False	**High**
use_dbscan	Use DBSCAN clustering to calculate inverse weights.	False	**Low**
n_bins	Adjust granularity of the sampling density (KMeans method)	8	**Medium**
w_power	Controls the strength of the sample weighting.	1.0	**Medium**
max_clusters	Upper limit for the number of clusters with KMeans.	10	**Medium**
focus_regions	Specifies regions to prioritize during sampling.	None	**Low**
normalize_sample_weights	Put all sample weights on a comparable scale.	False	**Low**

Geographic Density Sampler Tips

- **use_kmeans** and **use_kde**: These methods are used to estimate inverse sampling densities for weighting samples during training. Gets used with the weighted loss function.
- **use_dbscan**: This method is highly experimental still. Use with caution.
- **w_power**: Increase to make sample weights more aggressive.
- **use_weighted**: Supported options are "none" or "loss". Enable "loss" weighting to focus model training on underrepresented regions to mitigate sampling imbalance.
- **oversample_method**: Enable this to generate synthetic samples in underrepresented regions in order to balance sampling densities. Supported options are "none" or "kmeans".

Outlier Detection

GeoGenIE can remove outliers flagged as distant from nearby samples in spatial and genetic contexts:

Option	Description	Default	Importance
detect_outliers	Remove samples deviating from expected geographic and genetic patterns.	False	**High**
min_nn_dist	Threshold (meters) to consider samples as outliers.	1000	**Medium**
scale_factor	Adjust geographic distance scaling for outlier detection.	100	**Low**
significance_level	Set the p-value threshold for identifying outliers.	0.05	**Medium**
maxk	Set number of nearest neighbor range considered in outlier detection.	50	**Medium**

Outlier Detection Tips

- **detect_outliers**: Use this option if you suspect your study system has a history of e.g., translocations.
- **min_nn_dist**: Increase to detect only very distant outliers. Useful to exclude neighbors in close proximity.
- **scale_factor**: Best not to mess with, unless necessary.

Bootstrapping for Error Estimates

To obtain confidence intervals for locality predictions, enable bootstrapping with the `--do_bootstrap` boolean option. Bootstrapping is parallelized, and you can set the number of CPU threads with `--n_jobs <n_cpus>` or `--n_jobs -1` to use all available CPU threads.

Using bootstrapping generates additional plots showing confidence intervals for each sample, saved in `<output_dir>/plots/bootstrapped_sample_ci/<prefix>_bootstrap_ci_plot_<test/val/pr`

The file type for output plots can be specified with `--filetype "pdf"`, `--filetype "png"`, or `--filetype "jpg"`. The number of bootstrap replicates can be changed with `--nboots <integer>`.

Embedding Settings

GeoGenIE offers several embedding options for input features (i.e., loci). We recommend starting without using embeddings, but if you have very high-dimensional data or are getting poor performance due to many uninformative loci, try using one of the embedding methods:

Option	Description	Default	Importance
embedding_type	Embedding input SNPs to reduce dimensionality.	"none"	**High**

Option	Description	Default	Importance
n_components	Set the number of components to retain in the embedding.	None	**Medium**
embedding_sensitivity	Adjust the sensitivity for determining number of components.	1.0	**Medium**
tsne_perplexity	Control the balance between local and global aspects T-SNE.	30	**Medium**
polynomial_degree	Set the polynomial degree if "polynimial" method is used.	2	**Low**
n_init	Set number of embedding initializations.	4	**Low**

Embedding Setting Tips

- **embedding_type**: Supported options include: "none", "kernelpca", "nmf", "lle", "mca", "mds", "polynomial", and "tsne". We recommend starting with "none". This option is most useful if you have many loci that are uninformative. "lle" = Locally Linear Embedding, mca = "Multiple Correspondence Analysis", "nmf" = "Non-negative Matrix Factorization", "mds" = "Multi-Dimensional Scaling", "tsne" = "T-distributed Stochastic Neighbor Embedding", "polynomial" = "PolynomialFeatures".
- **n_components**: Number of components (dimensions) to retain with embedding.
- **polynomial_degree**: Only used if "embedding_type" is set to "polynomial". **CAUTION**: Setting this value higher than 2 can lead to extremely heavy computational loads.

Plot Settings

Set plotting parameters to customize the visualizations:

Option	Description	Default	Importance
show_plots	Control whether plots are displayed interactively (in-line).	False	**Low**
fontsize	Set the font size for all text in the plots.	24	**Low**
filetype	Specify the file format for saving plots.	"png"	**Low**
plot_dpi	Set the resolution for image format plots.	300	**Low**
remove_splines	Control whether axis lines are removed from plots.	False	**Low**
shapefile	Specify the shapefile to use as a base map.	Continental USA	**Low**
basemap_fips	Subset the basemap to focus on a specific region using FIPS code.	None	**Low**
highlight_basemap_counties	Highlight counties on the base map by name.	None	**Low**
samples_to_plot	Specify samples to plot with bootstrap contours.	None	**Low**
n_contour_levels	Set the number of contour levels for Kriging plots.	20	**Low**
min_colorscale	Set the minimum value for the color scale in Kriging plots.	0	**Low**
max_colorscale	Set the maximum value for the color scale in Kriging plots.	300	**Low**

Option	Description	Default	Importance
sample_point_scale	Adjusts the size of sample points in plots.	2	**Low**
bbox_buffer	Adds a buffer around the sampling area in map visualizations.	0.1	**Low**

Output and Miscellaneous

Option	Description	Default	Importance
prefix	Set a prefix for all output files.	"output"	**High**
output_dir	Specify the directory for storing output files.	"output"	**High**
n_jobs	Number of CPU threads used for parallel processing.	-1	**High**
gpu_number	Specify the GPU to use for computation.	None (CPU only)	**Low**
seed	Set a random seed for reproducible results results.	None	**Low**
sqldb	Store Optuna optimization results in SQLite3 database	None	**Low**
verbose	Set the level of detail for logging messages.	1	**Low**

Output Files and File Structure

Outputs are saved to the directory specified by `--output_dir <my_output_dir>/<prefix_>_*`. The prefix is specified with `--prefix <prefix>`. The directory structure of `<output_dir>` includes:

Directory	Description
benchmarking	Execution times for model training and prediction.
bootstrapped_sample_ci	One plot per sample showing confidence intervals on a map.
bootstrap_metrics	Files with evaluation metrics per bootstrap.
bootstrap_predictions	CSV files containing predictions for each bootstrap replicate.
bootstrap_summaries	Bootstrap summary statistics (aggregated).
data	Text files with detected outliers.
logfiles	Logs with INFO, WARNING, and ERROR messages.
models	Trained PyTorch models saved as ".pt" files.
optimize	Optuna results, including the best-found parameters (JSON file).

Directory	Description
plots	All plots and visualizations.

CAUTION: Re-running GeoGenIE with the same `output_dir` and `prefix` will overwrite all outputs except the Optuna SQL database.

Plot Descriptions

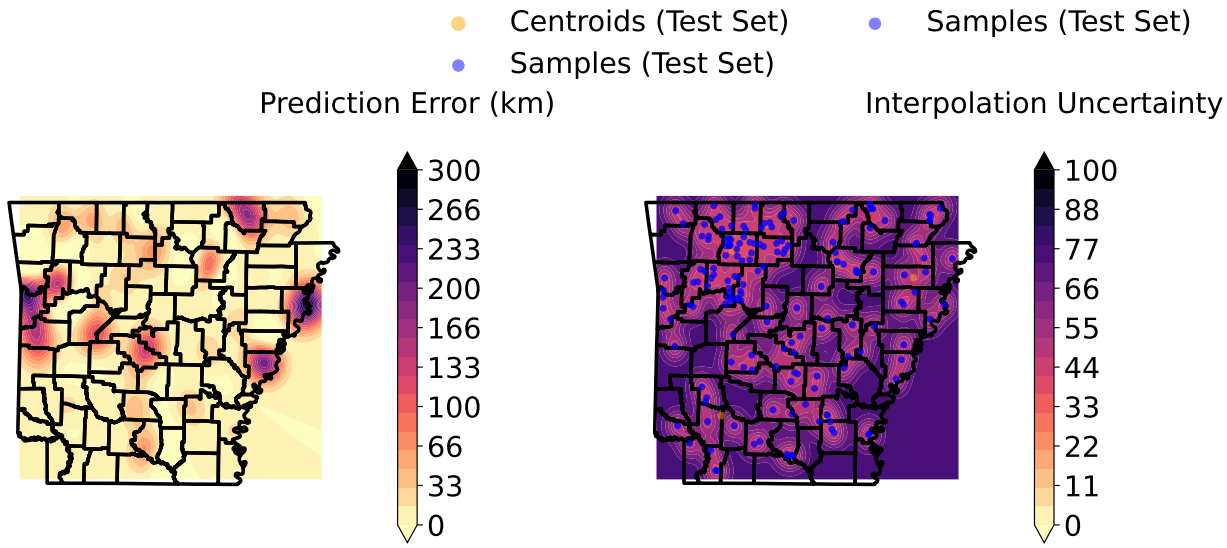

Figure 2: Geographic error distribution of the model predictions interpolated across the Arkansas landscape. Interpolated contour levels represent error magnitudes. Prediction error is Haversine distance between the predicted and recorded localities, in km. This hold-out test dataset was used to obtain realistic predition error estimates.

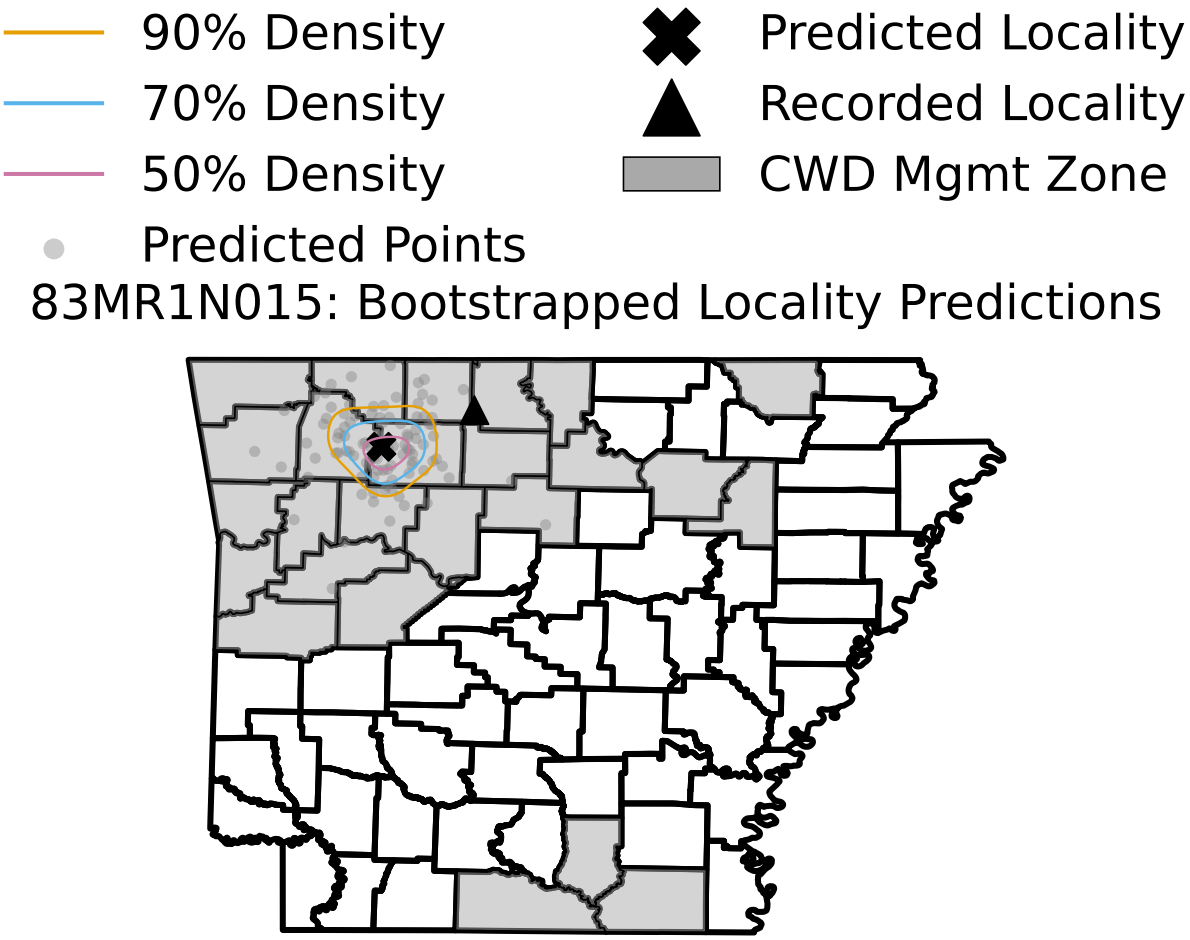

Figure 3: GeoGenIE bootstrap predictions (gray circles; N=100), with the geographic centroid of the bootstrap replicates being marked by **X** and the recorded locality as ▲. Orange, blue, and pink contours contain 90, 70, and 50 percent of the bootstrap replicates, respectively. This hold-out test dataset was used to obtain realistic predition error estimates.

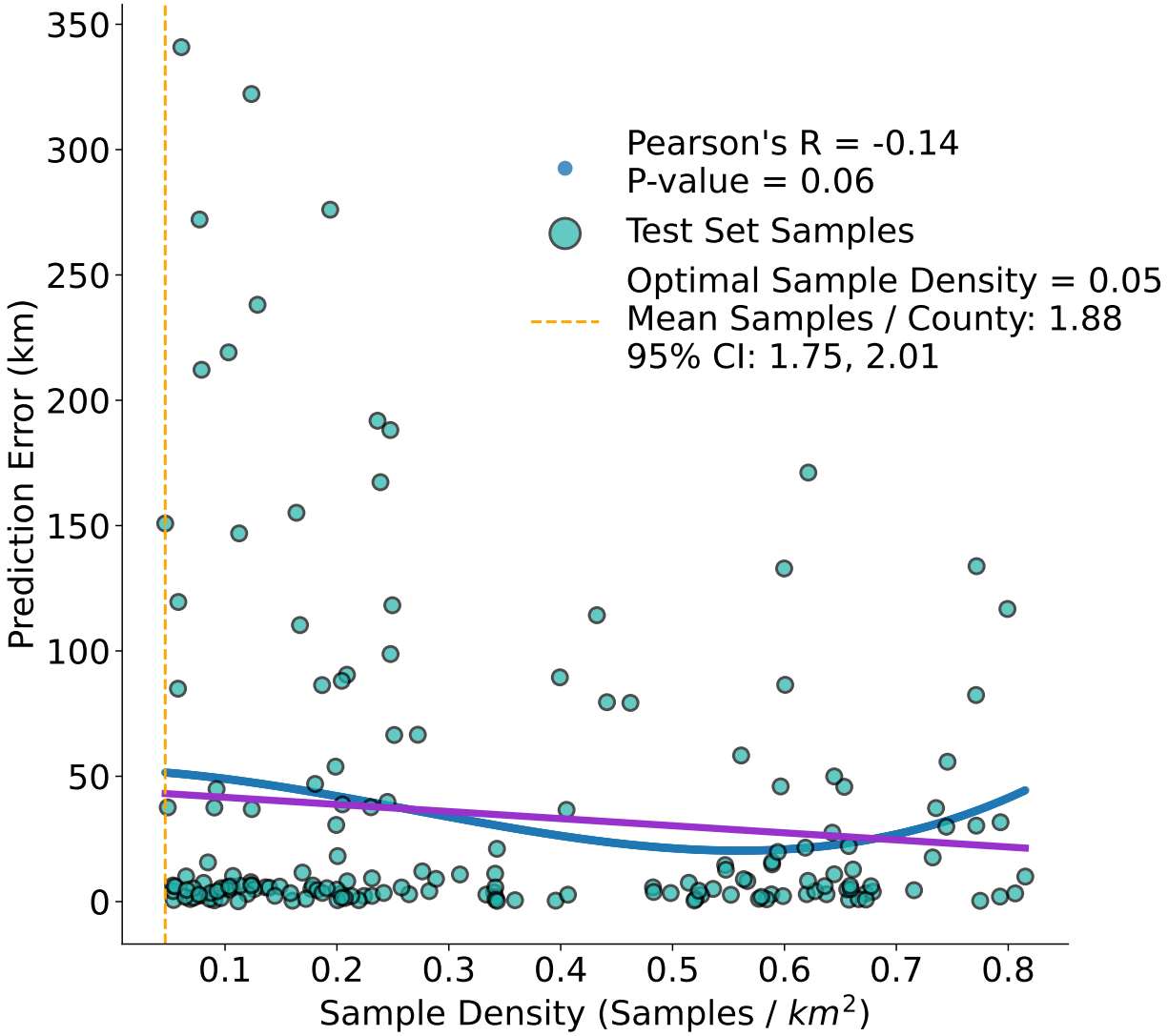

Figure 4: Linear and non-linear (3 *rd* order polynomial) regressions between sampling density (samples / km^2) and prediction error (km). Prediction error is the Haversine distance between the predicted and recorded localities. The orange dashed line represents optimal sampling density as the knee of the polynomial curve, beyond which sampling efforts may yield diminishing returns. This hold-out test dataset was used to obtain realistic predition error estimates.

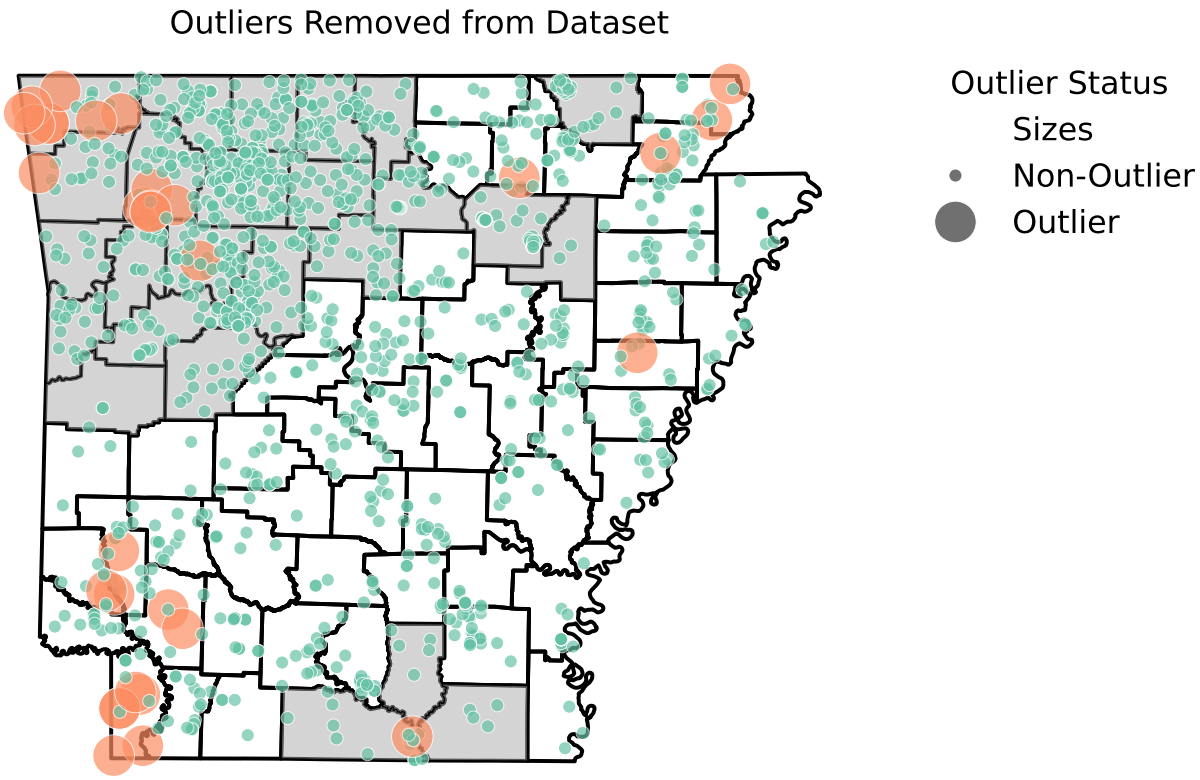

Figure 5: Map depicting sample outliers (large orange circles) removed from the training dataset by our algorithm adapted from GGOutlieR. Non-outliers are illustrated as the smaller green circles.

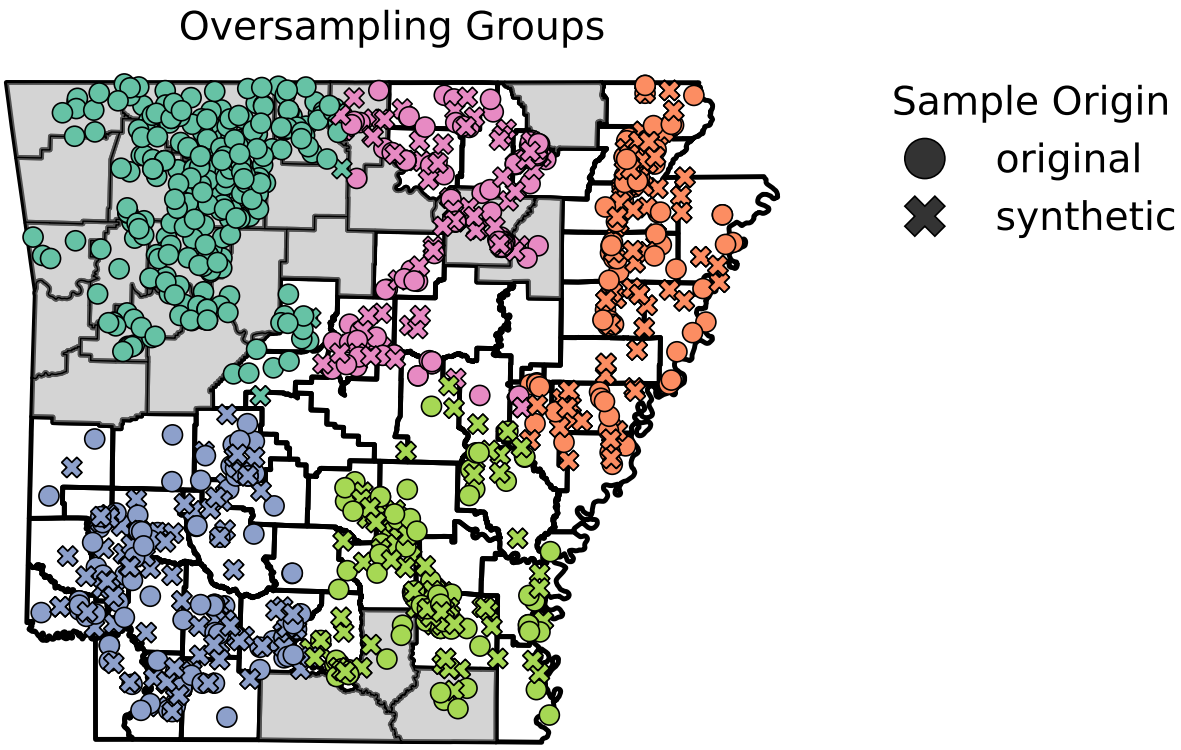

Figure 6: Training dataset samples, with "x" markers depicting synthetically created samples via our custom Mendelian inheritance interpolation method algorithm from a regression-based SMOTE method. Synthetic sample generation frequencies are inversely proportional to the sampling density (samples / km^2). Circles represent real samples that were not synthetically created.

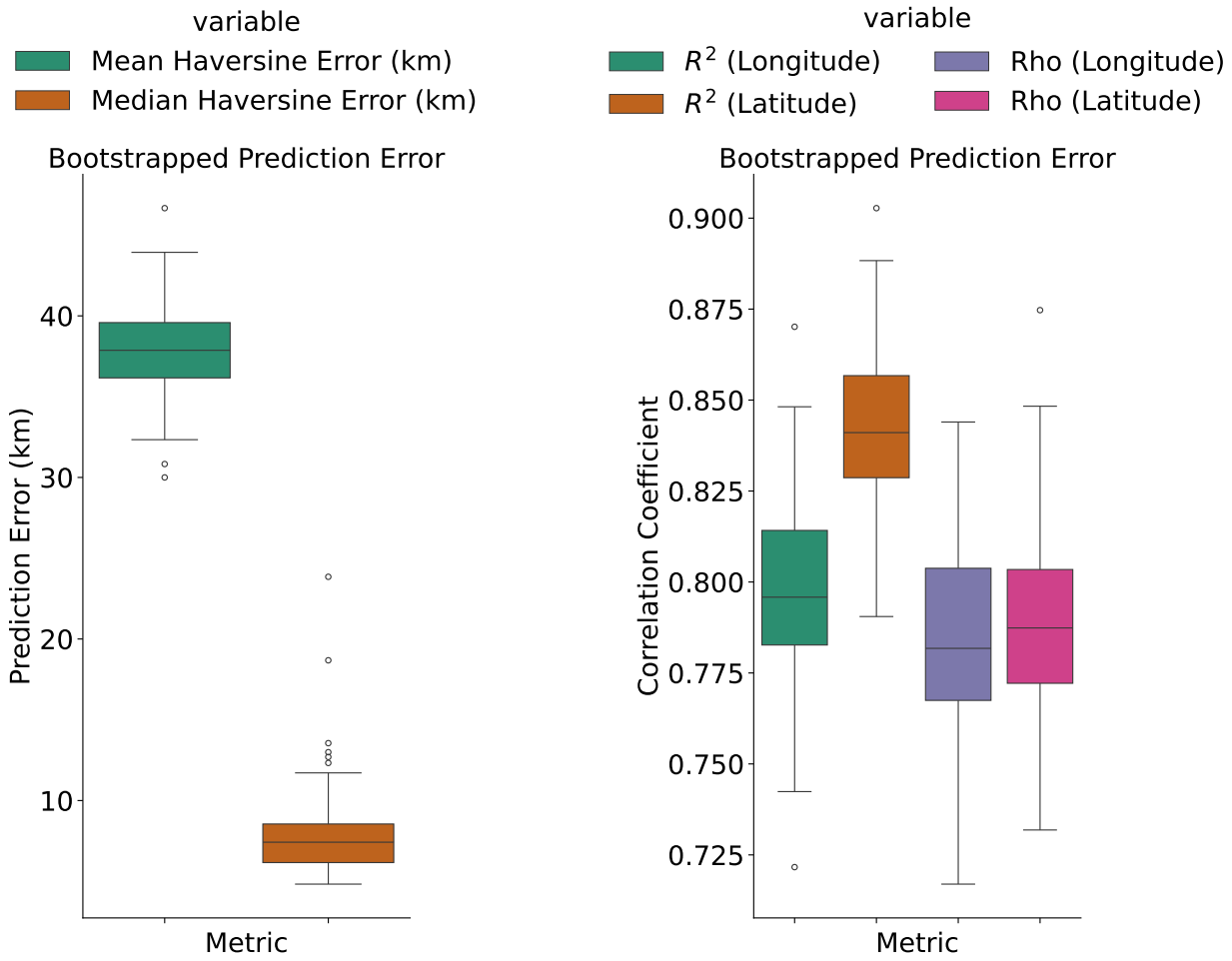

Figure 7: Boxplots, summarized across '–nboots' bootstrap replicates, showing (Left) the mean and median prediction error, represented as the Haversine distance between predicted and recorded localities (in Kilometers). (Right) Pearson's and Spearman's correlation coefficients depicting the correlation between the predicted and recorded localities. This hold-out test dataset was used to obtain realistic predition error estimates.

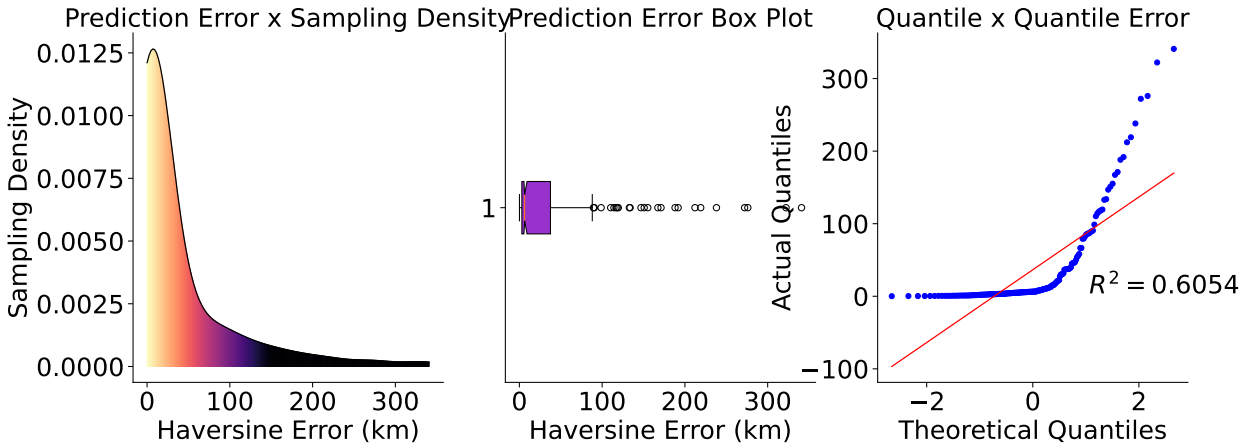

Figure 8: (Left) Area plot depicting prediction error (i.e., Haversine distance between predicted and recorded localities, in km) versus sampling density (samples / km^2). The color gradient corresponds to the geographic interpolation of prediction error in (Figure 2). (Middle) Boxplot showing the mean prediction error. (Right) Quantile X quantile regression plot of mean prediction error. This hold-out test dataset was used to obtain realistic predition error estimates.

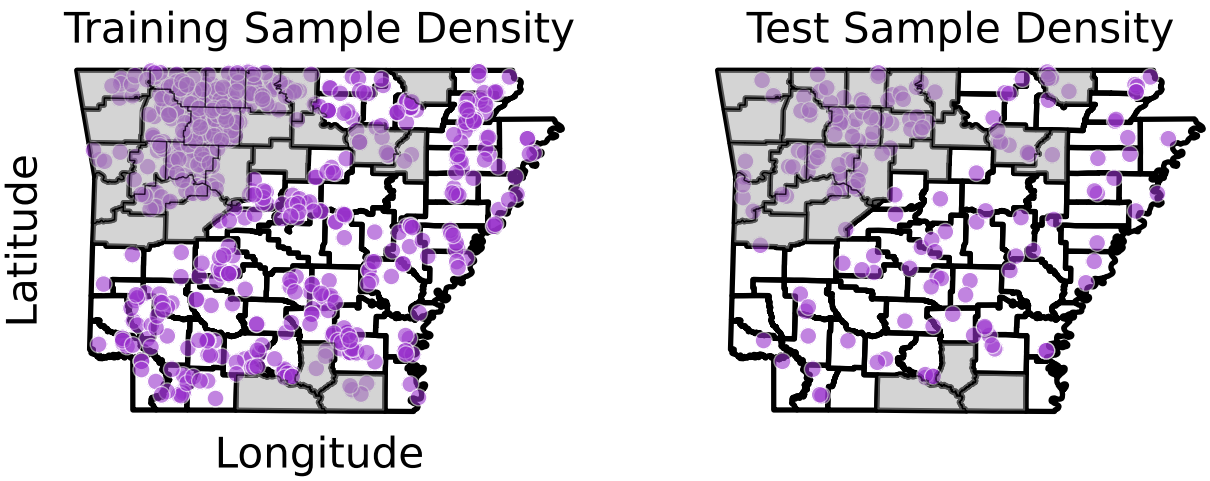

Figure 9: Samples (purple circles) selected for the training and test (i.e., hold-out) datasets and visualized on a map of Arkansas.

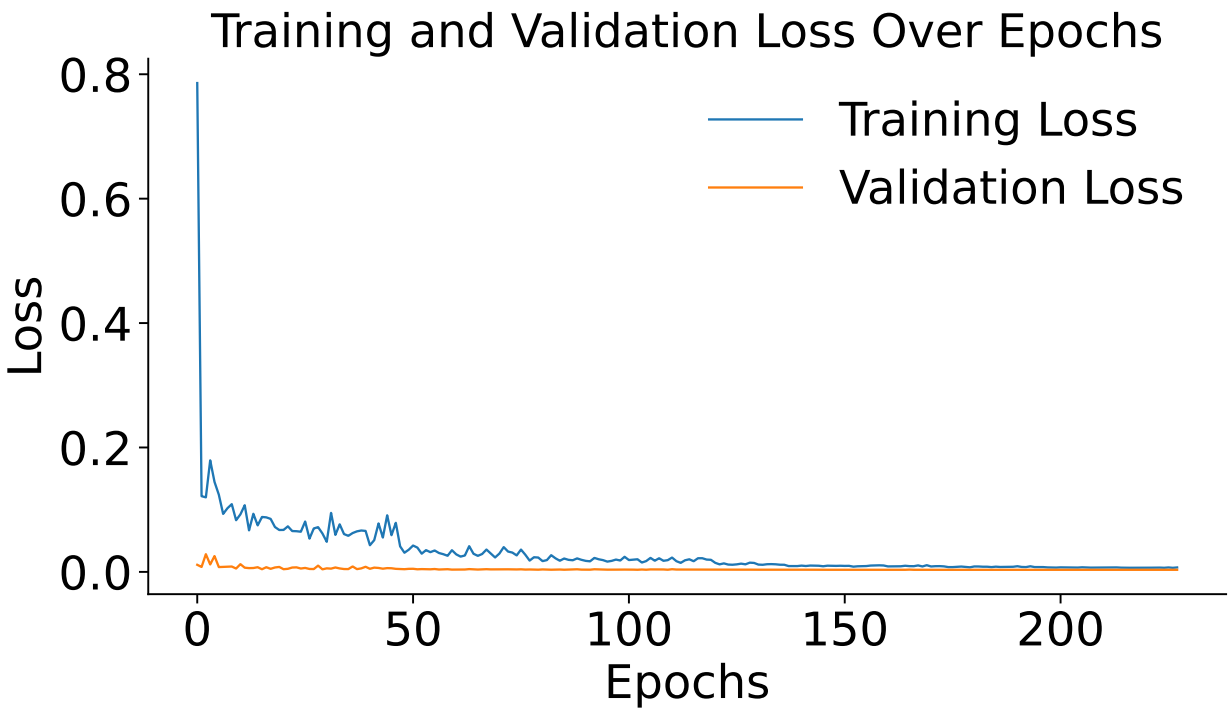

Figure 10: Training and validation loss over all epochs, visualizing the model's learning process and allowing diagnosis of potential overfitting or underfitting.

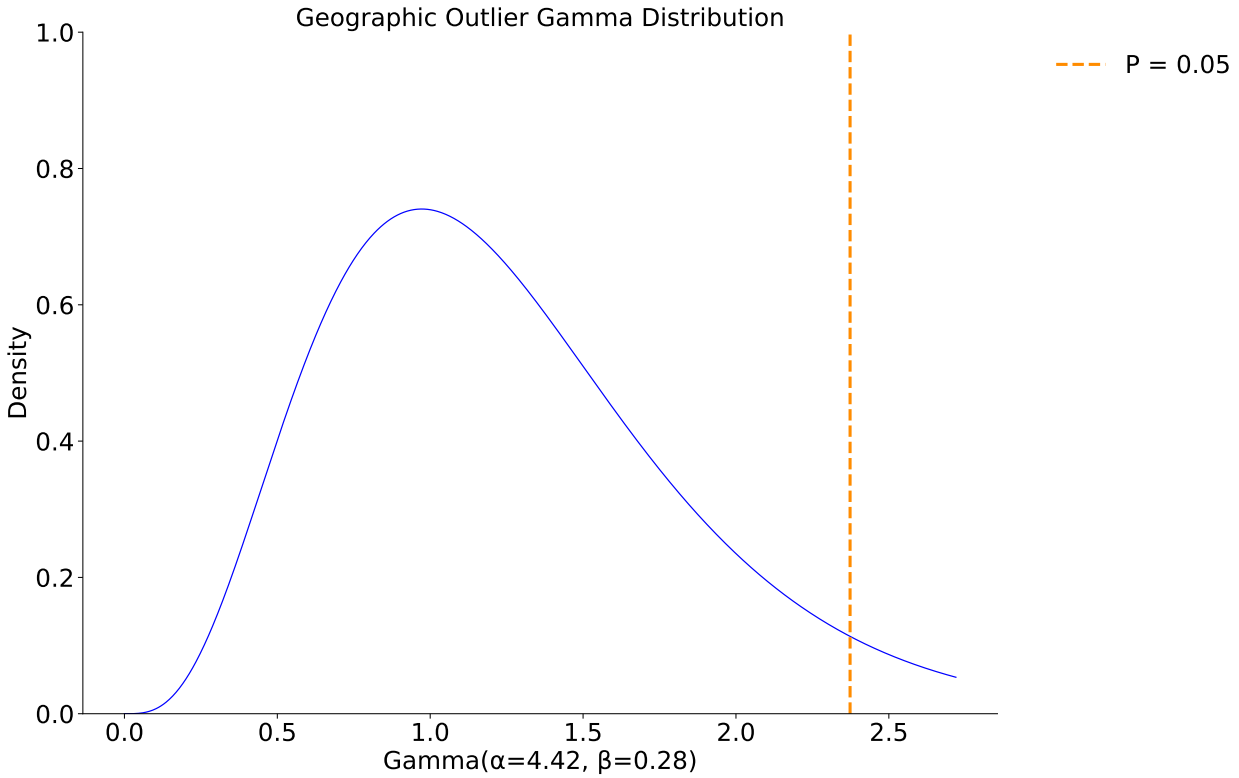

Figure 11: Geographic outlier gamma distribution used to identify the geographic outliers via our outlier removal algorithm adapted from GGOutlieR. The gamma distribution fit allows significant ($P < 0.05$) geographic outliers to be detected and removed from the training dataset.

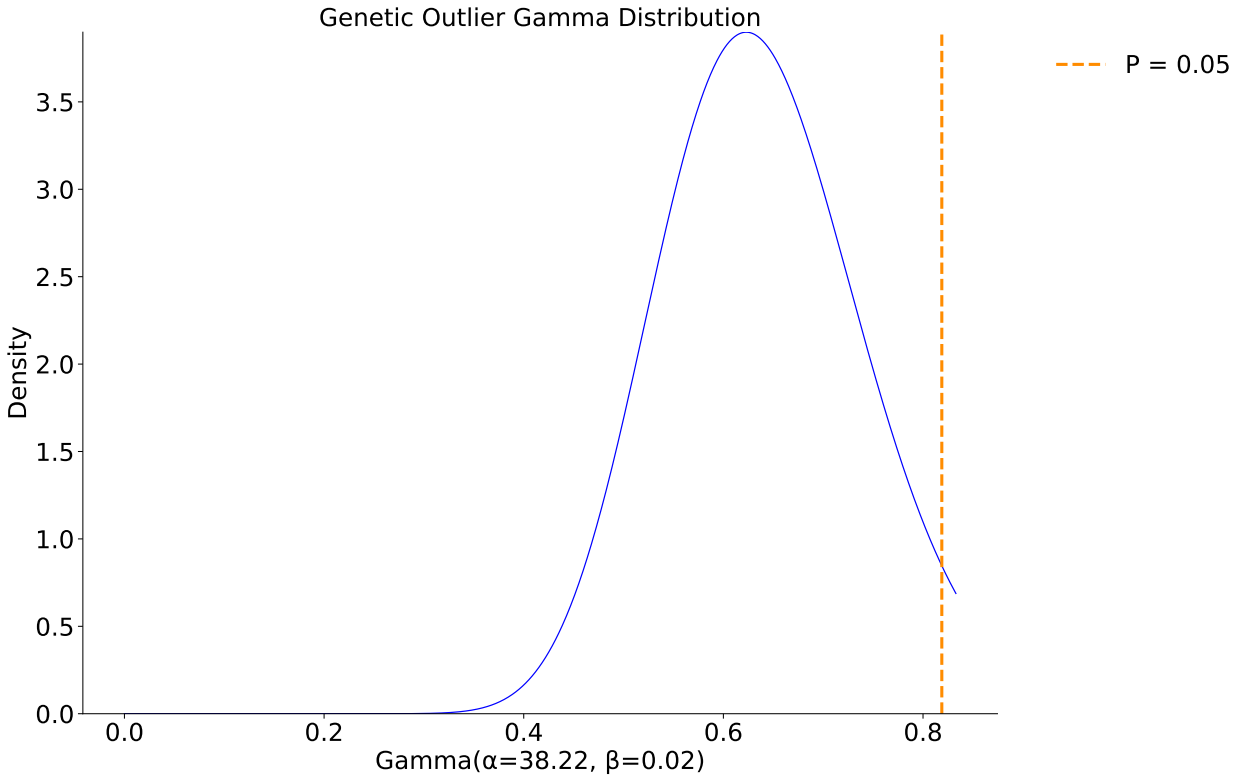

Figure 12: Geneetic outlier gamma distribution used to identify the genetic outliers via our outlier removal algorithm adapted from GGOutlieR. The gamma distribution fit allows significant ($P < 0.05$) genetic outliers to be detected and removed from the training dataset.

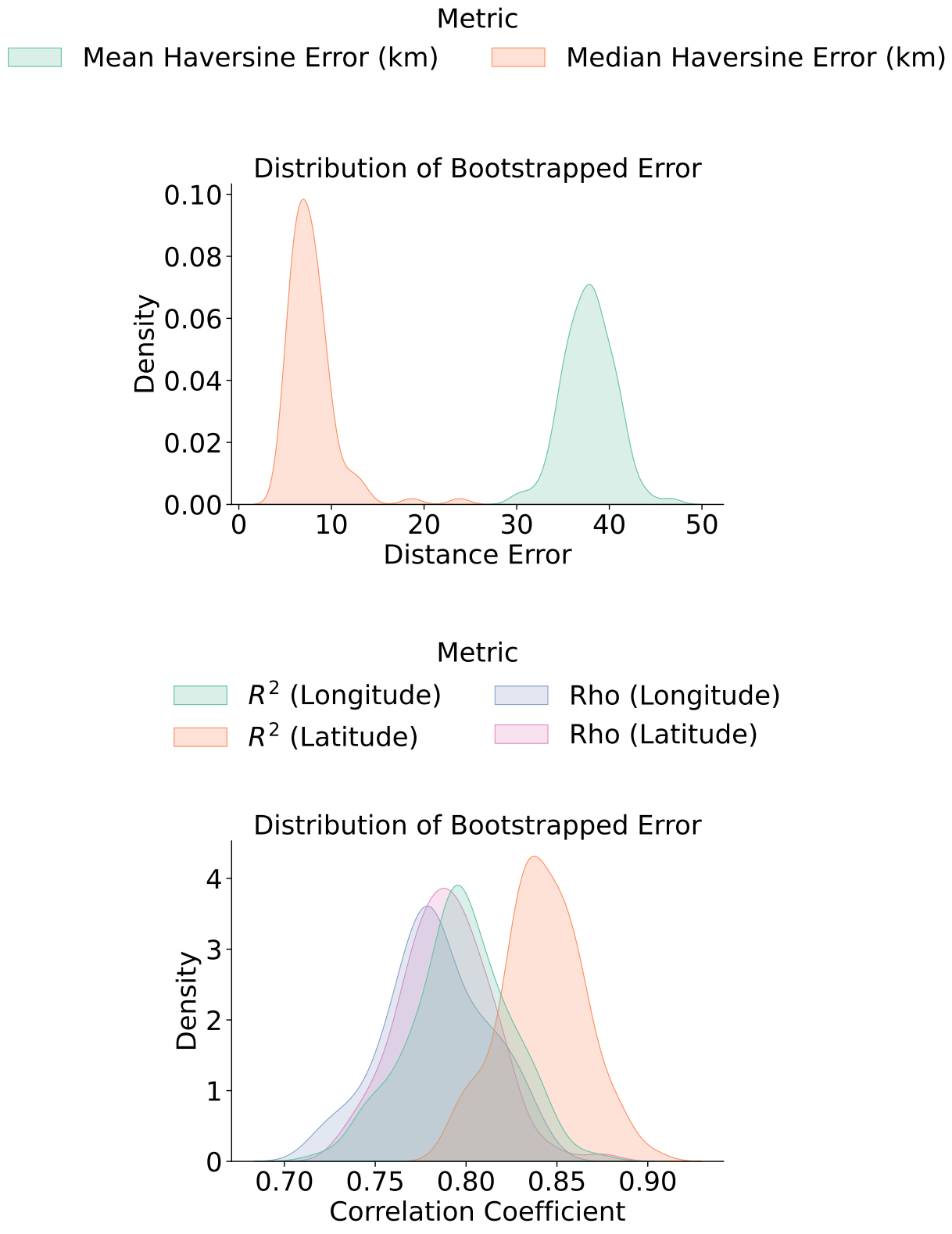

Figure 13: Distribution of prediction errors (i.e., Haversine distance between predicted and recorded localities, in km) across N=100 bootstrap replicates. It visualizes the variability and spread of prediction errors, providing insights into the model's robustness and consistency.

Metric Descriptions

Metric	Description
Root Mean Squared Error (RMSE)	Measures the square root of the average squared differences between predicted and actual values. Lower values indicate better performance.
Mean Absolute Error (MAE)	Calculates the average absolute differences between predicted and actual values. Less sensitive to outliers compared to RMSE.
Huber Loss	Combines RMSE and MAE, balancing sensitivity to outliers and overall accuracy. Useful for datasets with outliers.
Mean Distance (mean_dist)	Measures the average distance between predicted and actual geographic coordinates. Lower values indicate better predictive accuracy.
Median Distance (median_dist)	Represents the middle value of the distance distribution between predicted and actual geographic coordinates. Less sensitive to extreme values.
Standard Deviation of Distance (stdev_dist)	Measures the dispersion of distances between predicted and actual geographic coordinates. Lower values indicate consistency.
Kolmogorov-Smirnov Statistic (kolmogorov_smirnov)	Quantifies the maximum difference between the empirical distributions of predicted and actual distances.
Kolmogorov-Smirnov p-value (kolmogorov_smirnov_pval)	Indicates the statistical significance of the Kolmogorov-Smirnov test. Lower values suggest significant differences in distributions.
Skewness (skewness)	Measures the asymmetry of the distance distribution. Positive values indicate a longer right tail; negative values, a longer left tail.
Spearman's Rank Correlation Coefficient (rho)	Measures the monotonic relationship strength and direction between predicted and actual coordinates. Values close to 1 or -1 indicate strong relationships.
Spearman's Rank Correlation p-value (rho_p)	Assesses the statistical significance of Spearman's rho. Lower values indicate significant relationships.
Spearman Correlation for Longitude	Measures the Spearman correlation between predicted and actual longitude values. Higher values indicate stronger relationships.
Spearman Correlation for Latitude	Measures the Spearman correlation between predicted and actual latitude values. Higher values indicate stronger relationships.

Metric	Description
Spearman p-value for Longitude	Assesses the statistical significance of Spearman correlation for longitude. Lower values indicate significant relationships.
Spearman p-value for Latitude	Assesses the statistical significance of Spearman correlation for latitude. Lower values indicate significant relationships.
Pearson Correlation for Longitude	Measures the Pearson correlation between predicted and actual longitude values. Higher values indicate stronger linear relationships.
Pearson Correlation for Latitude	Measures the Pearson correlation between predicted and actual latitude values. Higher values indicate stronger linear relationships.
Pearson p-value for Longitude	Assesses the statistical significance of Pearson correlation for longitude. Lower values indicate significant linear relationships.
Pearson p-value for Latitude	Assesses the statistical significance of Pearson correlation for latitude. Lower values indicate significant linear relationships.
Mean Absolute Deviation Haversine (mad_haversine)	Calculates the mean absolute deviation using the Haversine formula, accounting for Earth's curvature. Measures average absolute distances.
Coefficient of Variation (coefficient_of_variation)	Ratio of the standard deviation to the mean distance. Standardized measure of distance dispersion.
Interquartile Range (interquartile_range)	Measures the spread of the middle 50% of the distance distribution. Calculated as the difference between the 75th and 25th percentiles.
25th Percentile (percentile_25)	Represents the value below which 25% of distances fall in the distribution.
50th Percentile (percentile_50)	Represents the median value of the distance distribution, indicating the middle distance.
75th Percentile (percentile_75)	Represents the value below which 75% of distances fall in the distribution.
Percent Within 20km (percent_within_20km)	Indicates the percentage of predicted coordinates within 20 km of actual coordinates. Higher values indicate better accuracy.
Percent Within 50km (percent_within_50km)	Indicates the percentage of predicted coordinates within 50 km of actual coordinates. Higher values indicate better accuracy.
Percent Within 75km (percent_within_75km)	Indicates the percentage of predicted coordinates within 75 km of actual coordinates. Higher values indicate better accuracy.

Metric	Description
Mean Absolute Z-Score (mean_absolute_z_score)	Measures the average absolute z-score of distances, providing a standardized measure of distance deviation from the mean.

Glossary

Term	Definition
Activation Function	A mathematical function applied to each neuron in a neural network to introduce non-linearity. Common examples include ReLU, sigmoid, and tanh.
Backpropagation	A training algorithm where the error is propagated backward through the network to update the weights, minimizing the loss function.
Batch Normalization	A technique to normalize inputs to each layer, stabilizing and speeding up the training of deep neural networks.
Bootstrapping	A statistical method that involves resampling a dataset with replacement to estimate variability and create confidence intervals.
Confidence Intervals	A range of values likely to contain the true value of a parameter, providing a measure of uncertainty in the estimate.
Convolutional Neural Network	A type of deep learning model effective for image and spatial data processing using convolutional layers.
Cross-Validation	A technique to evaluate model performance by dividing data into subsets for training and testing in different combinations.
Dropout	A regularization technique that randomly sets a fraction of input units to zero during training to prevent overfitting.
Early Stopping	A regularization method that halts training when the validation performance stops improving, preventing overfitting.
Epoch	One complete pass through the entire training dataset during model training.
Feedforward Neural Network	A simple neural network where connections between nodes do not form a cycle.
Gradient Boosting	A machine learning technique that builds an ensemble of weak models, typically decision trees, to correct errors sequentially.
Haversine Formula	A formula to calculate the distance between two points on a sphere, accounting for Earth's curvature, using latitude and longitude.
Hyperparameter Optimization	The process of tuning hyperparameters like learning rate or number of layers using methods such as grid search or Bayesian optimization.
Imbalanced Sampling	A situation where some classes are overrepresented or underrepresented, leading to biased models.
KMeans Clustering	An algorithm to partition data into K clusters by grouping data points with the nearest mean.
Learning Rate	A hyperparameter that controls how much to update the model weights during training.

Term	Definition
Mean Absolute Error (MAE)	Measures the average absolute differences between predicted and actual values. Less sensitive to outliers than RMSE.
Mendelian Inheritance	Principles of heredity describing the segregation and independent assortment of alleles.
Minor Allele Count (MAC)	The count of the less common allele in a population. A minimum MAC threshold helps filter out rare variants.
Neural Network	A computational model inspired by the human brain, composed of interconnected layers of nodes for tasks like classification.
Optuna	A hyperparameter optimization framework using techniques like Bayesian optimization to efficiently search the parameter space.
Detecting Outliers	The process of identifying and removing data points that deviate significantly from the dataset, improving model accuracy.
Overfitting	A modeling error where the model learns noise or details in training data, reducing performance on unseen data.
Principal Component Analysis (PCA)	A dimensionality reduction technique transforming data into uncorrelated variables called principal components.
Regularization	Techniques like L1 and L2 that add penalties to the loss function to prevent overfitting.
Root Mean Squared Error (RMSE)	Measures the square root of the average squared differences between predicted and actual values. Lower values indicate better performance.
Sampling Density	The concentration of samples in a given area, affecting the balance of the dataset.
SMOTE (Synthetic Minority Over-sampling Technique)	Generates synthetic samples for the minority class by interpolating between existing samples.
Spearman's Rank Correlation Coefficient	A non-parametric measure of monotonic relationships between two variables, ranging from -1 to 1.
Synthetic Oversampling	Generating synthetic data points to balance an imbalanced dataset, improving model performance.
T-SNE (t-distributed Stochastic Neighbor Embedding)	A dimensionality reduction technique for visualizing high-dimensional data.
Underfitting	A modeling error where the model is too simple to capture data structure, resulting in poor performance.

Term	Definition
Validation Split	The portion of the dataset used to evaluate model performance during training to detect overfitting.
Weighted Loss Function	Assigns different weights to samples based on importance, focusing on areas with lower sampling densities.
Xavier Initialization	A weight initialization method ensuring equal variances of input and output, improving convergence speed during training.

References

Akiba, Takuya, Shotaro Sano, Takeru Yanase, Toshihiko Ohta, and Masanori Koyama. 2019. "Optuna: A Next-Generation Hyperparameter Optimization Framework." In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2623–31. ACM.

Battey, C. J., P. L. Ralph, and A. D. Kern. 2020. "Predicting Geographic Location from Genetic Variation with Deep Neural Networks." *eLife* 9: e54507. https://doi.org/10.7554/eLife.54507.

Chang, Author et al. 2023. "GGoutlieR: An r Package to Identify and Visualize Unusual Geo-Genetic Patterns of Biological Samples." *Journal of Open Source Software* 8 (91): 5687. https://doi.org/10.21105/joss.05687.

Lemaître, Guillaume, Fernando Nogueira, and Christos K Aridas. 2017. "Imbalanced-Learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning." *Journal of Machine Learning Research* 18 (17): 1–5.