Supplement of Geosci. Model Dev., 16, 5825–5845, 2023
https://doi.org/10.5194/gmd-16-5825-2023-supplement

Geoscientific Model Development
Open Access
EGU

Supplement of

Ensemble of optimised machine learning algorithms for predicting surface soil moisture content at a global scale

Qianqian Han et al.

Correspondence to: Bob Su (z.su@utwente.nl)

1. Analysis of Feature Importance Across Regression Models

In this section, we present the results of our feature importance analysis across three different regression models: KNR, RFR, and XB. Our aim was to assess the relative contributions of the input variables in predicting the target variable, highlighting the varying importance assigned to each feature by different models. The result is in Figure S1.

In the KNR model, Year, API, and DOY are the most important predictor variables. This suggests that temporal trends, seasonal variations, and accumulated past precipitation have a substantial impact on soil moisture predictions in KNR. In the RFR model, API, lon, Silt, and Sand are the most important predictor variables. This highlights the relevance of accumulated past precipitation, soil properties, geographical information on soil moisture predictions in RFR. In the XB model, API, DOY, lon are the most important predictor variables. This underscores the influence of accumulated past precipitation, seasonal patterns, and geographical location in driving soil moisture predictions within the XB model.

To gain a comprehensive understanding of the relative importance of input variables across different regression models, we computed the average feature importance scores from KNR, RFR and XB. Notably, API plays an important role across all three ML models, underscoring its crucial role in predicting the target variable. This emphasis highlights the vital influence of past precipitation patterns. Furthermore, temporal factors DOY also exhibit substantial importance in the aggregated results, signifying its effectiveness in capturing temporal trends and seasonal variations. Geographical considerations also hold a significant role, as evidenced by the consistent appearance of longitude (lon) in the aggregated analysis. This underscores the impact of geographical location on shaping predictions across diverse models. Soil-related features such as silt content, Sand content and clay content underscore the enduring influence of soil characteristics in the predictive capacity of the models. Moreover, other variables contribute to the collective predictive performance with varying degrees of importance. The aggregated feature importance results solidify the pivotal role of specific variables, such as API, lon, and DOY, in predicting the target variable across different models. The collective contribution of all 19 input variables is evident, as they collectively enhance the predictive capacity of the models. Interestingly, our analysis suggests that the variable precipitation demonstrates relatively lower importance across these three ML models on predicting soil moisture. This intriguing revelation prompts the consideration of future research endeavors aimed at unraveling the nuanced relationship between precipitation and soil moisture.

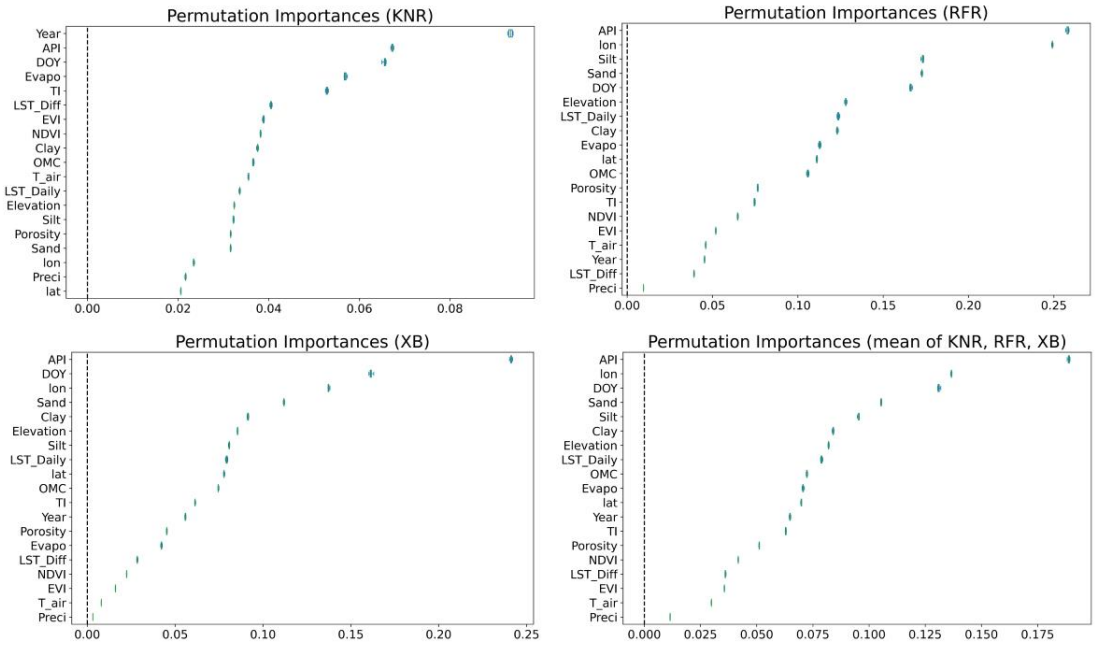

Figure S1. Permutation importances of KNR, RFR, XB and the mean permutation importances of KNR, RFR and XB.

Table S1. Hyperparameters values explored in the tuning process for the optimisation of the considered machine learning algorithms, the identified best combinations and their the r^2 score, and the training time of the optimisation procedure.

ID	ML algorithm	Parameters considered in the optimisation procedure	Combination of parameters that achieved the best performance metrics	r^2 score value achieved by the best combination of parameters (on train set)	Training time of the tuning procedure (mins)
A1	RFR	'n_estimators': [start=10, stop=500, num=10] 'max_features' = ['sqrt', 'log2'] 'max_depth': [5, 6, 7, 9] 'min_samples_split': [2, 3, 4] 'min_samples_leaf': [1, 2, 4]	['n_estimators': 10, 'max_features': 'log2' 'max_depth': None, 'min_samples_split': 4 'min_samples_leaf': 2]	0.8590	30
A2	KNR	'n_neighbors': [3,4,5,6,7] 'weights': ['uniform','distance'] 'p': [1,2] 'leaf_size': [20,30,40] 'algorithm': ['auto','ball_tree']	['n_neighbors': 4, 'weights': 'distance', 'p': 1, 'leaf_size': 20, 'algorithm': 'ball_tree']	0.8848	297

A3	AB	‘estimator’: [DecisionTreeRegressor] ‘estimator__max_depth’: [3, 5, 10] ‘estimator__criterion’: [“squared_error”, “friedman_mse”, “poisson”] ’n_estimators’: [10,20,30,40,50] ’learning_rate’: [0.2,0.4,0.6,0.8,1.0] ’loss’: [’linear’]	[‘estimator’:DecisionTreeRegressor, ‘estimator__max_depth’: 10, ‘estimator__criterion’:“ squared_error”, ’n_estimators’:30, ’learning_rate’:0.2, ’loss’:’linear]	0.6547	50
A4	SGDR	'loss': ['huber', 'epsilon_insensitive', 'squared_epsilon_insensitive'] 'penalty': ['l2', 'l1', 'elasticnet'] 'learning_rate': ['invscaling', 'constant', 'optimal', 'adaptive'] 'average': [False, True] 'warm_start': [False, True] 'alpha': [10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{-5}]	['alpha': 0.01, 'average': True, 'learning_rate': 'constant', 'loss': 'epsilon_insensitive', 'penalty': 'l2', 'warm_start': False]	0.4140	69
A5	MLR	-	-	0.4148	0.0054
A6	MLPR	'activation': ['identity', 'logistic', 'tanh', 'relu'] 'learning_rate': ['adaptive', 'invscaling', 'constant'] 'tol': array([10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}]) 'alpha': array([10^{0}, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}]) 'early_stopping': [True, False] 'hidden_layer_sizes' (number of neurons in the first hidden layer, neurons number in the second hidden layer, and so on): ‘layers’: [1, 2, 3, 4], ‘neurons’: [1, 5, 10, 15, 19, 25, 30, 38, 57, 76]	['activation': 'tanh' 'learning_rate': 'adaptive' 'tol': 10^{-7} 'alpha': 10^{-6} 'early_stopping': False ‘max_iter’: 1000 'hidden_layer_sizes': (30, 30)]	0.7638	75
A7	XB	'n_estimators': [400, 500, 600, 700, 800], 'max_depth': [3, 4, 5, 6, 7, 8, 9, 10], 'min_child_weight': [1, 2, 3, 4, 5, 6],	['n_estimators': 800 'max_depth': 10 'min_child_weight': 1 'gamma':0 'subsample': 0.8 'colsample_bytree': 0.9	0.9139	1521

		'gamma': [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6] 'subsample': [0.6, 0.7, 0.8, 0.9], 'colsample_bytree': [0.6, 0.7, 0.8, 0.9] 'reg_alpha': [0.05, 0.1, 1, 2, 3] 'reg_lambda': [0.05, 0.1, 1, 2, 3] 'learning_rate': [0.01, 0.05, 0.07, 0.1, 0.2]	'reg_alpha':0.05 'reg_lambda':0.1 'learning_rate':0.1]		
A8	GB	'n_estimators': [80, 90, 100, 110, 120] 'max_depth': [1, 2, 3, 4, 5] 'learning_rate': [0.1, 0.3, 0.5]	['n_estimators': 120, 'max_depth': 5, 'learning_rate': 0.5}	0.7977	17

Abbreviations: ML=Machine Learning, RFR=Random Forest Regressor, KNR=K-neighbours Regressor, AB=AdaBoost, SGDR=Stochastic Gradient Descent Regressor, MLR=Multiple Linear Regressor, MLPR=Multi-layer Perceptron Regressor, XB=Extreme Gradient Boosting, GB=GradientBoosting.

Notes: (ID=A1): "n_estimators" is the number of trees in the forest; "max_depth" is the maximum depth of the tree; "min_samples_split" is the minimum number of samples required to split an internal node; while "min_samples_leaf" represents the minimum number of samples required to be at a leaf node. **(ID=A2)**: "n_neighbors" is the number of neighbours to use by default for k-neighbours queries; "weights" is weight function used in prediction; "p" is the power parameter for the Minkowski metric; "leaf_size" is the leaf size passed to BallTree or KDTree, and can affect the speed of the construction and query, as well as the memory required to store the tree; while the "algorithm" represents the algorithm used to compute the nearest neighbours. **(ID=A3)**: "estimator" is the base estimator from which the boosted ensemble is built, if set to "None", then the base estimator is DecisionTreeRegressor; "estimator__max_depth" is the maximum depth of the tree, if set to "None", then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples; "criterion" is the function to measure the quality of a split; "n_estimators" is the maximum number of estimators at which boosting is terminated; "learning_rate" is the weight applied to each regressor at each boosting iteration; while "loss" represents the loss function to use when updating the weights after each boosting iteration. **(ID=A4)**: "loss" is the loss function to be used; "penalty" represents the regularisation term to be used; "average", the averaging method selected when set to "True", it computes the averaged SGD weights across all updates and stores the result in the "coef_attribute"; "warm_start" when set to "True", the algorithm reuses the solution of the previous call to fit as initialization, otherwise, it just erases the previous solution; while "alpha" is the constant that multiplies the regularisation term. **(ID=A6)**: "activation" is the activation function for the hidden layer; "tol" represents the tolerance for the optimisation; "alpha" is the strength of the L2 regularisation term; while "early_stopping" indicates whether to use early stopping to terminate training when validation score is not improving; 'hidden_layer_sizes' is a array-like of shape (,), and the ith

element represents the number of neurons in the ith hidden layer. **(ID=A7)**: "n_estimators" is the number of gradient boosted trees; "max_depth" is the maximum tree depth for base learners; "min_child_weight" represents the minimum sum of instance weight (hessian) needed in a child; "gamma" codifies the minimum loss reduction required to make a further partition on a leaf node of the tree, "subsample" is the subsample ratio of the training instance; "colsample_bytree" is the subsample ratio of columns when constructing each tree; "reg_alpha" is the L1 regularisation term on weights (xgb's alpha); "reg_lambda" is the L2 regularisation term on weights (xgb's lambda); while "learning_rate" represents the boosting learning rate. **(ID=A8)**: "n_estimators" is the number of boosting stages to perform; "max_depth" is the maximum depth of the individual regression estimators; while "learning_rate" codifies the learning rate that shrinks the contribution of each tree by "learning_rate".

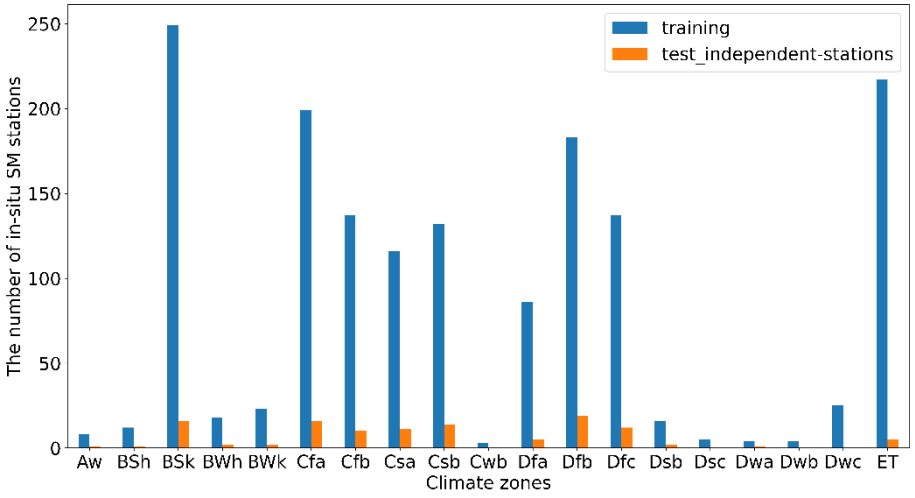

Figure S2. Number of in-situ SSM stations in "train" set and "test_independent-stations" set over each climate zone.

Abbreviations: Aw (Tropical wet and dry or savanna climate), BSh (Hot semi-arid climate), BSk (Cold semi-arid climate), BWh (Hot desert climate), BWk (Cold desert climate), Cfa (Humid subtropical climate), Cfb (Temperate oceanic climate), Csa (Hot-summer Mediterranean climate), Csb (Warm-summer Mediterranean climate), Cwb (Subtropical highland climate), Dfa (Hot-summer humid continental climate), Dfb (Warm-summer humid continental climate), Dfc (Subarctic climate), Dsb (Mediterranean-influenced warm-summer humid continental climate), Dsc (Mediterranean-influenced subarctic climate), Dwa (Monsoon-influenced hot-summer humid continental climate), Dwb (Monsoon-influenced warm-summer humid continental climate), Dwc (Monsoon-influenced subarctic climate), and ET (Tundra climate).

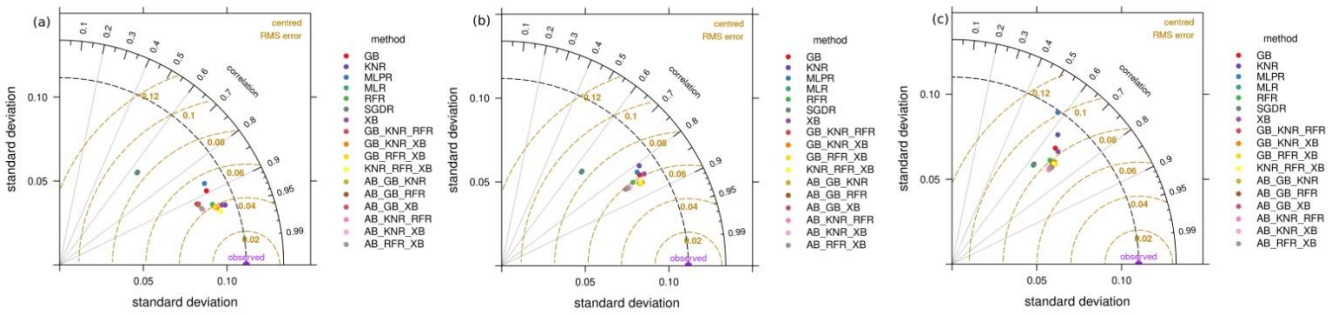

Figure S3. Taylor diagram on (a) "test_random", (b) "test_temporal", and (c) "test_independent-stations"

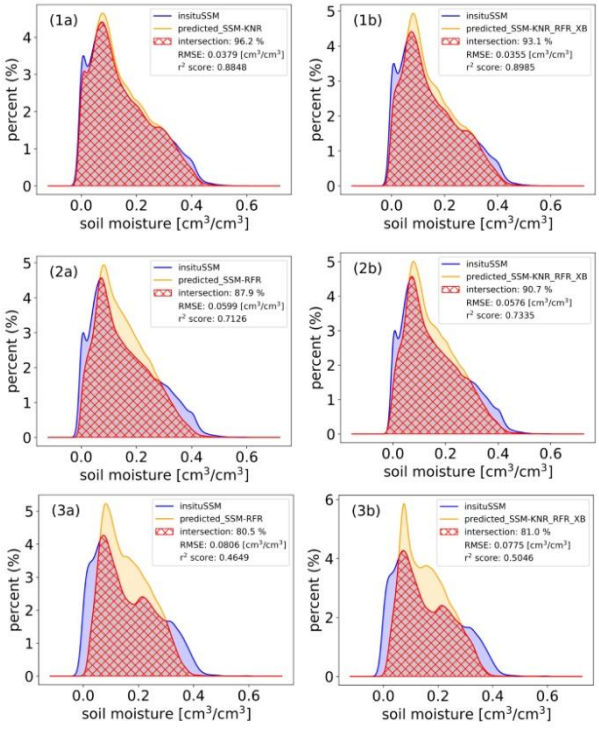

Figure S4. In-situ SSM with predicted SSM from the best single, optimised ML model **(1a)** and best ensemble model **(1b)** on the "test_random" set (KNR, KNR_RFR_XB); in-situ SSM with predicted SSM from the best single, optimised ML **(2a)** and best ensemble algorithm **(2b)** on the "test_temporal" set (RFR, KNR_RFR_XB); in-situ SSM with predicted SSM from the best single, optimised ML **(3a)** and best ensemble algorithm **(3b)** on the "test_independent-stations" set (RFR, KNR_RFR_XB).