Problem Statement

Multiclass classification problem : Classify the data into any one of the 4 classes - built-up, barren, green or water.

Understanding the data

- Generating Basic Summaries using df.describe()
- Checking target distribution
- Checking Missing Values
- Performing detailed Multivariate Analysis
- Checking correlated variables(Spearman Correlation)
- Outliers detection using Box Plot

  For more details check `socialcops/notebooks/01-Exploratory Data Analysis.ipynb`

Preprocessing Involved

- Outlier Removal
- Removing Correlated Features
- Scaling Features using Standard Scalar

  For more details check `socialcops/notebooks/02-Model_Building.ipynb`

Models Evaluation

We have used K-fold cross validation strategy with k = 10 and Micro F1 - Score to evaluate our models. CVMean is the cross validation mean across all 10 folds and cvstd is the standard deviation.

	cv_mean	**cv_std**
Linear SVC	0.90698	0.0281
Gaussian NB	0.8789	0.0575
LDA	0.9055	0.0425
KNN - 5	0.9055	0.0425
Extra Trees	0.9567	0.012
Random Forest	0.9568	0.013

We tried different Machine Learning Algorithms but **Random Forest or in more general tree based models** works best in this dataset. Tree based models are also the best methods to try when you dont know much about the data. It does not need any scaling and also works best with missing data. In next section we will take a look at Random Forest.

Basics of Random Forest

In simple words, random forest is just **bagged decision tree**. In random forest, only a subset of features are selected at random to construct a tree (while often not subsample instances). The benefit is that random forest decorrelates the trees.

There are mainly two algorithms to build trees : one is CART and other is ID3 which uses gini index and entropy as a metrics to build trees .One of the main hyper parameters of random forest is n estimators that is how many trees to be build parallel on different subsample of the same dataset. It predictions can be interpreted using shapley values.

Further Improvement, Code Optimization and Interpretability

- Generate New Features using **Feature Tools** (Automated Feature Engineering)
- Use XGBoost, LightGBM.
- Hyperparameter Optimization.
- **Ensembling Models** usually lead to higher accuracy although not much helpful in production.
- Use **Faster SVM** (Thunder SVM)
- A much better approach would be to train multiple models using different hyperparameters in parallel. Training one model at a time is time taking.
- Use Cython (much faster than python)
- Deep Learning would be an option in structured data if you have a lot of time at hand. Usually that's why Deep Learning fail for structured data a lot. Need a lot of hyperparameter tuning.
- Use **Shapley** (tree based models) and **LIME** to make models more interpretable.

Final Evaluation

We achieved an **Micro F1-score** of 0.9568 using random forest model.