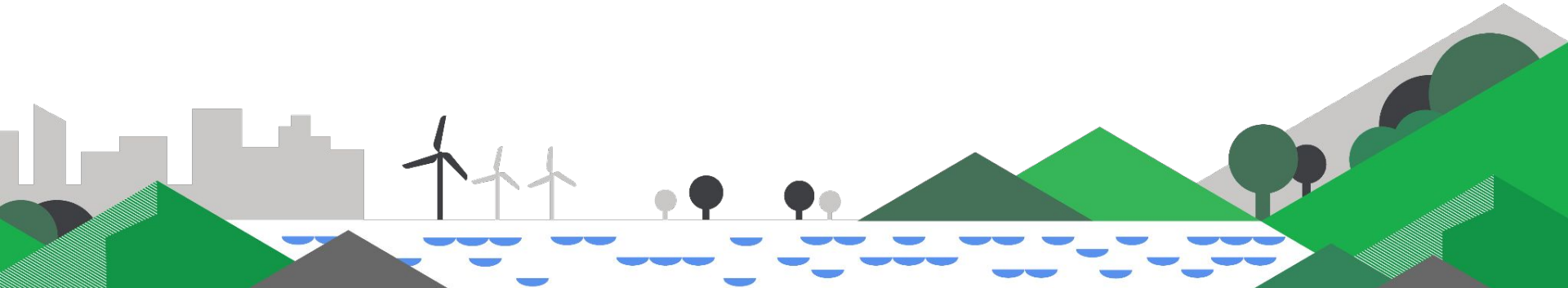
Intro to Machine Learning in Earth Engine

Emily Schechter, Noel Gorelick

Google

October 2022 | #GeoForGood22



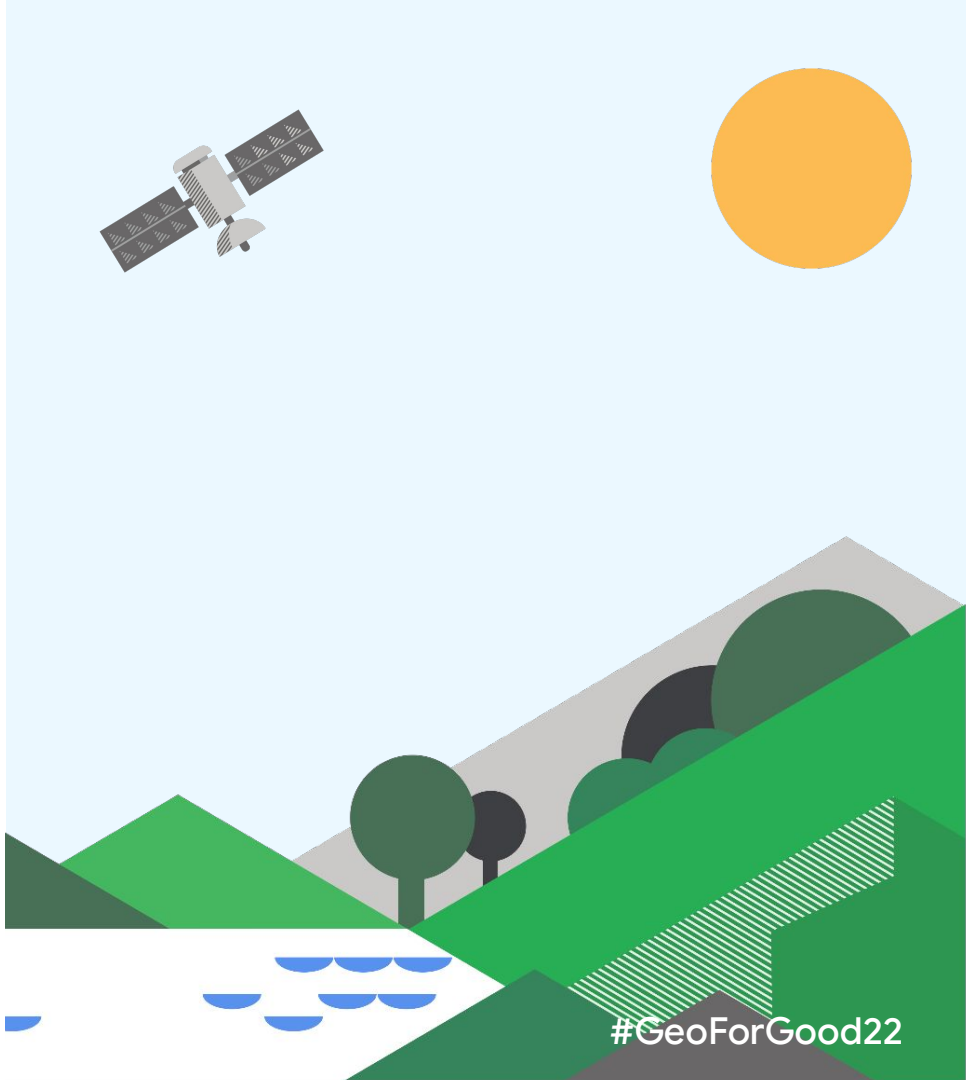
Agenda

01 Introduction to ML

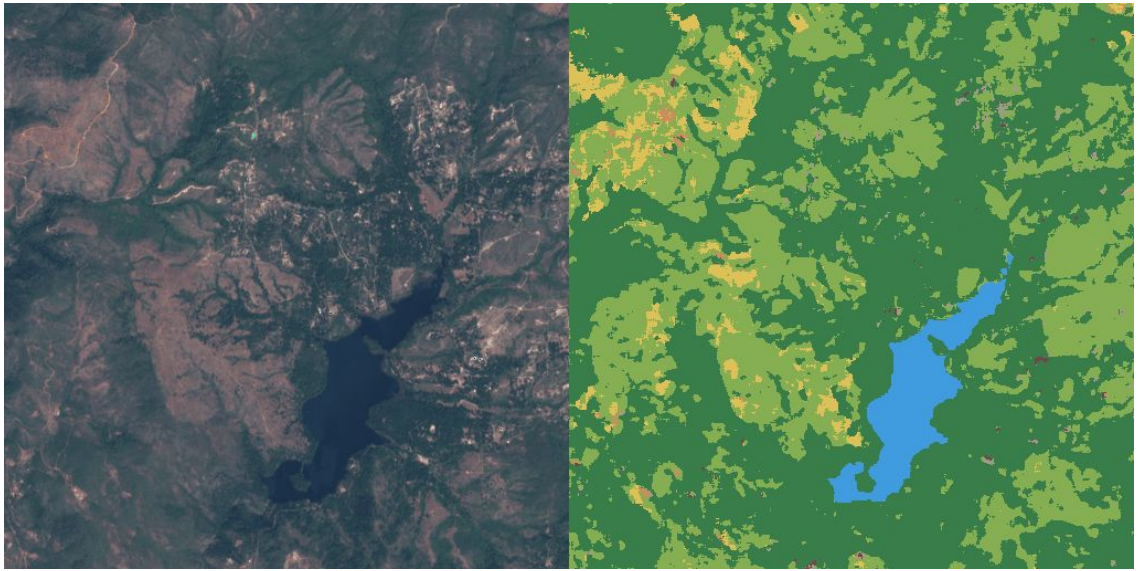
02 Classification in Practice

03 Techniques

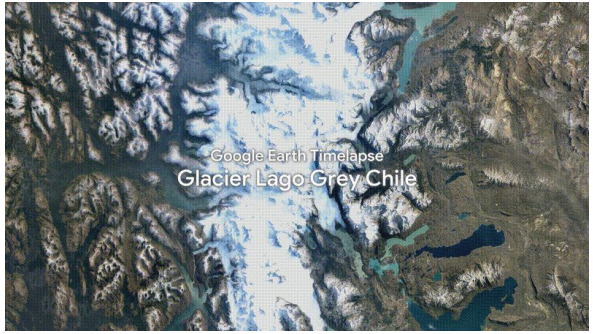
04 Issues and Limitations



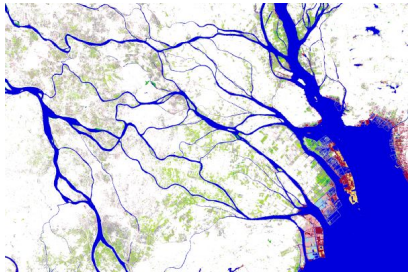
1. Introduction to ML



Deforestation

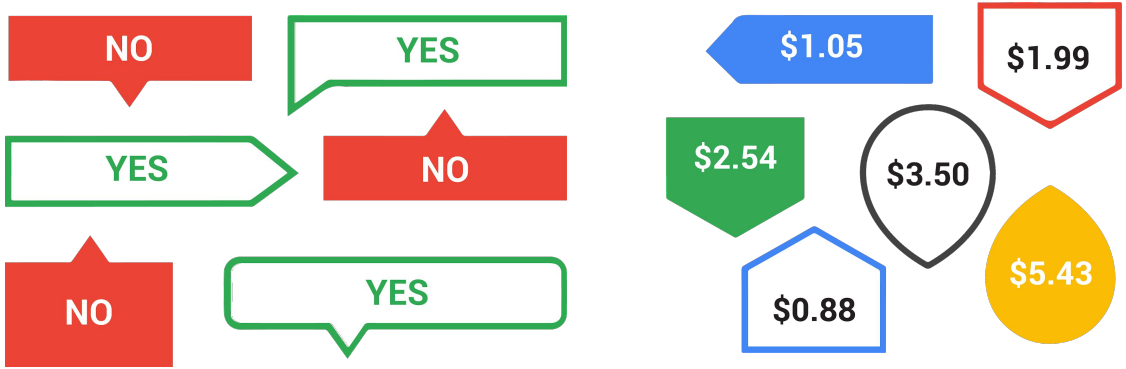


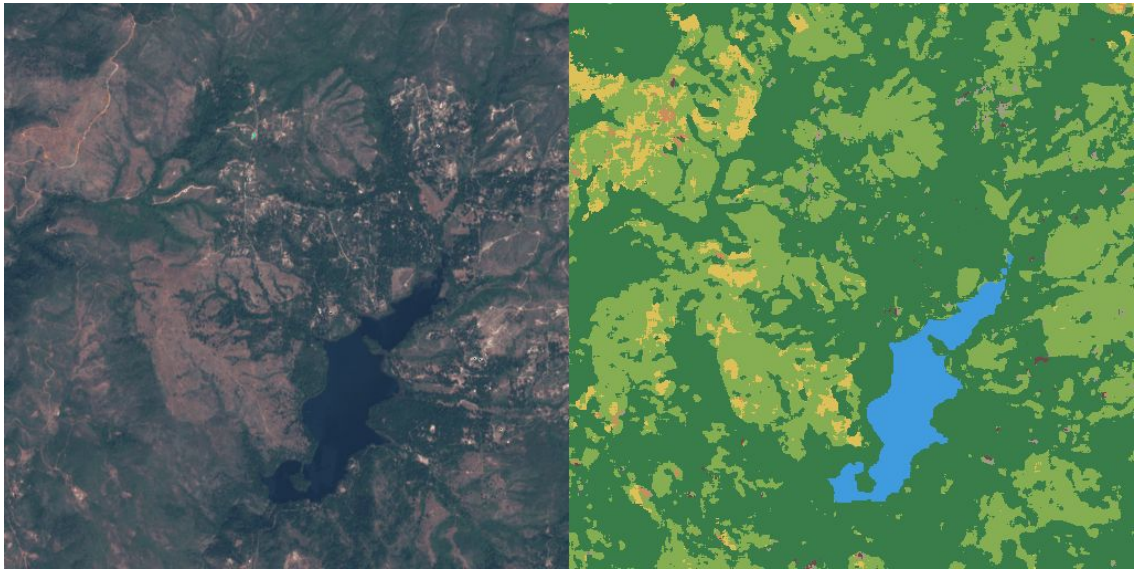
Climate change



Access to water

Machine learning is an approach to making lots of small decisions





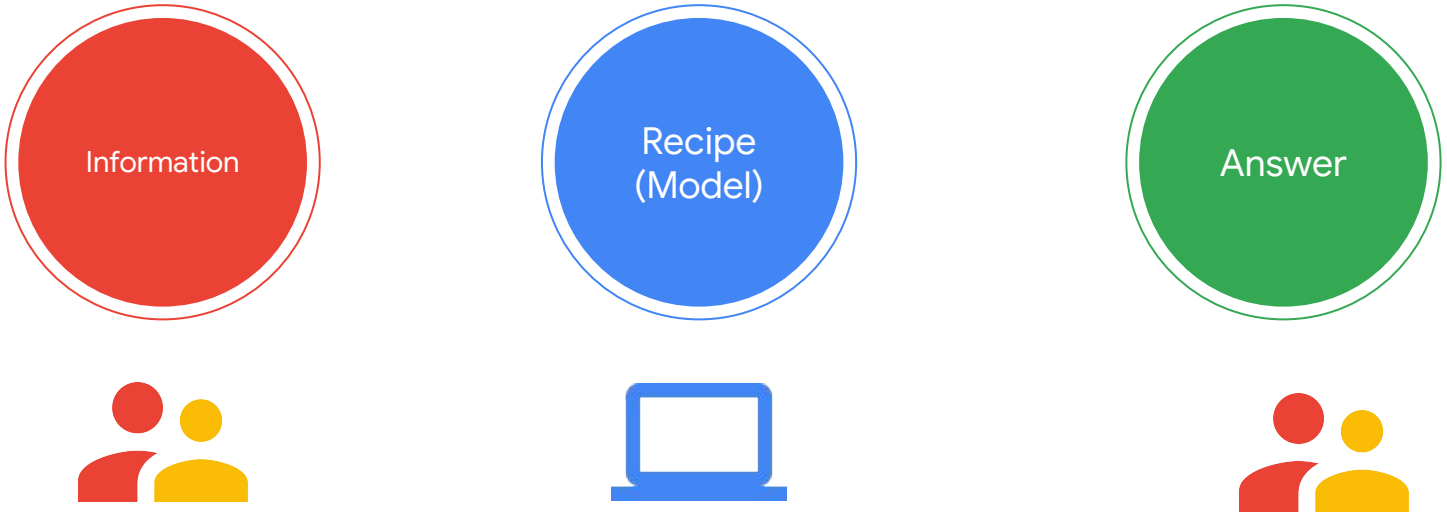

Information


Recipe


Answer

Traditional programming: code the recipe

Machine learning: learn the recipe from data



Training data: examples the system learns from

Features: individual predictors in training data

Model: recipe used to make decisions

Human supervision?

Supervised learning

Training data includes output labels
Classification, regression

Unsupervised learning

Training data is unlabeled
Clustering

Human supervision?

Supervised learning

Training data includes output labels
Classification, regression

Unsupervised learning

Training data is unlabeled
Clustering

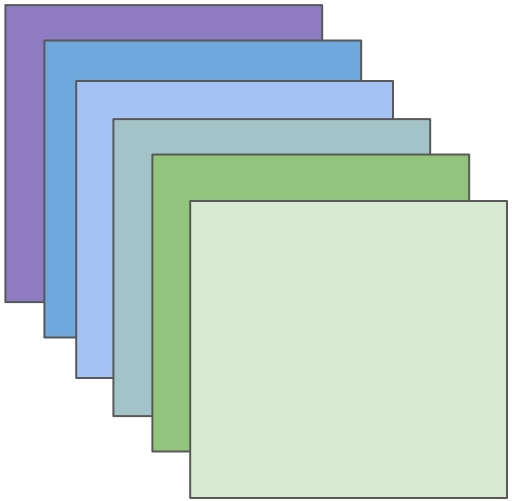
1. Decide on inputs & outputs
2. Get training data
3. Select model
4. Use training data to train model
5. Predict on new data
6. Assess

1. **Decide on inputs & outputs**
2. Get training data
3. Select model
4. Use training data to train model
5. Predict on new data
6. Assess

Input

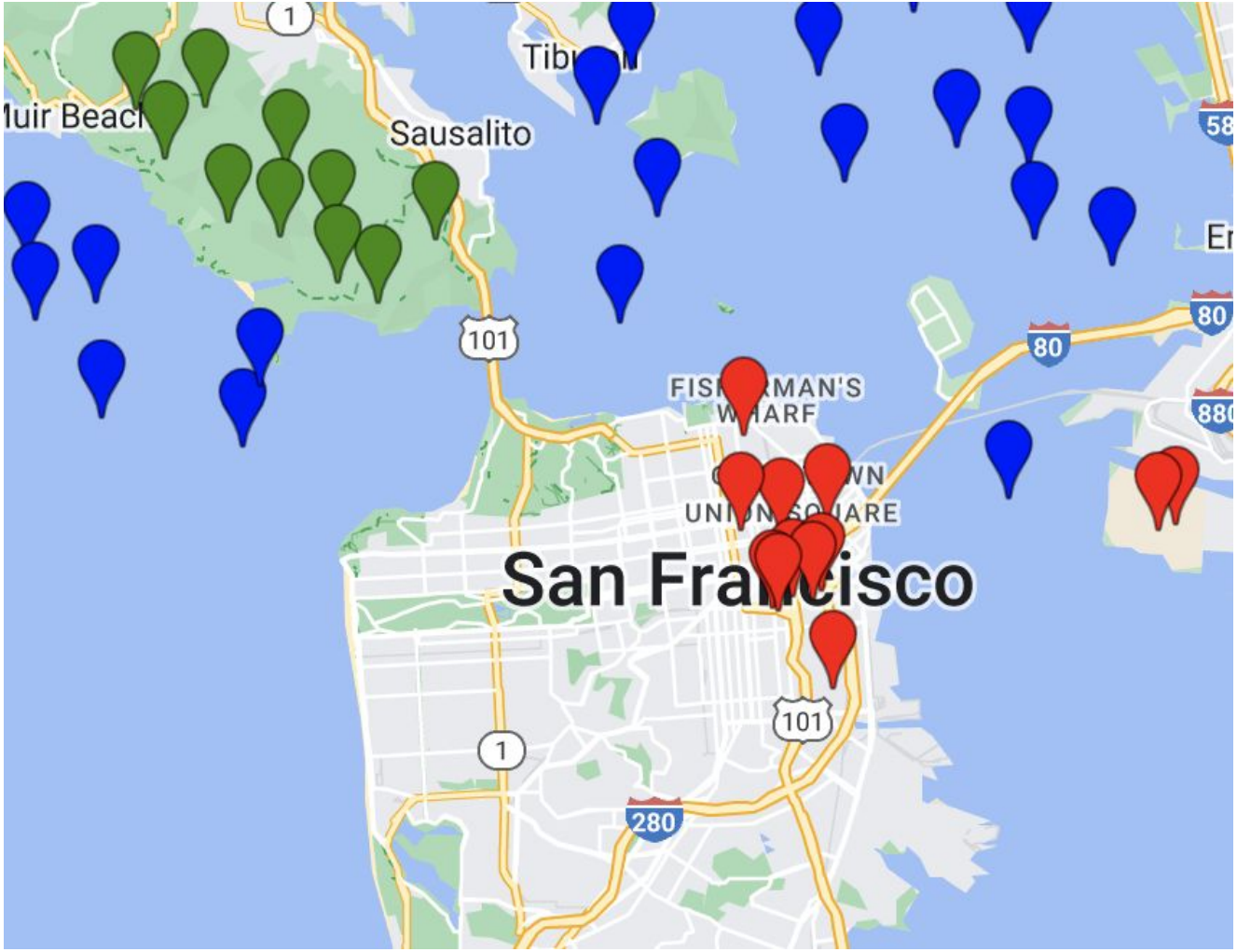
Recipe

Output

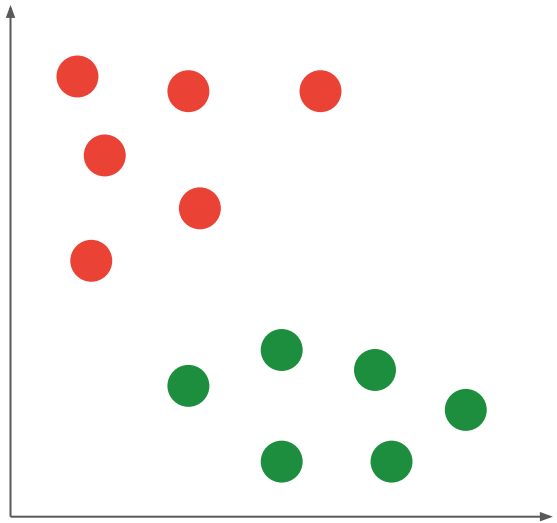


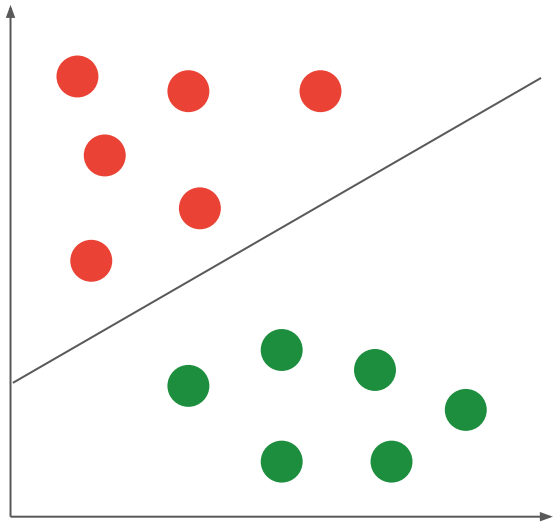
“vegetation”
“water”
“urban”

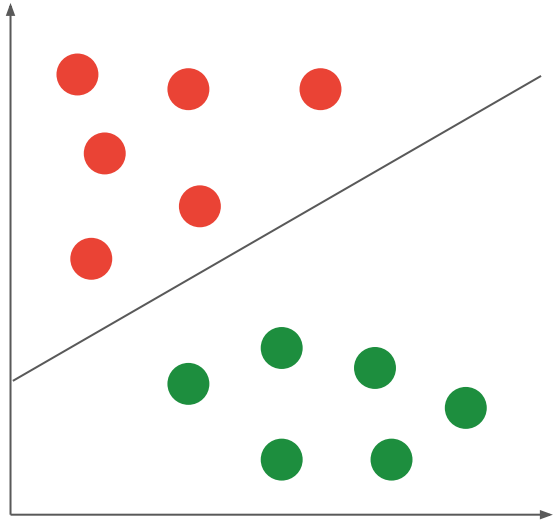
1. Decide on inputs & outputs
2. **Get training data**
3. Select model
4. Use training data to train model
5. Predict on new data
6. Assess


Sausalito
101
San Francisco
1
280
80

1. Decide on inputs & outputs
2. Get training data
3. **Select model**
4. Use training data to train model
5. Predict on new data
6. Assess







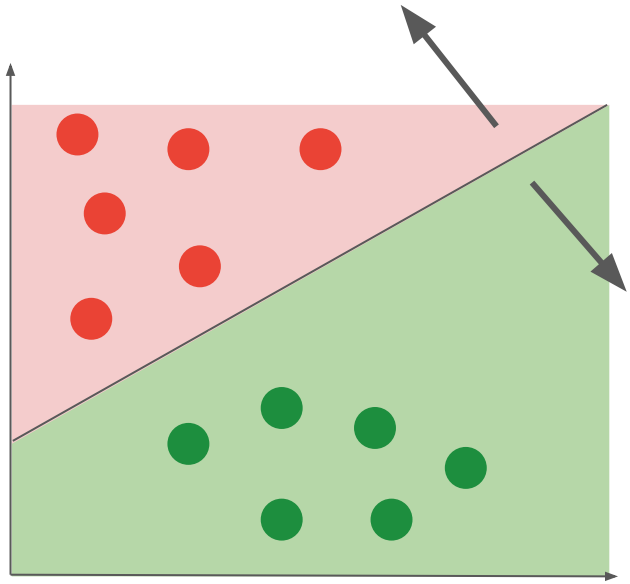
Algorithm selection:

Support Vector Machine

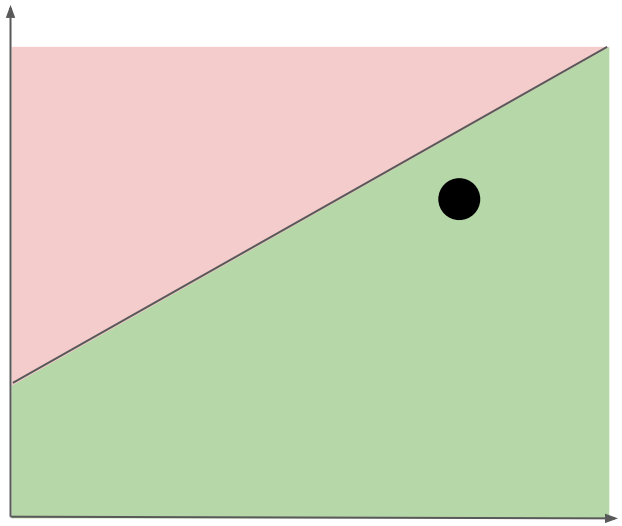
Decision Tree

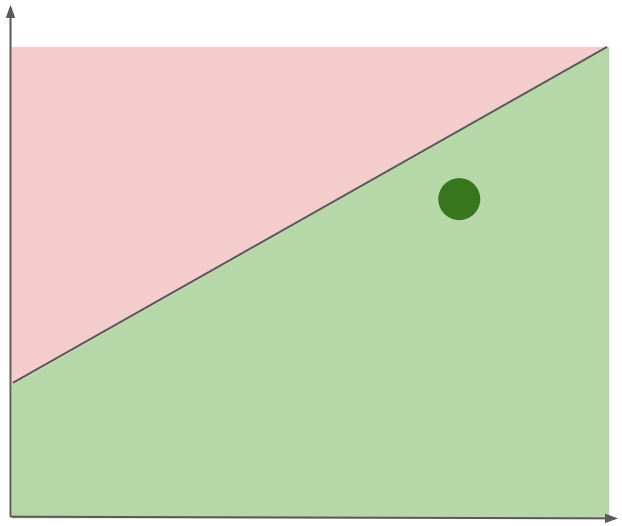
Neural Network

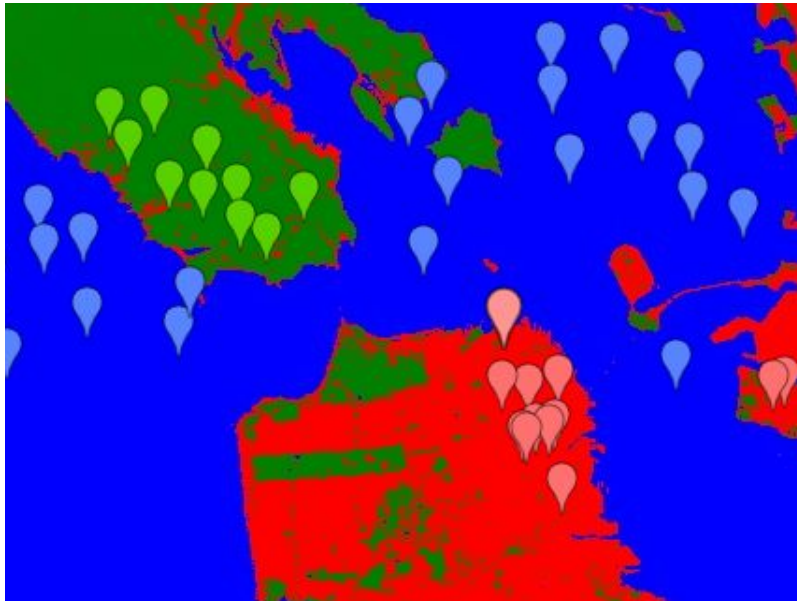
1. Decide on inputs & outputs
2. Get training data
3. Select model
4. **Use training data to train model**
5. Predict on new data
6. Assess



1. Decide on inputs & outputs
2. Get training data
3. Select model
4. Use training data to train model
5. **Predict on new data**
6. Assess







1. Decide on inputs & outputs
2. Get training data
3. Select model
4. Use training data to train model
5. Predict on new data
6. **Assess**

Challenges of machine learning

Training set too small

Need more data

Bad training data

Not representative, noisy, or has irrelevant features

Model simplicity

Overfit or underfit

2. Classification in Practice

Basic Classification with Holdout Validation

```
var points = image.sample(region, scale).randomColumn()

// 30% holdout
var training = points.filter("random < 0.7")
var holdout = points.filter("random >= 0.7")

var classifier = ee.Classifier.smileCart().train(training, "actual")
var classMap = image.classify(classifier, "predicted")

var validation = holdout.classify(classifier)
var errorMatrix = validation.errorMatrix("actual", "predicted")
print(errorMatrix.accuracy())
```

https://code.earthengine.google.com/23c89bb239ad90d38d2978de9477abac

Types of Classifiers

Cart - Classification and regression trees

Gradient Tree Boost - Gradient boosted trees

Maxent - Species distribution modeling

Minimum Distance - (incl. mahalanobis and spectral angle)

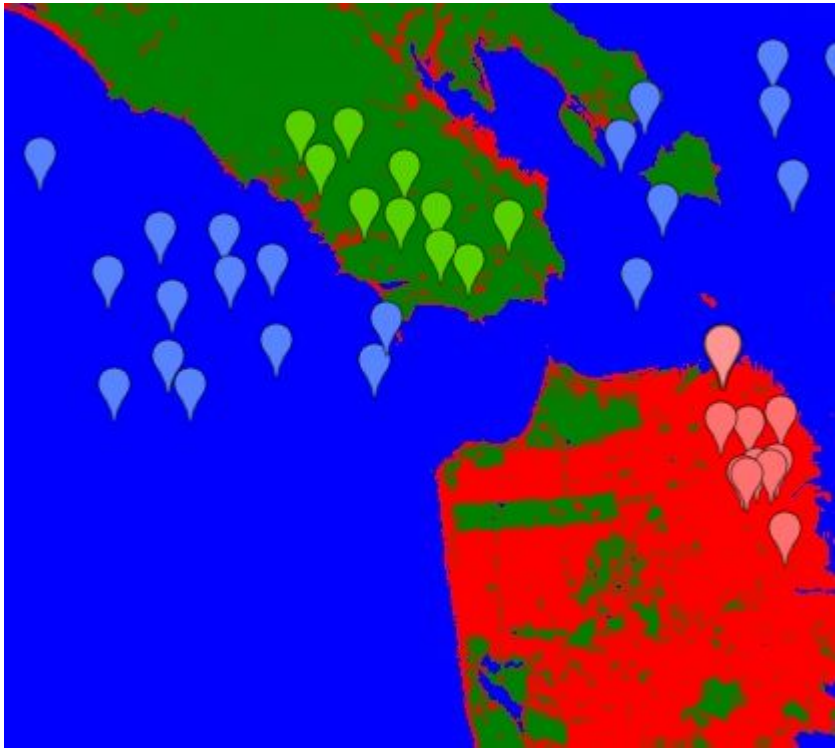
Naive Bayes - Bayesian probability

Random Forest - Random Decision Forest

SVM - Support Vector Machine

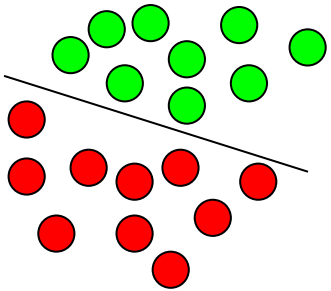
Decision Tree - Create a classifier from saved tree(s)

A Cart Example



```
tree:
n= 78
1) root 78 51.513 2
 2) NIR<=0.0818974 31 0.0000 2 *
 3) NIR>0.0818974 47 23.489 0
  6) GREEN<=0.126542 24 1.9167 1
   12) NIR<=0.156172 1 0.0000 0 *
   13) NIR>0.156172 23 0.0000 1 *
  7) GREEN>0.126542 23 0.0000 0 *
```

Output Modes



Classification

Regression - Continuous output (predicting a value)

Classification - Discrete integer classes (predicting a label)

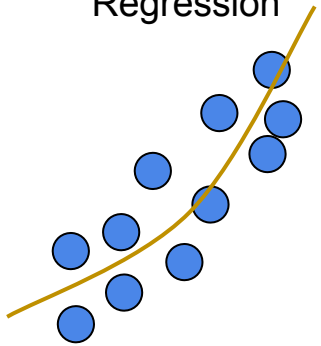
Probability* - The probability that a classification is correct.

Multiprobability - An array of probabilities for each class.

Raw - The internal representation of classification results. For example, votes from each tree in a random forest.

Raw_Regression: An array of the internal representation of regression results. For example, predictions from multiple regression trees.

Regression



Output Modes

	Classification	Regression	Probability	Mulitprobability	Raw	Raw Regression
Cart	✔	✔	✔	✔	✘	✘
Gradient Tree Boost	✔	✔	✔	✔	✘	✘
Maxent	✘	✘	✔	✘	✘	✘
Minimum Distance	✔	✔	✘	✘	✔	✘
Naive Bayes	✔	✘	✔	✔	✘	✘
Random Forest	✔	✔	✔	✔	✔	✔
SVM C_SVC	✔	✘	✔	✔	✘	✘
SVM NU_SVC	✔	✘	✔	✔	✘	✘
SVM One class	✔	✘	✘	✘	✘	✘
SVM Epsilon SVR	✘	✔	✘	✘	✘	✘
SVM Nu SVR	✘	✔	✘	✘	✘	✘

ee.Classifier.minimumDistance()

Finds the minimum distance to the mean of each training class, using a specific distance metric.

- Mahalanobis distance
- Euclidean distance
- Spectral angle

In REGRESSION mode, this classifier will return the distance to the closest class, which can help you find or exclude outliers.

Spectral Angle Distance example: classifies urban, water, agriculture and road.

ee.Classifier.decisionTree()

Creates a decision tree classifier from a R-style tree description.

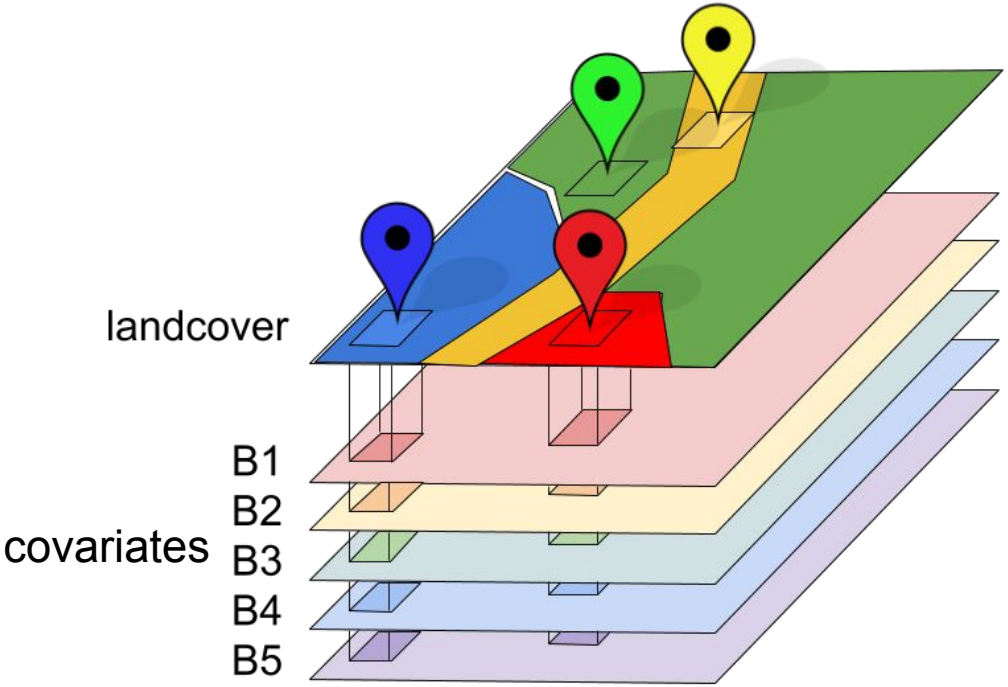
You can load a tree from Google Cloud Storage using ee.Blob().

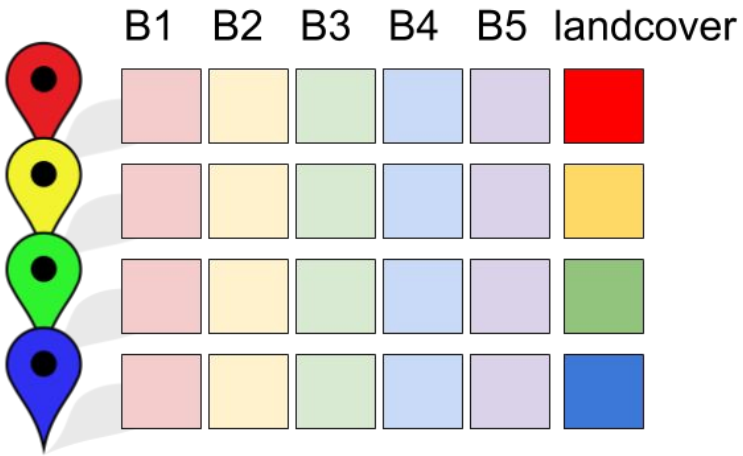
```
var blob = ee.Blob("gs://bucket/decision_tree.txt")
var classifier = ee.Classifier.decisionTree(blob.string())
var classified = image.classify(classifier)
```

https://code.earthengine.google.com/f98cb96e6a0c5f54b16929956ef18713

Sampling and Training

Sampling





Feature Vectors

Sampling

```
var training = image.sample(region, scale)
```

- Exhaustive sampling, but can do subsampling
- Each pixel becomes a feature
- Each band becomes a property
- Discards sparse feature vectors (e.g.: missing `B10`).

```
var training = image.sampleRegions(collection, properties, scale)
```

- Copies properties to each output feature.
- Each band becomes an additional property.
- No random sampling

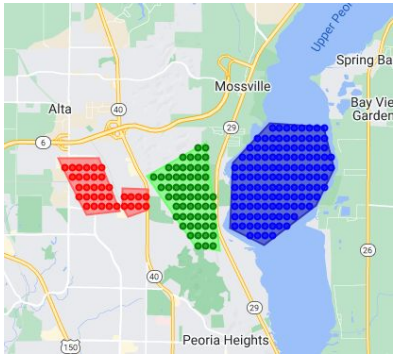
Sampling

```
var training = image.sample(region, scale)
```

- Exhaustive sampling, but can do subsampling
- Each pixel becomes a feature
- Each band becomes a property
- Discards sparse feature vectors (e.g.: missing `B10`).

```
var training = image.sampleRegions(collection, properties, scale)
```

- Copies properties to each output feature.
- Each band becomes an additional property.
- No random sampling



Random Sampling

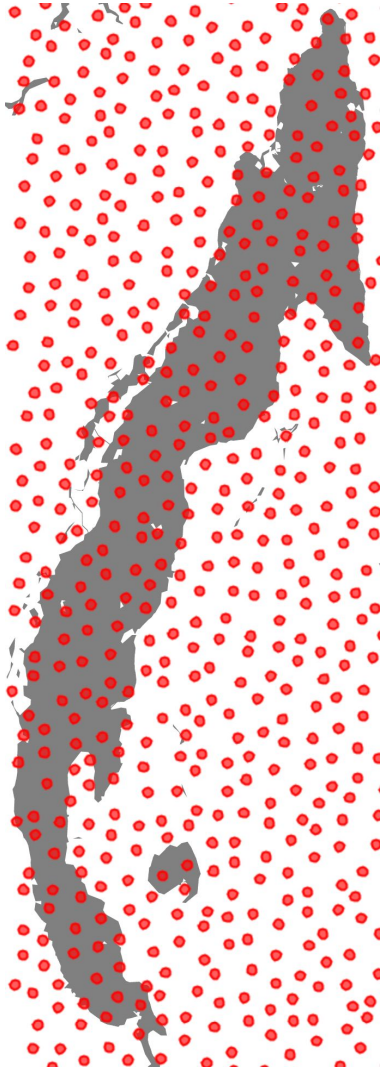
```
var training = image.sample(region, numPixels)
```

same as

```
var pts = ee.FeatureCollection.randomPoints(region, numPixels)
var training = image.sample(pts)
```

Plan B:

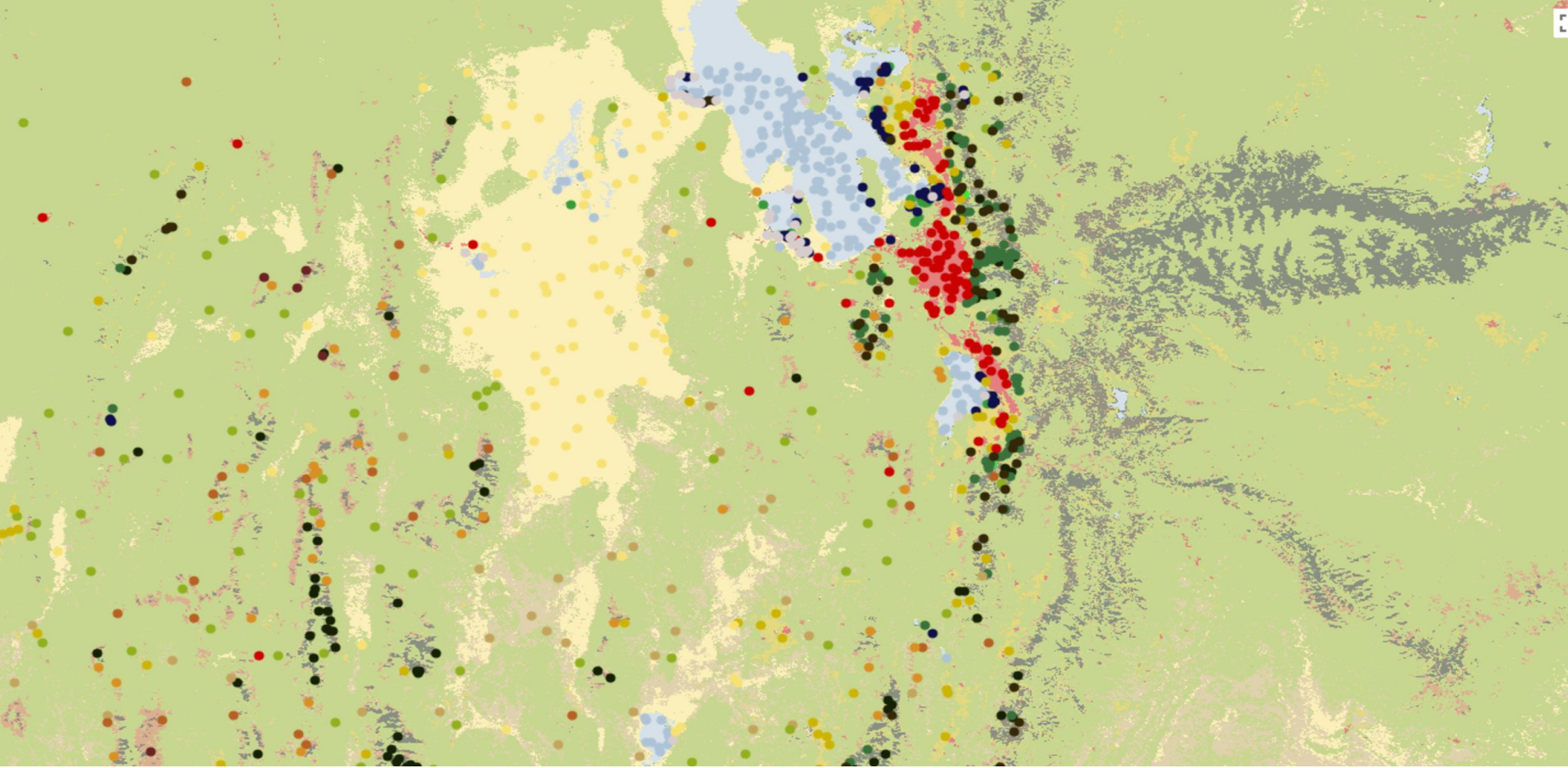
```
var training = image.sample(region)
var subsample = training.randomColumn().filter("random > 0.1")
```



Stratified Random Sampling

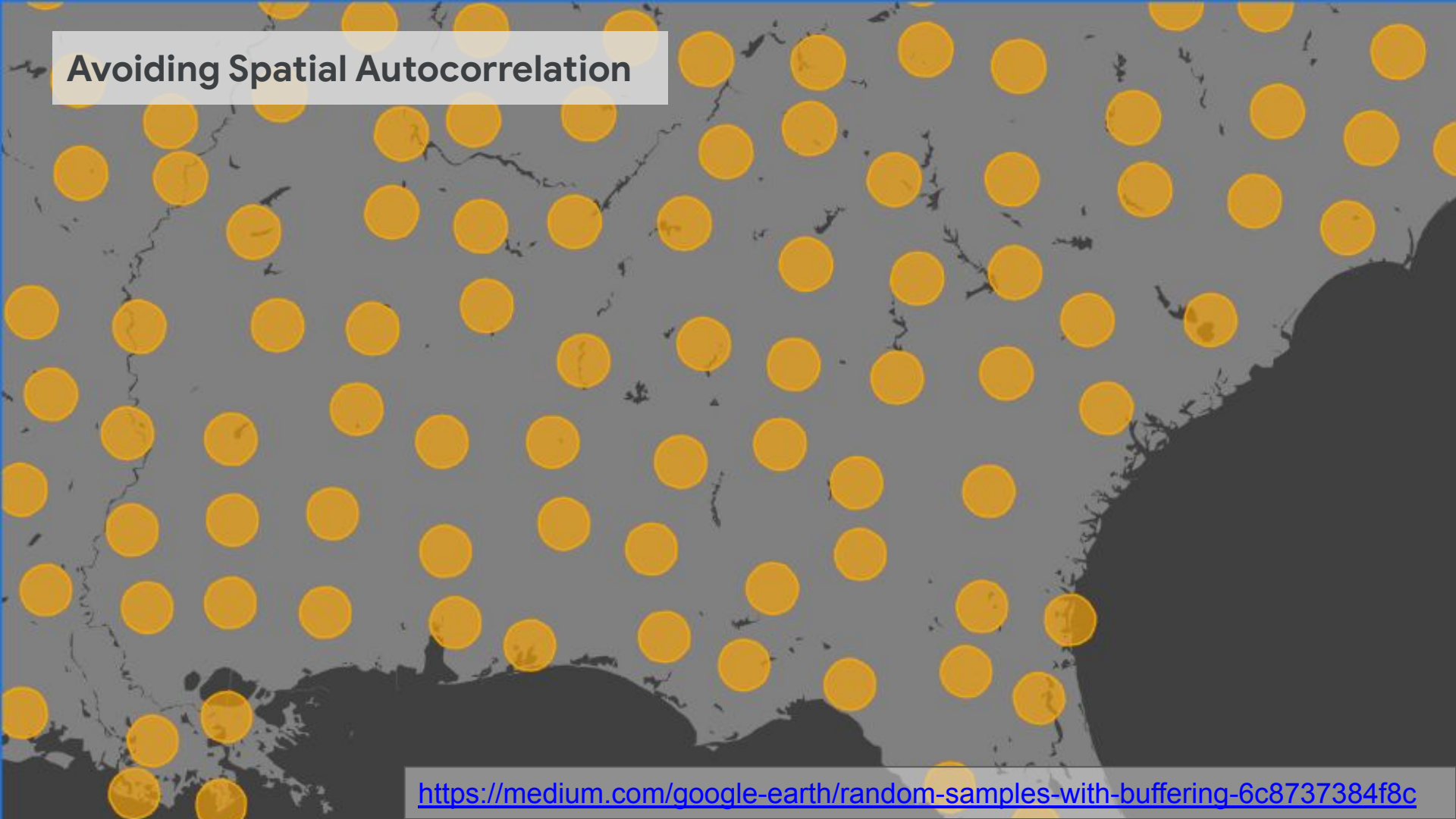
```
var training = image.stratifiedSample({
    classBand: "classes",
    numPoints: 3000,                  // Get 3000 points per class,
    classValues: [3, 5, 8],           // but for these 3 classes,
    classPoints: [500, 1000, 1000]    // get fewer points.
})
```

- Samples up to N points per class (with option to override for individual classes)
- If less than N pixels exist, uses all
- Each band becomes a property
- Discards sparse feature vectors
- Use if you want fine control over number of points or have patchy masks.



https://code.earthengine.google.com/a9ba80f9d412c918f3499261f5d435e7

Avoiding Spatial Autocorrelation



https://medium.com/google-earth/random-samples-with-buffering-6c8737384f8c

Accuracy Assessment

Confusion Matrix

```
matrix = table.errorMatrix("actual", "predicted")
matrix.accuracy()
matrix.consumersAccuracy()
matrix.producersAccuracy()
matrix.kappa()
matrix.fscore()
```

	0	1	2	3	4	5	6	7
0	493	2	2	0	0	3	9	2
1	2	3	0	0	0	4	0	0
2	0	0	2	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	1	1	0	5	0	0	0
5	0	4	6	0	9	0	0	0
6	0	43	55	0	101	0	12	0
7	0	10	12	0	57	0	0	0

Resubstitution validation (don't use this one)

```
matrix = classifier.confusionMatrix()
```

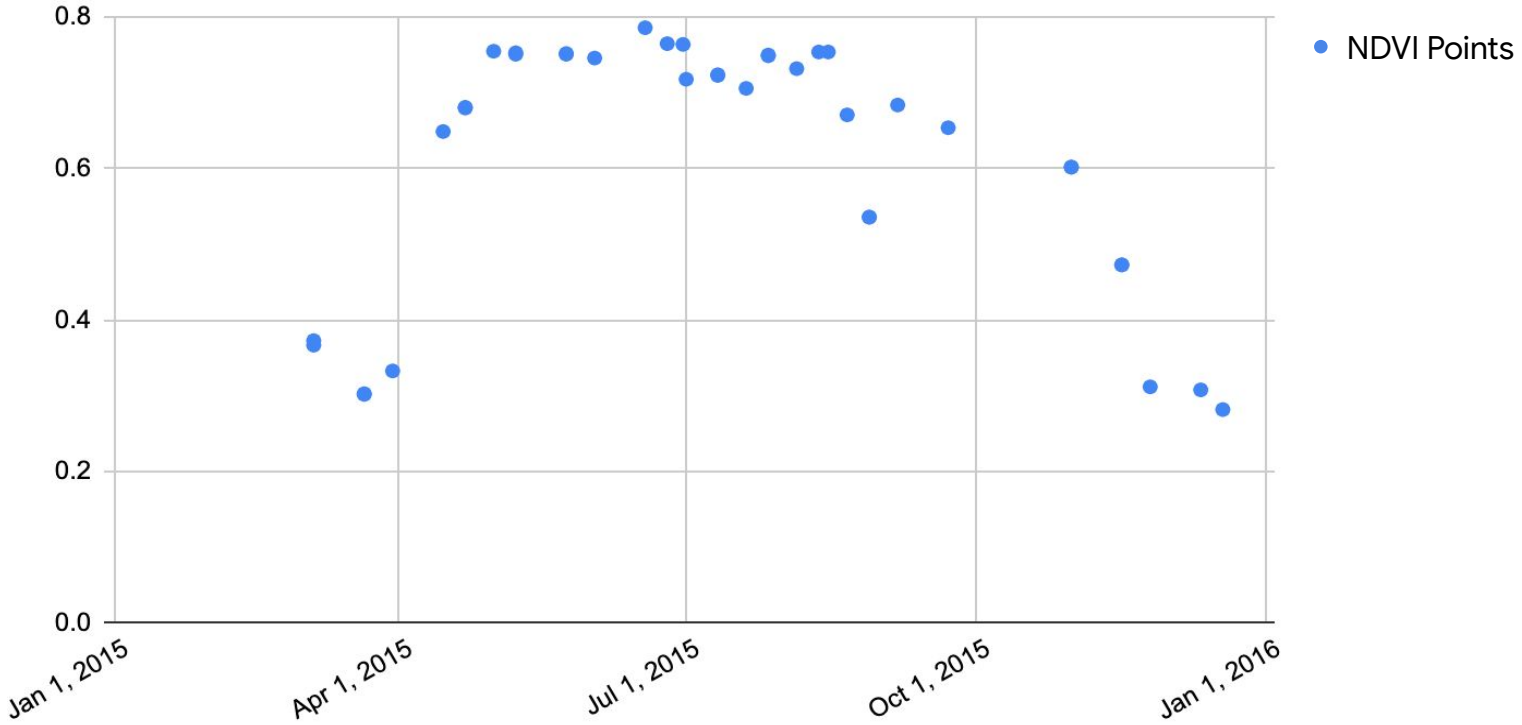
Techniques

1. Temporal Context

2. Spatial Context

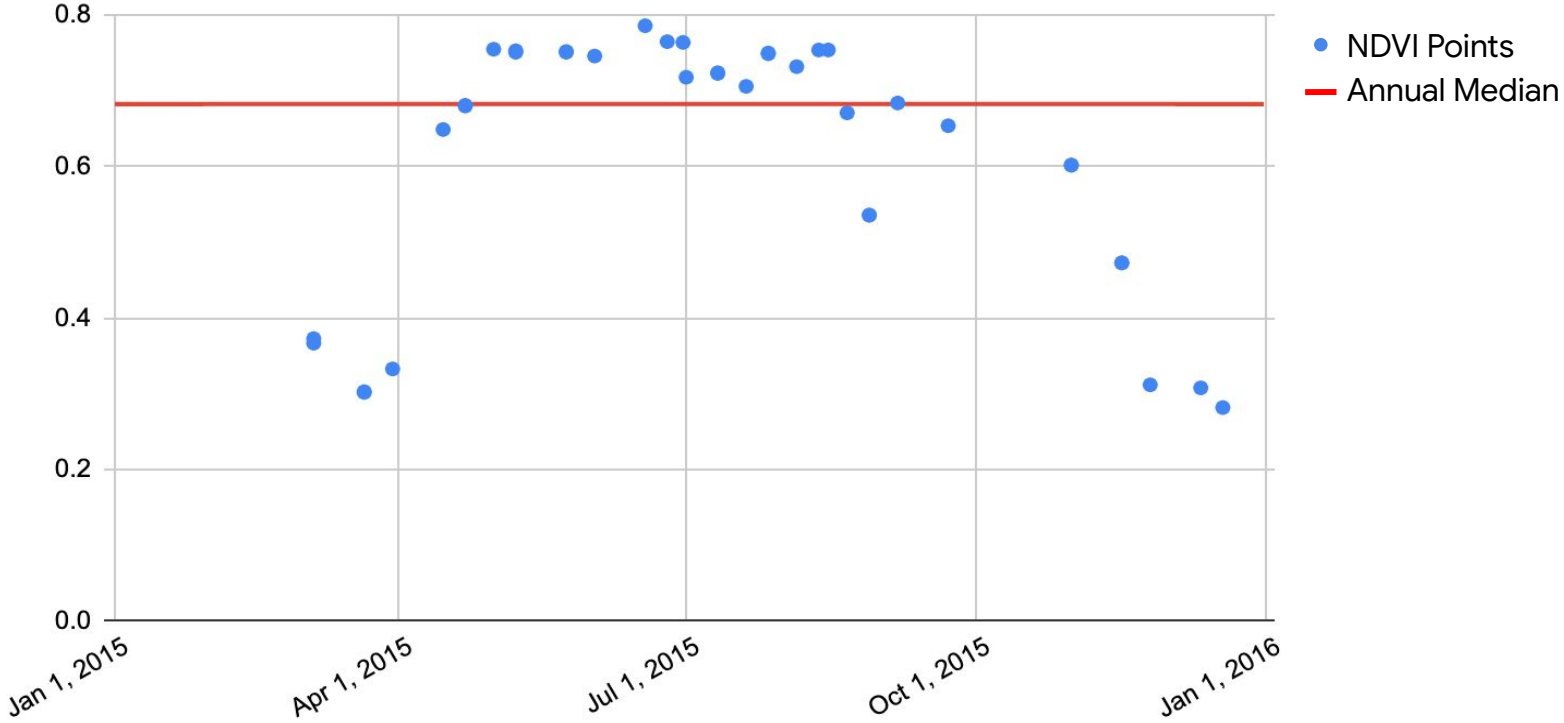
3. Object Based Image Analysis

Temporal Context



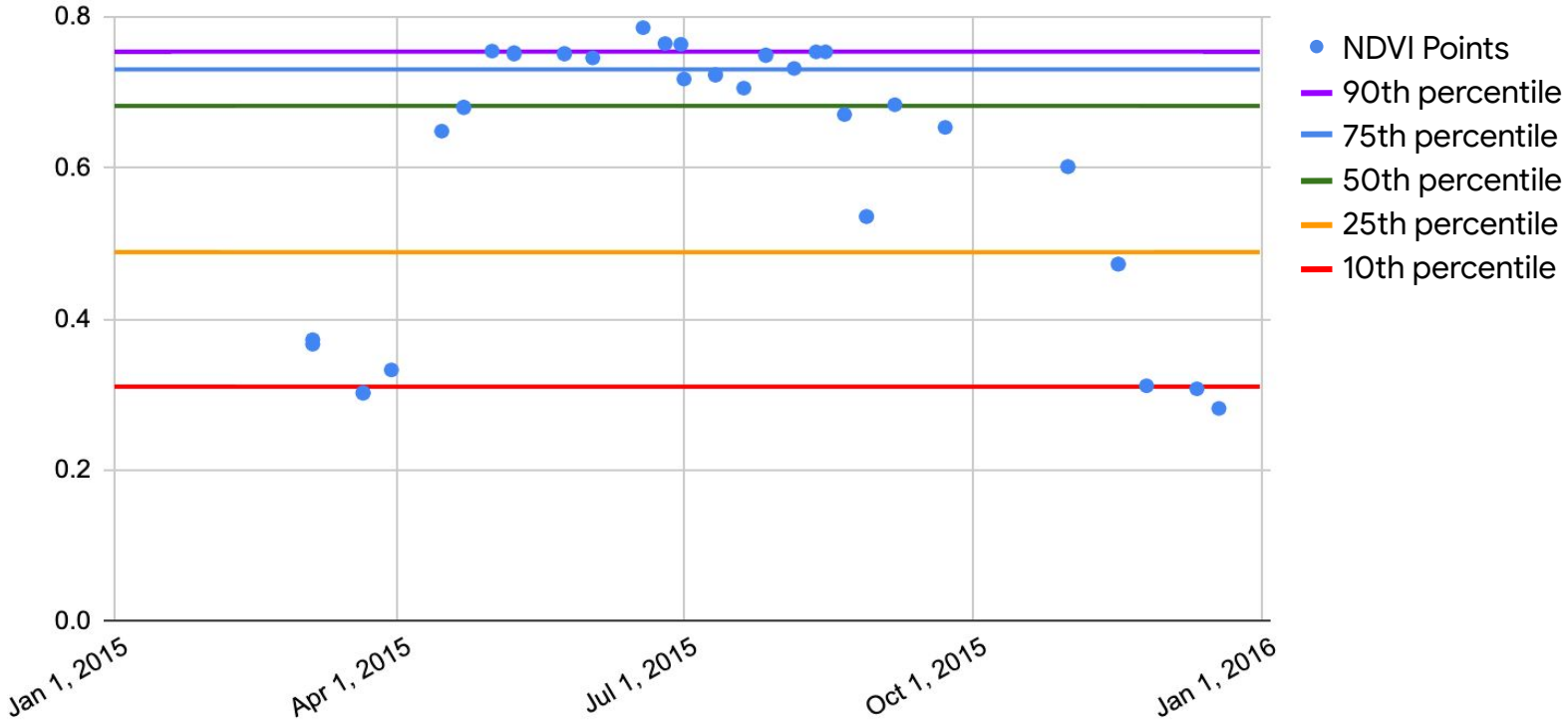
Temporal Context

Annual Median Composite



Temporal Context

Normalizing Statistics Composites

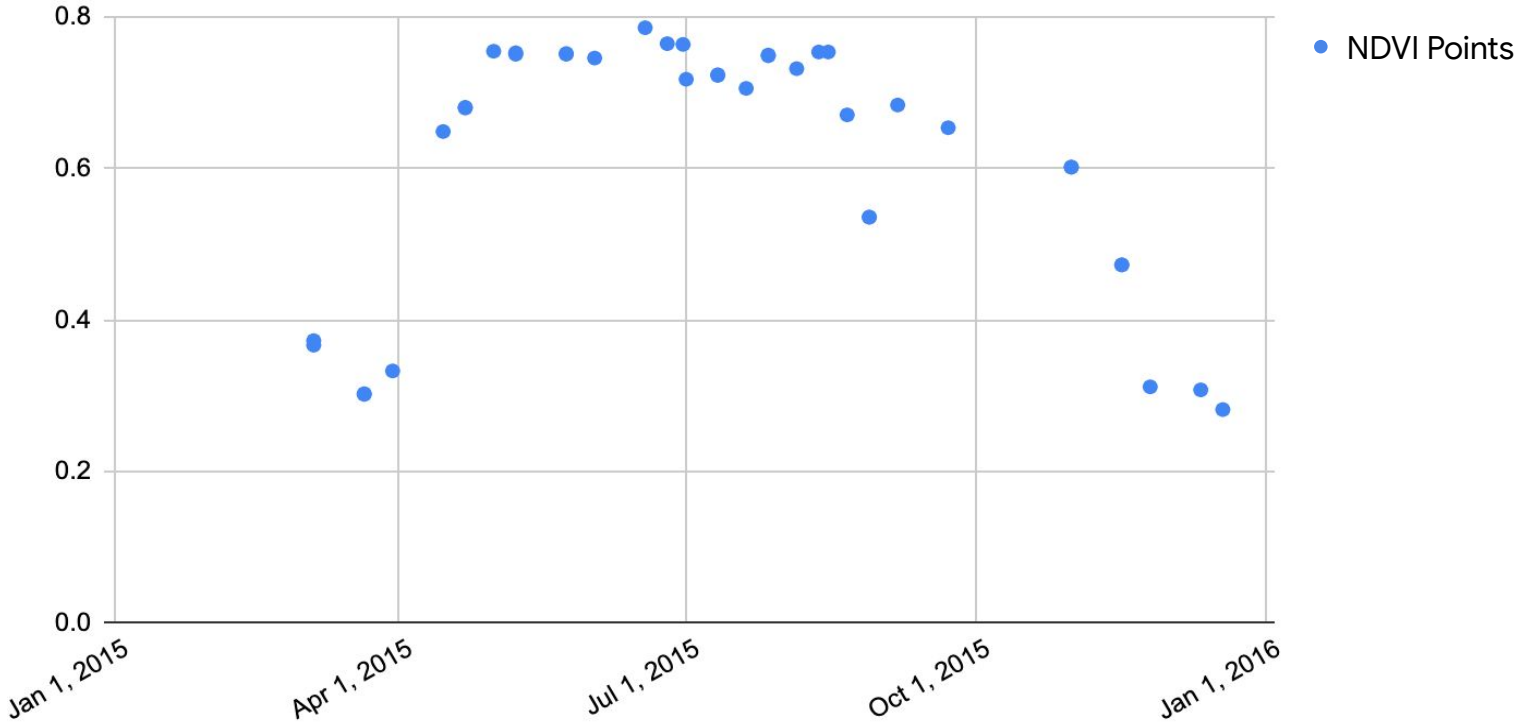


Temporal Context

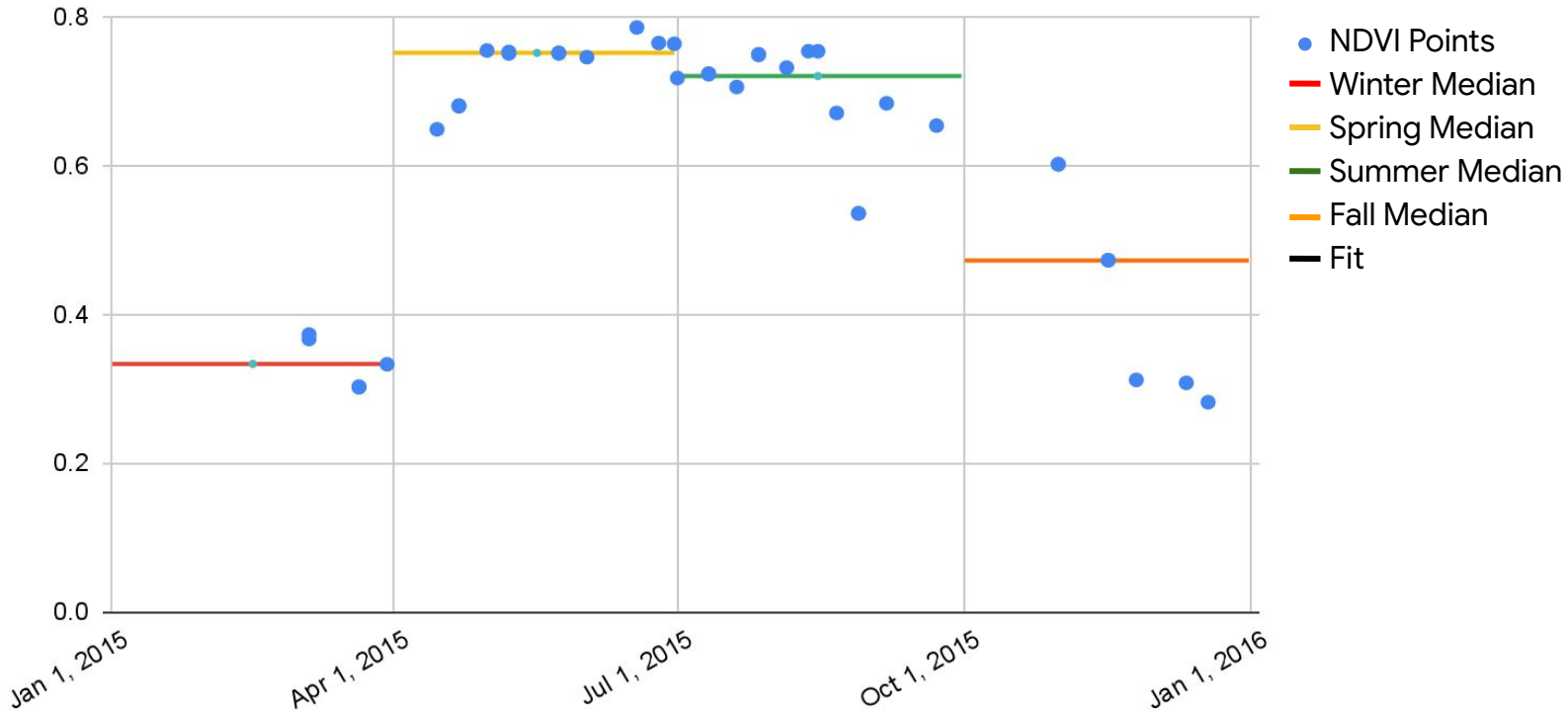
Normalizing Statistics

```
// Normalizing statistics composites
var reducer = ee.Reducer().percentile([10, 25, 50, 75, 90])
var stats = collection.reduce(reducer)
composite = composite.addBands(stats)
```

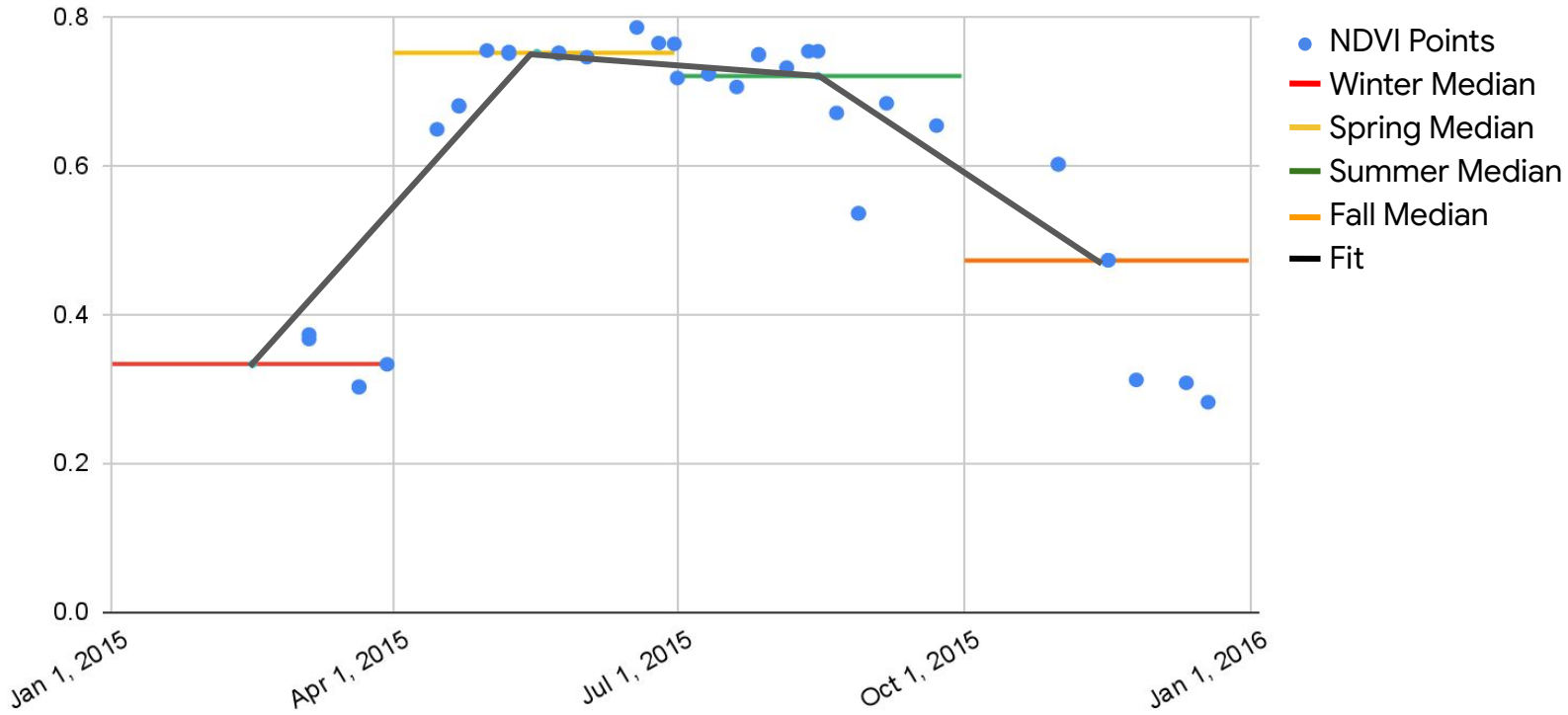
Temporal Context



Temporal Context



Temporal Context



Temporal Context

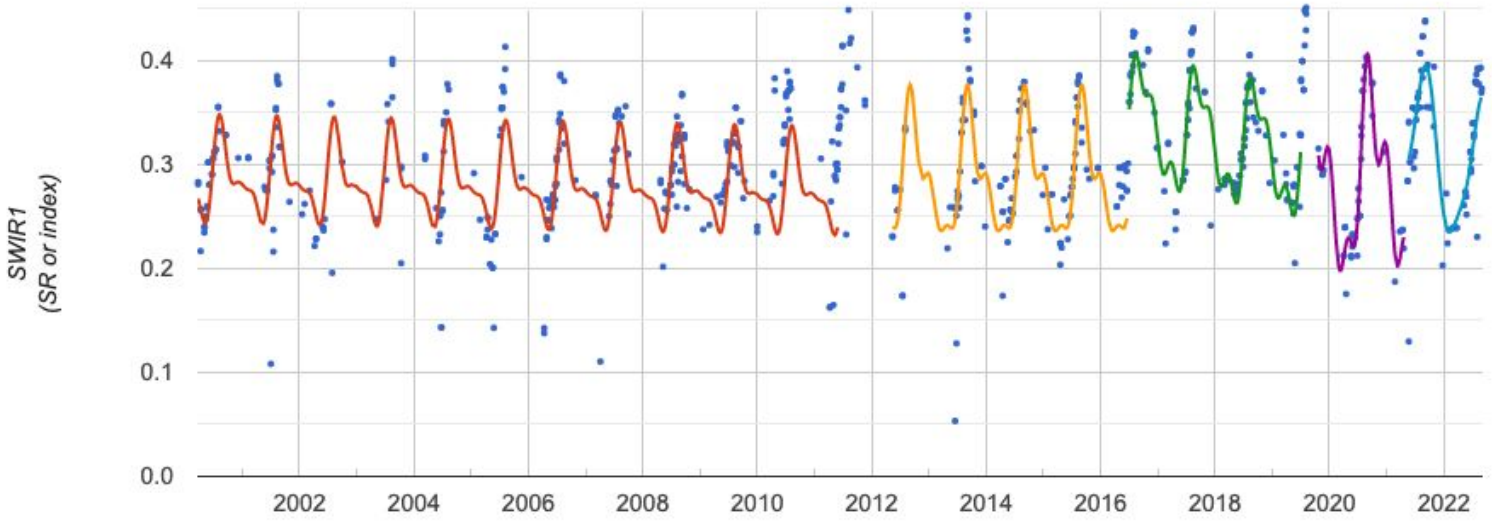
Seasonal Mosaics and Temporal Statistics

```
function seasonalComposite(start) {
  return collection
    .filter(ee.Filter.calendarRange(start, start.add(2), "month"))
    .median()
}

// Make mosaics from months 1-3, 4-6, 7-9 and 10-12
var seasons = ee.List([1, 4, 7, 10]).map(seasonalComposite)
var composite = ee.ImageCollection(seasons).toBands()
```

Temporal Context

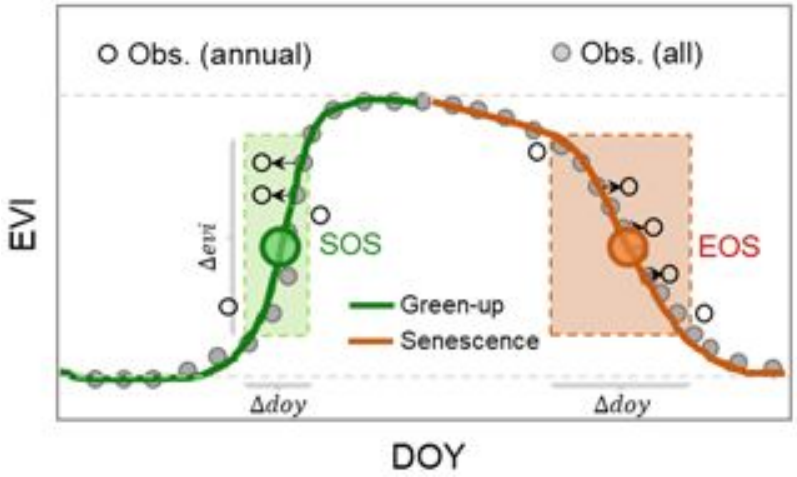
Fitting with CCDC

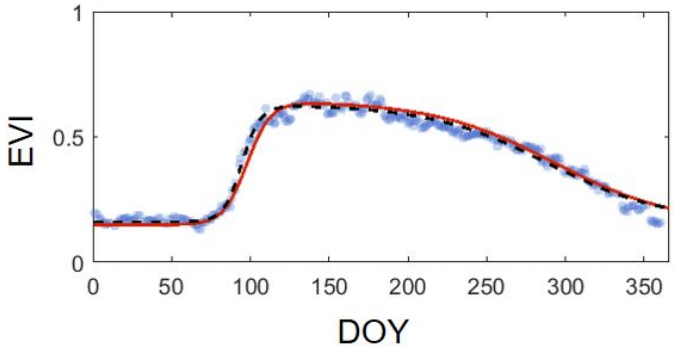


Session: Exploring the global Landsat archive with CCDC, Wednesday 10:15,

Temporal Context

Fitting with Double Logistics





Long: 92.72° W

Lat: 32.79° N

Type: forest

Li, et al., A dataset of 30 m annual vegetation phenology indicators (1985–2015) in urban areas of the conterminous United States, Earth Syst. Sci. Data, 11, 881–894, 2019

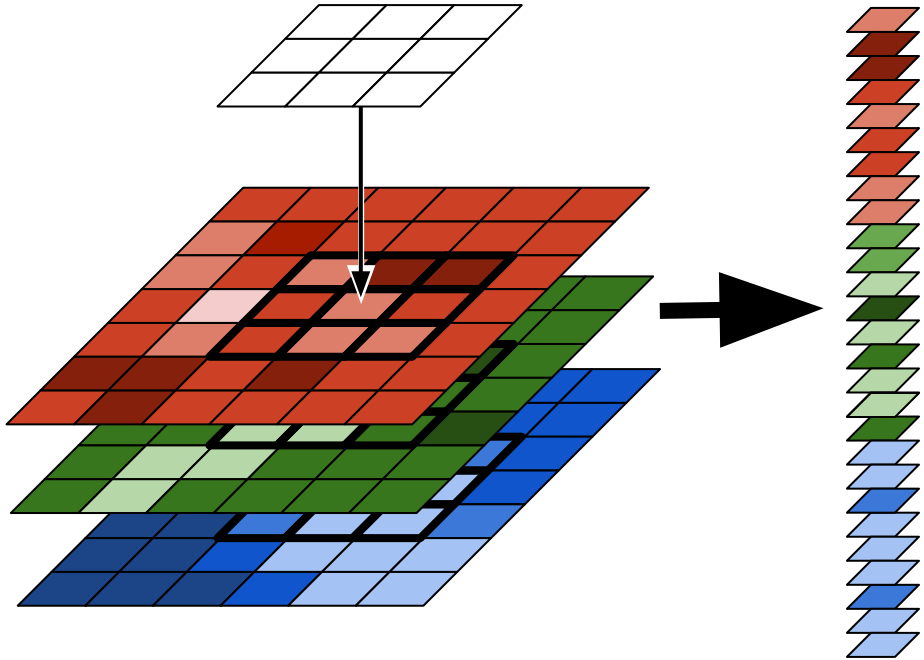
Spatial Context

Adding some texture measure – any measure! - to a classification usually improves the accuracy

Mryka Hall-Beyer https://prism.ucalgary.ca/handle/1880/51900

Spatial Context

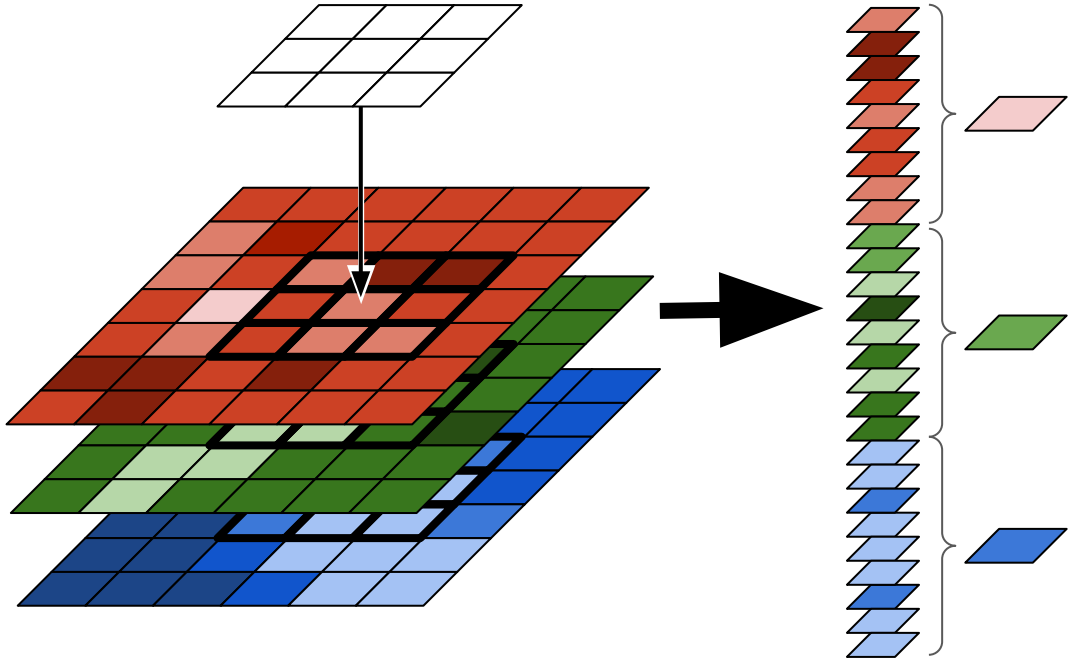
NeighborhoodToBands



Spatial Context

NeighborhoodToBands

ReduceNeighborhood

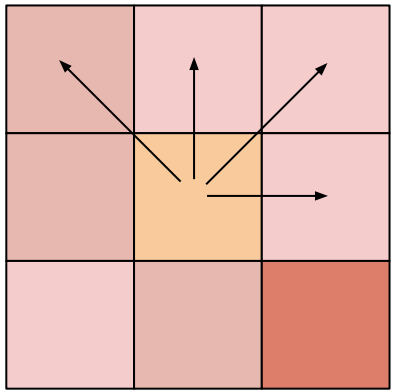


Spatial Context

NeighborhoodToBands

ReduceNeighborhood

GLCMTexture



$$ASM = \sum_i \sum_j \{p(i,j)\}^2$$

$$IDM = \sum_i \sum_j \frac{1}{1+(i-j)^2} p(i,j)$$

$$CON = \sum_{n=0}^{N_g-1} n^2 \left\{ \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} p(i,j) \right\}, |i-j| = n$$

$$ENT = -\sum_i \sum_j p(i,j) \log(p(i,j))$$

$$COR = -\sum_{i,j} \frac{(i-\mu_x)(j-\mu_y)}{\sqrt{(\sigma_x \sigma_y)}} p(i,j)$$

Object Based Image Analysis

Super pixels with SNIC

reduceConnectedComponents()

OBIA Presentation and Video

Optional eyebrow

Issues and Limitations

1. Memory Limits

2. Saved Classifiers

3. Control

Memory Constraints

100 MB

There's a 100 MB limit on the size of a table (sampling) and the size of a trained classifier.

Memory Constraints

You can (re)train a classifier more than once

```
var trainings =  ee.List.sequence(0, 3).map(function(cover) {
    return image.addBands(landcover.eq(cover).stratifiedSample(...)
})

var classifier = ee.Classifier.smileCart()
    .train(trainings.get(0), "cover")
    .train(trainings.get(1), "cover")
    .train(trainings.get(2), "cover")
    .train(trainings.get(3), "cover")
```

Memory Constraints

You can limit the size of classifier

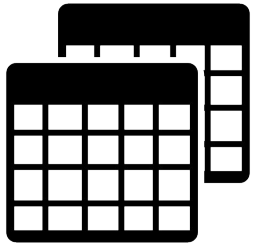
```
var classifier = ee.Classifier.smileRandomForest({
    numberOfTrees: 100,
    minLeafPopulation: 5,
    maxNodes: 10000
})
```

Saved Classifiers

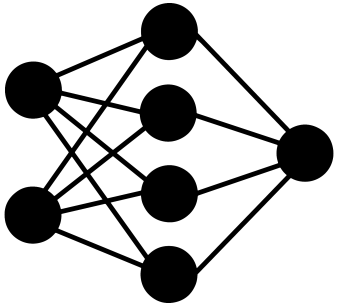
32 MB

There's a 32 MB limit on the size of the string representation of a classifier.

Control



Too Much Data



Complex Models

More Features

Session: Deep Learning with TensorFlow and Earth Engine
Tuesday, 2:30PM

Participate in an Earth Engine Machine Learning User Study

Do you apply Machine Learning to Remote Sensing / Geospatial Analysis?

We want to hear from you! Sign up to participate in an upcoming user study.

Know someone who would be interested? Please pass it on!

https://forms.gle/EA2uBxNkcyZi5D3z7


Thank you!