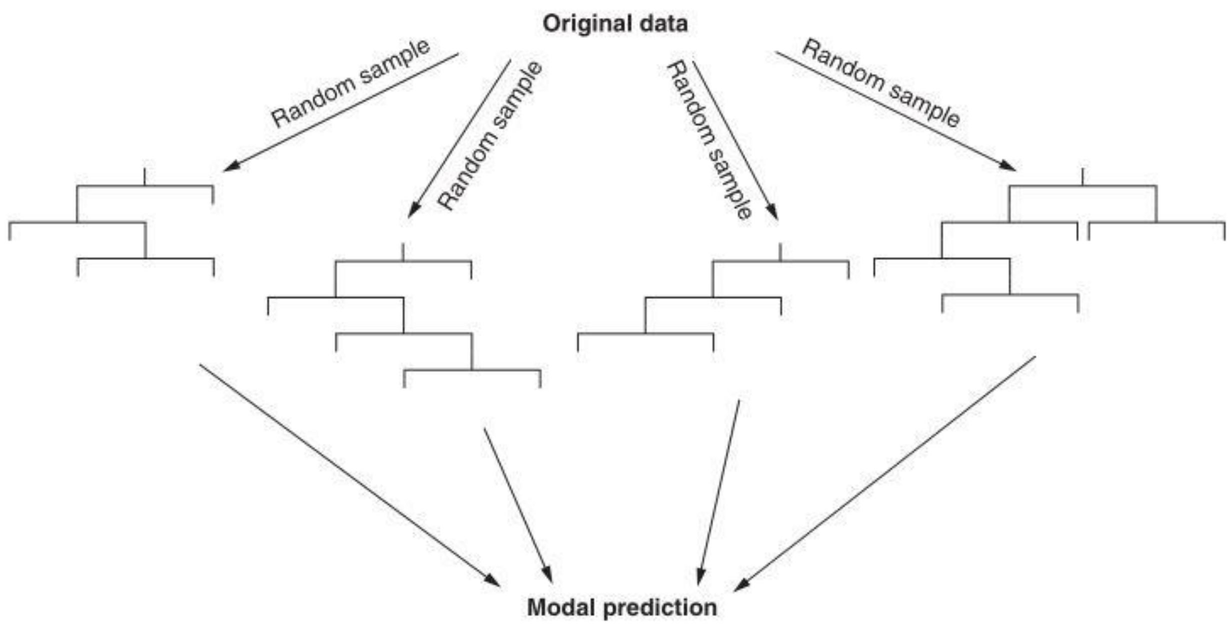
Land Use Land Cover Classification using Machine Learning Methods

Eduardo Ribeiro Lacerda

Fluminense Federal University (UFF)
International Institute for Sustainability (IIS)

What is this course about?

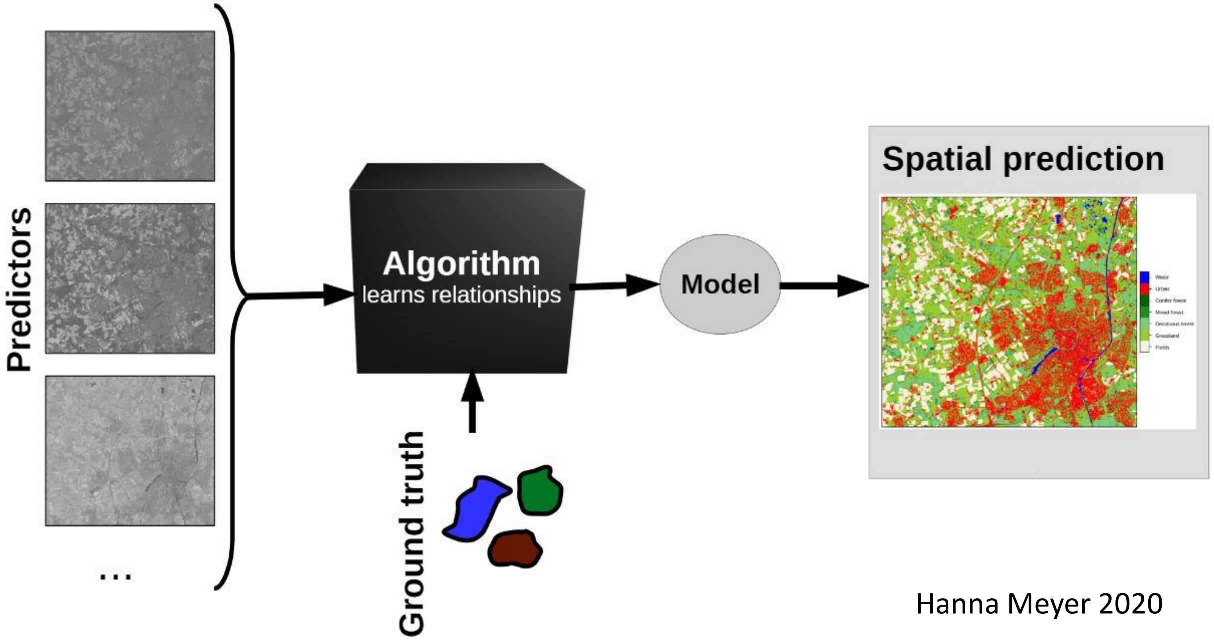


```r
rTask <- mlr::makeClassifTask(data = amostras_df, target = "class") # create task
rf = mlr::makeLearner("classif.randomForest", predict.type = "prob") # create learner
rfModel <- mlr::train(rf, rTask) # train the model
kFold <- mlr::makeResampleDesc("RepCV", folds = 10, reps = 50) # cross validation parameters
rfFoldCV <- mlr::resample(learner = rf, task = rTask, resampling = kFold, measures = list(mmce, kappa))
```

What will we learn?

- How to work with raster and vector data in R and Google Earth Engine
- How to get some good samples to train your model
- How to clean and prepare your data
- How to apply machine learning methods like Random Forest to classify satellite images
- How to improve your model
- How to validate your results
- Export and visualize output data

So…the main idea is to…



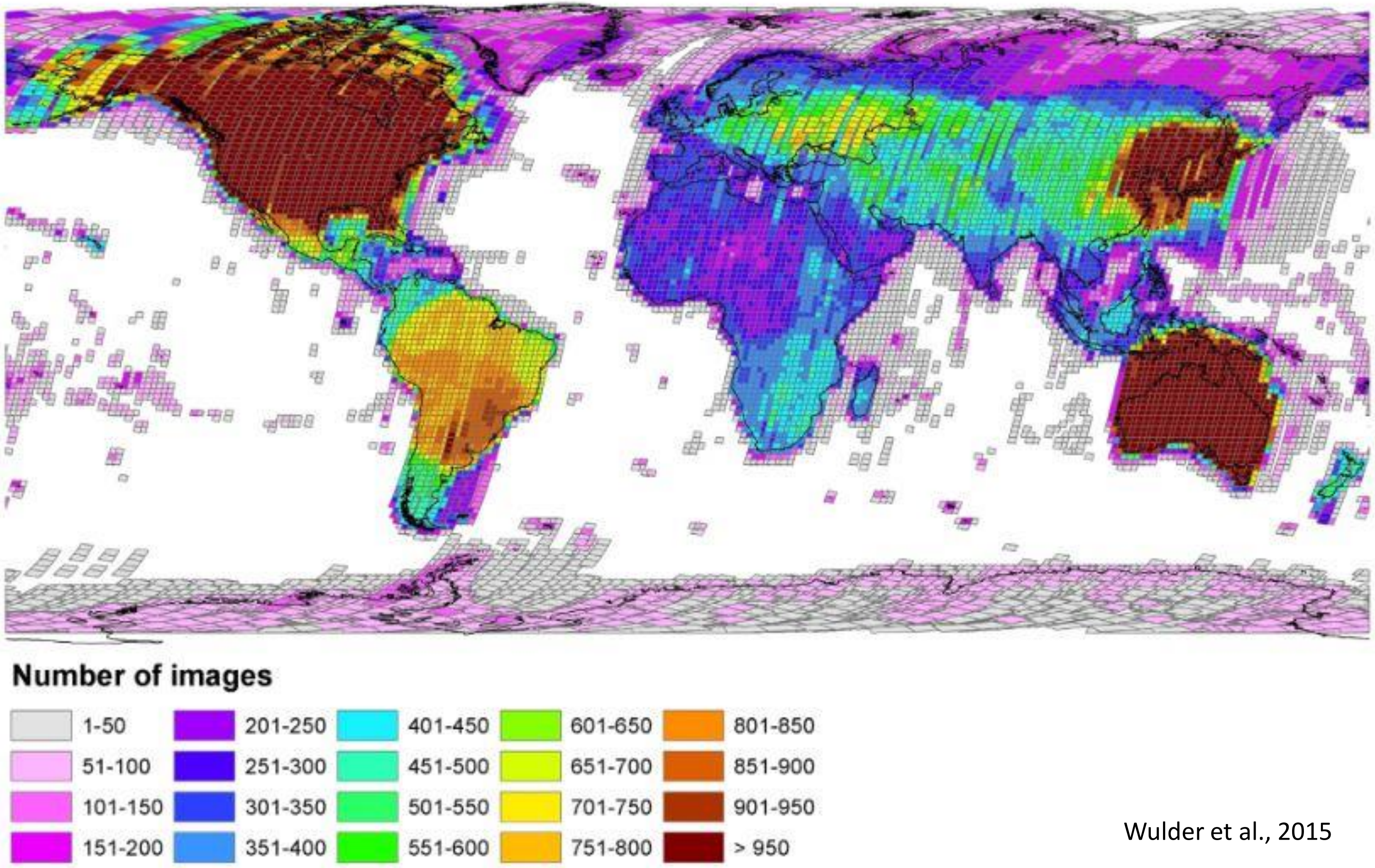
Hanna Meyer 2020

- First get some good input satellite image to serve as predictor variables
- Collect good samples to train the model
- Train the model to do some tests to improve even more the final results
- Use the model to classify the satellite raster data and create a beautiful final land use map :-)

About the input data: examples of available satellite/sensor data

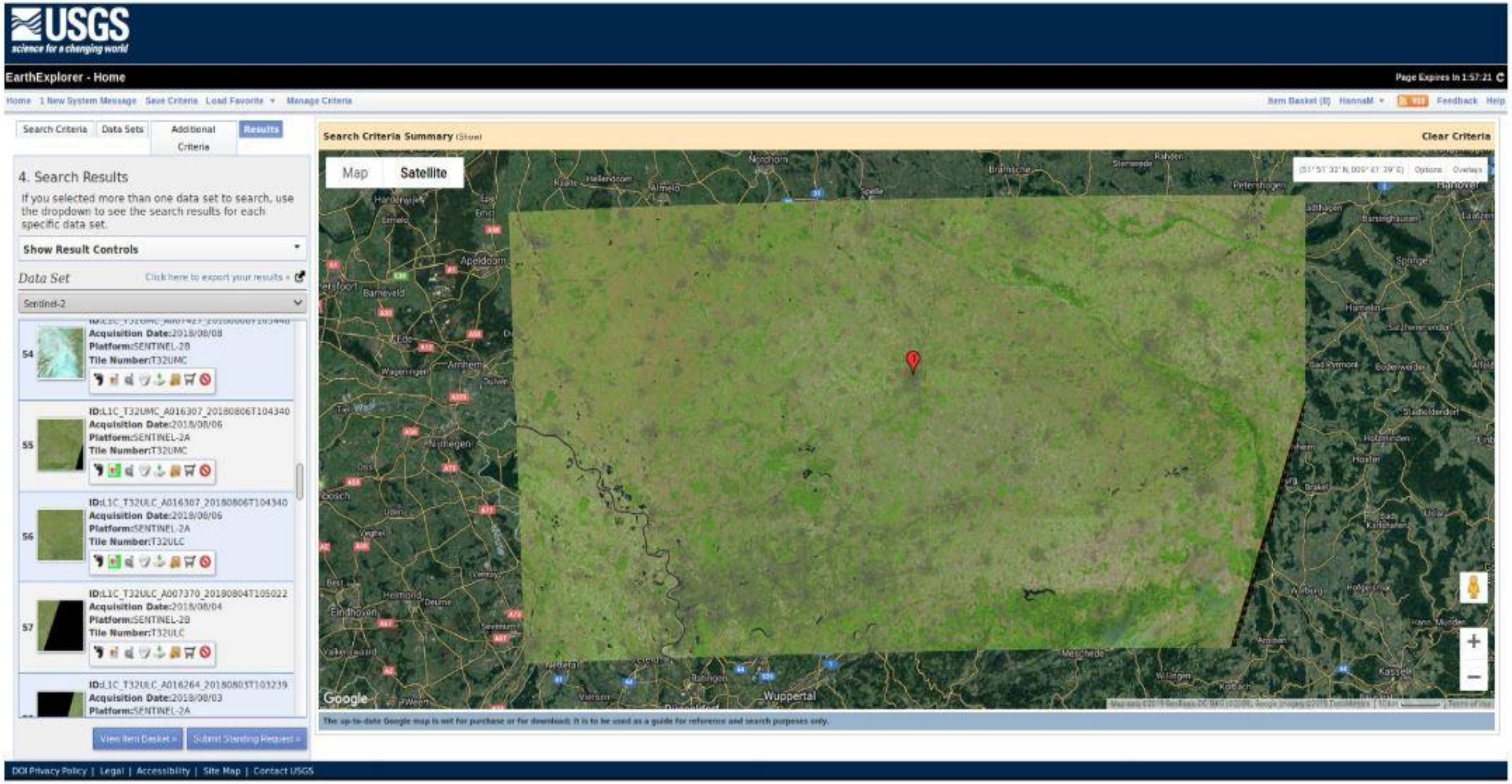
Platform/Sensor	Spatial Resolution	Temporal Resolution	Availability
Landsat MSS	79	16 days	started in 1972
Landsat TM	30	16 days	started in 1982
Landsat ETM+	30	16 days	started in 1999
Landsat 8 OLI	30	16 days	started in 2013
Landsat 9 OLI-2	30	16 days	mid 2021
Sentinel 2	10-20	5/10 days	started in 2014
MODIS	250-1000	4 per day	started in 2000

Accessible Landsat data



Wulder et al., 2015

And how we can access them?



And how we can access them?

Earth Engine Data Catalog

Search

English

Home | View all datasets | Browse by tags | Landsat | MODIS | Sentinel | API Docs

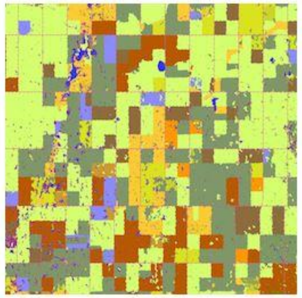
Earth Engine Data Catalog

Earth Engine's public data catalog includes a variety of standard Earth science raster datasets. You can import these datasets into your script environment with a single click. You can also upload your own raster data or vector data for private use or sharing in your scripts.

Looking for another dataset not in Earth Engine yet? Let us know by suggesting a dataset.

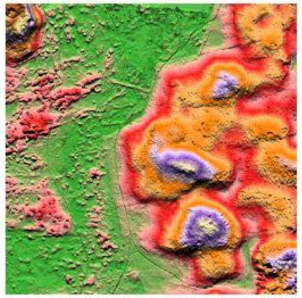
Filter list of datasets

Canada AAFC Annual Crop Inventory



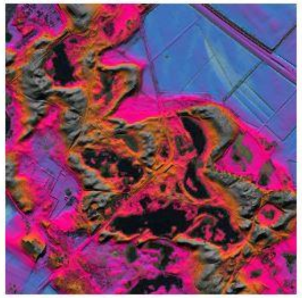
Starting in 2009, the Earth Observation Team of the Science and Technology Branch (STB) at Agriculture and Agri-Food Canada (AAFC) began the process of generating annual crop type digital maps. Focusing on the Prairie Provinces in 2009

AHN Netherlands 0.5m DEM, Interpolated



The AHN DEM is a 0.5m DEM covering the Netherlands. It was generated from LIDAR data taken in the spring between 2007 and 2012. It contains ground level samples with all other items above ground (such as buildings, bridges, trees etc.) removed.

AHN Netherlands 0.5m DEM, Non-Interpolated



The AHN DEM is a 0.5m DEM covering the Netherlands. It was generated from LIDAR data taken in the spring between 2007 and 2012. It contains ground level samples with all other items above ground (such as buildings, bridges, trees etc.) removed.

AHN Netherlands 0.5m DEM, Raw Samples

The AHN DEM is a 0.5m DEM covering the Netherlands. It was generated from LIDAR data taken in the spring between 2007 and 2012. This version contains both ground level samples and items above ground level (such as buildings, bridges, trees etc).

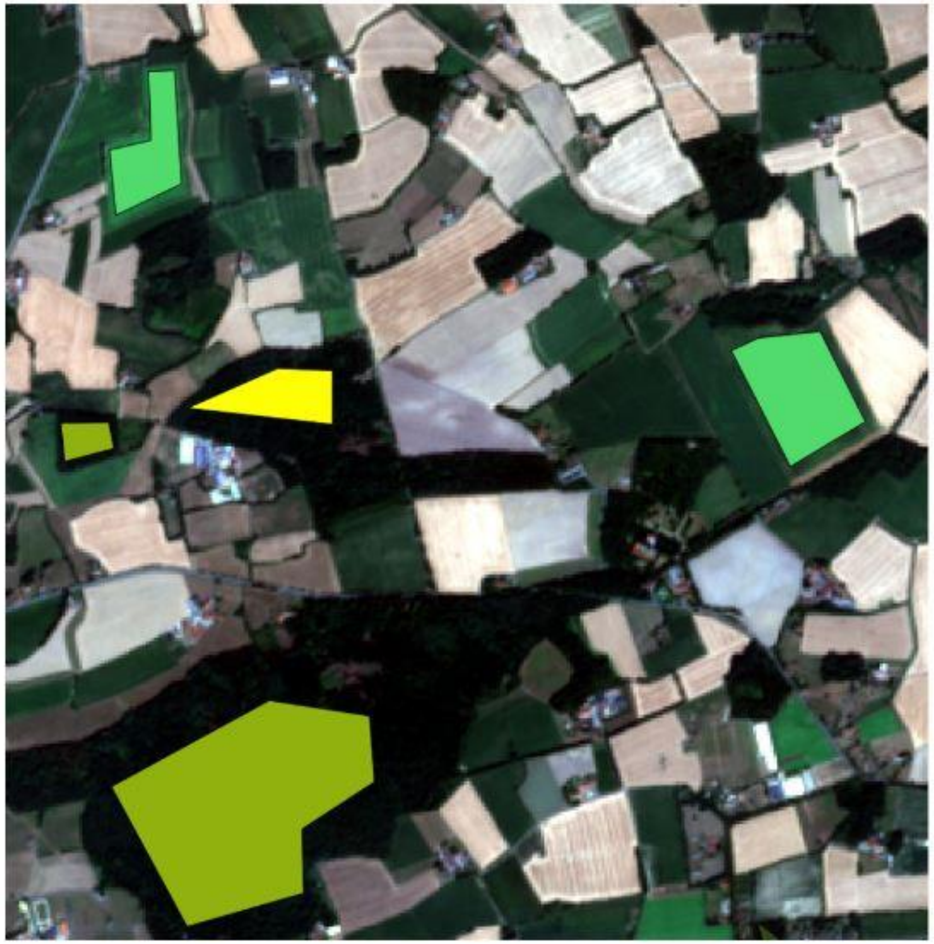
ASTER L1T Radiance

The Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER) is a multispectral imager that was launched on board NASA's Terra spacecraft in December, 1999. ASTER can collect data in 14 spectral bands from the

I downloaded the image. What do I do now?

Point data

Polygon data



Hanna Meyer 2020

Now we choose the algorithm to use

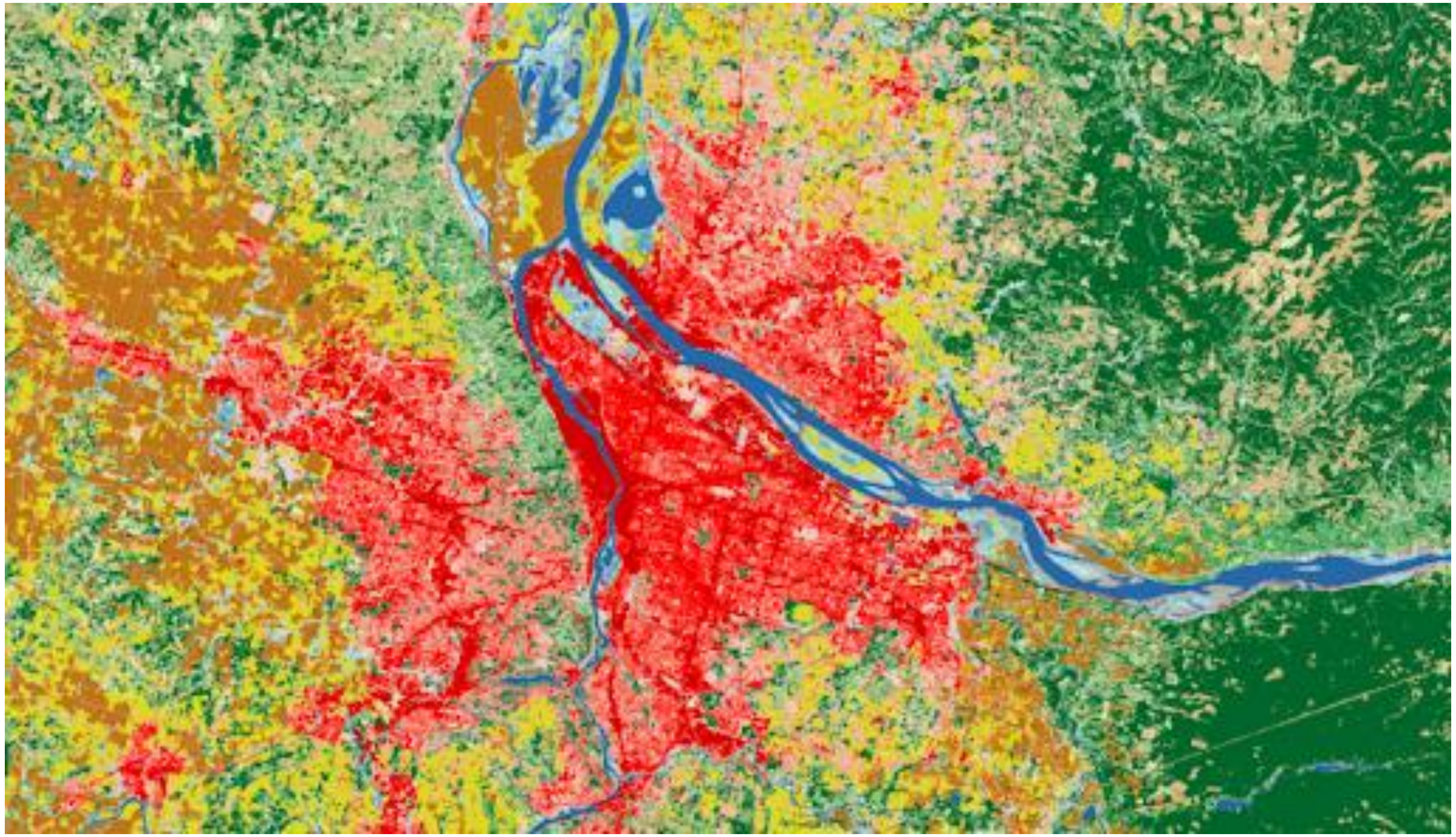
There are many options…

- ee.Classifier
  - ee.Classifier.cart(crossvalidationFactor, maxDepth, minLeafPopula...
  - ee.Classifier.decisionTree(treeString)
  - ee.Classifier.decisionTreeEnsemble(treeStrings)
  - ee.Classifier.gmoMaxEnt(weight1, weight2, epsilon, minIterations, ...
  - ee.Classifier.libsvm(decisionProcedure, svmType, kernelType, shri...
  - ee.Classifier.minimumDistance(metric)
  - ee.Classifier.naiveBayes(lambda)
  - ee.Classifier.randomForest(numberOfTrees, variablesPerSplit, min...
  - ee.Classifier.smileCart(maxNodes, minLeafPopulation)
  - ee.Classifier.smileNaiveBayes(lambda)
  - ee.Classifier.smileRandomForest(numberOfTrees, variablesPerSpl...
  - ee.Classifier.spectralRegion(coordinates, schema)
  - ee.Classifier.svm(decisionProcedure, svmType, kernelType, shrinki...

Filter

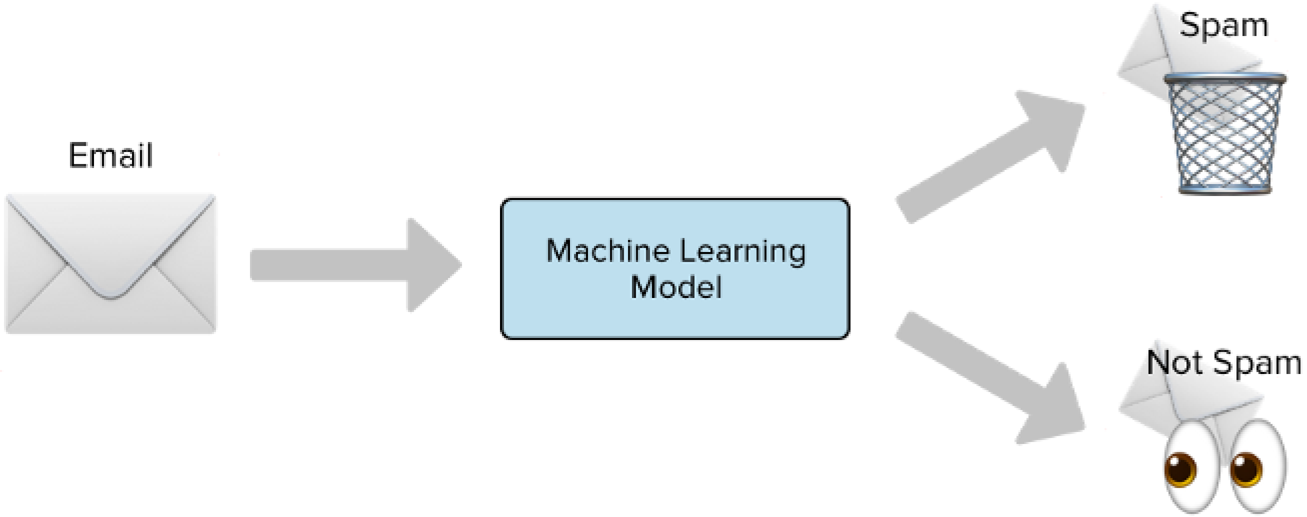
	class	name	short.name	package
1	classif.ada	ada Boosting	ada	ada,rpart
2	classif.adaboostm1	ada Boosting M1	adaboostm1	RWeka
3	classif.bartMachine	Bayesian Additive Regression Trees	bartmachine	bartMachine
4	classif.binomial	Binomial Regression	binomial	stats
5	classif.boosting	Adabag Boosting	adabag	adabag,rpart
6	classif.bst	Gradient Boosting	bst	bst,rpart
7	classif.C50	C50	C50	C50
8	classif.cforest	Random forest based on conditional inference trees	cforest	party
9	classif.clusterSVM	Clustered Support Vector Machines	clusterSVM	SwarmSVM,LiblineaR
10	classif.ctree	Conditional Inference Trees	ctree	party
11	classif.cvglmnet	GLM with Lasso or Elasticnet Regularization (Cross Validated...	cvglmnet	glmnet
12	classif.dbnDNN	Deep neural network with weights initialized by DBN	dbn.dnn	deepnet
13	classif.dcSVM	Divided-Conquer Support Vector Machines	dcSVM	SwarmSVM,e1071
14	classif.earth	Flexible Discriminant Analysis	fda	earth,stats
15	classif.evtree	Evolutionary learning of globally optimal trees	evtree	evtree
16	classif.extraTrees	Extremely Randomized Trees	extraTrees	extraTrees
17	classif.fdausc.glm	Generalized Linear Models classification on FDA	fdausc.glm	fda.usc
18	classif.fdausc.kernel	Kernel classification on FDA	fdausc.kernel	fda.usc
19	classif.fdausc.knn	fdausc.knn	fdausc.knn	fda.usc
20	classif.fdausc.np	Nonparametric classification on FDA	fdausc.np	fda.usc
21	classif.featureless	Featureless classifier	featureless	mlr
22	classif.fnn	Fast k-Nearest Neighbour	fnn	FNN
23	classif.gamboost	Gradient boosting with smooth components	gamboost	mboost
24	classif.gaterSVM	Mixture of SVMs with Neural Network Gater Function	gaterSVM	SwarmSVM

Output data



- Urban
- Forest
- Bare Soil
- Grasslands
- Pasture
- Others?

Bias-variance trade-off



Bias-variance trade-off

Underfitting and **Overfitting** are two important sources of error in model building. It also reduce the *generalizability* of our model.

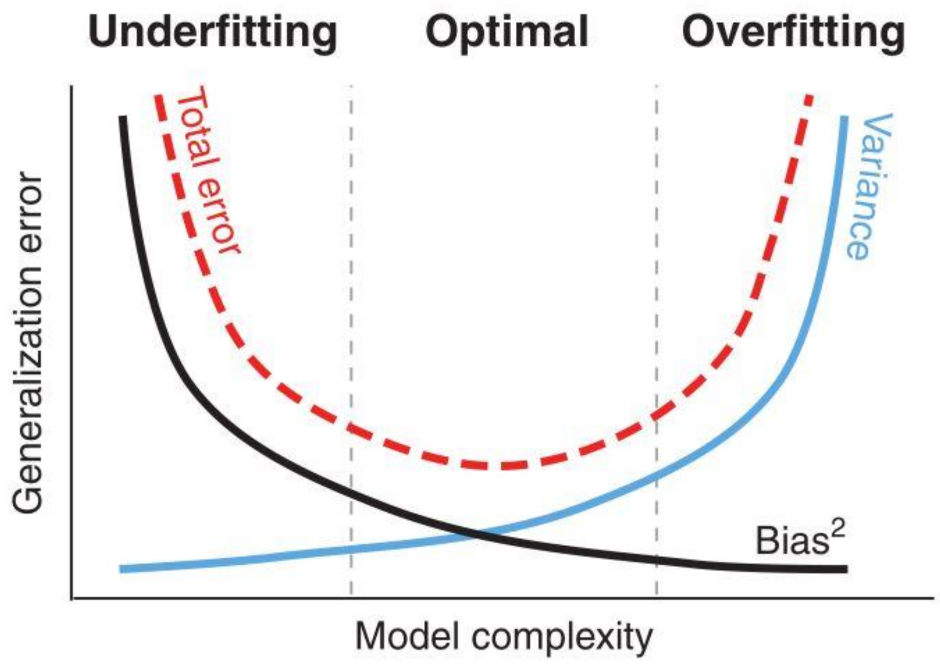
Underfitted – Your model is too simple and is biased towards misclassifying certain types of classes. A model that is underfitted will perform poorly on both the data we use to train it and with new data.

Overfitted – Your model is too complex and is modeling noise in the data that you used to train it. A model that is overfitted will perform well on the data used to train it, but poorly on new data. So, that not good!

With **overfitting** we are modeling noise. **And the pattern of noise is very specific to an individual dataset!**

Ryns, 2020

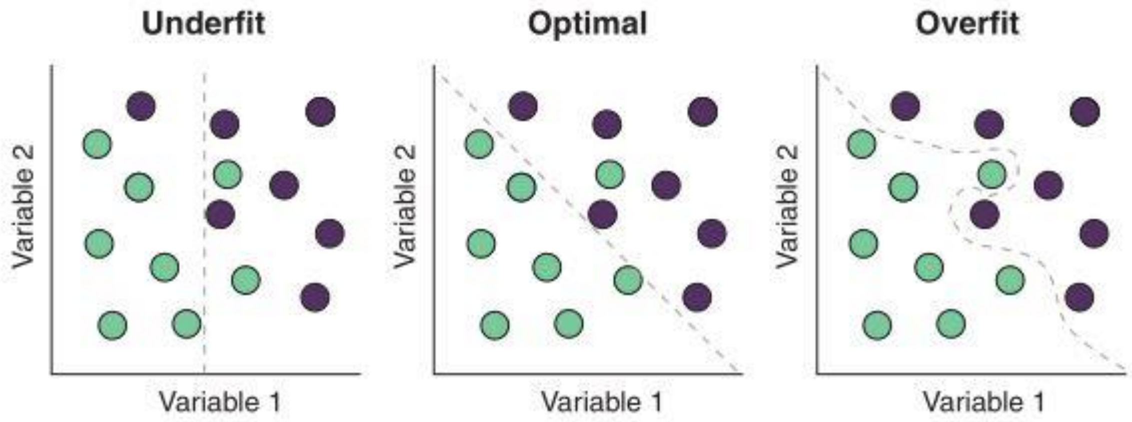
Bias-variance trade-off



- Generalization error is the proportion of erroneous predictions a model makes and is a result of overfitting and underfitting.
- The error associated with overfitting (too complex a model) is variance.
- The error associated with underfitting (too simple a model) is bias.
- An optimal model balances this trade-off.

Ryns, 2020

Bias-variance trade-off



- The dotted line represents a decision boundary (model)
- The more complex model is more likely to moss local differences in our data!
- So, how can I tell if I'm underfitting or overfitting? The question is a technique called cross-validation.

Ryns, 2020

Cross-validation

The solution is to evaluate the performance of our model using data that the model hasn't seen yet! We can do that collection future data, but its better to just split the training set as training and test set.

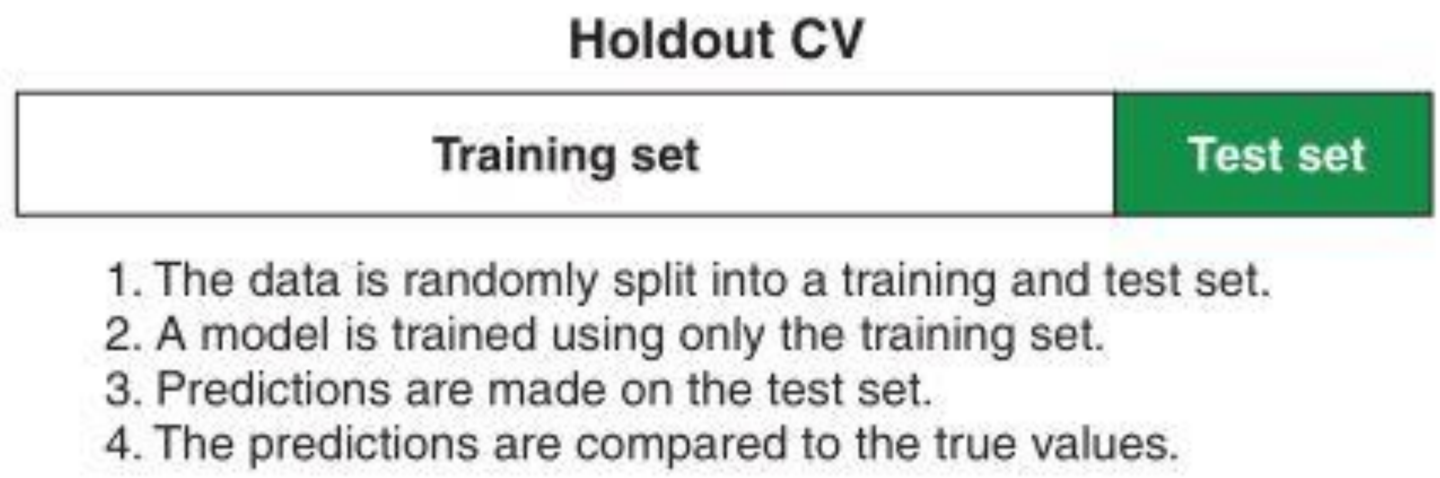
Doing that we can use some performance metrics to show how well the model will perform on unseen data.

Types of cross-validation

- Holdout cross-validation
- K-fold cross-validation
- Leave-one-out cross-validation

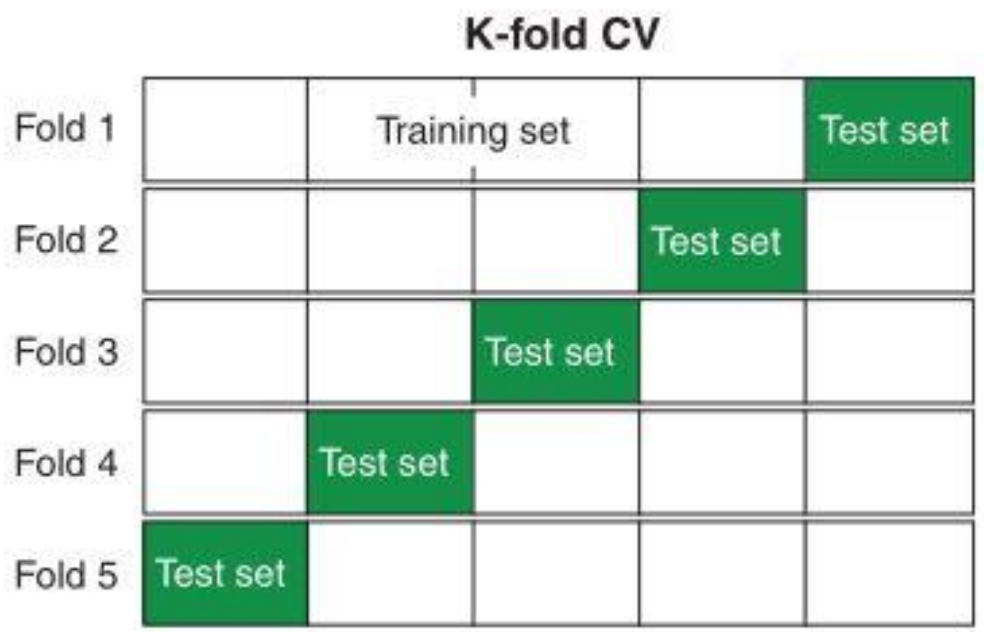
Ryns, 2020

Cross-validation



Ryns, 2020

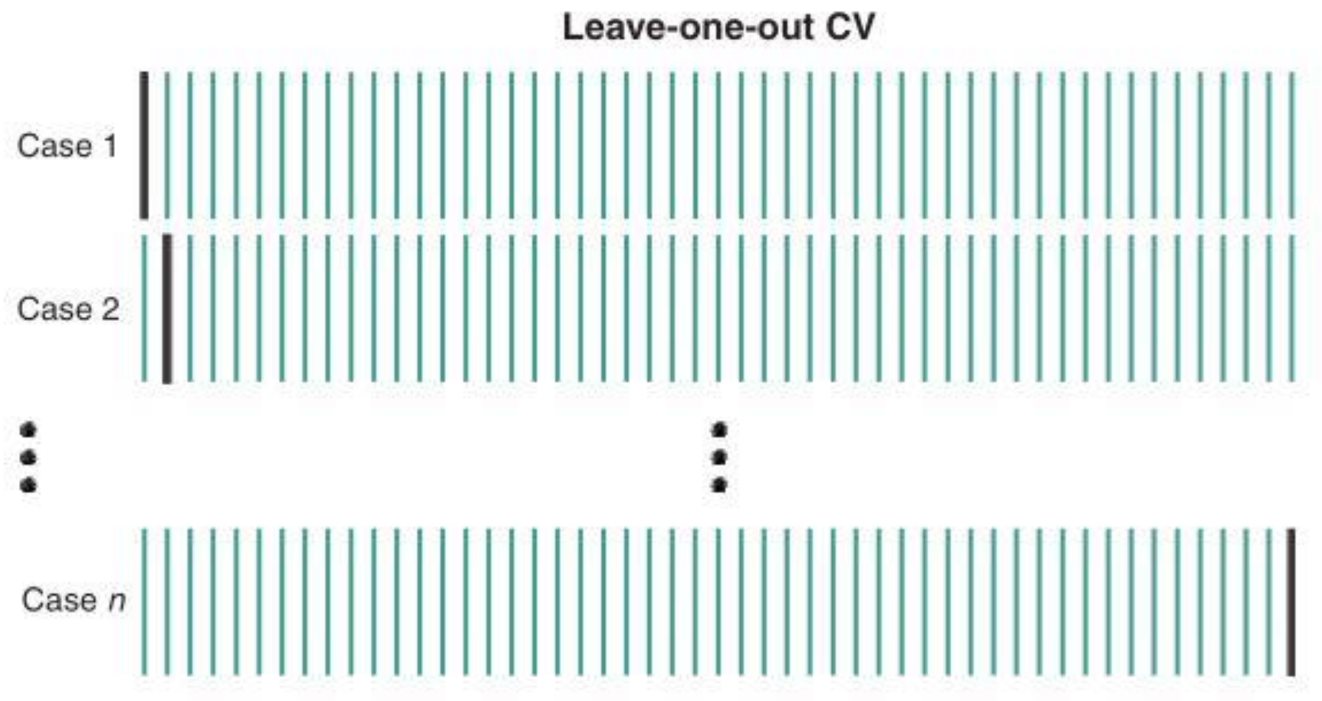
Cross-validation



1. The data is randomly split into *k* equal-sized folds.
2. Each fold is used as the test set once, where the rest of the data makes the training set.
3. For each fold, predictions are made on the test set.
4. The predictions are compared to the true values.

Ryns, 2020

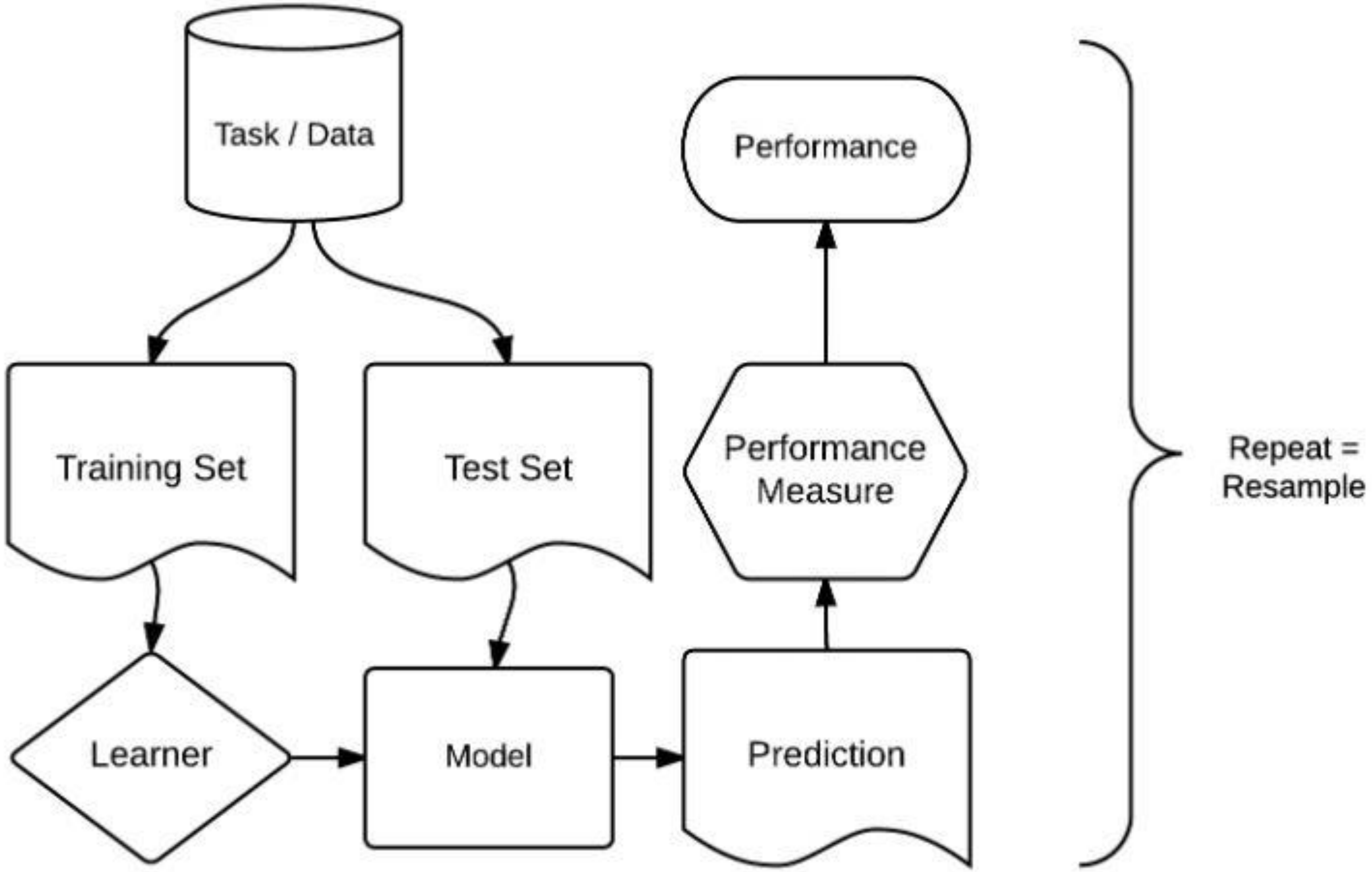
Cross-validation



1. Use all of the data except a single case as the training set.
2. Predict the value of the single test case.
3. Repeat until every case has been the test case.
4. The predictions for each case are compared to the true values.

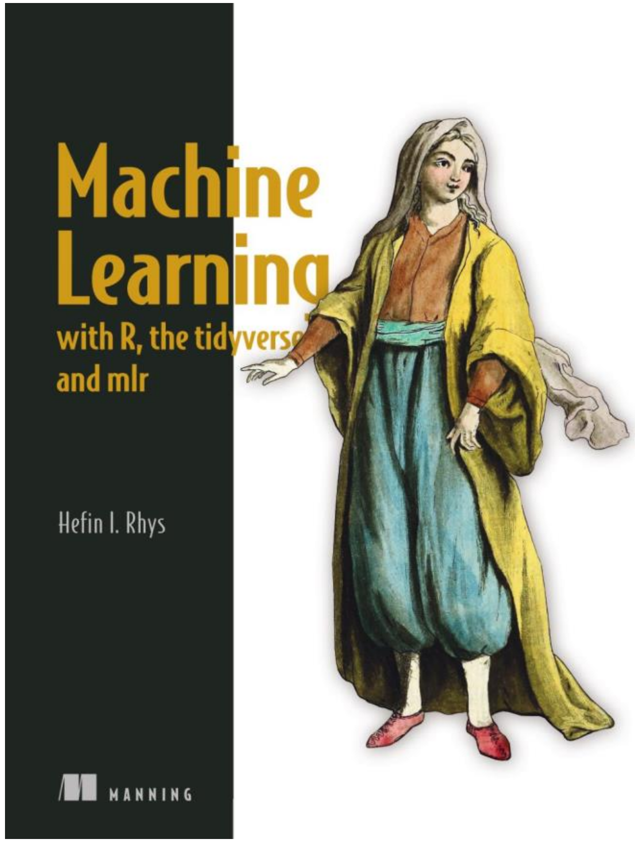
Ryns, 2020

The typical workflow



Practice time!

Extra material



Decision Trees:

https://www.youtube.com/watch?v=7VeUPuFGJHk&ab_channel=StatQuestwithJoshStarmer

Random Forest:

https://www.youtube.com/watch?v=J4Wdy0Wc_xQ&t=7s&ab_channel=StatQuestwithJoshStarmer

https://www.youtube.com/watch?v=nyxTdL_4Q-Q&ab_channel=StatQuestwithJoshStarmer

Extra material

MLR documentation:

https://arxiv.org/pdf/1609.06146.pdf

MLR3 documentation:

https://mlr3book.mlr-org.com/

Tidymodels:

https://www.tidymodels.org/
https://www.tmwr.org/