Appendix D: Custom classification methods detail

This document describes the data and software prerequisites and the steps needed to produce the custom land use classification that was integrated into the final land use classification layers for 2001 and 2014 in Merced County. A git repository containing the data and scripts needed to conduct a custom classification are here: https://bitbucket.org/tncgeo/tnc-merced

Software requirements

➢ R (≥ v3.3.2; r-project.org)
➢ GDAL (≥ v2.0; gdal.org)
➢ Orfeo Toolbox (≥ v5.8; orfeo-toolbox.org)
➢ Google Earth Engine (web account)
➢ Optional: Linux-based computing cluster with at least 24 CPUs (Google Cloud, AWS, etc.)

Data requirements

Base data
➢ LANDFIRE EVT (existing vegetation type)
➢ Prediction raster stack consisting of:
  • Multispectral satellite imagery:
    o Early-year: median mosaic using December-March scenes
    o Mid-year: median mosaic using April-August scenes
    o Late-year: median mosaic using September-November scenes
  • NIRv (1) early, mid, late year derived from multispectral mosaics
  • Optional: Satellite radar imagery: early, mid, late year (as above) mosaics
    o VV polarization, ascending orbit
    o VH polarization, ascending orbit
  • Elevation
  • Slope

Ground truth data
➢ CalAg Permits, DWR Land Use Survey, or other large source of field-level crop data
➢ National Wetlands Inventory (if substantial wetland areas present in county)
➢ Manual collection of field-level polygons for remaining natural land use classes
Methodology

Overview

The Merced County Climate and Multi-Benefit Assessment requires contiguous spatial data over the entire county at 30 m resolution. In order to create the GHG inventory, land cover change projections, and GHG flux baseline, each spatial data layer must have at least two time periods as close as possible to the temporal coverage boundaries (2001 and 2014). The final data layers must be created from stable data sources and replicable methodologies so the assessment can be repeated for other jurisdictions.

To maintain linkage and consistency with CA state agency efforts to assess and monitor GHG reduction potential from natural and working lands we rely heavily on the same products in use by the state agencies. The main base data layer (LANDFIRE) is produced at the national level and used for many other purposes, leading to known inconsistencies and inaccuracies at the county- and landscape-level. In order to address these shortcomings, we created the final land use classification data layers as a combination of LANDFIRE data and a custom classification for specific land use classes that were not of sufficient accuracy. Additionally, there were two other non-custom, non-LANDFIRE data sources that were integrated as part of the final land use layer (irrigated pasture and rice).

A brief outline of the workflow:

1. Geospatial “ground truth” data was collected and manually cleaned
   a. Script: none
2. Prediction raster stack created using Google Earth Engine
   a. Script: create_prediction_raster.js
3. Preprocessing all data using GDAL
   a. Script: preprocess_data.R
4. Segmentation procedure to create field-level boundaries using Orfeo Toolbox
   a. Script: segmentation.sh
5. Create model training dataset using the field data and the prediction raster stack
   a. Script: extract_training_data.R
6. Tuning (optimization) of machine learning classification model in R
   a. Script: model_tuning.R
7. Model application to full prediction raster stack in R
   a. Script: apply_model_to_image.R
8. Post processing cleanup in R
   a. Script: postprocess_data.R
9. Integration of custom classification and non-LANDFIRE data sources with LANDFIRE classes
Ground truth collection and cleaning

Associated script: None

Merced County’s Department of Agriculture provided us with their pesticide permit program GIS data layer, which contains vector polygon data on field-level crop types that may be planted for a given year. We used data from 2014-2015 to create the 2014 ground truth data layer for orchard, vineyard, and non-woody crop types. Using the “crop_list” attribute, we filtered the data for crop types that matched these general crop classes. Polygons with mixed crop classes and pasture or rangeland classes were excluded.

The California Department of Water Resources Land Use Survey of Merced County in 2002 contains vector polygon data on field-level crop types and was used for the 2001 ground truth data layer for orchard, vineyard, rice, and non-woody crop types. Using the “CLASS1” attribute we filtered the data for crop types that matched these general crop classes.

Due to the substantial presence of wetlands in Merced County, vector polygon data from the National Wetlands Inventory was used as ground truth for the custom wetland land use class. We used the “fresh emergent wetland” type and filtered out any polygons which were less than 20 acres. Since most of the wetlands in Merced County are federal or state protected areas, we used this layer for both 2001 and 2014.

For the remaining custom land use class (grasslands), we manually created ground truth polygons in Google Earth Engine using Landsat imagery from 2001 and 2014 as background. Additionally, using this same method we collected manual ground truth polygons from forest, woodland, shrubland, and open water. While these were not part of the custom classification, they were used in the classification model development to increase the ability of the classifier to correctly identify the target custom classes.

For each year’s associated data, separate combined layers were produced containing all the land use class ground truth polygons. A new attribute was created with a custom land use class code. A negative buffer of 45 m (150% of a 30 m pixel) was applied to all polygons to remove any overlapping pixels with other land use classes and to make each field boundary distinct upon conversion to a raster layer. The buffered polygon data was then rasterized to 30 m resolution using the custom land use code as the resulting raster value.

Prediction Raster Stack

Associated script: create_prediction_raster.js

The prediction raster stack is used for training the classifier model and the final classification layer. It is a multiband raster stack composed mainly of seasonal multispectral satellite imagery and derived index of plant growth. The raster stack is created entirely within the Google Earth Engine web platform (in JavaScript).
For the 2014 classification layer, we used satellite imagery from 2015 due to the severe drought conditions during 2014 that led to abnormal growth conditions. While 2015 was still a drought year, the conditions were better suited for classification of land use types. We used the Landsat 8 Surface Reflectance dataset filtered to early (December-March), middle (April-August), and late (September-November) season collections. We used these time periods to take advantage of the highly seasonal changes in both natural and agricultural land use types, which aids the classification process. After masking for clouds using the built in cloud mask flag, we created median mosaics of each seasonal collection to produce a single 6 band scene (Bands 2-7).

Using the NIR and Red bands from these seasonal median mosaics, we calculated the NIRv (1) index as a proxy for vegetation growth. We also included elevation and slope data from the SRTM (Shuttle Radar Topography Mission) 30m dataset. For the 2014 dataset only, we included seasonal mosaics (same time periods as above) of Sentinel-1 radar. This is a C-band synthetic aperture radar that produces imagery related to the structure of the land surface. We used the 10 m (resampled to 30 m on output) interferometric wide swath instrument mode on an ascending orbit. Mosaics using the VV and VH polarizations were both produced for each season. We did not use satellite radar imagery for 2001 due to the poor quality of imagery available during that year.

**Data Preprocessing**

Associated script: preprocess_data.R

All data layers need to be reprojected to the same spatial resolution, extent, and SRS (spatial reference system). Using GDAL, each layer is reprojected to 30m resolution (if raster), and to the NAD83/CA Albers (EPSG:3310) SRS with the same spatial extent (both raster and vector layers).

Using each ground truth raster containing the custom land use codes, a secondary raster must be created with each feature given a unique ID. This feature ID layer will be used to partition the training dataset below into calibration and validation subsets.

**Segmentation**

Associated script: segmentation.sh

A segmentation is performed on the seasonal NIRv data layers in order to define the boundaries of agricultural fields and other homogenous areas of a landscape. This will be used in the postprocessing of the classification layer. For each time period, the three seasonal NIRv layers are extracted from the prediction stack. They must be reprojected into the EPSG:4326 SRS for the segmentation algorithm to work. A no-data mask must also be produced (using Orfeo Toolbox) from the reprojected NIRv layers. These are all fed into the Orfeo Toolbox segmentation algorithm. This uses a mean shift segmentation algorithm to identify nearly
homogenous areas of a raster, and outputs a vector polygon file with each homogenous feature having a unique ID. The vector file is then rasterized at 30 m resolution with the unique ID as the resulting raster value.

Create Training Dataset

Associated script: extract_training_data.R

The training dataset is needed for the classification model development, and is split into a calibration and validation subset. The validation subset is never used in the model development, instead it serves as an independent dataset for evaluating the final model’s performance. Using the ground truth data layer with the custom land use codes, all values of the feature ID layer and prediction raster stack are extracted to a data frame. The final training data frame will consist of \( N \) rows consisting of each extracted pixel and \( M \) columns containing the variables associated with the pixel’s class code, its unique feature ID, and the prediction raster stack (satellite bands, growth indices, topography).

From this full training dataset, two subsets (50-50 split) were created by randomly selecting groups of pixels (rows) that all belong to the same feature ID and putting them in either the calibration or validation subset.

Classification Model Development

Associated script: model_tuning.R

We used a supervised machine learning classification framework for our model development. This approach is suitable for handling large (tens of thousands to millions of records) and complex (non-linear) classification problems. The two algorithms we chose to work with are the Random Forest (RF) and Gradient Boosting (GB) classifiers. Using the calibration sub-dataset as the input, we used 10-fold cross validation procedure to fit hundreds of models of each algorithm across a large range of hyperparameters (i.e., the parameters that allow us to “tune” the model to give better outcomes). Searching across the hyperparameter space for the most accurate model is called tuning or optimizing your classifier. For each combination of hyperparameters, the 10-fold cross validation and an independent validation gives you model accuracy metrics that are used to assess that particular model. Once you have found the optimal set of hyperparameters, you re-run your model on the entire training dataset without 10-fold cross validation to produce your final optimized model. (You have already performed two performance checks on the data used to fit the model so additional assessment is not needed).

We used the H2O machine learning package for R (http://h2o.ai). For the RF classifier, we optimized the number of trees (ntrees), the number of variables sampled at each split (mtries), and the maximum tree depth (max_depth). All combinations of the following hyperparameter sets were evaluated, ntrees (500-2000, steps:100), mtries (2-12, steps: 2), max_depth (10-20, ...
steps: 2), resulting in 576 unique combinations of hyperparameters. For the GB classifier, we optimized the number of trees (ntrees), the maximum tree depth (max_depth), and the learning rate (learn_rate). All combinations of the following hyperparameter sets were evaluated, ntrees (500-2000, steps:100), max_depth (2-18, steps: 2), and learn_rate (0.1, 0.01, 0.001), resulting in 432 unique combinations of hyperparameters.

The hyperparameter space that you can assess and the number of cross-fold validations you can perform will be dependent on the amount of computing power you have access to. If using limited computing resources, we recommend starting with a very large hyperparameter space as above, but running a coarse search across that space (i.e., much larger steps), and then refining to a more constrained space based on the results of the coarse search. Another option for limited computing resources would be to reduce the amount of calibration data that you use in the model. The more data you are able to train your model with, the better that model will be at classifying land use, but more data means longer run times for the model to fit.

In multiclass classification, there are many ways to assess the “best” model. We used the overall accuracy (mean accuracy across all classes) as well as examining the accuracy of each custom class from both the 10-fold cross validation calibration and the independent validation results. Once the best model has been identified from the hyperparameter optimization search, a final model is re-run on the entire training dataset without the cross validation using the hyperparameters found from the optimization. This final model will be used to create the custom classification layer.

**Custom Classification Layer**

Associated script: apply_model_to_image.R

To create the final custom classification layer, the final optimized model is applied to the original prediction raster stack. The prediction raster stack is first converted to a data frame and a separate data frame of the raster coordinates. The prediction data frame is then passed into the H2O prediction function, which takes the final model and predicts the land use class based on the values of the prediction data frame. The output is a single column data frame with predicted land use classes. This predicted data frame then converted back to a raster using the coordinates of the original prediction raster. The result is a single layer raster of the custom land use classes.

**Postprocessing Cleanup**

Associated script: postprocess_data.R

First, any land uses classes that were needed for model development but are not part of the custom classification are removed.
We then used the segmentation raster produced above to clean the custom raster. In a pixel-level classification there tends to be “speckle” or misclassified pixels scattered throughout what would otherwise be a correctly classified area. For example, in a large orchard field there may be small number of misclassified pixels as vineyard. The segmentation raster identified agricultural field boundaries that represent homogenous crop types. For every unique feature in the segmentation raster, we calculated the majority land use class from the custom raster and then applied that class to pixels within the entire feature.

For natural (non-ag) areas, we used a different procedure to clean up any misclassified speckle. This involved using a multipass filter with a 270 m kernel (9 pixels) applied to only areas that were not classified as agriculture. Any areas within the kernel that are not valid—in this case wetlands or grasslands—are converted to the majority class of the kernel window. This is repeated 10 times.

**Integration of Custom with Base Dataset**

Associated script: output_final_classification.R

After the final cleaned custom raster is finished, it must be integrated into the base data layer—in this case LANDFIRE EVT. The two rasters are merged such that all valid values from the custom classification raster supersede any values from the LANDFIRE EVT layer. LANDFIRE EVT values from the custom classes (grassland, wetland, orchard, vineyard, non-woody crop) were removed prior to merging. We then re-added the open water and developed land use classes from LANDFIRE back to the data layer, superseding any custom classification values that may have misclassified these areas. Finally, we added the irrigated pasture and rice (for 2014 only) to the data layer (see below for explanation).

For the irrigated pasture land use class we did not perform a custom classification, instead using data provided to us by the California Department of Conservation (DOC). The DOC data is created statewide every two years using manual identification of irrigated pasture fields using aerial imagery. Data layers for Merced County 2002 and 2014 were used directly in the final land use layer. Due to the very small amount of rice fields in Merced County and the exclusive use of those fields during the year for rice, we used fields identified as rice from the CalAg permit data from 2014 and the DWR Land Use Survey from 2002 directly in the final land use layer.

**Reference:**