Thesis

---

A Thesis
Presented to
The Division of Mathematics and Natural Sciences
Reed College

---

In Partial Fulfillment
of the Requirements for the Degree
Bachelor of Arts

---

Sarah Maebius

December 6th, 2020

Approved for the Division
(Mathematics)

_____

Tom Allen

# Acknowledgements

Deeply grateful to...

# Table of Contents

# List of Tables

# List of Figures

# Abstract

The preface pretty much says it all.

Second paragraph of abstract starts here.

# Introduction

## 0.1 Research Problem and Background

Western redcedar trees are evergreen trees that typically grow up to 75 feet tall and are located throughout the Pacific Northwest, making it an organism with a tolerance for shaded regions with moist environments. These trees are native to the land and have served many purposes to people and animals living in the vicinity of the trees, including medicinal, building, and habitat functions (Peterson, n.d.). Western redcedars were culturally important to indigenous peoples who valued the strong wood and utilized it for construction and everyday necessities ("Western red cedar," n.d.). Over the past decade, reports of dead western redcedars have been increasing, suggesting that something other than natural causes is killing off this species. In general, western redcedars experience several hardships in surviving in the Pacific Northwest, with common causes of death such as forest fires, clearcutting, small animals eating the saplings, and harsh weather including strong winds that easily uproot the trees (Peterson, n.d.). Dying redcedar trees can be identified by their branches which turn brownish yellow or fall off completely. Another sign is that the top of a tree will turn brown and lose leaves ("Western redcedar Dieback," n.d.). Losing this native tree would have a detrimental effect on animals in the area who rely heavily on the trees for their lifestyle. Scientists have speculated that western redcedar decline might be caused by recent dry summers, the spread of tree disease, insects, or other weather related events ("Western redcedar Dieback," n.d.). Since this is a recent issue, there is not a lot of resources explaining the decline. This research aims to provide more insight into the cause of the western redcedar decline by first predicting the location of the western redcedar trees in Portland and then predicting their condition in terms of health. Having more insight helps to prevent further tree deaths and save the western redcedar species, which also extends to similar tree species and provides more knowledge about environmental changes in the Pacific Northwest region.

Modelling in this research will be conducted by combining information gathered from remotely sensed images with ground level information. There are several sources publishing research done using satellite imagery for land classification (Castelluccio, Poggi, Sansone, & Verdoliva, 2015) and for predicting tree species (Fricker et al., 2019), which will be the groundwork for this project's application of remote sensing models to the specific topic of western redcedar mortality.

Remote sensing is a process for identifying the physical aspects of a large area of land by collecting and measuring the reflected light using airplanes or satellites. This method provides access to lot of information about the region of interest that would otherwise be limited from a human's ground-level persepective. Remote sensing is used to detect characteristics on land as well as the sea. Some remote sensing land applications include tracking forest fires, volcanic eruptions, city growth, and also forest changes.

Satellites orbit the earth, carrying sensors that record levels of electromagnetic radiation detected by the reflection of the sun on the earth. The data collected is the measured radiation from different regions of the spectrum (visible light, infrared, ultraviolet, etc.). Depending on the surface, sunlight gets absorbed or reflected back

at the orbiting satellite. The remotely sensed data is available at certain resolutions depending on the instruments used. Spatial resolution is the area imaged by a satellite image, where a detailed image has smaller spatial resolution hence smaller pixels. Spectral resolution is a measure of the size of the wavelength interval, where smaller bands and smaller wavelengths improve the spectral resolution of an image and enhance the level of detail. Satellite images need to be corrected to combine the bands into an image depending on the analysis.

Imagery come in various band widths and different number of bands. Hyperspectral imagery has narrow bands of sizes 10-20nm and hundreds of bands. So, hyperspectral imagery is successful in distinguishing fine details, but at a cost of more complexity. Multispectral imagery has wider bands and up to 10 total bands. This type of imagery is not complex to work with, but misses some details that hyperspectral imagery detects.

Previous literature has been published in utilizing satellite imagery for predictive models, however, many issues arise in classifying land type through remote sensing mainly due to image quality. A single image can be composed of image strips taken over the course of multiple flight paths. Consequently, these images are taken at different times of day, which compiles into a single image with a lot of variation due to changes in the weather as well as the different angles of the sun's position (Castelluccio et al., 2015). Ideally, a solid classifying model surpasses any error introduced by imperfection in the satellite images. Common approaches to classification on satellite images include support vector machines, random forests, and decision trees. One study on land type classification explores the performance of convolutional neural networks (CNN) as classification models (Castelluccio et al., 2015).

Another related study identifies 7 tree species by applying a hyperspectral CNN model. This study was completed using field data with measured information about tree count, tree species, and mortality status and hyperspectral imagery data. The hyperspectral data is converted into the form of a tree canopy height model with circular polygons centered at individual tree canopies. The study analyzes the performance of CNNs for both hyperspectral imagery data and a Red-Green-Blue (RGB) subset of the hyperspectral imagery data. The results of the experiment conclude that training a CNN on the hyperspectral data outperforms the CNN trained on RGB data in classifying land type on the UC Merced Land Use dataset. The RGB model does not perform as well as the hyperspectral model in terms of distinguishing tree classes, but does perform around the same level of accuracy in terms of genus classification. For this project, the data comes from Planet.com and only has RGB and IR data available for the Portland region.

This thesis follows along with the methods in (Fricker et al., 2019) for combining the satellite imagery data with ground data by creating spatial polygons and extracting pixel-level information to train classification models. The work in this thesis differs from the methods in the cited literature by using random forests and support vector machines as classification models instead of CNNs and uses the RGB model instead of hyperspectral model. Finally, this thesis applies established classification methods for satellite imagery data to answer the specific question about the cause of death of western redcedars in the Pacific Northwest. The models predict the loca-

tions for each tree species to facilitate the identification of any patterns in the species mortality over the past decade.

## 0.2 Overview

### 0.2.1 A Statistical Learning Approach to Identifying Location of Western Redcedars

This work combines RGB imagery data from Planet.com with ground level tree data from the RStudio `pdxTrees` library and applies random forest and support vector machine classification methods to predict the location of tree species (specifically western redcedars) in Portland, Oregon and model the condition of western redcedars to ultimately understand the cause of death in this species. Chapter 1 describes the two levels of data in this research: ground level and pixel level data, where the data comes from, and how the data is combined for modeling. Chapter 2 discusses the process behind training random forest and support vector machine models. Chapter 3 summarizes how the data is prepared for the final model and the outcome of the research followed by a discussion and conclusion.

# Chapter 1

# Data

There are two sources of data in this project. Data at a pixel level come from satellite images downloaded from Planet.com (`https://www.planet.com`), and data at the ground level come from library `pdxTrees` in RStudio (`https://github.com/mcconvil/pdxTrees`).

### 1.0.1 Imaging

Satellite images come from Planet.com's PS2.SD instrument found on Dove-R satellites that were launched in 2017. Satellites have an approximate frame size of 20 km x 12 km parallel to flight path. From a single pass, 4 to 5 overlapping framed scenes are combined to form a tile. To separate the light into red, blue, green, and near-infrared (NIR) channels, the telescope has a high-performance butcher-block filter made of 4 individual pass-band filters. Planet.com equates this pass-band filter with that of Sentinel-2.

As it orbits the Earth, the satellite captures continuous strips of single frame images that are split into a RGB frame and NIR frame. The butcher-block pass separately photographs the red, blue, green, and NIR bands and then combines all four to form an image. Images are are downloaded already fully processed and ready to be analyzed. The process includes corrections for radiometric calibration, terrain distortions corrections, elevation corrections, and atmospheric corrections.

The images used in this research were taken on September 2nd, 2020 at an altitude of 475 km. Images are taken from the summer months to reduce the chance of rain clouds and capture peak greenness of the trees. The compiled multi-spectral image is composed of 4 bands: blue, green, red, and NIR, with center wavelengths 490 nm, 565 nm, 665 nm, and 865 nm respectively, with an average bandwidth of 41 nm. The average spatial extent is around 25 km by 8 km. The images cover the entire Portland area, with specific measurements of 26 km from west to east and 30 km from north to south. The spatial resolution of the pixels is approximately 3 meters.

Planet.com provides a way to download free images that are already processed and corrected for analysis. Some drawbacks to using this data are the limited number of downloads available per free account, the low spatial resolution, which decreases the performance of statistical models trying to predict tree species location, and the low

spectral resolution, so the satellite sensor cannot detect wavelengths in detail.

### 1.0.2 Ground Data

Ground level data is available from the 'pdxTrees' package in RStudio. This data was collected as part of the Portland's Parks and Recreation's Urban Forestry Tree Inventory Project, which collected park tree information from 2017 to 2019. Originally, the inventory project started with the goal of improving the city's tree management plans. Under the guidance of US Forestry staff, volunteers around the city's neighborhoods are trained in identifying tree species and provided with tools necessary to record tree measurements. Measurements in feet were made using diameter tape. The Portland park trees inventory consists of over 25,000 trees, each with information about tree location, size, species, and health. For the purposes of this project, the following variables were selected:

- Spatial information: Longitude and latitude of tree. Location of tree is recorded on a tablet with Collector for ArcGIS and recorders manually select the tree from a satellite image on the screen. The tree's location is mapped over the satellite images in QGIS for analysis.

- Species: Tree genus, species, and common name. The tree species is used to train the model and predict the tree species of pixels in the test data.

- Crown size: Crown width from north to south, crown width from east to west, and the base height of the crown in feet. The crown of the tree is the tree's above ground leaves, so the crown width is the longest horizontal distance that can be measured between the leaves of the tree. A measuring wheel is used to measure this distance.

- Tree Condition: Categorical variable with four categories (from best to worst), good, fair, poor, and dead. A tree was considered good if the tree is strong and has no apparent issues, fair if the tree is average condition with possibly a few dead branches, poor if the tree has major wounds and dead major canopy loss, and dead if the tree has no live leaves.

To prepare the ground level data for training the model, the data was filtered by species and crown size. The following tree species were identified and included: maples, oak, Douglas-fir, Western Redcedar, and sequoia. These species are all native to the Pacific Northwest, so they benefit the ecosystem and having a model of these trees will help track the species' locations. Since the data is used to identify trees in the satellite images, the ground data was also filtered to only include the trees with a crown width from north to south of 20 feet or more, this ensures that there will be around 9 to 20 pixels per tree.

Portland's Tree Inventory includes a variable for identifying trees including Western Redcedars as well as variables for filtering to include larger trees canopies (at least 6m in diameter) in the dataset, which helps construct a training set of tree polygons

over a satellite image. This data would be even better suited for the purposes of this project if the street trees subdata also contained variables for tree canopy size.

### 1.0.3 Training Data

#### Characteristics of Training Data

The training data at the ground-level was filtered by tree size and tree type. Trees with at least 20 feet in north to south crown width were included in the training dataset to ensure that the polygons around each tree contain around 9 to 20 pixels. Figure 1.1 plots the north to south crown width values against the west to east crown width values to present the ranges of crown widths in the entire pdxTrees parks dataset and to justify only using the north to south crown width for filtering the dataset since the two variables have a strong positive correlation.

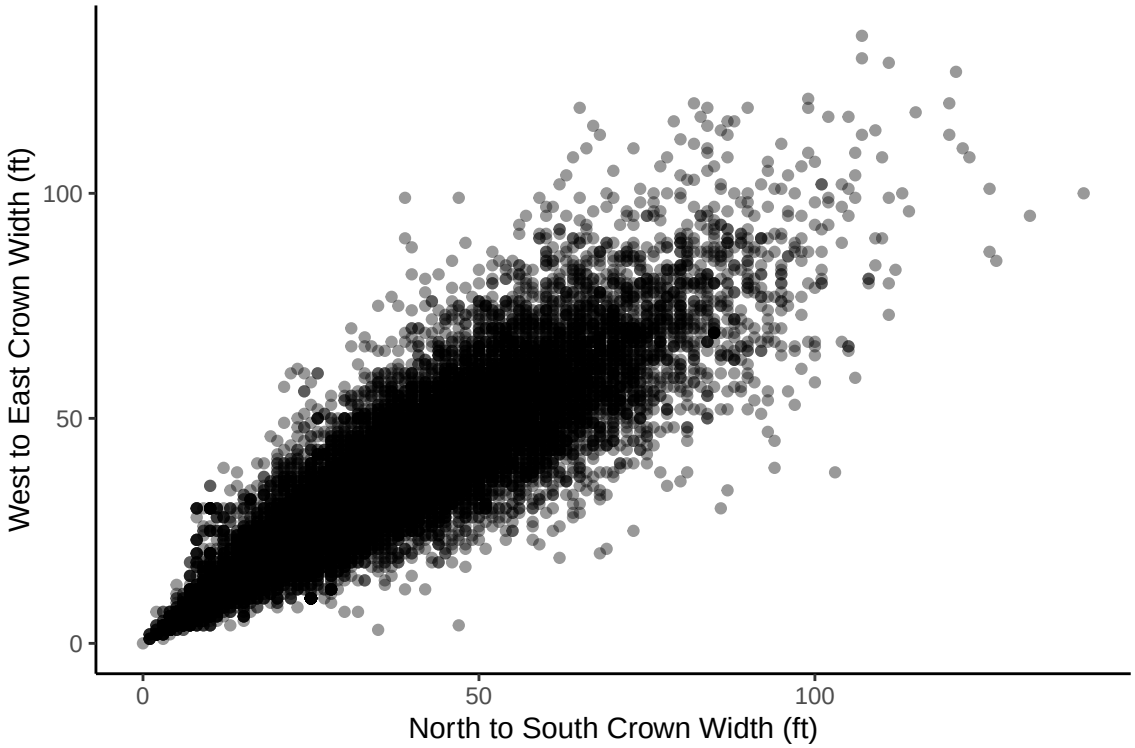


Figure 1.1: Plot comparing North to South Crown Width (ft) to West to East Crown Width (ft). A tree with a higher north-south crown width has a higher west-east crown width, so without loss of generality the north-south crown width variable is used to filter the dataset for larger trees only.

Six tree types were determined to have large enough canopies on average and appear frequently enough to construct a training dataset with plenty of observations for each species of tree: Douglas-Fir, English Oak, Giant Sequoia, Maple, Western Redcedar. Table 1.1 provides the number of trees under each common name category.

Table 1.1: Common tree names included in the tidy data and their total counts.

| Tree Name | Tree Count |
|---|---|
| Douglas-Fir | 6485 |
| Norway Maple | 1406 |
| Western Redcedar | 652 |
| Bigleaf Maple | 464 |
| Giant Sequoia | 315 |
| English Oak | 135 |

The largest group contains 6485 observations of Douglas-Fir trees, and the smallest group contains 135 observations of English Oak trees.

### Creating Spatial Polygons

The training dataset was converted into shapefiles for each type of tree and exported into QGIS where polygons were manually drawn around 100 of the trees for each species. The point shapefile layers were displayed over the raster images downloaded from Planet.com in QGIS, which provided a guide point for locating individual trees. Then using QGIS's drawing tool, a polygon is carefully drawn around the tree canopy (see Figure 1.2 and Figure 1.3). Most polygons turned out to be four to six-sided polygons to retain the general shape of the tree canopy. Trees polygons were only created if the outline of the tree canopy was clearly visible or surrounded by other trees of the same species in order to avoid including pixels from the wrong species in that polygon. Also, the shadows of the trees are visible in the raster images, so the polygons were drawn with the intention of not including the tree shadow. For the five different species of trees, at least 100 polygons were drawn around trees of that type, with each polygon containing at least 6 pixels and at most 20 pixels. The polygon shapefiles were then exported into RStudio where the rest of the analysis was conducted.

Table 1.2 displays the number of polygons per raster strip and the amount of pixels from those polygons. Table 1.3 displays these counts in terms of tree species.

Table 1.2: Number of polygons per raster strip

| Raster Strip | Polygons | n |
|---|---|---|
| a | 197 | 3614 |
| b | 257 | 4887 |
| c | 36 | 359 |
| d | 8 | 98 |
| e | 177 | 2395 |

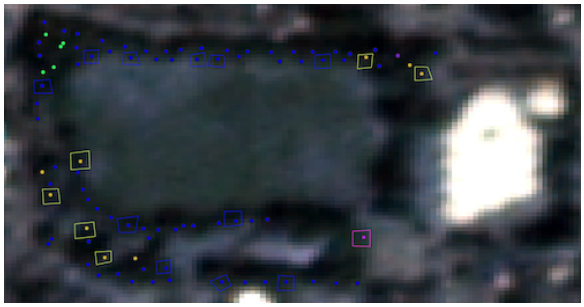

Figure 1.2: Polygons drawn around different tree species: maples (blue), sequoias (yellow), redcedars (pink). Note there are no polygons around trees with canopies that overlap with canopies of different tree species, like douglas-firs (green), to avoid misclassifying pixels from different tree species.

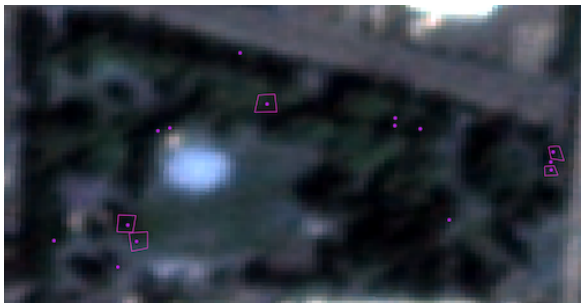

Figure 1.3: Polygons drawn around redcedar trees

### Polygon Limitations

Ideally, the raster strips contain training trees of each species, and as a result, the polygons are evenly distributed across the raster strips, but due to limitations of our ground-level data as well as the coverage of the strips across the city of Portland, this is not achievable. Each of the raster strips contains some number of manually drawn polygons, however, raster strips 'C' and 'D' are missing some of the tree species. The raster strips do not all cover Portland to the same extent. Strips 'A' and 'B'

Table 1.3: Number of polygons per tree type per raster strip

| Raster Strip | Tree Name | Polygons | n |
|---|---|---|---|
| a | Bigleaf Maple | 28 | 283 |
| a | Douglas-Fir | 35 | 432 |
| a | English Oak | 30 | 288 |
| a | Giant Sequoia | 26 | 336 |
| a | grass | 1 | 1158 |
| a | Norway Maple | 48 | 747 |
| a | Western Redcedar | 29 | 370 |
| b | Bigleaf Maple | 26 | 279 |
| b | Douglas-Fir | 38 | 604 |
| b | English Oak | 62 | 618 |
| b | Giant Sequoia | 32 | 399 |
| b | grass | 3 | 1579 |
| b | Norway Maple | 70 | 1026 |
| b | Western Redcedar | 26 | 382 |
| c | Bigleaf Maple | 9 | 55 |
| c | Douglas-Fir | 19 | 202 |
| c | English Oak | 2 | 39 |
| c | Western Redcedar | 6 | 63 |
| d | Douglas-Fir | 5 | 56 |
| d | Western Redcedar | 3 | 42 |
| e | Bigleaf Maple | 37 | 268 |
| e | Douglas-Fir | 6 | 86 |
| e | English Oak | 6 | 80 |
| e | Giant Sequoia | 54 | 701 |
| e | grass | 1 | 186 |
| e | Norway Maple | 37 | 567 |
| e | Western Redcedar | 36 | 507 |

Table 1.4: Variables and pixel values included in the pixels dataset

| ID | red | green | blue | ir | rstrip | id_type | Genus | Species | Cmmn_Nm | Cr_W_NS | Cr_W_EW | Crw_B_H | Conditn | St_Wdth | ndvi |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 43 | 4707 | 4125 | 3187 | 6627 | a | acpl | Acer | ACPL | Norway Maple | 28 | 28 | 8 | Fair | NA | 0.1694018 |
| 43 | 4384 | 3860 | 2864 | 7111 | a | acpl | Acer | ACPL | Norway Maple | 28 | 28 | 8 | Fair | NA | 0.2372336 |
| 43 | 4446 | 3803 | 2807 | 7443 | a | acpl | Acer | ACPL | Norway Maple | 28 | 28 | 8 | Fair | NA | 0.2520818 |
| 43 | 4619 | 3911 | 2859 | 7801 | a | acpl | Acer | ACPL | Norway Maple | 28 | 28 | 8 | Fair | NA | 0.2561997 |
| 43 | 4695 | 4060 | 3201 | 5996 | a | acpl | Acer | ACPL | Norway Maple | 28 | 28 | 8 | Fair | NA | 0.1216911 |
| 43 | 4344 | 3719 | 2728 | 6747 | a | acpl | Acer | ACPL | Norway Maple | 28 | 28 | 8 | Fair | NA | 0.2166622 |

encompass most of the city while strips 'C' and 'D' only make up a small portion that covers the corners of the city not reached by strips 'A' and 'B'.

The polygons are drawn with the intention of outlining pixels for a known tree species so that the extracted pixels truly represent that species and so that there is a sufficient amount of pixels per species, however, limitations of the satellite images' resolution make some error inevitable. Many parks trees have canopies that overlap, which cause some polygons to contain pixels from trees of different species. Other pixel errors might come from including pixels that are cast in shadow by the polygon's training tree or by surrounding infrastructure. To avoid these errors, polygons with uncertain canopies are cross-checked with the satellite filter on Google Maps. For example, a point in QGIS that is indicated as a douglas-fir tree might appear as one tree on the low-resolution image from Planet.com, but a closer look at Google Maps shows that it is actually two trees in close proximity. Having low spatial resolution decreases the number of pixels available within a polygon, but that has to be balanced out with the need for a large number of pixels. With regards to the number of pixels per tree species, having around the same number of tree polygons per species results in differing amounts of pixels per species due to the different average sizes of tree canopies for different species. After drawing the images, the pixels totals per tree species is computed to ensure that the training pixels data contains at least 800 pixels per tree species.

### Combining Ground-level Data with Pixel-level Data

The first spatial join in RStudio was conducted to match up each Spatial Polygon with a point in the training dataset. Then the raster images were loaded and joined with the polygons to extract the pixel values inside each polygon for all 4 bands (red, green, blue, infrared). Ultimately this turned into a pixel table with rows representing each pixel with its corresponding polygon, light reflection intensity values for all 4 bands, and the ground information about that tree. Table 1.4 displays the first few entries of the pixels dataset. Table 1.5 contains the summary of the total number of pixels for each tree type. Figure 1.4 displays the range of reflection intensity pixel values per tree type. Each tree species has similar ranges of values over the red, green, blue, and infared bands. Of the species included in the training data, the Giant Sequoia trees appear to have higher density counts for the red, green, and blue bands.

Table 1.5: Number of pixels per tree type

| Tree Name | Pixels Count |
|---|---|
| Bigleaf Maple | 885 |
| Douglas-Fir | 1380 |
| English Oak | 1025 |
| Giant Sequoia | 1436 |
| grass | 2923 |
| Norway Maple | 2340 |
| Western Redcedar | 1364 |

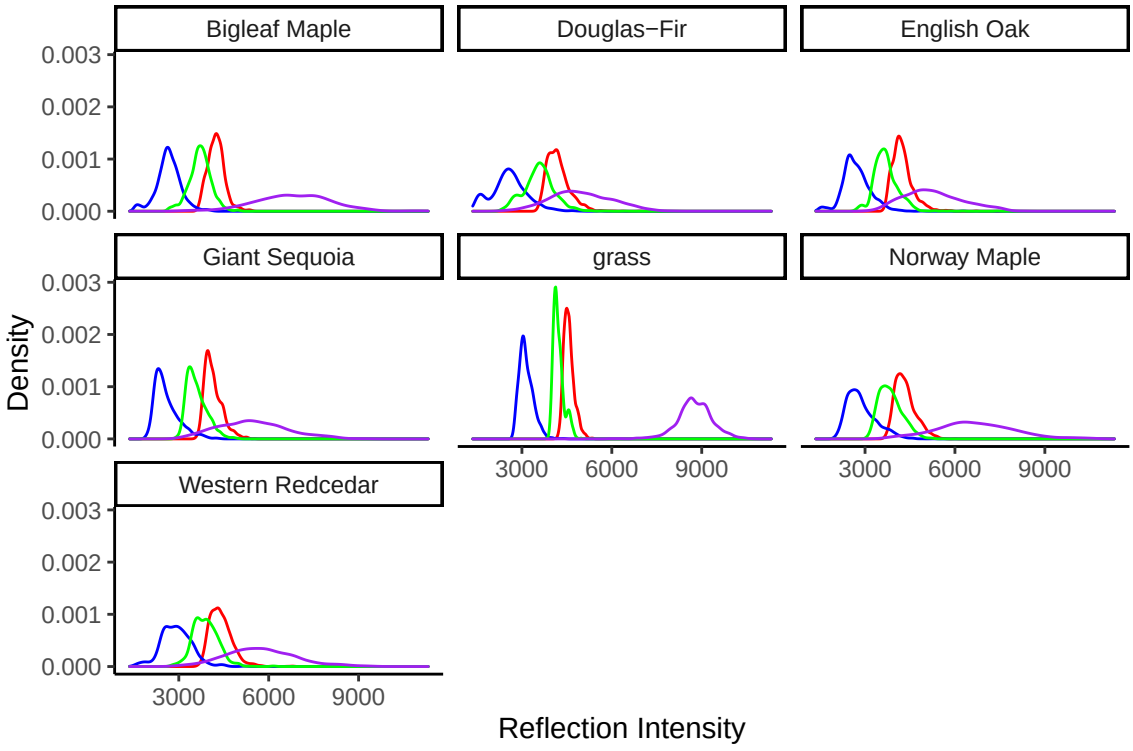


Figure 1.4

## 1.1 Preparing Raster Images

The training model has to be applied to the entire raster image to predict the location of Western redcedars, however, extracting all the pixels from raster images is a slow process. Filtering the rasters to keep vegetation and to recognize the difference between grass pixels and tree pixels alleviates the computational intensity of extracting all the pixels and improves the performance of the models. To reduce the size of computations, all five raster strips are masked and cropped to only include pixels within the city boundary of Portland, since that is the region of interest, and that is the extent of the ground level data.

### 1.1.1 Masking Vegetation

The satellite images in RStudio need to be masked to reduce the number of pixels that the model has to classify and prevent the chance of a non-vegetation surface being predicted as a tree. A common mask applied to raster images is a Normalized Difference Vegetation Index (NDVI) mask. This index is a measure of the greenness of a pixel, with higher values indicating vegetation and lower values indicating infertile areas such as a rock. The formula for NDVI is $NDVI = \frac{NIR-Red}{NIR+Red}$. Figure 1.5 displays the density of NDVI values over the raster images. To remove the pixels that are not vegetation, an NDVI threshold is determined to be 0.00 to create a mask dividing the pixels into vegetation (1) and non-vegetation (0). When the trained model is applied to the raster image, it is applied to the masked raster image that only keeps pixels with NDVI mask values of 1.

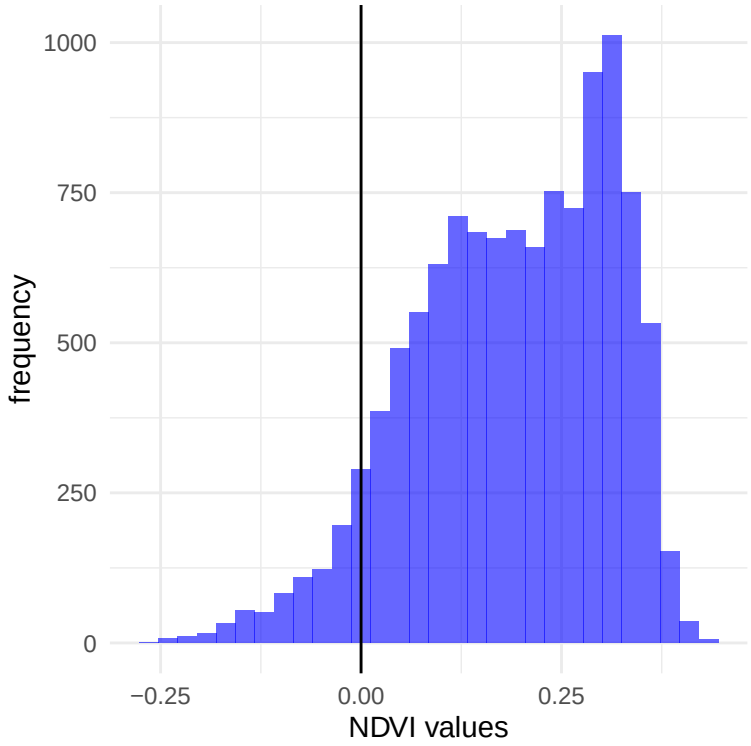


Figure 1.5: Normalized Difference Vegetation Index (NDVI) is a measure of the greenness of a pixel. The histogram displays the frequency of the tree pixels' NDVI values. A good threshold for masking out the nonvegetation pixels is an NDVI of 0.

### 1.1.2 Masking Grass

A second mask is applied to the NDVI masked raster image to distinguish tree pixels from grass pixels. Inspired by previous research, a grass index was created by averaging the values of the four bands, $GRASS = \frac{RED+BLUE+GREEN+NIR}{4}$ (Qian, Zhou, Nytch, Han, & Li, 2020). Grass polygons were created in QGIS to add a grass at-

tribute in the pixels dataset. Five fields of grass were outlined as polygons in QGIS. In RStudio, the pixels were extracted from the grass polygons for a total of 2,923 grass pixels. Figure 1.6 displays how the average band values for the grass pixels differ from the other tree types average band values. Based on the figure, pixels in the raster image that have an average band value above 4900 are filtered out. Masking the grass pixels ensures that a field of grass will not be classified as a tree when applied to the entire raster image

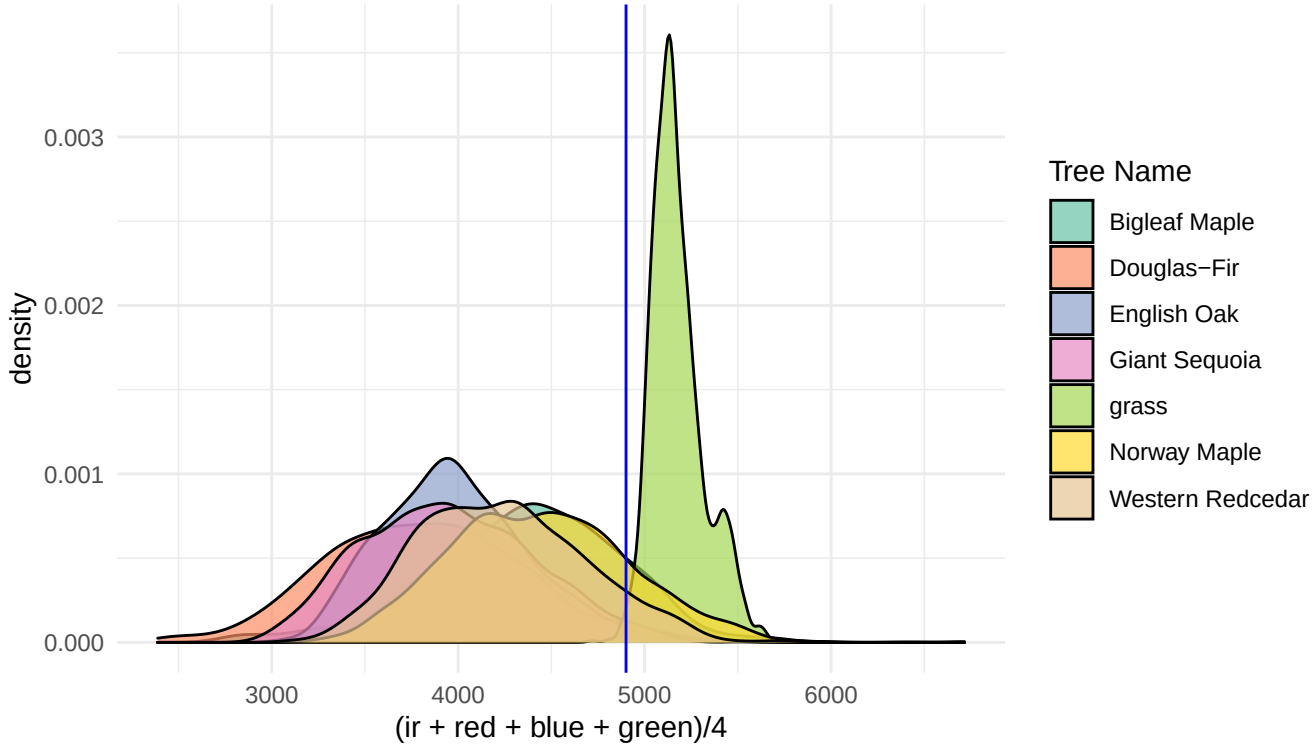


Figure 1.6: Average of band values distribution across different tree types in comparison with grass pixel average band values. Most grass pixels have an average value above 4900, so a mask is applied to the raster image to remove pixels with average values above this threshold.

Figure 1.7 displays a raster strip with every pixel included (top), only the pixels with NDVI values above 0.00 (middle), and only pixels with NDVI values above 0.00 and average band values below 4900 (bottom). Ideally, the bottom image is left with only the tree pixels in the image.

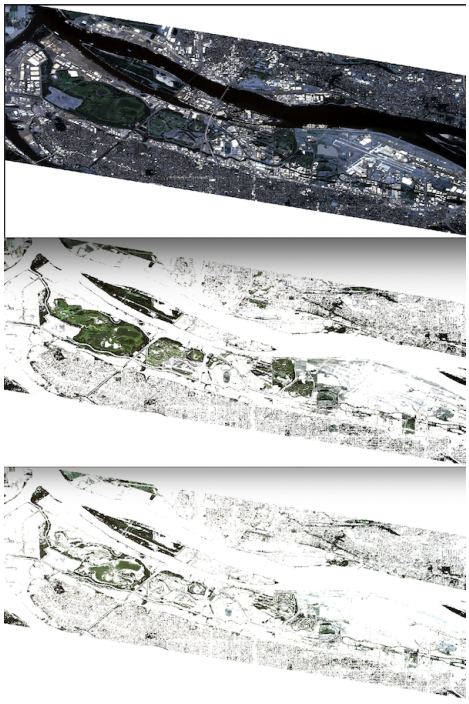

Figure 1.7: A raster strip with every pixel included (top), only the pixels with NDVI values above 0.00 (middle), and only pixels with NDVI values above 0.00 and average band values below 4900 (bottom)

### 1.1.3 Masking Limitations

The low resolution of the raster images cause shapes to appear blurry, and especially tree canopies that are close together appear as a single formation.

Masked raster images were compared in QGIS and, in general, the masks are successful in keeping the forest pixels while still removing building structures and roads. If a field of grass is a brownish color, it gets removed, but some greener fields are not masked out. The masks consistently remove river pixels but not lake pixels if the lake is a greenish color. For example, Smith Lake in North Portland is considered an urban wetland, which signifies a body of water with a lot of vegetation. From the perspective of a satellite, this looks like a field of grass. However, the average band pixel values are less than 4900, so they are not masked out by the final GRASS mask stage, and instead are treated as tree pixels in the model (see Figure 1.8).

To address leftover grass pixels in the filtered raster image, access to lidar data would provide a method of determining the height of certain pixels. Setting a height threshold would ensure that grass pixels are removed while higher vegetation pixels are kept. This would also address the masks' limitations in removing lakes with high vegetation, since those pixels would not meet the threshold for height.

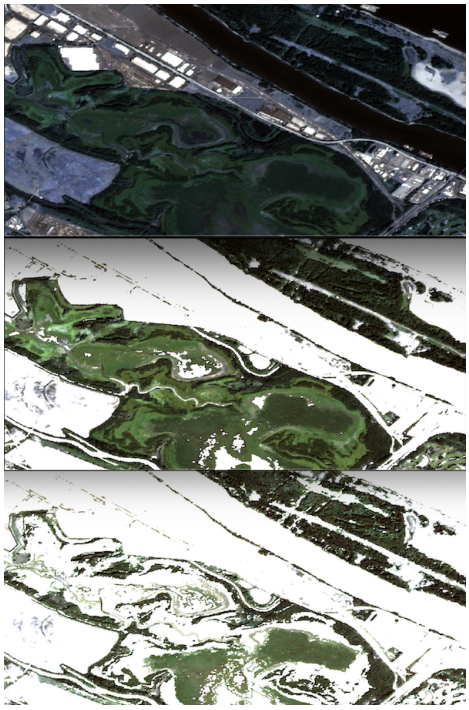

Figure 1.8: Satellite view of Smith Lake (top), after NDVI mask applied (middle), and NDVI mask and GRASS mask applied (bottom). The final masked image still contains parts of the lake because the aquatic vegetation fits the pixel values descriptions set by the masks.

# Chapter 2

# Methods

## 2.1 Current Methods

Since the Western redcedar decline is recent, there are currently no published work for identifying tree species like Western redcedars for the Pacific Northwest using satellite imaging. However, a similar study identifies tree species for a strip of land in California by combining ground-level data with satellite images (Fricker et al., 2019). Their methods involved drawing polygons around rastered images that are layered with ground data as explained in Chapter 1, but the pixels from the polygons are used to train a convolutional neural network (CNN) model instead of the random forests and support vector machines in this study. The CNN model was then evaluated with k-fold cross validation. CNN models are appropriate in a classification setting and when working with satellite images because they account for spatial relations, which is likely to appear in classifying tree species. This study follows the methods of preparing the data for modelling as well as the cross validation method to evaluate the performance of the random forests and support vector machines models. Random forests use random feature selections to create decision trees and increase the number of correctly classified observations. Random forests have the advantage of being simplier to train and still performs at a similar level as CNN models.

### 2.1.1 Random Forest

In building a decision tree for classification, recursive binary splits are made by minimizing the Gini index ($G$), the total variance across $K$ classes:

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}),$$

where $\hat{p}_{mk}$ is the proportion of observations in the $m^{\text{th}}$ region from the $k^{\text{th}}$ class. A small Gini index value represents a region containing mostly observations from a single class, while a large Gini index value represents a lot of variation within classes. Then the decision tree is constructed by repeatedly considering different attributes and making splits where the Gini index is minimized. Note that this top-down, greedy

approach tends to overfit the training data, so finding the decision tree that performs the best on test data involves a cost-complexity pruning algorithm to obtain smaller trees and apply k-fold cross validation to choose the best tree that minimizes the average error.

In the classification setting, a stronger predictive model than a decision tree is a random forest, which uses an element of randomness to decorrelate bootstrapped decision trees. Bootstrapped samples from the training dataset are used to construct decision trees, where splits are based on minimizing the Gini index when selecting from a random sample $m$ of attributes from all $p$ available. In common random forest applications, the number of attributes considered at a split is $m \approx \sqrt{p}$. Theoretically, by forcing only a subset of attributes to be featured in the tree, this expands the number of possible subtrees that might not have been achieved by the top-down, greedy approach. Hence, random forests provide a method of predicting classes with low variance while also keeping bias at a minimum.

### 2.1.2 Support Vector Machine

Another method investigated in this study imagines the multidimensional data in space and classifies the data using hyperplanes positioned to minimize misclassification error and maximize distance from observations to separate the data. Support vectors are vectors in space that represent the subset of observations which influence the hyperplane classifier. For example, in two dimensions, imagine a line separating A's one one side and B's on the other side. This line is the maximal margin hyperplane if it is the farthest possible from the observations and also separates the two classes perfectly. Observations that are closer to the hyperplane (support vectors) have more influence over the line if they move than observations that are further. Consequently, more support vectors indicates a classifier with lower variance and high bias. A tuning parameter can be altered to expand or shrink the margin of error surrounding the hyperplane (how far away the observations have to be from the plane) and decide the level of tolerance for misclassifying observations.

Support vector machines can be applied in a linear or nonlinear setting with the use of kernels. A kernel is a function $K(x_i, x_{i'})$ of two observations specifying the similarities between two points. Given data with $n$ observations, let $x_i$ be an observation, where $i = 1, ..., n$. A linear kernel is

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

with a support vector classifier

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle.$$

The parameters $\alpha_i$ and $\beta_0$ are estimated using the inner products of the observations, but $\alpha_i \neq 0$ if and only if $x_i$ is a support vector. Note that this function is

similar to a linear regression function. Extending this model to a nonlinear context involves a similar approach as adding nonlinear terms to a regression model.

The polynomial kernel of degree $d$ is defined to be

$$K(x_i, x_{i'}) = \left(1 + \sum_{j=1}^{p} x_{ij} x_{i'j}\right)^d$$

with support vector classifier

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i K(x, x_i).$$

For some positive constant $k$, the radial kernel is defined to be

$$K(x_i, x_{i'}) = \exp\left(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2\right),$$

which makes the support vector machine more sensitive to nearby observations as opposed to further observations.

Support vector machines are less sensitive to outliers and successful predictors of categorical variables in high-dimensional data. A support vector machine does not perform well for data with observations that overlap and if the kernel is incorrectly selected.

## 2.2 Training Models

To model the tree type of the pixels in raster image, a random forest model and a support vector machine model are trained using the pixels training data. First the data is separated into a training set and then a test set by a ratio of 70% training data and 30% testing data. The `dplyr` function `slice_sample` was used to take a stratified sample from the entire pixels dataset where 70% of each tree species in an individual raster strip was randomly selected. For example, 70% of the bigleaf maple trees in raster strip 'A' were randomly selected for the training set. To train these models, k-fold cross validation is used. The data gets split into $k = 10$ folds and ten models are trained each time withholding one of the ten folds until the process goes through every fold. This process aids in understanding how the models will perform against data that has not been included in training the model.

The selected variables for training the models are the four bands (red, green, blue, infrared) and an NDVI variable, and the predictive variable is the tree species name. Some predictor variables were manually computed using ratios of bands. Typically, a long over short band ratio provides some more information to train the model. The following bands were created: red/blue, ir/red, red/green, and blue/green. Examining a correlation matrix with these predictors reveals possibly useful and not too highly correlated variables are red/green, red/blue, and blue/green (see Figure 2.1.

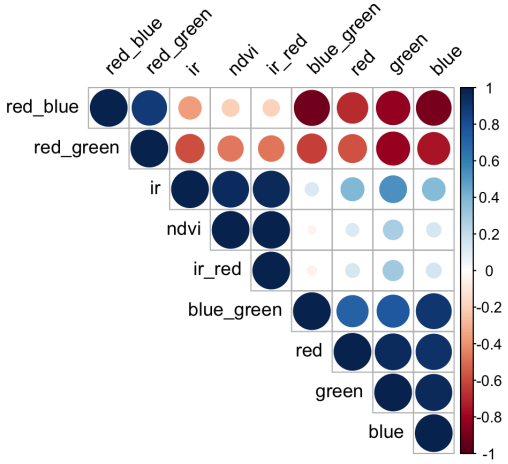


Figure 2.1: Correlation matrix of possible predictor variables to include in the model

The `caret` package (Kuhn, 2020) in RStudio stands for classification and regression training, and it contains functions for facilitating the process of creating training models in R and determining the tuning parameters.

### 2.2.1 Random Forest Tuning

The `train` function from the `caret` package (Kuhn, 2020) with the specified `method = rf` option is used to train a random forest model on the training dataset to predict the tree species. The function takes parameters for data, the predictive variable, and the method of training the model. To perform k-fold cross validation, the `trainControl` function is used to create an object that tells the `train` function to perform cross validation 10 times with a different fold left out each time. The function automatically tries multiple number of tries at each split (mtry) and determines the optimal number through cross-validation. Three different models were considered with varying number of predictors:

- 5 predictor model: red, green, blue, infrared, NDVI
- 7 predictor model: red, green, blue, infrared, NDVI, $\frac{\text{red}}{\text{green}}$, $\frac{\text{blue}}{\text{green}}$
- 8 predictor model: red, green, blue, infrared, NDVI, $\frac{\text{red}}{\text{green}}$, $\frac{\text{red}}{\text{blue}}$, $\frac{\text{blue}}{\text{green}}$

#### 5 predictor Random Forest

Figure 2.2 displays the accuracy of the trained random forest model for three different numbers of randomly selected predictors over 10 repeated cross-validation sets for a

model with 5 predictors: red, green, blue, infrared, and NDVI band values. This model had the highest overall accuracy with 0.53 when mtry was 1. The variable with the highest predictive power was infrared band values followed by NDVI values.

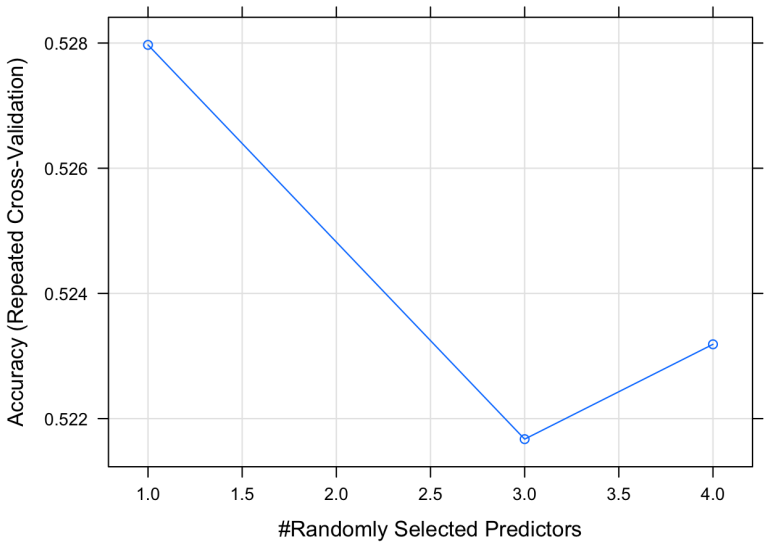


Figure 2.2: Accuracy of the 5 predictor trained random forest model for different mtry numbers over 10-fold cross-validation.

### 7 predictor Random Forest

The 7 predictor random forest model uses the same predictors from the 5 predictor model and also includes $\frac{\text{red}}{\text{green}}$ and $\frac{\text{blue}}{\text{green}}$ predictors. Figure 2.3 displays the accuracy of the model with a range of mtry values. The 7 predictor model with the highest accuracy (determined by cross-validation) randomly selected 2 predictors at each split with an accuracy of 0.54. The variable with the highest predictive power was still infrared band values with NDVI values second highest predictive power.

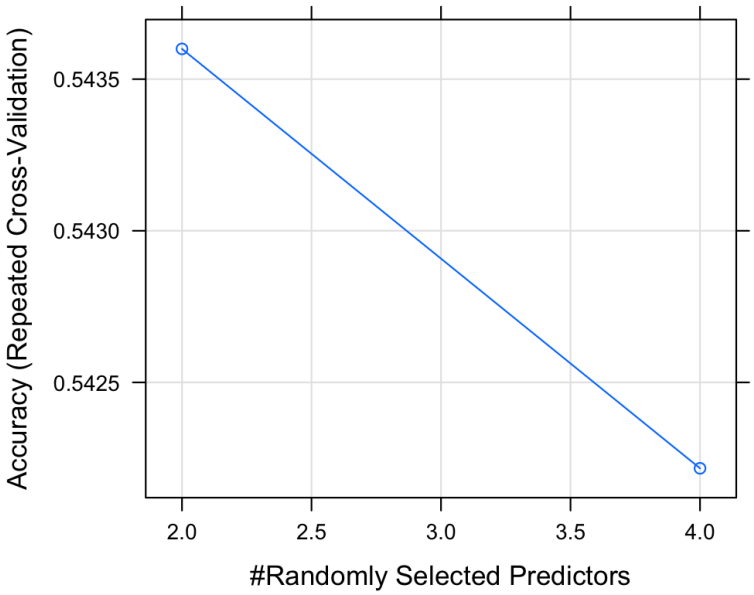


Figure 2.3: Accuracy of the 7 predictor trained random forest model for different mtry numbers over 10-fold cross-validation.

### 8 predictor Random Forest

The 8 predictor random forest model added a $\frac{\text{blue}}{\text{green}}$ predictor to the 7 predictor model. Figure 2.4 displays the results of a different number of randomly selected predictors to be considered at each split in the tree. The optimal mtry value was 4 with an accuracy of 0.55. The variables with the higher predictive power in decreasing order were infrared, NDVI, and green band values predictors.

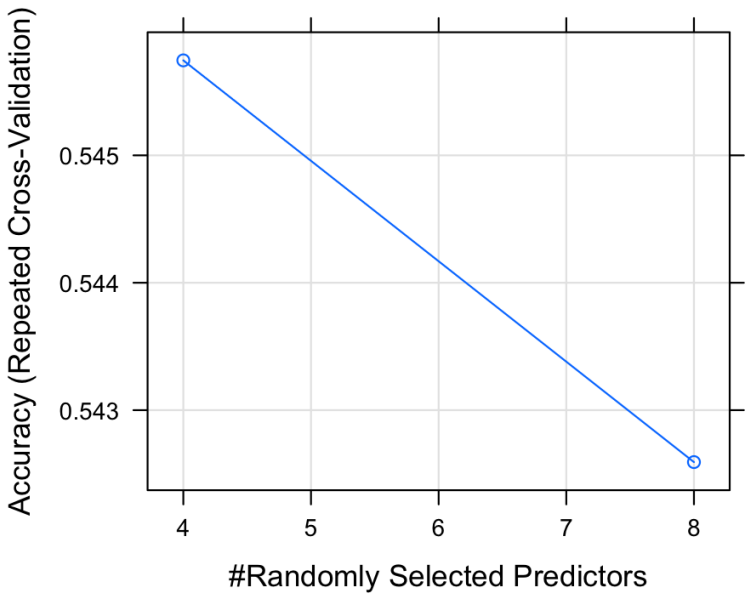


Figure 2.4: Accuracy of the 8 predictor trained random forest model for different mtry numbers over 10-fold cross-validation.

### Best Random Forest

Figure 2.5 compares the results of the models based on overall model accuracy in predicting the test data.

Table 2.1: Accuracy of random forest models with different number of predictors and mtry determined by ten-fold cross validation.

| Random Forest Model | Accuracy | Kappa | mtry |
|---|---|---|---|
| 5 predictors | 0.5280 | 0.4251 | 1 |
| 7 predictors | 0.5436 | 0.4449 | 2 |
| 8 predictors | 0.5457 | 0.4479 | 4 |

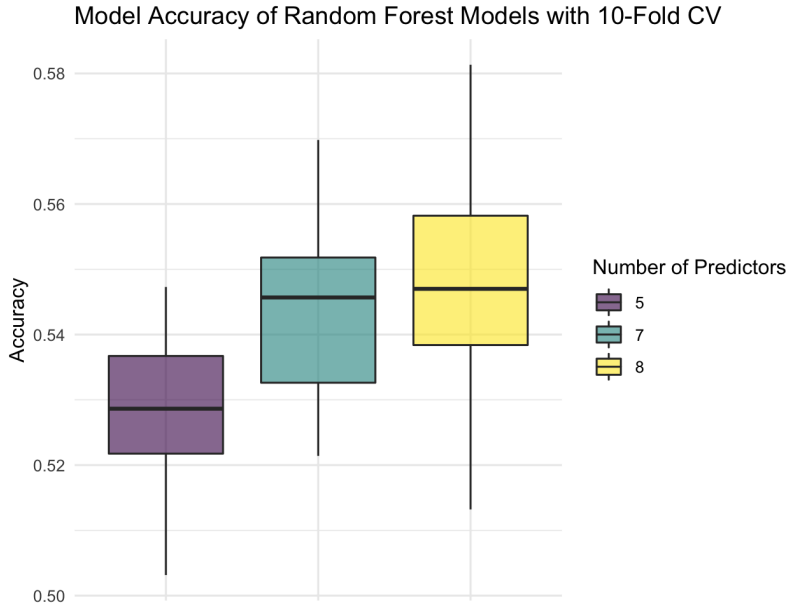


Figure 2.5: Comparison of results from random forest models with different predictors included

The model with the highest number of predictors included has a slightly higher median accuracy and mean accuracy than the 7 predictor model. A confusion matrix was constructed for each model averaging the entry counts over all ten cross validation resamples to investigate model performance just predicting Western redcedar tree pixels. Both the 7 and the 8 predictor models performed equally classifying Western redceder pixels, the 7 predictor model and the 8 predictor model had a prediction accuracy of 38%, while the 5 predictor model had 35%. Across all three models, Western redcedars tend to be inaccurately classified under douglas-firs, english oak, and norway maple trees.

Table 2.1 compares the hyperparameters accross all random forest models.

### 2.2.2 Support Vector Machine Tuning

The `train` function has the `method = svmLinear/svmRadial/svmPoly` option to train a support vector machine model on the dataset to classify pixels into species of trees. This option also has parameters for cost, loss function, class weights, and

normalized variables. For this project, three support vector machines were trained: linear, radial basis, and polynomial basis, and all three were specified to normalize the variables. The `caret` package selects the best cost tuning parameter based on accuracy through cross-validation.

### Linear Support Vector Machine

The linear kernel SVM model had the highest training accuracy at 0.51 when cost parameter held constant at a value of 1. The final model contained 5915 support vectors.

### Radial Support Vector Machine

Figure 2.6 displays the results of different cost parameters on training accuracy for the radial basis function kernel. The radial kernel SVM model performed best when cost was 16 and $\sigma = 0.25$ for a training accuracy of 0.56. The final model had 5587 support vectors.

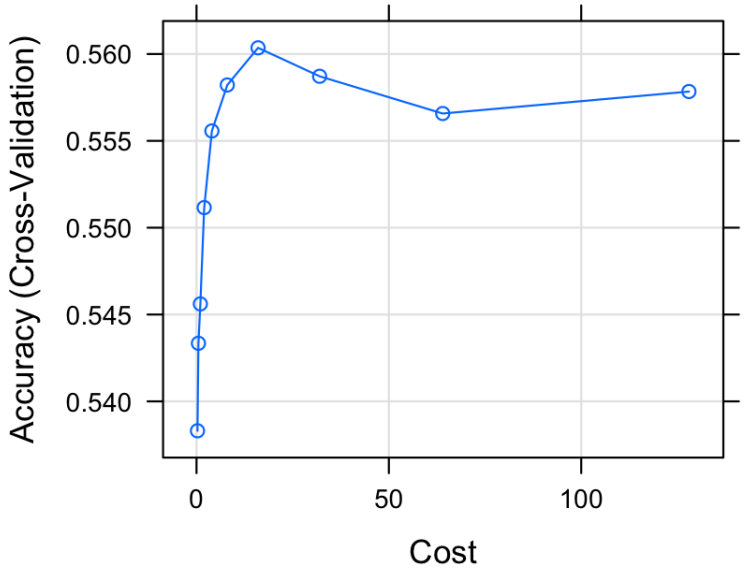


Figure 2.6: Comparison of accuracy results from different cost parameters with sigma held constant at 0.25 for radial basis function kernel.

### Polynomial Support Vector Machine

Figure 2.7 displays the results of different cost parameters on training accuracy for the polynomial basis function kernel. The final values used for the model were degree = 3, scale = 0.1, and cost = 1, which corresponded to a training accuracy of 0.54. The final model had a total of 5697 support vectors.

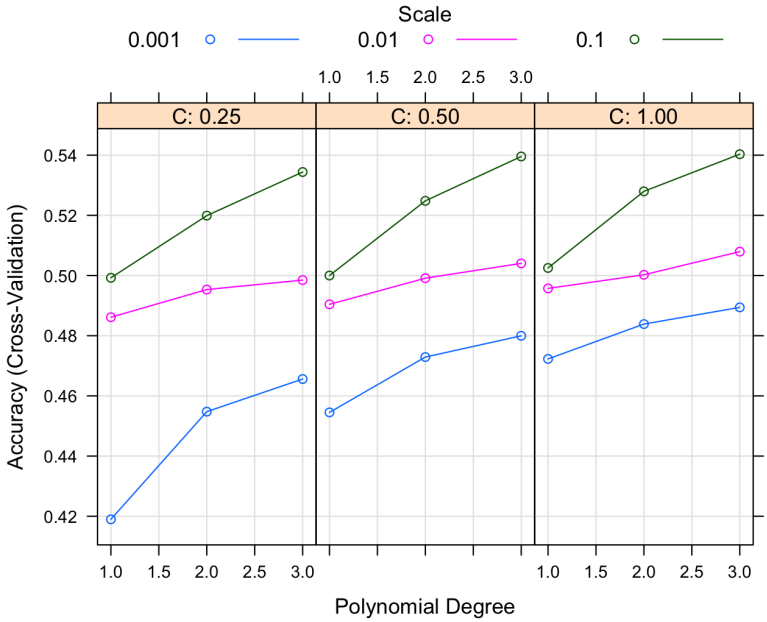


Figure 2.7: Comparison of accuracy results from different cost parameters with scale ranging from 0.001 to 0.1, C from 0.25 to 1, and degree from 1 to 3 for polynomial SVM.

### Best Support Vector Machine

Figure 2.8 displays the results of the linear, radial basis, and polynomial basis support vector machine models. Based on the overall prediction accuracy, the radial basis is an appropriate kernel choice for the pixels data since it has a higher median model accuracy than the other two kernel options.

Table 2.2: Accuracy of support vector machine models with different kernels. Best parameter is determined by ten-fold cross validation.

| SVM Model Kernel | Accuracy | Kappa | Parameter |
|---|---|---|---|
| Linear | 0.5053 | 0.3843 | C = 1 |
| Polynomial | 0.5403 | 0.4322 | C = 16 sigma = 0.2483 |
| Radial | 0.5604 | 0.4614 | C = 1 , scale = 0.1 , degree = 3 |

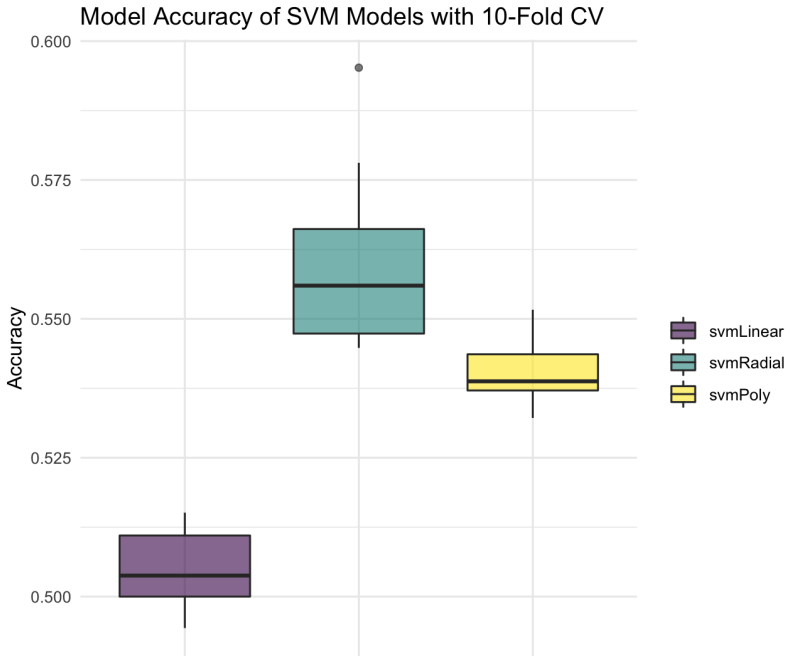


Figure 2.8: Comparison of results from support vector machine models with different kernel types: linear, radial, and polynomial.

As with the random forest model comparisons, a confusion matrix was constructed using the average counts over all ten cross-validation resamples to obtain the model results predicting Western redcedars. The support vector machine model that performed best overall was the radial basis kernel model, and it had the highest prediction accuracy of the Western redcedar class, 41%. The polynomial basis kernel had a 34% prediction accuracy and the linear kernel had 32% accuracy for Western redcedar.

Table 2.2 compares the hyperparameters accross all support vector machine models. The models are saved to be used for later prediction over the entire region of Portland.

## 2.3 Testing Models

The best models from the random forest models and the support vector machine models were used to predict the tree type on the test dataset. The results for the

Table 2.3: Overall accuracy of 8 predictor RF model compared to Radial SVM model on test data

| RF Accuracy | SVM Accuracy |
|---|---|
| 0.5472637 | 0.5604331 |

Table 2.4: Western redcedar accuracy of 8 predictor RF model compared to Radial SVM model on test data

| RF Accuracy | SVM Accuracy |
|---|---|
| 0.3430657 | 0.3260341 |

overall accuracy and the prediction accuracy for Western redcedars are shown in Table 2.4. The polygon prediction accuracy on the test dataset was also computed to see how pixels of the same tree were classified compared to individual pixel classification. A "correct" prediction was indicated if more than half the pixels in a polygon are correctly predicted. Table 2.6 displays the results over polygons in the test dataset.

Table 2.5: Accuracy of 8 predictor random forest model for polygons in test data. A 'correct' prediction is considered to be a polygon with more than half of the pixels correctly classified.

| result | n |
|---|---|
| correct | 205 |
| incorrect | 444 |

Table 2.6: Accuracy of radial support vector machine model for polygons in test data. A 'correct' prediction is considered to be a polygon with more than half of the pixels correctly classified.

| result | n |
|---|---|
| correct | 233 |
| incorrect | 416 |

# Chapter 3

# Results

## 3.1 Training Results

The best hyperparameters settings for the random forest training model were an mtry value of 4 predictors at each split and the following predictors: red, green, blue, infrared, NDVI, $\frac{\text{red}}{\text{green}}$, $\frac{\text{red}}{\text{blue}}$, $\frac{\text{blue}}{\text{green}}$. This model had a training predictive accuracy of 0.55 overall and 0.38 for Western redcedars specifically. On the test dataset, this model still accurately predicted 55% of the pixels, while the accuracy for predicting more than half the pixels correctly in the same polygon was 32%. The selected hyperparameter for the support vector machine model was a radial basis kernel with cost equal to 16 and $\sigma = 0.25$. The radial support vector machine had a training accuracy of 0.56 overall, 0.41 for just Western redcedars, and testing accuracy of 0.56 overall and 0.32 for Western redcedar pixels. For polygon predictions, the radial support vector machine had an accuracy of 36%. Between the random forest model and the support vector machine model, the radial basis kernel support vector machine model slightly outperforms the random forest 8 predictor model in overall accuracy, but the random forest has a slightly higher performance in predicting Western redcedars. Since these models produced similar results, both models are used to proceed with the final model over Portland.

## 3.2 Modelling Tree Species in Portland

After masking the raster images of the entire region of Portland, the pre-trained models are used to predict the tree species of individual pixels in the masked raster images. The `predict.raster` function from RStudio's `raster` package takes a model and applying that model to each cell of the raster, leaving a raster with tree classification predictions. This function is used to predict tree species over all five raster strips, and the raster strips are merged to display predictions for the entire city. Figure 3.1 displays the tree classification predictions given by the optimal random forest model over Portland and figure 3.2 displays the predictions from the support vector machine model.

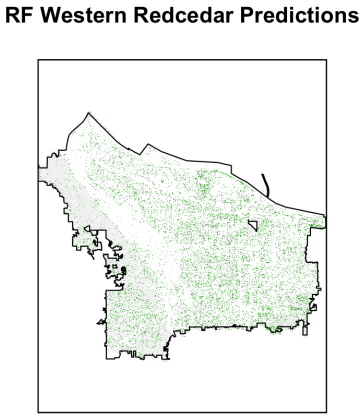


Figure 3.1: Model tree classification predictions over entire Portland region using the random forest model with 8 predictors.

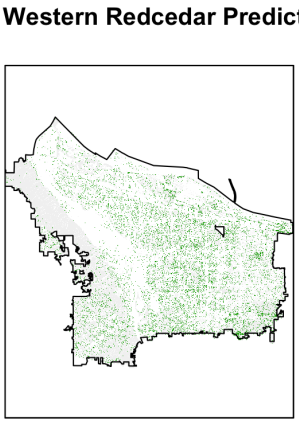


Figure 3.2: Model tree classification predictions over entire Portland region using the support vector machine model with radial basis kernel.

To assess the performance of the model over the Portland region, the results are compared to the remaining Western redcedars from `pdxTrees` both street and park data. The shapefiles for `pdxTrees` Western redcedars ground data are mapped over the classified pixel predictions, and the number of correctly and incorrectly classified pixels is recorded. Figure 3.3 visualizes this comparison.

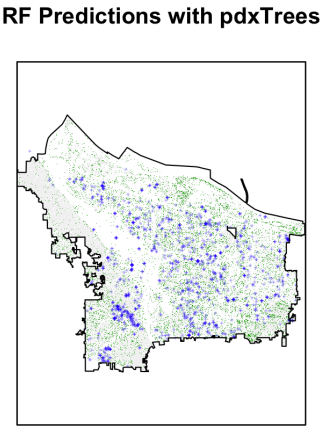


Figure 3.3: Comparison of Wester redcedar RF pixel predictions (in green) to Western redcedar trees from pdxTrees (in blue).

# Appendix A

# The First Appendix

This first appendix includes all of the R chunks of code that were hidden throughout the document (using the `include = FALSE` chunk tag) to help with readibility and/or setup.

**In the main Rmd file**

```
# This chunk ensures that the thesisdown package is
# installed and loaded. This thesisdown package includes
# the template files for the thesis.
if(!require(devtools))
  install.packages("devtools", repos = "http://cran.rstudio.com")
if(!require(thesisdown))
  devtools::install_github("ismayc/thesisdown")
library(thesisdown)
```

**In Chapter ??:**

```
# This chunk ensures that the thesisdown package is
# installed and loaded. This thesisdown package includes
# the template files for the thesis and also two functions
# used for labeling and referencing
if (!require(remotes)) {
  if (params$`Install needed packages for {thesisdown}`) {
    install.packages("remotes", repos = "https://cran.rstudio.com")
  } else {
    stop(
      paste(
        'You need to run install.packages("remotes")',
        "first in the Console."
      )
    )
  }
}
```

```
if (!require(dplyr)) {
  if (params$`Install needed packages for {thesisdown}`) {
    install.packages("dplyr", repos = "https://cran.rstudio.com")
  } else {
    stop(
      paste(
        'You need to run install.packages("dplyr")',
        "first in the Console."
      )
    )
  }
}
if (!require(ggplot2)) {
  if (params$`Install needed packages for {thesisdown}`) {
    install.packages("ggplot2", repos = "https://cran.rstudio.com")
  } else {
    stop(
      paste(
        'You need to run install.packages("ggplot2")',
        "first in the Console."
      )
    )
  }
}
if (!require(bookdown)) {
  if (params$`Install needed packages for {thesisdown}`) {
    install.packages("bookdown", repos = "https://cran.rstudio.com")
  } else {
    stop(
      paste(
        'You need to run install.packages("bookdown")',
        "first in the Console."
      )
    )
  }
}
if (!require(thesisdown)) {
  if (params$`Install needed packages for {thesisdown}`) {
    remotes::install_github("ismayc/thesisdown")
  } else {
    stop(
      paste(
        "You need to run",
        'remotes::install_github("ismayc/thesisdown")',
```

```
        "first in the Console."
      )
    )
  }
}
library(thesisdown)
library(dplyr)
library(ggplot2)
library(knitr)
flights <- read.csv("data/flights.csv", stringsAsFactors = FALSE)
```

# Appendix B

# The Second Appendix, for Fun

# References


Castelluccio, M., Poggi, G., Sansone, C., & Verdoliva, L. (2015). Land Use Classification in Remote Sensing Images by Convolutional Neural Networks. *arXiv:1508.00092 [Cs]*. Retrieved from http://arxiv.org/abs/1508.00092

Fricker, G. A., Ventura, J. D., Wolf, J. A., North, M. P., Davis, F. W., & Franklin, J. (2019). A Convolutional Neural Network Classifier Identifies Tree Species in Mixed-Conifer Forest from Hyperspectral Imagery. *Remote Sensing*, *11*(19), 2326. http://doi.org/10.3390/rs11192326

Kuhn, M. (2020). *Caret: Classification and regression training*. Retrieved from https://CRAN.R-project.org/package=caret

Peterson, J. S. (n.d.). WESTERN RED CEDAR, 3.

Qian, Y., Zhou, W., Nytch, C. J., Han, L., & Li, Z. (2020). A new index to differentiate tree and grass based on high resolution image and object-based methods. *Urban Forestry & Urban Greening*, *53*, 126661. http://doi.org/10.1016/j.ufug.2020.126661

Western red cedar. (n.d.). Retrieved from https://www.oregonencyclopedia.org/articles/western_red_cedar/#.YB7fUy1h1N0

Western redcedar Dieback. (n.d.). *PPO Home*. Retrieved from https://ppo.puyallup.wsu.edu/plant-health-concerns/redcedar/