Supervised Learning Classification Comparison on Solar Panels

Abstract

The purpose of this paper is to evaluate the supervised classification accuracy of two different machine learning algorithms for classifying solar panels in remotely sensed imagery. Machine learning algorithms have been proven useful in classification of imagery and in this context the intention is to

evaluate their usefulness at identifying solar panels located on residential roofs allowing for the estimation of photovoltaic energy produced at a neighborhood scale. For this application R packages 'raster' and 'e1071' are used. The orthoimage is a scene in the Miramar neighborhood of San Diego taken on a cloudless day in 2008. The neighborhood was chosen because of the high likelihood of solar adoption amongst residents due to favorable economic and environmental factors providing a suitable number of solar panels to use for training and classification. The results show that there is a small difference between the two algorithms classification performance. The classification algorithms proved to be useful starting points for such an analysis although further work is required to truly create an index for measuring the extent of solar panel adoption in a community.

Introduction

There has been immense growth in the amount of remotely sensed imagery available from several different satellites at varying resolutions. High resolution satellite and aerial images have enabled us to observe the earth and its phenomena from a high level. The exponential growth in satellites and available imagery necessitate a computational approach to derive information from the data produced. The growth in machine learning algorithms and computer vision in conjunction with the amount of available imagery means that there are endless novel applications for object detection from imagery.

Due to financial and time constraints, this paper will focus primarily on machine learning applications to object detection, namely solar panels. Solar energy derived from residential photo voltaic sources are widely considered a vital part of a clean energy future. Metrics are few and far between for calculating residential solar adoption. Overall statistics produced at the national and state level include the industrial and commercial grade solar installations as well. A more telling look at our solar energy adoption rate is the per capita figure at the national level. The US ranks 19th per capita of solar adoption rates in the world [1]. Introducing a raw method to determine the level of PV adoption rates at the municipality level could help communities' better position themselves for federal and state level grants for renewable energy technologies.

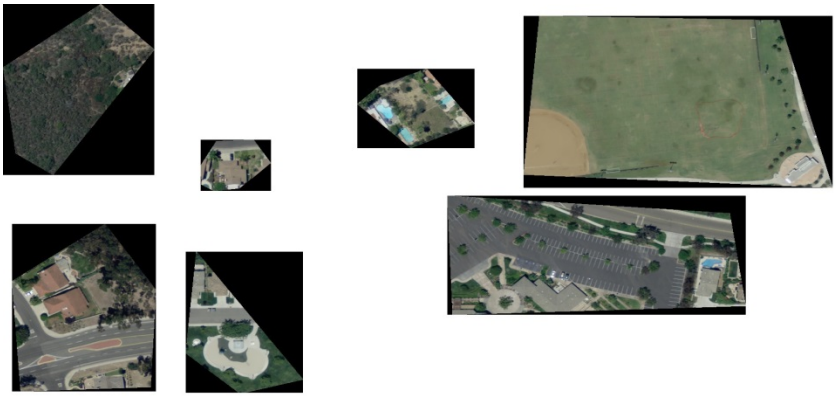
Other potential uses for this information include assessment of risk for personal and commercial lines of insurance. Insurance underwriters need to know the underlying risk for fire departments responding to a fire at a location equipped with solar panels. A house with solar panels requires a different approach in a structure fire emergency than those with traditional roofing. The presence of solar panels fundamentally changes the way that firefighters will approach a structure fire. The National Fire Protection Agency has produced papers regarding the risk firefighter's face and the approaches required to ensure their safety. Due to the relative newness of the technology, there is little data available to the public and emergency responders about the installation and operation of solar panels ranging from small residential to large commercial systems. The work produced here can help create a baseline from which first responding agencies can use to determine presence of solar (photovoltaic or solar thermal energy) [2].

Supervised classification has been one of the widest used classification schemes for satellite and aerial imagery for almost two decades. Literature on remote sensing suggests many favorable results of supervised classification techniques to classify remotely sensed imagery. Methods used for classification and object detection include Support Vector Machines, Random Forest, Maximum Likelihood Classification, Multinomial Logistic Regression and Neural Networks [3]. Supervised learning techniques all require creation of training sets of dependent and independent variables. In remote sensing, each object on the ground has a unique spectral signature that can be used to help segment and ultimately classify different features within images. The spectral signature is defined as the average RGB values for each object intended for classification. The mean RGB values for the solar class are 84, 91 and 106 respectively. The class of non-solar panels signature spans the entire 1-255 spectrum while the solar class inherits a very narrow range which will help differentiate it. This can also present itself as a problem since the non-solar class will span through the solar class which is intended to be isolated. Other dark features may present themselves in the solar class such as shadows and new roads.

Training Samples were digitized through the open source image manipulation software GIMP. Pixels from rooftops which have solar panels have been extracted and placed in a .JPG file to be used as training samples. The non-solar training sample images were cut out and placed into their own .JPG file. The samples used are pictured below. The training samples were turned into data frames to hold their RGB values, black and white background pixels introduced during digitization were removed from both images.

Figure 1a. *Solar Panel Training Sample*

Figure 1b. *Non Solar Panel Training Sample*



Code w/ Documentation

Load and plot the scanned map

```
setwd("C:/Users/i52025/Documents/DataScienceCertificateProgram/MachineLearning/ResearchProject")
require(raster)
```

```
## Loading required package: raster
```

```
## Warning: package 'raster' was built under R version 3.2.5
```

```
## Loading required package: sp
```

```
## Warning: package 'sp' was built under R version 3.2.5
```

```
require(e1071)
```

```
## Loading required package: e1071
## Attaching package: 'e1071'
```

```
## The following object is masked from 'package:raster':
##
##     interpolate
```

```
require(rasclass)
```

```
## Loading required package: rasclass
```

```
## Warning: package 'rasclass' was built under R version 3.2.5

##
## Attaching package: 'rasclass'

## The following object is masked from 'package:raster':
##
##     writeRaster

miramar2008 <- brick("Miramar2008.tif")

plotRGB(miramar2008)
```

```
# training and non training
training.panels.img <- brick("Solar.tif")
training.nonpanels.img <- brick("nonSolarPanels.jpg")
```

Put background data into data frame

```
training.panels.df <- data.frame(getValues(training.panels.img))
names(training.panels.df) <- c("r", "g", "b")

# Remove black and white background pixels
# Create new variable indicating pipeline pixels

training.panels.df$Panel <- "1"
training.panels.df <- training.panels.df[(training.panels.df$r > 1 & training.pan
els.df$g > 1 & training.panels.df$b > 1),]
```

```
training.panels.df <- training.panels.df[(training.panels.df$r < 254 & training.p
anels.df$g < 254&training.panels.df$b < 254),]

# Put training data into data frame

training.nonpanels.df <- data.frame(getValues(training.nonpanels.img))
names(training.nonpanels.df) <- c("r", "g", "b")

# Remove white and black background pixels

training.nonpanels.df <- training.nonpanels.df[(training.nonpanels.df$r > 1 & tra
ining.nonpanels.df$g > 1 & training.nonpanels.df$b > 1),]
training.nonpanels.df <- training.nonpanels.df[(training.nonpanels.df$r < 254 & t
raining.nonpanels.df$g < 254 & training.nonpanels.df$b < 254),]

# Create new variable indicating non solar panel pixels
training.nonpanels.df$Panel <- "0"

#remove NA row
training.panels.df<- training.panels.df[, c(1:3,5)]

# Combine data frames and subset only 10000 random values from the non-pipeline t
raining data
training.df <- rbind(training.panels.df, training.nonpanels.df[sample(nrow(traini
ng.nonpanels.df), 10000),])
# Turn classification variable into factor
training.df$Panel <- as.factor(training.df$Panel)

attach(training.df)
# Divide training data into a train-subset and a test-subset
train <- sample(nrow(training.df), round((nrow(training.df) - 1) / 2, 0))
test <- c(1:nrow(training.df))[!(c(1:nrow(training.df)) %in% train)]
trainset.df <- training.df[train,]
testset.df <- training.df[test,]

# Fit best SVM using tuning

svm.fit <-  svm(Panel~., data = trainset.df, cost = 10)
svm.fit2 <- best.svm(Panel~., data = training.df, cost = 10)

# Fit predictions and print error matrix

require(caret)

## Loading required package: caret

## Warning: package 'caret' was built under R version 3.2.5

## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.2.5
```

```
svm.pred <- predict(svm.fit, testset.df[,1:3])
svm.tab <- table(pred = svm.pred, true = testset.df[,4])
confusionMatrix(svm.tab)
```

```
## Confusion Matrix and Statistics
##
##     true
## pred    0    1
##    0 4957   74
##    1   66  731
##
##                Accuracy : 0.976
##                  95% CI : (0.9717, 0.9798)
##     No Information Rate : 0.8619
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8987
##  Mcnemar's Test P-Value : 0.5541
##
##             Sensitivity : 0.9869
##             Specificity : 0.9081
##          Pos Pred Value : 0.9853
##          Neg Pred Value : 0.9172
##              Prevalence : 0.8619
##          Detection Rate : 0.8505
##    Detection Prevalence : 0.8632
##       Balanced Accuracy : 0.9475
##
##        'Positive' Class : 0
##
```

```
svm.pred2 <- predict(svm.fit2, trainset.df[,1:3])
svm.tab2 <- table(pred = svm.pred2, true = trainset.df[,4])
confusionMatrix(svm.tab2)
```

```
## Confusion Matrix and Statistics
##
##     true
## pred    0    1
##    0 4929   56
##    1   48  794
##
##                Accuracy : 0.9822
##                  95% CI : (0.9784, 0.9854)
##     No Information Rate : 0.8541
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9281
##  Mcnemar's Test P-Value : 0.4925
##
##             Sensitivity : 0.9904
##             Specificity : 0.9341
##          Pos Pred Value : 0.9888
```

```
##          Neg Pred Value : 0.9430
##              Prevalence : 0.8541
##          Detection Rate : 0.8459
##    Detection Prevalence : 0.8555
##       Balanced Accuracy : 0.9622
##
##        'Positive' Class : 0
##
```

```
# Prepare Miramar map for predictions
## miramar.df <- data.frame(getValues(miramar2008))
## (miramar.df) <- c("r", "g", "b")
# Assign predicted values to target map

## miramar.pred <- predict(svm.fit2, miramar.df)
## system.time(predict(svm.fit, miramar.df)) # optional to see stats on intense c
omputational processing
# Assign predicted values to target map

## classified.img <- miramar2008[[1:3]]
## values(classified.img) <- miramar.pred

## image(classified.img)
```

Random Forest Classification [4]

RFClassification

```
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
# Load and plot the scanned map
setwd("C:/Users/i52025/Documents/DataScienceCertificateProgram/MachineLearning/Re
searchProject")


require(e1071)
```

```
## Loading required package: e1071
```

```
##
## Attaching package: 'e1071'
```

```
## The following object is masked from 'package:raster':
##
##     interpolate
```

```
require(caret)
```

```
## Loading required package: caret

## Warning: package 'caret' was built under R version 3.2.5

## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.2.5

##
## Attaching package: 'ggplot2'

## The following object is masked from 'package:randomForest':
##
##     margin

miramar2008 <- brick("Miramar2008.tif")
```

Code duplicated to create the training dataset for random forest classifier has been removed for aesthetic reasons. Same test and training datasets were used here.

```
rfm_mira <- randomForest(Panel ~ ., trainset.df)
p_mira   <- predict(rfm_mira, testset.df)

confusionMatrix(table(testset.df[,4], p_mira))

## Confusion Matrix and Statistics
##
##    p_mira
##        0    1
##   0 4907   61
##   1   81  779
##
##                Accuracy : 0.9756
##                  95% CI : (0.9713, 0.9794)
##     No Information Rate : 0.8559
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9022
##  Mcnemar's Test P-Value : 0.1108
##
##             Sensitivity : 0.9838
##             Specificity : 0.9274
##          Pos Pred Value : 0.9877
##          Neg Pred Value : 0.9058
##              Prevalence : 0.8559
##          Detection Rate : 0.8420
##    Detection Prevalence : 0.8524
##       Balanced Accuracy : 0.9556
##
##        'Positive' Class : 0
##
```

```
mean(testset.df[,4]==p_mira)

## [1] 0.9756349

importance(rfm_mira)

##   MeanDecreaseGini
## r         342.5820
## g         353.6519
## b         619.9324

p_miramar   <- predict(rfm_mira, miramar.df)

rfMiramar.class <- ifelse(p_miramar == "1", 1, 0)

rfClassified.img <- miramar2008[[1]]
values(rfClassified.img) <- rfMiramar.class
#image(rfClassified.img)

writeRaster(rfClassified.img, "RandomForestResults1.tif")
```

The code above loaded the packages and images both training and the full scene which are used in the analysis. It then created training data sets out of the solar and non-solar .JPG files. The models are fitted to the training sets for the respective models and then the accuracy matrices are computed. The models are then used to predict the solar panels in the full scene and finally the resulting rasters are saved to disk for analysis in a GIS.

Each cell was given a 1 for in the solar panel class or a 0 for non-solar panel class. The ones are displayed in yellow below.

Discussion

At a first glance it looks like each of the algorithms classified the playground and the new roads in the northwestern portion of the scene as solar panels. The end results of the training algorithms displayed in *Figure 2a.* and *Figure 2b.* show that the algorithms did not perform as accurately as initially hoped.

Figure 2a. *Support Vector Machine Classification Results*

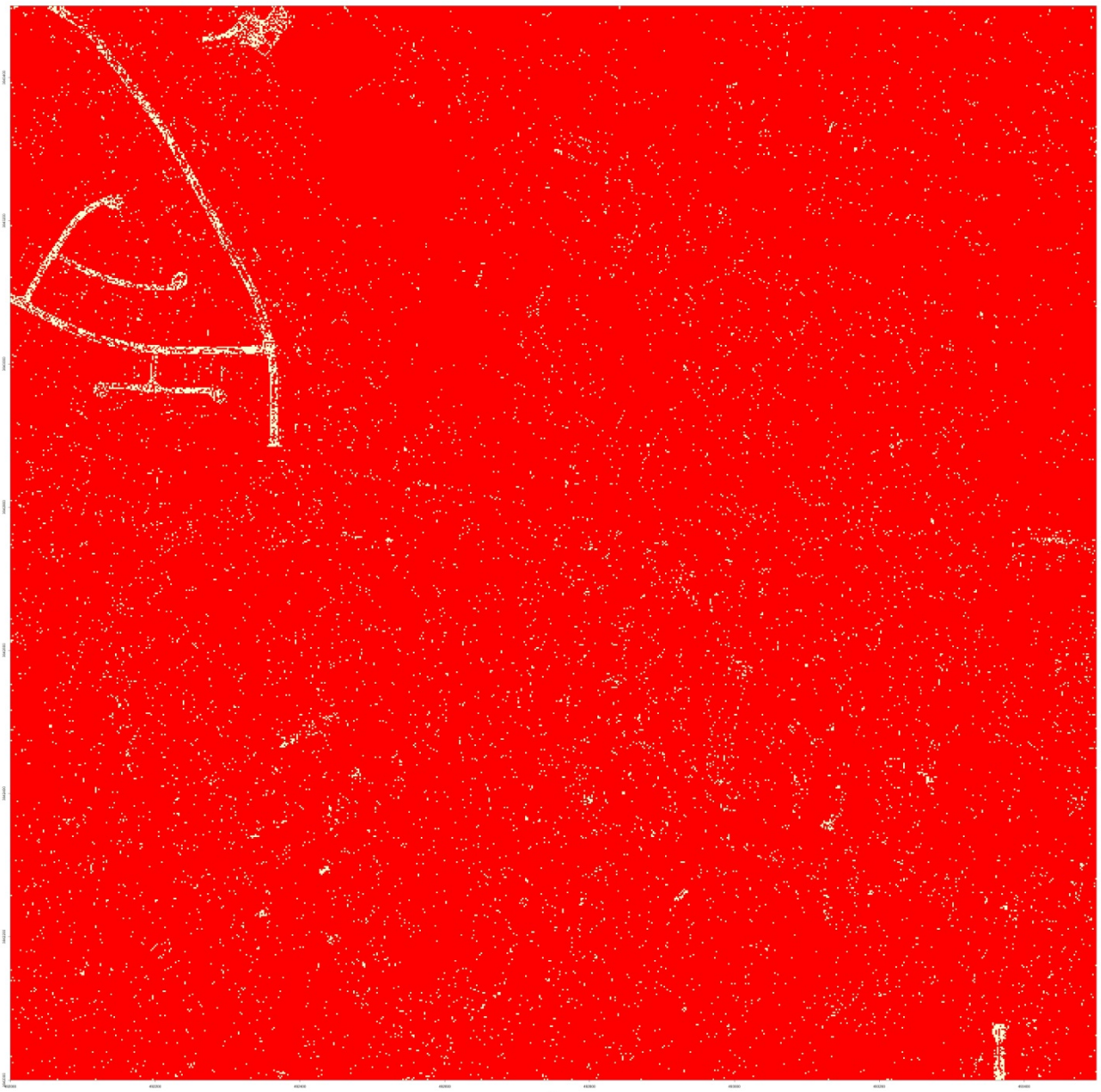
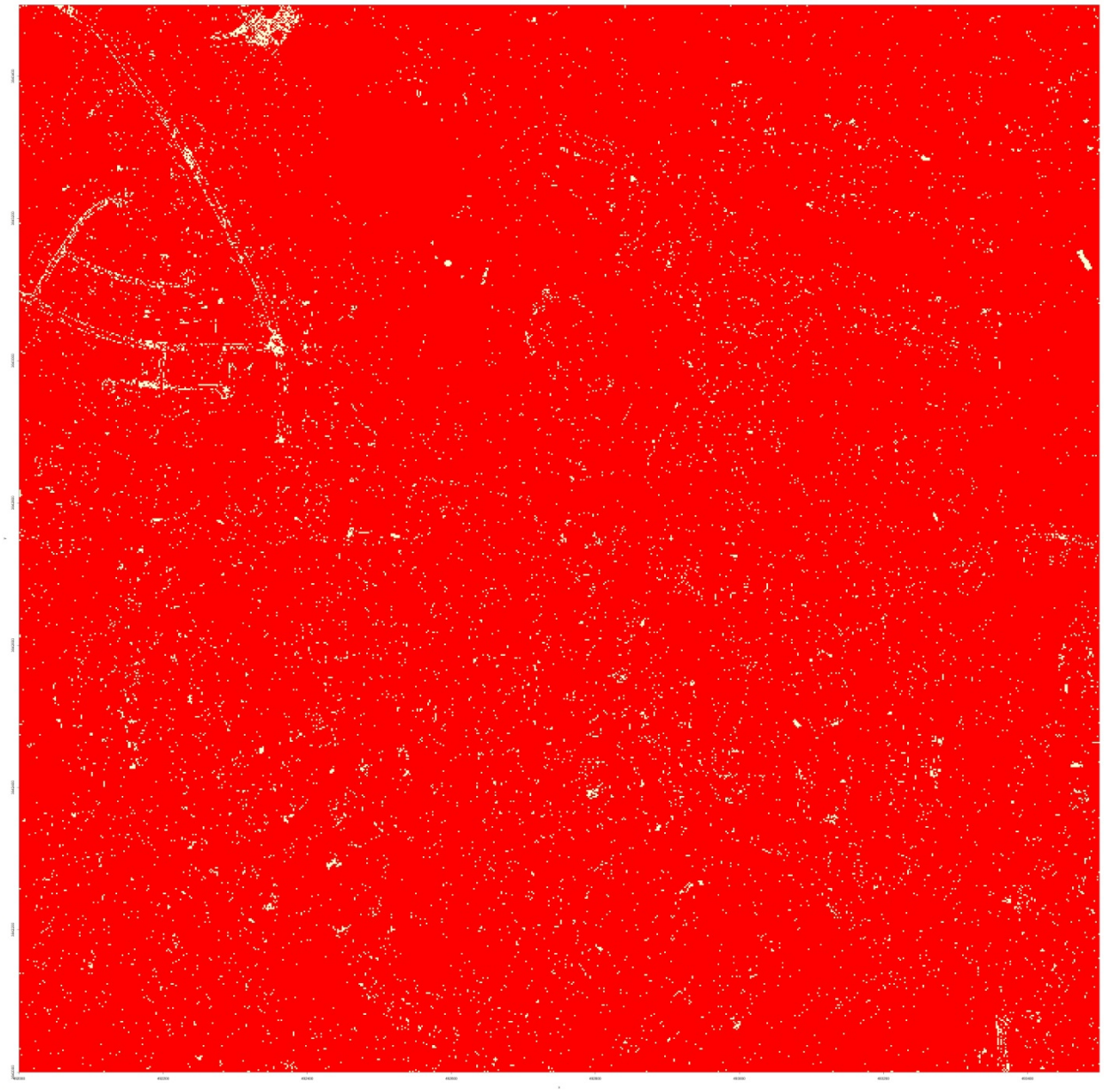


Figure 2b. *Random Forest Classification Results*



The Brieman random forest technique as well as the Support Vector Machine algorithm were performed in R to classify the solar panels in the image. The original data set was obtained through the USGS Earth Explorer data portal. The orthoimagery was taken from 9500 ft altitude and projected on the UTM 11N NAD83 stateplane coordinate system and has been georeferenced and corrected. The resolution of the imagery was 0.3 meters which proved crucial to be able to identify individual solar panels in the image.

The classifiers were not as accurate as their confusion matrices indicated from the training sets. The support vector machine confusion matrix on the training set listed the accuracy as 98.22 percent while the random forest dataset displayed an accuracy of 97.56 percent. The classifiers did perform admirably in identifying solar panels but there were many false positives such as cars and natural shadows cast from objects in the scene which were also picked up by both classifiers.

Results from the support vector machine algorithm showed that it could correctly predict the presence of solar panels but were also full of false positives, i.e. where the algorithm thought there to be solar panels present but were objects of another class. The pixels at the edges of newer roads where the classes were beginning to transition to other cover types were particularly sensitive to the support vector classifier. Roads have similar spectral signatures to the solar panels so in this instance it was more difficult to distinguish the two classes. Supervised classification is highly dependent upon the input data and the natural class distinctions to create an accurate decision boundaries. The results did capture solar panels, but were very liberal in the classification. This may be due to the Cost being quite low. A cost of 1000 may have made the classification criteria more rigid, enabling a better classification accuracy. Due to computational processing power restraints, I couldn't perform the algorithm on a vector of different costs. Each run of the svm took over an hour to predict on just a single class on the 25 million pixel image. These processes would be best run on a server or hardware with additional processing power and memory. Another contributor to the amount of false positives is the fact that the non-solar power training set contained the full 256 color spectrum for each red green and blue band while the solar training set only comprised of a narrow range of the three bands.

The random forest algorithm was less computationally intensive and but performed almost as well as the support vector machine algorithm. The algorithm correctly classified many panels in the image but also interestingly semi-classified solar panels. The random forest output found most of the panels but on many there were outlines of the panels that got classified rather than the full set of pixels that comprised the object in the image. The output also showed quite a few false positives including swimming pools and cars. *Figure 3c.* showed that the algorithm did correctly identify the pixels much like the svm classifier performed however the areas particularly on the west side of the image were outlines while the svm was more robust in classifiying the majority of the feature.

Generally these algorithms performed well but each of the models did suffer from overfitting, producing a high amount of false positives. The computational processing limitations prevented the svm model from receiving the best tuning and fit and instead resorted to having to estimate the correct parameters. These algorithms would work as an excellent starting point to this analysis. Alberto Bietti in 2012 showed that the support vector machine technique in conjunction with crowd sourced information from Amazon Mechanical Turk could accurately identify objects in remotely sensed imagery [5]. Computer vision algorithms which take into account the shapes of objects may also prove useful in this analysis as solar panels are overwhelmingly rectangular shaped. R does prove to be a capable tool in the processing and classification of satellite images in this application.

Figure 3a. *RGB Truecolor image in QGIS. This particular area of the image had a higher density of solar panels than average.*

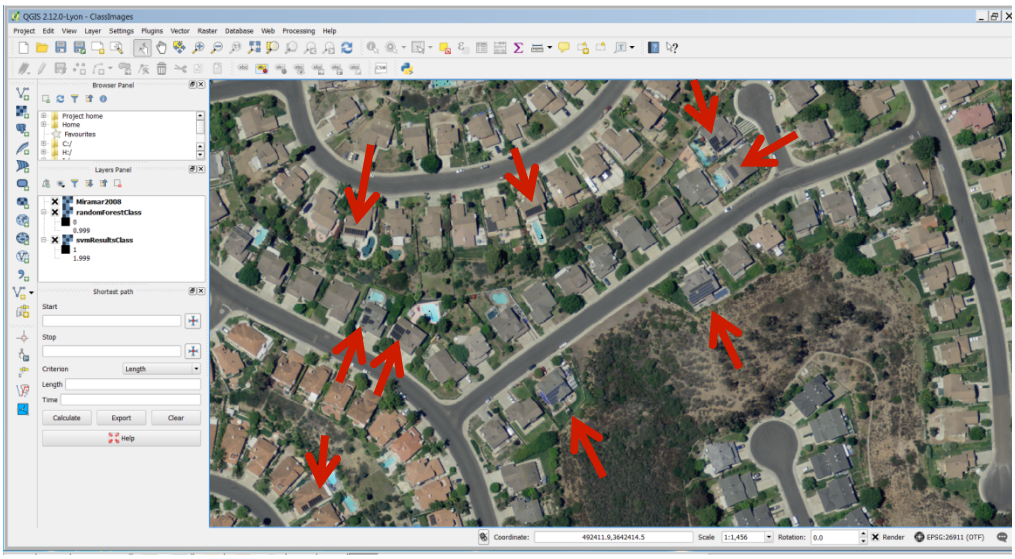


Figure 3b. *Random Forest Classification highlighted area*

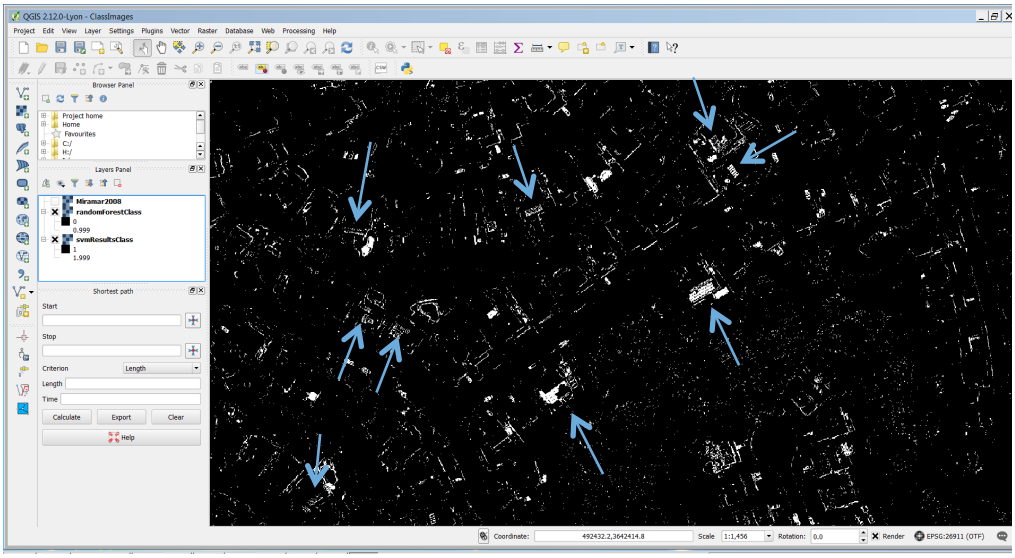
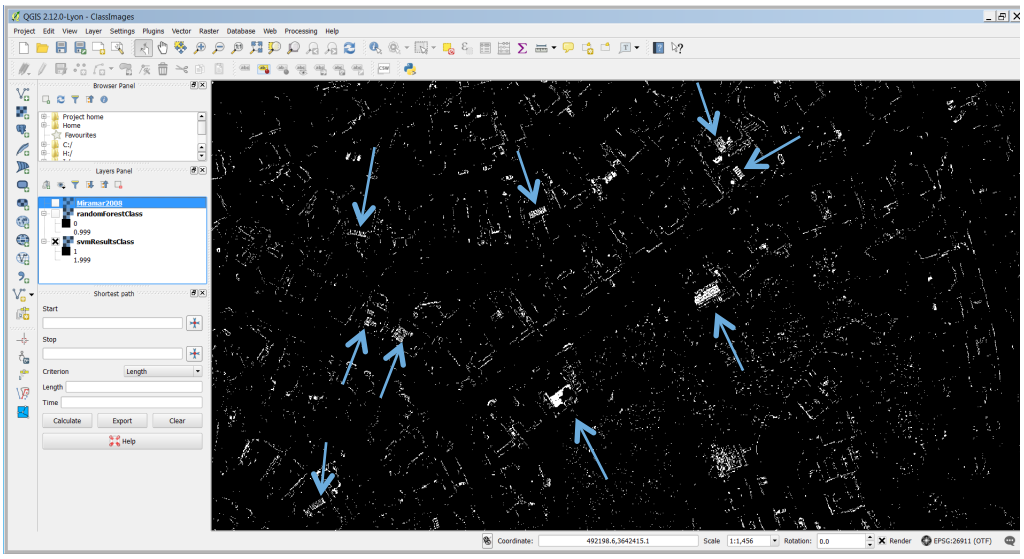


Figure 3c. *Support Vector Machine classification highlighted area*



References

1. US Solar Adoption Pretty Dismal, Closer Look Reveals. (2015, May 25). Retrieved from http://breakingenergy.com/2015/05/25/actually-the-us-is-a-dismal-solar-power-adopter/
2. Grant, C. C. (2010). *Fire Fighter Safety and Emergency Response for Solar Power Systems* (Tech.). Quincy, MA: National Fire Protection Association.
3. Wieland, M., & Pittore, M. (2014). Performance Evaluation of Machine Learning Algorithms for Urban Pattern Recognition from Multi-spectral Satellite Images. *Remote Sensing, 6*(4), 2912-2939. doi:10.3390/rs6042912
4. Hunziker, P. (2013, March 10). Supervised Image Classification in R Using Support Vector Machines. Retrieved from https://philipphunziker.wordpress.com/2013/03/10/supervised-image-classification-in-r-using-support-vector-machines/
5. Bietti, A. (2012). *Active Learning for Object Detection on Satellite Images* (Tech.). Pasadena, CA.
6. Hijmans, R. J. (2015, December 19). CRAN - Package raster. Retrieved from https://cran.r-project.org/web/packages/raster/index.html