# Classification of Regions with gabor filter

*Eduarda Chagas*

*March 11, 2020*

### Importing the packages

```
# Clear workspace:
rm(list = ls())

# Load some packages:
if(!require(caret)) install.packages("caret")
```

```
## Loading required package: caret
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
if(!require(MLmetrics)) install.packages("MLmetrics")
```

```
## Loading required package: MLmetrics
```

```
##
## Attaching package: 'MLmetrics'
```

```
## The following objects are masked from 'package:caret':
##
##     MAE, RMSE
```

```
## The following object is masked from 'package:base':
##
##     Recall
```

```
setwd("/home/eduarda/Desktop/Research/Repositories/PolSARfromITQualitative/Code/Classification")
```

### Importing the dataset

For this analysis, three SAR images with different regions were used, they are:

- Sierra del Lacandon National Park, Guatemala (purchased April 10, 2015), available at [https://uavsar.jpl.nasa.gov/cgi-bin/product.pl?jobName=Lacand_30202_15043_ 006_150410_L090_CX_01 # data] (https://uavsar.jpl.nasa.gov/cgi-bin/product.pl?jobName=Lacand_30202_15043_006_150410_L090_CX_01 # data);
- Oceanic regions of Cape Canaveral (acquired on September 22, 2016);
- Urban area of the city of Munich, Germany (acquired on June 5, 2015).

A total of 160 samples were considered during the investigation, with 40 forest regions in Guatemala, 80 ocean regions in Cape Canaveral and 40 urban regions in the city of Munich.

```
n.total = 160
regions = c(rep("Forest",40), rep("Sea",80), rep("Urban", 40))
gabor.energy = read.csv(file="../../Data/gabor.csv", header=TRUE, sep=",")
gabor = data.frame(gabor.energy, regions)
```

```
split = 0.85
trainIndex = createDataPartition(gabor$regions, p = split, list = FALSE)

x = data.frame(gabor[trainIndex,1:80])
y = factor(gabor$regions[trainIndex])

x_validation = data.frame(gabor[-trainIndex,1:80])
y_validation = factor(gabor$regions[-trainIndex])

gabor = data.frame(gabor[trainIndex,1:80], "regions" = gabor$regions[trainIndex])
```

## KNN Classifier

### Creating KNN model and predicting

```
set.seed(123)
ctrl = trainControl(method="repeatedcv", number = 10, repeats = 10)
knnFit = train(regions ~., data = gabor, method = "knn",
               trControl = ctrl,
               preProcess = c("center","scale"),
               tuneLength = 20)

pred = predict(knnFit, newdata = x_validation)

xtab = table(pred, y_validation)
confusionMatrix(xtab)
```

```
## Confusion Matrix and Statistics
##
##         y_validation
## pred     Forest Sea Urban
##   Forest      6   0     0
##   Sea         0  12     0
##   Urban       0   0     6
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.8575, 1)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : 5.96e-08
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Forest Class: Sea Class: Urban
## Sensitivity                   1.00        1.0         1.00
## Specificity                   1.00        1.0         1.00
## Pos Pred Value                1.00        1.0         1.00
```

```
## Neg Pred Value                1.00        1.0         1.00
## Prevalence                    0.25        0.5         0.25
## Detection Rate                0.25        0.5         0.25
## Detection Prevalence          0.25        0.5         0.25
## Balanced Accuracy             1.00        1.0         1.00
```

```
knnFit
```

```
## k-Nearest Neighbors
##
## 136 samples
##  80 predictor
##   3 classes: 'Forest', 'Sea', 'Urban'
##
## Pre-processing: centered (80), scaled (80)
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 122, 122, 122, 122, 123, 123, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    5  0.9992857  0.9988430
##    7  0.9970330  0.9951369
##    9  0.9925110  0.9879042
##   11  0.9807308  0.9703892
##   13  0.9548132  0.9306798
##   15  0.9114542  0.8628685
##   17  0.9114542  0.8628685
##   19  0.9114542  0.8628685
##   21  0.9100733  0.8606513
##   23  0.9085348  0.8581504
##   25  0.9012747  0.8468321
##   27  0.8990055  0.8433648
##   29  0.8886502  0.8271334
##   31  0.8818846  0.8164135
##   33  0.8774817  0.8093449
##   35  0.8737363  0.8034666
##   37  0.8692857  0.7964157
##   39  0.8649304  0.7897272
##   41  0.8577802  0.7788800
##   43  0.8517106  0.7691011
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 5.
```

```
cat("Accuracy: ", Accuracy(pred, y_validation), " Recall: ", Recall(pred, y_validation), " Precision: "
```

```
## Accuracy:  1  Recall:  1  Precision:  1  F1-Score:  1
```

## SVM Classifier

### Creating SVM model and predicting

```
svmFit <- train(regions ~., data = gabor, method = "svmRadial",
                trControl=ctrl,
```

```
                preProcess = c("center", "scale"),
                tuneLength = 20)
pred = predict(svmFit, newdata = x_validation)

xtab = table(pred, y_validation)
confusionMatrix(xtab)
```

```
## Confusion Matrix and Statistics
##
##          y_validation
## pred      Forest Sea Urban
##   Forest       4   0     0
##   Sea          0  12     0
##   Urban        2   0     6
##
## Overall Statistics
##
##                Accuracy : 0.9167
##                  95% CI : (0.73, 0.9897)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : 1.794e-05
##
##                   Kappa : 0.8667
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Forest Class: Sea Class: Urban
## Sensitivity                 0.6667        1.0       1.0000
## Specificity                 1.0000        1.0       0.8889
## Pos Pred Value              1.0000        1.0       0.7500
## Neg Pred Value              0.9000        1.0       1.0000
## Prevalence                  0.2500        0.5       0.2500
## Detection Rate              0.1667        0.5       0.2500
## Detection Prevalence        0.1667        0.5       0.3333
## Balanced Accuracy           0.8333        1.0       0.9444
```

```
cat("Accuracy: ", Accuracy(pred, y_validation), " Recall: ", Recall(pred, y_validation), " Precision: "
```

```
## Accuracy:  0.9166667  Recall:  1  Precision:  0.6666667  F1-Score:  0.8
```

## Random Forest Classifier

### Creating Random Forest model and predicting

```
rfFit <- train(regions ~., data = gabor, method = "rf",
               trControl = ctrl,
               preProcess = c("center","scale"),
               tuneLength = 20)
pred = predict(rfFit, newdata = x_validation)
```

```r
cat("Accuracy: ", Accuracy(pred, y_validation), " Recall: ", Recall(pred, y_validation), " Precision: "
```

```
## Accuracy:  1  Recall:  1  Precision:  1  F1-Score:  1
```