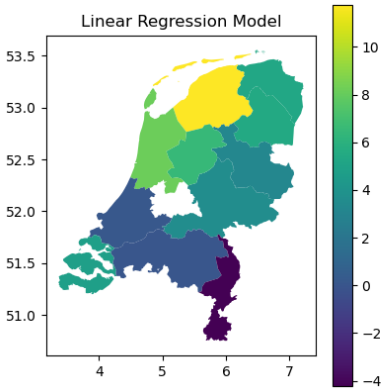
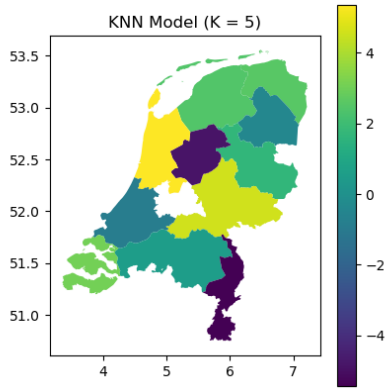
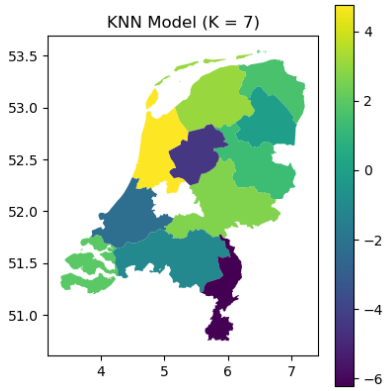
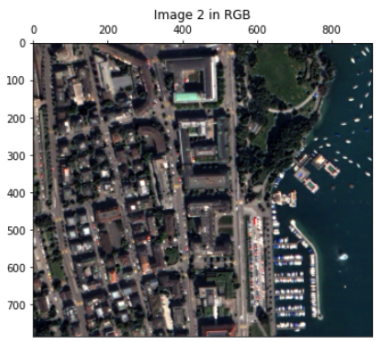
# Machine Learning - MGI Project

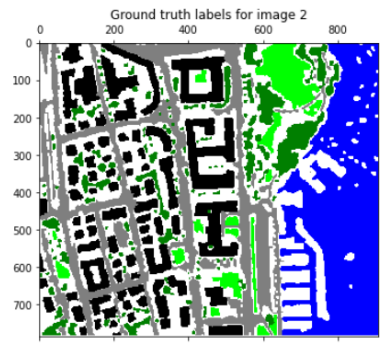
Sam Groen - 1184261
Jacotte Monroe - 1010220

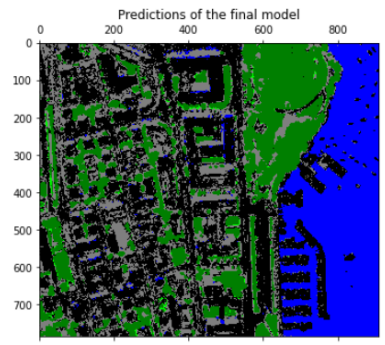


# Project Week 1 - Regression

The aim of this first project was to evaluate the methods for predicting potato yield in the Netherlands. The dataset was first visualized to examine its distribution. Then, a linear regression and KNN-model were fitted and their performances were compared. Lastly, the error of the models per region was mapped for the Netherlands.

## Exercise 1 - Data Visualization

### 1.1 Yield Plots

The crop yield dataset was first visualized by plotting a histogram of the yield for the training and test sets (Figure 1). The histograms indicate a normally distributed dataset, with an approximate mean around 43.

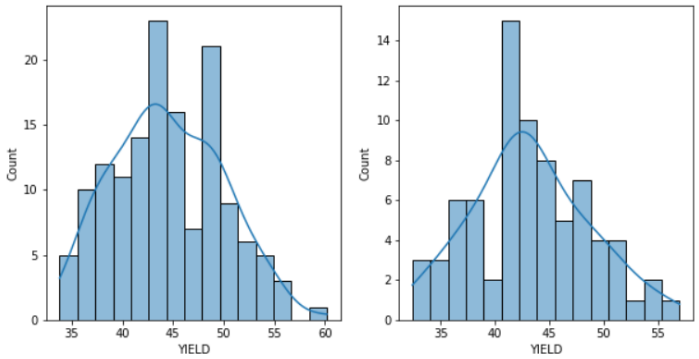



*Figure 1: Histogram of the crop yield for the training dataset (left) and testing dataset (right).*

### 1.2-3 Summary Statistics

We derived the summary statistics for potato crop yield of both datasets as seen in Table 1. The overall crop yield dataset was made up of 220 observations ranging from 1999 to 2018. The 143 observations from 1999 to 2011 were used to train the model and the remaining 77 observations from 2012 to 2018 were used to train the model. The test dataset was found to have a lower average crop yield of 43 compared to that of the training dataset, also shown in a boxplot in the code. The test dataset also had a standard deviation of 5.43 compared to 5.39 for the training dataset. This could be explained by the smaller sample size of the test dataset.

*Table 1: Summary statistics for crop yield of training and testing datasets.*

| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| **Yield_train** | 143.0 | 44.497902 | 5.386753 | 33.8 | 40.55 | 44.0 | 48.3 | 60.2 |
| **Yield_test** | 77.0 | 43.232468 | 5.426529 | 32.4 | 39.40 | 42.7 | 46.4 | 57.0 |

### 1.4-6 Correlations

The yield was plotted against the average daily air temperature (TAVG) and the accumulated daily precipitation (PREC) for both datasets, Figure 2. No clear correlations or patterns could be identified between the predictors and the yield. The correlation between the predictors and the yield was tested

for all predictors, see code. Furthermore, some of the predictors were found to correlate with each other. This was the case for the water-limited leaf area (WLAI) and the water-limited transpiration (TWC). Indeed, the more leaf surface a plant has, the more area there is for the plant to transpire. These two parameters could have therefore been combined or one could have been dropped to reduce redundancy in the model.

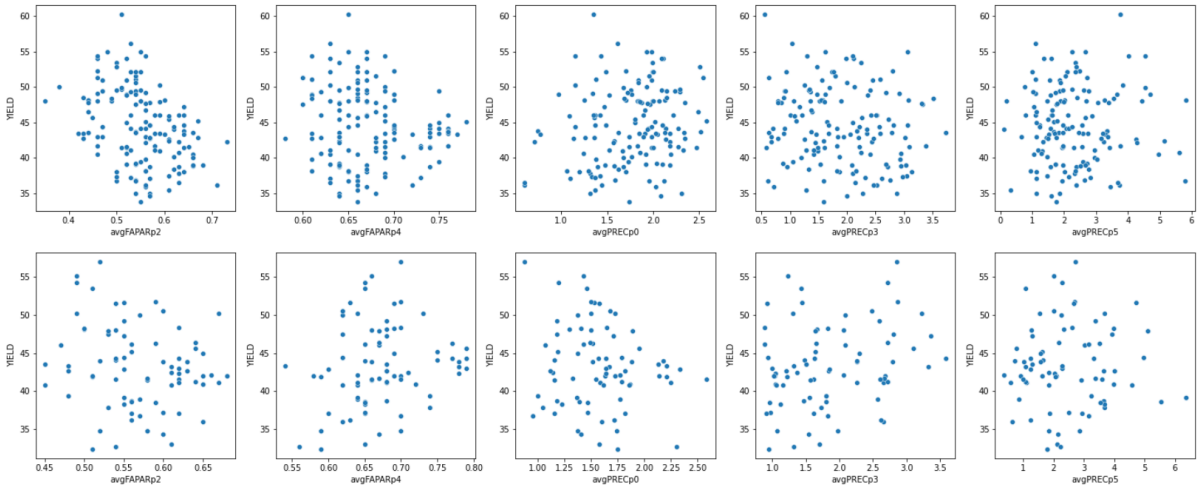



*Figure 2: Scatterplots of the yield as a function of the average daily air temperature (°C) and the sum of daily precipitation (mm) for the training set (top) and test set (bottom).*

# Exercise 2 - Linear Model

## 2.1-3 Trained Model Evaluation

In the attempt of finding a fitting model for the potato crop yield dataset, we made a first assumption that the dataset followed a linear form. A linear regression was thus fitted through the training dataset. This returned an R-squared of 0.62, while the estimated coefficients illustrated positive and negative correlations between the predictors and the crop yield, see code. For instance, a one-unit increase in max WLAI in phase 4 led to a -4.32 change in crop yield, in this case a decrease. Moreover, several predictors such as the max TWC in phase 4, the average TAVG in phase 0, and the average fraction of absorbed photosynthetically active radiation (FAPAR) in phase 2, showed significance with p-values below 0.05. On the other hand, the max water-limited dry weight biomass, the average climate water balance, and the average FAPAR of phase 4 were insignificant.

## 2.4-5 Fitted Model Evaluation

The results of testing the model to test data can be seen in Table 2. As expected, it can be seen that the train model performed better than the test model on all metrics. The R-squared value for the test model was found to be negative, indicating an arbitrarily worse model, according to Sckit Learn documentation. It was expected for the training model to obtain better results since the model was fitted to its own data and thus needed to predict data that it had already seen. On the other hand, the model was not fitted on the test data, it thus dealt with new data, making it harder to reach accurate predictions.

*Table 2: Overview of the model performance.*

| Model | MAE | RMSE | R-squared |
|---|---|---|---|
| Test | 5.326556 | 40.149747 | -0.381388 |
| Train | 2.656922 | 11.055894 | 0.616304 |

In Figure 3, the predicted responses of the test model were plotted against the observed responses. As can be seen, the model did not perform very well. If the model would have been very good, the plotted points would have been close to the linear ideal fit. Instead, our model followed the red line. Since the red line starts flattened out but slowly gains its slope, it can be seen that for lower response values the model tended to overestimate whereas the model tended to underestimate higher responses.

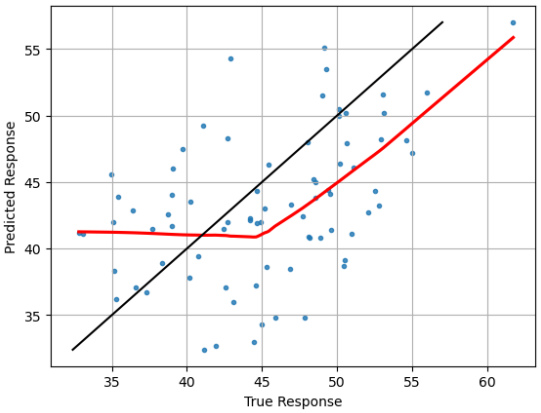



*Figure 3: Scatterplot of the predicted response versus the observed response with the smoothed trend of the data (red line) and ideal fit (black line).*

# Exercise 3 - KNN Model

## 3.1 K-value

We applied a K-neighbors regression model to predict crop yield. We reasoned that a lower K value would be the best option for this model, based on two arguments. First, the model had a high complexity due to its 22 predictors. As your model is more complex and has more predictors, more data is also needed. Second, due to the curse of dimensionality there were small chances of finding close neighbors as the distance between predictors was very big. Therefore, having a high K would result in far less accurate responses. We therefore decided to start with a more flexible K = 3 model. Table 3 provides an oversight of the assessment metrics compared to the linear model. It was seen that the KNN model outperformed the linear model on all three metrics.

*Table 3: A comparison of the KNN and Linear models.*

| Model | MAE | RMSE | R-squared |
|---|---|---|---|
| KNN (K = 3) | 4.130736 | 25.673449 | 0.116682 |
| Linear | 5.326556 | 40.149747 | -0.381388 |

## 3.2 Cross Validation

In order to optimize the KNN model, a cross validation was performed to gain insights into which value for K would result in the best model. For this cross validation, a value range of 1 to 15 was chosen for our K hyperparameter. This range was chosen based on our earlier assessment that a smaller K value would probably be the best fit. Running the cross validation got us the following results seen in Figure 4, with two values for K that performed best. First, based on the MAE, a value of 5 for K was best, having the lowest error of 6.4. Second, based on the RMSE and the R-squared, a value of 7 for K was best.

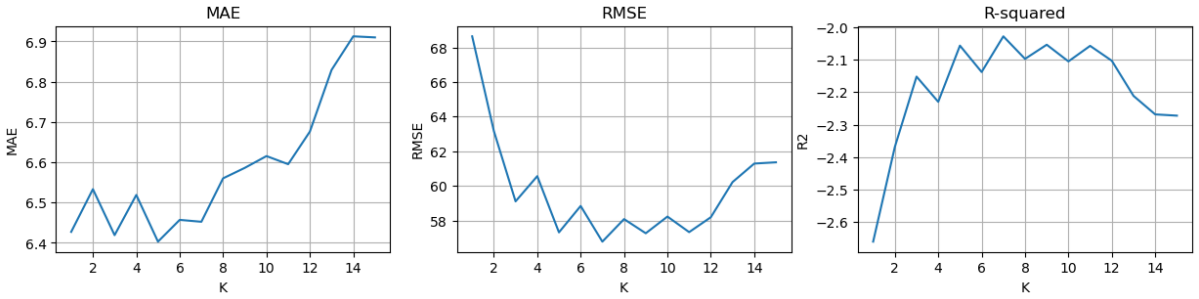



*Figure 4: Plots of mean absolute error, root mean square error, and R-squared for K hyperparameter tuning from 1 to 15.*

## 3.3 Model Evaluation

After tuning the hyperparameter K, the models were run on the test dataset. In Table 4, a comparison of all three models performed to analyze the crop yield data can be seen. As could be expected based on the earlier comparison of the KNN (K = 3) to the linear model, both KNN (K = 5) and KNN (K = 7) performed better on all three metrics. Overall, the KNN (K = 7) performed worse than KNN (K = 5) with a higher RMSE of 28.52, compared to 27.84 for KNN (K = 5), and a lower R-squared of 0.02, compared to 0.04. This difference in results when looking back at the cross-validation could be because KNN (K = 7) fitted more closely to the training and validation data, and thus had more bias.

*Table 4: Comparison of model performances for the three tested models.*

| Model | MAE | RMSE | R-squared |
|---|---|---|---|
| KNN (K = 5) | 4.326753 | 27.836717 | 0.042253 |
| KNN (K = 7) | 4.301855 | 28.521906 | 0.018678 |
| Linear | 5.326556 | 40.149747 | -0.381388 |

Furthermore, when we looked at the predicted responses versus the observed responses as presented in Figure 5, it was seen that the three different methods lead to different errors. As was expected, all three models did not follow a straight linear line. The KNN models showed a trend of diminishing return when predicting as the curve plateaued after a true value of 45. As a result, the KNN models were prone to heavily underestimate higher responses.

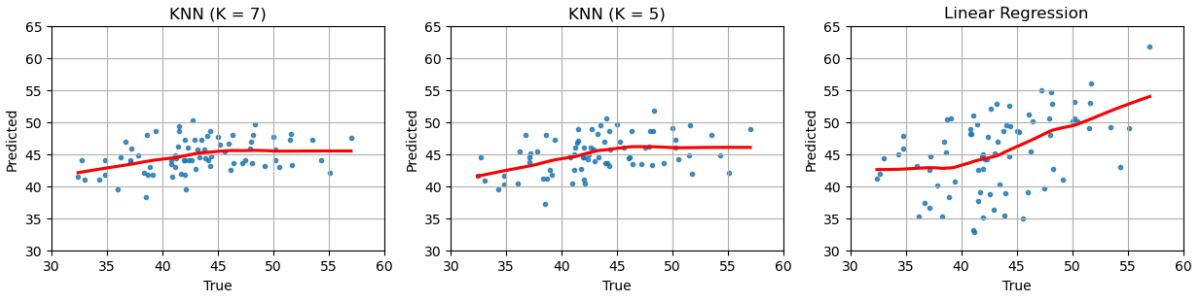



*Figure 5: Comparison of the true vs predicted trends (red line) of all three models.*

When looking at the range of errors of the three models, as presented in Figure 6, several observations were made. First, the linear model had the biggest range and highest median. This was expected as this was the worst model. Second, both KNN models looked quite similar, which was also to be expected since the metrics did not point to one best model in our cross validation. These mixed results could possibly be explained by the observation that the K = 7 model had more outliers, which was inline with the fact that a higher K value made the KNN model less flexible.

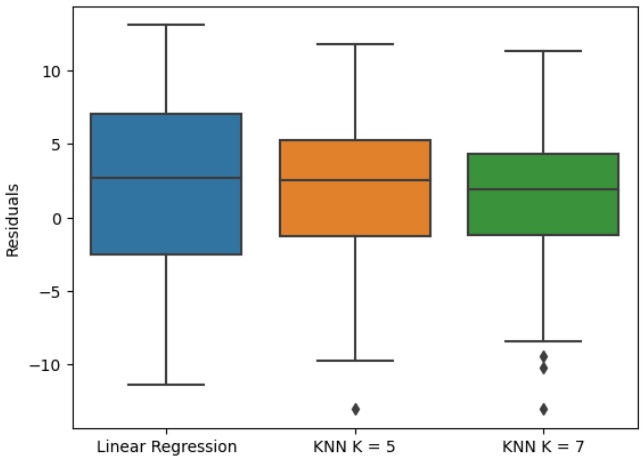



*Figure 6: Boxplot comparison of the errors for all three models.*

# Exercise 4 - Data Visualization

To assess how the model performs compared to regions, a map has been made to visualize this, as seen in Figure 7. As expected, both KNN models perform almost identically while the linear model shows different errors. For the linear model, it can be seen that it tends to significantly overestimate yields in Friesland. On the other hand, it underestimates yields for the Southern provinces, especially Limburg.

As far as the KNN models are concerned, both models overestimate yields from Noord-Holland. In addition, it underestimates yields in Flevoland and Limburg.

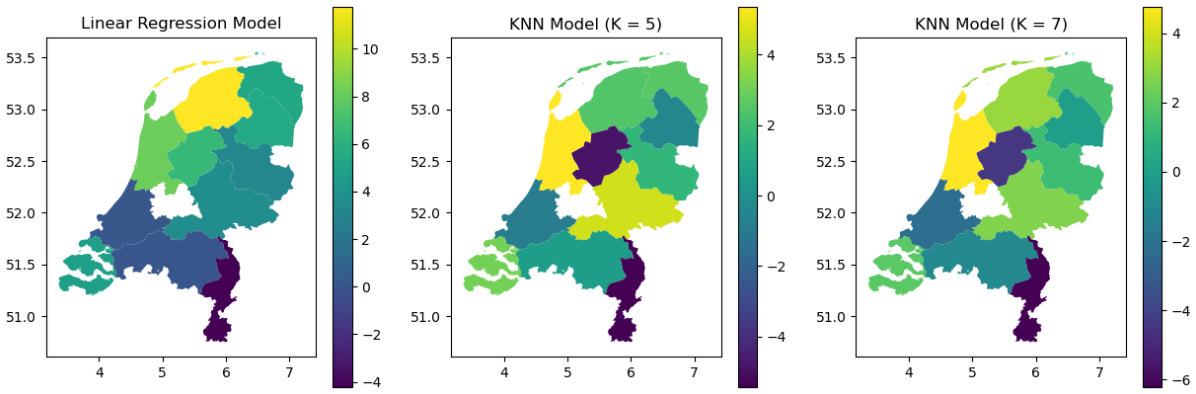



*Figure 7: Visualization of the errors compared to the region.*

# Project Week 2 - Classification

For this project, a remote sensing task was performed. The goal here was to classify land use from satellite images. Two satellite images were given, one for training and validating and another for testing. The training and validation image was first processed (feature generation, data splitting, etc.) to prepare it for our classifier. The classifier used here was a random forest classifier as this is a robust classifier due to its versatility.

## 1 Environment Setup

It was important to first set a seed value because this allows readers, programmers and others who read about your model to reproduce your work. If a seed is not set, the random value will result in different results for each run and thus the work would not be reproducible.

## 2 Feature Generation

An additional Normalized Difference Vegetation Index (NDVI) feature was generated to classify pixels in ways that was not possible based on only the RGB and NIR features. The index highlights the reflectance characteristics of vegetation and attenuates the spectral signatures of other objects. For example, we knew that vegetation had a high NDVI, whereas water did not. This can be seen in Figure 8.

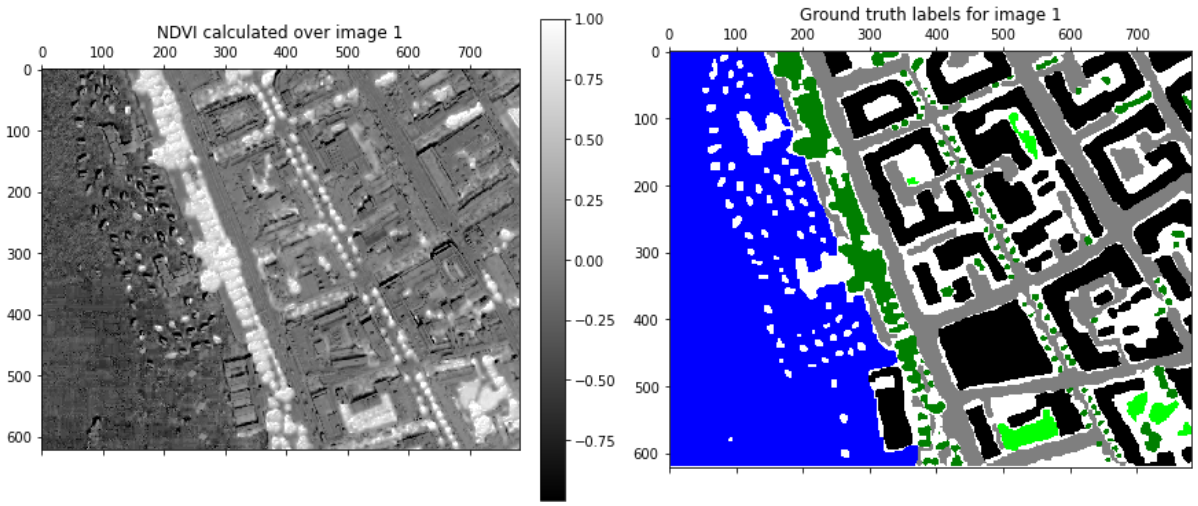



*Figure 8: Visualization of the NDVI (left) and the ground truth (right) of image 1.*

# 3 Data Splitting

The training set was further divided into a training and validation set, as shown in Figure 9. This training image could be divided because it is a full dataset of pixels, instead of a single sample. By dividing the set horizontally, all classes were present in both sets. If divided vertically, one set might not have water in it, or at least very few water pixel samples. Same goes for dividing the set randomly. It could also be argued that the pixels were already randomly divided because human infrastructure is kind of random itself. More importantly, selecting pixels at random would have resulted in a correlation between the training and validation set as there would be a close distance between the pixels from both subsets. To avoid that, we split the training set spatially.

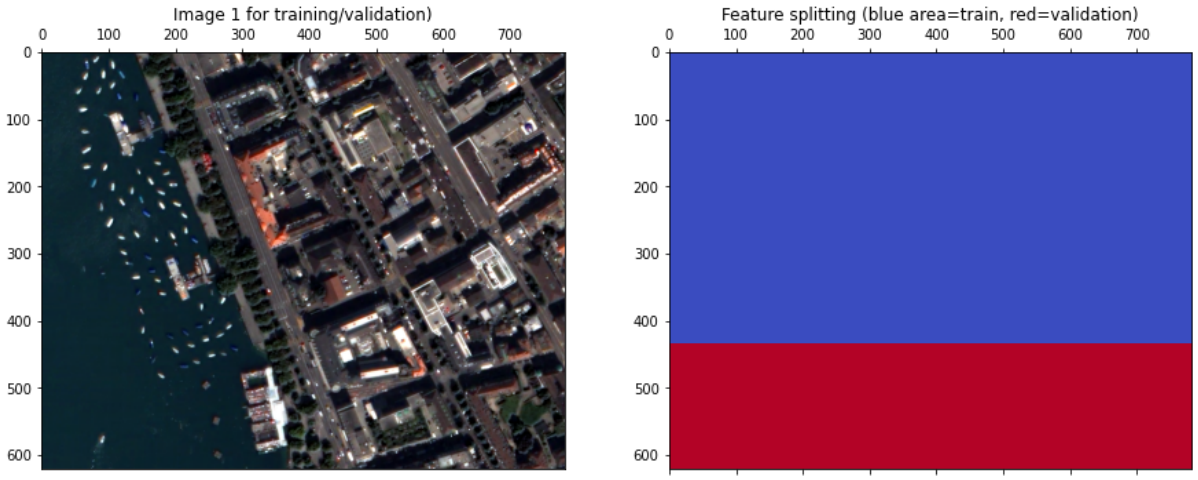



*Figure 9: Division of the training set into a train and validation set.*

The second image was chosen as a test set and thus not used as either training or validation because in this manner the test data remained independent and thus allowed for more robust test results.

# 4 Model Parameter Tuning

When building a Random Forest Classifier, we tuned two hyperparameters: *number of trees* and *number of samples per leaf*. In order to develop a robust model, parameter searching was performed

to explore which combination of parameters would result in both an accurate and computationally efficient model. Only 10% of the training data was used for this. Figure 10 shows a visualization of the explored models.

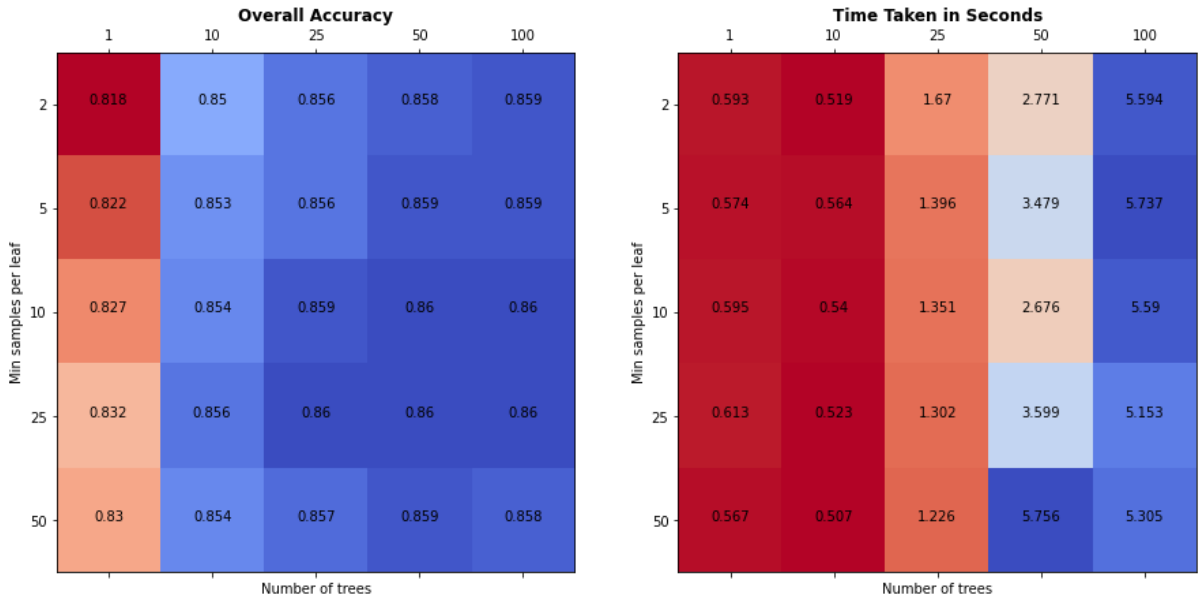



*Figure 10: Visualization of the explored models in the parameter tuning process.*

As expected, the accuracy grew as both the number of trees and minimum samples per leaf increased. This is because as the number of trees grew, the model became more robust, reducing bias as the trees explored more angles of the problem. Also, as the minimum samples per leaf increased, the variance of the model was reduced as the model became less flexible. The resulting time needed for each model was also no surprise since the computational power needed for the models grows as the number of trees increases because there are simply more trees to build.

Our choice was to go with the model that used a value of 25 for both *number of trees* and *minimum samples per leaf*. This combination resulted in one of the best performing models with an accuracy of 0.86, and was the fastest (1.302 seconds) of all models.

## 5-6 Feature Selection

In order to further optimize the model, feature selection was performed to reduce the computational complexity and thus reduce training time. To assess the importance of the features, two metrics were used: *GINI coefficient* and *Entropy*. Both metrics assessed the impurity of samples and thus the separability of the classes. A high score indicated more impurity for both metrics. As can be seen in Figure 11, the variables NIR_loc_avg, EF2, EF4 and EF5 score high. Therefore, these features were dropped from the model. Removing these features did not result in a worse model as accuracy dropped only 0.0008. However, training time dropped by almost 30%.

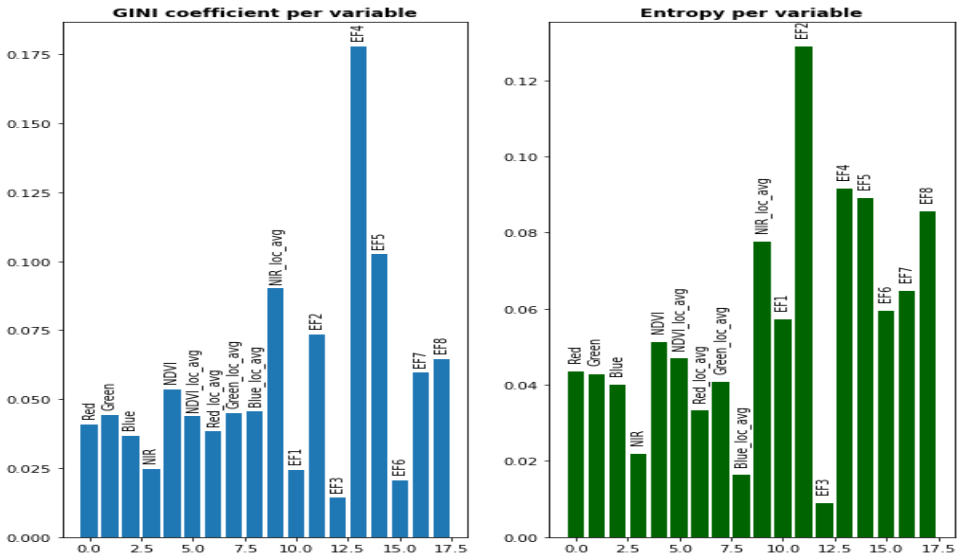



*Figure 11: Overview of the feature importance based on the GINI coefficient (left) and Entropy (right).*

# 7 - 13 Inference & Prediction Quality Control

We performed inference on Image 2 of our dataset to predict the land covers. A first classification was performed on the image, the result is shown in Figure 12. This initial attempt successfully distinguished water from land and other classes. Likewise, the real buildings were identified in the predicted image. Nevertheless, there was a significant misclassification of the roads and the marina as buildings. Additionally, the trees class was overrepresented compared to the grass class. The model also attributed shaded areas adjacent to buildings as part of the water class. For future experimentation, a new predictor could be added to have a better differentiation between grass and trees. More ground truthing could be performed to fill the no data gaps, especially in the marina and near buildings to avoid an overclassification of the buildings and trees classes. This modification could be accompanied by adding a new land cover class for boats.

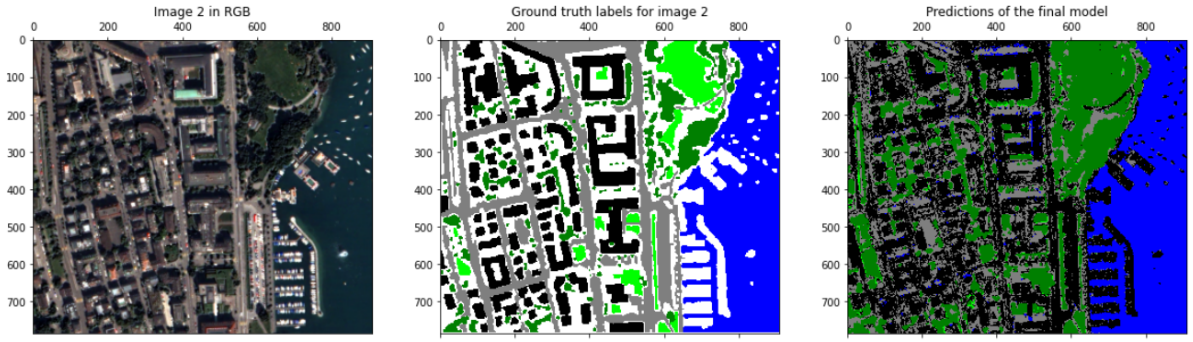



*Figure 12: Visualization of image 2 in RGB (left), with the ground truth labels (center), and with the Random Forest predicted land covers (right).*

In this project, a mask was applied to the predicted image to avoid classifying pixels with no ground truth labels. As expected, this second classification attempt gave an overall accuracy of 67%, with significantly less misclassification of pixels as trees or buildings. Regardless, the roads and grass classes remained incorrectly classified, see Figure 13. It is worth noting that the obtained accuracy was overestimated and not reflective of the model quality due to the manual addition of white background pixels to the image.

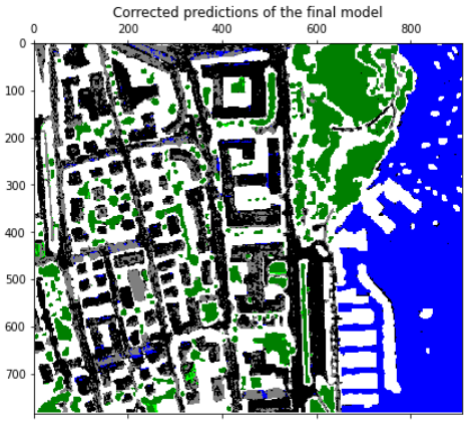



*Figure 13: Predicted land cover for image 2, with white background pixels for areas with no ground truth data.*

Furthermore, the confusion matrix was derived to provide more indications on strengths and weaknesses of the model, see Figure 14. As expected from the visualized result, 98% of the true grass pixels were misclassified as tree pixels. This could be due to the similar reflectance behaviors of the two vegetation types compared to the other classes. Additionally, the season in which the image was taken could have an effect on the classification of grass, depending on its growth stage. Likewise, the infrastructure classes (roads and buildings) largely overlapped, with 63% of the true road pixels labeled as buildings and 23% of the true building pixels classified as roads. The similar construction material, spectral signatures, and shape of these objects could be a reason for this confusion.

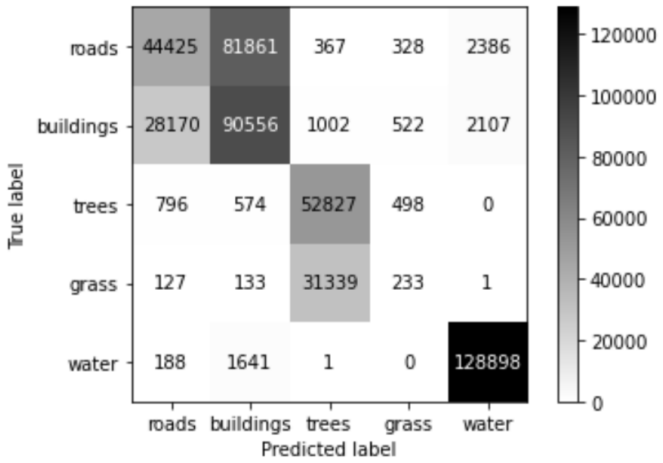



*Figure 14: Confusion matrix of corrected predicted land cover of image 2.*

More importantly, the recall and precision were calculated for each class to illustrate the true quality of the model, see code. The recall is an indicator for how accurate the predicted land cover image is compared to the ground truth. It calculates the percentage of the true class that has been correctly classified. On the other hand, the precision illustrates how reliable the model is for predicting land cover. It is the percentage of the predicted class that has been assigned to the correct class, it measures the ability to avoid false positives. Indeed, we observed that the high recall and precision of the large water class, 0.966 and 0.986, respectively, led to a skewed overall accuracy. This high accuracy was not representative of the real model performance. We saw that the model has trouble accurately predicting the grass class, with a recall of 0.147 and a precision of 0.007. While the trees class had a precision of 0.966, its recall was of 0.618 due to the grass pixels being classified as trees. The model was thus able to correctly spot and classify most tree pixels, but was not able to identify grass pixels. In the future, the ground truth may want to avoid areas of grass shaded by tree cover to

make the tree and grass classes more separable. Moreover, the 0.343 precision for roads and the 0.518 recall for buildings indicated a misidentification of road pixels as buildings. The precision of buildings of 0.74 also hinted at some difficulties in identifying buildings. Ground truthing may need to consider the effect of shading in streets, as well as varying roof materials that make buildings seem like roads.

# Project Week 3 - Clustering with K-Means

The aim of this project was to use the K-means clustering method for observing statistical patterns in the characteristics of cities around the world. The project was two-fold. First the K-means clustering algorithm was performed on a toy dataset of the cities to set up the classifier function. Then, the algorithm was applied to a real dataset, where conclusions were drawn on spatial and statistical patterns of the cities and their corresponding regions.

## 1-5 K-Means on a Toy Dataset

The toy dataset consisted of 200 cities and 2 unknown features describing each city. The data was normalized to avoid having features that weighed more than others in the clustering algorithm, which would result in non-elliptical overlapping clusters. Once the data was normalized, we initialized our K-mean classifier by randomly selecting centroids for our four clusters. The outcome of the clustering greatly depends on the location of the starting cluster centroids. Therefore, we chose these locations randomly to minimize bias in our procedure. The classifier was run over 10 iterations. At each iteration, the cluster centroids were updated based on the locations of the nearest points, and each point was designated to its closest centroid. Figure 15 displays the data divided into four clusters plotted on the two variables for the first 4 iterations. As expected, the normalization done in data preprocessing resulted in more elliptical non-overlapping clusters. More importantly, the cluster centroids reached an equilibrium at iteration 3, resulting in more distributed and compact elliptical clusters.

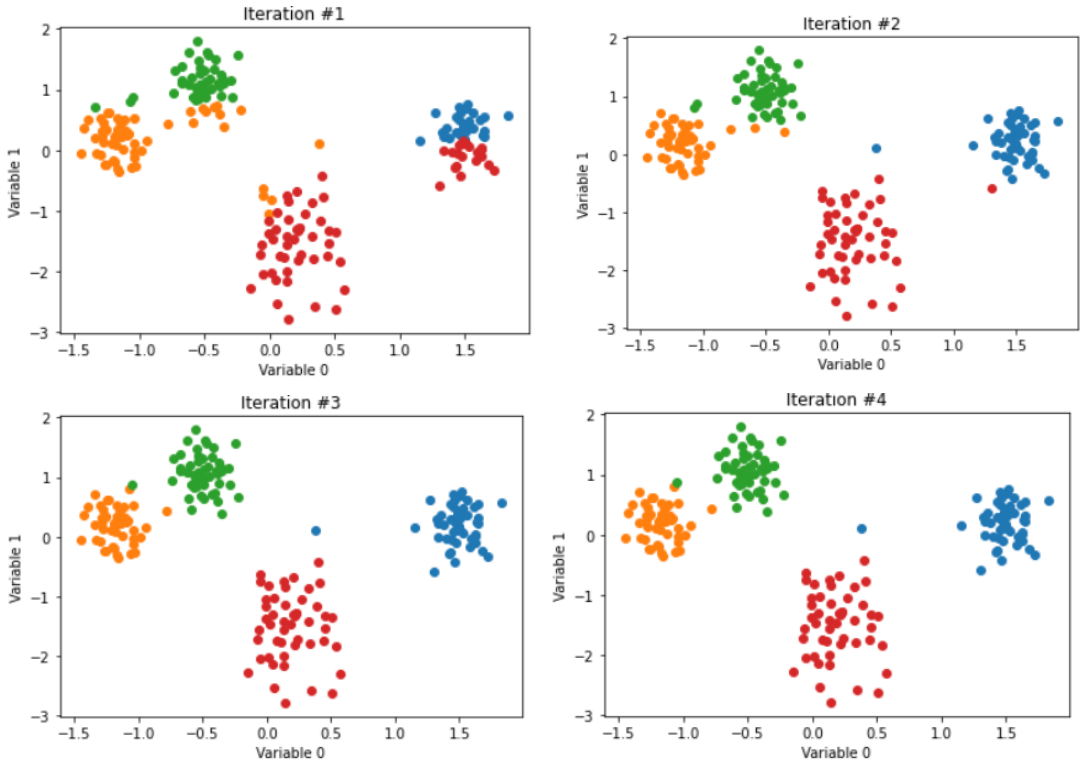



*Figure 15: Toy cities dataset plotted on the two variables for iterations 1 through 4. All points are assigned one of four clusters, shown in a different color.*

# 6-8 K-Means on Real City Data

Next, our K-Means classifier was used on a dataset with real city data. This dataset contained 280 observations along with 8 features. First, as the features had different units, the dataset was normalized to avoid that some features had outsized effects on our classifier due to a difference in weight. The dataset was also transformed to make sure it was ready to be fed to the classifier function.

Our K-Means analysis was performed using 4 clusters. As output, the following clusters were obtained, as seen in Figure 16.

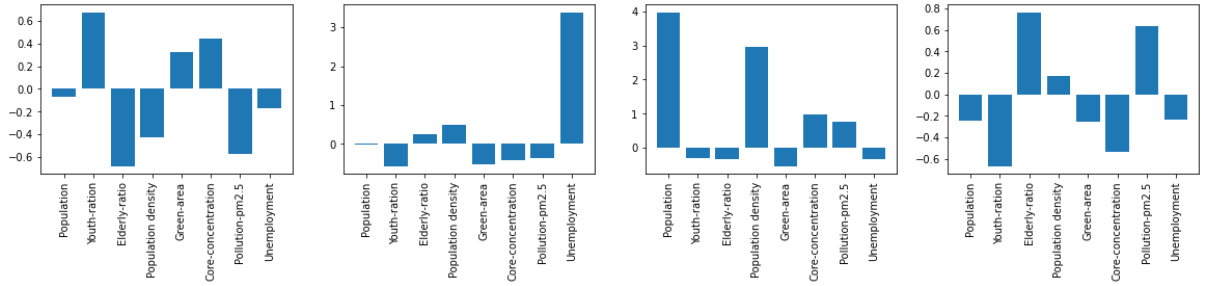



*Figure 16: Overview of the characteristics for the 4 obtained clusters (1 to 4 from left to right).*

From Figure 16, several observations can be made. First, it was seen that Cluster 1 was characterized by its high youth ratio and its low elderly people ratio. In addition, the population and population density of the cities in this cluster were relatively low. Another notable observation for Cluster 1 was that its pollution was quite low. This could be due to the relatively large green areas, which absorbed pollution.

Second, Cluster 2 was defined by its high unemployment ratio. Also, these cities had a relatively low youth ratio and relatively few green areas. Furthermore, the core concentration of these cities was also relatively low. As far as pollution was concerned, these cities had relatively low pollution.

Third, due to its relatively high populations and population density the cities in Cluster 3 were likely to be megacities. In addition, these cities had a relatively high amount of pollution. Also, these cities had relatively low unemployment rates and few green areas. Therefore, these cities were probably very industrial cities.

At last, Cluster 4 was characterized by its high elderly ratio and low youth ratio. In addition, air pollution was relatively high in these cities. In addition, despite its relatively low population these cities had relatively high population density and thus these cities were presumably smaller cities. Unemployment was also relatively low in these cities.

To further examine the clusters, the clusters were dissected by region. This can be seen in Figure 17.

Cluster 1 contained mostly Northern American cities, which make up around 55% of the cluster. In addition, approximately a quarter of the cities in this cluster were located in Latin America.

Cluster 2 only contained cities from Southern Europe. At first this was quite surprising since this cluster was characterized by its high unemployment. However, the analytics on the website of the OECD data supported this observation as Southern European countries such as Spain and Greece have had some of the highest unemployment rates in 2020 and 2021.

In Cluster 3, around half of the cities were located in East Asia. The rest of the cluster was divided into Western European countries and North and Latin American countries. This made sense since in East Asia there are lots of megacities, and also a lot of production of goods takes place in these cities.

At last, Cluster 4 contained cities from Central Europe and East Asia mostly. This also made sense since for example Japan has a problem of an aging population. The presence of Central

European cities also made sense since this cluster presumably contains smaller cities given its relatively low population and high population density.

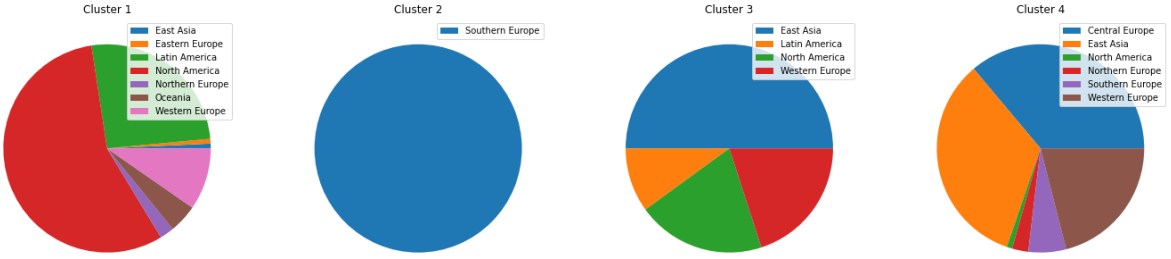



*Figure 17: Visualization of the clusters by region.*

Now turning our focus onto different regions, several observations can be made. First, regarding Europe it can be said that it contains a very wide variety of cities as European cities are represented in all four clusters. What further stood out in Europe are the high unemployment rates in its Southern cities. In addition, Western Europe had more big cities compared to Central Europe, given that Western Europe was part of Cluster 3 which can be characterized as megacities, whereas Central Europe is part of Cluster 4. In addition, Central Europe seems to have a population that is older than Northern and Western Europe.

Second, Northern America can be characterized by its youthful population. In addition, several megacities are located in Northern America whereas the region is not really home to smaller cities as it was only minimally represented in Cluster 4.

Third, as far as Latin America was concerned, it was observed that this region also has a youthful population, which was seen by its presence in Cluster 1. In addition, it is home to several big cities as it was also represented in Cluster 3.

At last, East Asia can be characterized by its megacities as it made up the majority of Cluster 3. In addition, it has quite an aging population as can be seen from its presence in Cluster 4. Since East Asian cities were represented in both Cluster 3 and 4, these cities can also be characterized by their relatively high pollution.

It must be said that due to the use of pie charts, the results obtained here are not necessarily valid. This is because the total number of cities in one cluster can differ to the total number of members of another cluster and thus could present a distorted picture. However, the results here are in line with common geographical knowledge and supported by sources. For example, East Asia has a lot of megacities and pollution due to their production of goods for Western regions. Additionally, it is relevant to note that the outcomes of K-means clustering depend on the initial splitting of the data into the clusters. Hence, in further practice, multiple iterations of the model should be run with different starting cluster distributions. This clustering should be regarded as an indication of the structure of the dataset and not a final result, it should be a starting point for more analysis.