GEOMATICS IV ASSIGNMENT 2

POINT CLOUD CLASSIFICATION

May 27, 2018

Student ID: HLLYAS001
Name: Yaseen Hull
University of Cape Town
Department of Geomatics

Contents

Chapter 1

Introduction

Mass acquisition of spatial data is commonly practiced to enhance and make decisions about the environment we live in. It provides another dimension of comprehending data and is regularly used by all faculties of society. Amongst the reliable and most used data acquisition techniques is aerial LIDAR (light amplification by simulated emission and radiation) because of its ability to operate in an all weather at any time, its effectiveness for modeling urban and rural terrain and its ability to penetrate through vegetation. LIDAR systems measure the distance to a object point by illuminating the target with a laser pulse. The time taken for the pulse emission, reflection off the target and back to the sensor is used to calculate the distance. The differences in return time and wavelengths can be used to generate a 3D representation in the form of a point cloud.

1.1 Objective

The objective of assignment is to classify a point cloud by generating and identifying features of each point. By analysing the intensities, heights, eigen vector/value pairs of the neighboring points within a specified neighborhood features can be generated on each point. The features can then be used as a basis upon which an unsupervised classification can be performed.

1.2 Tools Used

1. Cloud compare: Pre-processing and Visualisation
2. Anaconda: Normalisation and Classification (unsupervised)
3. Arcscene: Vectorisation and Accuracy Assessment

Chapter 2

Theory - Design cycle

2.1 Pre-processing

The volume of point cloud data can range from 1000 to millions of points. Working with a large number of points can be computationally intensive. It is therefore advisable to segment the data to work only with a sample and to make use of algorithms which are computationally effective. All segmentation was done in Cloud compare but other point cloud software could've also been used.

2.2 Nearest neighbour classifier

The objective of nearest neighbour methods is to find a user defined number of samples closest in distance (usually euclidean but can be defined in any other metric measure) to the point or centroid. With k-nearest neighbour the sample number is user defined hence we set k to a value to define our neigbourhood search. Radius-based neighbour learning is also a possibility which will change according to the local density of points in the sample space.
To increase performance and reduce time complexity the nearest neighbor methods in the sklearn library makes use of fast indexing structures. In particular we are interested in the kd tree algorithm as we working with a large number of samples. All the nearest neighbour algorithms act under the principle of reducing the amount of distance calculations. The ideas states that if A is distant from B and B and C are close then A is distant from C as well, without having to calculate the distance between them explicitly. The kd tree is a binary tree data structure which is very fast as it partitions data along the data's axis. It generalises between 2-dimensional Quad-trees and 3-dimensional Oct-trees to an arbitrary number of dimensions. Although it is very efficient for low-dimensional neighbor searches, it is not suitable if the dimensions exceed 20.

2.3 NORMALISATION

Normalisation is a scaling function which processes individual samples to within a range of 0 and 1. All values in the dataset can be considered to calculate the normalised value. For x in a data set Y we find the normal value:

$$n(x) = \frac{x - minimum(Y)}{maximum(Y) - minimum(Y)}$$

The sigmoid function is also available to normalise values and is as follows:

$$G(x) = \frac{1}{1 + e^{-}x}$$

The sigmoid function is also used as a classifier for class determination.

The importance of normalisation is realised when generating clusters based on features. Each feature is a influencing factor for class nomination of the data. By normalising features of points we rid the data of any bias in the feature attributes which the classification technique may recognise.

2.4 CHOOSING FEATURES

2.4.1 Eigen vector/value pairs and Dimensionality Reduction

In a higher dimensional datasets it is important to establish along which axis most variation in data exists and where data is highly correlated. This can be achieved by identifying the principle components through the PCA (Principle Component Analysis) method. The PCA method reduces the dimensionality of the space by finding the principle components with the highest variation thus omitting data with low correlation. The main objective of PCA is reducing the dimensionality of high dimension data and projecting it in a lower dimension space while retaining most of the information. The output is computed as follows:

- Normalise data
- Calculate the eigenvector/value pairs from the correlation matrix or covariance matrix
- Sort the eigenvector/value pairs in descending order and choose k number of eigenvectors that correspond to the k largest eigenvalues where k is the dimension number of the new feature subspace
- Construct a projection matrix B from the k eigenvectors
- Transform the original dataset X via B to obtain a k-dimensional feature subspace Y.

The eigenvectors are in fact the principle components of the data set and determine the direction of the new feature subspace where the corresponding eigenvalues determine their magnitude. The eigenvector/value pairs are also used to determine the best fitting plane to set of data points in the neighbourhood as well obtaining the normal to the plane. The normal vector can be used as a feature for classification alongside normalised eigenvalues.

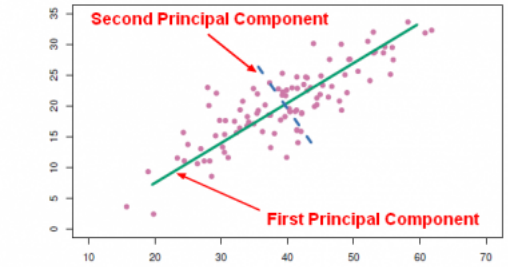

Figure 2.1: PCA

2.4.2 Feature Generation

There are a number of other features we can focus on however only a select few were chosen for our classification. The extra features used are RGB values which is an indication of height in the point cloud as well as the intensity values representing the measure of return strength of the laser pulse that generated the point.

2.5 Classification

There are generally two methods to classification. One in which the underlying theme classes are known and the other where classes are unknown. These are referred to as Supervised and unsupervised classification or clustering.
In Unsupervised classification or clustering the classes are unknown and the feature space represents the distribution of samples as oppose to classes. Much like supervised classification sampling is done to populate the feature space however there isn't any classifier training. In unsupervised classification object points are assigned unlabeled classes based on some classifier or decision rules. Essentially clusters of points are identified and after the points are assigned classes some are combined and some split based on their properties. The process must converge before the user identifies these classes and labels them accordingly. Some of the classifiers used in Unsupervised classification are K-means, Iso data and Mean shift classifiers.
The classifier used in this instance is the k-means which consists of 3 steps to achieve a classification. In a k means classifier a specified number of clusters are generated using feature values of points to define the cluster or class. Firstly, k samples of dataset X is chosen to be the initial centroids. After

the centroids are generated the other samples in the dataset are assigned to the closest or nearest centroid. The third step is generating a new centroid based on the mean (of features) of the samples surrounding the previous centroid. Steps 2 and 3 are iterated up to a point where the centroids no longer have significant movement. This is achieved by setting a threshold to which the difference between the old and new centroid adheres to.

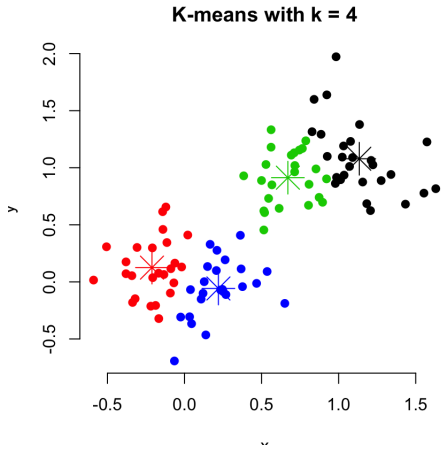

Figure 2.2: K-means

2.6 EVALUATION

Evaluation or testing of the samples is generally conveyed through the confusion matrix also known as the error matrix. The confusion matrix plots the predicted points of samples against actual or true ground point values. The confusion matrix can be seen as a measure of performance of the classifier and is represented as a table. From the matrix we can obtain information about the accuracy, misclassification rate, precision, specificity, kappa score, true positives (tp), true negatives (tn), false positives (fp), false negatives (fn), and prevalence. We can calculate the accuracy and misclassification variables by:

Meaning	Formula
How often is it right?	$accuracy = \frac{tp+tn}{total}$
How often is it wrong	$misclass = \frac{fp+fn}{total}$

Before we can compute the confusion matrix however we have to provide true ground point values to the original image/object. Once classification takes places these points in their same geographic position are overlaid on the classification to see in which classes they fall. We can achieve this by performing an accuracy assessment which randomly plots a specified number of points over a classified aerial image of the point cloud. The values of true ground points are then adjusted to their true classes using the original image.

Chapter 3

Results - Design cycle implementation

3.1 Preprocessing

Lidar data from the City of Cape Town was used to complete the practical application. Two sets of data were segmented of which both contained extra feature values such as RGB, intensity and normal values. The two sets are considerably different in the sense that one depicts a more urbanised area of the city including over half a million points and the other is a smaller and more sparse cloud of data with elements of vegetation and much smaller structures.

3.2 Nearest Neighbor classifier

The k nearest neighbour in the sklearn library was implemented to find the neighbours to each point. Considering the number of points available in each cloud k was set to 30. Using a higher value for k yields a more averaged solution when calculating the mean values of features. This also relies on the principle that points which are closer together tend to have similar values for there respective features. Both ball tree and kd tree nearest neighbour algorithms were used to find neighbours in the smaller dataset. The time complexity in these cases were relatively the same however kd tree seemed more efficient for the bigger set. The kd tree data structure stores the data and upon query isolates data which is relevant to the search. It tends to create a sort of bounding box (2-dimensional data) around the data closest to the centroid/origin neglecting all irrelevant data. This improves efficiency by reducing the time complexity of the search.

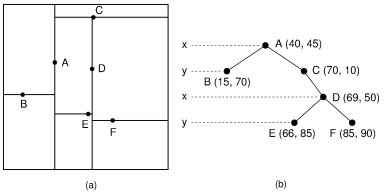

Figure 3.1: 2D KD tree visualisation

3.3 NORMALISATION OF FEATURES

The normalisation or standardisation of data is important especially if the data components are on different scales. This is particularly important for PCA as all bias needs to be eliminated from the data. Although it is unnecessary to normalise the x, y, z coordinates for the PCA, it was however important to normalise other features. Using the sklearn preprocessing normalisation methods RGB values, eigenvalues and intensities were normalised and averaged for each neighbourhood.

3.4 FEATURE SELECTION

A number of feature combinations of the two datasets were generated to to test the impact on the classification. The features used are as follows:

- Normalised eigenvalues and point normals
- Normalised eigenvalues
- Normalised intensities and point normals
- Normalised RGB values, intensities and point normals
- Point normals
- Normalised RGB values and point normals
- Normalised RGB values

These features were useful to distinguish classes from one another as well as identifying objects in the same class.

3.5 CLASSIFICATION

As mentioned in chapter one a k means unsupervised classification was completed to determine classes for the point cloud. The number of clusters was set to 150 however, increasing this value could yield better results as more discrete objects in the point cloud could be identified. The different outputs are depicted below and is discussed.

3.5.1 Raw point clouds

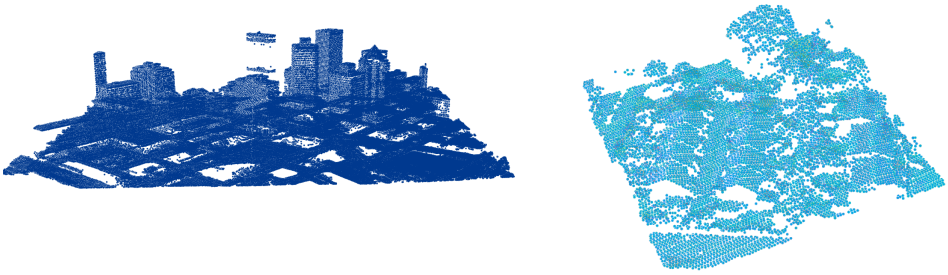

Figure 3.2: Dataset A and B point clouds

3.5.2 Classification: Normalised eigenvalue and point normals

Figure 3.3: Two oblique views of the point cloud

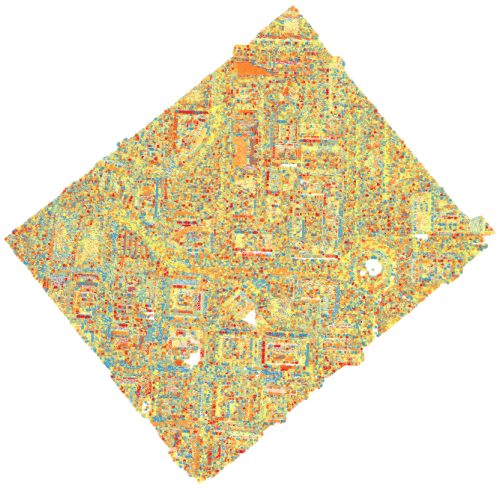

Figure 3.4: Aerial view of the point cloud

The points cloud above displays the effect of Normalised eigenvalues and the point normal on classification. In chapter one the eigenvalues were identified as the magnitude of the eigenvectors. More sensibly the eigenvalues explain the variance in data along a axis. The point normals are perpendicular or orthogonal to the surface produced by the neighbourhood of points. We can see from the oblique views that different orientations of the building faces reveal different colours. This is proof of the normals acting at each point making up the surface. The areal view is much more clustered, though this is a result of the combination of normals and eigenvalues. A road network can be identified through all the points and increasing the scale would reveal much more detail.

3.5.3 Classification: Normalised eigenvalue

Figure 3.5: Two oblique views of the point cloud

Figure 3.6: Aerial view of the point cloud

It is evident that the normals have a much better influence on bigger structures as in the previous

classification. The aerial view in this classification is however much better and roof tops of building structures and road networks are easily distinguishable. From our PCA we produced 3 eigenvectors/value pairs. The eigenvector with the lowest eigenvalue is also the normal to the plane.

3.5.4 Classification: Normalised intensities and point normal

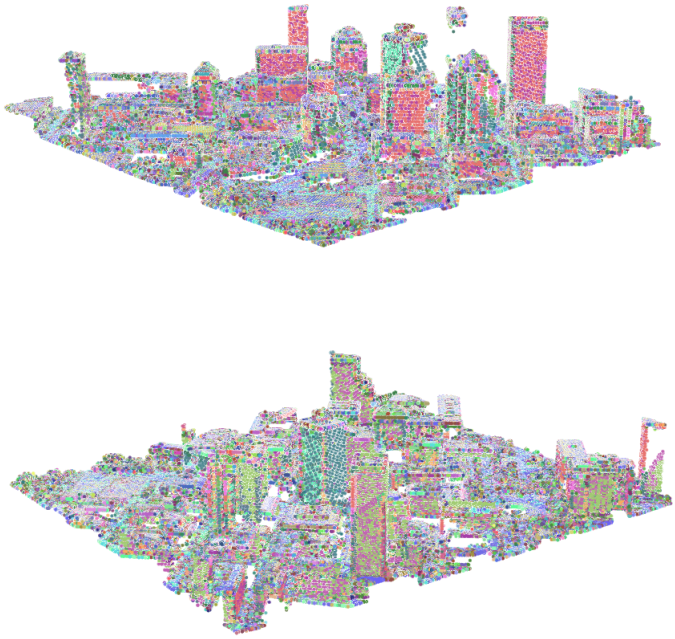

Figure 3.7: Two oblique views of the point cloud

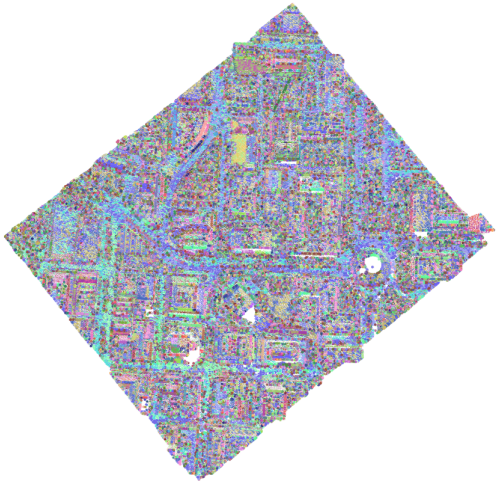

Figure 3.8: Aerial view of the point cloud

The intensities display the reflective strength of the laser. This property is also dependent on the reflectivity of the surface. The obliques display that reflective building surfaces are easily classified by using intensity as a feature for classification. Adding point normals in assists in producing different clusters on different orientations of the point cloud. Roof top surfaces are also more defined however the complex shapes influence normals and hence different classification clusters are generated.

3.5.5 Classification: Normalised RGB, intensities and point normal

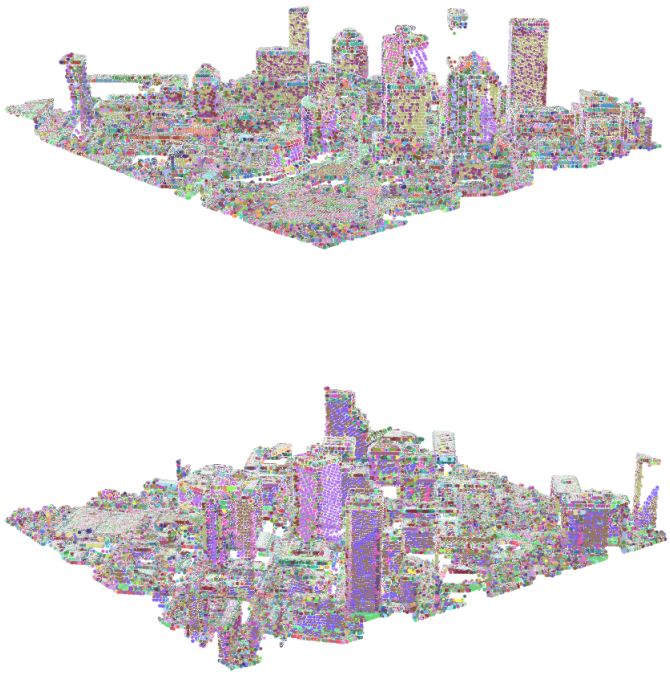

Figure 3.9: Two oblique views of the point cloud

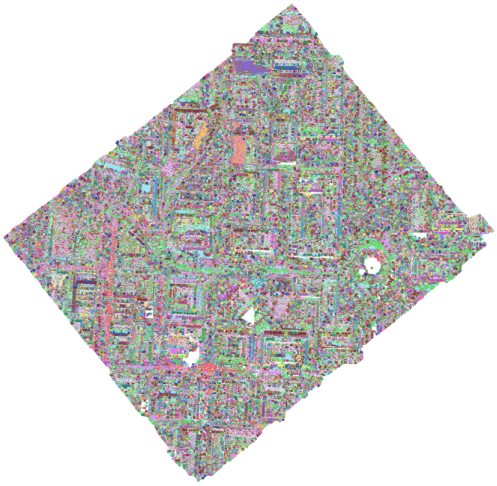

Figure 3.10: Aerial view of the point cloud

The classification is derivative of the previous however throwing in RGB makes the classification more

complex especially from a aerial perspective. Using too many features in this case wasn't a good idea.

3.5.6 Classification: Point normal

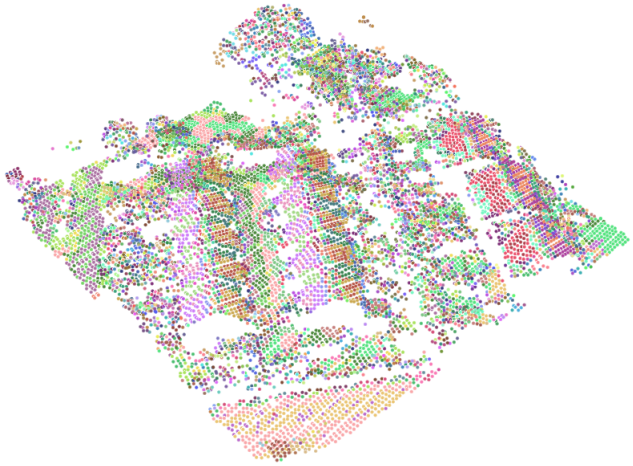

Figure 3.11: Oblique view of the point cloud

Figure 3.12: Aerial view of the point cloud

The point normal classification acted as expected. Regions with a particular orientations are assigned cluster values and generally roof tops are easily identified. Vegetation however is a mix of clusters as the point normals reflect in different directions. The ground normals are uniform in certain regions and are spotted when little to no vegetation is covering the area.

3.5.7 Classification: Normalised RGB and point normal

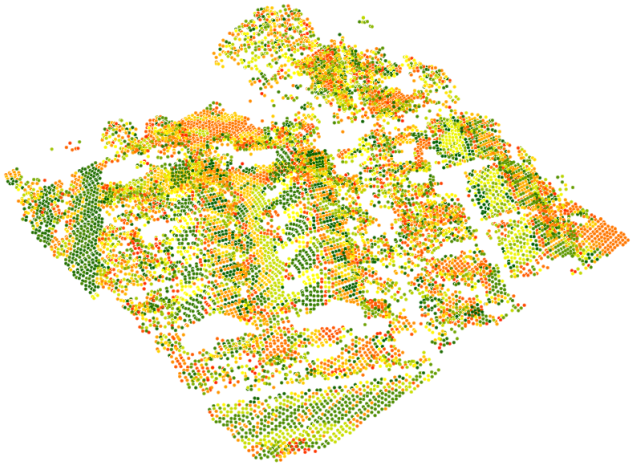

Figure 3.13: Oblique view of the point cloud

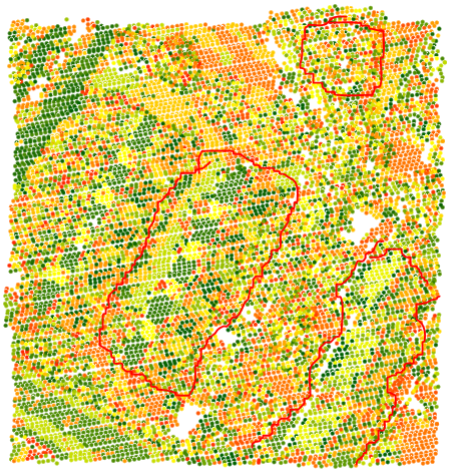

Figure 3.14: Aerial view of the point cloud

3.5.8 Classification: Normalised RGB

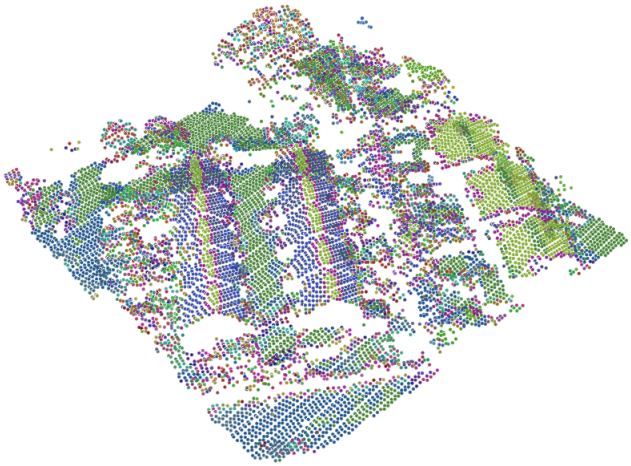

Figure 3.15: Oblique view of the point cloud

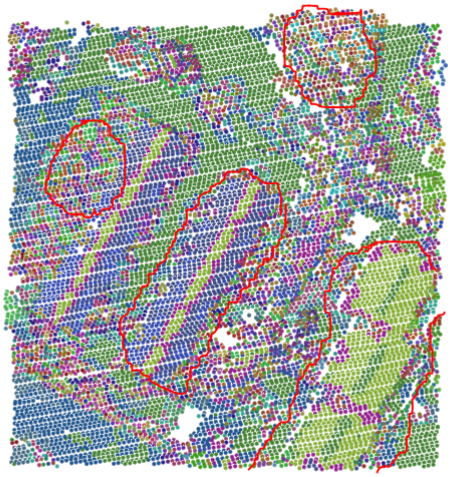

Figure 3.16: Aerial view of the point cloud

With RGB heights are made clear. Discrete clusters exists for different intervals of height. In both images these patterns can be seen.

3.6 EVALUATION

To evaluate the point cloud tools such as the confusion matrix/error matrix exists to distinguish whether points have been classified correctly. This however is only relevant for aerial images of the point cloud since existing aerial imagery exists. The oblique view imagery are much harder to come by and this would be time consuming to compute. The confusion matrix in this case can only be generated for 2 dimensional dataset. Evaluating 3 dimensional data requires the application of the data to be known and interpretation of the point cloud is a responsibility of the analyst.

Chapter 4

Conclusion - Accuracy of solution

Point cloud classification is a complex objective and requires a number of features and combinations to yield a result. Although in most of the scenarios above some form of classification was achieved, the number of points used in these datasets are cumbersome as they introduce much more detail and with that more clusters as well as time complexity. As mentioned before the classification is dependent on its application and how the analyst interprets the point cloud can influence decision making. To refine the point models further one could merge clusters decreasing the number of classes and uncertainty that exists in the point cloud. Another classification could also be done by finding the angle between the normals and vertical. Classes could then be generated for a certain range of angle values.