PES University

Cloud Computing and Big Data

PRESENTATION TITLE

POLYGON GENERATOR

Team Details

PESU CCBD

NAME: Pavan N
SRN: PES2UG19CS277
CAMPUS: EC Campus

NAME: Abhishek V
SRN: PES2UG19CS012
CAMPUS: EC Campus

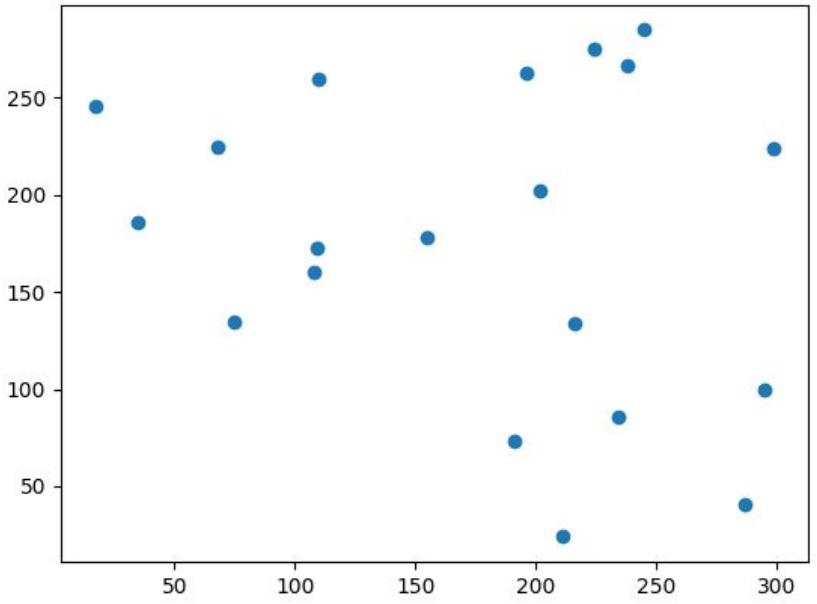
NAME: Abhishek Mishra
SRN: PES2UG19CS009
CAMPUS: EC Campus

GENERATING N RANDOM COORDINATES FOR THE POLYGON

PESU CCBD

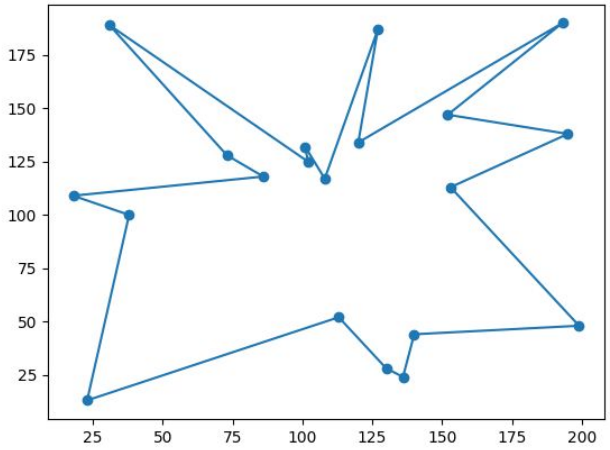
To generate a n sided polygon , we first needed to generate n random points , which would be the vertices for the polygon. This was a really easy task to do in python , by using the random function.

The only thing to be taken care here was that the coordinate points (x,y) do not repeat i.e. each coordinate generated should be unique. This was resolved by using the sample function instead of random function.



Generating 20 random non colliding points

JOINING THE RANDOM POINTS TO FORM A CLOSED POLYGON



Generating a 20 sided polygon using the polar angle and square distance method

This was one of the challenging and at the same time a very interesting task to solve , since many things had to be taken care of here , like overlapping sides when joining 2 vertices , the perfect order of coordinates so that while joining them it forms a perfect clockwise closed polygon.

There were a few approaches to this :

1. Concave/Convex hull method
2. Polar angle and square distance method

We chose the second method , since it ensured that all the points were involved in the polygon formation which was not the case in the first method.

JOINING THE RANDOM POINTS TO FORM A CLOSED POLYGON

PESU CCBD

Algorithm used here is :

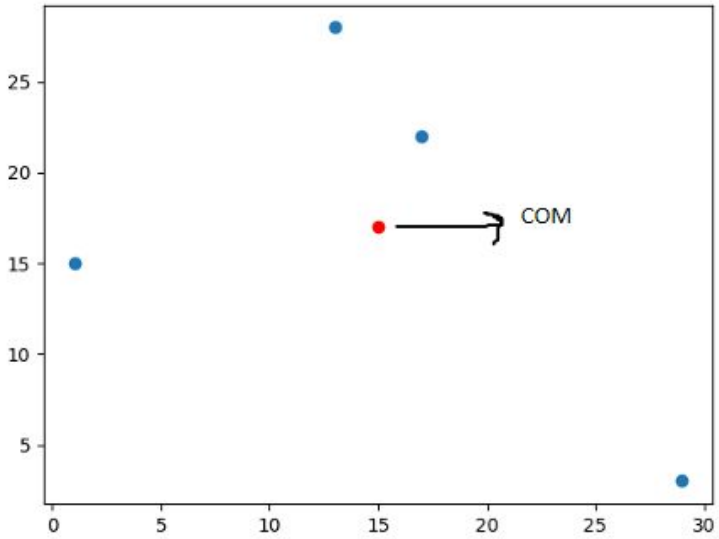
1.) Determine some good point to start with, not necessary one of the given points. Use some heuristic. For instance, the "Centre of mass" of the given points could be a useful choice. But any choice of a point within the convex hull will produce a polygon that will have non-intersecting edges. Depending on the choice, the polygon may be less or more "smooth".

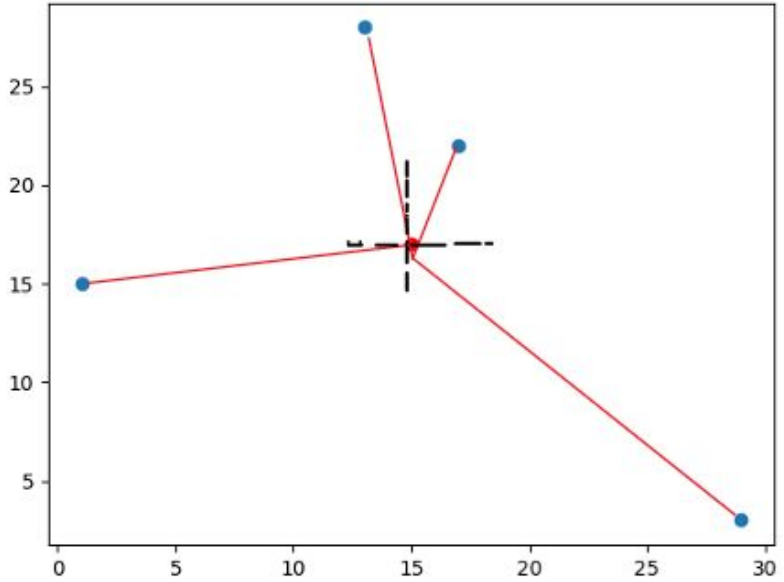
2.) Translate the coordinates of the given points to polar coordinates, where the point chosen in step one is the origin (0, 0).

3.) Sort the points by their polar coordinates: first by polar angle, then by radius (or actually the square of the radius to omit using the square root function).

4.) Draw the polygon using this ordering.

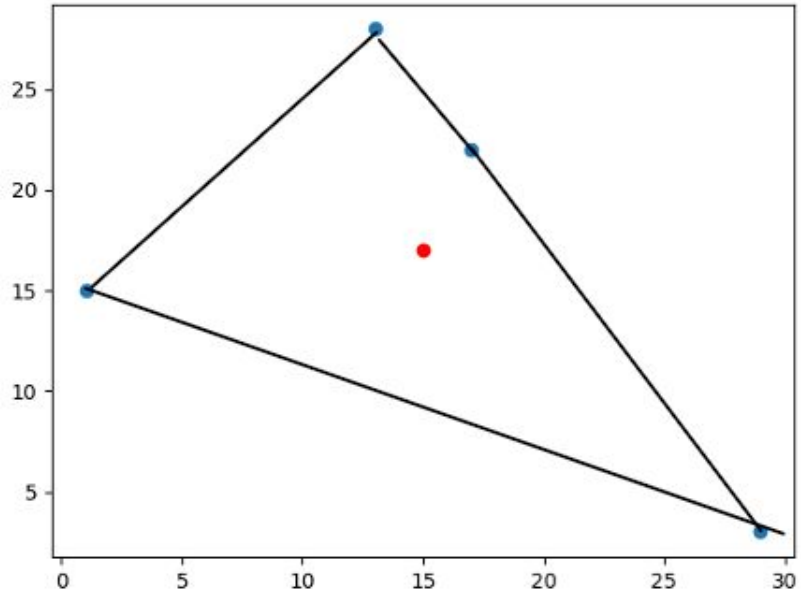
JOINING THE RANDOM POINTS TO FORM A CLOSED POLYGON





Here we are finding the polar angle and the square distance from com to all other vertices

JOINING THE RANDOM POINTS TO FORM A CLOSED POLYGON

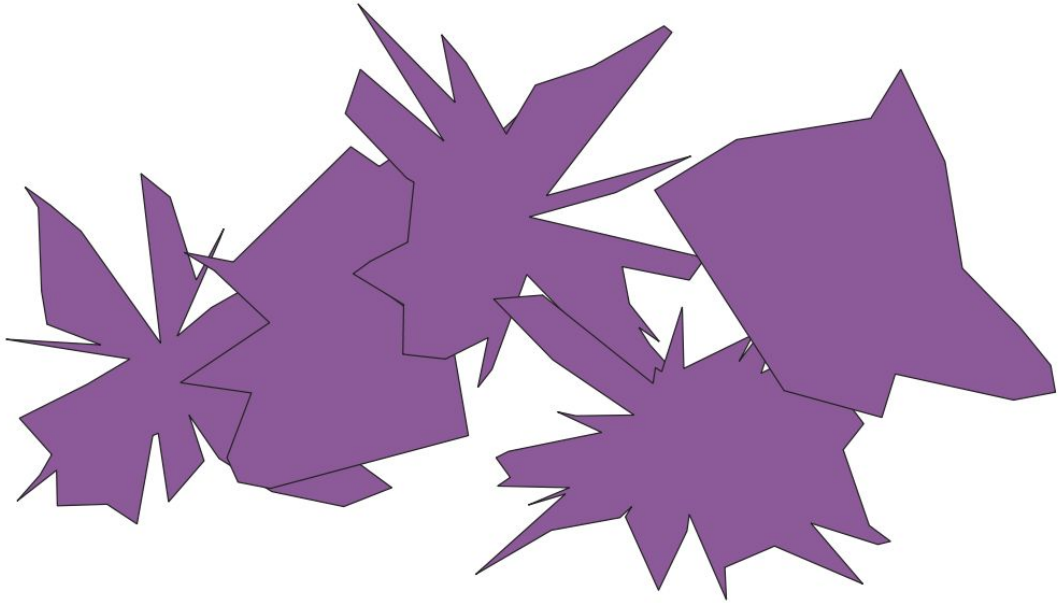


And finally joining the points in the decreasing order of their polar angle and square distance to get a polygon

FINAL GENERATING N POLYGONS ON A SINGLE CANVAS

PESU CCBD

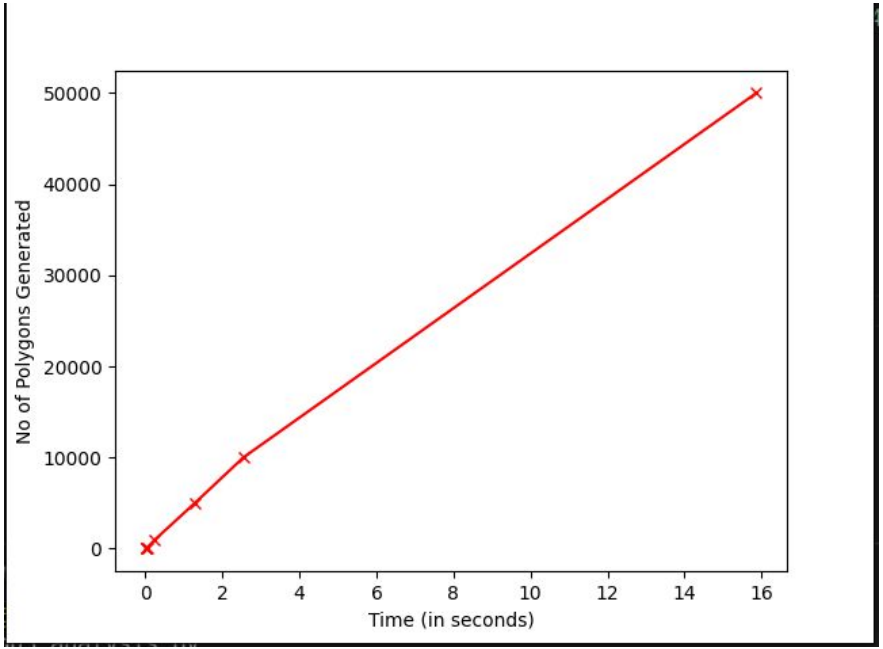
We stored the polygons generated in WKT file and to plot it , we used the qgis software

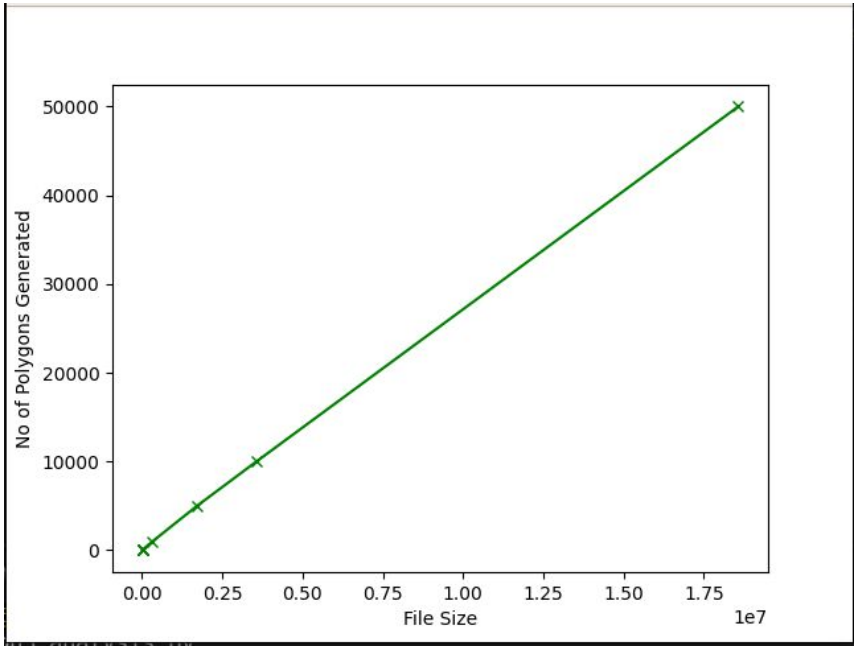


Plotting all the polygons
on a single canvas

TIME ANALYSIS

We generated a bunch of polygons and analyzed the time taken to generate the polygon as well as the file size in which it was stored.





TIME ANALYSIS

PESU CCBD

It was observed that with more number of polygons increasing , the time take was almost increasing linearly and for generating 50000 polygons it took around 16 seconds. The problem was in the file size because , the size of increasing almost exponentially and for about 50000 polygons it took 18574546 bytes of space !

This could be a problem because loading such a big data as it is in the qgis software , made it to lag and slow while plotting all the polygons

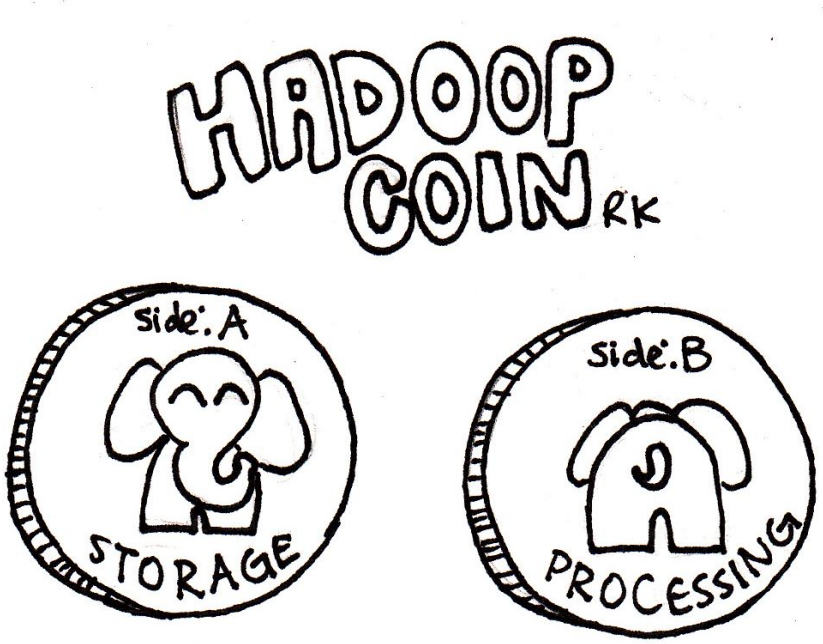
Hence something very similar to **how hadoop splits and process the data individually (like map-reduce)** is best suited for loading and constructing the polygon

TRYING TO FIX THE SLOW LOADING OF LARGE WKT FILE IN QGIS

PESU CCBD

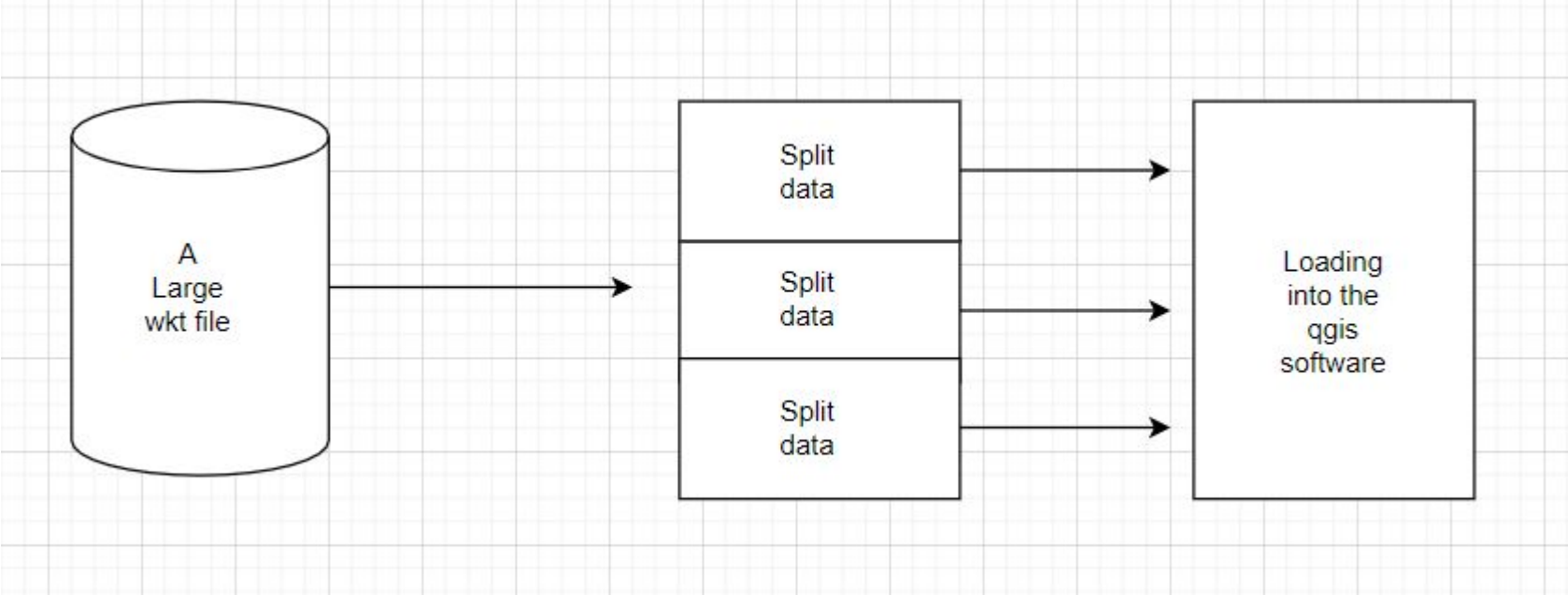
As and when the number of polygon increase the file size also increases and hence loading and plotting the polygon from the single file becomes a problem.

A naïve approach of solving this would be storing the wkt polygon in multiple files and loading them individually in the qgis software.



TRYING TO FIX THE SLOW LOADING OF LARGE WKT FILE IN QGIS

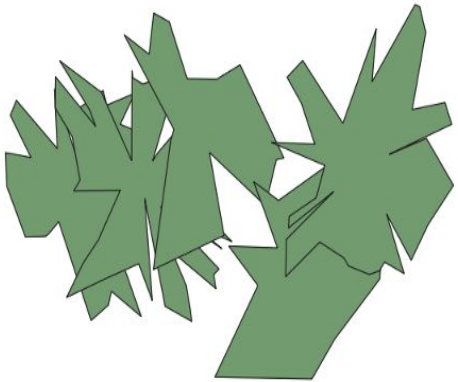
PESU CCBD

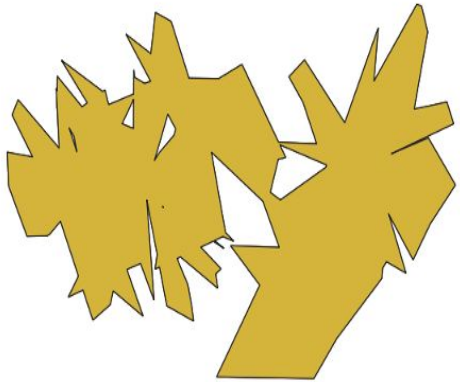


OVERLAPPING POLYGONS

While initially we were plotting all the set of polygons onto a single canvas it was overlapping , and hence it did not have a definitive boundary .

so to overcome this we merged all the polygons with the help of shapely module and thus we were able to get a well-defined boundary for all the polygons drawn together





A well defined boundary to the overlapping polygon

Data Distribution for various Objects

PESU CCBD

Here we can broadly classify the object into 2 parts

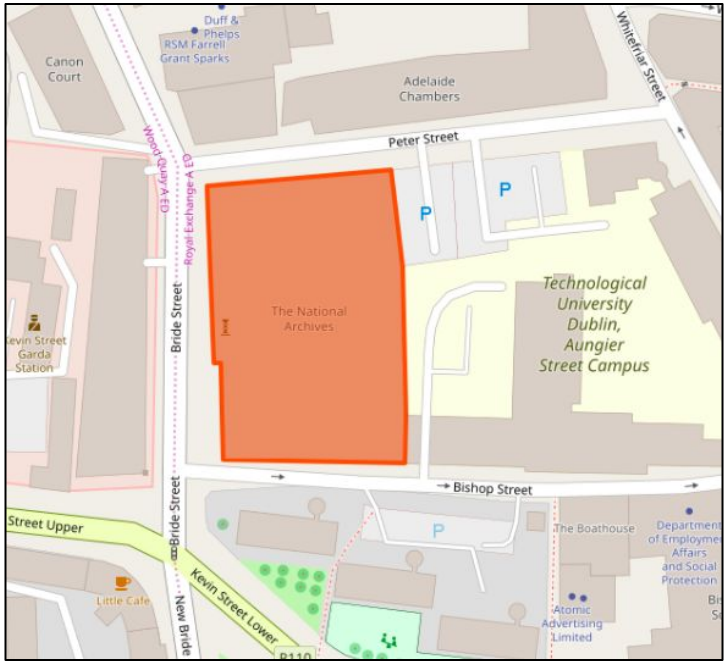
Man-made

Here a man made created landscape is fairly easy to generate or distinguish , because Man made construction have definite boundaries (which are like mostly rectangular) for example any building , agricultural land or even manmade parks

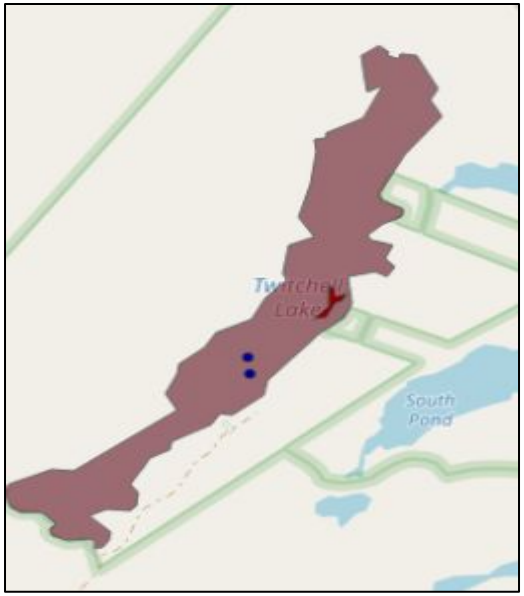
Naturally occuring

This is a naturally occurring lake and it does not have a very well define boundary as well as the number of vertices and the irregularity in the polygon are much greater.

Data Distribution for various Objects



A man made building , has approximately 4-5 vertices and the edge angles are almost at 90 deg



This is a naturally occurring lake and it does not have a very well define boundary as well as the number of vertices and the irregularity in the polygon are much greater.

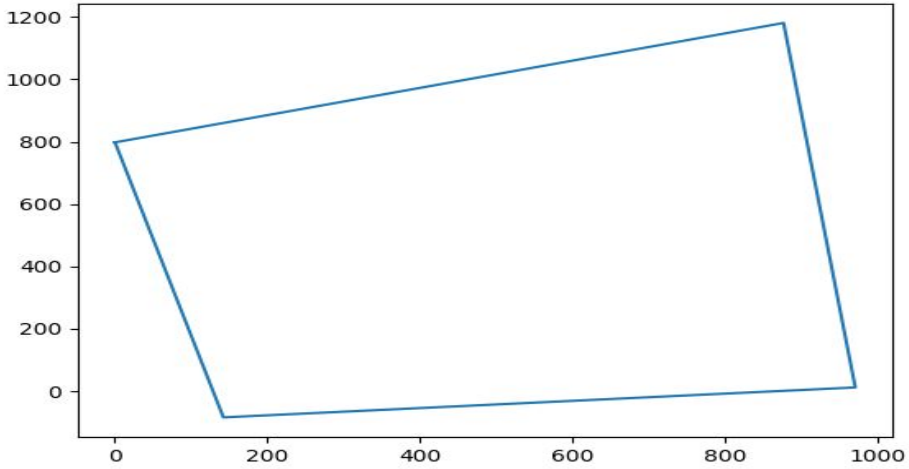
Data Distribution for various Objects

PESU CCBD

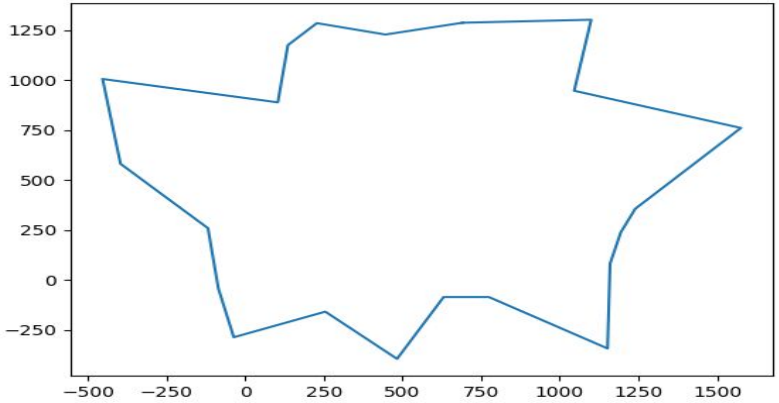
So we can say that Natural occurring can have any kind of skewness but man made should have a threshold.

So we can implement this in our working model by putting conditions such as the number of vertices for an agricultural land or a building should be less and whereas for natural objects like ponds , forests the number of vertices could be more and the irregularity should be high

Data Distribution for various Objects



This could be considered a manmade object like agricultural land



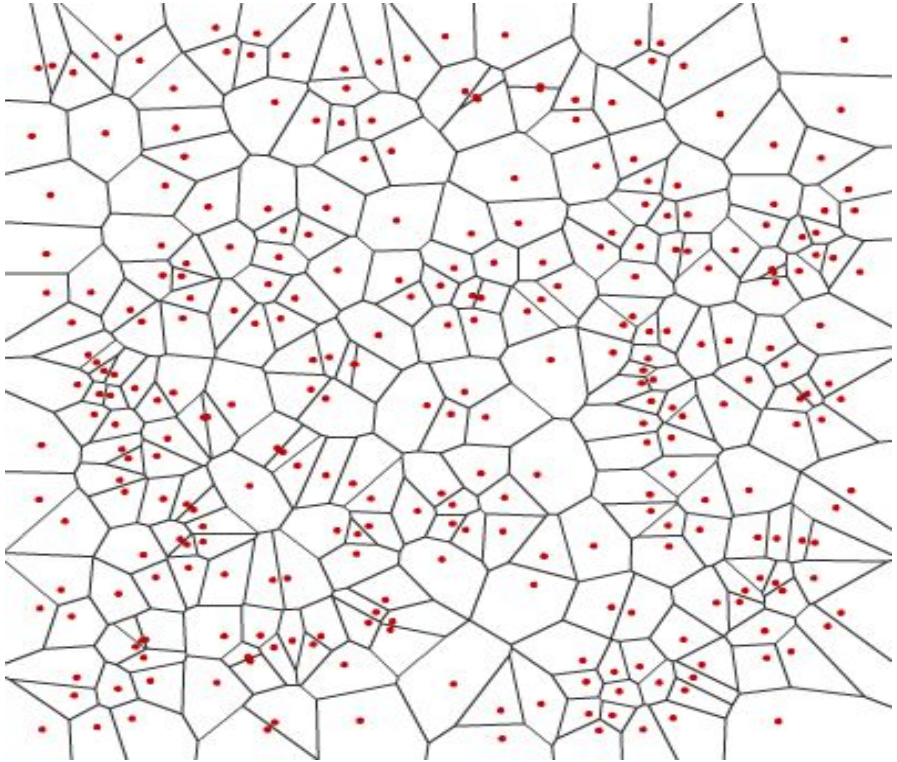
This could be considered a natural object like a pond or a lake

What we can do better

PESU CCBD

We can improve the algorithm of generating the polygon , by using the concave/ convex hull method or using knn method to form multiple polygons at once.

So instead of generating fixed amount of points at once , we can generate a large amount of points altogether and form a join the points using knn method or by using the concave/convex hull method to form like a constellation of polygon.



PES University

Cloud Computing and Big Data

THANK YOU!