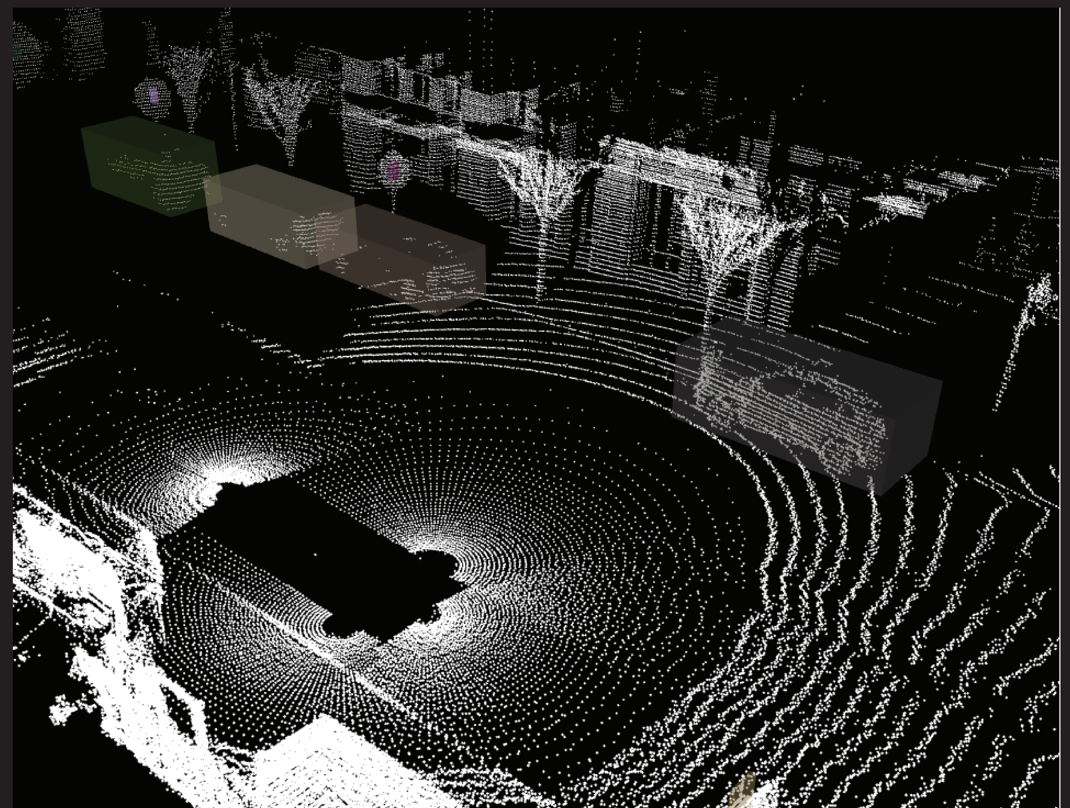
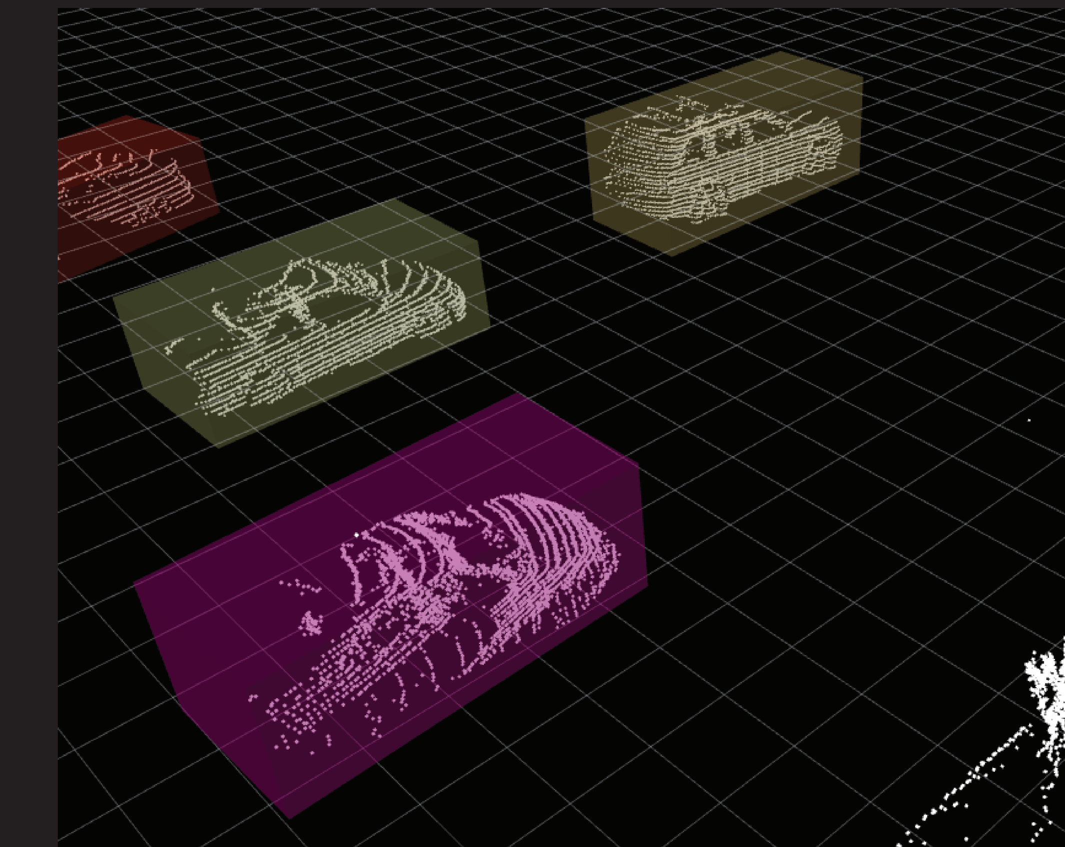


OBJECT DETECTION AND CLASSIFICATION IN POINTCLOUD DATA

Amy Phung & Connor Novak

Machine Learning Fall 2019



FEATURES

Volume - Calculated using 2D hull area multiplied by distance between lowest and highest Z point

Density - Calculated using number of points in region divided by volume

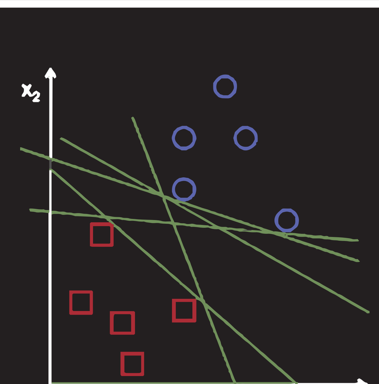
Eigenvalues (x,y,z) - Calculated from covariance matrix of cluster

Max Intensity - Maximum lidar intensity value within cluster

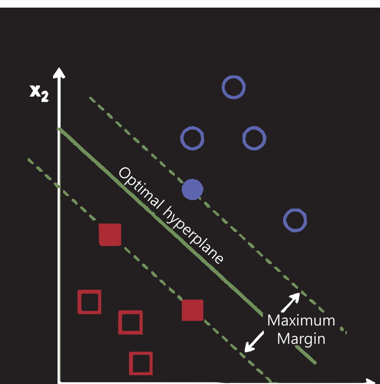
Average Intensity - Average intensity of points within cluster

Intensity Variance - Variance of intensity of points within cluster

MODEL



Linear Regression



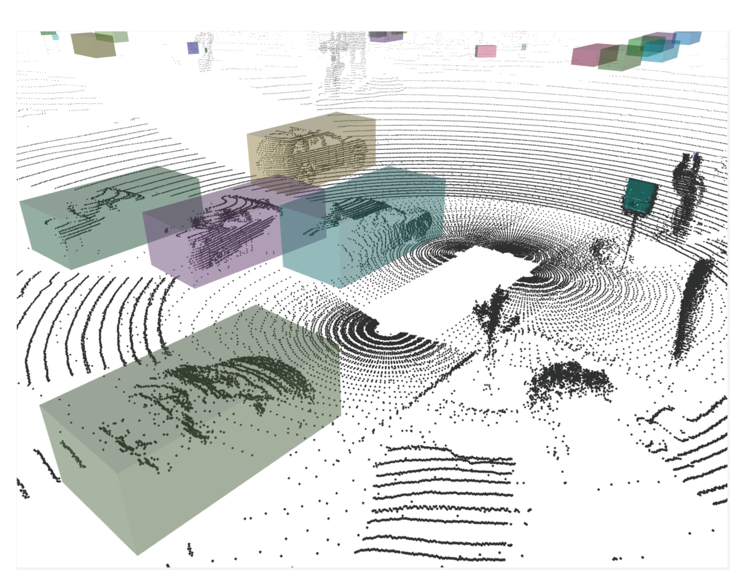
SVM
(Support Vector Machine)

We used both a linear regression model and an SVM to classify our dataset with similar accuracies for both models. We chose to use linear regression first to get a better understanding of our dataset, then progressed to an SVM based on a survey of popular approaches to the problem.

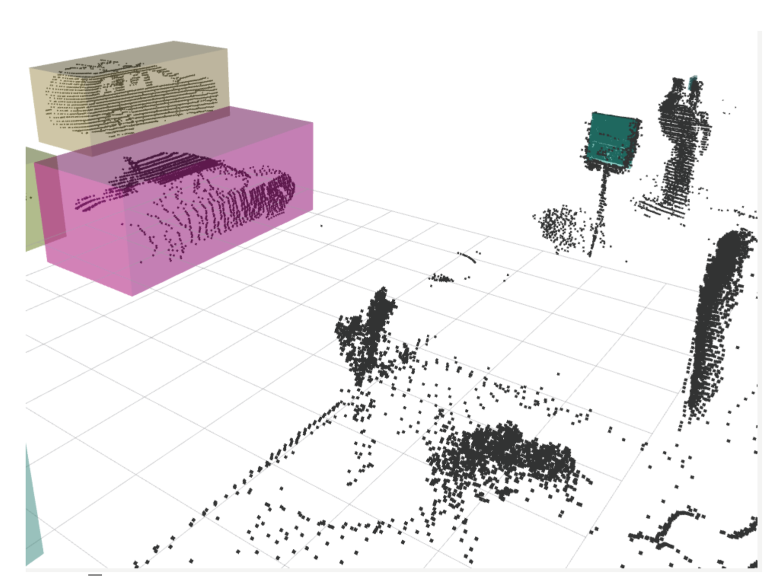
ABSTRACT

In this project, we attempted to use machine learning to detect and classify different objects in a 3D pointcloud using readily available labeled datasets from Waymo. After picking a rudimentary feature set, we applied a linear regression model and an SVM to our data to classify the points with relatively little success. Upon closer inspection of the data, we found that we were able to significantly improve our results by focusing on the quality of the features we chose to observe, rather than looking into a more complex model.

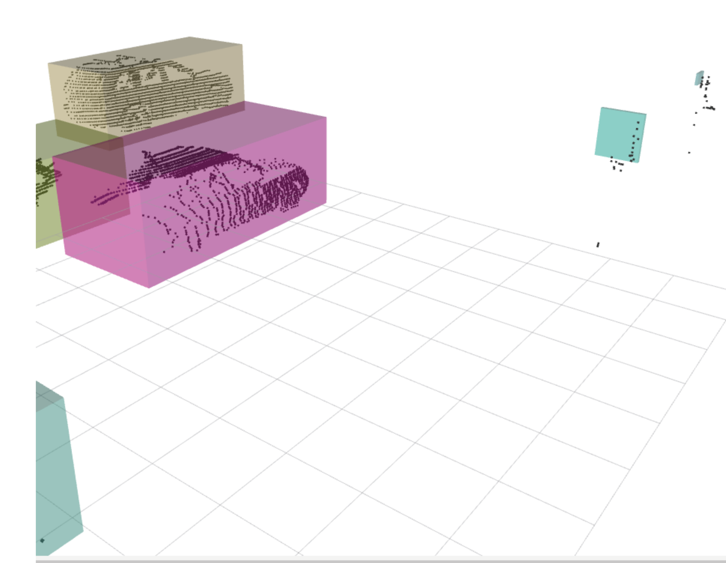
DATA PIPELINE



1 | Load in data



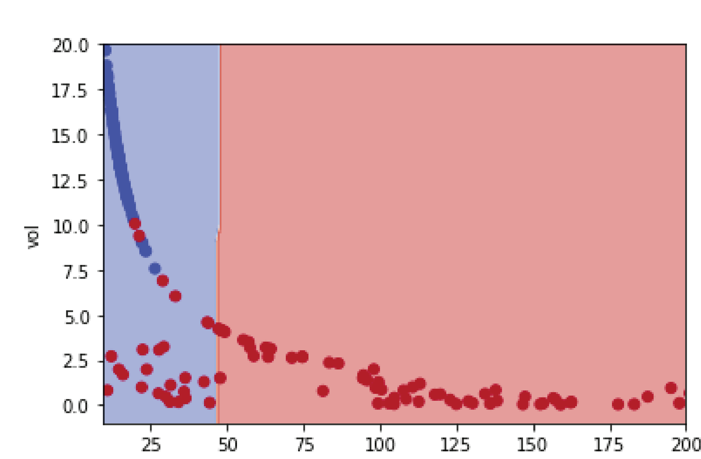
2 | Remove ground plane



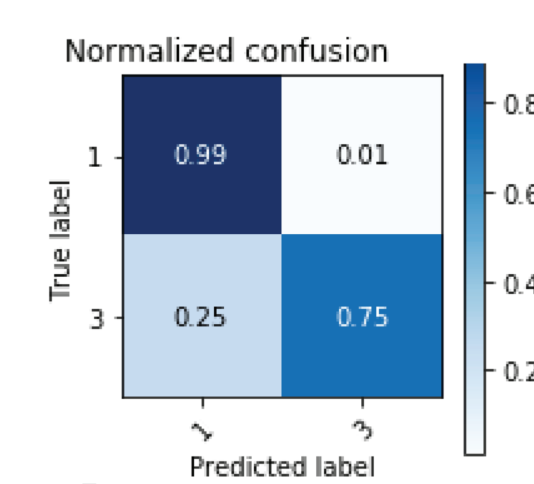
3 | Segment pointcloud by bounding boxes

```
def extract_cluster_parameters(cluster, display=
    """Calculate features for net training from
    containing all features.

    Args:
        cluster: ndarray (n * 4) of cluster poir

    Returns:
        parameters: list of parameters [e_x, e_y
            x = cluster COM along x axis
            y = cluster COM along y axis
```

4 | Compute Features



5 | Use features to train model



Accuracy: 0.87

6 | Check training accuracy

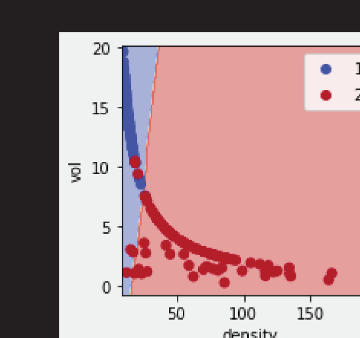
RESULTS

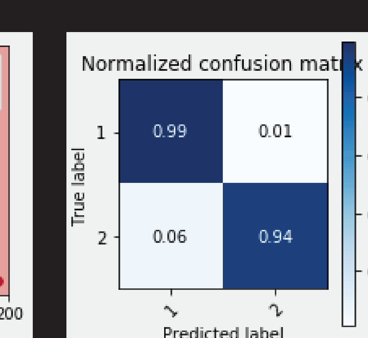
We found that the accuracy of our model was more strongly influenced by the quality of our features than the quality of our model. Because of this, we spent most of our time focusing on how the accuracy changed based on what features we put into the model. Below we share some of the results using the same model but different inputs.

Each of the models below were trained using a linear SVM using density and volume as features.

Cars (1)
People (2)

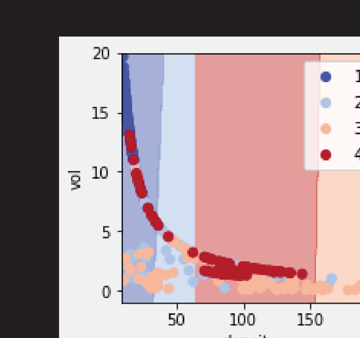
Accuracy: 96.5%

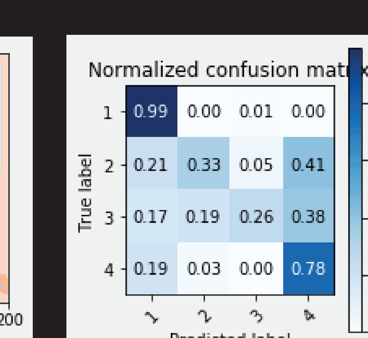




Cars (1)
People (2)
Signs (3)
Bikes (4)

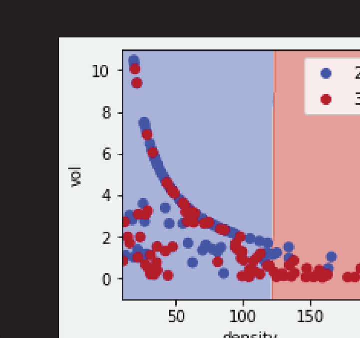
Accuracy: 59%

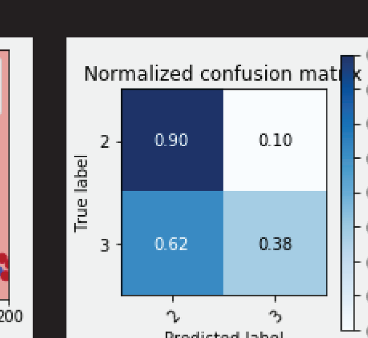




People (2)
Signs (3)

Accuracy: 64%





NEXT STEPS

- Nonlinear SVM
- Histogram-based ground model
- Bounding box cluster padding
- Application to custom 3D dataset

References: Waymo Dataset https://waymo.com/open/data/ | Velodyne Lidar-based 3D Object Perception https://velodynelidar.com/lidar/hdlpressroom/pdf/Articles/LIDAR-based%203D%20Object%20Perception.pdf | Classifying 3D objects in LiDAR point clouds with a back-propagation neural network https://link.springer.com/content/pdf/10.1186%2Fs13673-018-0152-7.pdf