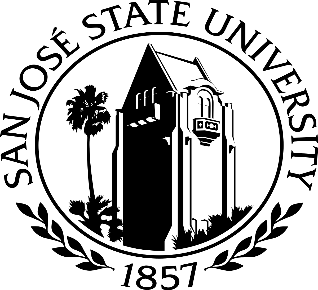
Project Report  
Milestone 1 and Milestone 2

CMPE 257: Machine Learning  
San Jose State University  
Master of Science in Data Analytics



**Group 8**

Aarathi Nadathur – 013704405  
Amar Ippili- 013728975

Divya Puraswani - 013755391

Sai Chaitanya Tolem - 013008788

Sameer Rajput - 013802841

Shubh Johri - 013750672

Selected Machine Learning Algorithm: CNN and Residual Neural Networks

Link: Google Colaboratory:

Milestone 1: <https://colab.research.google.com/drive/1RsvUuIq2oEIHBJ0gHIYIuvorLvGnlHSG>

Milestone 2:

<https://colab.research.google.com/drive/1k_yORhgKA_UJj2sLu65EjLeuT-ytJtOs>

**Who Did What!**

|  |  |  |
| --- | --- | --- |
| **Milestone 1** | | |
| **What** | **Description** | **Who** |
| Technical | 1. Data Preparation and EDA | Sai Chaitanya |
| 1. Data Validation, Data Preprocessing | Sai Chaitanya, Sameer Rajput |
| 1. Logistic Model | Amar Ippili |
| 1. Random Forest, KNN | Shubh, Aarathi |
| 1. Decision Tress, Naïve Bayes | Shubh, Aarathi |
| 1. SVM, CNN | Divya, Sameer, Chaitanya |
| Presentation | 1. PowerPoint Preparation | Divya, Amar |
| 1. In Class PowerPoint Presentation | Entire Team |
| Documentation | 1. Report Preparation | Sameer, Divya |
| 1. Report Editing | Shubh, Aarathi |

**Who Did What!**

|  |  |  |
| --- | --- | --- |
| **Milestone 2** | | |
| **What** | **Description** | **Who** |
| Technical | 1. Data Preparation and EDA | Sai Chaitanya |
| 1. Data Validation, Data Preprocessing | Sai Chaitanya, Sameer Rajput |
| 1. Logistic Model, CNN and Resnet:   bus (vehicle 1)+combinations from vehicle 2  bicycle (vehicle 1)+combinations from vehicle 2  train (vehicle 1)+combinations from vehicle 2  pickup\_truck (vehicle 1)+combinations from vehicle 2  motorcycle (vehicle 1)+combinations from vehicle 2 | Divya  Amar  Shubh  Sameer  Aarathi |
| Presentation | 1. PowerPoint Preparation | Aarathi, Divya |
| 1. In Class PowerPoint Presentation | Entire Team |
| Documentation | 1. Report Preparation and Report Editing | Shubh, Amar |

Table of Contents

[1. About the dataset 5](#_Toc7984499)

[2. Problem 5](#_Toc7984500)

[3. Software and libraries 5](#_Toc7984501)

[4. Data Preparation 5](#_Toc7984502)

[5. Data Validation 6](#_Toc7984503)

[6. Data Pre-processing 7](#_Toc7984504)

[7. Machine Learning Algorithms 7](#_Toc7984505)

[7.1 Logistic Regression 7](#_Toc7984506)

[7.2 KNN: 8](#_Toc7984507)

[7.3 Naïve Bayes: 9](#_Toc7984508)

[7.4 Random Forests: 10](#_Toc7984509)

[7.5 Decision Tree Classifier: 10](#_Toc7984510)

[7.6 SVM: 11](#_Toc7984511)

[8. Selected Machine Learning Algorithm: CNN 12](#_Toc7984512)

[8.1 What is CNN? 12](#_Toc7984513)

[8.2 Why is CNN better than other algorithms? 12](#_Toc7984514)

[8.2 What is Resnet? 15](#_Toc7984515)

[9. Conclusion 18](#_Toc7984516)

[10. Bibliography 19](#_Toc7984517)

# About the dataset

The CIFAR-100 dataset has 100 classes containing 600 images each. There are 500 training images and 100 testing images per class. The 100 classes in the CIFAR-100 are grouped into 20 super classes. Each image comes with a "fine" label (the class to which it belongs) and a "coarse" label (the superclass to which it belongs) (CIFAR 100 DATASET).

|  |  |
| --- | --- |
| **Super Class** | **Class** |
| Vehicle 1 | Bicycle, Bus, Motorcycle, Pickup truck, Train |
| Vehicle 2 | Rocket, Tank, Tractor, Streetcar, Lawn-mower |

# Problem

Milestone 1: Our goal is to perform a binary classification of images in vehicle 1 and vehicle 2 superclass pairs.

Milestone 2: The main goal of this project milestone is to design algorithms that would be able to predict a sub-class not available during the training stage.

# Software and libraries

**Software: Google Colaboratory Environment** (adventuresinmachinelearning, n.d.)

Google colaboratory environment is based on Python Jupyter notebooks which gives the user free access to Tesla K80 GPUs. It provides the ability to experiment with deep learning. To access the environment, you must have a signed in Google Drive account. The .ipynb files that you create will be saved in your Google Drive account. We have signed in using SJSU Email Id to access the environment and work with cifar100 dataset.

Python Libraries: Keras, Sklearn, Numpy, Matplotlib, Pandas, Scipy, Tensorflow

Keras is a high-level neural networks API, written in Python and we are running it on top of TensorFlow, CNTK, or Theano. It supported both convolutional networks and recurrent networks, as well as combinations of the two and ran seamlessly on CPU and GPU. (Keras Documentation, n.d.)

# Data Preparation

**(Milestone 1 and Milestone 2)**

Cifar 100 dataset consists of train, test and meta data. We loaded them into colaboratory and combined the images(X), coarse labels(Y) (0-19) data into data frame and filtered this data frame on target variable (vehicle1 =18, vehicle2 = 19) as in Figure 1.

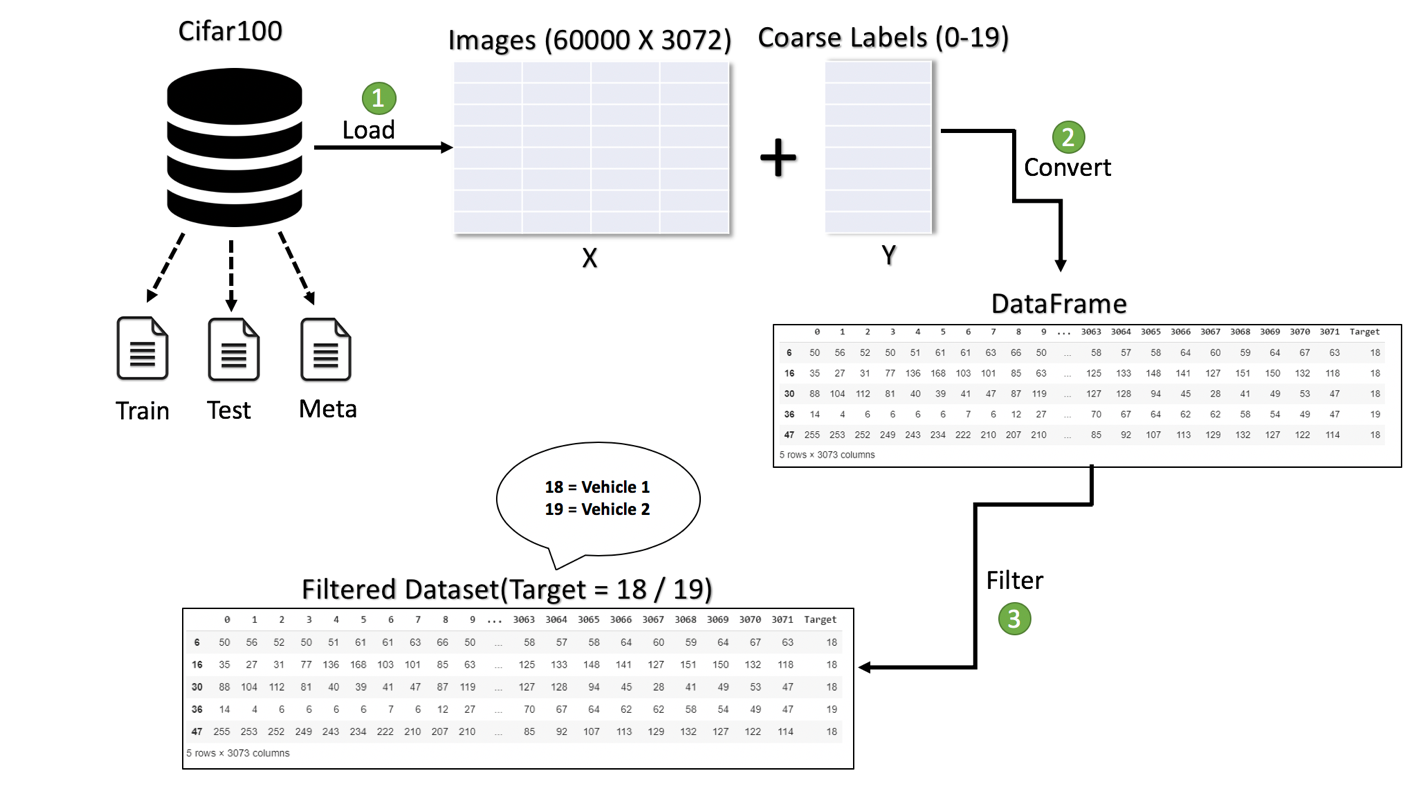


Figure 1

# Data Validation

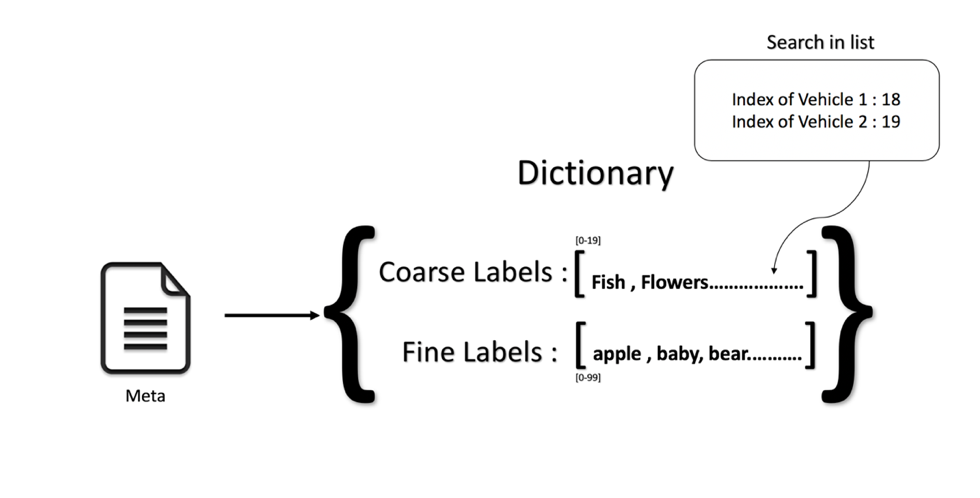


Figure 2

We used the meta data file in cifar 100 which consists of a serialized dictionary object to search for index of vehicle 1, vehicle 2 in the coarse label list as in Figure 2 and validated that 18 corresponds to vehicle 1 and 19 corresponds to vehicle 2 through code and we have crosschecked it by plotting few images from the filtered dataset.

# Data Pre-processing

**(for Milestone 1 and Milestone 2)**

To split the dataset into two sets, one for training and the other for testing, we have split the dataset in an 80–20 ratio such that 80% data is trained.

To reshape our dataset inputs (X\_train and X\_test) to the shape that our model expects when we train the model, we used astype to cast a pandas object into float. And then we used ‘one-hot-encode’ on our target variable using get\_dummies to convert categorical variable into dummy variables and to\_categorical to convert a class vector (integers) to binary class matrix. This means that a column will be created for each output category and a binary variable is inputted for each category.

**(for Milestone 2)**

Selection Of two Selected Sub- Classes As Testing Data:

A screenshot of a cell phone

Description automatically generated

Figure 3

# Machine Learning Algorithms

**(Milestone 1)**

## 7.1 Logistic Regression

**Accuracy Achieved**: 60.08%

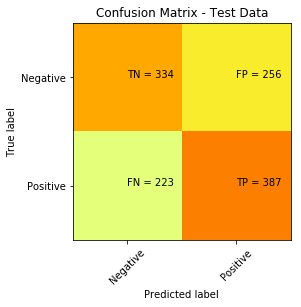


Figure 4

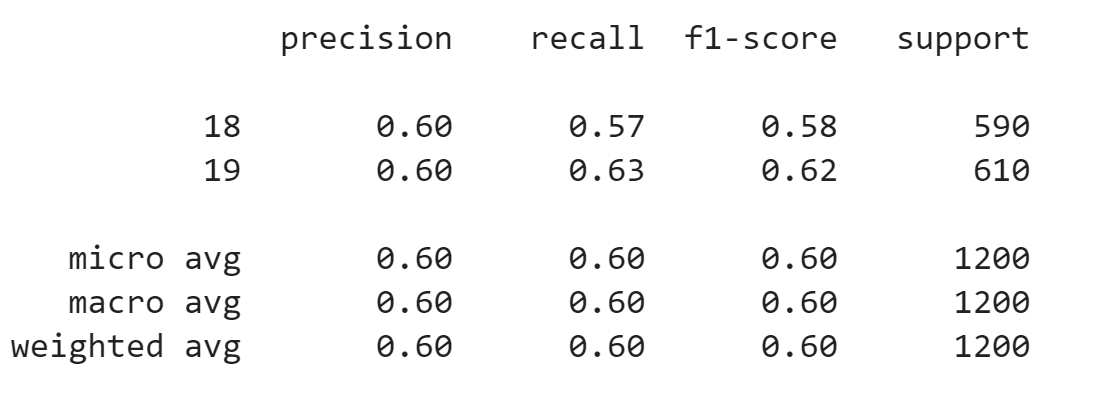


Figure 5

7.2 KNN:

**Accuracy Achieved**: 61.66%

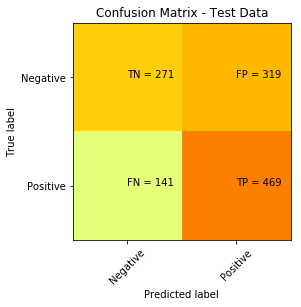


Figure 6

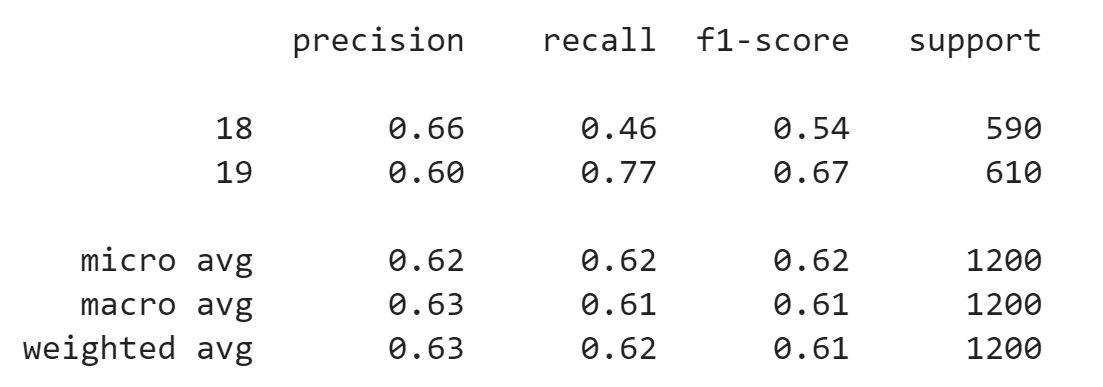


Figure 7

7.3 Naïve Bayes:

**Accuracy achieved:** 55.83%



|  |  |
| --- | --- |
| Figure 8    Figure 9 |  |
|  |  |

7.4 Random Forests:

**Accuracy achieved:** 69.83%

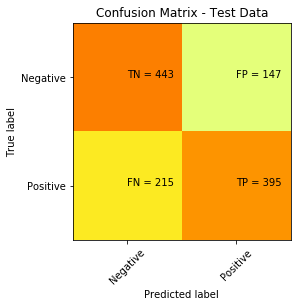


Figure 10

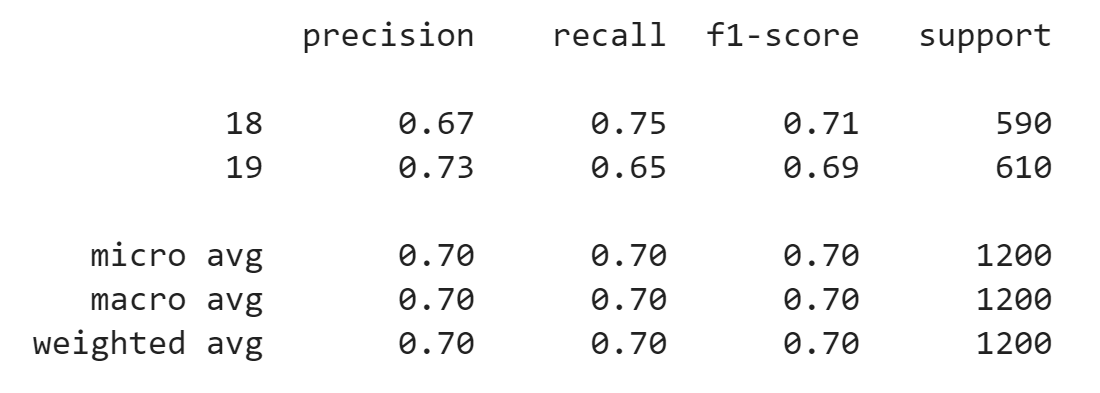


Figure 11

7.5 Decision Tree Classifier:

**Accuracy achieved:** 58.16%

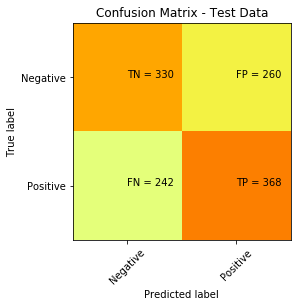


Figure 12

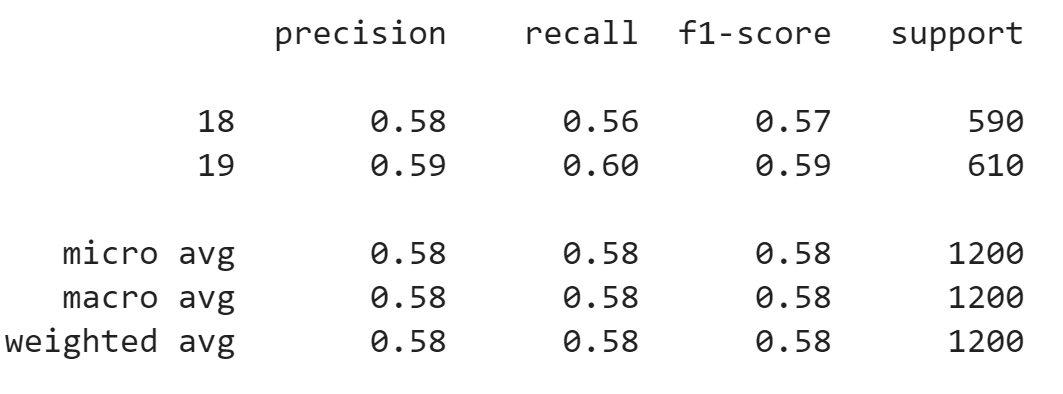


Figure 13

## 7.6 SVM:

**Accuracy achieved:** 68.88%

Confusion Matrix – Test Data

|  |  |  |
| --- | --- | --- |
| **Predicated Label** | **Negative** | **Positive** |
| **True Label** |
| **Negative** | TN = 696 | FP = 281 |
| **Positive** | FN = 335 | TP = 668 |

Figure 14

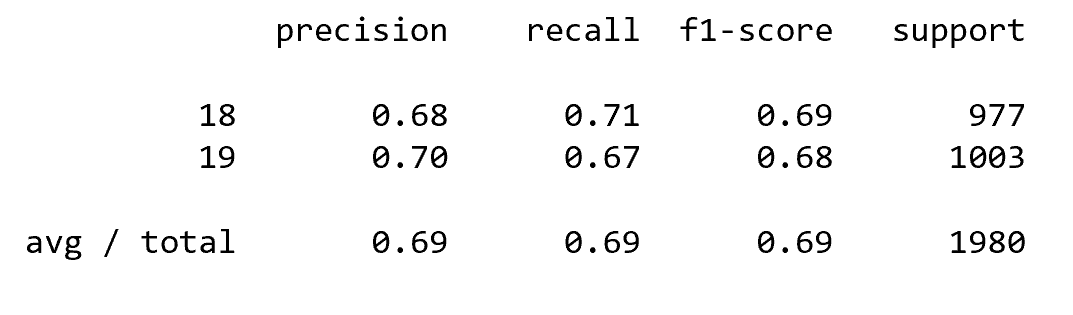


Figure 15

# Selected Machine Learning Algorithm: CNN

## 8.1 What is CNN?

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics. (Towards DataScience, n.d.)

Our CNN model runs with 200 epochs and batch size of 128 with steps per epoch as 18 having MaxPool and using dropout and image augmentation we tried to prevent overfitting of our model with a learning rate which is neither too high nor too low. (Towards DataScience, n.d.)



Figure 16

## 8.2 Why is CNN better than other algorithms?

The goal of CNN is to reduce the images into a form which is easier to process without losing features which are critical for getting a good prediction. This means that useful attributes from an already trained CNN can be extracted with its trained weights by feeding data on each level and tune the CNN a bit for the specific task.

**Accuracy achieved**: 83.67%

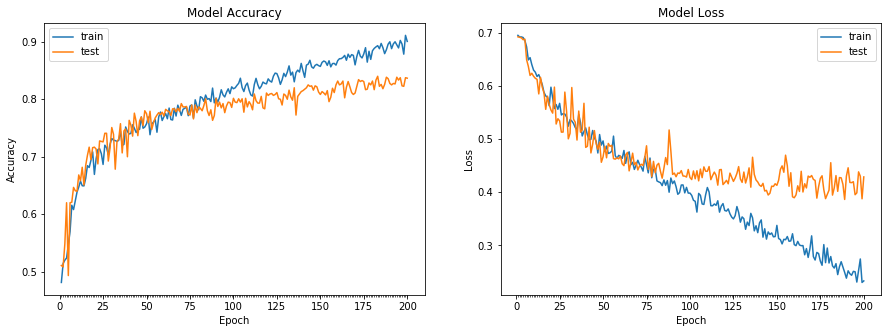


Figure 17

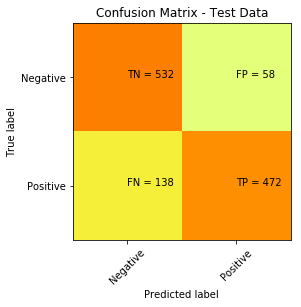


Figure 18

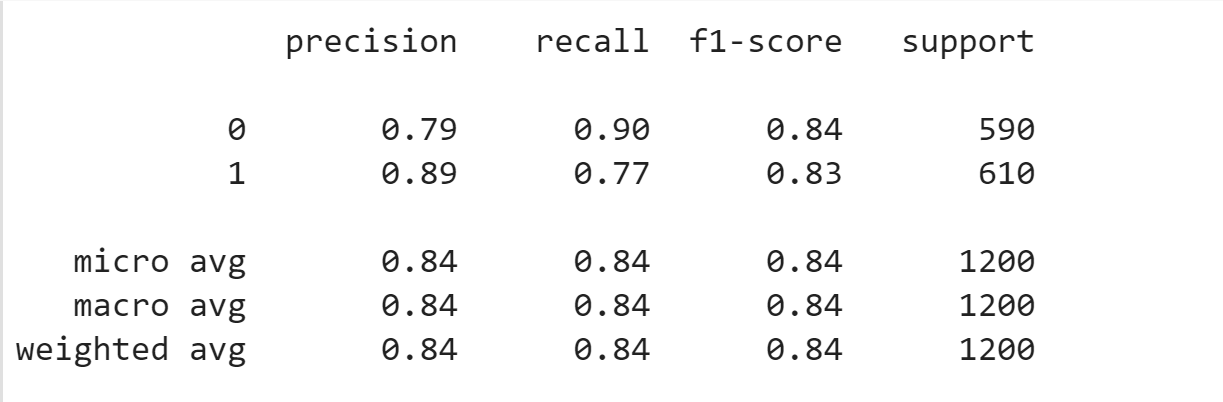
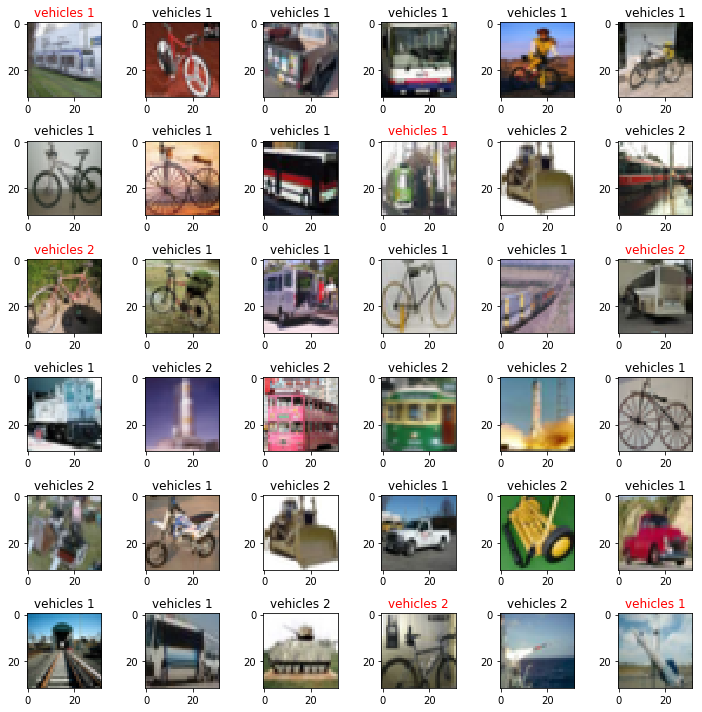


Figure 19

Output



*Figure 20*

**Milestone 2**

Less Accuracies from Model in Milestone 1 made us work on one of the classic networks- Resnet.

## What is Resnet?

Residual Networks makes it possible to train up to hundreds or thousands of layers and still achieve compelling performance which can be used as a backbone for many computer vision tasks.

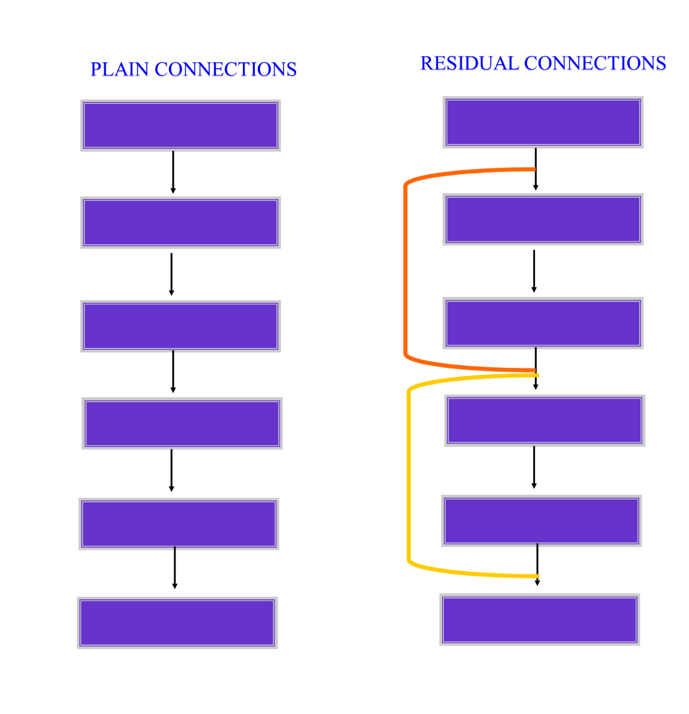


Figure 21

The Resnet in milestone 2 comprises of 29 layers using Batch normalization as a technique for preventing overfitting and understanding concept of dynamic learning rates where callbacks ReduceLROnPlateau in (keras) function was applied at given stages of the training procedure such that learning rate can be reduced when a metric has stopped improving. Keeping parameter patience=5, number of epochs can be decided. Resnet also helps in preserving the problem of vanishing gradient and regularizer kernel is used as well.

**Selected Pair:**

We have selected motorcycle from vehicle1 and lawn\_mower from vehicle2 as they closely resemble cycle, streetcar of vehicle 1, vehicle 2 respectively and our CNN model should be able to classify them easily even though motorcycle, lawn\_mower is not present in the training dataset.

Summarization Chart for 25 trails:

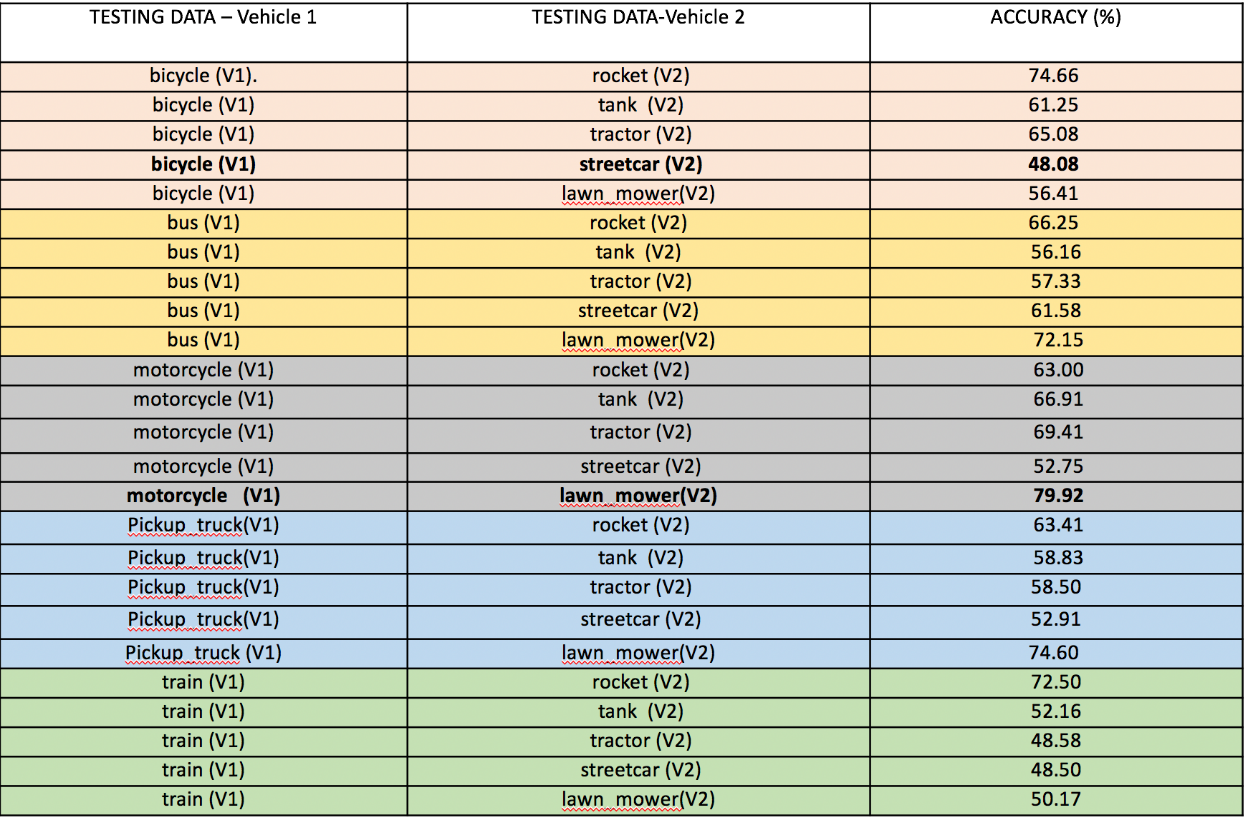


Figure 22

The best pair is motorcycle (vehicle 1) and lawn\_mower (vehicle 2) with accuracy of 79.92%.

Confusion matrix, ROC curve and Classification report for which is as below:

A screenshot of a cell phone

Description automatically generated

Figure 23

A screenshot of a cell phone

Description automatically generated

Figure 24

A picture containing object

Description automatically generated

Figure 25

**Note:** The model is saved when it reached 79.92% accuracy at 11th epoch automatically and reloaded this saved model to plot the confusion matrix, classification report and ROC curve.

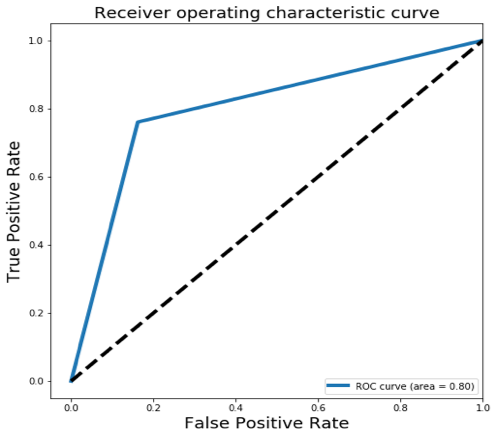


Figure 26

Output: 8 out of 36 images are misclassified in a random plot of images.

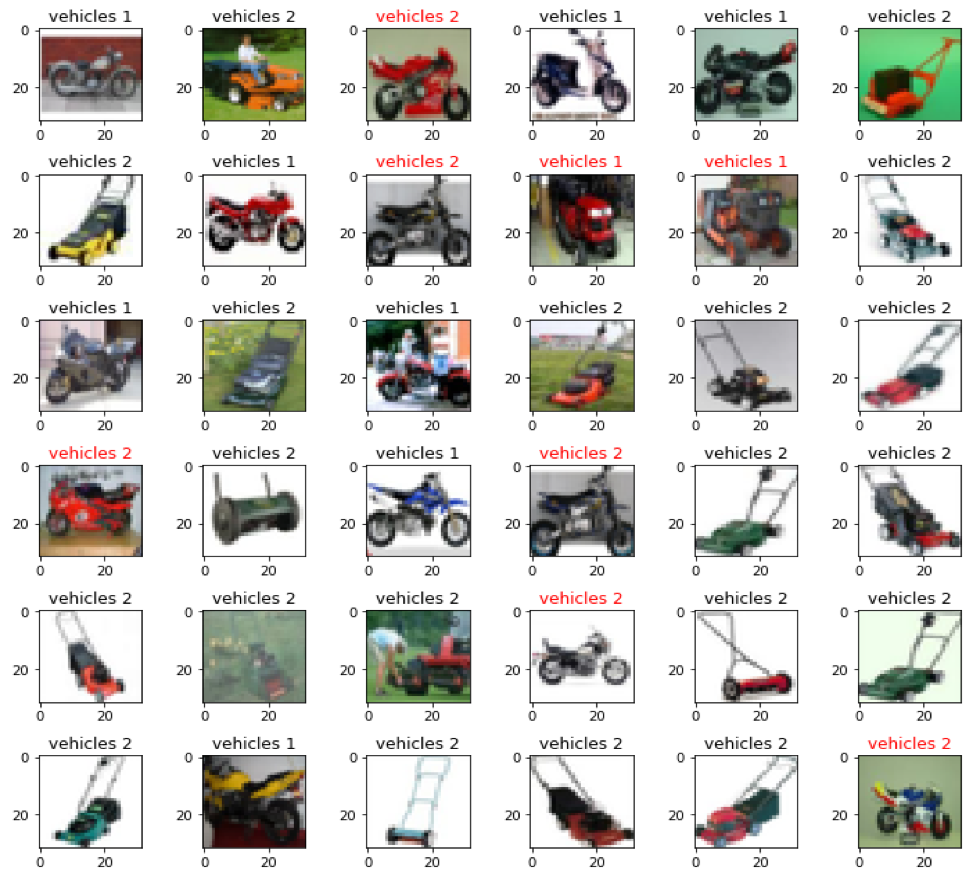


Figure 27

# Conclusion

**(Milestone 1)**

Accuracy, Computation and loss graphs matter. SVM Algorithm took enough time for generating results with accuracy of nearly 70%. Logistic regression and other algorithms did not generate optimal accuracy and the loss graph which made us decide to use CNN for achieving goal of our project.

Initial CNN model did not result in good accuracy and also was overfitted as it was run without using data augmentation. Later we tried to prevent overfitting and also simultaneously increase accuracy by tuning CNN parameters of learning rate, number of epochs, batch size, steps per epoch, using data augmentation and dropout which should be in 20%-50% range. Trying different layer types and number also helped in achieving accuracy.

**(Milestone 2)**

CNN model in milestone 1 did not result in good accuracy. Later we tried to work on one of the classic networks- Resnet. Batch normalization layer prevent overfitting and also simultaneously increased accuracy by tuning CNN parameters of dynamic learning rate (using callbacks), number of epochs, batch size, steps per epoch. Trying “more number of hidden layers, better the accuracy” we used 29 layers which helped in achieving accuracy but loss graph after 15 epochs was not optimal.

# Bibliography

(n.d.). Retrieved from https://adventuresinmachinelearning.com/introduction-to-google-colaboratory/

(n.d.). Retrieved from https://adventuresinmachinelearning.com/introduction-to-google-colaboratory/

(n.d.). Retrieved from adventuresinmachinelearning: https://adventuresinmachinelearning.com/introduction-to-google-colaboratory/

(n.d.). Retrieved from Keras Documentation: https://keras.io/

(n.d.). Retrieved from Towards DataScience: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

(n.d.). Retrieved from Towards DataScience: https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9

*CIFAR 100 DATASET*. (n.d.). Retrieved from CIFAR 100 DATASET: https://www.cs.toronto.edu/~kriz/cifar.html

<https://keras.io/examples/cifar10_resnet/>