Traffic Sign Recognition

Anshul Bhardwaj

Problem Statement

- Automated Traffic Sign recognition is an important part of self-driving vehicles.
- Traffic Signs can provide range of variations between classes in terms of colors and shape.
- In this project, I develop Deep Learning algorithms that will train on German Traffic Signs image dataset and them use these algorithms to classify unlabeled Traffic Signs images. The deep learning models will be built using Convolutional Neural Network and Transfer Learning.

Clients / Intended Audience

- This model can be used by anyone who wants to what makes an automated self driving system.

Dataset

- The dataset used for this project is taken from Kaggle
- https://www.kaggle.com/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign
- The dataset contains around 40000 train images and 12000 test images. Metadata is also provided.

Exploratory Data Analysis

- Distribution of Classes

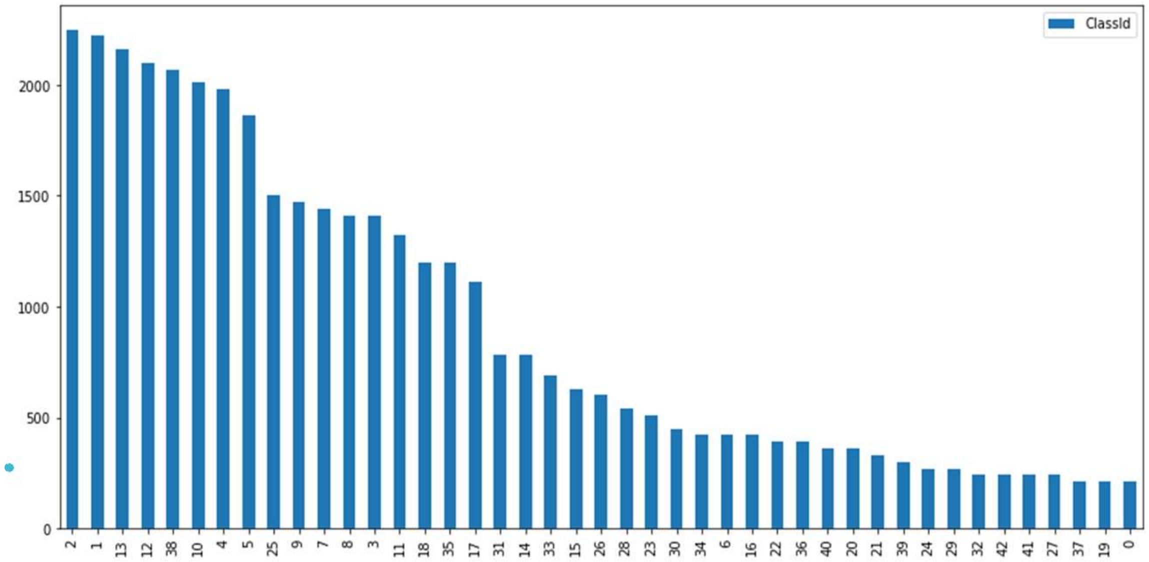
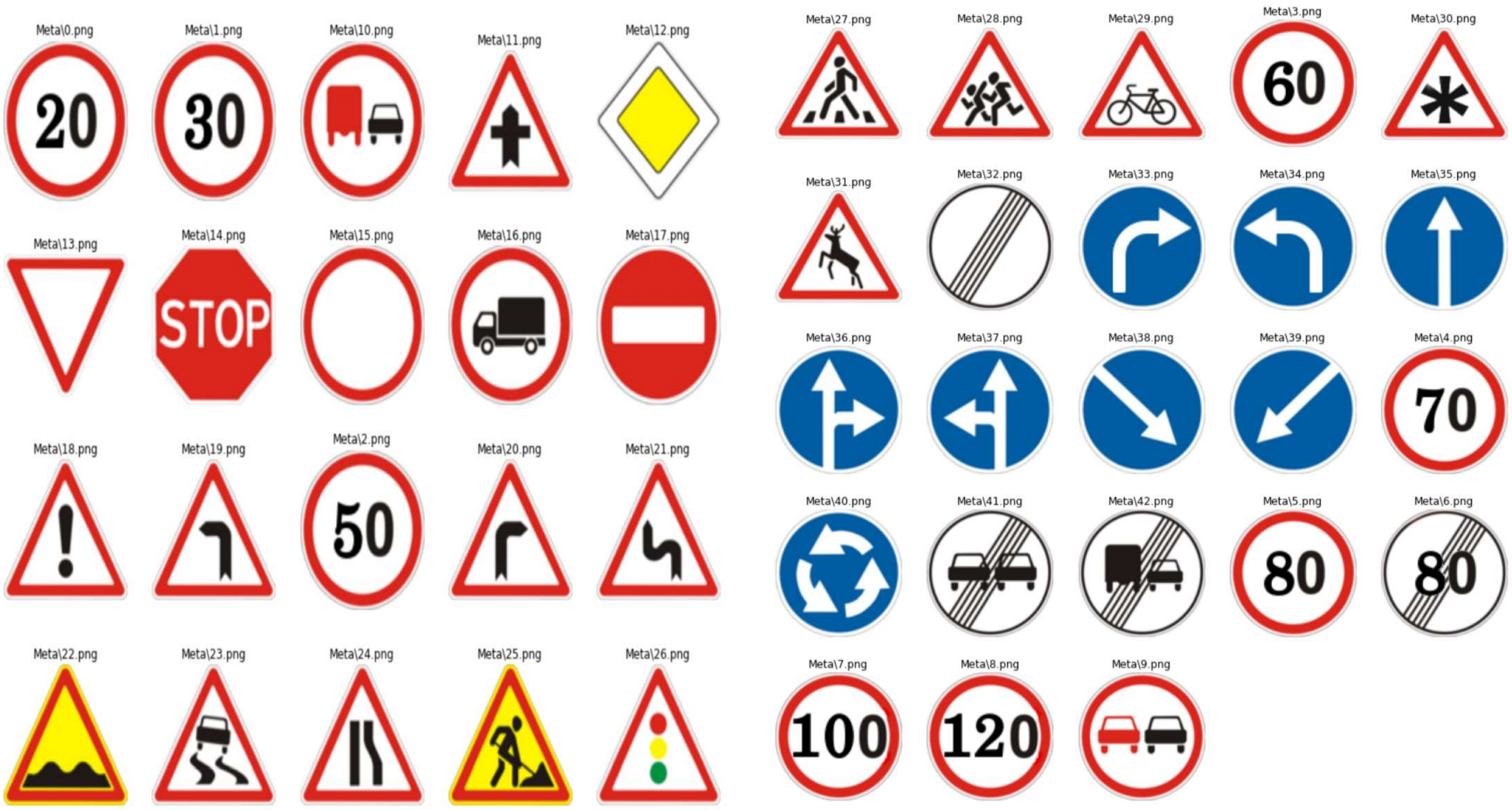


Image : ClassId vs Frequency

- This barplot depicts ClassIds and their respective frequency in image dataset.
- Images with ClassIds 2, 1, 13 etc. had the highest frequency (>2000)
- Images with ClassIds 37, 19, 0 etc. had the least frequency (~=200)

Meta Images



Images : Meta Images of 43 Image Classes of Input dataset.

- The above are the 43 Meta Images with their respective ClassIDs

Image Augmentation

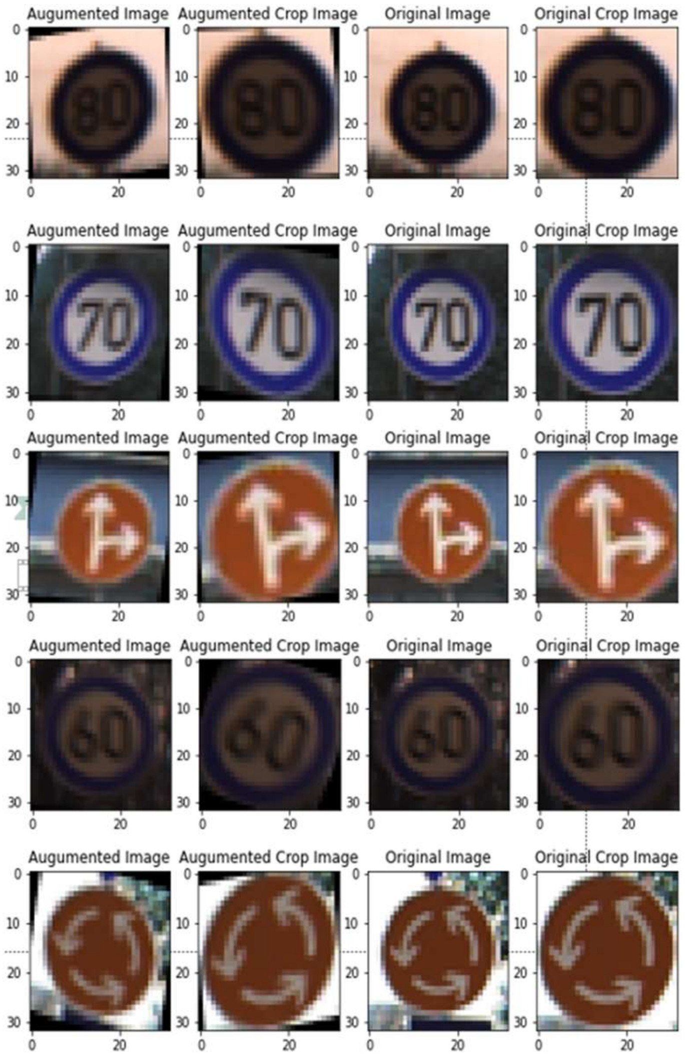


Image : Sample Augmented Images and Original Images.

- These are some of the Augmented Images and their respective Original Image
- I have done two affine transformation (Rotation and Shear) on all Image to increase number of Image.
- Image Augmentation also helps in generalizing the model.

Code snippet :

```
seq = iaa.Sequential([ iaa.Affine(
rotate=(-20, 20)
shear = (10,-10)) ])
```

Training

- I build three model using CNN and Transfer Learning
- First model was build using Transfer Learning with ResNet50V2 Model
- Second model was built using CNNs
- Third model was build using Transfer Learning with VGG19 Model
- Accuracy was used as metric in all three models
- ADAM optimizer was used in all three models

Model Performances / Histories

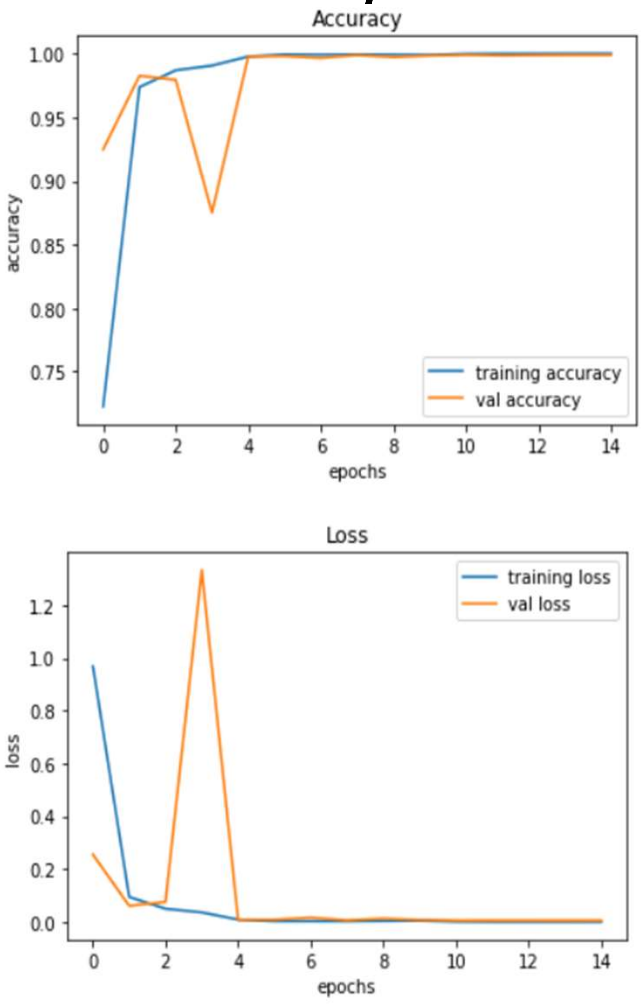


Image 1: ResNet50V2 Model accuracy and Loss plot

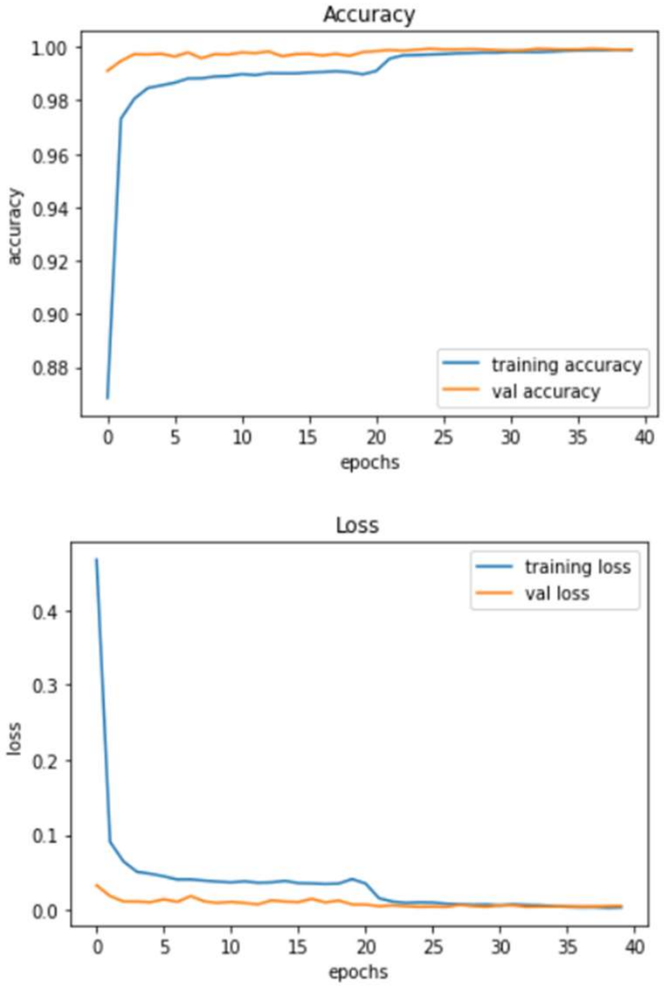


Image 2: CNN based Model accuracy and Loss plot

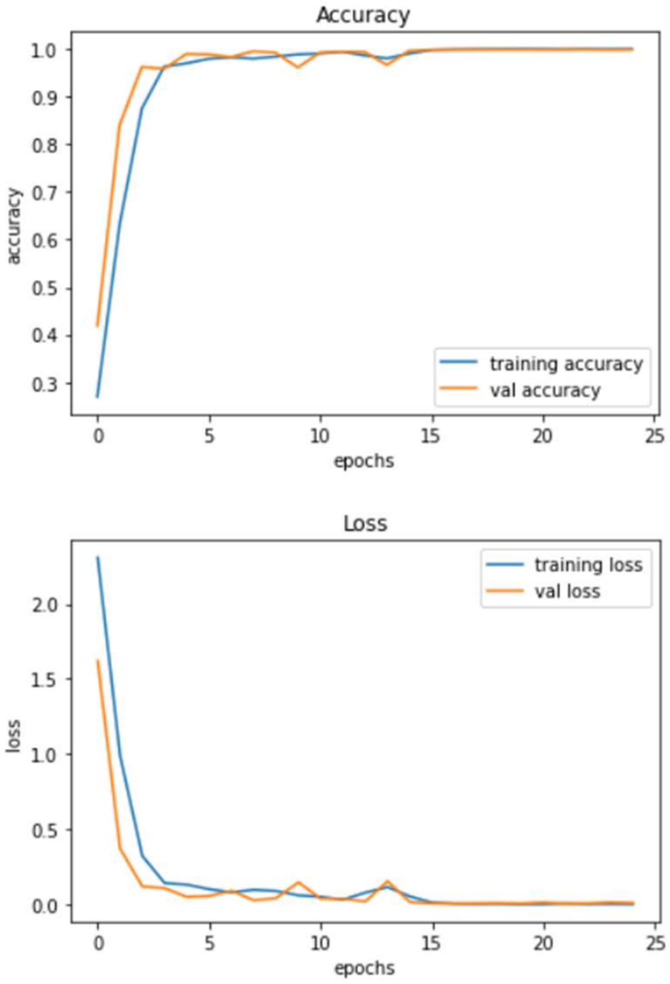


Image 3: VGG19 Model accuracy and Loss plot

Model Testing

- Model accuracies for the models were as follows

Model Name	Accuracy
CNN Model	98.72%
ResNet50V2 Model	96.61%
VGG19 Model	98.33%

- CNN model performed the best.

Confusion Matrix Heatmap for CNN model

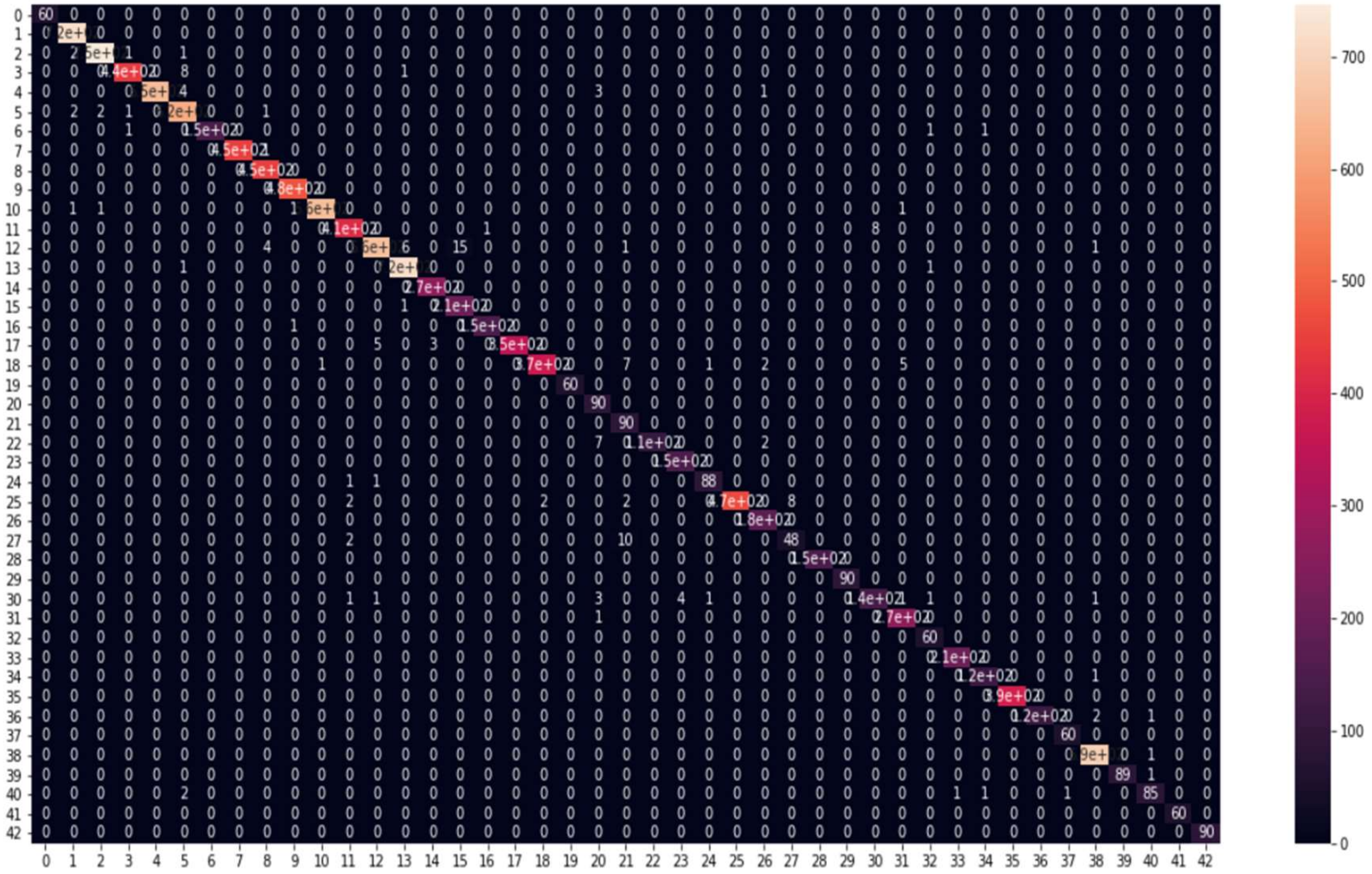


Image : Confusion Matrix for CNN based Model

Important Observations :

- 10 Images belonging to ClassId 27 are being assigned to ClassId 21.
- This CNN model's overall performance is really good with no other significant misclassifications.

Ensemble of the Models

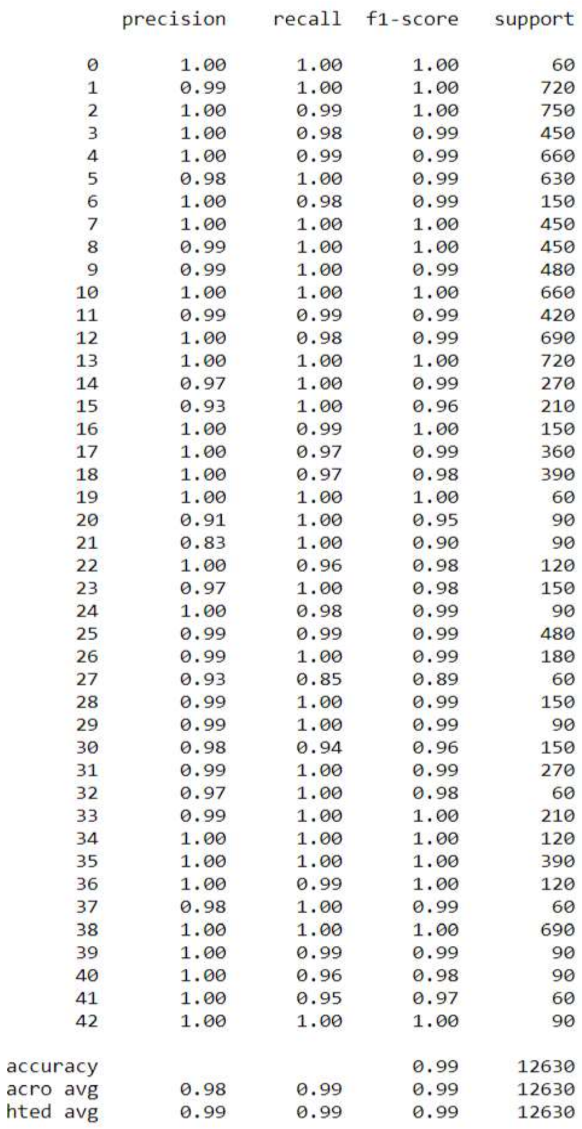
- Ensembling of the models had shown the significant improvement in accuracy

Ensemble model performance were as follows –

Type of Ensemble	Accuracy
Mean Ensemble (Soft Voting)	99.07 %
Weighted Ensemble	99.10 %
Mode Ensemble (Hard Voting)	98.72 %

- Weighted Ensemble had shown highest accuracy.

Classification Report of Weighted Ensemble



```
          precision    recall  f1-score   support

       0       1.00      1.00      1.00        60
       1       0.99      1.00      1.00       720
       2       1.00      0.99      1.00       750
       3       1.00      0.98      0.99       450
       4       1.00      0.99      0.99       660
       5       0.98      1.00      0.99       630
       6       1.00      0.98      0.99       150
       7       1.00      1.00      1.00       450
       8       0.99      1.00      1.00       450
       9       0.99      1.00      0.99       480
      10       1.00      1.00      1.00       660
      11       0.99      0.99      0.99       420
      12       1.00      0.98      0.99       690
      13       1.00      1.00      1.00       720
      14       0.97      1.00      0.99       270
      15       0.93      1.00      0.96       210
      16       1.00      0.99      1.00       150
      17       1.00      0.97      0.99       360
      18       1.00      0.97      0.98       390
      19       1.00      1.00      1.00        60
      20       0.91      1.00      0.95        90
      21       0.83      1.00      0.90        90
      22       1.00      0.96      0.98       120
      23       0.97      1.00      0.98       150
      24       1.00      0.98      0.99        90
      25       0.99      0.99      0.99       480
      26       0.99      1.00      0.99       180
      27       0.93      0.85      0.89        60
      28       0.99      1.00      0.99       150
      29       0.99      1.00      0.99        90
      30       0.98      0.94      0.96       150
      31       0.99      1.00      0.99       270
      32       0.97      1.00      0.98        60
      33       0.99      1.00      1.00       210
      34       1.00      1.00      1.00       120
      35       1.00      1.00      1.00       390
      36       1.00      0.99      1.00       120
      37       0.98      1.00      0.99        60
      38       1.00      1.00      1.00       690
      39       1.00      0.99      0.99        90
      40       1.00      0.96      0.98        90
      41       1.00      0.95      0.97        60
      42       1.00      1.00      1.00        90

accuracy                           0.99     12630
acro avg       0.98      0.99      0.99     12630
hted avg       0.99      0.99      0.99     12630
```

Image : Classification Report of Weighted Ensemble Model

Observations –

- Most of the classes has very good f1 score of 1 or 0.99
- Precision for Class 21 is comparatively low.
- Recall for Class 27 is comparatively low.
- Overall f1-score is .99

Future Improvements

- More Image augmentation
- Class balancing through Image augmentation
- Increase the input image dimension
- Use More Image generators for better ram management
- Make the models reproducible.