Building Machine learning Models as Digit Recognizer

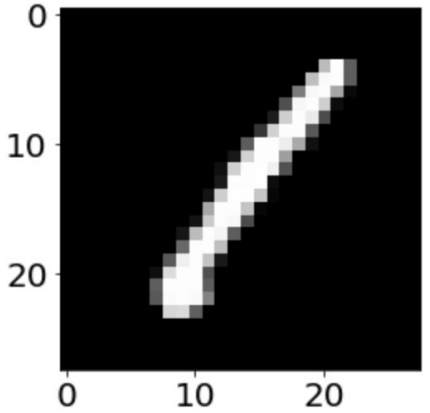
Group 8 - Xingyu Yang

Introduction

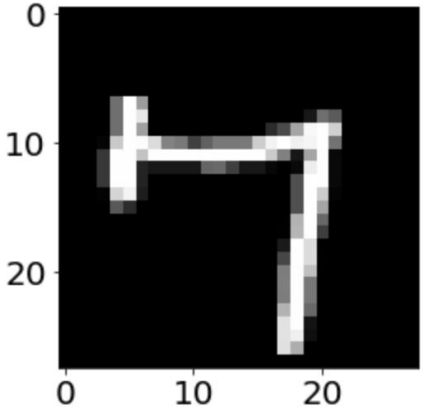
- Based on Kaggle Competition *Digit Recognizer - Learn computer vision fundamentals with the famous MNIST data*
  - https://www.kaggle.com/competitions/digit-recognizer/overview
- Proposal
  - Fit a model that predicts the hand-written digits most accurately
- Models
  - Support Vector Machines
  - Multiple Layer Perceptron Classifier
  - Dense Neural Network
  - Convolutional Neural Network

About the data

- Train data
  - 42000 rows, 785 columns
  - 1st column: label
  - 2nd - 785th column: pixel
- Test data
  - 28000 rows, 784 columns
  - 1nd - 784th column: pixel



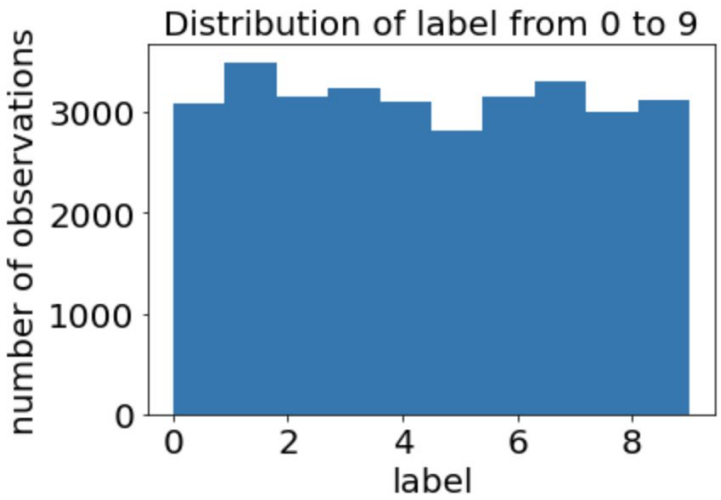
1st image



121th image

Data Preprocessing

- Train Set
  - X Train:
    - 31500 rows, 784 columns
    - Reshaped into shape 31500*28*28 for Neural Network
  - Y Train:
    - 31500 rows, 1 column
- Validation Set
  - X Validation
    - 10500 rows, 784 columns
    - Reshaped into shape 10500*28*28
  - Y Validation:
    - 10500 rows, 1 column



Support Vector Machines

- Tuning Hyperparameters:
  - Gamma: Scale, Auto
  - C: 0.1, 1, 10
- Selection of Hyperparameters
  - Gamma: Scale
  - C:10

C Value	0.1	1	10
Score	0.95276	0.97524	0.98086

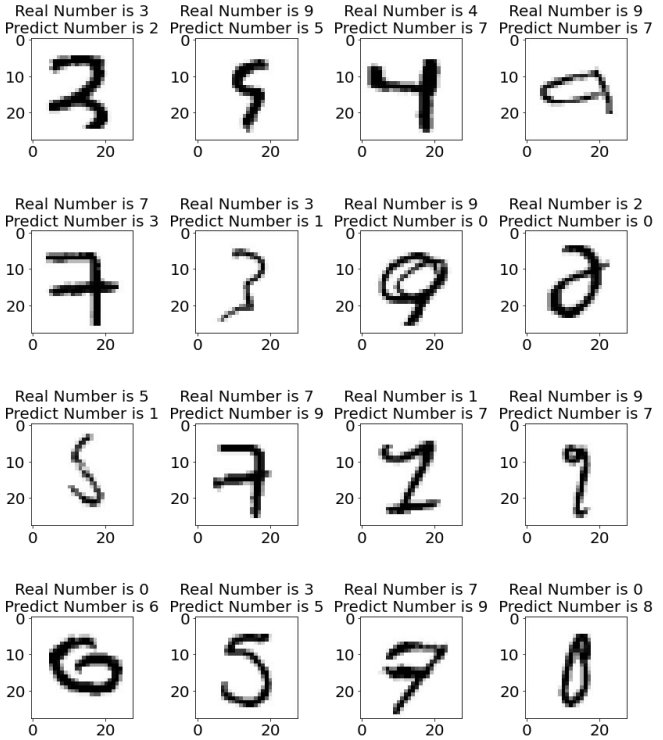
Support Vector Machines - results

Table of miss-predicted label

	True value	Predicted value
28	3	2
60	9	5
71	4	7
152	9	7
187	7	3
...	...	...
10066	3	5
10094	2	0
10281	7	4
10340	8	1
10425	1	4

201 rows × 2 columns

Plots of miss-predicted images



MLP Classifier

- Tuning Hyperparameters:
  - Activation function:
    - Relu
    - Tanh
- Selection of Hyperparameters
  - Activation function: Relu
  - Solver: Adam
  - Alpha: 0.0001

Activation function	Relu	Tanh
Score	0.96210	0.93152

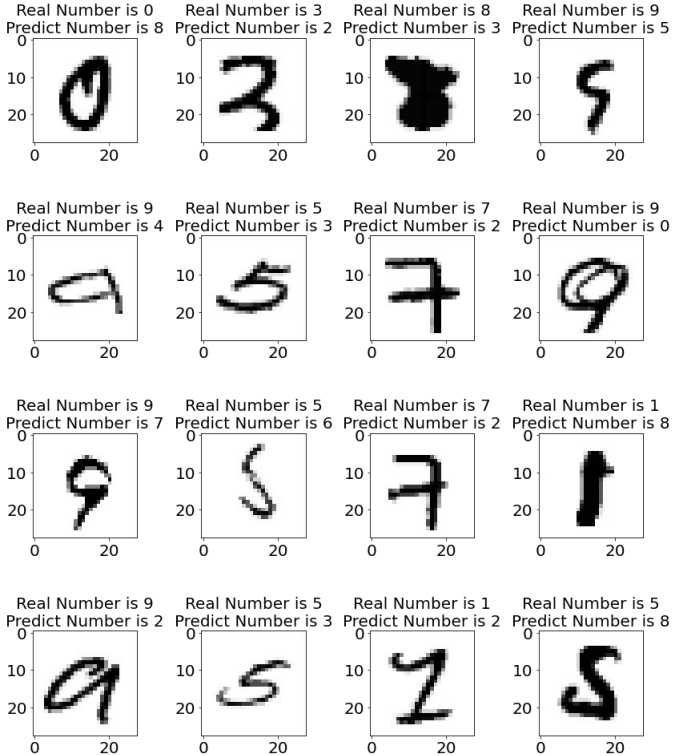
MLP Classifier - results

Table of miss-predicted label

	True value	Predicted value
13	0	8
28	3	2
51	8	3
60	9	5
152	9	4
...	...	...
10383	8	3
10425	1	4
10439	8	9
10464	2	7
10476	8	5

398 rows × 2 columns

Plots of miss-predicted images



Dense Neural Network

- Network Architecture
  - 5 hidden layers
  - Relu ->Tanh ->relu ->Tanh->Relu
  - Units: 50,75,50,100,50
  - Batch Normalization: after each transformation
- Training:
  - Batch size:20
  - Epoch: 20
  - Train epoch: 12

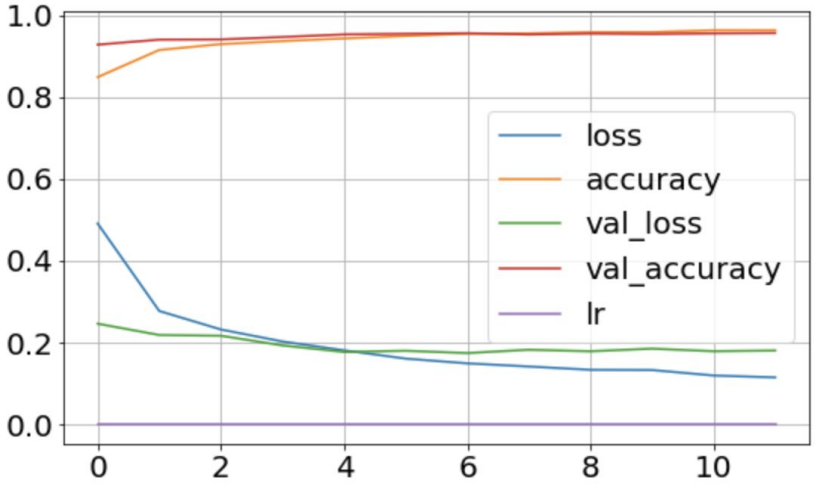
Layer (type)	Output Shape	Param #
flatten (Flatten)	(None, 784)	0
batch_normalization (BatchNormalization)	(None, 784)	3136
dense (Dense)	(None, 50)	39250
batch_normalization_1 (BatchNormalization)	(None, 50)	200
dense_1 (Dense)	(None, 75)	3825
batch_normalization_2 (BatchNormalization)	(None, 75)	300
dense_2 (Dense)	(None, 50)	3800
batch_normalization_3 (BatchNormalization)	(None, 50)	200
dense_3 (Dense)	(None, 100)	5100
batch_normalization_4 (BatchNormalization)	(None, 100)	400
dense_4 (Dense)	(None, 50)	5050
batch_normalization_5 (BatchNormalization)	(None, 50)	200
dense_5 (Dense)	(None, 10)	510

Dense Neural Network - results

Best model accuracy

```
329/329 [==============================] - 1s 3ms/step - loss: 0.1742 - accuracy: 0.9559
[0.17420229315757751, 0.9559047818183899]
```

Learning curve



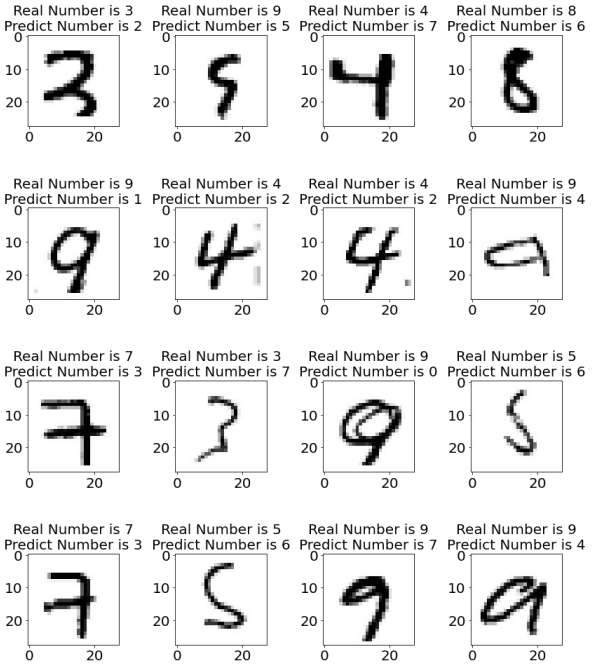
Dense Neural Network - results

Table of miss-predicted label

	True value	Predicted value
28	3	2
60	9	5
71	4	7
89	8	6
110	9	1
...	...	...
10366	6	5
10413	0	6
10425	1	4
10428	5	6
10476	8	5

463 rows × 2 columns

Plots of miss-predicted images



Convolutional Neural Network

- Network Architecture
  - 3 Conv2D
  - 3 MaxPooling
  - 3 Batch Normalization
  - 2 dropout with a rate of 0.2
- Training:
  - Batch size:20
  - Epoch: 20
  - Train epoch: 4

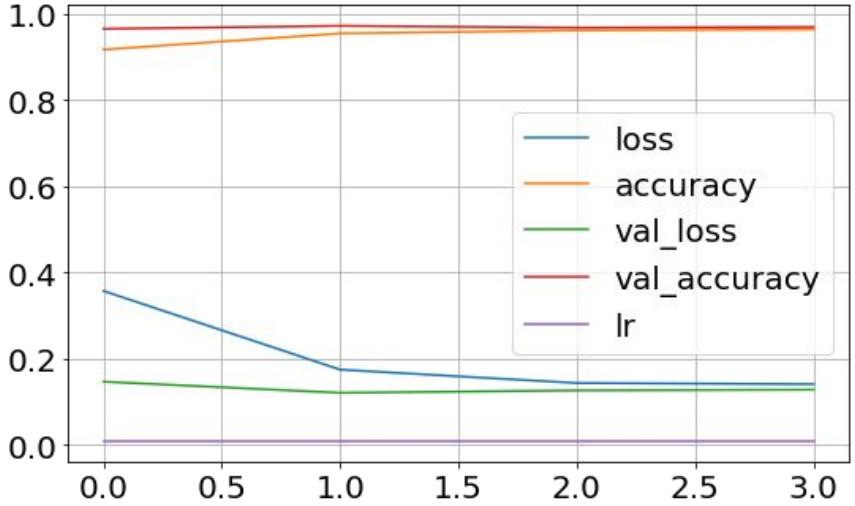
conv2d_3 (Conv2D)	(None, 26, 26, 64)	640
max_pooling2d_3 (MaxPooling 2D)	(None, 13, 13, 64)	0
batch_normalization_9 (Batc hNormalization)	(None, 13, 13, 64)	256
conv2d_4 (Conv2D)	(None, 11, 11, 128)	73856
max_pooling2d_4 (MaxPooling 2D)	(None, 5, 5, 128)	0
batch_normalization_10 (Bat chNormalization)	(None, 5, 5, 128)	512
dropout_2 (Dropout)	(None, 5, 5, 128)	0
conv2d_5 (Conv2D)	(None, 3, 3, 256)	295168
max_pooling2d_5 (MaxPooling 2D)	(None, 1, 1, 256)	0
batch_normalization_11 (Bat chNormalization)	(None, 1, 1, 256)	1024
dropout_3 (Dropout)	(None, 1, 1, 256)	0
flatten_2 (Flatten)	(None, 256)	0
dense_8 (Dense)	(None, 512)	131584
dense_9 (Dense)	(None, 10)	5130

Convolutional Neural Network - results

Best model accuracy

```
329/329 [==============================] - 8s 24ms/step - loss: 0.1211 - accuracy: 0.9722
[0.12110534310340881, 0.9721904993057251]
```

Learning curve



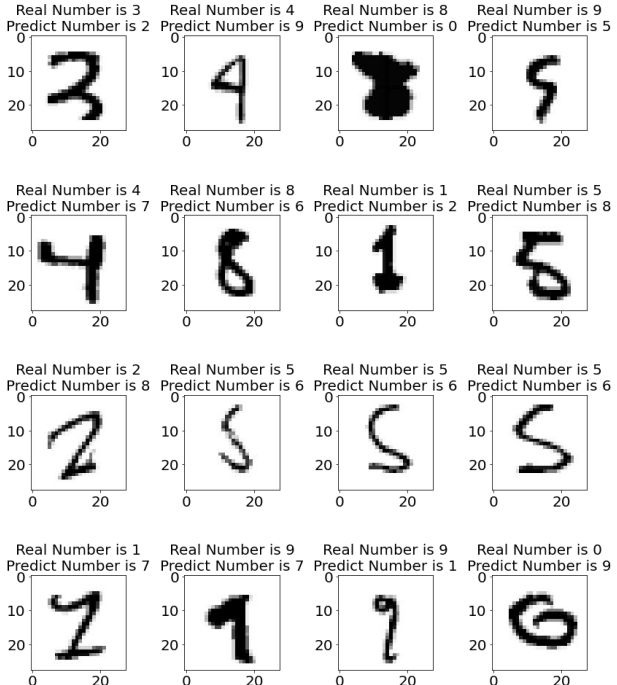
Convolutional Neural Network - results

Table of miss-predicted label

	True value	Predicted value
28	3	2
37	4	9
51	8	0
60	9	5
71	4	7
...	...	...
10094	2	4
10165	2	4
10279	5	6
10323	7	2
10428	5	6

292 rows × 2 columns

Plots of miss-predicted images



Conclusion

- According to the Score given by Kaggle:
  - The SVM model performs the best
  - DNN model performs the worst

submission_cnn.csv a few seconds to go by yangxi47 add submission details	0.96953
submission_dnn.csv a few seconds to go by yangxi47 add submission details	0.95532
submission_mlp.csv a few seconds to go by yangxi47 add submission details	0.96250
submission_svm.csv just now by yangxi47 add submission details	0.97964

Thanks for Watching!