Digit Recognizer

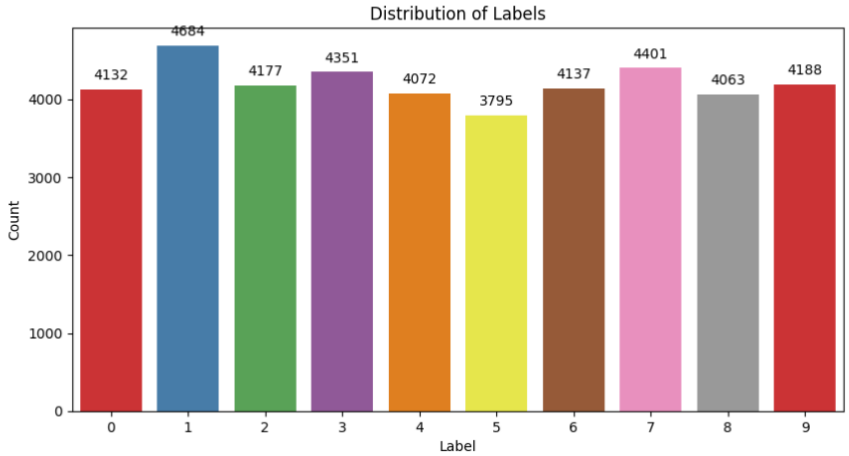
Data Description

The data was loaded directly from Kaggle and consisted of a training set and a test set. The training set contained labels, but the test set did not. The data files train.csv and test.csv contain gray-scale images of hand-drawn digits, from zero through nine. Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255, inclusive.

The training data set, (train.csv), has 785 columns. The first column, called "label", is the digit that was drawn by the user. The rest of the columns contain the pixel-values of the associated image.

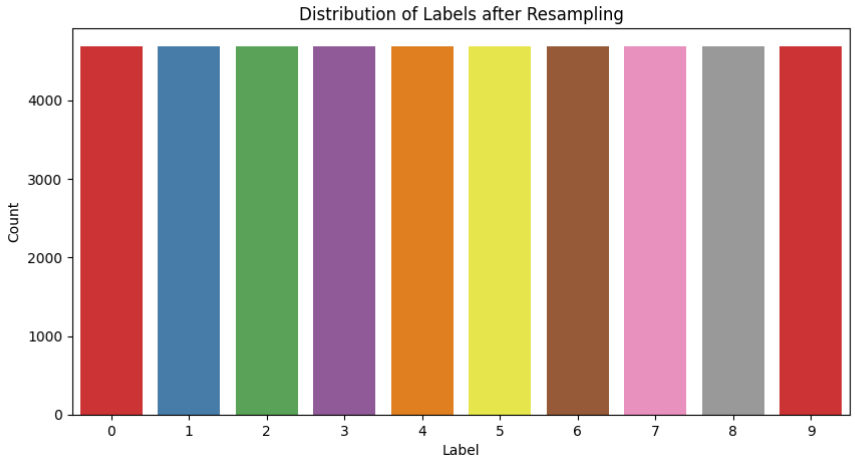
Each pixel column in the training set has a name like pixelx, where x is an integer between 0 and 783, inclusive. To locate this pixel on the image, suppose that we have decomposed x as x = i * 28 + j, where i and j are integers between 0 and 27, inclusive. Then pixelx is located on row i and column j of a 28 x 28 matrix, (indexing by zero).

The data was visualized using count plots to understand the distribution of the labels.



An imbalance in the data was observed, which was addressed using oversampling. The count plot was visualized again to confirm that the imbalance had been corrected.

The data was then split into a training set and a validation set.

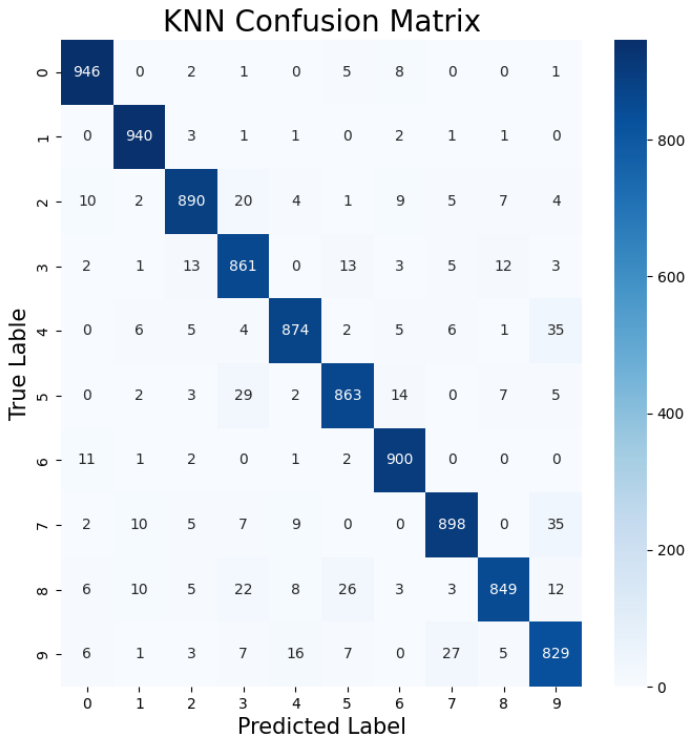


K-Nearest Neighbors (KNN) Model

The KNN model was trained using the scaled training data. The number of neighbors was set to 3. The model was then used to make predictions on the validation set.

Class	Precision	Recall	F1-score
0	0.96	0.98	0.97
1	0.97	0.99	0.98
2	0.96	0.93	0.95
3	0.90	0.94	0.92
4	0.96	0.93	0.94
5	0.94	0.93	0.94
6	0.95	0.98	0.97
7	0.95	0.93	0.94
8	0.96	0.90	0.93
9	0.90	0.92	0.91

The KNN model achieved an accuracy of 94.47% on the validation set. The precision, recall, and F1-score for each class were also calculated and were generally high, indicating good performance of the model. A confusion matrix was also generated for the KNN model.



Convolutional Neural Network (CNN) Model

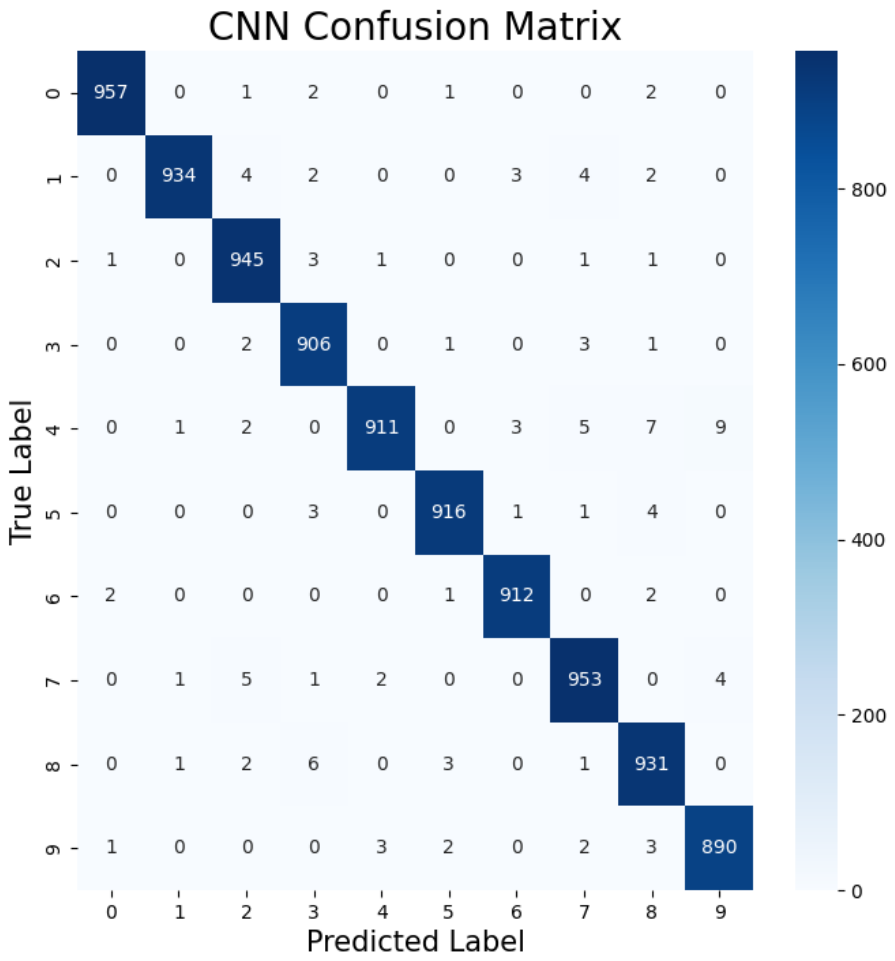
The CNN model was built using several layers, including Conv2D, MaxPooling2D, Dropout, Flatten, and Dense layers. The model was compiled with the Adam optimizer and the categorical cross-entropy loss function.

The model was trained for 10 epochs, with a validation split of 20%. The learning rate was reduced when the validation loss stopped improving, using the ReduceLROnPlateau callback.

Epoch 1/10
937/937 [==============================] - 68s 71ms/step - loss: 0.3208 - accuracy: 0.9012 - val_loss: 0.0805 - val_accuracy: 0.9761
Epoch 2/10
937/937 [==============================] - 65s 70ms/step - loss: 0.1063 - accuracy: 0.9708 - val_loss: 0.0605 - val_accuracy: 0.9825
Epoch 3/10
937/937 [==============================] - 70s 74ms/step - loss: 0.0732 - accuracy: 0.9792 - val_loss: 0.0550 - val_accuracy: 0.9851
Epoch 4/10
937/937 [==============================] - 71s 75ms/step - loss: 0.0559 - accuracy: 0.9847 - val_loss: 0.0528 - val_accuracy: 0.9861
Epoch 5/10

```
937/937 [==============================] - 69s 74ms/step - loss: 0.0398 - accuracy:
0.9883 - val_loss: 0.0581 - val_accuracy: 0.9877
Epoch 6/10
937/937 [==============================] - 61s 65ms/step - loss: 0.0369 - accuracy:
0.9891 - val_loss: 0.0634 - val_accuracy: 0.9859
Epoch 7/10
937/937 [==============================] - 78s 83ms/step - loss: 0.0292 - accuracy:
0.9914 - val_loss: 0.0747 - val_accuracy: 0.9877
Epoch 8/10
937/937 [==============================] - 84s 90ms/step - loss: 0.0244 - accuracy:
0.9929 - val_loss: 0.0684 - val_accuracy: 0.9883
Epoch 9/10
937/937 [==============================] - 90s 96ms/step - loss: 0.0245 - accuracy:
0.9932 - val_loss: 0.0793 - val_accuracy: 0.9869
Epoch 10/10
937/937 [==============================] - 72s 77ms/step - loss: 0.0230 - accuracy:
0.9931 - val_loss: 0.0652 - val_accuracy: 0.9880
293/293 [==============================] - 5s 14ms/step
[7 9 1 ... 9 7 0]
```

The CNN model achieved a validation accuracy of 98.99%. The model was then used to make predictions on the validation set, and the predicted labels were extracted. A confusion matrix was also generated for the CNN model.



Conclusion

Both the KNN and CNN models performed well on the validation set, with the CNN model achieving a higher accuracy, 98.99%. The next step would be to use these models to make predictions on the test set and print out the results.

Thao Tran

Kimthoatran77@gmail.com