CPSC 501- Assignment 2 Part 1: MNIST Logistic Regression

```
print("--Make model--")
model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(512, activation='relu'),
  tf.keras.layers.Dense(10, activation='softmax')
])
model.compile(optimizer='Adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

print("--Fit model--")
model.fit(x_train, y_train, batch_size=128, epochs=10, verbose=2)
```

I've changed the Keras model from a two layer model into a three layer model.

The second layer of the model is densely connected layer with 512 output. The ReLu activation function was chosen. I have chosen ReLu activation function because it does not activate all the neurons at the same time which appears to be a better fit than other activation functions like sigmoid

The third layer of the model is densely connected layer with 10 outputs. I chose softmax activation function because softmax goes step further and make sure sum of the 10 probabilities are 1 in total. Softmax function outputs a vector that represents the probability distributions of a list of potential outcomes which works better with classification model.

I also changed the optimizer used from SGD to Adam because it pays attention to that moving window idea of how much has it been getting better and at what rate to decide if its learning factor should be increase or decreased.

Finally, I've changed the batch_size to 128 and epochs to 10 when fitting the model. The larger epoch allows the model to train for many cycles of the training data which improved model accuracy in this case.

Result on training data: 99.9%
Result on testing data: 98.2%

```
--Evaluate model--
1875/1875 - 4s - loss: 0.0057 - accuracy: 0.9990 - 4s/epoch - 2ms/step
313/313 - 1s - loss: 0.0578 - accuracy: 0.9822 - 814ms/epoch - 3ms/step
Train / Test Accuracy: 99.9% / 98.2%
```