Debarghya Dey 510519087 Assignment 5

In [44]:
```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
import numpy as np
import cv2
from PIL import Image, ImageOps


from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remou
nt=True).
```

Importing Dataset & Exploration

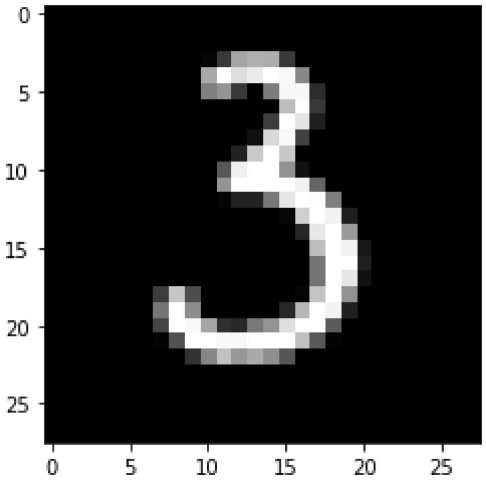
In [9]:
```
(X1, y1), (X2, y2) = tf.keras.datasets.mnist.load_data()
```

In [10]:
```
X_total = np.concatenate([X1, X2])
y_total = np.concatenate([y1, y2])

X, X_test, y, y_test = train_test_split(X_total, y_total, test_size=0.1)
print (X.shape,X_test.shape)
```

```
(63000, 28, 28) (7000, 28, 28)
```

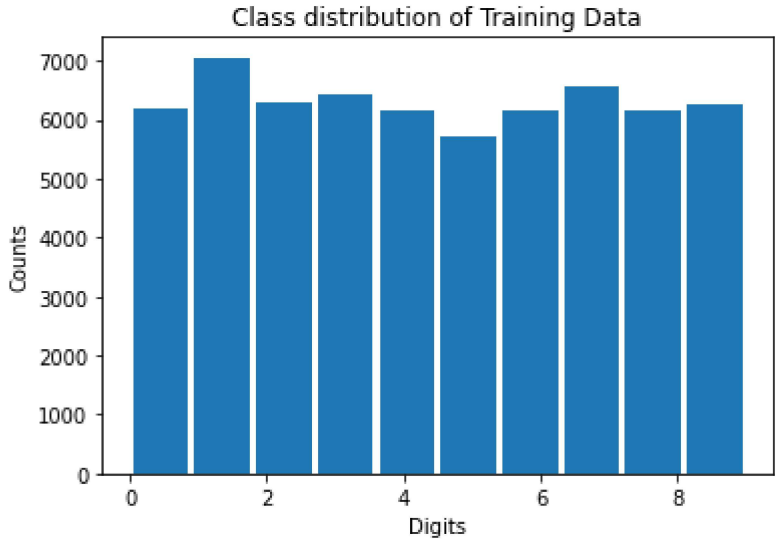
In [11]:
```
def displayMNIST(imageAsArray, label):
    imageAsArray = imageAsArray.reshape(28, 28);
    plt.imshow(imageAsArray, cmap='gray')
    plt.show()
    print("Label : ", label)
```

In [12]:
```
idx = 7
displayMNIST(X[idx], y[idx])
```



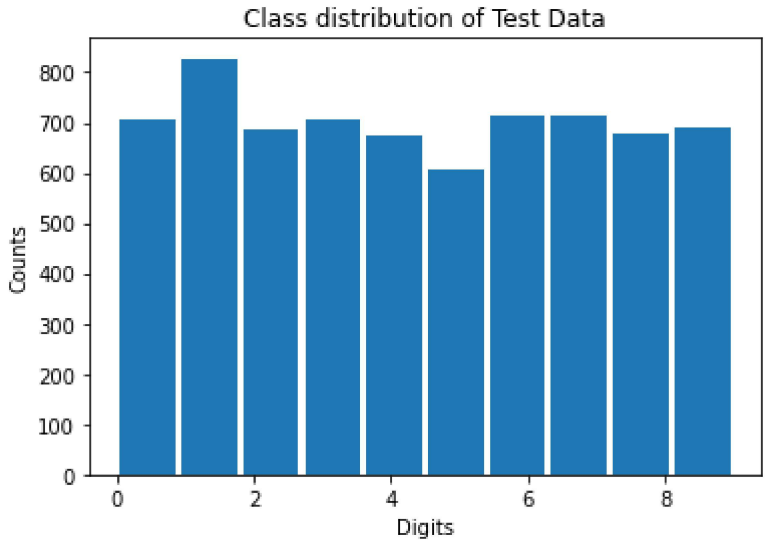
```
Label :  3
```

In [13]:
```
plt.hist(y, bins=10, rwidth=0.9)
plt.ylabel('Counts')
plt.xlabel('Digits')
plt.title('Class distribution of Training Data')
plt.show()
```



In [14]:
```
plt.hist(y_test, bins=10, rwidth=0.9)
plt.ylabel('Counts')
plt.xlabel('Digits')
```

```
plt.title('Class distribution of Test Data')
plt.show()
```



In [15]:
```
X, X_test = X/255.0, X_test/255.0
```

In [16]:
```
def plot_results(history, num):
  fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(10, 4))
  fig.suptitle('Model {} - Accuracy and Loss Plots'.format(num))

  ax1.plot(history.history['accuracy'])
  ax1.plot(history.history['val_accuracy'])
  ax1.legend(['train', 'val'], loc='upper left')

  ax2.plot(history.history['loss'])
  ax2.plot(history.history['val_loss'])
  ax2.legend(['train', 'val'], loc='upper left')
```

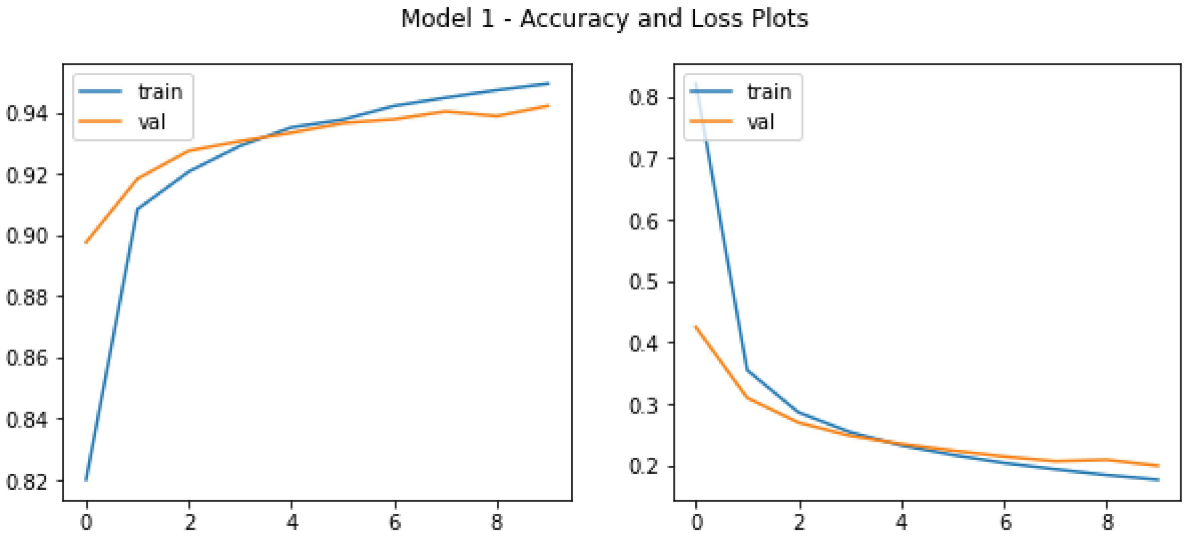
Model 1 - [16] with Sigmoid Activation

In [17]:
```
model1 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='sigmoid'),
    tf.keras.layers.Dense(10, activation='softmax')])
```

```
model1.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.001),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

history = model1.fit(X, y,validation_split = 0.166666, epochs=10)
```

```
Epoch 1/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.8211 - accuracy: 0.8200 - val_loss: 0.4251 - val_acc
uracy: 0.8975
Epoch 2/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.3554 - accuracy: 0.9083 - val_loss: 0.3102 - val_acc
uracy: 0.9183
Epoch 3/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.2864 - accuracy: 0.9206 - val_loss: 0.2698 - val_acc
uracy: 0.9274
Epoch 4/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.2543 - accuracy: 0.9290 - val_loss: 0.2483 - val_acc
uracy: 0.9306
Epoch 5/10
1641/1641 [==============================] - 3s 2ms/step - loss: 0.2326 - accuracy: 0.9351 - val_loss: 0.2351 - val_acc
uracy: 0.9333
Epoch 6/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.2172 - accuracy: 0.9376 - val_loss: 0.2242 - val_acc
uracy: 0.9365
Epoch 7/10
1641/1641 [==============================] - 3s 2ms/step - loss: 0.2042 - accuracy: 0.9421 - val_loss: 0.2146 - val_acc
uracy: 0.9377
Epoch 8/10
1641/1641 [==============================] - 3s 2ms/step - loss: 0.1939 - accuracy: 0.9448 - val_loss: 0.2070 - val_acc
uracy: 0.9403
Epoch 9/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.1845 - accuracy: 0.9472 - val_loss: 0.2093 - val_acc
uracy: 0.9388
Epoch 10/10
1641/1641 [==============================] - 3s 2ms/step - loss: 0.1769 - accuracy: 0.9494 - val_loss: 0.1996 - val_acc
uracy: 0.9421
```

In [18]:
```
plot_results(history, "1")
```



In [19]:
```
model1.evaluate(X_test, y_test)
```

```
219/219 [==============================] - 0s 2ms/step - loss: 0.2135 - accuracy: 0.9387
```

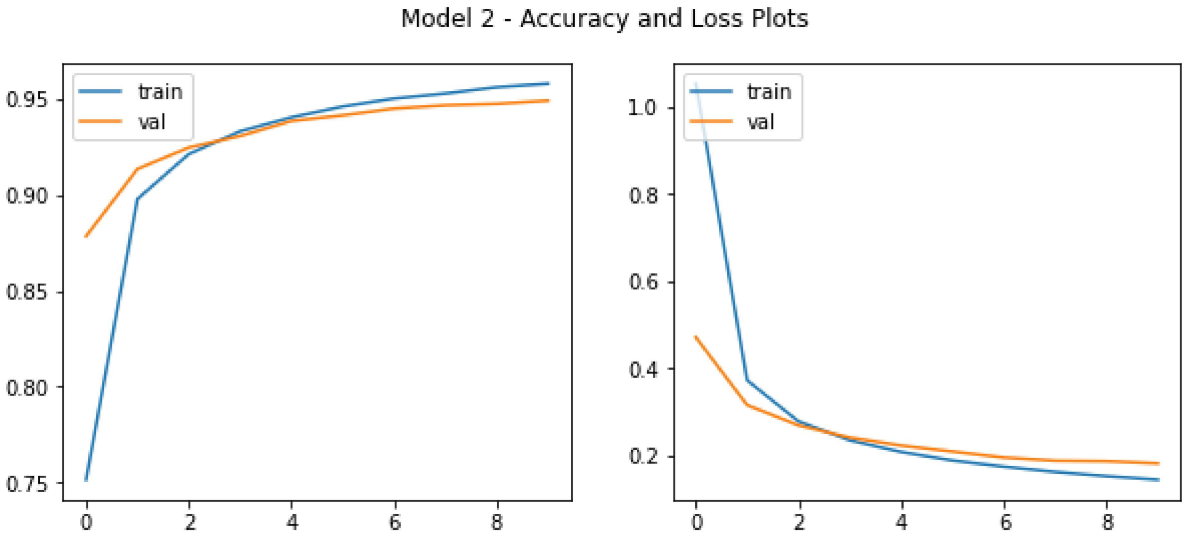
Out[19]:
```
[0.21349187195301056, 0.9387142658233643]
```

Model 2 - [16, 32] with Sigmoid Activation

In [20]:
```
model2 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='sigmoid'),
    tf.keras.layers.Dense(32, activation='sigmoid'),
    tf.keras.layers.Dense(10, activation='softmax')])

model2.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.001),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

history = model2.fit(X, y,validation_split = 0.166666, epochs=10)
```

```
Epoch 1/10
1641/1641 [==============================] - 4s 2ms/step - loss: 1.0518 - accuracy: 0.7516 - val_loss: 0.4706 - val_acc
uracy: 0.8783
Epoch 2/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.3722 - accuracy: 0.8976 - val_loss: 0.3155 - val_acc
uracy: 0.9132
Epoch 3/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.2780 - accuracy: 0.9210 - val_loss: 0.2690 - val_acc
uracy: 0.9244
Epoch 4/10
1641/1641 [==============================] - 3s 2ms/step - loss: 0.2346 - accuracy: 0.9329 - val_loss: 0.2408 - val_acc
uracy: 0.9304
Epoch 5/10
1641/1641 [==============================] - 5s 3ms/step - loss: 0.2079 - accuracy: 0.9401 - val_loss: 0.2226 - val_acc
uracy: 0.9382
Epoch 6/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.1884 - accuracy: 0.9457 - val_loss: 0.2089 - val_acc
uracy: 0.9411
Epoch 7/10
1641/1641 [==============================] - 3s 2ms/step - loss: 0.1742 - accuracy: 0.9499 - val_loss: 0.1952 - val_acc
uracy: 0.9447
Epoch 8/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.1621 - accuracy: 0.9525 - val_loss: 0.1879 - val_acc
uracy: 0.9464
Epoch 9/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.1524 - accuracy: 0.9558 - val_loss: 0.1867 - val_acc
uracy: 0.9471
Epoch 10/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.1443 - accuracy: 0.9576 - val_loss: 0.1818 - val_acc
uracy: 0.9488
```

In [21]:
```
plot_results(history, "2")
```



In [22]:
```
model2.evaluate(X_test, y_test)
```

```
219/219 [==============================] - 0s 1ms/step - loss: 0.1879 - accuracy: 0.9459
```

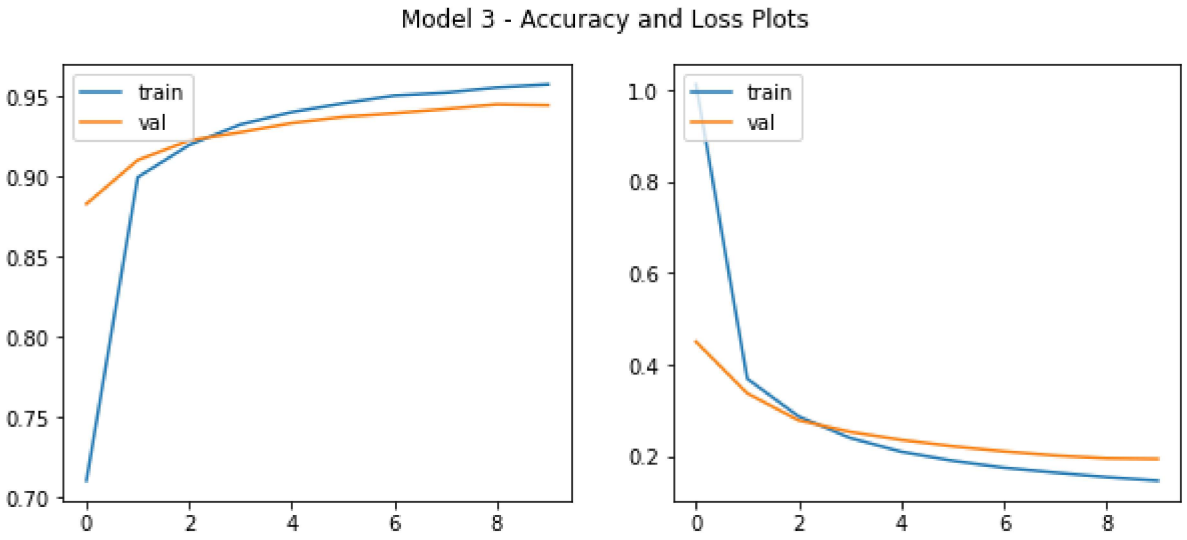
Out[22]:
```
[0.18788103759288788, 0.9458571672439575]
```

Model 3 - [16, 32, 64] with Sigmoid Activation

In [23]:
```
model3 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='sigmoid'),
    tf.keras.layers.Dense(32, activation='sigmoid'),
    tf.keras.layers.Dense(64, activation='sigmoid'),
    tf.keras.layers.Dense(10, activation='softmax')])

model3.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.001),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

history = model3.fit(X, y,validation_split = 0.166666, epochs=10)
```

```
Epoch 1/10
1641/1641 [==============================] - 5s 3ms/step - loss: 1.0124 - accuracy: 0.7106 - val_loss: 0.4503 - val_acc
uracy: 0.8829
Epoch 2/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.3695 - accuracy: 0.8994 - val_loss: 0.3376 - val_acc
uracy: 0.9101
Epoch 3/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.2871 - accuracy: 0.9195 - val_loss: 0.2786 - val_acc
uracy: 0.9223
Epoch 4/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.2402 - accuracy: 0.9323 - val_loss: 0.2534 - val_acc
uracy: 0.9274
Epoch 5/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.2100 - accuracy: 0.9399 - val_loss: 0.2358 - val_acc
uracy: 0.9331
Epoch 6/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.1903 - accuracy: 0.9454 - val_loss: 0.2220 - val_acc
uracy: 0.9370
Epoch 7/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.1751 - accuracy: 0.9502 - val_loss: 0.2107 - val_acc
uracy: 0.9392
Epoch 8/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.1644 - accuracy: 0.9521 - val_loss: 0.2017 - val_acc
uracy: 0.9419
Epoch 9/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.1544 - accuracy: 0.9553 - val_loss: 0.1956 - val_acc
uracy: 0.9449
Epoch 10/10
1641/1641 [==============================] - 4s 2ms/step - loss: 0.1467 - accuracy: 0.9572 - val_loss: 0.1942 - val_acc
uracy: 0.9443
```

In [24]:
```
plot_results(history, "3")
```



In [25]:
```
model3.evaluate(X_test, y_test)
```

```
219/219 [==============================] - 0s 1ms/step - loss: 0.1916 - accuracy: 0.9451
```

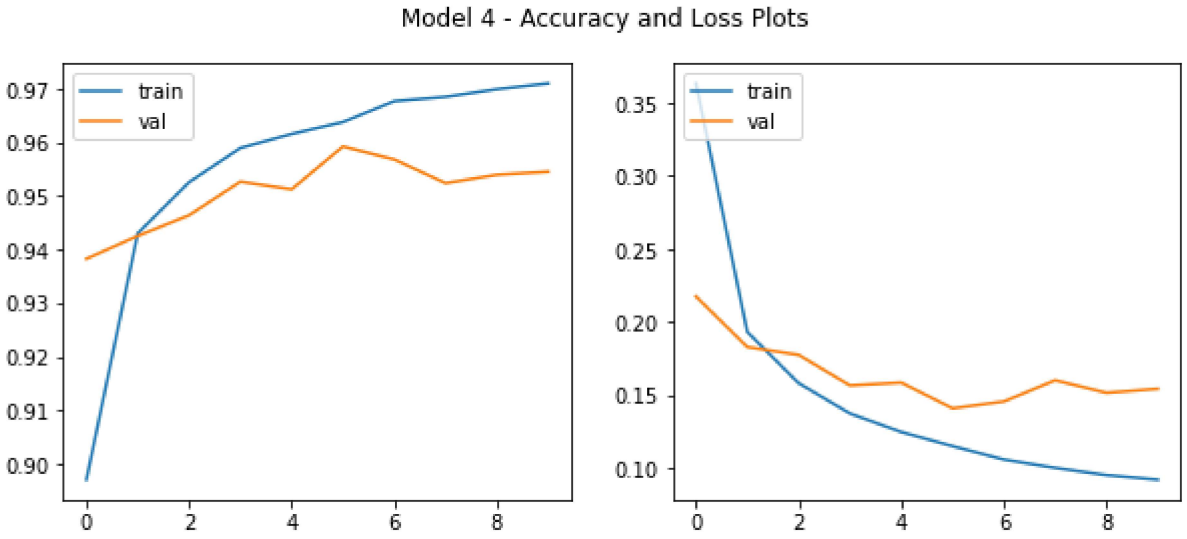
Out[25]:
```
[0.19161254167556763, 0.9451428651809692]
```

Model 4 - [16, 32, 64] with Tanh Activation

In [26]:
```
model4 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='tanh'),
    tf.keras.layers.Dense(32, activation='tanh'),
    tf.keras.layers.Dense(64, activation='tanh'),
    tf.keras.layers.Dense(10, activation='softmax')])

model4.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.001),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

#in the beginning, original dataset was split such that test set had 10% (split was to be done 80:10:10) and rest was fo
#here validation_spilt is chosen such that in the actual dataset, train and val appear 80:10
history = model4.fit(X, y,validation_split = 0.111111, epochs=10)
```

```
Epoch 1/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.3632 - accuracy: 0.8970 - val_loss: 0.2176 - val_acc
uracy: 0.9383
Epoch 2/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1934 - accuracy: 0.9431 - val_loss: 0.1830 - val_acc
uracy: 0.9426
Epoch 3/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1584 - accuracy: 0.9526 - val_loss: 0.1776 - val_acc
uracy: 0.9464
Epoch 4/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1376 - accuracy: 0.9590 - val_loss: 0.1568 - val_acc
uracy: 0.9527
Epoch 5/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1250 - accuracy: 0.9616 - val_loss: 0.1587 - val_acc
uracy: 0.9513
Epoch 6/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1153 - accuracy: 0.9639 - val_loss: 0.1412 - val_acc
uracy: 0.9593
Epoch 7/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1061 - accuracy: 0.9678 - val_loss: 0.1458 - val_acc
uracy: 0.9569
Epoch 8/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1005 - accuracy: 0.9686 - val_loss: 0.1603 - val_acc
uracy: 0.9524
Epoch 9/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.0955 - accuracy: 0.9700 - val_loss: 0.1518 - val_acc
uracy: 0.9540
Epoch 10/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.0924 - accuracy: 0.9711 - val_loss: 0.1545 - val_acc
uracy: 0.9546
```

In [27]:
```
plot_results(history, "4")
```



In [28]:
```
model4.evaluate(X_test, y_test)
```

```
219/219 [==============================] - 0s 2ms/step - loss: 0.1563 - accuracy: 0.9537
```

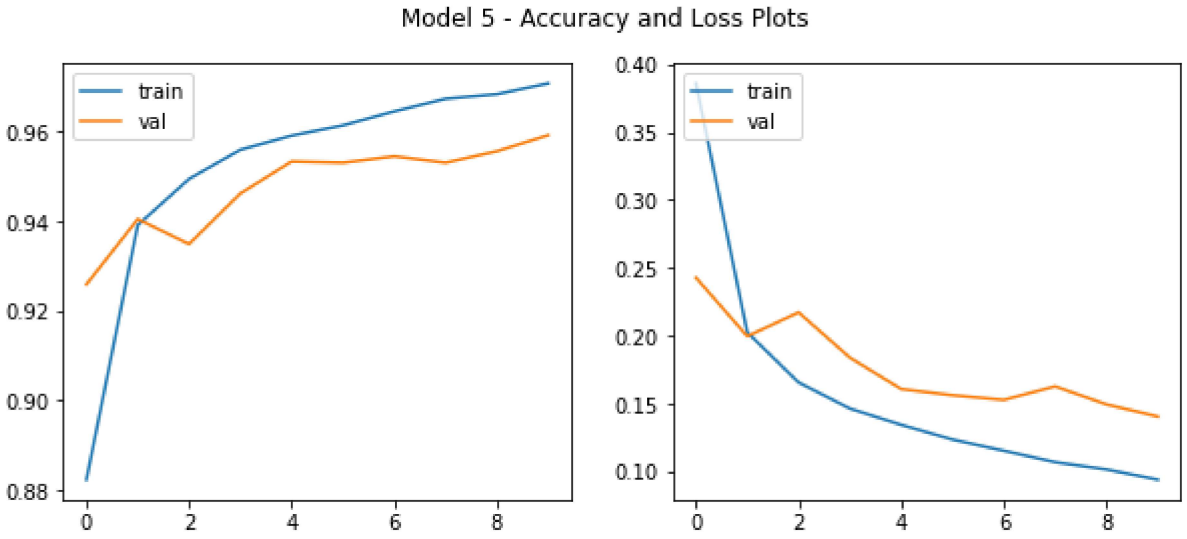
Out[28]:
```
[0.156306654214859, 0.9537143111228943]
```

Model 5 - [16, 32, 64] with ReLU Activation

In [29]:
```
model5 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(10, activation='softmax')])

model5.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.001),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

history = model5.fit(X, y, validation_split = 0.111111, epochs=10)
```

```
Epoch 1/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.3860 - accuracy: 0.8823 - val_loss: 0.2427 - val_acc
uracy: 0.9259
Epoch 2/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.2024 - accuracy: 0.9391 - val_loss: 0.1995 - val_acc
uracy: 0.9404
Epoch 3/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1655 - accuracy: 0.9494 - val_loss: 0.2171 - val_acc
uracy: 0.9349
Epoch 4/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1462 - accuracy: 0.9559 - val_loss: 0.1837 - val_acc
uracy: 0.9461
Epoch 5/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1342 - accuracy: 0.9591 - val_loss: 0.1606 - val_acc
uracy: 0.9533
Epoch 6/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1234 - accuracy: 0.9613 - val_loss: 0.1561 - val_acc
uracy: 0.9530
Epoch 7/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1151 - accuracy: 0.9645 - val_loss: 0.1527 - val_acc
uracy: 0.9544
Epoch 8/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1067 - accuracy: 0.9673 - val_loss: 0.1626 - val_acc
uracy: 0.9530
Epoch 9/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.1013 - accuracy: 0.9683 - val_loss: 0.1493 - val_acc
uracy: 0.9556
Epoch 10/10
1750/1750 [==============================] - 4s 2ms/step - loss: 0.0938 - accuracy: 0.9707 - val_loss: 0.1404 - val_acc
uracy: 0.9591
```

In [30]:
```
plot_results(history, "5")
```



In [31]:
```
model5.evaluate(X_test, y_test)
```

```
219/219 [==============================] - 0s 1ms/step - loss: 0.1387 - accuracy: 0.9589
```

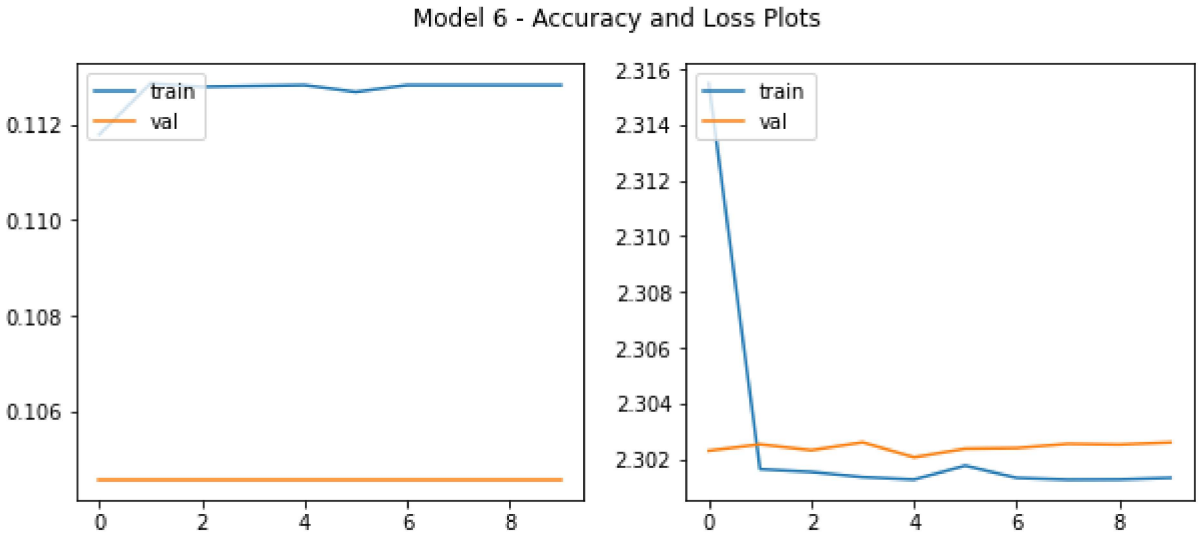
Out[31]:
```
[0.13870912790298462, 0.9588571190834045]
```

Model 6 - [16, 32, 64] with ReLU Activation, 0.9 Dropout

In [35]:
```
model6 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dropout(0.9),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dropout(0.9),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.9),
    tf.keras.layers.Dense(10, activation='softmax')])

model6.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.001),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

history = model6.fit(X, y, validation_split = 0.111111, epochs=10)
```

```
Epoch 1/10
1750/1750 [==============================] - 5s 3ms/step - loss: 2.3155 - accuracy: 0.1118 - val_loss: 2.3023 - val_acc
uracy: 0.1046
Epoch 2/10
1750/1750 [==============================] - 4s 2ms/step - loss: 2.3016 - accuracy: 0.1129 - val_loss: 2.3025 - val_acc
uracy: 0.1046
Epoch 3/10
1750/1750 [==============================] - 4s 2ms/step - loss: 2.3015 - accuracy: 0.1128 - val_loss: 2.3023 - val_acc
uracy: 0.1046
Epoch 4/10
1750/1750 [==============================] - 7s 4ms/step - loss: 2.3013 - accuracy: 0.1128 - val_loss: 2.3026 - val_acc
uracy: 0.1046
Epoch 5/10
1750/1750 [==============================] - 9s 5ms/step - loss: 2.3013 - accuracy: 0.1128 - val_loss: 2.3021 - val_acc
uracy: 0.1046
Epoch 6/10
1750/1750 [==============================] - 6s 4ms/step - loss: 2.3018 - accuracy: 0.1127 - val_loss: 2.3024 - val_acc
uracy: 0.1046
Epoch 7/10
1750/1750 [==============================] - 4s 2ms/step - loss: 2.3013 - accuracy: 0.1128 - val_loss: 2.3024 - val_acc
uracy: 0.1046
Epoch 8/10
1750/1750 [==============================] - 4s 2ms/step - loss: 2.3013 - accuracy: 0.1128 - val_loss: 2.3025 - val_acc
uracy: 0.1046
Epoch 9/10
1750/1750 [==============================] - 5s 3ms/step - loss: 2.3013 - accuracy: 0.1128 - val_loss: 2.3025 - val_acc
uracy: 0.1046
Epoch 10/10
1750/1750 [==============================] - 4s 2ms/step - loss: 2.3013 - accuracy: 0.1128 - val_loss: 2.3026 - val_acc
uracy: 0.1046
```

In [36]:
```
plot_results(history, "6")
```



In [ ]:
```
model6.evaluate(X_test, y_test)
```

```
313/313 [==============================] - 0s 1ms/step - loss: 2.3010 - accuracy: 0.1135
```

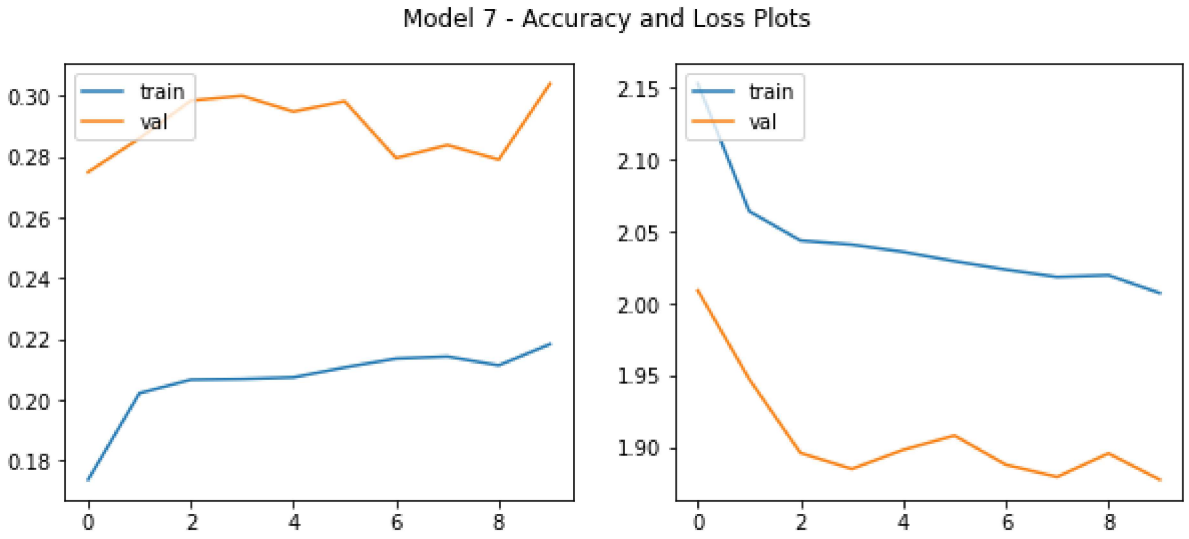
Out[ ]:
```
[2.3009984493255615, 0.11349999904632568]
```

Model 7 - [16, 32, 64] with ReLU Activation, 0.75 Dropout

In [ ]:
```
model7 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dropout(0.75),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dropout(0.75),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.75),
    tf.keras.layers.Dense(10, activation='softmax')])

model7.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.001),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

history = model7.fit(X, y, validation_split = 0.111111, epochs=10)
```

```
Epoch 1/10
1563/1563 [==============================] - 4s 2ms/step - loss: 2.1525 - accuracy: 0.1737 - val_loss: 2.0089 - val_acc
uracy: 0.2749
Epoch 2/10
1563/1563 [==============================] - 3s 2ms/step - loss: 2.0640 - accuracy: 0.2021 - val_loss: 1.9475 - val_acc
uracy: 0.2860
Epoch 3/10
1563/1563 [==============================] - 3s 2ms/step - loss: 2.0436 - accuracy: 0.2066 - val_loss: 1.8961 - val_acc
uracy: 0.2984
Epoch 4/10
1563/1563 [==============================] - 3s 2ms/step - loss: 2.0408 - accuracy: 0.2068 - val_loss: 1.8850 - val_acc
uracy: 0.3000
Epoch 5/10
1563/1563 [==============================] - 4s 2ms/step - loss: 2.0357 - accuracy: 0.2074 - val_loss: 1.8983 - val_acc
uracy: 0.2948
Epoch 6/10
1563/1563 [==============================] - 4s 3ms/step - loss: 2.0292 - accuracy: 0.2106 - val_loss: 1.9082 - val_acc
uracy: 0.2982
Epoch 7/10
1563/1563 [==============================] - 4s 2ms/step - loss: 2.0234 - accuracy: 0.2135 - val_loss: 1.8879 - val_acc
uracy: 0.2795
Epoch 8/10
1563/1563 [==============================] - 4s 2ms/step - loss: 2.0183 - accuracy: 0.2142 - val_loss: 1.8794 - val_acc
uracy: 0.2838
Epoch 9/10
1563/1563 [==============================] - 4s 2ms/step - loss: 2.0195 - accuracy: 0.2113 - val_loss: 1.8958 - val_acc
uracy: 0.2790
Epoch 10/10
1563/1563 [==============================] - 4s 2ms/step - loss: 2.0071 - accuracy: 0.2183 - val_loss: 1.8775 - val_acc
uracy: 0.3040
```

In [ ]:
```
plot_results(history, "7")
```



In [ ]:
```
model7.evaluate(X_test, y_test)
```

```
313/313 [==============================] - 0s 1ms/step - loss: 1.8630 - accuracy: 0.3085
```

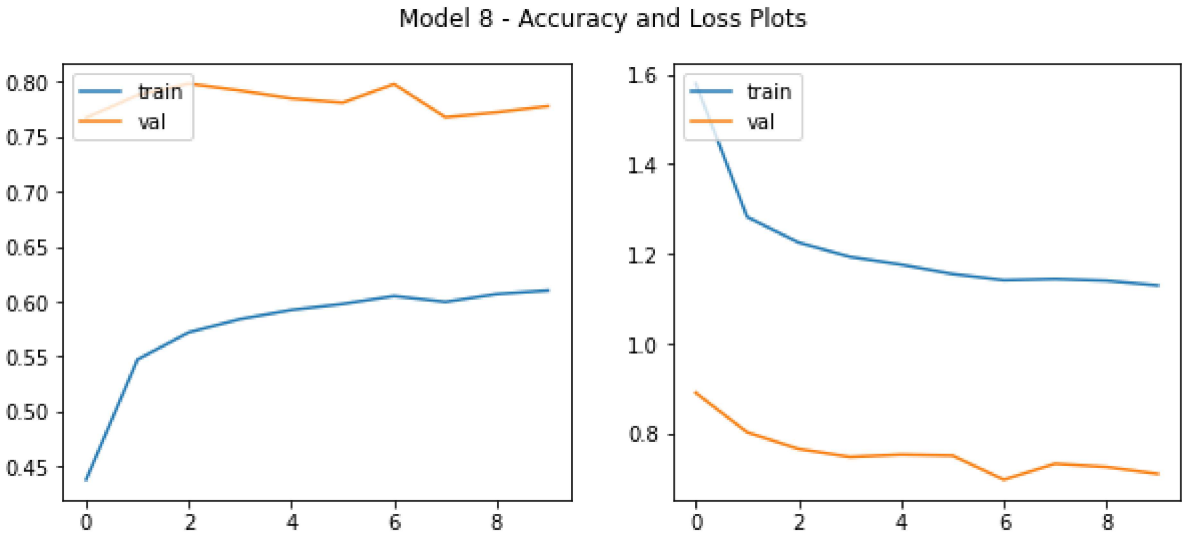
Out[ ]:
```
[1.8630034923553467, 0.3084999918937683]
```

Model 8 - [16, 32, 64] with ReLU Activation, 0.5 Dropout

In [ ]:
```
model8 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(10, activation='softmax')])

model8.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.001),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

history = model8.fit(X, y, validation_split = 0.111111, epochs=10)
```

```
Epoch 1/10
1563/1563 [==============================] - 5s 3ms/step - loss: 1.5790 - accuracy: 0.4376 - val_loss: 0.8901 - val_acc
uracy: 0.7672
Epoch 2/10
1563/1563 [==============================] - 3s 2ms/step - loss: 1.2820 - accuracy: 0.5469 - val_loss: 0.8022 - val_acc
uracy: 0.7878
Epoch 3/10
1563/1563 [==============================] - 3s 2ms/step - loss: 1.2250 - accuracy: 0.5718 - val_loss: 0.7651 - val_acc
uracy: 0.7983
Epoch 4/10
1563/1563 [==============================] - 3s 2ms/step - loss: 1.1930 - accuracy: 0.5838 - val_loss: 0.7472 - val_acc
uracy: 0.7920
Epoch 5/10
1563/1563 [==============================] - 3s 2ms/step - loss: 1.1757 - accuracy: 0.5922 - val_loss: 0.7525 - val_acc
uracy: 0.7848
Epoch 6/10
1563/1563 [==============================] - 3s 2ms/step - loss: 1.1548 - accuracy: 0.5976 - val_loss: 0.7503 - val_acc
uracy: 0.7810
Epoch 7/10
1563/1563 [==============================] - 3s 2ms/step - loss: 1.1410 - accuracy: 0.6048 - val_loss: 0.6965 - val_acc
uracy: 0.7979
Epoch 8/10
1563/1563 [==============================] - 3s 2ms/step - loss: 1.1436 - accuracy: 0.5995 - val_loss: 0.7321 - val_acc
uracy: 0.7677
Epoch 9/10
1563/1563 [==============================] - 3s 2ms/step - loss: 1.1399 - accuracy: 0.6067 - val_loss: 0.7249 - val_acc
uracy: 0.7722
Epoch 10/10
1563/1563 [==============================] - 3s 2ms/step - loss: 1.1293 - accuracy: 0.6099 - val_loss: 0.7095 - val_acc
uracy: 0.7778
```

In [ ]:
```
plot_results(history, "8")
```



In [ ]:
```
model8.evaluate(X_test, y_test)
```

```
313/313 [==============================] - 0s 1ms/step - loss: 0.7420 - accuracy: 0.7698
```

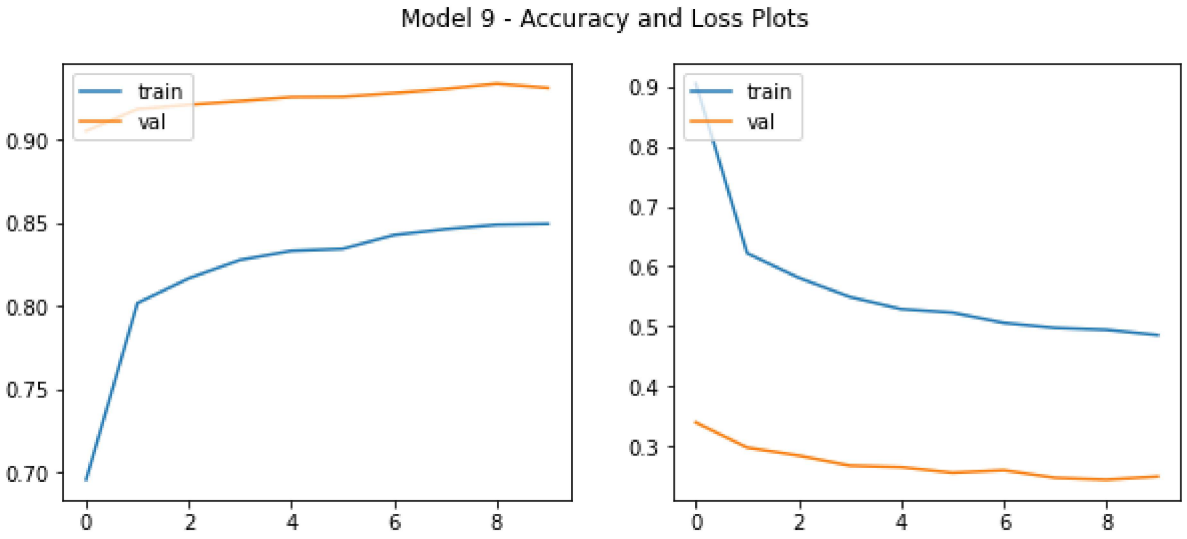
Out[ ]:
```
[0.7419830560684204, 0.7698000073432922]
```

Model 9 - [16, 32, 64] with ReLU Activation, 0.25 Dropout

In [ ]:
```
model9 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dropout(0.25),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dropout(0.25),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.25),
    tf.keras.layers.Dense(10, activation='softmax')])

model9.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.001),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

history = model9.fit(X, y, validation_split = 0.111111, epochs=10)
```

```
Epoch 1/10
1563/1563 [==============================] - 4s 2ms/step - loss: 0.9054 - accuracy: 0.6957 - val_loss: 0.3388 - val_acc
uracy: 0.9053
Epoch 2/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.6219 - accuracy: 0.8018 - val_loss: 0.2968 - val_acc
uracy: 0.9184
Epoch 3/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.5809 - accuracy: 0.8167 - val_loss: 0.2835 - val_acc
uracy: 0.9210
Epoch 4/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.5487 - accuracy: 0.8278 - val_loss: 0.2666 - val_acc
uracy: 0.9232
Epoch 5/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.5283 - accuracy: 0.8332 - val_loss: 0.2641 - val_acc
uracy: 0.9256
Epoch 6/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.5224 - accuracy: 0.8344 - val_loss: 0.2547 - val_acc
uracy: 0.9258
Epoch 7/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.5053 - accuracy: 0.8427 - val_loss: 0.2589 - val_acc
uracy: 0.9280
Epoch 8/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.4969 - accuracy: 0.8463 - val_loss: 0.2462 - val_acc
uracy: 0.9305
Epoch 9/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.4938 - accuracy: 0.8489 - val_loss: 0.2431 - val_acc
uracy: 0.9337
Epoch 10/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.4848 - accuracy: 0.8494 - val_loss: 0.2484 - val_acc
uracy: 0.9311
```

In [ ]:
```
plot_results(history, "9")
```



In [ ]:
```
model9.evaluate(X_test, y_test)
```

```
313/313 [==============================] - 0s 1ms/step - loss: 0.2680 - accuracy: 0.9249
```

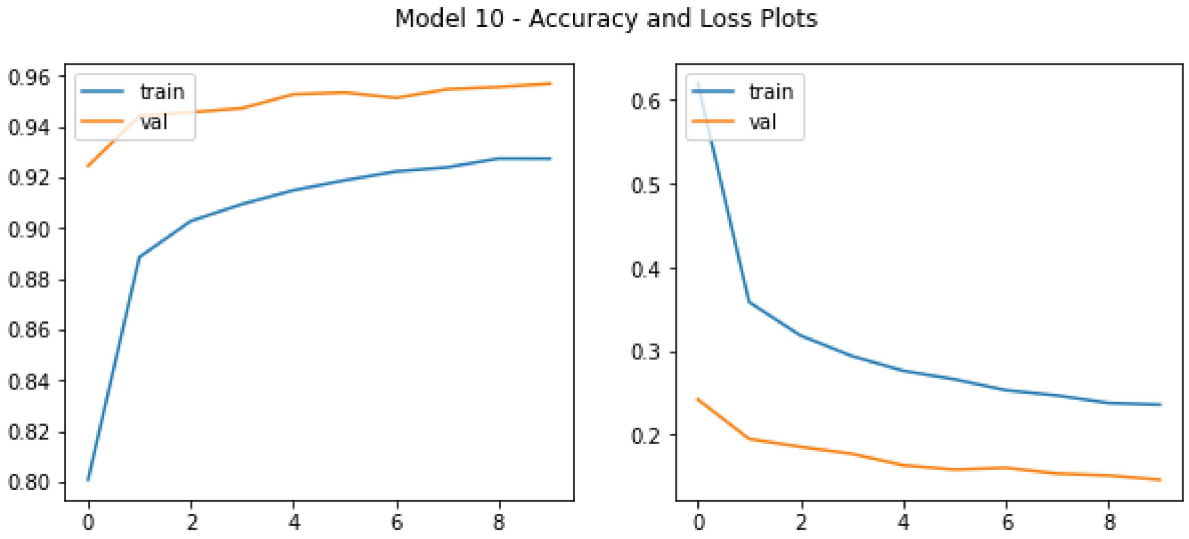
Out[ ]:
```
[0.2679823338985443, 0.9248999953269958]
```

Model 10 - [16, 32, 64] with ReLU Activation, 0.1 Dropout

In [ ]:
```
model10 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dropout(0.1),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dropout(0.1),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.1),
    tf.keras.layers.Dense(10, activation='softmax')])

model10.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.001),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

history = model10.fit(X, y, validation_split = 0.111111, epochs=10)
```

```
Epoch 1/10
1563/1563 [==============================] - 4s 2ms/step - loss: 0.6193 - accuracy: 0.8007 - val_loss: 0.2422 - val_acc
uracy: 0.9244
Epoch 2/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.3584 - accuracy: 0.8885 - val_loss: 0.1951 - val_acc
uracy: 0.9440
Epoch 3/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.3185 - accuracy: 0.9027 - val_loss: 0.1856 - val_acc
uracy: 0.9455
Epoch 4/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2940 - accuracy: 0.9093 - val_loss: 0.1773 - val_acc
uracy: 0.9471
Epoch 5/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2765 - accuracy: 0.9148 - val_loss: 0.1636 - val_acc
uracy: 0.9526
Epoch 6/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2663 - accuracy: 0.9187 - val_loss: 0.1584 - val_acc
uracy: 0.9533
Epoch 7/10
1563/1563 [==============================] - 4s 2ms/step - loss: 0.2533 - accuracy: 0.9222 - val_loss: 0.1605 - val_acc
uracy: 0.9512
Epoch 8/10
1563/1563 [==============================] - 4s 2ms/step - loss: 0.2469 - accuracy: 0.9238 - val_loss: 0.1537 - val_acc
uracy: 0.9546
Epoch 9/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2380 - accuracy: 0.9273 - val_loss: 0.1513 - val_acc
uracy: 0.9555
Epoch 10/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2360 - accuracy: 0.9272 - val_loss: 0.1463 - val_acc
uracy: 0.9568
```

In [ ]:
```
plot_results(history, "10")
```



In [ ]:
```
model10.evaluate(X_test, y_test)
```

```
313/313 [==============================] - 0s 1ms/step - loss: 0.1534 - accuracy: 0.9524
```

Out[ ]:
```
[0.15338517725467682, 0.9524000287055969]
```

Tuning Learning Rate [0.01, 0.001, 0.005, 0.0001, 0.0005]

From the models listed above, the best accuracy (on test set) was observed on Model 5 - [16, 32, 64] and ReLU Activation with no Dropout.

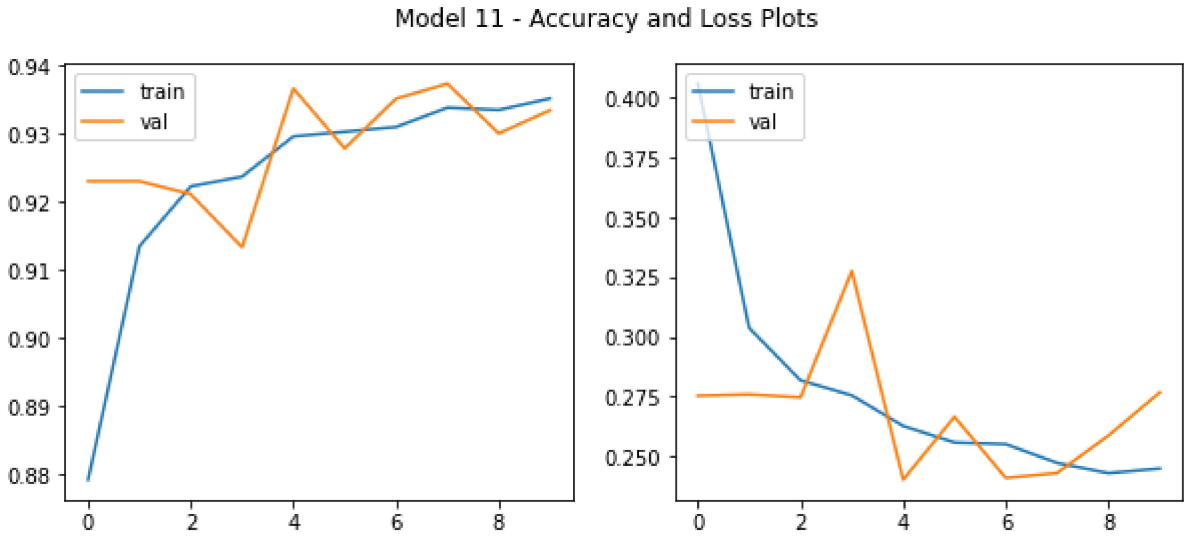
Learning Rate = 0.01

In [ ]:
```
model11 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(10, activation='softmax')])

model11.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.01),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
```

```
              metrics=['accuracy'])

history = model11.fit(X, y, validation_split = 0.111111, epochs=10)
```

```
Epoch 1/10
1563/1563 [==============================] - 4s 2ms/step - loss: 0.4061 - accuracy: 0.8792 - val_loss: 0.2754 - val_acc
uracy: 0.9230
Epoch 2/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.3038 - accuracy: 0.9134 - val_loss: 0.2759 - val_acc
uracy: 0.9230
Epoch 3/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2818 - accuracy: 0.9222 - val_loss: 0.2747 - val_acc
uracy: 0.9211
Epoch 4/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2755 - accuracy: 0.9237 - val_loss: 0.3276 - val_acc
uracy: 0.9133
Epoch 5/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2627 - accuracy: 0.9296 - val_loss: 0.2402 - val_acc
uracy: 0.9366
Epoch 6/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2558 - accuracy: 0.9302 - val_loss: 0.2665 - val_acc
uracy: 0.9278
Epoch 7/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2551 - accuracy: 0.9309 - val_loss: 0.2409 - val_acc
uracy: 0.9351
Epoch 8/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2472 - accuracy: 0.9338 - val_loss: 0.2430 - val_acc
uracy: 0.9373
Epoch 9/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2429 - accuracy: 0.9334 - val_loss: 0.2587 - val_acc
uracy: 0.9300
Epoch 10/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2448 - accuracy: 0.9351 - val_loss: 0.2767 - val_acc
uracy: 0.9334
```

In [ ]:
```
plot_results(history, "11")
```



In [ ]:
```
model11.evaluate(X_test, y_test)
```

```
313/313 [==============================] - 0s 1ms/step - loss: 0.2857 - accuracy: 0.9262
```

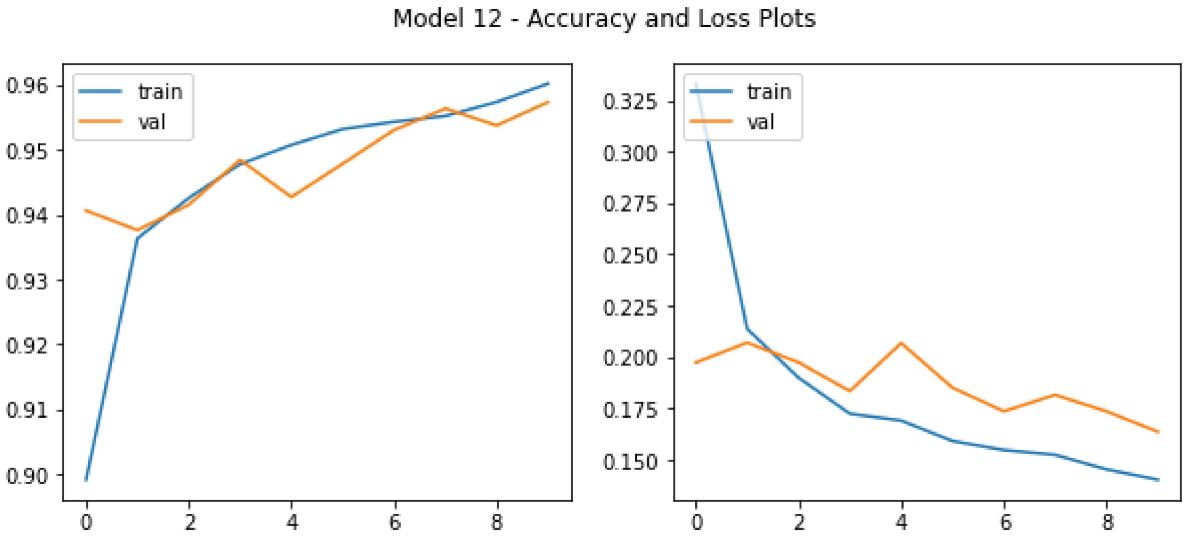
Out[ ]:
```
[0.28569531440734863, 0.9261999726295471]
```

Learning Rate 0.005

In [ ]:
```
model12 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(10, activation='softmax')])

model12.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.005),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

history = model12.fit(X, y, validation_split = 0.111111, epochs=10)
```

```
Epoch 1/10
1563/1563 [==============================] - 4s 2ms/step - loss: 0.3332 - accuracy: 0.8992 - val_loss: 0.1973 - val_acc
uracy: 0.9406
Epoch 2/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2137 - accuracy: 0.9363 - val_loss: 0.2070 - val_acc
uracy: 0.9376
Epoch 3/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1900 - accuracy: 0.9425 - val_loss: 0.1974 - val_acc
uracy: 0.9415
Epoch 4/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1724 - accuracy: 0.9477 - val_loss: 0.1834 - val_acc
uracy: 0.9484
Epoch 5/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1691 - accuracy: 0.9507 - val_loss: 0.2068 - val_acc
uracy: 0.9427
Epoch 6/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1591 - accuracy: 0.9531 - val_loss: 0.1851 - val_acc
uracy: 0.9478
Epoch 7/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1547 - accuracy: 0.9543 - val_loss: 0.1736 - val_acc
uracy: 0.9530
Epoch 8/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1524 - accuracy: 0.9552 - val_loss: 0.1816 - val_acc
uracy: 0.9563
Epoch 9/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1453 - accuracy: 0.9573 - val_loss: 0.1736 - val_acc
uracy: 0.9537
Epoch 10/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1403 - accuracy: 0.9601 - val_loss: 0.1636 - val_acc
uracy: 0.9573
```

In [ ]:
```
plot_results(history, "12")
```



In [ ]:
```
model12.evaluate(X_test, y_test)
```

```
313/313 [==============================] - 0s 1ms/step - loss: 0.1753 - accuracy: 0.9539
```

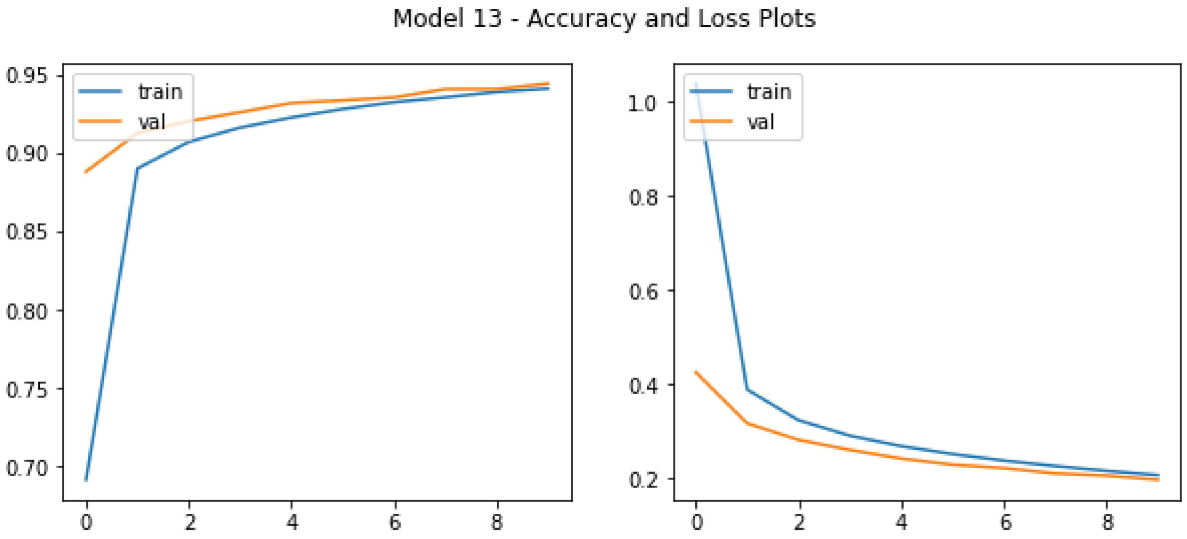
Out[ ]:
```
[0.17525003850460052, 0.9538999795913696]
```

Learning Rate 0.0001

In [ ]:
```
model13 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(10, activation='softmax')])

model13.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.0001),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

history = model13.fit(X, y, validation_split = 0.111111, epochs=10)
```

```
Epoch 1/10
1563/1563 [==============================] - 4s 2ms/step - loss: 1.0386 - accuracy: 0.6912 - val_loss: 0.4248 - val_acc
uracy: 0.8880
Epoch 2/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.3883 - accuracy: 0.8900 - val_loss: 0.3163 - val_acc
uracy: 0.9126
Epoch 3/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.3229 - accuracy: 0.9069 - val_loss: 0.2812 - val_acc
uracy: 0.9202
Epoch 4/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2900 - accuracy: 0.9161 - val_loss: 0.2597 - val_acc
uracy: 0.9259
Epoch 5/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2679 - accuracy: 0.9225 - val_loss: 0.2413 - val_acc
uracy: 0.9318
Epoch 6/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2510 - accuracy: 0.9280 - val_loss: 0.2285 - val_acc
uracy: 0.9335
Epoch 7/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2370 - accuracy: 0.9323 - val_loss: 0.2210 - val_acc
uracy: 0.9354
Epoch 8/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2255 - accuracy: 0.9355 - val_loss: 0.2098 - val_acc
uracy: 0.9407
Epoch 9/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2151 - accuracy: 0.9389 - val_loss: 0.2051 - val_acc
uracy: 0.9408
Epoch 10/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2062 - accuracy: 0.9410 - val_loss: 0.1964 - val_acc
uracy: 0.9442
```

In [ ]:
```
plot_results(history, "13")
```



In [ ]:
```
model13.evaluate(X_test, y_test)
```

```
313/313 [==============================] - 0s 1ms/step - loss: 0.2051 - accuracy: 0.9396
```

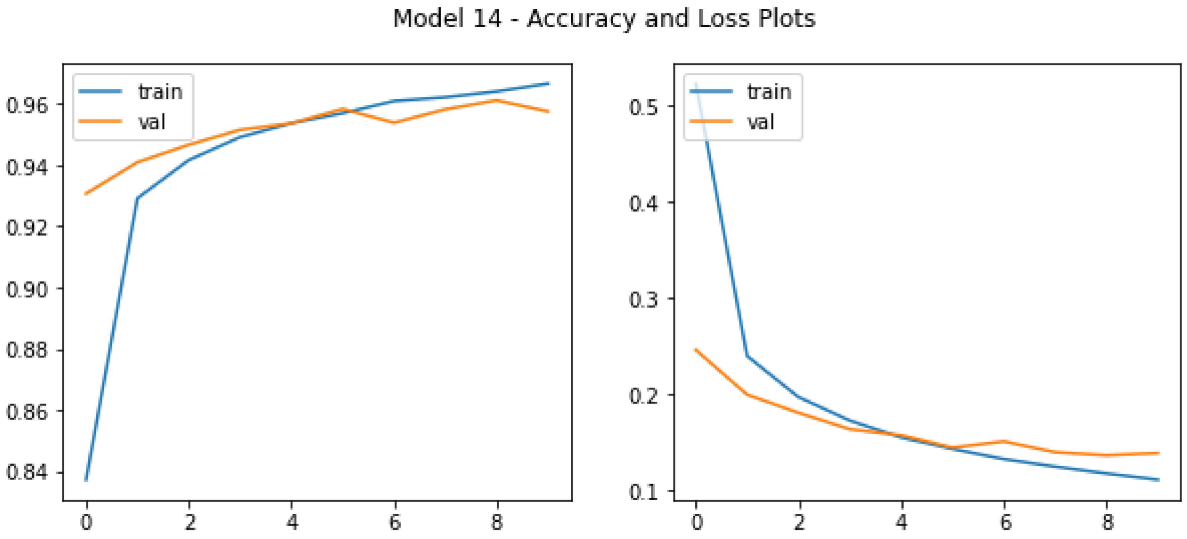
Out[ ]:
```
[0.2050841599702835, 0.9395999908447266]
```

Learning Rate 0.0005

In [ ]:
```
model14 = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(10, activation='softmax')])

model14.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.0005),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

history = model14.fit(X, y, validation_split = 0.111111, epochs=10)
```

```
Epoch 1/10
1563/1563 [==============================] - 4s 2ms/step - loss: 0.5234 - accuracy: 0.8373 - val_loss: 0.2462 - val_acc
uracy: 0.9307
Epoch 2/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.2399 - accuracy: 0.9292 - val_loss: 0.1997 - val_acc
uracy: 0.9409
Epoch 3/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1972 - accuracy: 0.9416 - val_loss: 0.1808 - val_acc
uracy: 0.9466
Epoch 4/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1727 - accuracy: 0.9491 - val_loss: 0.1637 - val_acc
uracy: 0.9515
Epoch 5/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1551 - accuracy: 0.9536 - val_loss: 0.1574 - val_acc
uracy: 0.9536
Epoch 6/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1432 - accuracy: 0.9569 - val_loss: 0.1445 - val_acc
uracy: 0.9583
Epoch 7/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1326 - accuracy: 0.9609 - val_loss: 0.1510 - val_acc
uracy: 0.9538
Epoch 8/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1248 - accuracy: 0.9621 - val_loss: 0.1398 - val_acc
uracy: 0.9582
Epoch 9/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1177 - accuracy: 0.9640 - val_loss: 0.1366 - val_acc
uracy: 0.9611
Epoch 10/10
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1112 - accuracy: 0.9666 - val_loss: 0.1389 - val_acc
uracy: 0.9575
```

In [ ]:
```
plot_results(history, "14")
```



In [ ]:
```
model14.evaluate(X_test, y_test)
```

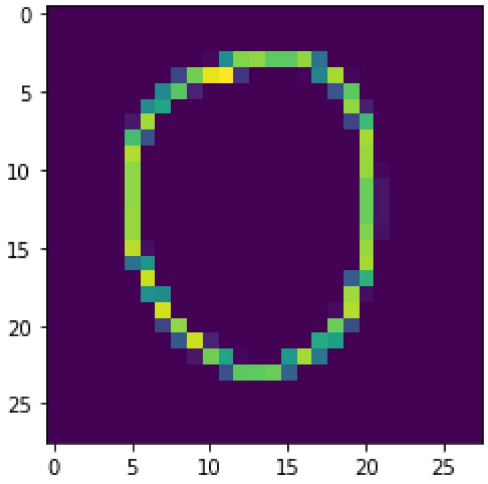
```
313/313 [==============================] - 0s 1ms/step - loss: 0.1374 - accuracy: 0.9588
```

Out[ ]:
```
[0.13742327690124512, 0.9588000178337097]
```

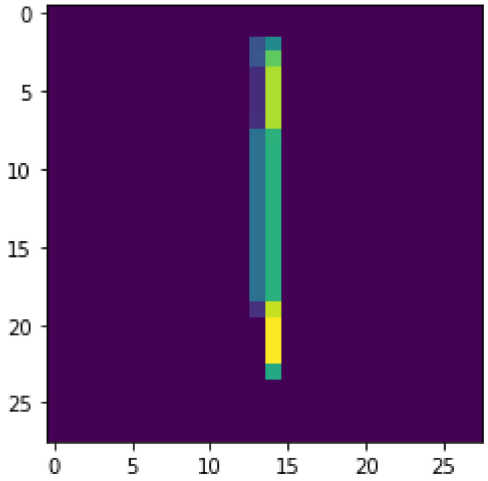
Best performing remains Model 5 with the Test Set accuracy of 96.23%

My own digits

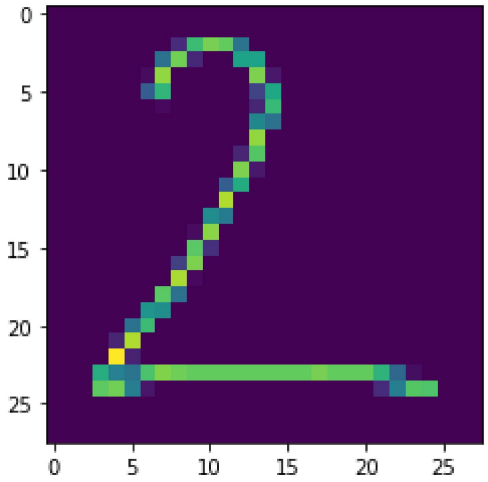
In [43]:
```
images = []
images.append(np.asarray(ImageOps.grayscale(Image.open("/content/drive/MyDrive/ML_DRIVE/Assign_5/0.png"))))
images.append(np.asarray(ImageOps.grayscale(Image.open("/content/drive/MyDrive/ML_DRIVE/Assign_5/1.png"))))
images.append(np.asarray(ImageOps.grayscale(Image.open("/content/drive/MyDrive/ML_DRIVE/Assign_5/2.png"))))
images.append(np.asarray(ImageOps.grayscale(Image.open("/content/drive/MyDrive/ML_DRIVE/Assign_5/3.png"))))
images.append(np.asarray(ImageOps.grayscale(Image.open("/content/drive/MyDrive/ML_DRIVE/Assign_5/8.png"))))

for i in range(5):
  images[i] = cv2.resize(images[i], (28,28), interpolation=cv2.INTER_AREA)
  images[i] = 255-images[i]
  plt.imshow(images[i])
  plt.show()
  prediction = np.argmax(model5.predict(np.expand_dims(images[i], axis=0)))
  print("Handwritten prediction", prediction, '\n')
```



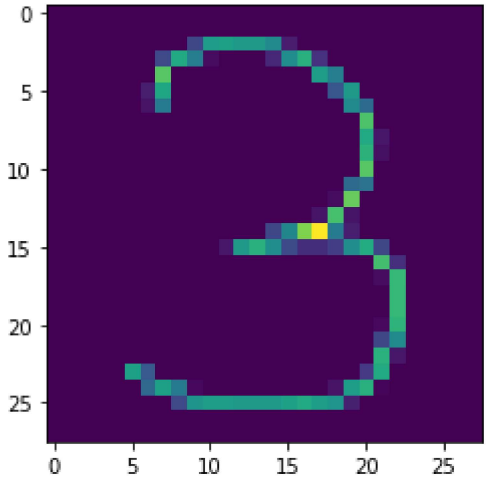
Handwritten prediction 0



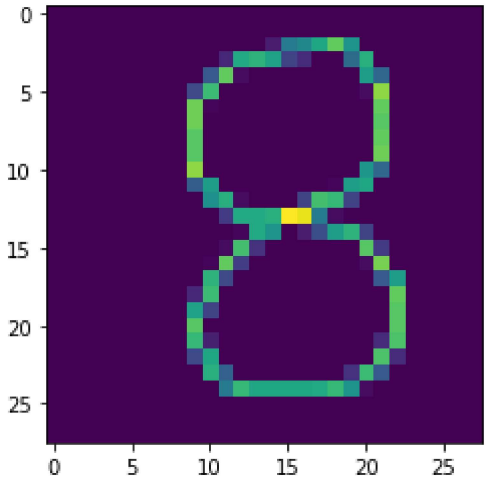
Handwritten prediction 1



Handwritten prediction 2



Handwritten prediction 3




```
Handwritten prediction 8
```

In [ ]: