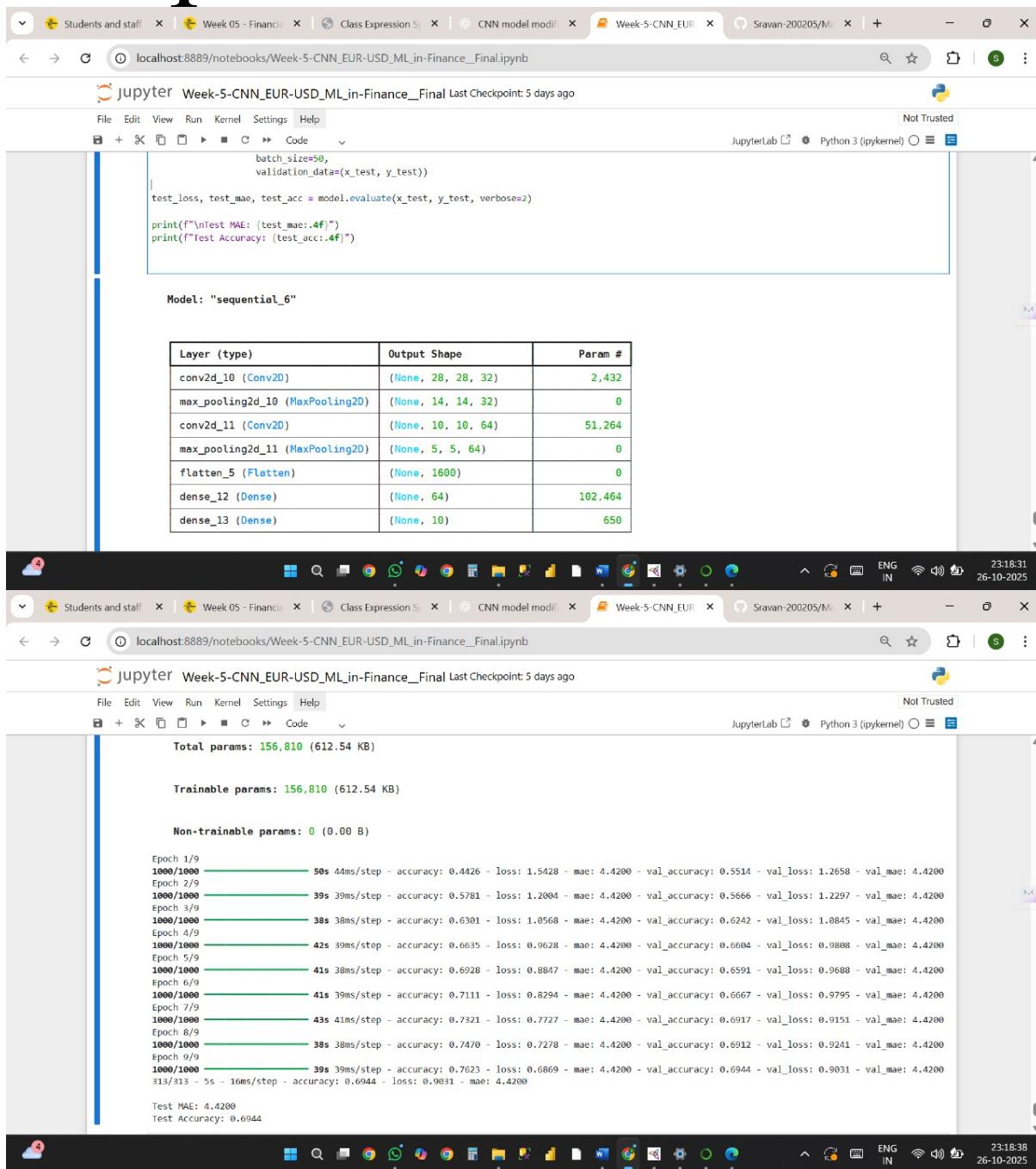


Week-5

Output



The screenshot shows a Jupyter Notebook interface with several tabs at the top: "Students and staff", "Week 05 - Finance", "Class Expression 5", "CNN model modif...", "Week-5-CNN_EUR", and "Sravan-202025/M...". The main area displays Python code and its output.

```
batch_size=50,
validation_data=(x_test, y_test))

test_loss, test_mae, test_acc = model.evaluate(x_test, y_test, verbose=2)

print(f"\nTest MAE: {test_mae:.4f}")
print(f"Test Accuracy: {test_acc:.4f}")

Model: "sequential_6"
```

| Layer (type) | Output Shape | Param # |
|---------------------------------|--------------------|---------|
| conv2d_10 (Conv2D) | (None, 28, 28, 32) | 2,432 |
| max_pooling2d_10 (MaxPooling2D) | (None, 14, 14, 32) | 0 |
| conv2d_11 (Conv2D) | (None, 10, 10, 64) | 51,264 |
| max_pooling2d_11 (MaxPooling2D) | (None, 5, 5, 64) | 0 |
| flatten_5 (Flatten) | (None, 1600) | 0 |
| dense_12 (Dense) | (None, 64) | 102,464 |
| dense_13 (Dense) | (None, 10) | 650 |

Total params: 156,810 (612.54 KB)

Trainable params: 156,810 (612.54 KB)

Non-trainable params: 0 (0.00 B)

Epoch 1/9
1000/1000 50s 44ms/step - accuracy: 0.4426 - loss: 1.5428 - mae: 4.4200 - val_accuracy: 0.5514 - val_loss: 1.2658 - val_mae: 4.4200
Epoch 2/9
1000/1000 39s 39ms/step - accuracy: 0.5781 - loss: 1.2004 - mae: 4.4200 - val_accuracy: 0.5666 - val_loss: 1.2297 - val_mae: 4.4200
Epoch 3/9
1000/1000 38s 38ms/step - accuracy: 0.6301 - loss: 1.0568 - mae: 4.4200 - val_accuracy: 0.6242 - val_loss: 1.0845 - val_mae: 4.4200
Epoch 4/9
1000/1000 42s 39ms/step - accuracy: 0.6635 - loss: 0.9628 - mae: 4.4200 - val_accuracy: 0.6604 - val_loss: 0.9888 - val_mae: 4.4200
Epoch 5/9
1000/1000 41s 38ms/step - accuracy: 0.6928 - loss: 0.8847 - mae: 4.4200 - val_accuracy: 0.6591 - val_loss: 0.9688 - val_mae: 4.4200
Epoch 6/9
1000/1000 41s 39ms/step - accuracy: 0.7111 - loss: 0.8294 - mae: 4.4200 - val_accuracy: 0.6667 - val_loss: 0.9795 - val_mae: 4.4200
Epoch 7/9
1000/1000 43s 41ms/step - accuracy: 0.7321 - loss: 0.7727 - mae: 4.4200 - val_accuracy: 0.6917 - val_loss: 0.9151 - val_mae: 4.4200
Epoch 8/9
1000/1000 38s 38ms/step - accuracy: 0.7470 - loss: 0.7278 - mae: 4.4200 - val_accuracy: 0.6912 - val_loss: 0.9241 - val_mae: 4.4200
Epoch 9/9
1000/1000 39s 39ms/step - accuracy: 0.7623 - loss: 0.6869 - mae: 4.4200 - val_accuracy: 0.6944 - val_loss: 0.9031 - val_mae: 4.4200
313/313 - 5s - 16ms/step - accuracy: 0.6944 - loss: 0.9031 - mae: 4.4200

Test MAE: 4.4200
Test Accuracy: 0.6944

The screenshot shows a Jupyter Notebook interface running on a local host at port 8889. The notebook title is "Jupyter Week-5-CNN_EUR-USD_ML_in-Finance_Final.ipynb". The code cell [48] contains Python code for building a Convolutional Neural Network (CNN) using TensorFlow and Keras. The code imports necessary modules, loads the CIFAR-10 dataset, preprocesses the data, defines a sequential model with two convolutional layers and two max pooling layers, adds a flatten layer, and two dense layers (one with 64 units and one with 10 units for softmax output). It then compiles the model with Adam optimizer, sparse categorical crossentropy loss, and metrics of mae and accuracy. Finally, it fits the model to the training data and evaluates it on the test data.

```
[48]:  
import tensorflow as tf  
from tensorflow.keras import datasets, layers, models  
from tensorflow.keras.optimizers import Adam  
(x_train, y_train), (x_test, y_test) = datasets.cifar10.load_data()  
x_train, x_test = x_train / 255.0, x_test / 255.0  
  
model = models.Sequential()  
    layers.Conv2D(32, (3,3), activation='relu', input_shape=(32,32,3)),  
    layers.MaxPooling2D((2,2)),  
    layers.Conv2D(64, (3,3), activation='relu'),  
    layers.MaxPooling2D((2,2)),  
    layers.Flatten(),  
    layers.Dense(64, activation='relu'),  
    layers.Dense(10, activation='softmax')  
]  
  
model.compile(optimizer=Adam(),  
              loss='sparse_categorical_crossentropy',  
              metrics=['mae', 'accuracy'])  
  
model.summary()  
  
history = model.fit(x_train, y_train,  
                     epochs=9,  
                     batch_size=50,  
                     validation_data=(x_test, y_test))  
  
test_loss, test_mae, test_acc = model.evaluate(x_test, y_test, verbose=2)
```