Ass. 6 Implement Boston housing price Prediction Problem by Linear Regression by using Deep Neural Network. Use Boston House price prediction dataset.

```
import tensorflow as tf
from tensorflow.keras.datasets import boston_housing
from sklearn import preprocessing
# Load the Boston Housing dataset
(train_x,train_y),(test_x,test_y) = boston_housing.load_data()

print("Train shape :",train_x.shape)
print("Test shape :",test_x.shape)
print("Actual Train output :",train_y.shape)
print("Actual test output :",test_y.shape)

test_x[4]
test_y[4]
# normalize the dataset
train_x = preprocessing.normalize(train_x)
test_x = preprocessing.normalize(test_x)

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import *
from keras.layers import Activation, Dense
# Build the model
def HousePricePredictionModel():
   model = Sequential()
   model.add(Dense(128,activation='relu',input_shape = (train_x[0].shape)))
   model.add(Dense(64,activation='relu'))
   model.add(Dense(32,activation='relu'))
   model.add(Dense(1))
# Compile the model
   model.compile(optimizer='rmsprop',loss='mse',metrics=['mae'])
   return model

import numpy as np
k = 4
num_val_samples = len(train_x)
num_epochs = 100
all_scores = []
# Train the model
model = HousePricePredictionModel()
history = model.fit(x=train_x, y=train_y, epochs = num_epochs, batch_size=1, verbose
=1, validation_data=(test_x,test_y))
# Test the model
test_input = [[1.52158193e-04, 0.00000000e+00, 9.55377269e-03,
0.00000000e+00,9.55377269e-04, 1.30241966e-02, 1.20858416e-01, 7.97410212e-
03,6.38336705e-03, 5.25563887e-01, 3.93640968e-02, 8.40795830e-01,1.79585393e-02]]
predicted_value = model.predict(test_input)
print("actual value is :",test_y[4])
print("predicted value is : ",predicted_value)
```

Ass.7 Classification using Deep neural network: Multiclass classification using Deep Neural Networks: Example: use the OCRLetter recognition dataset.

```
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.optimizers import RMSprop
from tensorflow.keras.datasets import mnist
import matplotlib.pyplot as plt
from sklearn import preprocessing
from sklearn import metrics
# Load the ocr dataset
(x_train, y_train), (x_test, y_test) = mnist.load_data()

print("X_train shape", x_train.shape)
print("y_train shape", y_train.shape)
print("X_test shape", x_test.shape)
print("y_test shape", y_test.shape)
# Reshape the dataset as it is 3 dimensional
x_train = x_train.reshape(60000, 784)
x_test = x_test.reshape(10000, 784)
x_train = x_train.astype('float32')

x_test = x_test.astype('float32')
x_train /= 255 # Each image has Intensity from 0 to 255
x_test /= 255

num_classes = 10
y_train = np.eye(num_classes)[y_train]
y_test = np.eye(num_classes)[y_test]
# Build the model
import tensorflow as tf
model = Sequential()
model.add(Dense(512, activation='relu', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation='softmax'))
# Compile the model
model.compile(loss='categorical_crossentropy',optimizer=RMSprop(),metrics=['accuracy'
])
# Train the model
batch_size = 128
epochs = 20
history = model.fit(x_train,
y_train,batch_size=batch_size,epochs=epochs,verbose=1,validation_data=(x_test,
y_test))
# Evaluate the model
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

Ass 8. Convolutional Neural Network(CNN): 1. Use MNIST Fashion dataset and create a classifier to classify fashion clothing into categories.

```
import tensorflow as tf
from tensorflow import keras
import numpy as np
import matplotlib.pyplot as plt
# Load the MNIST Fashion dataset
fashion_mnist = keras.datasets.fashion_mnist
(train_images, train_labels), (test_images, test_labels) = fashion_mnist.load_data()
# Define class names for the fashion categories
class_names = [
    'T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat',
    'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot'
]
# Preprocess the data
train_images = train_images / 255.0
test_images = test_images / 255.0
# Display some sample images
plt.figure(figsize=(10, 10))
for i in range(25):
    plt.subplot(5, 5, i + 1)
    plt.xticks([])
    plt.yticks([])
    plt.grid(False)
    plt.imshow(train_images[i], cmap=plt.cm.binary)
    plt.xlabel(class_names[train_labels[i]])
plt.show()
# Build the model
model = keras.Sequential([
    keras.layers.Flatten(input_shape=(28, 28)),
    keras.layers.Dense(128, activation='relu'),
    keras.layers.Dense(10, activation='softmax')
])
# Compile the model
model.compile(optimizer='adam',loss=tf.keras.losses.SparseCategoricalCrossentropy(fro
m_logits=True),metrics=['accuracy'])
# Train the model
model.fit(train_images, train_labels, epochs=10)
# Evaluate the model
test_loss, test_acc = model.evaluate(test_images, test_labels, verbose=2)
print('\nTest accuracy:', test_acc)
# Make predictions
predictions = model.predict(test_images)
```