```
import tensorflow as tf
from tensorflow import keras
import numpy as np
import matplotlib.pyplot as plt

N = 1000
#N = 5000
K = 10
M = 40000
#M = 60000
dt = 0.008
#dt = 0.001

Nbatch = 1
validation_split_ratio = 0.2
EPOCHS = np.int64(M/(N*(1-validation_split_ratio)/Nbatch))

np.random.seed(123)
tf.random.set_seed(124)
xs = np.random.uniform(-4,4,size=(N,1))

def f(x):
   return np.square(np.abs(x-0.5))
   #return np.cos(x)

ys = f(xs)

Input_layer = tf.keras.layers.InputLayer(1,)

Hidden_layer = keras.layers.Dense(units=K,
                                  activation="sigmoid",
                                  use_bias=True,

kernel_initializer='random_normal',
                                  bias_initializer='zeros',
                                  name="hidden_layer_1")

Output_layer = keras.layers.Dense(units=1,
                                  use_bias=False)

model = keras.Sequential([Input_layer, Hidden_layer,
Output_layer])
optimizer = tf.keras.optimizers.SGD(learning_rate=dt)

model.compile(optimizer=optimizer, loss='mean_squared_error')

history = model.fit(x=xs,
                   y=ys,
                   batch_size=Nbatch,
                   epochs=EPOCHS,
                   validation_split=validation_split_ratio,
                   verbose=1)

pts = np.linspace(-4,4,300).reshape(-1,1)
target_fcn_vals = f(pts)
alpha_vals = model(pts)

plt.figure('Alfa',figsize=(15,10))
plt.plot(pts,target_fcn_vals,label='f')
plt.plot(pts,alpha_vals,label='alfa')
plt.legend(['f','alfa'])
#plt.savefig('/Users/siobhanie/Desktop/result2.png')
plt.show()

plt.figure('Fel',figsize=(15,10))
plt.semilogy(history.history['loss'],label='Träningsfelet')
plt.semilogy(history.history['val_loss'],label='Generaliseringsfel
et')
plt.xlabel('Epok')
plt.ylabel('Medelkvadratfel')
plt.legend()
plt.grid(True)
#plt.savefig('/Users/siobhanie/Desktop/error2.png')
plt.show()
```