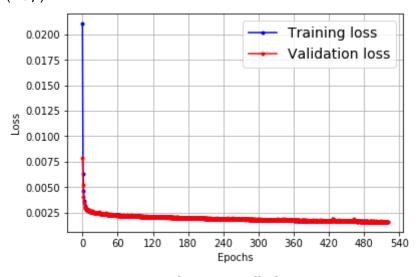
```
import matplotlib as mpl
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler
from scipy import stats
import seaborn as sns
import tensorflow as tf
from tensorflow import keras
import sklearn
import sys
import io
def plot learning curves(loss, val loss):
    plt.figure()
    plt.plot(np.arange(len(loss)), loss, "b.-", label="Training loss")
    plt.plot(np.arange(len(val_loss)), val_loss, "r.-", label="Validation loss")
    plt.gca().xaxis.set_major_locator(mpl.ticker.MaxNLocator(integer=True))
   plt.legend(fontsize=14)
   plt.xlabel("Epochs")
   plt.ylabel("Loss")
    plt.grid(True)
n \text{ steps} = 50
forecast = 10
#EXTRACT FLOW, Z SCORE, OUTLIERS
data west = pd.read csv('denoised data 1015 west.csv')
data west o = np.array(data west.FLOW)
print(data west o.shape)
#EXRTEND DATA
array to concatinate = data west o
for iter in range (35):
    data west o = np.concatenate([data west o,array to concatinate])
#SCALE AND RESHAPE DATA
scaler = MinMaxScaler()
array = data_west_o.reshape(-1, 1)
array scaled = scaler.fit transform(array)
flow_reshaped = array_scaled[:(len(array_scaled) - (len(array_scaled) % (n_steps+forecast
print(flow reshaped.shape)
#TRAIN SET, VALIDATION SET, TEST SET
test = int(0.7 * flow reshaped.shape[0])
valid = int(0.9 * flow_reshaped.shape[0])
X train = flow reshaped[:test, :n steps]
X valid = flow reshaped[test:valid, :n steps]
X_test = flow_reshaped[valid:, :n_steps]
print(X test.shape)
print(X test[-1:].shape)
#prepare targets
Y = np.empty((flow_reshaped.shape[0], n_steps, forecast))
for step ahead in range(1, forecast + 1):
    Y[:, :, step ahead - 1] = flow reshaped[:, step ahead:step ahead + n steps, 0]
y train = Y[:test]
y valid = Y[test:valid]
y test = Y[valid:]
#DEFINE AND COMPILE MODEL
model = keras.models.Sequential([
   keras.layers.LSTM(20, return sequences=True, input shape=[None, 1]),
    keras.layers.LSTM(20, return sequences=True),
    keras.layers.TimeDistributed(keras.layers.Dense(forecast))
])
```

```
def last_time_step_mse(Y_true, Y_pred):
    return keras.metrics.mean squared error(Y true[:, -1], Y pred[:, -1])
model.compile(loss="mse", optimizer="adam", metrics=[last_time_step_mse])
early stopping cb = keras.callbacks.EarlyStopping(patience=15, restore best weights=True)
history = model.fit(X_train, y_train, epochs=700,
                     validation_data=(X_valid, y_valid), callbacks=[early_stopping_cb])
model.save("lstm 1015 west.h5")
plot learning curves(history.history["loss"], history.history["val loss"])
#50 MINS PREDICTION
flow unscaled = array[:(len(array) - (len(array) % (n steps + forecast)))].reshape(-1, (n
y test unscaled = flow unscaled[valid:, n steps:, 0]
y_real_rescaled = y_test_unscaled[-1, :].reshape(-1, 1)
print(y_real_rescaled.shape)
flow_not_reshaped = array[:(len(array) - (len(array) % (n_steps+forecast)))]
#flow prediction
y_pred = model.predict(X_test[-1, :].reshape(-1, n_steps, 1)) #shape (1, 50, 10)
y_pred = y_pred[-1,-1,:].reshape(-1,1)
y_pred_rescaled = scaler.inverse_transform(y_pred).reshape(-1, 1) #shape (10, 1)
print(y pred rescaled.shape)
#t.ime
time not reshaped = np.array(data west['TIME'][:(len(data west['TIME']) - (len(data west[
time reshaped = np.array(data west['TIME'][:(len(data west['TIME']) - (len(data west['TIME'))
    reshape(-1, (n steps+forecast), 1)
valid_time = int(0.9 * time_reshaped.shape[0])
y time test = time reshaped[valid time:, n steps:, 0]
print(y time test[-1, :].shape)
def plot prediction(y real resacled, y pred rescaled, flow not reshaped, time not reshape
    plt.figure()
    plt.title("50 minutes prediction", fontsize=14)
    plt.plot(time_not_reshaped[-300:-forecast], flow_not_reshaped[-300:-forecast], 'b-')
    plt.plot(y_time_test[-1, :], y_real_resacled, 'ro-', label = 'Real values')
plt.plot(y_time_test[-1, :], y_pred_rescaled, 'gx-', label = 'Predicted values')
    plt.legend(loc="upper left")
    plt.xlabel("Time (in 5 minutes intervals)")
    plt.ylabel('Volume (veh/h)')
plot prediction(y real rescaled, y pred rescaled, flow not reshaped, time not reshaped, y
plt.show()
```

С→

```
Epoch 511/700
Epoch 512/700
Epoch 513/700
Epoch 514/700
Epoch 515/700
Epoch 516/700
Epoch 517/700
Epoch 518/700
Epoch 519/700
Epoch 520/700
Epoch 521/700
7499/7499 [=====
     ========= ] - 17s 2ms/sample - loss: 0.0016 - 1a
(10, 1)
(10, 1)
(10,)
```





Time (in 5 minutes intervals)