Лабораторная работа № 3.

Многослойные сети. Алгоритм обратного распространения ошибки

Целью работы является исследование свойств многослойной нейронной сети прямого распространения и алгоритмов ее обучения, применение сети в задачах классификации и аппроксимации функции.

[1]:
```
import os
os.environ["TF_CPP_MIN_LOG_LEVEL"] = "3"

import itertools
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
```

Задание 1

Вспомогательные константы

[2]:
```
BATCH = 20
EPOCHES = 200
EPS = 1e-6
N = 100
M = 3
```

Уравнение эллипса в параметрическом виде и поворот точек на угол phi

[3]:
```
def ellipse(t, a, b, x0, y0):
    x = x0 + a * np.cos(t)
    y = y0 + b * np.sin(t)
    return x, y

def rotate(x, y, phi):
    xr = x * np.cos(phi) - y * np.sin(phi)
    yr = x * np.sin(phi) + y * np.cos(phi)
    return xr, yr
```

[4]:
```
a = [0.4, 0.7, 1.0]
b = [0.15, 0.5, 1.0]
alpha = [np.pi / 6, -np.pi / 3, 0]
x0 = [-0.1, 0, 0]
y0 = [0.15, 0, 0]
```

Подготовка обучающих данных

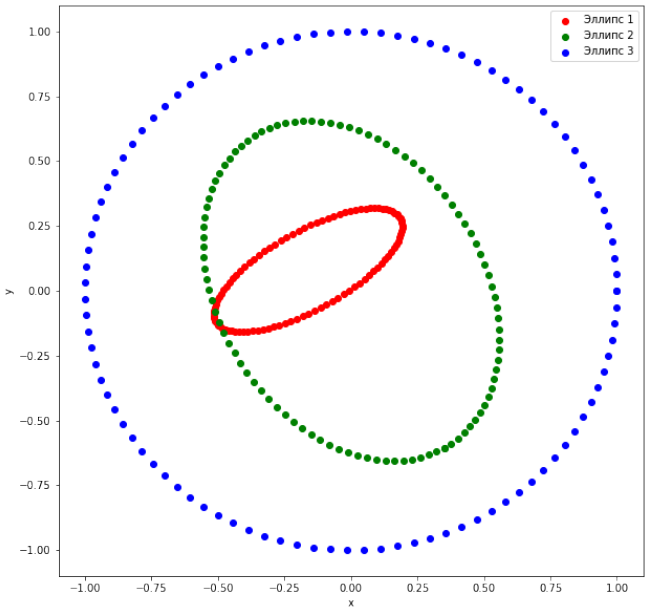
[5]:
```
t = np.linspace(0, 2 * np.pi, N)

train_x = np.empty([0, 2])
train_y = np.empty([0, M])
```

```
clrs = ["r", "g", "b"]
lbls = ["Эллипс 1", "Эллипс 2", "Эллипс 3"]
figure = plt.figure(figsize = (10, 10))

for i in range(M):
    xe, ye = ellipse(t, a[i], b[i], x0[i], y0[i])
    xr, yr = rotate(xe, ye, alpha[i])
    plt.scatter(xr, yr, color = clrs[i], label = lbls[i])
    fig_x = np.array(list(zip(xr, yr)))
    train_x = np.concatenate((train_x, fig_x), axis = 0)
    fig_y = np.full([N, M], EPS)
    for j in range(N):
        fig_y[j][i] = 1.0 - fig_y[j][i]
    train_y = np.concatenate((train_y, fig_y), axis = 0)

plt.ylabel("y")
plt.xlabel("x")
plt.legend()
plt.show()
```



Построение и обучение модели

[6]:
```
model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(32, input_dim = 2, activation = "relu"),
    tf.keras.layers.Dense(M, activation = "sigmoid")
])
```

```
model.compile(
    optimizer = tf.keras.optimizers.Adam(learning_rate = 1e-2),
    loss = "mse",
    metrics = ["mae"]
)

hst = model.fit(x = train_x, y = train_y, batch_size = BATCH, epochs = EPOCHES,␣
 ↪verbose = False, shuffle = True)
```

Визуализация результата

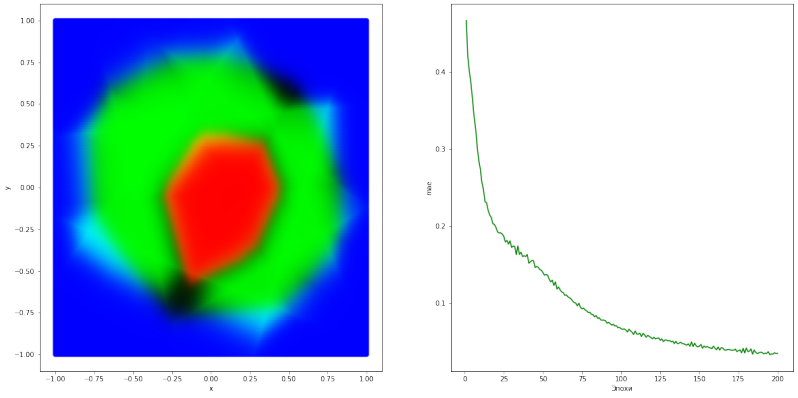
[7]:
```
x = np.linspace(-1, 1, 200)
y = np.linspace(-1, 1, 200)

mx, my = np.meshgrid(x, y)
xy = np.array(list(itertools.product(x, y)))

figure = plt.figure(figsize = (20, 10))
axes = figure.add_subplot(121)
pred = model.predict(xy)
plt.scatter(mx, my, c = pred)
plt.ylabel("y")
plt.xlabel("x")

axes = figure.add_subplot(122)
epticks = [(i + 1) for i in range(len(hst.history["mae"]))]
plt.plot(epticks, hst.history["mae"], "g")
plt.ylabel("mae")
plt.xlabel("Эпохи")

plt.show()
```



Задание 2

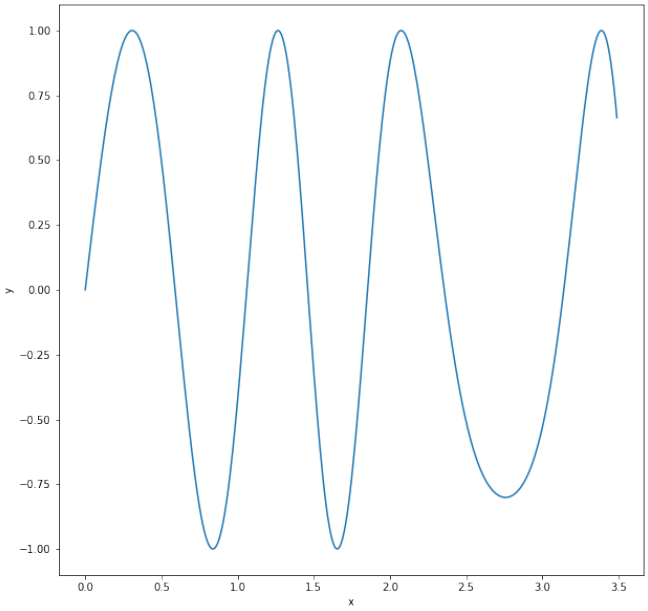
[8]:
```
def x(t):
    return np.sin(np.sin(t) * t * t + 5 * t)
```

Подготовка обучающих данных

[9]:
```
h = 0.01

train_x = np.arange(0, 3.5, h)
train_y = x(train_x)

figure = plt.figure(figsize = (10, 10))
plt.plot(train_x, train_y)
plt.ylabel("y")
plt.xlabel("x")
plt.show()
```



Построение и обучение модели

[10]:
```
model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(64, input_dim = 1, activation = "tanh"),
    tf.keras.layers.Dense(32, activation = "tanh"),
    tf.keras.layers.Dense(1, activation = "tanh")
])

model.compile(
```

```
    optimizer = tf.keras.optimizers.Adam(learning_rate = 1e-2),
    loss = "mse",
    metrics = ["mae"]
)

hst = model.fit(x = train_x, y = train_y, batch_size = BATCH, epochs = EPOCHES,␣
 ↪verbose = False, shuffle = True)
```

Визуализация результата

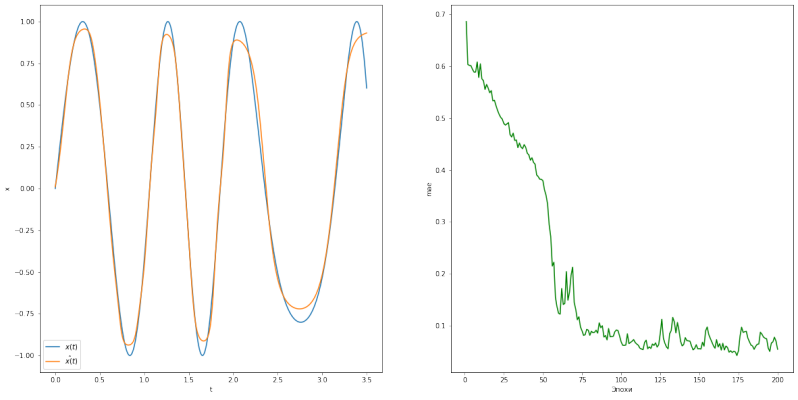
[11]:
```
valid_x = np.arange(0, 3.5, h / 10)
valid_y = x(valid_x)

figure = plt.figure(figsize = (20, 10))

axes = figure.add_subplot(121)
plt.plot(valid_x, valid_y, label = "$x(t)$")
plt.plot(valid_x, model.predict(valid_x), label = "$x\^(t)$")
plt.ylabel("x")
plt.xlabel("t")
plt.legend()

axes = figure.add_subplot(122)
epticks = [(i + 1) for i in range(len(hst.history["mae"]))]
plt.plot(epticks, hst.history["mae"], "g")
plt.ylabel("mae")
plt.xlabel("Эпохи")

plt.show()
```



Вывод

В ходе выполнения лабораторной работы я ознакомился с многослойными нейронными сетями и их свойствами. Реализовал две многослойные модели для решения задач классификации и апрос-

кимации.

[ ]: