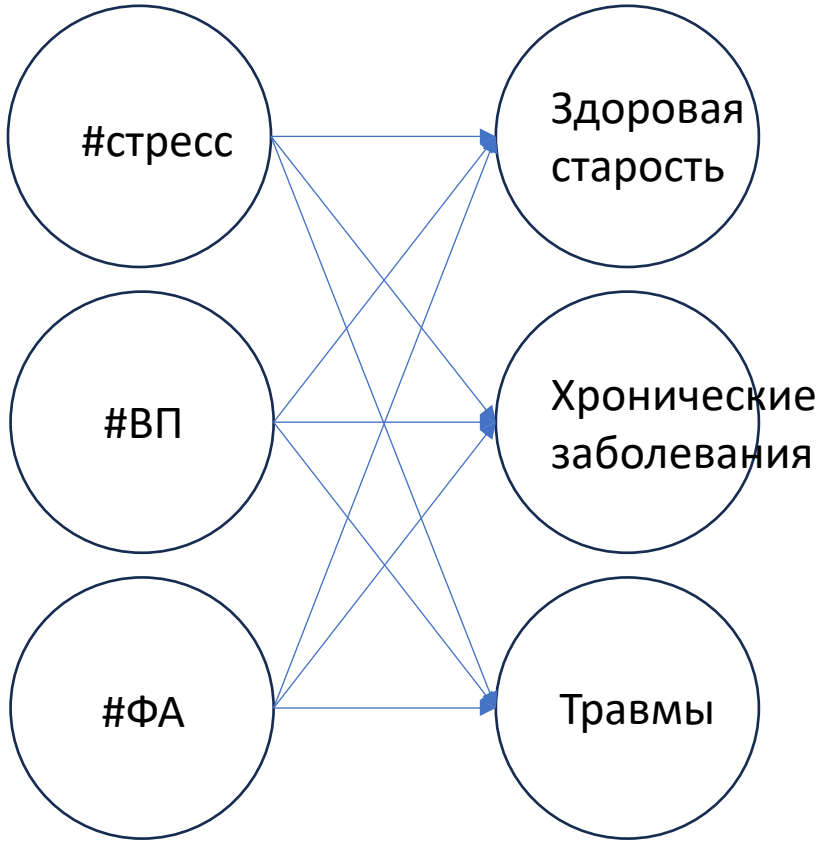
Введение в глубокое обучение

Лабораторная работа №2

1. Реализуйте корректировку весов для градиентного спуска пункт #4 программы




```python
# 1. Чистая нейронная сеть с несколькими входами и выходами

#Веса
          #стресс   #ВП   #ФА
weights = [[0.01, 0.09, 0.01], # здоровая старость
           [0.01, -0.04, 0.0], # хронические заболевания
           [0.0, 0.17, 0.03]] # травмы

# Взвешенная сумма (свертка) двух тензоров
def w_sum(a,b):
    assert(len(a) == len (b))
    output = 0
    for i in range(len(a)):
        output += a[i]*b[i]
    return output

# Векторное умножение матриц (тензоров)
def vect_mat_mul(vect, matrix):
    assert(len(vect) == len(matrix))
    output = [0,0,0]
    for i in range(len(vect)):
        output[i] = w_sum(vect, matrix[i])
    return output

def neural_network(input, weights):
    pred = vect_mat_mul(input, weights)
    return pred
```

```
# 2. ПРОГНОЗ: получение прогноза и вычисление среднеквадратичной ошибки и чистой ошибки
# Входные данные
stress = [5, 8, 15, 6, 30]     # стресс
bad_habits = [8, 4, 2, 6, 1]   # плохие привычки
phys_active =  [1, 1, 2, 1, 3] # физическая активность

input = [stress[0], bad_habits[0], phys_active[0]]

# Верные прогноз

healthy = [0.43, 0.62, 0.64, 0.5, 0.51]  # здоровая старость
diseses = [3, 7, 5, 8, 3]                # хронические заболевания
hirts = [1, 2, 3, 2, 5]                  # травмы

true = [healthy[0], diseses[0], hirts[0]]

alpha = 0.001 # альфа-коэффициент, чтобы сеть не шла в расхождение при больших значениях input

error = [0,0,0]
delta = [0.0,0.0,0.0]

for inter in range(100):
    pred = neural_network(input, weights)
    print("Pred: " + str(pred))
    for i in range(len(true)):
        error[i] = (pred[i] - true[i])**2
        delta[i] = pred[i] - true[i]
```

```
# 3. СРАВНЕНИЕ: вычисление каждого приращения (производной) weight_delta

def outer_prod(vec_a, vec_b):
     out = [[0, 0, 0],
            [0, 0, 0],
            [0, 0, 0]]
    for i in range(len(vec_a)):
        for j in range(len(vec_b)):
            out[i][j] = vec_a[i]*vec_b[j]
    return out

    weight_deltas = outer_prod(input, delta)
```

#4 Корректировка весов
do your code here

2. Проверьте правильность работы сети. Сравните вывод предсказания сети со значениями true в коде программы. За сколько итерация сеть научилась давать правильный прогноз? Проверьте правильность прогноза для всех 5 наборов входных данных (переменная input).
3. Задайте свои входные данные и верные прогнозы и проверьте правильность работы сети.

Пункт #4 в коде для проверки (для корректировки весов)

```
# 4. КОРРЕКТИРОВКА весов
 for i in range(len(true)):
        for j in range(len(true)):
                weights[i][j] -= (weight_deltas[j][i]*alpha)
```