`

Министерство образования Республики Беларусь

Учреждение образования

«Брестский Государственный технический университет»

Кафедра ИИТ

Отчет по лабораторной работе 3

Специальность ИИ-23

Выполнил:

Макаревич Н.Р.

Студент группы ИИ-23

Проверил:

Андренко К. В.
Преподаватель-стажёр
Кафедры ИИТ,
«___» _____ 2025 г.

Брест 2025

`

Цель: научиться осуществлять предобучение нейронных сетей с помощью RBM

Код программы:

```
# -*- coding: utf-8 -*-

"""iad_4.ipynb


Automatically generated by Colab.


Original file is located at

    https://colab.research.google.com/drive/1QqZP4gO9Cs6z0RH-Kvl8x_mQAWCqMaht
"""


pip install ucimlrepo


import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns


from ucimlrepo import fetch_ucirepo

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler, LabelEncoder

from sklearn.metrics import accuracy_score, classification_report, confusion_matrix


from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Input
```

```
from sklearn.neural_network import BernoulliRBM

from tqdm import tqdm


maternal_health_risk = fetch_ucirepo(id=863)

X = maternal_health_risk.data.features

y = maternal_health_risk.data.targets


le = LabelEncoder()

y = le.fit_transform(y.values.ravel())


scaler = StandardScaler()

X_scaled = scaler.fit_transform(X)


X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2,
random_state=42)


def build_model():

    model = Sequential([

        Input(shape=(X_train.shape[1],)),

        Dense(64, activation='relu'),

        Dense(32, activation='relu'),

        Dense(3, activation='softmax')

    ])

    model.compile(optimizer='adam', loss='sparse_categorical_crossentropy',
metrics=['accuracy'])

    return model
```

`

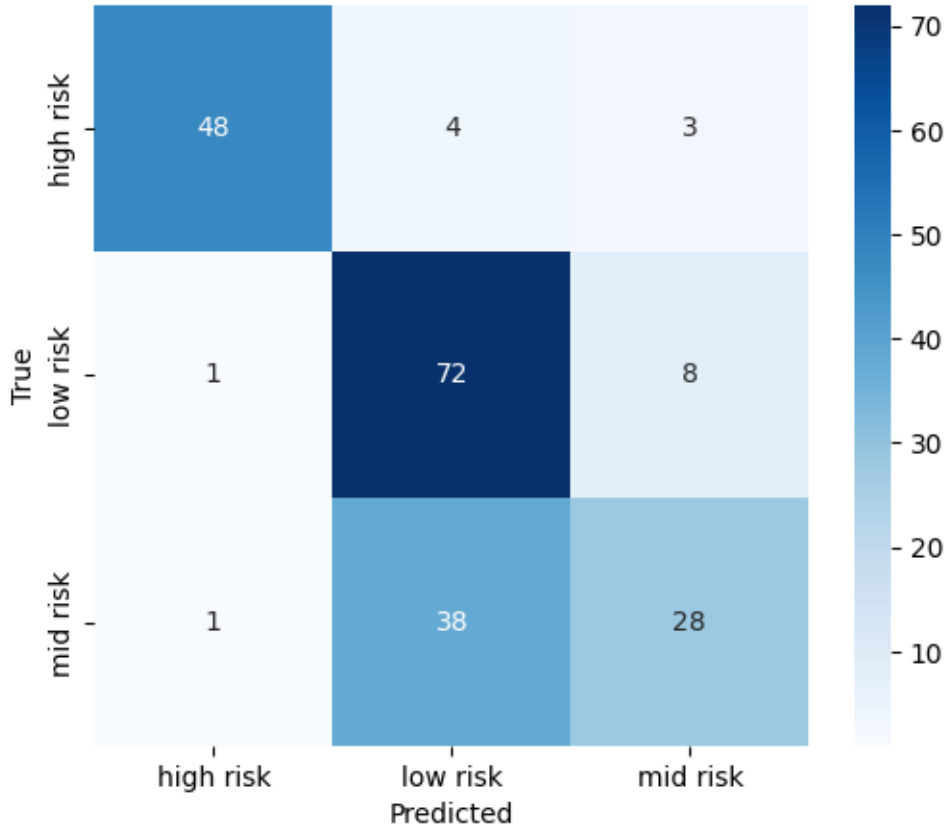
```
model_base = build_model()

history_base = model_base.fit(X_train, y_train, epochs=50, batch_size=16,
validation_split=0.2, verbose=0)


y_pred_base = np.argmax(model_base.predict(X_test), axis=1)

print("Точность без предобучения:", accuracy_score(y_test, y_pred_base))

print(classification_report(y_test, y_pred_base))


rbm1 = BernoulliRBM(n_components=64, learning_rate=0.05, n_iter=15, random_state=42,
verbose=True)

rbm2 = BernoulliRBM(n_components=32, learning_rate=0.05, n_iter=15, random_state=42,
verbose=True)


X_rbm1 = rbm1.fit_transform(X_train)


X_rbm2 = rbm2.fit_transform(X_rbm1)


print("Форма после RBM:", X_rbm2.shape)


model_rbm = Sequential([
    Input(shape=(X_train.shape[1],)),
    Dense(64, activation='relu'),
    Dense(32, activation='relu'),
    Dense(3, activation='softmax')
])


model_rbm.layers[0].set_weights([rbm1.components_.T, np.zeros(64)])
```

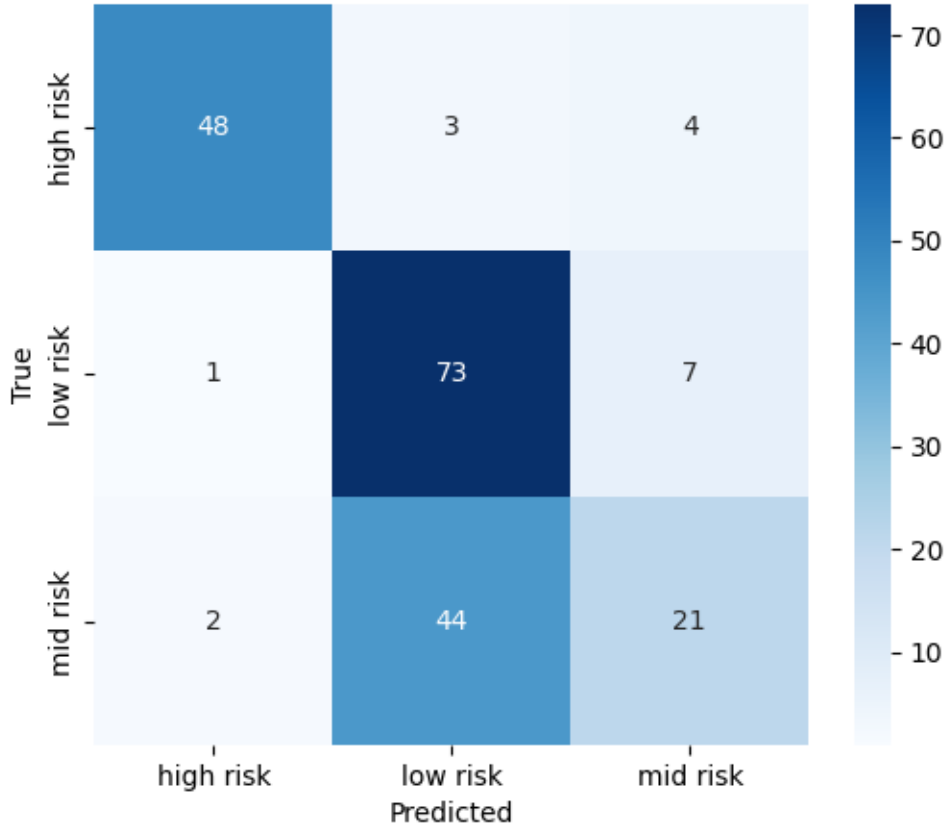
`

```
model_rbm.layers[1].set_weights([rbm2.components_.T, np.zeros(32)])


model_rbm.compile(optimizer='adam', loss='sparse_categorical_crossentropy',
metrics=['accuracy'])


history_rbm = model_rbm.fit(X_train, y_train, epochs=50, batch_size=16,
validation_split=0.2, verbose=0)


y_pred_rbm = np.argmax(model_rbm.predict(X_test), axis=1)

print("Точность после предобучения RBM:", accuracy_score(y_test, y_pred_rbm))

print(classification_report(y_test, y_pred_rbm))
```

Результат 3 лаб работы:

`


high risk
low risk
mid risk
True
48
4
3
1
72
8
1
38
28
high risk
low risk
mid risk
Predicted
70
60
50
40
30
20
10


high risk
low risk
mid risk
True
48
3
4
1
73
7
2
44
21
high risk
low risk
mid risk
Predicted
70
60
50
40
30
20
10

`

```
Baseline (no pretraining):
Accuracy: 0.729064039408867
Macro F1: 0.7270163798465684

With pretraining:
Accuracy: 0.6995073891625616
Macro F1: 0.6854236536016316
```

Датасет 4 лаб работы:

```
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
Точность без предобучения: 0.6896551724137931
              precision    recall  f1-score   support

           0       0.79      0.87      0.83        47
           1       0.63      0.80      0.70        80
           2       0.71      0.46      0.56        76

    accuracy                           0.69       203
   macro avg       0.71      0.71      0.70       203
weighted avg       0.70      0.69      0.68       203
```

```
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
Точность после предобучения RBM: 0.39408866995073893
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        47
           1       0.39      1.00      0.57        80
           2       0.00      0.00      0.00        76

    accuracy                           0.39       203
   macro avg       0.13      0.33      0.19       203
weighted avg       0.16      0.39      0.22       203
```

Вывод: научился предобучать НС с помощью RBM.