|  |  |  |
| --- | --- | --- |
| Лабораторная работа № 8. Искусственные нейронные сети | Студент | Бокова О. Д. |
| Группа | ИВТ-363 |
| Преподаватель | Фокин Р.О. |
| Оценка |  |
| Дата | 17.04.2023 |

Для выбранного или произвольного набора данных из репозитория

UC Irvine Machine Learning Repository решить задачу классификации (кластеризации или распознавания образов) с использованием ИНС. Разработать собственную ИНС. Создать топологию ИНС. Обучить ИНС. Выполнить эмуляцию сети.

Код:

import pandas as pd

from sklearn.preprocessing import LabelEncoder

from sklearn.model\_selection import train\_test\_split

from keras.models import Sequential

from keras.layers import Dense

# Загрузка датасета

df = pd.read\_csv('biofam.csv', sep=';')

print(df['idhous'].isna().sum()) # Проверка наличия значений NaN

print(df['idhous'].isin([np.inf, -np.inf]).sum()) # Проверка наличия значений inf или -inf

# Замена значений NaN на среднее значение

mean\_value = df['idhous'].mean()

df['idhous'].fillna(mean\_value, inplace=True)

# Замена значений inf на другое значение (например, -999)

df.replace([np.inf, -np.inf], -999, inplace=True)

# Преобразование столбцов в целочисленный вид

df['idhous'] = df['idhous'].astype('int')

df['birthyr'] = df['birthyr'].astype('int')

from sklearn.preprocessing import LabelEncoder

# Создание кодировщика

le = LabelEncoder()

# Преобразование строковых значений в числовые значения

df['nat\_1\_02'] = le.fit\_transform(df['nat\_1\_02'])

df['plingu02'] = le.fit\_transform(df['plingu02'])

df['p02r01'] = le.fit\_transform(df['p02r01'])

df['p02r04'] = le.fit\_transform(df['p02r04'])

df['cspfaj'] = le.fit\_transform(df['cspfaj'])

df['cspmoj'] = le.fit\_transform(df['cspmoj'])

df['nat\_1\_02'] = df['nat\_1\_02'].astype('int')

df['plingu02'] = df['plingu02'].astype('int')

df['p02r01'] = df['p02r01'].astype('int')

df['p02r04'] = df['p02r04'].astype('int')

df['cspfaj'] = df['cspfaj'].astype('int')

df['cspmoj'] = df['cspmoj'].astype('int')

df['a15'] = df['a15'].astype('int')

df['a16'] = df['a16'].astype('int')

df['a17'] = df['a17'].astype('int')

df['a18'] = df['a18'].astype('int')

df['a19'] = df['a19'].astype('int')

df['a20'] = df['a20'].astype('int')

df['a21'] = df['a21'].astype('int')

df['a22'] = df['a22'].astype('int')

df['a23'] = df['a23'].astype('int')

df['a24'] = df['a24'].astype('int')

df['a25'] = df['a25'].astype('int')

df['a26'] = df['a26'].astype('int')

df['a27'] = df['a27'].astype('int')

df['a28'] = df['a28'].astype('int')

df['a29'] = df['a29'].astype('int')

df['a30'] = df['a30'].astype('int')

df['wp00tbgp'] = df['wp00tbgp'].astype('int')

df['wp00tbgs'] = df['wp00tbgs'].astype('int')

# Кодирование категориальных признаков (столбец 'sex')

le = LabelEncoder()

df['sex'] = le.fit\_transform(df['sex'])

# Разделение датасета на обучающую и тестовую выборки

X = df.drop(['nat\_1\_02'], axis=1) # Все столбцы, кроме 'nat\_1\_02', являются признаками

y = df['nat\_1\_02'] # Столбец 'nat\_1\_02' является целевой переменной

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Создание нейронной сети

model = Sequential()

model.add(Dense(32, input\_shape=(X\_train.shape[1],), activation='relu'))

model.add(Dense(16, activation='relu'))

model.add(Dense(1))

# Компиляция модели

model.compile(loss='mse', optimizer='adam')

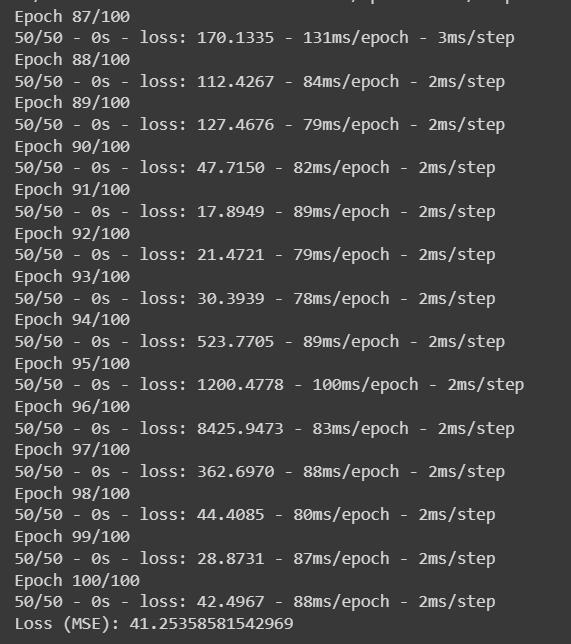
# Обучение модели

model.fit(X\_train, y\_train, epochs=100, batch\_size=32, verbose=2)

# Оценка точности модели

score = model.evaluate(X\_test, y\_test, verbose=0)

print('Loss (MSE):', score)



from sklearn.metrics import mean\_squared\_error, r2\_score

# Предсказание модели на тренировочных данных

y\_train\_pred = model.predict(X\_train)

train\_mse = mean\_squared\_error(y\_train, y\_train\_pred)

train\_r2 = r2\_score(y\_train, y\_train\_pred)

print('Train MSE:', train\_mse)

print('Train R^2:', train\_r2)

# Предсказание модели на тестовых данных

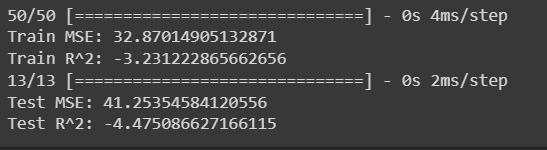
y\_test\_pred = model.predict(X\_test)

test\_mse = mean\_squared\_error(y\_test, y\_test\_pred)

test\_r2 = r2\_score(y\_test, y\_test\_pred)

print('Test MSE:', test\_mse)

print('Test R^2:', test\_r2)



import matplotlib.pyplot as plt

# График точности предсказания модели

plt.scatter(y\_test, y\_test\_pred)

plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], 'k--', lw=2)

plt.xlabel('True values')

plt.ylabel('Predictions')

plt.show()

