ЦЕНТР ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ
МГТУ им. Н.Э. Баумана

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА по курсу «Data Science»

Прогнозирование конечных свойств новых материалов (композиционных материалов)

Зиянгирова Зинаида Михайловна

ЦЕНТР ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ
МГТУ им. Н.Э. Баумана

Характеристики датасета

В исследуемом датасете (X_set) содержится 13 столбцов и 1023 строки. Произведен анализ датасета, получена информация и описательная статистика. В данных отсутствуют пропуски и строки-дубликаты.

```
X_set = X_bp.join(X_nup, how='inner')
X_set.head()
```

	Соотношение матрица-наполнитель	Плотность, кг/м3	модуль упругости, ГПа	Количество отвердителя, м.%	Содержание эпоксидных групп,%_2	Температура вспышки, С_2	Поверхностная плотность, г/м2	Модуль упругости при растяжении, ГПа	Прочность при растяжении, МПа	Потребление смолы, г/м2	Угол нашивки, град	Шаг нашивки	Плотность нашивки
0	1.857143	2030.0	738.736842	30.00	22.267857	100.000000	210.0	70.0	3000.0	220.0	0	4.0	57.0
1	1.857143	2030.0	738.736842	50.00	23.750000	284.615385	210.0	70.0	3000.0	220.0	0	4.0	60.0
2	1.857143	2030.0	738.736842	49.90	33.000000	284.615385	210.0	70.0	3000.0	220.0	0	4.0	70.0
3	1.857143	2030.0	738.736842	129.00	21.250000	300.000000	210.0	70.0	3000.0	220.0	0	5.0	47.0
4	2.771331	2030.0	753.000000	111.86	22.267857	284.615385	210.0	70.0	3000.0	220.0	0	5.0	57.0

```
X_set.shape
```

```
(1023, 13)
```

```
X_set.describe().T
```

	count	mean	std	min	25%	50%	75%	max
Соотношение матрица-наполнитель	1023.0	2.930366	0.913222	0.389403	2.317887	2.906878	3.552660	5.591742
Плотность, кг/м3	1023.0	1975.734888	73.729231	1731.764635	1924.155467	1977.621657	2021.374375	2207.773481
модуль упругости, ГПа	1023.0	739.923233	330.231581	2.436909	500.047452	739.664328	961.812526	1911.536477
Количество отвердителя, м.%	1023.0	110.570769	28.295911	17.740275	92.443497	110.564840	129.730366	198.953207
Содержание эпоксидных групп,%_2	1023.0	22.244390	2.406301	14.254985	20.608034	22.230744	23.961934	33.000000
Температура вспышки, С_2	1023.0	285.882151	40.943260	100.000000	259.066528	285.896812	313.002106	413.273418
Поверхностная плотность, г/м2	1023.0	482.731833	281.314690	0.603740	266.816645	451.864365	693.225017	1399.542362
Модуль упругости при растяжении, ГПа	1023.0	73.328571	3.118983	64.054061	71.245018	73.268805	75.356612	82.682051
Прочность при растяжении, МПа	1023.0	2466.922843	485.628006	1036.856605	2135.850448	2459.524526	2767.193119	3848.436732
Потребление смолы, г/м2	1023.0	218.423144	59.735931	33.803026	179.627520	219.198882	257.481724	414.590628
Угол нашивки, град	1023.0	44.252199	45.015793	0.000000	0.000000	0.000000	90.000000	90.000000
Шаг нашивки	1023.0	6.899222	2.563467	0.000000	5.080033	6.916144	8.586293	14.440522
Плотность нашивки	1023.0	57.153929	12.350969	0.000000	49.799212	57.341920	64.944961	103.988901

```
X_set.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1023 entries, 0 to 1022
Data columns (total 13 columns):
 #   Column                                 Non-Null Count  Dtype
---  ------                                 --------------  -----
 0   Соотношение матрица-наполнитель        1023 non-null   float64
 1   Плотность, кг/м3                       1023 non-null   float64
 2   модуль упругости, ГПа                  1023 non-null   float64
 3   Количество отвердителя, м.%            1023 non-null   float64
 4   Содержание эпоксидных групп,%_2        1023 non-null   float64
 5   Температура вспышки, С_2               1023 non-null   float64
 6   Поверхностная плотность, г/м2          1023 non-null   float64
 7   Модуль упругости при растяжении, ГПа   1023 non-null   float64
 8   Прочность при растяжении, МПа          1023 non-null   float64
 9   Потребление смолы, г/м2                1023 non-null   float64
 10  Угол нашивки, град                     1023 non-null   int64
 11  Шаг нашивки                            1023 non-null   float64
 12  Плотность нашивки                      1023 non-null   float64
dtypes: float64(12), int64(1)
memory usage: 111.9 KB
```

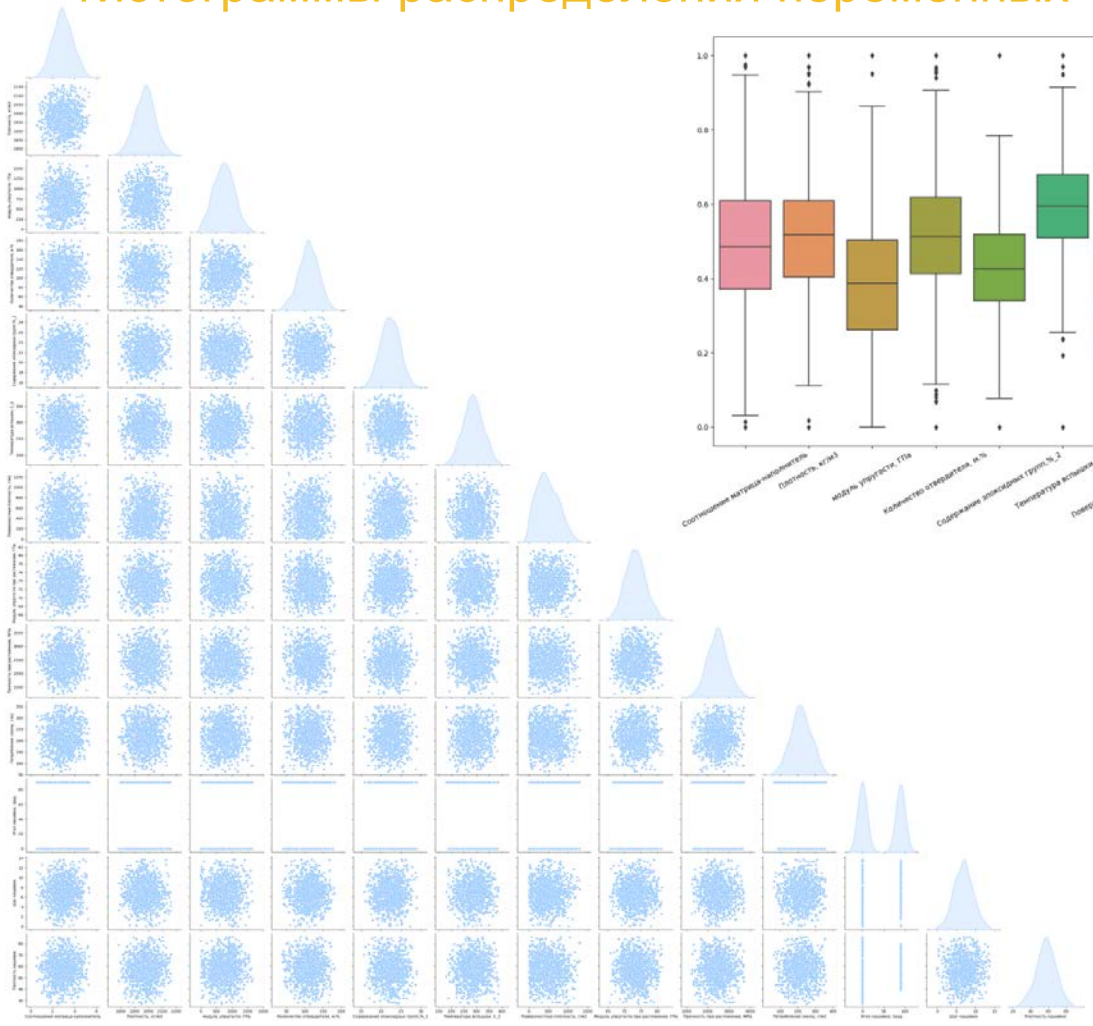
```
#количество уникальных значений
X_set.nunique()
```

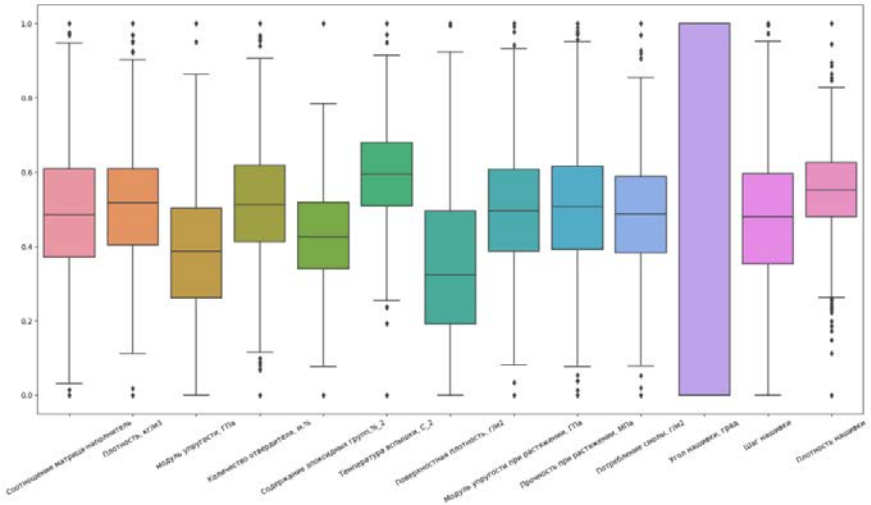
```
Соотношение матрица-наполнитель          1014
Плотность, кг/м3                         1013
модуль упругости, ГПа                    1020
Количество отвердителя, м.%              1005
Содержание эпоксидных групп,%_2          1004
Температура вспышки, С_2                 1003
Поверхностная плотность, г/м2            1004
Модуль упругости при растяжении, ГПа     1004
Прочность при растяжении, МПа            1004
Потребление смолы, г/м2                  1003
Угол нашивки, град                          2
Шаг нашивки                               989
Плотность нашивки                         988
dtype: int64
```

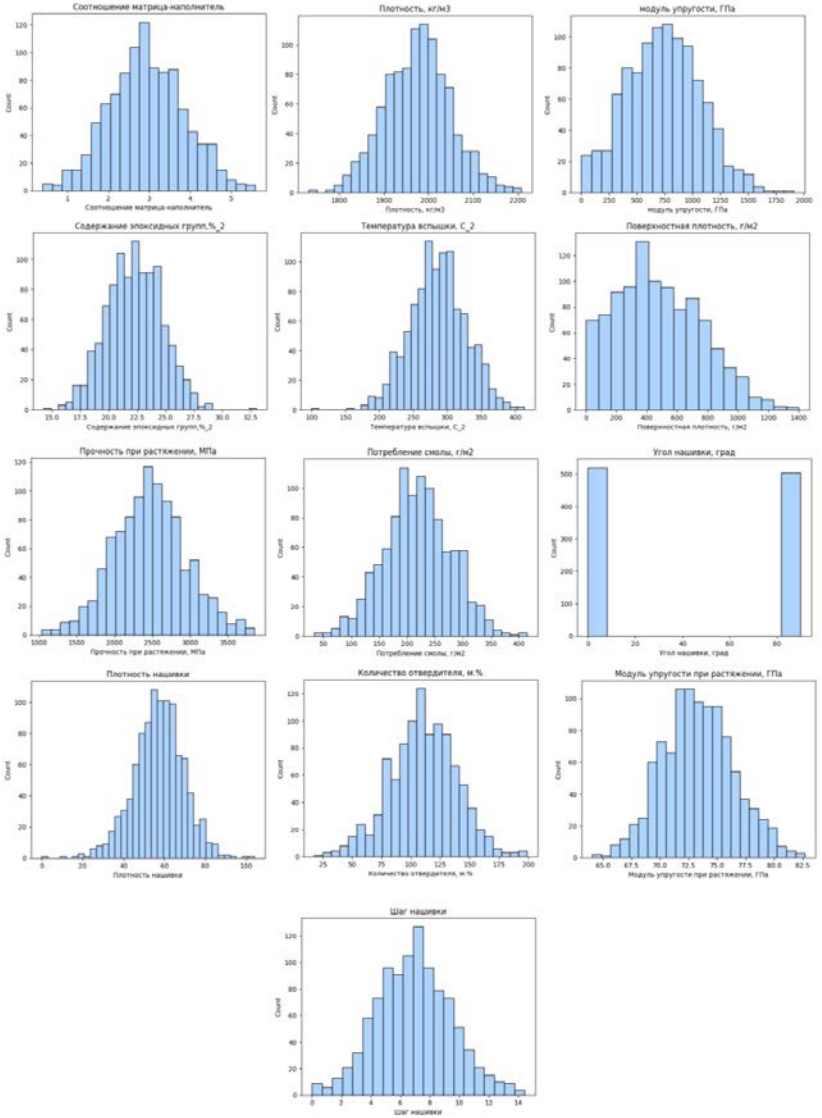
ЦЕНТР ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ
МГТУ им. Н.Э. Баумана

Визуализация данных

попарные графики рассеивания точек, ящики с усами и гистограммы распределения переменных

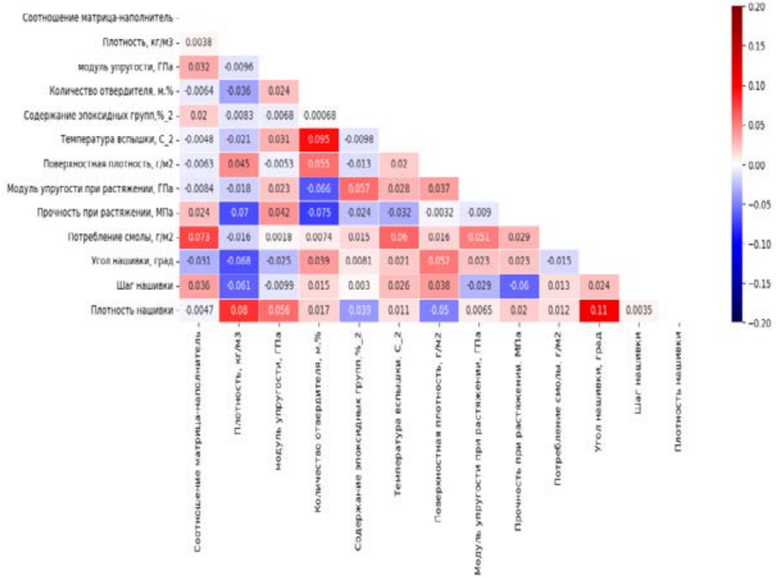


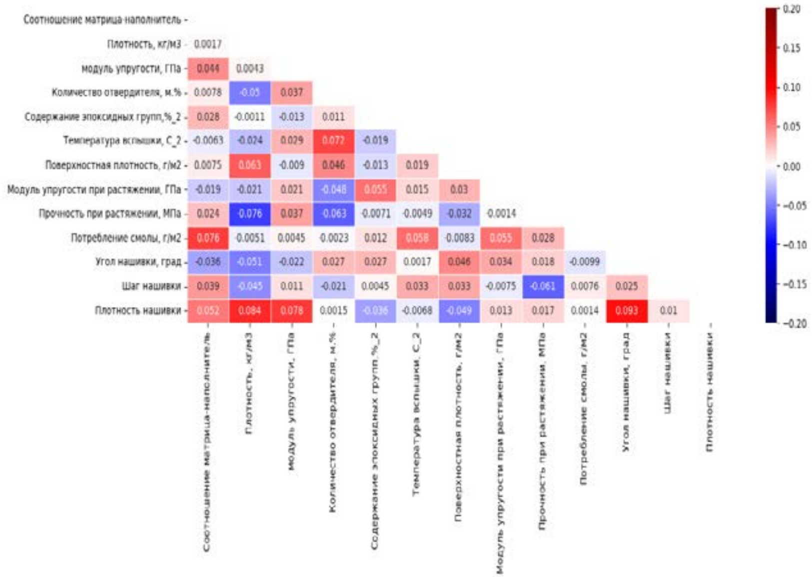


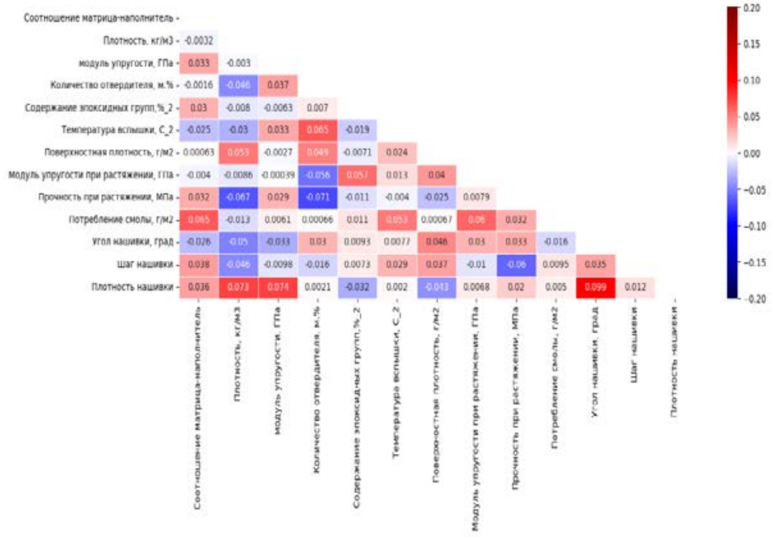


Тепловая карта корреляции

Карты корреляции исходного датасета; датасета, очищенного от выбросов; датасета, в котором выбросы заменены на средние значения







ЦЕНТР ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ
МГТУ им. Н.Э. Баумана

Определение модуля упругости при растяжении

Использовались модели линейной регрессии, случайного леса и метода K- ближайших соседей.

```
[37]  inputs_1 = ['Количество отвердителя, м.%', 'Содержание эпоксидных групп,%_2', 'Потребление смолы, г/м2']
      output_1 = ['Модуль упругости при растяжении, ГПа']

      X_1 = df_minmax[inputs_1]
      y_1 = df[output_1]
```

```
[38]  X_train1, X_test1, y_train1, y_test1 = train_test_split(X_1, y_1, train_size=0.3, random_state=1)
      linear_model1 = LinearRegression()
      linear_model1.fit(X_train1, y_train1)
      y_pred_linear1 = linear_model1.predict(X_test1)
```

```
[41]  r_forest1 = RandomForestRegressor(n_estimators=15, max_depth=7, random_state=1)
      r_forest1.fit(X_train1, np.ravel(y_train1))
      y_pred_forest1 = r_forest1.predict(X_test1)
```

```
[44]  kn1 = KNeighborsRegressor(n_neighbors=50)
      kn1.fit(X_train1, y_train1)
      y_pred_kn1 = kn1.predict(X_test1)
```

```
accuracy1 = pd.DataFrame({'Model': ['LinearRegression', 'RandomForestRegressor', 'KNeighborsRegressor'],
                          'MAE': [MAE_linear1, MAE_forest1, MAE_kn1],
                          'MSE': [MSE_linear1, MSE_forest1, MSE_kn1],
                          'R2': [R2_linear1, R2_forest1, R2_kn1],
                          'error': [ep_linear1, ep_forest1, ep_kn1],
                          'accuracy': [ac_linear1, ac_forest1, ac_kn1]})
accuracy1
```

	Model	MAE	MSE	R2	error	accuracy
0	LinearRegression	2.504425	9.603236	-0.028179	3.420074	96.579926
1	RandomForestRegressor	2.566119	10.393928	-0.112835	3.504324	96.495676
2	KNeighborsRegressor	2.489067	9.493850	-0.016468	3.399100	96.600900

ЦЕНТР ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ
МГТУ им. Н.Э. Баумана

Определение прочности при растяжении

Использовались модели линейной регрессии, случайного леса и метода K- ближайших соседей.

```
inputs_2 = ['Плотность, кг/м3', 'Количество отвердителя, м.%', 'Шаг нашивки', 'Угол нашивки, град', 'Потребление смолы, г/м2', 'Соотношение матрица-наполнитель' ]
output_2 = ['Прочность при растяжении, МПа']

X_2 = df_minmax[inputs_2]
y_2 = df[output_2]
```

```
[49]  X_train2, X_test2, y_train2, y_test2 = train_test_split(X_2, y_2, train_size=0.3, random_state=1)
      linear_model2 = LinearRegression()
      linear_model2.fit(X_train2, y_train2)
      y_pred_linear2 = linear_model2.predict(X_test2)
```

```
[52]  r_forest2 = RandomForestRegressor(n_estimators=15, max_depth=7, random_state=33)
      r_forest2.fit(X_train2, np.ravel(y_train2))
      y_pred_forest2 = r_forest2.predict(X_test2)
```

```
kn2 = KNeighborsRegressor(n_neighbors=100)
kn2.fit(X_train2, y_train2)
y_pred_kn2 = kn2.predict(X_test2)
```

```
accuracy2 = pd.DataFrame({'Model': ['LinearRegression', 'RandomForestRegressor', 'KNeighborsRegressor'],
                          'MAE': [MAE_linear2, MAE_forest2, MAE_kn2],
                          'MSE': [MSE_linear2, MSE_forest2, MSE_kn2],
                          'R2': [R2_linear2, R2_forest2, R2_kn2],
                          'error': [ep_linear2, ep_forest2, ep_kn2],
                          'accuracy': [ac_linear2, ac_forest2, ac_kn2]})
accuracy2
```

	Model	MAE	MSE	R2	error	accuracy
0	LinearRegression	374.136895	222258.062113	-0.006573	15.239563	84.760437
1	RandomForestRegressor	391.188940	245345.126459	-0.111130	15.934137	84.065863
2	KNeighborsRegressor	373.442818	222241.455938	-0.006497	15.211292	84.788708

ЦЕНТР ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ
МГТУ им. Н.Э. Баумана

Нейронная сеть для соотношения матрица - наполнитель

Модель нейронной сети и график потерь

```
model = tf.keras.models.Sequential()
model.add(layers.Dense(128, input_dim=X_3.shape[1], activation='relu'))
model.add(layers.Dropout(0.12))
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dropout(0.12))
model.add(layers.Dense(1))
model.add(layers.Dense(64, activation='tanh'))
model.summary()
```

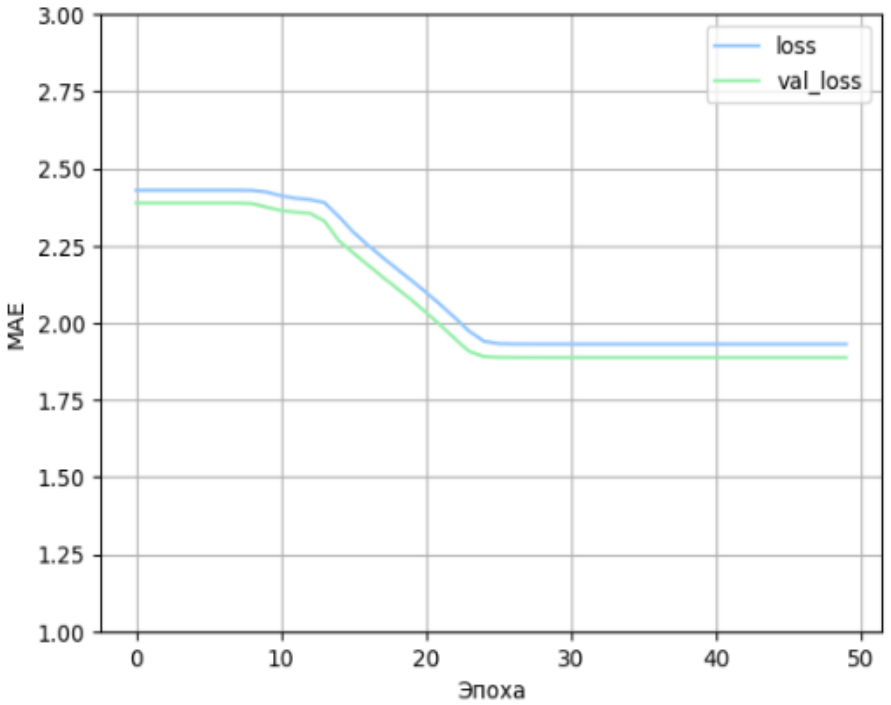
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 128)               1664

 dropout (Dropout)           (None, 128)               0

 dense_1 (Dense)             (None, 64)                8256

 dropout_1 (Dropout)         (None, 64)                0

 dense_2 (Dense)             (None, 1)                 65

 dense_3 (Dense)             (None, 64)                128

=================================================================
Total params: 10,113
Trainable params: 10,113
Non-trainable params: 0
_________________________________________________________________
```

```
[61]  earlystop_ns = EarlyStopping(monitor='mae', patience=10, mode='auto')
```

```
[62]  dfmodel = model.compile(optimizer='adam', loss='mae', metrics=['mae'])
      history = model.fit(
          X_train3,
          y_train3,
          validation_split=0.2,
          batch_size = 64,
          verbose=1, epochs=50
          )
```

```
def plot_loss(history):
    plt.plot(history.history['loss'], label='loss')
    plt.plot(history.history['val_loss'], label='val_loss')
    plt.ylim([1, 3])
    plt.xlabel('Эпоха')
    plt.ylabel('MAE')
    plt.legend()
    plt.grid(True)
```

```
plot_loss(history)
```



ЦЕНТР ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ
МГТУ им. Н.Э. Баумана

Приложение

```
def input_variable():

  x1 = float(input('Введите значение переменной Количество отвердителя, м.%: '))
  x2 = float(input('Введите значение переменной Содержание эпоксидных групп,%_2: '))
  x3 = float(input('Введите значение переменной Потребление смолы, г/м2: '))

  return x1,x2,x3

def input_proc(X):
  print('Вызов модели')
  res = model_1.predict(X)
  return res

def app_model():
  job_x = load('/content/drive/MyDrive/Colab Notebooks/VKR/X_1.joblib')
  job_y = load('/content/drive/MyDrive/Colab Notebooks/VKR/y_1.joblib')
  model_1 = load('/content/drive/MyDrive/Colab Notebooks/VKR/filename.joblib')
  print('Приложение прогнозирует значения модуля упругости при растяжении')
  for i in range(110):
    try:
      print('введите 1 для прогноза, 2 для выхода')
      check = input()

      if check == '1':
        print('Введите данные')
        X = input_variable()
        X = job_x.transform(np.array(X_1).reshape(1,-1))
        print(['Модуль упругости при растяжении, ГПа'])
        print(job_y.inverse_transform(input_proc(X_1)))


      elif check == '2':
        break
      else:
        print('Повторите выбор')
    except:
      print('Неверные данные. Повторите операцию')
app_model()
```

- Для прогнозирования значения модуля упругости при растяжении на основании модели линейной регрессии приложение просит поэтапно ввести значения параметров Количество отвердителя, Содержание эпоксидных групп и Потребление смолы, на основании которых выдает итоговое значение.

```
Приложение прогнозирует значения модуля упругости при растяжении
введите 1 для прогноза, 2 для выхода
1
Введите данные
Введите значение переменной Количество отвердителя, м.%: 0.5
Введите значение переменной Содержание эпоксидных групп,%_2: 0.5
Введите значение переменной Потребление смолы, г/м2: 0.5
Неверные данные. Повторите операцию
введите 1 для прогноза, 2 для выхода
```

ЦЕНТР
ДОПОЛНИТЕЛЬНОГО
ОБРАЗОВАНИЯ
МГТУ им. Н.Э. Баумана

do.bmstu.ru