ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР *МГТУ им. Н. Э. Баумана*

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

по курсу «Data Science»

Тема: прогнозирование конечных свойств новых материалов (композиционных материалов)

Н.А. Ядов

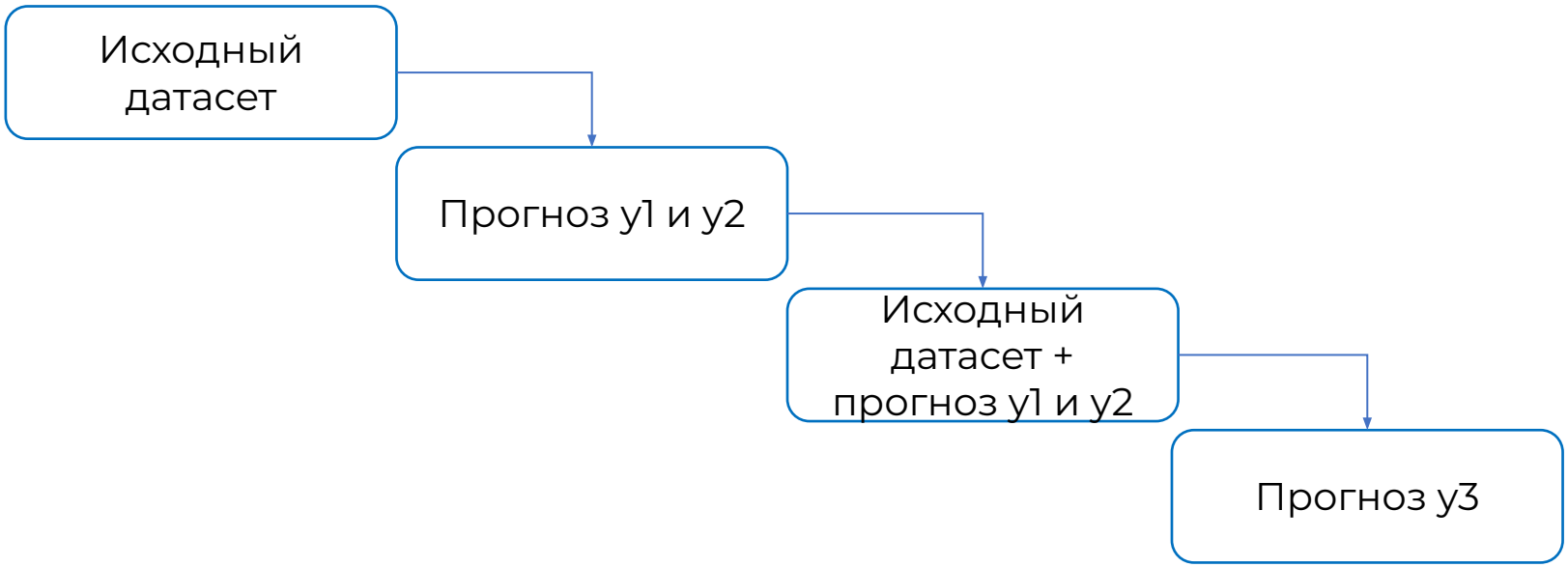
Цели и задачи работы

Прогнозирование ряда конечных свойств композиционных материалов:

- модуль упругости при растяжении – y1;
- прочность при растяжении – y2;
- соотношение матрица-наполнитель – y3.

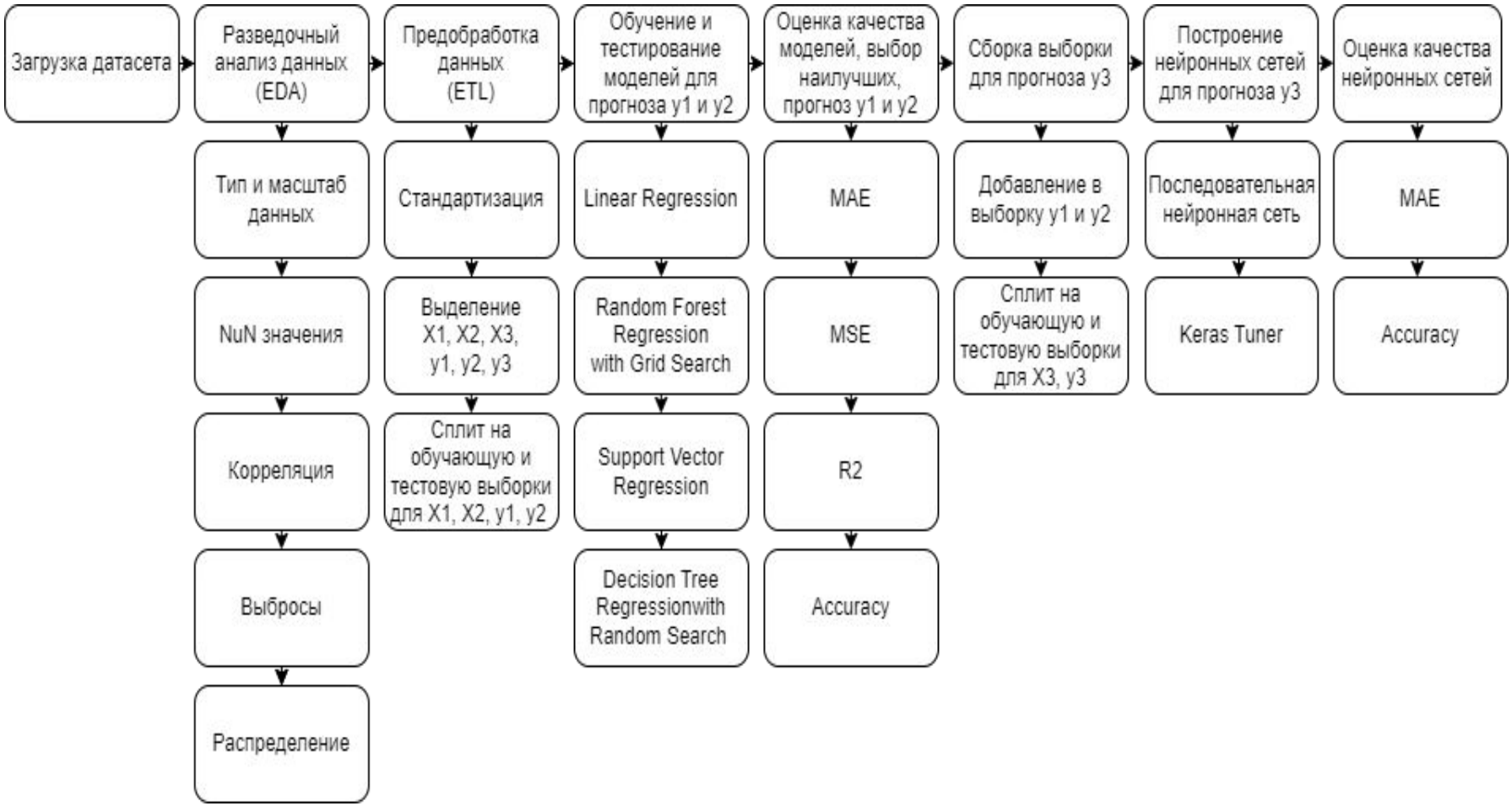
ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Цели и задачи работы



ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Pipline



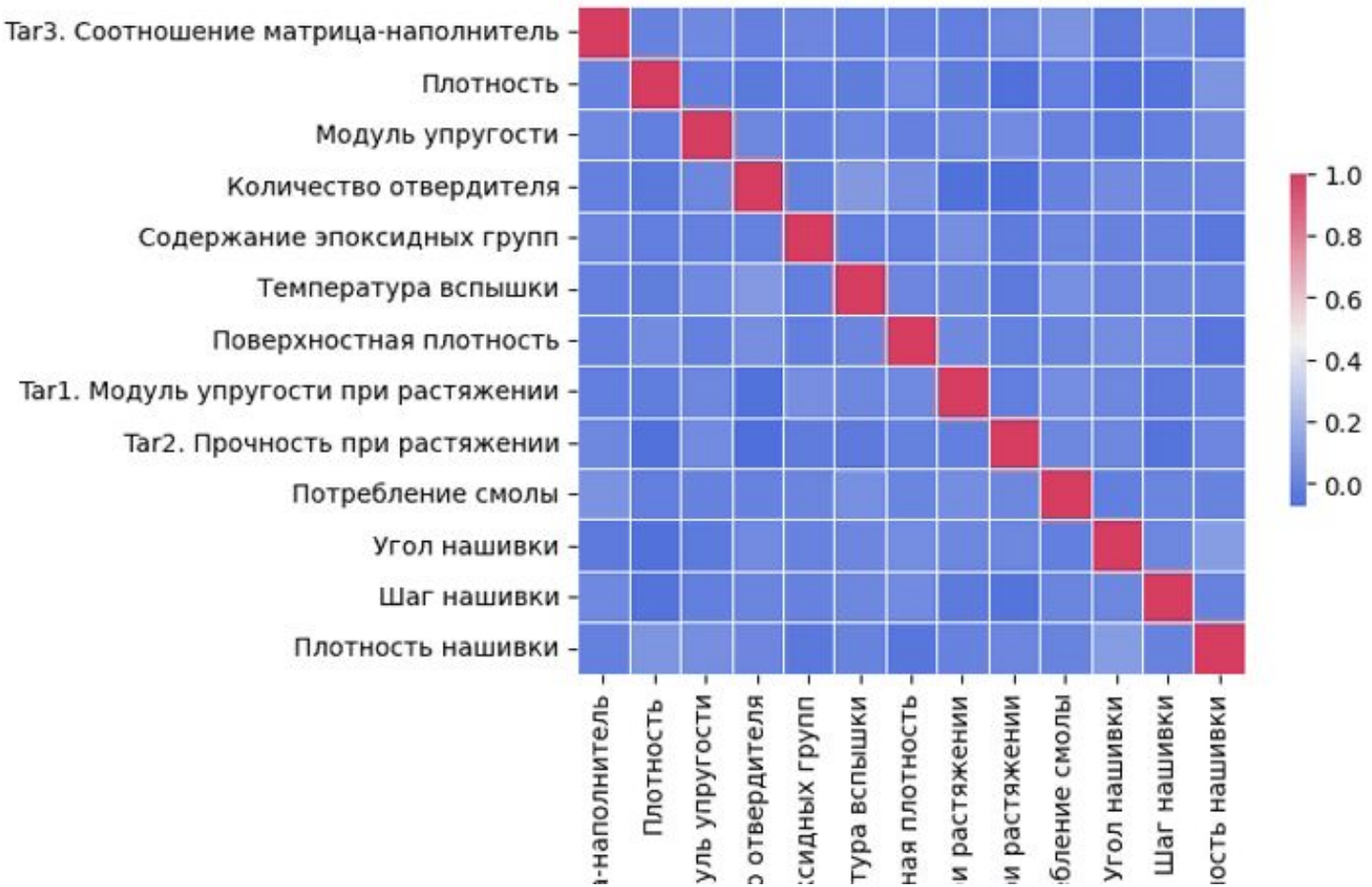
ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Exploratory Data Analysis

- отсутствуют пустые NuN значения;
- int64 и float64;
- отсутствие значимых корреляций;
- отсутствие выбросов;
- отсутствие значимых смещений в распределении.

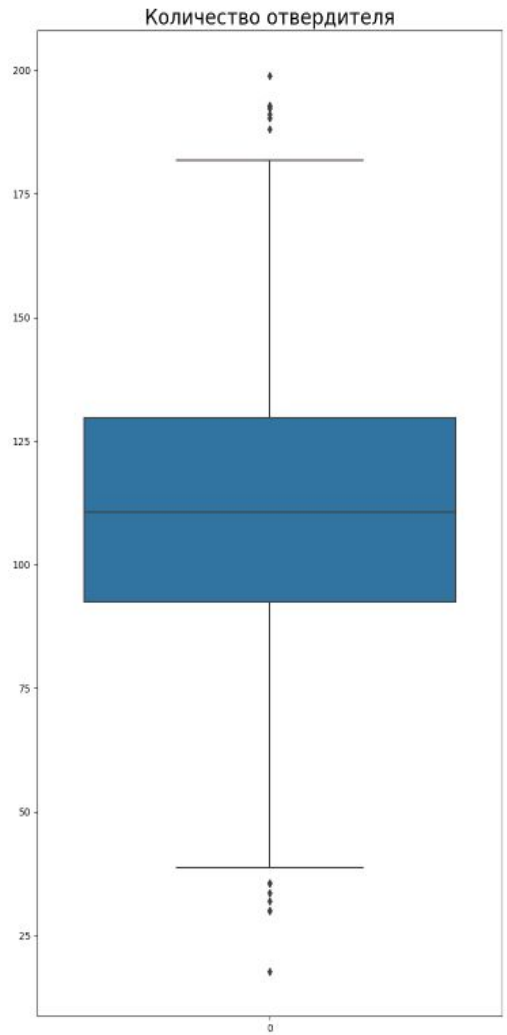
ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

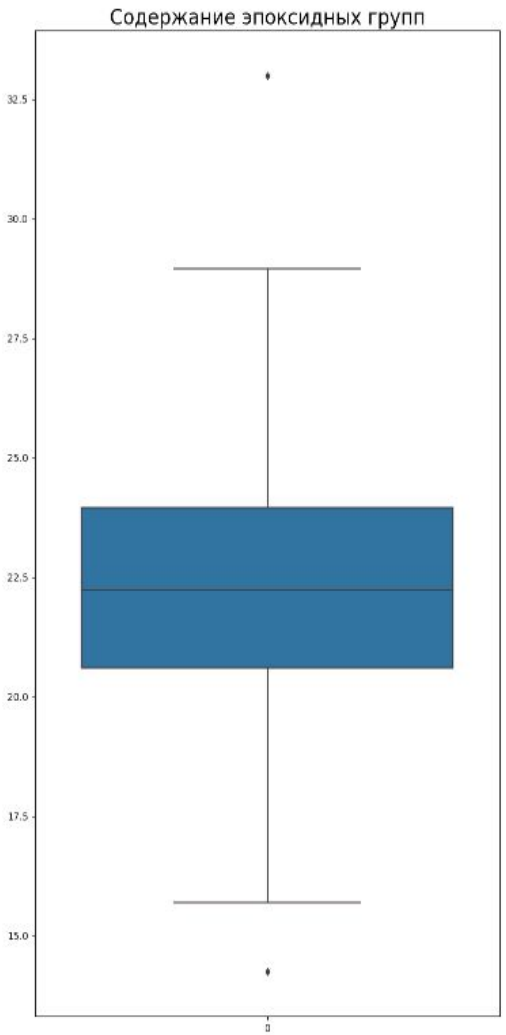
Матрица корреляций

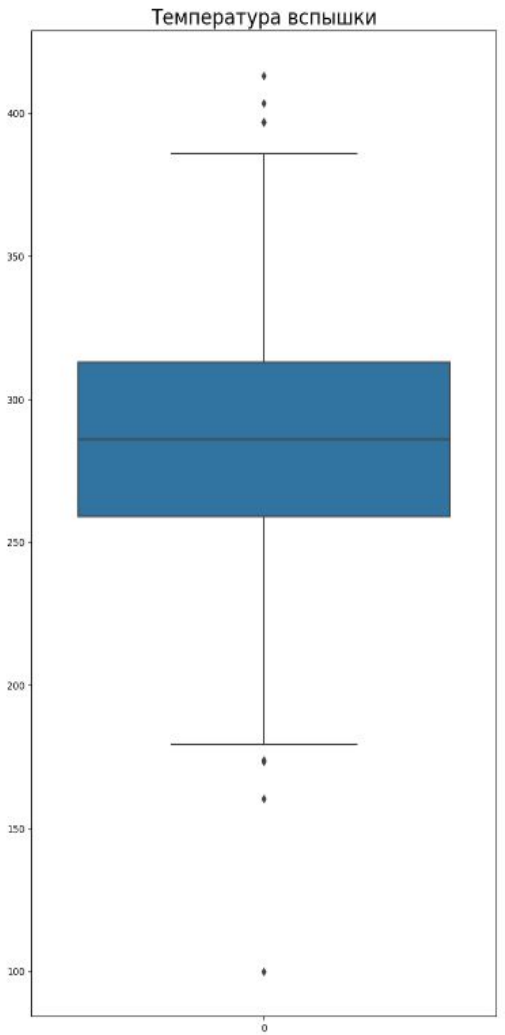


ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Анализ выбросов boxplot, ящик с усами

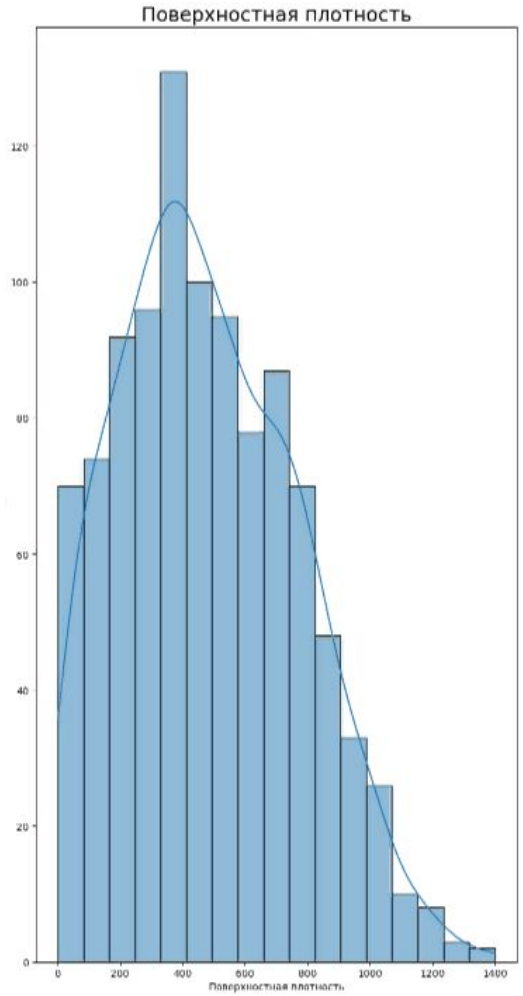


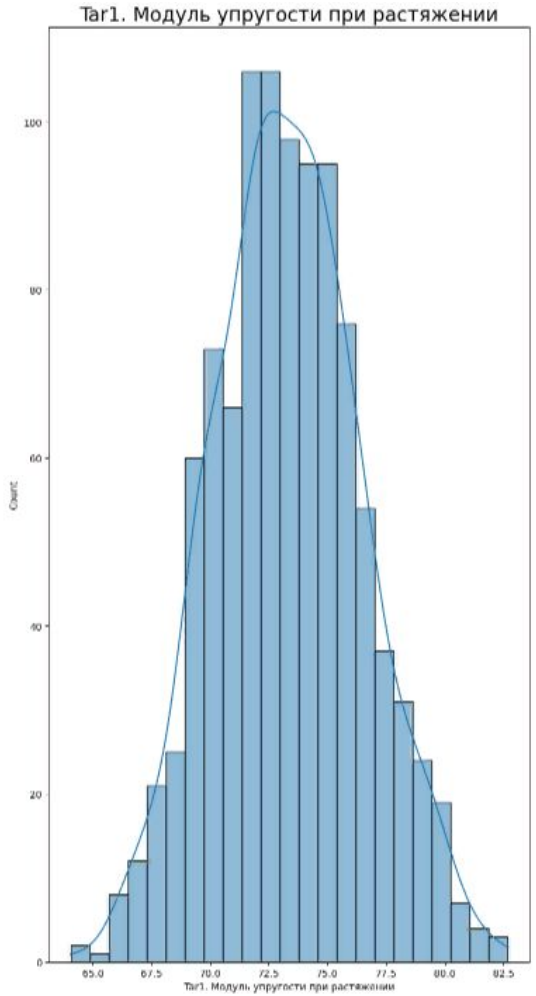


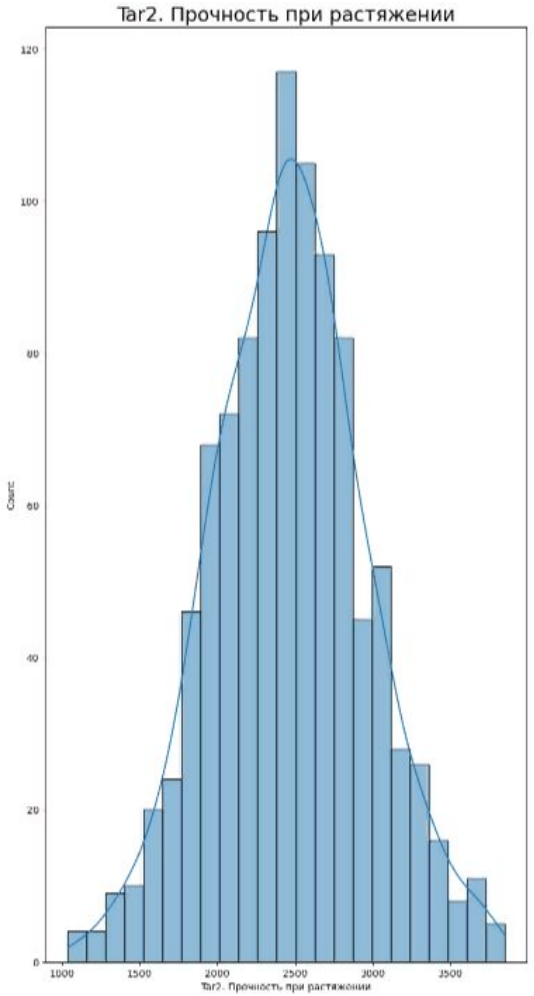


ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Анализ распределения







ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Extract, Transformer, Load

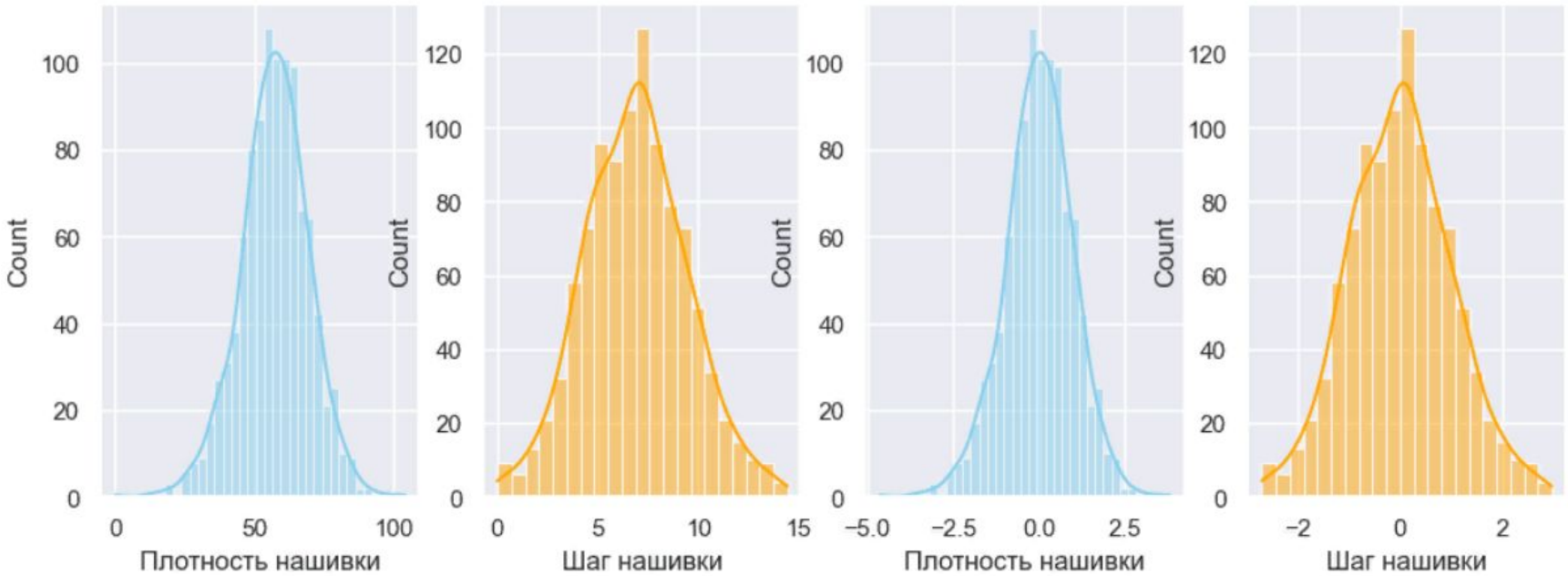
- разделение на тестовые и обучающие выборки методом train_test_split

```
y1 = set_split['Tar1. Модуль упругости при растяжении']
y2 = set_split['Tar2. Прочность при растяжении']
y3 = set_split['Tar3. Соотношение матрица-наполнитель']

X1 = set_split.drop(['Tar3. Соотношение матрица-наполнитель', 'Tar1. Модуль упругости при растяжении'], axis = 1)
X2 = set_split.drop(['Tar3. Соотношение матрица-наполнитель', 'Tar2. Прочность при растяжении'], axis = 1)
X3 = set_split.drop(['Tar1. Модуль упругости при растяжении','Tar2. Прочность при растяжении',
                     'Tar3. Соотношение матрица-наполнитель'], axis = 1)
```

```
X1_train, X1_test, y1_train, y1_test = train_test_split(X1, y1, test_size=0.3, shuffle = True, random_state=42)
X2_train, X2_test, y2_train, y2_test = train_test_split(X2, y2, test_size=0.3, shuffle = True, random_state=42)
```

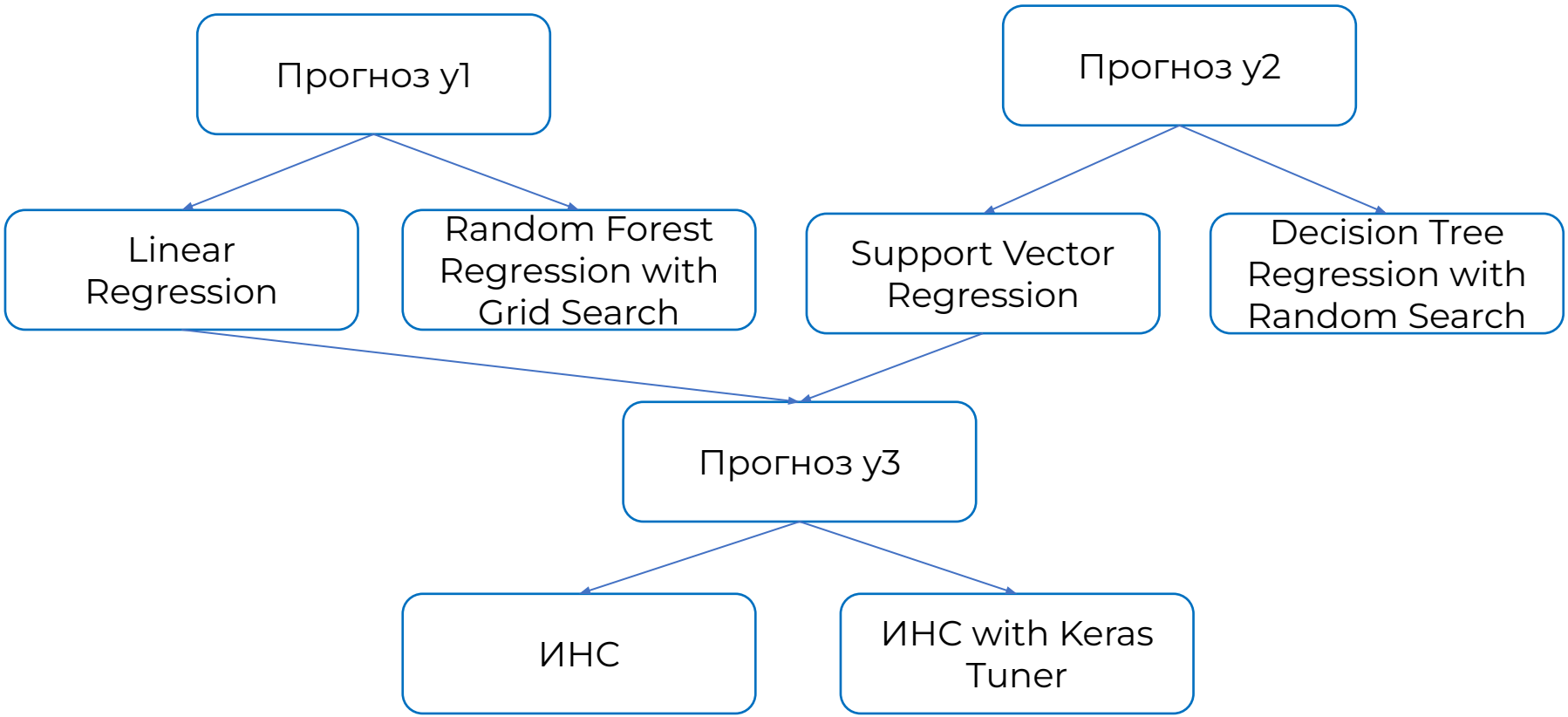
Extract, Transformer, Load

- масштабирование StandartScaler



ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Прогнозирование. Модели и ИНС



ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Прогнозирование. Модели

Target	Model	MAE	MSE	R2	accuracy
Tar1. Модуль упругости при растяжении	Linear Regression	2.557104	10.118841	-0.018564	96.526208
Tar1. Модуль упругости при растяжении	RandomForest Regression with Grid Search	2.565373	10.149580	-0.021658	96.514974
Tar2. Прочность при растяжении	Support Vector Regression	379.490730	224363.417124	-569632.206020	84.651504
Tar2. Прочность при растяжении	TDecision Tree Regressor with Random Search	380.952804	226299.789139	-577680.645878	84.592370

```
MAE_train: 2.4613192952688316
MAE_test: 2.5571035925102468
MSE_train: 9.379429049269936
MSE_test: 10.118841188115416
r2_train: 0.02063112061922867
r2_test: -0.018563613121253653
Точность модели на трейне: 96.63787934635941
Точность модели на тесте: 96.52620775333057
```


ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Прогнозирование. ИНС

```
model_ns = tf.keras.models.Sequential()
model_ns.add(tf.keras.Input(shape=(12,)))
model_ns.add(Dense(1, input_dim=12, activation='linear'))

model_ns.compile(
    optimizer=tf.keras.optimizers.Adam(0.001),
    loss='mean_absolute_error')
```

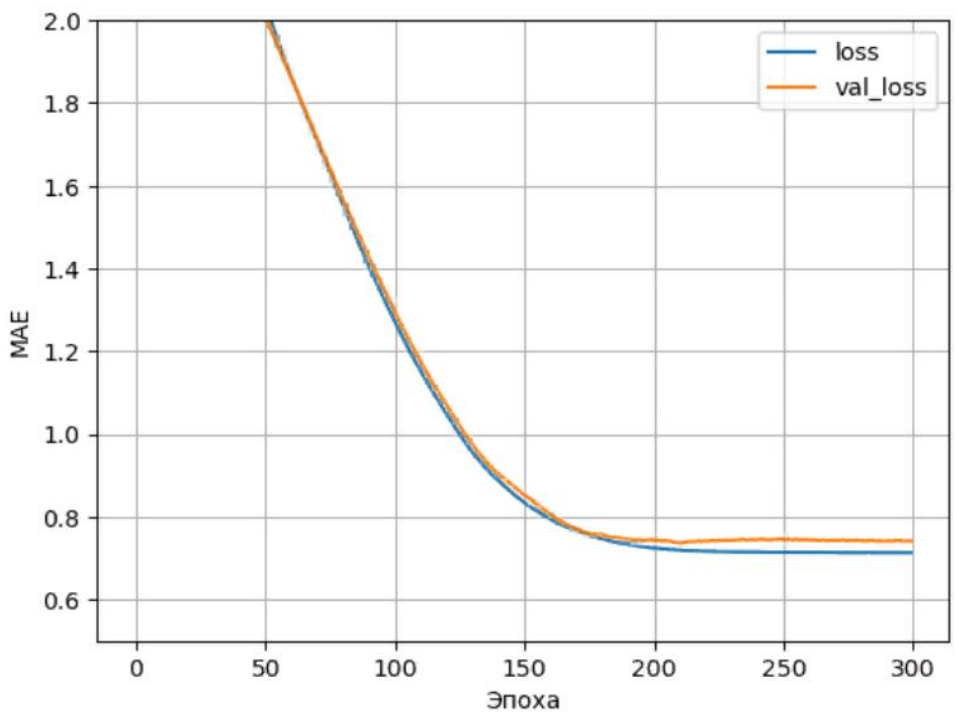
```
model_ns.summary()
```

```
Model: "sequential_14"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_16 (Dense)            (None, 1)                 13

=================================================================
Total params: 13
Trainable params: 13
Non-trainable params: 0
```


ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Прогнозирование. ИНС

```
history_ns = model_ns.fit(
    X3_train,
    y3_train,
    epochs=300,
    verbose=0,
    validation_split = 0.2)
```




```
mae_ns = model_ns.evaluate(X3_test, y3_test, verbose=0)
print('MAE:', mae_ns)
print('Точность модели:', 100- (mae_ns / y3_test.mean() * 100))
```

```
MAE: 0.7650960087776184
Точность модели: 74.07735800719789
```

```
mae_ns = model_ns.evaluate(X3_test, y3_test, verbose=0)
print('MAE:', mae_ns)
print('Точность модели:', 100- (mae_ns / y3_test.mean() * 100))
```

```
MAE: 0.7639831304550171
Точность модели: 74.11506405455322
```


ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Прогнозирование. ИНС with Keras Tuner

```
def tuner_model(hp):
    model = Sequential()

    activation_choice = hp.Choice('activation', values=['relu', 'sigmoid', 'tanh', 'elu', 'selu'])

# сетка параметров входного слоя

    model.add(Dense(units=hp.Int('units_input',
                                 min_value=12,
                                 max_value=36,
                                 step=2),
                                 input_dim = X3_train.shape[1], #input_shape=(X3_train.shape[1])
                                 activation = activation_choice))

# сетка кол-ва слоев и их параметров:

    for i in range(hp.Int('num_layers', 2, 5)):
        model.add(layers.Dense(units=hp.Int('units_' + str(i),
                                            min_value=32,
                                            max_value=128,
                                            step=32),
                                            activation = activation_choice))

    model.add(Dense(1, activation='linear'))

    model.compile(
        optimizer=hp.Choice('optimizer', values=['adam','rmsprop','SGD']),
        loss='mean_absolute_error',
        metrics=['mean_absolute_error'])

    return model
```


ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Прогнозирование. ИНС with Keras Tuner

```
def tuner_model(hp):
    model = Sequential()

    activation_choice = hp.Choice('activation', values=['relu', 'sigmoid', 'tanh', 'elu', 'selu'])

# сетка параметров входного слоя

    model.add(Dense(units=hp.Int('units_input',
                                 min_value=12,
                                 max_value=36,
                                 step=2),
                                 input_dim = X3_train.shape[1], #input_shape=(X3_train.shape[1])
                                 activation = activation_choice))

# сетка кол-ва слоев и их параметров:

    for i in range(hp.Int('num_layers', 2, 5)):
        model.add(layers.Dense(units=hp.Int('units_' + str(i),
                                            min_value=32,
                                            max_value=128,
                                            step=32),
                                            activation = activation_choice))

    model.add(Dense(1, activation='linear'))

    model.compile(
        optimizer=hp.Choice('optimizer', values=['adam','rmsprop','SGD']),
        loss='mean_absolute_error',
        metrics=['mean_absolute_error'])

    return model
```


ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Прогнозирование. ИНС with Keras Tuner

```
best_model.summary() #архитектура лучшей модели
```

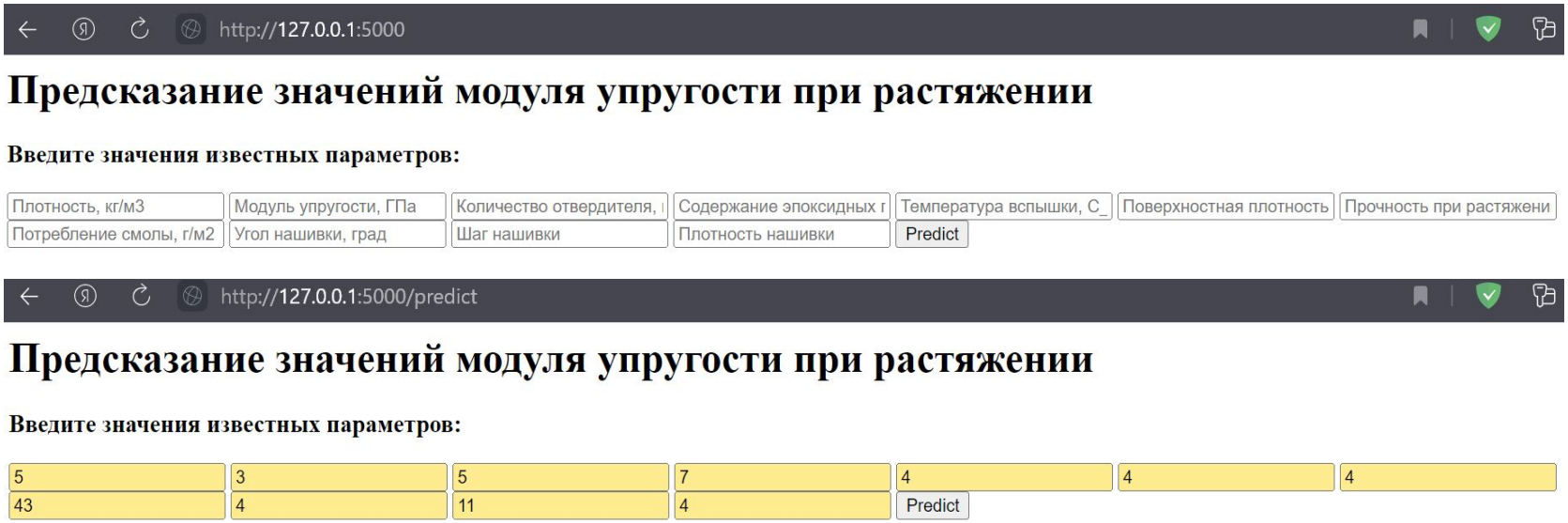
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 24)                312

 dense_1 (Dense)             (None, 32)                800

 dense_2 (Dense)             (None, 96)                3168

 dense_3 (Dense)             (None, 96)                9312

 dense_4 (Dense)             (None, 32)                3104

 dense_5 (Dense)             (None, 1)                 33

=================================================================
Total params: 16,729
Trainable params: 16,729
Non-trainable params: 0
```

```
best_model = tuner.get_best_models(num_models=1)[0] #полуаем лучшую модель
loss, mae = best_model.evaluate(X3_test, y3_test) #метрики лучшей модели
print('Точность сети:', 100 - (mae / y3_test.mean() * 100))
```

```
10/10 [==============================] - 0s 3ms/step - loss: 0.7552 - mean_absolute_error: 0.7552
Точность сети: 74.41249370205179
```


ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Flask-приложение




```
test = np.array([5, 3, 5, 7, 4, 4, 4, 43, 4, 11, 4])
test = test.reshape(1, -1)
test_pred = model.predict(test)
test_pred
```

```
array([76.84437407])
```


ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана

Спасибо за внимание!

ОБРАЗОВАТЕЛЬНЫЙ ЦЕНТР МГТУ им. Н. Э. Баумана