```
In [ ]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
       from sklearn.datasets import make_classification
        x_data_generated, y_data_generated = make_classification(scale=1, random_state=42)
        x_data_generated.shape, y_data_generated.shape
Out[]: ((100, 20), (100,))
In [ ]: from sklearn.model_selection import cross_val_score
        from sklearn.linear_model import LogisticRegression
        def cvs_score(x_data, y_data, model=None):
           if model is None:
               model = LogisticRegression()
            return cross_val_score(model, x_data, y_data, scoring='accuracy').mean()
        cvs_score(x_data_generated, y_data_generated)
Out[]: 0.9800000000000001
In [ ]: from sklearn.feature_selection import VarianceThreshold
        selector = VarianceThreshold(threshold=0.9)
        x_data_generated = selector.fit_transform(x_data_generated)
        x_data_generated.shape
Out[]: (100, 14)
In [ ]: plt.figure(figsize=(12, 10))
        stacked_data = np.hstack((x_data_generated, y_data_generated.reshape(-1, 1)))
        corr_matrix = np.corrcoef(stacked_data.T)
        sns.heatmap(corr_matrix, square=True, annot=True, fmt='.2f', cmap='coolwarm', center=0)
        plt.show()
           1.00 0.07 -0.04 -0.00 0.04 -0.07 -0.11 -0.10 -0.06 0.02 -0.12 0.08 -0.10 -0.07 -0.09
       - - 0.07 1.00 -0.10 -0.14 -0.62 -0.20 0.05 -0.96 0.03 -0.02 -0.03 0.08 -0.04 -0.04 -0.76
                                                                                                                        - 0.75
       ~ -0.04 -0.10 1.00 0.03 0.14 0.06 0.08 0.07 -0.02 0.00 0.05 -0.03 -0.04 0.00 0.09
       m - -0.00 -0.14 0.03 1.00 0.10 0.20 -0.07 0.13 0.00 -0.07 0.01 -0.04 -0.21 0.04 0.12
                                                                                                                        - 0.50
                                     1.00 0.17 -0.09 0.36 -0.12 0.07 0.04 -0.03 -0.02 0.00 0.11
       → - 0.04 -0.62 0.14 0.10
       பு - -0.07 -0.20 0.06 0.20 0.17 1.00 -0.05 0.18 -0.13 -0.10 0.03 0.09 -0.08 -0.04 0.20
                                                                                                                        - 0.25
       σ - -0.11 0.05 0.08 -0.07 -0.09 -0.05 1.00 -0.02 0.12 0.15 0.18 0.11 0.05 -0.13 -0.02
       ► - -0.10 -0.96 0.07 0.13 0.36 0.18 -0.02 1.00 0.02 -0.01 0.02 -0.09 0.05 0.04 0.86
                                                                                                                        - 0.00
       \infty - -0.06 0.03 -0.02 0.00 -0.12 -0.13 0.12 0.02 1.00 -0.01 0.08 -0.09 0.07 0.16 0.06
       o - 0.02 -0.02 0.00 -0.07 0.07 -0.10 0.15 -0.01 -0.01 1.00 0.01 0.18 -0.01 -0.10 0.02
                                                                                                                        - -0.25
       9 - -0.12 -0.03 0.05 0.01 0.04 0.03 0.18 0.02 0.08 0.01 1.00 0.08 0.07 -0.01 0.08
       - -0.50
       <u>∽</u> - -0.10 -0.04 -0.04 -0.21 -0.02 -0.08 0.05 0.05 0.07 -0.01 0.07 -0.03 1.00 -0.02 0.10
       \underline{m} - -0.07 -0.04 0.00 0.04 0.00 -0.04 -0.13 0.04 0.16 -0.10 -0.01 -0.12 -0.02 1.00 -0.06
                                                                                                                         - -0.75
       4 - -0.09
                               0.12
                                     0.11 0.20 -0.02
                                                         0.86
                                                                0.06
                                                                      0.02
                 -0.76
                        0.09
                                                                             0.08
                                                                                  -0.02
                                                                                          0.10
                                                                                                -0.06
                                                                                                       1.00
                          2
                                                                        9
                                                                              10
                                                                                     11
                                                                                           12
                                                                                                  13
                                                                                                        14
             0
In []: x_data_generated = np.delete(x_data_generated, 1, axis=1) # удаляем признак сильно скоррелированный с другими признаками (1)
        x_data_generated.shape
Out[]: (100, 13)
        Повторите п. 2 на отобранных признаках в п. 3a, п. 3b.
In [ ]: cvs_score(x_data_generated, y_data_generated)
Out[]: 0.96
        Осуществите отбор признаков на основе дисперсионного анализа:
       from sklearn.feature_selection import SelectKBest, f_classif
        selector = SelectKBest(f_classif, k=5)
        x_data_generated2 = selector.fit_transform(x_data_generated, y_data_generated)
        x_data_generated2.shape
        cvs_score(x_data_generated2, y_data_generated)
Out[ ]: 1.0
        Отбор с использованием моделей:
        from sklearn.preprocessing import StandardScaler
        from sklearn.linear_model import Lasso
        scaler = StandardScaler()
        x_data_generated_scaled = scaler.fit_transform(x_data_generated)
        model_list = []
        coef_list = []
        x_{\text{test\_data}} = list(2**np.linspace(-10,1,150))
        for a in x_test_data:
           model = Lasso(alpha=a, random_state=42)
           model.fit(x_data_generated_scaled, y_data_generated)
           model_list.append(model)
           coef_list.append(model.coef_)
        colors = {
           0: "red",
           1: "green",
           2: "blue",
           3: "yellow",
           4: "black",
           5: "purple",
           6: "orange",
           7: "brown",
           8: "pink",
           9: "gray",
           10: "cyan",
           11: "magenta",
           12: "lime",
           13: "teal"
        plt.figure(figsize=(10, 7))
        for i, x in enumerate(x_test_data):
           for j in range(len(coef_list[i])):
               plt.plot(x, coef_list[i][j], 'o', color=colors[j], markersize=3, label='Feature {}'.format(j))
        plt.grid()
        handles, labels = plt.gca().get_legend_handles_labels()
        unique_labels = dict.fromkeys(labels)
        handles = [handles[labels.index(label)] for label in unique_labels]
        labels = list(unique_labels)
        plt.legend(handles, labels, loc='best')
        plt.yscale('symlog', linthresh=0.01)
        # plt.xscale('symlog', linthresh=0.01)
        plt.xlim(-0.01, 0.1)
        plt.title('Lasso coefficients')
        plt.show()
                                                        Lasso coefficients
                                                                                                         Feature 0
                                                                                                         Feature 1
                                                                                                         Feature 2
                                                                                                         Feature 3
        10^{-1}
                                                                                                         Feature 4
                                                                                                         Feature 5
                                                                                                         Feature 6
                                                                                                         Feature 7
        10^{-2}
                                                                                                         Feature 8
                                                                                                         Feature 9
                                                                                                         Feature 10
                                                                                                         Feature 11
                                                                                                         Feature 12
            0
       -10^{-2}
       -10^{-1}
                      0.00
                                        0.02
                                                           0.04
                                                                              0.06
                                                                                                0.08
                                                                                                                   0.10
In [ ]: from sklearn.feature_selection import SelectFromModel
        selector = SelectFromModel(Lasso(alpha=0.03, random_state=42))
        x_data_generated3 = selector.fit_transform(x_data_generated, y_data_generated)
        x_data_generated3.shape
        cvs_score(x_data_generated3, y_data_generated)
        0.9800000000000001
In [ ]: from sklearn.ensemble import RandomForestClassifier
        clf = RandomForestClassifier(random_state=42)
        clf.fit(x_data_generated, y_data_generated)
        si = np.argsort(clf.feature_importances_)
        le = np.arange(len(clf.feature_importances_))
        plt.barh(le, clf.feature_importances_[si])
        plt.yticks(ticks=le, labels=[*si])
        plt.show()
        cvs_score(x_data_generated[:, 6:7], y_data_generated)
        6 -
        0 -
        7 -
        3 -
        4 -
       11 -
        9 -
        2 -
        1 -
       10 -
       12 -
        5
        8 -
                               0.2
                                                     0.4
                    0.1
                                          0.3
                                                                0.5
         0.0
Out[]: 0.95
In [ ]: from sklearn.feature_selection import SequentialFeatureSelector
        sfs = SequentialFeatureSelector(clf, n_features_to_select=5, direction="forward")
        x_data_generated4 = sfs.fit_transform(x_data_generated, y_data_generated)
        display(x_data_generated4.shape)
        display(sfs.get_support(indices=True))
        cvs_score(x_data_generated4, y_data_generated)
      (100, 5)
       array([ 1, 2, 3, 6, 12], dtype=int64)
Out[]: 0.99
```