基于机器学习算法预测钙钛矿材料带隙

本节介绍机器学习算法在计算材料学领域应用，主要包含以下内容：

- 材料数据集预处理及数据可视化
- 机器学习模型搭建及评估
- 材料特征工程应用

请用户使用JAMIP机器学习模块前先熟悉Readme.pdf内容，简单了解机器学习整体流程，Input文件夹包含数据集excel、csv格式，Output文件夹包含Script.py脚本输出文件。

注意：脚本在使用时会出现评估指标会出现略微差距，是模型随机性的影响，不改变整体趋势。

JAMIP机器学习模块使用

1.材料数据集预处理及可视化

用户完成高通量计算任务后,可从JAMIP数据库提取所需要材料信息数据,搭建初始数据集,内部可以包括数字型特征和字符串型特征,本示例中构建钙钛矿数据集，其结构中包括0维、1维、2维、3维钙钛矿共696种，特征集考虑元素的离子半径、范德华半径、电子亲和能、电负性和填充系数、晶系等共54种特征。

```
# 载入初始数据集
>>> import pandas as pd
>>> original_df = pd.read_csv('Input/example.csv', index_col = False, usecols =
range(2,57)

# 初始特征集合
>>> original_df.columns
Index(['phase', 'ionic_radius_a', 'atomic_number_b1',
       'atomic_volume_b1', 'atomic_radius_b1', 'atomic_weight_b1',
       'dipole_polarizability_b1', 'electron_affinity_b1', 'fusion_heat_b1',
       'period_b1', 'vdw_radius_b1', 'covalent_radius_cordero_b1', 'ip_1_b1',
       'ip_2_b1', 'ip_3_b1', 'eletronegativity_b1', 'ionic_radius_b1',
       'atomic_number_b2', 'atomic_volume_b2', 'atomic_radius_b2',
       'atomic_weight_b2', 'dipole_polarizability_b2', 'electron_affinity_b2',
       'fusion_heat_b2', 'period_b2', 'vdw_radius_b2',
       'covalent_radius_cordero_b2', 'ip_1_b2', 'ip_2_b2', 'ip_3_b2',
       'eletronegativity_b2', 'ionic_radius_b2', 'atomic_number_x',
       'atomic_volume_x', 'atomic_radius_x', 'atomic_weight_x',
       'dipole_polarizability_x', 'electron_affinity_x', 'fusion_heat_x',
       'period_x', 'vdw_radius_x', 'covalent_radius_cordero_x', 'ip_1_x',
       'ip_2_x', 'ip_3_x', 'eletronegativity_x', 'ionic_radius_x', 'a', 'b',
       'c', 'natoms', 'volume', 'PF', 'bx_mean', 'band_gap'],
      dtype='object')
```

数据清理

检查整体数据集是否存在缺失值，并分析目标性质分布情况

```
>>> from jamip.ml.preprocessing import DataCleaning
>>> dc = DataCleaning(dataset_df = original_df, target = 'band_gap')
>>> dc.precheck_dataframe()
{}      # 数据集无缺失值
```

```
# 检查数字型特征与字符串型特征
>>> dc.number_object_cols().number_feature   # 数字型特征
numberic feature : ['ionic_radius_a', 'atomic_number_b1', 'atomic_volume_b1',
'atomic_radius_b1', 'atomic_weight_b1', 'dipole_polarizability_b1',
'electron_affinity_b1', 'fusion_heat_b1', 'period_b1', 'vdw_radius_b1',
'covalent_radius_cordero_b1', 'ip_1_b1', 'ip_2_b1', 'ip_3_b1', 'eletronegativity_b1',
'ionic_radius_b1', 'atomic_number_b2', 'atomic_volume_b2', 'atomic_radius_b2',
'atomic_weight_b2', 'dipole_polarizability_b2', 'electron_affinity_b2', 'fusion_heat_b2',
'period_b2', 'vdw_radius_b2', 'covalent_radius_cordero_b2', 'ip_1_b2', 'ip_2_b2',
'ip_3_b2', 'eletronegativity_b2', 'ionic_radius_b2', 'atomic_number_x',
'atomic_volume_x', 'atomic_radius_x', 'atomic_weight_x', 'dipole_polarizability_x',
'electron_affinity_x', 'fusion_heat_x', 'period_x', 'vdw_radius_x',
'covalent_radius_cordero_x', 'ip_1_x', 'ip_2_x', 'ip_3_x', 'eletronegativity_x',
'ionic_radius_x', 'a', 'b', 'c', 'natoms', 'volume', 'PF', 'bx_mean']
>>> dc.number_object_cols().object_feature   # 字符串特征
['phase']  # 晶系特征
```

晶系特征转码

将字符串型特征{ ‘**晶系**’ }转为数字型特征，样本包括{'Cubic', 'Orthorhombic ', 'Tetragonal', 'Trigonal', 'Monoclinic', 'Hexagonal'} 6种, 根据**二进制转码**规则依次将其转为 **[001,010,011,100,101,110]** 共三维, 相较于其他转码方式,特征维度最低。

```
>>> from jamip.ml.preprocessing import CategoryCoding
>>> cc = CatagoryCoding(dataset_df = original_df, target = 'band_gap')

>>> binary_df = cc.category_coding_x(x = 'phase', encode_method = 'binary')
>>> binary_df = binary_df.transform_dataframe()
Binary encoding is used for columns ['phase']
>>> binary_df.columns
Index(['ionic_radius_a', 'atomic_number_b1', 'atomic_volume_b1',
       'atomic_radius_b1', 'atomic_weight_b1', 'dipole_polarizability_b1',
       'electron_affinity_b1', 'fusion_heat_b1', 'period_b1', 'vdw_radius_b1',
       'covalent_radius_cordero_b1', 'ip_1_b1', 'ip_2_b1', 'ip_3_b1',
       'eletronegativity_b1', 'ionic_radius_b1', 'atomic_number_b2',
       'atomic_volume_b2', 'atomic_radius_b2', 'atomic_weight_b2',
       'dipole_polarizability_b2', 'electron_affinity_b2', 'fusion_heat_b2',
       'period_b2', 'vdw_radius_b2', 'covalent_radius_cordero_b2', 'ip_1_b2',
       'ip_2_b2', 'ip_3_b2', 'eletronegativity_b2', 'ionic_radius_b2',
       'atomic_number_x', 'atomic_volume_x', 'atomic_radius_x',
       'atomic_weight_x', 'dipole_polarizability_x', 'electron_affinity_x',
       'fusion_heat_x', 'period_x', 'vdw_radius_x',
       'covalent_radius_cordero_x', 'ip_1_x', 'ip_2_x', 'ip_3_x',
       'eletronegativity_x', 'ionic_radius_x', 'a', 'b', 'c', 'natoms',
       'volume', 'PF', 'bx_mean', 'phase_0', 'phase_1', 'phase_2', 'band_gap'],
```

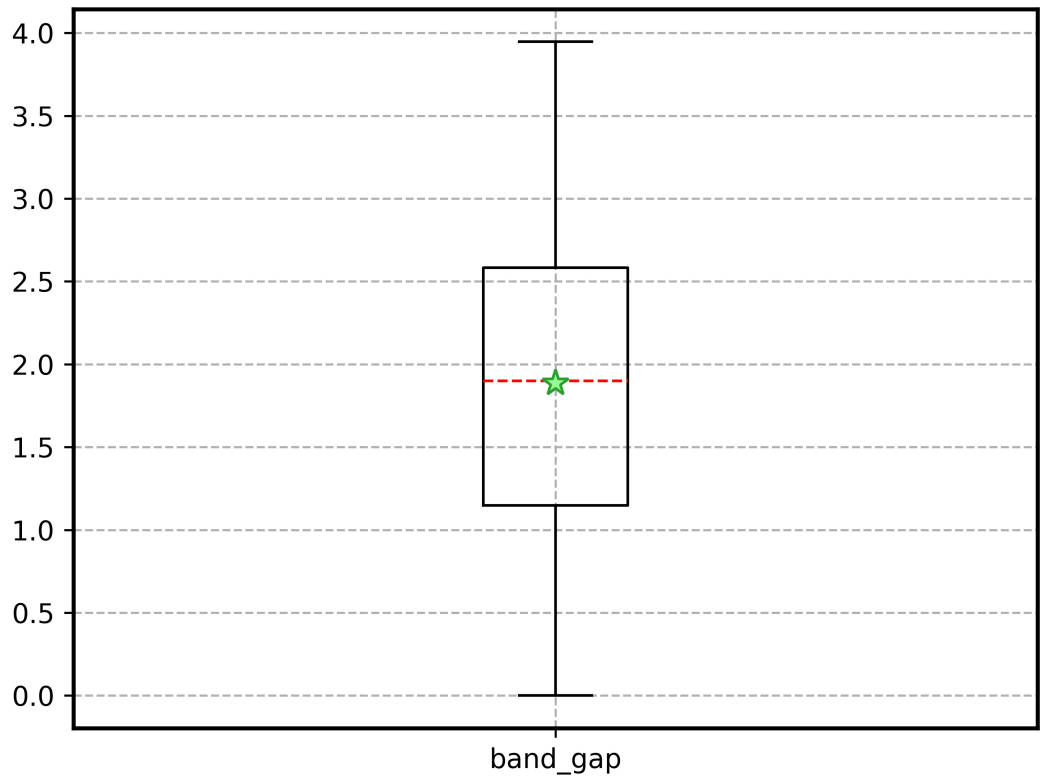
```
      dtype='object')
>>> number_feature = cc.number_object_cols().number_feature # 更新数字特征
```

箱式图分析带隙数据分布情况

箱式图是一种常见的数据分析,主要用于识别数据中的异常值并了解数据的离散分布情况

```
>>> from jamip.ml.plot import Plot
>>> Plot(df = binary_df).plot_box(features = 'band_gap', file_name = 'box')
```

带隙数据分布范围在箱式图上下限之间，无异常值数据，适合直接输入模型拟合。



2.机器学习模型搭建及评估

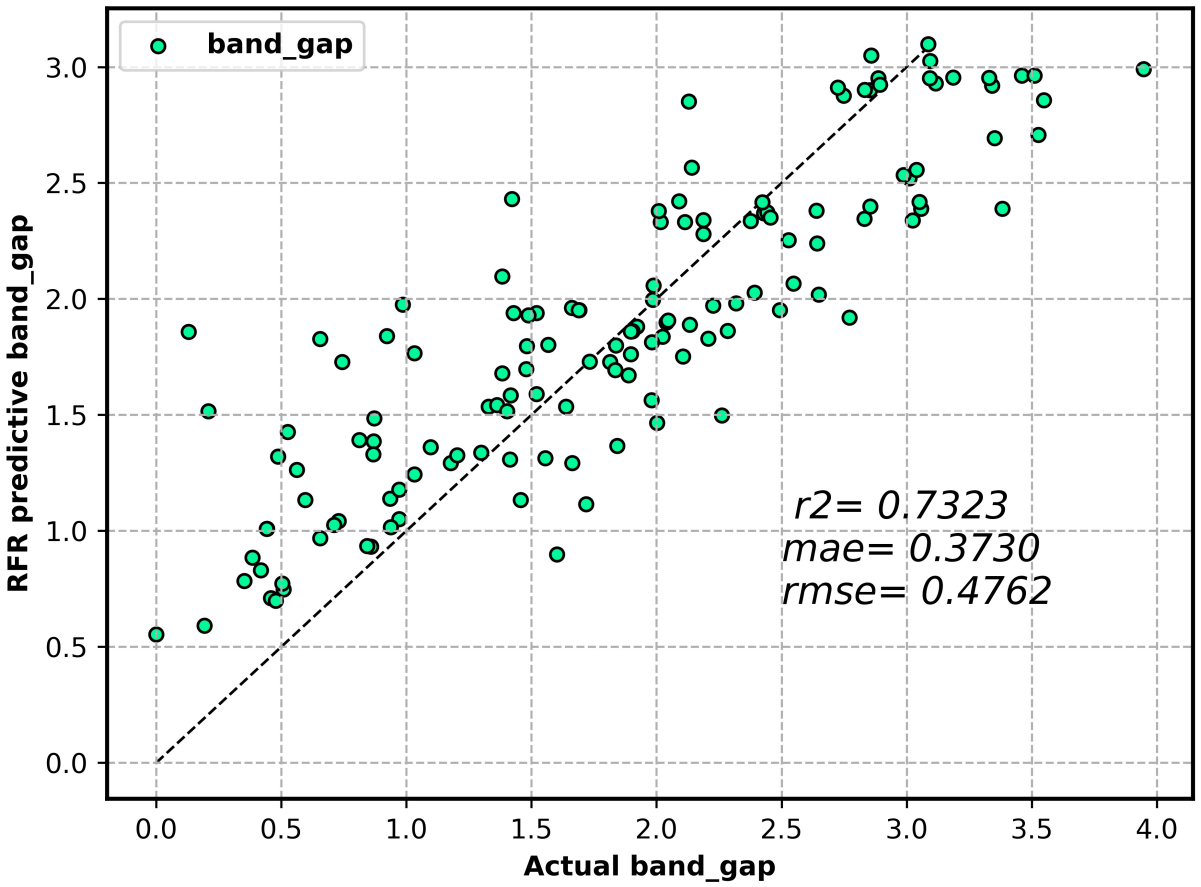
完成数据预处理后，可以将数据输入机器学习模型中，JAMIP提供多种类型机器学习模型， 接下来将以随机森林、支持向量机、LASSO正则模型为例。

```
# 调用模型构建类和后处理类
>>> from jamip.ml.model import RegressionModelbuilder, RegressionModelfitProcessing
>>> from sklearn.model_selection import train_test_split
>>> X_train, X_test, y_train, y_test = train_test_split(binary_df[number_feature],
                                                        binary_df['band_gap'],
                                                        test_size = 0.2, train_size =0.8)
```

随机森林

随机森林是通过集成算法将多个决策树集成的一种算法，其基本单元是决策树。

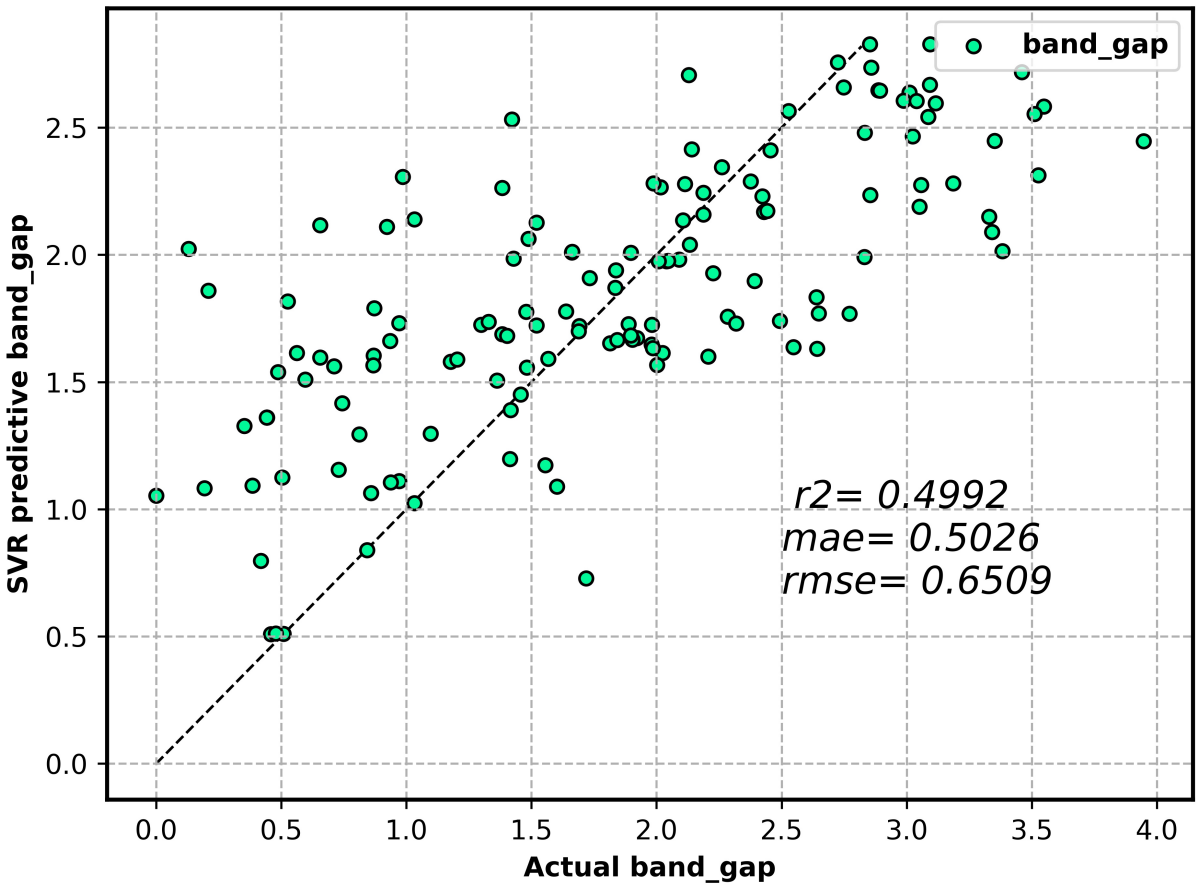
```
# 设置模型、超参数
# model_name可输入'RandomForestRegressor' or 'rfr'
>>> rfr = RegressionModelbuilder(model_name = 'rfr')
>>> rfr = rfr.model_set.hyper_set(n_estimators = 200,
                                  max_depth = 5,
                                  max_features = 'sqrt')
>>> prf = PlotRegression(model = 'RFR', fixed_model = rfr)
>>> prf.plot_actuality_predict_scatter('band_gap', # target
                                       X_train,
                                       y_train,
                                       X_test,
                                       y_test,
                                       evaluation_index= ['r2','mae','rmse'],
                                       file_name = 'RF_Actuality_Predict',
                                       derandomization = True,    # 去随机化
                                       iteration_number = 50)    # 迭代次数
```



支持向量机

通过将数据映射到高维空间，来解决原始低维空间问题，是一种传统机器学习模型。

```
>>> svr = RegressionModelbuilder(model_name = 'svr')
>>> svr = svr.model_set.hyper_set()
>>> prsvm = PlotRegression(model = 'SVR', fixed_model =svr)
>>> prsvm.plot_actuality_predict_scatter('band_gap',
                                         X_train,
                                         y_train,
                                         X_test,
                                         y_test,
                                         evaluation_index= ['r2','mae','rmse'],
                                         file_name = 'SVR_Actuality_Predict',
                                         derandomization = True,
                                         iteration_number = 10)
```

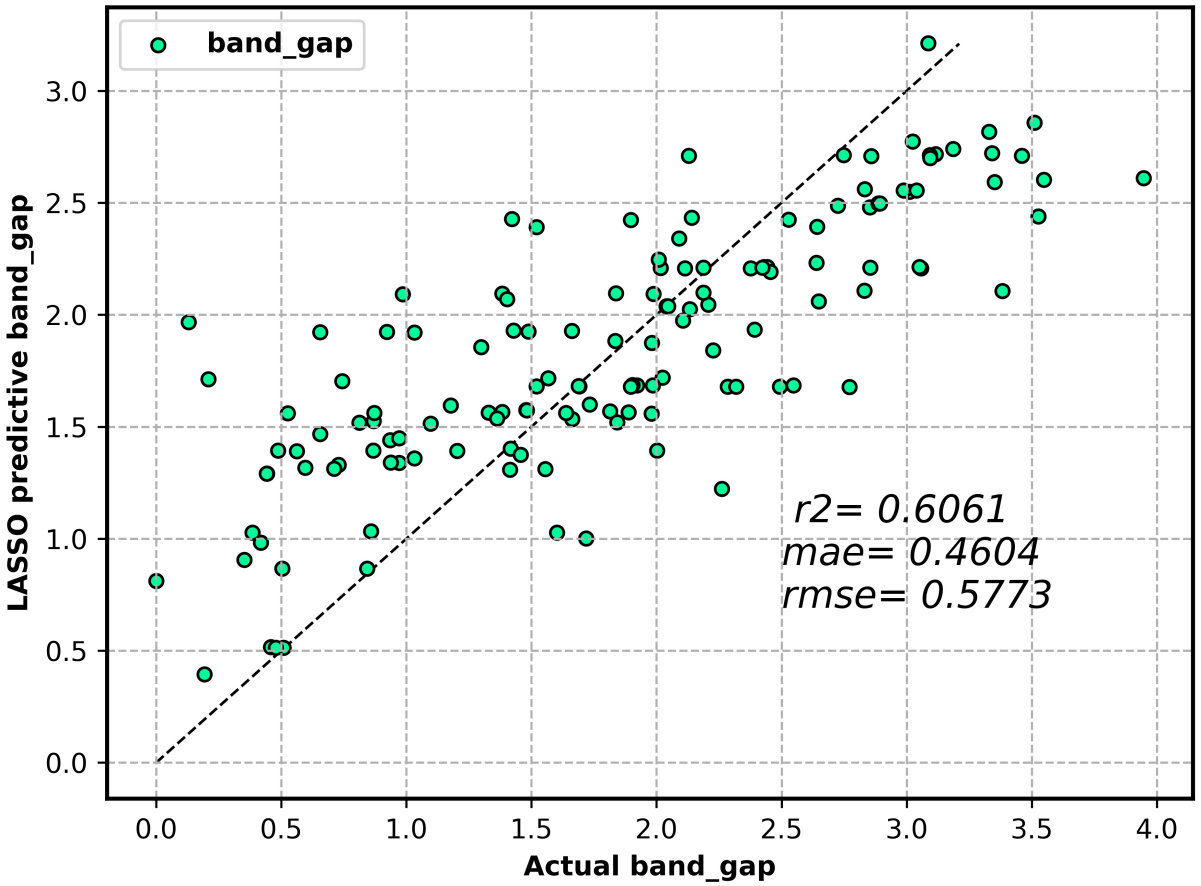


LASSO

在线性模型基础上采用L1范数防止模型过拟合现象的发生，同时也可用于特征选择。

```
>>> lasso = RegressionModelbuilder(model_name = 'lasso')
>>> lasoo = lasso.model_set.hyper_set()
>>> prlasso = PlotRegression(model = 'LASSO', fixed_model = lasso)
>>> prlasso.plot_actuality_predict_scatter('band_gap',
                                           X_train,
                                           y_train,
                                           X_test,
                                           y_test,
                                           evaluation_index= ['r2','mae','rmse'],
```

```
                                        file_name = 'LASSO_Actuality_Predict',
                                        derandomization = True,
                                        iteration_number = 10)
```



从三种模型拟合效果中来看，**随机森林**模型拟合效果最好，r2 = 0.7354, mae = 0.3585，rmse = 0.6006，误差比实际带隙范围小一个数量级，可以接受。

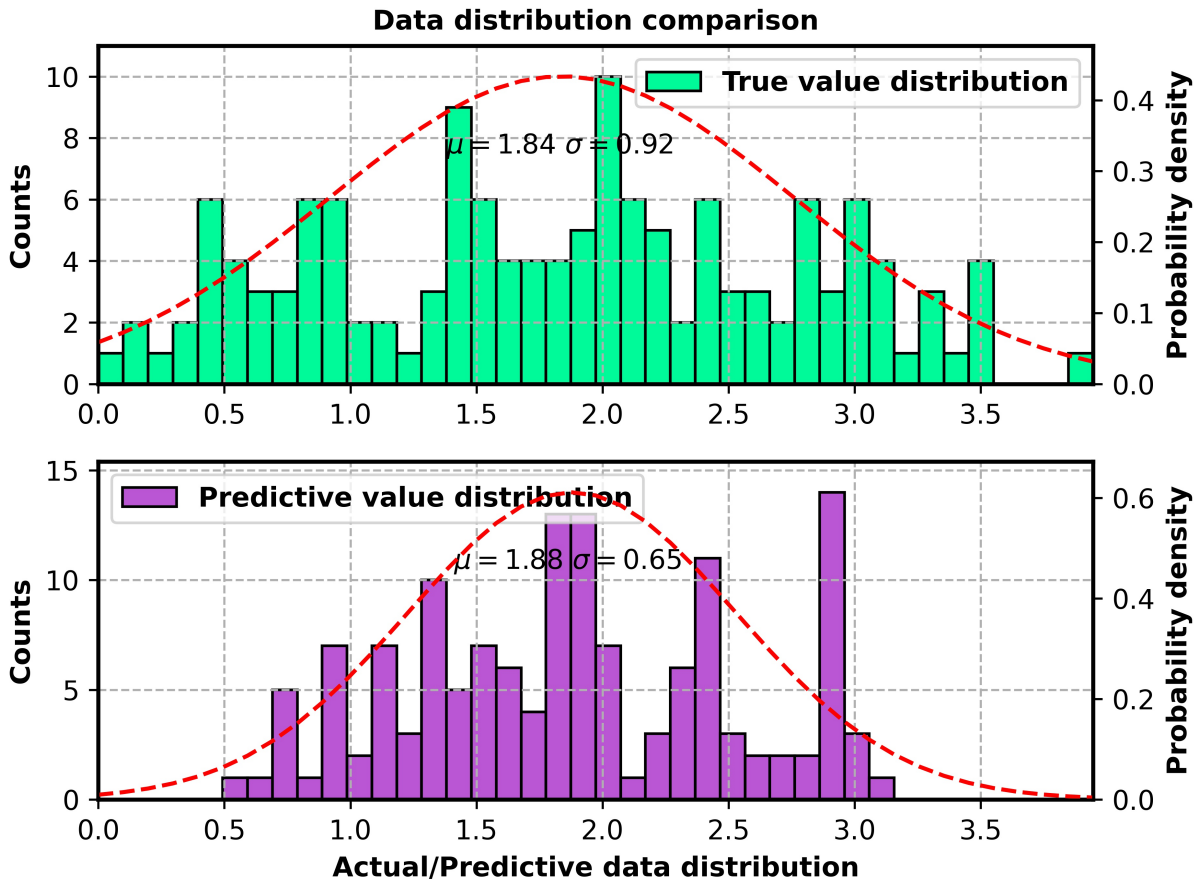
由于随机森林模型属于集成模型，每次进行拟合时结果略有差距但不会很大，若想消除这种影响，采取两种方案：

1.增大iteration_number, 平均化消除随机性;

2.在模型参数(hyper_set)功能设置random_state参数。

接下来将具体查看随机森林模型的实际-预测值分布图。

```
>>> PlotRegression('RF', rfr).plot_actuality_predict_scatter(X_train,
                                                             y_train,
                                                             X_test,
                                                             y_test)
```



从图像上看出，预测结果与实际值差距还是有一定差距，均值较接近，但标准差差距较大，接下来将进行特征工程，尝试提高拟合效果。

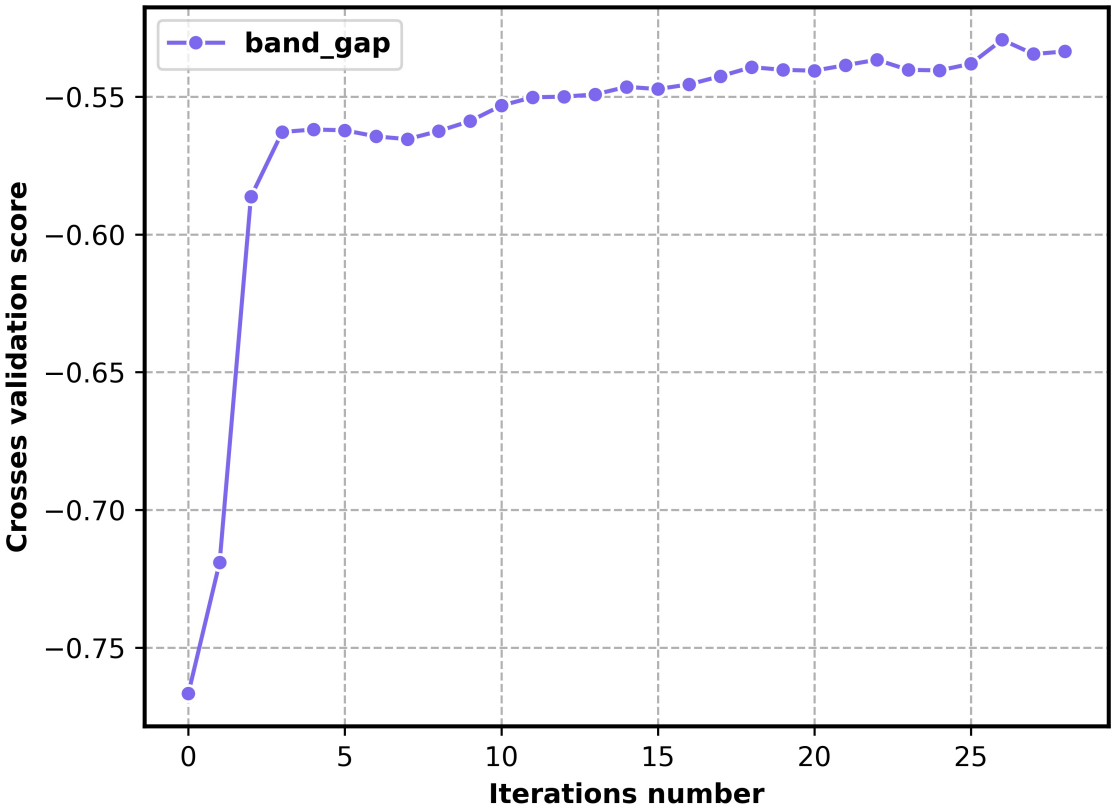
3.材料特征工程应用

树模型(随机森林、梯度提升决策树等)自带特征权重的属性，可以凭借这点剔除掉冗余特征提高模型拟合精度还可以挖掘出与目标性质相关特征进一步理解材料。接下来我们采用**递归特征删除**的方法，完成后模型整体预测性能略有升高并挖掘出重要特征。

注：不同次迭代效果，对删除特征种类差距很小，下例展示为最普遍效果，用户可执行Script.py自行测试。

```
>>> from jamip.ml.feature_engineering import FeatureSelection, RecursiveFeatureSelection

>>> rfss = RecursiveFeatureSelection(model = 'rfr',
                                     dataset_df = binary_df,
                                     target = 'band_gap',
                                     learning_task = 'regression',
                                     selected_min_features = 1,
                                     scoring = 'neg_root_mean_squared_error',
                                     cv = 5,
                                     step=2)
>>> rfss.set_model_hyper(n_estimators = 300,
                         max_depth = 5,
                         max_features = 'sqrt').set_selector().fit()
>>> rfss.plot_rfe_cv('RFE')    # 输出递归特征选择图
```

由图像可知，当进行第二次迭代时，模型拟合效果最好，即剔除了4个特征




```
# 且Packing factor为最重要的特征
>>> fitted_rfr.get_feature_importances(number_feature)
         RandomForestRegressor  weight_score      # 特征排序
0                           PF      0.096059
1              atomic_volume_b1      0.057188
2           electron_affinity_b1      0.048173
3    covalent_radius_cordero_b1      0.038040
4               ionic_radius_b1      0.037837
5                       bx_mean      0.033092
6                 vdw_radius_b1      0.031413
7           electron_affinity_x      0.030204
8              atomic_radius_b1      0.030127
9            eletronegativity_x      0.028512
10                       ip_1_x      0.027461
11    covalent_radius_cordero_x      0.024008
12     dipole_polarizability_b1      0.022586
13              atomic_radius_x      0.022389
14                       ip_3_x      0.021925
15                      ip_2_b1      0.020932
16                       ip_2_x      0.020705
17               ionic_radius_x      0.019773
18              atomic_weight_x      0.018962
19                     period_x      0.018857
20                      ip_3_b2      0.018277
21              atomic_number_x      0.017865
22                 vdw_radius_x      0.016813
23                fusion_heat_x      0.016334
```

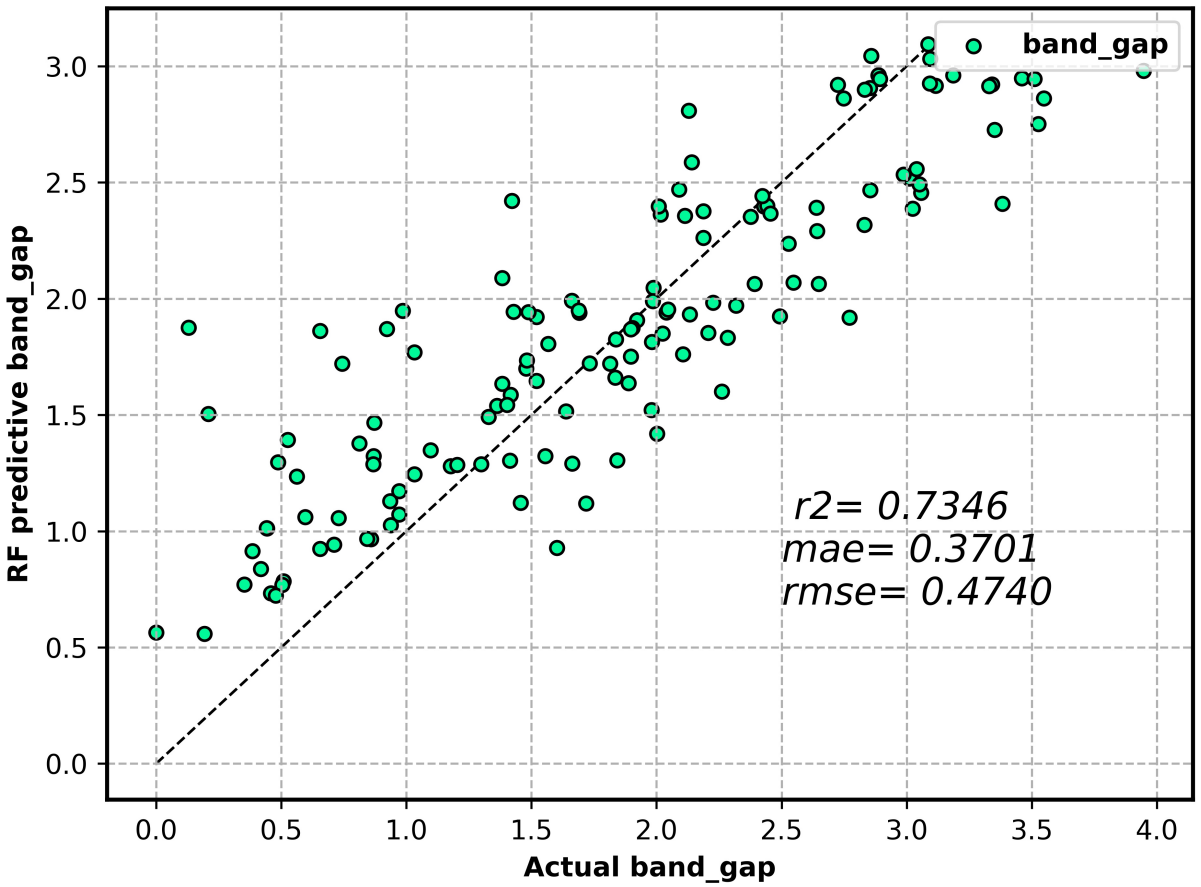
```
24            atomic_number_b1      0.015193
25            atomic_weight_b1      0.014272
26                     ip_3_b1      0.014267
27     dipole_polarizability_x      0.014193
28                           c      0.013701
29             atomic_volume_x      0.013145
30                     ip_2_b2      0.013073
31                      volume      0.012768
32                           b      0.012370
33              vdw_radius_b2      0.011868
34                      natoms      0.011367
35                     phase_1      0.010653
36        electron_affinity_b2      0.010621
37              fusion_heat_b1      0.009879
38                   period_b1      0.009871
39         eletronegativity_b1      0.009010
40                           a      0.008495
41                     ip_1_b1      0.007970
42              fusion_heat_b2      0.007839
43                     ip_1_b2      0.007537
44    dipole_polarizability_b2      0.007310
45         eletronegativity_b2      0.006570
46            atomic_volume_b2      0.006401
47            atomic_weight_b2      0.006266
48            atomic_number_b2      0.005852
49             ionic_radius_b2      0.004589
50            atomic_radius_b2      0.004576
51  covalent_radius_cordero_b2      0.004403
52              ionic_radius_a      0.003706
53                   period_b2      0.002737
54                     phase_0      0.001447
55                     phase_2      0.000522

>>> rfss.features_rank          # 递归删除结果
                          rfr  rank
0             atomic_number_b1     1
1             atomic_volume_b1     1
2             atomic_radius_b1     1
3             atomic_weight_b1     1
4     dipole_polarizability_b1     1
5         electron_affinity_b1     1
6               fusion_heat_b1     1
7                    period_b1     1
8                vdw_radius_b1     1
9   covalent_radius_cordero_b1     1
10                    ip_1_b1     1
11                    ip_2_b1     1
12                    ip_3_b1     1
13         eletronegativity_b1     1
14             ionic_radius_b1     1
15            atomic_number_b2     1
16            atomic_volume_b2     1
17            atomic_radius_b2     1
18            atomic_weight_b2     1
19    dipole_polarizability_b2     1
20        electron_affinity_b2     1
21              fusion_heat_b2     1
22               vdw_radius_b2     1
```

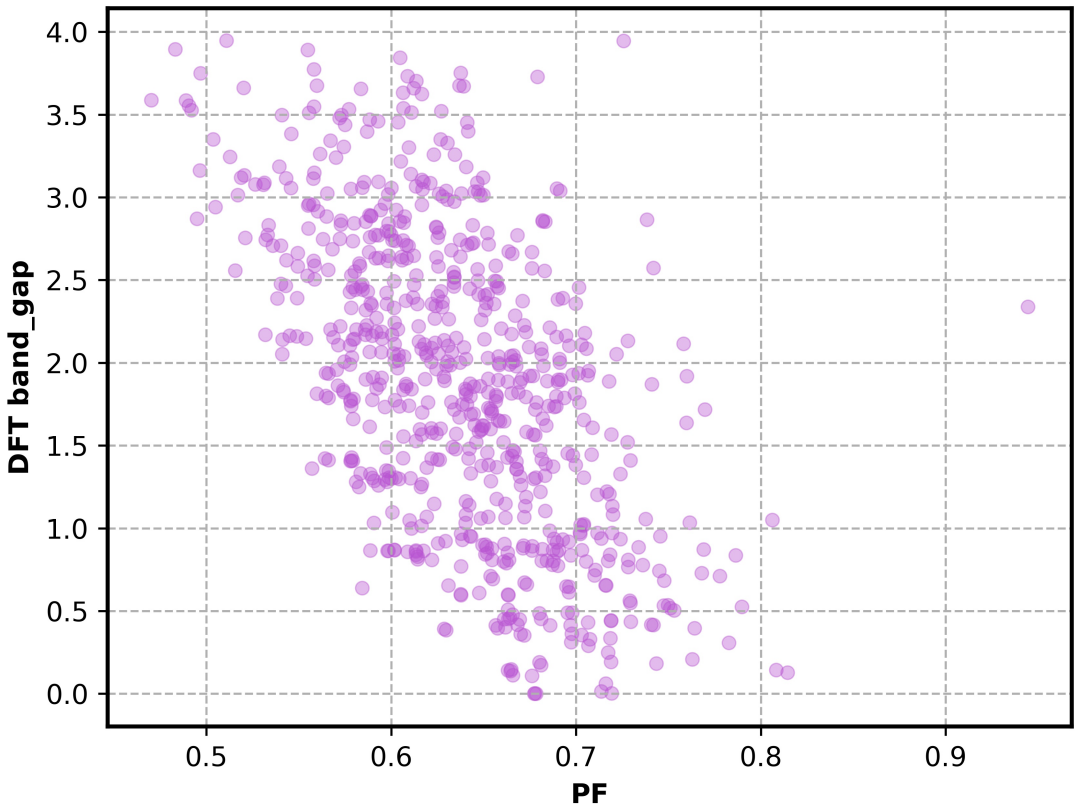
```
covalent_radius_cordero_b2     1
                   ip_1_b2     1
                   ip_2_b2     1
                   ip_3_b2     1
       eletronegativity_b2     1
           ionic_radius_b2     1
           atomic_number_x     1
           atomic_volume_x     1
           atomic_radius_x     1
           atomic_weight_x     1
   dipole_polarizability_x     1
       electron_affinity_x     1
             fusion_heat_x     1
                  period_x     1
              vdw_radius_x     1
 covalent_radius_cordero_x     1
                    ip_1_x     1
                    ip_2_x     1
                    ip_3_x     1
        eletronegativity_x     1
            ionic_radius_x     1
                         a     1
                         b     1
                         c     1
                   natoms     1
                   volume     1
                       PF     1
                  bx_mean     1
                  phase_1     1
            ionic_radius_a     2
                 period_b2     2
                   phase_0     3
                   phase_2     3
```

根据RecursiveFeatureSelection结果删除对目标性质预测产生负影响的特征

```
# 输出特征排序, ['ionic_radius_a', 'period_b2','phase_0','phase_2']被删除;
>>> deled_number = rfss.features_rank['rank'][rfss.features_rank['rank'] != 1].keys()
>>> deled_features = list(rfss.features_rank['rfr'][deled_number])
```

```
>>> for i in deled_features:
>>>     number_feature.remove(i)
>>> pr = PlotRegression('RF', rfr)
>>> pr.plot_actuality_predict_scatter('band_gap',
                                      X_train[number_feature],
                                      y_train,
                                      X_test[number_feature],
                                      y_test,
                                      evaluation_index= ['r2','mae','rmse'],
                                      file_name = 'Actuality_Predict')
```





可以发现，特征剔除后，模型预测效果略有增加，其中R2效果增加最为明显，约0.005，证明特征选择的有效性。 由上述特征选择可知，Packing factor为最重要特征，且高于第二位特征权重接近两

倍，如上图，可以看出packing facotr与带隙整体呈现**负相关**，说明机器学习有能力挖掘描述符与材料特征性质之间的关系。