```
#import packages

import pandas as pd
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import cross_val_score, train_test_split
from bayes_opt import BayesianOptimization

#Load data

wn = pd.read_csv('winequality2.csv')
wn.isnull().values.any()
x=wn.drop('quality', axis=1)
y=wn.quality
from sklearn.model_selection import train_test_split
x_train,x_test,y_train, y_test= train_test_split(x,y, test_size=0.2, random_state=42)

#Bayesian on RandomForest
# Define objective function for optimization

RandomForestRegressor._get_param_names()


def objective(n_estimators, max_depth, min_samples_split, max_features,
max_leaf_nodes, max_samples, min_impurity_decrease, min_samples_leaf,
min_weight_fraction_leaf):
   model = RandomForestRegressor(n_estimators=int(n_estimators),
                                      max_depth=int(max_depth),
                                      min_samples_split=int(min_samples_split),
                                      max_features=min(max_features,0.999),
                                       max_leaf_nodes=min(max_leaf_nodes,2),
                                     max_samples=min(max_samples,0.999),
                                     min_impurity_decrease=int(min_impurity_decrease),
                                     min_samples_leaf=int(min_samples_leaf),
                                     min_weight_fraction_leaf=int(min_weight_fraction_leaf),
                                      random_state=42)

   return -1.0 * cross_val_score(model, x_train, y_train, cv=3,
scoring="neg_mean_squared_error").mean()
```

```
#Define Parameters

param_bounds = {
   'n_estimators': (10, 500),
   'max_depth': (1, 500),
   'min_samples_split': (2, 200),
   'max_features': (0.1,0.999),
   'max_leaf_nodes': (2,200),
   'max_samples': (0,1),
   'min_impurity_decrease': (0,1000),
   'min_samples_leaf': (0,1000),
   'min_weight_fraction_leaf': (0,0.5),
}

#Run it

opt= BayesianOptimization(f=objective, pbounds=param_bounds, random_state=42)
opt.maximize(init_points=5, n_iter=25)

#Best params

best_params = opt.max['params']
best_params



#Final model

final_model = RandomForestRegressor(n_estimators=int(best_params['n_estimators']),
                                    max_depth=int(best_params['max_depth']),
                                    min_samples_split=int(best_params['min_samples_split']),
                                    max_features=best_params['max_features'],
                                     max_leaf_nodes=int(best_params['max_leaf_nodes']),
                                     max_samples=best_params['max_samples'],

                       min_impurity_decrease=int(best_params['min_impurity_decrease']),
                                     min_samples_leaf=int(best_params['min_samples_leaf']),

                       min_weight_fraction_leaf=int(best_params['min_weight_fraction_leaf']),
                                 random_state=42)
final_model.fit(x_train, y_train)
score = final_model.score(x_test, y_test)
print({score})



#New random forest

opt_rf = RandomForestRegressor(**best_params_f, random_state=42)
opt_rf.fit(x_train, y_train)

score = opt_rf.score(x_test, y_test)
print({score})


#SVM Bayesian Opt

import pandas as pd
from sklearn.svm import SVC
from sklearn.model_selection import cross_val_score, train_test_split
from bayes_opt import BayesianOptimization

#Load data

wn = pd.read_csv('winequality2.csv')
wn.isnull().values.any()
x=wn.drop('quality', axis=1)
y=wn.quality
from sklearn.model_selection import train_test_split
x_train,x_test,y_train, y_test= train_test_split(x,y, test_size=0.2, random_state=42)

#Define objective function

SVC._get_param_names()
def objective(C, cache_size, coef0, gamma):
   model = SVC(C=int(C),

                                cache_size=int(cache_size),
                    coef0=int(coef0),
                   gamma=int(gamma),
                    random_state=42)

  return -1.0 * cross_val_score(model, x_train, y_train, cv=3,
scoring="neg_mean_squared_error").mean()



param_bounds = {
  'C': (0,100),
  'cache_size': (2, 200),
  'coef0': (2,200),
  'gamma': (0,1000),
}

#Run it

opt2= BayesianOptimization(f=objective, pbounds=param_bounds, random_state=42)
opt2.maximize(init_points=5, n_iter=25)

#Best params

best_params = opt2.max['params']
best_params



#Final model



final_model2 = SVC(C=int(best_params['C']),

                     cache_size=int(best_params['cache_size']),
                     coef0=best_params['coef0'],
                      gamma=best_params['gamma'],
                    probability=int(best_params['probability']),
                       random_state=42)
final_model2.fit(x_train, y_train)
score2 = final_model2.score(x_test, y_test)
print({score2})



#New random forest

opt2_rf = SVC(**best_params, random_state=42)
opt2_rf.fit(x_train, y_train)

score2 = opt2_rf.score(x_test, y_test)
print({score2})
```