```
  [file,path] = uigetfile({'*.xlsx';'*.xls'},'File Selector');
        filepath=strcat(path,file);
        Data = xlsread(filepath,'sheet1');
        Datasize=size(Data);
        X=Data;

        msgbox("导入数据成功，正在计算，请稍后");
        %%  第二步：判断是否需要正向化
        [n,~] = size(X);
        %% 第三步：对正向化后的矩阵进行标准化
        Z = X ./ repmat(sum(X.*X) .^ 0.5, n, 1);
        %% 让用户判断是否需要增加权重
        weight = Entropy_Method(Z);
        %% 第四步：计算与最大值的距离和最小值的距离，并算出得分
        D_P = sum([(Z - repmat(max(Z),n,1)) .^ 2 ] .* repmat(weight,n,1) ,2) .^ 0.5;    % D+ 与最
        D_N = sum([(Z - repmat(min(Z),n,1)) .^ 2 ] .* repmat(weight,n,1) ,2) .^ 0.5;    % D- 与最
        S = D_N ./ (D_P+D_N);     % 未归一化的得分
        % disp('最后的得分为：')
        stand_S = S / sum(S);
%         stand_S = 1./stand_S;

        pathname = '.\';
        filename = 'stand_S.mat';
        filepath2=strcat(pathname,filename);
        save(filepath2,'stand_S');

        [~,index] = sort(stand_S ,'descend');

        dates=Data(:,1);
        result = [dates index stand_S];
        result = sortrows(result,2);
        A=result(:,1);
        B=result(:,2);
        C=result(:,3);

        plot(app.UIAxes,dates,stand_S);

        R=[A B C];

        app.Output.Data=R;
```