In [ ]:

```
import xlsxwriter
import xlrd


def extract(fname):
    wb = xlrd.open_workbook(fname)
    sheet = wb.sheet_by_index(0)

    ind_comp = []
    temp = []

    for i in range(sheet.nrows):
        for j in range(sheet.ncols):
            str1 = sheet.cell_value(i, j)
            str1 = str(str1).upper()
            if (str1.__contains__("FROM DATE")):
                ind_comp = [i, j]
                temp.append(ind_comp)

    count = 0
    start_row = temp[0][0] + 1
    head_col = []
    temp_list = []
    for i in temp:
        for j in range(sheet.ncols):
            count = i[0]
            str1 = sheet.cell_value(count, j)
            str1 = str(str1).upper()
            if (str1 != ''):
                head_col.append([j, str1])
    temp.append([sheet.nrows, 0])
    print(temp)

    # if(count < temp._len_()):
    #     count+=1

    print(head_col)

    data_dict = {}
    for k in head_col:
        data_dict[k[1]] = []

    for i in range(len(temp) - 1):
        start = temp[i][0]
        print(start)
        end = temp[i + 1][0]
        print(end)
        for j in range(sheet.ncols):
            rl = []
            if (sheet.cell_value(start, j) != ""):
                name = str(sheet.cell_value(start, j)).upper()
                print(name)
                for k in range(start + 1, end):
                    str2 = str(sheet.cell_value(k, j)).upper()
                    if str2 != '':
                        rl.append(str(sheet.cell_value(k, j)).upper())
                    else:
                        break
                data_dict[name] = rl


    for i in data_dict.keys():
        print(i)
        print(data_dict[i])

    workbook = xlsxwriter.Workbook("data1.xlsx", {'strings_to_numbers': True
})
    worksheet = workbook.add_worksheet()
    row = 0
    col = 0
    # dict1={"a":[1,2,3],"b":[4,5,6],"c":[7,8,9]}
    for i in data_dict.keys():
        row = 0
        worksheet.write(row, col, i)
        for j in data_dict[i]:
            row += 1
            worksheet.write(row, col, j)
        col += 1
    workbook.close()

file_names=["1mar_april","2april_may","3may_june","4june_july","5july_aug","6
aug_sept","7sept_oct","8oct_nov"]
for i in file_names:

    extract("data/"+i+'.xlsx')





data.drop_duplicates(inplace=True)  # removing duplicates

data.dropna()


limit = {}


def IQR_outlier(dt, name):
    q1 = dt.quantile(.25)
    q3 = dt.quantile(.75)
    iqr = q3 - q1
    l_limit = q1 - 1.5 * iqr
    r_limit = q3 + 1.5 * iqr
    l_limit = round(l_limit, 2)
    r_limit = round(r_limit, 2)
    limit[name] = [l_limit, r_limit]


def removal_outlier(st, name):
    st = st[st < limit[name][1]]
    st = st[st > limit[name][0]]
    return st


for i in range(len(names)):
    IQR_outlier(data[names[i]], names[i])
```

```
for i in range(len(names)):
    data[names[i]] = removal_outlier(data[names[i]], names[i])




from scipy.stats import zscore

# data.apply(zscore)  # feature scaling


data = (data - data.min()) / (data.max() - data.min())


from sklearn.model_selection import train_test_split
train1, test1 = train_test_split(data, test_size=0.25)  # splitting data


train = pd.read_excel('dt1_train.xlsx')
test = pd.read_excel('dt1_test.xlsx')
names = train.columns


for i in range(len(names)):
    sns.distplot(data[names[i]], hist=True, kde=True, bins=100, color = 'red'
,hist_kws={'edgecolor':'black'})
    plt.xlabel('Range of '+str(names[i]))
    plt.title('Histogram + Density for '+str(names[i]))
    plt.show()
    plt.clf()
    plt.cla()

for i in range(len(names)):
    for j in range(len(names)):
        if (names[i] != names[j]):
            train[names[i]] = train[names[i]].apply(np.sqrt)
            train[names[j]] = train[names[j]].apply(np.sqrt)
            test[names[i]] = test[names[i]].apply(np.sqrt)
            test[names[j]] = test[names[j]].apply(np.sqrt)

            x_train = train[names[i]]
            y_train = train[names[j]]
            x_test = test[names[i]]
            y_test = test[names[j]]

            x_train = np.array(x_train)
            y_train = np.array(y_train)
            x_test = np.array(x_test)
            y_test = np.array(y_test)

            x_train = x_train.reshape(-1, 1)
            x_test = x_test.reshape(-1, 1)

            clf = LinearRegression(normalize=True)
            clf.fit(x_train, y_train)
            y_pred = clf.predict(x_test)

            n11 = r2_score(y_test, y_pred)
            if (n11 > 0.7) :
                print(names[i], 'and', names[j])
                print(n11)
                st1 = names[i] + ' and ' + names[j] + '\n'

                sns.regplot(y_test, y_pred)
                plt.xlabel(names[i])
                plt.ylabel(names[j])
                plt.title('Scatter plot of ' + str(names[i]) + ' and ' + str(
names[j]) + ' ' + str(round(n11)))

                plt.savefig('pics/' + str(names[i]) + '_' + names[j] + '.png'
)
                plt.show()
                plt.clf()
                plt.cla()


import statsmodels.api as sm
X = NBA['W']
y = NBA[['PTS', 'oppPTS']]
X = sm.add_constant(X)
model11 = sm.OLS(y, X).fit()
model11.summary()
```