```
# -*- coding: utf-8 -*-

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import scipy.stats as ss
import cluster_tools as ct
import sklearn.cluster as cluster
import sklearn.metrics as skmet
import sklearn.preprocessing as pp
import scipy.optimize as opt
import errors as err

def read_data_all(filename, countries):

    """



    Parameters
    ----------
    filename : TYPE
        DESCRIPTION.
    countries : TYPE
        DESCRIPTION.

    Returns
    -------
    df : TYPE
        DESCRIPTION.
    df_t : TYPE
        DESCRIPTION.
```

```
    """

    # read the data
    df = pd.read_csv(filename, skiprows=4)

    # set index
    df.index = df.iloc[:, 0]
    df = df.iloc[:, 1:]

    # transpose the data
    df_t = df.T
    df_t.index = df_t.index.astype(int)
    df = df.loc[countries, np.arange(1990, 2021).astype(str)].T
    return df, df_t


def poly(x, a, b, c, d):
    """


    Parameters
    ----------
    x : TYPE
        DESCRIPTION.
    a : TYPE
        DESCRIPTION.
    b : TYPE
        DESCRIPTION.
    c : TYPE
        DESCRIPTION.
```

```
    d : TYPE
        DESCRIPTION.

    Returns
    -------
    f : TYPE
        DESCRIPTION.

    """
    """ Calulates polynominal"""

    x = x - 1990
    return a + b*x + c*x**2 + d*x**3

def build_cluster_graph(country, carbon_emission_df_t, forest_area_df_t):
    """



    Parameters
    ----------
    country : TYPE
        DESCRIPTION.

    Returns
    -------
    df_cluster : TYPE
        DESCRIPTION.

    """
    df_cluster = pd.DataFrame({'co2': carbon_emission_df_t[country],
                               'forest_area': forest_area_df_t[country]}).dropna()
```

```
    df_norm, _, _ = ct.scaler(df_cluster)

    ncluster = 2
    kmeans = cluster.KMeans(n_clusters=ncluster, n_init=20)
    kmeans.fit(df_norm)

    labels = kmeans.labels_
    cen = ct.backscale(kmeans.cluster_centers_, _, _)



    # calculate silhouette clusters
    xkmeans, ykmeans = cen[:, 0], cen[:, 1]
    x, y = df_cluster['co2'], df_cluster['forest_area']

    cmap = plt.get_cmap("tab10")
    plt.figure(figsize=(15, 8), dpi=300)
    scatter = plt.scatter(x, y, 25, labels, cmap=cmap, marker="o", edgecolors='k', linewidth=0.8)
    plt.scatter(xkmeans, ykmeans, 150, "k", marker="D", label="Cluster Centers", edgecolors='w',
linewidth=1.5)
    plt.scatter(xkmeans, ykmeans, 150, "y", marker="+", label="Centroid", edgecolors='k',
linewidth=1.5)

    plt.legend()
    plt.grid(True, linestyle="--", alpha=0.7)
    plt.title(f"Demonstration of graph with clusters for {country}", fontsize=20, color="black")
    plt.xlabel("Co2 emission (kt)", fontsize=15, color="black")
    plt.ylabel("Forest area (% of land area)", fontsize=15, color="black")
    plt.colorbar(scatter, label='Two clusters', ticks=range(ncluster))
    plt.savefig(f"cluster_of_{country}.png", dpi=300, va="center")
    plt.show()
    return df_cluster
```

```
def build_fitting_graph(df_cluster, indicator, title):
    df_cluster["Year"] = df_cluster.index
    params, covar = opt.curve_fit(poly, df_cluster["Year"], df_cluster[indicator])

    df_cluster["fit"] = poly(df_cluster["Year"], *params)

    year = np.arange(1990, 2030)
    forecast = poly(year, *params)
    sigma = err.error_prop(year, poly, params, covar)
    low, up = forecast - sigma, forecast + sigma

    df_cluster["fit"] = poly(df_cluster["Year"], *params)

    plt.figure(figsize=(15, 8), dpi=250)
    plt.plot(df_cluster["Year"], df_cluster[indicator], label=indicator)
    plt.plot(year, forecast, label="Forecast")

    plt.xlabel("Year", fontsize=15)
    plt.ylabel(f"{indicator} ({title.split()[0]} {title.split()[1]})", fontsize=15)
    plt.title(title, fontsize=20)

    plt.fill_between(year, low, up, color="yellow", alpha=0.6, label="Confidence margin")
    plt.savefig(f"{title}.png", dpi=300, va="center")

    plt.legend()
    plt.show()

# Main part
countries = ["Indonesia", "France"]
carbon_emission_df_t = read_data_all("carbon_emission.csv", countries)
forest_area_df_t = read_data_all("forest_area.csv", countries)
```

```
for country in countries:
    df_cluster = build_cluster_graph(country, carbon_emission_df_t, forest_area_df_t)
    build_fitting_graph(df_cluster, "forest_area", f"Forest area of {country}")
    build_fitting_graph(df_cluster, "co2", f"Co2 emission of {country}")
```