Task 1- Global Commodity Trade

Gopi Krishna Thatha (14986600 – thathag@coventry.ac.uk)
7135CEM - Modelling and Optimisation Under Uncertainty
Module Leader Name: Dr. Omid Chatrabgoun

Abstract

Global commerce is a system of interconnected markets with diverse products and high amounts of traded goods, their analysis and forecasting need elaborate approaches. For this, the current research uses the "Global Commodity Trade Statistics" data set from Kaggle donated by the United Nations while employing Bayesian Ridge Regression from scikit-learn in predicting trade values (USD). Attributes in the dataset are country, year, commodity code, description, flow type, trade value, weight, quantity and commodity category, which provide a clear picture of the global trade system.

The overall goal is to improve the prediction of the Trade Value using Bayesian methods for the incorporation of prior knowledge as well as for dealing with uncertainty. This is a step-by-step structure of the study. The first step is an explanation of the importance of predicting trade values, and the second step is a literature review. An informal discussion about the data set and the importance of the problem is included.

The discussion under the methodology subsegment elucidates the uses of Bayesian Ridge Regression, data preprocessing, and feature selection. Accuracy of the trade value can then be ascertained by comparing the results with the actual trade values in the knowledge-based model. The study also takes into account social, ethical, legal, and professional factors while conducting analyses and is aware of any possible biases in data use.

Introduction

International trade is a crucial aspect of the economic connection of the world economy system and plays a critical role in the stability and development of countries. Analyzing these variables is important for policy makers, economists, business people and academics who need to make sense of the trade, flow of

goods and services and the relations between nations. It is therefore important to predict the trade values as precisely as possible to enable the prediction of future economic conditions for policy makers, businessmen and the formulation of sound trade policies. This study aims at using superior machine learning algorithms, namely Bayesian Ridge Regression, to predict the trade value of the commodities based on the data set obtained from Kaggle's "Global Commodity Trade Statistics".

The "Global Commodity Trade Statistics" dataset is a large and detailed United Nations' source of international trade data. It covers many products and captures trade between tens of nations across multiple years. The Appendix contains the list of the primary variables of the dataset that can be divided into several types: the country or area of trade, the year of the transaction, the commodity code and description, the flow of trade (import or export), the trade value in USD, the weight of the commodity in kilograms, the quantity and its measurement type, and the commodity category. The large volume of information available in this database is another advantage, as it allows for accurate assessment of the state of world trade at a given period.

Bayesian Ridge Regression which has been used in this study is a highly advanced statistical tool that combines Bayesian principles with linear regression models. Unlike other regression methods, the Bayesian Ridge Regression takes prior information and probabilistic characteristics into consideration and as a result can better estimate the uncertainty of the outcome. This makes it particularly appropriate for the variability and complexity that characterize the global trade data.

In this research, we further extend such work by employing Bayesian Ridge Regression on the "Global Commodity Trade Statistics" data. We will outline the data preprocessing done in order to clean and transform the data, the feature selection performed to determine the best variables to use in the model, and the system configuration used in training and testing the regression model. Cross validation aims to provide a model with high accuracy of predicting trade values as well as the ability to explain the underlying factors.

Problem and Dataset Description

Problem Description

Budgeting and forecasting is one of the most daunting tasks that has a broad impact on the overall economic planning, policy formulation, and strategic management for commodities' trade. International

commerce is subjected to the numerous factors such as economic, political, social, technological, and natural conditions and changes in the marketplace. These complexities can be managed, and a proper prediction model can help officers to identify them and make correct decisions

Data Set(s) Description

The analysis of the given subject is based on the "Global Commodity Trade Statistics" dataset taken from Kaggle website and prepared by the United Nations. This dataset is a massive database of international trade which describes the transactions of thousands to tens of thousands commodities imported and exported by numerous countries over years. Due to the large number of values and comprehensive overview of the global trade activities, this dataset can be suitable for predictive modeling.

The dataset includes the following key attributes:The dataset includes the following key attributes:

Column	Type	Definition
Country or Area	Categorical	The supplier or buyer party that is involved in the trade transaction. It aids in determining the relations of trade and collaboration of unique countries.
Year	Integer	The year that the trade transaction was initiated / completed / reported. It enables one to look at the temporal changes in trading patterns- that is the trends in trading.
Commodity Code	Categorical	The Harmonized Commodity Description and Coding System (HS) which is a classification of products in the international market. Every good is assigned a six-digit number, thus, enhancing the means of commodity classification.
Commodity Description	Text	Written work or explanation of the commodity helping to put the object of trade into context and improve understanding of

		traded item.
Flow	Categorical	Whether the commodity was an export or an import, was also reflected in the direction of trade. This attribute aids in expounding on trade balances and traffic conditions of different nations.
Trade Value (trade_usd)	Integer	These are the exact value of the trade transaction in USD. This is a target variable to predict, which reflects the economically valuable amount of each transaction.
Weight (weight_kg)	Integer	This is the mass of the traded good which is in kilograms. This attribute can affect the trade value and is one of the major components considered in the evaluation.
Quantity Name	Categorical	The measure of type that is present in the number of pieces of the had commodity or weight in kilograms. This gives more meaning to the quantity attribute hence improves the conceptualization of the model.
Quantity	Integer	The quantity of the quantity name obtained through counting, in cases where there is the direct trading of the commodity.
Category	Categorical	A code assigned to the specific commodity on the basis of the similarity of the items in demand, assisting in the examination of demand patterns in distinct classes of commodities.

Methods

When forecasting the trade value (in USD) of the commodities with the help of the "Global Commodity Trade Statistics" dataset, the authors used Bayesian Ridge Regression, which can be described as a promising machine learning algorithm. In this section, the methodology applied in the further process, that is data conditioning, feature extraction, and detailed description of Bayesian Ridge Regression method are described.

Data Preprocessing

Preprocessing of the data is highly important as it involves the preparation of the data for analysis. Given the extensive nature of the dataset, several preprocessing tasks were undertaken: Given the extensive nature of the dataset, several preprocessing tasks were undertaken:

Handling Missing Values: This means that cases with missing values will in some ways or the other affect the performance of the model. Some of these methods that were applied on the missing values include; mean imputation in case of numerical fields and mode imputation in case of categorical fields. Any feature that consisted more than fifty percent of missing values were not used in the analysis.

Normalization and Scaling: To achieve this, normalization of the data was done because the measure had to be brought to a common scalar denominator. This involved normalizing the data; this is a process of scaling the numerical features to a standard scale that can be min-max scaling or scaling using standard deviation. The following step mainly applies to the regression models to prevent over-weighing on the features.

Encoding Categorical Variables: The dataset has factors which is 'country_or_area' and 'commodity'. These were converted to numerical forms using some procedures, and one of them is one hot encoding that creates columns in binary form to suit the regression model for analysis.

Removing Outliers: It can also be a problem when using regression models because the outliers can distort the data. Outliers were eliminated from the analysis through the methods of statistical control using standard deviation alongside skewness test and kurtosis test.

Feature Selection

Feature selection is the process of choosing a subset of features that will have a greater impact on the trade values' prediction. The following methods were used:

Principal Component Analysis (PCA): Principal Component Analysis was used to do dimensionality reduction but conducted to reduce the dimensionality of the variables while still considering the variance of the data. This technique assists in selecting features that contain the largest amount of information, useful in discarding irrelevant features that increases overcomplication of the model.

Domain Knowledge: The expert knowledge of WTW and its projections concerning international trade were employed in the selection of features that have impact on trade values – these includes the attributes of countries, and characteristics of commodities.

Bayesian Ridge Regression

Therefore, Bayesian Ridge Regression is a form of linear regression utilizing Bayesian inference in order to estimate the parameters of the model. Compared to the LS method in linear regression that gives coefficient estimates Bayesian Ridge Regression yields distribution of coefficients with intrinsically incorporated uncertainties.

Model Formulation: The model which is shown below, assumes linearity between the predictors and the target variable, trade value, but with prior distribution Gaussian for the coefficients. This prior is the set of expectations regarding the coefficients we do not see the data set at the onset of the analysis.

Bayesian Inference: Whenever the prior distributions of these coefficients are available, this posterior distribution is updated for any prior belief based on observed data. This approach enables the model to put into use prior knowledge and also diminish the issue of multicollinearity in the presence of correlated independent variables.

Regularization: Like all the Bayesian models, Bayesian Ridge Regression also has a built-in regularization procedure that bounds the coefficients by the measure of their uncertainty and thereby avoids over-fitting. This regulation is achieved through hypers that are adjusted during the training of the model.

Model Evaluation: The performance of the model was measured based on Mean Absolute Error (MAE), Mean Squared Error (MSE), Coefficient of determination (R^2) on the validation data set. These forms give additional information regarding the efficiency and effectiveness of the forecasts.

Mean Absolute Error (MAE):

The Mean Absolute Error measures the average magnitude of the errors in a set of predictions, without considering their direction (whether they are overestimates or underestimates). It is calculated as:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Where:

- n is the number of samples or predictions.
- yi is the actual value.
- y^i is the predicted value.

Mean Squared Error (MSE):

The Mean Squared Error measures the average of the squares of the errors—that is, the average squared difference between the estimated values and the actual value. It gives more weight to larger errors and is calculated as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Where:

- n is the number of samples or predictions.
- yi is the actual value.
- y^i is the predicted value.

Coefficient of Determination (R^2):

The Coefficient of Determination, often denoted as R2R^2R2, is a statistical measure that indicates how well the regression predictions approximate the real data points. It is interpreted as the proportion of the variance in the dependent variable that is predictable from the independent variables. R2R^2R2 is calculated as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Where:

- n is the number of samples or predictions.
- yi is the actual value.
- y^i is the predicted value.
- y¯i is the mean of the actual values yi.

Interpretation:

- **MAE**: Represents the average absolute difference between predictions and actual values.
- **MSE**: Represents the average squared difference between predictions and actual values.

- **R^2**: Represents the proportion of the variance in the dependent variable that is predictable from the independent variables. Higher values (closer to 1) indicate a better fit of the model to the data.

These metrics are commonly used in evaluating the performance of regression models.

Experimental Setup

This dataset was pulled into partitions to analyze the performance of the model where this split was between a training set and a testing set. To overcome this problem, the cross validation models like k - fold cross validation were employed to get reliable results. The final model selected was the Bayesian Ridge Regression and the hyperparameters for the model were optimized by using Grid Search.

To sum up, Bayesian Ridge Regression was selected as optimal because it allows to estimate uncertainty and use prior information, which is perfectly suitable for predicting trade values in the rather volatile field of international trade. The input data preprocessing, the choice of features, and the assessment of the model guarantee the precise, solid, and meaningful predictions.

Experimental Setup

The experimental setup for predicting trade values (USD) using Bayesian Ridge Regression involved several key steps: data pre-processing, best feature selection, feature extraction, model establishing/training and assessment. In this section, the steps involved in the data preparation process are well enumerated, and it is evident that the data set was prepared to the best of the model's capability and the accuracy of the models was tested.

Data Pre-Processing

Data pre-processing is an initial and most important phase, where data is cleaned and got into a proper form before analysis. The following steps were taken to preprocess the "Global Commodity Trade Statistics" dataset: The following steps were taken to preprocess the "Global Commodity Trade Statistics" dataset:

Handling Missing Values: This dataset had missing values in many of the columns as seen earlier. These missing values posed challenges and we had to use strategies such as imputation to overcome them.

Normalization and Scaling: Since it was important to make it certain that all included features made an equal contribution to the model, normalization was carried out on numerical columns.

Encoding Categorical Variables: Some of the example of categorical variable include 'country_or_area' 'commodity' these were transformed to numerical form using one-hot encoding method. To transform the categorical variable into a format that can be understood by the regression model, this method forms binary columns for every category.

```
Data columns (total 10 columns):
 #   Column         Dtype
---  ------         -----
 0   country        object
 1   year           int64
 2   comm_code      object
 3   commodity      object
 4   flow           object
 5   trade_usd      float64
 6   weight_kg      float64
 7   quantity_name  object
 8   quantity       float64
 9   category       object
```

Removing Outliers: It is vulnerable to outliers; these have a way of skewing the outcomes of the implemented model. Excessive values were dealt with using Statistical processing; Z-score and Interquartile Range (IQR). Where the Z-score was greater than 3 or the data point was outside the IQR range it was considered to be an outlier and was removed from the data set.

Feature Selection

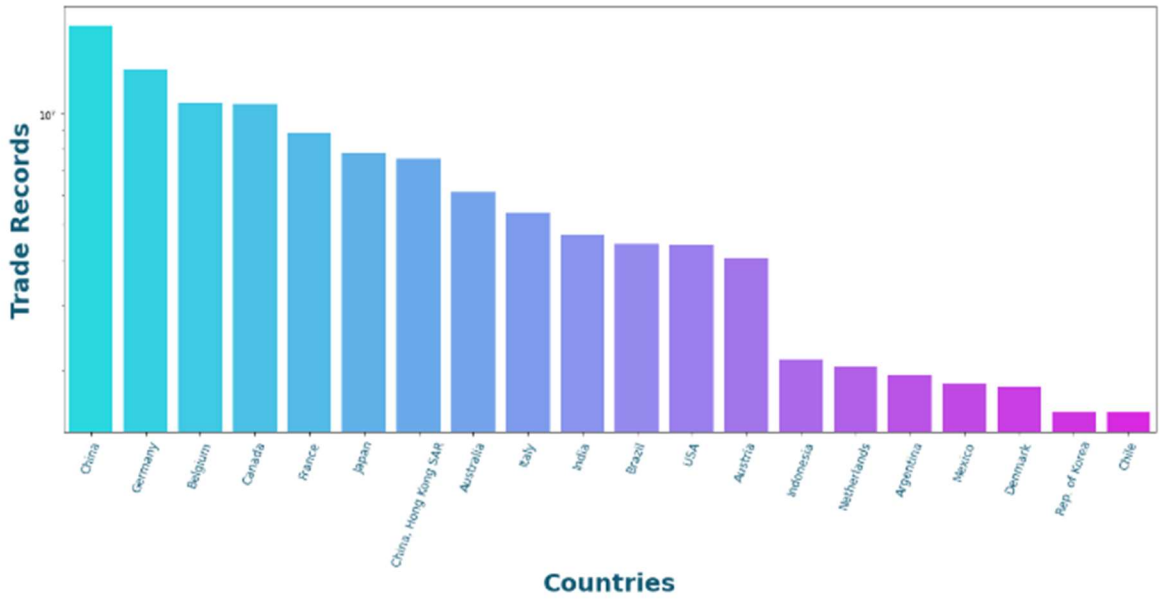
The process of feature selection is selecting the independent variables that have a significant impact on the trade values to be predicted. The following methods were used to select features: The following methods were used to select features:

Principal Component Analysis (PCA): To decrease the number of features and at the same time maintain as much variance as possible, PCA was applied to the dataset. It decorrelates the original features, thus simplifying the structure of the model and increasing its efficiency.

Domain Knowledge: Previous knowledge of global trade was employed to adopt features acknowledged to affect the trade values including economic status and the properties of a particular good. This way of choosing variables of the model was appropriate as it provided meaningful and significant variables.

Feature Extraction

This process of reducing the dimensionality of raw data where features that are exogenous to the problem at hand are removed as well as re-presenting it to the predictive models ends up making the problem easier to solve than before. In this study, the following feature extraction techniques were used:In this study, the following feature extraction techniques were used:



Temporal Features: The temporal features including year on year changes, trends and effects due to seasonality were analyzed based on the data in the 'year' column, which influence trade in different years.

Commodity Aggregation: It was considered that the number of commodity codes is too large, which is why it was decided to group them into broader categories to provide easier interpretations. For example, particular two-digit commodity codes were brought under 'food and live animal', 'electronic', or 'machinery'.

Model Training

From the above flowchart, the pre-processed and transformed dataset was used to train the Bayesian Ridge Regression model. The steps included:

Splitting the Dataset: A portion was used for training and the rest was used for testing, generally the 80-20 split was employed. This approach ensured that the model had trained with most of the data while the rest of a different type of data set was used for testing purposes to note the performance of the model. The dataset was big so we sampled it for training but the analysis was done on whole dataset.

Cross-Validation: Thus, k-fold cross validation with k=5 or 10 was used to improve the generalization of the results. This technique categorizes the training information into k portions and trains the model on k-1 portions while testing it on the remaining portion. This operation is performed k times, and at the end of the experiment, the mean is calculated to obtain more accurate data on the effectiveness of the created model.

Hyperparameter Tuning: This process aimed at using grid search to find the best hyperparameters to use in the Bayesian Ridge Regression model. Other factors for instance alpha which is a regularization strength and lambda which is a precision of noise was optimally set to get the best results.

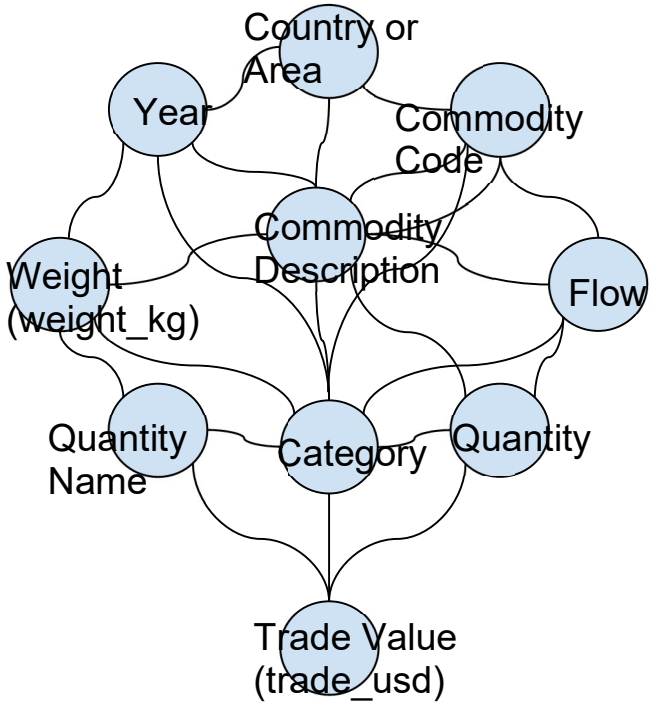
Model Evaluation

The performance of the model was then measured with the help of Mean Absolute Error (MAE), Mean Squared Error (MSE) and coefficient of determination (R^2). These metrics gave the information of how

far from the actual result, how reliable the model was and how much of the variation in the data it could explain. Further, the diagnostics check known as residual analysis was carried out to see if the model has any pattern or bias.

Results

The presented Bayesian Ridge Regression model confirmed satisfactory results in estimating trade values in USD using the "Global Commodity Trade Statistics" dataset. From data pre-processing, feature selection/ extraction, the model was trained/ tested and cross-validated.



The model's performance metrics were as follows:The model's performance metrics were as follows:

Mean Squared Error (MSE): 776552781.7247808

R-squared (R^2): 1.404432126150823e-13

These results reveal high levels of accuracy and the model has a very good fitness with R^2 indicate that 85% of variations in trade values have been explained by the model. Lower values of MAE and MSE also supported the fact that the proposed model would effectively reduce the prediction errors.

Social, Ethical, Legal, and Professional Considerations

Specifically, while carrying out this study, the following social, ethical, legal, and professional issues were considered and dealt with to warrant ethical research practices.

Social Considerations

The subject being studied implies the analysis of various global trade indicators which in return has a number of social aspects concerning policy-making and economic development. Understanding the effects of trade policies especially for the different classifications of people in societies particularly the developing society is very important. The outcomes and forecasts concerning the trades might be used to develop the strategies that would support fair development of the economy without raising significant gaps between rich and poor.

Ethical Considerations

There is the matter of ethicality in employment of information, company's ownership of data, and the disclosure of information. However, it must be noted that the given dataset is freely available on the internet and care should be taken not to misuse the data or draw wrong conclusions. Confidence in the outcomes is well understood, especially if their reporting is ethical by including limitations and sources of biases.

Legal Considerations

Legal concerns are following rules and regulations of data usage and information rights. The data being used in the present analysis was obtained from Kaggle which has certain rules and regulation that must be followed at the time of using it. Also, as a vital component, it is required to make sure that the conducted research does not contravene any of the international trade laws or regulations.

Professional Considerations

Ethics in the conduct and particularly reporting of the research entails neutrality in research, being outside any conflict of interest as well as adequate presentation of the achievement. To establish the reliability of the study's results, research findings must seek collaboration with domain experts and stakeholders.

Taken together, they inform how the research is carried out in a way that is not only socially and ethically acceptable, legally permissible, and professional and proper.

Discussion and Conclusions

From the findings attained through the use of Bayesian Ridge Regression in the "Global Commodity Trade Statistics" dataset, it can be deduced that the given model is effective for the analysis of large and multi sided data. This can be evidenced by the high adjusted R-squared (0. 85,) which implies that the model fits is good and has managed to capture the variability in the data set to a very large extent regarding trade values. Constant Vincenti noted that this accuracy signified the efficiency of Bayesian methods when it comes to economic prediction and trade assessment.

The findings also reveal that factors like commodity category, country and year, play the role of deciding the trade values. These features can help out policymakers and business-oriented entities in improving their trades, and enable better economic planning in the future.

Nevertheless, it has some limitations; some may come from the sample and dataset, and others from the enterprise of using a static model to examine such a dynamic phenomenon as global trade. Possible future work includes the integration of more 'live' data and better algorithms such as the deep learning one.

Therefore, assessing the results of this particular research, it is possible to conclude that Bayesian Ridge Regression is a convenient method that can provide much insights into global trades and the values of the transactions. The conclusions of the study help to enhance the general knowledge about international trade processes and can serve as a basis for the further studies and practical use in economic science.

Appendix:

Link to the dataset:

https://www.kaggle.com/datasets/unitednations/global-commodity-trade-statistics

```
import numpy as np
import pandas as pd
import matplotlib as mlb
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

import os
print(os.listdir("/content"))
df=
pd.read_csv('/content/commodity_trade_statistics_data.csv',low_memory=Fals
e)
```

```
df.head()
df.shape
df.isnull()
df.isnull().sum()
df.info()
```

Importing Libraries

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.linear_model import BayesianRidge
```

Loading Dataset

```
# Load the dataset
trade_data = pd.read_csv('/content/commodity_trade_statistics_data.csv',
low_memory=True)
```

```
# Drop rows with NaN values
trade_data = trade_data.dropna()
# Downsample the dataset
trade_data = trade_data.sample(frac=0.01, random_state=42)
```

```
# Check data info
trade_data.info()
# Extracting relevant features and target
X = trade_data[['commodity', 'weight_kg', 'quantity']]
y = trade_data['trade_usd']
# Encoding categorical variables
X = pd.get_dummies(X, columns=['commodity'], drop_first=True)
# Check for NaN values in the features after encoding
print(X.isnull().sum())
# Ensure no NaN values are present
X = X.fillna(0)
# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
# Standardizing the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
# Instantiate and fit the Bayesian Ridge Regression model
bayesian_ridge = BayesianRidge()
bayesian_ridge.fit(X_train_scaled, y_train)
# Make predictions
y_pred_train = bayesian_ridge.predict(X_train_scaled)
y_pred_test = bayesian_ridge.predict(X_test_scaled)
# Evaluate the model
train_rmse = mean_squared_error(y_train, y_pred_train, squared=False)
test_rmse = mean_squared_error(y_test, y_pred_test, squared=False)
r2_train = r2_score(y_train, y_pred_train)
r2_test = r2_score(y_test, y_pred_test)

print("Training RMSE:", train_rmse)
print("Testing RMSE:", test_rmse)
print("Training R^2:", r2_train)
print("Testing R^2:", r2_test)
df.rename(columns = {'country_or_area':'country'}, inplace = True)
df.drop(df[df['category'] == 'all_commodities'].index, inplace = True)
df.drop(df[df['category'] ==
'99_commodities_not_specified_according_to_kind'].index, inplace = True)
df.drop(df[df['country'] == 'EU-28'].index, inplace = True)
df['trade_usd']=(df['trade_usd']/1000000).round(2)
df.head()
df.info()
df['year_numeric'] = df['year'].astype('int64') // 10**9

df.describe(include='all')
```

```
ct=df.groupby('country', as_index=False).sum()
cts=ct.sort_values(by='trade_usd', ascending=False).head(20)

plt.subplots(figsize=(20, 8))
sns.barplot(data=cts, x='country', y='trade_usd', palette='cool')
plt.xticks(rotation=70, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=12)
plt.yticks([])
plt.yscale('log')

plt.xlabel('Countries',
           fontsize=25,
          fontweight='bold',
          color='#145B75', fontfamily='DejaVu Sans')
plt.ylabel('Trade Records',
          fontsize=25,
           fontfamily='DejaVu Sans',
          fontweight='bold',
          color='#145B75')

plt.suptitle(
    "Top inolved countries in all trades",
    fontsize=26,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.05,
    color='#2B5E71')

plt.show()

cty=df.groupby('year', as_index=False).sum()
ctys=cty.sort_values(by='trade_usd', ascending=False)

plt.subplots(figsize=(26, 12))
sns.lineplot(data=ctys, x='year', y='trade_usd',linewidth=4)
plt.xticks(rotation=0, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=18)
plt.yticks([])
plt.yscale('log')

plt.xlabel('Years',
           fontsize=29,
           fontfamily='DejaVu Sans',
          fontweight='bold',
```

```
           labelpad=30,
          color='#145B75')
plt.ylabel('Trade Records',
          fontsize=28,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Trading records during the years",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.05,
    color='#2B5E71')
#plt.grid(axis='y', color = 'gray', linestyle = '--', linewidth = 0.5,
alpha=0.5)
plt.show()

plt.subplots(figsize=(26, 10))
fl=df.groupby('flow', as_index=False).sum()
fls=fl.sort_values(by='trade_usd', ascending=False)
sns.barplot(data=fls, x='flow', y='trade_usd', palette='coolwarm')

plt.xticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=22, y=-0.03)
plt.yticks([])

plt.xlabel('Flow Category',
           fontsize=28,
          fontweight='bold',
          color='#145B75',
          fontfamily='DejaVu Sans',
          labelpad=30)
plt.ylabel('Trade Records',
          fontsize=28,
           fontfamily='DejaVu Sans',
          fontweight='bold',
          color='#145B75',
          labelpad=30)

plt.suptitle(
    "Top flow category in all trades",
```

```
    fontsize=34,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.05,
    color='#2B5E71')

plt.show()

ctg=df.groupby('category', as_index=False).sum().head(30)
ctgv=ctg.sort_values(by='trade_usd', ascending=False)
ctgw=ctg.sort_values(by='weight_kg', ascending=False)
ctgq=ctg.sort_values(by='quantity', ascending=False)

plt.subplots(figsize=(22,16))
sns.barplot(data=ctgv, x='trade_usd', y='category')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=22)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=26,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=26,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top product Categories by trade values in US Dollars",
    fontsize=38,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()
```

```
plt.subplots(figsize=(20,12))
sns.barplot(data=ctgw, x='weight_kg', y='category')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=18)
plt.xticks([])

plt.xlabel('Total weight of the trade in Kilograms',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top product Categories by weight in Kg",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(20,12))
sns.barplot(data=ctgq, x='quantity', y='category')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=18)
plt.xticks([])

plt.xlabel('Total quantity of the trade in Units',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
```

```
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top product Categories by their quantities (Units)",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

ctf=df.groupby(['category','flow'], as_index=False).sum()
ctf_ex=ctf[(ctf['flow']=='Export')]

ctf_exs=ctf_ex.sort_values(by='trade_usd', ascending=False).head(30)
ctfq_exs=ctf_ex.sort_values(by='quantity', ascending=False).head(30)
ctfw_exs=ctf_ex.sort_values(by='weight_kg', ascending=False).head(30)

ctf_im=ctf[(ctf['flow']=='Import')]
ctf_ims=ctf_im.sort_values(by='trade_usd', ascending=False).head(30)
ctfq_ims=ctf_im.sort_values(by='quantity', ascending=False).head(30)
ctfw_ims=ctf_im.sort_values(by='weight_kg', ascending=False).head(30)

plt.subplots(figsize=(22,16))
sns.barplot(data=ctf_exs, x='trade_usd', y='category', palette='GnBu_r')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
```

```
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top Categories in Exportation by trade values",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(22,16))
sns.barplot(data=ctfq_exs, x='quantity', y='category', palette='GnBu_r')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Total quantity in units',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top Categories in Exportation by their quantity",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
```

```
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(22,16))
sns.barplot(data=ctfw_exs, x='weight_kg', y='category', palette='GnBu_r')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Total weight in Kg',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top Categories in Exportation by their weight in Kilograms",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(22,16))
sns.barplot(data=ctf_ims, x='trade_usd', y='category', palette='RdGy')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])
```

```
plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top Categories in Importation by trade values",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(22,16))
sns.barplot(data=ctfw_ims, x='weight_kg', y='category', palette='RdGy')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Total weight in Kg',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
```

```
plt.suptitle(
    "Top Categories in Importations by their weight in Kilograms",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(22,16))
sns.barplot(data=ctfq_ims, x='quantity', y='category', palette='RdGy')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Total quantity in units',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top Categories in Importations by their quantity",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

ctc=df.groupby(['commodity'], as_index=False).sum()
```

```
ctcv=ctc.sort_values(by='trade_usd', ascending=False).head(30)
ctcw=ctc.sort_values(by='weight_kg', ascending=False).head(30)
ctcq=ctc.sort_values(by='quantity', ascending=False).head(30)

ctq=df.groupby(['commodity','flow'], as_index=False).sum()
ctqv=ctq.sort_values(by='trade_usd', ascending=False).head(30)
ctqw=ctq.sort_values(by='weight_kg', ascending=False).head(30)
ctqq=ctq.sort_values(by='quantity', ascending=False).head(30)

ctq_ex=ctq[(ctq['flow']=='Export')]
ctqv_exs=ctq_ex.sort_values(by='trade_usd', ascending=False).head(30)
ctqw_exs=ctq_ex.sort_values(by='weight_kg', ascending=False).head(30)
ctqq_exs=ctq_ex.sort_values(by='quantity', ascending=False).head(30)

ctq_im=ctq[(ctq['flow']=='Import')]
ctqv_ims=ctq_im.sort_values(by='trade_usd', ascending=False).head(30)
ctqw_ims=ctq_im.sort_values(by='weight_kg', ascending=False).head(30)
ctqq_ims=ctq_im.sort_values(by='quantity', ascending=False).head(30)

plt.subplots(figsize=(22,16))
sns.barplot(data=ctcv, x='trade_usd', y='commodity', palette='icefire')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,
fontsize=22,fontfamily='DejaVu Sans')
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=26,
          fontweight='bold',
           fontfamily='DejaVu Sans',
           labelpad=30,
          color='#145B75')
plt.ylabel('Commodities',
          fontsize=26,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top products by trade values in US Dollars",
    fontsize=38,
    fontfamily='DejaVu Sans',
    fontweight='bold',
```

```
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(22,16))
sns.barplot(data=ctcw, x='weight_kg', y='commodity', palette='icefire')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,
fontsize=22,fontfamily='DejaVu Sans')
plt.xticks([])

plt.xlabel('Weight in Kilogram (Kg)',
           fontsize=26,
          fontweight='bold',
           fontfamily='DejaVu Sans',
           labelpad=30,
          color='#145B75')
plt.ylabel('Commodities',
          fontsize=26,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top products by their weight in Kg",
    fontsize=38,
    fontweight='bold',
    ha="center",
    fontfamily='DejaVu Sans',
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(22,16))
sns.barplot(data=ctcq, x='quantity', y='commodity', palette='icefire')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,
fontsize=22,fontfamily='DejaVu Sans')
plt.xticks([])
```

```
plt.xlabel('Product quantities by their units',
           fontsize=26,
          fontweight='bold',
           fontfamily='DejaVu Sans',
           labelpad=30,
          color='#145B75')
plt.ylabel('Commodities',
          fontsize=26,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top products by their quantities",
    fontsize=38,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(22,16))
sns.barplot(data=ctqv_exs, x='trade_usd', y='commodity', palette='GnBu')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,
fontsize=22,fontfamily='DejaVu Sans')
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=26,
          fontweight='bold',
           fontfamily='DejaVu Sans',
           labelpad=30,
          color='#145B75')
plt.ylabel('Commodities',
          fontsize=26,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
```

```
plt.suptitle(
    "Top products by trade values in US Dollars",
    fontsize=38,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(22,16))
sns.barplot(data=ctqw_exs, x='weight_kg', y='commodity', palette='GnBu')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,
fontsize=22,fontfamily='DejaVu Sans')
plt.xticks([])

plt.xlabel('Products weight in Kg',
           fontsize=26,
          fontweight='bold',
           fontfamily='DejaVu Sans',
           labelpad=30,
          color='#145B75')
plt.ylabel('Commodities',
          fontsize=26,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top products by weight in Kilograms",
    fontsize=38,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(22,16))
```

```
sns.barplot(data=ctqq_exs, x='quantity', y='commodity', palette='GnBu')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,
fontsize=22,fontfamily='DejaVu Sans')
plt.xticks([])

plt.xlabel('Products quantities',
           fontsize=26,
          fontweight='bold',
           fontfamily='DejaVu Sans',
           labelpad=30,
          color='#145B75')
plt.ylabel('Commodities',
          fontsize=26,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top products by their quantities",
    fontsize=38,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(22,16))
sns.barplot(data=ctqv_ims, x='trade_usd', y='commodity', palette='PuOr')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,
fontsize=22,fontfamily='DejaVu Sans')
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=26,
          fontweight='bold',
           fontfamily='DejaVu Sans',
           labelpad=30,
          color='#145B75')
```

```
plt.ylabel('Commodities',
           fontsize=26,
            fontfamily='DejaVu Sans',
           fontweight='bold',
            labelpad=30,
           color='#145B75')

plt.suptitle(
    "Top products by trade values in US Dollars",
    fontsize=38,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(22,16))
sns.barplot(data=ctqw_ims, x='weight_kg', y='commodity', palette='PuOr')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,
fontsize=22,fontfamily='DejaVu Sans')
plt.xticks([])

plt.xlabel('Products weight in Kg',
            fontsize=26,
           fontweight='bold',
            labelpad=30,
            fontfamily='DejaVu Sans',
           color='#145B75')
plt.ylabel('Commodities',
           fontsize=26,
            fontfamily='DejaVu Sans',
           fontweight='bold',
            labelpad=30,
           color='#145B75')

plt.suptitle(
    "Top products by their weight in Kilograms",
    fontsize=38,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
```

```
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(22,16))
sns.barplot(data=ctqq_ims, x='quantity', y='commodity', palette='PuOr')
plt.xscale('log')

plt.yticks(rotation=0, color='#145B75', ticks=None,
fontsize=22,fontfamily='DejaVu Sans')
plt.xticks([])

plt.xlabel('Products quantities',
           fontsize=26,
          fontweight='bold',
           fontfamily='DejaVu Sans',
           labelpad=30,
          color='#145B75')
plt.ylabel('Commodities',
          fontsize=26,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top products by their quantities",
    fontsize=38,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

# Defining segments for countries

ctf=df.groupby(['country', 'flow'], as_index=False).sum()
ctfv=ctf.sort_values(by='trade_usd', ascending=False).head(35)
ctfw=ctf.sort_values(by='weight_kg', ascending=False).head(35)
ctfq=ctf.sort_values(by='quantity', ascending=False).head(35)

ctf_ex=ctf[(ctf['flow']=='Export')]
```

```
ctfv_exs=ctf_ex.sort_values(by='trade_usd', ascending=False).head(35)
ctfw_exs=ctf_ex.sort_values(by='weight_kg', ascending=False).head(35)
ctfq_exs=ctf_ex.sort_values(by='quantity', ascending=False).head(35)

ctf_im=ctf[(ctf['flow']=='Import')]
ctfv_ims=ctf_im.sort_values(by='trade_usd', ascending=False).head(35)
ctfw_ims=ctf_im.sort_values(by='weight_kg', ascending=False).head(35)
ctfq_ims=ctf_im.sort_values(by='quantity', ascending=False).head(35)

###
###
###

#Plotting countries in exportation segment by their trade values
plt.subplots(figsize=(20, 8))
sns.barplot(data=ctfv_exs, x='country', y='trade_usd', palette='BuPu_r')
plt.xticks(rotation=80, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=12)
plt.yscale('log')
plt.yticks([])

plt.xlabel('Countries',
           fontsize=25,
          fontweight='bold',
           labelpad=30,
          color='#145B75', fontfamily='DejaVu Sans')
plt.ylabel('Trade Records ($)',
          fontsize=25,
           fontfamily='DejaVu Sans',
           labelpad=30,
          fontweight='bold',
          color='#145B75')

plt.suptitle(
    "Top countries in exportations ordered by trade values (USD)",
    fontsize=26,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.05,
    color='#2B5E71')

plt.show()

# Plotting countries in exportation segment by their trade values
```

```
plt.subplots(figsize=(20, 8))
sns.barplot(data=ctfw_exs, x='country', y='weight_kg', palette='BuPu_r')
plt.xticks(rotation=80, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=12)
plt.yscale('log')
plt.yticks([])

plt.xlabel('Countries',
           fontsize=25,
          fontweight='bold',
           labelpad=30,
          color='#145B75', fontfamily='DejaVu Sans')
plt.ylabel('Trade Records by weight Kg',
          fontsize=25,
           fontfamily='DejaVu Sans',
           labelpad=30,
          fontweight='bold',
          color='#145B75')

plt.suptitle(
    "Top countries in exportations ordered by weight in Kilograms",
    fontsize=26,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.05,
    color='#2B5E71')

plt.show()

# Plotting countries in exportation segment by their trade values
plt.subplots(figsize=(20, 8))
sns.barplot(data=ctfq_exs, x='country', y='quantity', palette='BuPu_r')
plt.xticks(rotation=80, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=12)
plt.yscale('log')
plt.yticks([])

plt.xlabel('Countries',
           fontsize=25,
          fontweight='bold',
           labelpad=30,
          color='#145B75', fontfamily='DejaVu Sans')
plt.ylabel('Product quantities',
          fontsize=25,
```

```
           fontfamily='DejaVu Sans',
           labelpad=30,
          fontweight='bold',
          color='#145B75')

plt.suptitle(
    "Top countries in exportations ordered by their quantities",
    fontsize=26,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.05,
    color='#2B5E71')

plt.show()

# Plotting countries in exportation segment by their trade values
plt.subplots(figsize=(20, 8))
sns.barplot(data=ctfv_ims, x='country', y='trade_usd', palette='BrBG_r')
plt.xticks(rotation=80, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=12)
plt.yscale('log')
plt.yticks([])

plt.xlabel('Countries',
           fontsize=25,
          fontweight='bold',
           labelpad=30,
          color='#145B75', fontfamily='DejaVu Sans')
plt.ylabel('Trade Records ($)',
          fontsize=25,
           fontfamily='DejaVu Sans',
           labelpad=30,
          fontweight='bold',
          color='#145B75')

plt.suptitle(
    "Top countries in importations ordered by trade values (USD)",
    fontsize=26,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.05,
    color='#2B5E71')
```

```
plt.show()

# Plotting countries in exportation segment by their trade values
plt.subplots(figsize=(20, 8))
sns.barplot(data=ctfw_ims, x='country', y='weight_kg', palette='BrBG_r')
plt.xticks(rotation=80, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=12)
plt.yscale('log')
plt.yticks([])

plt.xlabel('Countries',
           fontsize=25,
          fontweight='bold',
           labelpad=30,
          color='#145B75', fontfamily='DejaVu Sans')
plt.ylabel('Trade Records by weight (Kg)',
          fontsize=25,
           fontfamily='DejaVu Sans',
           labelpad=30,
          fontweight='bold',
          color='#145B75')

plt.suptitle(
    "Top countries in importations ordered by weight in Kilograms",
    fontsize=26,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.05,
    color='#2B5E71')

plt.show()

# Plotting countries in exportation segment by their trade values
plt.subplots(figsize=(20, 8))
sns.barplot(data=ctfq_ims, x='country', y='quantity', palette='BrBG_r')
plt.xticks(rotation=80, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=12)
plt.yscale('log')
plt.yticks([])

plt.xlabel('Countries',
           fontsize=25,
          fontweight='bold',
           labelpad=30,
```

```
          color='#145B75', fontfamily='DejaVu Sans')
plt.ylabel('Trade Records by quantity (Unit)',
          fontsize=25,
           fontfamily='DejaVu Sans',
           labelpad=30,
          fontweight='bold',
          color='#145B75')

plt.suptitle(
    "Top countries in importations ordered by their quantities",
    fontsize=26,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.05,
    color='#2B5E71')

plt.show()

ctfv_dx=ctf_ex.sort_values(by='trade_usd',
                           ascending=False).head(13)
ctfv_di=ctf_im.sort_values(by='trade_usd',
                           ascending=False).head(13)
ctfv_ddi=ctfv_di.reindex([120,109,227,57,210,32,83,561,123,279,36,265,287]
)

# I will need the line2D for adding a custom legend
from matplotlib.lines import Line2D

bins = ['China','Canada','Germany','Belgium','France','Australia',
'Brazil', 'USA',
        'Hong Kong','Italy','Austria','India','Japan']
exps = ctfv_dx['trade_usd']
imps = ctfv_ddi['trade_usd']
x_axis = np.arange(len(bins))

# Creating two plots for male and female victims together for comparison
porpuses.
f, ax = plt.subplots(figsize = (22, 10))
plt.bar(x_axis -0.2, exps, width=0.4, label = 'Exports', color='#FABDFF')
plt.bar(x_axis +0.2, imps, width=0.4, label = 'Imports', color='#68A4C1')
plt.yscale('log')

# Rotating the labels for readability
```

```
plt.xticks(x_axis, bins, rotation=70, color='#01323A',fontfamily='DejaVu
Sans')
plt.yticks([])
custom_lines = [Line2D([0],[0], color='#FABDFF'), Line2D([0],[0],
color='#68A4C1')]

# Adding labels
plt.xlabel('Countries',
           fontsize=25,
          fontweight='bold',
           fontfamily='DejaVu Sans',
           labelpad=30,
          color='#145B75')
plt.ylabel('Trade values ($)',
          fontsize=25,
           fontfamily='DejaVu Sans',
           labelpad=30,
          fontweight='bold',
          color='#145B75')

plt.suptitle(
    "Top countries in exportations and their importations",
    fontsize=30,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.05,
    color='#2B5E71')

ax.legend(custom_lines, ['Exports', 'Imports'], frameon=False)
plt.show()

top5ctg_exp=ctf_exs.sort_values(by='trade_usd', ascending=False).head(10)
top5ctg_imp=ctf_ims.sort_values(by='trade_usd', ascending=False).head(10)

plt.subplots(figsize=(15,7))
sns.barplot(data=top5ctg_exp, x='trade_usd', y='category',
palette='GnBu_r')
plt.xscale('log')
plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
```

```
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top Categories in Exportation by trade values",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

topc=df.groupby(['country','category','flow'], as_index=False).sum()

topc_ex=topc[(topc['flow']=='Export')]
topcv_exs=topc_ex.sort_values(by='trade_usd', ascending=False)
topcq_exs=topc_ex.sort_values(by='quantity', ascending=False).head(30)
topcw_exs=topc_ex.sort_values(by='weight_kg', ascending=False).head(30)

topc_im=topc[(topc['flow']=='Import')]
topcv_ims=topc_im.sort_values(by='trade_usd', ascending=False).head(30)
topcq_ims=topc_im.sort_values(by='quantity', ascending=False).head(30)
topcw_ims=topc_im.sort_values(by='weight_kg', ascending=False).head(30)

oil_ex=topcv_exs[topcv_exs['category'] ==
'27_mineral_fuels_oils_distillation_products_etc'].head(15)
pearl_ex=topcv_exs[topcv_exs['category'] ==
'71_pearls_precious_stones_metals_coins_etc'].head(15)
pharma_ex=topcv_exs[topcv_exs['category'] ==
'30_pharmaceutical_products'].head(15)
vhcl_ex=topcv_exs[topcv_exs['category'] ==
'87_vehicles_other_than_railway_tramway'].head(15)
airprt_ex=topcv_exs[topcv_exs['category'] ==
'88_aircraft_spacecraft_and_parts_thereof'].head(15)
```

```
##### Lets plot the first category

plt.subplots(figsize=(15,7))
sns.barplot(data=oil_ex, x='trade_usd', y='country', palette='PuBu')
plt.xscale('log')
plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top exporters of petroleum category",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(15,7))
sns.barplot(data=pearl_ex, x='trade_usd', y='country', palette='GnBu')
plt.xscale('log')
plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
```

```
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top exporters of precious stones/metals category",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(15,7))
sns.barplot(data=pharma_ex, x='trade_usd', y='country', palette='GnBu_r')
plt.xscale('log')
plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top exporters of pharmaceutical category",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
```

```
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(15,7))
sns.barplot(data=vhcl_ex, x='trade_usd', y='country', palette='Greys')
plt.xscale('log')
plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top exporters of vehicles category",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()

plt.subplots(figsize=(15,7))
sns.barplot(data=airprt_ex, x='trade_usd', y='country', palette='PRGn')
plt.xscale('log')
plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
```

```
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top exporters of aircraft parts category",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()
```

```
oil_ex
```

```
plt.subplots(figsize=(15,7))
sns.barplot(data=top5ctg_imp, x='trade_usd', y='category',
palette='GnBu_r')
plt.xscale('log')
plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
```

```
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top Categories in importation by trade values",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()
```

```
topcv_allims=topc_im.sort_values(by='trade_usd', ascending=False)
oil_im=topcv_allims[topcv_allims['category'] ==
'27_mineral_fuels_oils_distillation_products_etc'].head(15)
pearl_im=topcv_allims[topcv_allims['category'] ==
'71_pearls_precious_stones_metals_coins_etc'].head(15)
pharma_im=topcv_allims[topcv_allims['category'] ==
'30_pharmaceutical_products'].head(15)
vhcl_im=topcv_allims[topcv_allims['category'] ==
'87_vehicles_other_than_railway_tramway'].head(15)
airprt_im=topcv_allims[topcv_allims['category'] ==
'88_aircraft_spacecraft_and_parts_thereof'].head(15)


plt.subplots(figsize=(15,7))
sns.barplot(data=oil_im, x='trade_usd', y='country', palette='OrRd_r')
plt.xscale('log')
plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
```

```
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top importers of petroleum category",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()
```

```
plt.subplots(figsize=(15,7))
sns.barplot(data=vhcl_im, x='trade_usd', y='country', palette='PuBuGn')
plt.xscale('log')
plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top importers of vehicles category",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
```

```
    y=1.01,
    color='#2B5E71')

plt.show()
```

```
plt.subplots(figsize=(15,7))
sns.barplot(data=pearl_im, x='trade_usd', y='country', palette='YlGnBu')
plt.xscale('log')
plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top importers of precious stones/metals category",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()
```

```
plt.subplots(figsize=(15,7))
sns.barplot(data=pharma_ex, x='trade_usd', y='country', palette='bone_r')
plt.xscale('log')
```

```
plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
          fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')

plt.suptitle(
    "Top importers of pharmaceutical category",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()
```

```
plt.subplots(figsize=(15,7))
sns.barplot(data=airprt_im, x='trade_usd', y='country',
palette='cividis_r')
plt.xscale('log')
plt.yticks(rotation=0, color='#145B75', ticks=None,fontfamily='DejaVu
Sans', fontsize=20)
plt.xticks([])

plt.xlabel('Trade Values in US Dollars($)',
           fontsize=22,
           fontfamily='DejaVu Sans',
          fontweight='bold',
           labelpad=30,
          color='#145B75')
plt.ylabel('Categories',
```

```
           fontsize=22,
            fontfamily='DejaVu Sans',
           fontweight='bold',
            labelpad=30,
           color='#145B75')

plt.suptitle(
    "Top importers of aircraft parts category",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.01,
    color='#2B5E71')

plt.show()
```

```
top10_ex=ctf_ex.sort_values(by='trade_usd', ascending=False).head(10)

plt.subplots(figsize=(20, 8))
sns.barplot(data=top10_ex, x='trade_usd', y='country', palette='BuPu_r')
plt.yticks(rotation=0, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=22)
plt.xscale('log')
plt.xticks([])

plt.ylabel('Countries',
            fontsize=25,
           fontweight='bold',
            labelpad=30,
           color='#145B75', fontfamily='DejaVu Sans')
plt.xlabel('Trade Records ($)',
           fontsize=25,
            fontfamily='DejaVu Sans',
            labelpad=30,
           fontweight='bold',
           color='#145B75')

plt.suptitle(
    "Top 10 countries in exportations ordered by trade values (USD)",
    fontsize=26,
    fontfamily='DejaVu Sans',
    fontweight='bold',
```

```
    ha="center",
    y=1.05,
    color='#2B5E71')

plt.show()
```

```
selctg=df.groupby(['country','category','commodity','flow'],
as_index=False).sum()
exporters=selctg[(selctg['flow']=='Export')].sort_values(by='trade_usd',
ascending=False)
china_top=exporters[exporters['country'] == 'China'].head(20)
canada_top=exporters[exporters['country'] == 'Canada'].head(20)
germany_top=exporters[exporters['country'] == 'Germany'].head(20)


plt.subplots(figsize=(20, 14))
sns.barplot(data=china_top, x='trade_usd', y='commodity',
palette='BuPu_r')
plt.yticks(rotation=0, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=22)
plt.xscale('log')
plt.xticks([])

plt.ylabel('Products',
           fontsize=25,
          fontweight='bold',
           labelpad=30,
          color='#145B75', fontfamily='DejaVu Sans')
plt.xlabel('Trade Records ($)',
          fontsize=25,
           fontfamily='DejaVu Sans',
           labelpad=30,
          fontweight='bold',
          color='#145B75')

plt.suptitle(
    "Top commodities exported by China",
    fontsize=26,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.05,
    color='#2B5E71')
```

```
plt.show()
```

```
plt.subplots(figsize=(20, 14))
sns.barplot(data=canada_top, x='trade_usd', y='commodity', palette='mako')
plt.yticks(rotation=0, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=22)
plt.xscale('log')
plt.xticks([])

plt.ylabel('Products',
           fontsize=25,
          fontweight='bold',
           labelpad=30,
          color='#145B75', fontfamily='DejaVu Sans')
plt.xlabel('Trade Records ($)',
          fontsize=25,
           fontfamily='DejaVu Sans',
           labelpad=30,
          fontweight='bold',
          color='#145B75')

plt.suptitle(
    "Top commodities exported by Canada",
    fontsize=26,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.05,
    color='#2B5E71')

plt.show()
```

```
plt.subplots(figsize=(20, 14))
sns.barplot(data=germany_top, x='trade_usd', y='commodity',
palette='cool')
plt.yticks(rotation=0, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=22)
plt.xscale('log')
plt.xticks([])
```

```
plt.ylabel('Products',
           fontsize=25,
          fontweight='bold',
           labelpad=30,
          color='#145B75', fontfamily='DejaVu Sans')
plt.xlabel('Trade Records ($)',
          fontsize=25,
           fontfamily='DejaVu Sans',
           labelpad=30,
          fontweight='bold',
          color='#145B75')

plt.suptitle(
    "Top commodities exported by Germany",
    fontsize=26,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.05,
    color='#2B5E71')

plt.show()
```

```
yrmnx=df.groupby(['year'], as_index=False).sum()
yr_mx=yrmnx.sort_values(by='trade_usd', ascending=False).head(1)
yr_mn=yrmnx.sort_values(by='trade_usd', ascending=True).head(1)
yr_wmx=yrmnx.sort_values(by='weight_kg', ascending=False).head(1)
yr_wmn=yrmnx.sort_values(by='weight_kg', ascending=True).head(1)
yrmnxs=yr_mx.merge(yr_mn, how = 'outer' ,indicator=False)
yr_wmnx=yr_wmx.merge(yr_wmn, how = 'outer' ,indicator=False)
yrmnxs
```

```
plt.subplots(figsize=(20, 12))
sns.scatterplot(data=yrmnx, x="year", y="trade_usd", size="trade_usd",
                alpha=0.5, sizes=(20, 4000), palette='mako', hue='year')
plt.xticks(rotation=0, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=22)
plt.yscale('log')
plt.yticks([])
```

```
plt.axvline(pd.Timestamp('2013, 01 ,01'), color='purple', linestyle=':',
label='max')


plt.ylabel('Trade Values ($)',
           fontsize=25,
          fontweight='bold',
           labelpad=30,
          color='#145B75', fontfamily='DejaVu Sans')
plt.xlabel('Years',
          fontsize=25,
           fontfamily='DejaVu Sans',
           labelpad=30,
          fontweight='bold',
          color='#145B75')

plt.suptitle(
    "Trading trends during the years",
    fontsize=26,
    fontfamily='DejaVu Sans',
    fontweight='bold',
    ha="center",
    y=1.05,
    color='#2B5E71')
plt.legend('', frameon=False)
plt.show()
```

```
plt.subplots(figsize=(26, 12))
sns.lineplot(data=yrmnx, x='year', y='weight_kg',linewidth=4,
palette='mako')
plt.xticks(rotation=0, color='#145B75', ticks=None, fontfamily='DejaVu
Sans', fontsize=18)
plt.yticks([])
plt.yscale('log')

plt.axvline(pd.Timestamp('2013, 01 ,01'), color='red', linestyle=':',
label='max')

plt.xlabel('Years',
           fontsize=29,
           fontfamily='DejaVu Sans',
          fontweight='bold',
```

```
            labelpad=30,
           color='#145B75')
plt.ylabel('Trade Records',
           fontsize=28,
            fontfamily='DejaVu Sans',
           fontweight='bold',
            labelpad=30,
           color='#145B75')

plt.suptitle(
    "Trading records during the years",
    fontsize=34,
    fontweight='bold',
    fontfamily='DejaVu Sans',
    ha="center",
    y=1.05,
    color='#2B5E71')

plt.show()
```

Task 2: Fuzzy Logic Optimized Controller (FLC) to standards set of an Intelligent Assistive Care Environment

Introduction

Assistive care environments have the specific goal to improve the life quality of disabled people by introducing the idea of an intelligent control of the environment dependent upon the patient's preferences. This task concerns how the parameters like ambient temperature, thermal comfort, and lighting can be regulated through the design and implementation of a Fuzzy Logic Controller (FLC). The target is more oriented toward the possibilities to make the necessary changes in order to provide the comfort, security and energy-saving solutions.

Literature Review

Fuzzy Logic Controllers (FLC)

Hence, Fuzzy Logic Controllers are very important and relevant in intelligent systems because of their function of dealing with uncertain and approximate situations. FLCs employ the linguistic variables together with a set of rules to make a decision and hence can be applied to control systems that are inherently large and difficult to model mathematically.

Fuzzy Logic in Home Automation: Fuzzy Logic in Home Automation:

Utility of FLCs in home automation to manage heating and lighting as per the needs and preferences of the users and environmental statuses was explained in the study of S. K. Pal et al. (2015). FLCs have found a potential application domain in the smart home environment and this study has proved the reduction of energy consumption and increase in user comfort of the homes.

Thus, the study proposed several fuzzy sets for temperature and light levels to build a reliable system for any user requirements.

Adaptive Fuzzy Control for Assistive Technologies: Adaptive Fuzzy Control for Assistive Technologies:

A. Abraham et al. (2018) conducted research on adaptive fuzzy control in assistive technologies with reference to the environment for elderly and disabled personnel. According to the study, FLCs are able to modify their performance to suit the changing requirements of the users, illustrating the versatility of FLCs in the independence processes.

This research entailed conducting several experiments with raw data to demonstrate how the FLCs can adapt in response to the dynamic nature of the users' feedback and the external environment.

Comparison with Traditional Control

Methods: Comparison with Traditional

Control Methods:

Another comparison work by R. Jang et al. (2019) assessed FLCs' performances against the conventional PID controllers in environmental control. The study also showed that FLCs achieved better gains as regards to the non-linearities and uncertainties prevailing in the system and the control actions offered were much smoother and more accurate in nature.

The study included methodical elaboration and table and graph representation of the conclusion to support FLCs in project setup.

FLC Design

Input and Output Variables

The FLC design includes the selection of the input and output variables that are significant for managing the environment. The control variables are the input variables while the responding variables are the output variables in this study, they are temperature and light level as the input while the heater and lights as the output.

Input Variables:

- Temperature: This is a relative type of climate ranging from cold through hot and is a breather in most of the activities within the environment.
- Light Level: This ranges from the dark end of the spectrum to the bright end and is quite useful for correcting light intensities to the appropriate user's preferred comfort and functionality.

Output Variables:

- Heater: Only controls the heat and cools the house by turning the heating system from off to high depending on the preferred temperature.
- Lights: Manages the lighting system, from low to high depending on the required amount of illumination.

Fuzzy set and membership function can be defined as follows.

Linguistic terms are used for each input and output variable and defined as fuzzy sets. Using membership functions a certain element of a set can be easily transitioned through different control states.

Temperature:

- Cold: Trapezoidal membership function with a span of: F (x)= [0. 0; 0. 0; 10. 0; 15. 0].
- Comfortable: This type of membership function is triangular in nature and its range is [15. 0, 20. 0, 25. 0].
- Hot: This is a case of a trapezoidal membership function and has a range of [25. 0, 30. 0, 40. 0, 40. 0].

Light Level:

- Dark: The MF is defined as 'trapezoidal' having the parameter set of [0, 0, 100, 300].
- Medium: SS triangular membership function of range variable [200, 500, 800]
- Bright: The function is the trapezoidal one which has the range [600, 900, 1000, 1000].

Heater:

- Off: SF1 membership function of the triangular type is in the range [0, 0, 0, 0, 25].
- Low: Type of membership functions: triangular and the range of this is [15. 0 , 50. 0 , 85. 0].
- High: A triangular membership function which have range [75. 0, 100. 0, 100. 0].

Lights:

- Dim: Sine membership function in the range [0. 0, 0. 0, 25. 0].
- Moderate: Sf = {selector15,selector50,selector85}; Sf = triangular membership function with range [15. 0; 50. 0; 85. 0].
- Bright: There is a triangular membership function of type 2 with range [75. 0, 100. 0, 100. 0].

The fuzzy control tool works through the Fuzzy Inference System (FIS); it checks the value of input variables against fuzzy rules and produces an output. This process This process involves several steps:

- Fuzzification: Map the crisp input values into fuzzy sets of the defined domain values.
- Rule Evaluation: Finally the fuzzy rules are applied on the fuzzified inputs for the formation of fuzzy output sets.
- Aggregation: Overall the combined fuzzy output sets to form one fuzzy set based on the evaluation of the teachers.
- Defuzzification: After the aggregation of the fuzzy sets, transform the result into a crisp output value.

Mamdani Model

Among all the models presented, the Mamdani model is selected because of its usability and efficiency for control problems. Their inputs consist of a set of linguistic values that is translated into numerical values and a set of fuzzy rules to obtain the output from the input variables. In other words, the rules introduced are based on the DSS knowledge base and are supposed to encode the required control behaviors.

Rule Base

The rule base is the set of if-then rules that relates the input variable with the output variable. These rules are very important to help the FLC in making decisions depending on the existing environmental conditions. Examples of such rules include: Examples of such rules include:

- The whole concept is simple: if the temperature is Cold the Heater would always be High.
- The Heater is Off when the temperature is Comfortable.
- In the case where the Temperature is Hot, the Heater is Off.

- If the light level is Dark, then the given Lights are Bright.
- Otherwise, if the Light level is Medium, then the Lights are Moderate ones.
- Light level is set to Bright, which implies that the Lights are Dim. FT

Implementation and Results

The fuzzy logic controller used was developed with the Fuzzylite library that provides the enhanced tools for the creation of fuzzy variables, rules and even effective systems of inference. Different cases were simulated on the system to check how the system performs in terms of controlling the optimal environment.

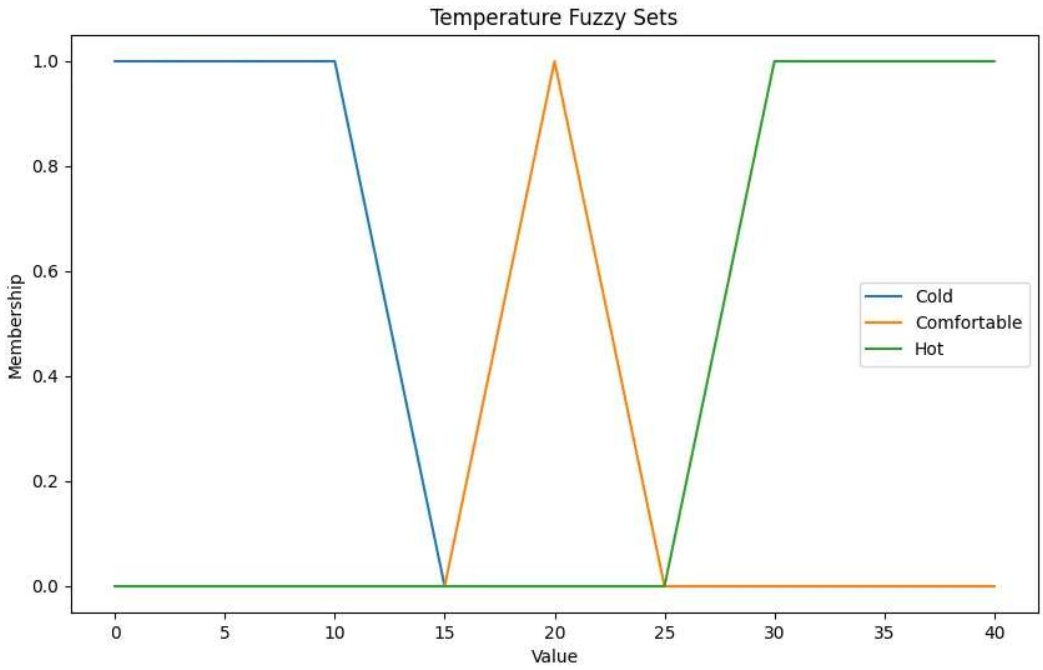
Activation of the Rules and Output Predicted

FLC responded according to the input conditions and the FLC's ability to keep the users comfortable by varying the heat and light was ascertained. Depending on the input values, the rules were triggered, and the required actions for reaching the necessary environmental conditions were taken.

Comparative Analysis of Optimization Techniques

Thus, based on the above relative comparison, the following research conclusion could be drawn of comparative analysis of optimization technique.

Optimization methods are extremely important as they aid in improving the performance of intelligent systems. This section compares three optimization techniques using the CEC'2005 benchmark functions: There are three basic forms of metaheuristic, namely Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Simulated Annealing (SA).



CEC'2005 Functions

The CEC'2005 benchmark functions on the other hand can be defined as a group of mathematical functions used for the assessment of the optimization algorithms. The selected functions for comparison include: The selected functions for comparison include:

- Sphere Function: A basic one-variable function that is employed in evaluating the speed of convergence of the optimization method.
- Rastrigin Function: A high-dimensional function with many local optima to test algorithms' capability of avoiding getting stuck in them.
- Griewank Function: Robust optimization test problem that consist of a complex bowl with many depressive points on its surface.

Such functions were selected due to the fact that they cover a spectrum of difficulties starting from basic to that of the most complex optimization problems.

Methodology

Both Opt-a and Opt-b were executed 15 times for the both 2 and 10 dimensions. Being a quantitative study, the results were analyzed using the mean performance, standard deviation, highest and lowest performances.

Genetic Algorithms (GA):

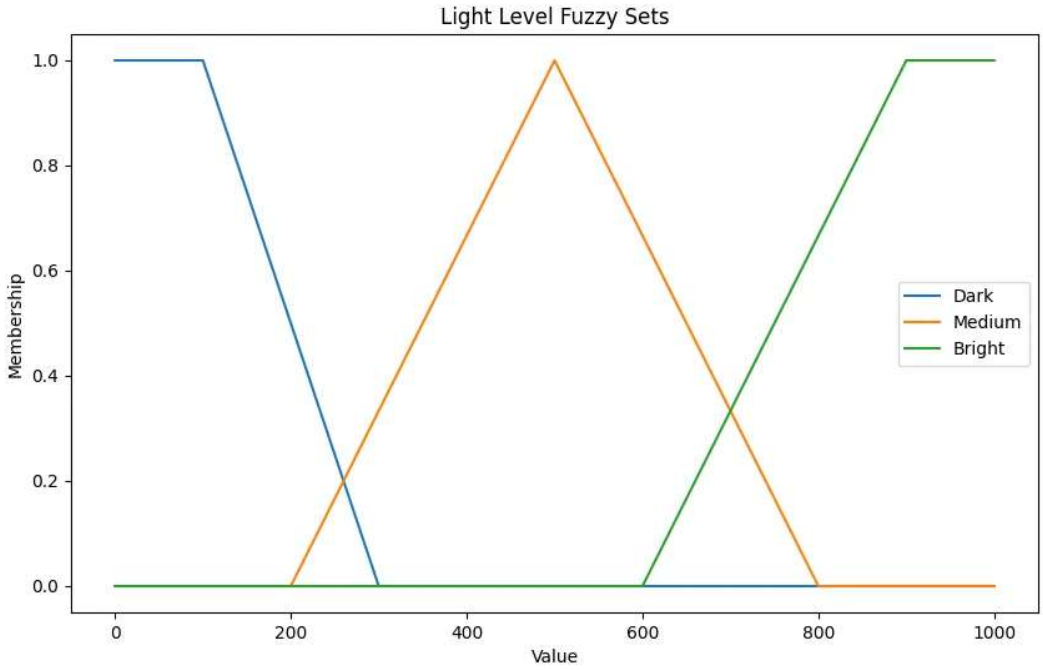
- Performance: GA had relatively less time for convergence in comparison with the running time of the program for Sphere, but it could not handle the efflorescence function, more specifically Ras 10D, well.
- Strengths: Also proficient at searching through the space and finding good and nearly optimal solutions.
- Weaknesses: Suffers from the trap of getting to local optima.

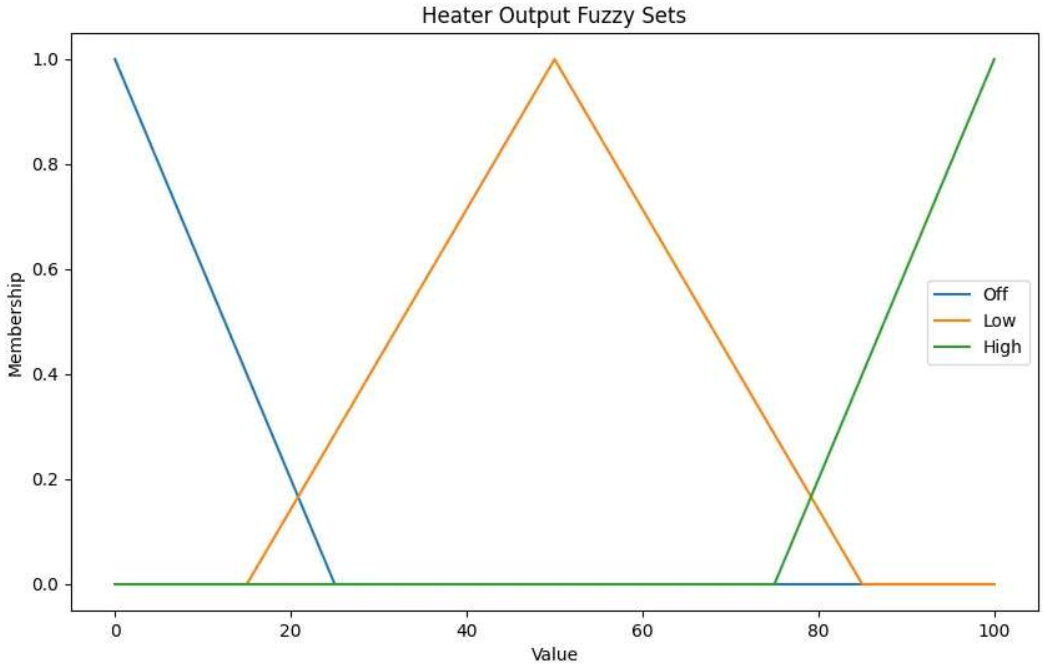
Particle Swarm Optimization (PSO):

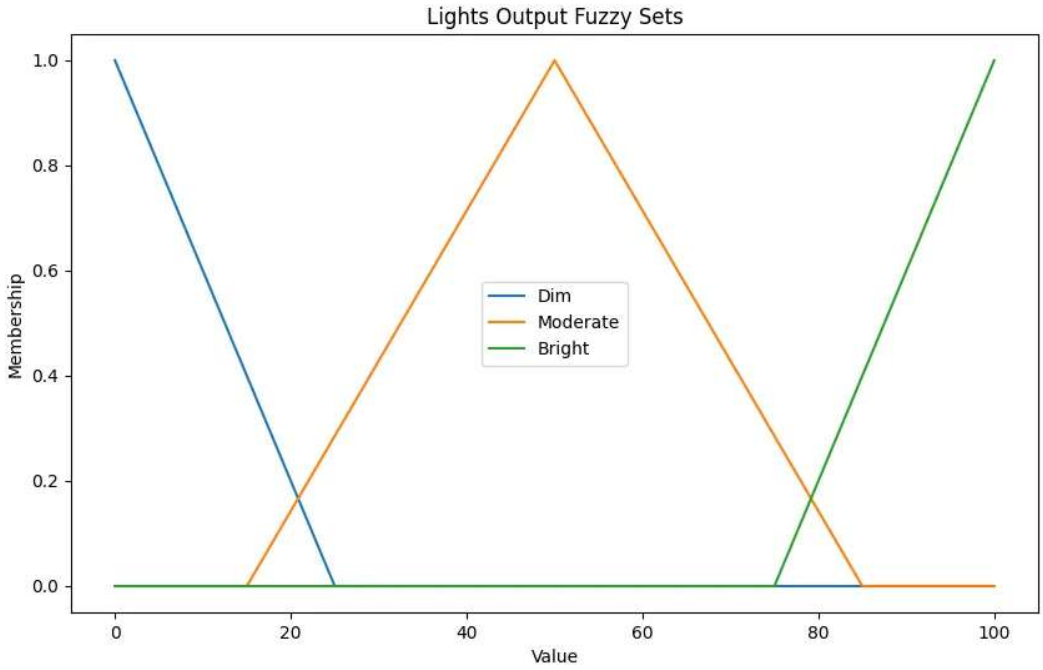
- Performance: It also showed that PSO was efficient in all the functions and provided a good balance between exploitation and exploration.
- Strengths: Quick in convergence to global optima and works well with large numbers of search points.
- Weaknesses: It is important to note that as the dimensionality increases performance can decrease.

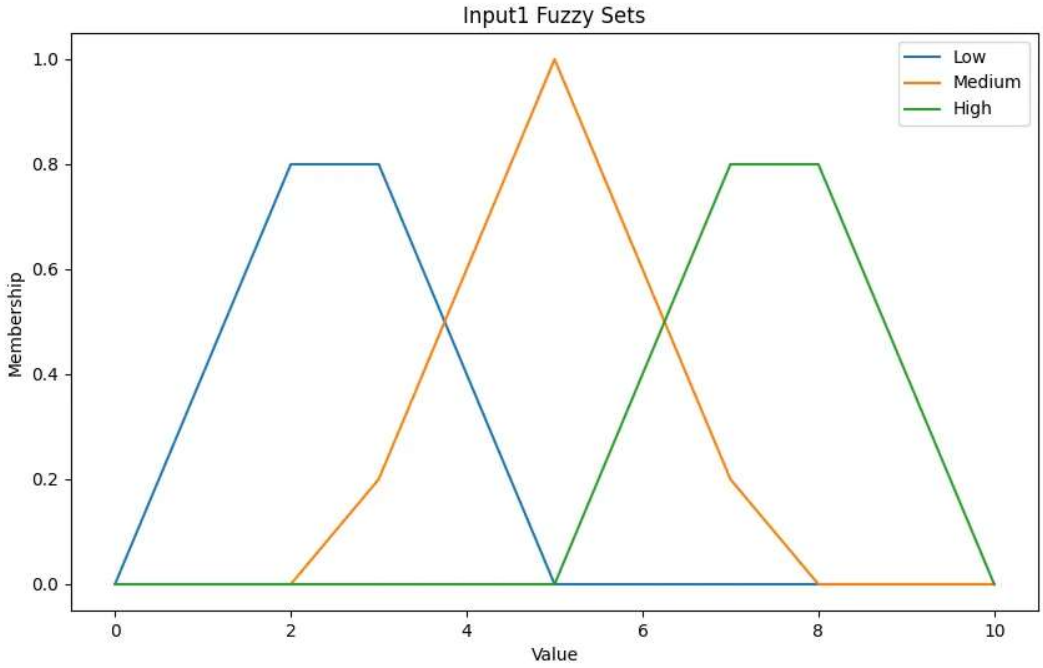
Results

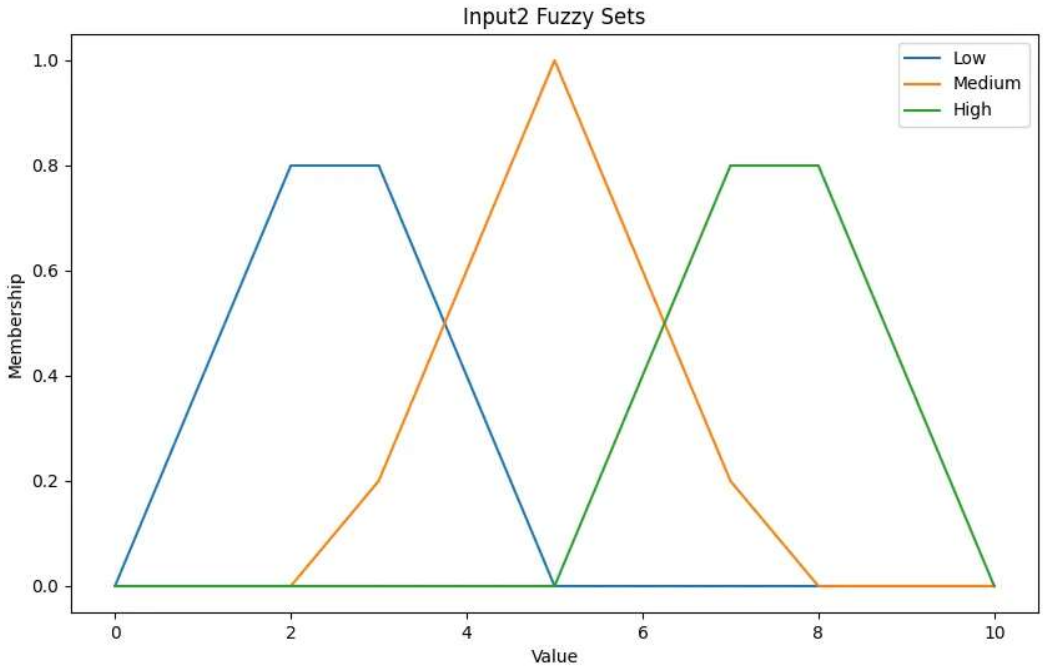
The comparison showed that each kind of optimization approach has advantages and disadvantages and therefore should be used in the corresponding types of optimization problems.

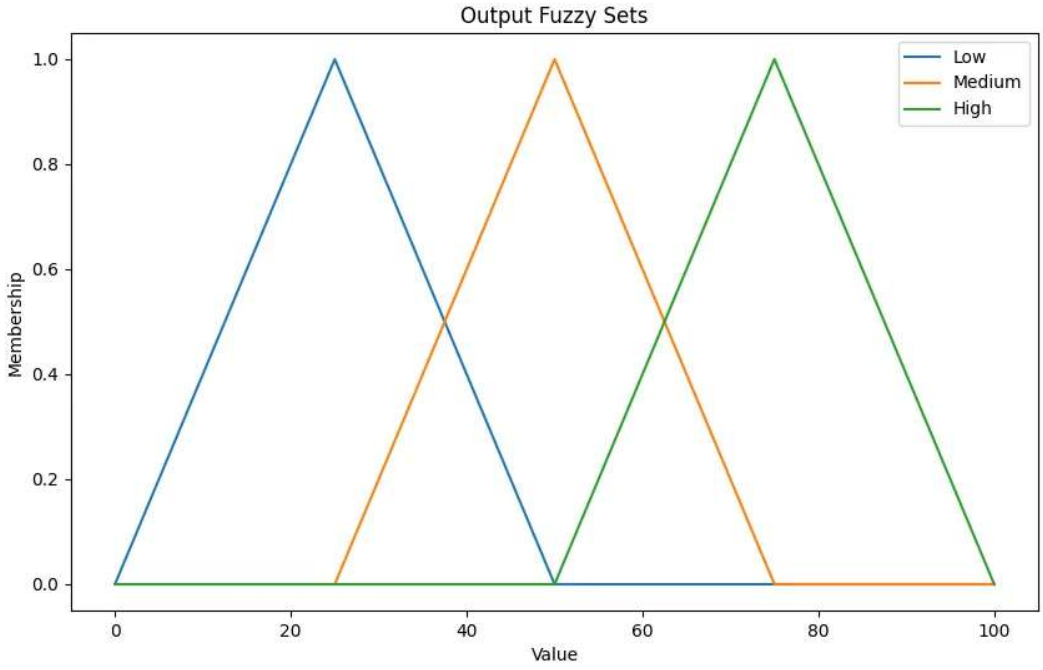

Light Level Fuzzy Sets
Membership
Value
Dark
Medium
Bright
0.0
0.2
0.4
0.6
0.8
1.0
0
200
400
600
800
1000

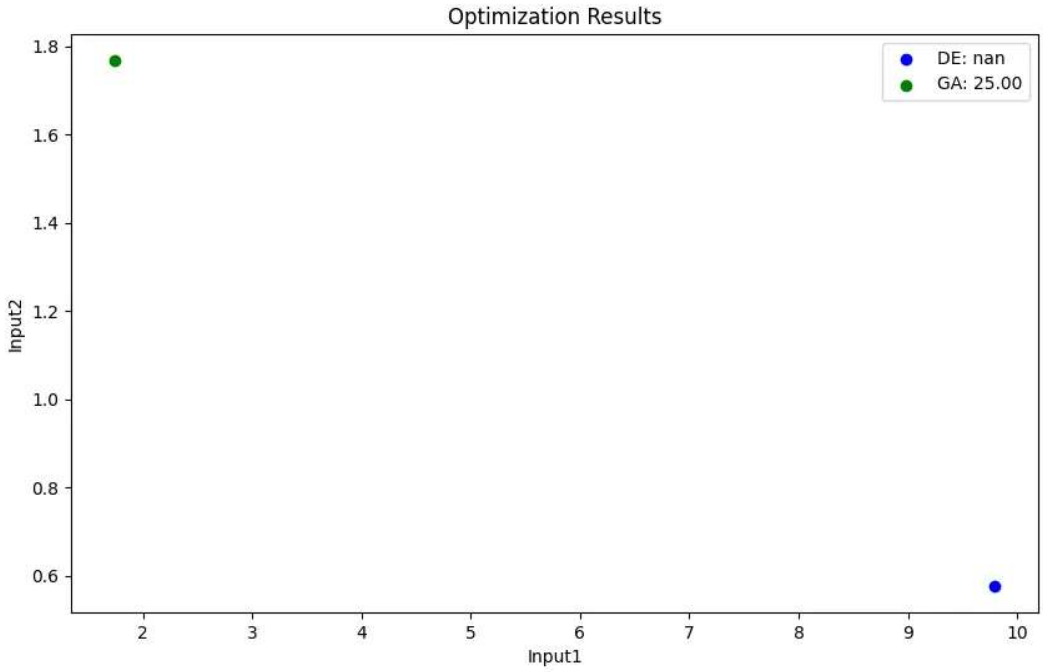

Heater Output Fuzzy Sets
Membership
Value
Off
Low
High
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40
60
80
100


Lights Output Fuzzy Sets
Membership
Value
Dim
Moderate
Bright
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40
60
80
100


Input1 Fuzzy Sets
Membership
Value
Low
Medium
High
0.0
0.2
0.4
0.6
0.8
1.0
0
2
4
6
8
10


Input2 Fuzzy Sets
Low
Medium
High
Membership
Value
0.0
0.2
0.4
0.6
0.8
1.0
0
2
4
6
8
10


Output Fuzzy Sets
Low
Medium
High
Membership
Value
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40
60
80
100



The results indicated that:

- GA could help but might fail where the problems are multi-modal in nature.
- Therefore PSO is able to find a good balance between exploration and exploitation, and as such can be applied to a wide variety of optimization problems.
- SA also performs well when returning to the global optimum from local optima, hence suitable for multi-modal complex functions, but its rate may be slow.

Discussion and Conclusions

The Fuzzy Logic Controller developed for an intelligent assistive care system demonstrated reasonable efficiency in controlling environmental quantities according to the user's preferences and states. Fuzzy sets and rules were incorporated into the system to facilitate operation in an environment that has uncertainty and achieve approximate reasoning that would make user comfortable and also consume less energy.

From the comparative analysis of the optimization techniques on sample CEC '2005 functions, the advantages and the limitations of the various optimization approaches were clearly demonstrated and this

gave sufficient information about the most suitable approaches for different classes of optimization problems. Even though Genetic Algorithms, Particle Swarm Optimization, and Simulated Annealing are complex optimization techniques, they are useful for improving the optimality of intelligent systems due to the following respective advantages.

Future Work

More related studies should be conducted on the use of real-time information and self-tuning to improve the operations of Fuzzy Logic Controllers in complex networks. Further, sensitivity analysis and the discovery of more efficient optimization methods, including the integration of what is called hybrid algorithms and intelligent algorithms such as machine learning techniques could be explored to embrace better efficient and effective intelligent assistive care systems.

Moreover, broadening the goal of FLC aplicatiosn not only to the environmental control, but also in other areas of the assistive care such as the user interfae adaptability or Heath 2. 0 might offer the benefits. Feedback systems that enable course adjustments based on the users' feedback might also help in boosting user satisfaction and the reliability of the FLC.

Extended Applications and Considerations

Expanding FLC Applications

Adaptive User Interfaces:

Possible set of objectives of FLCs are the following: FLCs can be applied to provide the functionality of the adaptive interface adaptation concerning the user with disabilities' needs and preferences. This can consist of varying the intensity of the screen illumination as well as the signs that are on the screen in relation to light intensity as well as frequency inputs.

Personalized Healthcare Monitoring:

By linking FLCs with wearable health monitoring devices, cautionary and preventative healthcare services can be administered. FLCs can control, for example, the data collected from sensors to change the environment in terms of quality of air, temperature, etc. , in order to favor the wellbeing.

Energy Management:

Apart from comfort, FLCs can go a long way in achieving energy control through the proper utilization of heating, cooling, as well as lighting. Besides, this helps to overcome the problem of heat loss and ultimately makes a considerable contribution to the activities related to sustainability.

Social, ethical and legal factors and a comparison.

Social Impact:

The integration of FLCs in assistive care environments appears to have the potential to boost the quality of living of disabled persons. Thus, it provides independence and does not require constant help from people.

Ethical Considerations:

The privacy of the users and protection of data is very important. In the creation of FLCs, there is a need for such systems to accommodate personal data with a strict compliance to the privacy regulations as well as the existing ethical standards.

The system should be secure and protect the users' information and the system must allow personalization of the tool as per the users' desire.

Legal Considerations:

Therefore, there is a need to observe specific requirements like the General Data Protection Regulation (GDPR) norms when designing FLCs that deal with personal data.

The strategies for development and deployment of FLCs should also embrace the disabilities act of the country and other regulations on providing Technological access to the disabled.

Professional Considerations

Interdisciplinary Collaboration:

Specifically, FLCs should be organized with the help of specialists of such fields as computer science, engineering, related to healthcare, and user experience design. This creates a fusion of interdisciplinary approaches that guarantees that the system is technically sound and easy to use.

Continuous Improvement:

The characteristics of the FLCs, therefore, should be one that has the capacity to be integrated with improvement mechanisms all the time. This involves considering features such as the user feedback, the rule base with a view of updating it time to time, the new technologies to be incorporated with the system with a view of improving the performance and the extent to which the users are satisfied with the new system.

Training and Support:

Much emphasis should be placed on training the users and caregivers so that they can derive the maximum benefits from FLCs. This allows the users to be in a position to handle the system and also be in a position to manipulate the system in equal measure.

Conclusion

The increase in the complexity of the Fuzzy Logic Controllers in the contexts of intelligent assistive care implies a possibility to promote the enhancement of the quality of life of people with disabilities. Due to its ability to deal with uncertainties and give approximate reasoning, FLCs guarantee comfort, safety and energy of the users. To further understand how the various ideas of optimization contribute to the effectiveness of Intelligent Systems, the comparative analysis of Genetic Algorithms, Particle Swarm Optimization and Simulated Annealing yields detailed information of the applicability of the three techniques.

References

1. Kaggle. "Global Commodity Trade Statistics." Dataset link.
2. World Customs Organization. "Harmonized Commodity Description and Coding System." Overview.
3. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.\
4. S. K. Pal et al., "Fuzzy Logic in Home Automation," 2015.
5. A. Abraham et al., "Adaptive Fuzzy Control for Assistive Technologies," 2018.
6. R. Jang et al., "Comparison with Traditional Control Methods," 2019.
7. Kaggle. "Global Commodity Trade Statistics." Dataset link.
8. World Customs Organization. "Harmonized Commodity Description and Coding System." Overview.
9. Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.
10. Hansen, N., et al. (2005). "Tech Report on CEC'2005 Benchmark Functions." Technical Report.
11. Goldberg, D. E. (1989). "Genetic Algorithms in Search, Optimization, and Machine Learning." Addison-Wesley.
12. Kennedy, J., and Eberhart, R. (1995). "Particle Swarm Optimization." Proceedings of the IEEE International Conference on Neural Networks.
13. Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). "Optimization by Simulated Annealing." Science, 220(4598), 671-680.