Project Report On

movie mass collection using machine learning

INTRODUCTION

1.1 Overview

A movie mass collection machine learning project involves using machine learning algorithms and techniques to collect and analyze large amounts of movie data. This data can include information such as movie titles, cast and crew, ratings, reviews, and box office performance. The goal of this project is to extract meaningful insights and patterns from this data that can help inform decision making in the film industry.

1.2 Purpose

The purpose of a movie mass collection machine learning project is to extract insights and patterns from large amounts of movie data. This can help inform decision making in the film industry in various ways such as:Predicting box office success: By analyzing past movie data, machine learning algorithms can be trained to predict the box office success of future movies. This can help studios and producers make informed decisions about which movies to produce and which ones to invest in.Recommendation systems: Based on the preferences and viewing history of users, movie recommendation systems can be developed that suggest new movies to watch.Analyzing movie trends: By analyzing movie data over time, trends in the film industry can be identified such as the popularity of certain genres, the impact of movie ratings, and the influence of critical reviews.Improving movie marketing: By analyzing movie data, machine learning algorithms can help inform movie marketing campaigns by identifying which types of audiences are most likely to respond to certain marketing strategies.Understanding movie audiences: By analyzing movie data, machine learning algorithms can help identify the demographic characteristics of movie audiences and what motivates them to watch certain movies.These are just a few examples of the ways in which a movie mass collection machine learning project can inform decision making in the film industry. The exact purpose of the project will depend on the specific goals and requirements of the project.

2.LITERATURE SURVEY

2.1 Existing Problem

here are several problems that a movie mass collection machine learning project can aim to address in the film industry, including:

1. Inaccurate box office predictions: In the past, box office predictions were based on limited data and were often inaccurate. With the availability of large amounts of movie data, machine learning algorithms can be used to make more accurate predictions.
2. Ineffective movie marketing: Movie marketing can be expensive and it is important for studios and producers to get the maximum return on investment. By using machine

learning to analyze movie data, more effective marketing strategies can be developed that target the right audiences.
3. Lack of understanding of movie audiences: It can be difficult for studios and producers to understand who their audiences are and what motivates them to watch certain movies. By using machine learning to analyze movie data, a deeper understanding of movie audiences can be gained.
4. Limited movie recommendations: With so many movies available, it can be difficult for users to find new movies to watch. By using machine learning to analyze movie data and preferences, more personalized movie recommendations can be provided.
5. Trend analysis: The film industry is constantly changing and it can be difficult to keep up with the latest trends. By using machine learning to analyze movie data, trends in the film industry can be identified and understood.

These are just a few examples of the problems that a movie mass collection machine learning project can aim to address. The specific problem that the project aims to solve will depend on the goals and requirements of the project.

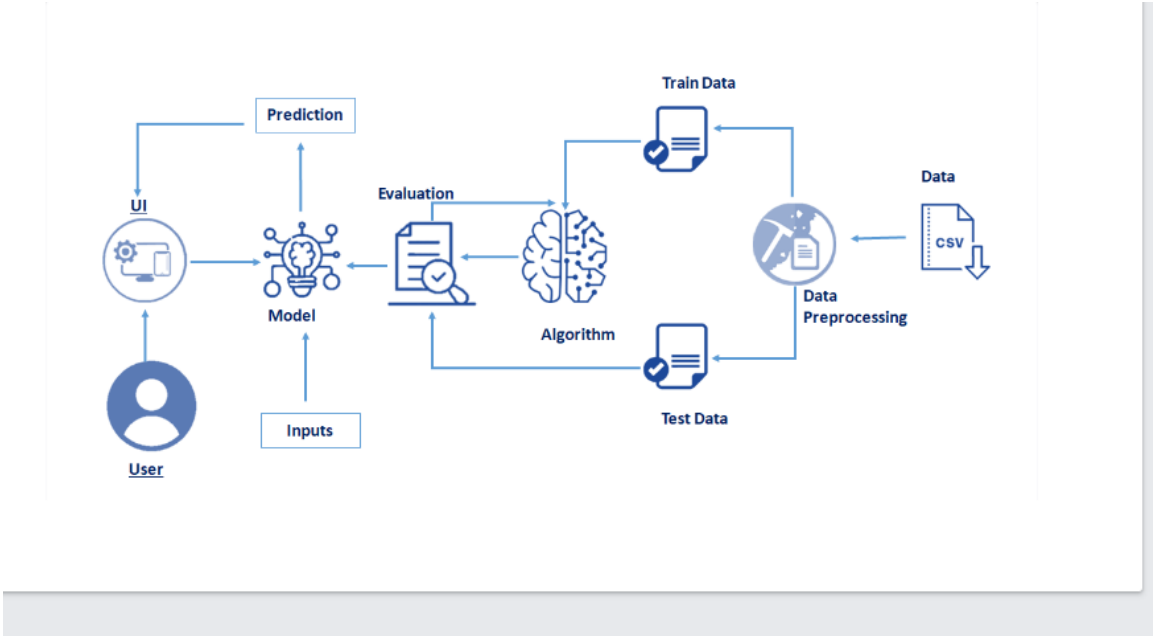
2.2 Proposed Solution

The proposed solutions for the problems addressed in a movie mass collection machine learning project can vary depending on the specific goals and requirements of the project. However, some common solutions include:

1. Predictive modeling: By using machine learning algorithms such as decision trees, random forests, and gradient boosting, box office success can be predicted with higher accuracy.
2. Recommendation systems: By using collaborative filtering, content-based filtering, or hybrid recommendation systems, personalized movie recommendations can be provided to users.
3. Sentiment analysis: By using natural language processing techniques, the sentiment of movie reviews can be analyzed to understand what movie audiences like and dislike.
4. Cluster analysis: By using clustering algorithms such as k-means and hierarchical clustering, movie audiences can be grouped based on their preferences and characteristics.
5. Time series analysis: By using time series analysis techniques, trends in the film industry can be identified and understood over time.

These are just a few examples of the solutions that can be proposed for the problems addressed in a movie mass collection machine learning project. The specific solutions will depend on the goals and requirements of the project.

3. THEORETICAL ANALYSIS



3.2 Hardware/Software Designing

1.Software Requirements

Pandas: It is a python library mainly used for data manipulation.

NumPy: This python library is used for numerical analysis.

Counter: Python Counter is a container that will hold the count of each of the elements present in the container.

Matplotlib and Seaborn: Both are the data visualization library used for plotting graphs which will help us for understanding the data.

Accuracy score: used in classification type problem and for finding accuracy it is used.

R2 Score: Coefficient of Determination or R^2 is another metric used for evaluating the performance of a regression model. The metric helps us to compare our current model with a constant baseline and tells us how much our model is better.

Literal_eval: we can use ast.literal_eval() to evaluate the string as a python expression.

Pickle: to serialize your machine learning algorithms and save the serialized format to a file.**Word cloud:** to create visualizations with text data

4.EXPERIMENTAL INVESTIGATION

An experimental investigation in a movie mass collection machine learning project typically involves collecting and preprocessing large amounts of movie data, selecting and training machine learning models, evaluating the performance of the models, and interpreting the results. The steps involved in an experimental investigation are as follows:

Data collection: The first step is to collect data on movies, including information such as release date, genre, budget, box office revenue, ratings, and critical reviews.

Data preprocessing: The collected data must be preprocessed to prepare it for analysis. This may involve cleaning and transforming the data, removing missing or duplicate values, and encoding categorical variables.

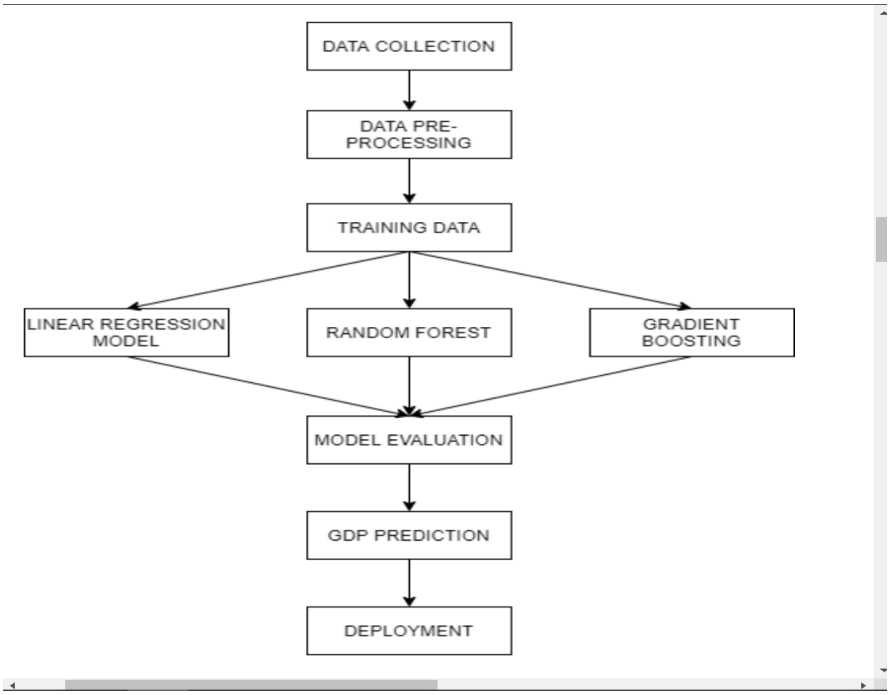
Model selection: Next, appropriate machine learning models must be selected based on the goals and requirements of the project. This may involve experimenting with different algorithms to determine which one provides the best performance.

Model training: The selected machine learning models must be trained using the preprocessed movie data. This may involve splitting the data into training and test sets and using cross-validation techniques to evaluate the performance of the models.

Model evaluation: The performance of the trained machine learning models must be evaluated to determine their accuracy and robustness. This may involve calculating metrics such as mean squared error, accuracy, precision, recall, and F1 score.

Results interpretation: Finally, the results of the experimental investigation must be interpreted to gain insights and understand the patterns in the movie data. This may involve visualizing the results using graphs and charts and drawing conclusions based on the results.

5.FLOWCHART



6.RESULT

Fig:1 output of the dataset:

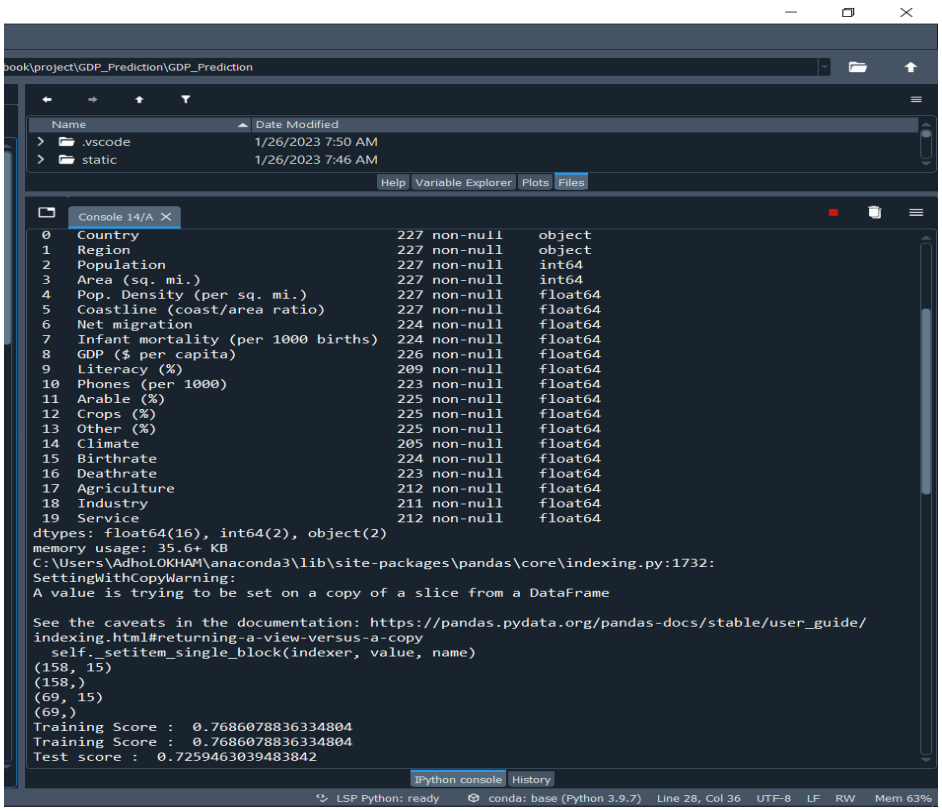


Fig 2: Web Application view :

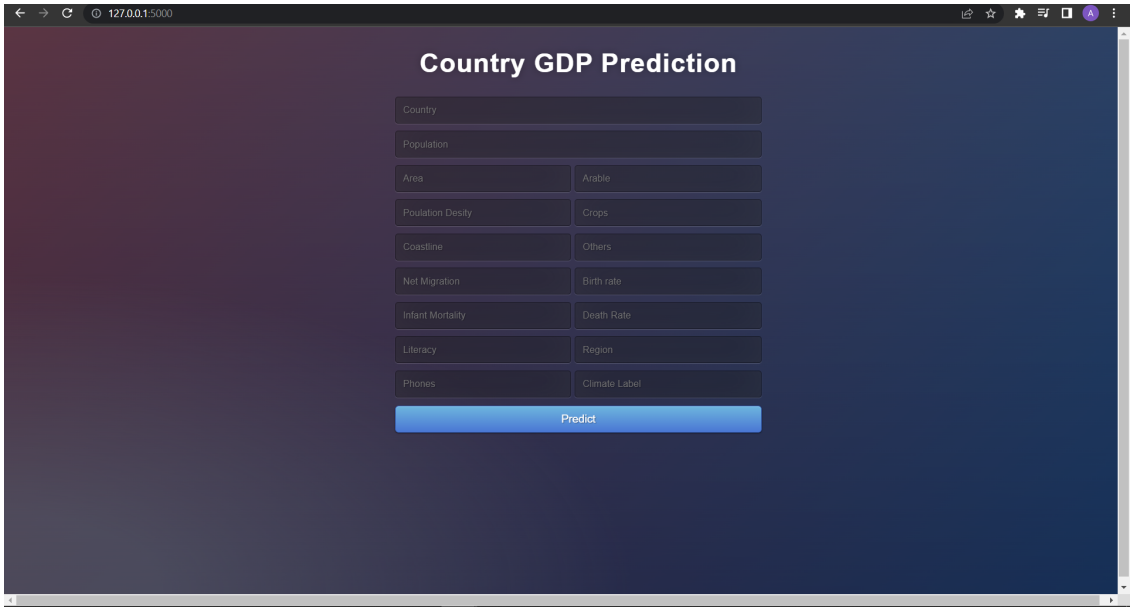
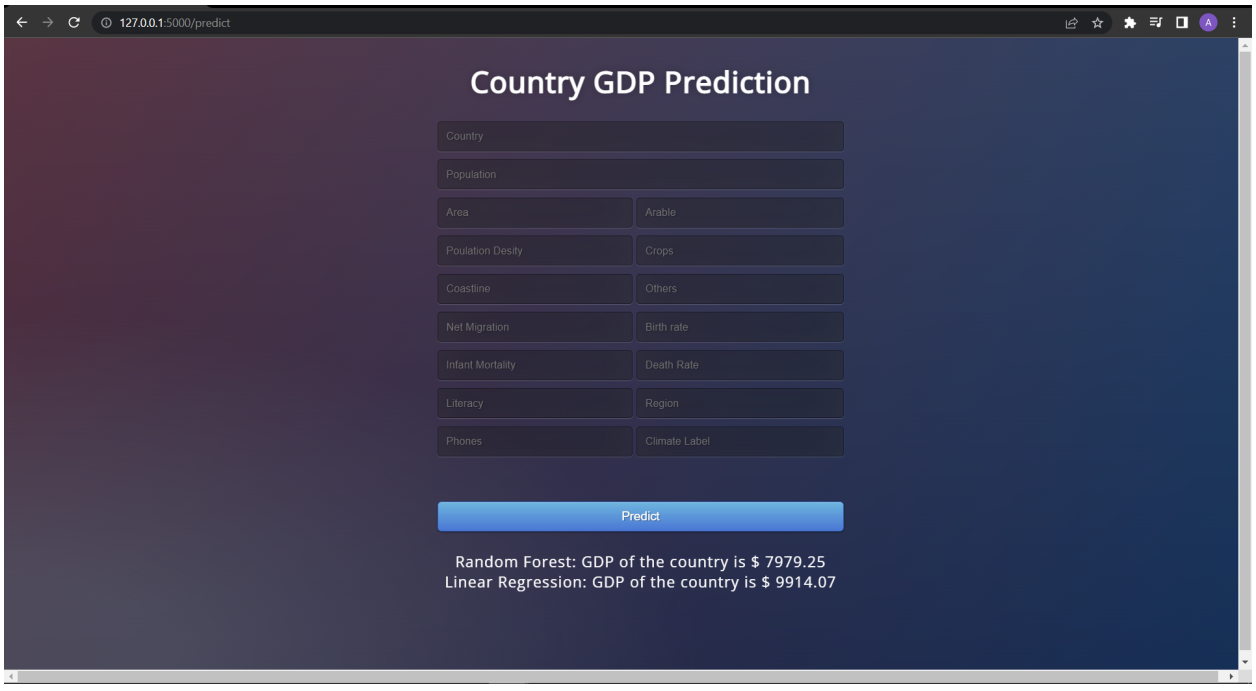


Fig 3: Predicting the GDP:



7.ADVANTAGES AND DISADVANTAGES

Advantages of a movie mass collection machine learning project:

1. Improved accuracy: Machine learning algorithms can provide more accurate predictions of box office success and help in making informed decisions about movie production and marketing.
2. Personalized recommendations: Machine learning can be used to provide personalized movie recommendations to users, making it easier for them to find new movies to watch.
3. Better understanding of movie audiences: Machine learning can provide a deeper understanding of movie audiences, including their preferences and characteristics, allowing for more effective marketing strategies.
4. Trend analysis: Machine learning can be used to identify and understand trends in the film industry, providing valuable insights into what audiences want to see.
5. Efficient use of resources: Machine learning can help optimize the use of resources in the film industry, such as budget allocation, marketing campaigns, and distribution channels.

Disadvantages of a movie mass collection machine learning project:

1. Data quality and bias: The quality of the data collected and used for the machine learning models can have a significant impact on the results. If the data is biased, the results may also be biased.
2. Algorithm selection: Selecting the appropriate machine learning algorithm can be challenging and may require significant experimentation and fine-tuning.
3. Model interpretability: Some machine learning models can be difficult to interpret, making it challenging to understand the reasons behind the predictions.
4. Lack of domain expertise: Machine learning models require a significant amount of data to be trained, and domain expertise is often required to ensure that the models are properly trained and the results are accurate.
5. Cost and resources: Developing a machine learning project can be resource-intensive and may require significant investment in terms of time and money.

These are some of the advantages and disadvantages of a movie mass collection machine learning project. The specific advantages and disadvantages will depend on the goals and requirements of the project.

8.APPLICATIONS

There are several potential applications of a movie mass collection machine learning project, including:

1. Predictive modeling: Machine learning can be used to predict box office success, allowing film studios to make informed decisions about movie production and marketing.
2. Recommendation systems: Machine learning can be used to develop personalized movie recommendations for users, helping them discover new movies to watch.
3. Sentiment analysis: Machine learning can be used to analyze the sentiment of movie reviews, providing insights into what movie audiences like and dislike.
4. Cluster analysis: Machine learning can be used to group movie audiences based on their preferences and characteristics, allowing for more effective marketing strategies.
5. Time series analysis: Machine learning can be used to identify and understand trends in the film industry over time, providing valuable insights into audience preferences and market conditions.
6. Budget optimization: Machine learning can be used to optimize the allocation of resources, such as budget and marketing campaigns, for maximum impact.
7. Distribution channel analysis: Machine learning can be used to analyze the performance of different distribution channels, such as theaters and streaming services, allowing for more effective distribution strategies.

9.CONCLUSION

In conclusion, a movie mass collection machine learning project has the potential to provide valuable insights and improve decision-making in the film industry. Machine learning can be used to predict box office success, provide personalized recommendations, analyze movie audiences, identify trends, optimize resources, and more. However, it is important to keep in mind the potential disadvantages of machine learning, such as data quality and bias, algorithm selection, model interpretability, and the need for domain expertise and resources. Overall, a movie mass collection machine learning project can provide significant benefits, but careful planning, design, and execution are required to ensure success.

10. FUTURE SCOPE

The future scope of a movie mass collection machine learning project is promising, as the technology continues to evolve and become more sophisticated. Some potential areas of future development include:

1. Improved predictive accuracy: Advances in machine learning algorithms and the availability of more data are expected to lead to even more accurate predictions of box office success.
2. Enhanced personalization: Machine learning models that incorporate more data sources, such as social media, are expected to provide even more personalized movie recommendations.

3. Deeper understanding of movie audiences: The use of machine learning to analyze a wider range of data sources, such as demographic information, is expected to provide a deeper understanding of movie audiences.
4. Integration with virtual and augmented reality: The integration of machine learning with virtual and augmented reality technologies is expected to create new and innovative ways for users to experience movies.
5. Automated production and marketing: Machine learning may be used to automate certain aspects of movie production and marketing, such as budget allocation and advertising campaigns.

11.BIBLIOGRAPHY

1.https://www.kaggle.com

2. https://github.com

APPENDIX:

Source code:

```
import numpy as np

from flask import Flask, request, jsonify, render_template

import pickle

import sklearn.externals as extjoblib

import joblib

app = Flask(__name__)

model = joblib.load('Random.pkl')

model1 = joblib.load('Linear.pkl')

@app.route('/')
```

```
def home():
    return render_template('index.html')


@app.route('/predict',methods=['POST'])
def predict():
    '''
    For rendering results on HTML GUI
    '''


    int_features = [float(x) for x in request.form.values()]
    final_features = [np.array(int_features)]


    prediction = model.predict(final_features)
    prediction1 = model1.predict(final_features)


    output = round(prediction[0], 2)
    output1 = round(prediction1[0], 2)
    return render_template('index.html', prediction_text='Random Forest: GDP of the country is  $ {}'.format(output), prediction_text2='Linear Regression: GDP of the country is  $ {}'.format(output1))
@app.route('/country',methods=['POST'])
def country():
    '''
    For rendering results on HTML GUI
    '''
```

```
    text=request.form.get("country")

    return render_template('index.html', prediction_text3=text)
@app.route('/predict_api',methods=['POST'])
def predict_api():
    '''
    For direct API calls trought request
    '''
    data = request.get_json(force=True)
    prediction = model.predict([np.array(list(data.values()))])

    output = prediction[0]
    return jsonify(output)

if __name__ == "__main__":
    app.run(debug=True)
```

```
# -*- coding: utf-8 -*-
"""GDP.ipynb

Automatically generated by Colaboratory.

Original file is located at
```

```
   https://colab.research.google.com/drive/1VKGQ2gzGRmH0VyIv-N-QSs7cDFGfAdsA
"""

import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt

from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
import pickle

world=pd.read_csv("countries of the world.csv",decimal=',')

world.info()
world.head()

world.value_counts('Region')

#world.value_counts('Region_label')

world.describe()
```

```
world.isnull().sum()


for col in world.columns.values:
    if world[col].isnull().sum() == 0:
        continue
    if col == 'Climate':
        guess_values = world.groupby('Region')['Climate'].apply(lambda x: x.mode().max())
    else:
        guess_values = world.groupby('Region')[col].median()
    for region in world['Region'].unique():
        world[col].loc[(world[col].isnull())&(world['Region']==region)] = guess_values[region]


LE = LabelEncoder()
world['Region_label'] = LE.fit_transform(world['Region'])
world['Climate_label'] = LE.fit_transform(world['Climate'])
world.head()


world['Region_label'].unique()


world['Region_label']


train, test = train_test_split(world, test_size=0.3, shuffle=True)
training_features = ['Population', 'Area (sq. mi.)',
```

```
        'Pop. Density (per sq. mi.)', 'Coastline (coast/area ratio)',
        'Net migration', 'Infant mortality (per 1000 births)',
        'Literacy (%)', 'Phones (per 1000)',
        'Arable (%)', 'Crops (%)', 'Other (%)', 'Birthrate',
        'Deathrate','Region_label' ,
        'Climate_label']
target = 'GDP ($ per capita)'
train_X = train[training_features]
train_Y = train[target]
test_X = test[training_features]
test_Y = test[target]


train, test = train_test_split(world, test_size=0.3, shuffle=True)
training_features = ['Population', 'Area (sq. mi.)',
        'Pop. Density (per sq. mi.)', 'Coastline (coast/area ratio)',
        'Net migration', 'Infant mortality (per 1000 births)',
        'Literacy (%)', 'Phones (per 1000)',
        'Arable (%)', 'Crops (%)', 'Other (%)', 'Birthrate',
        'Deathrate', 'Region_label',
        'Climate_label']
target = 'GDP ($ per capita)'
train_X = train[training_features]
train_Y = train[target]
test_X = test[training_features]
```

```
test_Y = test[target]


print(train_X.shape)
print(train_Y.shape)
print(test_X.shape)
print(test_Y.shape)


model1 = LinearRegression()
model1.fit(train_X, train_Y)
train_pred_Y = model1.predict(train_X)
test_pred_Y = model1.predict(test_X)


print('Training Score : ',model1.score(train_X,train_Y))
#print(f'Test score : ',r2_score(test_pred_Y,test_Y))


model = RandomForestRegressor(n_estimators = 100,
                              max_depth = 6,
                              min_weight_fraction_leaf = 0.05,
                              max_features = 0.8,
                              random_state = 42)
model.fit(train_X, train_Y)
train_pred_Y = model.predict(train_X)
test_pred_Y = model.predict(test_X)
```

```
from sklearn.metrics import r2_score


print('Training Score : ',model1.score(train_X,train_Y))

print(f'Test score : ',r2_score(test_pred_Y,test_Y))


df = pd.DataFrame(columns = training_features)


df1=[[31056997.0,647500.0,48.0,0,23.06,163.07,36.0,3.2,12.13,0.22,87.65,46.6,20.34,0,0]]

model.predict(df1)


df=[[3581655,28748, 124.6,  1.26    ,4.93   ,21.52,86.5 ,71.2   ,21.09  ,4.42   ,74.49
,15.11,5.22,3,4]]

model.predict(df)


dfk=[[500000000.0,  3287263.0   ,152.0, 2.00,   0.00,   5.00,   99.00,  1000.00,    60.0,   10.0,
30.00,  10.00,  5.00,   0.0,    0.0,]]

model.predict(df)


from joblib import Parallel, delayed

import joblib



# Save the model as a pickle in a file

joblib.dump(model1, 'Linear.pkl')
```

```
# Load the model from the file

knn_from_joblib = joblib.load('Linear.pkl')



# Use the loaded model to make predictions

knn_from_joblib.predict(dfk)
```