Envisioning Success: Predicting University Scores With Machine Learning

1.Introduction

1.1Project Overview

Giving scores to universities can serve several important purposes. Firstly, it allows prospective students and their families to make informed decisions about where to pursue higher education based on objective criteria such as academic reputation, research output, faculty quality, and student outcomes. Secondly, it provides universities with valuable feedback on areas of strength and weakness, which can help them identify areas for improvement and enhance their overall quality. Thirdly, university rankings can also have an impact on funding and reputation, with higher-ranked institutions often receiving more research grants, attracting top talent, and enjoying greater prestige in the academic community. Overall, scoring universities can help ensure that students receive a high-quality education and that universities are held accountable for their performance. In this project, we have some characteristics of the universities as a dataset. The target variable of this dataset is the Score. This score is predicted on the basis of the following characteristics: Quality of education, Alumni Employment, Quality of faculty, Publications, Influence, Citations, and Patents. For making a better decision about your education, you can use this web application to predict your university score. The main purpose of the Score Prediction system is to predict the score of the university based on certain parameters.

1.2 Project Objective

The primary objective of "Envisioning Success: Predicting University Scores with Machine Learning" is to develop a predictive model that accurately forecasts university scores based on a set of input features. The model aims to identify the most influential factors that impact a student's academic performance, providing insights for educators, administrators, and policymakers to make data-driven decisions.

2. Project Initialization and Planning Phase

The "Project Initialization and Planning Phase" marks the project's outset, defining goals, scope, and stakeholders. This crucial phase establishes project parameters, identifies key team members, allocates resources, and outlines a realistic timeline. It also involves risk assessment and mitigation planning. Successful initiation sets the foundation for a well-organized and efficiently executed machine learning project, ensuring clarity, alignment, and proactive measures for potential challenges.

Activity 1: Define Problem Statement

Problem Statement: University scores are a critical factor in determining an academic score success. Predicting the scores can challenging due to the complexity of factor involved. A machine learning-based approach can provide a more accurate and efficient solution.

Ref. template: **Click Here**

Envisioning Success: Predicting University Scores With Machine Learning

Problem Statement Report: Click_Here

Activity 2: Project Proposal (Proposed Solution)

The proposed solution is to develop a CatBoost model, which is a gradient boosting algorithm that can handle categorical features effectively. The model will be trained using a dataset of student information and will predict university scores based on the input features.

Ref. template: **Click Here**

Envisioning Success: Predicting University Scores With Machine Learning

Project Proposal Report: **click Here**

Activity 3: Initial Project Planning

Initial Project Planning involves outlining key objectives, defining scope, and identifying the yield prediction. It encompasses setting timelines, allocating resources, and determining the overall project strategy. During this phase, the team establishes a clear understanding of the dataset, formulates goals for analysis, and plans the workflow for data processing. Effective initial planning lays the foundation for a systematic and well-executed project, ensuring successful outcomes.

Ref. template: [Click Here]
Envisioning Success: Predicting University Scores With Machine Learning
Initial Project Planning: : [click Here]

3. Data Collection and Preprocessing Phase

Dataset variables will be statistically analyzed to identify patterns and outliers, with Python employed for preprocessing tasks like normalization and feature engineering. Data cleaning will address missing values and outliers, ensuring quality for subsequent analysis and modeling, and forming a strong foundation for insights and predictions.

Activity 1: Data Collection Plan, Raw Data Sources Identified

Search for datasets related to Envisioning Success: Predicting University Scores With Machine Learning, and College Score predicting.
The raw data sources for this project include datasets obtained from Kaggle, the popular platform for data science competitions and repositories. The provided sample data represents a subset of the collected information, encompassing variables such rainfall, temperature.

Ref. template: [Click Here]
Envisioning Success: Predicting University Scores With Machine Learning
Raw Data Sources Report: [click Here]

Activity 2: Data Quality Report

The Data Quality Report Template will summarize data quality issues from the selected source, including severity levels and resolution plans. It will aid in systematically identifying and rectifying data discrepancies.

Ref. template: Click Here

Envisioning Success: Predicting University Scores With Machine Learning

Data Quality Report: click Here

Activity 3: Data Exploration and Preprocessing

Dataset variables will be statistically analyzed to identify patterns and outliers, with Python employed for preprocessing tasks like normalization and feature engineering. Data cleaning will address missing values and outliers, ensuring quality for subsequent analysis and modeling, and forming a strong foundation for insights and predictions.

Ref. template: Click Here

Envisioning Success: Predicting University Scores With Machine Learning

Data Exploration and Preprocessing Report: click Here

4. Model Development Phase

The Model Development Phase entails crafting a predictive model for loan approval. It encompasses strategic feature selection, evaluating and selecting models (Linear Regression, Lasso Regression, Decision Tree, Random Forest), initiating training with code, and rigorously validating and assessing model performance for informed decision-making in the lending process.

Activity 1: Feature Selection Report

In the forthcoming update, each feature will be accompanied by a brief description. Users will indicate whether it's selected or not, providing reasoning for their decision. This process will streamline decision-making and enhance transparency in feature selection.

Ref. template: Click Here

Envisioning Success: Predicting University Scores With Machine Learning

Feature Selection Report: click Here

Activity 2:Model Selection Report

In the forthcoming Model Selection Report, various models will be outlined, detailing their descriptions, hyperparameters, and performance metrics, including Accuracy or F1 Score. This comprehensive report will provide insights into the chosen models and their effectiveness.

Ref. template: Click Here

Envisioning Success: Predicting University Scores With Machine Learning

Model Selection Report: click Here

Activity 3: Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include a summary and training and validation performance metrics for multiple models, presented through respective screenshots.

Ref. template: Click Here

Envisioning Success: Predicting University Scores With Machine Learning

Model Development Phase Template: click Here

5. Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves refining machine learning models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

Activity 1: Hyperparameter Tuning Documentation

Hyperparameter tuning involves adjusting the parameters that govern the training process of machine learning models to optimize their performance. It includes methods such as grid search, random search, and Bayesian optimization. Proper documentation helps in understanding the impact of different hyperparameters, streamlining the tuning process, and replicating results. Clear records of hyperparameter settings and their outcomes are essential for achieving the best model accuracy and efficiency.

Ref. template: Click Here

Envisioning Success: Predicting University Scores With Machine Learning

Hyperparameter Tuning Report: click Here

Activity 2: Performance Metrics Comparison Report

A Performance Metrics Comparison Report systematically evaluates the effectiveness of various machine learning models or algorithms by comparing key metrics such as MAE,MSE,R-Square. This report highlights the strengths and weaknesses of each model, providing insights into their performance on different datasets or tasks. By presenting a clear, detailed analysis, the report aids in selecting the most suitable model for deployment, ensuring optimal performance in real world applications.

Ref. template: Click Here

Envisioning Success: Predicting University Scores With Machine Learning

Performance Metrics comparison Report: click Here

Activity 3: Final Model Selection Justification

The Final Model Selection Justification explains the rationale behind choosing the optimal machine learning model for a given task. This decision is based on a thorough analysis of various models' performance metrics, including Mae, Additionally, factors such as computational efficiency, interpretability, and scalability are considered. The justification ensures that the selected model not only performs well on the test data but also meets practical requirements, making it the most suitable choice for deployment in real-world scenarios.

Ref. template: Click Here

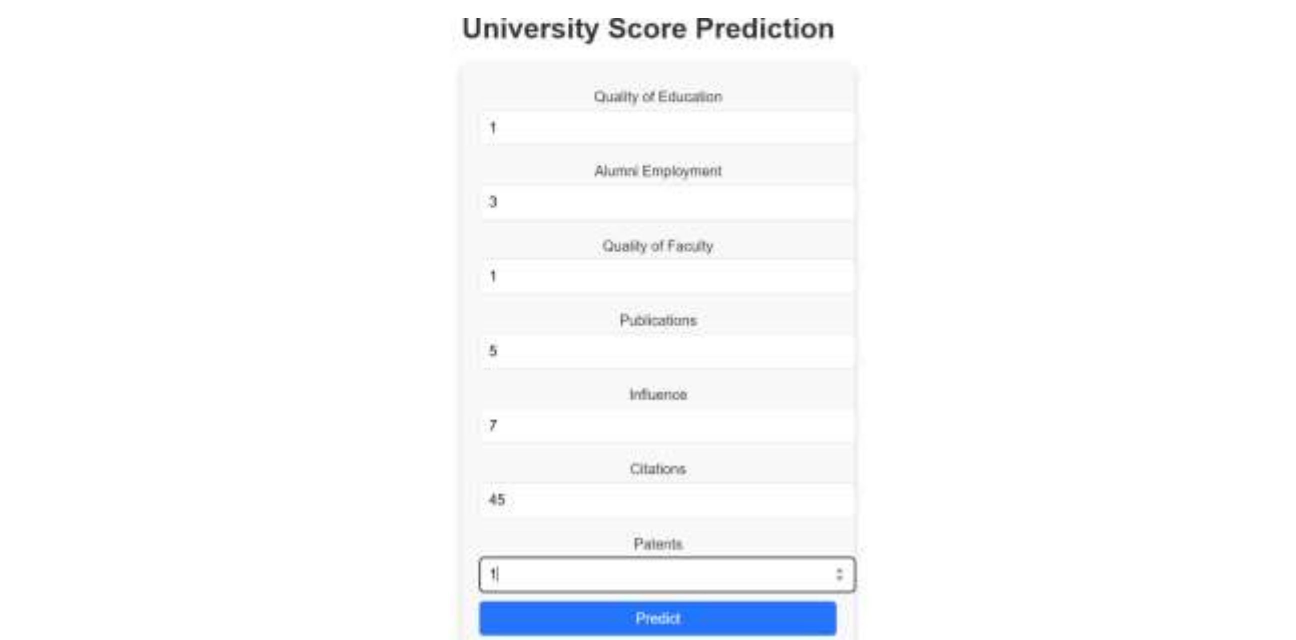
Envisioning Success: Predicting University Scores With Machine Learning

Final Model Selection Justification Report: click Here

6.RESULT

HOME PAGE

PREDICTION PAGE



RESULT PAGE

7.ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

1. Improved Student Outcomes:

The machine learning model can help identify students who are at risk of poor academic performance, enabling administrators to provide targeted support and interventions to improve student outcomes.

2. Data-Driven Decision-Making:

The model provides a data-driven framework for evaluating the effectiveness of academic programs and support services, enabling administrators to make informed decisions about resource allocation.

3. Personalized Support:

The model can help identify key factors that contribute to student success, enabling administrators to provide personalized support and resources to students who need it most.

4. Early Intervention:

The model can detect early warning signs of poor academic performance, enabling administrators to intervene early and provide support to students before they fall behind.

5. Scalability:

The machine learning model can be scaled to accommodate large datasets and provide predictions for a large number of students.

DISADVANTAGES:

1. Data Quality Issues:

The accuracy of the model depends on the quality of the data used to train it. Poor data quality can lead to biased or inaccurate predictions.

2. Model Complexity:

The machine learning model can be complex and difficult to interpret, making it challenging to identify the factors that contribute to student success.

3. Limited Data Availability:

The model may not have access to all relevant data, which can limit its ability to make accurate predictions.

4. Bias in the Model:

The model may perpetuate existing biases in the data, leading to unfair or discriminatory outcomes.

5. Dependence on Technology:

The model relies on technology and data, which can be prone to errors or downtime.

8.CONCLUSION

The project "Envisioning Success: Predicting University Scores With Machine Learning" aims to develop a web application that predicts university scores based on various characteristics. This project has several key components, including problem definition, data collection, preparation, exploratory data analysis, model building, model deployment, and documentation. It leverages Flask for web integration and a pre-trained machine learning model (usp.pkl) to make predictions. The business problem addressed here is to help prospective students and their families make informed decisions about where to pursue higher education by providing them with a university score based on objective criteria. This project contributes to the field of education by enabling better decision-making and accountability for universities. The project "Envisioning Success: Predicting University Scores with Machine Learning" exhibits promising prospects for future development and broader applications. As it stands, the project offers a valuable tool for prospective students and their families to make informed decisions about higher education by predicting university scores based on objective criteria. In the future, this tool could be enriched with additional features, such as user reviews, cost of education, and student feedback, providing a comprehensive evaluation of universities. Real-time data integration and regular updates can ensure users have access to the most current information in the ever evolving higher education landscape. Implementing a recommendation system that considers individual preferences and career goals can personalize the user experience. Allowing users to create profiles, developing a mobile application, and expanding international coverage can make the platform more accessible and user-friendly. Quality assurance, integration with admission systems, data analytics for universities, and AI powered chatbot support are additional avenues for expansion. Ultimately, the future of this project lies in continuous updates, user feedback, and the integration of emerging technologies to empower students and universities in the higher education ecosystem.

9.FUTURE SCOPE

1. Expansion to Other Educational Institutions: The model can be expanded to other educational institutions, such as high schools, community colleges, and online education platforms, to provide a more comprehensive understanding of student success.

2. Integration with Other Data Sources: The model can be integrated with other data sources, such as social media, learning management systems, and student information systems, to provide a more complete picture of student behavior and performance.

3. Real-Time Predictions: The model can be developed to provide real-time predictions, enabling administrators to respond quickly to changes in student performance and provide timely interventions.

4. Personalized Learning Paths: The model can be used to create personalized learning paths for students, tailoring the curriculum to their individual needs and abilities.

5. Faculty Performance Evaluation: The model can be used to evaluate faculty performance, providing insights into the effectiveness of different teaching methods and instructors.

6. Resource Allocation Optimization: The model can be used to optimize resource allocation, identifying areas where resources can be most effectively deployed to support student success.

7. Early Warning Systems: The model can be used to develop early warning systems, identifying students who are at risk of dropping out or failing, and providing targeted interventions to support them.

8. Career Outcome Predictions: The model can be expanded to predict career outcomes, providing students with insights into their potential career paths and enabling administrators to develop targeted career support services.

9. Multi-Institutional Collaborations: The model can be used to facilitate multi-institutional collaborations, enabling institutions to share data and best practices to improve student outcomes.

10. Continuous Model Improvement: The model can be continuously improved through ongoing data collection and analysis, ensuring that it remains accurate and effective in predicting student success.

10.APPENDIX

10.1 SOURCE CODE:

Index.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Envisioning Success</title>
  <style>
    body {
      margin: 0;
      padding: 0;
      font-family: Arial, sans-serif;
      color: #333; /* Set text color to dark gray */
    }

    .hero-section {
      background: url('/static/assests/img/uni.jpg') no-repeat center center;
      background-size: cover;
      height: 100vh;
      display: flex;
      justify-content: center;
      align-items: center;
      color: white;
      text-align: center;
      padding: 0 20px;
    }

    .hero-section h1 {
      font-size: 3rem;
      margin-bottom: 20px;
      font-weight: bold; /* Make the heading bold */
      color: black; /* Set heading color to black */
    }

    .hero-section p {
      font-size: 1.5rem;
      margin-bottom: 40px;
      color: black; /* Set paragraph color to black */
```

```
}

.hero-section .btn-primary {
    font-size: 1.25rem;
    color: black; /* Set button text color to black */
    text-decoration: none;
    border: 1px solid black; /* Add a black border to the button */
    padding: 10px 20px; /* Add padding to the button */
}

.about-section, .contact-section {
    padding: 60px 0;
    background-color: #f9f9f9; /* Light gray background */
}

.about-section h2, .contact-section h2 {
    text-align: center;
    margin-bottom: 40px;
    font-weight: bold; /* Make the section headings bold */
    color: black; /* Set section heading color to black */
}

.about-section p, .contact-section ul {
    color: black; /* Set text color inside sections to black */
    text-align: justify;
    margin: 0 auto;
    max-width: 800px;
}

.contact-section ul {
    list-style-type: none;
    padding: 0;
    text-align: center;
}

.contact-section li {
    margin-bottom: 10px;
}

.container {
    max-width: 1200px;
    margin: 0 auto;
    padding: 0 20px;
```

```
        }
    </style>
</head>
<body>
    <header class="hero-section">
        <div class="container text-center">
            <h1>Welcome to Envisioning Success</h1>
            <p>Predict University Scores with Machine Learning</p>
            <a href="/predict" class="btn btn-primary">Go to Prediction</a>
        </div>
    </header>

    <section id="about" class="about-section">
        <div class="container">
            <h2>About Us</h2>
            <p>Welcome to Envisioning Success! Our mission is to harness the power of machine learning to provide accurate predictions for university scores, helping students make informed decisions about their education. Our platform analyzes various factors, including quality of education, alumni employment, faculty expertise, research publications, and more, to deliver comprehensive insights.</p>
            <p>We believe in the transformative power of education and aim to support students in their journey towards academic excellence. By utilizing advanced algorithms and data-driven techniques, we strive to provide reliable and actionable predictions that can guide students in choosing the right university and maximizing their potential.</p>
            <p>Our team of experts continuously works to improve our models and ensure the highest level of accuracy. We are committed to innovation and excellence, making Envisioning Success a trusted resource for students, educators, and institutions alike.</p>
        </div>
    </section>

    <section id="contact" class="contact-section">
        <div class="container">
            <h2>Contact Us</h2>
            <ul>
                <li>Phone: +91 7780240811</li>
                <li>Email: university@domain.com</li>
                <li>Address: 123 Main Street, Your City, Your Country</li>
            </ul>
        </div>
    </section>

    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>
</body>
```

</html>

Prediction.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>University Score Prediction</title>
  <style>
    body {
      margin: 0;
      padding: 0;
      font-family: Arial, sans-serif;
      background: white; /* White background */
      height: 100vh; /* Full viewport height */
      display: flex;
      justify-content: center;
      align-items: center;
      color: #333; /* Dark gray text color */
      text-align: center;
    }

    #formbox {
      background: #f9f9f9; /* Light gray background for the form */
      padding: 20px;
      border-radius: 10px;
      box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
      max-width: 400px;
      width: 100%;
    }

    #formbox label {
      display: block;
      margin: 10px 0 5px;
      color: #333; /* Dark gray label text color */
    }

    #formbox input {
      width: 100%;
      padding: 10px;
      margin-bottom: 10px;
      border: 1px solid #ddd; /* Light gray border */
      border-radius: 5px;
      font-size: 1rem;
    }

    #bu {
      width: 100%;
      padding: 10px;
```

```
            border: none;
            border-radius: 5px;
            background-color: #2575fc;
            color: white;
            font-size: 1rem;
            cursor: pointer;
        }

        #bu:hover {
            background-color: #6a11cb;
        }

        #head, #prediction_text {
            margin-bottom: 20px;
            color: #333; /* Dark gray text color */
        }
    </style>
</head>
<body>
    <div>
        <h1 id="head">University Score Prediction</h1>
        <div id="formbox">
            <form action="{{ url_for('predict') }}" method="POST" id="form">
                <label for="qoe">Quality of Education</label>
                <input type="number" name="qoe" id="qoe" step="0.01" required>
                <br>
                <label for="ae">Alumni Employment</label>
                <input type="number" name="ae" id="ae" step="0.01" required>
                <br>
                <label for="qof">Quality of Faculty</label>
                <input type="number" name="qof" id="qof" step="0.01" required>
                <br>
                <label for="publ">Publications</label>
                <input type="number" name="publ" id="publ" step="0.01" required>
                <br>
                <label for="inf">Influence</label>
                <input type="number" name="inf" id="inf" step="0.01" required>
                <br>
                <label for="cit">Citations</label>
                <input type="number" name="cit" id="cit" step="0.01" required>
                <br>
                <label for="pat">Patents</label>
                <input type="number" name="pat" id="pat" step="0.01" required>
                <br>
                <button id="bu" type="submit">Predict</button>
            </form>
        </div>
        <h1 id="prediction_text">{{ prediction_text }}</h1>
    </div>
</body>
```

```
</html>
```

Result.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>University Score Prediction Result</title>
    <style>
        body {
            margin: 0;
            padding: 0;
            font-family: Arial, sans-serif;
            background: url('/static/assets/img/bg.jpg') no-repeat center center;
            background-size: cover;
            height: 100vh;
            display: flex;
            flex-direction: column;
            justify-content: center;
            align-items: center;
            color: white;
            text-align: center;
        }

        .container {
            background: rgba(0, 0, 0, 0.6); /* Semi-transparent background */
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
            max-width: 600px;
            width: 100%;
            text-align: center;
        }

        h1 {
            margin-bottom: 20px;
            font-size: 2rem;
        }

        .nav-links {
            position: absolute;
            top: 20px;
            left: 20px;
        }

        .nav-links a {
            color: white;
            text-decoration: none;
            background: rgba(0, 0, 0, 0.7);
            padding: 10px 15px;
```

```
            border-radius: 5px;
        }

        .btn {
            display: inline-block;
            margin-top: 20px;
            padding: 10px 20px;
            background-color: #2575fc;
            color: white;
            text-decoration: none;
            border-radius: 5px;
        }

        .btn:hover {
            background-color: #6a11cb;
        }
    </style>
</head>
<body>
    <div class="nav-links">
        <a href="/">Home</a>
    </div>
    <div class="container">
        <h1>{{ prediction_text }}</h1>
        <a href="/predict" class="btn">Try Another Prediction</a>
    </div>
</body>
</html>
```

App.py

```
from flask import Flask, request, render_template
import numpy as np
import pickle
import joblib

app = Flask(__name__)

# Load the model
model = joblib.load('usp.pkl')

@app.route("/")
def index():
    return render_template("index.html")

# Prediction page route
@app.route('/predict', methods=['GET', 'POST'])
def predict():
    if request.method == 'POST':
        # Extracting input features from the form
        qoe = float(request.form['qoe'])
        ae = float(request.form['ae'])
        qof = float(request.form['qof'])
        publ = float(request.form['publ'])
        inf = float(request.form['inf'])
        cit = float(request.form['cit'])
        pat = float(request.form['pat'])

        # Assuming you have 6 more features, extract them similarly
        # Adjust the number of features accordingly based on your model

        # Making prediction using the model
        input_features = np.array([[qoe, ae, qof, publ, inf, cit, pat, 0, 0, 0, 0, 0, 0]])
        prediction = model.predict(input_features)
        predicted_score = prediction[0]

        # Pass the prediction result to the result.html page
        return render_template('result.html', prediction_text=f"Predicted University Score: {predicted_score:.2f}")

    # If GET method or any other method, return predict.html
    return render_template('predict.html')

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

CODE SNIPPETS

DATA COLLECTION

Importing necessary libraries

```
[1] import pandas as pd
    import numpy as np
    from sklearn.preprocessing import LabelEncoder
    from sklearn.preprocessing import StandardScaler
    from sklearn.model_selection import train_test_split
    from sklearn import linear_model
    from sklearn.linear_model import LinearRegression
    from sklearn.svm import SVR
    from sklearn.tree import DecisionTreeRegressor
    from sklearn.ensemble import RandomForestRegressor
    from xgboost import XGBRegressor
    from sklearn.metrics import mean_absolute_error
    from sklearn.metrics import mean_squared_error
    from sklearn.metrics import r2_score
    import seaborn as sns
    import matplotlib.pyplot as plt
    %matplotlib inline
    import pickle
    import warnings
    warnings.filterwarnings('ignore')
```

Reading Dataset

```
[8] cwur = pd.read_csv("/content/cwurData.csv")
```

```
cwur.head()
```

	world_rank	institution	country	national_rank	quality_of_education	alumni_employment	quality_of_faculty	publications	influence
0	1	Harvard University	USA	1	7	9	1	1	1
1	2	Massachusetts Institute of Technology	USA	2	9	17	3	12	4
2	3	Stanford University	USA	3	17	11	5	4	2
3	4	University of Cambridge	United Kingdom	1	10	24	4	16	16
4	5	California Institute of Technology	USA	4	2	29	7	37	22

Dataset shape

```
[10] cwur.shape
```

```
(2200, 14)
```

DATA PREPROCESSING

Datatypes

```
cwur.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2200 entries, 0 to 2199
Data columns (total 14 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   world_rank           2200 non-null   int64  
 1   institution          2200 non-null   object 
 2   country              2200 non-null   object 
 3   national_rank        2200 non-null   int64  
 4   quality_of_education 2200 non-null   int64  
 5   alumni_employment    2200 non-null   int64  
 6   quality_of_faculty   2200 non-null   int64  
 7   publications         2200 non-null   int64  
 8   influence            2200 non-null   int64  
 9   citations            2200 non-null   int64  
 10  broad_impact         2000 non-null   float64
 11  patents              2200 non-null   int64  
 12  score                2200 non-null   float64
 13  year                 2200 non-null   int64  
dtypes: float64(2), int64(10), object(2)
memory usage: 240.8+ KB
```

Handling null Values

```
np.sum(cwur.isnull())
```

```
world_rank              0
institution             0
country                 0
national_rank           0
quality_of_education    0
alumni_employment       0
quality_of_faculty      0
publications            0
influence               0
citations               0
broad_impact            0
patents                 0
score                   0
year                    0
dtype: int64
```

Handling Categorical Values

```
datTypeSeries = cwur.dtypes
print("Data type of each column of timesData Dataframe :")
print(datTypeSeries)
```

```
Data type of each column of timesData Dataframe :
world_rank                 int64
institution               object
country                   object
national_rank              int64
quality_of_education       int64
alumni_employment          int64
quality_of_faculty         int64
publications               int64
influence                  int64
citations                  int64
broad_impact             float64
patents                    int64
score                    float64
year                       int64
dtype: object
```

```
[16] from sklearn.preprocessing import LabelEncoder
     le=LabelEncoder()
```

```
[17] cwur["institution"]=le.fit_transform(cwur["institution"])
     cwur["country"]=le.fit_transform(cwur["country"])
```

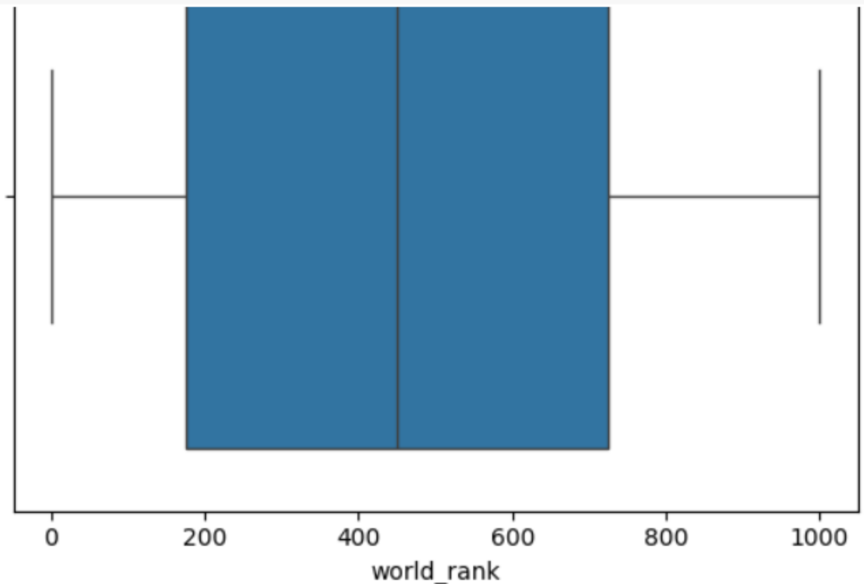
```
cwur.head()
```

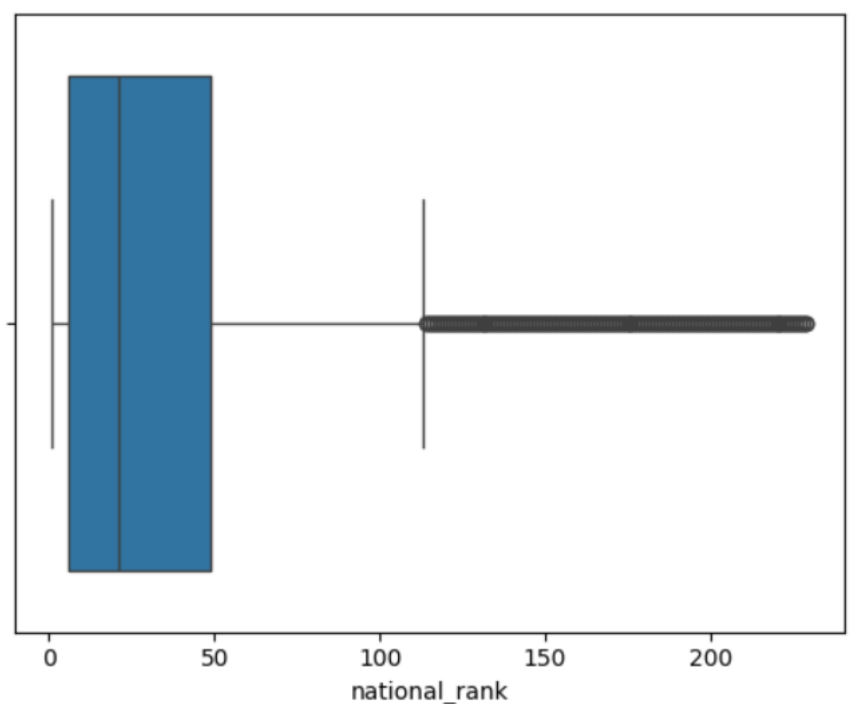
	world_rank	institution	country	national_rank	quality_of_education	alumni_employment	quality_of_faculty	publications	influence	ci
0	1	184	54	1	7	9	1	1	1	
1	2	312	54	2	9	17	3	12	4	
2	3	511	54	3	17	11	5	4	2	
3	4	637	57	1	10	24	4	16	16	
4	5	53	54	4	2	29	7	37	22	

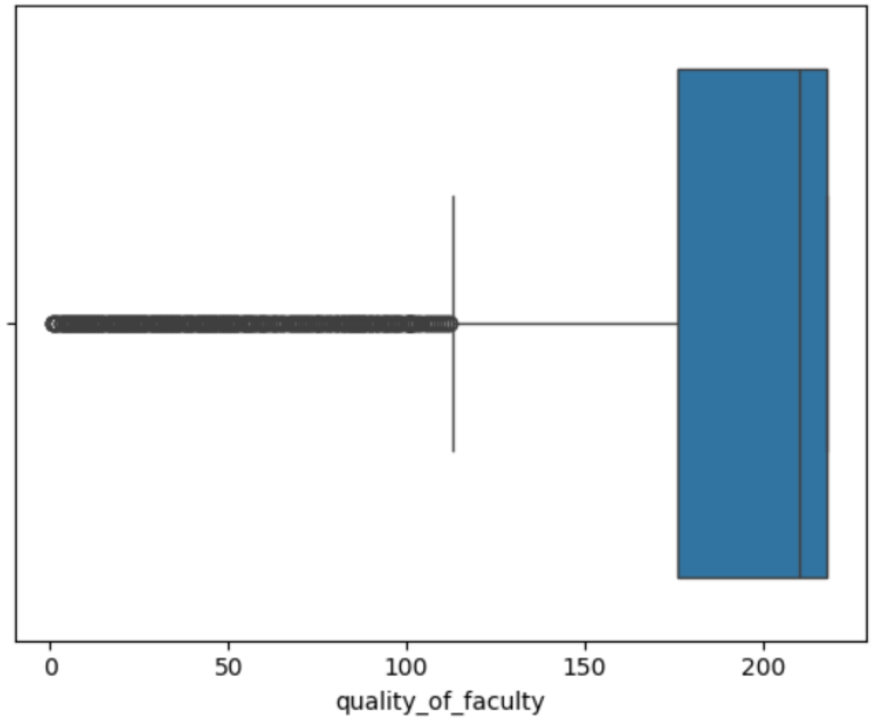
Handling outliers

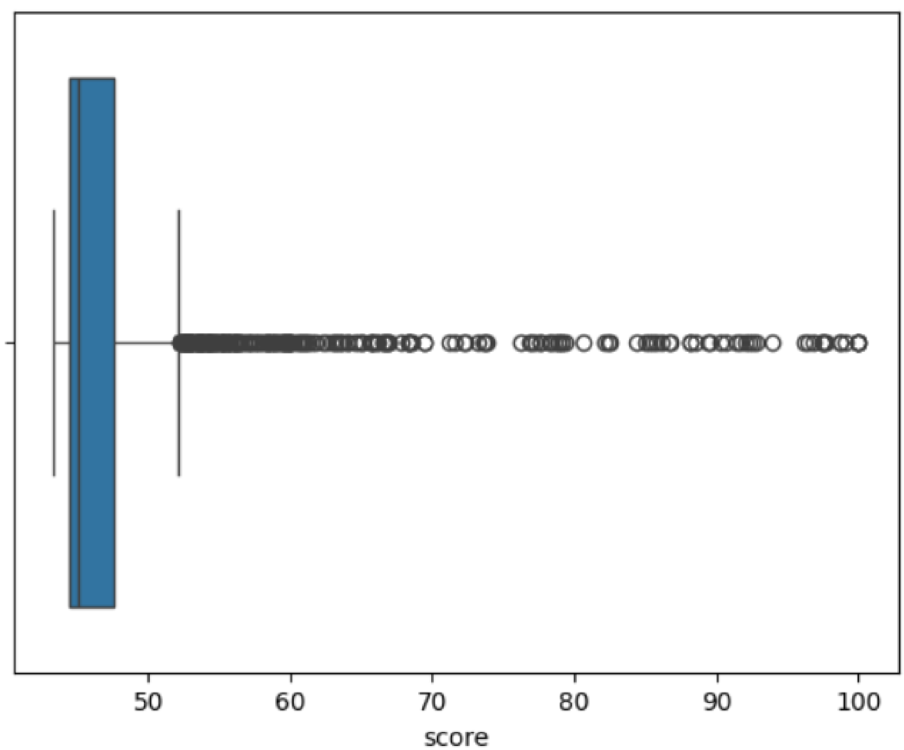
[19]
```
def fun(col):
  sns.boxplot(x=col,data=cwur)
  plt.show()
```

```
for i in cwur.columns:
  fun(i)
```





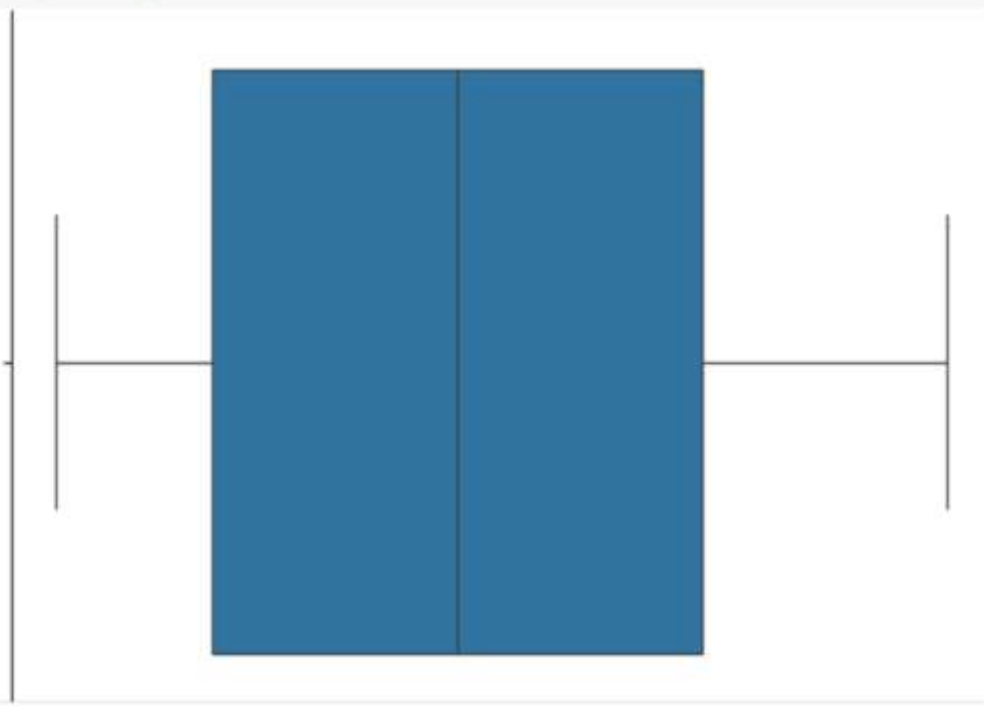

0
50
100
150
200
quality_of_faculty


50
60
70
80
90
100
score

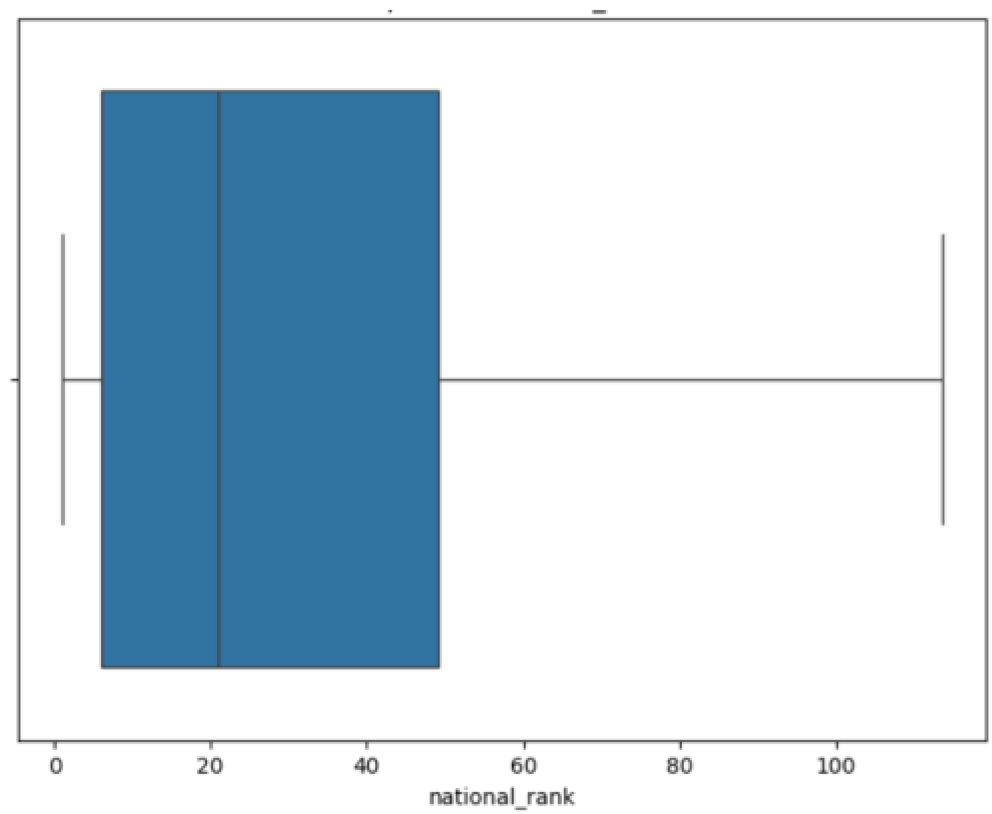
Data after removing outliers

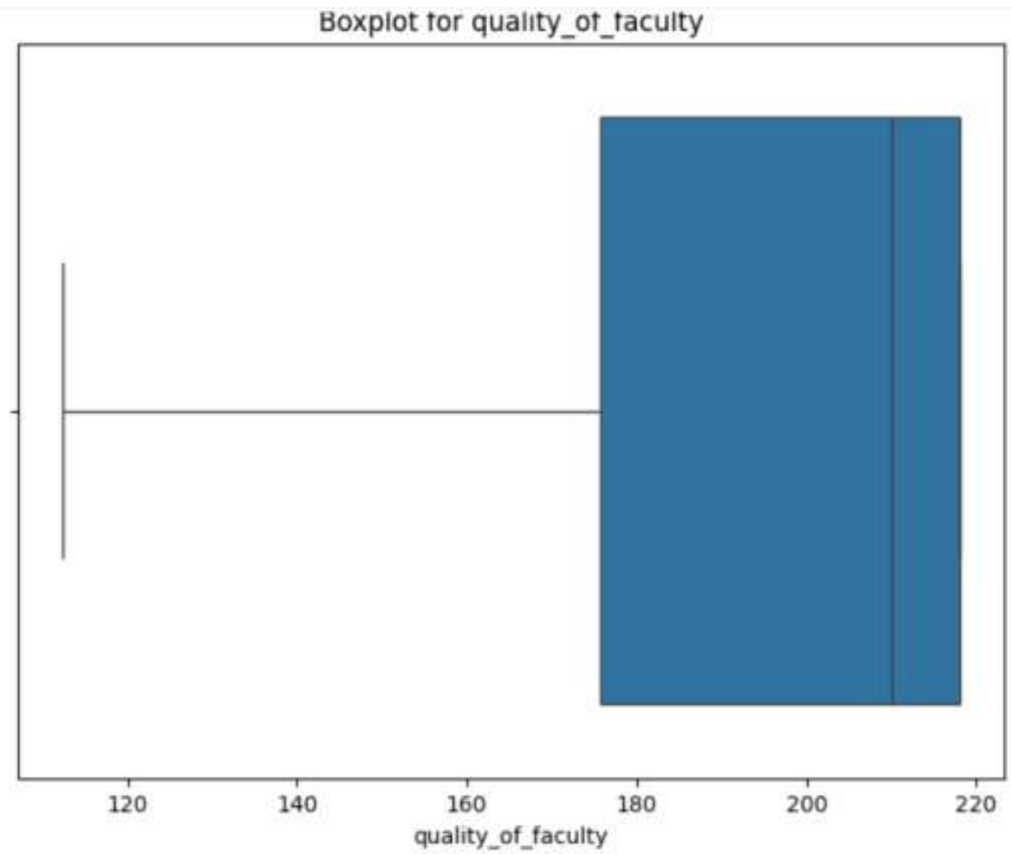
```
# Iterate over each column
for column in cwur.columns:
    # Check if the column contains numeric data
    if pd.api.types.is_numeric_dtype(cwur[column]):
        # Calculate quantiles
        quant = cwur[column].quantile(q=[0.75, 0.25])
        Q3 = quant.loc[0.75]
        Q1 = quant.loc[0.25]

        # Calculate IQR
        IQR = Q3 - Q1

        # Calculate lower and upper bounds for outliers
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR

        # Replace outliers with values within the bounds
        cwur[column] = np.where(cwur[column] < lower_bound, lower_bound,cwur[column])
        cwur[column] = np.where(cwur[column] > upper_bound, upper_bound, cwur[column])

# Now all outliers in the specified columns have been replaced with values within the IQR bounds
```

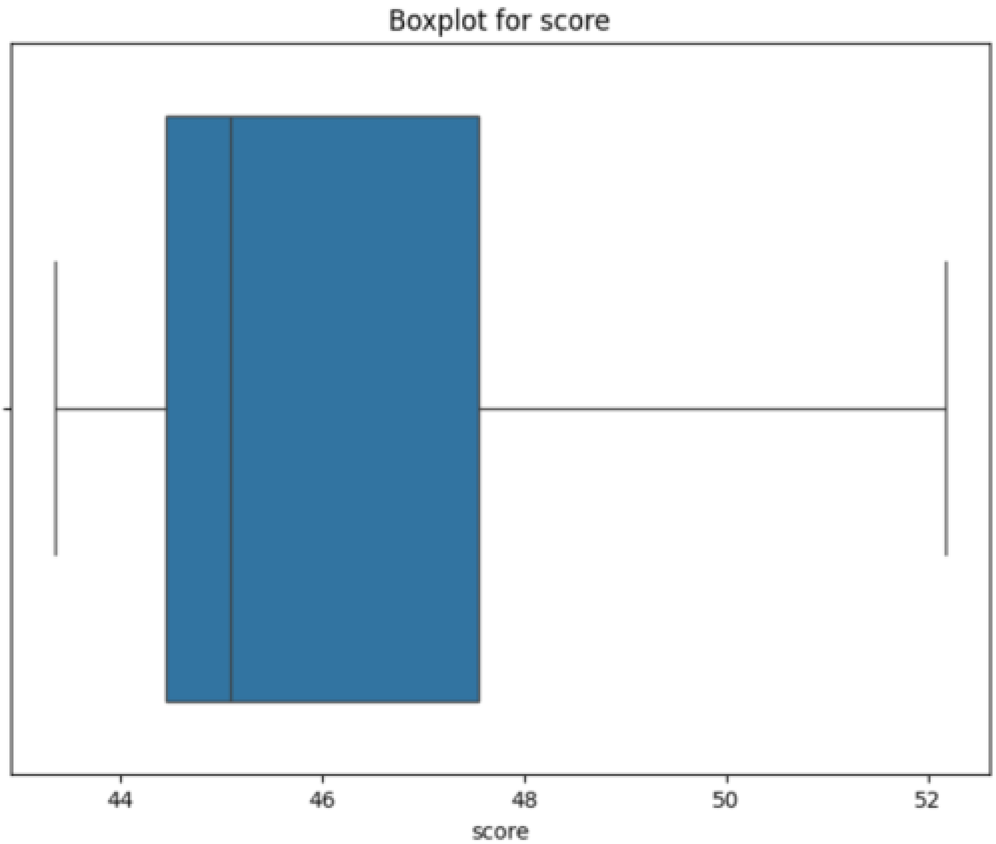
```
# Iterate over each column and plot boxplot
for column in cwur.columns:
    plt.figure(figsize=(8, 6))  # Adjust the figure size as needed
    sns.boxplot(x=cwur[column])
    plt.title(f'Boxplot for {column}')
    plt.xlabel(column)
    plt.show()
```



✓ 0s completed at 3:05PM


0
20
40
60
80
100
national_rank


Boxplot for quality_of_faculty
120
140
160
180
200
220
quality_of_faculty



Descriptive statistical

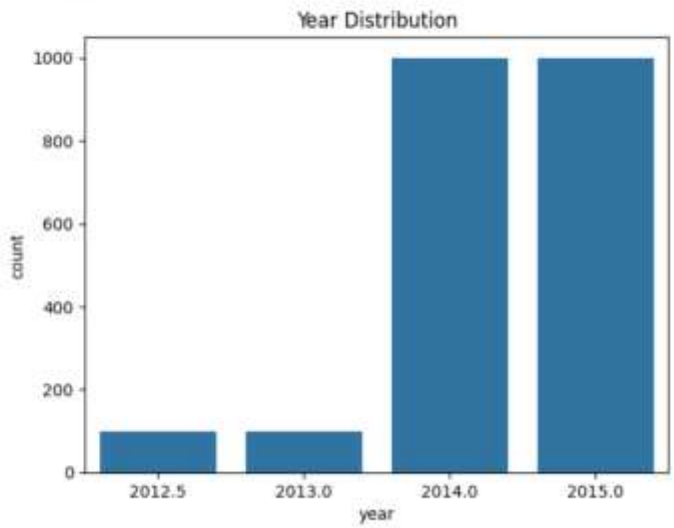
```
cwur.describe(include = "all")
```

	world_rank	institution	country	national_rank	quality_of_education	alumni_employment	quality_of_faculty	publications	in
count	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000	2200
mean	459.590909	516.390909	34.110455	34.161818	275.100455	357.116818	189.660000	459.908636	459
std	304.320363	294.908607	19.211020	35.642332	121.935100	186.779252	41.673073	303.760352	303
min	1.000000	0.000000	0.000000	1.000000	1.000000	1.000000	112.375000	1.000000	1
25%	175.750000	263.750000	17.000000	6.000000	175.750000	175.750000	175.750000	175.750000	175
50%	450.500000	521.000000	33.000000	21.000000	355.000000	450.500000	210.000000	450.500000	450
75%	725.250000	770.250000	54.000000	49.000000	367.000000	478.000000	218.000000	725.000000	725
max	1000.000000	1023.000000	58.000000	113.500000	367.000000	567.000000	218.000000	1000.000000	991

Visual Analysis

Univariate Analysis

```
sns.countplot(x='year', data=cwur)
plt.title('Year Distribution')
plt.show()
```

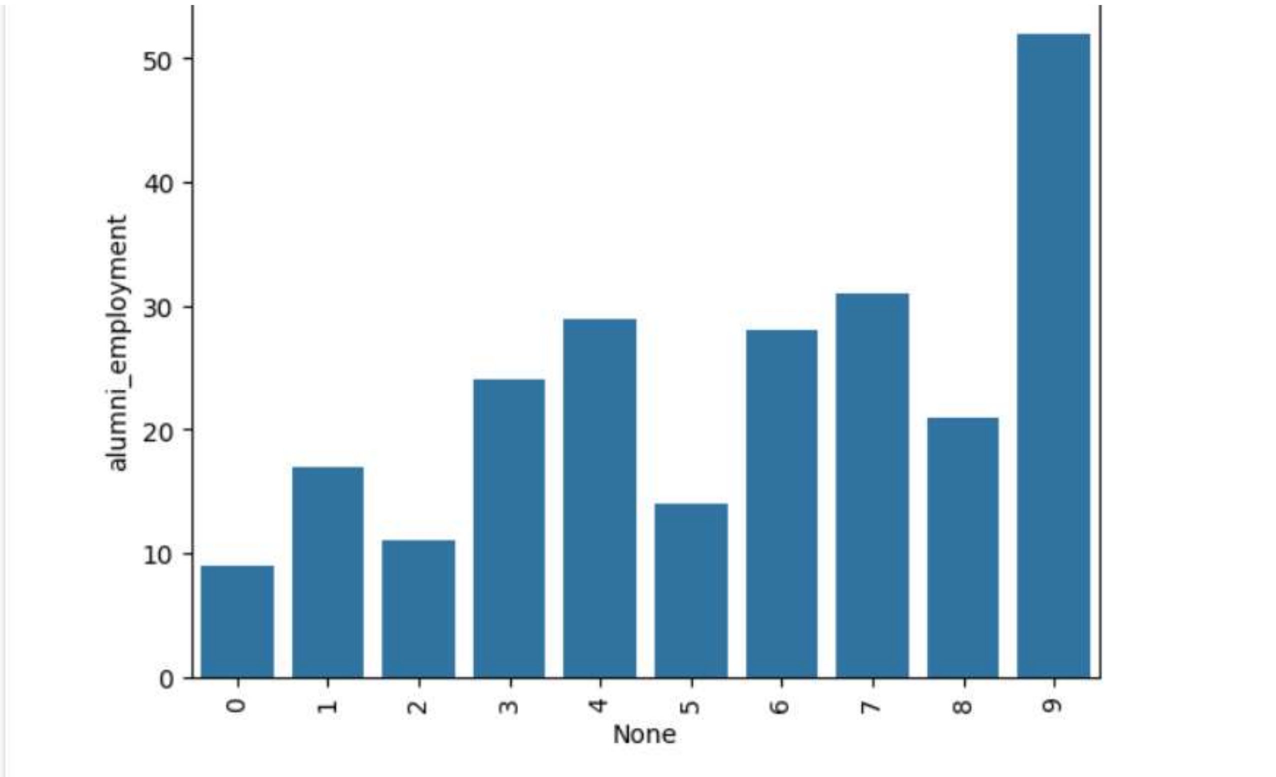


Bivariate Analysis

```
Top10 = cwur.head(10)
```

```
sns.barplot(x = Top10.index,y = 'alumni_employment',data = Top10).set_xticklabels(labels = Top10.index,rotation = 90)
```

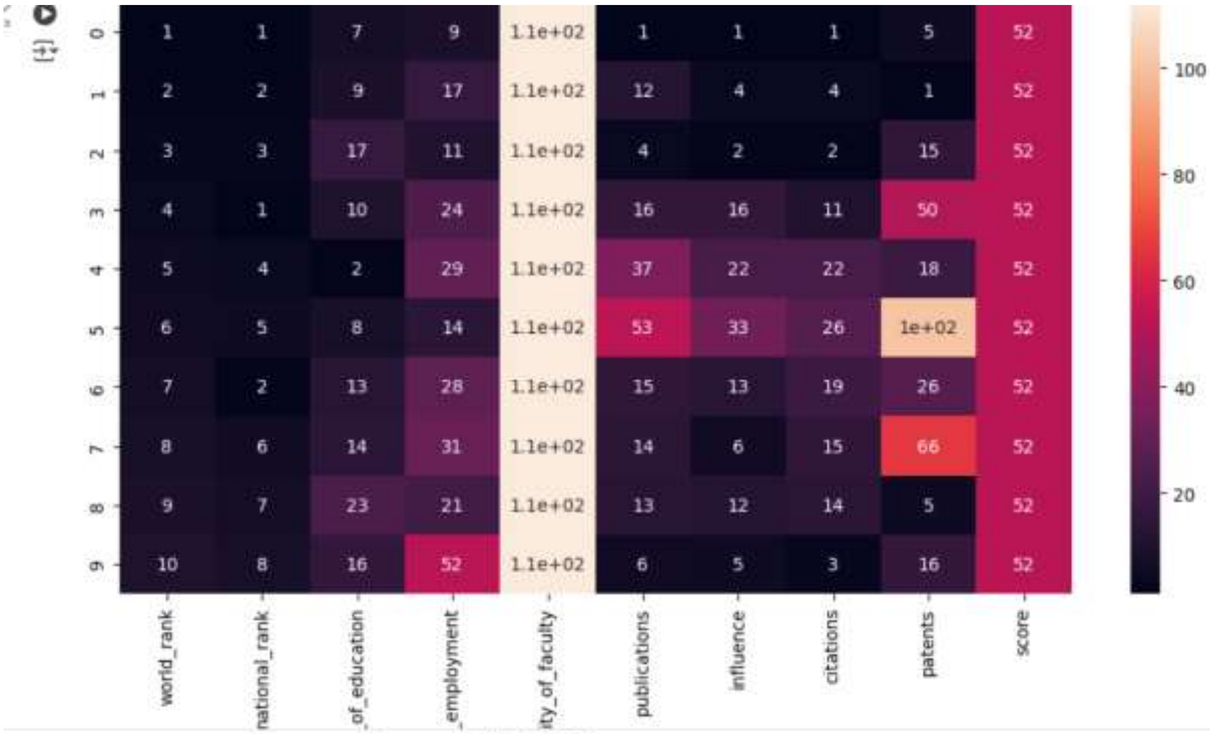
```
[Text(0, 0, '0'),
 Text(1, 0, '1'),
 Text(2, 0, '2'),
 Text(3, 0, '3'),
 Text(4, 0, '4'),
 Text(5, 0, '5'),
 Text(6, 0, '6'),
 Text(7, 0, '7'),
 Text(8, 0, '8'),
 Text(9, 0, '9')]
```



Multivariate Analysis

```
topi = cwur.head(10)
#topi.info()
topi_f = topi.loc[:,['world_rank','national_rank','quality_of_education','alumni_employment','quality_of_faculty','publications','influen
plt.figure(figsize = (12,6))
sns.heatmap(data = topi_f,annot = True)
plt.show
```

```
matplotlib.pyplot.show
def show(*args, **kwargs)

/usr/local/lib/python3.10/dist-packages/matplotlib/pyplot.py
Display all open figures.

Parameters
----------
block : bool, optional
```



✓ 0s completed at 3:05PM

Splitting data into train and test

```
[30] cwur.head()
```

	world_rank	institution	country	national_rank	quality_of_education	alumni_employment	quality_of_faculty	publications	influence	c
0	1.0	184.0	54.0	1.0	7.0	9.0	112.375	1.0	1.0	
1	2.0	312.0	54.0	2.0	9.0	17.0	112.375	12.0	4.0	
2	3.0	511.0	54.0	3.0	17.0	11.0	112.375	4.0	2.0	
3	4.0	637.0	57.0	1.0	10.0	24.0	112.375	16.0	16.0	
4	5.0	53.0	54.0	4.0	2.0	29.0	112.375	37.0	22.0	

Next steps: Generate code with cwur View recommended plots

```
[31] x = cwur.drop('score',axis=1)
     y = cwur['score']
```

```
[32] print(x)
```

```
      world_rank  institution  country  national_rank  quality_of_education  \
0            1.0        184.0     54.0            1.0                   7.0
1            2.0        312.0     54.0            2.0                   9.0
2            3.0        511.0     54.0            3.0                  17.0
3            4.0        637.0     57.0            1.0                  10.0
4            5.0         53.0     54.0            4.0                   2.0
...          ...          ...      ...            ...                   ...
2195       996.0        954.0     37.0            7.0                 367.0
```

```
[33] print(y)
```

```
0       52.1725
1       52.1725
2       52.1725
3       52.1725
4       52.1725
         ...
2195    44.0300
2196    44.0300
2197    44.0300
2198    44.0200
2199    44.0200
Name: score, Length: 2200, dtype: float64
```

```
[34] x.shape
```

```
(2200, 13)
```

```
y.shape
```

```
(2200,)
```

```
[36] x_train,x_test,y_train,y_test = train_test_split(x,y,train_size = 0.8 , random_state=42)
```

Training The Model In Multiple Algorithms

Linear Regression model

```
[37] linReg = LinearRegression()
     linReg.fit(x_train,y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
[38] y_pred = linReg.predict(x_test)
```

```
[39] accuracy = linReg.score(x_test,y_test)
     print(accuracy)
```

```
0.7439493774592185
```

Lasso Regression model

```
[40] lassoReg = linear_model.Lasso(alpha = 0.1)
     lassoReg.fit(x,y)
```

```
Lasso
Lasso(alpha=0.1)
```

```
[41] y_pred = lassoReg.predict(x_test)
```

```
[42] accuracy = lassoReg.score(x_test,y_test)
     print(accuracy)
```

```
0.7444199332854502
```

SVM Model:

```
[43] svr = SVR().fit(x,y)
```

```
[44]
     y_pred = svr.predict(x_test)
```

```
[45] accuracy = svr.score(x_test,y_test)
     print(accuracy)
```

```
0.8223608225568596
```

Decision Tree Model:

```
[46] dt = DecisionTreeRegressor(random_state = 0)
     dt.fit(x,y)
```

```
DecisionTreeRegressor
DecisionTreeRegressor(random_state=0)
```

```
[47] y_pred = dt.predict(x_test)
```

```
[48] accuracy = dt.score(x_test,y_test)
     print(accuracy)
```

```
1.0
```

Random Forest Model:

```
[49] rf = RandomForestRegressor(n_estimators = 100 , random_state = 0)
     rf.fit(x,y)
```

```
RandomForestRegressor
RandomForestRegressor(random_state=0)
```

```
[50] y_pred = rf.predict(x_test)
```

```
[51] accuracy = rf.score(x_test,y_test)
     print(accuracy)
```

```
0.9998704251212489
```

Testing The Model

```python
cwur.head()
```

	world_rank	institution	country	national_rank	quality_of_education	alumni_employment	quality_of_faculty	publications	influence	c
0	1.0	184.0	54.0	1.0	7.0	9.0	112.375	1.0	1.0	
1	2.0	312.0	54.0	2.0	9.0	17.0	112.375	12.0	4.0	
2	3.0	511.0	54.0	3.0	17.0	11.0	112.375	4.0	2.0	
3	4.0	637.0	57.0	1.0	10.0	24.0	112.375	16.0	16.0	
4	5.0	53.0	54.0	4.0	2.0	29.0	112.375	37.0	22.0	

Next steps: Generate code with cwur | View recommended plots

```python
[53] cwur.tail()
```

world_rank	institution	country	national_rank	quality_of_education	alumni_employment	quality_of_faculty	publications	influence	citat
996.0	954.0	37.0	7.0	367.0	567.0	218.0	926.0	845.0	8
997.0	11.0	14.0	4.0	236.0	566.0	218.0	997.0	908.0	6
998.0	132.0	4.0	18.0	367.0	549.0	218.0	830.0	823.0	8
999.0	576.0	48.0	40.0	367.0	567.0	218.0	886.0	974.0	8
1000.0	74.0	8.0	83.0	367.0	567.0	218.0	861.0	991.0	8

```python
dt.predict([[1.0, 184.0,  54.0, 1.0,  7.0,  9.0,  112.375,  1.0 ,1.0, 1.0,496.0,5.0,2012.5]])
```

```
array([52.1725])
```

```python
[55] dt.predict([[996.0,954.0,37.0,7.0,367.0,567.0 ,218.0,926.0,845.0,812.0,969.0,816.0,2015.0]])
```

```
array([44.03])
```


```python
[53] cwur.tail()
```

world_rank	institution	country	national_rank	quality_of_education	alumni_employment	quality_of_faculty	publications	influence	citat
996.0	954.0	37.0	7.0	367.0	567.0	218.0	926.0	845.0	8
997.0	11.0	14.0	4.0	236.0	566.0	218.0	997.0	908.0	6
998.0	132.0	4.0	18.0	367.0	549.0	218.0	830.0	823.0	8
999.0	576.0	48.0	40.0	367.0	567.0	218.0	886.0	974.0	8
1000.0	74.0	8.0	83.0	367.0	567.0	218.0	861.0	991.0	8

```python
dt.predict([[1.0, 184.0,  54.0, 1.0,  7.0,  9.0,  112.375,  1.0 ,1.0, 1.0,496.0,5.0,2012.5]])
```

```
array([52.1725])
```

```python
[55] dt.predict([[996.0,954.0,37.0,7.0,367.0,567.0 ,218.0,926.0,845.0,812.0,969.0,816.0,2015.0]])
```

```
array([44.03])
```


Testing Model With Multiple Evaluation Metrics

Compare the Model

[56]
```
# Assuming 'x_test' is available in the environment and is a pandas DataFrame or a NumPy array.
y_pred = linReg.predict(x_test)  # Predict on the entire x_test dataset

print("Prediction Evaluation using Linear Regression")
print('Mean Absolute Error:', mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test, y_pred)))
print('R-squared:', r2_score(y_test, y_pred))
```

```
Prediction Evaluation using Linear Regression
Mean Absolute Error: 0.9264657671450711
Mean Squared Error: 1.7890643253785259
Root Mean Squared Error: 1.337559092294066
R-squared: 0.7439493774592185
```

```
y_pred = lassoReg.predict(x_test)
print("Prediction Evaluation using lasso Regression")
print('Mean Absolute Error:', mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test, y_pred)))
print('R-squared:', r2_score(y_test, y_pred))
```

```
Prediction Evaluation using lasso Regression
Mean Absolute Error: 0.9352851280381133
Mean Squared Error: 1.7857764808364731
Root Mean Squared Error: 1.3363294806433303
R-squared: 0.7444199332854502
```

[58]
```
y_pred = svr.predict(x_test)
print("Prediction Evaluation using  support vector Regression")
print('Mean Absolute Error:', mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test, y_pred)))
print('R-squared:', r2_score(y_test, y_pred))
```

```
Prediction Evaluation using  support vector Regression
Mean Absolute Error: 0.5454340693726399
Mean Squared Error: 1.2411917299771091
Root Mean Squared Error: 1.1140878466158355
R-squared: 0.8223608225568596
```

```
y_pred = dt.predict(x_test)
print("Prediction Evaluation using Decision Regression ")
print('Mean Absolute Error:', mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test, y_pred)))
print('R-squared:', r2_score(y_test, y_pred))
```

```
Prediction Evaluation using Decision Regression
Mean Absolute Error: 5.264475724040743e-15
Mean Squared Error: 2.7561365735867205e-28
Root Mean Squared Error: 1.6601616106833456e-14
R-squared: 1.0
```

```
[60] y_pred = rf.predict(x_test)
     print("Prediction Evaluation using Random Regression")
     print('Mean Absolute Error:', mean_absolute_error(y_test, y_pred))
     print('Mean Squared Error:', mean_squared_error(y_test, y_pred))
     print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test, y_pred)))
     print('R-squared:', r2_score(y_test, y_pred))
```

```
Prediction Evaluation using Random Regression
Mean Absolute Error: 0.010686590909099649
Mean Squared Error: 0.0009053592244319952
Root Mean Squared Error: 0.0300891878327082
R-squared: 0.9998704251212489
```

Saving the model

```
[61] import pickle
     pickle.dump(rf,open('usp.pkl','wb'))
```

10.2 Github & project Demo Link: