

Introduction and Summary of the Project

The Global AI Index is the first index to benchmark nations on their level of investment, innovation and implementation of artificial intelligence. We have chosen the required dataset which is providing the analysis of multiple indicators affecting the index on 62 countries. The dataset draws upon a wide range of sources, including scientific publications, investment reports, and government documents, to provide a holistic overview of global AI progress. Countries are scored across several key dimensions: AI research (number of publications and citations), investment in AI (private and public spending), AI talent (availability of skilled AI professionals), AI startups and companies, and government strategies (such as national AI policies and regulatory frameworks). This multifaceted approach allows the dataset to capture not just technological advancement but also the broader economic, policy, and social contexts that drive AI development.

We have used predictive analytics to forecast future trends in AI development by analyzing the current data, providing insights into which countries are likely to emerge as AI leaders in the coming years. With the growing importance of AI in shaping industries and economies, datasets like this are crucial for stakeholders looking to understand the competitive AI landscape globally.

Some of the interesting outcome of the research

1. Research, Development, Talent, and Commercial variables have the most tangible contribution to Total score of AI Index
2. Government Strategy and Infrastructure have less impact on the overall AI Index
3. The United States is the top country in the number of AI talents (an average score of all factors is 100)
4. India does not fit into the top ten list of the countries by Total score, although it holds the second position in terms of AI Talent score
5. The Top 5 countries with the highest AI implementation level are the United States, The Netherlands, Canada, China, and Great Britain.
6. Only 35% of analyzed countries have an AI implementation score higher than 50.
7. China's Infrastructure Is Most Appropriate for AI
8. The Top 5 countries with the highest level of innovation in AI are the United States, China, South Korea, Australia, and Switzerland.

9. However, only the first two ones from the point above have an innovation score higher than 50. It testifies to the low level of AI development among the countries (in comparison with the leader - the USA) despite the shreds of evidence of AI worldwide exponential growth.

US-based specialists generate more AI ideas

1. AI development in the USA is at the highest Level
2. Almost 70% of the countries in the dataset are ready for AI evolution at the governmental level; their Government Strategy indicator score is higher than 50.
3. The Top 5 countries in the Government AI Strategy rate are Canada, China, Saudi Arabia, Spain, and France (it is 5 out of 6 countries with a score higher than 90).
4. The United States is leading in the commercialization of AI and has no direct competition in this field

As researchers , this dataset is valuable in analyzing how nations are advancing AI. It helps identify areas where countries excel or need improvement, supporting informed decisions regarding AI policies, research funding, and international collaboration.

About the Data

The Dataset "**AI Global index**" includes The Global AI Index itself and seven indicators affecting the Index on 62 countries, as well as general information about the countries (region, cluster, income group and political regime).

The **Global AI Index** is the first index to benchmark nations on their level of investment, innovation and implementation of artificial intelligence.

Talent, Infrastructure and Operating Environment are the factors of **AI Implementation** group of indicators, which represents the application of artificial intelligence by professionals in various sectors, such as businesses, governments, and communities.

- **Talent** indicator focuses on the availability of skilled practitioners for the provision of artificial intelligence solutions.
- **Infrastructure** indicator focuses on the reliability and scale of access infrastructure, from electricity and internet, to super computing capabilities.
- **Operating Environment** indicator focuses on the regulatory context, and public opinion surrounding artificial intelligence.

Research and Development are the factors of **Innovation** group of indicators, which reflects the progress made in technology and methodology, which signify the potential for artificial intelligence to evolve and improve.

- **Research** indicator focuses on the extent of specialist research and researchers; investigating the amount of publications and citations in credible academic journals.
- **Development** indicator focuses on the development of fundamental platforms and algorithms upon which innovative artificial intelligence projects rely.

Government Strategy and Commercial are the factors of **Investment** group of indicators, which reflects financial and procedural commitments to artificial intelligence.

- **Government Strategy** indicator focuses on the depth of commitment from national government to artificial intelligence; investigating spending commitments and national strategies.
- **Commercial indicator** focuses on the level of startup activity, investment and business initiatives based on artificial intelligence.

```
In [1]: import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import matplotlib.cm as cm
import seaborn as sns

import plotly.graph_objects as go
from plotly.subplots import make_subplots
import plotly.express as px
import plotly.offline
```

Initial Data Inspection

```
In [2]: #from google.colab import drive
#drive.mount('/content/drive')

import pandas as pd
import gdown

# Modify the Google Drive link to a downloadable format
file_url = "https://drive.google.com/uc?id=1JIF7XyyL0fhe8vGds6UB_DayYh1PtJKz

# Download the file using gdown
gdown.download(file_url, 'data.csv', quiet=False)
```

```
# Read the downloaded CSV file
df = pd.read_csv('data.csv')

# Display the first few rows of the dataframe
df.head()
```

```
Downloading...
From: https://drive.google.com/uc?id=1JIF7XyyL0fhe8vGds6UB_DayYhlPtJKz
To: c:\SanDiago\Project\data.csv
100%|██████████| 6.38k/6.38k [00:00<00:00, 6.20MB/s]
```

Out[2]:

| | Country | Talent | Infrastructure | Operating Environment | Research | Development | Gov |
|---|--------------------------|--------|----------------|-----------------------|----------|-------------|-----|
| 0 | United States of America | 100.00 | 94.02 | 64.56 | 100.00 | 100.00 | |
| 1 | China | 16.51 | 100.00 | 91.57 | 71.42 | 79.97 | |
| 2 | United Kingdom | 39.65 | 71.43 | 74.65 | 36.50 | 25.03 | |
| 3 | Canada | 31.28 | 77.05 | 93.94 | 30.67 | 25.78 | |
| 4 | Israel | 35.76 | 67.58 | 82.44 | 32.63 | 27.96 | |

After the initial data inspection, we can confirm what conveys on the dataset

✓ The Global AI Index (GAI) is an indicator to rank countries based on the capacity for AI, specifically by measuring investment, innovation, and implementation levels. According to the previous and current values of the GAI, Tortoise Media divides countries into five clusters - power players, traditional champions, rising stars, waking-up players, and nascent players.

According to the outcome of the above program you can see the two power players are on the map - the United States and China

The following two countries, the United Kingdom and Canada are traditional champions .

Germany and France are the other two traditional champions

The other rising stars are Australia, Ireland, Finland, Denmark, Japan, and Spain, Israel, Singapore, South Korea, and the Netherlands

All the other countries (among the analyzed 62 countries) except Sri Lanka, Egypt, Kenya, Nigeria, and Pakistan are the waking-up

players; the last are nascent AI market players.

```
In [3]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 62 entries, 0 to 61
Data columns (total 13 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Country                               62 non-null     object
1   Talent                               62 non-null     float64
2   Infrastructure                         62 non-null     float64
3   Operating Environment                 62 non-null     float64
4   Research                             62 non-null     float64
5   Development                           62 non-null     float64
6   Government Strategy                   62 non-null     float64
7   Commercial                           62 non-null     float64
8   Total score                           62 non-null     float64
9   Region                               62 non-null     object
10  Cluster                              62 non-null     object
11  Income group                          62 non-null     object
12  Political regime                      62 non-null     object
dtypes: float64(8), object(5)
memory usage: 6.4+ KB
```

```
In [4]: df.describe()
```

```
Out[4]:
```

| | Talent | Infrastructure | Operating Environment | Research | Development | Go |
|--------------|------------|----------------|-----------------------|------------|-------------|----|
| count | 62.000000 | 62.000000 | 62.000000 | 62.000000 | 62.000000 | |
| mean | 16.803065 | 63.503710 | 66.925484 | 16.610000 | 14.824677 | |
| std | 15.214963 | 20.217525 | 20.000424 | 17.413996 | 19.419279 | |
| min | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | |
| 25% | 7.365000 | 55.857500 | 58.107500 | 3.032500 | 1.202500 | |
| 50% | 13.445000 | 65.230000 | 69.505000 | 12.930000 | 9.005000 | |
| 75% | 24.567500 | 75.947500 | 80.500000 | 25.412500 | 19.980000 | |
| max | 100.000000 | 100.000000 | 100.000000 | 100.000000 | 100.000000 | 1 |

We can see that

- There are no missing values in the dataset , with 62 countries (records) presented there to describe their AI potential as of 2023
- There are 8 numeric and 4 categorical variables in the dataset
- All numeric variables have their values in the range from 0 to 100
- *Total score* variable can be considered to be the *target* variable whereas other variables can describe the various effects on it

Express EDA of Numeric Variables

numeric variables in our dataset.

Univariate Analysis: Numerical variables

Let's look at the distribution of the numeric variables

```
In [5]: fig = make_subplots(rows=2, cols=4, subplot_titles=('<b>Distribution of Tale
                                                    '<b>Distribution of Infr
                                                    '<b>Distribution of Oper
                                                    '<b>Distribution of Rese
                                                    '<b>Distribution of Deve
                                                    '<b>Distribution of Gove
                                                    '<b>Distribution of Comm
                                                    '<b>Distribution of Tota
                                                    ))

fig.add_trace(go.Histogram(x=df['Talent'], nbinsx=30), row=1, col=1)
fig.add_trace(go.Histogram(x=df['Infrastructure'], row=1, col=2)
fig.add_trace(go.Histogram(x=df['Operating Environment'], nbinsx=30), row=1,
fig.add_trace(go.Histogram(x=df['Research'], nbinsx=30), row=1, col=4)

fig.add_trace(go.Histogram(x=df['Development'], nbinsx=30), row=2, col=1)
fig.add_trace(go.Histogram(x=df['Government Strategy'], row=2, col=2)
fig.add_trace(go.Histogram(x=df['Commercial'], nbinsx=30), row=2, col=3)
fig.add_trace(go.Histogram(x=df['Total score'], nbinsx=30), row=2, col=4)

# Update visual layout
fig.update_layout(
    showlegend=False,
    width=800,
    height=500,
    autosize=False,
    margin=dict(t=15, b=0, l=5, r=5),
    template="plotly_white",
)

# update font size at the axes
fig.update_coloraxes(colorbar_tickfont_size=10)
# Update font in the titles: Apparently subplot titles are annotations (Subp
fig.update_annotations(font_size=10)
# Reduce opacity
fig.update_traces(opacity=0.75)

fig.show()
```

We find that

- Infrastructure and Operating Environment are skewed to the right a little
- Other numeric variables are significantly left-skewed
- The values of Talent, Research, Development, Commercial and Total score have clear outliers to the right side of the distribution (it is related to two countries in Power players cluster)

Multivariate Analysis: Correlation between numerical variables

```
In [6]: sns.set_theme(style="white")

# make a dataframe with only numeric variables, without 'Outcome'
d = df.select_dtypes(include=np.number).copy() # Select only numeric columns

# Compute the correlation matrix for numeric features
corr = d.corr()

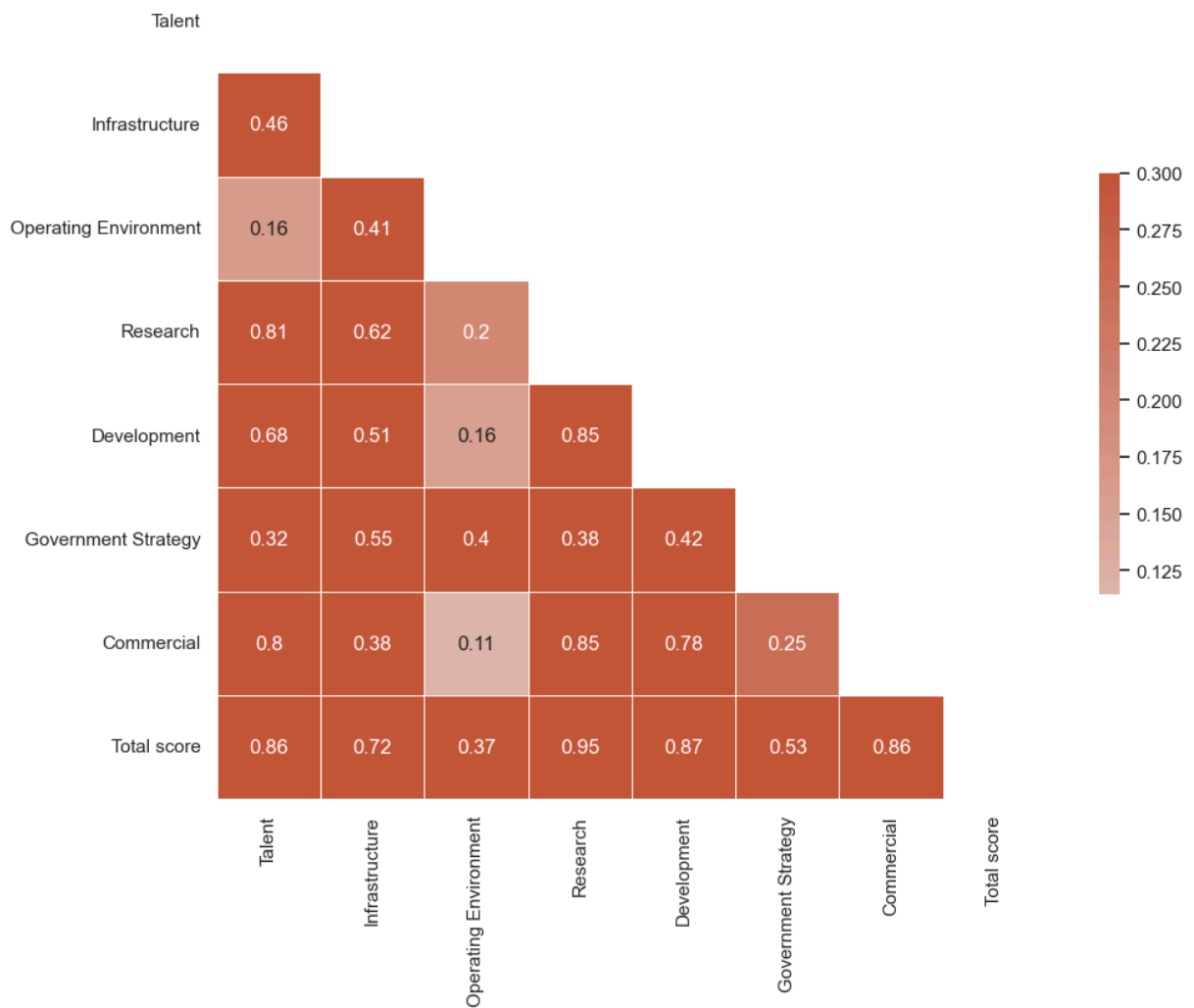
# Generate a mask for the upper triangle
mask = np.triu(np.ones_like(corr, dtype=bool))

# Set up the matplotlib figure
f, ax = plt.subplots(figsize=(11, 9))

# Generate a custom diverging colormap
cmap = sns.diverging_palette(230, 20, as_cmap=True)

# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr, mask=mask, cmap=cmap, vmax=.3, center=0, annot=True,
            square=True, linewidths=.5, cbar_kws={"shrink": .5})
```

Out[6]: <Axes: >



We find that

- Total score is highly correlated with Research (0.95), Development (0.87), Talent (0.86), and Commercial (0.86) variables
- Total score displays medium correlations with Government Strategy and Operating Environment variables (that is, the contribution of such variables to the Total score is less than the highly correlated variables)

Express EDA of Categorical Variables


Let's briefly look at the categorical variables in the dataset.

Univariate Analysis: Categorical variables


```
In [7]: dfg = df['Region'].value_counts().reset_index()
dfg.columns = ['Region', 'Quantity']
fig = px.bar(dfg, x='Region', y='Quantity',
             title='Number of Countries in Report by Region (Continent)',
             color_discrete_sequence=px.colors.qualitative.Prism)
fig.show()
```

We find that the dataset contains the information about

- 29 countries in *Europe*
- 14 countries in *Asia-Pacific* region
- 8 countries in *Americas* region
- 6 countries in the *Middle East*
- 5 countries in *Africa*

 **Note:** The data in this dataset is presented for the countries that decided to start serious efforts to adopt AI technologies. Countries that, are not on the way to adapt AI technologies are not listed there.

```
In [8]: dfg = df['Cluster'].value_counts().reset_index()
dfg.columns = ['Cluster', 'Quantity']
fig = px.bar(dfg, x='Cluster', y='Quantity',
             title='Number of Countries in Report by Cluster',
             color_discrete_sequence=px.colors.qualitative.Prism)
fig.show()
```

We find that

- *Power players* cluster contains two leading nations in terms of adopting the AI on a serious scale (these are USA and China)
- *Traditional champions* cluster unites 4 nations (UK, Canada, France, and Germany) that follow the power players
- *Rising stars* cluster aggregates 11 countries that are on a serious rise in terms of the AI development/commercializing progress
- *Waking up* cluster comprises 40 countries that just started smooth progress in AI
- *Nascent* cluster is a union of 5 countries that are at the very beginning of their way to adapt AI on the national level

```
In [9]: dfg = df['Income group'].value_counts().reset_index()
dfg.columns = ['Income group', 'Quantity']
fig = px.bar(dfg, x='Income group', y='Quantity',
             title='Number of Countries in Report by Income groups',
             color_discrete_sequence=px.colors.qualitative.Prism)
fig.show()
```

We find that

- There are 43 countries with *High* income level (out of 62 presented in the dataset)
- There are 9 countries with *Upper middle* level of incomes
- There are 8 countries with *Lower middle* level of incomes

💡 **Note:** As we can see, AI is expensive. Only countries with High- or Middle-level incomes can afford to participate in the AI races at the moment.

```
In [10]: dfg = df['Political regime'].value_counts().reset_index()
dfg.columns = ['Political regime', 'Quantity']
fig = px.bar(dfg, x='Political regime', y='Quantity',
             title='Number of Countries in Report by Political regimes',
             color_discrete_sequence=px.colors.qualitative.Prism)
fig.show()
```

We find that

- 27 countries listed in the dataset operate under *Liberal democracy* regime
- 20 countries listed in the dataset operate under *Electoral democracy* regime
- 8 countries listed in the dataset operate under *Electoral autocracy* regime
- 7 countries listed in the dataset operate under *Closed autocracy* regime

Multivariate Analysis: Associations between categorical variables

Let's look at the associations between Region, Cluster, and Income groups first.

```
In [11]: agg_data = df[["Region", "Cluster", "Income group"]].groupby(["Region", "Cluster"])
# define figure element
fig = px.sunburst(
    agg_data,
    values='Count',
    path=["Region", "Cluster", "Income group"],
    title="Number of countries by region, cluster, and income group",
    color="Region",
    height=800,
    color_discrete_sequence=px.colors.qualitative.Pastel
)
# display the figure
fig.show()
```

We find that

- the biggest group of the countries in the dataset are European countries with High income level, fitting into Waking Up cluster
- countries with High income level, fitting into Waking Up cluster, are the most numerous group in Aasia-Pacific and Middle East regions, too
- the most 'crowdy' group in Americas is Waking Up, with Upper middle income level
- Traditional champions and Raising star countries are all with High level of incomes
- One of the Power players (USA) is with the High level of incomes, and another Power player (China) in Uppper middle income group

Now, let's look at the separation of the countries in AI Index dataset by the income group and political regimes.

```
In [12]: agg_data = df[["Region", "Income group", "Political regime"]].groupby(["Region",
# define figure element
fig = px.sunburst(
    agg_data,
    values='Count',
    path=["Region", "Income group", "Political regime"],
    title="Number of countries by region, income group, and political regime",
    color="Region",
    height=800,
    color_discrete_sequence=px.colors.qualitative.Pastel
)
# display the figure
fig.show()
```

We find that

- the biggest groups in Europe and Asia-Pacific regions represent the countries with High income and either Liberal or Electoral democracy
- the biggest group of countries in Americas contains the countries with High income level and Liberal democracy as a political regime
- the biggest group of countries in Middle East contains the countries with High income level and Closed autocracy as a political regime

Now, let's look at the associations between Region, Cluster, and Political regime.

```
In [13]: agg_data = df[["Region", "Cluster", "Political regime"]].groupby(["Region",
# define figure element
fig = px.sunburst(
    agg_data,
    values='Count',
    path=["Region", "Cluster", "Political regime"],
    title="Number of countries by region, cluster, and political regime",
    color="Region",
    height=800,
    color_discrete_sequence=px.colors.qualitative.Pastel
```

```
)  
# display the figure  
fig.show()
```

We find that

- Rasing stars and Traditional champions operate under Liberal or Electoral democracy, in every region
- One of the Power players (USA) operates under Liberal democracy, and another Power player (China) operates under Closed autocracy political regime

Deeper-Dive Insights

In the chapters below, we are going to look deeper at the business-level and public administration insights drawn from the data in AI dataset.

First of all, we are going to look at the relations between *Total score* and other variables in the dataset

Then we are going to outline the holistic insights on

- AI Implementation
- AI Research and Development
- AI Governance and Commercial contexts

Total Score Insights

```
In [14]: total_df = df[['Country', 'Region', 'Cluster', 'Income group', 'Political reg
```

Total Score: Geospacial View

```
In [15]: # set the size of the geo bubble  
def set_score_size(value):  
    '''  
    Takes the numeric value of a parameter to visualize on a map (Plotly Geo)  
    Returns a number to indicate the size of a bubble for a country which nu  
    was supplied as an input  
    '''  
    result = np.log(1+value/1200)
```

```

    if result < 0:
        result = 0.001
    return result

total_df['Size'] = total_df.apply(lambda x: set_score_size(x['Total score']))

fig = px.scatter_geo(
    total_df, locations="Country", locationmode='country names',
    color="Total score",
    size='Size', hover_name="Country",
    range_color= [0, 100],
    projection="natural earth",
    title='AI Total Score Across the Globe',
    color_continuous_scale="portland_r")

# Update visual layout
fig.update_layout(
    showlegend=True,
    width=800,
    height=400,
    autosize=False,
    margin=dict(t=40, b=0, l=5, r=5),
    template="plotly_white",
)

fig.show()

```

C:\Users\andani\AppData\Local\Temp\ipykernel_33208\2870983846.py:14: Setting WithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```

In [16]: total_df.sort_values('Total score',
                             ascending=False)[
                             [
                                 'Country',
                                 'Total score'
                             ]
                             ][:10].style.background_gradient(cmap='seismic')

```

Out[16]:

| | Country | Total score |
|---|--------------------------|-------------|
| 0 | United States of America | 100.000000 |
| 1 | China | 62.920000 |
| 2 | United Kingdom | 40.930000 |
| 3 | Canada | 40.190000 |
| 4 | Israel | 39.890000 |
| 5 | Singapore | 38.670000 |
| 6 | South Korea | 38.600000 |
| 7 | The Netherlands | 36.350000 |
| 8 | Germany | 36.040000 |
| 9 | France | 34.420000 |

We find that

- the Power players (USA, China) lead the ranks , although the gap between the total score of USA and China is significant
- some of the countries in Rasing stars cluster (Israel, Singapore, South Korea, The Netherlands) outperform some of the traditional champions (that is, Germany and France), in terms of the total score
- India does not fit into the top ten list of the countries by *Total score*, although it holds the second position in terms of AI Talent score

Total Score by Regions

```
In [17]: fig = px.violin(total_df, y="Total score", x="Region", box=True, points="all")
fig.show()
```

```
In [18]: # implement binning by Total Score categories
```

```
def set_score_level(x):
    res = "N/A"
    if x >= 80:
        res = "1 - Very high"
    elif x >= 60 and x < 80:
        res = "2 - High"
    elif x >= 40 and x < 60:
        res = "3 - Moderate"
    elif x >= 20 and x < 40:
        res = "4 - Low"
    else:
        res = "5 - Very low"
    return res
```

```
total_df["Score_level"] = total_df["Total score"].apply(set_score_level)

agg_data = total_df.groupby(['Region', 'Score_level']).size().reset_index(name='Country_count')

agg_data = agg_data.sort_values(by=["Score_level", "Country_count"])

fig = px.bar(
    agg_data,
    x='Region',
    y='Country_count',
    color='Score_level',
    title="Countries by Region and Total Score level",
    color_discrete_sequence=px.colors.qualitative.Prism,
    height=600)
fig.show()
```

C:\Users\andani\AppData\Local\Temp\ipykernel_33208\1587877859.py:17: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

We find that

- There is just one country with the Very high-level Total score (USA, in Americas region)
- There is just one country with the the High-level Total score (China, Asia-Pacific region)
- There are just two countries with the moderate-level Total score (Canada, UK); however, they are less than 1-2 units above the bunch of traditional champions and raising star countries with the Total score of 38+ or 39+ (the latter countries classified as Low-level in terms of their Total score)
- the distribution of Total score in Middle East and Africa shows these regions has more countries with lower Total score vs. Americas, Europe, and Asia-Pacific

Total Score by Clusters

```
In [19]: fig = px.violin(total_df, y="Total score", x="Cluster", box=True, points="all")
fig.show()
```

```
In [20]: agg_data = total_df.groupby(['Cluster', 'Score_level']).size().reset_index(name='Country_count')

agg_data = agg_data.sort_values(by=["Score_level", "Country_count"])
```

```
fig = px.bar(
    agg_data,
    x='Cluster',
    y='Country_count',
    color='Score_level',
    title="Countries by Region and Total Score level",
    color_discrete_sequence=px.colors.qualitative.Prism,
    height=600)
fig.show()
```

We find that

- the average Total score is higher in Power players and Traditional champions clusters
- some of the countries in Raising stars cluster are still scored higher then two of the traditional champions (Germany, France)

Total score by Income groups

```
In [21]: g = px.violin(total_df, y="Total score", x="Income group", box=True, points=
g.show()
```

```
In [22]: agg_data = total_df.groupby(['Income group', 'Score_level']).size().reset_in
agg_data = agg_data.sort_values(by=["Score_level", "Country_count"])

fig = px.bar(
    agg_data,
    x='Income group',
    y='Country_count',
    color='Score_level',
    title="Countries by Income group and Total score level",
    color_discrete_sequence=px.colors.qualitative.Prism,
    height=600)
fig.show()
```

We find that

- the overall trend is, the countries with High income level have higher average Total score vs. other income groups
- there are notable outliers in every income group
- USA, one of the Power players, stands out of the 'crowd' of the countries with the High income level
- China, another Power player, severely outperforms the rest of the countries in Upper middle income group
- India strongly outperforms the rest of the nations in Lower middle income group; actually, its score is higher then the score of all countries in Upper

middle income group (except for China), and 75% of the countries in High income group

Total score by Political regime

```
In [23]: g = px.violin(total_df, y="Total score", x="Political regime", box=True, poi
g.show()
```

```
In [24]: agg_data = total_df.groupby(['Political regime', 'Score_level']).size().rese

agg_data = agg_data.sort_values(by=["Score_level", "Country_count"])

fig = px.bar(
    agg_data,
    x='Political regime',
    y='Country_count',
    color='Score_level',
    title="Countries by Political regime and Total score level",
    color_discrete_sequence=px.colors.qualitative.Prism,
    height=600)
fig.show()
```

We find that

- There are much more countries with political regimes of Liberal democracy and Elecrotal democracy that ranked higher Total score in the list (the leading Power player, USA, is among them)
- At the same time, we can see some of the countries with autocratic regimes to be quite successful in AI, too (China, another Power player, is an example of that)

Holistic View on AI Implementation

Implementation metrics represent the application of artificial intelligence by professionals in various sectors, such as businesses, governments, and communities. The implementation factors group may be divided into three smaller groups

- Talent,
- Infrastructure, and
- Operating Environment.

```
In [25]: fig = px.scatter_3d(df, x='Region', y='Talent', z='Infrastructure', color='C
        title='Talent-Infrastructure-Operating Environment Inter-
# Update visual layout
fig.update_layout(
    showlegend=True,
    width=800,
    height=400,
    autosize=False,
    margin=dict(t=35, b=35, l=5, r=5),
    template="plotly_white",
)
fig.show()
```

```
In [26]: fig = px.scatter_3d(df, x='Cluster', y='Talent', z='Infrastructure', color='
        title='Talent-Infrastructure-Operating Environment Inter-
# Update visual layout
fig.update_layout(
    showlegend=True,
    width=800,
    height=400,
    autosize=False,
    margin=dict(t=35, b=35, l=5, r=5),
    template="plotly_white",
)
fig.show()
```

```
In [27]: fig = px.scatter_3d(df, x='Income group', y='Talent', z='Infrastructure', cc
        title='Talent-Infrastructure-Operating Environment Inter-
# Update visual layout
fig.update_layout(
    showlegend=True,
    width=800,
    height=400,
    autosize=False,
    margin=dict(t=35, b=35, l=5, r=5),
    template="plotly_white",
)
fig.show()
```

After the analysis of the visualizations above, we can confirm what [Global AI Race: Dominant Players and Aspiring Challengers] conveys on the AI Implementation indicators

✓ **Americans Are Most Talented in AI** The talent factors group emphasizes the presence of proficient experts who can deliver artificial intelligence solutions. They include the absolute and relative numbers of AI engineers, Data Scientists, and Machine engineers, commits related to AI on GitHub, and IT graduates, etc.

According to the provided dataset, the **United States** is the top country in the number of AI talents (an average score of all factors is 100). The following country with the lower score is India (a score of 45,3). The other three countries in the top-5 countries with skilled practitioners in AI are Great Britain, Singapore, and Israel, with almost equal scores (~40). The low scores of leaders and score median of 13.45 demonstrate an **unfavorable state** of the AI experts market for most countries.

✓ **China's Infrastructure Is Most Appropriate for AI**

Infrastructure factors focus on the dependability and scalability of the access infrastructure, ranging from electricity and internet to supercomputing capabilities. These factors include the ratio of the total population with access to electricity, the internet, the level of 5G implementation, etc.

As data say, China is better prepared for AI technologically (a score is 100). Hong Kong, Luxemburg, and the United States are just a little behind, with scores higher than 90. Ireland ends the top-5 rating with a score of 89.5. 80% of the analyzed countries have an Infrastructure indicators score higher than 50, while just one country has a total score of the Talent group indicators higher than 50. It illustrates the high global level of infrastructural capacity in comparison with the deficit of AI specialists.

✓ **Saudi Arabia's Society Is Most Friendly to AI**

The Operating Environment group of factors concentrates on the regulatory environment and the public perception of artificial intelligence. The group consists of such factors as the level of data privacy legislation, gender diversity of AI professionals, the share of people who trust AI, etc.

Generally, the world society is optimistic about AI (85% of the analyzed countries have an Operating Environment group score higher than 50); the five most benevolent countries are Saudi Arabia (a score is 100), Poland, Mexico, Slovenia, and Canada.

🚩 **Generic note:** Under weights for each indicator group, the Top 5 countries with the highest AI implementation level are the

United States, The Netherlands, Canada, China, and Great Britain. Only 35% of analyzed countries have an implementation score higher than 50.

Holistic View on AI Innovations

Innovation metrics reflect the progress made in technology and methodology, which signify the potential for artificial intelligence to evolve and improve. According to Tortoise Media, the innovation metrics are divided into two smaller groups

- Research
- Development

```
In [28]: fig = px.scatter_3d(df, x='Region', y='Research', z='Development', color='To
        title='R&D Inter-relations by Regions')
# Update visual layout
fig.update_layout(
    showlegend=True,
    width=800,
    height=400,
    autosize=False,
    margin=dict(t=35, b=35, l=5, r=5),
    template="plotly_white",
)
fig.show()
```

```
In [29]: fig = px.scatter_3d(df, x='Cluster', y='Research', z='Development', color='T
        title='R&D Inter-relations by Clusters')
# Update visual layout
fig.update_layout(
    showlegend=True,
    width=800,
    height=400,
    autosize=False,
    margin=dict(t=35, b=35, l=5, r=5),
    template="plotly_white",
)
fig.show()
```

```
In [30]: fig = px.scatter_3d(df, x='Income group', y='Research', z='Development', col
        title='R&D Inter-relations by Income groups')
# Update visual layout
fig.update_layout(
    showlegend=True,
    width=800,
    height=400,
```

```

    autosize=False,
    margin=dict(t=35, b=35, l=5, r=5),
    template="plotly_white",
)
fig.show()

```

```

In [31]: fig = px.scatter_3d(df, x='Political regime', y='Research', z='Development',
                             title='R&D Inter-relations by Political regimes')
# Update visual layout
fig.update_layout(
    showlegend=True,
    width=800,
    height=400,
    autosize=False,
    margin=dict(t=35, b=35, l=5, r=5),
    template="plotly_white",
)
fig.show()

```

The charts above confirms what [Global AI Race: Dominant Players and Aspiring Challengers]conveys on the AI Innovation indicators

✅ **US Specialists Are Generating More AI Ideas.** The Research indicator analyzes the degree of specialization among researchers and their research activities, specifically focusing on quantifying the number of publications and citations in reputable scholarly journals. This complex indicator accounts for total spending on research and development, the absolute and relative number of AI-related articles, researchers, and universities with AI-related courses.

The global state of the AI Research indicator is unpleasant — only two countries have scored higher than 50, one of which is the United States (a score is 100). It's not surprising since we have already found that the country has the most specialists related to AI. The second country is China (a score of 71,4); it has been investing heavily in AI and technology education in recent years to become a global leader in the AI sphere. The following three countries on the top-5 list are Switzerland, Singapore, and the United Kingdom.

✅ **AI Development in the USA Is at the Highest Level.** The group of Development indicators focuses on fundamental platforms and algorithms development upon which innovative artificial intelligence projects rely. It includes the absolute and relative number of software developers who are core creators of

open-source "AI packages," the number of commits to "open-source AI packages," the number of patents relating to "AI," etc. Globally, only three countries are on a high level of development in the AI field; they are the USA (a score is 100), China (a score is 80), and South Korea (a score is 77,3). The score of other countries is lower than 50. To complete the top-5 list, - the last two countries are Australia and Japan.

🚩 **Generic note:** The Top 5 countries with the highest level of innovation in AI are the United States, China, South Korea, Australia, and Switzerland. However, only the first two have an innovation score higher than 50. It testifies to the low level of AI development among the countries (in comparison with the leader - the USA) despite the shreds of evidence of AI worldwide exponential growth.

🌞 **Note:** Additionally, we can observe the interesting 'outlier' represented by **South Korea**. This country demonstrates quite a low value of AI Research while its AI Development index is within the top 5 list for the entire dataset. It can indicate the country to become a powerful hub for prototyping various AI-centric solutions, products, and hardware where the ideas/patents proposed by the specialists from other countries are being 'productized'/validated by the practical applications.

Holistic View on AI Investments

Investment metrics reflect financial and procedural commitments to artificial intelligence. Tortoise Media divides these metrics into two groups of indexes - Government Strategy and Commercial.

```
In [32]: fig = px.scatter_3d(df, x='Region', y='Government Strategy', z='Commercial',
                           title='AI Investments by Regions')
# Update visual layout
fig.update_layout(
    showlegend=True,
    width=800,
    height=400,
    autosize=False,
    margin=dict(t=35, b=35, l=5, r=5),
    template="plotly_white",
```

```
)  
fig.show()
```

```
In [33]: fig = px.scatter_3d(df, x='Cluster', y='Government Strategy', z='Commercial'  
                                title='AI Investments by Clusters')  
# Update visual layout  
fig.update_layout(  
    showlegend=True,  
    width=800,  
    height=400,  
    autosize=False,  
    margin=dict(t=35, b=35, l=5, r=5),  
    template="plotly_white",  
)  
fig.show()
```

```
In [34]: fig = px.scatter_3d(df, x='Income group', y='Government Strategy', z='Commer  
                                title='AI Investments by Income groups')  
# Update visual layout  
fig.update_layout(  
    showlegend=True,  
    width=800,  
    height=400,  
    autosize=False,  
    margin=dict(t=35, b=35, l=5, r=5),  
    template="plotly_white",  
)  
fig.show()
```

```
In [35]: fig = px.scatter_3d(df, x='Political regime', y='Government Strategy', z='Co  
                                title='AI Investments by Political regimes')  
# Update visual layout  
fig.update_layout(  
    showlegend=True,  
    width=800,  
    height=400,  
    autosize=False,  
    margin=dict(t=35, b=35, l=5, r=5),  
    template="plotly_white",  
)  
fig.show()
```

The charts above confirms what [Global AI Race: Dominant Players and Aspiring Challengers]conveys on the AI Investment indicators

✓ **The Canadian Government Is the Most Inclined Toward the AI Growth.** The Government Strategy indicator focuses on the extent of national commitment to artificial intelligence by examining both spending allocations and national-level plans. It factors in the amount of dedicated investment in AI by the government, the level of tax credit for research and

development, dedicated spending on AI, etc.

Almost 70% of the countries in the dataset are ready for AI evolution at the governmental level; their Government Strategy indicator score is higher than 50. The first five countries in the rate are Canada, China, Saudi Arabia, Spain, and France (it is 5 out of 6 countries with a score higher than 90).

✅ **The United States Is an Absolute Leader in the AI Commercialisation.** Commercial indicators evaluate the degree of startup engagement, investment, and business ventures that rely on artificial intelligence. These indicators include the absolute and relative numbers of AI companies and startups, their total and average funding, etc.

The United States is leading in the commercialization of AI and has no direct competition in this field. China is the closest country to the USA but is 66% behind. The three following countries are Israel, the United Kingdom, and Singapore; their commercial scores are lower than 30.

Although the United States Government Strategy in AI is not even in the top-5 of the strongest ones, it is scoring high on Investment indicators and is ranking first. The following four countries are China, Israel, the United Kingdom, and Canada; their Investment level is far behind the USA.

Talent Insights

```
In [36]: talent_df = df[['Country', 'Region', 'Cluster', 'Income group', 'Political re
```

Talent: Geospacial view

```
In [37]: # set the size of the geo bubble
def set_talent_size(value):
    """
    Takes the numeric value of a parameter to visualize on a map (Plotly Geo
    Returns a number to indicate the size of a bubble for a country which nu
    was supplied as an input
```



```

    ...
    result = np.log(1+value/1200)

    if result < 0:
        result = 0.001
    return result

talent_df['Size'] = talent_df.apply(lambda x: set_talent_size(x['Talent']),

fig = px.scatter_geo(
    talent_df, locations="Country", locationmode='country names',
    color="Talent",
    size='Size', hover_name="Country",
    range_color= [0, 100],
    projection="natural earth",
    title='AI Talent Across the Globe',
    color_continuous_scale="portland_r")

# Update visual layout
fig.update_layout(
    showlegend=True,
    width=800,
    height=400,
    autosize=False,
    margin=dict(t=40, b=0, l=5, r=5),
    template="plotly_white",
)

fig.show()

```

C:\Users\andani\AppData\Local\Temp\ipykernel_33208\2351959951.py:14: Setting WithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```

In [38]: talent_df.sort_values('Talent',
                                ascending=False)[
                                [
                                    'Country',
                                    'Talent'
                                ]
                                ][[:10]].style.background_gradient(cmap='seismic')

```

Out[38]:

| | Country | Talent |
|----|--------------------------|------------|
| 0 | United States of America | 100.000000 |
| 16 | India | 45.270000 |
| 2 | United Kingdom | 39.650000 |
| 5 | Singapore | 39.380000 |
| 4 | Israel | 35.760000 |
| 7 | The Netherlands | 33.830000 |
| 3 | Canada | 31.280000 |
| 11 | Ireland | 29.930000 |
| 9 | France | 28.320000 |
| 18 | Sweden | 28.210000 |

Talent By Regions

```
In [39]: fig = px.violin(talent_df, y="Talent", x="Region", box=True, points="all", t
fig.show()
```

```
In [40]: # implement binning by Talent categories
```

```
def set_talent_level(x):
    res = "N/A"
    if x >= 80:
        res = "1 - Very high"
    elif x >= 60 and x < 80:
        res = "2 - High"
    elif x >= 40 and x < 60:
        res = "3 - Moderate"
    elif x >= 20 and x < 40:
        res = "4 - Low"
    else:
        res = "5 - Very low"
    return res
```

```
talent_df["Talent_level"] = talent_df["Talent"].apply(set_talent_level)
```

```
agg_data = talent_df.groupby(['Region', 'Talent_level']).size().reset_index()
```

```
agg_data = agg_data.sort_values(by=["Talent_level", "Country_count"])
```

```
fig = px.bar(
    agg_data,
    x='Region',
    y='Country_count',
    color='Talent_level',
    title="Countries by Region and Talent level",
```

```
color_discrete_sequence=px.colors.qualitative.Prism,  
height=600)  
fig.show()
```

C:\Users\andani\AppData\Local\Temp\ipykernel_33208\1648788115.py:17: Setting
WithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

Talent by Clusters

```
In [41]: fig = px.violin(talent_df, y="Talent", x="Cluster", box=True, points="all",  
fig.show()
```

```
In [42]: agg_data = talent_df.groupby(['Cluster', 'Talent_level']).size().reset_index  
  
agg_data = agg_data.sort_values(by=["Talent_level", "Country_count"])  
  
fig = px.bar(  
    agg_data,  
    x='Cluster',  
    y='Country_count',  
    color='Talent_level',  
    title="Countries by Cluster and Talent level",  
    color_discrete_sequence=px.colors.qualitative.Prism,  
    height=600)  
fig.show()
```

Talent by Income groups

```
In [43]: g = px.violin(talent_df, y="Talent", x="Income group", box=True, points="all",  
g.show()
```

```
In [44]: agg_data = talent_df.groupby(['Income group', 'Talent_level']).size().reset_  
  
agg_data = agg_data.sort_values(by=["Talent_level", "Country_count"])  
  
fig = px.bar(  
    agg_data,  
    x='Income group',  
    y='Country_count',  
    color='Talent_level',  
    title="Countries by Income group and Talent level",
```

```
color_discrete_sequence=px.colors.qualitative.Prism,  
height=600)  
fig.show()
```

Talent by Political regimes

```
In [45]: fig = px.violin(talent_df, y="Talent", x="Political regime", box=True, points=True,  
fig.show()
```

```
In [46]: agg_data = talent_df.groupby(['Political regime', 'Talent_level']).size().reset_index(name='Country_count')  
  
agg_data = agg_data.sort_values(by=["Talent_level", "Country_count"])  
  
fig = px.bar(  
    agg_data,  
    x='Political regime',  
    y='Country_count',  
    color='Talent_level',  
    title="Countries by Political regime and Talent level",  
    color_discrete_sequence=px.colors.qualitative.Prism,  
    height=600)  
fig.show()
```

Infrastructure Insights

```
In [47]: infra_df = df[['Country', 'Region', 'Cluster', 'Income group', 'Political regime', 'Infrastructure level']]
```

Infrastructure by Regions

```
In [48]: fig = px.violin(infra_df,  
    y="Infrastructure",  
    x="Region",  
    box=True,  
    points="all",  
    title="Infrastructure by Regions",  
    hover_data=infra_df.columns)  
fig.show()
```

```
In [49]: # implement binning by Infrastructure categories  
  
def set_infrastructure_level(x):  
    res = "N/A"  
    if x >= 80:  
        res = "1 - Very high"
```

```

elif x >= 60 and x < 80:
    res = "2 - High"
elif x >= 40 and x < 60:
    res = "3 - Moderate"
elif x >= 20 and x < 40:
    res = "4 - Low"
else:
    res = "5 - Very low"
return res

infra_df["Infrastructure_level"] = infra_df["Infrastructure"].apply(set_infr
agg_data = infra_df.groupby(['Region', 'Infrastructure_level']).size().reset
agg_data = agg_data.sort_values(by=["Infrastructure_level", "Country_count"])

fig = px.bar(
    agg_data,
    x='Region',
    y='Country_count',
    color='Infrastructure_level',
    title="Countries by Region and Infrastructure level",
    color_discrete_sequence=px.colors.qualitative.Prism,
    height=600)
fig.show()

```

C:\Users\andani\AppData\Local\Temp\ipykernel_33208\4255209223.py:17: Setting WithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

Infrastructure by Clusters

```

In [50]: fig = px.violin(infra_df,
                        y="Infrastructure",
                        x="Cluster",
                        box=True,
                        points="all",
                        title="Infrastructure by Clusters",
                        hover_data=infra_df.columns)
fig.show()

```

```

In [51]: agg_data = infra_df.groupby(['Cluster', 'Infrastructure_level']).size().reset
agg_data = agg_data.sort_values(by=["Infrastructure_level", "Country_count"])
fig = px.bar(

```

```

agg_data,
x='Cluster',
y='Country_count',
color='Infrastructure_level',
title="Countries by Cluster and Infrastructure level",
color_discrete_sequence=px.colors.qualitative.Prism,
height=600)
fig.show()

```

Infrastructure by Income groups

```

In [52]: fig = px.violin(infra_df,
                        y="Infrastructure",
                        x="Income group",
                        box=True,
                        points="all",
                        title="Infrastructure by Income groups",
                        hover_data=infra_df.columns)

fig.show()

```

```

In [53]: agg_data = infra_df.groupby(['Income group', 'Infrastructure_level']).size()

agg_data = agg_data.sort_values(by=["Infrastructure_level", "Country_count"])

fig = px.bar(
    agg_data,
    x='Income group',
    y='Country_count',
    color='Infrastructure_level',
    title="Countries by Income group and Infrastructure level",
    color_discrete_sequence=px.colors.qualitative.Prism,
    height=600)
fig.show()

```

Infrastructure by Political Regimes

```

In [54]: fig = px.violin(infra_df,
                        y="Infrastructure",
                        x="Political regime",
                        box=True,
                        points="all",
                        title="Infrastructure by Political regimes",
                        hover_data=infra_df.columns)

fig.show()

```

```

In [55]: agg_data = infra_df.groupby(['Political regime', 'Infrastructure_level']).si

agg_data = agg_data.sort_values(by=["Infrastructure_level", "Country_count"])

```

```

fig = px.bar(
    agg_data,
    x='Political regime',
    y='Country_count',
    color='Infrastructure_level',
    title="Countries by Political regime and Infrastructure level",
    color_discrete_sequence=px.colors.qualitative.Prism,
    height=600)
fig.show()

```

Infrastructure: Geospacial view

```

In [56]: # set the size of the geo bubble
def set_infrastructure_size(value):
    """
    Takes the numeric value of a parameter to visualize on a map (Plotly Geo)
    Returns a number to indicate the size of a bubble for a country which number
    was supplied as an input
    """
    result = np.log(1+value/1200)

    if result < 0:
        result = 0.001
    return result

infra_df['Size'] = infra_df.apply(lambda x: set_infrastructure_size(x['Infrastructure']), axis=1)

fig = px.scatter_geo(
    infra_df, locations="Country", locationmode='country names',
    color="Infrastructure",
    size='Size', hover_name="Country",
    range_color= [0, 100],
    projection="natural earth",
    title='AI Infrastructure Across the Globe',
    color_continuous_scale="portland_r")

# Update visual layout
fig.update_layout(
    showlegend=True,
    width=800,
    height=400,
    autosize=False,
    margin=dict(t=40, b=0, l=5, r=5),
    template="plotly_white",
)

fig.show()

```

```
C:\Users\andani\AppData\Local\Temp\ipykernel_33208\257039703.py:14: SettingWithCopyWarning:
```

A value is trying to be set on a copy of a slice from a DataFrame.
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
In [57]: infra_df.sort_values('Infrastructure',
                             ascending=False)[
                             [
                                 'Country',
                                 'Infrastructure'
                             ]
                             ][[:10]].style.background_gradient(cmap='seismic')
```

```
Out[57]:
```

| | Country | Infrastructure |
|----|--------------------------|----------------|
| 1 | China | 100.000000 |
| 19 | Hong Kong | 96.110000 |
| 14 | Luxembourg | 94.880000 |
| 0 | United States of America | 94.020000 |
| 11 | Ireland | 89.500000 |
| 6 | South Korea | 85.230000 |
| 15 | Japan | 84.580000 |
| 5 | Singapore | 84.300000 |
| 7 | The Netherlands | 81.990000 |
| 33 | United Arab Emirates | 79.160000 |

Operating Environment Insights

```
In [58]: ops_df = df[['Country', 'Region', 'Cluster', 'Income group', 'Political region', 'Operating Environment']]
```

Operating Environment by Regions

```
In [59]: fig = px.violin(ops_df,
                          y="Operating Environment",
                          x="Region",
                          box=True,
                          points="all",
```



```

        title="Operating Environment by Regions",
        hover_data=ops_df.columns)
fig.show()

```

In [60]: *# implement binning by Operating Environment categories*

```

def set_ops_level(x):
    res = "N/A"
    if x >= 80:
        res = "1 - Very high"
    elif x >= 60 and x < 80:
        res = "2 - High"
    elif x >= 40 and x < 60:
        res = "3 - Moderate"
    elif x >= 20 and x < 40:
        res = "4 - Low"
    else:
        res = "5 - Very low"
    return res

ops_df["Ops_Environment_level"] = ops_df["Operating Environment"].apply(set_
agg_data = ops_df.groupby(['Region', 'Ops_Environment_level']).size().reset_
agg_data = agg_data.sort_values(by=["Ops_Environment_level", "Country_count"]

fig = px.bar(
    agg_data,
    x='Region',
    y='Country_count',
    color='Ops_Environment_level',
    title="Countries by Region and Operating Environment level",
    color_discrete_sequence=px.colors.qualitative.Prism,
    height=600)
fig.show()

```

C:\Users\andani\AppData\Local\Temp\ipykernel_33208\1132965378.py:17: Setting WithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

Operating Environment by Clusters

```

In [61]: fig = px.violin(ops_df,
                        y="Operating Environment",
                        x="Cluster",
                        box=True,

```

```

        points="all",
        title="Operating Environment by Clusters",
        hover_data=ops_df.columns)
fig.show()

```

```

In [62]: agg_data = ops_df.groupby(['Cluster', 'Ops_Environment_level']).size().reset
agg_data = agg_data.sort_values(by=["Ops_Environment_level", "Country_count"])

fig = px.bar(
    agg_data,
    x='Cluster',
    y='Country_count',
    color='Ops_Environment_level',
    title="Countries by Cluster and Operating Environment level",
    color_discrete_sequence=px.colors.qualitative.Prism,
    height=600)
fig.show()

```

Operating Environment by Income Groups

```

In [63]: fig = px.violin(ops_df,
                        y="Operating Environment",
                        x="Income group",
                        box=True,
                        points="all",
                        title="Operating Environment by Income Groups",
                        hover_data=ops_df.columns)
fig.show()

```

```

In [64]: agg_data = ops_df.groupby(['Income group', 'Ops_Environment_level']).size().
agg_data = agg_data.sort_values(by=["Ops_Environment_level", "Country_count"])

fig = px.bar(
    agg_data,
    x='Income group',
    y='Country_count',
    color='Ops_Environment_level',
    title="Countries by Income Group and Operating Environment level",
    color_discrete_sequence=px.colors.qualitative.Prism,
    height=600)
fig.show()

```

Operating Environment by Political Regimes

```

In [65]: fig = px.violin(ops_df,
                        y="Operating Environment",
                        x="Political regime",

```

```

        box=True,
        points="all",
        title="Operating Environment by Political Regimes",
        hover_data=ops_df.columns)
fig.show()

```

```

In [66]: agg_data = ops_df.groupby(['Political regime', 'Ops_Environment_level']).size
agg_data = agg_data.sort_values(by=["Ops_Environment_level", "Country_count"])

fig = px.bar(
    agg_data,
    x='Political regime',
    y='Country_count',
    color='Ops_Environment_level',
    title="Countries by Political Regime and Operating Environment level",
    color_discrete_sequence=px.colors.qualitative.Prism,
    height=600)
fig.show()

```

Operating Environment: Geospacial view

```

In [67]: # set the size of the geo bubble
def set_op_env_size(value):
    """
    Takes the numeric value of a parameter to visualize on a map (Plotly Geo)
    Returns a number to indicate the size of a bubble for a country which number
    was supplied as an input
    """
    result = np.log(1+value/1200)

    if result < 0:
        result = 0.001
    return result

ops_df['Size'] = ops_df.apply(lambda x: set_op_env_size(x['Operating Environment']),
                             axis=1)

fig = px.scatter_geo(
    ops_df, locations="Country", locationmode='country names',
    color="Operating Environment",
    size='Size', hover_name="Country",
    range_color= [0, 100],
    projection="natural earth",
    title='AI Operating Environment Across the Globe',
    color_continuous_scale="portland_r")

# Update visual layout
fig.update_layout(
    showlegend=True,
    width=800,
    height=400,
    autosize=False,

```

```
margin=dict(t=40, b=0, l=5, r=5),
template="plotly_white",
)

fig.show()
```

C:\Users\andani\AppData\Local\Temp\ipykernel_33208\1725242863.py:14: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
In [68]: ops_df.sort_values('Operating Environment',
                           ascending=False)[
                           [
                               'Country',
                               'Operating Environment'
                           ]
                           ][[:10]].style.background_gradient(cmap='seismic')
```

Out[68]:

| | Country | Operating Environment |
|----|-----------------|-----------------------|
| 25 | Saudi Arabia | 100.000000 |
| 27 | Poland | 99.560000 |
| 43 | Mexico | 97.030000 |
| 28 | Slovenia | 94.550000 |
| 3 | Canada | 93.940000 |
| 1 | China | 91.570000 |
| 29 | New Zealand | 90.350000 |
| 40 | Slovakia | 88.710000 |
| 22 | Estonia | 88.670000 |
| 7 | The Netherlands | 88.050000 |

References

- 'T-Minus AI': A look at the intersection of geopolitics and autonomy (Aug 17, 2020) - <https://www.c4isrnet.com/opinion/2020/08/16/t-minus-ai-a-look-at-the-intersection-of-geopolitics-and-autonomy/>
- Artificial Intelligence and Great Power Competition (Mar 28, 2023) - <https://www.cfr.org/podcasts/artificial-intelligence-and-great-power->

competition-paul-scharre

- AI is supposedly the new nuclear weapons — but how similar are they, really? (Jun 29, 2023) - <https://www.vox.com/future-perfect/2023/6/29/23762219/ai-artificial-intelligence-new-nuclear-weapons-future>
- U.S.-China tech battle entering its 'primetime' — and generative A.I. could be the next frontier (Jun 22, 2023) - <https://www.cnbc.com/2023/06/23/us-china-tech-war-why-generative-ai-could-be-the-next-battleground.html>
- MACHINE POLITICS: EUROPE AND THE AI REVOLUTION (Jul. 1, 2019) - <https://www.jstor.org/stable/resrep21907>
- Military Uses of AI: A Risk to International Stability? (Jan 1, 2021) - <https://www.jstor.org/stable/resrep28649.5>
- Global AI Race: Dominant Players and Aspiring Challengers (Jun 26, 2023) - <https://intersog.com/blog/ai-dominant-players-and-aspiring-challengers/>
- State of AI Report - <https://stateof.ai>

This notebook was converted with convert.ploomber.io