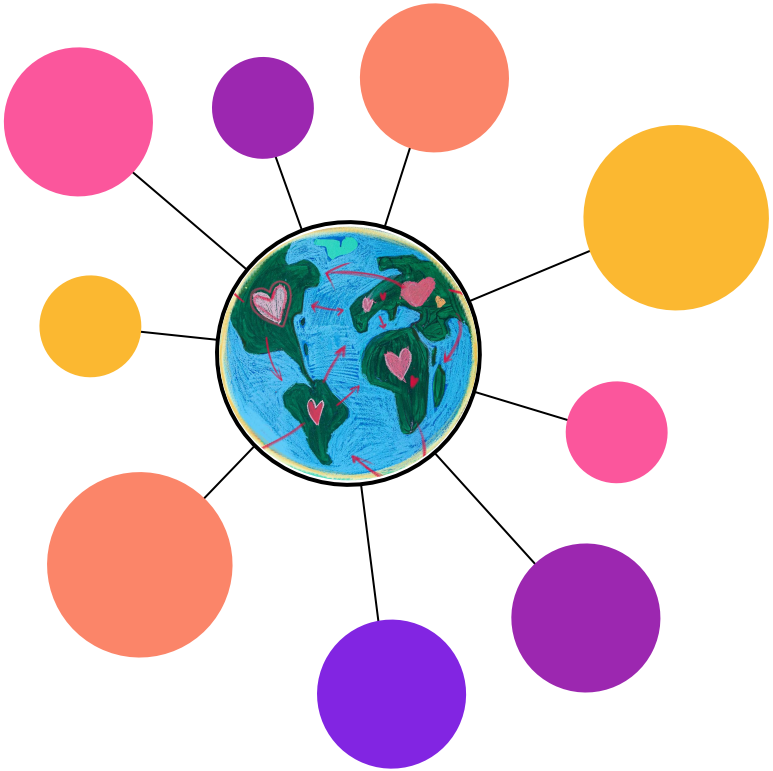


World Happiness Prediction

USING MACHINE LEARNING

Tudor Cardas, Yassine Turki, and Paula González Álvarez

Table of Contents

Introduction

Background

- We wanted to see how one could measure happiness because it seemed very complicated to measure
- Every year since 2015, the World Happiness Report is published where around 150 counties are ranked based on happiness
- European countries are consistently on the top of the rankings

Objective

- Find which variables amongst the ones they use are the most relevant
- Try to add new variables that we consider could be important and see what happens when we include them in the model
- Find biases which lead to biased happiness rankings
- Find the correlation between the variables
- Learn how to handle missing data when large chunks of data are missing

Original Ranking

How Real Happiness Report Variables are Measured

- GDP per capita is in terms of Purchasing Power Parity (PPP)
- Corruption: **“Is corruption widespread throughout the government or not?” and “Is corruption widespread within businesses or not?”**
- Social support: **“If you were in trouble, do you have relatives or friends you can count on to help you whenever you need them, or not?”**
- Freedom: **“Are you satisfied or dissatisfied with your freedom to choose what you do with your life?”**
- Generosity: **“Have you donated money to a charity in the past month?”**

World Happiness Ranking Data

Here is the dataset we used to run the regression with the original variables:

	Country	Happiness Score	Economy (GDP per Capita)	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity
0	Switzerland	7.587	1.39651	0.94143	0.66557	0.41978	0.29678
1	Iceland	7.561	1.30232	0.94784	0.62877	0.14145	0.43630
2	Denmark	7.527	1.32548	0.87464	0.64938	0.48357	0.34139
3	Norway	7.522	1.45900	0.88521	0.66973	0.36503	0.34699
4	Canada	7.427	1.32629	0.90563	0.63297	0.32957	0.45811
...	...	...	...	...	...	...	...
153	Rwanda	3.465	0.22208	0.42864	0.59201	0.55191	0.22628
154	Benin	3.340	0.28665	0.31910	0.48450	0.08010	0.18260
155	Syria	3.006	0.66320	0.72193	0.15684	0.18906	0.47179
156	Burundi	2.905	0.01530	0.22396	0.11850	0.10062	0.19727
157	Togo	2.839	0.20868	0.28443	0.36453	0.10731	0.16681

Variable Importance

Linear Regression:

1. Freedom: 1.829
2. Economy (GDP per Capita): 1.297
3. Health (Life Expectancy): 1.286
4. Generosity: 0.566
5. Trust (Government Corruption): 0.505

Ridge Regression:

1. Freedom: 1.747
2. Economy (GDP per Capita): 1.308
3. Health (Life Expectancy): 1.267
4. Generosity: 0.550
5. Trust (Government Corruption): 0.512

Lasso Regression:

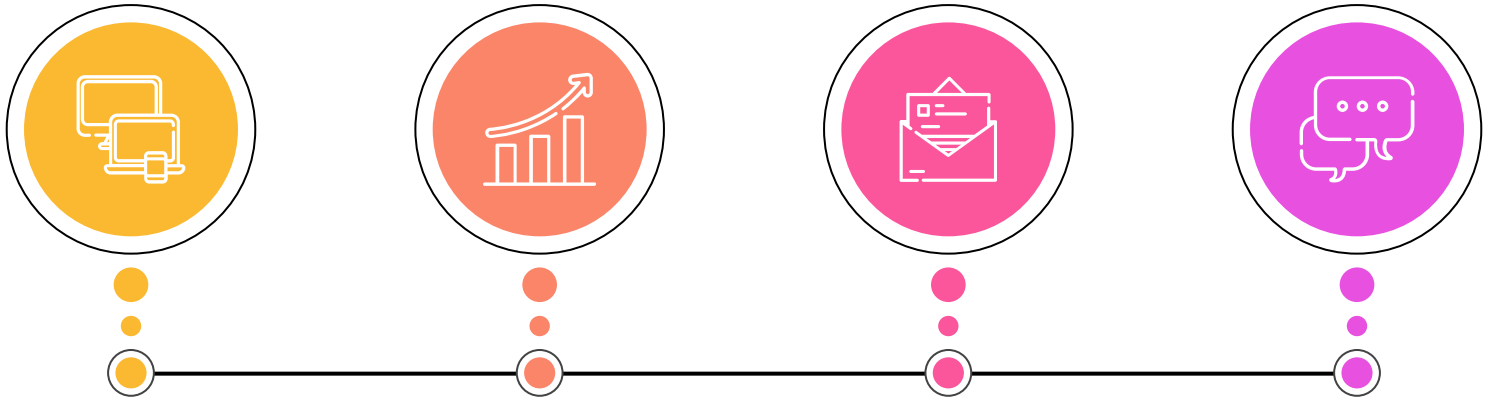
1. Freedom: 1.813
2. Economy (GDP per Capita): 1.306
3. Health (Life Expectancy): 1.270
4. Generosity: 0.517
5. Trust (Government Corruption): 0.442

ElasticNet Regression:

1. Economy (GDP per Capita): 1.359
2. Freedom: 1.312
3. Health (Life Expectancy): 1.138
4. Trust (Government Corruption): 0.447
5. Generosity: 0.409

Ranking with Extensive Data

Data Acquisition and Data Processing



Data acquisition

Researched online and chose to use World Development Indicators dataset from World Bank

Data processing

Went through all the 1478 indicators they have and agreed on the most relevant ones

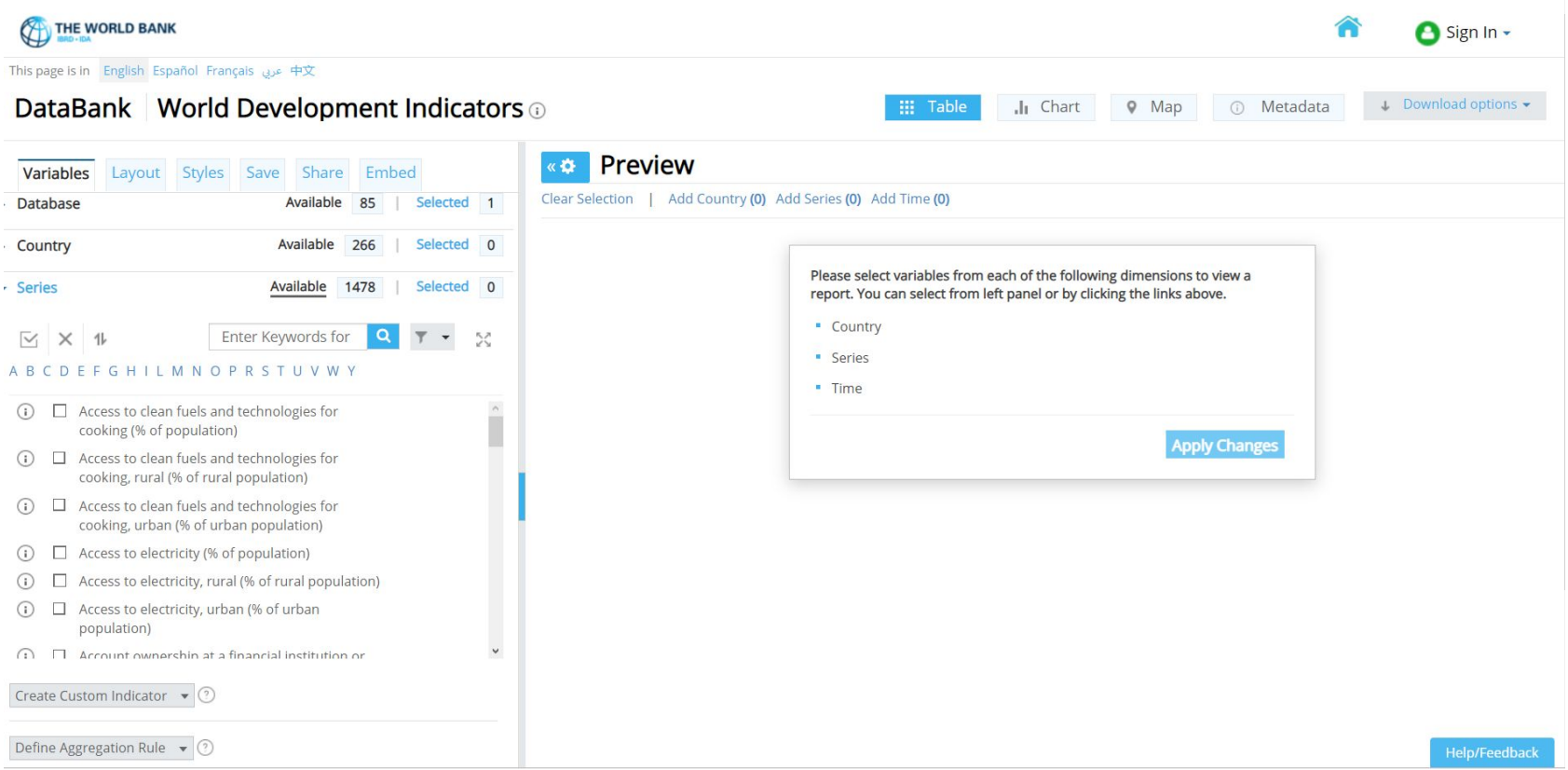
Data processing

Went through the list of countries and selected the ones which appeared in the happiness rankings

Data acquisition

Downloaded all of the data into an excel that we formatted and then imported into Jupyter

Downloading and Preparing the Data



Downloading and Preparing the Data

Row Labels	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011
Access to clean fuels and technologies for cooking (% of population)	8391.46	8477	8562.89	8648.54	8730.92	8812.15	8892.73	8970.05	9047.95	9121.17	9194.54	9265
Compulsory education, duration (years)	1076	1096	1118	1141	1206	1219	1230	1232	1248	1269	1298	1313
Current health expenditure (% of GDP)	775.7772618	795.291703	824.9765407	842.6929358	853.9516452	853.9389486	852.4551752	850.5863436	863.7433138	923.4968421	916.5130489	918.6266629
Death rate, crude (per 1,000 people)	1349.057	1335.983	1328.66	1318.279	1292.19	1283.824	1264.183	1252.234	1236.511	1214.525	1207.258	1181.892
Electric power consumption (kWh per capita)	453674.2934	460938.7455	469292.5637	477444.5637	485762.0341	500521.7115	512113.0388	521021.9613	531074.8513	513000.5607	533870.2138	532231.6162
Employment to population ratio, 15+, total (%) (national estimate)	4392.189985	4548.469997	4686.250015	4889.530022	4886.170002	5260.539982	5286.430019	5362.849989	5194.48	5450.640005	5986.029984	5702.960009
Fertility rate, total (births per woman)	449.339	440.9225	434.7375	430.0905	426.572	422.19	419.934	417.876	417.128	413.534	409.08	403.9685
Fixed telephone subscriptions (per 100 people)	2758.315553	2781.5563	2778.615961	2775.590076	2825.438745	2829.245522	2821.185045	2824.619053	2828.827504	2844.215312	2799.832096	2764.415736
Forest area (% of land area)	4145.151445	4139.554591	4134.26927	4128.144224	4120.382394	4114.450642	4194.127378	4190.581303	4186.649154	4182.582362	4178.458698	4172.833844
GDP per capita growth (annual %)	450.0760186	275.1878815	297.3080487	401.6439829	668.6356015	549.786876	653.5178752	619.8717178	358.5388679	-212.6198273	481.4475871	359.636393
Human capital index (HCI) (scale 0-1)	0	0	0	0	0	0	0	0	0	0	56.65930489	0
Individuals using the Internet (% of population)	1195.145966	1480.171858	1887.038646	2194.063627	2566.578801	2911.667805	3263.370635	3680.070101	4067.386731	4453.06	4882.7	5272.01201
Inflation, consumer prices (annual %)	1119.55522	955.3010065	774.833412	770.0411377	719.4724902	831.2847012	809.0834606	784.6730982	1428.67106	600.2903143	634.2836872	947.5893355
Intentional homicides (per 100,000 people)	743.6173605	805.8057701	801.7782137	711.6088084	741.1102401	702.1997487	682.563288	763.4781027	853.4546488	832.8529523	873.7305073	836.1842019
Labor force participation rate, total (% of total population ages 15+)	5539.530016	5411.240005	5339.009998	5471.979988	5608.609997	6096.699995	5968.299995	6074.330009	6082.79002	6201.890007	6600.479984	6229.139988
Labor force, total	2651848576	2687635139	2724305406	2762814932	2802723772	2843998378	2886189949	2928520572	2964793371	2999145863	3029623147	3062147013
Life expectancy at birth, total (years)	9650.838366	9693.989	9736.170976	9784.012585	9835.796341	9885.698244	9938.80878	9985.922854	10035.67795	10099.02859	10142.19934	10216.2891
Lifetime risk of maternal death (%)	188.3818683	180.9494166	174.5084694	167.1479962	159.7913777	152.0068808	145.4482749	139.1534053	133.7569026	128.6024503	122.8548821	117.8890068
Literacy rate, adult total (% of people ages 15 and above)	2900.157475	2011.31837	1162.138542	695.0452805	1257.038183	1678.537931	1907.29418	2445.815147	2119.457058	2562.73077	3382.673523	4503.014488
Mobile cellular subscriptions (per 100 people)	2502.101911	3254.856035	3876.656295	4508.148076	5479.303303	6629.050878	8035.15255	9651.339562	11032.497	12120.84039	13197.92093	14219.69598
Mortality rate, infant (per 1,000 live births)	5390.5	5177.3	4968.2	4768.2	4585.7	4393.5	4219.4	4066.2	3936.1	3785.8	3664.1	3522.2
Mortality rate, under-5 (per 1,000 live births)	7971.6	7634.4	7294.6	6967.7	6680.8	6361.9	6076.2	5810.6	5591	5322.6	5131	4875.2
People practicing open defecation (% of population)	1875.347251	1823.13449	1768.069402	1711.789206	1655.973214	1607.792129	1554.38173	1499.079984	1444.796038	1390.089961	1335.79632	1282.077732
People using safely managed drinking water services (% of populatic	6193.466415	6220.985186	6254.511942	6289.153247	6332.055605	6562.566111	6693.782193	6741.732156	6708.237902	6757.144325	6807.569283	6857.792529
People using safely managed sanitation services (% of population)	4863.570435	4949.475775	5059.742362	5100.816813	5151.578134	5225.854	5310.493607	5371.321178	5432.819726	5499.300035	5569.197836	5640.941306
Population ages 0-14 (% of total population)	4543.718628	4486.752619	4430.585294	4374.303958	4317.941102	4262.58359	4209.344288	4161.234451	4118.824208	4081.343709	4050.152491	4024.426098
Population ages 15-64 (% of total population)	8794.80451	8837.605983	8878.942771	8920.881148	8962.748431	9003.972996	9044.745609	9082.712928	9116.285314	9143.952062	9164.218914	9174.784119
Population ages 65 and above (% of total population)	1061.476861	1075.641398	1090.471935	1104.814894	1119.310467	1133.443415	1145.910103	1156.052622	1164.890479	1174.704229	1185.628595	1200.789783
Prevalence of undernourishment (% of population)	0	1727.8	1683.7	1641.6	1594.3	1534.4	1467	1410.1	1361.8	1284.5	1216	1152.4
Ratio of female to male labor force participation rate (%) (national es	6072.9659	5956.216524	5940.334362	6236.462268	6468.619909	6703.47169	6663.107624	6506.463607	6454.614162	6775.44725	7323.958055	6961.875288
Refugee population by country or territory of origin	8291187	8107290	7327319	6592161	6528233	5912951	6962188	8699456	7817992	7804170	7833276	7154660
School enrollment, secondary (% gross)	7558.451849	7462.067832	7888.224602	7936.990756	8271.034701	8538.578134	8318.79855	8458.711361	8209.171816	8174.549646	8583.905295	8700.672379
Suicide mortality rate (per 100,000 population)	1708.1	1677.9	1669.1	1645	1625.8	1630.9	1600.3	1577.8	1593.5	1600.6	1562	1537.5
Survival to age 65, female (% of cohort)	10476.17863	10522.09121	10571.32203	10630.9571	10685.84678	10766.66328	10847.40871	10911.9581	10974.82484	11064.94572	11123.76978	11239.07286
Survival to age 65, male (% of cohort)	9021.033037	9073.555667	9129.585714	9192.455982	9265.367649	9335.844751	9411.097248	9483.750791	9567.047832	9662.882089	9739.161062	9842.604189
Total greenhouse gas emissions (kt of CO2 equivalent)	31725940.18	32117659.75	32500389.96	33707459.76	35078910.08	36193389.52	37222710.01	38447790.05	38741179.84	38360980.03	40322889.56	41515810.55
Unemployment, total (% of total labor force) (national estimate)	824.8599961	818.4100015	801.4100008	828.3899987	814.6399966	867.7100009	754.6000024	783.1899984	717.9699992	890.3299985	927.4500019	823.2799973
Wage and salaried workers, total (% of total employment) (modeled	7955.279975	7959.530004	7974.510015	7997.529956	8046.259969	8092.970009	8158.060021	8229.970025	8279.039999	8290.75999	8342.239991	8402.379997
Grand Total	2692451316	2728454114	2764735729	2803726222	2844953633	2886745495	2931028799	2976333336	3012029424	3045971118	3078465274	3111503377

Adding Columns

Next, we downloaded the World Happiness Report scores as well as the GDP per capita for each year and added them into our dataset. This gave us issues because at the beginning it wasn't aligned correctly and it was giving us bad results.

```
data["GDP Per Capita"]=np.zeros(len(data))
data["Score"]=np.zeros(len(data))
# Adding the Score column for our data:
end_of_countries=data.index[data['Countries'] == "2016"][0]
countries_in_dataframe=data.iloc[:end_of_countries,:1]
listof_countries_in_data=countries_in_dataframe["Countries"].to_list()
for i, y in enumerate(years):
    for j in range(y.shape[0]):
        if y.iloc[j][1] in listof_countries_in_data:
            data.at[data.loc[data["Countries"] == y.iloc[j][1]].index[i], 'GDP Per Capita'] =  y.iloc[j][2]
            data.at[data.loc[data["Countries"] == y.iloc[j][1]].index[i], 'Score'] =  y.iloc[j][0]
```

Missing Values

Filtering Factors

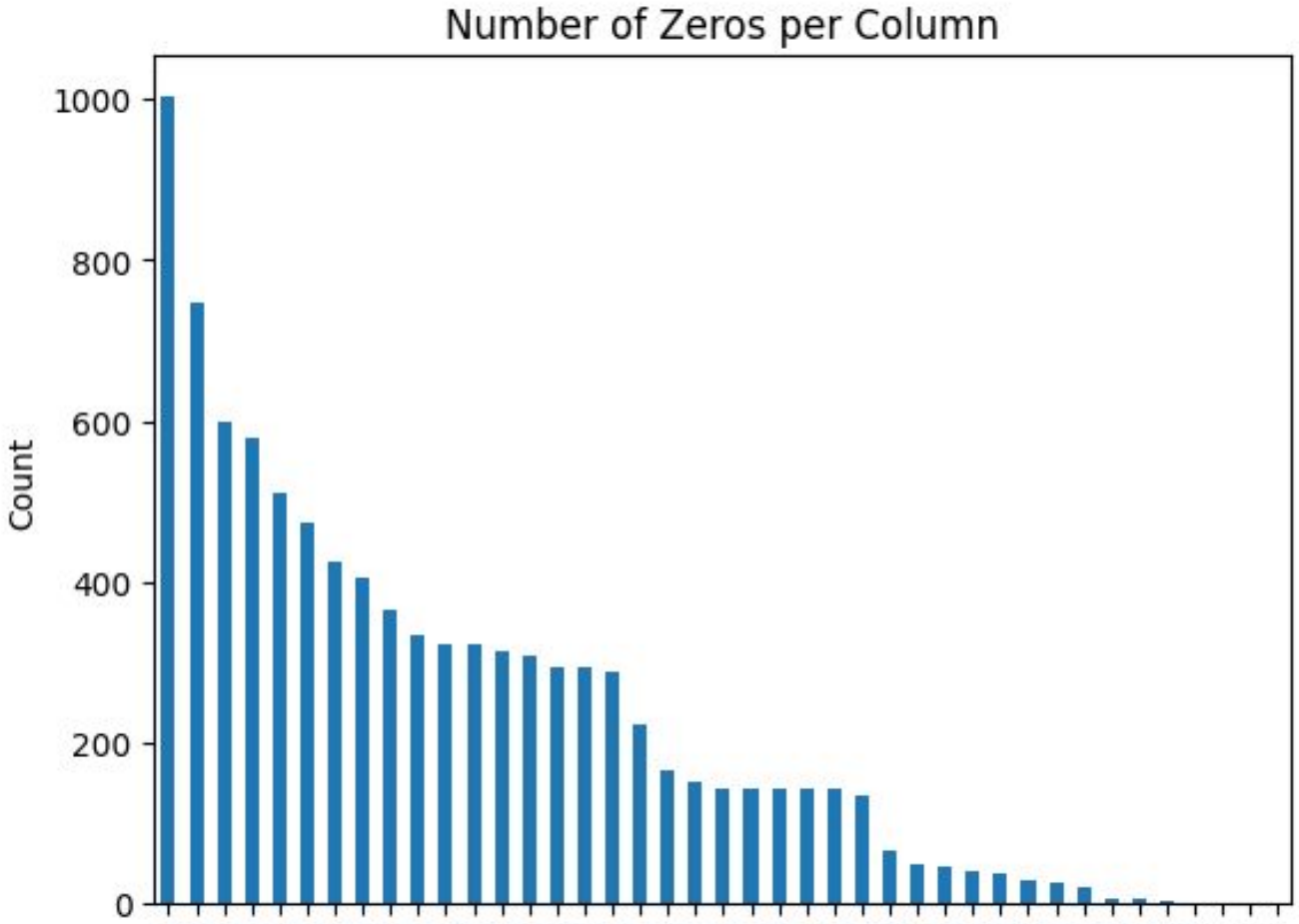


Table with missing values

	2015	2016	2017	2018	2019	2020	2021	sum
Electric power consumption (kWh per capita)	144	144	144	144	144	145	138	1003
Literacy rate, adult total (% of people ages 15 and above)	109	115	116	87	118	120	81	746
Human capital index (HCI) (scale 0-1)	145	143	8	12	140	12	138	598
Lifetime risk of maternal death (%)	1	1	4	145	145	145	138	579
People practicing open defecation (% of population)	53	54	60	64	69	73	138	511
Intentional homicides (per 100,000 people)	46	49	51	59	57	74	138	474
People using safely managed drinking water services (% of population)	47	47	47	48	47	52	138	426
People using safely managed sanitation services (% of population)	43	42	43	44	45	50	138	405
School enrollment, secondary (% gross)	38	39	41	44	45	56	102	365
Ratio of female to male labor force participation rate (%) (national estimate)	48	44	40	47	43	51	60	333
Employment to population ratio, 15+, total (%) (national estimate)	46	43	36	44	41	52	60	322
Labor force participation rate, total (% of total population ages 15+) (national estimate)	45	41	37	45	42	51	60	321
Unemployment, total (% of total labor force) (national estimate)	45	40	36	43	40	50	60	314
Current health expenditure (% of GDP)	3	4	4	5	9	145	138	308

Handling missing data

Next, we dropped all of the factors that we had initially selected that had a very large number of data missing:

```
data=data.drop(columns=["Electric power consumption (kWh per capita)",
                        "Literacy rate, adult total (% of people ages 15 and above)",
                        "Human capital index (HCI) (scale 0-1)",
                        "Lifetime risk of maternal death (%)",
                        "People practicing open defecation (% of population)",
                        "Intentional homicides (per 100,000 people)",
                        "People using safely managed drinking water services (% of population)",
                        "People using safely managed sanitation services (% of population)",
                        ]) # Dropping Features that have a lot of missing data
data=data[:-144] # Dropping observations for year 2021
data = data[data['Score'] != 0] # Lastly, dropping all observations that lack a Happiness Score
```

Imputing missing data

The code belows how we imputed our missing data using a K-nearest neighbors method:

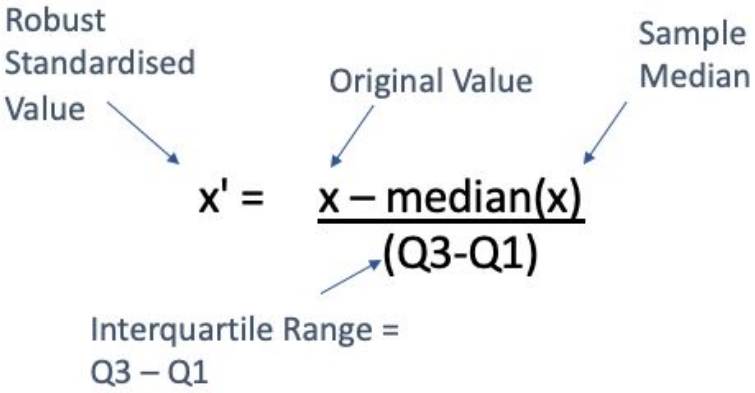
```
# The method works only for numerical variables so we discard the "Countries" column
data_without_countries = data.drop('Countries', axis=1)

# We impute missing values with a K-nearest neighbors type method
imputer = KNNImputer(missing_values = 0, n_neighbors = 5, weights = 'distance')

imputed_data = imputer.fit_transform(data_without_countries)

# We merge back the "Countries" column with the newly formed data to get the new dataframe without any missing values
imputed_data_with_countries = np.column_stack((data['Countries'].to_numpy(),imputed_data))
full_data = pd.DataFrame(imputed_data_with_countries, columns=data.columns)


cols_to_convert = full_data.columns.difference(['Countries'])  # Exclude 'countries' column
full_data[cols_to_convert] = full_data[cols_to_convert].apply(pd.to_numeric, errors='coerce')
```

Data overview: Need for Normalization

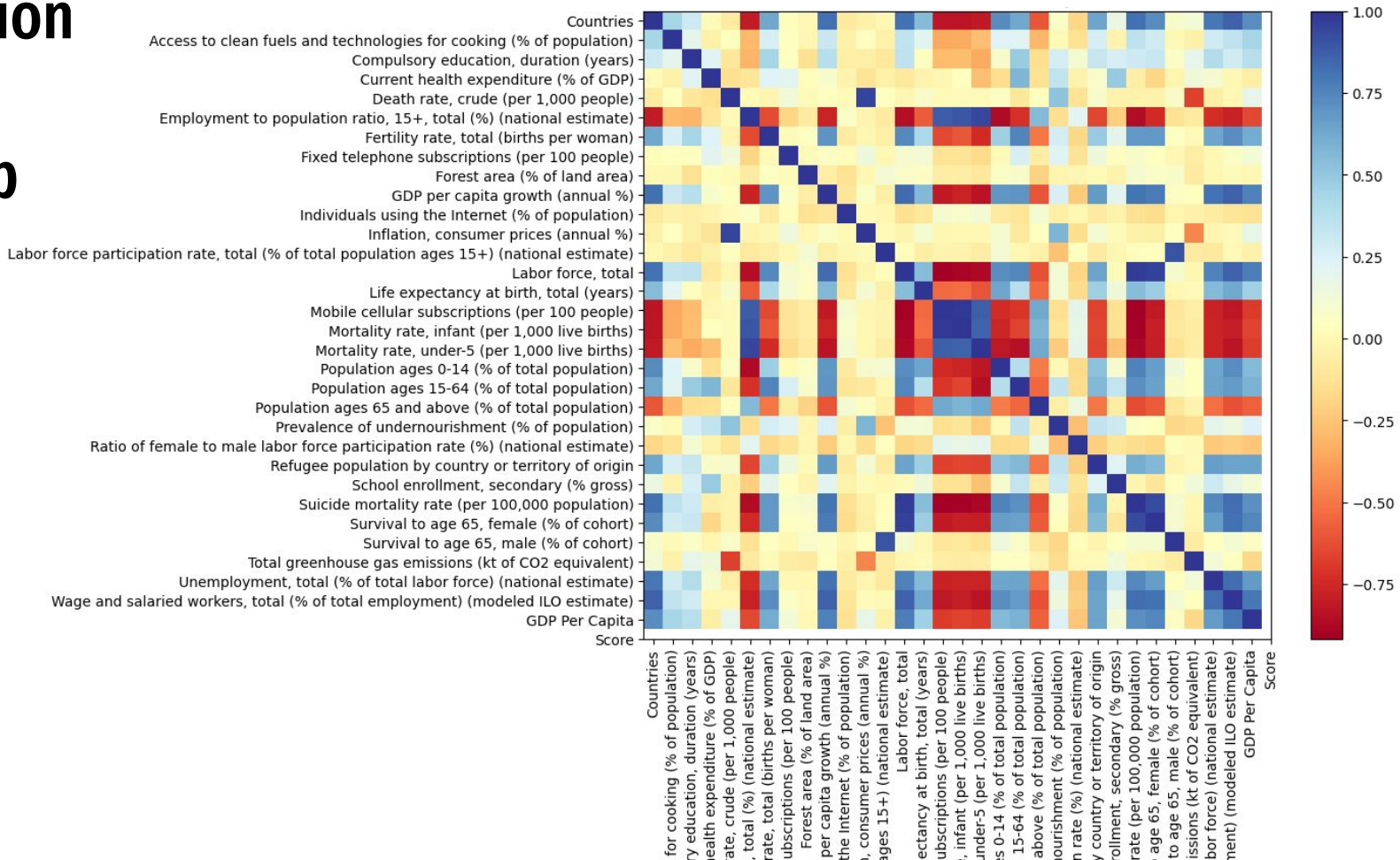
	Access to clean fuels and technologies for cooking (% of population)	Compulsory education, duration (years)	Current health expenditure (% of GDP)	Death rate, crude (per 1,000 people)	Employment to population ratio, 15+, total (%) (national estimate)	Fertility rate, total (births per woman)	Fixed telephone subscriptions (per 100 people)	Forest area (% of land area)	GDP per capita growth (annual %)	Individuals using the Internet (% of population)	...	Refugee population by country or territory of origin
count	823.000000	823.000000	823.000000	823.000000	823.000000	823.000000	823.000000	823.000000	823.000000	823.000000	...	8.230000e+02
mean	68.065213	9.696233	5.470224	8.015463	41.013998	2.578566	16.507509	28.605680	0.696699	54.642178	...	5.219201e+04
std	37.818179	3.013364	3.522260	2.882243	26.825728	1.298570	15.665179	20.349544	4.676493	30.262231	...	2.450979e+05
min	0.000000	0.000000	0.000000	1.014000	0.000000	0.868000	0.000000	0.000000	-34.776559	0.000000	...	0.000000e+00
25%	31.325000	9.000000	3.477337	6.115000	0.000000	1.594000	1.969924	11.162336	-0.674882	24.900000	...	1.415000e+02
50%	87.800000	10.000000	5.654695	7.352000	53.439999	2.097000	13.137804	28.791971	1.520007	61.762212	...	1.469000e+03
75%	100.000000	12.000000	7.851398	9.600000	60.019999	3.145000	25.641853	42.734823	3.332766	80.865833	...	1.367700e+04
max	100.000000	17.000000	20.413414	18.000000	84.839996	7.211000	61.007900	91.551209	30.496301	100.000000	...	2.727556e+06

Data Normalization: Robust Scaler

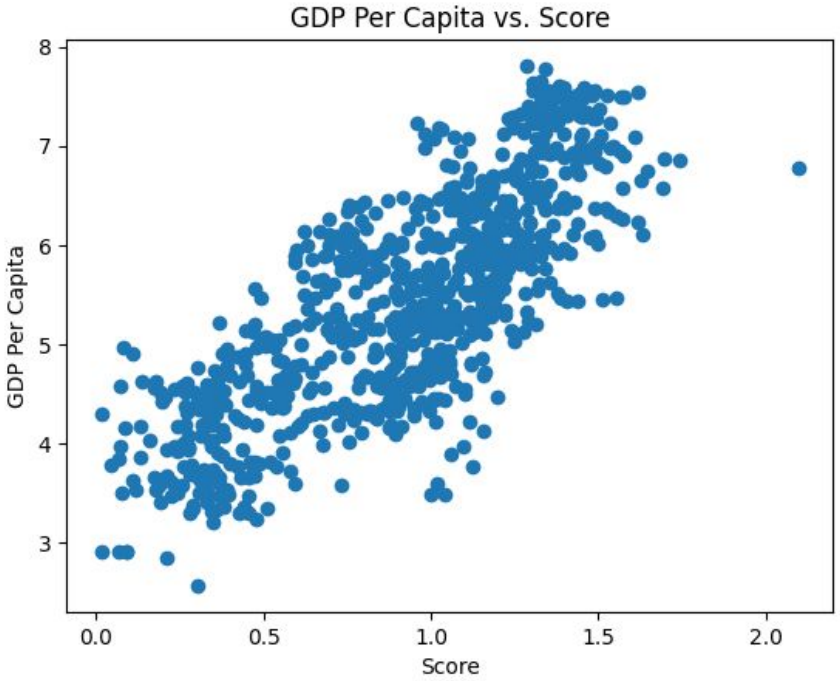
```
X=full_data.drop(columns=["Score","Countries"],axis=1)
y=full_data["Score"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
scaler = RobustScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

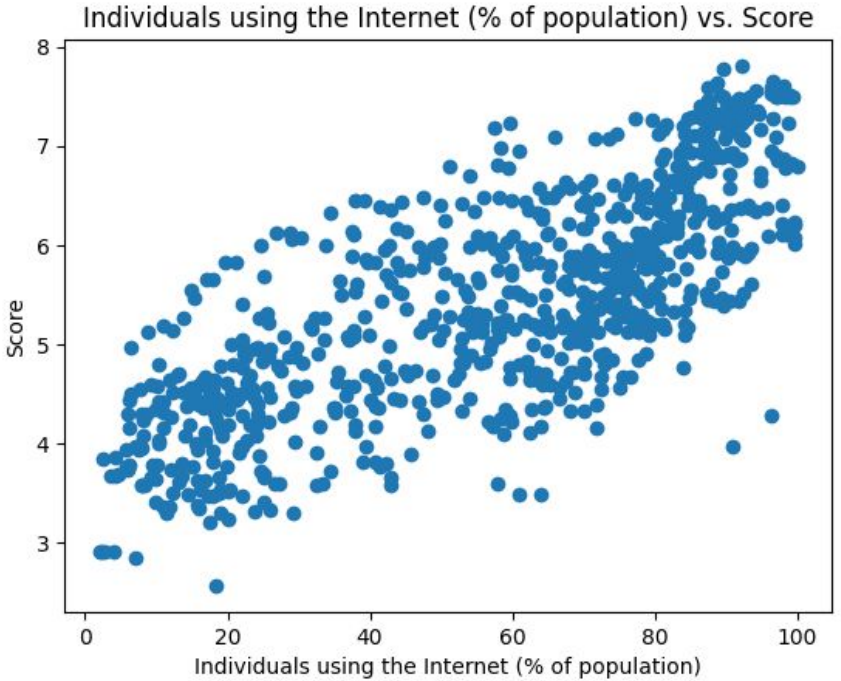


Correlation Matrix Heatmap

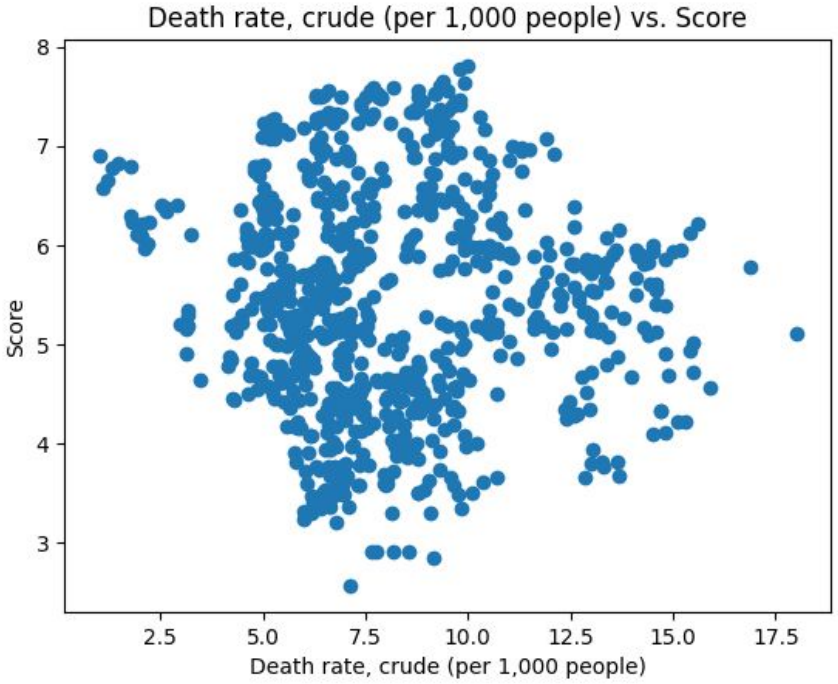


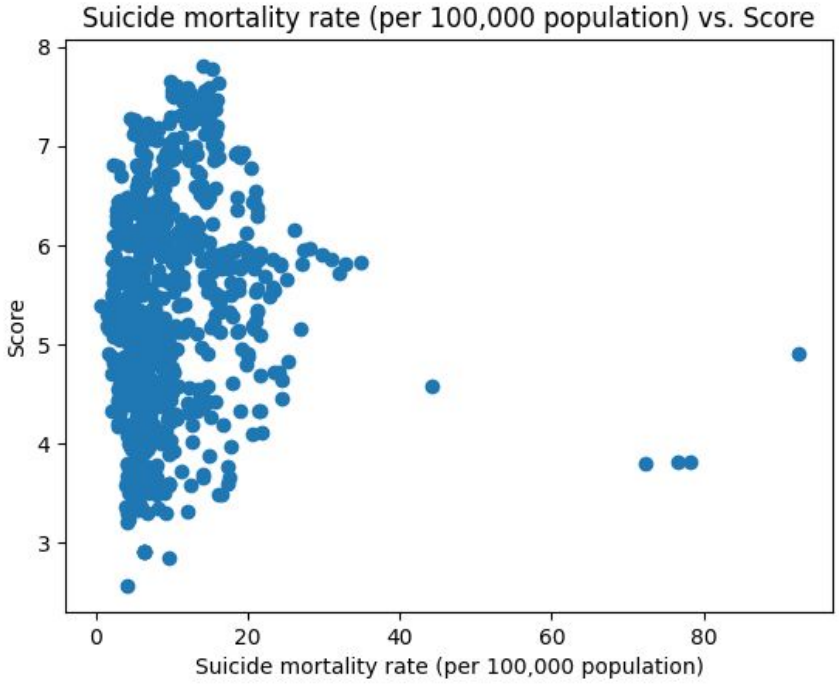
Best Factors Determining Happiness



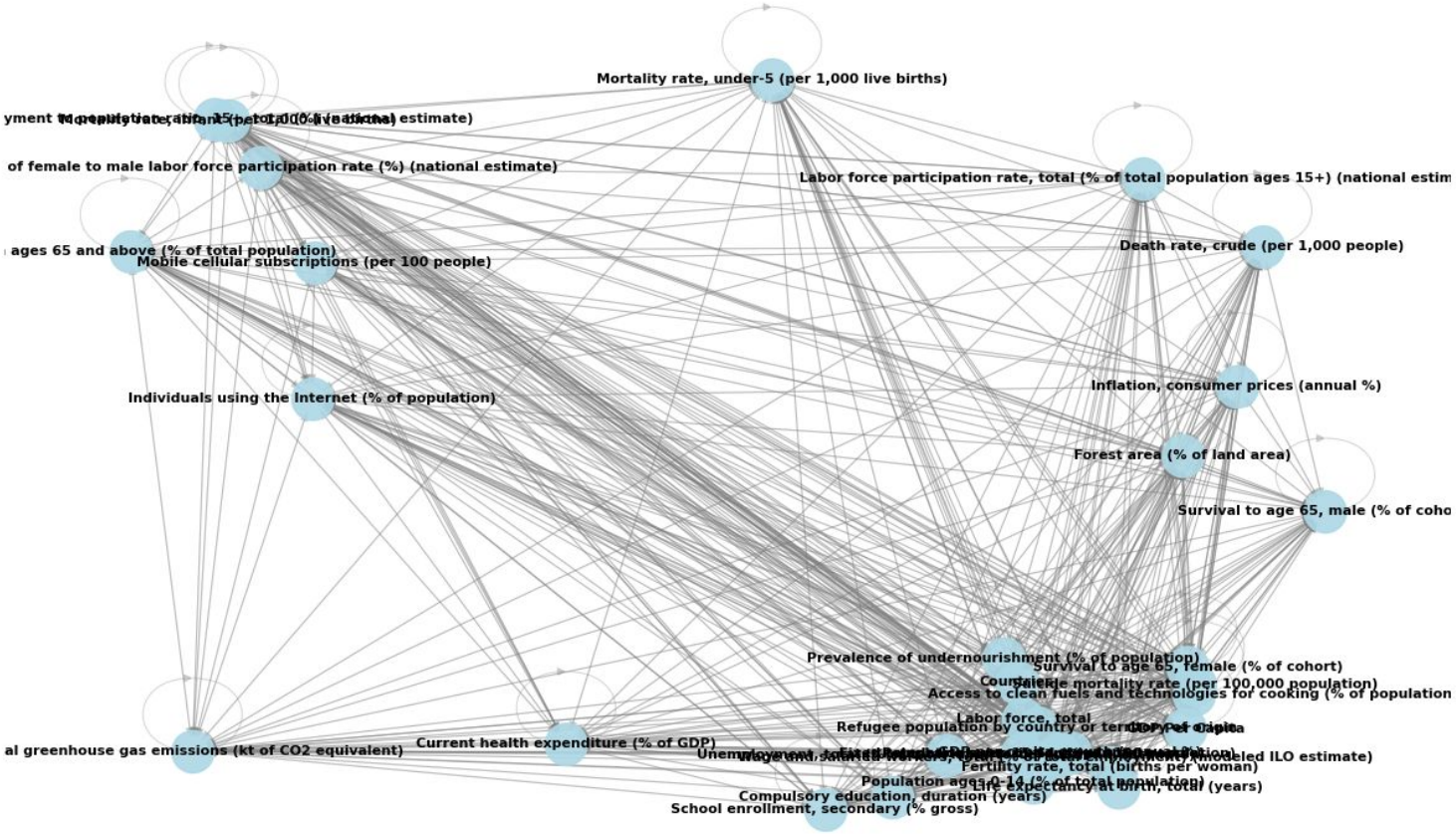


Worst Factors Determining Happiness



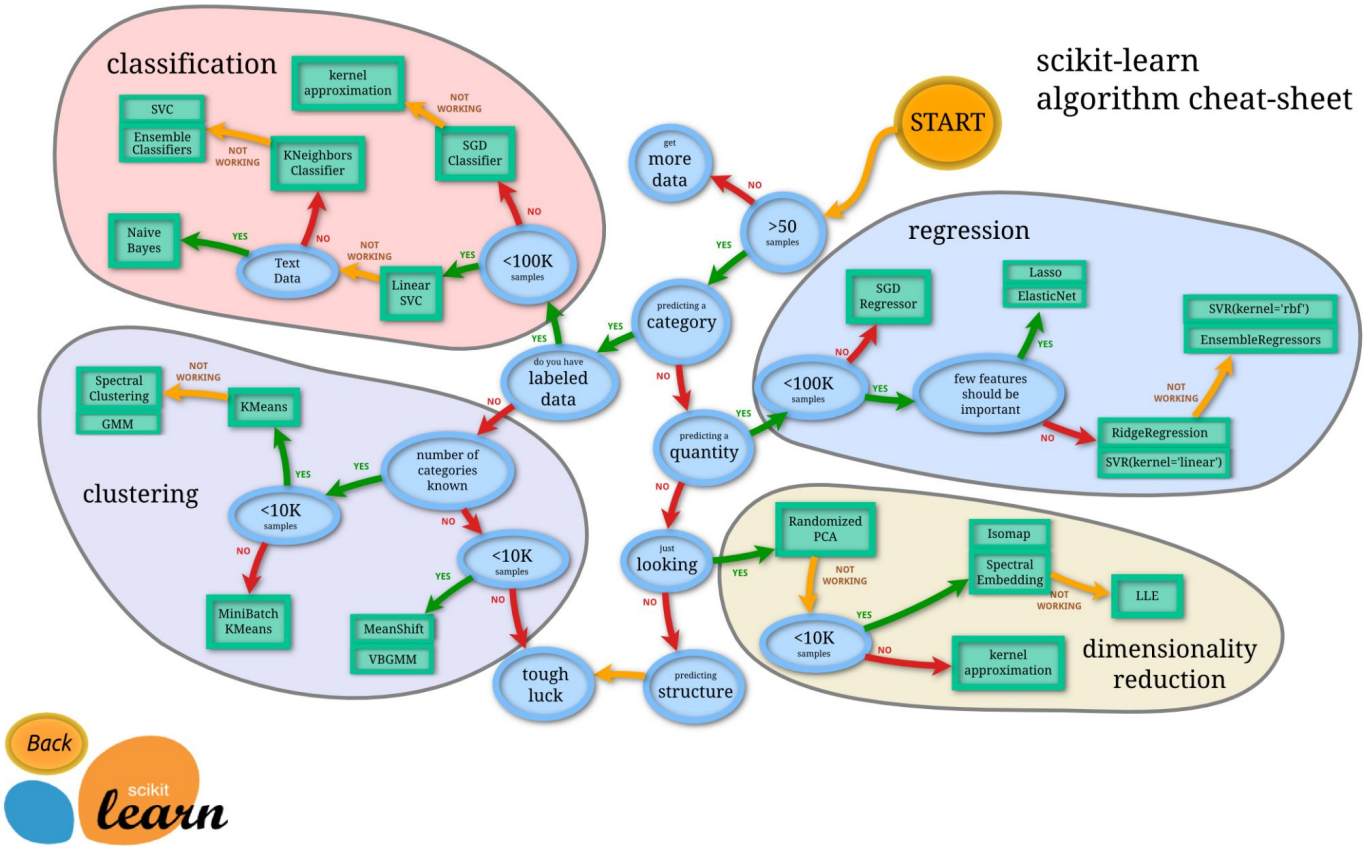


Correlation Network Graph



Regressions

How we chose appropriate models for our predictions:



Linear Regression

```
linear_regression = LinearRegression()

linear_regression.fit(X_train_scaled, y_train)

y_pred = linear_regression.predict(X_test_scaled)

mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
print("Mean Squared Error (MSE) For Linear Regression:", mse)

print("Mean Absolute Error (MAE)For Linear Regression:", mae)
```

```
Mean Squared Error (MSE) For Linear Regression: 0.27993570236620274
Mean Absolute Error (MAE)For Linear Regression: 0.4196652564860049
```

Variable Importance - Linear Regression:

1. Life expectancy at birth, total (years): 1.12

2. Survival to age 65, female (% of cohort): 0.77

3. GDP Per Capita: 0.61

4. Mortality rate, under-5 (per 1,000 live births): 0.42

5. Survival to age 65, male (% of cohort): 0.38

6. Individuals using the Internet (% of population): 0.32

7. Population ages 65 and above (% population): 0.30

8. Access to clean fuels and technologies for cooking (% of population): 0.30

9. Fertility rate, total (births per woman): 0.26

10. Population ages 0-14 (% of total population): 0.24

21. Population ages 65 and above (% of total population): 0.042

22. Fixed telephone subscriptions (per 100 people): 0.064

23. Employment to population ratio, total (%): 0.05

24. Death rate, crude (per 1,000 people): 0.0496

25. Suicide mortality rate (per 100,000 population): 0.033

26. GDP per capita growth (annual %): 0.016

27. Labor force, total: 0.008

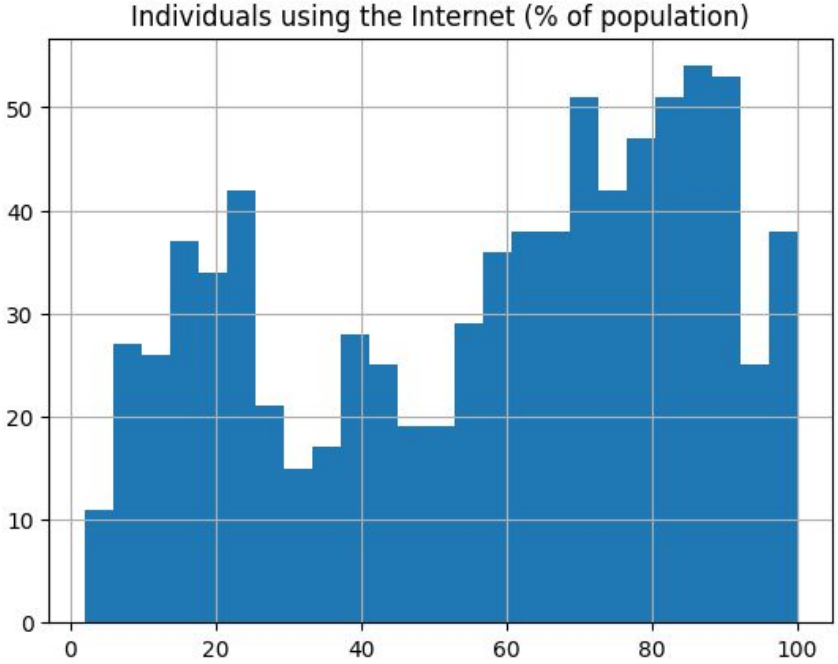
28. Inflation, consumer prices (annual %): 0.006

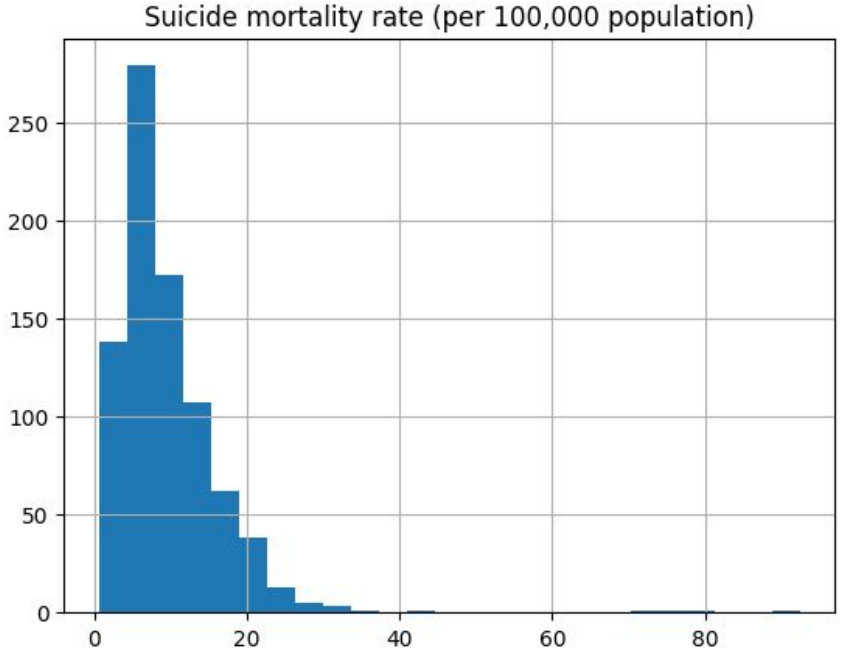
29. Refugee population by country or territory of origin: 0.0027

30. Wage and salaried workers, total (% of total employment): 0.0007

31. Total greenhouse gas emissions (kt of CO2 equivalent): 0.0001

Feature Distributions





Ridge and Lasso Regressions

```
#Ridge regression
ridge_model = Ridge(alpha=1.0)
ridge_model.fit(X_train_scaled, y_train)
y_pred_ridge = ridge_model.predict(X_test_scaled)
mse_ridge = mean_squared_error(y_test, y_pred_ridge)
mae_ridge=mean_absolute_error(y_test, y_pred_ridge)

#Lasso regression
lasso_model = Lasso(alpha=1.0)
lasso_model.fit(X_train_scaled, y_train)
y_pred_lasso = lasso_model.predict(X_test_scaled)
mse_lasso = mean_squared_error(y_test, y_pred_lasso)
mae_lasso=mean_absolute_error(y_test, y_pred_lasso)

print("Mean Squared Error (MSE) - Ridge Regression:", mse_ridge)
print("Mean Absolute Error (MAE) - Ridge Regression:", mae_ridge)

print("Mean Squared Error (MSE) - Lasso Regression:", mse_lasso)
print("Mean Absolute Error (MAE) - Lasso Regression:", mae_lasso)
```

```
Mean Squared Error (MSE) - Ridge Regression: 0.28952036351375227
Mean Absolute Error (MAE) - Ridge Regression: 0.42654257615850394
Mean Squared Error (MSE) - Lasso Regression: 0.8765919036264572
Mean Absolute Error (MAE) - Lasso Regression: 0.7633678209479647
```

Hyper Parameter tuning Ridge & Lasso

```
listofalphas=[]
i=0.001
while i<=10:
    listofalphas.append(i)
    i+=0.001

# Ridge
ridge_model_cv = RidgeCV(alphas=listofalphas)
ridge_model_cv.fit(X_train_scaled, y_train)
y_pred_ridge = ridge_model_cv.predict(X_test_scaled)
mse_ridge = mean_squared_error(y_test, y_pred_ridge)
mae_ridge=mean_absolute_error(y_test, y_pred_ridge)
print("Best alpha for Ridge regression:", ridge_model_cv.alpha_)
print("Mean Squared Error (MSE) - Ridge Regression:", mse_ridge)
print("Mean Absolute Error (MAE) - Ridge Regression:", mae_ridge)
```

```
# Lasso
lasso_model_cv = LassoCV(alphas=[0.001,0.01,0.1,1,10])
lasso_model_cv.fit(X_train_scaled, y_train)
y_pred_lasso = lasso_model_cv.predict(X_test_scaled)
mse_lasso = mean_squared_error(y_test, y_pred_lasso)
mae_lasso=mean_absolute_error(y_test, y_pred_lasso)

print("Best alpha for Lasso regression:", lasso_model_cv.alpha_)
print("Mean Squared Error (MSE) - Lasso Regression:", mse_lasso)
print("Mean Absolute Error (MAE) - Lasso Regression:", mae_lasso)
```

```
Best alpha for Ridge regression: 3.658999999999708
Mean Squared Error (MSE) - Ridge Regression: 0.2939483629501764
Mean Absolute Error (MAE) - Ridge Regression: 0.4302050328894518
Best alpha for Lasso regression: 0.001
Mean Squared Error (MSE) - Lasso Regression: 0.28864928776645477
Mean Absolute Error (MAE) - Lasso Regression: 0.42665275126462776
```

Variable Importance - Lasso Regression:

1. GDP Per Capita: 0.621

2. Survival to age 65, female (% of cohort): 0.57

3. Life expectancy at birth, total (years): 0.52

4. Survival to age 65, male (% of cohort): 0.40

5. Individuals using the Internet (% of population): 0.33

6. Access to clean fuels and technologies for cooking (% of population): 0.26

7. Current health expenditure (% of GDP): 0.23

8. Unemployment, total (% of total labor force): 0.21

9. Compulsory education, duration (years): 0.16

10. Mortality rate, under-5 (per 1,000 live births): 0.15

21. Suicide mortality rate (per 100,000 population): 0.029

22. GDP per capita growth (annual %): 0.019

23. Labor force, total: 0.0063

24. Inflation, consumer prices (annual %): 0.0058

25. Refugee population by country or territory of origin: 0.0033

26. Total greenhouse gas emissions (kt of CO2 equivalent): 0.0015

27. Wage and salaried workers, total (% of total employment) : 0.0

28. Employment to population ratio, total (%): 0.0

29. Population ages 0-14 (% of total population): 0.0

30. Population ages 65 and above (% of total population): 0.0

31. Mortality rate, infant (per 1,000 live births): 0.0

Variable Importance - Ridge Regression:

1. GDP Per Capita: 0.62

2. Survival to age 65, female (% of cohort): 0.39

3. Survival to age 65, male (% of cohort): 0.38

4. Individuals using the Internet (% of population): 0.34

5. Life expectancy at birth, total (years): 0.30

6. Access to clean fuels and technologies for cooking (% of population): 0.25

7. Current health expenditure (% of GDP): 0.23

8. Mortality rate, under-5 (per 1,000 live births): 0.22

9. Unemployment, total (% of total labor force): 0.20

10. Compulsory education, duration (years): 0.16

21. Population ages 65 and above (% of total population):0.042

22. Suicide mortality rate (per 100,000 population): 0.035

23. GDP per capita growth (annual %): 0.022

24. Fixed telephone subscriptions (per 100 people): 0.0197

25. Employment to population ratio, total (%): 0.013

26. Population ages 0-14 (% of total population): 0.0070

27. Inflation, consumer prices (annual %): 0.0060

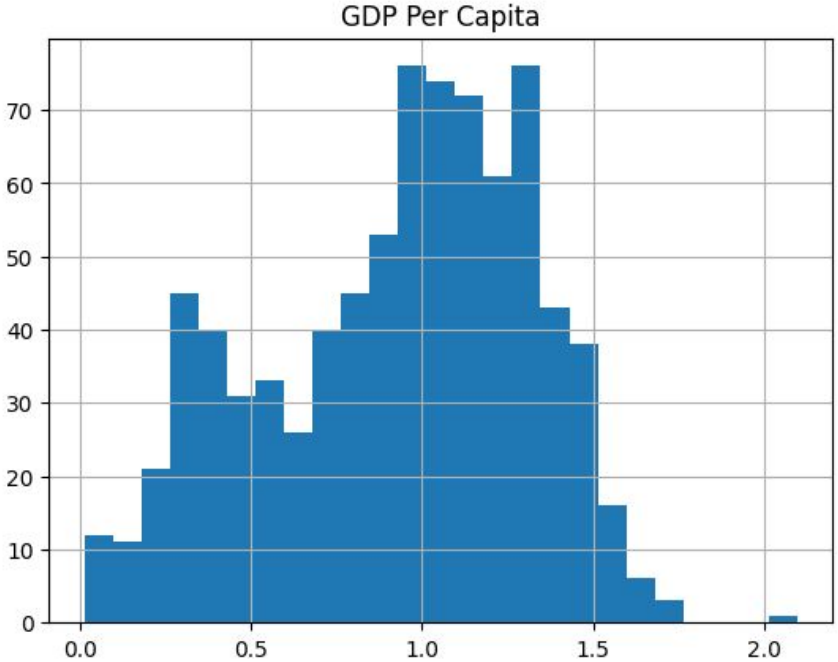
28. Labor force, total: 0.0051

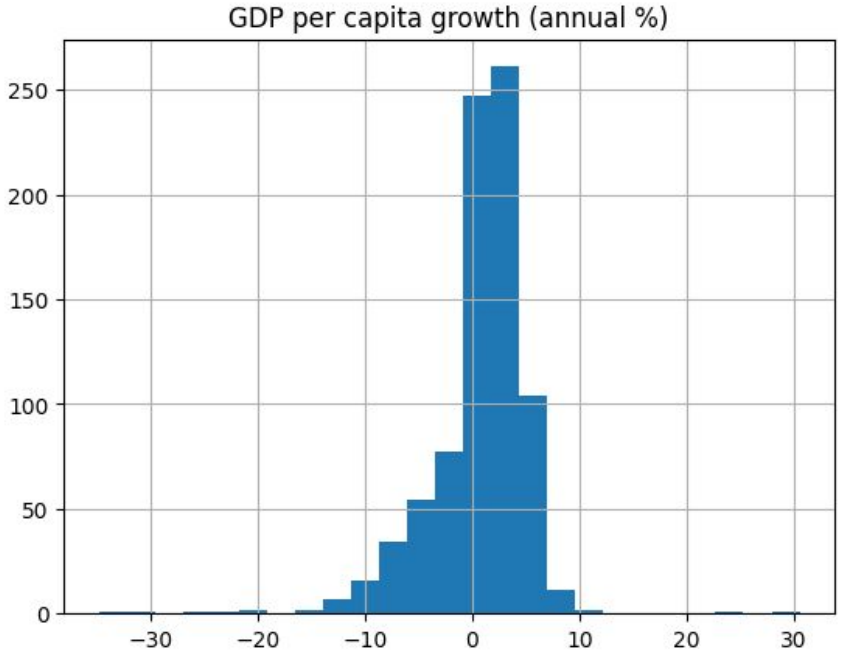
29. Refugee population by country or territory of origin: 0.0034

30. Total greenhouse gas emissions (kt of CO2 equivalent): 0.0024

31. Wage and salaried workers, total (% of total employment): 0.00058

Feature Distributions





ElasticNet

```
enet = ElasticNet()
enet.fit(X_train_scaled, y_train)

y_pred = enet.predict(X_test_scaled)

mse = mean_squared_error(y_test, y_pred)
mae=mean_absolute_error(y_test, y_pred)
print("Mean Squared Error (MSE): ElasticNet", mse)
print("Mean Absolute Error (MAE): ElasticNet", mae)
```

```
Mean Squared Error (MSE): 0.725731915528155
Mean Absolute Error (MAE): 0.6959528782940738
```

Hyper Parameter Tuning: ElasticNet

```
#Cross Validation for ElasticNet

param_grid = {
    'alpha': [0.01, 0.1, 1, 10],
    'l1_ratio': [0.1, 0.3, 0.5, 0.7, 0.9],
    'max_iter': [100, 500, 1000, 10000]
}

# Perform grid search with cross-validation
elasticnet_cv = GridSearchCV(enet, param_grid, cv=5, scoring='neg_mean_squared_error')
elasticnet_cv.fit(X_train, y_train)

# Get the best hyperparameters
best_params = elasticnet_cv.best_params_
print("Best Hyperparameters:", best_params)

# Predict on the test set using the best model
y_pred = elasticnet_cv.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print("Mean Squared Error =",mse)
mae=mean_absolute_error(y_test, y_pred)
print("Mean Absolute Error (MAE):", mae)
```

```
Best Hyperparameters: {'alpha': 0.01, 'l1_ratio': 0.3, 'max_iter': 10000}
Mean Squared Error = 0.27498639305741307
Mean Absolute Error (MAE): 0.41702191956187173
```

Variable Importance - ElasticNet Regression:

1. GDP Per Capita: 0.61

2. Individuals using the Internet (% of population): 0.337

3. Survival to age 65, male (% of cohort): 0.336

4. Survival to age 65, female (% of cohort): 0.26

5. Current health expenditure (% of GDP): 0.24

6. Access to clean fuels and technologies for cooking (% of population): 0.23

7. Unemployment, total (% of total labor force): 0.203

8. Life expectancy at birth, total (years): 0.201

9. Compulsory education, duration (years): 0.16

10. Death rate, crude (per 1,000 people): 0.15

21. Suicide mortality rate (per 100,000 population): 0.034

22. GDP per capita growth (annual %): 0.023

23. Inflation, consumer prices (annual %): 0.0058

24. Refugee population by country or territory of origin: 0.0037

25. Total greenhouse gas emissions (kt of CO2 equivalent): 0.0034

26. Labor force, total: 0.00335

27. Employment to population ratio, 15+, total (%) : 0.000332

28. Fixed telephone subscriptions (per 100 people): 0.0

29. Wage and salaried workers, total (% of total employment): 0.0

30. Population ages 0-14 (% of total population): 0.0

31. Mortality rate, infant (per 1,000 live births): 0.0

Tree Models

Decision Tree Regressor

```
decision_tree = DecisionTreeRegressor(random_state=42)

hyperparameters = {'max_depth': range(1, 10),
                   'min_samples_leaf': range(20, 50, 5)}
grid = GridSearchCV(decision_tree,
                    param_grid=hyperparameters,
                    cv=5,
                    scoring='neg_mean_squared_error')

grid.fit(X_train, y_train)
print('Best Parameters:', grid.best_params_)

y_pred = grid.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
print("Mean Squared Error (MSE) For Decision Tree:", mse)
print("Mean Absolute Error (MAE):", mae)
```

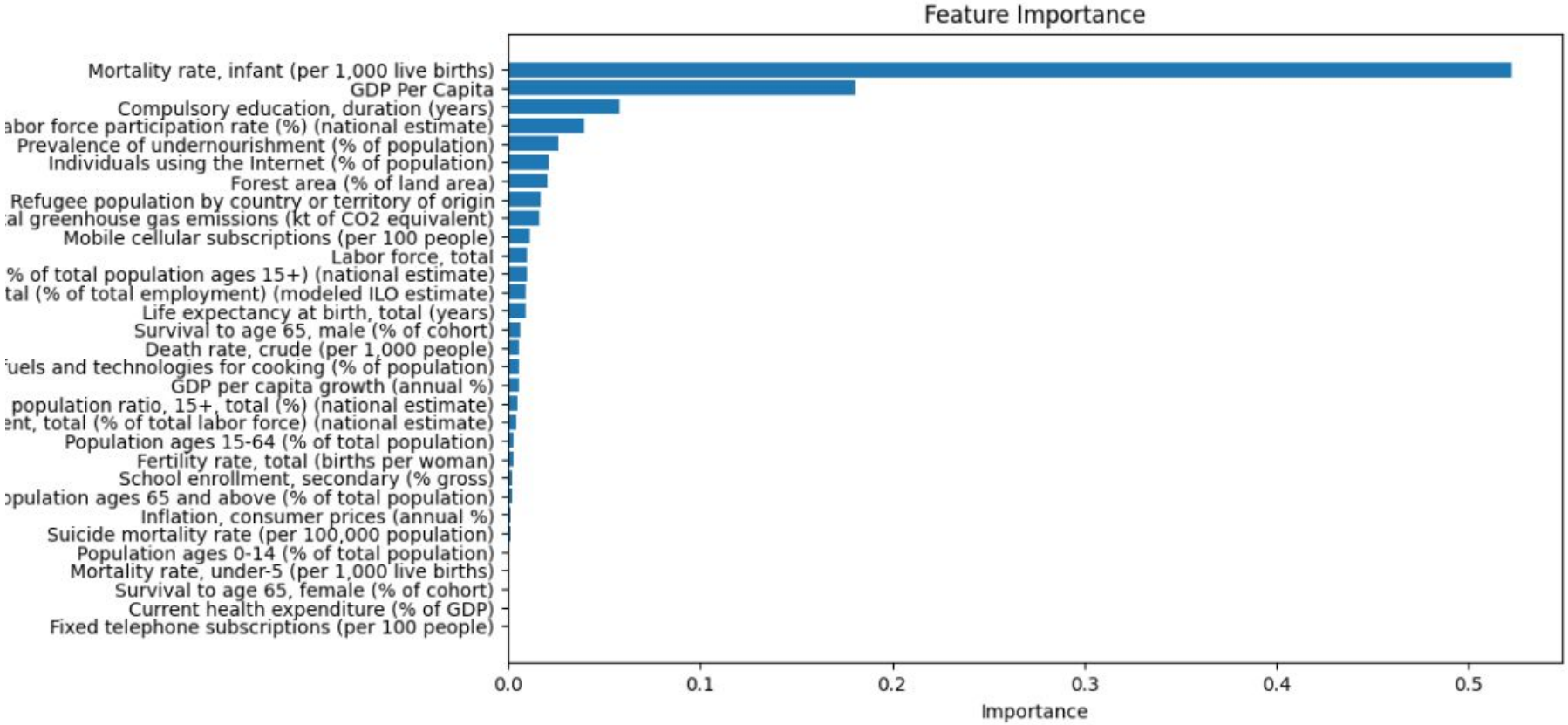
```
Best Parameters: {'max_depth': 7, 'min_samples_leaf': 20}
Mean Squared Error (MSE) For Decision Tree: 0.2797573342333845
Mean Absolute Error (MAE): 0.4067104905423586
```

Random Forest Regressor

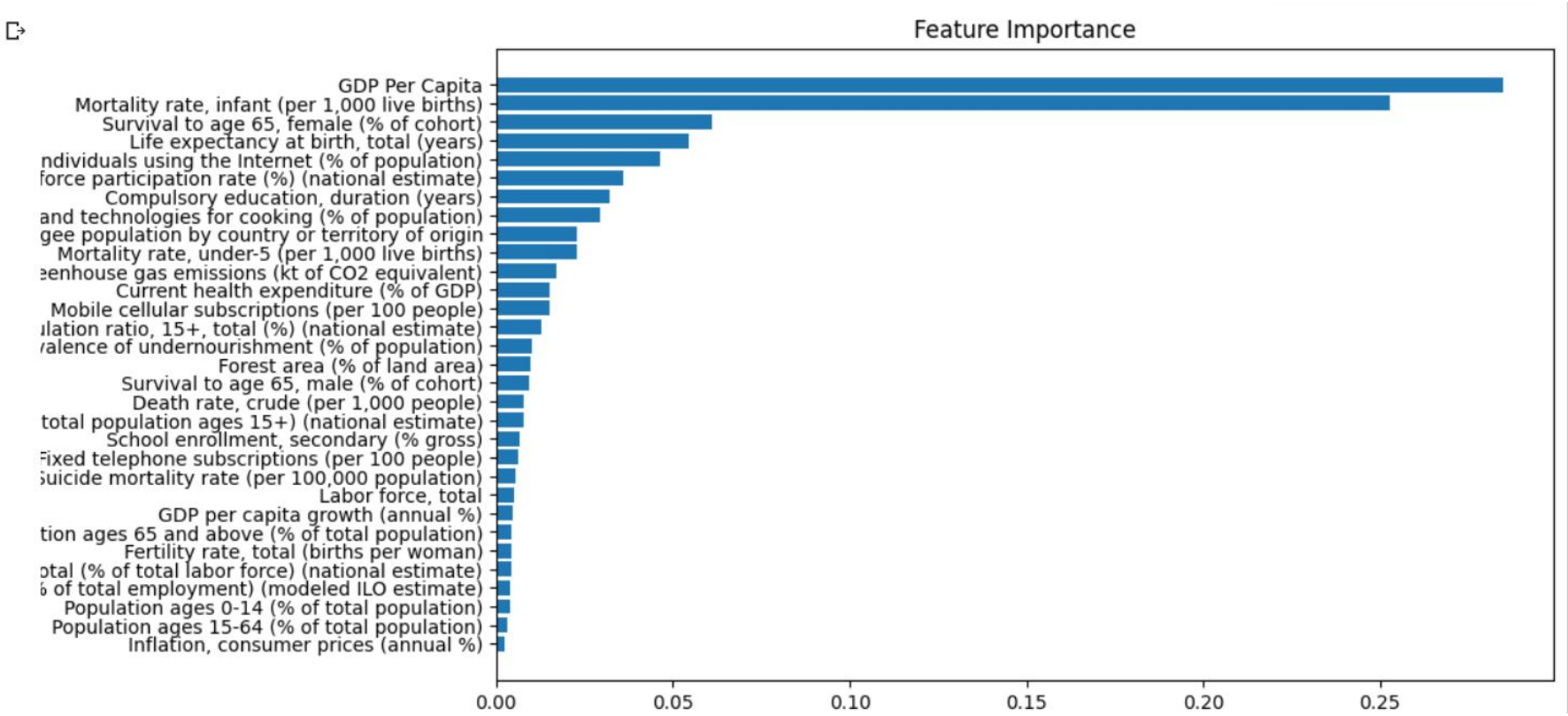
```
random_forest = RandomForestRegressor(random_state=42)

hyperparameters = {'max_depth': range(1, 10),
                   'min_samples_leaf': range(20, 50, 5)}
grid = GridSearchCV(random_forest,
                    param_grid=hyperparameters,
                    cv=5,
                    scoring='neg_mean_squared_error')

grid.fit(X_train, y_train)
print('Best Parameters:', grid.best_params_)

y_pred = grid.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
print("Mean Squared Error (MSE) For Random Forest:", mse)
print("Mean Absolute Error (MAE):", mae)
```

```
Best Parameters: {'max_depth': 6, 'min_samples_leaf': 20}
Mean Squared Error (MSE) For Random Forest: 0.2121827919607727
Mean Absolute Error (MAE): 0.36936815299226083
```

Feature Importance for Decision Tree Regressor



Feature Importance for Random Forest Regressor



Summary of MSE / MAE

Error	Ridge	ElasticNet	Lasso	Linear	Decision Tree	Random Forest
MSE	0.29395	0.29290	0.28864	0.27994	0.27976	0.21218
MAE	0.43021	0.42977	0.42665	0.41967	0.40671	0.36937

Testing

We created a function that takes a model and a given year, trains the data on the other 5 available years and then predicts with the given model what would the Happiness Ranking in that year be.

```
def predicted_ranking(model,year):
  indexx = [0,137,273,410,547,684,823]
  if year <2015 or year > 2020:
    raise ValueError("Year must be between 2015 and 2020.")
  else:
    i = year - 2015
    X_test = full_data.iloc[indexx[i]:indexx[i+1]].copy()
    y_test = X_test['Score']
    countries = X_test['Countries']
    X_test.drop(columns=['Countries', 'Score'], inplace=True)

    X_train = full_data.copy()
    X_train.drop(X_train.index[indexx[i]:indexx[i+1]], inplace=True)
    y_train = X_train['Score']
    X_train.drop(columns=['Countries', 'Score'], inplace=True)

    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

    model.fit(X_train_scaled,y_train)
    y_pred = model.predict(X_test_scaled)

    y_test_df = y_test.reset_index().rename(columns={'Score': 'Score'})
    y_test_df['Prediction'] = y_pred
    y_test_df.index = countries
    y_test_df = y_test_df.drop("index", axis=1)
```

```
y_test_df = y_test_df.sort_values("Score",ascending=False)
ord1 = y_test_df.index.to_numpy()
y_test_df = y_test_df.sort_values("Prediction",ascending = False)
ord2 = y_test_df.index.to_numpy()

delta = np.zeros(len(ord2),dtype=int)
for i,elem in enumerate(ord2):
  ord1_index = np.where(ord1 == elem)[0][0]
  delta[i] = i - ord1_index
avg_delta = np.mean(np.absolute(delta))

res = {"Place":range(1,len(ord1)+1),"Actual Ranking":ord1,"Predicted Ranking":ord2,"Delta":delta}
results = pd.DataFrame(res)

print(results.head(25))
print(results.tail(25))
print("The average delta of the prediction is ",avg_delta)
return results, avg_delta
```

Some Predicted Rankings

```
result,delta = predicted_ranking(LinearRegression(),2016)
print(result)
print("Average delta for Linear Regression is ",delta)
```

	Place	Actual Ranking	Predicted Ranking	Delta
0	1	Denmark	Norway	-3
1	2	Switzerland	Switzerland	0
2	3	Iceland	United States	-10
3	4	Norway	Iceland	1
4	5	Finland	Sweden	-5
..	...	...	...	...
131	132	Rwanda	Madagascar	4
132	133	Benin	Afghanistan	-1
133	134	Afghanistan	Haiti	15
134	135	Togo	Rwanda	3
135	136	Burundi	Uganda	10

```
[136 rows x 4 columns]
Average delta for Linear Regression is  13.867647058823529
```

```
result,delta = predicted_ranking(elasticnet_cv,2017)
print(result)
print("Average delta for Elastic Net is ",delta)
```

	Place	Actual Ranking	Predicted Ranking	Delta
0	1	Norway	Sweden	-8
1	2	Denmark	Norway	1
2	3	Iceland	Switzerland	-1
3	4	Switzerland	United States	-10
4	5	Finland	Iceland	2
..	...	...	...	...
132	133	Guinea	Burundi	-4
133	134	Togo	Madagascar	5
134	135	Rwanda	Chad	12
135	136	Tanzania	Rwanda	1
136	137	Burundi	Afghanistan	11

```
[137 rows x 4 columns]
Average delta for Elastic Net is  13.722627737226277
```

Some Predicted Rankings

```
result,delta = predicted_ranking(lasso_model_cv,2018)
print(result)
print("Average delta for Lasso Regression is ",delta)
```

	Place	Actual Ranking	Predicted Ranking	Delta
0	1	Finland	United Arab Emirates	-19
1	2	Norway	Sweden	-7
2	3	Denmark	Switzerland	-2
3	4	Iceland	Norway	2
4	5	Switzerland	Iceland	1
..	...	...	...	...
132	133	Liberia	Burundi	-4
133	134	Rwanda	Haiti	2
134	135	Yemen	Rwanda	1
135	136	Tanzania	Madagascar	9
136	137	Burundi	Afghanistan	8

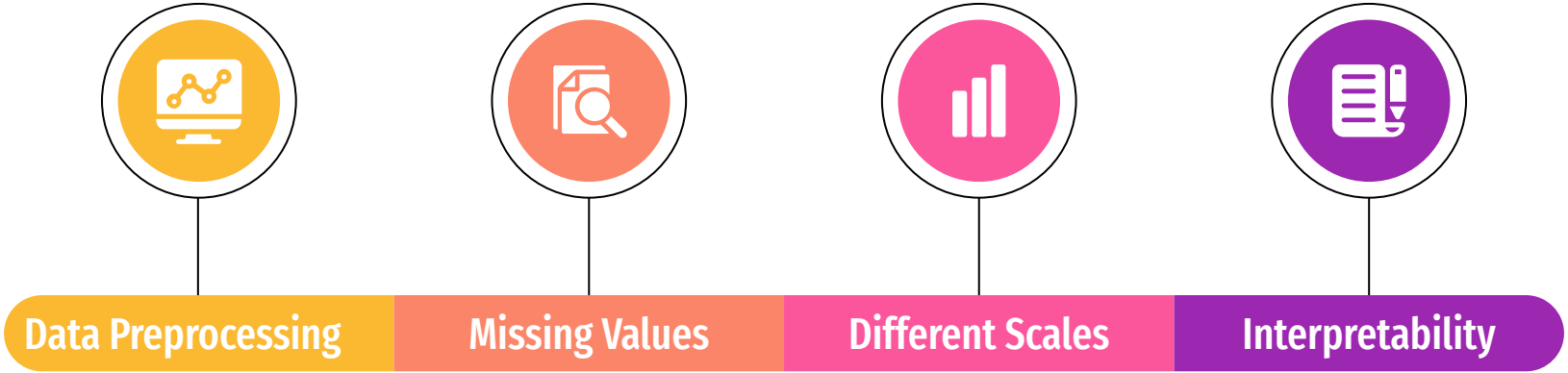
```
[137 rows x 4 columns]
Average delta for Lasso Regression is  13.532846715328468
```

```
result,delta = predicted_ranking(ridge_model_cv,2019)
print(result)
print("Average delta for Ridge Regression is ",delta)
```

	Place	Actual Ranking	Predicted Ranking	Delta
0	1	Finland	Sweden	-6
1	2	Denmark	Norway	-1
2	3	Norway	Switzerland	-3
3	4	Iceland	Iceland	0
4	5	Netherlands	United States	-14
..	...	...	...	...
132	133	Malawi	Haiti	2
133	134	Yemen	Chad	17
134	135	Rwanda	Afghanistan	-2
135	136	Tanzania	Rwanda	1
136	137	Afghanistan	Madagascar	10

```
[137 rows x 4 columns]
Average delta for Ridge Regression is  13.97080291970803
```

Main difficulties



Conclusions

- Norway or Sweden are most often predicted as the happiest country in the world, while Ireland and United Arab Emirates both win it on 2 of our simulations

- Survival to age 65, life expectancy at birth and internet user percentage have a big impact regardless of the chosen model

- GDP and life expectancy which are the top 3 most important variables in the original rankings usually stand in our top 5 most important variables but we believe it is because we do the predictions based on their scores more so than because out of the whole list they predict happiness best

- The real World Happiness Rankings have highly subjective data and could benefit from including more factors that are not self-reported in surveys

References

Find the code here: https://github.com/yassine-turki/CSE204/tree/main

Happiness Scores from Kaggle: https://www.kaggle.com/datasets/mathurinache/world-happiness-report?select=2022.csv

World Happiness Report (general document to understand the approach):
https://happiness-report.s3.amazonaws.com/2022/WHR+22.pdf

Background Information on the World Happiness Report:
https://www.euronews.com/next/2023/03/20/these-are-the-worlds-10-happiest-countries-in-2023-and-most-of-them-are-in-europe

World Development Indicators Data Source: https://databank.worldbank.org/source/world-development-indicators#

Thank you!

Any Questions?