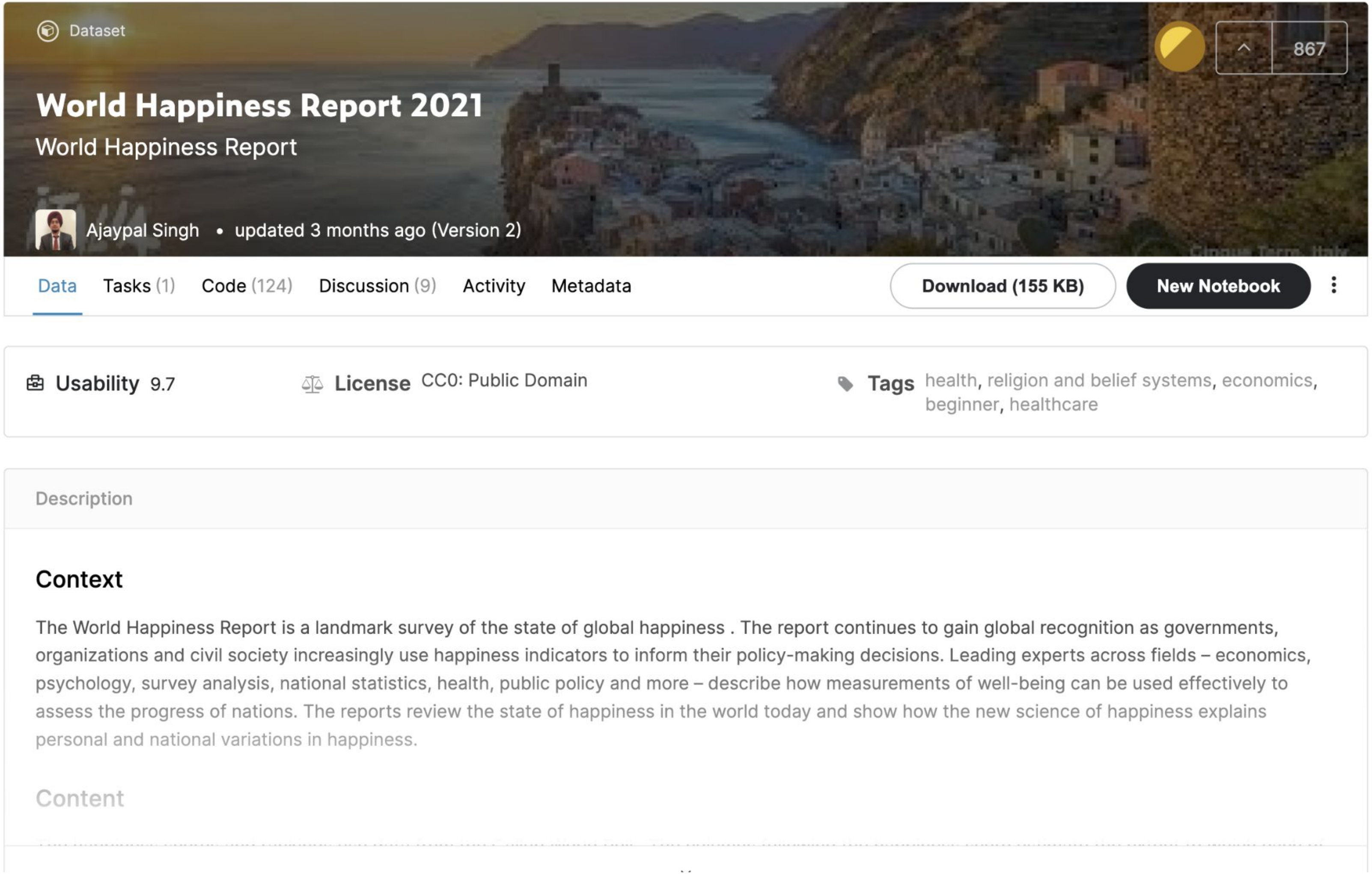
Group 8
World Happiness Report

Andy Lyu, Jax Luo, Shirley Yu,
Morgan Zhang


Dataset

World Happiness Report 2021

World Happiness Report

Ajaypal Singh • updated 3 months ago (Version 2)

867

Data Tasks (1) Code (124) Discussion (9) Activity Metadata

Download (155 KB) New Notebook

Usability 9.7

License CC0: Public Domain

Tags health, religion and belief systems, economics, beginner, healthcare

Description

Context

The World Happiness Report is a landmark survey of the state of global happiness . The report continues to gain global recognition as governments, organizations and civil society increasingly use happiness indicators to inform their policy-making decisions. Leading experts across fields – economics, psychology, survey analysis, national statistics, health, public policy and more – describe how measurements of well-being can be used effectively to assess the progress of nations. The reports review the state of happiness in the world today and show how the new science of happiness explains personal and national variations in happiness.

Content

Processing Data

```python
# 2. We delte the countries with only one row of values, for which we cannot find the current column mean and there
#    is no reason to set these "nan" values to "0"
#    For example, if we check country Cuba:

CUBA = pd.DataFrame(world_happiness_dict.get('Cuba'))
print(CUBA)
# Therefore we find all of such countries and delete them from our dataset
country_with_one_row_data = []
for country in world_happiness_dict.keys():
    if len(world_happiness_dict.get(country)) == 1:
        country_with_one_row_data.append(country)
print("the countries with only one row of data are:", country_with_one_row_data)
print("We delete these 5")
for country in country_with_one_row_data:
    world_happiness_dict.pop(country, None)
```

```
      0      1    2     3      4      5    6    7      8      9
0  2006  5.418  NaN  0.97  68.44  0.281  NaN  NaN  0.647  0.277
the countries with only one row of data are: ['Cuba', 'Guyana', 'Maldives', 'Oman', 'Suriname']
We delete these 5
```

Processing Data

```
# 3. There is another case that we miss one of the columns for one country and we cannot find mean for that row
#    Therefore we traverse all the keys in the dictionary and for each country,
filled_data = defaultdict()
for country in world_happiness_dict.keys():
    df = pd.DataFrame(world_happiness_dict.get(country))
    filled_df = df.fillna(df.mean())

    if filled_df.isnull().values.any():
    # we illustrate the conditions of those countries before and after the procedure:
        print(country,"BEFORE:")
        print(filled_df)

        filled_df = df.fillna(0)
        print(country, "AFTER:")
        print(filled_df)


    filled_data[country] = filled_df
```

China BEFORE:

	0	1	2	3	4	5	6	7	8	9
0	2006	4.560	8.696	0.747	66.88	0.8457	-0.170643	NaN	0.809	0.170
1	2007	4.863	8.824	0.811	67.06	0.8457	-0.176000	NaN	0.817	0.159
2	2008	4.846	8.911	0.748	67.24	0.8530	-0.092000	NaN	0.817	0.147
3	2009	4.454	8.996	0.798	67.42	0.7710	-0.160000	NaN	0.786	0.162
4	2010	4.653	9.092	0.768	67.60	0.8050	-0.133000	NaN	0.765	0.158
5	2011	5.037	9.179	0.787	67.76	0.8240	-0.186000	NaN	0.820	0.134
6	2012	5.095	9.249	0.788	67.92	0.8080	-0.185000	NaN	0.821	0.159
7	2013	5.241	9.319	0.778	68.08	0.8050	-0.158000	NaN	0.836	0.142
8	2014	5.196	9.386	0.820	68.24	0.8457	-0.217000	NaN	0.854	0.112
9	2015	5.304	9.449	0.794	68.40	0.8457	-0.244000	NaN	0.809	0.171
10	2016	5.325	9.510	0.742	68.70	0.8457	-0.228000	NaN	0.826	0.146
11	2017	5.099	9.571	0.772	69.00	0.8780	-0.175000	NaN	0.821	0.214
12	2018	5.131	9.632	0.788	69.30	0.8950	-0.159000	NaN	0.856	0.190
13	2019	5.144	9.688	0.822	69.60	0.9270	-0.173000	NaN	0.891	0.147
14	2020	5.771	9.702	0.808	69.90	0.8910	-0.103000	NaN	0.789	0.245

China AFTER:

	0	1	2	3	4	5	6	7	8	9
0	2006	4.560	8.696	0.747	66.88	0.000	0.000	0.0	0.809	0.170
1	2007	4.863	8.824	0.811	67.06	0.000	-0.176	0.0	0.817	0.159
2	2008	4.846	8.911	0.748	67.24	0.853	-0.092	0.0	0.817	0.147
3	2009	4.454	8.996	0.798	67.42	0.771	-0.160	0.0	0.786	0.162
4	2010	4.653	9.092	0.768	67.60	0.805	-0.133	0.0	0.765	0.158
5	2011	5.037	9.179	0.787	67.76	0.824	-0.186	0.0	0.820	0.134
6	2012	5.095	9.249	0.788	67.92	0.808	-0.185	0.0	0.821	0.159
7	2013	5.241	9.319	0.778	68.08	0.805	-0.158	0.0	0.836	0.142
8	2014	5.196	9.386	0.820	68.24	0.000	-0.217	0.0	0.854	0.112
9	2015	5.304	9.449	0.794	68.40	0.000	-0.244	0.0	0.809	0.171
10	2016	5.325	9.510	0.742	68.70	0.000	-0.228	0.0	0.826	0.146
11	2017	5.099	9.571	0.772	69.00	0.878	-0.175	0.0	0.821	0.214
12	2018	5.131	9.632	0.788	69.30	0.895	-0.159	0.0	0.856	0.190
13	2019	5.144	9.688	0.822	69.60	0.927	-0.173	0.0	0.891	0.147
14	2020	5.771	9.702	0.808	69.90	0.891	-0.103	0.0	0.789	0.245

```
Algeria BEFORE:
      0      1      2      3      4      5      6      7      8      9
0  2010  5.464  9.287    NaN  64.50  0.593 -0.205  0.618    NaN    NaN
1  2011  5.317  9.297  0.810  64.66  0.530 -0.181  0.638  0.550  0.255
2  2012  5.605  9.311  0.839  64.82  0.587 -0.172  0.690  0.604  0.230
3  2014  6.355  9.335  0.818  65.14    NaN    NaN    NaN  0.626  0.177
4  2016  5.341  9.362  0.749  65.50    NaN    NaN    NaN  0.661  0.377
5  2017  5.249  9.354  0.807  65.70  0.437 -0.167  0.700  0.642  0.289
6  2018  5.043  9.348  0.799  65.90  0.583 -0.146  0.759  0.591  0.293
7  2019  4.745  9.337  0.803  66.10  0.385  0.005  0.741  0.585  0.215
Algeria AFTER:
      0      1      2         3      4         5         6      7         8  \
0  2010  5.464  9.287  0.803571  64.50  0.593000 -0.205000  0.618  0.608429
1  2011  5.317  9.297  0.810000  64.66  0.530000 -0.181000  0.638  0.550000
2  2012  5.605  9.311  0.839000  64.82  0.587000 -0.172000  0.690  0.604000
3  2014  6.355  9.335  0.818000  65.14  0.519167 -0.144333  0.691  0.626000
4  2016  5.341  9.362  0.749000  65.50  0.519167 -0.144333  0.691  0.661000
5  2017  5.249  9.354  0.807000  65.70  0.437000 -0.167000  0.700  0.642000
6  2018  5.043  9.348  0.799000  65.90  0.583000 -0.146000  0.759  0.591000
7  2019  4.745  9.337  0.803000  66.10  0.385000  0.005000  0.741  0.585000

          9
0  0.262286
1  0.255000
2  0.230000
3  0.177000
4  0.377000
5  0.289000
6  0.293000
7  0.215000
```

Processing Data

```
Training data dimensions:  (1515, 10)
Testing data dimensions:  (429, 10)
        0      1      2      3      4      5      6      7      8      9
3    2011  3.832  7.620  0.521  51.92  0.496  0.162  0.731  0.611  0.267
2    2010  4.758  7.647  0.539  51.60  0.600  0.121  0.707  0.618  0.275
7    2015  3.983  7.702  0.529  53.20  0.389  0.080  0.881  0.554  0.339
5    2013  3.572  7.725  0.484  52.56  0.578  0.061  0.823  0.621  0.273
6    2014  3.131  7.718  0.526  52.88  0.509  0.104  0.871  0.532  0.375
..    ...    ...    ...    ...    ...    ...    ...    ...    ...    ...
4    2010  4.682  7.729  0.857  46.70  0.665 -0.093  0.828  0.748  0.122
1    2007  3.280  7.666  0.828  42.86  0.456 -0.082  0.946  0.661  0.265
11   2017  3.638  8.016  0.754  55.00  0.753 -0.098  0.751  0.806  0.224
14   2020  3.160  7.829  0.717  56.80  0.643 -0.009  0.789  0.703  0.346
0    2006  3.826  7.711  0.822  41.58  0.431 -0.076  0.905  0.715  0.297

[1515 rows x 10 columns]
```

Goal

- To find what the world needs to improve on the most to achieve a higher quality of life.
- Create a regression model that can predict the happiness score given certain social parameters.
- Explain some decisions the model took to arrive at a given conclusion.

Approach

- A random forest regressor trained on the world happiness dataset provided via Kaggle
  - Why random forest?
    - We combine the low bias and high variance trait of decision trees, merge a “forest” of trees to arrive at a more desired outcome.
    - Through visualizing each individual decision tree, we can somewhat reason why the model decided on the decision that was outputted.
- Elevate each individual parameter by a percentage and seeing how much that would impact the happiness prediction outcome.
  - Take the maximum happiness that is achieved through raising all parameters by the same scale factor.
- Why not use linear regression?
  - The human mind returns to equilibrium
  - Infinite money does not correlate with infinite happiness

Methodology

Random forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time.

What is the decision tree?

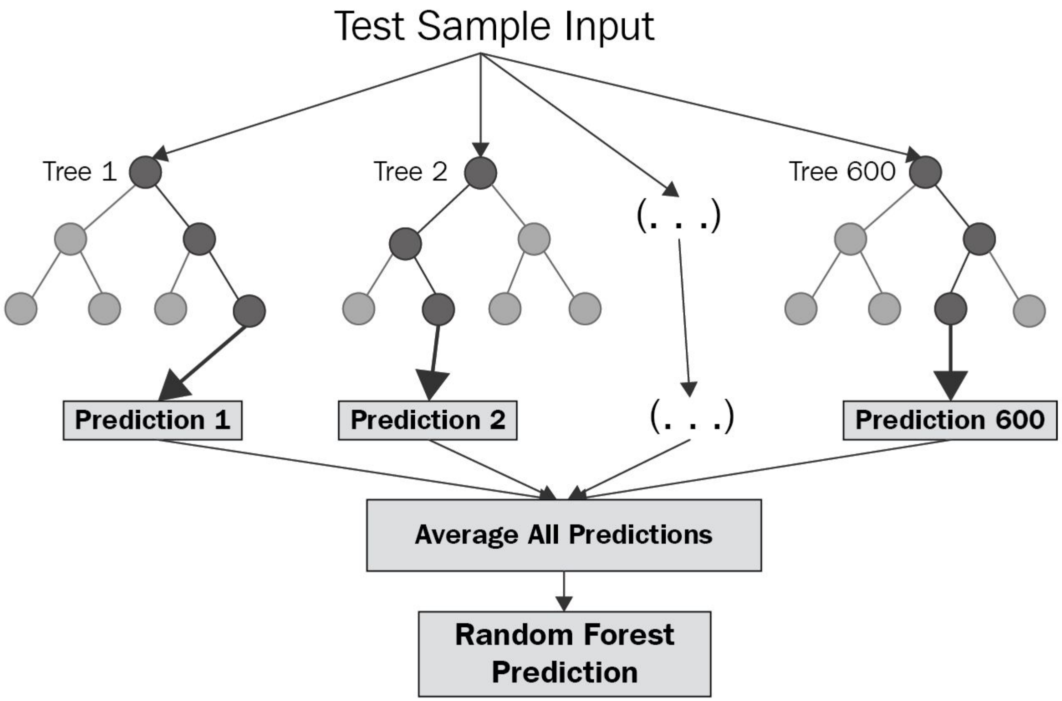
- ➢ A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility.
- ➢ Decision trees has low bias and high variance, the tradeoff depending on the depth of the tree and how it splits.
  - ○ Low bias high variance means each tree fits its training model too well, causing overfitting and learning the noise associated with the dataset.

Methodology

Why the name ‘random forest?’

- Each decision tree in the forest considers a random subset of features when forming questions and only has access to a random set of the training data points.
- The diversity in the forest leads to more robust overall predictions and the name ‘random forest.’
- A random forest regressor aggregates each tree decision and takes an average

Methodology

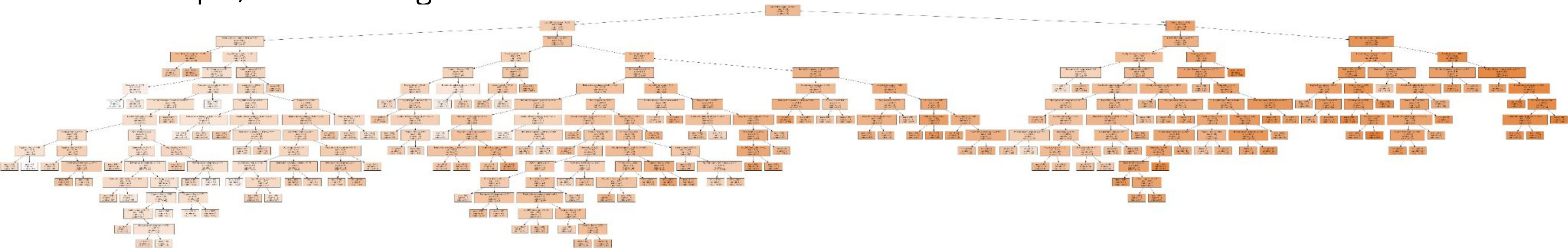


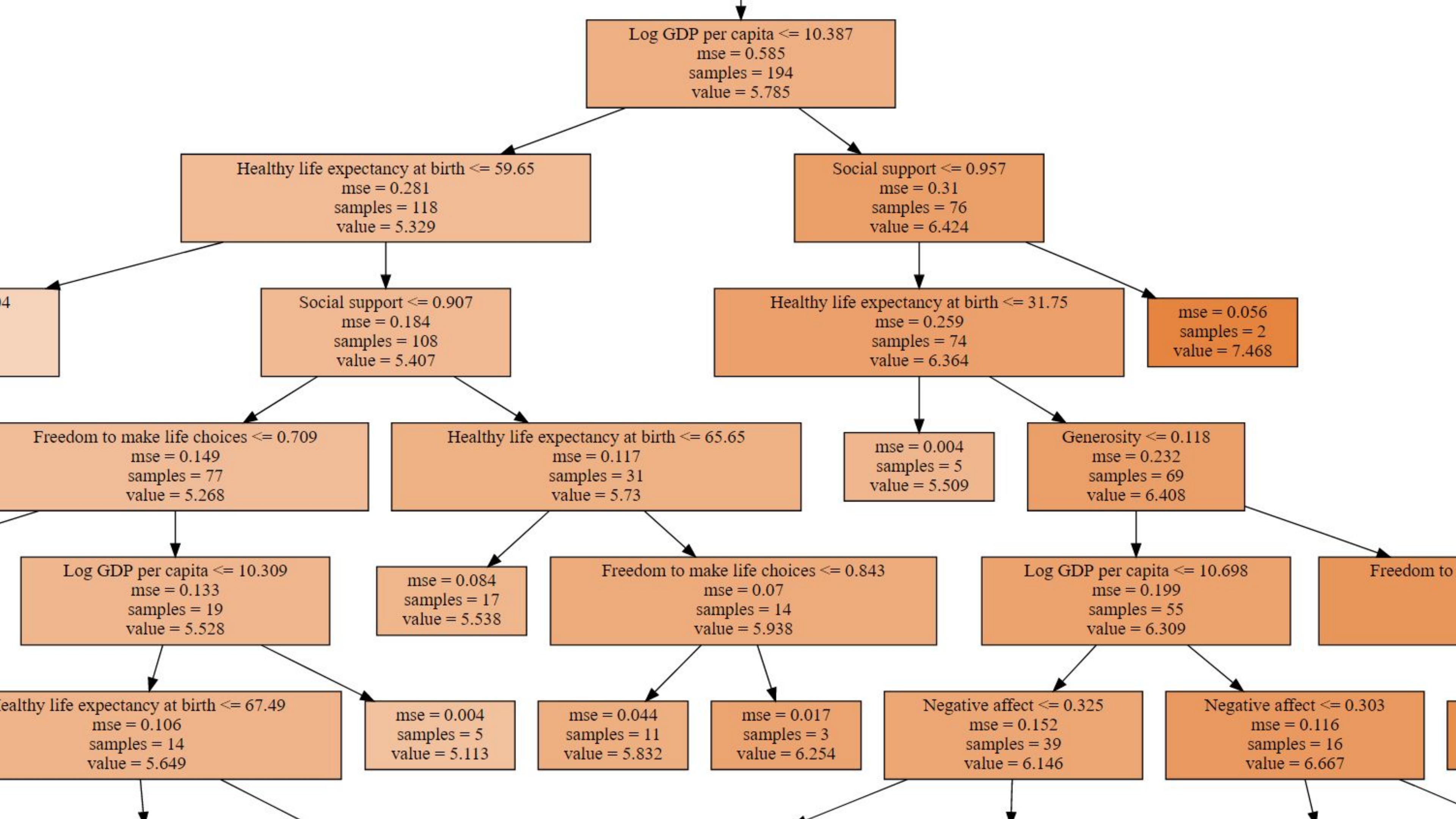
Implementation

To arrive at an estimate, we

- set a series of questions to narrow possible values until reaching a single prediction.
- repeat this decision process over and over again.
- Optimal testing reveals 300 trees, each with 150 nodes max per tree

For example, here is a single tree in our model random forest visualized:




Log GDP per capita <= 10.387
mse = 0.585
samples = 194
value = 5.785
Healthy life expectancy at birth <= 59.65
mse = 0.281
samples = 118
value = 5.329
Social support <= 0.957
mse = 0.31
samples = 76
value = 6.424
Social support <= 0.907
mse = 0.184
samples = 108
value = 5.407
Healthy life expectancy at birth <= 31.75
mse = 0.259
samples = 74
value = 6.364
mse = 0.056
samples = 2
value = 7.468
Freedom to make life choices <= 0.709
mse = 0.149
samples = 77
value = 5.268
Healthy life expectancy at birth <= 65.65
mse = 0.117
samples = 31
value = 5.73
mse = 0.004
samples = 5
value = 5.509
Generosity <= 0.118
mse = 0.232
samples = 69
value = 6.408
Log GDP per capita <= 10.309
mse = 0.133
samples = 19
value = 5.528
mse = 0.084
samples = 17
value = 5.538
Freedom to make life choices <= 0.843
mse = 0.07
samples = 14
value = 5.938
Log GDP per capita <= 10.698
mse = 0.199
samples = 55
value = 6.309
Freedom to
ealthy life expectancy at birth <= 67.49
mse = 0.106
samples = 14
value = 5.649
mse = 0.004
samples = 5
value = 5.113
mse = 0.044
samples = 11
value = 5.832
mse = 0.017
samples = 3
value = 6.254
Negative affect <= 0.325
mse = 0.152
samples = 39
value = 6.146
Negative affect <= 0.303
mse = 0.116
samples = 16
value = 6.667

Results

- Average difference on testing set: 3%
  - Average of (abs(predicted_y - actual_y) / actual_y)
- Coefficient of determination R^2 (forest score): 0.84
  - Better accuracy the closer it is to 1
  - $R^2 = (1 - u/v)$
    - u defined as the residual sum of squares ((y_true - y_pred) ** 2).sum()
    - v defined as the total sum of squares ((y_true - y_true.mean()) ** 2).sum()

Results

- Feature best to improve the world with 1% scale up:
  - Social support
- Feature best to improve the world with 10%, 50%, and 100% scale up:
  - Log GDP per Capita
- Feature scaling needed to improve world happiness by 5%:
  - Log GDP per Capita : 6%
  - Social Support : 8%
  - Healthy life expectancy at birth: : 7%
  - All other features : undetermined (will be explained later as a part of limitations)

Limitations

- Regressor outputs limited by the upper and lower bound of the given training data
  - Acceptable for our purpose because we want to find realistic improvements the world need to make in order to achieve better happiness (i.e. what we need to improve to reach Finland levels)
- Cannot miss out on features
  - If you disregard the individual decision trees that rely on the features which are missing, it is possible to arrive at a prediction however the result may be highly inaccurate.

Questions?

Thank you :)