PSTAT100_Final_Project

June 17, 2023

1 Final Project: World Happiness Variations

Charles Delgado, Brandelyn Nie, Giovani Gutierrez

1.1 World Happiness Data Description

The data contains 2199 observations with 10 variables `Year`, `Life Ladder`, `Log GDP per capita`, `Social support`, `Healthy life expectancy at birth`, `Freedom to make life choices`, `Generosity`, `Perceptions of corruption`, `Positive affect`, and `Negative affect`, for 165 unique countries. The years span from 2005 to 2022, but not every country has observations recorded for each year in this range. There are not any variables that are missing a large proportion of values, most are missing less than 5%.

The table below provides variable descriptions and units for each column in the dataframe:

Variable	Description	Data Type
Country name	Name of country	string
year	Year the observation was recorded	categorical
Life Ladder	Happiness score	numeric
Log GDP per capita	Log gross domestic product per capita	numeric
Social support	National average of the binary responses (either 0 or 1) to the GWP question "If you were in trouble, do you have relatives or friends you can count on to help you whenever you need them, or not?"	numeric
Healthy life expectancy at birth	Life expectany in years	numeric

Variable	Description	Data Type
Freedom to make life decisions	The national average of responses to the GWP question “Are you satisfied or dissatisfied with your freedom to choose what you do with your life?”	numeric
Generosity	The residual of regressing national average of response to the GWP question “Have you donated money to a charity in the past month?” on GDP per capita.	numeric
Perceptions of corruption	National average of the survey responses to two questions in the GWP: “Is corruption widespread throughout the government or not” and “Is corruption widespread within businesses or not?”	numeric
Positive affect	Average of three positive affect measures in GWP: laugh, enjoyment and doing interesting things, the measures were three GWP questions	numeric
Negative affect	The average of three negative affect measures in GWP. They are worry, sadness and anger, the measures were three GWP questions	numeric

`Life Ladder` is, according to the world happiness website, “Happiness score or subjective well-being … is the national average response to the question of life evaluations. The English wording of the question is ‘Please imagine a ladder, with steps numbered from 0 at the bottom to 10 at the top. The top of the ladder represents the best possible life for you and the bottom of the ladder represents the worst possible life for you. On which step of the ladder would you say you personally feel you stand at this time?’ This measure is also referred to as Cantril life ladder, or just life ladder in our analysis.”

1.2 Question of Interest

Which variables are driving variation in the data? Having quanitified variables relating to emotional feelings was a pretty interesting way to collect this data, so let’s investigate further on which variables contributed the most variation to this happiness data.

1.3 Exploratory Data Analysis (EDA) & Pre-Processing

[1]:
```python
# packages used
import numpy as np
import pandas as pd
import altair as alt
from statsmodels.multivariate.pca import PCA
import warnings
from sklearn.cluster import KMeans

# ignore warnings
warnings.filterwarnings("ignore")

# disable row limit for plotting
alt.data_transformers.disable_max_rows()

# load in the dataset
whr_data = pd.read_csv('data/whr-2023.csv')

# render plots for pdf
alt.renderers.enable('mimetype')
```

[1]:
```
RendererRegistry.enable('mimetype')
```

1.3.1 Missing Values

First, we look at thr proportion of missing values for each variable. Notice that these proportions are very low. Hence, we keep all these attributes.

[2]:
```python
# proportion of missingness
whr_data.isna().mean()
```

[2]:
```
Country name                         0.000000
year                                 0.000000
Life Ladder                          0.000000
Log GDP per capita                   0.009095
Social support                       0.005912
Healthy life expectancy at birth     0.024557
Freedom to make life choices         0.015007
Generosity                           0.033197
Perceptions of corruption            0.052751
Positive affect                      0.010914
Negative affect                      0.007276
```

```
dtype: float64
```

Exploring the data, there were certain countries that were missing values for an entire variable for each year, so these countries were dropped since there was no way to impute values for them.

These would be interesting countries to do further research on why they did not record these metrics for these countries. From this, we only lost 9 countries (65 observations), leaving us with 156 countries.

Moreover, not every single year was recorded for every country. To make our analyses more simple, we grouped by country and took the mean for each numeric attribute. Notice that `year` was coerced to a categorical type. In total, we are left with 156 countries/observations.

Finally, we create two indicator variables which will be useful in clustering later on. `Higher_Life_Ladder` indicates countries with a `Life Ladder` value greater than (or equal to) 5.0. `Higher_Social_support` indicates countries with a higher `Social_support` value. The rest of the variables have the same definitions as before.

[3]:
```python
# grouping to find which countries are missing all values in a column
df1 = whr_data.groupby('Country name', dropna = False).mean(numeric_only =␣
 ↪True).isna()

# the countries missing all values, 9 total
df1_drop = df1[df1.any(axis = 1)]

# converting to list to drop from data set
list_drop = df1_drop.index.values.tolist()

# dropping
whr_data2 = whr_data[~whr_data['Country name'].isin(list_drop)]

# checking what's left in the dataset
print('Number of observations dropped (before grouping): {}'.format(whr_data.
 ↪shape[0] - whr_data2.shape[0]))
print('Number of countries left: {}\n'.format(whr_data2['Country name'].
 ↪nunique()))

# coerce `year` to category
whr_data2['year'] = whr_data2['year'].astype('category')

# group by country & take mean
whr_grouped = whr_data2.groupby('Country name', as_index = False).
 ↪mean(numeric_only = True)

# for use in Clustering
whr_grouped['Higher_Life_Ladder'] = np.where(whr_grouped['Life Ladder'] >= 5.0,␣
 ↪True, False)  # ndicator variable for Life Ladder
whr_grouped['Higher_Social_support'] = np.where(whr_grouped['Social support']␣
 ↪>= .5, True, False) # indicator variable for Social Support
```

```python
# preview
whr_grouped.head()
```

```
Number of observations dropped (before grouping): 65
Number of countries left: 156
```

[3]:

```
  Country name  Life Ladder  Log GDP per capita  Social support
0  Afghanistan     3.346643            7.585615        0.484500  \
1      Albania     5.047933            9.396933        0.715800
2      Algeria     5.377400            9.339800        0.814889
3       Angola     4.420250            8.985750        0.738250
4    Argentina     6.283588           10.030412        0.902412

   Healthy life expectancy at birth  Freedom to make life choices  Generosity
0                         52.533929                      0.498571    0.060000  \
1                         68.505333                      0.683133   -0.074733
2                         66.080000                      0.530875   -0.141000
3                         52.150000                      0.456250   -0.090500
4                         66.664706                      0.774529   -0.152471

   Perceptions of corruption  Positive affect  Negative affect
0                   0.842786         0.433286         0.364357  \
1                   0.869600         0.557267         0.293267
2                   0.697750         0.535667         0.267222
3                   0.866750         0.625750         0.351250
4                   0.838647         0.739000         0.287588

   Higher_Life_Ladder  Higher_Social_support
0               False                  False
1                True                   True
2                True                   True
3               False                   True
4                True                   True
```

1.4 Principal Components Analysis (PCA)

PCA is a useful multivariate analysis technique that can help with dimension reduction and help visualize high-dimensional data. In our case, this technique will help us determine which variables are driving the most variation in our data.

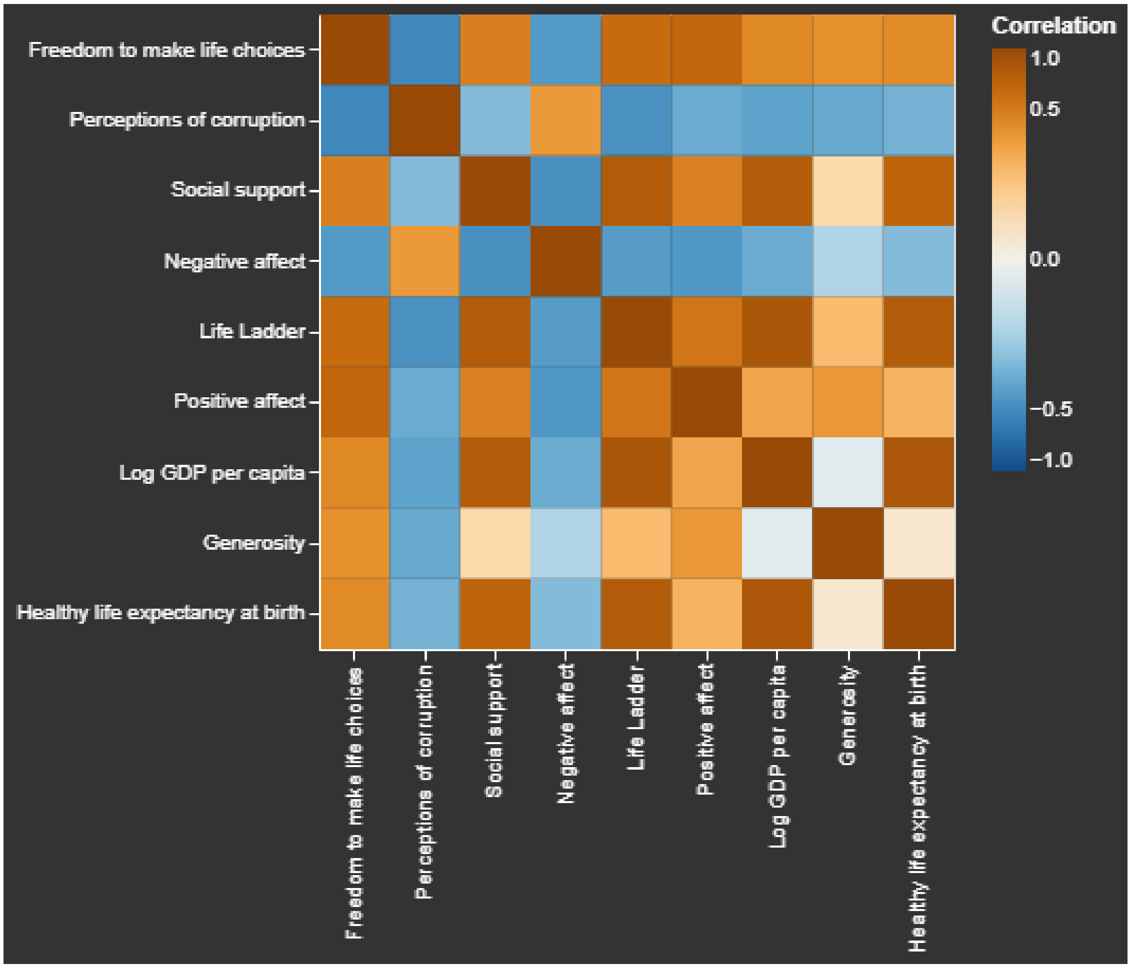
1.4.1 Correlation Matrix

PCA identifies variable combinations that capture covariation by decomposing the correlation matrix, so it will be helpful to look at the correlation matrix to see which variables vary together.

[4]:

```
# x_matrix creation
x_mx = whr_grouped.drop(columns = ['Country name', 'Higher_Life_Ladder',␣
↪'Higher_Social_support'])
corr_mx = x_mx.corr()
corr_mx.loc[:, 'Life Ladder'].sort_values()

# melt corr_mx
corr_mx_long = corr_mx.reset_index().rename(
    columns = {'index': 'row'}
).melt(
    id_vars = 'row',
    var_name = 'col',
    value_name = 'Correlation'
)

# construct plot
alt.Chart(corr_mx_long).mark_rect().encode(
    x = alt.X('col', title = '', sort = {'field': 'Correlation', 'order':␣
↪'ascending'}),
    y = alt.Y('row', title = '', sort = {'field': 'Correlation', 'order':␣
↪'ascending'}),
    color = alt.Color('Correlation',
                      scale = alt.Scale(scheme = 'blueorange', # diverging␣
↪gradient
                                        domain = (-1, 1), # ensure white = 0
                                        type = 'sqrt'), # adjust gradient scale
                     legend = alt.Legend(tickCount = 5)) # add ticks to␣
↪colorbar at 0.5 for reference
).properties(width = 300, height = 300)
```

[4]:



Looking at the correlation matrix, there are a lot of dark oranges and blues showing strong positive and strong negative correlations respectively. An example of positive correlation is `Positive affect` and `Freedom to make life choices`, and an example of negative is `Perceptions of corruption` and `Freedom to make life choices`.

Interestingly, `Log GDP per capita` and `Generosity` are the only variables that are not correlated in some manner.

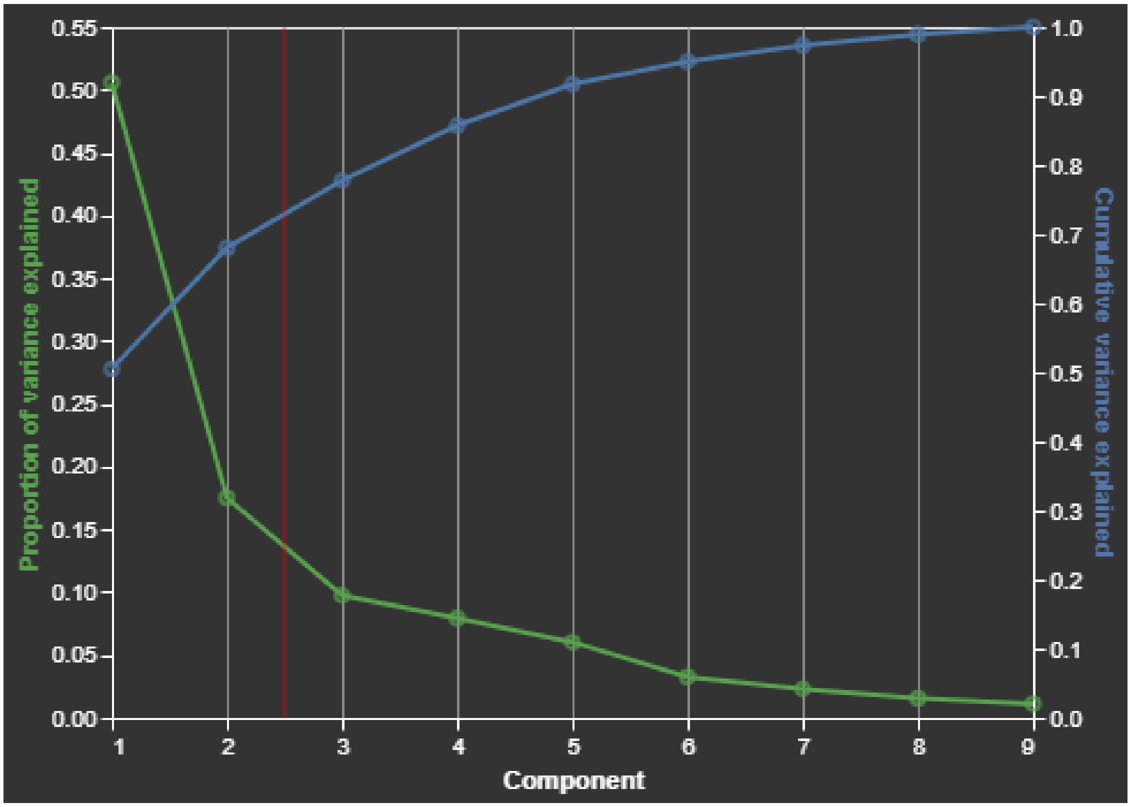
1.4.2 Compute the PC's and Variance Ratios

Here, we compute the principal components and their respective variance ratios. With these, we can examine the proportion of variatio explained for each PC and the cumulative variance exaplained. This will help us determine the amount of PCs to use in our analysis. To more easily examine these values, we can graph these values.

[5]:
```
#------ compute components
pca = PCA(x_mx, normalize = False, standardize = True)

# inspect loadings
```

```python
pca.loadings

# compute variance ratios
var_ratios = pca.eigenvals/pca.eigenvals.sum()

# store proportion of variance explained as a dataframe
pca_var_explained = pd.DataFrame({
    'Component': np.arange(1, 10),
    'Proportion of variance explained': var_ratios})

# add cumulative sum
pca_var_explained['Cumulative variance explained'] = var_ratios.cumsum()

#------ graphing portion of cell

# encode component axis only as base layer
base = alt.Chart(pca_var_explained).encode(
    x = 'Component')

# make a base layer for the proportion of variance explained
prop_var_base = base.encode(
    y = alt.Y('Proportion of variance explained',
              axis = alt.Axis(titleColor = '#57A44C'))
)

# make a base layer for the cumulative variance explained
cum_var_base = base.encode(
    y = alt.Y('Cumulative variance explained', axis = alt.Axis(titleColor =␣
 ↪'#5276A7'))
)

# add points and lines to each base layer
line = alt.Chart(pd.DataFrame({'Component': [2.5]})).mark_rule(opacity = 0.3,␣
 ↪color = 'red').encode(x = 'Component')
prop_var = prop_var_base.mark_line(stroke = '#57A44C') + prop_var_base.
 ↪mark_point(color = '#57A44C') + line
cum_var = cum_var_base.mark_line() + cum_var_base.mark_point() + line

# layer the layers
var_explained_plot = alt.layer(prop_var, cum_var).resolve_scale(y =␣
 ↪'independent')

# display
var_explained_plot
```

[5]:



The graphic above shows how much proportion of variance each component covers, as well as the cumulative amount of proportion of variance covered by multiple PC's. Find that 2 PC's covers about 60% of the total variation, though PC1 itself covers about 50% of the total variation.

The next step is to examine the loadings to understand just *which* variables the components combine with significant weight. The loadings are the *weights* with which the variables are combined to form the principal components.

[6]:
```
# subset loadings
loading_df = pca.loadings.iloc[:, 0:2]

# rename columns
loading_df = loading_df.rename(columns = dict(zip(loading_df.columns, ['PC' +␣
↪str(i) for i in range(1, 3)])))

# print
loading_df
```

[6]:

	PC1	PC2
Life Ladder	0.437373	-0.125421
Log GDP per capita	0.387745	-0.359136
Social support	0.392639	-0.225781
Healthy life expectancy at birth	0.361079	-0.375832

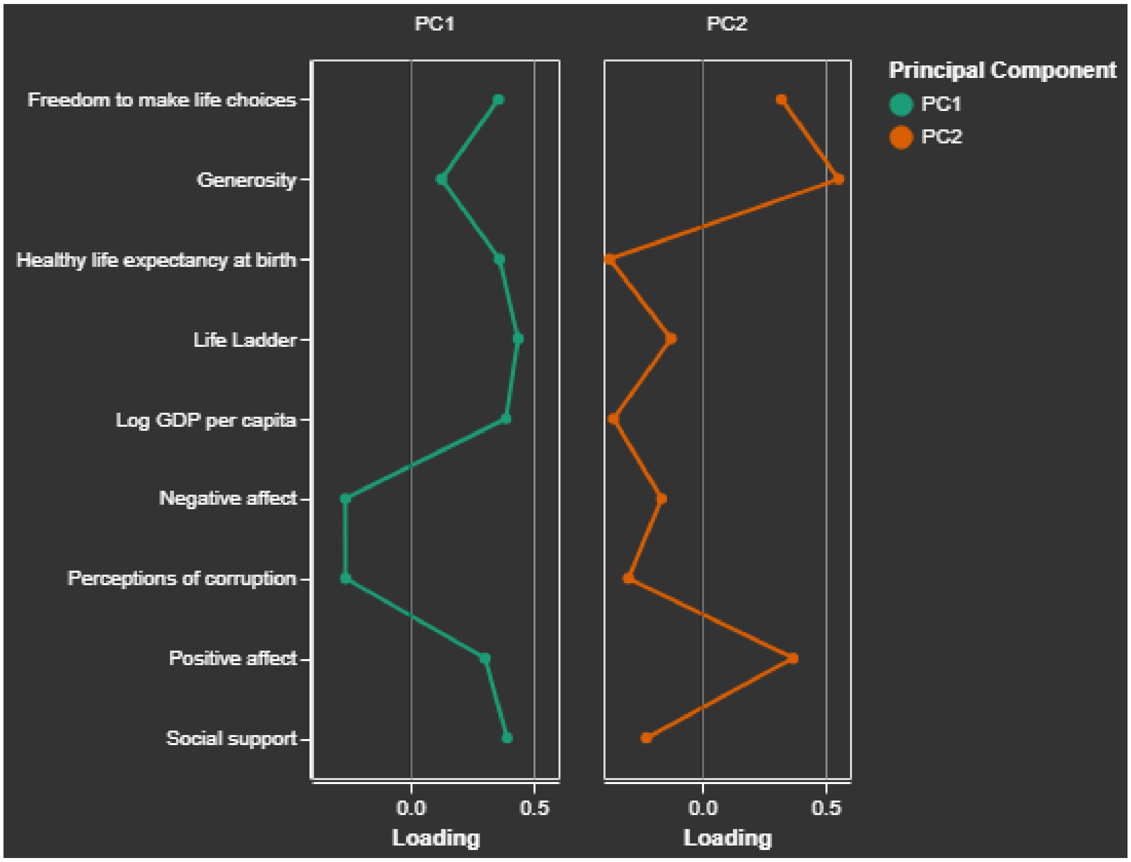
```
Freedom to make life choices      0.357443  0.321550
Generosity                        0.126142  0.554023
Perceptions of corruption        -0.262515 -0.299063
Positive affect                   0.303109  0.369302
Negative affect                  -0.263342 -0.164027
```

1.4.3 Visualize the Loadings

By visualizing the loadings, the variables that are the most influential for each component can be found, and also which variables seem to drive total variation in the data.

[7]:
```
# melt from wide to long
loading_plot_df = loading_df.reset_index().melt(
    id_vars = 'index',
    var_name = 'Principal Component',
    value_name = 'Loading'
).rename(columns = {'index': 'Variable'})

# add a column of zeros to encode for x = 0 line to plot
loading_plot_df['zero'] = np.repeat(0, len(loading_plot_df))

# create base layer
base = alt.Chart(loading_plot_df)

# create lines + points for loadings
loadings = base.mark_line(point = True).encode(
    y = alt.X('Variable', title = ''),
    x = 'Loading',
    color = alt.Color('Principal Component', scale = alt.Scale(scheme =␣
 ↪'dark2'))
)

# create line at zero
rule = base.mark_rule().encode(x = alt.X('zero', title = 'Loading'), size = alt.
 ↪value(0.05))

# layer
loading_plot = (loadings + rule).properties(width = 120, height = 350)

# show
loading_plot.facet(column = alt.Column('Principal Component', title = ''))
```

[7]:



Interpreting the principal components, for PC1 the variables with negative loadings are `Negative affect` and `Perceptions of corruption`. The variables with positive loadings are `Life Ladder` and `Social support`. These variables seem to be a representation of postive and negative emotions. If there is a higher value for PC1, then they have higher than average positive feelings, and if PC1 is small then they have higher than average negative feelings.

For PC2, the negative variables are `Health life expectancy at birth` and `Log GDP per capita`. The positive loading variables is `Generosity`. These varibles seems to represent ecominical factors. If there is a higher value for PC2, then they have higher than average spending, and if PC2 is small then they have higher than average GDP and life expectancy.

Recall that that first 2 PCs explain about 70% of the total variation in the data. Of this 70%, PC1 explains about 60% of this variation. Hence, moving forward, we will focus on the PC1. Below, we find that the two most influential variables for PC1 are `Life Ladder` and `Social support`. This can also be seen in the graphic above.

[8]:
```
# sorting the loadings
sort_loading = loading_df['PC1'].abs().sort_values(ascending=False)

# find most influential variable for PC1
pc1_most_influential_variable = loading_df['PC1'].abs().idxmax()
pc1_2ndmost_influential_variable = sort_loading.index[1]
```

```
# find respective loadings
pc1_most_influential_variable_loading = (loading_df['PC1']).
 ↪loc[pc1_most_influential_variable]
pc1_2ndmost_influential_loading = (loading_df['PC1']).
 ↪loc[pc1_2ndmost_influential_variable]

# print
print('Most influential variable for PC1: {}'.
 ↪format(pc1_most_influential_variable))
print('Second most influential variable for PC1: {}'.
 ↪format(pc1_2ndmost_influential_variable))
```

```
Most influential variable for PC1: Life Ladder
Second most influential variable for PC1: Social support
```

Since we have chosen to go with 2 PCs (with a focus on PC1), we subset the dataframe containing the scores. Hence, we are only left with the scores for the first two PCs. Furthermore, we concatenate the variables from our original dataset to these scores to create a new dataframe.

[9]:
```
# subset scores
score_df = pca.scores.iloc[:, 0:2]

# rename columns
score_df = score_df.rename(
    columns = dict(zip(score_df.columns, ['PC' + str(i) for i in range(1, 3)]))
)

# add original variables to dataframe
score_df[whr_grouped.columns.values.tolist()] = whr_grouped[whr_grouped.columns.
 ↪values.tolist()]

# print
score_df.head()
```

[9]:
```
        PC1       PC2 Country name  Life Ladder  Log GDP per capita
0 -4.828629  0.231621  Afghanistan     3.346643            7.585615  \
1 -0.906973 -1.240222      Albania     5.047933            9.396933
2 -0.803451 -1.718915      Algeria     5.377400            9.339800
3 -2.736381 -0.754709       Angola     4.420250            8.985750
4  1.181542 -1.135789    Argentina     6.283588           10.030412

   Social support  Healthy life expectancy at birth
0        0.484500                         52.533929  \
1        0.715800                         68.505333
2        0.814889                         66.080000
```

```
3        0.738250                          52.150000
4        0.902412                          66.664706

   Freedom to make life choices  Generosity  Perceptions of corruption
0                      0.498571    0.060000                   0.842786  \
1                      0.683133   -0.074733                   0.869600
2                      0.530875   -0.141000                   0.697750
3                      0.456250   -0.090500                   0.866750
4                      0.774529   -0.152471                   0.838647

   Positive affect  Negative affect  Higher_Life_Ladder  Higher_Social_support
0         0.433286         0.364357               False                  False
1         0.557267         0.293267                True                   True
2         0.535667         0.267222                True                   True
3         0.625750         0.351250               False                   True
4         0.739000         0.287588                True                   True
```

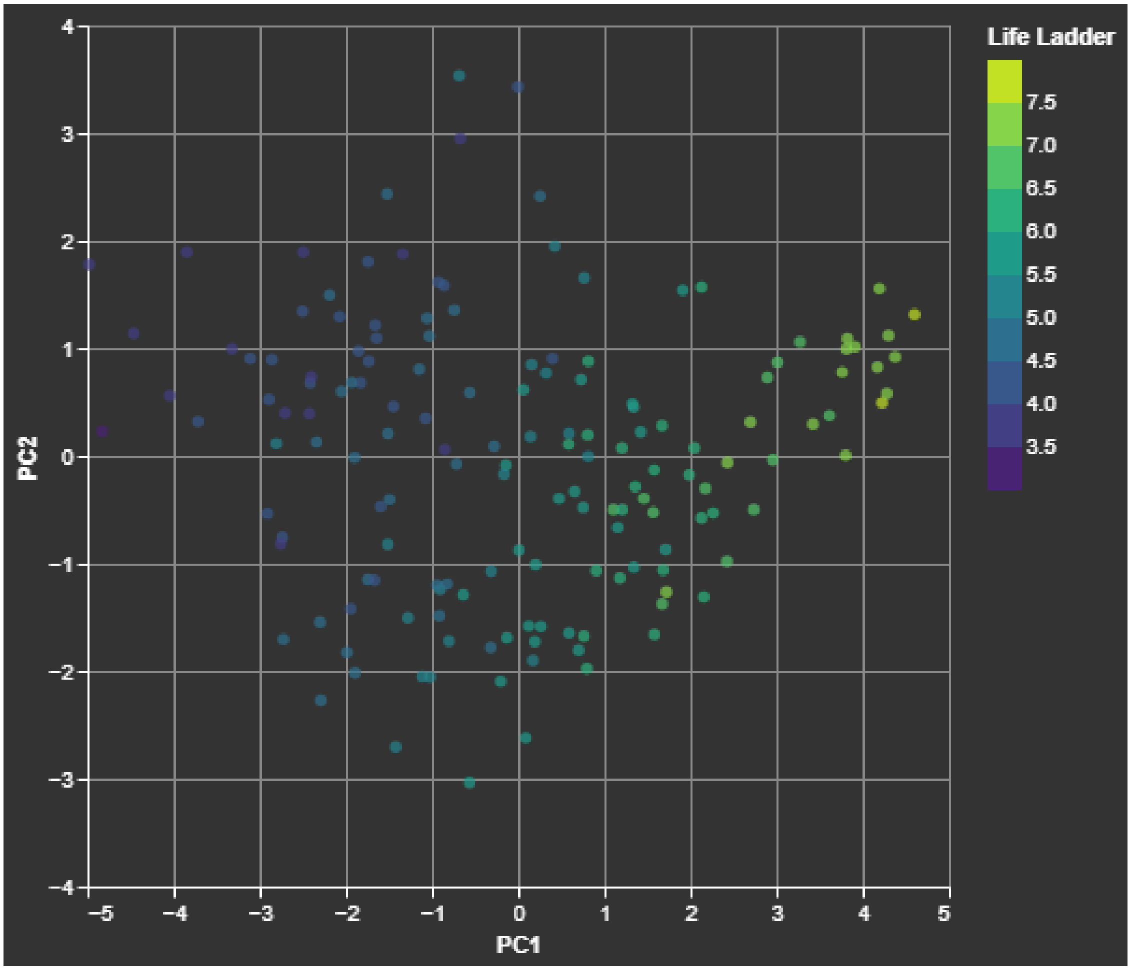
1.4.4 EDA with PCA

Using our two PCs, we can now use visualizations to look for any patterns or trends in the data. Since we noted that `Life Ladder` and `Social support` were the most influential variables for PC1, we color the plots by each variable.

This first plot is colored by `Life Ladder`. Notice that higher values of `PC1` correspond to observations/points that have a higher value of `Life Ladder`. On the other hand, lower values of PC1 indicate a lower value of `Life Ladder`. This is definielty in line with our previous interpretation of PC1.

[10]:
```
alt.Chart(score_df).mark_circle().encode(
    x = alt.X('PC1', title = 'PC1'),
    y = alt.Y('PC2', title = 'PC2'),
    color = alt.Color('Life Ladder',
                      bin = alt.Bin(maxbins = 10),
                      scale = alt.Scale(scheme = 'viridis'),
                      title = 'Life Ladder')
).properties(width = 400, height = 400)
```

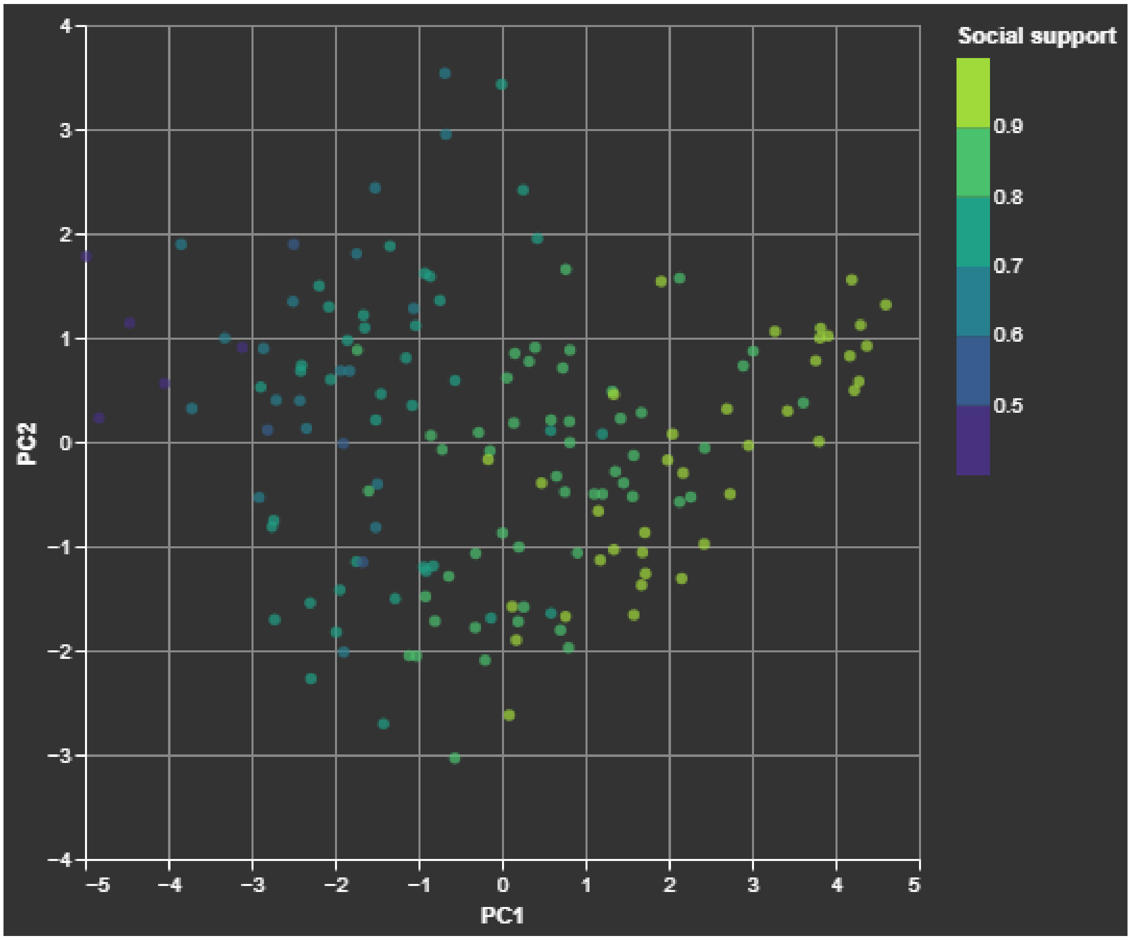
[10]:



This second plot is colored by `Social support`. Notice that higher values of `PC1` correspond to observations/points that have a higher value of `Social support`. On the other hand, lower values of PC1 indicate a lower value of `Social support`. Again, this is definielty in line with our previous interpretation of PC1.

[11]:
```
alt.Chart(score_df).mark_circle().encode(
    x = alt.X('PC1', title = 'PC1'),
    y = alt.Y('PC2', title = 'PC2'),
    color = alt.Color('Social support',
                      bin = alt.Bin(maxbins = 8),
                      scale = alt.Scale(scheme = 'viridis'),
                      title = 'Social support')
).properties(width = 400, height = 400)
```

[11]:



1.5 K-Means Clustering with PCA

From our analysis, we can infer that there are two general groups in which these observations (in terms of the two PCs) are apart of. Using K-Means clustering, we can further identify these groups and discover any more patterns among our data.

Since we have infered about two general groups, we will use 2 clusters:

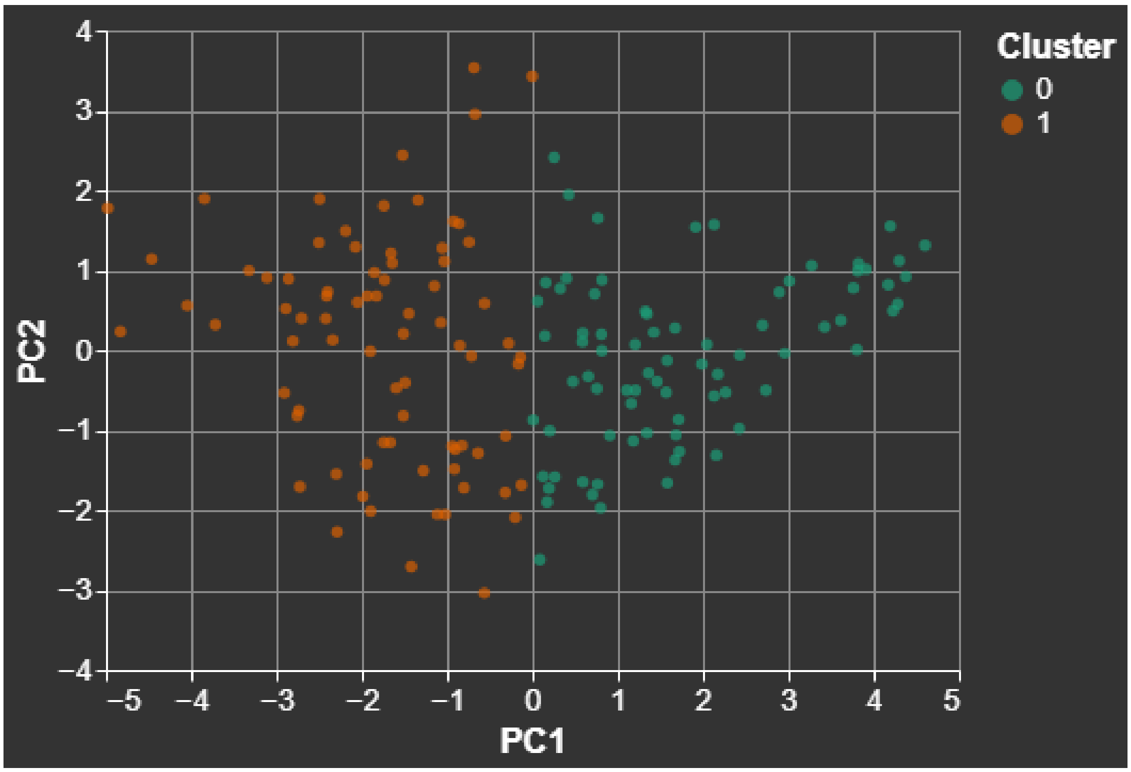
[12]:
```
clust = KMeans(n_clusters = 2, random_state = 123)
clust.fit(pca.scores.iloc[:, 0:2])
clust_labels = clust.predict(pca.scores.iloc[:, 0:2])
```

Now, we can visualize the clustered PCs:

[13]:
```
plot_df = pca.scores.iloc[:, 0:2].copy()
plot_df['cluster'] = clust_labels

alt.Chart(plot_df).mark_circle(opacity = 0.7).encode(
    x = alt.X('comp_0:Q', title = 'PC1'),
```

```
    y = alt.Y('comp_1:Q', title = 'PC2'),
    color = alt.Color('cluster:N', title = 'Cluster',
                      scale = alt.Scale(scheme = 'dark2'))
).configure_axis(
    labelFontSize = 14,
    titleFontSize = 16
).configure_legend(
    labelFontSize = 14,
    titleFontSize = 16
)
```

[13]:



We can now explore what any distinctions between these two clusters/groups. Since we are focusing on PC1, we will look at the cluster composition by the two variables which most influence PC1.

1.5.1 Cluster Composition by `Life Ladder`

First, we look at the composition of these clusters by `Life Ladder`.

[14]:
```
label_df = pd.DataFrame({'cluster': clust_labels}, index = x_mx.index)

pd.merge(whr_grouped, label_df, left_index = True, right_index = True
         ).groupby(['Higher_Life_Ladder', 'cluster']).size().reset_index(
         ).pivot(columns = 'Higher_Life_Ladder', index = 'cluster')
```

[14]:
```
                           0
Higher_Life_Ladder False True
cluster
0                      2   76
1                     62   16
```

Notice that each cluster (mostly) contains differing observations according to `Life Ladder`. Cluster '0' is mostly made up of those observations with higher values of `Life Ladder`. On the other hand, Cluster '1' is mostly made up of those observations with lower values of `Life Ladder`. Hence, we can say that one cluster is mostly representative of the "happier" countries and the other cluster is mostly representative of the "less happier" countries.

We can also identify those 2 countries in cluster '0' which had lower `Life Ladder` scores:

[15]:
```
countries_labeled = pd.merge(whr_grouped, label_df, left_index = True,␣
 ↪right_index = True)

countries_labeled[(countries_labeled.cluster == 0) & (countries_labeled.
 ↪Higher_Life_Ladder == False)]
```

[15]:
```
    Country name  Life Ladder  Log GDP per capita  Social support  \
76          Laos     4.995400            8.671900        0.723900
129    Sri Lanka     4.326067            9.299667        0.828667

     Healthy life expectancy at birth  Freedom to make life choices  \
76                          58.722000                      0.900667
129                         65.293333                      0.809467

     Generosity  Perceptions of corruption  Positive affect  Negative affect  \
76       0.2464                   0.639333         0.731400         0.274500
129      0.1468                   0.817333         0.722467         0.227333

     Higher_Life_Ladder  Higher_Social_support  cluster
76                False                   True        0
129               False                   True        0
```

1.5.2 Cluster Composition by `Social support`

Next, we look at the composition of these clusters by `Social support`.

[16]:
```
label_df = pd.DataFrame({'cluster': clust_labels}, index = x_mx.index)

pd.merge(whr_grouped, label_df, left_index = True, right_index = True
        ).groupby(['Higher_Social_support', 'cluster']).size().reset_index(
        ).pivot(columns = 'Higher_Social_support', index = 'cluster')
```

[16]:
```
                              0
Higher_Social_support False True
```

```
cluster
0                        NaN  78.0
1                        5.0  73.0
```

Notice that both clusters are mostly made up of those observations with higher values of `Social support`. Here, there does not seem to be a distinction between clusters/groups based on `Social support`.

However, we may identify those 5 countries that we could consider "outliers" in this setting. Notice that these countries also all have lower values of `Life Ladder`:

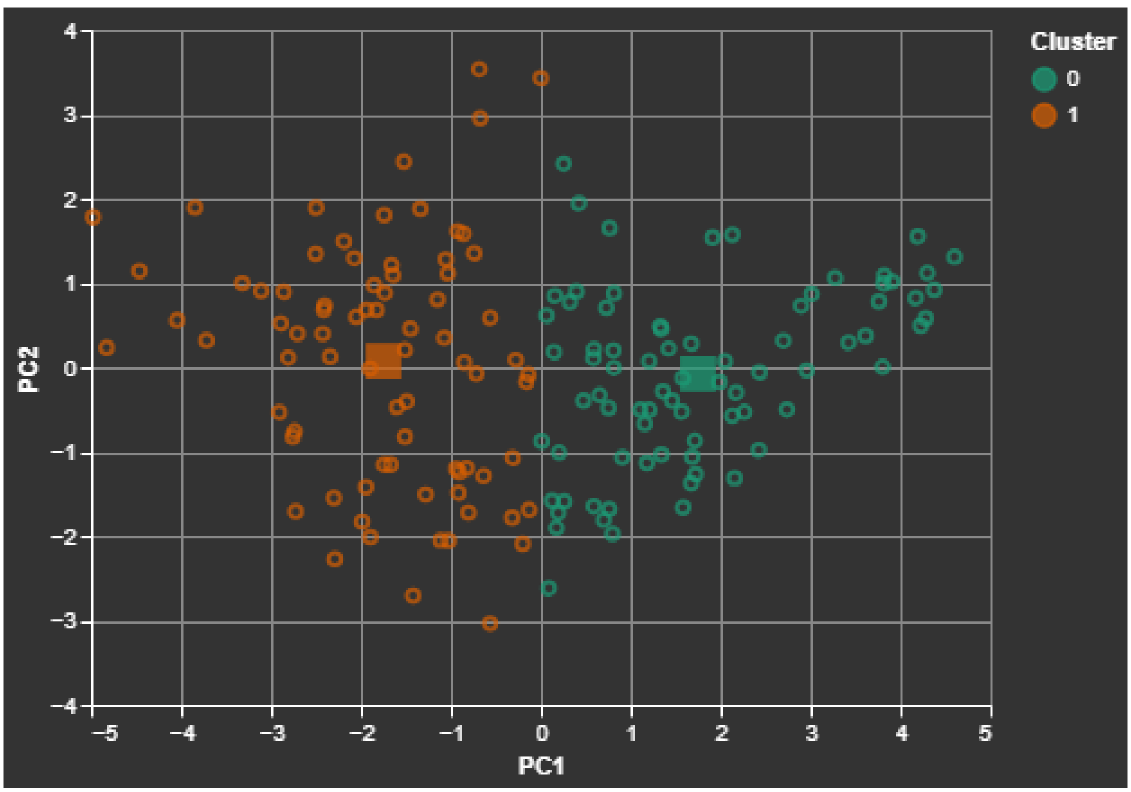
[17]:
```
countries_labeled[(countries_labeled.cluster == 1) & (countries_labeled.
 ↪Higher_Social_support == False)]
```

[17]:
```
                 Country name  Life Ladder  Log GDP per capita  \
0                 Afghanistan     3.346643            7.585615
14                      Benin     4.091786            7.978071
22                    Burundi     3.548200            6.682200
26   Central African Republic     3.515000            6.894800
140                      Togo     3.661000            7.532909

     Social support  Healthy life expectancy at birth  \
0          0.484500                         52.533929
14         0.466286                         54.417143
22         0.417800                         52.008000
26         0.402400                         43.374000
140        0.480545                         54.405455

     Freedom to make life choices  Generosity  Perceptions of corruption  \
0                        0.498571    0.060000                   0.842786
14                       0.738143   -0.052857                   0.757571
22                       0.450800   -0.034600                   0.732400
26                       0.680400    0.030600                   0.842000
140                      0.629545   -0.037364                   0.791636

     Positive affect  Negative affect  Higher_Life_Ladder  \
0           0.433286         0.364357               False
14          0.582500         0.352214               False
22          0.570400         0.244200               False
26          0.540000         0.391400               False
140         0.566909         0.419182               False

     Higher_Social_support  cluster
0                    False        1
14                   False        1
22                   False        1
26                   False        1
140                  False        1
```

We could also plot the actual centroids, as done below. Further analysis could include identifying those observations farthest or closest from the centroids.

[18]:
```
plot_df = pca.scores.iloc[:, 0:2].copy()
plot_df['cluster'] = clust_labels

scatter = alt.Chart(plot_df).mark_point(opacity = 0.7).encode(
    x = alt.X('comp_0:Q', title = 'PC1'),
    y = alt.Y('comp_1:Q', title = 'PC2'),
    color = alt.Color('cluster:N', title = 'Cluster')
)

centers = pd.concat([pd.DataFrame(clust.cluster_centers_), pd.
 ↪DataFrame({'cluster':[0, 1]})], axis = 1)
centers.columns.values[0:2] = ['x', 'y']

mark = alt.Chart(centers).mark_square(size = 250, opacity = 0.7).encode(
    x = 'x:Q',
    y = 'y:Q',
    color = alt.Color('cluster:N', scale = alt.Scale(scheme = 'dark2'))
)

scatter + mark
```

[18]:



2 Summary of Findings

This report investigated which variables drove the variation in the data. Initially, by creating the correlation matrix, many of the variables were found to be correlated with other ones. From this observation, Principal Component Analysis was a good next step. The two PC's used were an Emotional PC (PC1) and an Economical PC (PC2). These two PC's cover around 70% of the variation in the data, with PC1 covering around 60% alone. Since PC1 covered such a large percentage, the next step was to look further into the variables with highest weight within this PC, which were `Life Ladder` and `Social support`.

Furthermore, using K-Means clustering on the PCs, we found that the observations are "grouped" into two general groups: "happier" countries and "less happier/sad" countries. In terms of `Social support`, there were no defining distinctions between the clusters.

All in all, `Life Ladder` and `Social support` drive the most variation in the data, with `Life Ladder` being the defining variable, as one would intuitively think.