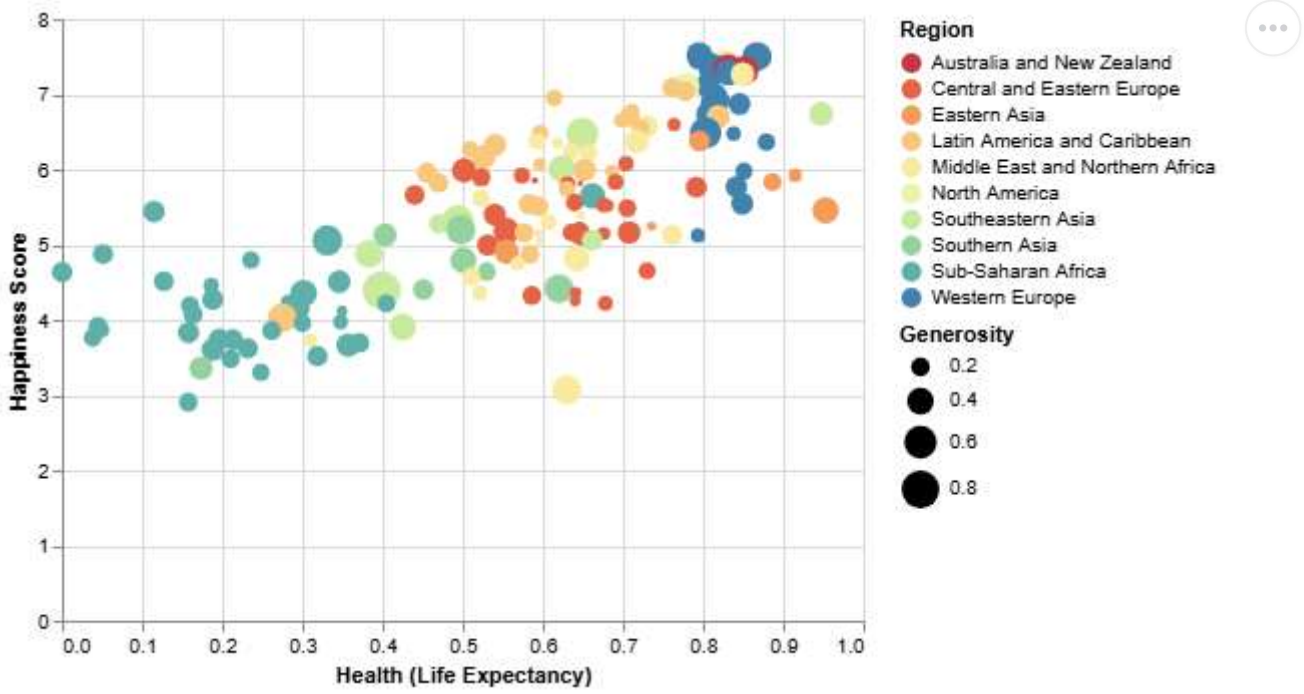
In [18]:
```
# Import our data processing library (note: you may have to install this!)
import pandas as pd
import altair as alt

# Let's use this to upload a sample dataset and show the start of the dataset
data= pd.read_csv("WHR_2016.csv")
data.head()
```

Out[18]:

	Country	Region	Happiness Rank	Happiness Score	Lower Confidence Interval	Upper Confidence Interval	Economy (GDP per Capita)	Family	Health (Life Expectancy)	Freedom	Trust (Government Corruption)	Generosity
0	Denmark	Western Europe	1	7.526	7.460	7.592	1.44178	1.16374	0.79504	0.57941	0.44453	0.36171
1	Switzerland	Western Europe	2	7.509	7.428	7.590	1.52733	1.14524	0.86303	0.58557	0.41203	0.28083
2	Iceland	Western Europe	3	7.501	7.333	7.669	1.42666	1.18326	0.86733	0.56624	0.14975	0.47678
3	Norway	Western Europe	4	7.498	7.421	7.575	1.57744	1.12690	0.79579	0.59609	0.35776	0.37895
4	Finland	Western Europe	5	7.413	7.351	7.475	1.40598	1.13464	0.81091	0.57104	0.41004	0.25492

In [19]:
```
selection = alt.selection(type='multi', fields=['Region'])

alt.Chart(data).mark_circle().encode(
    x = "Health (Life Expectancy)",
    y    = "Happiness Score",
    color=alt.Color('Region', scale=alt.Scale(scheme='spectral')),
    size="Generosity",
    tooltip=["Country", "Happiness Score"],
    opacity=alt.condition(selection,alt.value(1),alt.value(.2))
).add_selection(selection)
```

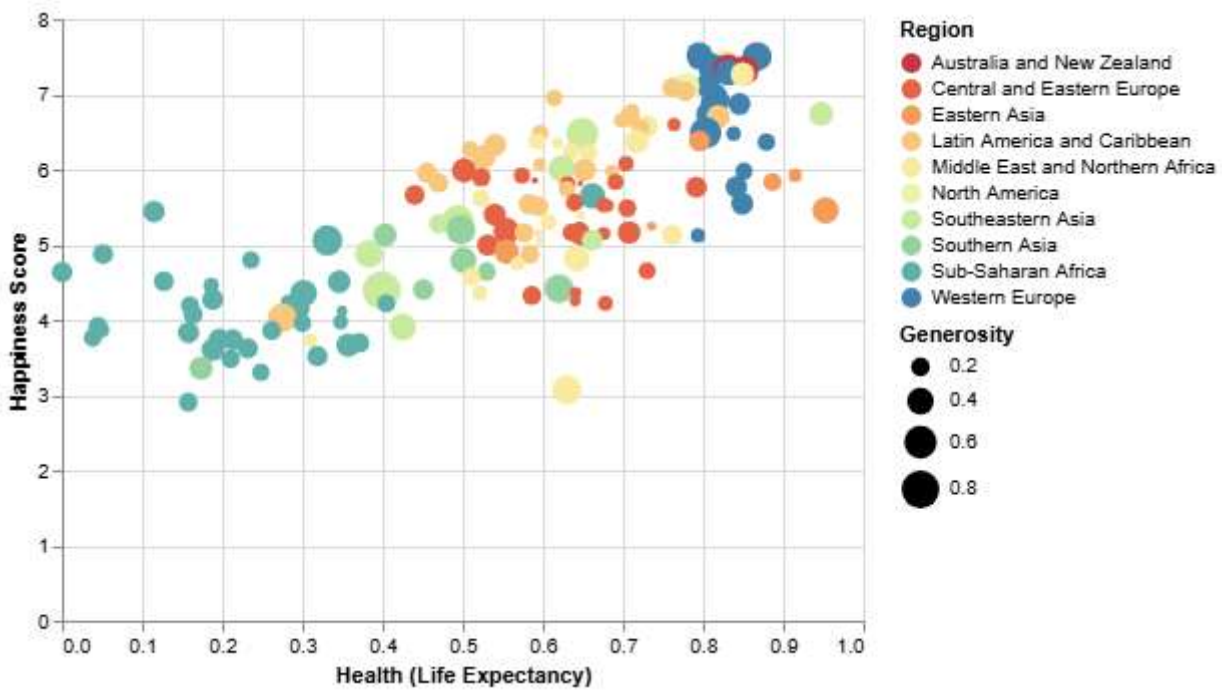
Out[19]:



In [21]:

```
selection = alt.selection(type='multi', fields=['Region'], on='mouseover', nearest=True)

alt.Chart(data).mark_circle().encode(
    x = "Health (Life Expectancy)",
    y    = "Happiness Score",
    color=alt.Color('Region', scale=alt.Scale(scheme='spectral')),
    size="Generosity",
    tooltip=["Country", "Happiness Score"],
    opacity=alt.condition(selection,alt.value(1),alt.value(.2))
).add_selection(selection)
```

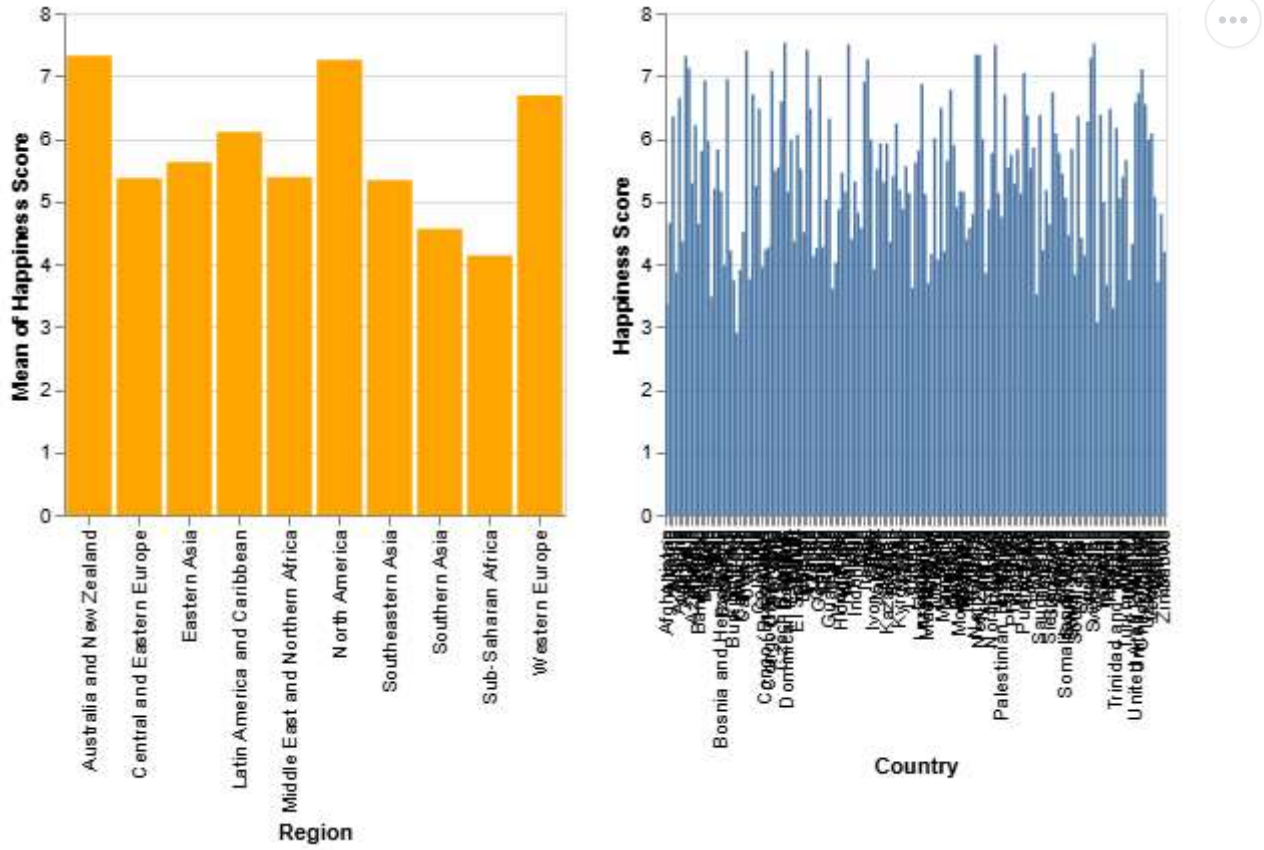
Out[21]:



In [22]:

```
# Let's implement filtering using dynamic queries.
selection = alt.selection(type="multi", fields=["Region"])

# Create a container for our two different views
base =  alt.Chart(data).properties(width=500, height=250)

# Let's specify our overview chart
overview = alt.Chart(data).mark_bar().encode(
    y = "mean(Happiness Score)",
    x = "Region",
    color=alt.condition(selection, alt.value("orange"), alt.value("lightgrey"))
).add_selection(selection).properties(height=250, width=250)

# Create a detail chart
detail = hist = base.mark_bar().encode(
    y = "Happiness Score",
    x = "Country"
).transform_filter(selection).properties(height=250, width=250)

overview | detail
```

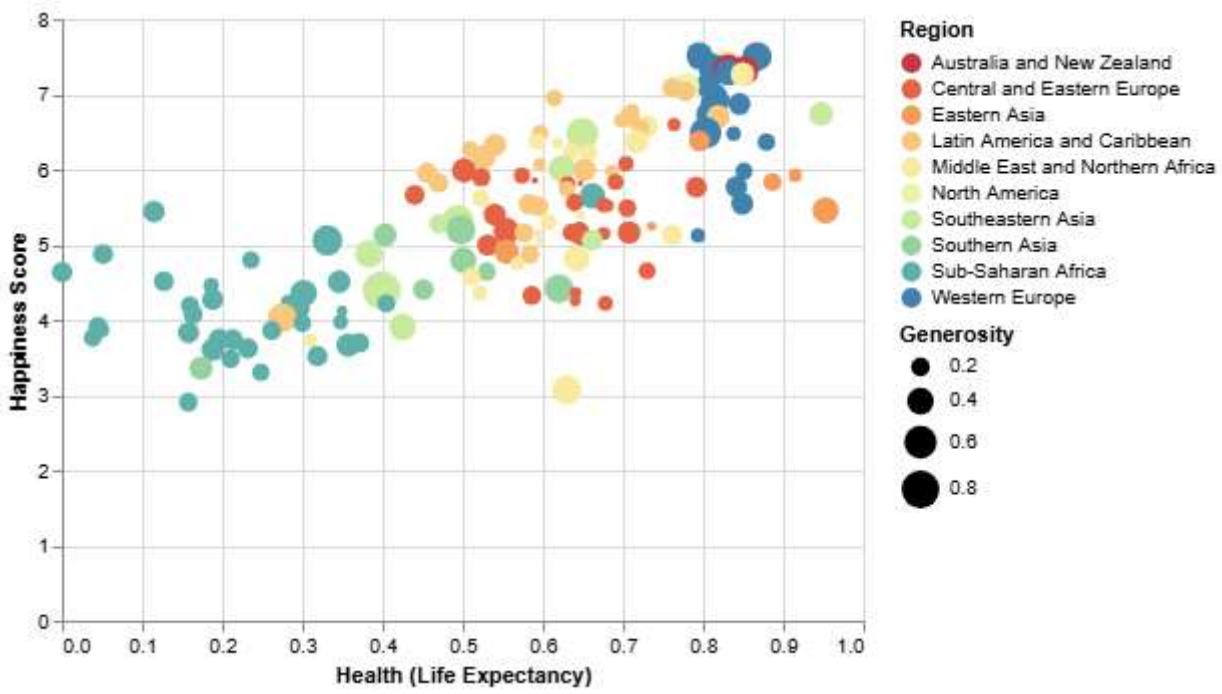
Out[22]:



In [23]:

```
# Bind our selection to the legend
selection = alt.selection(type='multi', fields=['Region'], bind='legend')

alt.Chart(data).mark_circle().encode(
    x = "Health (Life Expectancy)",
    y    = "Happiness Score",
    color=alt.Color('Region', scale=alt.Scale(scheme='spectral')),
    size="Generosity",
    tooltip=["Country", "Happiness Score"],
    opacity=alt.condition(selection,alt.value(1),alt.value(.2))
).add_selection(selection)
```

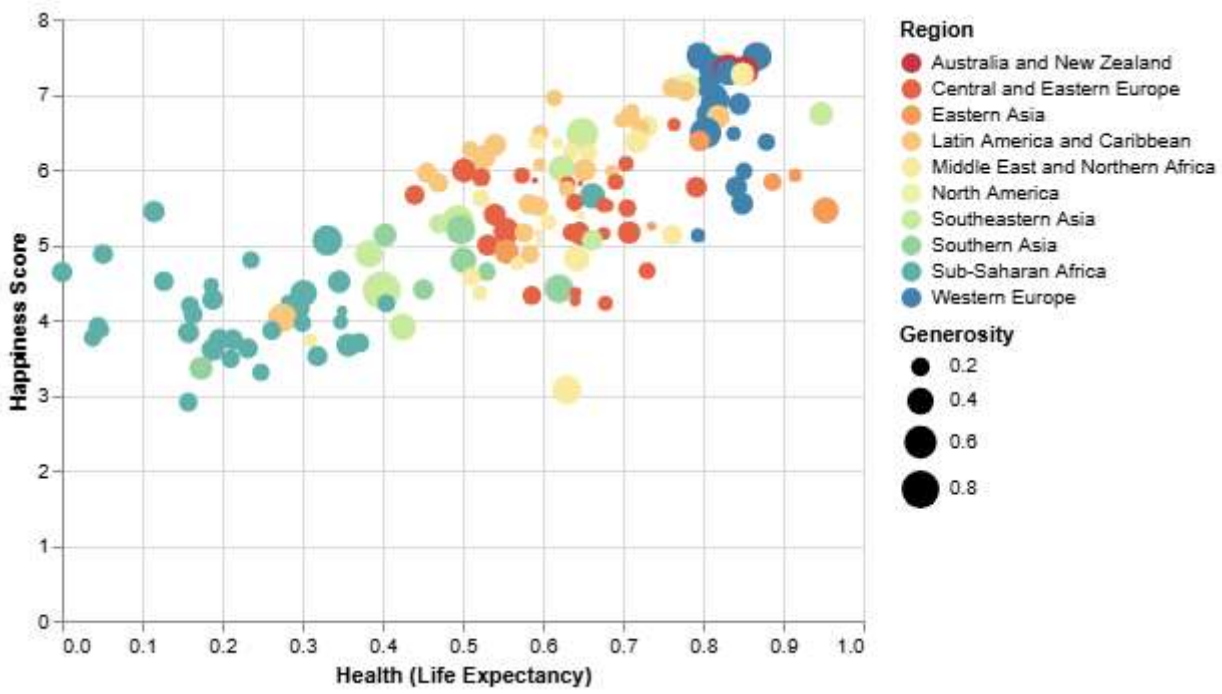
Out[23]:



In [24]:

```
# Let's implement filtering using dynamic queries.
dropdown = alt.binding_select (options=data["Region"].unique(), name="Select a region:")

# Create a new selection that uses my dynamic query widget
selection = alt.selection(type="single", fields=["Region"], bind=dropdown)

# Let's specify our chart
alt.Chart(data).mark_circle().encode(
    x = "Health (Life Expectancy)",
    y = "Happiness Score",
    color=alt.Color('Region', scale=alt.Scale(scheme='spectral')),
    size="Generosity",
    tooltip=["Country", "Happiness Score"],
    opacity=alt.condition(selection,alt.value(1),alt.value(.2))
).add_selection(selection)
```

Out[24]:

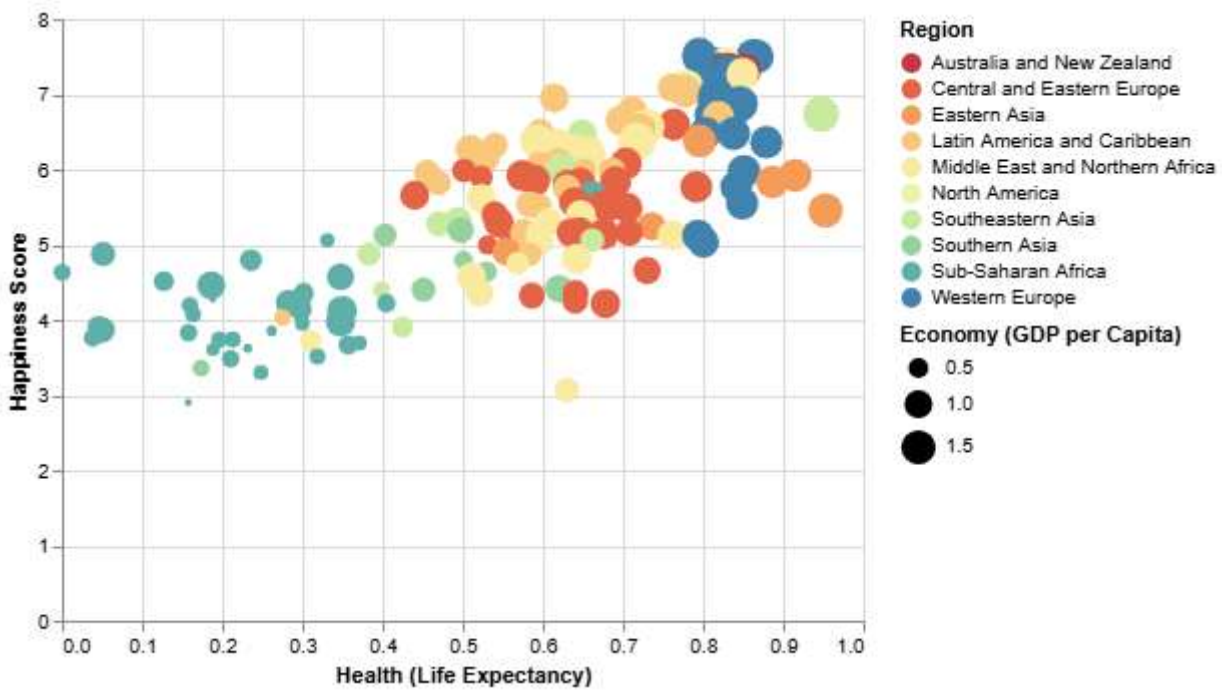


Select a region: Western Europe

In [25]:

```
# Let's implement filtering using dynamic queries.
##dropdown = alt.binding_select (options=data["Country"].unique(), name="Select a country:")
dropdown = alt.binding_select (options=['Japan','India','China'], name="Select a country:")

# Create a new selection that uses my dynamic query widget

selection = alt.selection(type="single", fields=["Country"], bind=dropdown)

# Let's specify our chart
alt.Chart(data).mark_circle().encode(
    x = "Health (Life Expectancy)",
    y = "Happiness Score",
    color=alt.Color('Region', scale=alt.Scale(scheme='spectral')),
    size="Economy (GDP per Capita)",
    tooltip=["Country", "Happiness Score","Economy (GDP per Capita)"],
    opacity=alt.condition(selection,alt.value(1),alt.value(.2))
).add_selection(selection)
```

Out[25]:

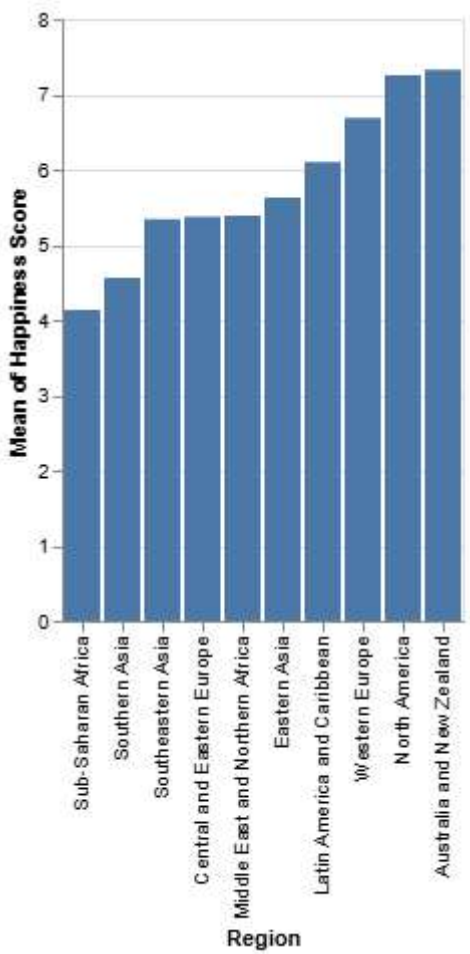


Select a country: Japan

In [9]:

```
# Let's specify our chart
alt.Chart(data).mark_bar().encode(
    y = "mean(Happiness Score)",
    x = alt.X(field='Region', type='nominal', sort=alt.EncodingSortField(field='Happiness Score', op='mean'))
)
```

Out[9]:

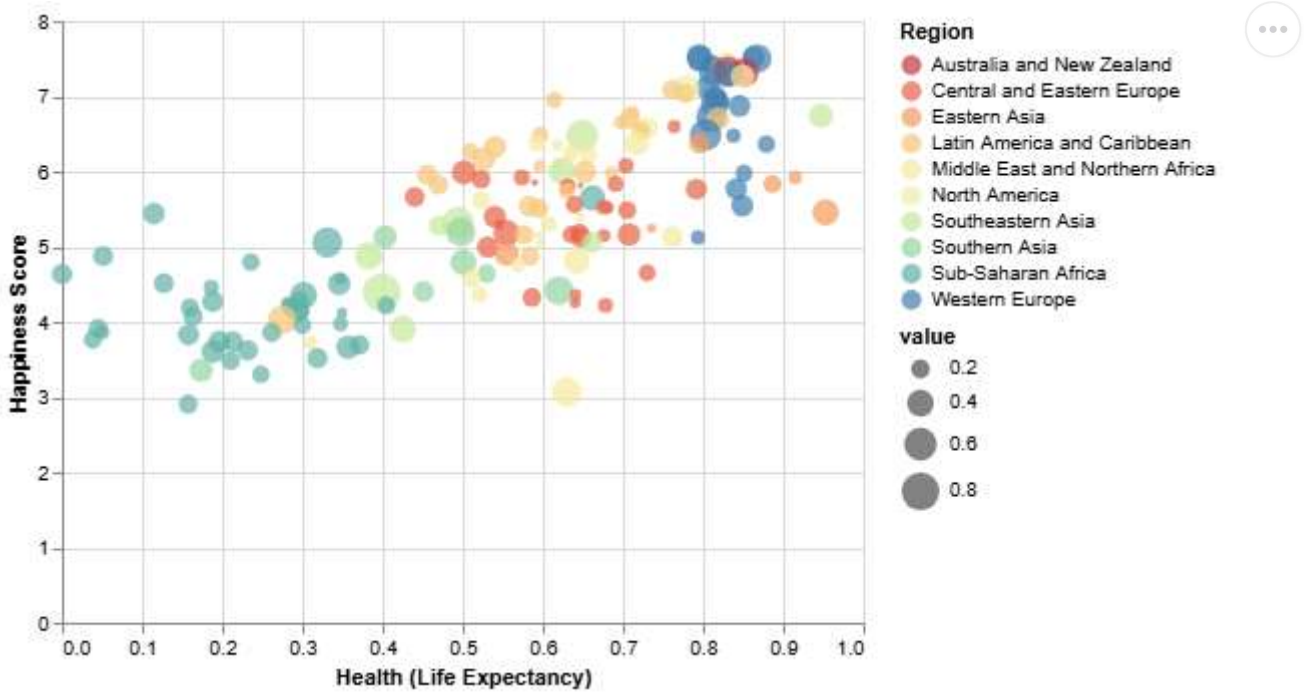


In [28]:

```
# Let's implement filtering using dynamic queries.
dropdown = alt.binding_select (options=["Generosity", "Family", "Freedom"], name="Select a size variable:")

# Create a new selection that uses my dynamic query widget
selection = alt.selection(type="single", fields=['column'], bind=dropdown, init={'column':'Generosity'})

# Let's specify our chart
alt.Chart(data).transform_fold(
    ["Generosity", "Family", "Freedom"],
    as_=['column', 'value']
).transform_filter(
    selection
).mark_circle().encode(
    x = "Health (Life Expectancy)",
```

```
    y = "Happiness Score",
    color=alt.Color('Region', scale=alt.Scale(scheme='spectral')),
    size="value:Q",
    tooltip=["Country", "Happiness Score"],
).add_selection(selection)
```

Out[28]:



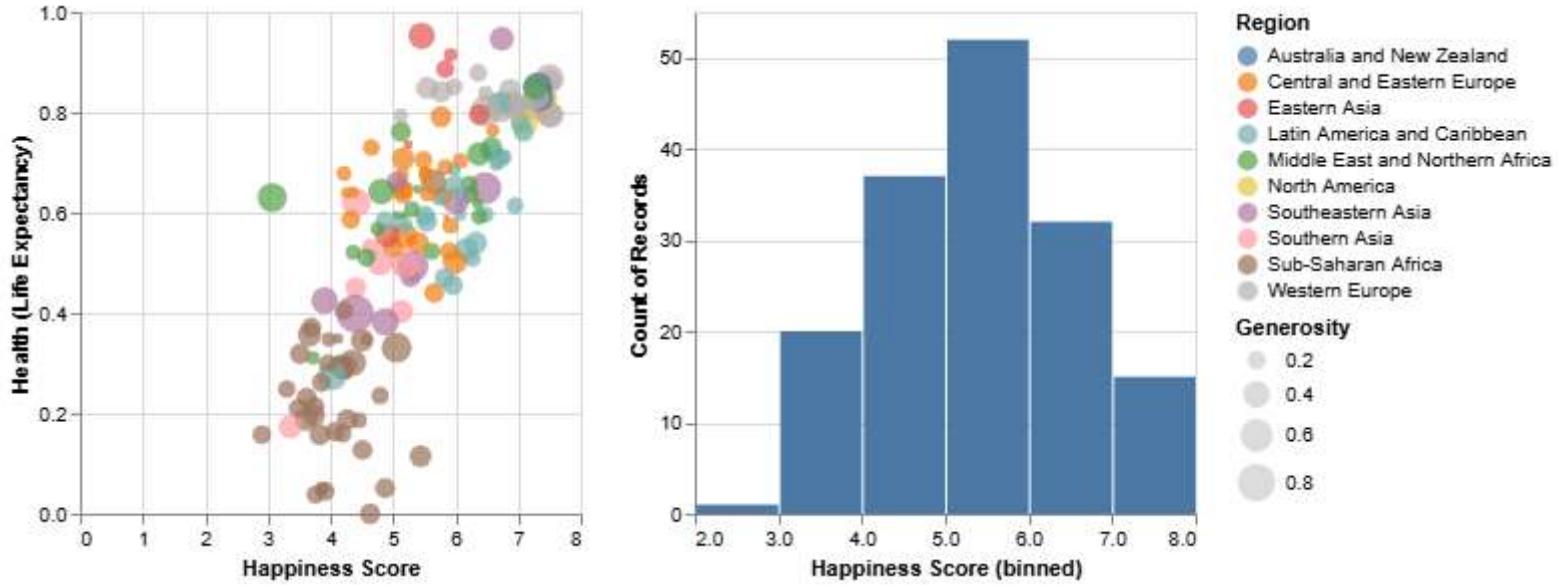
Select a size variable: Generosity

In [26]:

```
# Linked views
# Creating a selection:
selection = alt.selection(type="single", fields=["Region"])

# Create a container for our two different views
base =  alt.Chart(data).properties(width=250, height=250)

# Create our scatterplot
scatterplot = base.mark_circle().encode(
    x = 'Happiness Score',
    y = 'Health (Life Expectancy)',
    size = "Generosity",
    color = alt.condition(selection, "Region", alt.value('lightgray'))
).add_selection(selection)
```

```
# Create a histogram
hist = base.mark_bar().encode(
    x = alt.X("Happiness Score", bin=alt.Bin(maxbins=5)),
    y = "count()"
).transform_filter(selection)

# Connect our charts using the pipe operation
scatterplot | hist
```

Out[26]:

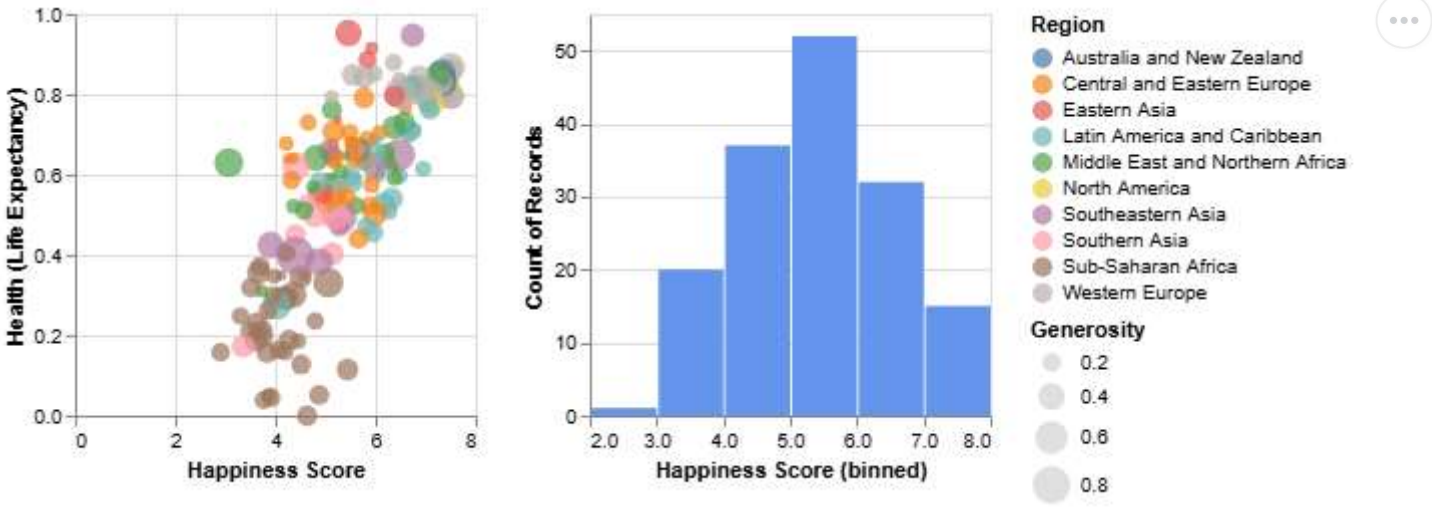


In [29]:

```
# This selection is going to be an interval selection
selection = alt.selection(type="interval", encodings=["x", "y"])

# Create our scatterplot
scatterplot = alt.Chart(data).mark_circle().encode(
    x = 'Happiness Score',
    y = 'Health (Life Expectancy)',
    size = "Generosity",
    color = alt.condition(selection, "Region", alt.value('lightgray'))
).properties(
    width = 200,
    height = 200
).add_selection(selection)

# Define our background chart
base = alt.Chart().mark_bar(color="cornflowerblue").encode(
    x = alt.X("Happiness Score", bin=alt.Bin(maxbins=5)),
```

```
    y = "count()"
).properties (
    width=200,
    height = 200
)

# Grey background to show the selection range in the scatterplot
background = base.encode(color=alt.value('lightgray')).add_selection(selection)

# Blue highlights to show the transformed (brushed) data
highlight = base.transform_filter(selection)

# Layer the two charts
layers = alt.layer(background, highlight, data = data)

scatterplot | layers
```

Out[29]:



In [5]:

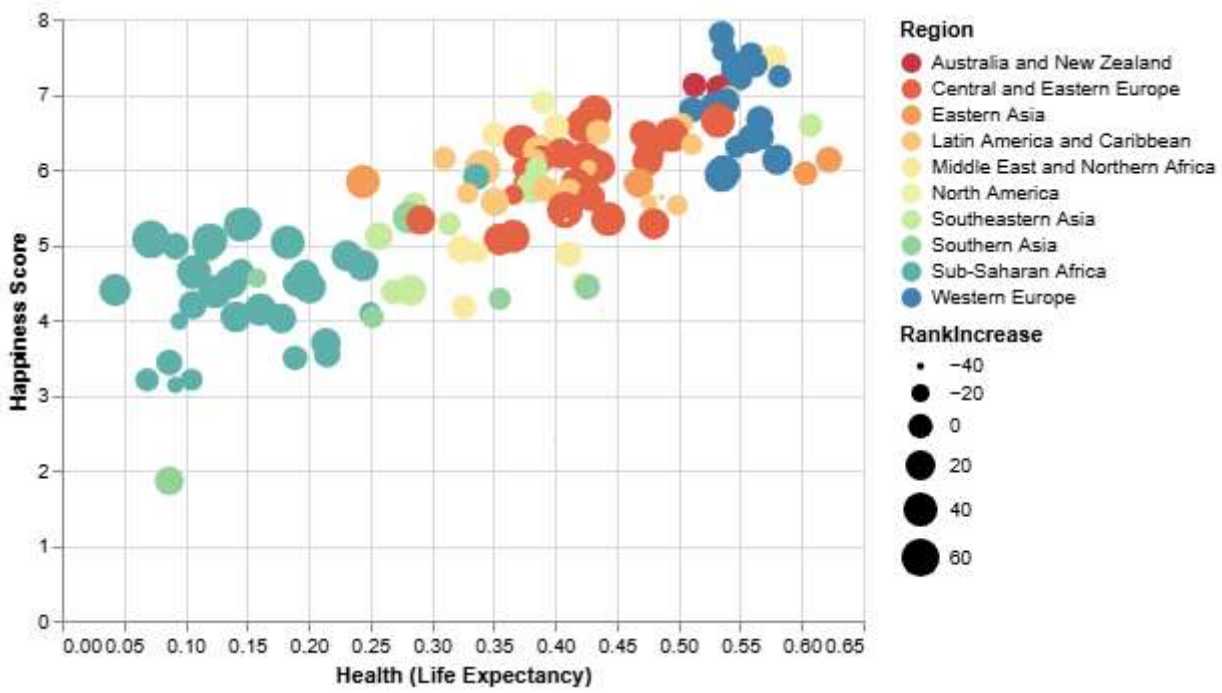
```
import pandas as pd
import altair as alt

# Let's use this to upload a sample dataset and show the start of the dataset
Historydata= pd.read_csv("WH2023compare.csv")
Historydata.head()
```

Out[5]:

	Country	Region	Happiness Score	Happiness Rank	Social_support	Health__Life_Expectancy_	Freedom_to_make_life_choices	Generosity	Perceptio
0	Finland	Western Europe	7.804	1	0.969	71.150	0.961	-0.019	
1	Denmark	Western Europe	7.586	2	0.954	71.250	0.934	0.134	
2	Iceland	Western Europe	7.530	3	0.983	72.050	0.936	0.211	
3	Israel	Middle East and Northern Africa	7.473	4	0.943	72.697	0.809	-0.023	
4	Netherlands	Western Europe	7.403	5	0.930	71.550	0.887	0.213	

In [30]:
```
# Let's implement filtering using dynamic queries.
##dropdown = alt.binding_select (options=data["Country"].unique(), name="Select a country:")
dropdown = alt.binding_select (options=['Japan','Canada','China','Germany','United States'], name="Select a country:")

# Create a new selection that uses my dynamic query widget

selection = alt.selection(type="single", fields=["Country"], bind=dropdown)

# Let's specify our chart
alt.Chart(Historydata).mark_circle().encode(
    x = "Health (Life Expectancy)",
    y = "Happiness Score",
    color=alt.Color('Region', scale=alt.Scale(scheme='spectral')),
    size="RankIncrease",
    tooltip=["Country", "Happiness Score","Economy GDP per Capita","Region","RankIncrease"],
    opacity=alt.condition(selection,alt.value(1),alt.value(.2))
).add_selection(selection)
```

Out[30]:



Select a country: Japan

In [ ]:

In [ ]: