Machine Learning

| Big Picture?

Prof. Dr. Mostafa Elhosseini

https://youtube.com/drmelhosseini

Big Picture

Agenda

- Linear Model
- Training Model
- Cost function
- Model Parameters
- Inference

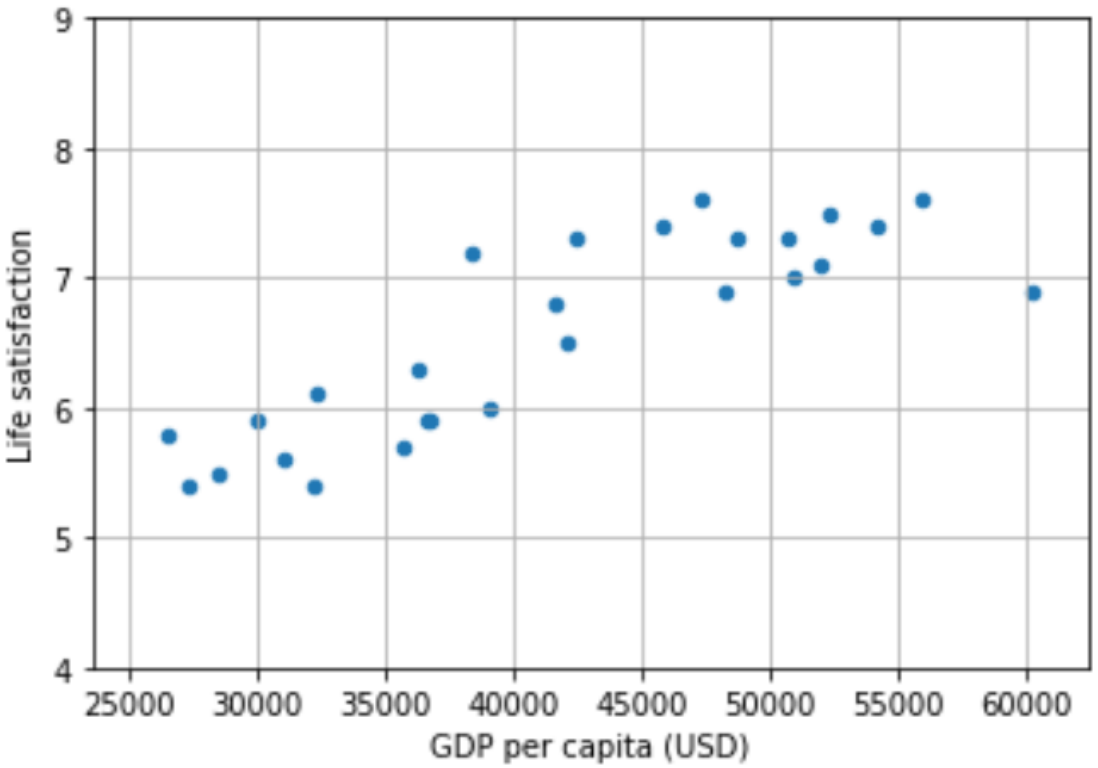
Big Picture

- Does money make people happy?
  - Download the Better Life Index data from the OECD's website https://www.oecdbetterlifeindex.org/
  - and World Bank stats about gross domestic product (GDP) per capita الناتج المحلي الإجمالي للفرد https://ourworldindata.org/.
  - Join the tables and sort by GDP per capita.

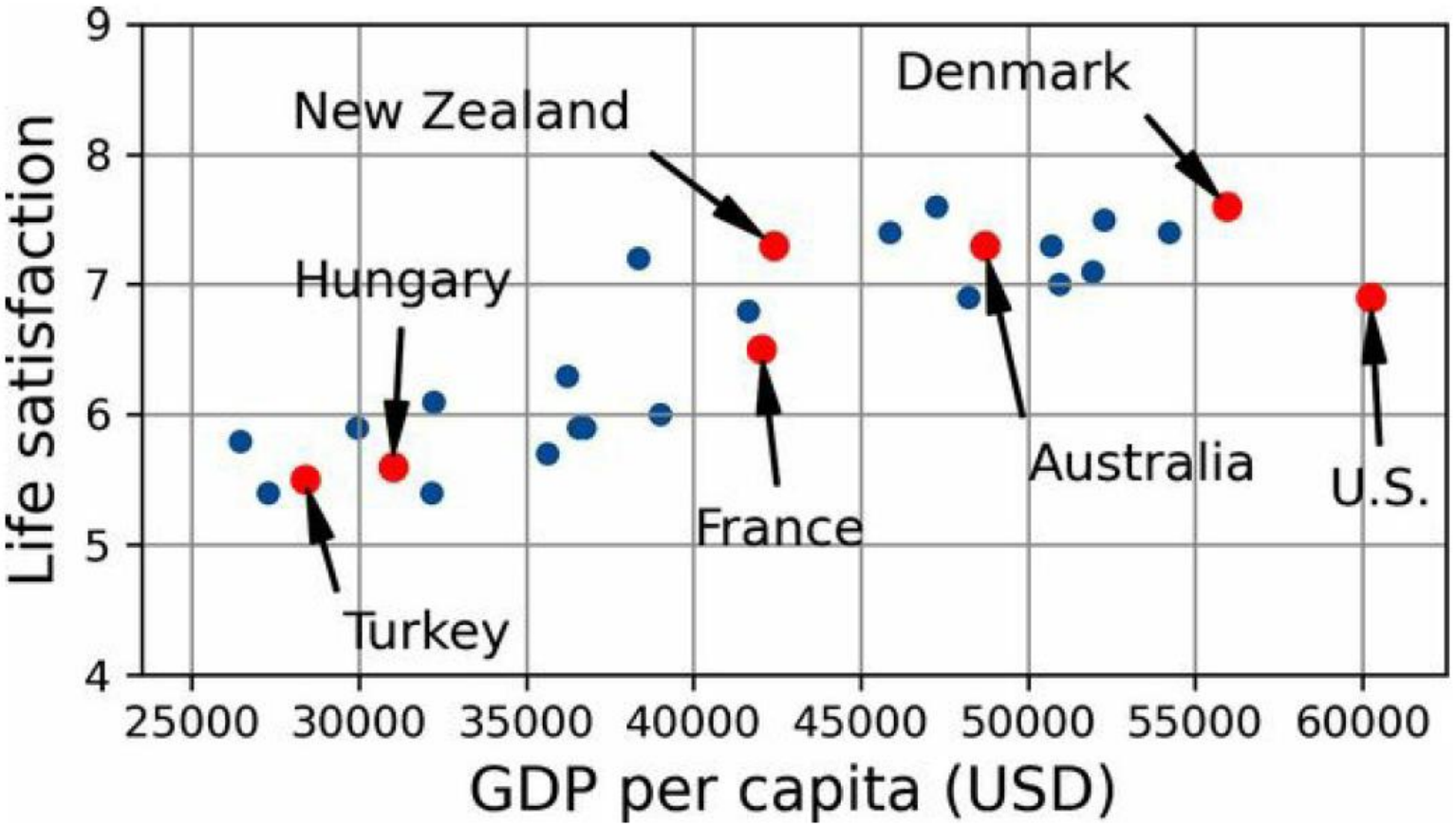
Country	GDP per capita (USD)	Life satisfaction
Turkey	28,384	5.5
Hungary	31,008	5.6
France	42,026	6.5
United States	60,236	6.9
New Zealand	42,404	7.3
Australia	48,698	7.3
Denmark	55,938	7.6

Big Picture

- There seems to be an emerging trend
  - Data is noisy - partly random
  - Life satisfaction goes up more or less linearly as the country's GDP per capita increases.
  - Model life satisfaction as a **linear function of GDP per capita**. This step is called model selection.

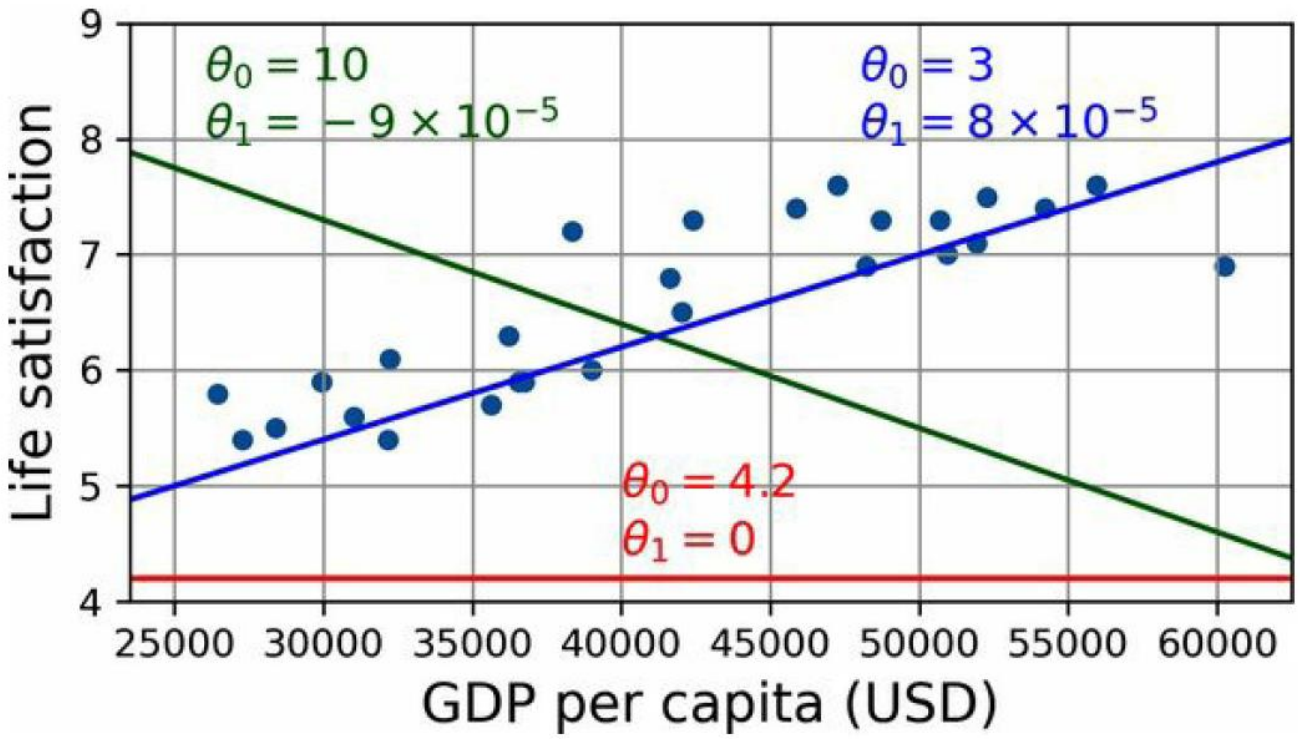


Big Picture


Life satisfaction
GDP per capita (USD)
New Zealand
Denmark
Hungary
France
Australia
U.S.
Turkey
9
8
7
6
5
4
25000
30000
35000
40000
45000
50000
55000
60000

Big Picture

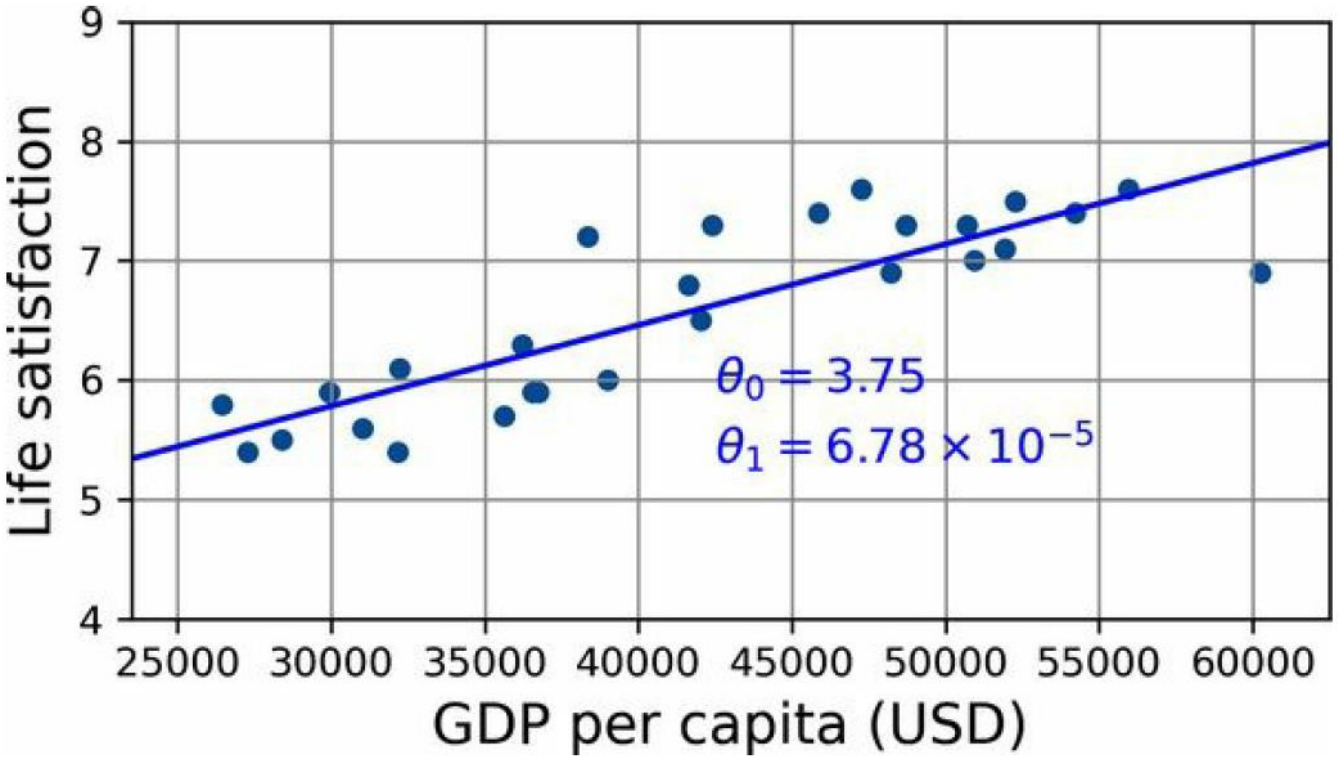
- you selected a linear model of life satisfaction with just one attribute, GDP per capita
- $L = \theta_1 G + \theta_0$
  - θ_1: Slope
  - θ_0: Intercept
  - Model Parameters
- By tweaking these parameters, you can make your model represent any linear function – **Training Model**



Big Picture

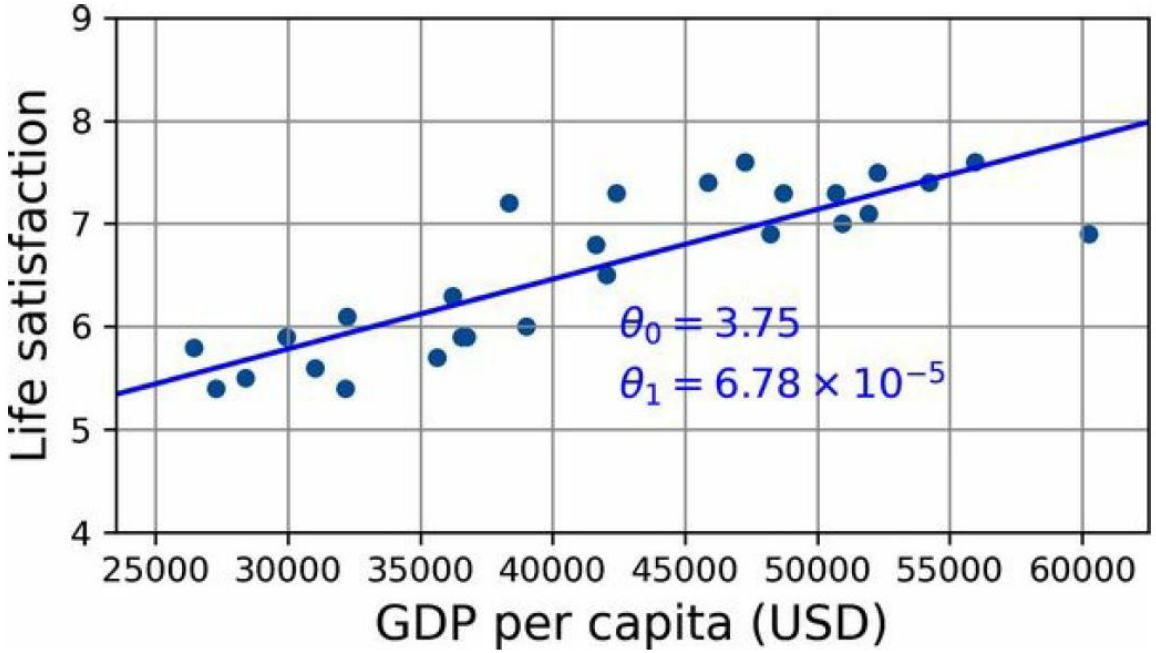
- How can you know which values will make your model perform best?
  - Specify a **performance measure**.
  - Utility function (or **fitness function**) that measures how good your model is
  - you can define a **cost function** that measures how bad it is
  - For **linear regression problems**, people typically use a **cost function** that measures the distance between the linear model's predictions and the training examples;
  - The objective is to minimize this distance
- This is where the **linear regression algorithm** comes in:
  - you feed it your training examples, and it finds the parameters that make the linear model fit best to your data

Big Picture



- $L = \theta_1 G + \theta_0$
- $L = 6.78 \times 10^{-5}\, G + 3.75$

Big Picture



- $L = \theta_1 G + \theta_0$
- $L = 6.78 \times 10^{-5}\, G + 3.75$
- Model can refer to:
  - Model Type - linear regression
  - Fully specified model architecture - linear regression with one input and one output
  - Final trained model ready to be used for predictions

Big Picture

- Now the model fits the training data as closely as possible (for a linear model)
- You are finally ready to run the model to make predictions
  - Inferences
  - How happy Cypriots are, and the OECD data does not have the answer
  - Cyprus's GDP per capita, find $37,655
  - Apply your model and find that life satisfaction
  - $L = 6.78 \times 10^{-5} \times 37{,}655 + 3.75 = 6.3$

Big Picture

In [1]:
```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
```

In [2]:
```
# Download and prepare the data
data_root = "https://github.com/ageron/data/raw/main/"
lifesat = pd.read_csv(data_root + "lifesat/lifesat.csv")
X = lifesat[["GDP per capita (USD)"]].values
y = lifesat[["Life satisfaction"]].values
```

In [3]:
```
# Visualize the data
lifesat.plot(kind='scatter', grid=True,
x="GDP per capita (USD)", y="Life satisfaction")
plt.axis([23_500, 62_500, 4, 9])
plt.show()
```

Big Picture

In [4]:
```
# Select a linear model
model = LinearRegression()
```

In [5]:
```
# Train the model
model.fit(X, y)
```

In [6]:
```
# Make a prediction for Cyprus
X_new = [[37_655.2]] # Cyprus' GDP per capita in 2020
print(model.predict(X_new)) # output: [[6.30165767]]
```

Big Picture

- If all went well, your model will make good predictions. If not, you may need to:
  - Use more attributes (employment rate, health, air pollution, etc.),
  - Get more or better-quality training data, or
  - Perhaps select a more powerful model (e.g., a polynomial regression model).

Project Overflow

- You studied the data.
- You selected a model.
- You trained it on the training data (i.e., the learning algorithm searched for the model parameter values that minimize a cost function).
- Finally, you applied the model to make predictions on new cases (this is called **inference**), hoping that this model will generalize well.