Building Blocks of Supervised Machine Learning

Swati Mishra

Applications of Machine Learning (4AL3)

Fall 2024

McMaster University

ENGINEERING

Supervised Machine Learning

- It is defined as the technique of mapping a given input to a target.
- For instance, if a given dataset D where each data instance is x_i, and the target is y , then the goal of supervised learning is to find f, such that:

$$f\left(x_1, x_2, \ldots, x_p\right) \rightarrow y$$

Supervised Machine Learning

- It is defined as the technique of mapping a given input to a target.
- For instance, if a given dataset D where each data instance is x_i, and the target is y , then the goal of supervised learning is to find f, such that:

$$\underbrace{f}_{\text{Model}}\underbrace{\left(x_1, x_2, \dots, x_p\right)}_{\text{Dataset}} \rightarrow \underbrace{y}_{\text{Target}}$$

Supervised Machine Learning

- It is defined as the technique of mapping a given input to a target.
- For instance, if a given dataset D where each data instance is x_i, and the target is y , then the goal of supervised learning is to find f, such that:

$$\underbrace{f}_{\text{Model}}\underbrace{\left(x_1, x_2, \dots, x_p\right)}_{\text{Dataset}} \rightarrow \underbrace{y}_{\text{Target}}$$

- We build the Predictive Model $\longleftarrow$ Dataset + Algorithm

Supervised Machine Learning - Objective

or

Predictive

Inferential

Supervised Machine Learning - Example

Question: Are you happy if you are rich?

Happiness score (h): Cantril Ladder Score which asks respondents to evaluate their life on a scale from 0 to 10.

Richness Score (r): GDP per capita. It indicates that the amount of output or income per person in an economy can indicate average productivity or average living standards.

Country	h	r
Luxembourg	7.0903	114164.470
Singapore	6.2620	98336.950
Qatar	6.3745	91461.620
Ireland	7.0211	83340.390
UAE	6.8245	71550.555
Switzerland	7.4802	70558.560
Norway	7.5539	64341.258
United States	6.8923	61355.650
Hong Kong	5.4304	61055.340
Iceland	7.4936	56816.363

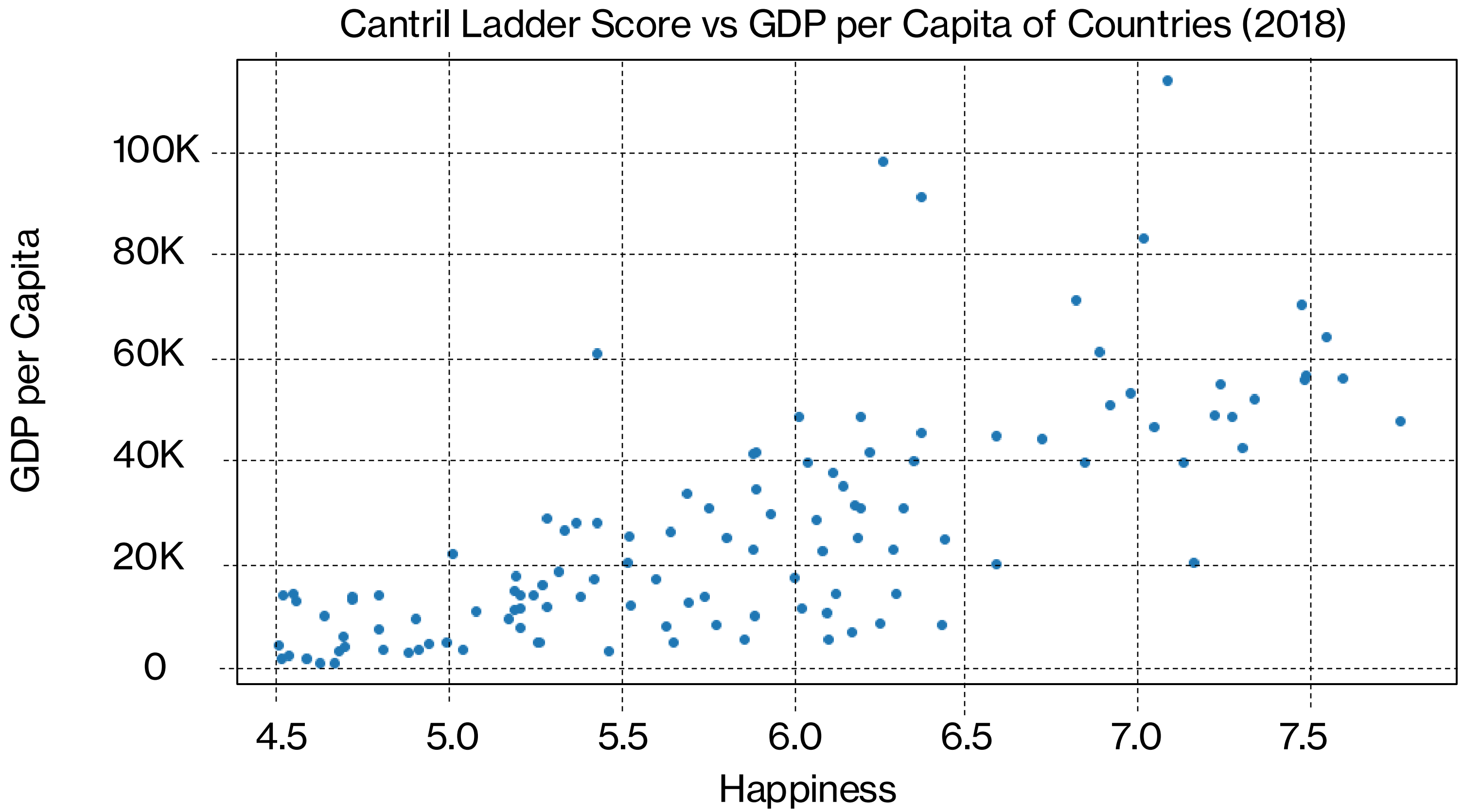
Supervised Machine Learning - Example

Our **goal**: Find the relationship between *happiness* (h) and *richness* (r) of an individual.

Formally speaking: Find f such that $h = f(r)$; $h \in \{h_1, h_2, \ldots, h_n\}$ and $r \in \{r_1, r_2, \ldots, r_n\}$

Our **hypothesis**: There is a linear relationship between the h and r.

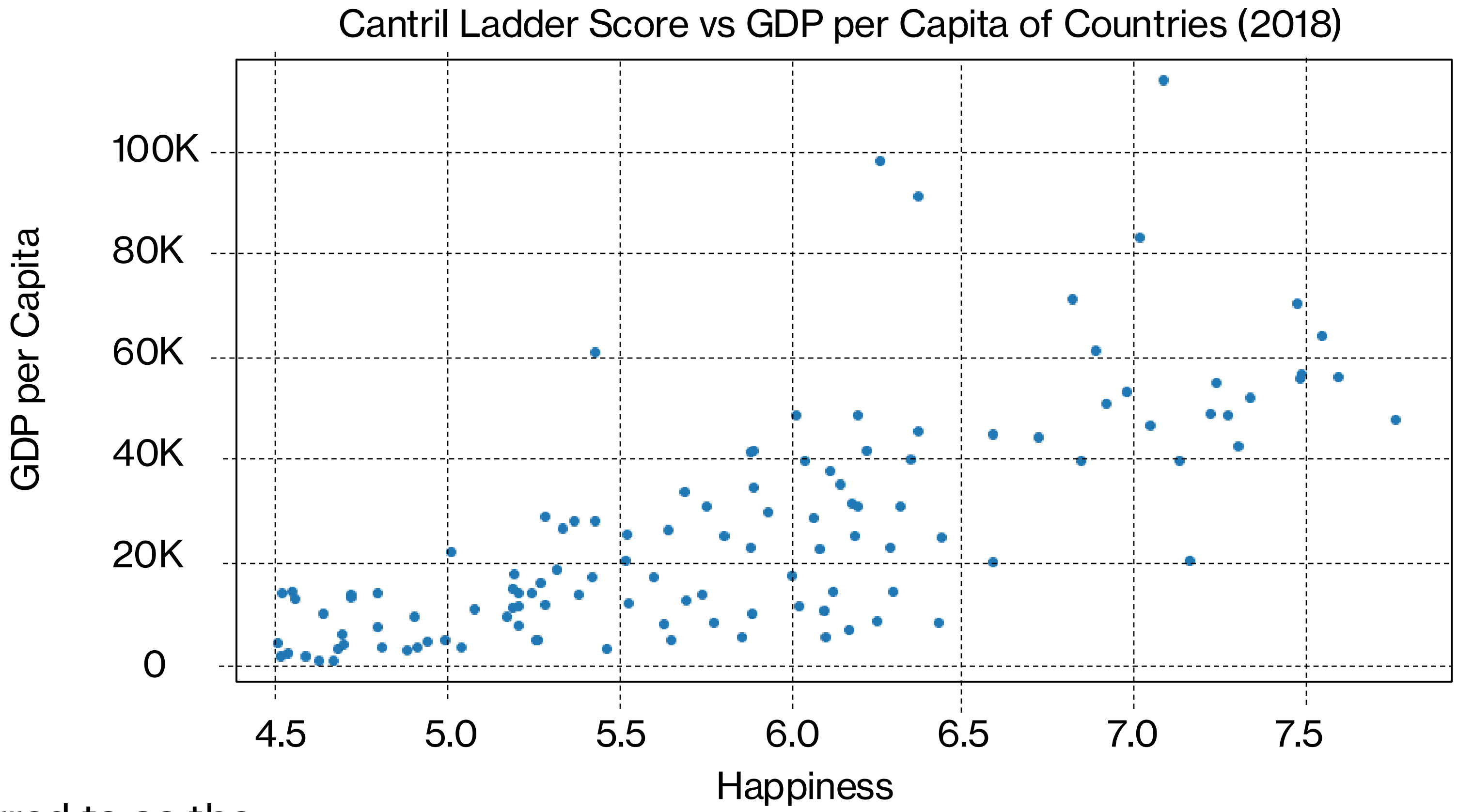
Linear Regression



$$y = f(\beta) = \beta_0 + \beta_1 * x$$

McMaster University

Linear Regression



Here β_i is referred to as the parameters of the model

$$y = f(\beta) = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + \cdots + \beta_p * x_p$$

Supervised Machine Learning - Terminology

- **Learner**: It is the *statistical model* that we are learning for the task.
- **Observation**: The single *data instance* of the model.
- **Training set**: The set of *estimation samples* used for computing the model.
- **Test set**: The *out-of-sample observation* that the learner has not seen before.
- **Algorithm**: The *estimation method* that is used to compute the model.
- **Features**: The vector *representation* of a single data instance used in the algorithm.

$$f(x_1, x_2, \ldots, x_p) \rightarrow y$$

Supervised Machine Learning - Terminology

- **Learner**: It is the *statistical model* that we are learning for the task.
- **Observation**: The single *data instance* of the model.
- **Training set**: The set of *estimation samples* used for computing the model.
- **Test set**: The *out-of-sample observation* that the learner has not seen before.
- **Algorithm**: The *estimation method* that is used to compute the model.
- **Features**: The vector *representation* of a single data instance used in the algorithm.

$$f(x_1, x_2, \dots, x_p) \to y$$

Supervised Machine Learning - Terminology

- **Learner**: It is the *statistical model* that we are learning for the task.
- **Observation**: The single *data instance* of the model.
- **Training set**: The set of *estimation samples* used for computing the model.
- **Test set**: The *out-of-sample observation* that the learner has not seen before.
- **Algorithm**: The *estimation method* that is used to compute the model.
- **Features**: The vector *representation* of a single data instance used in the algorithm.

$$f(x_1, x_2, \dots, x_p) \rightarrow y$$

Supervised Machine Learning - Terminology

- **Learner**: It is the *statistical model* that we are learning for the task.
- **Observation**: The single *data instance* of the model.
- **Training set**: The set of *estimation samples* used for computing the model.
- **Test set**: The *out-of-sample observation* that the learner has not seen before.
- **Algorithm**: The *estimation method* that is used to compute the model.
- **Features**: The vector *representation* of a single data instance used in the algorithm.

$$f(x_1, x_2, \ldots, x_p) \rightarrow y$$

McMaster University

Supervised Machine Learning - Terminology

- **Learner**: It is the *statistical model* that we are learning for the task.
- **Observation**: The single *data instance* of the model.
- **Training set**: The set of *estimation samples* used for computing the model.
- **Test set**: The *out-of-sample observation* that the learner has not seen before.
- **Algorithm**: The *estimation method* that is used to compute the model.
- **Features**: The vector *representation* of a single data instance used in the algorithm.

$$f(x_1, x_2, \ldots, x_p) \rightarrow y$$

Supervised Machine Learning - Terminology

- **Learner**: It is the *statistical model* that we are learning for the task.
- **Observation**: The single *data instance* of the model.
- **Training set**: The set of *estimation samples* used for computing the model.
- **Test set**: The *out-of-sample observation* that the learner has not seen before.
- **Algorithm**: The *estimation method* that is used to compute the model.
- **Features**: The vector *representation* of a single data instance used in the algorithm.

$$f\left(x_1, x_2, \ldots, x_p\right) \to y$$

McMaster University

Parametric Models

- A learning model that summarizes data with a set of parameters of fixed size.

Parametric Models

- A learning model that summarizes data with a set of parameters of fixed size.
- Examples:
  - Linear Discriminant Analysis
  - Perceptron
  - Naive Bayes
  - Simple Neural Networks

$$\beta_2 * x_2 + \beta_1 * x_1 + \beta_0 = 0$$

Parametric Models

- A learning model that summarizes data with a set of parameters of fixed size.
- Benefits:
  - **Speed**: Parametric models are very fast to learn from data.
  - **Less Data**: No large datasets necessary and can work well with imperfect data.

Parametric Models

- A learning model that summarizes data with a set of parameters of fixed size.
- Benefits:
  - **Speed**: Parametric models are very fast to learn from data.
  - **Less Data**: No large datasets necessary and can work well with imperfect data.
- Limitations:
  - **Constrained**: By choosing a functional form these methods are highly constrained to the specified form and hence suited to simpler problems.
  - **Not very accurate**: In practice the methods are unlikely to match the underlying mapping function

Non-Parametric Models

- These are algorithms that do not make a lot of assumptions about the data

Non-Parametric Models

- These are algorithms that do not make a lot of assumptions about the data
- Examples: .
    - k-Nearest Neighbors
    - Decision Trees like CART and C4.5
    - Support Vector Machines

Non-Parametric Models

- These are algorithms that do not make a lot of assumptions about the data .
- Benefits of Nonparametric Machine Learning Algorithms:
  - **Flexibility**: Capable of fitting many functional forms.
  - **Power**: No assumptions (or weak assumptions) about the underlying function.
  - **Performance**: Can result in higher performance models for prediction.

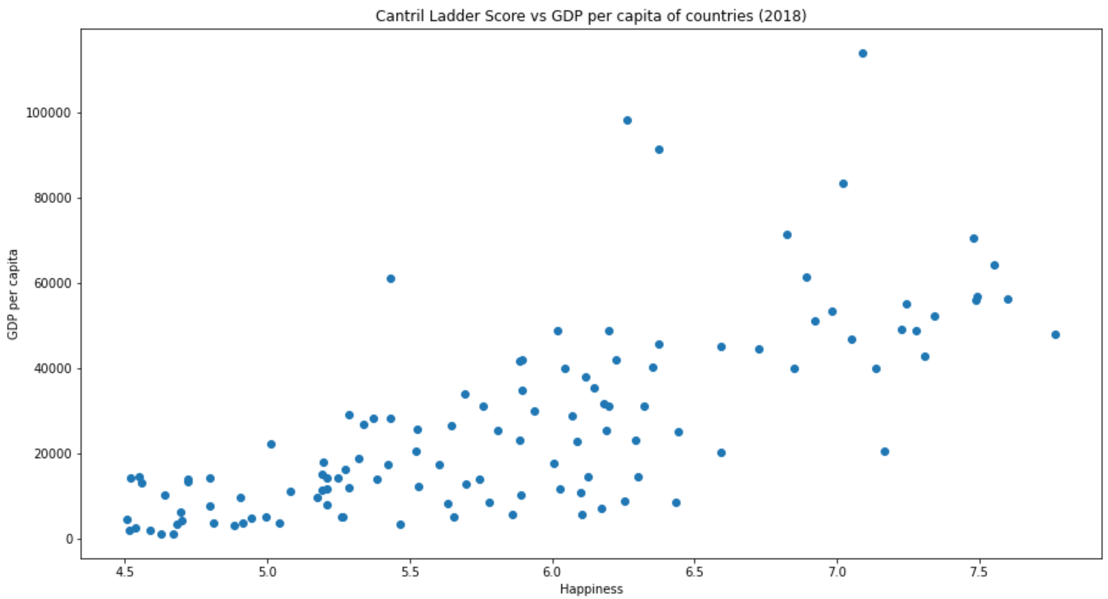
Non-Parametric Models

- These are algorithms that do not make a lot of assumptions about the data .
- Benefits of Nonparametric Machine Learning Algorithms:
  - **Flexibility**: Capable of fitting many functional forms.
  - **Power**: No assumptions (or weak assumptions) about the underlying function.
  - **Performance**: Can result in higher performance models for prediction.
- Limitations of Nonparametric Machine Learning Algorithms:
  - **More data**: Require a lot more training data to estimate the mapping function.
  - **Slower**: A lot slower to train as they often have far more parameters to train.
  - **Overfitting**: High risk of overfitting
  - **Explainability**: it is harder to explain why specific predictions are made.

Supervised Learning – Linear Regression

$$y' = \beta_0 + \beta_1 * \mathrm{x}_1 + \beta_2 * \mathrm{x}_2 + \cdots + \beta_p * \mathrm{x}_p$$

$$\mathrm{y}_1' = \beta_0 + \beta_1 * \mathrm{x}_{11} + \beta_2 * \mathrm{x}_{12} + \cdots + \beta_n * \mathrm{x}_{1p}$$
$$\mathrm{y}_2' = \beta_0 + \beta_1 * \mathrm{x}_{21} + \beta_2 * \mathrm{x}_{22} + \cdots + \beta_n * \mathrm{x}_{2p}$$
$$\mathrm{y}_n' = \beta_0 + \beta_1 * \mathrm{x}_{n1} + \beta_2 * \mathrm{x}_{n2} + \cdots + \beta_n * \mathrm{x}_{np}$$



β_i = feature weights

x_i = the feature value

n = number of features

Supervised Learning – Linear Regression

$$
\begin{aligned}
y_1' &= \beta_0 + \beta_1 * x_{11} + \beta_2 * x_{12} + \cdots + \beta_n * x_{1p} \\
y_2' &= \beta_0 + \beta_1 * x_{21} + \beta_2 * x_{22} + \cdots + \beta_n * x_{2p} \\
y_n' &= \beta_0 + \beta_1 * x_{n1} + \beta_2 * x_{n2} + \cdots + \beta_n * x_{np}
\end{aligned}
$$

$$
X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}
$$

p =number of features

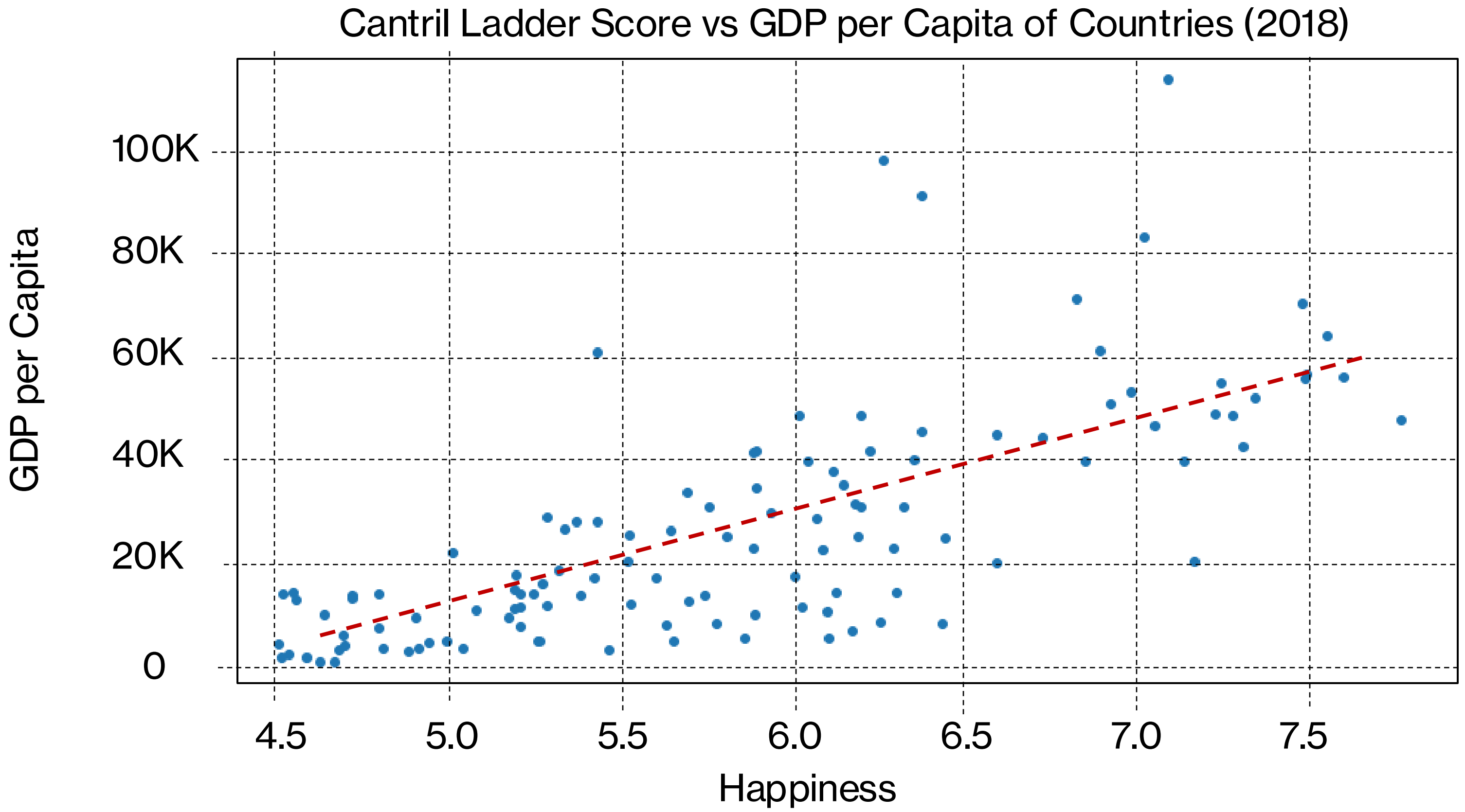
n = number of observations

$$Y' = h_\beta(x) = \beta . X$$

h_β is the hypothesis function

β is model parameters

Ordinary Least Squares



Find β such that it minimizes

$$\sum_{i=0}^{n} d^2$$

Ordinary Least Squares

Input data

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}$$

Target

$$y = \begin{matrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{matrix}$$

Objective

$$MSE = \frac{1}{n}\sum_{n=1}^{n}(y_i - y_i')^2$$

Loss function (K) **=** Mean Squared Error

Find β

Ordinary Least Squares

Input data

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}$$

Target

$$y = \begin{matrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{matrix}$$

Objective

$$MSE = \frac{1}{n}\sum_{n=1}^{n}(y_i - y_i')^2$$

Loss function (K) **=** Mean Squared Error

Find β

Closed Form Equation

$$\beta' = (\mathrm{X}^T X)^{-1}\mathrm{X}^T y$$

Ordinary Least Squares

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}$$

```
#arrange in matrix format
X = np.column_stack((np.ones(len(self.input)),self.input))
```

Input data

```
def __init__(self,x_:list,y_:list) -> None:

    self.input = np.array(x_)
    self.target = np.array(y_)
```

Target

$$y = \begin{matrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{matrix}$$

```
#arrange in matrix format
Y = (np.column_stack(y_train)).T
```

Closed Form Equation

$$\beta' = (X^T X)^{-1} X^T y$$

```
def train(self, X, Y):

    #compute beta
    return np.linalg.inv(X.T.dot(X)).dot(X.T).dot(Y)
```

Ordinary Least Squares

After finding β

Test data

$$X\_test = \begin{pmatrix} x'_{11} & x'_{12} & \cdots & x'_{1p} \\ x'_{21} & x'_{22} & & x'_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x'_{n1} & x'_{n2} & \cdots & x'_{np} \end{pmatrix}$$

Find y for test set

$$y' = ?$$

$$Y' = h_\beta(x) = \beta.X$$

```python
def predict(self, X_test,beta):

    #predict using beta
    Y_hat = X_test*beta.T
    return np.sum(Y_hat,axis=1)
```

Ordinary Least Squares

After finding β

Test data

$$X\_test = \begin{pmatrix} x'_{11} & x'_{12} & \cdots & x'_{1p} \\ x'_{21} & x'_{22} & & x'_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x'_{n1} & x'_{n2} & \cdots & x'_{np} \end{pmatrix}$$

Find y for test set

$$y' = ?$$

Evaluate using : $MSE = \frac{1}{n}\sum_{n=1}^{n}(y_i - y_i')^2$

```python
def predict(self, X_test,beta):

    #predict using beta
    Y_hat = X_test*beta.T
    return np.sum(Y_hat,axis=1)
```

Ordinary Least Squares

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}$$

```
#arrange in matrix format
X = np.column_stack((np.ones(len(self.input)),self.input))
```

Input data

```
def __init__(self,x_:list,y_:list) -> None:

    self.input = np.array(x_)
    self.target = np.array(y_)
```

Target

$$y = \begin{matrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{matrix}$$

```
#arrange in matrix format
Y = (np.column_stack(y_train)).T
```

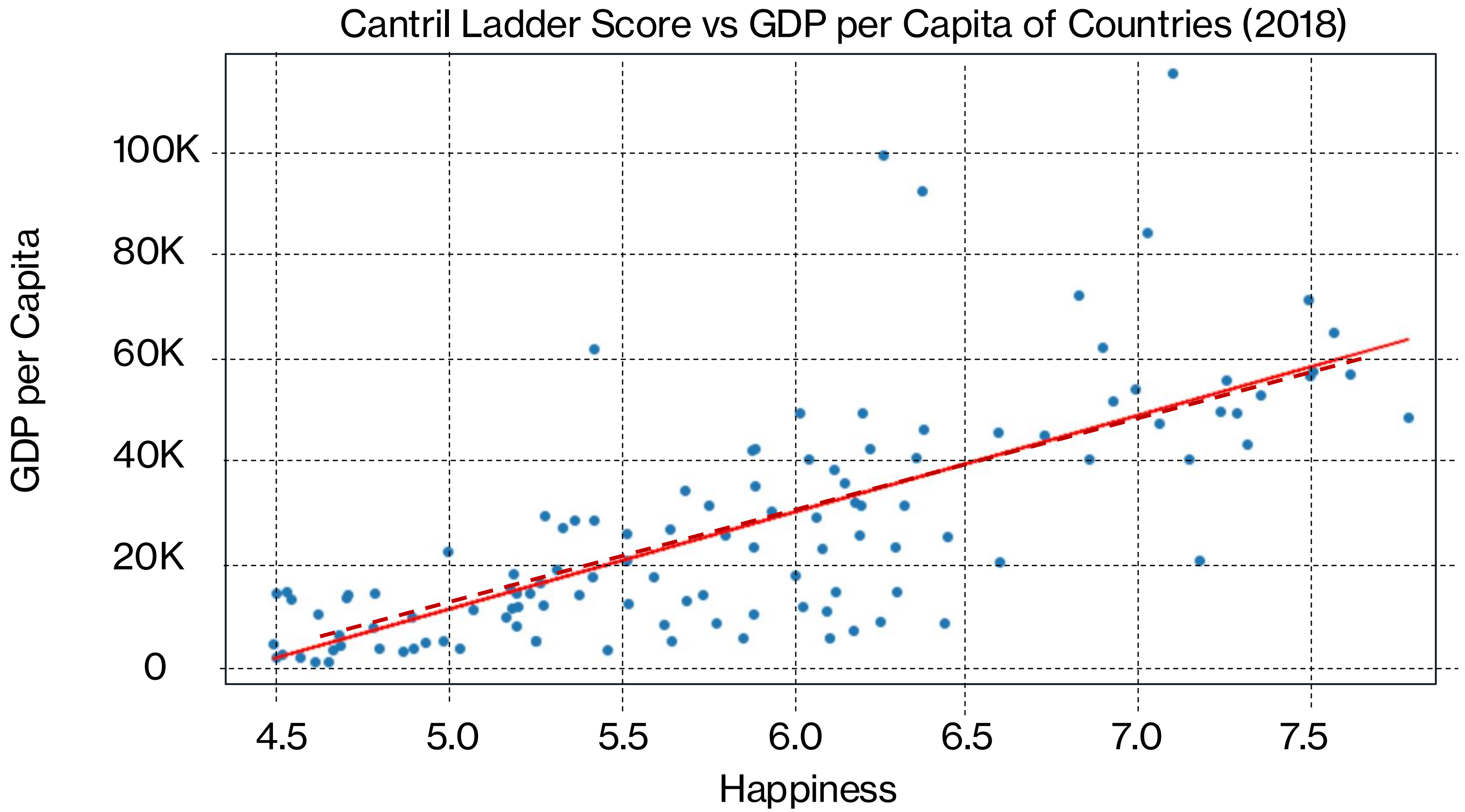
Closed Form Equation

$$\beta' = (X^T X)^{-1} X^T y$$

```
def train(self, X, Y):

    #compute beta
    return np.linalg.inv(X.T.dot(X)).dot(X.T).dot(Y)
```

Ordinary Least Squares



Other Loss Functions

- $Root\ Mean\ Square\ Error\ (RMSE)$ $$\sqrt{\frac{1}{n}\sum_{n=1}^{n}(y'_i\ - y_i)^2}$$

- $Mean\ Absoluter\ Error\ (MAE)$ $$\frac{1}{n}\sum_{n=1}^{n}|y'_i\ - y_i|$$

Thank You