Machine Learning: Algorithms and Applications

Philip O. Ogunbona

Advanced Multimedia Research Lab
University of Wollongong

Regression

Outline

Regression - Historical and general ideas

- Regression is a supervised learning method often used in prediction tasks (with modification, also in classification)
- Regression as a scientific method first appeared around 1885
- Francis Galton developed the ideas in the studies of hereditary stature - comparison of height of parents and their children (Izenman 2008)
- Galton did not link the least squares method and regression which was discovered 80 years later
- Linear regression models can be simple, multiple or multivariate
  1. simple linear regression - one input and one output
  2. multiple regression - many inputs and one output
  3. multivariate regression - many inputs and many outputs
- In general there is the output (also called the dependent variable) that is assumed to be linearly related to the input(s) (also called the independent variables; input space)
- Independent variables could be formed from a linear combination of a fixed set of nonlinear functions (basis functions) of input variables
- It is the coefficients of the function of relatedness that we want to determine and obtain an equation for use in prediction on new observed variables

Regression - Theoretical development

Regression problem

- Let $\mathcal{X}$ denote the input space and $\mathcal{Y}$ a measurable subset of $\mathbb{R}$.
- Denote by $\mathcal{D}$ an unknown distribution over $\mathcal{X}$ according to which the inputs are drawn
- Let $f : \mathcal{X} \to \mathcal{Y}$ be the target labelling function
- Learner receives a labelled sample $S = \{(x_1, y_1), \ldots, (x_m, y_m)\} \in (\mathcal{X} \times \mathcal{Y})$ with $x_1, \ldots, x_m$ drawn i.i.d from $\mathcal{D}$ and $y_i = f(x_i)$ for all $i \in [1, m]$.
- Denote by $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ the loss function measuring the magnitude of error
  - Commonly, squared error is used: $\mathcal{L}(y, \hat{y}) = ||y - \hat{y}||^2$ for all $y, \hat{y} \in \mathcal{Y}$
  - Generally, $\mathcal{L}_p$ loss function may be used: $\mathcal{L}_p(y, \hat{y}) = ||y - \hat{y}||^p$ for all $y, \hat{y} \in \mathcal{Y}$ and some $p \geq 1$
- Given a hypothesis set $\mathcal{H}$ of functions mapping $\mathcal{X}$ to $\mathcal{Y}$, regression problem consists of using the labelled sample S to find the hypothesis $h \in \mathcal{H}$ with small expected loss or generalization error $\mathcal{R}(h)$ with respect to the target function, f:

$$\mathcal{R}(h) = E_{x \sim \mathcal{D}}[\mathcal{L}(h(x), f(x))] \tag{1}$$

- Empirical loss is:

$$\hat{\mathcal{R}}(h) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(h(x_i), y_i) \tag{2}$$

Regression - Theoretical development

Regression problem

- Let $\mathcal{X}$ denote the input space and $\mathcal{Y}$ a measurable subset of $\mathbb{R}$.
- Denote by $\mathcal{D}$ an unknown distribution over $\mathcal{X}$ according to which the inputs are drawn
- Let $f : \mathcal{X} \to \mathcal{Y}$ be the target labelling function
- Learner receives a labelled sample $S = \{(x_1, y_1), \ldots, (x_m, y_m)\} \in (\mathcal{X} \times \mathcal{Y})$ with $x_1, \ldots, x_m$ drawn i.i.d from $\mathcal{D}$ and $y_i = f(x_i)$ for all $i \in [1, m]$.
- Denote by $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ the loss function measuring the magnitude of error
  - Commonly, squared error is used: $\mathcal{L}(y, \hat{y}) = ||y - \hat{y}||^2$ for all $y, \hat{y} \in \mathcal{Y}$
  - Generally, $\mathcal{L}_p$ loss function may be used: $\mathcal{L}_p(y, \hat{y}) = ||y - \hat{y}||^p$ for all $y, \hat{y} \in \mathcal{Y}$ and some $p \geq 1$
- Given a hypothesis set $\mathcal{H}$ of functions mapping $\mathcal{X}$ to $\mathcal{Y}$, regression problem consists of using the labelled sample S to find the hypothesis $h \in \mathcal{H}$ with small expected loss or generalization error $\mathcal{R}(h)$ with respect to the target function, f:

$$\mathcal{R}(h) = E_{x \sim \mathcal{D}}[\mathcal{L}(h(x), f(x))] \tag{1}$$

- Empirical loss is:

$$\hat{\mathcal{R}}(h) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(h(x_i), y_i) \tag{2}$$

Regression - Theoretical development

Regression problem

- Let $\mathcal{X}$ denote the input space and $\mathcal{Y}$ a measurable subset of $\mathbb{R}$.
- Denote by $\mathcal{D}$ an unknown distribution over $\mathcal{X}$ according to which the inputs are drawn
- Let $f : \mathcal{X} \to \mathcal{Y}$ be the target labelling function
- Learner receives a labelled sample $S = \{(x_1, y_1), \ldots, (x_m, y_m)\} \in (\mathcal{X} \times \mathcal{Y})$ with $x_1, \ldots, x_m$ drawn i.i.d from $\mathcal{D}$ and $y_i = f(x_i)$ for all $i \in [1, m]$.
- Denote by $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ the loss function measuring the magnitude of error
  - Commonly, squared error is used: $\mathcal{L}(y, \hat{y}) = ||y - \hat{y}||^2$ for all $y, \hat{y} \in \mathcal{Y}$
  - Generally, $\mathcal{L}_p$ loss function may be used: $\mathcal{L}_p(y, \hat{y}) = ||y - \hat{y}||^p$ for all $y, \hat{y} \in \mathcal{Y}$ and some $p \geq 1$
- Given a hypothesis set $\mathcal{H}$ of functions mapping $\mathcal{X}$ to $\mathcal{Y}$, regression problem consists of using the labelled sample S to find the hypothesis $h \in \mathcal{H}$ with small expected loss or generalization error $\mathcal{R}(h)$ with respect to the target function, f:

$$\mathcal{R}(h) = E_{x \sim \mathcal{D}}[\mathcal{L}(h(x), f(x))] \tag{1}$$

- Empirical loss is:

$$\hat{\mathcal{R}}(h) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(h(x_i), y_i) \tag{2}$$

Regression - Theoretical development

Regression problem

- Let $\mathcal{X}$ denote the input space and $\mathcal{Y}$ a measurable subset of $\mathbb{R}$.
- Denote by $\mathcal{D}$ an unknown distribution over $\mathcal{X}$ according to which the inputs are drawn
- Let $f : \mathcal{X} \to \mathcal{Y}$ be the target labelling function
- Learner receives a labelled sample $S = \{(x_1, y_1), \ldots, (x_m, y_m)\} \in (\mathcal{X} \times \mathcal{Y})$ with $x_1, \ldots, x_m$ drawn i.i.d from $\mathcal{D}$ and $y_i = f(x_i)$ for all $i \in [1, m]$.
- Denote by $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ the loss function measuring the magnitude of error
  - Commonly, squared error is used: $\mathcal{L}(y, \hat{y}) = ||y - \hat{y}||^2$ for all $y, \hat{y} \in \mathcal{Y}$
  - Generally, $\mathcal{L}_p$ loss function may be used: $\mathcal{L}_p(y, \hat{y}) = ||y - \hat{y}||^p$ for all $y, \hat{y} \in \mathcal{Y}$ and some $p \geq 1$
- Given a hypothesis set $\mathcal{H}$ of functions mapping $\mathcal{X}$ to $\mathcal{Y}$, regression problem consists of using the labelled sample S to find the hypothesis $h \in \mathcal{H}$ with small expected loss or generalization error $\mathcal{R}(h)$ with respect to the target function, f:

$$\mathcal{R}(h) = E_{x \sim \mathcal{D}}[\mathcal{L}(h(x), f(x))] \tag{1}$$

- Empirical loss is:

$$\hat{\mathcal{R}}(h) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(h(x_i), y_i) \tag{2}$$

Regression - Theoretical development

Regression problem

- Let $\mathcal{X}$ denote the input space and $\mathcal{Y}$ a measurable subset of $\mathbb{R}$.
- Denote by $\mathcal{D}$ an unknown distribution over $\mathcal{X}$ according to which the inputs are drawn
- Let $f : \mathcal{X} \to \mathcal{Y}$ be the target labelling function
- Learner receives a labelled sample $S = \{(x_1, y_1), \ldots, (x_m, y_m)\} \in (\mathcal{X} \times \mathcal{Y})$ with $x_1, \ldots, x_m$ drawn i.i.d from $\mathcal{D}$ and $y_i = f(x_i)$ for all $i \in [1, m]$.
- Denote by $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ the loss function measuring the magnitude of error
  - Commonly, squared error is used: $\mathcal{L}(y, \hat{y}) = ||y - \hat{y}||^2$ for all $y, \hat{y} \in \mathcal{Y}$
  - Generally, $\mathcal{L}_p$ loss function may be used: $\mathcal{L}_p(y, \hat{y}) = ||y - \hat{y}||^p$ for all $y, \hat{y} \in \mathcal{Y}$ and some $p \geq 1$
- Given a hypothesis set $\mathcal{H}$ of functions mapping $\mathcal{X}$ to $\mathcal{Y}$, regression problem consists of using the labelled sample S to find the hypothesis $h \in \mathcal{H}$ with small expected loss or generalization error $\mathcal{R}(h)$ with respect to the target function, f:

$$\mathcal{R}(h) = E_{x \sim \mathcal{D}}[\mathcal{L}(h(x), f(x))] \tag{1}$$

- Empirical loss is:

$$\hat{\mathcal{R}}(h) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(h(x_i), y_i) \tag{2}$$

Regression - Theoretical development

Regression problem

- Let $\mathcal{X}$ denote the input space and $\mathcal{Y}$ a measurable subset of $\mathbb{R}$.
- Denote by $\mathcal{D}$ an unknown distribution over $\mathcal{X}$ according to which the inputs are drawn
- Let $f : \mathcal{X} \to \mathcal{Y}$ be the target labelling function
- Learner receives a labelled sample $S = \{(x_1, y_1), \ldots, (x_m, y_m)\} \in (\mathcal{X} \times \mathcal{Y})$ with $x_1, \ldots, x_m$ drawn i.i.d from $\mathcal{D}$ and $y_i = f(x_i)$ for all $i \in [1, m]$.
- Denote by $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ the loss function measuring the magnitude of error
  - Commonly, squared error is used: $\mathcal{L}(y, \hat{y}) = ||y - \hat{y}||^2$ for all $y, \hat{y} \in \mathcal{Y}$
  - Generally, $\mathcal{L}_p$ loss function may be used: $\mathcal{L}_p(y, \hat{y}) = ||y - \hat{y}||^p$ for all $y, \hat{y} \in \mathcal{Y}$ and some $p \geq 1$
- Given a hypothesis set $\mathcal{H}$ of functions mapping $\mathcal{X}$ to $\mathcal{Y}$, regression problem consists of using the labelled sample $\mathcal{S}$ to find the hypothesis $h \in \mathcal{H}$ with small expected loss or generalization error $\mathcal{R}(h)$ with respect to the target function, f:

$$\mathcal{R}(h) = E_{x \sim \mathcal{D}}[\mathcal{L}(h(x), f(x))] \tag{1}$$

- Empirical loss is:

$$\hat{\mathcal{R}}(h) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(h(x_i), y_i) \tag{2}$$

Regression - Theoretical development

Regression problem

- Let $\mathcal{X}$ denote the input space and $\mathcal{Y}$ a measurable subset of $\mathbb{R}$.
- Denote by $\mathcal{D}$ an unknown distribution over $\mathcal{X}$ according to which the inputs are drawn
- Let $f : \mathcal{X} \to \mathcal{Y}$ be the target labelling function
- Learner receives a labelled sample $S = \{(x_1, y_1), \ldots, (x_m, y_m)\} \in (\mathcal{X} \times \mathcal{Y})$ with $x_1, \ldots, x_m$ drawn i.i.d from $\mathcal{D}$ and $y_i = f(x_i)$ for all $i \in [1, m]$.
- Denote by $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ the loss function measuring the magnitude of error
  - Commonly, squared error is used: $\mathcal{L}(y, \hat{y}) = ||y - \hat{y}||^2$ for all $y, \hat{y} \in \mathcal{Y}$
  - Generally, $\mathcal{L}_p$ loss function may be used: $\mathcal{L}_p(y, \hat{y}) = ||y - \hat{y}||^p$ for all $y, \hat{y} \in \mathcal{Y}$ and some $p \geq 1$
- Given a hypothesis set $\mathcal{H}$ of functions mapping $\mathcal{X}$ to $\mathcal{Y}$, regression problem consists of using the labelled sample $\mathcal{S}$ to find the hypothesis $h \in \mathcal{H}$ with small expected loss or generalization error $\mathcal{R}(h)$ with respect to the target function, f:

$$\mathcal{R}(h) = E_{x \sim \mathcal{D}}[\mathcal{L}(h(x), f(x))] \tag{1}$$

- Empirical loss is:

$$\hat{\mathcal{R}}(h) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(h(x_i), y_i) \tag{2}$$

Linear regression

- Let $\mathbf{\Phi} : \mathcal{X} \to \mathbb{R}^N$ be a feature mapping from input space $\mathcal{X}$ to $\mathbb{R}^N$
- Consider a family of linear hypotheses

$$\mathcal{H} = \{x \mapsto \mathbf{w}.\mathbf{\Phi}(x) + b : \mathbf{w} \in \mathbb{R}^N, b \in \mathbb{R}\} \tag{3}$$

- Linear regression seeks an hypothesis in $\mathcal{H}$ with the smallest mean squared error
- Given a sample set $\mathcal{S} = ((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \mathcal{Y})^m$ we need to solve the following optimization problem:

$$\min_{w,b} \frac{1}{m} \sum_{i=1}^{m} (\mathbf{w}.\mathbf{\Phi}(x_i) + b - y_i)^2 \tag{4}$$

- If we write $\boldsymbol{X} = \begin{bmatrix} \Phi(x_1) & \cdots & \Phi(x_m) \\ 1 & \cdots & 1 \end{bmatrix}$, $\boldsymbol{W} = \begin{bmatrix} w_1 \\ \vdots \\ w_N \\ 1 \end{bmatrix}$ and $\boldsymbol{Y} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}$ the optimization problem in (4) can be written compactly as

$$\min_{\boldsymbol{W}} F(\boldsymbol{W}) = \frac{1}{m} ||\boldsymbol{X}^T \boldsymbol{W} - \boldsymbol{Y}||^2 \tag{5}$$

Linear regression

- Consider the dimensions of the entries in Equation (5)
  - $\boldsymbol{X}^T \in \mathbb{R}^{m \times (N+1)}$
  - $\boldsymbol{W} \in \mathbb{R}^{N+1}$
  - $\boldsymbol{X}^T\boldsymbol{W} \in \mathbb{R}^m$
  - $\boldsymbol{Y} \in \mathbb{R}^m$
- In transforming Equation (4) to Equation (5) we have done the following:

$$\begin{aligned} y_i &= w_i x_i + b \\ &= w_i' x_i + 1 \end{aligned}$$

  where the bias b has been absorbed in the weight w'

- The optimization problem in Equation (5), $F(\boldsymbol{W})$, is convex, differentiable and has a global minimum that can be obtained by differentiating $F(\boldsymbol{W}) = \frac{1}{m}||\boldsymbol{X}^T\boldsymbol{W} - \boldsymbol{Y}||^2$ with respect to $\boldsymbol{W}$ and equating to zero
- $\nabla F(\boldsymbol{W}) = 0$; $\frac{2}{m}\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{W} - \boldsymbol{Y}) = 0$ from which $\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{W} = \boldsymbol{X}\boldsymbol{Y}$

$$\boldsymbol{W} = \begin{cases} (\boldsymbol{X}\boldsymbol{X}^T)^{-1}\boldsymbol{X}\boldsymbol{Y} & \text{if } \boldsymbol{X}\boldsymbol{X}^T \text{ is invertible} \\ (\boldsymbol{X}\boldsymbol{X}^T)^{\dagger}\boldsymbol{X}\boldsymbol{Y} & \text{otherwise; using the pseudo-inverse } \dagger \end{cases} \tag{6}$$

Linear Regression

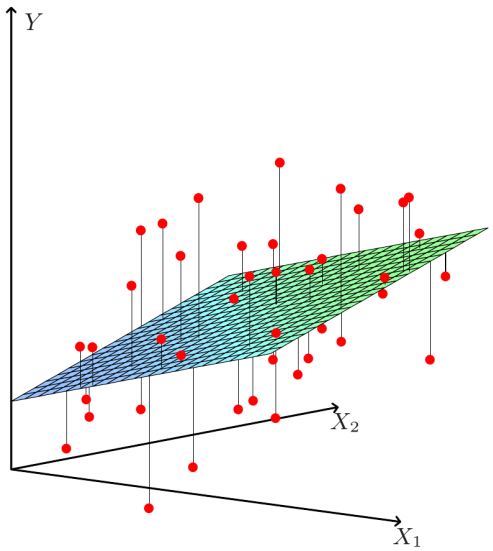

Figure 1: Linear least square fitting ($X \in \mathbb{R}^2$). We seek the linear function of X that minimizes sum of squared errors from Y (Hastie et al. 2001).

Linear Regression

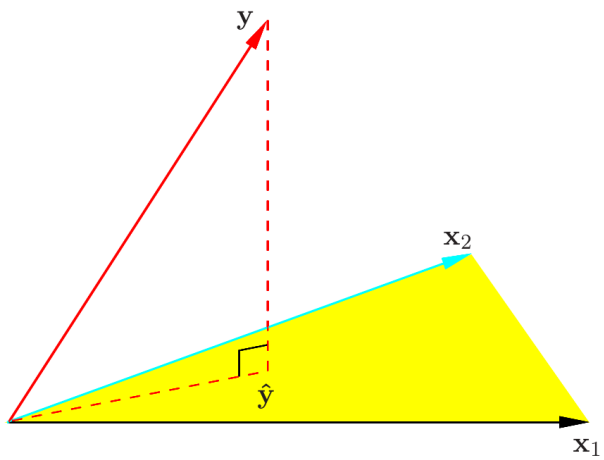

Figure 2: N-dimensional geometry of least squares regression with two independent variables $\boldsymbol{x_1}, \boldsymbol{x_2}$. Predicted y vector is orthogonally projected onto the hyperplane spanned by $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$. $\hat{y}$ represents the vector of the least squares predictions (Hastie et al. 2001).

Linear Regression

- Results shown in Equation (6) is also referred to as the least squares estimate of the weight vector (coefficients), $\boldsymbol{W}$, of the linear regression model
- Important notes on linear regression:
  - Prediction accuracy of least squares estimate often has low bias but large variance[1]
  - If there is a large number of independent variables it is desirable to know the key variables that exhibit strong effect
  - There is no strong generalization guarantee because we only minimize empirical error without controlling the norm (length) of the weight vector; there is no regularization
  - There is possibility of overfitting or underfitting

[1] See Figure (8) here

Linear Regression

Quick example:(Does money make people happier?) (Géron 2019, p.19)

Dataset

1. Better Life Index data from OECD's website (Life satisfaction data)
2. GDP per capita from IMF's website

Table 1: Does money make people happier?

Country	GDP per capita (USD)	Life satisfaction
Hungary	12,240	4.9
Korea	27,195	5.8
France	37,675	6.5
Australia	50,962	7.3
United States	55,805	7.2

Linear Regression

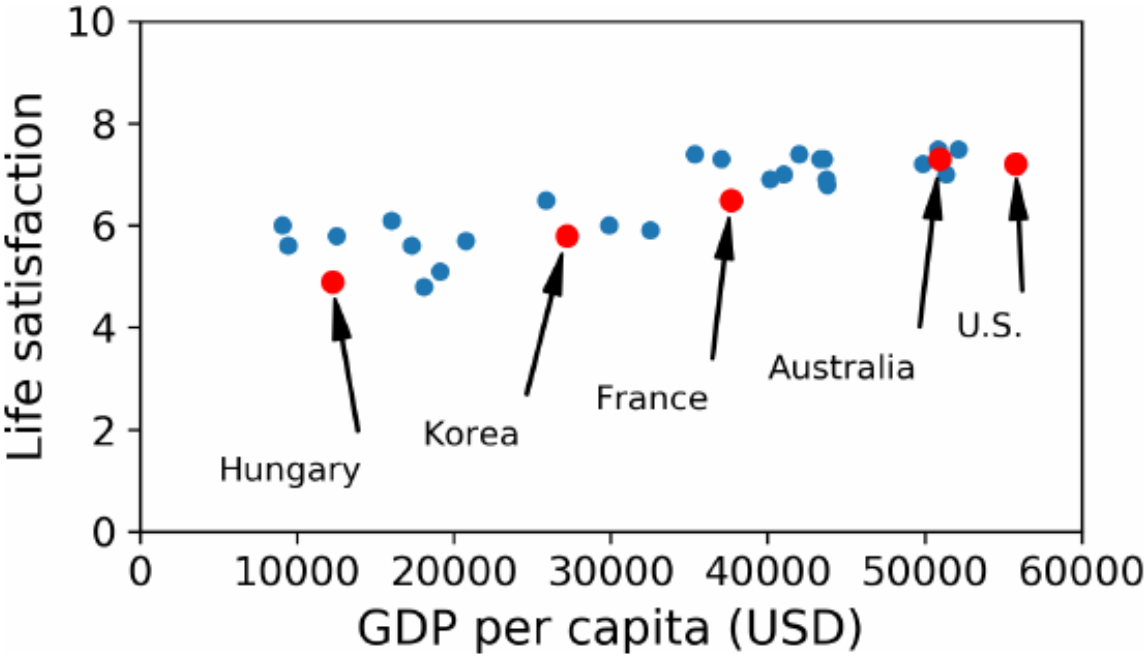

Figure 3: satisfaction v GDP per capita (Géron 2019)

- Notice the somewhat linear trend suggesting a linear relationship
- This is model selection - we assume a linear model

$$\text{life\_satisfaction} = \theta_0 + \theta_1 \times \text{GDP per capita} \tag{7}$$

Linear Regression

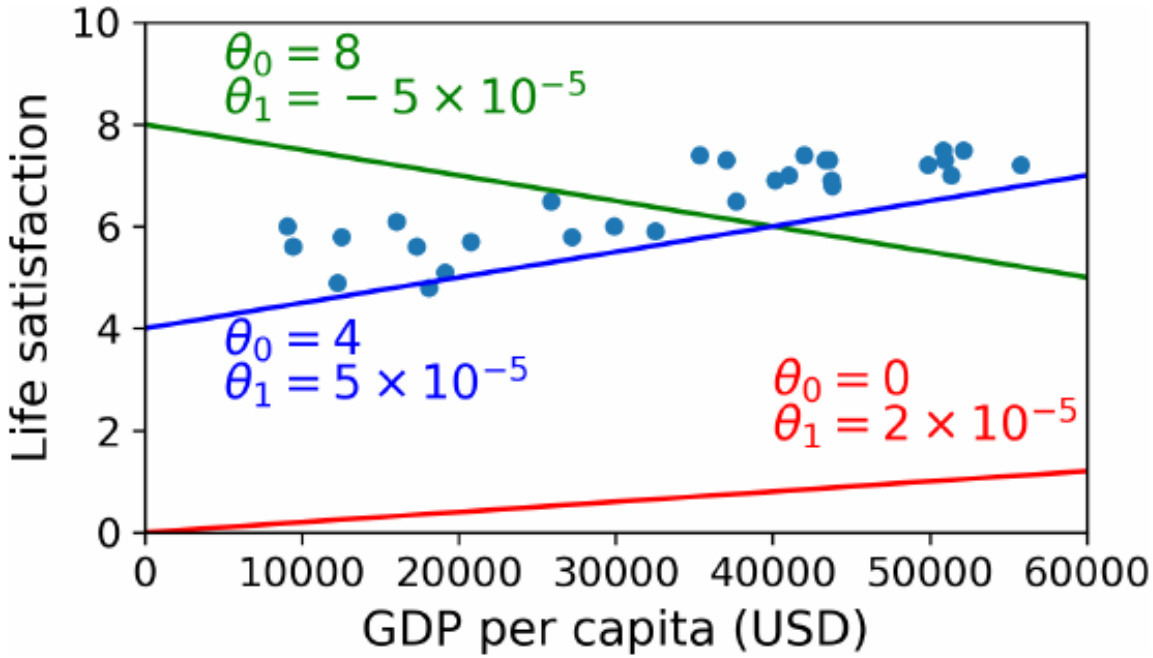

Figure 4: Different values of parameter lead to different models (Géron 2019)

Linear Regression

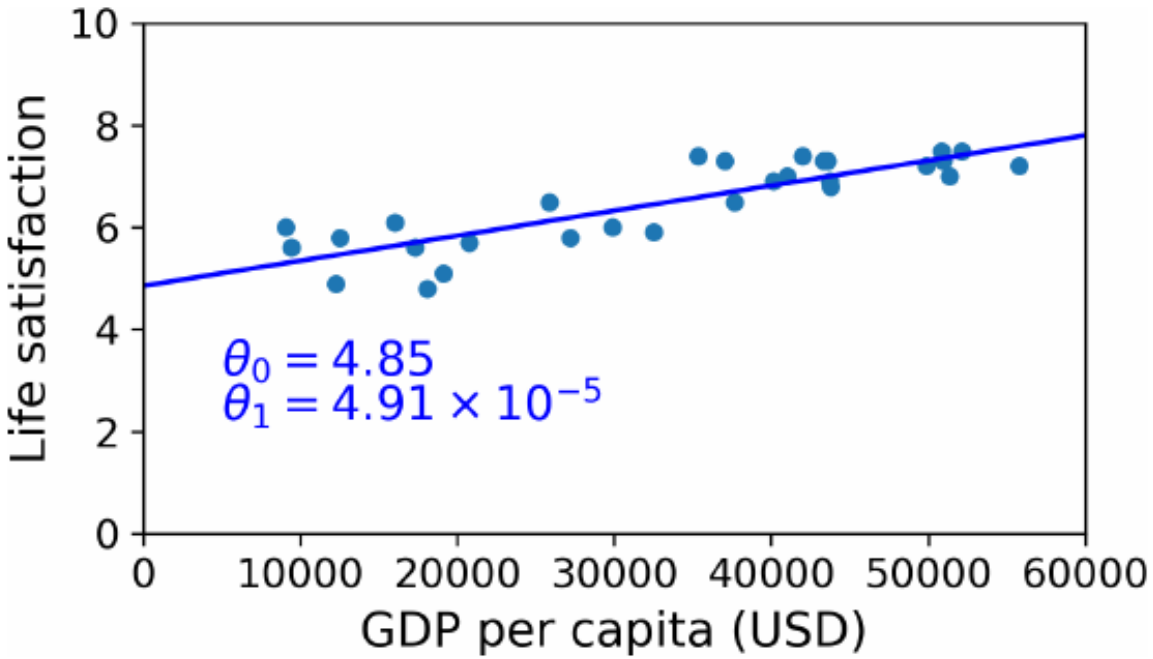

Figure 5: Best linear model fit to data (Géron 2019)

$$\text{life\_satisfaction} = 4.85 + 4.91 \times 10^{-5} \times \text{GDP per capita} \tag{8}$$

Linear Regression

1. There are several software libraries/frameworks available for implementation
2. Scikit-Learn (very popular Python-based) library
3. TensorFlow (Python-based framework by Google)
4. Pytorch (Python-based framework by Facebook)

Scikit-Learn implementation of linear regression

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import sklearn.linear_model
# Load the data
oecd_bli = pd.read_csv("oecd_bli_2015.csv", thousands=',')
gdp_per_capita = pd.read_csv("gdp_per_capita.csv",
thousands=',',delimiter='\t',
encoding='latin1', na_values="n/a")
```

Linear Regression

```
# Prepare the data
country_stats = prepare_country_stats(oecd_bli, gdp_per_capita)
X = np.c_[country_stats["GDP per capita"]]
y = np.c_[country_stats["Life satisfaction"]]

# Visualize the data
country_stats.plot(kind='scatter', x="GDP per capita",
y='Life satisfaction')
plt.show()
# Select a linear model
model = sklearn.linear_model.LinearRegression()
# Train the model
model.fit(X, y)
# Make a prediction for Cyprus
X_new = [[22587]] # Cyprus's GDP per capita
print(model.predict(X_new)) # outputs [[ 5.96242338]]
```

- There is no life satisfaction data for Cyprus, but our model can help us predict a life satisfaction index:
- `X_new = [[22587]]`
- life_satisfaction $= 4.85 + 4.91 \times 10^{-5} \times 22587 = 5.96$

Kernel Ridge regression

- Formulation is somewhat similar to linear regression; consider mapping from input space to a feature space but with a kernel $\Phi(\cdot)$
- This model gives better theoretical guarantees and improved performance in practice (there is a theorem that supports this claim) The optimization problem is written compactly as:

$$\min_{\boldsymbol{W}} F(\boldsymbol{W}) = \lambda ||\boldsymbol{W}||^2 + ||\boldsymbol{X}^T \boldsymbol{W} - \boldsymbol{Y}||^2 \tag{9}$$

  where λ is a positive parameter that determines the trade-off between the regularization term $||\boldsymbol{W}||^2$ and the empirical mean squared error; $\boldsymbol{X} \in \mathbb{R}^{N \times m}$ is the matrix of feature vectors, $\boldsymbol{X} = [\Phi(x_1), \ldots, \Phi(x_m)]$ and $\boldsymbol{W}$ and $\boldsymbol{Y}$ are as defined previously (see Equation (5))
- Optimization problem of Equation (9) is convex and differentiable with a global minimum if and only if

$$\nabla F(\boldsymbol{W}) = 0 \Leftrightarrow (\boldsymbol{X}\boldsymbol{X}^T + \lambda \boldsymbol{I})\boldsymbol{W} = \boldsymbol{X}\boldsymbol{Y} \Leftrightarrow \boldsymbol{W} = (\boldsymbol{X}\boldsymbol{X}^T + \lambda \boldsymbol{I})^{-1}\boldsymbol{X}\boldsymbol{Y} \tag{10}$$

  $\boldsymbol{X}\boldsymbol{X}^T + \lambda \boldsymbol{I}$ is always invertible [2]
- Alternative formulation of the kernel ridge regression

$$\min_{\boldsymbol{w}} \sum_{1}^{m} (\boldsymbol{w} \cdot \boldsymbol{\Phi}(x_i) - y_i)^2 \qquad \text{subject to} \qquad ||\boldsymbol{w}||^2 \leq \Lambda^2 \tag{11}$$

[2] because its eigenvalues are sum of non-negative eigenvalues of positive semi-definite matrix $\boldsymbol{X}\boldsymbol{X}^T$ and $\lambda > 0$

Kernel Ridge regression

Some properties of ridge regression:

- In essence it is a model selection method in which the ridge parameter λ helps select/weight the variables appropriately.
- The choice of the ridge parameter is a tool to balance the “bias-variance” trade-off. The larger the value of λ the larger the bias and the smaller the variance. The parameter can be determined using cross validation technique.
- The ridge regression estimator is a shrinkage estimator that shrinks the least square weights toward zero.
- It can be used with positive definite symmetric (PDS) kernels and hence can be extended to non-linear regression and more general feature spaces.

Ridge Regression

Example of ridge regression implementation with Scikit-Learn

```
from sklearn.linear_model import Ridge
ridge_reg = Ridge(alpha=1, solver="cholesky")
ridge_reg.fit(X, y)
ridge_reg.predict([[1.5]])
array([[1.55071465]])
```

- In this case the closed-form solution has been used and cholesky factorisation method chosen as the solver.

Ridge Regression

Example of ridge regression implementation with Scikit-Learn

```
sgd_reg = SGDRegressor(penalty="l2")
sgd_reg.fit(X, y.ravel())
sgd_reg.predict([[1.5]])
array([1.47012588])
```

- In this case the stochastic gradient optimisation has been used to find the optimum coefficients.
- The use case is when you have a lot of data items and features. In general this is what is used when training neural networks.
- We will revisit gradient descent algorithm later in the lectures.

Lasso Regression

- Our goal in prediction is to choose an economical (parsimonious) model that will balance the bias-variance trade-off.
- What variables are important for the prediction?
- Variable selection is another method of solving this problem
  1. Backward elimination: Begin with full set of variables and drop at each step the variable whose $F-$ratio is smallest:

$$F = \frac{(RSS_0 - RSS_1)/(df_0 - df_1)}{RSS_1/df_1} \tag{12}$$

  $RSS_0 = \sum_i (y_i - \hat{y}_i)^2$ computed with reduced model and with degree of freedom df_0 ;
  $RSS_1 = \sum_i (y_i - \hat{y}_i)^2$ computed with larger model and with degree of freedom df_1 ;
  The reduced model is refitted and the iteration is repeated.
  2. Forward selection: Begin with an empty set of variables and select the variable from the list that gives the largest F value[3].

[3] More on feature selection later in the lecture series.

Lasso Regression

- Lasso is a short for Least absolute shrinkage and selection operator
- Essentially it combines variable subset selection and shrinkage to improve accuracy
- This model does not allow an easy use of a PDS kernel; assume input space $\mathcal{X}$, is a subset of $\mathbb{R}^N$
- Consider a family of linear hypotheses

$$\mathcal{H} = \{x \mapsto \mathbf{w}.\boldsymbol{x} + b : \boldsymbol{w} \in \mathbb{R}^N, b \in \mathbb{R}\} \tag{13}$$

- Given a sample set $\mathcal{S} = ((x_1, y_1), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \mathcal{Y})^m$
- Lasso regression seeks an hypothesis in $\mathcal{H}$ that minimizes empirical squared error with a regularization term depending on the norm of the weight vector;
- Lasso uses L_1 norm instead of L_2 norm (ridge regression - see Equations (9) and (11)):

$$\min_{\boldsymbol{w},b} F(\boldsymbol{w}, b) = \lambda ||\boldsymbol{w}||_1 + \sum_{i=1}^{m} (\mathbf{w}.\boldsymbol{x}_i + b - y_i)^2 \tag{14}$$

- Equivalently:
  $\min_{\boldsymbol{w},b} \sum_{i=1}^{m} (\mathbf{w}.\boldsymbol{x}_i + b - y_i)^2$ subject to $||\boldsymbol{w}||_1 \leq \Lambda_1$; It is a Quadratic Program solvable by QP solvers

Lasso Regression

- Key property of Lasso is that it leads to sparse solution of $\boldsymbol{w}$ - one with few non-zero components
- Sparsity is encouraged by L_1 norm

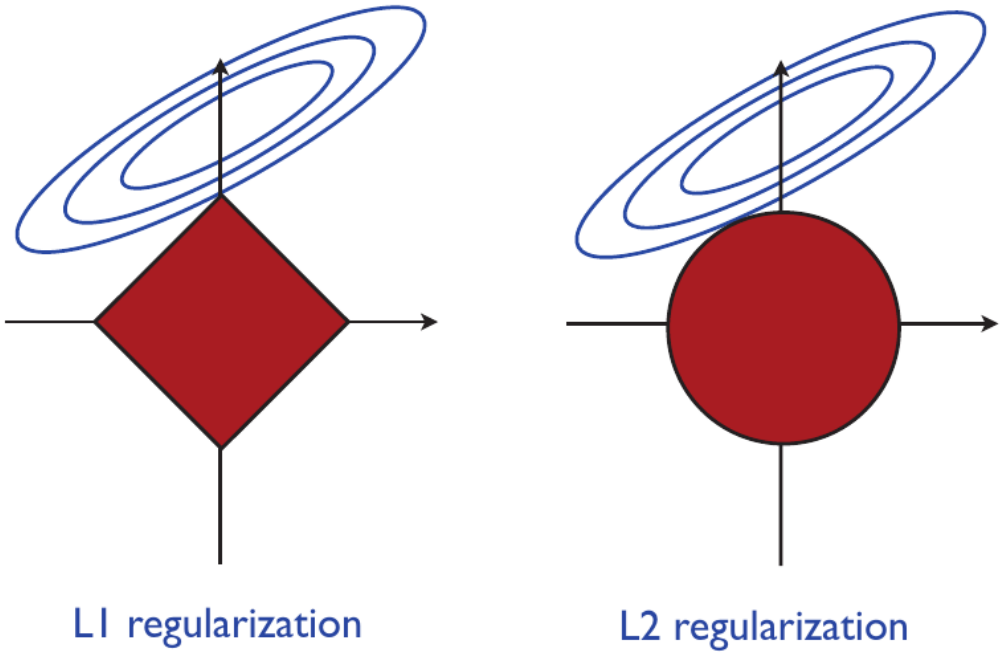

Figure 6: Comparision of Lasso and ridge regression solutions (Mohri et al. 2012)

- Objective function is quadratic and contours are ellipsoids (See Figure 6); Lasso solution is intersection with L_1 ball occurring at corner where some coordinates are zero, hence it promotes sparsity; contrast with L_2 regularization

Lasso Regression

Example of Lasso regression implementation with Scikit-Learn

```
from sklearn.linear_model import Lasso
lasso_reg = Lasso(alpha=0.1)
lasso_reg.fit(X, y)
lasso_reg.predict([[1.5]])
array([1.53788174])
```

Elastic net regularization

- Elastic net linear regression uses the penalties from both the lasso and ridge techniques to regularize regression models.

$$\min_{\boldsymbol{w},b} F(\boldsymbol{w}, b) = \lambda ||\boldsymbol{w}||_1 + \beta ||\boldsymbol{W}||_2^2 + \sum_{i=1}^{m} (\mathbf{w}.\boldsymbol{x}_i + b - y_i)^2 \tag{15}$$

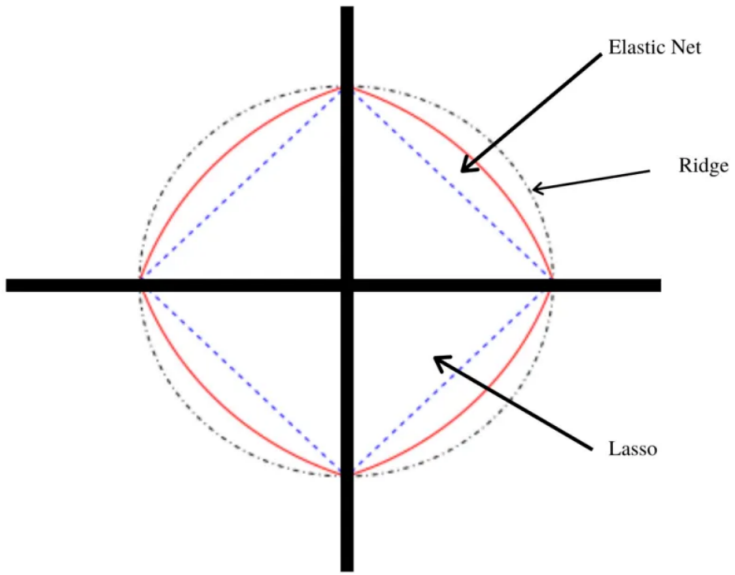

Figure 7: The ball of the various penalty norms

- Elastic net method estimator, involve two stages.
  - It first finds the ridge regression coefficients and then conducts the second step by using a lasso sort of shrinkage of the coefficients.

Elastic Net Regression

Partitioned feature space

In some situations, the feature space may be naturally partitioned into subsets, and it may be desirable to find a sparse solution that selects or omits entire subsets of features. A natural norm in this setting is the group or mixed norm $L2, 1$, which is a combination of the L_1 and L_2 norms (Mohri et al. 2012).

Elastic-net Regression

Example of Elastic-Net regression implementation with Scikit-Learn

```
from sklearn.linear_model import ElasticNet
elastic_net = ElasticNet(alpha=0.1, l1_ratio=0.5)
elastic_net.fit(X, y)
elastic_net.predict([[1.5]])
array([1.54333232])
```

Dealing with data

Non-representative Data

1. For the created model to generalise, training data must be representative of the new cases (or production data) to be used when deployed.
2. This is the same as saying that the underlying probability distribution of both training data and production data must be the same.
3. Possible solution is to ensure proper sampling - not too few samples (avoid sampling noise); representative sampling (avoid sampling bias)

Poor Quality Data

1. Sources of poor quality include: errors, outliers, and noise.
2. Outliers: discard of fix manually
3. Missing values: if there are too many for a feature it may be good to ignore the feature; replace missing values with median (or average); train one model with feature and one without.

Dealing with data

Irrelevant Features

1. These features that do not add any new information to model creation
2. Feature engineering can fix the problem of irrelevant features
3. Feature selection - from available features, select the most useful ones for model training; Regularisation could be used to select relevant features (Lasso or elastic-net regularization)
4. Feature extraction - combining features to produce more useful ones. Dimensionality reduction is an example.

Essential further reading

Chapter 2 of the text by Géron (Géron 2019) is an excellent reference on end-to-end design of a machine learning system.

Model Selection and variance-bias Trade-off

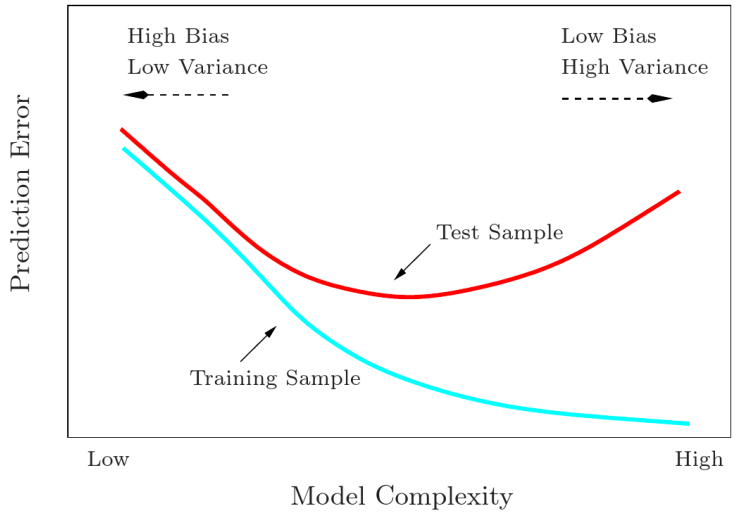

Figure 8: Typical training and test error behaviour as a function of model complexity (Hastie et al. 2001). Training error decreases as model complexity increases; model overfits leading to poor generalization and large variance. Test error increases if model is not complex enough; model underfits; lead to large bias and poor generalization. So there is a bias-variance trade-off.

- The prediction error has three parts:
  1. irreducible error (variance of the new test target) which is beyond our control
  2. Bias component - the squared difference between true mean of the estimate and the expected value of the estimate
  3. Variance component - variance of an average

Back to main

Bibliography

Alpaydin, E. (2010), *Introduction to Machine Learning*, second edn, The MIT Press, Cambridge Massachusetts.

Duda, R. O., Hart, P. E. & Stork, D. G. (2001), *Pattern Classification*, Second edn, John Wiley and Sons.

Géron, A. (2019), *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, Second edn, O'Reilly.

Goodfellow, I., Bengio, Y. & Courville, A. (2016), *Deep Learning*, MIT Press.

Hastie, T., Tibshirani, R. & Friedman, J. (2001), *The Elements of Statistical Learning - Data Mining, Inference and Prediction*, Springer Science+Business Media LLC.

Izenman, A. J. (2008), *Modern Multivariate Statistical Techniques - Regression, Classification and Manifold Learning*, Springer Science+Business Media LLC.

Kellleher, J. D., Namee, B. M. & D'Arcy, A. (2015), *Fundamentals of Machine Learning for Predictive Data Analytics - Algorithms, Worked Examples and Case Studies*, The MIT Press, Cambridge Massachusetts.

Mitchell, T. M. (1997), *Machine Learning*, WCB McGraw-Hill.

Mohri, M., Rostamizadeh, A. & Talwalkar, A. (2012), *Foundations of Machine Learning*, MIT Press.

Webb, A. (2002), *Statistical Pattern Recognition*, Second edn, John Wiley and Sons.