Introduction to Deep Learning

Arijit Mondal
Dept. of Computer Science & Engineering
Indian Institute of Technology Patna
arijit@iitp.ac.in

Feature Engineering

Machine Learning

- A form of applied statistics with
  - Increased emphasis on the use of computers to statistically estimate complicated function
  - Decreased emphasis on proving confidence intervals around these functions
- Two primary approaches
  - Frequentist estimators
  - Bayesian inference

Types of Machine Learning Problems

- Supervised
- Unsupervised
- Other variants
  - Reinforcement learning
  - Semi-supervised

Learning algorithm

- A ML algorithm is an algorithm that is able to learn from data
- Mitchelle (1997)
  - A computer program is said to learn from experience E with respect to some class of task T and performance measure P, if its performance at task in T as measured by P, improves with experience E.

Task

- A ML tasks are usually described in terms of how ML system should process an example
  - Example is a collection of features that have been quantitatively measured from some objects or events that we want the learning system process
    - Represented as $\boldsymbol{x} \in \mathbb{R}^n$ where x_i is a feature
    - Feature of an image — pixel values

Common ML Task

- Classification
  - Need to predict which of the k categories some input belong to
  - Need to have a function $f : \mathbb{R}^n \rightarrow \{1, 2, \ldots, k\}$
  - $y = f(\boldsymbol{x})$ input $\boldsymbol{x}$ is assigned category identified by y
  - Examples
    - Object identification
    - Face recognition
- Regression
  - Need to predict numeric value for some given input
  - Need to have a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$
  - Examples
    - Energy consumption
    - Amount of insurance claim

Common ML Task (contd.)

- Classification with missing inputs
  - Need to have a set of functions
  - Each function corresponds to classifying $\boldsymbol{x}$ with different subset of inputs missing
  - Examples
    - Medical diagnosis (expensive or invasive)

Common ML Task (contd.)

- Classification with missing inputs
  - Need to have a set of functions
  - Each function corresponds to classifying $\boldsymbol{x}$ with different subset of inputs missing
  - Examples
    - Medical diagnosis (expensive or invasive)
- Transcription
  - Need to convert relatively unstructured data into discrete, textual form
    - Optical character recognition
    - Speech recognition

Common ML Task (contd.)

- Classification with missing inputs
  - Need to have a set of functions
  - Each function corresponds to classifying $\boldsymbol{x}$ with different subset of inputs missing
  - Examples
    - Medical diagnosis (expensive or invasive)
- Transcription
  - Need to convert relatively unstructured data into discrete, textual form
    - Optical character recognition
    - Speech recognition
- Machine translation
  - Conversion of sequence of symbols in one language to some other language
    - Natural language processing (English to Spanish conversion)

Common ML Task (contd.)

- Structured output
  - Output is a vector with important relationship between the different elements
    - Mapping natural language sentence into a tree that describes grammatical structure
    - Pixel based image segmentation (eg. identify roads)

Common ML Task (contd.)

- Structured output
  - Output is a vector with important relationship between the different elements
    - Mapping natural language sentence into a tree that describes grammatical structure
    - Pixel based image segmentation (eg. identify roads)
- Anamoly detection
  - Observes a set of events or objects and flags if some of them are unusual
    - Fraud detection in credit card

Common ML Task (contd.)

- Structured output
  - Output is a vector with important relationship between the different elements
    - Mapping natural language sentence into a tree that describes grammatical structure
    - Pixel based image segmentation (eg. identify roads)
- Anamoly detection
  - Observes a set of events or objects and flags if some of them are unusual
    - Fraud detection in credit card
- Synthesis and sampling
  - Generate new example similar to past examples
    - Useful for media application
    - Text to speech

Performance measure

- Accuracy is one of the key measures
  - The proportion of examples for which the model produces correct outputs
  - Similar to error rate
    - Error rate often referred as expected 0-1 loss
- Mostly interested how ML algorithm performs on unseen data
- Choice of performance measure may not be straight forward
  - Transcription
    - Accuracy of the system at transcribing entire sequence
    - Any partial credit for some elements of the sequence are correct

Experience

- Kind of experience allowed during learning process
  - Supervised
  - Unsupervised

Supervised learning

- Allowed to use labeled dataset
- Example — Iris
  - Collection of measurements of different parts of Iris plant
  - Each plant means each example
  - Features
    - Sepal length/width, petal length/width
    - Also record which species the plant belong to

Supervised learning (contd.)

- A set of labeled examples $\langle x_1, x_2, \ldots, x_n, y \rangle$
  - x_i are input variables
  - y output variable
- Need to find a function $f : X_1 \times X_2 \times \ldots X_n \rightarrow Y$
- Goal is to minimize error/loss function
  - Like to minimize over all dataset
  - We have limited dataset

Unsupervised learning

- Learns useful properties of the structure of data set
- Unlabeled data
  - Tries to learn entire probability distribution that generated the dataset
  - Examples
    - Clustering, dimensionality reduction

Supervised vs Unsupervised learning

- Unsupervised attempts to learn implicitly or explicitly probability distribution of $p(x)$
- Supervised tries to predict y from x ie. $p(y|x)$

Supervised vs Unsupervised learning

- Unsupervised attempts to learn implicitly or explicitly probability distribution of $p(\boldsymbol{x})$
- Supervised tries to predict $\boldsymbol{y}$ from $\boldsymbol{x}$ ie. $p(\boldsymbol{y}|\boldsymbol{x})$
- Unsupervised learning can be decomposed as supervised learning

$$p(\boldsymbol{x}) = \prod_{i=1}^{n} p(x_i | x_1, x_2, \ldots, x_{i-1})$$

Supervised vs Unsupervised learning

- Unsupervised attempts to learn implicitly or explicitly probability distribution of $p(\boldsymbol{x})$
- Supervised tries to predict $\boldsymbol{y}$ from $\boldsymbol{x}$ ie. $p(\boldsymbol{y}|\boldsymbol{x})$
- Unsupervised learning can be decomposed as supervised learning

$$p(\boldsymbol{x}) = \prod_{i=1}^{n} p(x_i|x_1, x_2, \ldots, x_{i-1})$$

- Solving supervised learning using traditional unsupervised learning

$$p(y|\boldsymbol{x}) = \frac{p(\boldsymbol{x}, y)}{\sum_{y'} p(\boldsymbol{x}, y')}$$

Linear regression

- Prediction of the value of a continuous variable
  - Example — price of a house, solar power generation in photo-voltaic cell, etc.

Linear regression

- Prediction of the value of a continuous variable
  - Example — price of a house, solar power generation in photo-voltaic cell, etc.
- Takes a vector $\boldsymbol{x} \in \mathbb{R}^n$ and predict scalar $y \in \mathbb{R}$
  - Predicted value will be represented as $\hat{y} = \boldsymbol{w}^T\boldsymbol{x}$ where $\boldsymbol{w}$ is a vector of parameters
    - x_i receives positive weight — Increasing the value of the feature will increase the value of y
    - x_i receives negative weight — Increasing the value of the feature will decrease the value of y
    - Weight value is very high/large — Large effect on prediction

Performance

- Assume, we have m examples not used for training
  - This is known as test set

Performance

- Assume, we have m examples not used for training
  - This is known as test set
- Design matrix of inputs is $\boldsymbol{X}^{(\text{test})}$ and target output is a vector $\boldsymbol{y}^{(\text{test})}$
  - Performance is measured by Mean Square Error (MSE)

$$\text{MSE}_{(\text{test})} = \frac{1}{m} \sum_i \left(\hat{\boldsymbol{y}}^{(\text{test})} - \boldsymbol{y}^{(\text{test})} \right)_i^2 = \frac{1}{m} \|\hat{\boldsymbol{y}}^{(\text{test})} - \boldsymbol{y}^{(\text{test})}\|_2^2$$

  - Error increases when the Euclidean distance between target and prediction increases

Performance

- Assume, we have m examples not used for training
  - This is known as test set
- Design matrix of inputs is $\boldsymbol{X}^{(\text{test})}$ and target output is a vector $\boldsymbol{y}^{(\text{test})}$
  - Performance is measured by Mean Square Error (MSE)

$$\text{MSE}_{(\text{test})} = \frac{1}{m}\sum_i \left(\hat{\boldsymbol{y}}^{(\text{test})} - \boldsymbol{y}^{(\text{test})}\right)_i^2 = \frac{1}{m}\|\hat{\boldsymbol{y}}^{(\text{test})} - \boldsymbol{y}^{(\text{test})}\|_2^2$$

  - Error increases when the Euclidean distance between target and prediction increases
- The learning algorithm is allowed to gain experience from training set $(\boldsymbol{X}^{(\text{train})}, \boldsymbol{y}^{(\text{train})})$
- One of the common ideas is to minimize $\text{MSE}_{(\text{train})}$ for training set

Minimization of MSE

- We have the following now

$$\nabla_w \mathrm{MSE}_{(\text{train})} = 0$$

Minimization of MSE

- We have the following now

$$\nabla_w \mathrm{MSE}_{(\mathrm{train})} = 0$$

$$\Rightarrow \quad \nabla_w \frac{1}{m} \| \hat{\boldsymbol{y}}^{(\mathrm{train})} - \boldsymbol{y}^{(\mathrm{train})} \|_2^2 = 0$$

Minimization of MSE

- We have the following now

$$
\begin{aligned}
& \nabla_w \mathrm{MSE}_{(\mathrm{train})} = 0 \\
\Rightarrow \quad & \nabla_w \frac{1}{m} \|\hat{\boldsymbol{y}}^{(\mathrm{train})} - \boldsymbol{y}^{(\mathrm{train})}\|_2^2 = 0 \\
\Rightarrow \quad & \frac{1}{m} \nabla_w \|\boldsymbol{X}^{(\mathrm{train})} \boldsymbol{w} - \boldsymbol{y}^{(\mathrm{train})}\|_2^2 = 0
\end{aligned}
$$

Minimization of MSE

- We have the following now

$$
\begin{aligned}
& \nabla_w \mathrm{MSE}_{(\mathrm{train})} = 0 \\
\Rightarrow \quad & \nabla_w \frac{1}{m} \|\hat{\boldsymbol{y}}^{(\mathrm{train})} - \boldsymbol{y}^{(\mathrm{train})}\|_2^2 = 0 \\
\Rightarrow \quad & \frac{1}{m} \nabla_w \|\boldsymbol{X}^{(\mathrm{train})} \boldsymbol{w} - \boldsymbol{y}^{(\mathrm{train})}\|_2^2 = 0 \\
\Rightarrow \quad & \nabla_w (\boldsymbol{X}^{(\mathrm{train})} \boldsymbol{w} - \boldsymbol{y}^{(\mathrm{train})})^T (\boldsymbol{X}^{(\mathrm{train})} \boldsymbol{w} - \boldsymbol{y}^{(\mathrm{train})}) = 0
\end{aligned}
$$

Minimization of MSE

- We have the following now

$$
\begin{aligned}
& \nabla_w \text{MSE}_{(\text{train})} = 0 \\
\Rightarrow \quad & \nabla_w \frac{1}{m} \|\hat{\boldsymbol{y}}^{(\text{train})} - \boldsymbol{y}^{(\text{train})}\|_2^2 = 0 \\
\Rightarrow \quad & \frac{1}{m} \nabla_w \|\boldsymbol{X}^{(\text{train})} \boldsymbol{w} - \boldsymbol{y}^{(\text{train})}\|_2^2 = 0 \\
\Rightarrow \quad & \nabla_w (\boldsymbol{X}^{(\text{train})} \boldsymbol{w} - \boldsymbol{y}^{(\text{train})})^T (\boldsymbol{X}^{(\text{train})} \boldsymbol{w} - \boldsymbol{y}^{(\text{train})}) = 0 \\
\Rightarrow \quad & \nabla_w (\boldsymbol{w}^T \boldsymbol{X}^{(\text{train})T} \boldsymbol{X}^{(\text{train})} \boldsymbol{w} - 2\boldsymbol{w}^T \boldsymbol{X}^{(\text{train})T} \boldsymbol{y}^{(\text{train})} - \boldsymbol{y}^{(\text{train})T} \boldsymbol{y}^{(\text{train})}) = 0
\end{aligned}
$$

Minimization of MSE

- We have the following now

$$
\begin{aligned}
& \nabla_w \text{MSE}_{(\text{train})} = 0 \\
\Rightarrow\quad & \nabla_w \frac{1}{m} \|\hat{\boldsymbol{y}}^{(\text{train})} - \boldsymbol{y}^{(\text{train})}\|_2^2 = 0 \\
\Rightarrow\quad & \frac{1}{m} \nabla_w \|\boldsymbol{X}^{(\text{train})}\boldsymbol{w} - \boldsymbol{y}^{(\text{train})}\|_2^2 = 0 \\
\Rightarrow\quad & \nabla_w (\boldsymbol{X}^{(\text{train})}\boldsymbol{w} - \boldsymbol{y}^{(\text{train})})^T (\boldsymbol{X}^{(\text{train})}\boldsymbol{w} - \boldsymbol{y}^{(\text{train})}) = 0 \\
\Rightarrow\quad & \nabla_w (\boldsymbol{w}^T \boldsymbol{X}^{(\text{train})T} \boldsymbol{X}^{(\text{train})} \boldsymbol{w} - 2\boldsymbol{w}^T \boldsymbol{X}^{(\text{train})T} \boldsymbol{y}^{(\text{train})} - \boldsymbol{y}^{(\text{train})T} \boldsymbol{y}^{(\text{train})}) = 0 \\
\Rightarrow\quad & 2\boldsymbol{X}^{(\text{train})T} \boldsymbol{X}^{(\text{train})} \boldsymbol{w} - 2\boldsymbol{X}^{(\text{train})T} \boldsymbol{y}^{(\text{train})} = 0
\end{aligned}
$$

Minimization of MSE

- We have the following now

$$
\begin{aligned}
& \nabla_w \text{MSE}_{(\text{train})} = 0 \\
\Rightarrow \quad & \nabla_w \frac{1}{m} \|\hat{\boldsymbol{y}}^{(\text{train})} - \boldsymbol{y}^{(\text{train})}\|_2^2 = 0 \\
\Rightarrow \quad & \frac{1}{m} \nabla_w \|\boldsymbol{X}^{(\text{train})} \boldsymbol{w} - \boldsymbol{y}^{(\text{train})}\|_2^2 = 0 \\
\Rightarrow \quad & \nabla_w (\boldsymbol{X}^{(\text{train})} \boldsymbol{w} - \boldsymbol{y}^{(\text{train})})^T (\boldsymbol{X}^{(\text{train})} \boldsymbol{w} - \boldsymbol{y}^{(\text{train})}) = 0 \\
\Rightarrow \quad & \nabla_w (\boldsymbol{w}^T \boldsymbol{X}^{(\text{train})T} \boldsymbol{X}^{(\text{train})} \boldsymbol{w} - 2\boldsymbol{w}^T \boldsymbol{X}^{(\text{train})T} \boldsymbol{y}^{(\text{train})} - \boldsymbol{y}^{(\text{train})T} \boldsymbol{y}^{(\text{train})}) = 0 \\
\Rightarrow \quad & 2\boldsymbol{X}^{(\text{train})T} \boldsymbol{X}^{(\text{train})} \boldsymbol{w} - 2\boldsymbol{X}^{(\text{train})T} \boldsymbol{y}^{(\text{train})} = 0 \\
\Rightarrow \quad & \boldsymbol{w} = (\boldsymbol{X}^{(\text{train})T} \boldsymbol{X}^{(\text{train})})^{-1} \boldsymbol{X}^{(\text{train})} \boldsymbol{y}^{(\text{train})}
\end{aligned}
$$

Minimization of MSE

- We have the following now

$$
\begin{aligned}
& \nabla_w \mathrm{MSE}_{(\text{train})} = 0 \\
\Rightarrow\quad & \nabla_w \frac{1}{m} \|\hat{\boldsymbol{y}}^{(\text{train})} - \boldsymbol{y}^{(\text{train})}\|_2^2 = 0 \\
\Rightarrow\quad & \frac{1}{m} \nabla_w \|\boldsymbol{X}^{(\text{train})} \boldsymbol{w} - \boldsymbol{y}^{(\text{train})}\|_2^2 = 0 \\
\Rightarrow\quad & \nabla_w (\boldsymbol{X}^{(\text{train})} \boldsymbol{w} - \boldsymbol{y}^{(\text{train})})^T (\boldsymbol{X}^{(\text{train})} \boldsymbol{w} - \boldsymbol{y}^{(\text{train})}) = 0 \\
\Rightarrow\quad & \nabla_w (\boldsymbol{w}^T \boldsymbol{X}^{(\text{train})T} \boldsymbol{X}^{(\text{train})} \boldsymbol{w} - 2\boldsymbol{w}^T \boldsymbol{X}^{(\text{train})T} \boldsymbol{y}^{(\text{train})} - \boldsymbol{y}^{(\text{train})T} \boldsymbol{y}^{(\text{train})}) = 0 \\
\Rightarrow\quad & 2\boldsymbol{X}^{(\text{train})T} \boldsymbol{X}^{(\text{train})} \boldsymbol{w} - 2\boldsymbol{X}^{(\text{train})T} \boldsymbol{y}^{(\text{train})} = 0 \\
\Rightarrow\quad & \boldsymbol{w} = (\boldsymbol{X}^{(\text{train})T} \boldsymbol{X}^{(\text{train})})^{-1} \boldsymbol{X}^{(\text{train})} \boldsymbol{y}^{(\text{train})}
\end{aligned}
$$

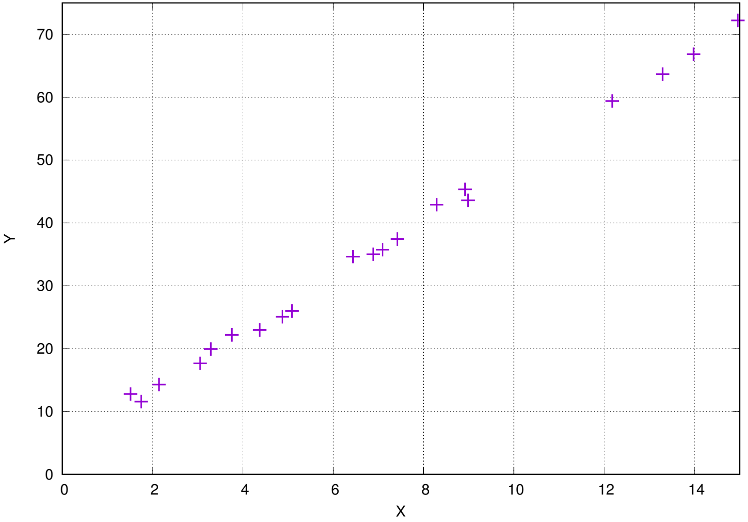
- Linear regression with bias term

$$\hat{y} = [\boldsymbol{w}^T \quad w_0][\boldsymbol{x} \quad 1]^T$$

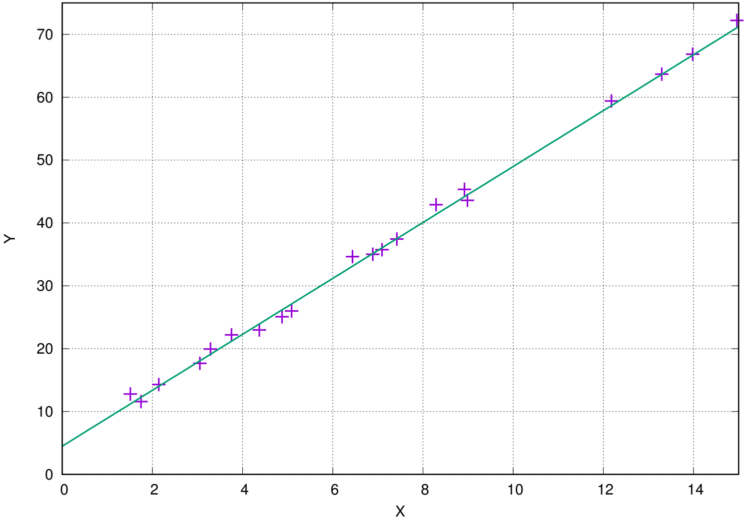
Moore-Penrose Pseudoinverse

- Let $\boldsymbol{A} \in \mathbb{R}^{n \times m}$
- Every $\boldsymbol{A}$ has pseudoinverse $\boldsymbol{A}^{+} \in \mathbb{R}^{m \times n}$ and it is unique
  - $\boldsymbol{A}\boldsymbol{A}^{+}\boldsymbol{A} = \boldsymbol{A}$
  - $\boldsymbol{A}^{+}\boldsymbol{A}\boldsymbol{A}^{+} = \boldsymbol{A}^{+}$
  - $(\boldsymbol{A}\boldsymbol{A}^{+})^{T} = \boldsymbol{A}\boldsymbol{A}^{+}$
  - $(\boldsymbol{A}^{+}\boldsymbol{A})^{T} = \boldsymbol{A}^{+}\boldsymbol{A}$
- $\boldsymbol{A}^{+} = (\boldsymbol{A}^{T}\boldsymbol{A})^{-1}\boldsymbol{A}^{T}$
- Example
  - If $\boldsymbol{A} = [1 \quad 2]^{T}$ then $\boldsymbol{A}^{+} = [\frac{1}{5} \quad \frac{2}{5}]$
  - If $\boldsymbol{A} = \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 1 & 5 \end{bmatrix}$ then $\boldsymbol{A}^{+} = \begin{bmatrix} 0.121212 & 0.515152 & -0.151515 \\ 0.030303 & -0.121212 & 0.212121 \end{bmatrix}$

Regression example



Regression example



Minimization of MSE: Gradient descent

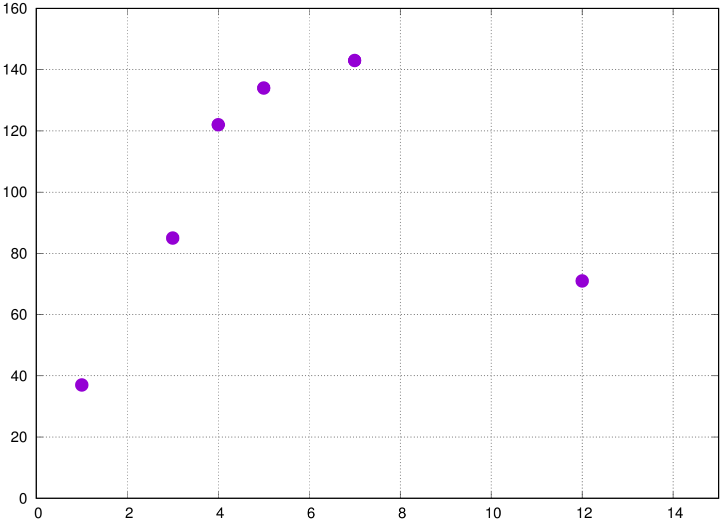
- Assuming $\text{MSE}_{(\text{train})} = J(w_1, w_2)$
- Target is to $\min_{w_1, w_2} J(w_1, w_2)$
- Approach
  - Start with some w_1, w_2
  - Keep modifying w_1, w_2 so that $J(w_1, w_2)$ reduces till the desired accuracy is achieved

Minimization of MSE: Gradient descent

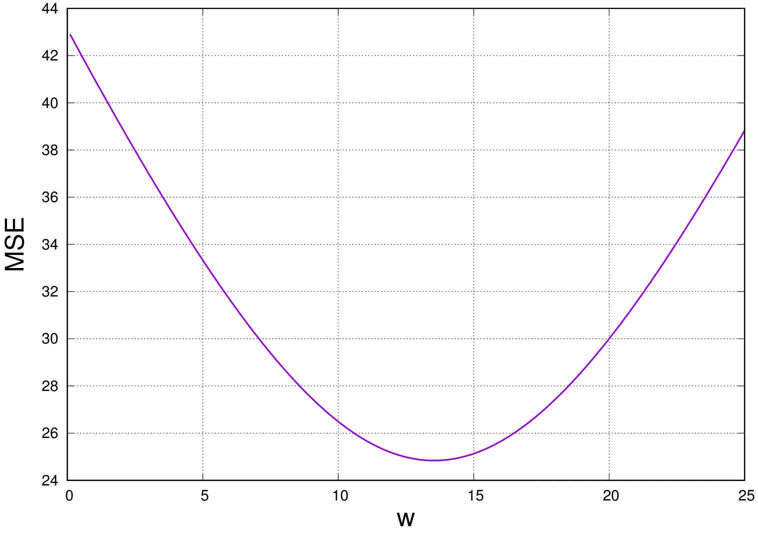
- Assuming $\text{MSE}_{(\text{train})} = J(w_1, w_2)$
- Target is to $\min\limits_{w_1, w_2} J(w_1, w_2)$
- Approach
  - Start with some w_1, w_2
  - Keep modifying w_1, w_2 so that $J(w_1, w_2)$ reduces till the desired accuracy is achieved
- Algorithm
  - Repeat the following until convergence

$$w_j = w_j - \frac{\partial}{\partial w_j} J(w_1, w_2)$$

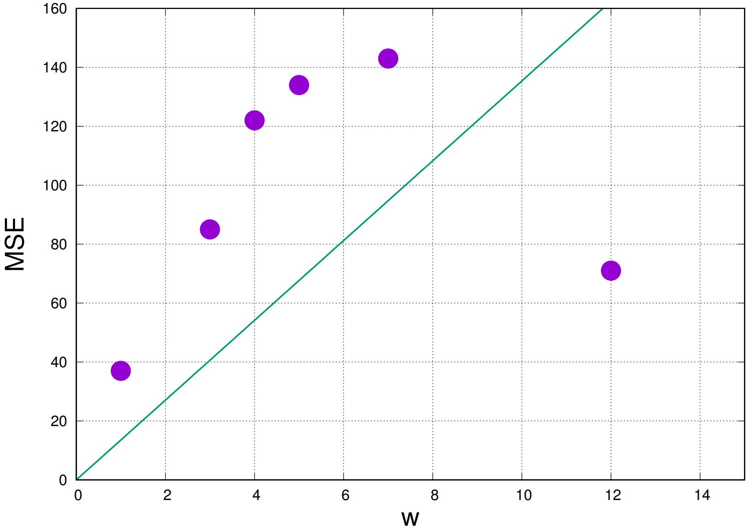
Example



Example



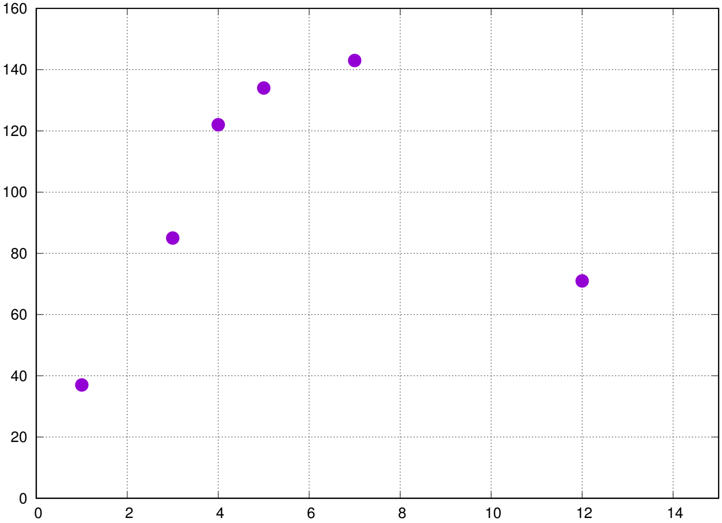
Example



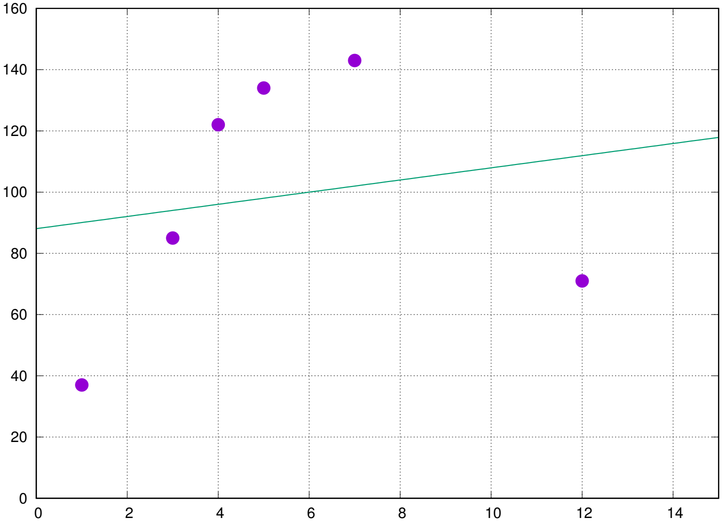
Error

- Training error - Error obtained on a training set
- Generalization error - Error on unseen data
- Data assumed to be independent and identically distributed (iid)
  - Each data set are independent of each other
  - Train and test data are identically distributed
- Expected training and test error will be the same
- It is more likely that the test error is greater than or equal to the expected value of training error
- Target is to make the training error is small. Also, to make the gap between training and test error smaller

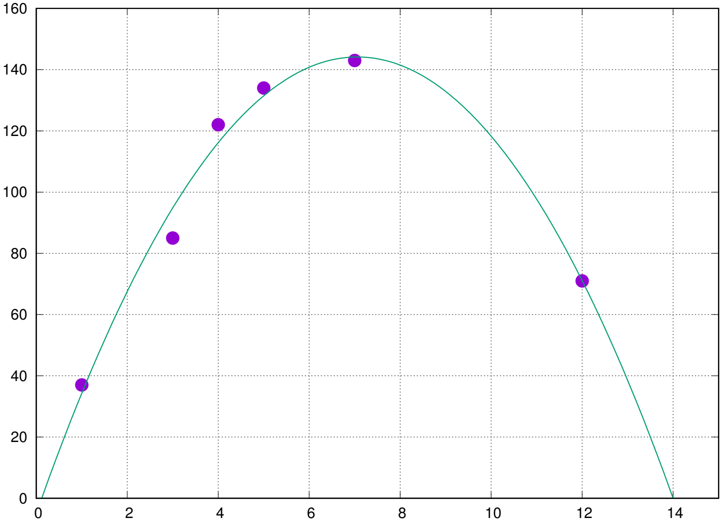
Regression example



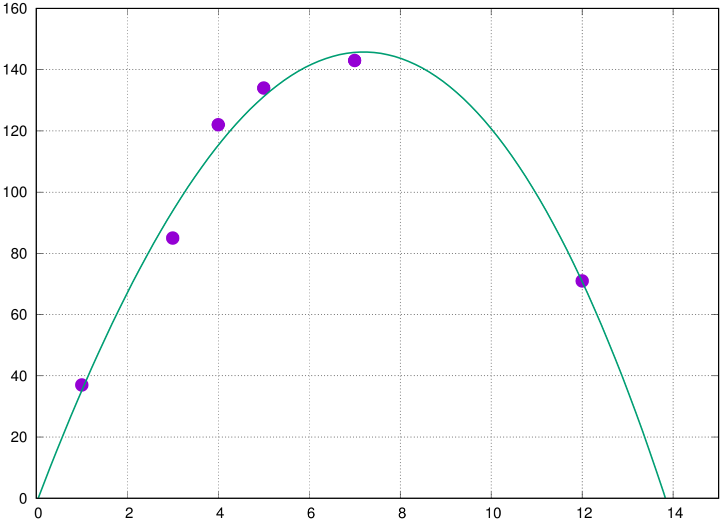
Regression example: degree 1



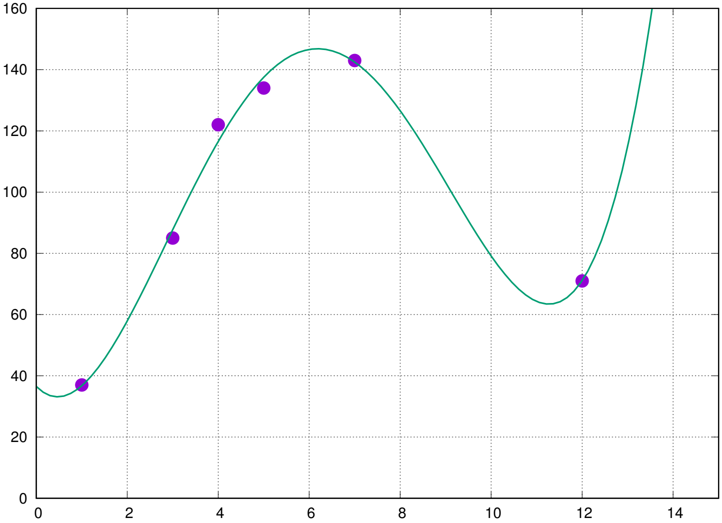
Regression example: degree 2



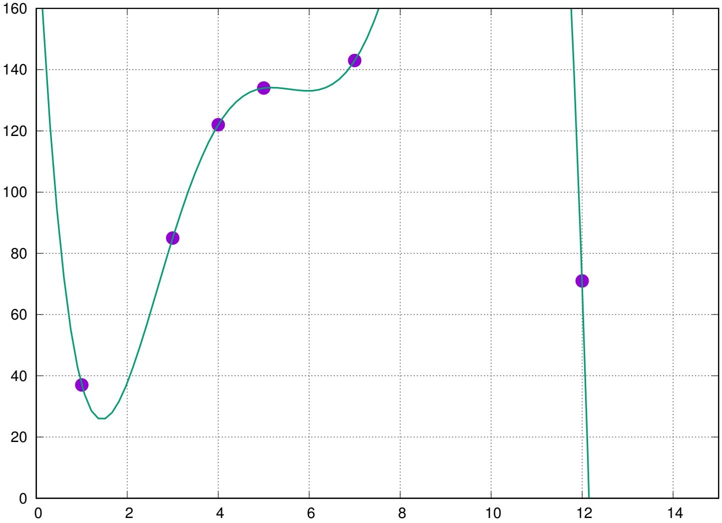
Regression example: degree 3



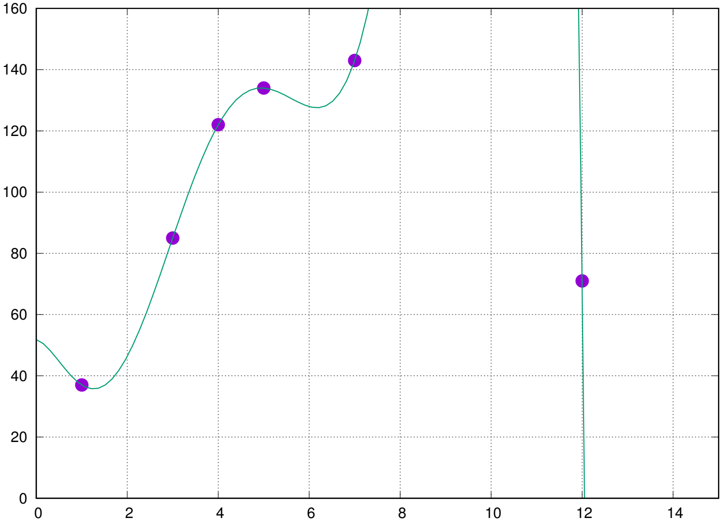
Regression example: degree 4



Regression example: degree 5



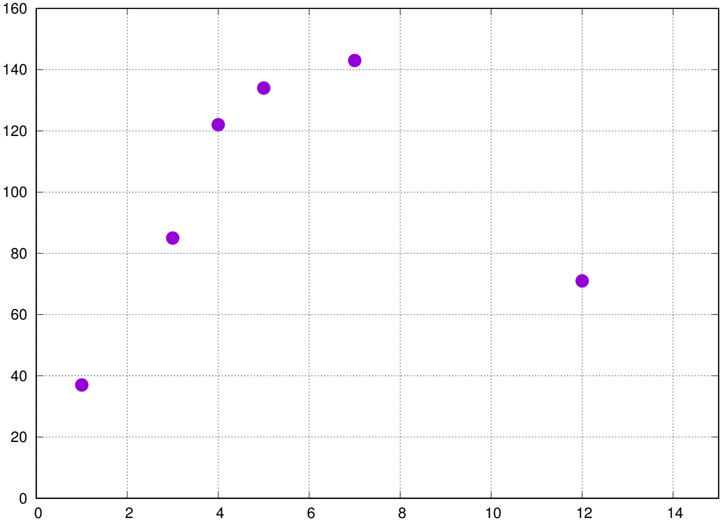
Regression example: degree 6



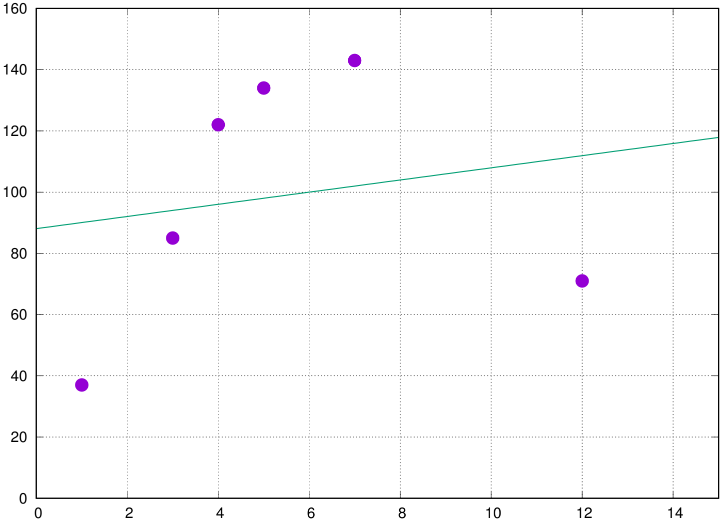
Underfitting & Overfitting

- Underfitting
  - When the model is not able to obtain sufficiently low error value on the training set
- Overfitting
  - When the gap between training set and test set error is too large

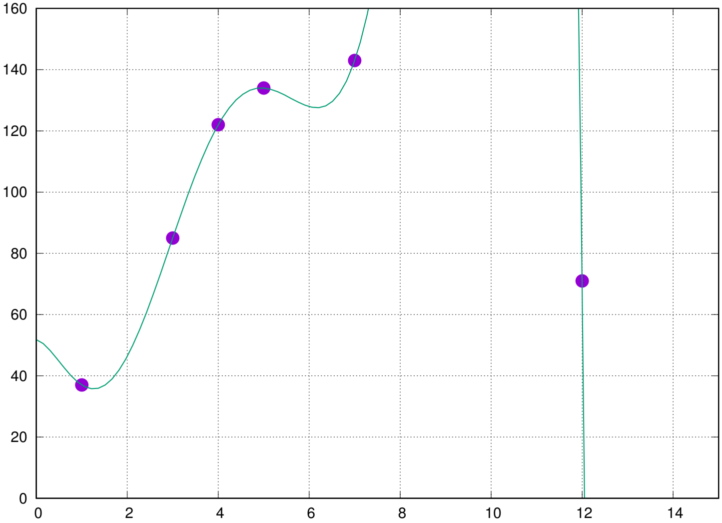
Example



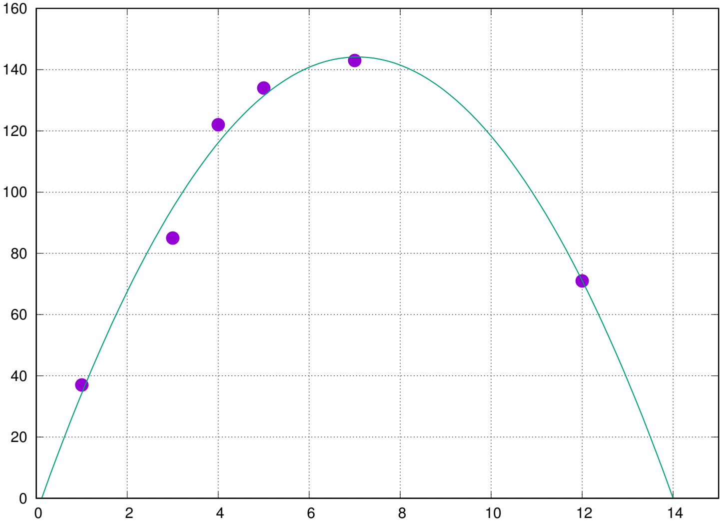
Underfitting example



Overfitting example



Better fit



Capacity

- Ability to fit wide variety of functions
  - Low capacity will struggle to fit the training set
  - High capacity will can overfit by memorizing the training set
- Capacity can be controlled by choosing hypothesis space
  - A polynomial of degree 1 gives linear regression $\hat{y} = b + wx$
  - By adding x^2 term, it can learn quadratic curve $\hat{y} = b + w_1 x + w_2 x^2$
    - Output is still a linear function of parameters
- Capacity of is determined by the choice of model (Representational capacity)
- Finding best function is very difficult optimization problem
  - Learning algorithm does not find the best function but reduces the training error
  - Imperfection in optimization algorithm can further reduce the capacity of model (effective capacity)

Capacity (contd.)

- Occam's razor
  - Among equally well hypotheses, choose the simplest one
- Vapnik-Chervonenski dimension - Capacity for binary classifier
  - Largest possible value of m for which a training set of m different $\boldsymbol{x}$ point that the classifier can label arbitrarily
- Training and test error is bounded from above by a quantity that grows as model capacity grows but shrinks as the number of training example increases
  - Bounds are usually provided for ML algorithm and rarely provided for DL
  - Capacity of deep learning model is difficult as the effective capacity is limited by optimization algorithm
    - Little knowledge on non-convex optimization