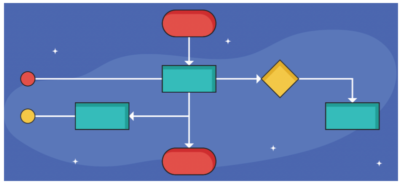
MACHINE LEARNING

COMP8220

02 – Workflow for ML Project



Xuyun Zhang (Department of Computing)

Lecture Outline

MACQUARIE University

- Workflow of Machine Learning Project
  - Data Pre-processing and Feature Engineering
  - Model Training and Evaluation
- Linear Regression Introduction
  - Linear Regression Model
  - Overfitting and Model Selection

Lecture Outline

- Workflow of Machine Learning Project
  - Data Pre-processing and Feature Engineering
  - Model Training and Evaluation

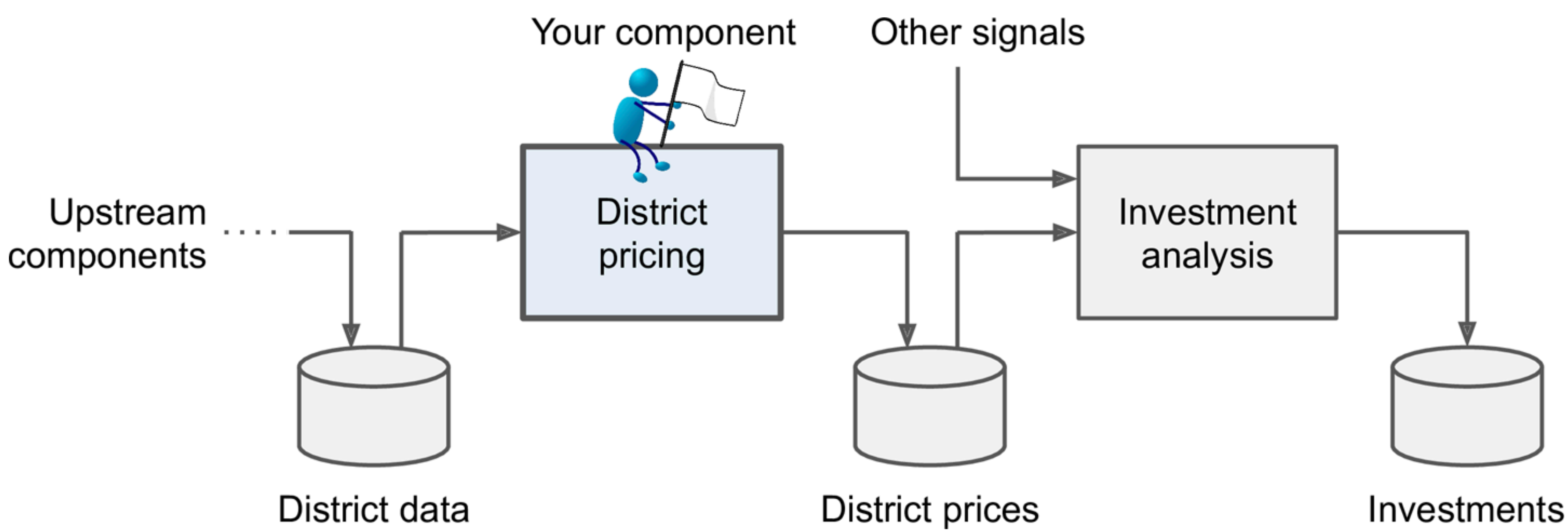
Workflow of ML Project

MACQUARIE University

- Problem framing
- Data collection
- Exploratory analysis and visualization
- Data Pre-processing
- Feature extraction and selection
- ML model training
- Model fine-tuning
- ML model deployment and maintenance

Framing Problem

- What is the business objective?
  - Building a model might not be the end goal
  - E.g., an ML pipeline for real estate investment

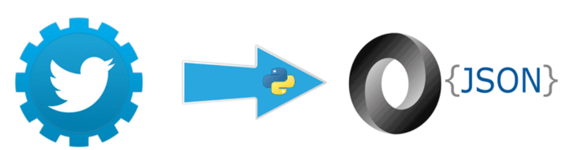


Framing Problem

- Requirement analysis
  - Similar to that in Software Engineering
  - Based on data and business requirement
- Typical examples of analysis questions
  - Is it a supervised or unsupervised learning task?
    - Are data labelled or not?
    - Will domain experts be involved in any stage?
  - Is it a online learning or batch learning?
    - Data streaming? Large datasets?
  - What is the performance measures?
  - Sequential or parallel implementation?
  - What are the interfaces to other components?

Data Collection

- Manually collect datasets (usually small)
- Automated tools to collect datasets
  - E.g., Web crawlers to collect data from web pages
- Public datasets
  - Data repositories, e.g.,
    - UC Irvine Machine Learning Repository
    - Kaggle datasets
    - Amazon's AWS datasets
    - Data Portals (meta portals listing repositories)
  - Web APIs for data queries, e.g.,
    - Twitter data APIs
    - Domain property data APIs



Exploratory Analysis

- Quick glance to get a general understanding
  - Metadata, e.g., data size, data types, etc.
  - Assist in problem framing and modelling
  - Less complex than machine learning models
  - Light-weight experiments (feature combination)
- Visualization
  - Human brains are very good at spotting patterns in pictures
  - Applicable to low (2 or 3) dimensional data
  - Dimension reduction might be necessary for visualization
    - PCA
    - Just randomly select 2 or 3 features for visualization (there are many possible combinations)

Exploratory Analysis

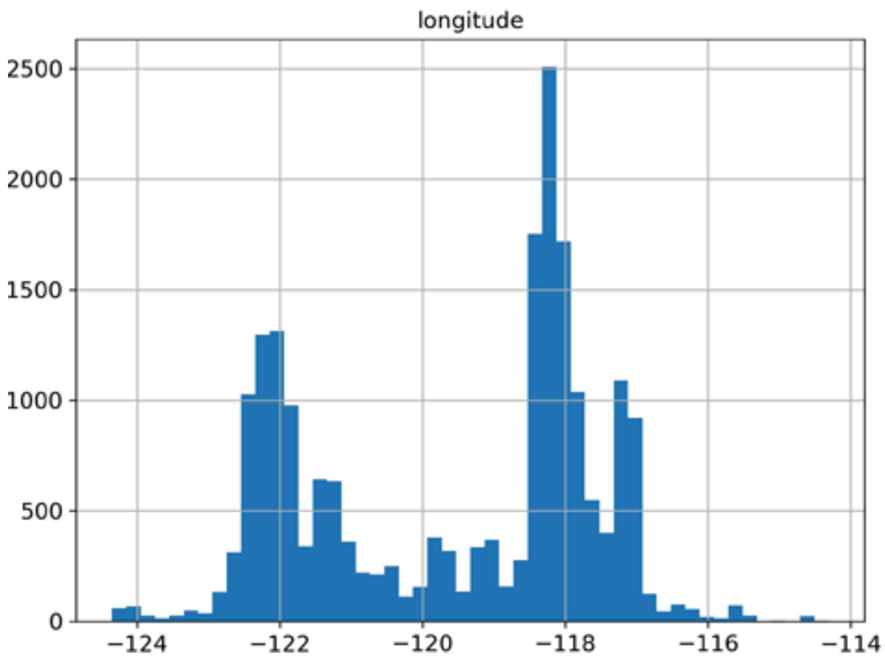
MACQUARIE University

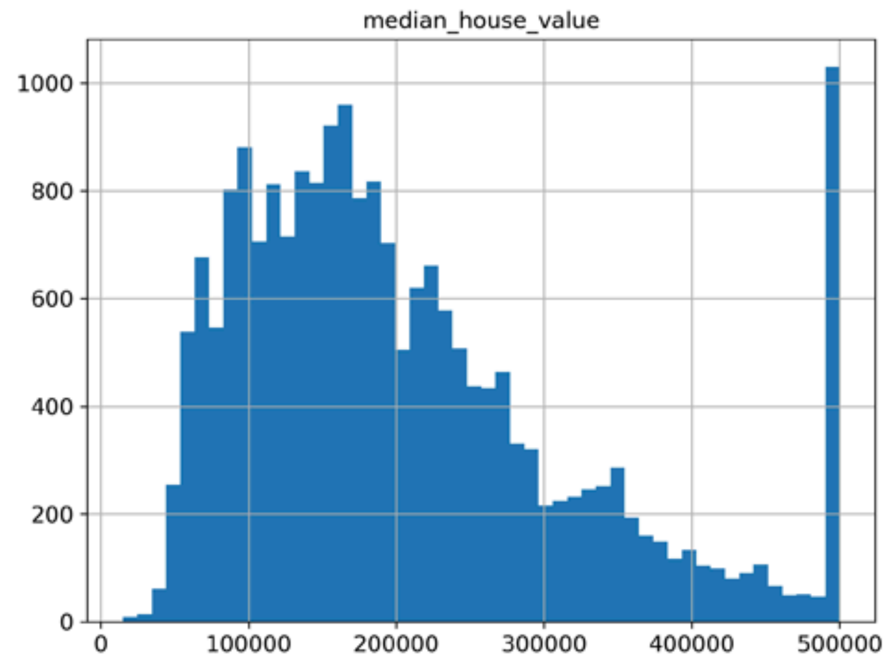
- Typical types of exploratory analysis
  - Statistics
  - Correlations

- Typical statistics
  - Count
  - Mean, median, mode
  - Std
  - Max, min
  - Quantile
  - …

Exploratory Analysis

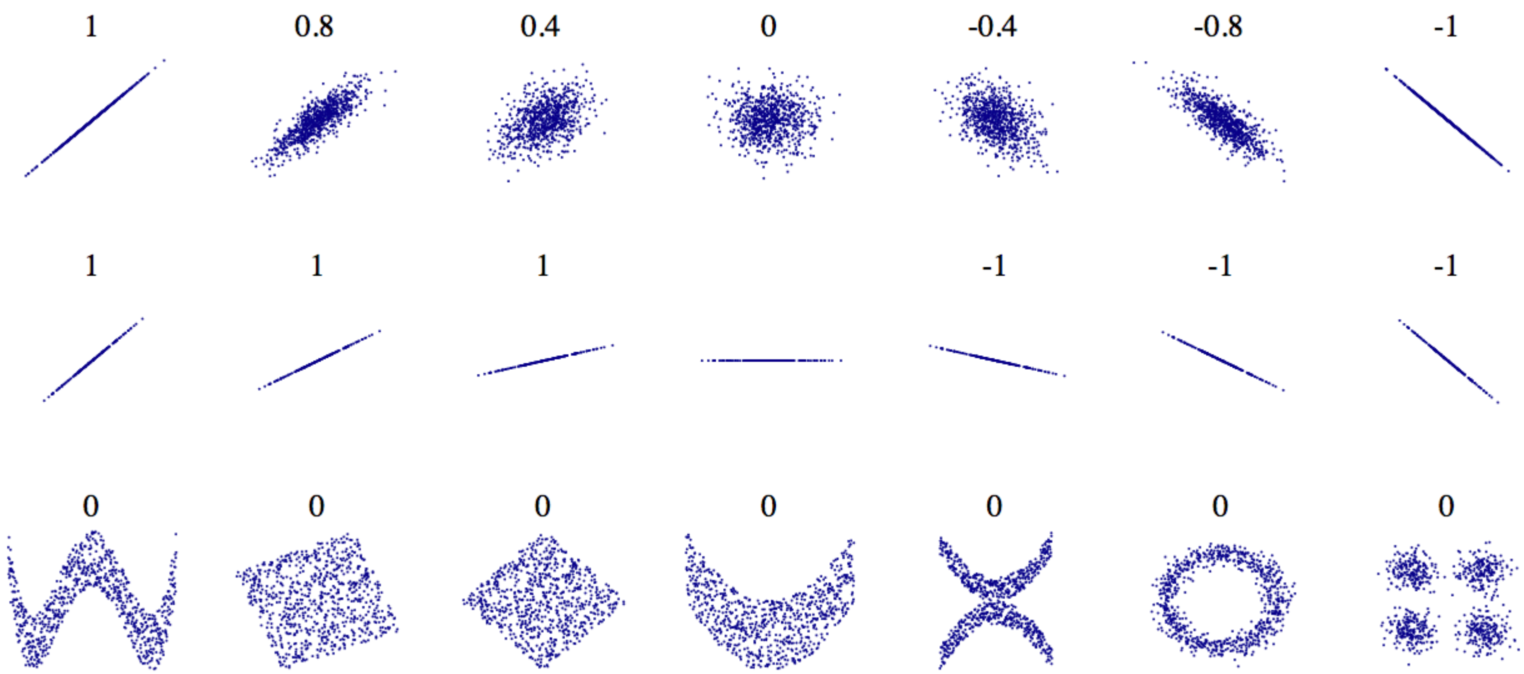
- Histogram
  - To visualize the statistics
  - E.g.,





Exploratory Analysis

- Correlations
  - Mainly refer to linear correlation (simple for exploration)
  - Between a feature and the target, or between two features
- Examples ('1' for strong positive correlation)



Data Pre-processing

- Missing values
  - Blanks, NaNs, or other placeholders
- Standardization
  - Attributes with varying scales
- Normalization
- Imbalanced data
  - Sampling
- Data duplication
- Outliers (noise)

Handling Missing Values

- Discarding the data instances with missing values
  - If the number of missing values is small
- Discarding the attributes with missing values
  - If the number of missing values is big
- Imputation
  - Manually: tedious + infeasible?
  - Automatically
    - A global constant
    - Mean, median, mode, or other statistics
    - Based on advanced techniques such as regression, nearest neighbors, matrix factorization, etc.

Missing Value Imputation

- Examples
  - Raw data: $$X = \begin{bmatrix} -1 & 1 \\ 6 & 2 \\ 3 & 3 \\ 3 & 4 \\ 1 & 4 \\ 9 & ? \end{bmatrix}$$
  - Mean: 2.8 $\left(\frac{1+2+3+4+4}{5} = 2.8\right)$
  - Median: 3 (3 is in the middle)
  - Mode: 4 (4 is the most frequent)
  - Regression: How? (result: 3.38)

Standardization

- Z-score (centered and scaled)

$$x' = \frac{x - \mu}{\sigma}$$

- Fixed range, e.g., $[0, 1]$ or $[-1, 1]$
  - For $[0, 1]$: $x' = \frac{x - x_{min}}{x_{max} - x_{min}}$
  - For $[-1, 1]$: ?
- Non-linear transformation (still order-preserving)
  - E.g., Box-Cox transform
- All transformation should be applied to both training and testing data

Standardization (Cont'd)

- Examples
  - Raw data: $$X = \begin{bmatrix} 1 & -1 & 2 \\ 2 & 0 & 0 \\ 0 & 1 & -1 \end{bmatrix}$$
  - Z-score: $$X' = \begin{bmatrix} 1.0 & -1.22 & 1.33 \\ 1.22 & 0.0 & -0.26 \\ -1.22 & 1.22 & -1.06 \end{bmatrix}$$
  - Fixed range $[0, 1]$: $$X' = \begin{bmatrix} 0.5 & 0.0 & 1.0 \\ 1.0 & 0.5 & 0.33 \\ 0.0 & 1.0 & 0.0 \end{bmatrix}$$

Normalization

- Scaling individual data instances to have unit norm
  - Metrics: L_1 norm: Manhattan; L_2 norm: Euclidean
  - i.e., the "length" of a feature vector is 1
  - The angle information is still preserved
- Examples
  - Raw data : $\mathrm{X} = \begin{bmatrix} 1 & -1 & 2 \\ 2 & 0 & 0 \\ 0 & 1 & -1 \end{bmatrix}$
  - L_2 normalized: $\mathrm{X}' = \begin{bmatrix} 0.40 & -0.40 & 0.81 \\ 1.00 & 0.0 & 0.0 \\ 0.0 & 0.70 & -0.70 \end{bmatrix}$
  - E.g., first row: $\sqrt{0.40^2 + (-0.40)^2 + 0.81^2} \approx 1.0$

Handling Imbalanced Data

- Sampling: obtaining data instances to represent the whole original data set
- Undersampling vs oversampling
  - The sample size is less than the original size
  - The sample size is greater than the original size
- With vs without replacement
  - Whether an instance is removed from the data set or not
- Simple random sampling (SRS) vs other sampling
  - SRS: each instance has an equal probability of being selected
  - Others: the probability might not be equal, e.g., stratified sampling (grouping data first; then sampling from groups)

Other Pre-processing

- Data duplication
  - Depending on specific ML models, some are sensitive

- Outlier (noise)
  - Caused by data entry problem, faulty instruments, …
  - The consequence also depends on specific ML models
  - E.g., ordinary linear regression is sensitive while tree models are often robust
  - Outlier detection itself can be the main ML task, particularly in the cyber security domain

Feature Engineering

- Feature generation
  - Manual feature engineering by domain experts
    - Domain experts can identify more informative features
  - Feature interaction and polynomials
- Feature selection
  - Some features might be redundant or irrelevant
  - Select a subset of (informative) features
- Feature extraction
  - Transform to another feature space (lower-dimensional)
- Deep learning
  - Automatic feature representation and extraction

Feature Interaction & Polynomial

- To enrich feature representation
  - Particularly for linear models (to capture non-linearity)
  - E.g., we used higher orders of polynomial terms in the linear regression running example (still remember?)
- Feature interaction example

$$\langle X_1, X_2 \rangle \rightarrow \langle 1, X_1, X_2, X_1^2, X_1X_2, X_2^2 \rangle$$

Index	X1	X2
0	0	1
1	2	3

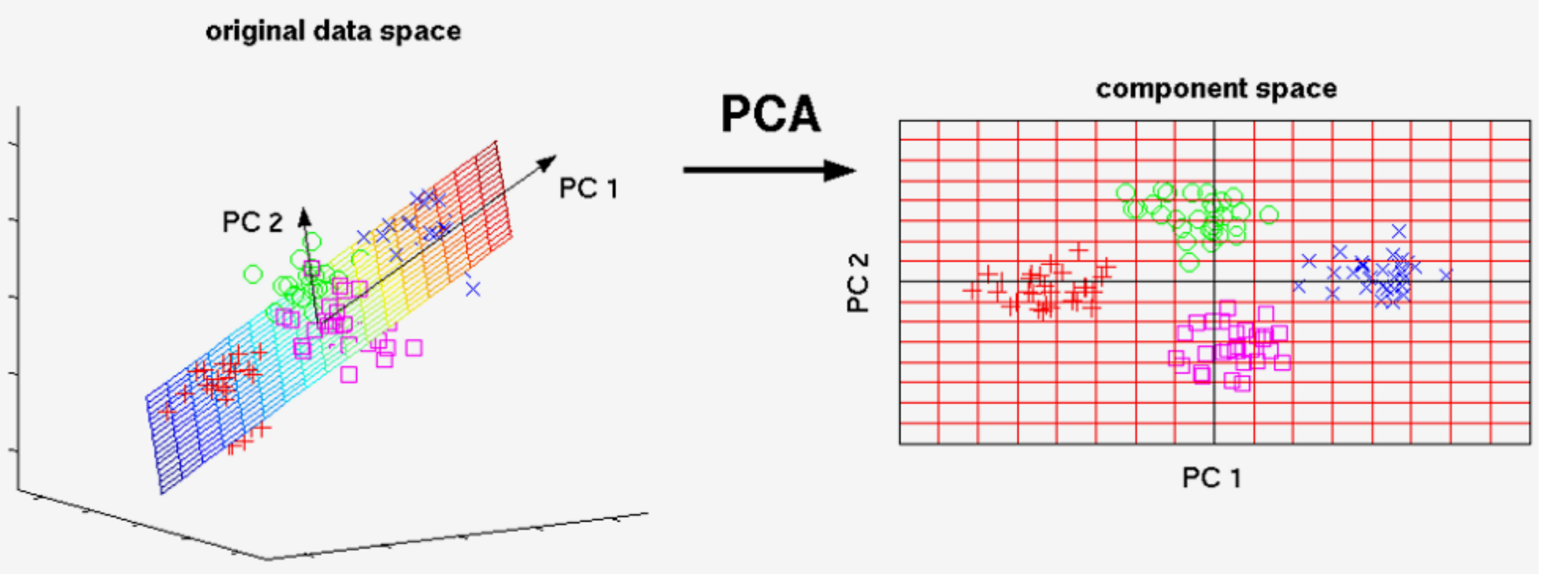
Index	1	X1	X2	X1X1	X1X2	X2X2
0	1	0	1	0	0	1
1	1	2	3	4	6	9

Feature Selection

- Filter approaches: univariate statistics
  - Test if an individual feature has statistically significant relationship with the target
  - E.g., analysis of variance (ANOVA), correlation coefficient
- Wrapper approaches: feature subset selection
  - Brute-force: try all possible subsets (NOT scalable)
  - Recursive feature elimination
    - Build a model; discard least important feature; repeat
- Embedded approaches: model-based selection
  - Use ML models to judge feature importance
  - E.g., decision tree models, Lasso regression coefficients

Feature Extraction

- Transformative
  - Linear or non-linear projection to another space
  - Dimensionality can usually be reduced
- Commonly-used techniques
  - E.g., Principal Component Analysis (PCA)



Deep Feature Representation

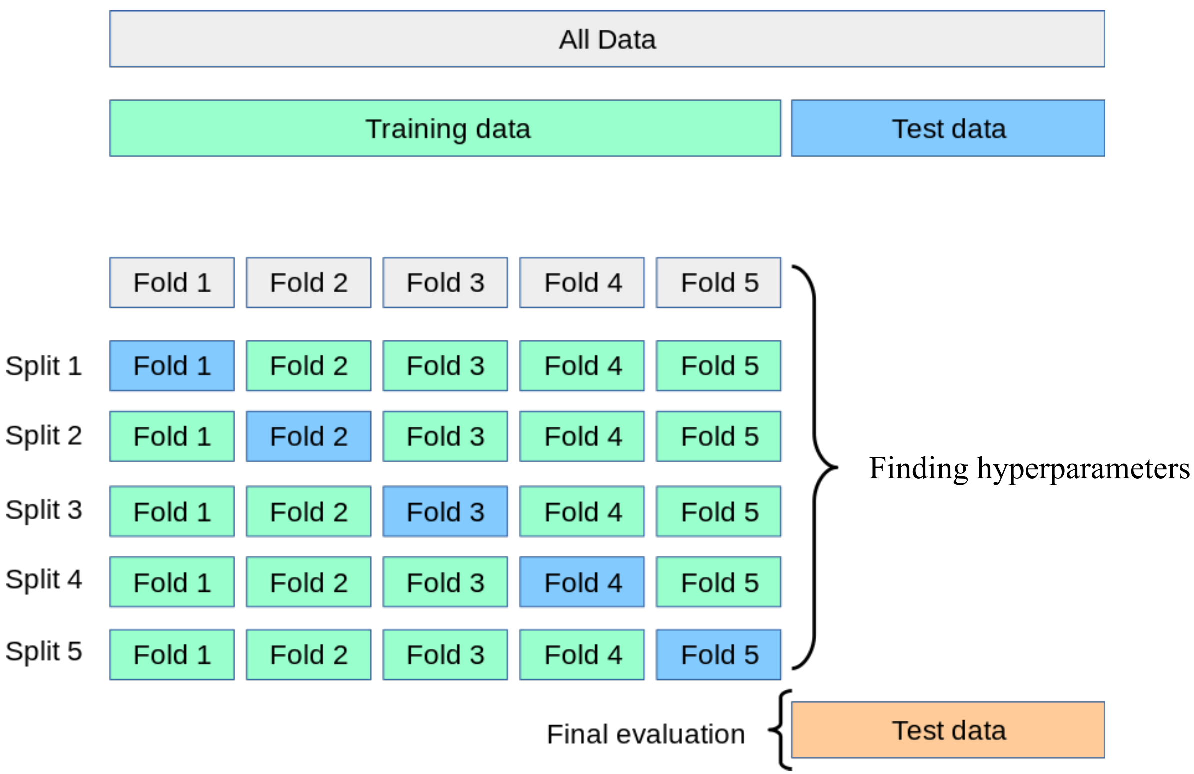
- Deep learning can automatically perform feature selection and extraction
  - Interpretability might be lost, e.g., feature importance
- Typical examples
  - Convolutional Neural Network (CNN)
    - Convolution can capture/activate features in images
    - For images/videos
  - Recurrent Neural Network (RNN)
    - Capture temporal/sequential information
  - Long Short-Term Memory networks (LSTM)
    - A special kind of RNN
    - For sequences, audio, time series, and text

Model Training

- Determine a machine learning model
  - Supervised vs unsupervised
  - Categorical vs numerical
  - Batch vs online
  - …
- Split data into training and testing data
  - Training data for building the model
  - Testing data for validating the learning model
  - Randomly partition the data
  - Can be 80% vs 20%, or 75% vs 25%
  - K-folder cross validation

Cross Validation

- k-fold cross validation
  - Usually, k is 5 or 10
  - Extreme case $k = n$, leave one out cross validation



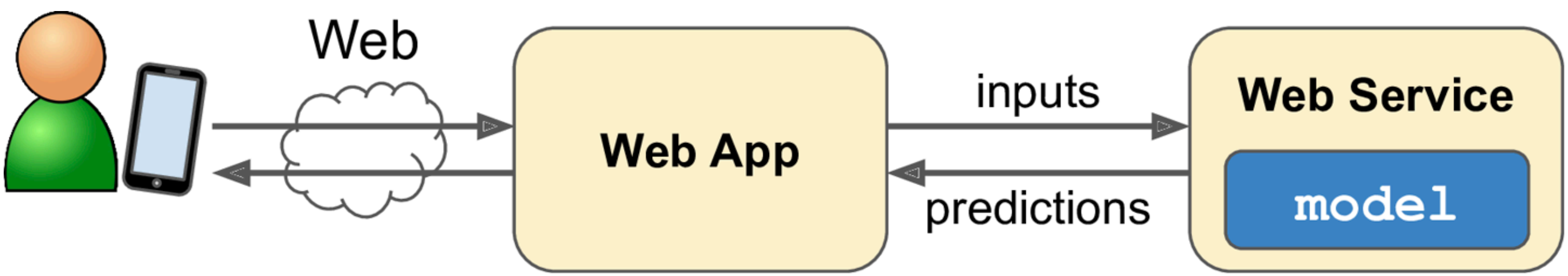
Model Fine-tuning

- Model tuning is a non-trivial job!
  - Model selection
  - Confusing terms
    - *Model parameters*
    - *Model hyperparameters*
    - *Learning algorithm parameters*
  - Tuning is performed w.r.t. model hyperparameters
- Typical tuning methods
  - Manual tuning based on human experts
  - Automated grid search for optimal model hyperparameters
  - Automated random search
  - Often work together with cross validation

Model Deployment

MACQUARIE University

- Deploy the trained model in a production environment
  - Need to save the trained model
    - E.g., using 'joblib' in Scikit-learn library
  - Integrated with other parts of a system
    - E.g., wrap the model within a dedicated web service that your web application can query through a REST API



- Other example is Google Cloud AI Platform

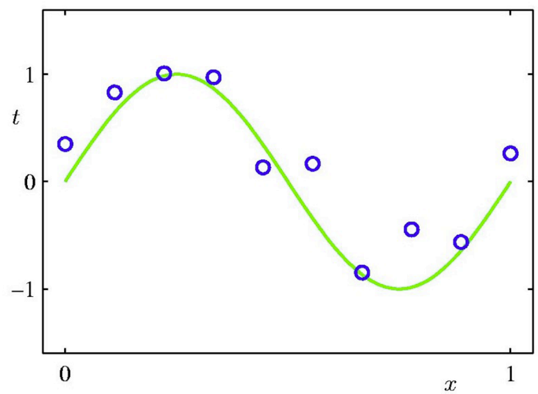
Lecture Outline

- Workflow of Machine Learning Project
  - Data Pre-processing and Feature Engineering
  - Model Training and Evaluation

- Linear Regression Introduction
  - Linear Regression Model
  - Overfitting and Model Selection

Regression

- Suppose we have 10 observations
  - $\mathcal{D} = \{\langle x_1, t_1\rangle, \cdots, \langle x_{10}, t_{10}\rangle\}$
  - Ground truth: $t = \sin(2\pi x)$
  - Noise added to t_i



- **Regression** task: estimate $t = y(x)$ from D

Linear Models

- A function is linear if

$$f(\alpha x_1 + \beta x_2) = \alpha f(x_1) + \beta f(x_2)$$

  - α, β are scalars
  - Additivity and homogeneity

- Linear regression model

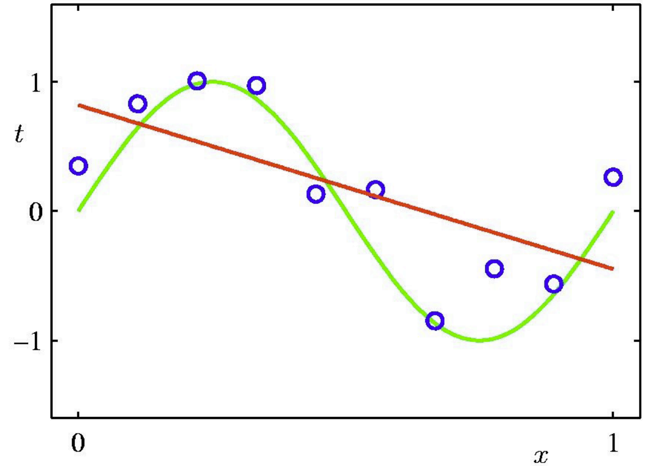
$$y(\boldsymbol{x}, \boldsymbol{\omega}) = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \cdots + \omega_D x_D$$

  - Note: D is the # of features, not the # of instances
  - The linearity of $y(\cdot, \omega)$ is with respect to ω
  - Note: we are learning ω

The Simplest From

- Dimensionality $M = 1$

$$y(x, \boldsymbol{\omega}) = \omega_0 + \omega_1 x$$



  - Can only model a "line"

- How to handle non-linearity?

Feature Extraction

- Transform x

$$\boldsymbol{x} \Rightarrow \boldsymbol{\Phi}(\boldsymbol{x}) = (\phi_0(\boldsymbol{x}), \phi_1(\boldsymbol{x}), \phi_2(\boldsymbol{x}), \cdots, \phi_M(\boldsymbol{x}))^T$$

  - $\phi_j(x)$, $j \geq 0$ form a family of basic functions
  - Transformation/feature function is often non-linear

- Linear basis function models

$$y(\boldsymbol{x}, \boldsymbol{\omega}) = \sum_{j=0}^{M} \omega_j \phi_j(\boldsymbol{x}) = \boldsymbol{\omega}^T \boldsymbol{\Phi}(\boldsymbol{x})$$

  - $\boldsymbol{\omega} = (\omega_0, \cdots, \omega_M)^T$, $\boldsymbol{\Phi} = (\phi_0, \cdots, \phi_M)^T$

Special Cases

- For $0 < j \leq D$, $\phi_j(x) = x_j$; $\phi_0(x) = 1$

$$y(\mathbf{x}, \boldsymbol{\omega}) = \sum_{j=0}^{D} \omega_j x_j$$

  - The original form

- $\phi_j(x) = x^j$, $0 < j \leq M$

$$y(x, \boldsymbol{\omega}) = \sum_{j=0}^{M} \omega_j x^j$$

  - Power series expansions

Loss Function for Regression

- How to learn/choose the parameter vector ω?
  - First, define a loss function $L(y(x), t)$
  - Then, find out $y(x)$ to minimize the loss (expectation)

$$y^*(\boldsymbol{x}) = \arg\min_{y(\boldsymbol{x})} \mathbb{E}[L] = \int\int L(y(\boldsymbol{x}), t)p(\boldsymbol{x}, t)d\boldsymbol{x}dt$$

  - E.g., general squared loss function:

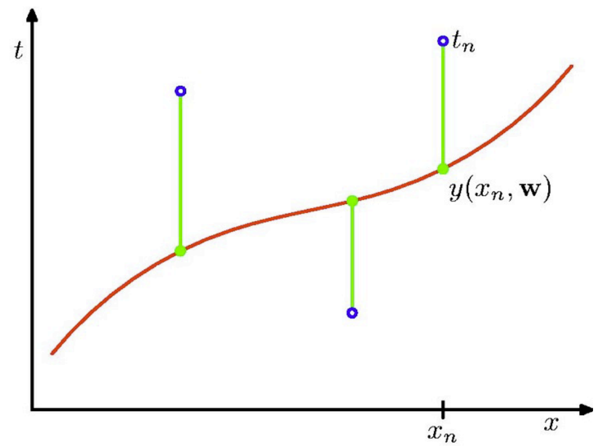
$$L(y(\boldsymbol{x}), t) = \frac{1}{2}||y(\boldsymbol{x}) - t||^2$$

  - In practice, we often use the loss function defined on training data

Least Squares

- Sum of squared error:

$$E(\omega) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \omega) - t_n\}^2$$



- Then, the specific learning problem

$$\omega^* = \arg\min_{\omega} E(\omega)$$

Optimization Problem

- Learning problem → optimization problem

$$\omega^* = \arg\min_{\omega} E(\omega),$$

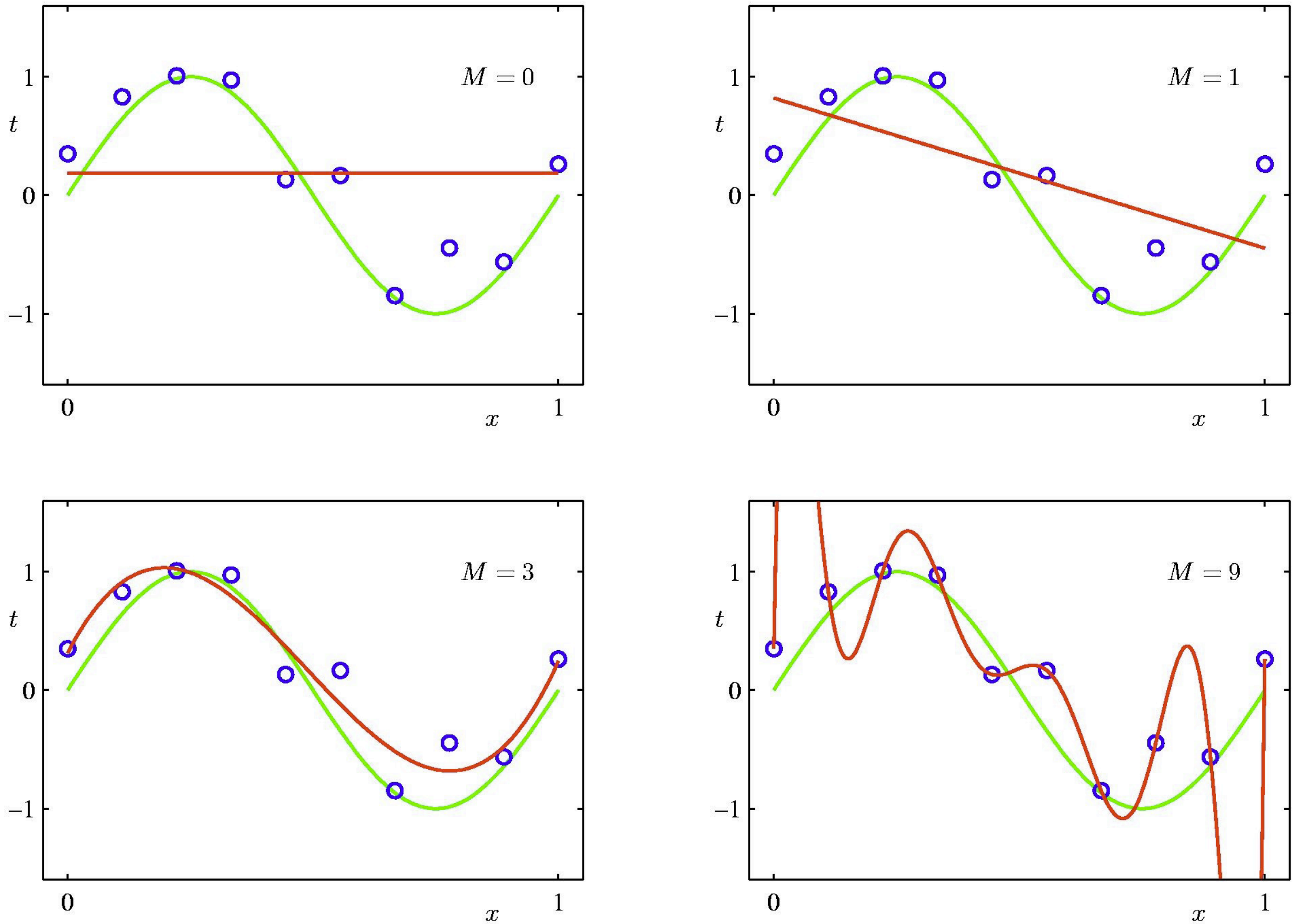
$$E(\omega) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \omega) - t_n\}^2$$

- How to solve? $$\frac{\partial E(\omega)}{\partial \omega} = 0$$
  - Close-form solution can be obtained
  - For the basis function model

$$\omega_{ML} = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\boldsymbol{t}$$

Different Degrees of Polynomials

MACQUARIE University



Observations

- $M = 0$ or $M = 1$
  - Only passing a couple of points
  - Models are too simple
- $M = 9$
  - Passing every points but oscillating wildly
  - **Overfitting** problem: Lack of generalisation capability
- $M = 3$
  - Much better solution
  - Good trade-o has been made

Training/Test Error

- Training/empirical error of the trained model $\hat{f}$ on a training dataset with size N, e.g.,

$$E_{emp}(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}(x_i))$$

  - Learning process usually tries to minimize this error

- Test error on a test data set with size N', e.g.,

$$E_{test}(\hat{f}) = \frac{1}{N'}\sum_{i=1}^{N'} L(y_i, \hat{f}(x_i))$$

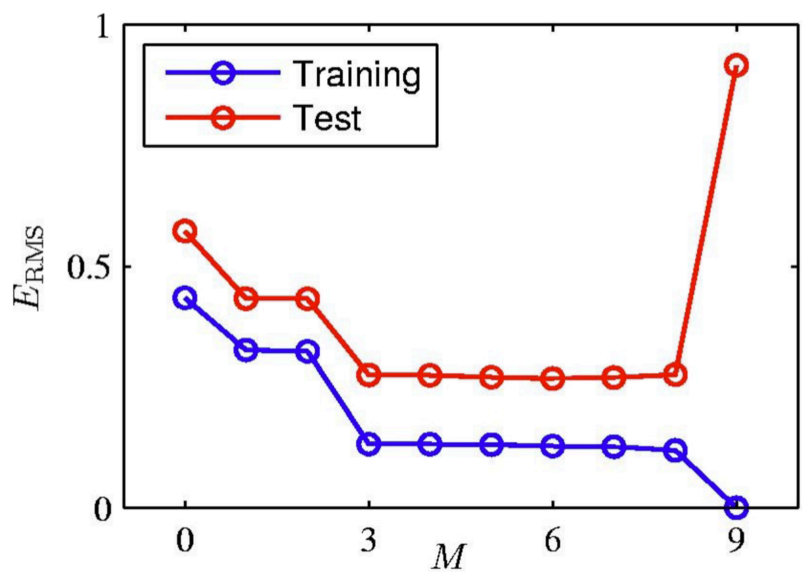
  - Test error indicates the generalisation capability of a learned model

Overfitting Issue

MACQUARIE University

- Test errors with respect to M
  - Test data containing 100 points
  - Root-Mean-Square (RMS) error

$$E_{RMS} = \sqrt{2E(\omega^*)/N}$$



Model Selection

- Observations on M
  - Different models generated by varying M
  - M controls the learning capability of the model
  - Unlike model parameter ω, M can NOT be learned from data (**hyperparameter**)
- Model selection
  - Models generated by different learning algorithms
  - Models generated by different algorithm parameter (model hyperparameter) configurations
- Model selection is a non-trial job!

Regularization

- Another way to the overfitting issue
  - Penalizing large coefficients in error function
  - E.g., quadratic regularizer

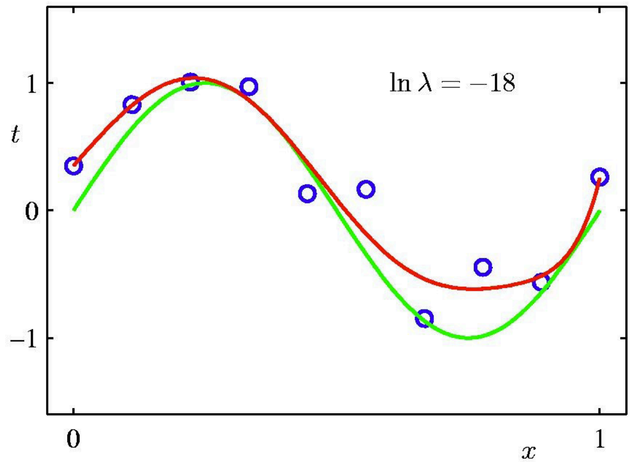
$$\tilde{E}(\boldsymbol{\omega}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \boldsymbol{\omega}) - t_n\}^2 + \frac{\lambda}{2}||\boldsymbol{\omega}||^2,$$

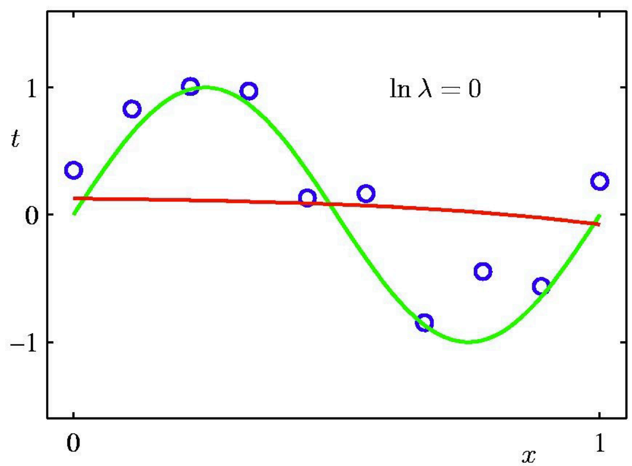
$$||\boldsymbol{\omega}||^2 = \boldsymbol{\omega}^T\boldsymbol{\omega} = \omega_0^2 + \omega_1^2 + \cdots + \omega_M^2$$

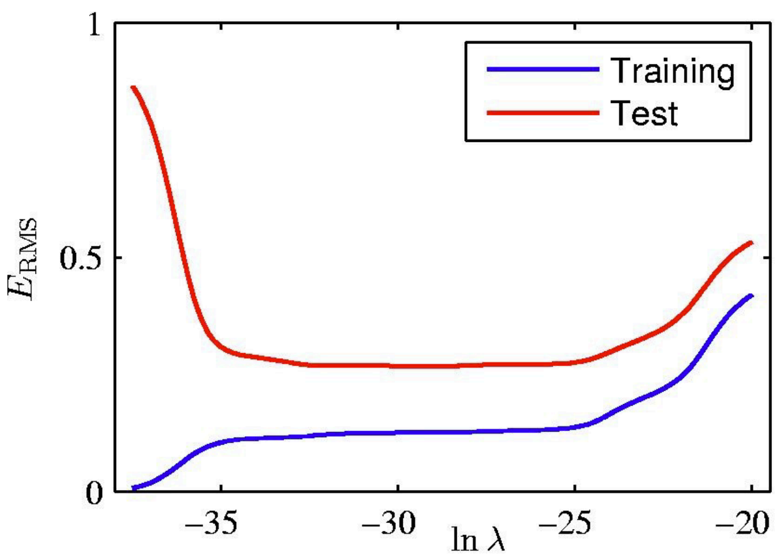
  - λ governs the trade-offs between the two terms
  - A.k.a. ridge regression
  - Close-form solution can be obtained
  - Lasso regression: the regularizer is $||\omega||_{L_1}$

Effects of λ

❖ Performance of different λ when $M = 9$







Scikit-learn (sklearn)

MACQUARIE University

- Data preparation
- Training/test data split
  - x_train, y_train, x_test, y_test
- Train the model
  - Build a model: model = xxxModel.xxxModel()
  - Train a model: model.fit(x_train, y_train)
  - Validate: cross_validate(xxxmodel, x_train, y_train, k, scoring)
- Test the learning model
  - Test the model: y_pred = xxxModel.predict(x_test)
  - Can use y_pred and y_test to calculate the error, e.g., mean_squared_error(y_test, y_pred)