https://www.sli.do/
#073374

Classical Machine Learning: Classification and Regression (V)

Learning Objectives

- Learn how to fine tune your machine learning model.
- Learn a few regression models.
- Learn the kernel method and regularization techniques.

Algorithm Tuning: AdaBoost, Gradient Boost

C-S David Chen, Department of Civil Engineering, National Taiwan University

Algorithm Tuning

- Algorithm tuning mainly involves tuning hyperparameters.
- Hyperparameters are parameters that are not directly learnt within estimators (algorithms).
- In scikit-learn they are passed as arguments to the constructor of the estimator classes. Typical examples include C, kernel and gamma for Support Vector Classifier, alpha for Lasso, etc.
- Two generic approaches to parameter search are provided in scikit-learn: for given values, **GridSearchCV** exhaustively considers all parameter combinations, while **RandomizedSearchCV** can sample a given number of candidates from a parameter space with a specified distribution.

sklearn.ensemble.AdaBoostClassifier

class `sklearn.ensemble.` **`AdaBoostClassifier`**(*base_estimator=None*, ***, *n_estimators=50*, *learning_rate=1.0*, *algorithm='SAMME.R'*, *random_state=None*) [source]

An AdaBoost classifier.

An AdaBoost [1] classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

sklearn.ensemble.GradientBoostingClassifier

class `sklearn.ensemble.` **`GradientBoostingClassifier`**(***, *loss='deviance'*, *learning_rate=0.1*, *n_estimators=100*, *subsample=1.0*, *criterion='friedman_mse'*, *min_samples_split=2*, *min_samples_leaf=1*, *min_weight_fraction_leaf=0.0*, *max_depth=3*, *min_impurity_decrease=0.0*, *min_impurity_split=None*, *init=None*, *random_state=None*, *max_features=None*, *verbose=0*, *max_leaf_nodes=None*, *warm_start=False*, *validation_fraction=0.1*, *n_iter_no_change=None*, *tol=0.0001*, *ccp_alpha=0.0*) [source]

Gradient Boosting for classification.

GB builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage `n_classes_` regression trees are fit on the negative gradient of the binomial or multinomial deviance loss function. Binary classification is a special case where only a single regression tree is induced.

Algo_Tuning.ipynb

Regression and Regression Algorithm Walkthrough

C-S David Chen, Department of Civil Engineering, National Taiwan University

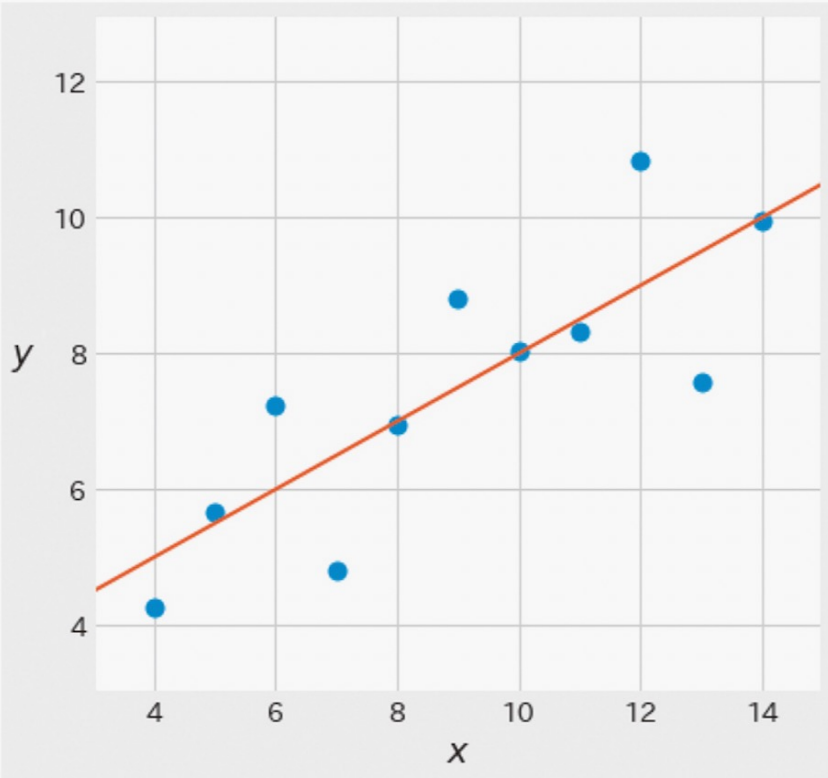
Supervised Learning: Regression

- Predict a value of a given continuous valued variable based on the values of other variables, assuming a linear or nonlinear model of dependency.
- Extensively studied in statistics, neural network fields.
- Examples:
  - Predicting sales amounts of new product based on advertising expenditure.
  - Predicting wind velocities as a function of temperature, humidity, air pressure, etc.
  - Time series prediction of stock market indices.

Simple Linear Regression

$$y \approx w_0 + w_1 x$$

- Predicting a quantitative response y on the basis of a single feature x (feature dimension = 1).

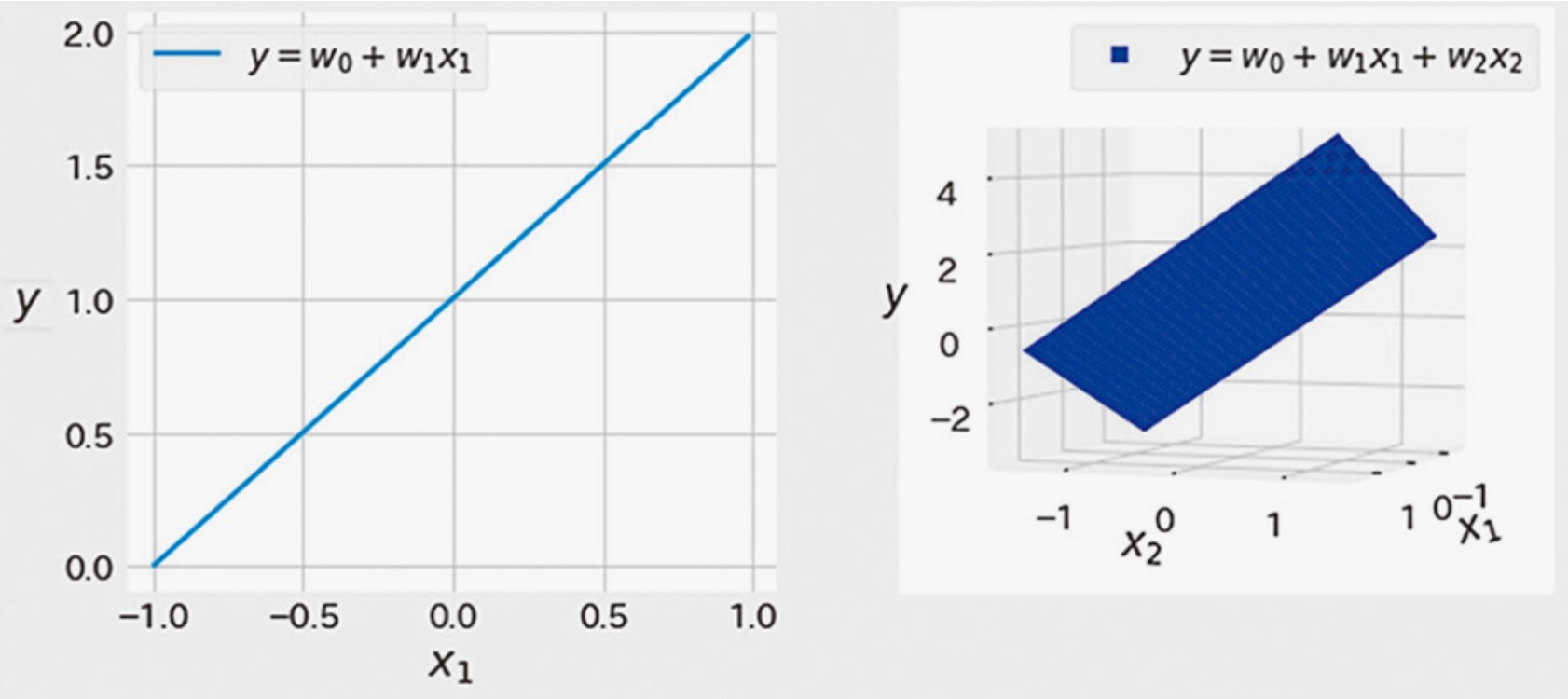


i	x	y
0	10.0	8.04
1	8.0	6.95
2	13.0	7.58
3	9.0	8.81
4	11.0	8.33
5	14.0	9.96
6	6.0	7.24
7	4.0	4.26
8	12.0	10.84
9	7.0	4.82
10	5.0	5.68

Multiple Linear Regression

$$y \approx w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_d x_d$$

- Predicting a quantitative response y on the basis of d distinct features.



Regression: Performance Metrics

- The square of sum errors (SSE) $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
- The mean square of sum errors (MSE) $MSE = 1/n \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
- R^2, Coefficient of Determinant $R^2 = 1 - \frac{SSE}{TSS}$

$TSS = \sum_{i=1}^{n}(y_i - \bar{y}_i)^2$ measures **the total variance** in the response y, and can be thought of as the amount of variability inherent in the response before the regression is performed.

Fun Time: which R^2 gives us a better regression:
(1) 1.0 (2) 0.5 (3) 0.0

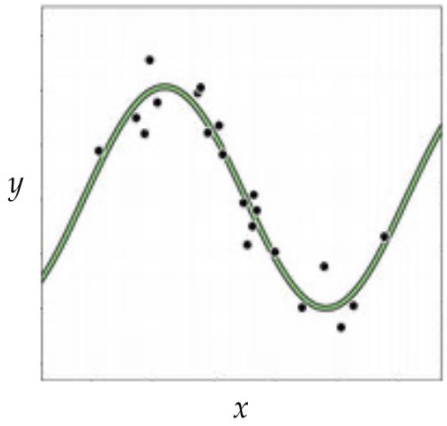
Linear_Regression.ipynb

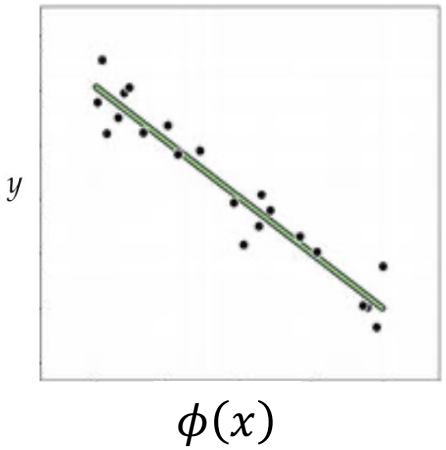
Nonlinear Regression: Feature Transformation, Basis Functions, Feature Exploration and Kernel Method

C-S David Chen, Department of Civil Engineering, National Taiwan University

Nonlinear Regression: Feature Transformation and Basis Function

- single feature x (feature dimension = 1).





$$y \approx w_0 + w_1\phi(x)$$

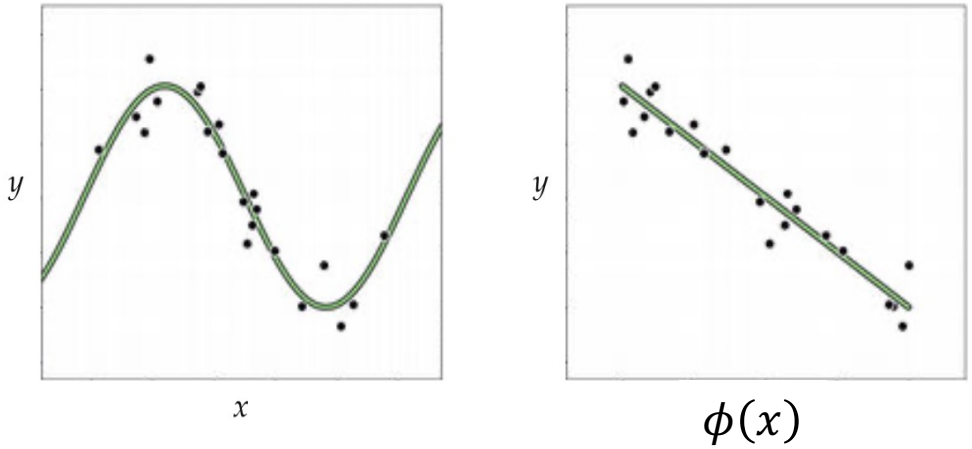
$$\phi(x) = \sin(\alpha_0 + \alpha_1 x)$$

- **Important observation**: $y \approx w_0 + w_1\phi(x)$ remains linear with respect to w_1
- $\phi(x)$ is called **feature transformation**
- To apply feature transformation systematically, we often use some **basis functions** for $\phi(x)$.

See Feature_Transformation.pdf

Nonlinear Regression: Single vs. Multiple Features

- single feature x (feature dimension = 1).



$$y \approx w_0 + w_1\phi(x)$$

$$\phi(x) = \sin(\alpha_0 + \alpha_1 x)$$

- multiple features

$$y \approx \overline{w}_0 + \overline{w}_1\phi(x_1) + \overline{w}_2\phi(x_2) \qquad \phi(x) = (\alpha_0 + \alpha_1 x + \alpha_0 x^2)$$

$$y \approx w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_1 x_2 + w_5 x_2^2$$

we are now dealing with five-dimensional instead of two-dimensional feature space.

Nonlinear Regression: Feature Explosion

Fun Time: let the input dimension d is of moderate size, e.g., $d = 100$, then the associated polynomial degree $n = 5$, what would be the resulting features? (1) ~ 10^3 (2) ~ 10^5 (3) ~ 10^7 (4) ~ 10^9

Nonlinear Regression: Feature Explosion

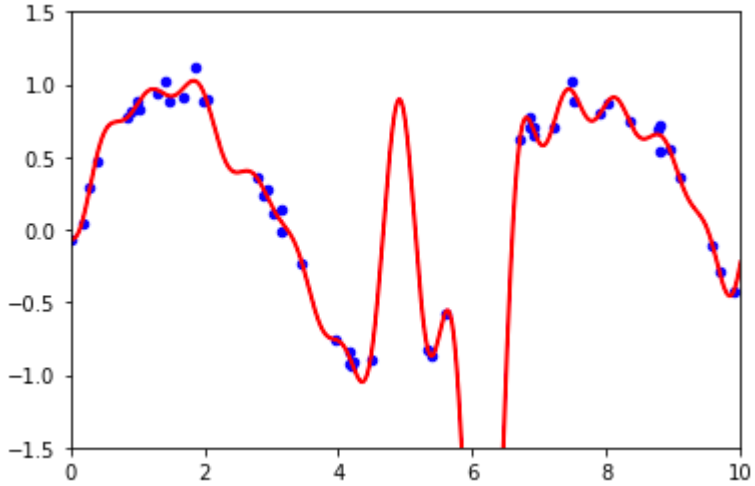
- **The exponential growth challenge of the feature dimension after feature transformation $\mathbf{x} \rightarrow \phi(\mathbf{x})$ can be cleverly solved by the kernel-based method.**
- **A kernel is just a form of generalized dot product $\phi(\mathbf{x}) \cdot \phi(\mathbf{z})$**
- **The key insight in kernel-based method is that you can rewrite many linear models in a way that doesn't require you to ever explicitly compute $\phi(\mathbf{x})$**
- **The topic is often called kernel tricks, a very important topic in ML but beyond the scope of this course.** (refs: Kernel_Trick01.pdf, Kernel_Trick02.pdf)

Nonlinear Regression: Overfit and Regularization

C-S David Chen, Department of Civil Engineering, National Taiwan University

Regularization

- High-order nonlinear regression can easily lead to **overfit**.



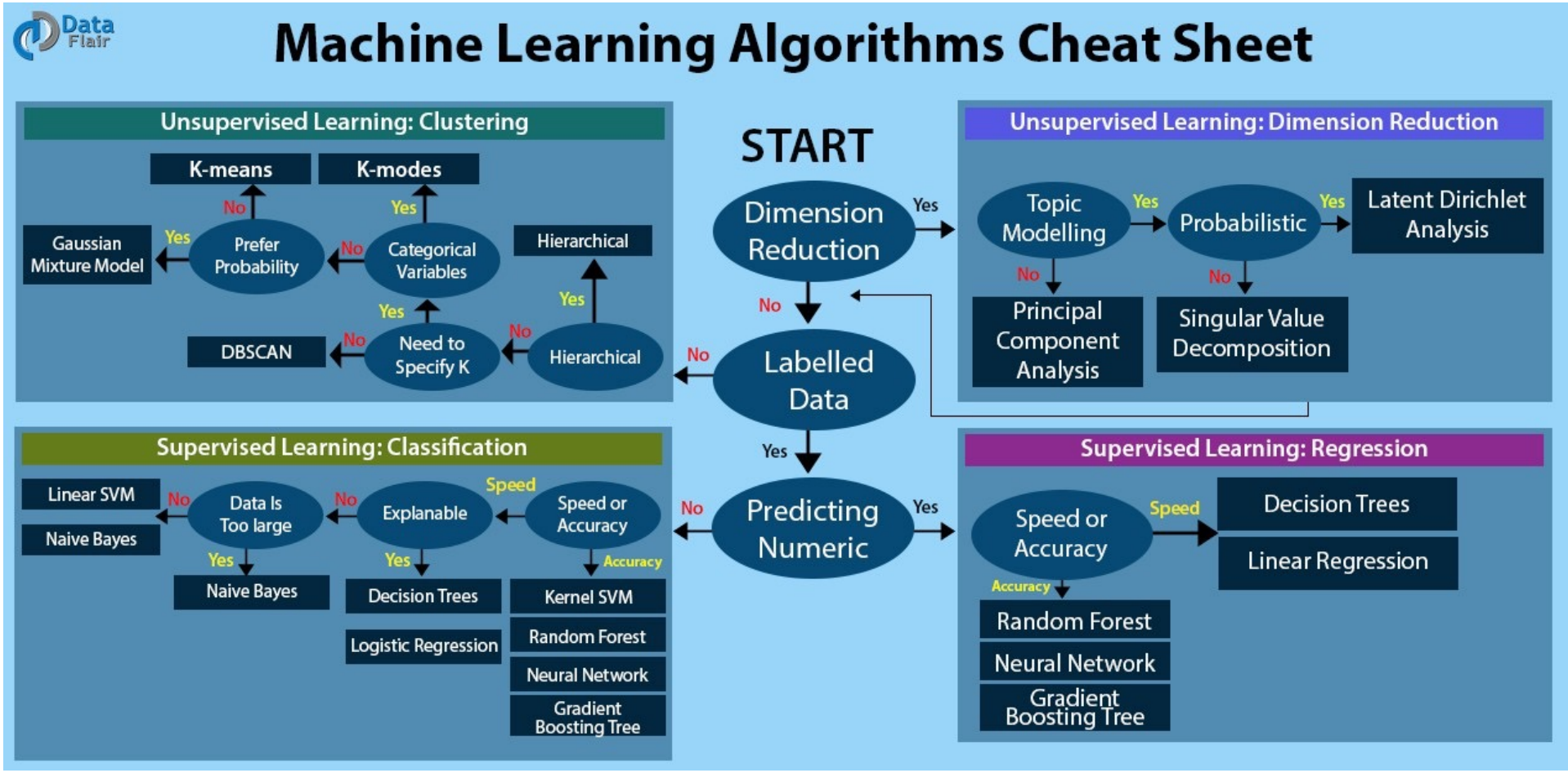
- **Regularization** can help!
  - Ridge (L2) regularization
  - Lasso (L1) regularization

Regularization.pdf

Summary: Regression

- Regression is used to predict a continuous value based some given features.
- Feature transformation $x \rightarrow \phi(x)$ allows us to extend linear models to nonlinear models.
- The exponential growth challenge of the feature dimension after feature transformation $x \rightarrow \phi(x)$ can be cleverly solved by the **kernel**-based method.
- The kernel method is also used in SVM.
- Nonlinear regression tends to overfit and ridge (L2) and lasso (L1) regularization techniques are often used to reduce overfit.
- Many classification methods such as SVM, decision tree, ensemble trees can also handle regression problems.

Cheat Sheet (備忘表)



https://medium.com/dataflair/beat-the-heat-with-machine-learning-cheat-sheet-365c25bd1c3