Machine Learning

Non-parametric Regression: k-NN Method and its variants. Bias-Variance trade-off for k-NN Regression. Mean (Absolute) Test Error.

Aleksandr Petiushko

ML Research

SOFIA UNIVERSITY

Content

Content

Content

1. Non-parametric Regression
2. k-NN Regression: Mean (Absolute) Test Error
3. Bias-Variance trade-off for k-NN Regression

Non-parametric Regression

- The main disadvantage of parametric models is that it is necessary to have a parametric model to describe the dependency

Non-parametric Regression

- The main disadvantage of parametric models is that it is necessary to have a parametric model to describe the dependency
- If it is impossible to select an adequate model, it makes sense to use non-parametric regression methods

Non-parametric Regression

- The main disadvantage of parametric models is that it is necessary to have a parametric model to describe the dependency
- If it is impossible to select an adequate model, it makes sense to use non-parametric regression methods

Assumption

Close objects correspond to similar answers

Non-parametric Regression

The simplest model

We approximate the desired dependence by a constant in some neighborhood

[1] https://en.wikipedia.org/wiki/Kernel_regression

Non-parametric Regression

The simplest model

We approximate the desired dependence by a constant in some neighborhood

Nadaraya-Watson kernel regression[1]

If there are several objects from the training sample in the vicinity of the point, then it is reasonable to use the weighted average as a prediction of the algorithm

$$a(x) = \frac{\sum\limits_i y_i \omega_i(x)}{\sum\limits_i \omega_i(x)},$$

where $\omega_i(x) = K_h(x, x_i)$, a function K_h is called a **kernel** with smoothing window width h.

[1] `https://en.wikipedia.org/wiki/Kernel_regression`

k-NN Regression: simplest prediction method

The simplest model: let us use $\omega_i(x) = \frac{1}{k}$ for the k-NN method.

k-NN Regression: simplest prediction method

The simplest model: let us use $\omega_i(x) = \frac{1}{k}$ for the k-NN method.

k-NN Regression prediction

Let us have for every x_0 k nearest neighbors $(x_1, \ldots, x_k)$ with the ground truth labels $(y_1, \ldots, y_k)$. Then the Nadaraya-Watson kernel regression formula will transform into the following:

$$a(x_0) = \frac{1}{k} \sum_{i=1}^{k} y_i$$

k-NN Regression: simplest prediction method

The simplest model: let us use $\omega_i(x) = \frac{1}{k}$ for the k-NN method.

k-NN Regression prediction

Let us have for every x_0 k nearest neighbors $(x_1, \ldots, x_k)$ with the ground truth labels $(y_1, \ldots, y_k)$. Then the Nadaraya-Watson kernel regression formula will transform into the following:

$$a(x_0) = \frac{1}{k} \sum_{i=1}^{k} y_i$$

Note: It means we are just averaging the labels of k nearest neighbors.

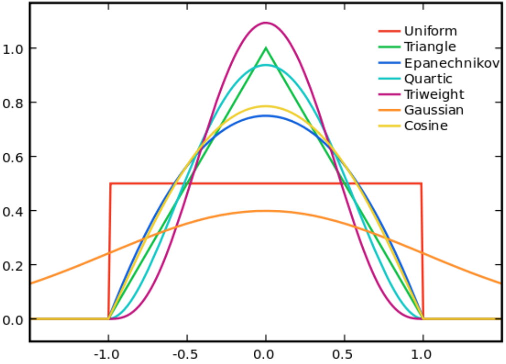
Examples of more complicated Kernels

- $K_h(x, x_i) = K(\frac{||x - x_i||}{h})$

[2] `https://en.wikipedia.org/wiki/Kernel_(statistics)`

Examples of more complicated Kernels

- $K_h(x, x_i) = K(\frac{||x - x_i||}{h})$
- Typical Examples: [2]



[2] `https://en.wikipedia.org/wiki/Kernel_(statistics)`

k-NN Regression: Mean Test Error

- Suppose that we have L test points $(x_1^t, \dots, x_L^t)$ with the corresponding ground truth labels $(y_1^t, \dots, y_L^t)$

k-NN Regression: Mean Test Error

- Suppose that we have L test points $(x_1^t, \dots, x_L^t)$ with the corresponding ground truth labels $(y_1^t, \dots, y_L^t)$
- For every test point $x_j^t, j = 1, \dots, L$ our k-NN Regression algorithm returns k nearest points $(x_1^j, \dots, x_k^j)$ with the corresponding labels $(y_1^j, \dots, y_k^j)$

k-NN Regression: Mean Test Error

- Suppose that we have L test points $(x_1^t, \ldots, x_L^t)$ with the corresponding ground truth labels $(y_1^t, \ldots, y_L^t)$
- For every test point $x_j^t, j = 1, \ldots, L$ our k-NN Regression algorithm returns k nearest points $(x_1^j, \ldots, x_k^j)$ with the corresponding labels $(y_1^j, \ldots, y_k^j)$
- But how to calculate the mean test error in this case?

k-NN Regression: Mean Test Error

- Suppose that we have L test points $(x_1^t, \ldots, x_L^t)$ with the corresponding ground truth labels $(y_1^t, \ldots, y_L^t)$
- For every test point $x_j^t, j = 1, \ldots, L$ our k-NN Regression algorithm returns k nearest points $(x_1^j, \ldots, x_k^j)$ with the corresponding labels $(y_1^j, \ldots, y_k^j)$
- But how to calculate the mean test error in this case?

k-NN Regression: Mean Test Error

1. For each test point calculate the prediction of the algorithm: $a(x_j^t) = \frac{1}{k}\sum_{i=1}^{k} y_i^j$

k-NN Regression: Mean Test Error

- Suppose that we have L test points $(x_1^t, \ldots, x_L^t)$ with the corresponding ground truth labels $(y_1^t, \ldots, y_L^t)$
- For every test point $x_j^t, j = 1, \ldots, L$ our k-NN Regression algorithm returns k nearest points $(x_1^j, \ldots, x_k^j)$ with the corresponding labels $(y_1^j, \ldots, y_k^j)$
- But how to calculate the mean test error in this case?

k-NN Regression: Mean Test Error

1. For each test point calculate the prediction of the algorithm: $a(x_j^t) = \frac{1}{k}\sum_{i=1}^k y_i^j$
2. Calculate the average error: $Err_1(a) = \frac{1}{L}\sum_{j=1}^L |a(x_j^t) - y_j^t|$

k-NN Regression: Mean Test Error

- Suppose that we have L test points $(x_1^t, \dots, x_L^t)$ with the corresponding ground truth labels $(y_1^t, \dots, y_L^t)$
- For every test point $x_j^t, j = 1, \dots, L$ our k-NN Regression algorithm returns k nearest points $(x_1^j, \dots, x_k^j)$ with the corresponding labels $(y_1^j, \dots, y_k^j)$
- But how to calculate the mean test error in this case?

k-NN Regression: Mean Test Error

1. For each test point calculate the prediction of the algorithm: $a(x_j^t) = \frac{1}{k}\sum_{i=1}^{k} y_i^j$
2. Calculate the average error: $Err_1(a) = \frac{1}{L}\sum_{j=1}^{L} |a(x_j^t) - y_j^t|$

Note: It means we are just averaging the absolute error for every point-wise prediction across the test set.

Reminder: bias-variance tradeoff

Definitions

Let $y = y(x) = f(x) + \varepsilon$ be the target dependence, where $f(x)$ is the deterministic function, $\varepsilon \sim N(0, \sigma^2)$ and $a(x)$ is the machine learning algorithm.

Reminder: bias-variance tradeoff

Definitions

Let $y = y(x) = f(x) + \varepsilon$ be the target dependence, where $f(x)$ is the deterministic function, $\varepsilon \sim N(0, \sigma^2)$ and $a(x)$ is the machine learning algorithm.

$$E(y-a)^2 = \sigma^2 + variance(a) + bias^2(f, a)$$

Bias and Variance of k-NN Regression

Bias

$$bias^2(f, a) = (E(f(x_0) - a(x_0)))^2 = \left(f(x_0) - \frac{1}{k} \sum_{i=1}^{k} f(x_i) \right)^2$$

Bias and Variance of k-NN Regression

Bias

$$bias^2(f, a) = (E(f(x_0) - a(x_0)))^2 = \left(f(x_0) - \frac{1}{k}\sum_{i=1}^{k} f(x_i)\right)^2$$

Variance

$$Variance(a) = D\left(\frac{1}{k}\sum_{i=1}^{k} y(x_i)\right) = \frac{1}{k^2}D\left(\sum_{i=1}^{k} y(x_i)\right) =$$

$$= \frac{1}{k^2}D\left(\sum_{i=1}^{k}(f(x_i) + \varepsilon_i)\right) = \frac{1}{k^2}D\left(\sum_{i=1}^{k} f(x_i)\right) + \frac{1}{k^2}D\left(\sum_{i=1}^{k} \varepsilon_i\right) =$$

$$= 0 + \frac{1}{k^2}k\sigma^2 = \frac{\sigma^2}{k}$$

Bias-Variance tradeoff of k-NN Regression

$$Error(x_0) = E(a(x_0) - f(x_0))^2 = \left(f(x_0) - \frac{1}{k} \sum_{i=1}^{k} f(x_i) \right)^2 + \frac{\sigma^2}{k} + \sigma^2$$

Bias-Variance tradeoff of k-NN Regression

$$Error(x_0) = E(a(x_0) - f(x_0))^2 = \left(f(x_0) - \frac{1}{k}\sum_{i=1}^{k} f(x_i)\right)^2 + \frac{\sigma^2}{k} + \sigma^2$$

- Higher k, lower variance

Bias-Variance tradeoff of k-NN Regression

$$Error(x_0) = E(a(x_0) - f(x_0))^2 = \left(f(x_0) - \frac{1}{k} \sum_{i=1}^{k} f(x_i) \right)^2 + \frac{\sigma^2}{k} + \sigma^2$$

- Higher k, lower variance
- Higher k, higher bias

Bias-Variance tradeoff of k-NN Regression

$$Error(x_0) = E(a(x_0) - f(x_0))^2 = \left(f(x_0) - \frac{1}{k}\sum_{i=1}^{k} f(x_i)\right)^2 + \frac{\sigma^2}{k} + \sigma^2$$

- Higher k, lower variance
- Higher k, higher bias

Note: Under "reasonable assumptions" the bias of the 1-NN estimator vanishes entirely as the size of the training set approaches infinity

Mandatory external links to read

1. Read the Introduction to K-Nearest Neighbor (kNN) algorithm for the Regression task
   - Main source, more thorough explanation, and rigorous algorithm overview.
   - A video covering a practical example.

Conclusion

- The main advantage of non-parametric regression is the absence of assumptions about the form of the dependence model,

Conclusion

- The main advantage of non-parametric regression is the absence of assumptions about the form of the dependence model,
- The method has a large number of variations to customize,

Conclusion

- The main advantage of non-parametric regression is the absence of assumptions about the form of the dependence model,
- The method has a large number of variations to customize,
    - Metric learning (e.g., l_*-metric variations),

Conclusion

- The main advantage of non-parametric regression is the absence of assumptions about the form of the dependence model,
- The method has a large number of variations to customize,
  - Metric learning (e.g., l_*-metric variations),
  - Number of nearest neighbors k,

Conclusion

- The main advantage of non-parametric regression is the absence of assumptions about the form of the dependence model,
- The method has a large number of variations to customize,
  - Metric learning (e.g., l_*-metric variations),
  - Number of nearest neighbors k,
  - Weights in the weighted version of the method,

Conclusion

- The main advantage of non-parametric regression is the absence of assumptions about the form of the dependence model,
- The method has a large number of variations to customize,
  - Metric learning (e.g., l_*-metric variations),
  - Number of nearest neighbors k,
  - Weights in the weighted version of the method,
  - Smoothing window width;
- To compute the k-NN Regression Prediction we are averaging the nearest neighbors labels,

Conclusion

- The main advantage of non-parametric regression is the absence of assumptions about the form of the dependence model,
- The method has a large number of variations to customize,
  - Metric learning (e.g., l_*-metric variations),
  - Number of nearest neighbors k,
  - Weights in the weighted version of the method,
  - Smoothing window width;
- To compute the k-NN Regression Prediction we are averaging the nearest neighbors labels,
- To compute the k-NN Regression Mean (Absolute) Test Error we are averaging the absolute error for every point-wise prediction across the test set.

Thank you!