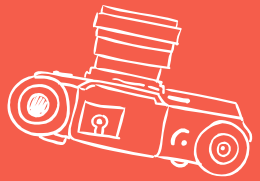
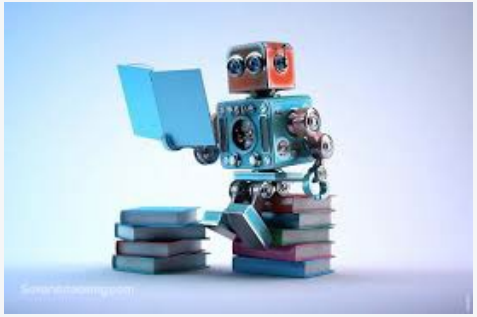
به نام خدا



یادگیری ماشین



یادگیری ماشین

آرش عبدی هجراندوست

arash.abdi.hejrandoost@gmail.com

دانشگاه علم و صنعت

دانشکده مهندسی کامپیوتر

~~نیم سال اول ۱۴۰۱-۱۴۰۲~~

مدل های بدون پارامتر

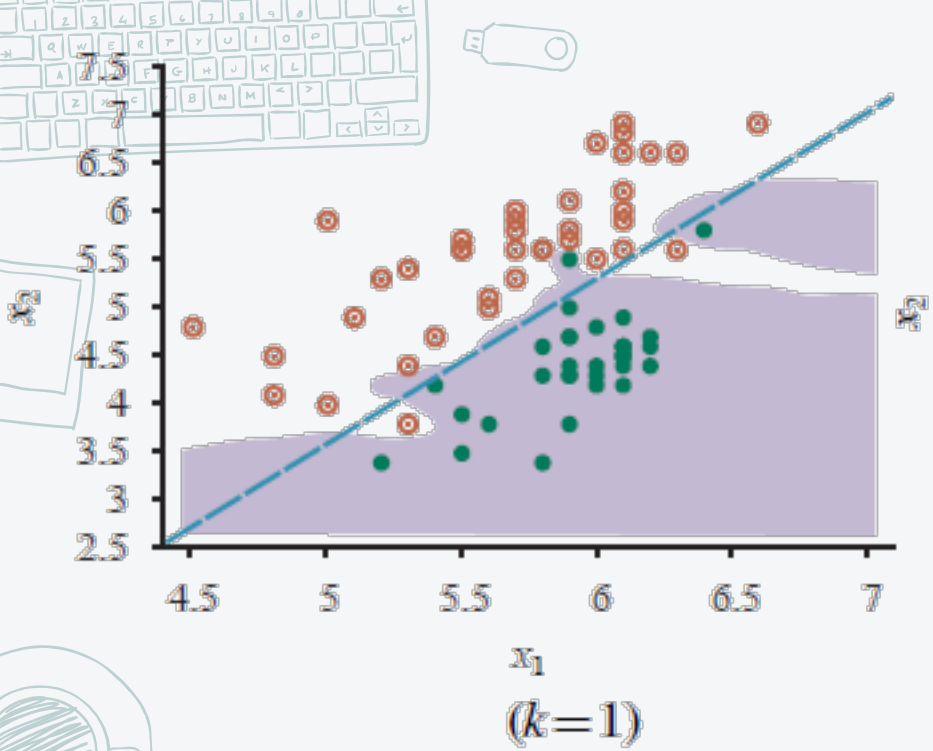
Nonparametric Models

- مدلهای با پارامتر
  - خلاصه سازی داده ها در فرمی ساده
  - درخت تصمیم
  - رگرسون خطی (لجستیک-پرسپترون)
- حجم داده بالا
- مردم خود نظر بدهند! نماینده نداشته باشند!

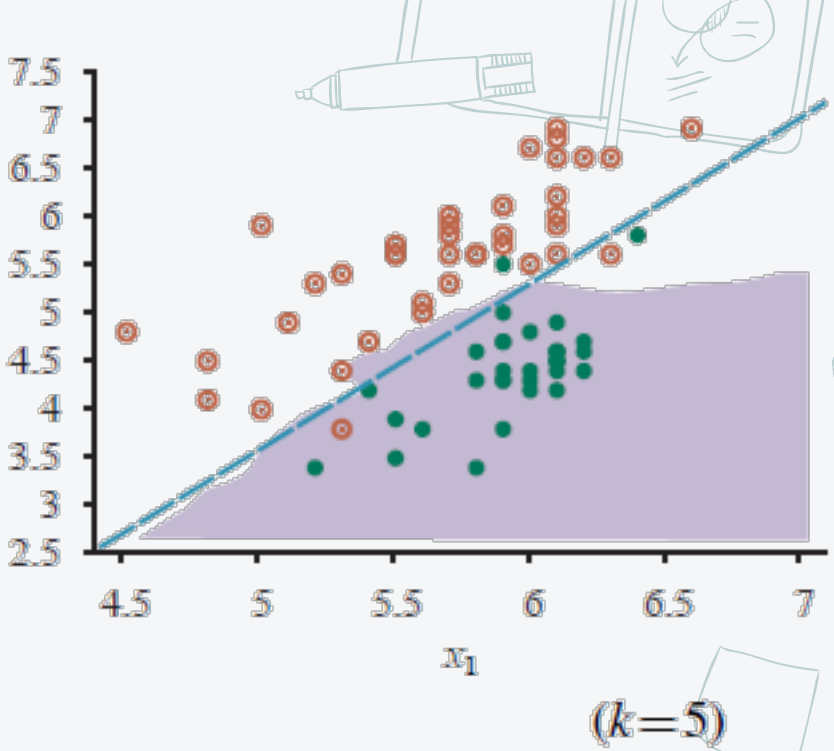
- یادگیری مبتنی بر نمونه – Instance based
- یادگیری مبتنی بر حافظه – Memory based

نزدیک ترین همسایه (۱)
Nearest-Neighbor

- Lookup Table
- K نزدیک ترین همسایه
- دسته بندی:
  - نظر اکثریت (ترجیحا تعداد فرد)
- تقریب تابع:
  - میانگین
  - میانه
    - (مزیت؟)
  - تقریب تابع خطی یا ...
    - piecewise linear function



$(k=1)$



$(k=5)$

Overfit ✗

Underfit ✗

Cross-validation ✗

معیار فاصله

- نرم های p (فاصله Minkowski)

$$L^P(\mathbf{x}_j, \mathbf{x}_q) = \left(\sum_i |x_{j,i} - x_{q,i}|^P\right)^{1/P}.$$

- P=2 : فاصله اقلیدسی
- P=1: فاصله منهتن
  - مقادیر بولین: فاصله همینگ (تعداد درایه های متفاوت)

نرمال سازی

- اگر بازه ویژگی ها در یک بردار ویژگی متفاوت باشد؟
  - قد برحسب متر
  - وزن برحسب گرم
  - فاصله دو نقطه از هم؟
- نرمالسازی در هر بعد:
  - $(x_{i,j} - \mu_i)/\sigma_i$
- فاصله Mahalanobis
- $\sqrt{(x-y)^T S^{-1}(x-y)}, \quad S = \text{covariance matrix}$
  - معادل تغییر مقیاس دادن بردارها برای داشتن واریانس واحد و سپس محاسبه فاصله اقلیدسی در فضای جدید
  - معیاری مستقل از مقیاس ویژگی ها

نفرین ابعاد – curse of dimensionality

- داده های حجیم با ابعاد کم ← کارآمدی NN
- ابعاد بالاتر ← فاصله همسایه ها از هم دور می‌شود!
  - همسایه ها از آنچه فکر می‌کنید از شما دورترند!
- مکعبی با طول واحد و ابعاد n
- تعداد N نمونه با توزیع یکنواخت
- تعریف همسایگی: کوچکترین مکعبی دور نقطه مد نظر که شامل k نقطه باشد.
  - طول مکعب همسایگی: L و حجم آن: L^n (حجم کل مکعب با N نمونه 1)

$$L^n = \frac{k}{N} \leftarrow L = \sqrt[\frac{1}{n}]{\frac{k}{N}}$$

$$k = 10,\ N = 1{,}000{,}000$$

- $n = 2 \leftarrow l = 0.003$
- $n = 3 \leftarrow l = 0.02$
- $n = 17 \leftarrow l = 0.5$
- $n = 200 \leftarrow l = 0.94$

- در مکعب مذکور، لایه بیرونی با قطر ۱ ٪ را در نظر بگیرید.
- داده های حاشیه ای و دشوار (برون یابی به جای درون یابی)
- در فضای یک بعدی: حجم لایه بیرونی: ۲ ٪
- در فضای ۲۰۰ بعدی: ۹۸ ٪ $(0.98^{200} = 0.0176)$

زمان اجرای NN

- ❌ $O(N)$
- ❌ داده های حجیم؟
- ❌ نیاز به سرعت بیشتر
- ❌ K-d tree

K-d tree

- داده ها در هر گره بر اساس مقدار میانه (m) به دو زیرگروه تقسیم شوند.
  - تاوقتی که یک نمونه داده در برگها بماند
- برای انتخاب ویژگی در هر گره (از n ویژگی):
  - در سطح i از درخت: ویژگی $i \bmod n$
  - انتخاب ویژگی با بیشترین واریانس

K-d tree

- جستجوی دقیق: OK
- جستجوی همسایه نزدیک؟
  - وقتی به برگ میرسیم لزوما نزدیک ترین همسایه نیست!
  - شاید در یکی از گره‌ها که به چپ پیچیدیم، اندکی راست تر، نزدیک ترین همسایه باشد!
    - میتوان تصمیمات لب مرزی را یادداشت کرد و در برگ ها چک کرد اگر لازم است طرف دیگر تصمیمات لب مرزی هم چک شود!
      - اگر k همسایه که فاصله شان از آن لب مرز کمتر نبود، پیدا نشد ...
      - کرکتیف کاری! ☺

K-d tree

- همسایه بعدی چگونه پیدا شود؟
- زمان اجرا؟
- زمان سریع تر؟
  - درهم سازی

درهم سازی مبتنی بر موقعیت
locality-sensitive hashing-LSH

- درهم سازی برای تطبیق دقیق است.
- ما به دنبال تقریب و همسایه هستیم.
- درهم سازی، ورودی ها را تصادفا در binهای خروجی توزیع میکند.
- میخواهیم نقاط نزدیک به هم در bin های یکسان قرار گیرند
  - LSH
- الگوریتمی **تقریبی** برای استفاده از درهم سازی در مساله NN
  - با درهم سازی نزدیکترین همسایه ها به طور دقیق قابل یافتن نیستند.

درهم سازی مبتنی بر موقعیت
locality-sensitive hashing-LSH

- صورت مساله:
  - با داشتن مجموعه ای از نقاط و ورودی x_q با احتمال بالا نقطه یا نقاطی نزدیک به x_q را بیاب.
- اگر دو نقطه در فضای n بعدی نزدیک به هم باشند، وقتی به فضای یک بعدی (یک خط) نگاشت شوند نیز نزدیک به هم خواهند بود.
- خط مذکور را به چند bin تقسیم می‌کنیم.
- با احتمال بالا نقاط همسایه در یک bin قرار می‌گیرند.
- برخی نقاط دور هم اشتباها در bin نقطه فعلی خواهند بود.

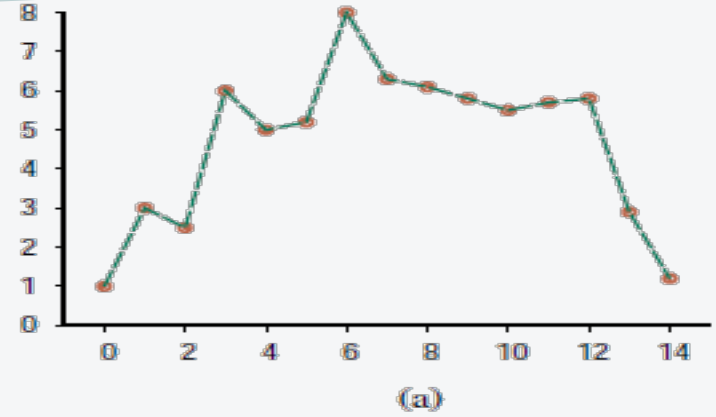
درهم سازی مبتنی بر موقعیت
locality-sensitive hashing-LSH

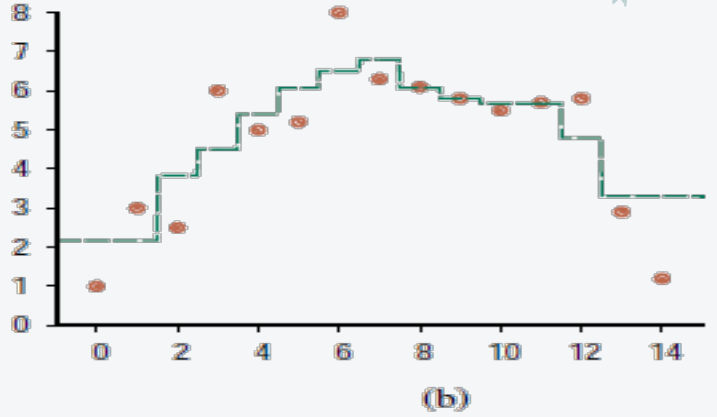
- SLH
  - در نظر گرفتن چند نگاشت تصادفی و ترکیب آنها
  - تبدیل داده ها به bit string
- l نگاشت تصادفی ← l جدول درهم سازی
- قرار دادن تمام نمونه ها در جداول
- برای داده ورودی x_q تمام نقاطی که هم bin نقطه ورودی در هر یک از جداول هستند به عنوان مجموعه کاندید در نظر گرفته می‌شوند.
- محاسبه فاصله واقعی نقاط کاندید با نقطه ورودی
- در نظر گرفتن k نزدیک همسایه به عنوان خروجی

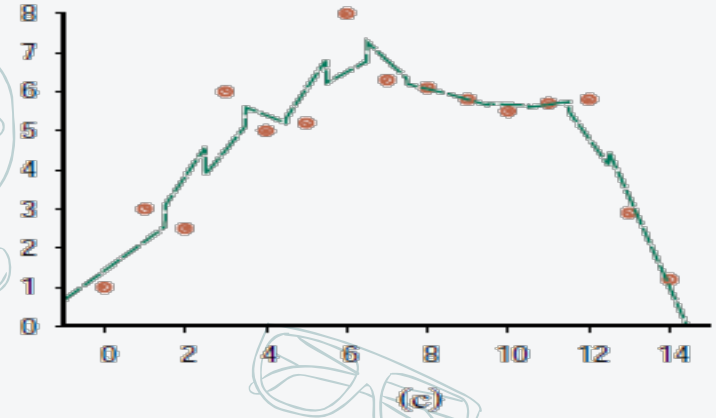
- ❌ تعداد bin ؟
- ❌ تعداد نگاشت ؟
- ❌ زمان؟

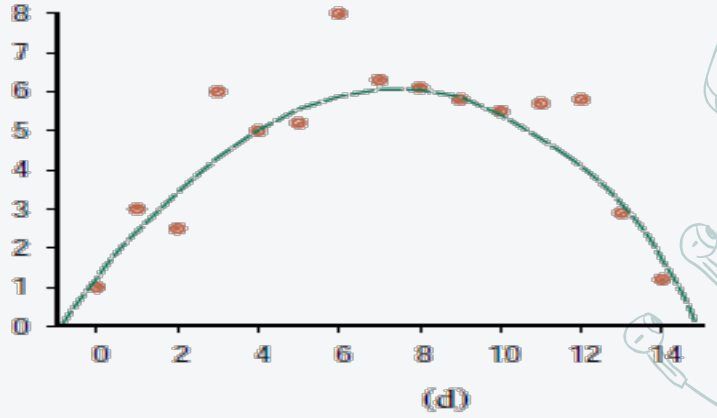
- ❌ در مجموعه ای از ۱۳ میلیون تصویر ۵۱۲ بعدی، SLH فقط با تست چند هزار تصویر نزدیک ترین همسایه ها را می‌یابد.

nonparametric regression – تقریب تابع بدون پارامتر









تقریب تابع بدون پاراصتر

✕ اتصال نقاط (a):

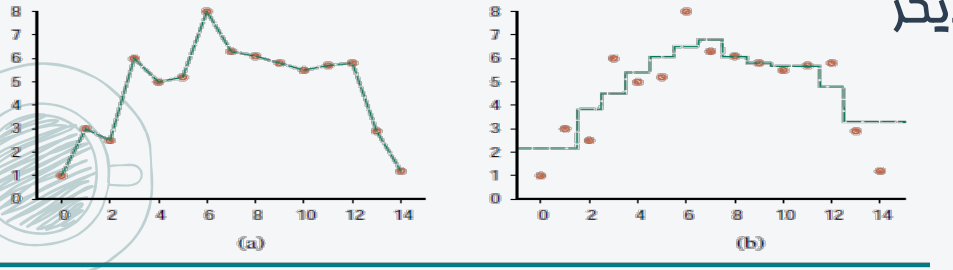
- یک نقطه قبل و یک نقطه بعد از نقطه ورودی را به هم وصل کن
- چند بعدی: قبل ؟ بعد؟
- برخورد با نویز؟
  - اطراف نویز، تقریب غلط خواهد بود

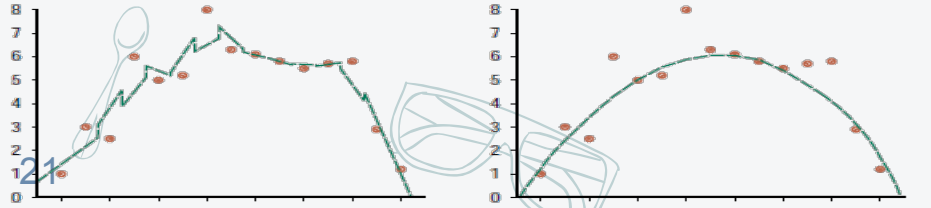
✕ تقریب با k نزدیک ترین همسایه

- به جای یک نقطه قبل و بعد: چند همسایه نزدیک (هر طرف که باشند)
- همسایه های بیشتر: خروجی تقریب نرم تر
- متوسط گیری (b)
- تقریب خطی k نزدیک ترین همسایه (c)
  - مقدار k ؟

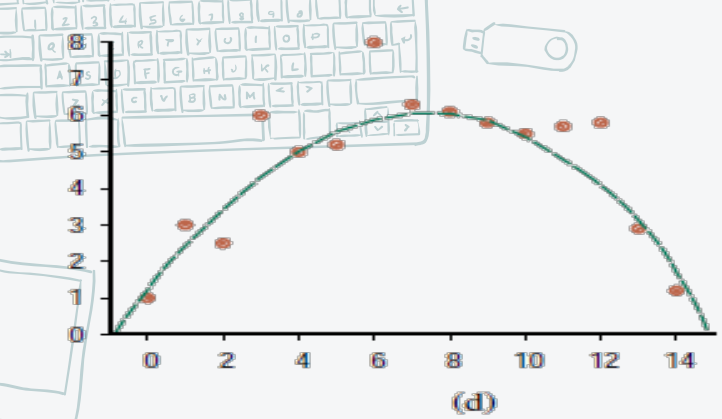
تقریب تابع بدون پاراصتر
تقریب وزن دار محلی – Locally Weighted Regression

- نمودارهای a تا c پرش و تغییر ناگهانی دارند.
- به همسایه نزدیک تر وزن بیشتر بدهیم!
- تابع kernel (هسته) تعیین کننده میزان اهمیت (وزن) است.
  - ورودی: فاصله بین نقطه ورودی و نقطه دیگر





تقریب وزن دار محلی



- یک هسته درجه ۲
  - $k(d) = \max(0, 1 - \left(\frac{2|d|}{w}\right)^2)$
  - W طول پنجره: مثلا ۱۰ (پارامتر آزاد)
- می‌توان هسته گاوسی یا ... در نظر گرفت.
- پنجره خیلی عریض: underfit
  - و برعکس
- نحوه تقریب:

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_j \mathcal{K}(Distance(\mathbf{x}_q, \mathbf{x}_j))\,(y_j - \mathbf{w} \cdot \mathbf{x}_j)^2$$

$$h(x_q) = w^*.x_q$$

- برای هر ورودی، حل بهینه سازی فوق (ان شاءالله(!) برای تعداد کمی نقطه)

با تشکر