# Học Máy

## (Machine Learning)

**Thân Quang Khoát**

*khoattq@soict.hust.edu.vn*

Viện Công nghệ thông tin và Truyền thông
Trường Đại học Bách Khoa Hà Nội

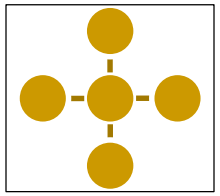
Năm 2015

# Nội dung môn học:

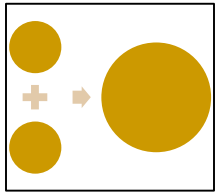
- Giới thiệu chung
- Các phương pháp học không giám sát
- **Các phương pháp học có giám sát**
  - **Học dựa trên các láng giềng gần nhất (Nearest neighbors learning)**
- Đánh giá hiệu năng hệ thống học máy

# Các bạn phân loại thế nào?

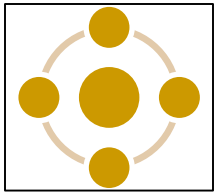
Class a



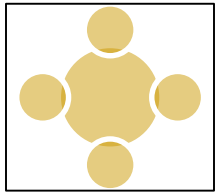
Class b



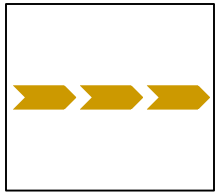
Class a



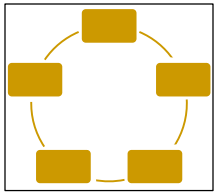
Class a



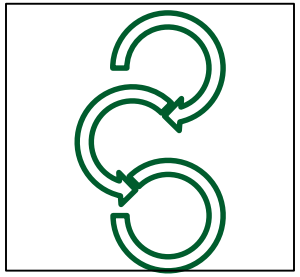
??



Class a



Class b



# Học dựa trên các láng giềng gần nhất

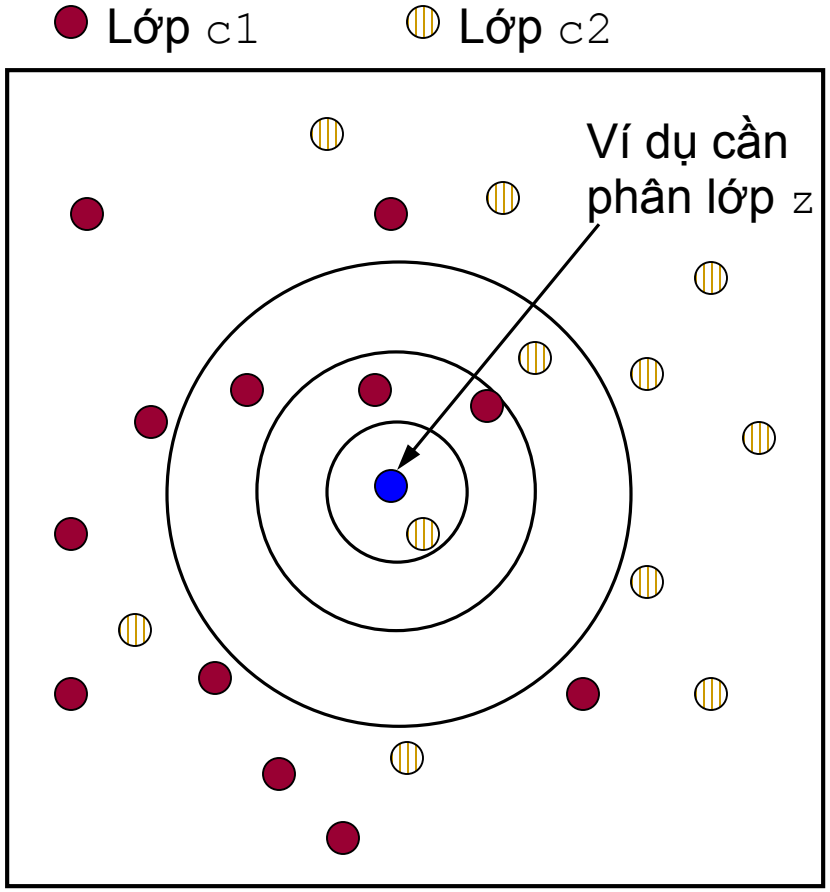
- Một số tên gọi khác của phương pháp **học dựa trên các láng giềng gần nhất (Nearest neighbors learning)**
  - Instance-based learning
  - Lazy learning
  - Memory-based learning
- Ý tưởng của phương pháp
  - Quá trình học
    - (Đơn giản là) lưu lại các ví dụ học
    - Không xây dựng một mô hình (mô tả) rõ ràng và tổng quát của hàm mục tiêu cần học
  - Đối với một ví dụ cần phân loại/dự đoán
    - Giá trị của hàm mục tiêu (một nhãn lớp, hoặc một giá trị thực) được suy ra từ các hàng xóm của nó.

# Học dựa trên các láng giềng gần nhất

- Biểu diễn đầu vào của bài toán
  - Mỗi ví dụ $\boldsymbol{x}$ được biểu diễn là một vectơ $n$ chiều trong không gian các vectơ $\boldsymbol{x} \in R^n$
  - $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, trong đó $x_i \in R$ là một số thực
- Có thể áp dụng được với cả 2 kiểu bài toán
  - Bài toán *phân lớp (classification)*
    - Hàm mục tiêu có giá trị rời rạc
    - Đầu ra của hệ thống là một trong số các giá trị rời rạc đã xác định trước (một trong các nhãn lớp)
  - Bài toán *hồi quy (regression)*
    - Hàm mục tiêu có giá trị liên tục
    - Đầu ra của hệ thống là một giá trị số thực

# Ví dụ: bài toán phân lớp

- Xét 1 láng giềng gần nhất
  → Gán `z` vào lớp `c2`
- Xét 3 láng giềng gần nhất
  → Gán `z` vào lớp `c1`
- Xét 5 láng giềng gần nhất
  → Gán `z` vào lớp `c1`



# Giải thuật k-NN cho phân lớp

- Mỗi ví dụ học $\boldsymbol{x}$ được biểu diễn bởi 2 thành phần:
  - Mô tả của ví dụ: $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, trong đó $x_i \in R$
  - Nhãn lớp : $c \in C$, với $C$ là tập các nhãn lớp được xác định trước
- Giai đoạn học
  - Đơn giản là lưu lại các ví dụ học trong tập học: $\boldsymbol{D}$
- Giai đoạn phân lớp: Để phân lớp cho một ví dụ (mới) $\boldsymbol{z}$
  - Với mỗi ví dụ học $\boldsymbol{x} \in \boldsymbol{D}$, tính khoảng cách giữa $\boldsymbol{x}$ và $\boldsymbol{z}$
  - Xác định tập $NB(\boldsymbol{z})$ – các láng giềng gần nhất của $\boldsymbol{z}$

    $\rightarrow$ Gồm $\boldsymbol{k}$ ví dụ học trong $\boldsymbol{D}$ gần nhất với $\boldsymbol{z}$ tính theo một hàm khoảng cách $d$
  - Phân $\boldsymbol{z}$ vào lớp chiếm số đông (the majority class) trong số các lớp của các ví dụ trong $NB(\boldsymbol{z})$

# Giải thuật k-NN cho hồi quy

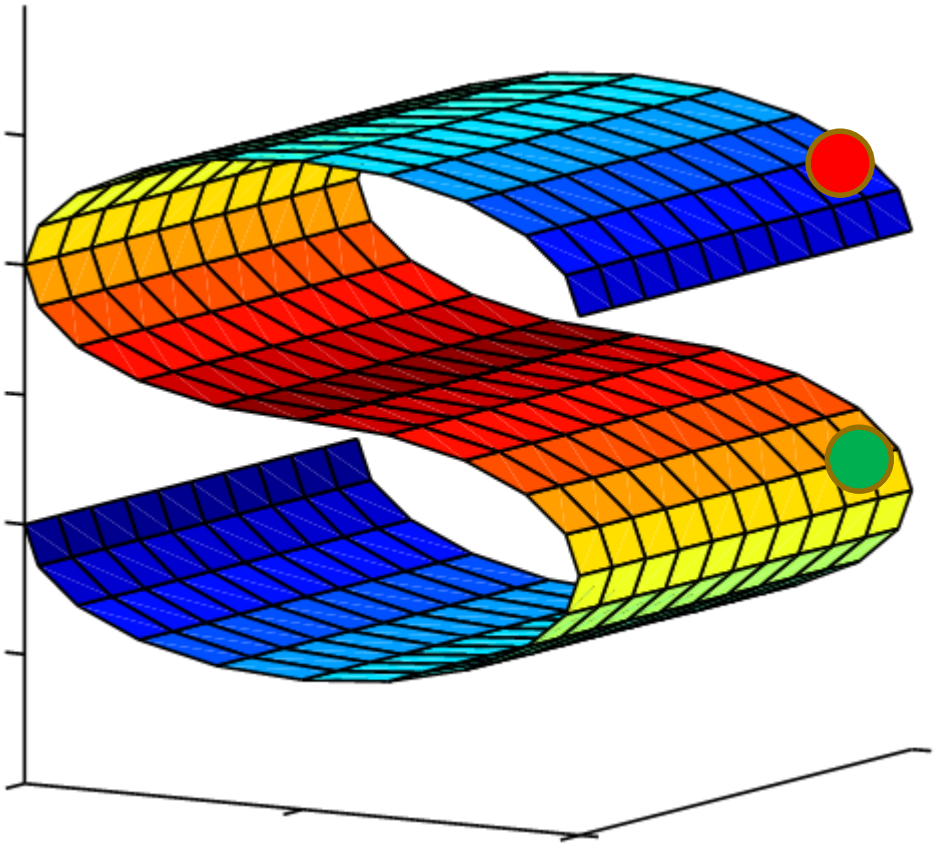
- Mỗi ví dụ học $\boldsymbol{x}$ được biểu diễn bởi 2 thành phần:
  - Mô tả của ví dụ: $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, trong đó $x_i \in R$
  - Giá trị đầu ra mong muốn: $y_x \in R$ (là một số thực)
- Giai đoạn học
  - Đơn giản là lưu lại các ví dụ học trong tập học $\boldsymbol{D}$
- Giai đoạn dự đoán: Để dự đoán giá trị đầu ra cho ví dụ $\boldsymbol{z}$
  - Đối với mỗi ví dụ học $\boldsymbol{x} \in \boldsymbol{D}$, tính khoảng cách giữa $\boldsymbol{x}$ và $\boldsymbol{z}$
  - Xác định tập $NB(\boldsymbol{z})$ – các láng giềng gần nhất của $\boldsymbol{z}$
    → Gồm $\boldsymbol{k}$ ví dụ học trong $\boldsymbol{D}$ gần nhất với $\boldsymbol{z}$ tính theo một hàm khoảng cách $d$
  - Dự đoán giá trị đầu ra đối với $\boldsymbol{z}$: $y_z = \frac{1}{k} \sum_{\boldsymbol{x} \in NB(\boldsymbol{z})} y_x$

# k-NN: Các vấn đề cốt lõi

Suy nghĩ khác nhau!

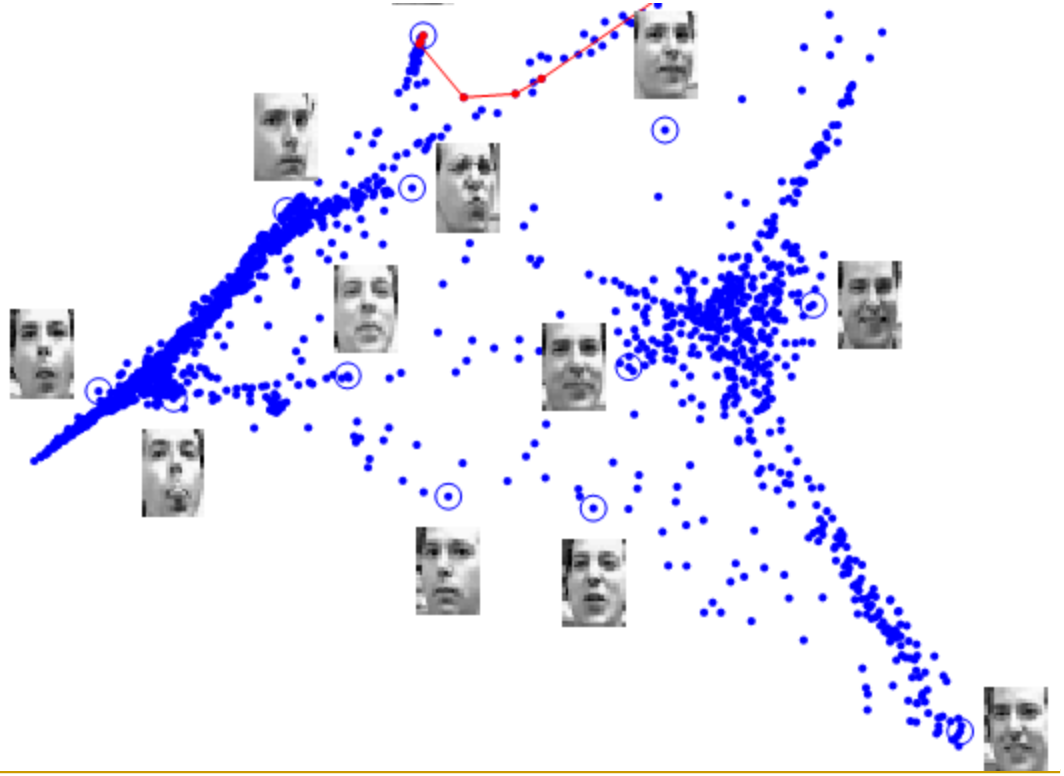
# k-NN: Các vấn đề cốt lõi

- Hàm khoảng cách
  - Mỗi hàm sẽ tương ứng với một cách nhìn về dữ liệu.
  - Vô hạn hàm!!!
  - Chọn hàm nào?



# k-NN: Các vấn đề cốt lõi

- Chọn tập láng giềng $NB(\boldsymbol{z})$
  - Chọn bao nhiêu láng giềng?
  - Giới hạn chọn theo vùng?



# Một hay nhiều láng giềng gần nhất?

- Việc phân lớp (hay dự đoán) chỉ dựa trên duy nhất một láng giềng gần nhất (là ví dụ học gần nhất với ví dụ cần phân lớp/dự đoán) thường *không* chính xác
  - Nếu ví dụ học này là một ví dụ bất thường, không điển hình (an outlier) – rất khác so với các ví dụ khác
  - Nếu ví dụ học này có nhãn lớp (giá trị đầu ra) sai – do lỗi trong quá trình thu thập (xây dựng) tập dữ liệu
- Thường xét $k$ $(> 1)$ các ví dụ học (các láng giềng) gần nhất với ví dụ cần phân lớp/dự đoán
- Đối với bài toán phân lớp có 2 lớp, $k$ thường được chọn là một số lẻ, để tránh cân bằng về tỷ lệ các ví dụ giữa 2 lớp
  - Ví dụ: $k = 3, 5, 7, \ldots$

# Hàm tính khoảng cách (1)

- Hàm tính khoảng cách ***d***
  - Đóng vai trò rất quan trọng trong phương pháp học dựa trên các láng giềng gần nhất
  - Thường được xác định trước, và không thay đổi trong suốt quá trình học và phân loại/dự đoán
- Lựa chọn hàm khoảng cách ***d***
  - *Các hàm khoảng cách hình học*: Dành cho các bài toán có các thuộc tính đầu vào là kiểu số thực ($x_i \in R$)
  - *Hàm khoảng cách Hamming*: Dành cho các bài toán có các thuộc tính đầu vào là kiểu nhị phân ($x_i \in \{0,1\}$)

# Hàm tính khoảng cách (2)

- Các hàm tính khoảng cách hình học (Geometry distance functions)
  - Hàm Minkowski ($p$-norm):

$$d(x,z) = \left( \sum_{i=1}^{n} \left| x_i - z_i \right|^p \right)^{1/p}$$

  - Hàm Manhattan ($p = 1$):

$$d(x,z) = \sum_{i=1}^{n} \left| x_i - z_i \right|$$

  - Hàm Euclid ($p = 2$):

$$d(x,z) = \sqrt{\sum_{i=1}^{n} (x_i - z_i)^2}$$

  - Hàm Chebyshev ($p = \infty$):

$$d(x,z) = \lim_{p \to \infty} \left( \sum_{i=1}^{n} \left| x_i - z_i \right|^p \right)^{1/p}$$

$$= \max_i \left| x_i - z_i \right|$$

# Hàm tính khoảng cách (3)

- Hàm khoảng cách Hamming
  - Đối với các thuộc tính đầu vào là kiểu nhị phân ($\{0,1\}$)
  - Ví dụ: $\boldsymbol{x} = (0,1,0,1,1)$

$$d(x, z) = \sum_{i=1}^{n} Difference \quad (x_i, z_i)$$

$$Difference \quad (a, b) = \begin{cases} 1, if \ (a \neq b) \\ 0, if \ (a = b) \end{cases}$$

# Chuẩn hóa miền giá trị thuộc tính

- Hàm tính khoảng cách Euclid: $d(x, z) = \sqrt{\sum_{i=1}^{n} (x_i - z_i)^2}$
- Giả sử mỗi ví dụ được biểu diễn bởi 3 thuộc tính: `Age`, `Income` (cho mỗi tháng), và `Height` (đo theo mét)
  - $\boldsymbol{x}$ = (`Age`=20, `Income`=12000, `Height`=1.68)
  - $\boldsymbol{z}$ = (`Age`=40, `Income`=1300, `Height`=1.75)
- Khoảng cách giữa $\boldsymbol{x}$ và $\boldsymbol{z}$
  - $d(\boldsymbol{x}, \boldsymbol{z}) = [(20 - 40)^2 + (12000 - 1300)^2 + (1.68 - 1.75)^2]^{0.5}$
  - Giá trị khoảng cách bị quyết định chủ yếu bởi giá trị khoảng cách (sự khác biệt) giữa 2 ví dụ đối với thuộc tính `Income`

    → Vì: Thuộc tính `Income` có miền giá trị rất lớn so với các thuộc tính khác
- Cần phải chuẩn hóa miền giá trị (đưa về cùng một khoảng giá trị)
  - Khoảng giá trị [0,1] thường được sử dụng
  - Đối với mỗi thuộc tính `i`: $x_i := x_i / \max(x_j)$

# Trọng số của các thuộc tính

- Hàm khoảng cách Euclid:

$$d(x,z) = \sqrt{\sum_{i=1}^{n} (x_i - z_i)^2}$$

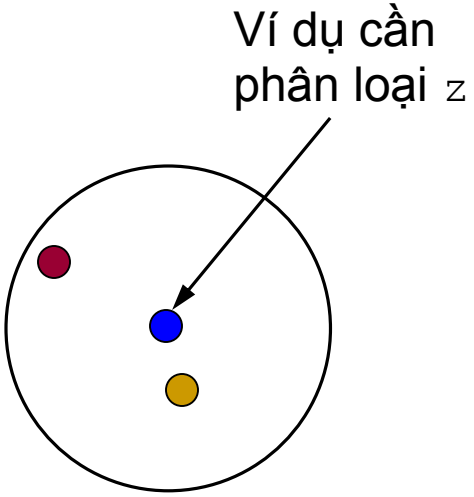
  - Tất cả các thuộc tính có cùng (như nhau) ảnh hưởng đối với giá trị khoảng cách
- **Các thuộc tính khác nhau** có thể (nên) có **mức độ ảnh hưởng khác nhau** đối với giá trị khoảng cách
- Cần phải tích hợp (đưa vào) các giá trị trọng số của các thuộc tính trong hàm tính khoảng cách
  - $w_i$ là trọng số của thuộc tính `i`:

$$d(x,z) = \sqrt{\sum_{i=1}^{n} w_i (x_i - z_i)^2}$$

- Làm sao để xác định các giá trị trọng số của các thuộc tính?
  - Dựa trên các tri thức cụ thể của bài toán (vd: được chỉ định bởi các chuyên gia trong lĩnh vực của bài toán đang xét)
  - Bằng một quá trình tối ưu hóa các giá trị trọng số (vd: sử dụng một tập học để học một bộ các giá trị trọng số tối ưu)

# Khoảng cách của các láng giềng (1)

- Xét tập $NB(\boldsymbol{z})$ – gồm $k$ ví dụ học gần nhất với ví dụ cần phân lớp/dự đoán $\boldsymbol{z}$
  - Mỗi ví dụ (láng giềng gần nhất) này có khoảng cách khác nhau đến $\boldsymbol{z}$
  - Các láng giềng này có ảnh hưởng như nhau đối với việc phân lớp/dự đoán cho $\boldsymbol{z}$? → KHÔNG!
- Nên gán các mức độ ảnh hưởng (đóng góp) của mỗi láng giềng gần nhất tùy theo khoảng cách của nó đến $\boldsymbol{z}$
  - Mức độ ảnh hưởng cao hơn cho các láng giềng gần hơn!



# Khoảng cách của các láng giềng (2)

- Gọi $v$ là hàm xác định trọng số theo khoảng cách
  - Đối với một giá trị $d(\boldsymbol{x},\boldsymbol{z})$ – khoảng cách giữa $\boldsymbol{x}$ và $\boldsymbol{z}$
  - $v(\boldsymbol{x},\boldsymbol{z})$ tỷ lệ nghịch với $d(\boldsymbol{x},\boldsymbol{z})$
- Đối với bài toán phân lớp: $c(z) = \arg\max_{c_j \in C} \sum_{x \in NB(z)} v(x,z).Identical\ (c_j, c(x))$

$$Identical\ (a,b) = \begin{cases} 1, if\ (a = b) \\ 0, if\ (a \neq b) \end{cases}$$

- Đối với bài toán dự đoán (hồi quy): $f(z) = \dfrac{\sum_{x \in NB(z)} v(x,z).f(x)}{\sum_{x \in NB(z)} v(x,z)}$
- Lựa chọn một hàm xác định trọng số theo khoảng cách:

$$v(x,z) = \frac{1}{\alpha + d(x,z)} \qquad v(x,z) = \frac{1}{\alpha + [d(x,z)]^2} \qquad v(x,z) = e^{-\frac{d(x,z)^2}{\sigma^2}}$$

# Lazy learning vs. Eager learning

- **Lazy learning**. Việc đánh giá hàm mục tiêu (target function) được hoãn lại cho đến khi xét ví dụ cần phân loại/dự đoán
  - Đánh giá (xấp xỉ) hàm mục tiêu *một cách cục bộ (locally)* và *riêng rẽ (diferrently)* cho mỗi ví dụ cần phân loại/dự đoán (*tại thời điểm phân loại/dự đoán của hệ thống*)
  - Tính toán nhiều lần các xấp xỉ *cục bộ* của hàm mục tiêu
  - Thường mất thời gian lâu hơn để đưa ra kết luận (phân lớp/dự đoán), và cần nhiều không gian nhớ hơn
  - Ví dụ: Nearest neighbor learner, Locally weighted regression
- **Eager learning**. Việc đánh giá hàm mục tiêu được hoàn thành trước khi xét đến bất kỳ ví dụ cần phân loại/dự đoán
  - Đánh giá (xấp xỉ) hàm mục tiêu *một cách tổng thể (globally)* đối với toàn bộ không gian các ví dự (*tại thời điểm học của hệ thống)*
  - Tính toán một xấp xỉ *duy nhất (ở mức tổng thể)* của hàm mục tiêu
  - Ví dụ: Linear regression, Support vector machines, Neural networks, ...

# k-NN: Ưu nhược điểm

- Các ưu điểm
  - Chi phí thấp cho quá trình huấn luyện (chỉ việc lưu lại các ví dụ học)
  - Hoạt động tốt với các bài toán phân loại gồm nhiều lớp
    → Không cần phải học $c$ bộ phân loại cho $c$ lớp
  - Phương pháp học k-NN ($k >> 1$) có khả năng xử lý nhiễu cao
    → Phân loại/dự đoán được thực hiện dựa trên $k$ láng giềng gần nhất
  - Rất Linh động trong việc chọn hàm khoảng cách.
    → Có thể dùng độ tương tự (similarity): cosine
    → Có thể dùng độ đo khác, chẳng hạn Kullback-Leibler divergence, Bregman divergence, …
- Các nhược điểm
  - Phải lựa chọn hàm tính khoảng cách (sự khác biệt) thích hợp với bài toán
  - Chi phí tính toán (thời gian, bộ nhớ) cao tại thời điểm phân loại/dự đoán
  - Có thể cho kết quả kém/sai với các thuộc tính không liên quan

# k-NN: bài tập

- K-NN khác gì so với phương pháp Least squares?
- Đánh giá khả năng overfitting của phương pháp k-NN?