Data Preprocessing

Why data preprocessing

- Today's real-world databases are noisy, contain missing values and inconsistent due to their huge size.
- A good preprocess data before learning not only "improve the quality of learning results" but also "ease the learning process".
- There are a number a data preprocessing techniques
  - Data Cleaning:- Remove noise, correct inconsistencies
  - Data Integration:- Merge data from multiple sources
  - Data Transformation:- Improve distance measurements
  - Data Reduction:- Eliminate redundant features

Data Cleaning

- Data cleaning routines
  - Missing values.
  - Identify outliers.
  - Correct inconsistencies.

- Missing values:-
  - You can note that many tuples have no recorded value for several attributes, such as country, Refund.

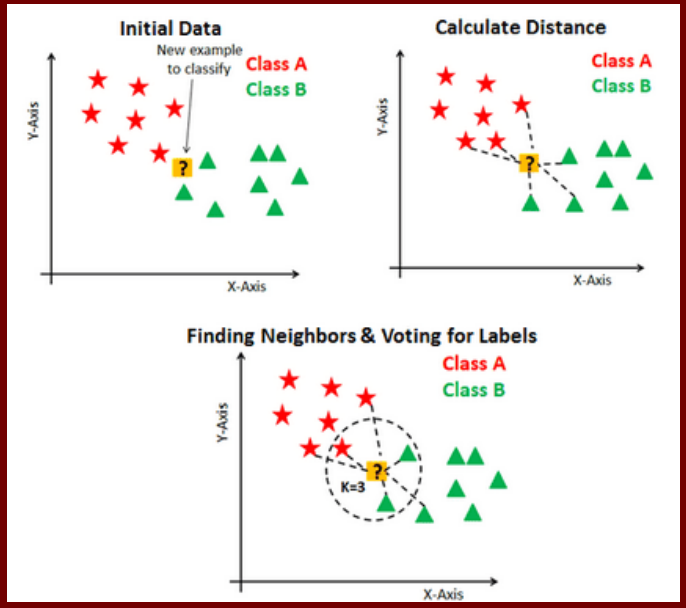
TID	Refund	Country	Taxable Income	Cheat
1	Yes	USA	125K	No
2		UK	100K	No
3	No	Australia	70K	No
4			120K	No
5	No	NZL	95K	Yes
6	No	USA	60K	No
7	Yes	UK	220K	No
8	No		85K	Yes
9	No	UK		No
10	No			Yes

Missing values Techniques

- Ignore the record:-
  - This is usually done when the class label is missing.
- Use the attribute mean to fill in the missing value:-
  - For example, take the average value of income attribute. Use this value to replace the missing value for income.
- Use the attribute Median to fill in the missing value:-
  - The main difference between Mean and Median is that Median is robust to noise.
- Use classification rules to fill in the missing value:-
  - This can be determined with decision tree, neural network, Bayesian Modeling.

Missing values Techniques (cont)

- K Nearest Neighbors
  - KNN is a supervised learning algorithm where the missing values are replaced based on nearest K neighbor. **Neighbors are determined based on distance measure**.
  - Once K neighbors are determined then we can use mean or median for missing value of continuous attribute.



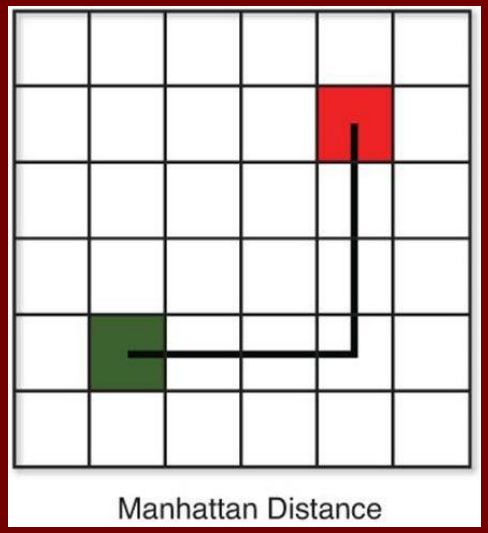
K Nearest Neighbors

Distance functions

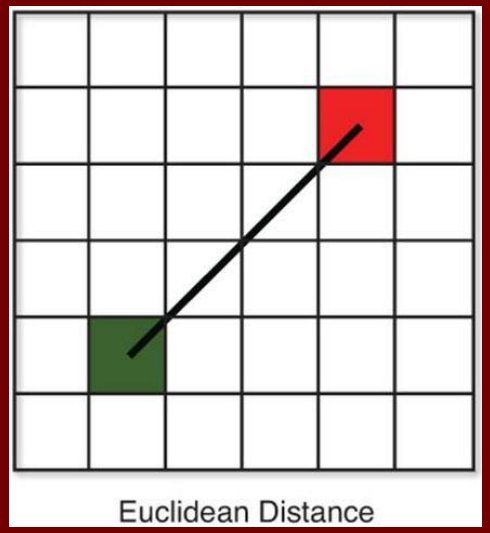
Euclidean $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$

Manhattan $\sum_{i=1}^{k}|x_i - y_i|$

Minkowski $\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$



Manhattan Distance



Euclidean Distance

KNN classifier example

Sepal Length	Sepal Width	Species
5.3	3.7	Setosa
5.1	3.8	Setosa
7.2	3.0	Virginica
5.4	3.4	Setosa
5.1	3.3	Setosa
5.4	3.9	Setosa
7.4	2.8	Virginica
6.1	2.8	Verscicolor
7.3	2.9	Virginica
6.0	2.7	Verscicolor
5.8	2.8	Virginica
6.3	2.3	Verscicolor
5.1	2.5	Verscicolor
6.3	2.5	Verscicolor
5.5	2.4	Verscicolor

Sepal Length	Sepal Width	Species
5.2	3.1	?

KNN classifier example

Sepal Length	Sepal Width	Species
5.3	3.7	Setosa
5.1	3.8	Setosa
7.2	3.0	Virginica
5.4	3.4	Setosa
5.1	3.3	Setosa
5.4	3.9	Setosa
7.4	2.8	Virginica
6.1	2.8	Verscicolor
7.3	2.9	Virginica
6.0	2.7	Verscicolor
5.8	2.8	Virginica
6.3	2.3	Verscicolor
5.1	2.5	Verscicolor
6.3	2.5	Verscicolor
5.5	2.4	Verscicolor

Sepal Length	Sepal Width	Species
5.2	3.1	?

Step 1: Find Distance

$$\text{Distance (Sepal Length, Sepal Width)} = \sqrt{(x-a)^2 + (y-b)^2}$$

$$\text{Distance (Sepal Length, Sepal Width)} = \sqrt{(5.2-5.3)^2 + (3.1-3.7)^2}$$

$$\text{Distance (Sepal Length, Sepal Width)} = 0.608$$

Sepal Length	Sepal Width	Species	Distance
5.3	3.7	Setosa	0.608

KNN classifier example

Sepal Length	Sepal Width	Species	Distance	Rank
5.3	3.7	Setosa	0.608	3
5.1	3.8	Setosa	0.707	6
7.2	3.0	Virginica	2.002	13
5.4	3.4	Setosa	0.36	2
5.1	3.3	Setosa	0.22	1
5.4	3.9	Setosa	0.82	8
7.4	2.8	Virginica	2.22	15
6.1	2.8	Verscicolor	0.94	10
7.3	2.9	Virginica	2.1	14
6.0	2.7	Verscicolor	0.89	9
5.8	2.8	Virginica	0.67	5
6.3	2.3	Verscicolor	1.36	12
5.1	2.5	Verscicolor	0.60	4
6.3	2.5	Verscicolor	1.25	11
5.5	2.4	Verscicolor	0.75	7

Step 2: Find Rank

KNN classifier example

Sepal Length	Sepal Width	Species	Distance	Rank
5.3	3.7	Setosa	0.608	3
5.1	3.8	Setosa	0.707	6
7.2	3.0	Virginica	2.002	13
5.4	3.4	Setosa	0.36	2
5.1	3.3	Setosa	0.22	1
5.4	3.9	Setosa	0.82	8
7.4	2.8	Virginica	2.22	15
6.1	2.8	Verscicolor	0.94	10
7.3	2.9	Virginica	2.1	14
6.0	2.7	Verscicolor	0.89	9
5.8	2.8	Virginica	0.67	5
6.3	2.3	Verscicolor	1.36	12
5.1	2.5	Verscicolor	0.60	4
6.3	2.5	Verscicolor	1.25	11
5.5	2.4	Verscicolor	0.75	7

Step 3: Find the Nearest Neighbor

If k = 1 – Setosa

If k = 2 – Setosa

If k = 5 – Setosa

Approximation of discrete-valued target function

- Let us first consider learning ***discrete-valued target functions*** of the form

$$f : \Re^n \rightarrow V.$$

- Where, V is the finite set $\{v_1, \ldots v_s\}$
- The k- Nearest Neighbor algorithm for approximation a **discrete-valued target function** is given below:

---

(1) **Training algorithm:**

- For each training example $\langle x, f(x) \rangle$, add the example to the list *training_examples*

(2) **Classification algorithm:**

- Given a query instance x_q to be classified,
  - Let $x_1 \ldots x_k$ denote the k instances from *training_examples* that are nearest to x_q
  - Return

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{k} \delta(v, f(x_i))$$

where $\delta(a, b) = 1$ if $a = b$ and where $\delta(a, b) = 0$ otherwise.

Approximation of discrete-valued target function

Sl. No.	Height	Weight	Target	Distance	Nearest Points
1	150	50	Medium	8.06	
2	155	55	Medium	2.24	1
3	160	60	Large	6.71	3
4	161	59	Large	6.40	2
5	158	65	Large	11.05	
6	157	54	?		

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{k} \delta(v, f(x_i))$$

Approximation of discrete-valued target function

Sl. No.	Height	Weight	Target	Distance	Nearest Points
1	150	50	Medium	8.06	
2	155	55	Medium	2.24	1
3	160	60	Large	6.71	3
4	161	59	Large	6.40	2
5	158	65	Large	11.05	
6	157	54	?		

K=3

$\delta(a,b)=1$ $a==b$

$\delta(a,b)=0$ $a \neq b$

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{k} \delta(v, f(x_i))$$

1+0+0=1

0+1+1=2

Hamming Distance for Nominal and binary attributes

$$D_H = \sum_{i=1}^{k} |x_i - y_i|$$

$$x = y \Rightarrow D = 0$$

$$x \neq y \Rightarrow D = 1$$

The Hamming distance between:

- "karolin" and "kathrin" is 3.
- "karolin" and "kerstin" is 3.
- 1011101 and 1001001 is 2.
- 2173896 and 2233796 is 3.

Burger liking example using hamming distance

	Pepper	Ginger	Chilly	Liked
A	True	True	True	False
B	True	False	False	True
C	False	True	True	False
D	False	True	False	True
E	True	False	False	True

New Example - Q: pepper: false, ginger: true, chilly : true

Burger liking example using hamming distance

	Pepper	Ginger	Chilly	Liked	Distance
A	True	True	True	False	1 + 0 + 0 = 1
B	True	False	False	True	1 + 1 + 1 = 3
C	False	True	True	False	0 + 0 + 0 = 0
D	False	True	False	True	0 + 0 + 1 = 1
E	True	False	False	True	1 + 1 + 1 = 3

New Example - Q: pepper: false, ginger: true, chilly : true

Burger liking example using hamming distance

	Pepper	Ginger	Chilly	Liked	Distance	3NN
A	True	True	True	False	1 + 0 + 0 = 1	2
B	True	False	False	True	1 + 1 + 1 = 3	
C	False	True	True	False	0 + 0 + 0 = 0	1
D	False	True	False	True	0 + 0 + 1 = 1	2
E	True	False	False	True	1 + 1 + 1 = 3	

New Example - Q: pepper: false, ginger: true, chilly : true

Weighted similarity measure

age	income	student	Credit rating	Buys computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

- Given the training data, predict the class of the following new example using k-Nearest Neighbour for k=5:
- age<=30, income=medium, student=yes, credit- rating=fair.

Weighted similarity measure

age	income	student	Credit rating	Buys computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

- For similarity measure use a simple match of attribute values:

$$\sum_{i=1}^{4} w_i * \frac{\partial(a_i, b_i)}{4}$$

- *where* $\partial(a_i, b_i)$ is 1 if a_i equals b_i and 0 otherwise.
- a_i and b_i are either age, income, student or credit_rating.
- Weights are all 1 except for income it is 2.

Weighted similarity measure

age	income	student	Credit rating	Buys computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

age<=30, income=medium, student=yes, credit-rating=fair

RID	Class	Similarity to New
1	No	(1*1+2*0+1*0+1*1)/4 = 0.5
2	No	(1*1+2*0+1*0+1*0)/4 = 0.25
3	Yes	(1*0+2*0+1*0+1*1)/4 = 0.25
4	Yes	(1*0+2*1+1*0+1*1)/4 = 0.75
5	Yes	(0+0+1+1)/4 = 0.5
6	No	(0+0+1+0)/4 = 0.25
7	Yes	(0+0+1+0)/4 = 0.25
8	No	(1+2+0+1)/4 = 1
9	Yes	(1+0+1+1)/4 = 0.75
10	Yes	(0+2+1+1)/4 = 1
11	Yes	(1+2+1+0)/4 = 1
12	Yes	(0+2+0+0)/4 = 0.5
13	Yes	(0+0+1+1)/4 = 0.5
14	No	(0+2+0+0)/4 = 0.5

Weighted similarity measure

- Among the **five nearest neighbours** four are from class *Yes* and one from class *No*.
- Hence, the k-NN classifier predicts **buys_computer=yes** for the new example.

age<=30, income=medium, student=yes, credit-rating=fair

RID	Class	Similarity to New
1	No	(1*1+2*0+1*0+1*1)/4 = 0.5
2	No	(1*1+2*0+1*0+1*0)/4 = 0.25
3	Yes	(1*0+2*0+1*0+1*1)/4 = 0.25
4	Yes	(1*0+2*1+1*0+1*1)/4 = 0.75
5	Yes	(0+0+1+1)/4 = 0.5
6	No	(0+0+1+0)/4 = 0.25
7	Yes	(0+0+1+0)/4 = 0.25
8	No	(1+2+0+1)/4 = 1
9	Yes	(1+0+1+1)/4 = 0.75
10	Yes	(0+2+1+1)/4 = 1
11	Yes	(1+2+1+0)/4 = 1
12	Yes	(0+2+0+0)/4 = 0.5
13	Yes	(0+0+1+1)/4 = 0.5
14	No	(0+2+0+0)/4 = 0.5

Distance-weighted Nearest Neighbor Algorithm

- The refinement to the k-NEAREST NEIGHBOR Algorithm is to weight the contribution of each of the k neighbors according to their distance to the query point xq, giving greater weight to closer neighbors.
- For example, in the k-Nearest Neighbor algorithm, which approximates discrete-valued target functions, we might weight the vote of each neighbor according to the inverse square of its distance from xq.
- Distance-Weighted Nearest Neighbor Algorithm for approximation a discrete-valued target functions

Distance-weighted Nearest Neighbor Algorithm

Training algorithm:

- For each training example $\langle x, f(x) \rangle$, add the example to the list *training_examples*

Classification algorithm:

- Given a query instance x_q to be classified,
  - Let $x_1 \ldots x_k$ denote the k instances from *training_examples* that are nearest to x_q
  - Return

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{k} w_i \delta(v, f(x_i))$$

where

$$w_i \equiv \frac{1}{d(x_q, x_i)^2}$$

Distance-weighted Nearest Neighbor Algorithm

Sl. No.	Height	Weight	Target	Distance	1/distance^2	Nearest Points
1	150	50	Medium	8.06		
2	155	55	Medium	2.24	0.45	1
3	160	60	Large	6.71	0.15	3
4	161	59	Large	6.40	0.16	2
5	158	65	Large	11.05		
6	157	54	?			

Distance-weighted Nearest Neighbor Algorithm

Sl. No.	Height	Weight	Target	Distance	1/distance^2	Nearest Points
1	150	50	Medium	8.06		
2	155	55	Medium	2.24	0.45 ✓	1
3	160	60	Large	6.71	0.15 ✓	3
4	161	59	Large	6.40	0.16 ✓	2
5	158	65	Large	11.05		
6	157	54	? medium			

$0.45\,\delta(M,M) + 0.45\,\delta(M,L) + 0.16\,\delta(M,L)$

$0.45 * 1 + 0.45 * 0 + 0.16 * 0 = 0.45$

$0.45\,\delta(L,M) + 0.45\,\delta(L,L) + 0.16\,\delta(L,L) = 0 + 0.15 + 0.16 = 0.31$

Approximation of real-valued target function

- The K- Nearest Neighbor algorithm for approximation a **real-valued target function** is given below $f : \Re^n \rightarrow \Re$

Training algorithm:

- For each training example $\langle x, f(x) \rangle$, add the example to the list *training_examples*

Classification algorithm:

- Given a query instance x_q to be classified,
  - Let $x_1 \ldots x_k$ denote the k instances from *training_examples* that are nearest to x_q
  - Return

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} f(x_i)}{k}$$

Distance-weighted Nearest Neighbor Algorithm

- Distance-Weighted Nearest Neighbor Algorithm for approximation a Real-valued target functions

---

Training algorithm:

- For each training example $\langle x, f(x) \rangle$, add the example to the list *training_examples*

Classification algorithm:

- Given a query instance x_q to be classified,
  - Let $x_1 \ldots x_k$ denote the k instances from *training_examples* that are nearest to x_q
  - Return

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} w_i f(x_i)}{\sum_{i=1}^{k} w_i}$$

where

$$w_i \equiv \frac{1}{d(x_q, x_i)^2}$$

Distance-weighted Nearest Neighbor Algorithm

Sl. No.	Height	Weight	Target	Distance	1/distance^2	Nearest Points
1	150	50	1.5	8.06		
2	155	55	1.2	2.24	0.45 ✓	1
3	160	60	1.8	6.71	0.15 ✓	3
4	161	59	2.1	6.40	0.16 ✓	2
5	158	65	1.7	11.05		
6	157	54	?			

Distance-weighted Nearest Neighbor Algorithm

Sl. No.	Height	Weight	Target	Distance	1/distance^2	Nearest Points
1	150	50	1.5	8.06		
2	155	55	1.2	2.24	0.45 ✓	1
3	160	60	1.8	6.71	0.15 ✓	3
4	161	59	2.1	6.40	0.16 ✓	2
5	158	65	1.7	11.05		
6	157	54	? 1.51			

$\hat{f}(x_q) = (0.45 \times 1.2 + 0.15 \times 1.8 + 0.16 \times 2.1) / (0.45 + 0.15 + 0.16)$

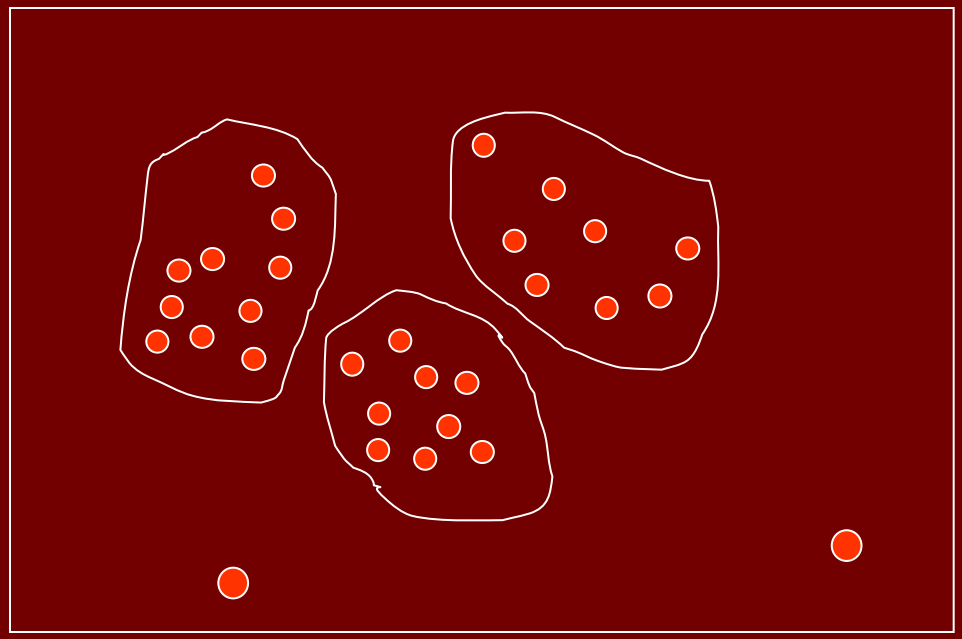
$= 1.146 / 0.76 = 1.51$

Noisy Data

- Noise is a random error or variance in a measured variable.
- Incorrect attribute values may be due to
  - faulty data collection tools
  - data entry problems
  - data transmission problems

Noisy Data (Clustering)

- Outliers may be detected by clustering, where similar values are organized into groups or “clusters”.
- Values which falls outside of the set of clusters may be considered outliers.



Data Integration

- Data integration:
  - Combines data from multiple sources into a coherent store as in data warehousing.
    - Sources may include
      - Databases
      - Data cubes
      - Flat files
- Problem(1): Schema integration
  - **Entity identification problem:** How we can sure that customer_id in one database, and cust_number in another refer to the same entity.
- Problem(2): Detecting and resolving data value conflicts
  - For the same entity, attribute values from different sources are different
  - possible reasons: different representations, different scales, e.g., Price in dollars vs. Price in Rupees.

Handling Redundant Data in Data Integration

- Redundant data occur often when integration of multiple databases
  - The same attribute may have different names in different databases e.g., customer_id and customer_number.
  - One attribute may be a “derived” attribute in another table,
- **Technique:**
  - Redundant data may be able to be detected by correlation analysis.

Handling Redundant Data in Data Integration

- Correlation between two attributes can be checked by:-

$$r_{A,B} = \frac{\Sigma(A - \bar{A})(B - \bar{B})}{(n-1)\sigma_A \sigma_B}$$

1. Resulting value > 0, then A and B are positively correlated. If A increase B will also increase. A or B can be removed
2. Resulting value = 0, then A and B are independent.
3. Resulting value < 0, then A and B are negatively correlated. If the value of A increases, the value B will decrease.

Data Transformation: Data Normalization

- The goal of normalization is to transform features to be on a similar scale. All integer attributes are normalized, so that their values fall within a small specified, such as 0 to 1.0 or -1.0 to 1.0.

- Normalization is particularly useful for classification algorithms or distance measurements such as nearest-neighbor classification and clustering.

person_name	Salary	Year_of_ experience	Expected Position Level
Aman	100000	10	2
Abhinav	78000	7	4
Ashutosh	32000	5	8
Dishi	55000	6	7
Abhishek	92000	8	3
Avantika	120000	15	1
Ayushi	65750	7	5

The attributes salary and year_of_experience are on different scale and hence attribute salary can take high priority over attribute year_of_experience in the model.

Data Normalization

- min-max normalization

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$$

  - Example:- Suppose that the minimum and maximum values for the attribute income are 12,000 and 98,000. By min-max normalization, a value of 73,600 for income is transformed to
    - ((73,000-12,000)/(98,000-12,000)) (1.0-0) + 0 = 0.716

Data Normalization

- z-score normalization
  - This method of normalization is useful when the actual minimum and maximum of any attribute are unknown.
  - Or when outliers which dominate the min-max normalization.

$$v' = \frac{v - mean_A}{stand\_dev_A}$$

- Decimal scaling normalization

$$v' = \frac{v}{10^j}$$

Where j is the smallest integer such that $\text{Max}(|v'|)<1$