Supervised Classification

Dr Muhammad Atif Tahir

Professor

School of Computer Science

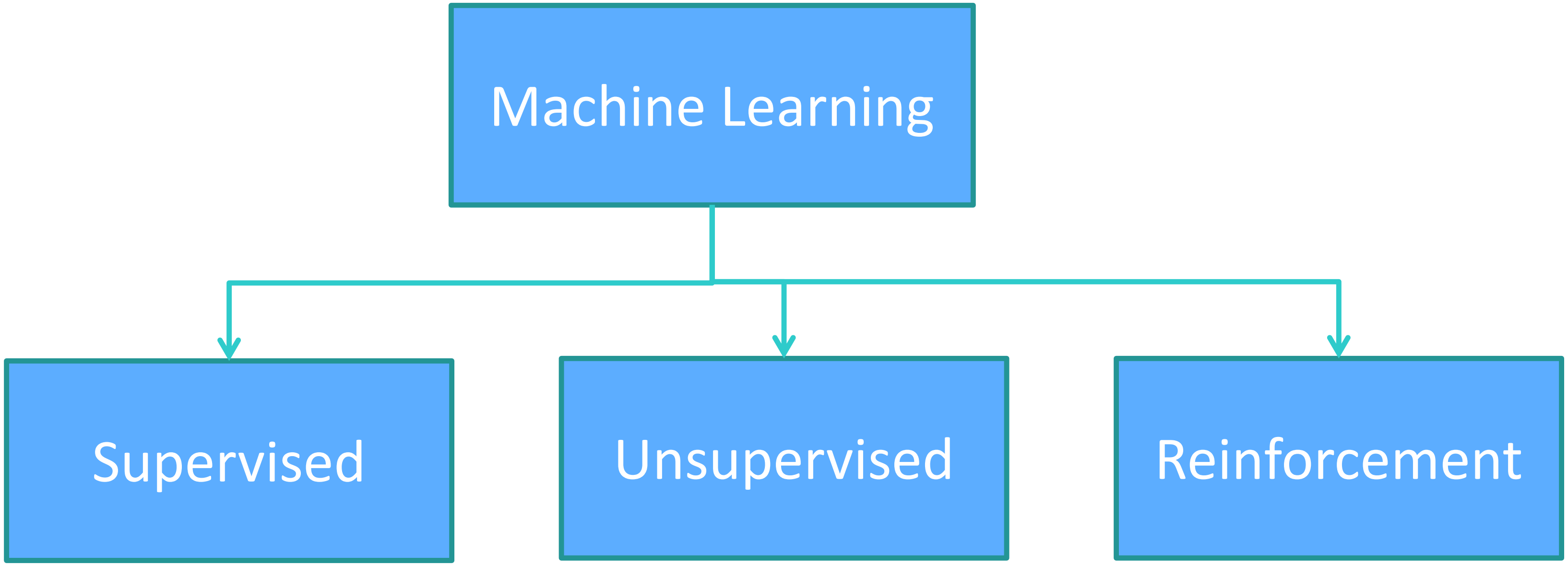
National University of Computing & Emerging Sciences

Karachi Campus

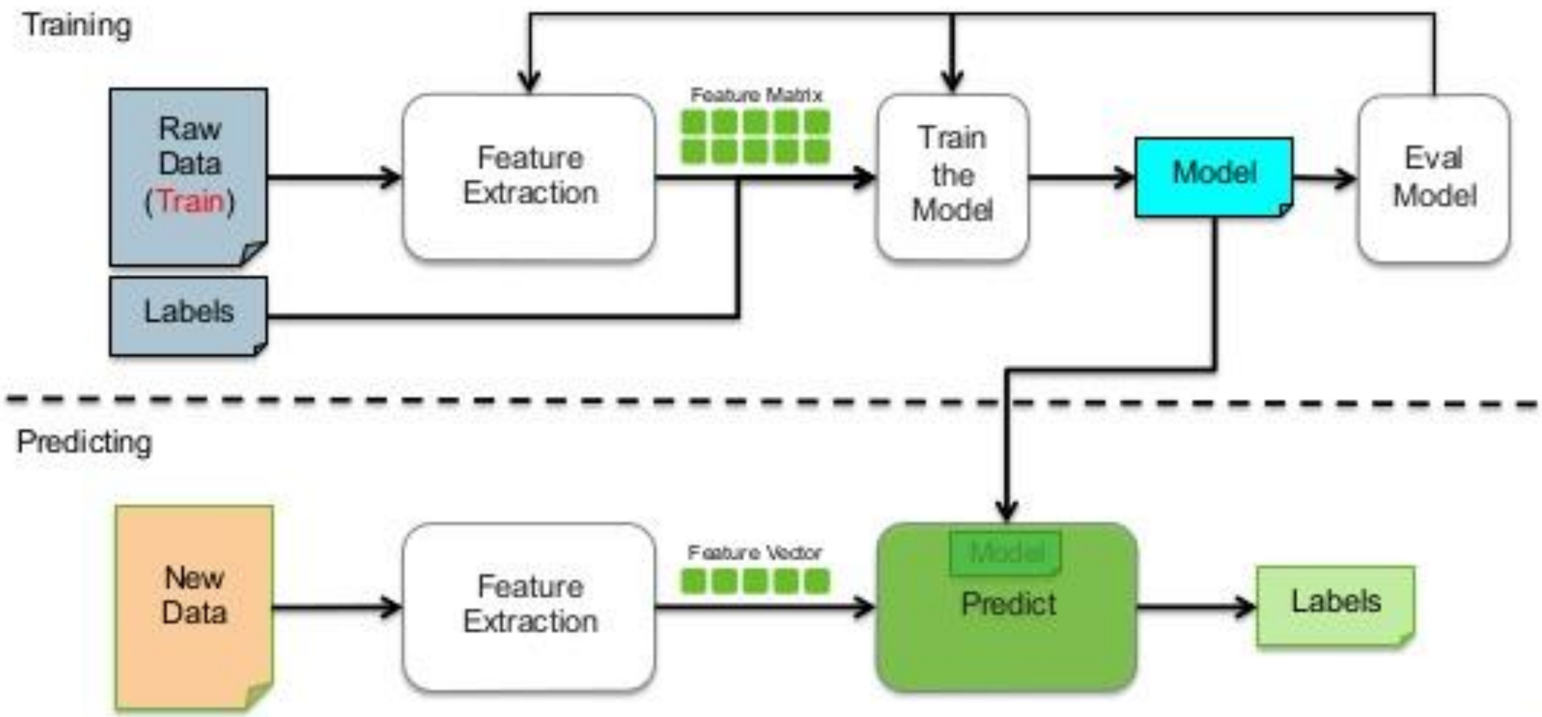
Contents

- kNN classifier
- Confusion Matrix
- Conclusions

Types of Machine Learning



Supervised Learning Workflow



Hortonworks

Instance Based Classifiers

- First Example of Supervised Classification
- Examples:
  - Rote-learner
    - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly
  - Nearest neighbor
    - Uses k “closest” points (nearest neighbors) for performing classification

Instance-Based Classifiers

Set of Stored Cases

Atr1	……...	AtrN	Class
			A
			B
			B
			C
			A
			C
			B

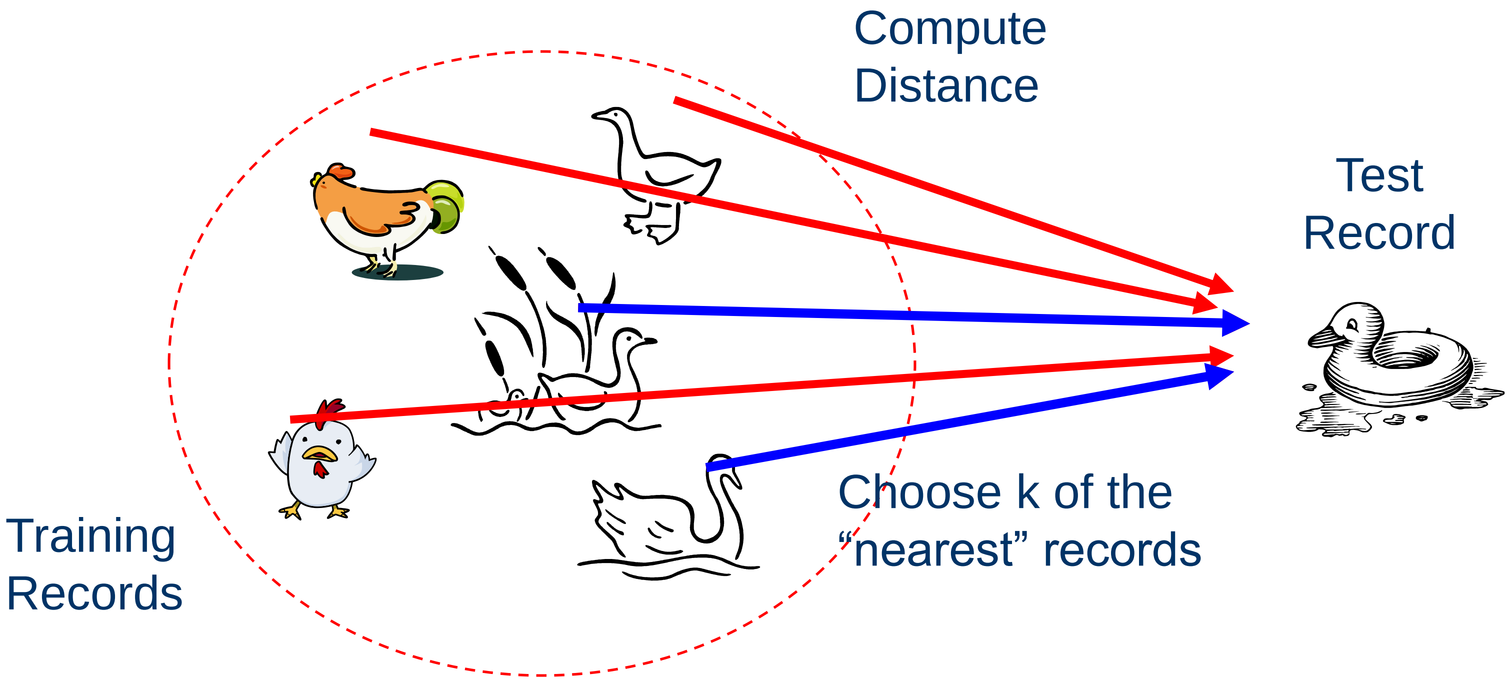
- Store the training records
- Use training records to predict the class label of unseen cases

Unseen Case

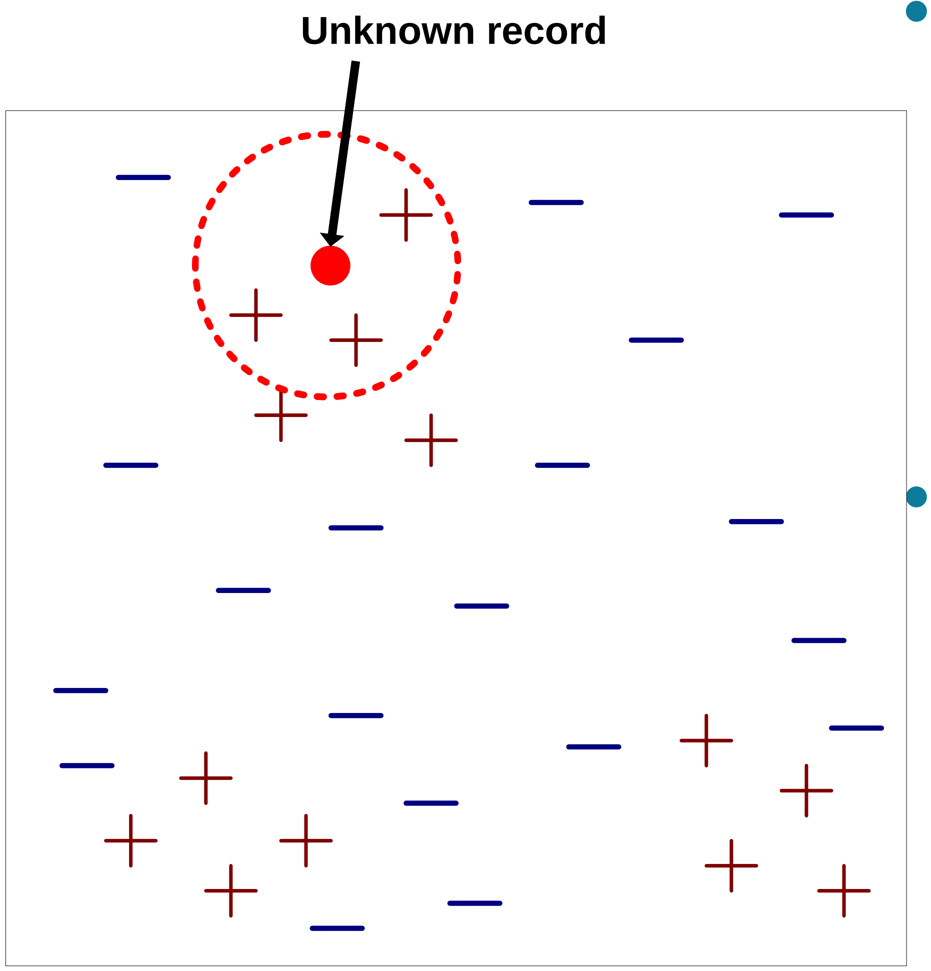
Atr1	……...	AtrN

Nearest Neighbor Classifiers

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



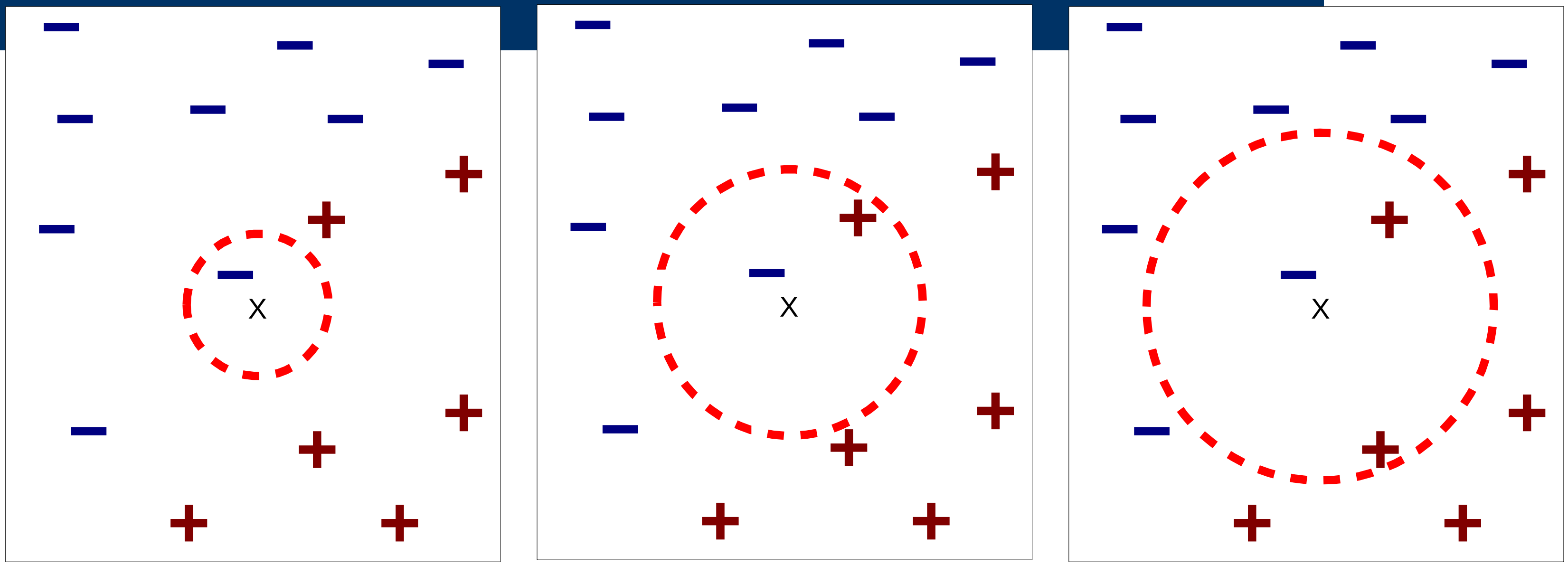
Nearest-Neighbor Classifiers



- Requires three things
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of *k*, the number of nearest neighbors to retrieve

- To classify an unknown record:
  - Compute distance to other training records
  - Identify *k* nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

Definition of Nearest Neighbor



(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

K-nearest neighbors of a record x are data points that have the k smallest distance to x

Nearest Neighbor Classification

- Compute distance between two points:
  - Euclidean distance

$$d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

$$d(p,q) = \sum_i abs(p_i - q_i)$$

- Determine the class from nearest neighbor list
  - take the majority vote of class labels among the k-nearest neighbors
  - Weigh the vote according to distance
    - weight factor, $w = 1/d^2$

Example (NN Classifier)

F1	F2	Class
1	5	0
0	8	0
0	6	1
1	2	1

Training Data

1	*3*	*?*
1	*4*	*?*
0	*3*	*?*
0	*4*	*?*

Test Data

Example (NN Classifier)

Step 1: Computer Distance from Test Sample 1 to Training Data

Step 2:

Distance from Test Sample 1 to All Training Samples		Class
1	$\lvert 1-1\rvert+\lvert 3-5\rvert = 0 + 2 = 2$	0
2	$\lvert 1-0\rvert+\lvert 3-8\rvert = 1 + 5 = 6$	0
3	$\lvert 1-0\rvert+\lvert 3-6\rvert = 1 + 3 = 4$	1
4	$\lvert 1-1\rvert+\lvert 3-2\rvert = 0 + 1 = 1$	1

Step 3: Assign the Test Sample to Class with minimum Distance, Here is Class 1. So Test Sample 1 belongs to Class 1

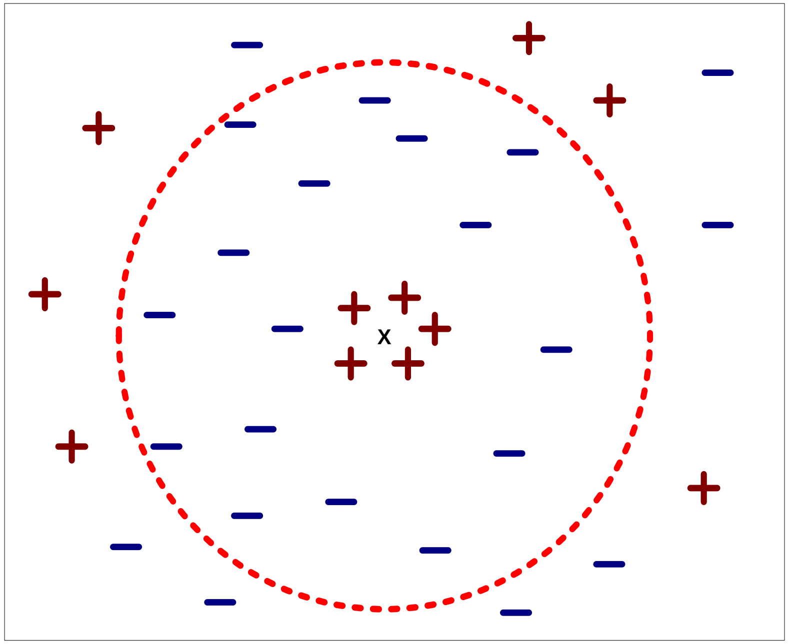
Example (NN Classifier)

Exercise: Calculate for other 3 Test Samples

ID	Actual	Predicted
1	0	1
2	0	0
3	1	1
4	1	0 or 1

Nearest Neighbor Classification…

- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes



Nearest Neighbor Classification…

- Scaling issues
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
  - Example:
    - height of a person may vary from 1.5m to 1.8m
    - weight of a person may vary from 90lb to 300lb
    - income of a person may vary from $10K to $1M

Example (NN Classifier)

Normalize Data from 0 to 1

F1	F2	Class
1	0.5	0
0	1	0
0	0.667	1
1	0	1

Training Data

1	*0.167*	*?*
1	*0.334*	*?*
0	*0.167*	*?*
0	*0.334*	*?*

Test Data

Example (NN Classifier)

After Normalization

ID	Actual	Predicted
1	0	1
2	0	0
3	1	1
4	1	1

Confusion Matrix

- In the field of machine learning, a **confusion matrix** is a specific table layout that allows visualization of the performance of an algorithm

	Predicted Negative	Predicted Positive
Actual Negative	True Negative	False Positive
Actual Positive	False Negative	True Positive

Confusion Matrix

- *TN* is the number of correct predictions that an instance is negative
- *FP* is the number of incorrect predictions that an instance is positive
- *FN* is the number of incorrect predictions that an instance is negative
- *TP* is the number of correct predictions that an instance is positive

Confusion Matrix

- Confusion Matrix from the example of Lecture 2 (without Normalization)

ID	Actual	Predicted
1	1	1
2	0	0
3	1	1
4	1	0

	Negative	Positive
Negative	1	0
Positive	1	2

Confusion Matrix

- Several standard terms have been defined for the 2 class matrix
- The *accuracy* (*AC*) is the proportion of the total number of predictions that were correct

$$Accuracy = \frac{TN + TP}{TN + FN + TP + FP}$$

- Accuracy = 3 / 4 = 75%

Confusion Matrix

- The *recall* or *true positive rate* (*TPR*) is the proportion of positive cases that were correctly identified

$$TPR = \frac{TP}{TP + FN}$$

- The *false positive rate* (*FPR*) is the proportion of negatives cases that were incorrectly classified as positive

$$FPR = \frac{FP}{FP + TN}$$

- TPR or recall = 2 / 3 = 66.7%
- FPR = 0 / 1 = 0 %

Confusion Matrix

- The *true negative rate* (*TNR*) is defined as the proportion of negatives cases that were classified correctly,

$$TNR = \frac{TN}{FP + TN}$$

- The *false negative rate* (*FNR*) is the proportion of positives cases that were incorrectly classified as negative

$$FNR = \frac{FN}{FN + TP}$$

- TNR = 1 / 1 = 100%
- FNR = 1 / 3 = 33.3%

Confusion Matrix

- *precision* (*P*) is the proportion of the predicted positive cases that were correct,

$$precision = \frac{tp}{tp + fp}$$

- precision = 2/2 = 100%
- F measure is harmonic mean of precision and recall

$$F_1 = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

- F1 = (2 * 1 * 0.667)/(1+0.667) = 0.8

Exercise

		Actual	
		Negative	Positive
Predicted	Negative	9760	40
	Positive	140	60

References

- Introduction to Data Mining by Tan, Steinbach, Kumar (Lecture Slides)
- http://robotics.stanford.edu/~ronnyk/glossary.html
- http://www.cs.tufts.edu/comp/135/Handouts/introduction-lecture-12-handout.pdf

Questions!