K – Nearest Neighbors and Classifiers Evaluation

Dr. Wedad Hussein
wedad.hussein@cis.asu.edu.eg

Dr. Mahmoud Mounir
mahmoud.mounir@cis.asu.edu.eg

Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- K-Nearest Neighbors
- Model Evaluation

K-Nearest Neighbors (KNN)

- K-Nearest Neighbors (KNN) is a supervised learning algorithm where the result of new instance query is classified based on majority of K-nearest neighbor category.
- The purpose of this algorithm is to classify a new object based on attributes and training samples.
- KNN used neighborhood classification as the prediction value of the new query instance.

K-Nearest Neighbors (KNN)

- Given data instance

  X1 = 3 and X2 = 7

Determine the suitable class of this instance using KNN algorithm.

X1	X2	Class
7	7	Bad
7	4	Bad
3	4	Good
1	4	Good

K-Nearest Neighbors (KNN)

Here is step by step on how to compute K-nearest neighbors KNN algorithm:

- Determine parameter K = number of nearest neighbors.
- Calculate the distance between the query-instance and all the training samples.
- Sort the distance and determine nearest neighbors based on the K-th minimum distance.
- Gather the category of the nearest neighbors.
- Use simple majority of the category of nearest neighbors as the prediction value of the query instance

K-Nearest Neighbors (KNN)

Here is step by step on how to compute K-nearest neighbors KNN algorithm:

- Determine parameter K = number of nearest neighbors.
  - Suppose k = 3
- Calculate the distance between the query-instance and all the training samples.
  - We will use the Euclidean distance

X1	X2	Distance
7	7	$\sqrt{(7-3)^2+(7-7)^2}$ = 4
7	4	$\sqrt{(7-3)^2+(4-7)^2}$ = 5
3	4	$\sqrt{(3-3)^2+(4-7)^2}$ = 3
1	4	$\sqrt{(1-3)^2+(4-7)^2}$ = $\sqrt{13}$ = 3.6

K-Nearest Neighbors (KNN)

- Sort the distance and determine nearest neighbors based on the K-th minimum distance.

X1	X2	Distance	Ranked distance	Is it included in the 3-Nerest Neighbors
7	7	4	3	Yes
7	4	5	4	No
3	4	3	1	Yes
1	4	3.6	2	Yes

K-Nearest Neighbors (KNN)

- Gather the category of the nearest neighbors.

X1	X2	Distance	Ranked distance	Is it included in the 3-Nerest Neighbors	Class
7	7	4	3	Yes	Bad
7	4	5	4	No	
3	4	3	1	Yes	Good
1	4	3.6	2	Yes	Good

- Use simple majority of the category of nearest neighbors as the prediction value of the query instance

K-Nearest Neighbors (KNN)

X1	X2	Distance	Ranked distance	Is it included in the 3-Nerest Neighbors	Class
7	7	4	3	Yes	Bad
7	4	5	4	No	
3	4	3	1	Yes	Good
1	4	3.6	2	Yes	Good

- Use simple majority of the category of nearest neighbors as the prediction value of the query instance
  - We have 2 good and 1 bad,
  - We conclude that data instance X1 = 3 and X2 = 7 is included in <u>Good category</u>.

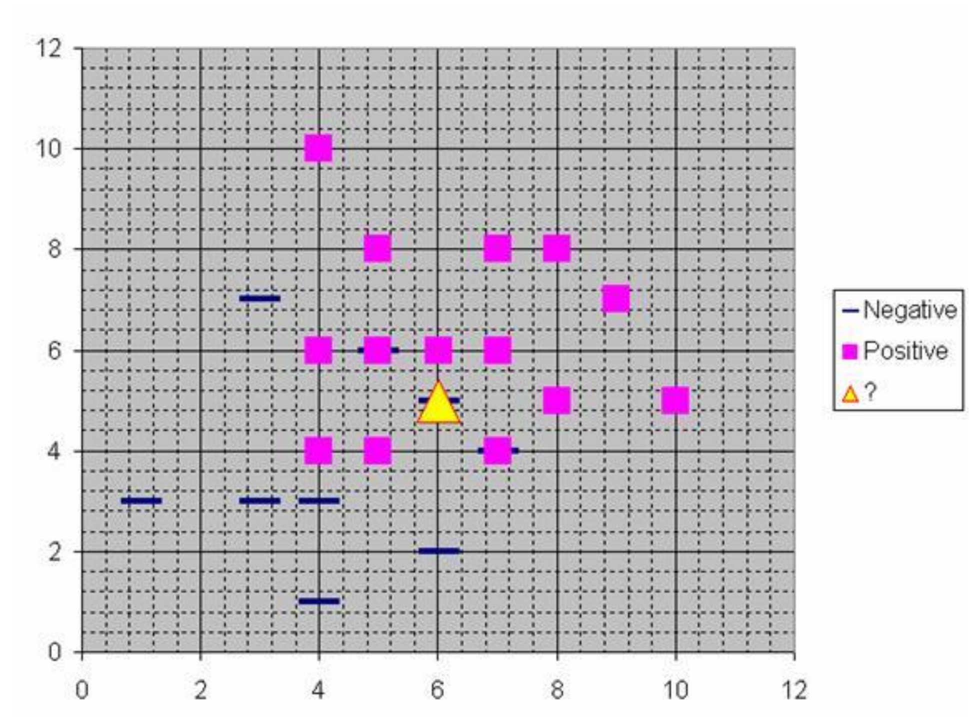
K-Nearest Neighbors (KNN)

Example

training data

X1	X2	Y
4	3	+
1	3	+
3	3	+
3	7	+
7	4	+
4	1	+
6	5	+
5	6	+
3	7	+
6	2	+
4	6	-
4	4	-
5	8	-
7	8	-
5	6	-
10	5	-
7	6	-
4	10	-
9	7	-
5	4	-
8	5	-
6	6	-
7	4	-
8	8	-
6	5	?

prediction



Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- K-Nearest Neighbors
- Model Evaluation

Model Evaluation and Selection

- Evaluation metrics: How can we measure accuracy? Other metrics to consider?
- Use **validation test set** of class-labeled tuples instead of training set when assessing accuracy
- Methods for estimating a classifier's accuracy:
  - Holdout method, random subsampling
  - Cross-validation
  - Bootstrap
- Comparing classifiers:
  - Confidence intervals
  - Cost-benefit analysis and ROC Curves

Classifier Evaluation Metrics: Confusion Matrix

Confusion Matrix:

Actual class\Predicted class	C_1	$\neg C_1$
C_1	**True Positives (TP)**	**False Negatives (FN)**
$\neg C_1$	**False Positives (FP)**	**True Negatives (TN)**

Example of Confusion Matrix:

Actual class\Predicted class	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	**6954**	**46**	7000
buy_computer = no	**412**	**2588**	3000
Total	7366	2634	10000

- Given *m* classes, an entry, $CM_{i,j}$ in a **confusion matrix** indicates # of tuples in class *i* that were labeled by the classifier as class *j*
- May have extra rows/columns to provide totals

Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

A\P	C	¬C	
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
	P'	**N'**	**All**

- **Classifier Accuracy,** or **recognition rate**: percentage of test set tuples that are correctly classified

$$Accuracy = \frac{TP+TN}{ALL}$$

- **Error rate:**

Error rate = 1 – accuracy, or

$$Error\ rate = \frac{FP+FN}{ALL}$$

Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

A\P	C	¬C	
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
	P'	**N'**	**All**

- **Sensitivity (Recall)**:
  - What % of positive tuples did the classifier label as positive?
  - True Positive recognition rate

$$\textit{Sensitivity} = \frac{TP}{P} = \frac{TP}{TP+FN}$$

- **Specificity**:
  - What % of negative tuples did the classifier label as negative?
  - True Negative recognition rate

$$\textit{Specificity} = \frac{TN}{N} = \frac{TN}{FP+TN}$$

Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

A\P	C	¬C	
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
	P'	**N'**	**All**

- **Precision**:
  - what % of tuples that the classifier labeled as positive are actually positive

$$Precision = \frac{TP}{P'} = \frac{TP}{TP+FP}$$

Classifier Evaluation Metrics: Confusion Matrix

Confusion Matrix:

Actual class\Predicted class	Positive **(H_o is True)**	Negative **(H_o is False)**
Positive **(Do not reject H_o)**	**(TP)**	**(FN) Type (II) Error**
Negative **(Reject H_o)**	**(FP) Type (I) Error**	**(TN)**

Example of Confusion Matrix:

Actual class\Predicted class	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	**6954**	**46**	7000
buy_computer = no	**412**	**2588**	3000
Total	7366	2634	10000

Classifier Evaluation Metrics: Example (1)

Actual Class\Predicted class	cancer = yes	cancer = no	Total	Recognition(%)
cancer = yes	**90**	**210**	300	30.00 (*sensitivity*
cancer = no	**140**	**9560**	9700	98.56 (*specificity)*
Total	230	9770	10000	96.50 (*accuracy*)

- ***Precision = TP/TP+FP*** *= 90/230 = 39.13%*
- ***Sensitivity (Recall) = TP/P*** *= 90/300 = 30.00%*
- ***Specificity = TN/N*** *= 9560/9700 = 98.56%*
- ***Accuracy = TP+TN/ALL*** *= 90+9560/10000 = 96.50%*

A\P	C	¬C	
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
	P'	**N'**	**All**

Classifier Evaluation Metrics: Example (2)

		Predicted		
		A	B	C
Actual	A	95	3	2
	B	5	90	5
	C	15	0	85

a) The recognition rate.

b) The sensitivity of Class C.

c) The error rate.

a) Recognition rate = *95+90+85/300 = 90%*

b) Sensitivity (Recall) = TP/P *= 85/100 = 85%*

c) Error rate = 1- recognition rate = 1-0.9 = 0.1 = 10%