Multivariate Analysis

Supervised Learning

Ira A. Fulton Schools of Engineering
Arizona State University

Objective

Objective

Define supervised learning and describe supervised learning methods

A Twitter Example

ID	Celebrity	Verified Account	# Followers	Influential?
1	Yes	No	1.25M	No
2	No	Yes	1M	No
3	No	Yes	600K	No
4	Yes	Unknown	2.2M	No
5	No	No	850K	Yes
6	No	Yes	750K	No
7	No	No	900K	Yes
8	No	No	700K	No
9	Yes	Yes	1.2M	No
10	No	Unknown	950K	Yes

Supervised Learning: The Process

- We are given a set of labeled records/instances
  - In the format (***X***,*y*)
  - ***X*** is a vector of features
  - *y* is the class attribute (commonly a scalar)
- [Training] supervised learning task is to build a model that maps **X** to y
  - Find a mapping, *m*, such that *m(**X**) = y*
- [Testing] Given an unlabeled instance (***X'***,?), we compute *m(**X'**)*
  - E.g, spam/non-spam prediction

Supervised Learning Algorithms

Classification

- Decision Tree Learning
- *k*-Nearest Neighbor Classifier

Regression

- Linear Regression

Decision Tree Learning

Decision Trees

- A decision tree is learned from the dataset
  - Training data with known classes
- Learned tree is later applied to predict class attribute value of new data
  - Test data with unknown classes
  - Only feature values are known

ID	Celebrity	Verified Account	# Followers	Influential?
1	Yes	No	1.25M	No
2	No	Yes	1M	No
3	No	Yes	600K	No
4	Yes	Unknown	2.2M	No
5	No	No	850K	Yes
6	No	Yes	750K	No
7	No	No	900K	Yes
8	No	No	700K	No
9	Yes	Yes	1.2M	No
10	No	Unknown	950K	Yes

Decision Trees Example

Multiple decision trees can be learned from the same dataset.

ID	Celebrity	Verified Account	# Followers	Influential?
1	Yes	No	1.25M	No
2	No	Yes	1M	No
3	No	Yes	600K	No
4	Yes	Unknown	2.2M	No
5	No	No	850K	Yes
6	No	Yes	750K	No
7	No	No	900K	Yes
8	No	No	700K	No
9	Yes	Yes	1.2M	No
10	No	Unknown	950K	Yes

(a) Learned Decision Tree 1

(b) Learned Decision Tree 2

Decision Tree Construction

- Decision Trees are constructed recursively
- After selecting a feature for each node, different branches are created
- Training set is then partitioned into subsets based on feature values
- When selecting features, we prefer features that partition the set of instances into subsets that are more *pure*
- A *pure* subset has instances that all have the same class attribute value

Stopping Criteria for Decision Tree Induction

When reaching pure (or highly pure) subsets under a branch

- Decision tree construction process no longer partitions the subset
- Creates a leaf node under the branch
- Assigns class attribute value (or the majority class attribute value) for subset instances as the leaf's predicted class attribute value

Measuring Purity

- To measure purity we can use/minimize entropy

- Over a subset of training instances, *T*, with a binary class attribute (values in {+,-}), the entropy of *T* is defined as:

$$entropy(T) = -p_+ \log p_+ - p_- \log p_-$$

Entropy Example

Assume there is a subset T that has 10 instances:

- Seven instances have a positive class attribute value
- Three have a negative class attribute value
- Denote *T* as [7+,3-]
- The entropy for subset *T* is:

$$entropy(T) = -\frac{7}{10}\log\frac{7}{10} - \frac{3}{10}\log\frac{3}{10} = 0.881$$

Entropy Values vs. Purity

$$entropy(T) = -p_+ \log p_+ - p_- \log p_-$$

- In a pure subset, all instances have the same class attribute value (**entropy is 0**)
- When is it **1**?
- If the subset contains an unequal number of positive and negative instances
  - The entropy is between 0 and 1

Nearest Neighbor Classifier

An Intuitive Illustration of *k*-NN

No.	Outlook (O)	Temperature (T)	Humidity (H)	Play Golf (PG)
1	sunny	hot	high	N
2	sunny	mild	high	N
3	overcast	hot	high	Y
4	rain	mild	high	Y
5	sunny	cool	normal	Y
6	rain	cool	normal	N
7	overcast	cool	normal	Y
8	sunny	mild	high	?

k-NN: Example

k-Nearest Neighbors

- *k*-nearest neighbors or *k*-NN
- Uses *k* nearest instances, called **neighbors**, to perform classification
- Instance being classified is assigned label (class attribute value) that majority of its *k* neighbors are assigned
- When *k* = 1, the closest neighbor's label is used as predicted label for instance being classified
- To determine the neighbors of an instance, we need to measure its distance to all other instances based on some distance metric

k-NN: Example

No.	Outlook (O)	Temperature (T)	Humidity (H)	Play Golf (PG)
1	sunny	hot	high	N
2	sunny	mild	high	N
3	overcast	hot	high	Y
4	rain	mild	high	Y
5	sunny	cool	normal	Y
6	rain	cool	normal	N
7	overcast	cool	normal	Y
8	sunny	mild	high	?

Similarity between row 8 and other data instances;
(Similarity = 1 if attributes have the same value, otherwise similarity = 0)

Data instance	Outlook	Temperature	Humidity	Similarity	Label	K	Prediction
2	1	1	1	3	N	1	**N**
1	1	0	1	2	N	2	**N**
4	0	1	1	2	Y	3	**N**
3	0	0	1	1	Y	4	**?**
5	1	0	0	1	Y	5	**Y**
6	0	0	0	0	N	6	**?**
7	0	0	0	0	Y	7	**Y**

k-NN: Algorithm

Algorithm 5.1 *k*-Nearest Neighbor Classifier

Require: Instance i, A Dataset of Real-Value Attributes, k (number of neighbors), distance measure d

1: **return** Class label for instance i
2: Compute k nearest neighbors of instance i based on distance mea- sure d.
3: l = the majority class label among neighbors of instance i. If more than one majority label, select one randomly.
4: Classify instance i as class l

k-NN: A Lazy Learning Algorithm

- Does *k*-NN learn?
- How fast is *k*-NN?

Regression

- $y = c$

- $y = \beta_0 + \beta_1 x$

- $\varepsilon = y - (\beta_0 + \beta_1 x)$

Interpolation

Linear Interpolation

Regression

In regression,

- Class values are real numbers as class values
  - In classification, class values are categorical

$$y \approx f(X)$$

Class attribute
(dependent variable)
$y \in R$

Features
(regressors)
$X = (x_1, x_2, \ldots, x_m)$

Goal: find the relation between y and vector $X = (x_1, x_2, \ldots, x_m)$

Linear Regression

$$Y = XW + \epsilon$$

$$\epsilon^2 = ||\epsilon^2|| = ||Y - XW||^2$$

Linear Regression

- we assume the relation between the class attribute *Y* and feature set ***X*** is linear
- *W* represents the vector of regression coefficients

Regression

- can be solved by estimating *W* and epsilon using the provided dataset and the labels *Y*
- “Least squares” is a popular method to solve regression