Data Analytics

Yong Zheng

Illinois Institute of Technology

Chicago, IL, 60616, USA

School of Applied Technology

ILLINOIS INSTITUTE OF TECHNOLOGY

Classification Algorithms

- KNN Classifier
  - Lazy classifier
  - Have to specify the value of K
  - Sensitive to initial clusters, distance measures, K
  - Cannot handle categorical data, have to transform data
- Naïve Bayes Classifier
- Logistic Regression
- Tree-Based Learning

Classification Algorithms

- KNN Classifier
- Naïve Bayes Classifier
  - Requirement: conditionally independent
  - Cannot handle numeric, have to transform data
  - May have imbalance issues in labels (general issue)
  - Laplace smoothing
- Logistic Regression
- Tree-Based Learning

Classification Algorithms

- KNN Classifier
- Naïve Bayes Classifier
- Logistic Regression
  - Utilize regression models for classifications
  - The y variable is log (odd)
  - Feature selections can also be applied
  - Make decisions by using odd or P (Y = 1)
- Tree-Based Learning

Classification Algorithms

- KNN Classifier
- Naïve Bayes Classifier
- Logistic Regression
- Tree-Based Learning
  - More complicated but much more effective sometimes
  - Tree-based learning: a machine learning method
  - Require feature selection
  - Require to handle overfitting problems

Classification Algorithms

- Ensemble Methods
- Multi-Label Classification

Ensembles of Classifiers

- **Basic idea** is to learn a set of classifiers (experts) and to allow them to vote.
- **Advantage:** improvement in predictive accuracy.
- **Disadvantage:** it is difficult to understand an ensemble of classifiers.

Ensemble Methods

- Bagging
- AdaBoosting
- Random Forest
- …. And more

Bagging

- Process in bagging:
  - Sample several training sets of size *n* (instead of just having one training set of size *n*)
  - Build a classifier for each training set
  - Combine the classifier's predictions by voting or averaging

Bagging classifiers

Classifier generation

Let *n* be the size of the training set.
For each of *t* iterations:
  Sample *n* instances with replacement from the training set.
  Apply the learning algorithm to the sample.
  Store the resulting classifier.

classification

For each of the *t* classifiers:
  Predict class of instance using classifier.
Return class that was predicted most often.

Voting and Averaging

- Voting is used for classifications, and averaging is used for regressions
- Voting: Hard and Soft voting

Hard voting

Predictions:

Classifier 1 predicts class A

Classifier 2 predicts class B

Classifier 3 predicts class B

2/3 classifiers predict class B, so **class B is the ensemble decision**.

Soft voting

Predictions (identical to the earlier example, but now in terms of probabilities. Shown only for class A here because the problem is binary):

Classifier 1 predicts class A with probability 99%

Classifier 2 predicts class A with probability 49%

Classifier 3 predicts class A with probability 49%

The average probability of belonging to class A across the classifiers is `(99 + 49 + 49) / 3 = 65.67%`. Therefore, **class A is the ensemble decision**.

Why does bagging work?

- Bagging reduces variance by voting/ averaging, thus reducing the overall expected error
  - In the case of classification there are pathological situations where the overall error might increase
  - Usually, the more classifiers the better

Boosting

- Also uses voting/averaging but models are weighted according to their performance
- Iterative procedure: new models are influenced by performance of previously built ones
  - New model is encouraged to become expert for instances classified incorrectly by earlier models
  - Assign more weights to the misclassified instances to improve the classification iteratively
- There are several variants of this algorithm

AdaBoost.M1

classifier generation

```
Assign equal weight to each training instance.
For each of t iterations:
  Learn a classifier from weighted dataset.
  Compute error e of classifier on weighted dataset.
  If e equal to zero, or e greater or equal to 0.5:
    Terminate classifier generation.
  For each instance in dataset:
    If instance classified correctly by classifier:
       Multiply weight of instance by e / (1 - e).
  Normalize weight of all instances.
```

classification

```
Assign weight of zero to all classes.
For each of the t classifiers:
  Add -log(e / (1 - e)) to weight of class predicted
  by the classifier.
Return class with highest weight.
```

Random Forest

Classifier generation

Let *n* be the size of the training set.

For each of *t* iterations:

(1) Sample *n* instances with replacement from the training set.

(2) Learn a decision tree s.t. the variable for any new node is the best variable among *m* randomly selected variables.

(3) Store the resulting decision tree.

Classification

For each of the *t* decision trees:

Predict class of instance.

Return class that was predicted most often.

Classification Algorithms

- Ensemble Methods
- Multi-Label Classification

Advanced Topic: Multi-Label Classification

- Applications

Data type	Application	Resource	Labels Description (Examples)
text	categorization	news article	Reuters topics (agriculture, fishing)
		web page	Yahoo! directory (health, science)
		patent	WIPO (paper-making, fibreboard)
		email	R&D activities (delegation)
		legal document	Eurovoc (software, copyright)
		medical report	MeSH (disorders, therapies)
		radiology report	ICD-9-CM (diseases, injuries)
		research article	Heart conditions (myocarditis)
		research article	ACM classification (algorithms)
		bookmark	Bibsonomy tags (sports, science)
		reference	Bibsonomy tags (ai, kdd)
		adjectives	semantics (object-related)
image	semantic annotation	pictures	concepts (trees, sunset)
video	semantic annotation	news clip	concepts (crowd, desert)
audio	noise detection	sound clip	type (speech, noise)
	emotion detection	music clip	emotions (relaxing-calm)
structured	functional genomics	gene	functions (energy, metabolism)
	proteomics	protein	enzyme classes (ligases)
	directed marketing	person	product categories

Multi-Label Classification: Example

- Movies and Emotions

Title	Actors	Director	Emo_before	Emo_during	Emo_after
Spider Man	XX	XXX	Sad	Exciting	Happy
Superman	Xxx	Xx	Exciting	Normal	Disappointed
…	…	…	…	…	…
…	…	…	…	…	…
New Movie	Xxxx	Xxx	?	?	?

Multi-Label Classification: Example

- Twitter and Hashtags
  - You may tweet some texts and use hashtags
  - For a single tweet, you may use more than one hashtag
  - Given a set of knowledge → tweet with hashtags, we want to build multi-label classification models
  - Given a new tweet, we predict or suggest the hashtags they can use

Multi-Label Classification

- Solutions
  - **Transformation Based Methods**
    Transform the task to binary/multi-class classifications
  - **Adaptation Based Methods**
    Develop new algorithms to solve the problem

Multi-Label Classification

- Transformation Based Methods
  - Binary Relevance
  - Classifier Chains
  - Label Powerset

Multi-Label Classification

- Binary Relevance

Example	Attributes	Label set
1	$\mathbf{x_1}$	$\{\lambda_1, \lambda_4\}$
2	$\mathbf{x_2}$	$\{\lambda_3, \lambda_4\}$
3	$\mathbf{x_3}$	$\{\lambda_1\}$
4	$\mathbf{x_4}$	$\{\lambda_2, \lambda_3, \lambda_4\}$

There are 4 labels, then we simply build 4 binary classifiers

Ex.	Label
1	λ_1
2	$\neg\lambda_1$
3	λ_1
4	$\neg\lambda_1$

(a)

Ex.	Label
1	$\neg\lambda_2$
2	$\neg\lambda_2$
3	$\neg\lambda_2$
4	λ_2

(b)

Ex.	Label
1	$\neg\lambda_3$
2	λ_3
3	$\neg\lambda_3$
4	λ_3

(c)

Ex.	Label
1	λ_4
2	λ_4
3	$\neg\lambda_4$
4	λ_4

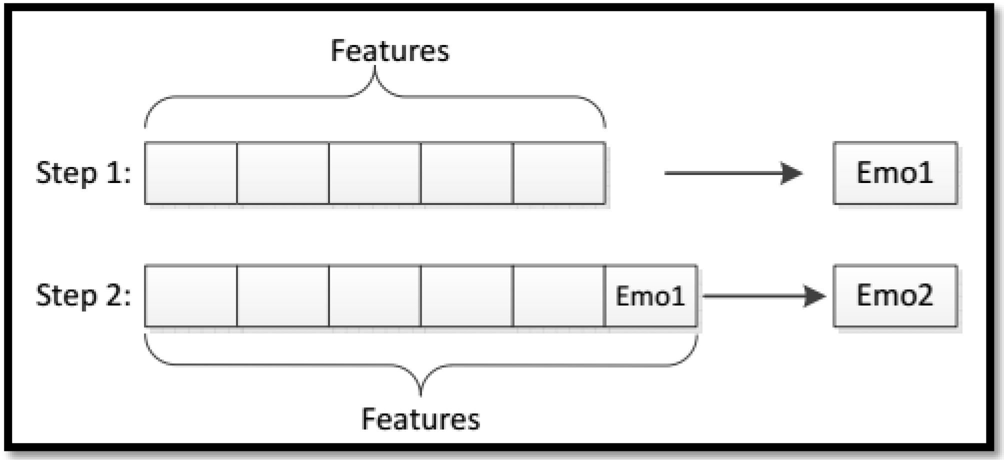
(d)

Multi-Label Classification

- Classifier Chains
  - The main drawback in binary relevance is that it ignores the label correlations
  - Classifier Chains build the model in a chain by taking label correlations into consideration
  - It uses the feature to perform binary classification on 1st label, the prediction on 1st label will be reused as the features into the 2nd step to predict the 2nd label
  - Repeat the process above until all of the labels are predicted

Multi-Label Classification

- Classifier Chains
  - It uses the feature to perform binary classification on 1st label, the prediction on 1st label will be reused as the features into the 2nd step to predict the 2nd label
  - Repeat the process above until all of the labels are predicted



Multi-Label Classification

- Label Powerset
  - Each subset of the label set will be a single label
  - Assign binary classification or multi-class classification to them
  - Find a way to aggregate the results

Multi-Label Classification Tools

- Mulan
  - Java Based
  - Reuse Weka library
  - No UI
  - http://mulan.sourceforge.net/
- Meka
  - Java Based
  - With UI
  - http://meka.sourceforge.net/

Multi-Label Classification

- References
  - G Tsoumakas, I Katakis, I Vlahavas, Mining multi-label data
  - G Tsoumakas, I Katakis , Multi-label classification: An overview
  - G Tsoumakas, E Spyromitros-Xioufis, J Vilce, Mulan: A java library for multi-label learning

Exam 2

- Time: April 25, 8:35 to 9:50 AM
- Location: Pending
- Knowledge covered: classifications
  - KNN, Naïve Bayes, Logistic regression
    - Similar to HW 8
    - Know how algorithms work, be evaluated
    - Know how to do manual calculations, similar to HW 8
    - Know how to run and read R outputs
  - Decision Tree and Ensemble Classifications
    - Understand how they work
    - Only one concept question, e.g., how bagging works

Assignments and Exam 1

Next Class

- Coding Practice: HW 7
- Coding Practice: HW 9