CAMCOS Report Day

December 9th, 2015
San Jose State University

Project Theme: Classification
On Classification: An Empirical Study of Existing Algorithms based on two Kaggle Competitions

<table>
<thead>
<tr>
<th>Team 1</th>
<th>Team 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wilson A. Florero-Salinas</td>
<td>Xiaoyan Chong</td>
</tr>
<tr>
<td>Carson Sprook</td>
<td>Minglu Ma</td>
</tr>
<tr>
<td>Dan Li</td>
<td>Yue Wang</td>
</tr>
<tr>
<td>Abhirupa Sen</td>
<td>Sha Li</td>
</tr>
<tr>
<td>Data Set</td>
<td>Data Set</td>
</tr>
<tr>
<td>Handwritten Digits</td>
<td>Spring Leaf</td>
</tr>
</tbody>
</table>

Team Advisor: Dr. Guanliang Chen
Outline

1. What is Classification?
2. The two Kaggle Competitions
3. Data Preprocessing
4. Classification Algorithms
5. Summary
6. Conclusions
7. Future Work
What is Classification?

start with a data set whose categories are already known

this data is called the *training set*

new data becomes available whose categories are *unknown*

this data set is called the *test set*

**Goal:** use the training set to predict the label of a new data point.
The Kaggle Competition

Kaggle is an international platform that hosts data prediction competitions

Students and experts in data science compete

Our CAMCOS team entered two competitions

Team 1: Digit Recognizer (Ends December 31st)

Team 2: Springleaf Marketing Response (Ended October 19th)
Team 1: The MNIST\textsuperscript{1} data set

- 28x28 images of handwritten images: 0, 1, ..., 9
- size-normalized and centered in a fixed size image
- 60,000 used for \textit{training}
- 10,000 used for \textit{testing}.

\textsuperscript{1}\text{subset of data collected by NIST, the US's National Institute of Standards and Technology}
Potential Applications

**Banking**: Check deposits

**Surveillance**: license plates

**Shipping**: Envelopes/Packages
Initial Challenges and work-arounds

High dimensional data set

Each image is stored in a vector

Computationally expensive

digits are written differently by different people

left-handed vs right-handed

Preprocess the data set

Reduce dimension → increase computation speed

apply some transformation to the images → enhance features important for classification
Data Preprocessing Methods

- In our experiments we have used the following methods
  - Principal Component Analysis (PCA)
  - Linear Discriminant Analysis (LDA)
  - 2D LDA
  - Nonparametric Discriminant Analysis (NDA)
  - Kernel PCA
  - t-Distributed Stochastic Neighbor Embedding (t-sne)
  - parametric t-sne
  - kernel t-sne
Deskewing

Before:

\[ 1 \quad 7 \]
\[ 8 \quad 2 \]

After:

\[ 1 \quad 7 \]
\[ 8 \quad 2 \]
Principal Component Analysis (PCA)

- Using too many dimensions (784) can be computationally expensive.
- Uses variance as dimensionality reduction criterion
- Throw away variables with least amount of variance
Linear Discriminant Analysis

- Reduce dimensionality, preserve as much class discriminatory information as possible

A projection with non-ideal separation

A projection with ideal separation
LDA and PCA on pairwise data

PCA:

LDA:
Classification Methods

- In our experiments we have used the following methods
  - Nearest Neighbors Methods (Instance based)
  - Naive Bayes (NB)
  - Maximum a Posterior (MAP)
  - Logistic Regression
  - Support Vector Machines (Linear Classifier)
  - Neural Networks
  - Random Forests
  - Xgboost
K nearest neighbors

k neighbors decide the group to which a test data belongs

The best result for our data set with $k = 8$ is 2.76% misclassification

The test data falls in class 1.

The neighborhood of the test data includes both the classes. However, majority belongs to class 1.
Prediction = class 1
K means

Situation 1: Data is well separated.
Each class has centroid/average.
The test data is closest to centroid 3
The test data is predicted to be from class 3.

Situation 2:
- Here test data is closer to centroid 1.
- The test data actually belongs to cluster 2
- is predicted to belong to class 1 as it is closer to the global centroid of class 1.
Solution: local k means

For every class local centroids are calculated around the test data.

The class with the local centroid closest to the test data is the prediction.

Results

- Some of best results came from local k means in the beginning.
- With $k = 14$ misclassification of 1.75%.
- Local PCA + local k means gave 1.53% misclassification.
- With deskewing this misclassification could be reduced to 1.14%.
Support Vector Machines (SVM)

Suppose you are given a data set in which their classes are known and want to classify to construct a **linear decision boundary** to classify new observations.
Support Vector Machines (SVM)

Decision boundary chosen to maximize the separation $m$ between classes.
SVM with multiple classes

SVM is a binary classifier

how can we extend SVM to more than two classes?

**One method**: One vs Rest

Construct one SVM model for each class

Separates on
Support Vector Machines (SVM)

What if data cannot be separated by a line?

Kernel SVM

Separation may be easier in higher dimensions

complex in low dimensions

simple in higher dimensions
Parameter selection for SVMs

- It is not obvious what parameters to use when training multiple models.
- Within each class, compute a corresponding gamma:
  \[ \gamma_{c_i} = \frac{1}{\text{for } j \in x_{c_i}} \sum ||x_j - kNN|| \]
- This gives a starting point for parameter selection.
- How to obtain an approximate range of parameters for training?
An alternative approach

- **Our approach**: Separately apply PCA to each digit space
- This extracts the patterns from each digit class
- We can use different parameters for each digit group.

![Diagram showing data flow and model connections for digits 0, 1, 9 with PCA and SVM(1 VS. all) models.]
Some Challenges for kernel SVM

• Using kNN, with k=5, gamma values for each class

• Error 1.25% using kNN different gamma on each model

\[ 0.03 \ 0.10 \ 0.023 \ 0.026 \ 0.031 \ 0.026 \ 0.032 \ 0.041 \ 0.024 \ 0.035 \]  

• Error 1.2% is achieved with the averaged gamma

\[
\bar{\gamma} = \frac{1}{10} \sum_{i} \gamma_{C_i} = 0.036
\]
Known results using PCA and Kernel SVM

<table>
<thead>
<tr>
<th>Method</th>
<th>Error</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>1.40%</td>
<td>743 sec</td>
</tr>
<tr>
<td>PCA + SVM</td>
<td>1.25%</td>
<td>234 sec</td>
</tr>
<tr>
<td>digit PCA + SVM</td>
<td>1.20%</td>
<td>379 sec</td>
</tr>
<tr>
<td>LDA + local Kmeans</td>
<td>7.77%</td>
<td>660 sec</td>
</tr>
</tbody>
</table>
Neural Networks

• inspired by biology
• mimic the way humans learn
• best result 1.30%
Neural Networks: Artificial Neuron

\[ \sum x_i w_i \rightarrow \varphi \rightarrow Output \]

Inputs: \( x_1, x_2, x_3, \ldots, x_n \)

Weights: \( w_1, w_2, w_3, \ldots, w_n \)

Threshold: \( \theta \)
Neural Networks: Learning
Summary of Results
Conclusion

UNDER CONSTRUCTION
Future work

UNDER CONSTRUCTION