Introduction to Machine Learning

Author: Victor Carbune

Day 2 - July 17, 2014
ROSEdu Summer Workshops

Outline

Quick Review

Model Complexity
- Bias - Variance Tradeoff

Performance
- N-Fold Cross-Validation

Features
- Feature Engineering
- Feature Preprocessing

Nonlinear Models
- Naive Bayes

Arrhythmia Classification

Quick Review: Discriminative versus Generative Models

- Both models use in the end $P(y|x)$ to assign the label
- Discriminative models learn it directly
- Generative models model $P(x, y)$ and then use Bayes rule

Quick Review: Discriminative versus Generative Models

- Both models use in the end $P(y|x)$ to assign the label
- Discriminative models learn it directly
- Generative models model $P(x, y)$ and then use Bayes rule
- Assume data set $\{(x_i, y_i)\} = \{(1, 0), (1, 1), (0, 1)\}$
- Complete the following tables

P(y|x)

	y = 0	y = 1
x = 0	?	?
x = 1	?	?

P(x,y)

	y = 0	y = 1
x = 0	?	?
x = 1	?	?

rosedu SUMMER WORKSHOPS

Quick Review: Discriminative versus Generative Models

- Both models use in the end $P(y|x)$ to assign the label
- Discriminative models learn it directly
- Generative models model $P(x, y)$ and then use Bayes rule
- Assume data set $\{(x_i, y_i)\} = \{(1, 0), (1, 1), (0, 1)\}$
- Complete the following tables

P(y|x)

	y = 0	y = 1
x = 0	0	1
x = 1	1/2	1/2

P(x,y)

	y = 0	y = 1
x = 0	0	1/3
x = 1	1/3	1/3

Quick Review: Discriminative versus Generative Models

- One should almost always prefer discriminative models over generative models, when possible

  ["On Discriminative vs. Generative classifiers", Andrew Ng and Michael Jordan, NIPS ’01]

The Gaussian Distribution (Normal Distribution)

- Parametrized by mean μ and variance σ^2 (moments)

The Gaussian Distribution (Normal Distribution)

- Parametrized by mean μ and variance σ^2 (moments)
- Formally, the function is as follows

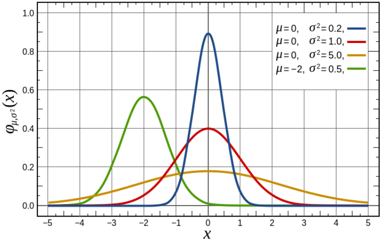
$$f(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

The Gaussian Distribution (Normal Distribution)

- Parametrized by mean μ and variance σ^2 (moments)
- Formally, the function is as follows

$$f(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- Plots for various values



http://en.wikipedia.org/wiki/Normal_distribution

The Gaussian Distribution: Motivation

- Why care about the normal distribution?

The Gaussian Distribution: Motivation

- Why care about the normal distribution?
- Because of **CLT** (Central Limit Theorem)

- Assume you have many *i.i.d.* observations of a random variable with unknown underlying distribution
- One can simply compute the mean μ and variance σ^2 of these observations and approximate the distribution through $\mathcal{N}(\mu, \sigma^2)$

The Gaussian Distribution: Motivation

- Why care about the normal distribution?
- Because of **CLT** (Central Limit Theorem)

- Assume you have many *i.i.d.* observations of a random variable with unknown underlying distribution
- One can simply compute the mean μ and variance σ^2 of these observations and approximate the distribution through $\mathcal{N}(\mu, \sigma^2)$

- ... it's also incredibly simple, though it might look complicated

The Bias - Variance Tradeoff

$$\text{MSE} = \frac{1}{n}\sum_i (\hat{y}_i - y_i)^2$$

- The mean squared error of a classifier can be split as
- MSE = bias^2 + variance + noise [homework: research the math behind it[1]]

[1] http://www.cc.gatech.edu/~lebanon/notes/estimators1.pdf

The Bias - Variance Tradeoff

$$\text{MSE} = \frac{1}{n}\sum_i (\hat{y}_i - y_i)^2$$

- The mean squared error of a classifier can be split as
- MSE = bias^2 + variance + noise [homework: research the math behind it[1]]

- To explain what the bias and variance are, assume that one trains the same model multiple times on the different subsets of data

[1] http://www.cc.gatech.edu/~lebanon/notes/estimators1.pdf

Bias of an Estimator

- The bias is defined as the difference between the average predictions of the trained models and the correct prediction

Bias of an Estimator

- The bias is defined as the difference between the average predictions of the trained models and the correct prediction
- Intuitevely, if we keep re-training the model on different datasets, for each value to be predicted, its estimates are off in some direction, on average, from the true prediction

Bias of an Estimator

- The bias is defined as the difference between the average predictions of the trained models and the correct prediction
- Intuitevely, if we keep re-training the model on different datasets, for each value to be predicted, its estimates are off in some direction, on average, from the true prediction

- Represents a systematic error due to the **model.**

Variance of an Estimator

- The variance of an estimator can be seen as the variability of the model for a specific data point

Variance of an Estimator

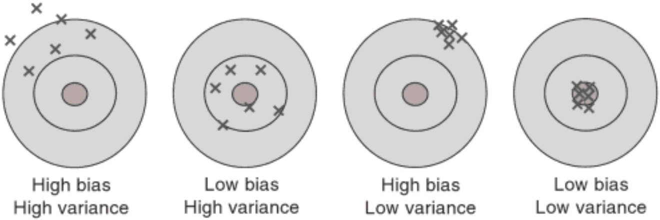
- The variance of an estimator can be seen as the variability of the model for a specific data point
- That is, how does the model's prediction vary, accross its different realizations (trainings)

Variance of an Estimator

- The variance of an estimator can be seen as the variability of the model for a specific data point
- That is, how does the model's prediction vary, accross its different realizations (trainings)

- Represents a systematic error due to the **impact of data variability on the model**

Dartboard: Bias and Variance



Dartboard analogy, Introduction to the Practice of Statistics, Moore & McCabe, 2002

Why do we say we *trade* one against the other in ML?

Can we do something to reduce them?

- First we should somehow make sure our evaluation methods are able to take them into account, and the most commonly used is **cross validation**

Can we do something to reduce them?

- First we should somehow make sure our evaluation methods are able to take them into account, and the most commonly used is **cross validation**
- Higher dimensionality often implies higher variance and one could reduce it through **principal component analysis** or **feature selection**

Can we do something to reduce them?

- First we should somehow make sure our evaluation methods are able to take them into account, and the most commonly used is **cross validation**
- Higher dimensionality often implies higher variance and one could reduce it through **principal component analysis** or **feature selection**
- Ensemble methods are often used, **bagging** is used to reduce variance, while **boosting** to reduce bias [homework!]

Testing: N-Fold Cross-Validation

- We have one dataset and we need to use it both for training and validation in order to evaluate our model. How can we do this best?

Testing: N-Fold Cross-Validation

- We have one dataset and we need to use it both for training and validation in order to evaluate our model. How can we do this best?
- Split the dataset into N subsets and use N-1 of those for training and one for testing

Testing: N-Fold Cross-Validation

- We have one dataset and we need to use it both for training and validation in order to evaluate our model. How can we do this best?
- Split the dataset into N subsets and use N-1 of those for training and one for testing
- Average the MSE accross all N trials

Testing: N-Fold Cross-Validation

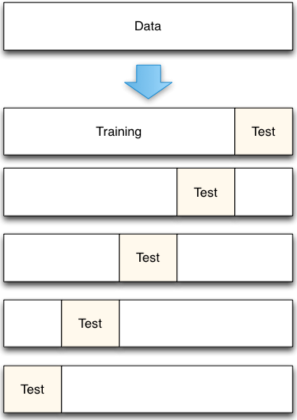


Figure : 5-Fold Cross-Validation

Testing: N-Fold Cross-Validation

- Advantage: makes best use of available data
- Disadvantage: very expensive computationally
- Leave-one-out (LOO): N = number of samples

- scikit-learn already has everything you need [2]

[2] http://scikit-learn.org/stable/modules/cross_validation.html

Quick Review

- General workflow of supervised learning

Quick Review

- General workflow of supervised learning
- Statistics behind the data
- Types of models and some particular models
- Optimization criterias and kinds of error rates

- How about getting the most out of the data we have?

Feature Engineering

- Tough problem, this is where a lot of engineering time goes in the industry

- Tough problem, this is where a lot of engineering time goes in the industry
- Everyone's dream of automating this part is getting closer to reality through deep learning, but there's still a long way

- Tough problem, this is where a lot of engineering time goes in the industry
- Everyone's dream of automating this part is getting closer to reality through deep learning, but there's still a long way

- Unfortunately, there's no recipe for this part and domain specific knowledge for designing features leads to huge improvements

Feature Engineering: Workflow

- Understand the problem you're trying to solve, read through bibliography what characteristics of data seem the most relevant (high chance somebody else did it before?)

Feature Engineering: Workflow

- Understand the problem you're trying to solve, read through bibliography what characteristics of data seem the most relevant (high chance somebody else did it before?)
- Some of the features you engineer might not help at all or even have a negative impact, when combined with the specific model you're using

Feature Engineering: Workflow

- Understand the problem you're trying to solve, read through bibliography what characteristics of data seem the most relevant (high chance somebody else did it before?)
- Some of the features you engineer might not help at all or even have a negative impact, when combined with the specific model you're using
- Understand how much the model is able ignore features that "hurt" it (e.g. in linear models, how many weights are zero?)

Feature Engineering: Workflow

- Understand the problem you're trying to solve, read through bibliography what characteristics of data seem the most relevant (high chance somebody else did it before?)
- Some of the features you engineer might not help at all or even have a negative impact, when combined with the specific model you're using
- Understand how much the model is able ignore features that "hurt" it (e.g. in linear models, how many weights are zero?)
- Experimentally vary the subset of features you're using

Feature Engineering: Workflow

- Understand the problem you're trying to solve, read through bibliography what characteristics of data seem the most relevant (high chance somebody else did it before?)
- Some of the features you engineer might not help at all or even have a negative impact, when combined with the specific model you're using
- Understand how much the model is able ignore features that "hurt" it (e.g. in linear models, how many weights are zero?)
- Experimentally vary the subset of features you're using
- Let's look at a couple of problems and their features

Feature Engineering: On-line Handwriting Recognition

Data: $\{p_i = (x_i, y_i, t_i)\}$ (two-dimensional points with time)

Two classes of features:

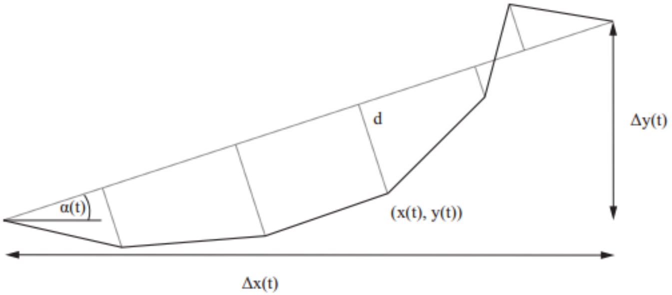
["Feature Selection for On-Line Handwriting Recognition of Whiteboard Notes", M. Liwicki and H. Bunke,

Feature Engineering: On-line Handwriting Recognition

Data: $\{p_i = (x_i, y_i, t_i)\}$ (two-dimensional points with time)

Two classes of features:

- Features for each point p_i, considering its neighbours



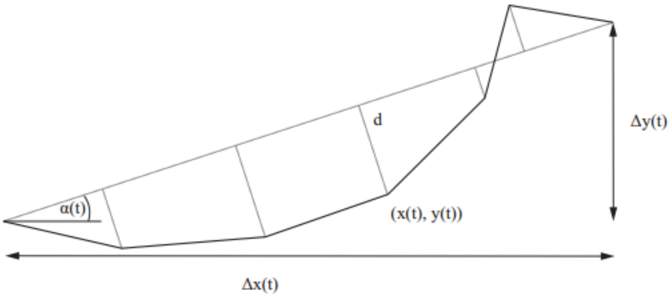
["Feature Selection for On-Line Handwriting Recognition of Whiteboard Notes", M. Liwicki and H. Bunke,

Feature Engineering: On-line Handwriting Recognition

Data: $\{p_i = (x_i, y_i, t_i)\}$ (two-dimensional points with time)

Two classes of features:

- Features for each point p_i, considering its neighbours



- Off-line matrix representation of the handwriting

["Feature Selection for On-Line Handwriting Recognition of Whiteboard Notes", M. Liwicki and H. Bunke,

Feature Engineering: On-line Handwriting Recognition

Data: $\{p_i = (x_i, y_i, t_i)\}$ (two-dimensional points with time)

Some of the first class features:

- [continous] normalized x, y coordinates
- [boolean] pen-up/pen-down
- [continous] cosine and sine of the writing direction
- [continous] average square distance to vicinity points
- [continous] length and aspect of trajectory
- [continous] angle of the straight line between vicinity ends

["Feature Selection for On-Line Handwriting Recognition of Whiteboard Notes", M. Liwicki and H. Bunke, '07]

Feature Engineering: Arrhythmia Analysis

Goal: identify one of the 16 types of arrhythmia
Data: ecg images of patients (279 features)

Types of features:

["A Supervised Machine Learning Algorithm for Arrhythmia Analysis", H. A. Guvenir et al., '98]

Feature Engineering: Arrhythmia Analysis

Goal: identify one of the 16 types of arrhythmia
Data: ecg images of patients (279 features)

Types of features:

- Patient personal records

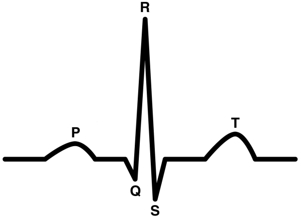
["A Supervised Machine Learning Algorithm for Arrhythmia Analysis", H. A. Guvenir et al., '98]

Feature Engineering: Arrhythmia Analysis

Goal: identify one of the 16 types of arrhythmia
Data: ecg images of patients (279 features)

Types of features:

- Patient personal records
- Features extracted from ECG



Agateller for Wikipedia
Public Domain

["A Supervised Machine Learning Algorithm for Arrhythmia Analysis", H. A. Guvenir et al., '98]

Feature Engineering: Arrhythmia Analysis

Goal: identify one of the 16 types of arrhythmia
Data: ecg images of patients

Some of the patient characteristics:

- [continous] age
- [boolean] sex
- [continous] height, weight

Some of the patient characteristics:

- [continous] average QRS duration in msec.
- [continous] average duration between onset of P and Q
- [continous] average width of Q, R, S waves
- [boolean] existence of notched R,P,T wave

["A Supervised Machine Learning Algorithm for Arrhythmia Analysis", H. A. Guvenir et al., '98]

Feature Engineering: Titanic [Kaggle Tutorial]

Goal: predict which types of people are going to survive
Data: passengers information

Some of the features:

- [continous] passenger class
- [boolean] sex
- [continous] age
- [continous] number of siblings/spouses aboard
- [continous] number of parents/children aboard
- [continous] ticket number
- [continous] port of embarkation

Check it out at: `http://www.kaggle.com/c/titanic-gettingStarted`

Feature Engineering: ?

- What particular problems are you interested in?

Feature Engineering: Preprocessing

- Preprocessing is an important step in helping the model get the most out of the data
- Luckily, scikit-learn implements a lot of these methods [3]

[3] http://scikit-learn.org/stable/modules/preprocessing.html

Feature Preprocessing: Missing Values

- The dataset might have missing or corrupted values, especially in the case where features were extracted manually by experts or through crowd-sourcing
- Solution 1: replace missing values with mean, median or the most frequent values
- Solution 2: first cluster entries together through similarity between known features, and then complete missing ones

Feature Prepocessing: Rescaling and Normalization

- Each numerical feature has it's own dimensionality (meters, kg, nanometers, seconds)

Feature Prepocessing: Rescaling and Normalization

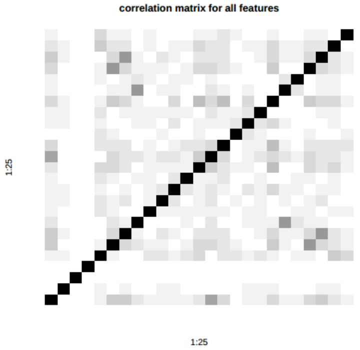
- Each numerical feature has it's own dimensionality (meters, kg, nanometers, seconds)
- Models might be sensible to these simple dimensionality variations
- Think of linear models, regularization and the impact of feature dimensionality on the weights

Feature Prepocessing: Rescaling and Normalization

- Each numerical feature has it's own dimensionality (meters, kg, nanometers, seconds)
- Models might be sensible to these simple dimensionality variations
- Think of linear models, regularization and the impact of feature dimensionality on the weights
- Common approach is to normalize features to have zero mean and unit variance, thus following $\mathcal{N}(0, 1)$, regardless whether they are actually normally distributed

Feature Prepocessing: Correlations

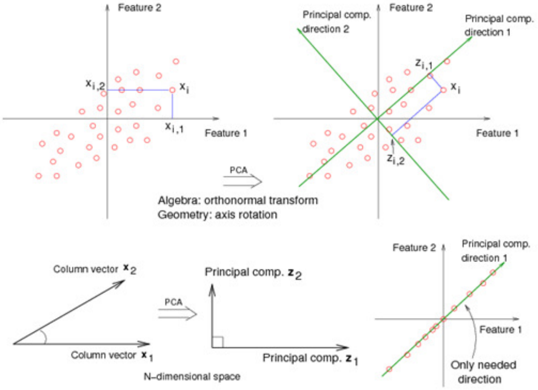
- Correlations between features [how do you identify them?] are subtle and can impact the performance of the underlying model



["Feature Selection for On-Line Handwriting Recognition of Whiteboard Notes" M. Liwicki and H.

Principal Component Analysis: Overview

- Simple unsupervised model for dimensionality reduction
- Build new K-dimensional of uncorrelated features



onlinecourses.science.psu.edu/stat857

Principal Component Analysis: Understanding Data

- Simple model to start and understand better the data
- Simple to implement, and available in scikit-learn

- Eigenfaces - 4096 features (64 x 64) $\rightarrow$ 16 features

`http://scikit-learn.org/0.13/modules/decomposition.html`

Nonlinear Models

- The input-output mapping function can be nonlinear
- Through nonlinear models one can learn this mapping
- Through nonlinear models, such as deep learning, one might also project data in a new higher-order feature spaces

Nonlinear Models: Naive Bayes

- **Naive Bayes** is a popular model, often used as benchmark

Nonlinear Models: Naive Bayes

- **Naive Bayes** is a popular model, often used as benchmark
- Recall Bayes Rule

$$P(y|x) = \frac{P(y)P(x|y)}{P(x)}$$

- Terminology

$$\text{Posterior} = \frac{\text{Prior x Likelihood}}{\text{Evidence}}$$

Nonlinear Models: Naive Bayes

- We're interested in defining the prior and likelihood
- The prior $P(y)$ can either be modeled from train data (counts), or if there's a known distribution of the class labels, it can be directly used

Nonlinear Models: Naive Bayes

- We're interested in defining the prior and likelihood
- The prior $P(y)$ can either be modeled from train data (counts), or if there's a known distribution of the class labels, it can be directly used
- Consider $x = (x_1, x_2, ..., x_n)$, then the likelihood is

$$P(x|y) = P(x_1, x_2, ..., x_n|y) =$$
$$P(x_1|y)P(x_2, x_3, ..., x_n|y, x_1) =$$
$$P(x_1|y)P(x_2|y, x_1)P(x_3, ..., x_n|y, x_1, x_2) =$$
$$P(x_1|y)P(x_2|y, x_1)P(x_3|y, x_1, x_2)..P(x_n|y, x_1, ..., x_{n-1})$$

Nonlinear Models: Naive Bayes

- The *naive* part: features are conditionally independent

Nonlinear Models: Naive Bayes

- The *naive* part: features are conditionally independent
- Mathematically, $P(x_i|y, x_1, ...x_{i-1}) = P(x_i|y)$

Nonlinear Models: Naive Bayes

- The *naive* part: features are conditionally independent
- Mathematically, $P(x_i|y, x_1, ...x_{i-1}) = P(x_i|y)$
- Back to our original problem

$$P(y|x_1, ..., x_n) \propto P(y) \prod_i P(x_i|y)$$

Nonlinear Models: Naive Bayes

- The *naive* part: features are conditionally independent
- Mathematically, $P(x_i|y, x_1, ...x_{i-1}) = P(x_i|y)$
- Back to our original problem
$$P(y|x_1, ..., x_n) \propto P(y) \prod_i P(x_i|y)$$
- So what about $P(x_i|y)$?

Nonlinear Models: Naive Bayes

- Think about spam classification of a document (words w_i): $P(y = spam|w_1, ...w_n) \propto P(y = spam)\prod_i P(w_i|y = spam)$

Nonlinear Models: Naive Bayes

- Think about spam classification of a document (words w_i): $P(y = spam|w_1, ...w_n) \propto P(y = spam) \prod_i P(w_i|y = spam)$
- How can we compute $P(w_i|y = spam)$ from training set?

Nonlinear Models: Gaussian Naive Bayes

- In the case of continous features, the probability is often modeled using a normal distribution
- The mean and variance are computed from the feature values found in the training data

$$P(x_i = v|y) = \frac{1}{\sqrt{2\pi\sigma^2_{x_i,y}}} \exp^{-\frac{(v-\mu_{x_i,y})^2}{2\sigma^2_{x_i,y}}}$$

Questions?

- Questions?

Questions?

- Questions?
- Next: classifying arrhythmia patterns using naive bayes