COMP755-Lect08

September 23, 2018

1 COMP 755

Plan for today

1. Review Generative models for classification
2. Naive Bayes with Bernoulli feature distribution
3. Tuning and evaluating models
    - Cross-Validation
    - ROC plots

2 Multivariate Gaussian distribution -- dependent case

Suppose we have n standard random variables (0 mean, unit variance)

$$z_i \sim \mathcal{N}(0, 1), \quad i = 1, \ldots n$$

and we are given a vector ¯ of length n and a full-rank matrix A of size $n \times n$.

Distribution of $\mathbf{x} = A\mathbf{z} + \mu$ is

$$p(\mathbf{x}) = (2\pi)^{-\frac{n}{2}} (\det \Sigma)^{-\frac{1}{2}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - ¯)^T \Sigma^{-1} (\mathbf{x} - ¯) \right\}$$

where $\Sigma = AA^T$.

- μ is **mean** of the Gaussian
- Σ is **covariance** matrix

3 Maximum likelihood estimates of mean and covariance

Given data $\{\mathbf{x}_i \in \mathbb{R}^n | i = 1, \ldots, T\}$ maximum likelihood estimates (MLE) of mean and covariance are:

$$¯^{\text{MLE}} = \frac{1}{T} \sum_{i=1}^{T} \mathbf{x}_i$$

$$\Sigma^{\text{MLE}} = \frac{1}{T} \sum_{i=1}^{T} \underbrace{\left(\mathbf{x}_i - ¯^{\text{MLE}}\right) \left(\mathbf{x}_i - ¯^{\text{MLE}}\right)^T}_{\text{a matrix of size } n \times n}$$

Dimensionality * ¯$^{\text{MLE}}$ is of same dimension as a single data point $n \times 1$. * Σ^{MLE} is a matrix of size $n \times n$

Note that $\mathbf{x}\mathbf{x}^T$ and $\mathbf{x}^T\mathbf{x}$ are not the same. Former is a matrix, latter is a scalar.

4 Generative models for classification

There are two ways to factorize joint probability of labels and features

$$p(y, \mathbf{x}|\theta) = p(y|\mathbf{x}, \theta)p(\mathbf{x}|\theta) = p(\mathbf{x}|y, \theta)p(y|\theta)$$

The second one given us a simple process to *GENERATE* data:

1. First select label according $p(y|\theta)$, say it was c
2. Now generate features $p(\mathbf{x}|y = c, \theta)$

Once we have such a model we can obtain the conditional probability $p(y|\mathbf{x})$ using Bayes rule

$$p(y = c|\mathbf{x}) = \frac{p(y = c|\theta)p(\mathbf{x}|y = c, \theta)}{\sum_k p(y = k|\theta)p(\mathbf{x}|y = k, \theta)}$$

and we can predict label for a new feature vector $\mathbf{x}$

5 Naive Bayes

$$p(y = c|\pi) = \pi_k$$
$$p(\mathbf{x}|y = c, \theta) = \prod_j p(x_j|y = c, \theta_{j,c})$$

Parameters are * π_c prior probability that a sample comes from the class c * $\theta_{j,c}$ parameters for the j^{th} feature for class c

In general, there are many variants of Naive Bayes.

You can choose different distributions for $p(x_j|y = c)$ * Gaussian -- continuous features * Bernoulli -- binary features * Binomial -- count of positive outcomes * Categorical -- discrete features * Multinomial -- count of particular discrete outcomes

6 Bag of Words representation

Review 188_7: "I'm a **huge classic** film **buff**, but am just **getting** in to **silent movies**. A **lot** of **silent films** don't **hold** my **attention**, ..."

Review 196_9: "... **fans** of the **silent era**, with many **cameos**, **adds** to the **overall fun** ..."

Converted into a row of word counts

Document_id	#attention	...	#classic	...	#fun	...	#silent	...
188_7	1	...	1	...	0	...	2	...
196_9	0	...	0	...	1	...	1	...
...	...	...	...	...	...	...	...	...

Features can also be word presence/absence, rather than counts as above.

These types of representations are called **bag-of-words**.

In your homework we use bag-of-words representation of movie reviews to predict sentiment.

7 Naive Bayes for spam classification

One approach to classifying e-mail spam (1) vs not spam (0) is to construct a Naive Bayes model using a bag-of-words representation.

Feature vector $\mathbf{x}$ is W long vector of word presence absence in an e-mail

$$\begin{aligned}
p(y=1) &= \pi && \text{prior probability that message is spam} \\
p(y=0) &= 1-\pi && \text{prior probability that message is not spam} \\
p(x_j=1|y=1) &= \theta_{1,j} && \text{probability that word } j \text{ appears in a spam e-mail} \\
p(x_j=0|y=1) &= 1-\theta_{1,j} && \\
p(x_j=1|y=0) &= \theta_{0,j} && \text{probability that word } j \text{ appears in non-spam e-mail} \\
p(x_j=0|y=0) &= 1-\theta_{0,j} &&
\end{aligned}$$

Q: What is the size of π?

Q: What is the size of θ?

8 Naive Bayes for spam classification

$$\begin{aligned}
p(y=1) &= \pi && \text{prior probability that message is spam} \\
p(y=0) &= 1-\pi && \text{prior probability that message is not spam} \\
p(x_j=1|y=1) &= \theta_{1,j} && \text{probability that word } j \text{ appears in a spam e-mail} \\
p(x_j=0|y=1) &= 1-\theta_{1,j} && \text{probability that word } j \text{ is absent in a spam e-mail} \\
p(x_j=1|y=0) &= \theta_{0,j} && \text{probability that word } j \text{ is absent in non-spam e-mail} \\
p(x_j=0|y=0) &= 1-\theta_{0,j} &&
\end{aligned}$$

More compactly for math purposes

$$\begin{aligned}
p(y) &= \pi^y (1-\pi)^{1-y} \\
p(x_j|y) &= \left[\theta_{1,j}^{x_j}(1-\theta_{1,j})^{1-x_j}\right]^y \left[\theta_{0,j}^{x_j}(1-\theta_{0,j})^{1-x_j}\right]^{1-y}
\end{aligned}$$

9 Naive Bayes for spam classification -- likelihood

$$\begin{aligned}
\mathcal{LL}(\theta, \pi \big| X, \mathbf{y}) = & \underbrace{\sum_{i=1}^{N}}_{\text{samples}} \left[y_i \log \pi + (1 - y_i) \log(1 - \pi) \right] \\
& + \underbrace{\sum_{i=1}^{N}}_{\text{samples}} \underbrace{\sum_{j=1}^{W}}_{\text{words}} y_i x_{i,j} \log \theta_{1,j} \\
& + \underbrace{\sum_{i=1}^{N}}_{\text{samples}} \underbrace{\sum_{j=1}^{W}}_{\text{words}} y_i (1 - x_{i,j}) \log(1 - \theta_{1,j}) \\
& + \underbrace{\sum_{i=1}^{N}}_{\text{samples}} \underbrace{\sum_{j=1}^{W}}_{\text{words}} (1 - y_i) x_{i,j} \log(\theta_{0,j}) \\
& + \underbrace{\sum_{i=1}^{N}}_{\text{samples}} \underbrace{\sum_{j=1}^{W}}_{\text{words}} (1 - y_i)(1 - x_{i,j}) \log(1 - \theta_{0,j})
\end{aligned}$$

Work out derivatives and updates for π_1 and θ_1, j on the board

10 Naive Bayes for spam classification -- learning

Given a training data with N labeled e-mails, training a Naive Bayes spam model is accomplished using

$$\pi_1 = \frac{\sum_{i=1}^{N} [y_i = 1]}{N} \qquad \text{frequency of spam e-mail in the training data}$$

$$\theta_{1,j} = \frac{\sum_{i=1}^{N} x_{i,j} * [y_i = 1]}{\sum_{i=1}^{N} [y_i = 1]} \qquad \text{freqency of word } j \text{ in spam e-mail}$$

$$\theta_{0,j} = \frac{\sum_{i=1}^{N} x_{i,j} * [y_i = 0]}{\sum_{i=1}^{N} [y_i = 0]} \qquad \text{frequency of word } j \text{ in non-spam e-mail}$$

11 Naive Bayes for spam classification -- prediction

Prediction for a new e-mail given its bag-of-words representation $\mathbf{x}$

$$p(y = 1 \big| \mathbf{x}) = \frac{p(y = 1, \mathbf{x})}{p(\mathbf{x})} = \frac{p(y = 1, \mathbf{x})}{p(y = 1, \mathbf{x}) + p(y = 0, \mathbf{x})}$$

If interested only in the most likely label of a message represented by $\mathbf{x}$

$$\operatorname*{argmax}_c \log p(y = c, \mathbf{x}) = \operatorname*{argmax}_c \log \pi_c + \sum_{j=1}^{W} \left[x_j \log \theta_{c,j} + (1 - x_j) \log(1 - \theta_{c,j}) \right]$$

12 Spam filtering -- smoothing

$$\pi_1 = \frac{\sum_{i=1}^{N}[y_i = 1]}{N} \quad \text{frequency of spam e-mail in the training data}$$
$$\theta_{1,j} = \frac{\sum_{i=1}^{N} x_{i,j} * [y_i = 1]}{\sum_{i=1}^{N}[y_i = 1]} \quad \text{freqency of word } j \text{ in spam e-mail}$$
$$\theta_{0,j} = \frac{\sum_{i=1}^{N} x_{i,j} * [y_i = 0]}{\sum_{i=1}^{N}[y_i = 0]} \quad \text{frequency of word } j \text{ in non-spam e-mail}$$

Q: What is the value of $\theta_{1,j}$ if the word has never been seen in a spam e-mail in the dataset? Is this a problem? If so, why and how would you fix it?

http://spamassassin.apache.org/

Uses Naive Bayes approach with smoothing -- even though a word has not been seen it does not have probability 0.

13 Hyperparameters

In penalized regression we added ridge term

$$\mathcal{LL}(\beta) - \frac{\lambda}{2}\sum_{j>0}\beta_j^2$$

Q: How do we choose λ? What happens if we try to maximize penalized log-likelihood with respect to λ?

Q: How do we choose which words to count in our document classifiers? All of them? Do we have enough samples?

14 Hyperparameters

Hyperparameters, unlike parameters, typically constrain the model complexity to avoid overfitting.

Overfitting occurs when our performance on training set is optimistic compared to performance on test set

15 How do we select hyperparameters?

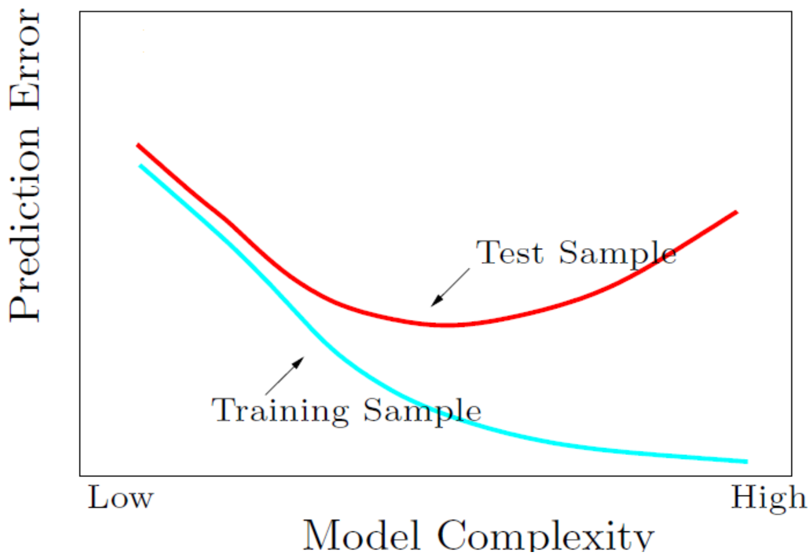
One approach is to split the dataset into three parts: 1. Training 2. Validation 3. Test

Training subset is used to train models for different settings of hyperparameters for example $\lambda \in \{0.1, 0.001, 0.0001..\}$

Validation subset is used to evaluate accuracy of the different models we trained and select the best model.

Test set is used to compute the accuracy of the model selected on validation set.

Q: Why can't we just report accuracy of the best model on the validation set?



16 How do we select hyperparameters if the dataset is small?

Splitting data into three parts makes sense on a large dataset, but can hurt your performance on a small dataset.

We can't look at the test set during training, but validation set seems to be just sitting around

Q: Can we somehow use more of the training and validation data for training? What is the problem of leaving just one sample for validation?

17 The idea: cross-validation

First take out test data, then split the remaining data into k parts and treat each part as validation while training on the rest.

An example of 4-fold cross validation:

0. Split data into disjoint subsets $\text{Data} = \text{Data}_1 \cup \text{Data}_2 \cup \text{Data}_3 \cup \text{Data}_4$
1. Train on $\text{Data}_2 \cup \text{Data}_3 \cup \text{Data}_4$ compute Error_1 on Data_1
2. Train on $\text{Data}_1 \cup \text{Data}_3 \cup \text{Data}_4$ compute Error_2 on Data_2
3. Train on $\text{Data}_1 \cup \text{Data}_2 \cup \text{Data}_4$ compute Error_3 on Data_3
4. Train on $\text{Data}_1 \cup \text{Data}_2 \cup \text{Data}_3$ compute Error_4 on Data_4

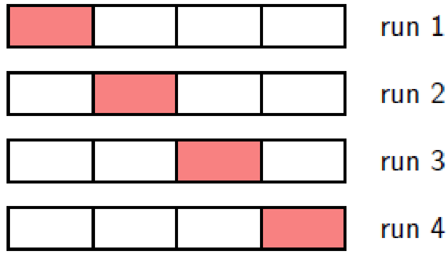
Report

$$\text{CVError} = \frac{\text{Error}_1 + \text{Error}_2 + \text{Error}_3 + \text{Error}_4}{4}$$

18 k-fold cross-validation for linear regression

19 Classification performance -- confusion matrix

Say we trained a classifier and want to see how well it works.



Foreach i=1:K

1. Set $\text{Test} = \text{Set}_i$ and $\text{Train} = \bigcup_{k \neq i} \text{Set}_k$
2. Learn β_0 and $\boldsymbol{\beta}$ on $\{(\mathbf{x}_i, y_i) : i \in \text{Train}\}$ using α
3. $\text{CVErr}_i(\alpha) \quad = \sum_{i \in \text{Test}} [(y_i - \beta_0 - \mathbf{x}_i' \boldsymbol{\beta})^2]$

$\text{CVErr}(\alpha) \quad = \frac{1}{n} \sum_k \text{CVErr}_k(\alpha)$

4-fold illustration

We can report a single number, accuracy, that tells us how often it is right.
However, this hides information about where the classifier fails.
Confusion matrix is given by:

Predicted True	$y = 1$	$y = 0$
$\hat{y} = 1$	True Positive	False Positive
$\hat{y} = 0$	False Negative	True Positive

20 Classification performance -- prediction rates

Prediction rates * true positive rate

$$TPR = \frac{TP}{TP + FN}$$

* false positive rate

$$FPR = \frac{FP}{TN + FP}$$

* true negative rate

$$TNR = \frac{TN}{TN + FP}$$

* false negative rate

$$FNR = \frac{FN}{TP + FN}$$

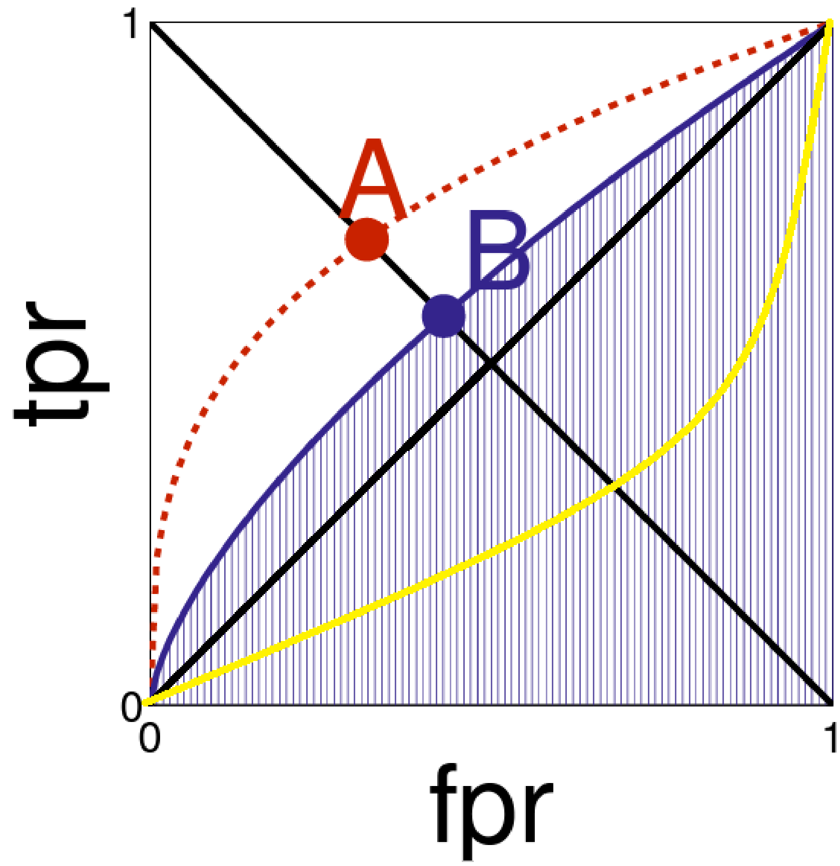
Note that $TP + FN$ is total number of positive examples.
Similarly $TN + FP$ is total number of negative examples.

21 Classification performance -- ROC curves

Predictions are based on a cutoff

$$p(y = 1|\mathbf{x}) > \tau$$

where τ is typically 0.5.


1
A
B
tpr
0
0
1
fpr