

Notes

Machine Learning by Andrew Ng on Coursera

Sparsh Jain

November 20, 2020

Contents

1	Introduction	5
1.1	Supervised Learning	5
1.1.1	Regression Problem	5
1.1.2	Classification Problems	5
1.2	Unsupervised Learning	5
I	Supervised Learning	6
2	Linear Regression with One Variable	7
2.1	Notations	7
2.2	Supervised Learning	7
2.3	Gradient Descent	8
2.4	Gradient Descent for Linear Regression	8
3	Linear Algebra	9
3.1	Matrix	9
3.2	Vector	9
3.3	Addition and Scalar Multiplication	10
3.4	Matrix Matrix Multiplication	10
3.5	Inverse and Transpose	10
4	Linear Regression with Multiple Variables	12
4.1	Notations	12
4.2	Hypothesis	12
4.3	Gradient Descent	13
4.3.1	Feature Scaling	13
4.3.2	Mean Normalization	14
4.3.3	Learning Rate	14
4.4	Features and Polynomial Regression	14
4.4.1	Features	14
4.4.2	Polynomial Regression	14

4.5	Normal Equation	15
4.5.1	Non Invertibility of $X^T X$	15
5	Octave Tutorial	16
6	Classification	17
6.1	Logistic Regression	17
6.2	Decision Boundary	18
6.3	Cost Function	18
6.4	Advanced Optimization:	20
6.5	Multi-Class Classification	21
6.5.1	One vs All	21
7	Regularization	22
7.1	Problem of Overfitting	22
7.2	Addressing Overfitting	22
7.3	Cost Function	23
7.4	Regularized Linear Regression	23
7.5	Regularized Logistic Regression	25
8	Neural Networks	26
8.1	Non-Linear Hypothesis	26
8.2	Neural Networks	26
8.3	Model Representation	26
8.4	Notations	27
8.5	Forward Propagation	28
8.6	Multi-Class Classification	29
9	Back Propagation	30
9.1	Notations	30
9.2	Cost Function	31
9.3	Backpropagation Algorithm	31
9.3.1	Gradient Computation	31
9.4	Implementation Note	33
9.4.1	Unrolling Parameters	33
9.4.2	Learning Algorithm	34
9.5	Gradient Checking	34
9.5.1	Numerical Estimation of gradients	34
9.5.2	Implementation Note:	36
9.6	Random Initialization	36
9.7	Putting it all together	37

10 Applying Machine Learning	39
10.1 Deciding what to try next	39
10.1.1 Debugging a Learning Algorithm	39
10.1.2 Machine Learning Diagnostics	39
10.2 Evaluating your Hypothesis	40
10.2.1 The Standard Way	40
10.2.2 Train/Test Procedure	40
10.3 Model Selection	40
10.3.1 Evaluating your hypothesis	41
10.4 Bias vs Variance	42
10.4.1 Effect of regularization	42
10.5 Learning Curves	43
10.5.1 High Bias	43
10.5.2 High Variance	44
10.6 Deciding what to try next (Revisited)	44
10.6.1 Neural Networks	44
11 Machine Learning System Design	45
11.1 Prioritizing what to work on: Spam Classification Example	45
11.1.1 Time Management?	45
11.2 Error Analysis	46
11.2.1 Recommended approach	46
11.2.2 Numerical Evaluation	46
11.3 Skewed Classes	47
11.3.1 Precision/Recall	47
11.3.2 Trade-Off	48
11.3.3 Working with Large Data	48
12 Support Vector Machines	49
12.1 Optimization Objective	49
12.1.1 Alternative view of logistic regression	49
12.1.2 SVM Hypothesis	50
12.2 Large Margin	51
12.2.1 Intuition	51
12.2.2 Math	51
12.3 Kernels	52
12.3.1 Non-linear Decision Boundary	52
12.3.2 Choosing the Landmarks	53
12.3.3 SVM Parameters:	54
12.4 Using SVM	54
12.4.1 Multi-Class Classification	55

12.4.2 Logistic Regression vs SVM	55
II Unsupervised Learning	56
13 Clustering	57
13.1 Unsupervised Learning	57
13.2 Clustering	57
13.2.1 Application	57
13.3 K -means	57
13.3.1 Algorithm	57
13.3.2 K -means for non-separated clusters	58
13.3.3 Optimization Objective	58
13.3.4 Random Initialization	59
13.3.5 Choosing K	59
14 Dimensionality Reduction	60
14.1 Motivation	60
14.1.1 Data Compression	60
14.1.2 Data Visualization	60
14.2 Principle Component Analysis	60
14.2.1 Problem Formulation	60
14.3 Algorithm	61
14.3.1 Data Preprocessing	61
14.3.2 Procedure	61
14.3.3 Reconstruction	62
14.4 Choosing the number of principle components	62
14.4.1 Algorithm	63
14.5 Applying PCA	63
14.5.1 Supervised Learning Speedup	63
14.5.2 Visualization	63
14.5.3 Bad use of PCA: To prevent overfitting	64
14.5.4 PCA is sometimes used where it shouldn't be	64
15 Anomaly Detection	65
15.1 Example	65
15.1.1 Aircraft Engine!	65
15.1.2 Fraud Detection	65
15.1.3 Manufacturing	65
15.1.4 Monitoring computers in a data center	65
15.2 Gaussian (Normal) Distribution	66
15.2.1 Definition	66

15.2.2	Parameter Estimation	66
15.3	Algorithm	66
15.3.1	Density Estimation	66
15.3.2	Anomaly Detection Algorithm	67
15.4	Developing and Evaluating an Anomaly Detection System	68
15.4.1	The importance of real-number evaluation	68
15.4.2	Algorithm Evaluation	68
15.5	Anomaly Detection vs Supervised Learning	69
15.6	Choosing what features to use	69
15.6.1	Transformations	69
15.6.2	Error analysis for anomaly detection	70
15.6.3	Other techniques:	70
15.7	Multivariate Gaussian (Normal) Distribution	70
15.8	Anomaly Detection using Multivariate Gaussian Distribution . . .	71
15.8.1	Model	71
15.8.2	Anomaly Detection	71
15.8.3	Relationship to original model	72

Part I

Supervised Learning

Part II

Unsupervised Learning

Chapter 15

Anomaly Detection

15.1 Example

15.1.1 Aircraft Engine!

Features:

x_1 = heat generated
 x_2 = vibration intensity
...

Dataset: $\{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$

Question: Is x_{test} anomalous?

Answer: Model $p(x)$ from dataset, and if $p(x_{test}) < \epsilon$ then flag anomaly otherwise OK!

15.1.2 Fraud Detection

Identify unusual users by checking $p(x) < \epsilon$

15.1.3 Manufacturing

Just like Aircraft Engine

15.1.4 Monitoring computers in a data center

Check which system is probably required a review by a system administrator.

15.2 Gaussian (Normal) Distribution

15.2.1 Definition

Say $x \in \mathbb{R}$. If x is distributed *Gaussian* with mean μ and variance σ^2 (σ is the standard deviation).

$$x \sim \mathcal{N}(\mu, \sigma^2)$$
$$p(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Bell shaped curve centered at μ and width varying by σ

15.2.2 Parameter Estimation

Given a dataset, estimate μ and σ or σ^2

Maximum Likelihood Estimate (MLE):

$$\mu = \frac{1}{m} \sum_{i=1}^m x^{(i)}$$
$$\sigma^2 = \frac{1}{m} \sum_{i=1}^m (x^{(i)} - \mu)^2$$

15.3 Algorithm

15.3.1 Density Estimation

Training Set: $\{x^{(1)}, \dots, x^{(m)}\}$

Each example is $x \in \mathbb{R}$

Model $p(x)$ as such:

$$p(x) = p(x_1)p(x_2)\dots p(x_n)$$

Assume each feature is distributed Gaussian independently

$$\begin{aligned}x_1 &\sim \mathcal{N}(\mu_1, \sigma_1^2) \\x_2 &\sim \mathcal{N}(\mu_2, \sigma_2^2) \\&\vdots \\x_n &\sim \mathcal{N}(\mu_n, \sigma_n^2) \\p(x) &= p(x_1; \mu_1, \sigma_1^2) p(x_2; \mu_2, \sigma_2^2) \dots p(x_n; \mu_n, \sigma_n^2) \\&= \prod_{j=1}^n p(x_j; \mu_j, \sigma_j^2)\end{aligned}$$

In practice, it works fine even if the features are not really *independent*.

15.3.2 Anomaly Detection Algorithm

1. Choose features x_i that might be indicative of anomaly.
2. Given a training set $\{x^{(1)}, \dots, x^{(m)}\}$, fit parameters $\mu_1, \dots, \mu_n, \sigma_1^2, \dots, \sigma_n^2$

$$\begin{aligned}\mu &= \frac{1}{m} \sum_{i=1}^m x^{(i)} \\\sigma^2 &= \frac{1}{m} \sum_{i=1}^m (x^{(i)} - \mu)^2\end{aligned}$$

3. Given a new example x , compute $p(x)$

$$\begin{aligned}p(x) &= \prod_{j=1}^n p(x_j; \mu_j, \sigma_j^2) \\&= \prod_{j=1}^n \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left(-\frac{(x_j - \mu_j)^2}{2\sigma_j^2}\right)\end{aligned}$$

4. Anomaly if $p(x) < \epsilon$

15.4 Developing and Evaluating an Anomaly Detection System

15.4.1 The importance of real-number evaluation

When developing a learning algorithm (choosing features, etc.), making decisions is much easier if we have a way of evaluating our learning algorithm.

Assume we have some labelled data, of anomalous and non-anomalous examples. ($y = 0$ if normal, $y = 1$ if anomalous).

Training set: $x^{(1)}, x^{(2)}, \dots, x^{(m)}$ (assume normal)

Cross validation set: $(x_{cv}^{(1)}, y_{cv}^{(1)}), \dots, (x_{cv}^{(m_{cv})}, y_{cv}^{(m_{cv})})$

Test set: $(x_{test}^{(1)}, y_{test}^{(1)}), \dots, (x_{test}^{(m_{test})}, y_{test}^{(m_{test})})$

Assume we include examples in cross validation set and test set which are known to be anomalous.

Aircraft Engines Example

Recommended split:

- 10000 good (normal) engines
- 20 - 50 flawed (anomalous) engines
- Training Set: 6000 good engines
- CV: 2000 good engines, 10 anomalous
- Test: 2000 good engines, 10 anomalous

15.4.2 Algorithm Evaluation

1. Fit model $p(x)$ on training set $\{x^{(1)}, \dots, x^{(m)}\}$
2. On a cross-validation/test example x , predict

$$y = \begin{cases} 1 & \text{if } p(x) < \epsilon \text{ (anomaly)} \\ 0 & \text{otherwise} \end{cases}$$

3. Possible evaluation metrics

- True positive, false positive, false negative, true negative
- Precision/Recall
- F_1 -score

4. Can also use cross validation set to choose parameter ϵ

15.5 Anomaly Detection vs Supervised Learning

If we have labelled data, why not use supervised learning?

Anomaly Detection	Supervised Learning
Very small number of positive examples ($y = 1$).	Large number of positive and negative examples.
Large number of negative examples ($y = 0$).	
Many different <i>types</i> of anomalies. Hard for any algorithm to learn from positive examples what the anomalies look like.	Enough positive examples for algorithm to get a sense of what positive examples are like.
Future anomalies may look nothing like any of the anomaly examples we've seen so far.	Future positive examples likely to be similar to ones in training set.
Fraud Detection	Email spam classification
Manufacturing (e.g. aircraft engine)	Weather prediction (sunny/rainy/etc.)
Monitoring machines in a data center	Cancer classification
\vdots	\vdots

15.6 Choosing what features to use

15.6.1 Transformations

Works fine even if data isn't gaussian, but usually is a good sanity check for a feature.

If not gaussian, play with different kinds of transformations and get something similar to gaussian, like log transformation.

Other transformations:

$$x_1 \leftarrow \log x_1$$

$$x_1 \leftarrow \log x_2 + c$$

$$x_3 \leftarrow \sqrt{x_3} = x_3^{\frac{1}{2}}$$

$$x_4 \leftarrow x_4^{\frac{1}{3}}$$

15.6.2 Error analysis for anomaly detection

Want $p(x)$ large for normal examples x .

$p(x)$ small for anomalous examples x .

Most common problem: $p(x)$ is comparable (say, both large) for normal and anomalous examples.

Look at such anomalous examples and try to get a new feature to distinguish.

15.6.3 Other techniques:

Choose features that might take on unusually large or small values in the event of an anomaly.

Example: Monitoring computers in a data center running servers.

x_1 = memory use

x_2 = disk access

x_3 = CPU load

x_4 = network traffic

Possible anomaly = infinite loop, so $x_5 = \frac{\text{CPU Load}}{\text{network traffic}}$

15.7 Multivariate Gaussian (Normal) Distribution

Example

Monitoring machines in a data center. Consider x_1 = CPU load and x_2 = Memory Use. An outlier in 2D plot would not be triggered anomalous when treating x_1 and x_2 .

$x \in \mathbb{R}^n$. Don't model $p(x_1), p(x_2), \dots$, etc. separately. Model $p(x)$ all in one go.

Parameters: $\mu \in \mathbb{R}^n, \Sigma \in \mathbb{R}^{n \times n}$ (covariance matrix)

Equation:

$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu)\right)$$

where $|\Sigma|$ = determinant of Σ = `det`(Σ) (in octave)

15.8 Anomaly Detection using Multivariate Gaussian Distribution

15.8.1 Model

Parameters: μ, Σ

Parameter Fitting: Given training set $\{x^{(1)}, \dots, x^{(m)}\}$

$$\mu = \frac{1}{m} \sum_{i=1}^m x^{(i)}$$
$$\Sigma = \frac{1}{m} \sum_{i=1}^m (x^{(i)} - \mu)(x^{(i)} - \mu)^T$$

15.8.2 Anomaly Detection

1. Fit model $p(x)$ by setting μ, Σ
2. Given a new example x , compute

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

3. Flag an anomaly if $p(x) < \epsilon$

15.8.3 Relationship to original model

Original model corresponds to multivariate gaussian where the contours of the gaussian are axis aligned. The constraint (mathematically) is

$$\Sigma = \begin{bmatrix} \sigma_1^2 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & 0 & \dots & 0 \\ 0 & 0 & \ddots & & 0 \\ 0 & \dots & 0 & \sigma_{n-1}^2 & 0 \\ 0 & \dots & 0 & 0 & \sigma_n^2 \end{bmatrix}$$

Original Model

$$\prod_{j=1}^n p(x_j; \mu_j, \sigma_j^2)$$

Manually create features to capture anomalies where x_1, x_2 take unusual combination of values.

Computationally cheaper (scales better to large n , say 100,000).

OK even if m (training set size) is small.

Multivariate Gaussian

$$\frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

Automatically captures correlations between features.

Computationally more expensive.

Must have $m > n$, or else Σ is non-invertible. (Typical rule of thumb: $m \geq 10n$)

Remark. If Σ is singular (non-invertible), either $m > n$ fails, or you have redundant (linearly dependent) features.