Notes

Machine Learning by Andrew Ng on Coursera

Sparsh Jain

November 20, 2020

Contents

Part I

Supervised Learning

Part II

Unsupervised Learning

Chapter 15

Anomaly Detection

15.1 Example

15.1.1 Aircraft Engine!

Features:

$$\begin{aligned} x_1 &= \text{heat generated} \\ x_2 &= \text{vibration intensity} \\ &\dots \end{aligned}$$

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 Likelyhood 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: $\{x^{(1)}, \ldots, x^{(m)}\}$
Each example is $x \in \mathbb{R}$
Model $p(x)$ as such:

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

Assume each feature is distributed Gaussian independently

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

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 $\{x^{(1)}, \ldots, x^{(m)}\}$, fit paramters $\mu_1, \ldots, \mu_n, \sigma_1^2, \ldots, \sigma_2^2$

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

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: $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$ (assume normal)

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

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

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 $\{x^{(1)}, \ldots, x^{(m)}\}$
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 negative examples ($y = 0$).	Large number of positive and negative examples.
Many different *types* 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.

$$x_1 = \text{memory use}$$
$$x_2 = \text{disk access}$$
$$x_3 = \text{CPU load}$$
$$x_4 = \text{network traffic}$$

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

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 $p(x_1), p(x_2), \ldots$, 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 $\left\{x^{(1)}, \ldots, x^{(m)}\right\}$

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

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 contraint (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	**Multivariate Gaussian**
$\prod_{j=1}^{n} p(x_j; \mu_j, \sigma_j^2)$	$\frac{1}{(2\pi)^{\frac{n}{2}} \lvert\Sigma\rvert^{\frac{1}{2}}} e^{\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)}$
Manually create features to capture anomalies where x_1, x_2 take unusual combination of values.	Automatically captures correlations between features.
Computationally cheaper (scales better to large n, say 100,000).	Computationally more expensive.
OK even if m (training set size) is small.	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.

Check Lecture15.pdf for more details.