Regression Problem

(adapted from) Prof. Alexander Ihler
Overview

- Regression Problem Definition and define parameters $\Theta$.
- Prediction using $\Theta$ as parameters
- Measure the error
- Finding good parameters $\Theta$ (direct minimization problem)
- Non-Linear regression problem
Example of Regression

- Vehicle price estimation problem
  - Features  \( x \): Fuel type, The number of doors, Engine size
  - Targets  \( y \): Price of vehicle
  - Training Data: (fit the model)

<table>
<thead>
<tr>
<th>Fuel type</th>
<th>The number of doors</th>
<th>Engine size (61 – 326)</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>70</td>
<td>11000</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>120</td>
<td>17000</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>310</td>
<td>41000</td>
</tr>
</tbody>
</table>

- Testing Data: (evaluate the model)

<table>
<thead>
<tr>
<th>Fuel type</th>
<th>The number of doors</th>
<th>Engine size (61 – 326)</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>150</td>
<td>21000</td>
</tr>
</tbody>
</table>
Example of Regression

- Vehicle price estimation problem

<table>
<thead>
<tr>
<th>Fuel type</th>
<th>#of doors</th>
<th>Engine size</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>70</td>
<td>11000</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>120</td>
<td>17000</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>310</td>
<td>41000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fuel type</th>
<th>The number of doors</th>
<th>Engine size (61 – 326)</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>150</td>
<td>21000</td>
</tr>
</tbody>
</table>

- $\Theta = [-300, -200, 700, 130]$
- Ex #1 = $-300 + -200 * 1 + 4 * 700 + 70 * 130 = 11400$
- Ex #2 = $-300 + -200 * 1 + 2 * 700 + 120 * 130 = 16500$
- Ex #3 = $-300 + -200 * 2 + 2 * 700 + 310 * 130 = 41000$
  - Mean Absolute Training Error = $1/3 * (400 + 500 + 0) = 300$
- Test = $-300 + -200 * 2 + 4 * 700 + 150 * 130 = 21600$
  - Mean Absolute Testing Error = $1/1 * (600) = 600$
Supervised learning

- **Notation**
  - Features: \( x \) (input variables)
  - Targets: \( y \) (output variables)
  - Predictions: \( \hat{y} \)
  - Parameters: \( \theta \)

**Program ("Learner")**
- Characterized by some "parameters" \( \theta \)
- Procedure (using \( \theta \)) that outputs a prediction

**Training data (examples)**
- Features

**Feedback / Target values**

**Evaluation of the model (measure error)**

**Error** = Distance between \( y \) and \( \hat{y} \)

**Learning algorithm**
- Change \( \theta \)
- Improve performance
Overview

- Regression Problem Definition and parameters.
- Prediction using $\Theta$ as parameters
- Measure the error
- Finding good parameters $\Theta$ (direct minimization problem)
- Non-Linear regression problem

(c) Alexander Ihler
Linear regression

- Define form of function $f(x)$ explicitly
- Find a good $f(x)$ within that family

\[ Y = 5 + 1.5X_1 \]

New instance with $X_1=8$
Predicted value $= 17$

\[ \tilde{y} = \Theta_0 + \Theta_1 \times X_1 \]
return $\tilde{y}$

$\tilde{y}$ = Predicted target value (Black line)

(c) Alexander Ihler
More dimensions?

\[ \hat{y}(x) = \theta \cdot x^T \]

\[ \theta = [\theta_0 \ \theta_1 \ \theta_2] \]

\[ x = [1 \ x_1 \ x_2] \]

(c) Alexander Ihler
Notation

\[ \hat{y}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots \]

Define “feature” \( x_0 = 1 \) (constant)

Then

\[ \hat{y}(x) = \theta^T x \]

\[ \theta = [\theta_0, \ldots, \theta_n] \]

\[ x = [1, x_1, \ldots, x_n] \]

\( \tilde{Y} \) is a plane in \( n+1 \) dimension space

\( n = \text{the number of features in dataset} \)
Overview

- Regression Problem Definition and parameters.
- Prediction using $\Theta$ as parameters
- Measure the error
- Finding good parameters $\Theta$ (direct minimization problem)
- Non-Linear regression problem
Supervised learning

- Notation
  - Features $x$ (input variables)
  - Targets $y$ (output variables)
  - Predictions $\hat{y}$
  - Parameters $\theta$

Program ("Learner")
Characterized by some "parameters" $\theta$
Procedure (using $\theta$) that outputs a prediction

Error = Distance between $y$ and $\hat{y}$

Learning algorithm
Change $\theta$
Improve performance

Training data (examples)
Features
Feedback / Target values
Evaluation of the model (measure error)
Measuring error

Red points = Real target values
Black line = $\tilde{y}$ (predicted value)
$\tilde{y} = \Theta_0 + \Theta_1 \cdot X$
Blue lines = Error (Difference between real value $y$ and predicted value $\tilde{y}$)

$$y - \hat{y}(x) = (y - \theta \cdot x^T)$$
Mean Squared Error

- How can we quantify the error?

\[ \text{MSE}, \ J(\theta) = \frac{1}{m} \sum_{j} (y^{(j)} - \hat{y}(x^{(j)}))^2 \]

\( m = \) number of instance of data

\[ = \frac{1}{m} \sum_{j} (y^{(j)} - \theta \cdot x^{(j)T})^2 \]

- \( Y = \) Real target value in dataset,
- \( \hat{y} = \) Predicted target value by \( \Theta \cdot X \)

- Training Error: \( m = \) the number of training instances,
- Testing Error: Using a partition of Training error to check predicted values. \( m = \) the number of testing instances,
MSE cost function

\[ J(\theta) = \frac{1}{m} \sum_{j} (y^{(j)} - \hat{y}(x^{(j)}))^2 \]

\[ = \frac{1}{m} \sum_{j} (y^{(j)} - \theta \cdot x^{(j)T})^2 \]

- Rewrite using matrix form

\[ \theta = [\theta_0, \ldots, \theta_n] \]

\[ y = \begin{bmatrix} y^{(1)} & \ldots & y^{(m)} \end{bmatrix}^T \]

\[ X = \begin{bmatrix} x_0^{(1)} & \ldots & x_n^{(1)} \\ \vdots & \ddots & \vdots \\ x_0^{(m)} & \ldots & x_n^{(m)} \end{bmatrix} \]

\[ J(\theta) = \frac{1}{m} (y^T - \theta X^T) \cdot (y^T - \theta X^T)^T \]

(Matlab)

\[ >> e = y' - \theta \cdot X'; \quad J = e' \cdot e / m; \]

(c) Alexander Ihler
Visualizing the error function

The plane is the value of $J$, not the plane fitted to output values.

$J$ is error function.

Dimensions are $\Theta_0$ and $\Theta_1$ instead of $X_1$ and $X_2$
Output is $J$ instead of $y$ as target value

Representation of $J$ in 2D space.
Inner red circles has less value of $J$
Outer red circles has higher value of $J$
Overview

- Regression Problem Definition and parameters.
- Prediction using $\Theta$ as parameters
- Measure the error
- Finding good parameters $\Theta$ (direct minimization problem)
- Non-Linear regression problem
Supervised learning

- **Notation**
  - Features $x$
  - Targets $y$
  - Predictions $\hat{y}$
  - Parameters $\theta$

**Program ("Learner")**

Characterized by some "parameters" $\theta$

Procedure (using $\theta$) that outputs a prediction

**Learning algorithm**

Change $\theta$

Improve performance

**Training data (examples)**

- Features
- Feedback / Target values

**Evaluation of the model (measure error)**
Finding good parameters

- Want to find parameters which minimize our error…

- Think of a cost “surface”: error residual for that $\theta$…

$$\hat{\theta} = \arg \min_{\theta} J(\theta)$$
MSE Minimum \((m \leq n+1)\)

- Consider a simple problem
  - One feature, two data points
  - Two unknowns and two equations:

\[
y^{(1)} = \theta_0 + \theta_1 x^{(1)} \\
y^{(2)} = \theta_0 + \theta_1 x^{(2)}
\]

\(n + 1 = 1 + 1 = 2\)

\(m = 2\)

- Can solve this system directly:

\[
y^T = \theta X^T \quad \Rightarrow \quad \hat{\theta} = y^T (X^T)^{-1}
\]

Theta gives a line or plane that exactly fit to all target values.
SSE Minimum \((m > n+1)\)

- Most of the time, \(m > n\)
  - There may be no linear function that hits all the data exactly
  - Minimum of a function has gradient equal to zero (gradient is a horizontal line.)

\[
\nabla J(\theta) = \left[ y^T - \theta X^T \right] \cdot X = 0
\]

- Reordering, we have

\[
y^T X - \theta X^T \cdot X = 0
\]

\[
y^T X = \theta X^T \cdot X
\]

\[
\theta = y^T X (X^T X)^{-1}
\]

Just need to know how to compute parameters.

(c) Alexander Ihler
Effects of Mean Square Error choice

**outlier data**: An outlier is an observation that lies an abnormal distance from other value.

- $16^2$ cost for this one datum
- Heavy penalty for large errors
- Distract line from other points.

(c) Alexander Ihler
Absolute error

\[ \ell_1(\theta) = \sum_j |y^{(j)} - \hat{y}(x^{(j)})| \]

\[ = \sum_j |y - \theta \cdot x^T| \]
Error functions for regression

(Mean Square Error)
\[ \ell_2 : (y - \hat{y})^2 \]

(Mean Absolute Error)
\[ \ell_1 : |y - \hat{y}| \]

Something else entirely…

(???)
\[ c - \log(\exp(-(y - \hat{y})^2) + c) \]

“Arbitrary” Error functions can’t be solved in closed form…
So as alternative way, use gradient descent

(c) Alexander Ihler
Overview

• Regression Problem Definition and parameters.
• Prediction using $\Theta$ as parameters
• Measure the error
• Finding good parameters $\Theta$ (direct minimization problem)
• Non-Linear regression problem
Nonlinear functions

- Single feature $x$, predict target $y$:

$$D = \{(x^{(j)}, y^{(j)})\}$$

$$\hat{y}(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$$

Add features:

$$D = \{([x^{(j)}, (x^{(j)})^2, (x^{(j)})^3], y^{(j)})\}$$

$$\hat{y}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

Linear regression in new features

- Sometimes useful to think of “feature transform”

$$\Phi(x) = \begin{bmatrix} 1, x, x^2, x^3, \ldots \end{bmatrix}$$

$$\hat{y}(x) = \theta \cdot \Phi(x)$$

- Convert a non-linear function to linear function and then solve it.
Higher-order polynomials

- Are more features better?
- “Nested” hypotheses
  - 2nd order more general than 1st,
  - 3rd order ““ than 2nd, ...
- Fits the observed data better
Test data

- After training the model
- Go out and get more data from the world
  - New observations \((x,y)\)
- How well does our model perform?

(c) Alexander Ihler
Training versus test error

- Plot MSE as a function of model complexity
  - Polynomial order
- Decreases
  - More complex function fits training data better
- What about new data?
- 0th to 2nd order
  - Error decreases
  - Underfitting
- Higher order
  - Error increases
  - Overfitting

(c) Alexander Ihler
Summary

• Regression Problem Definition
  – Vehicle Price estimation
• Prediction using \( \Theta: \hat{y}(x) = \theta x^T \)
• Measure the error: difference between \( y \) and \( \hat{y} \)
  – e.g. Absolute error, MSE
• direct minimization problem
  – Two cases \( m \leq n+1 \) and \( m > n+1 \)
• Non-Linear regression problem
  – Finding best \( n^{th} \) order polynomial function for each problem (not overfitting and not under fitting)