10-701
Machine Learning

Regression
Where we are

- Inputs to Density Estimator result in probability prediction.
- Inputs to Classifier result in category prediction.
- Inputs to Regressor predict real numbers.

Today
Choosing a restaurant

- In everyday life we need to make decisions by taking into account lots of factors.
- The question is what weight we put on each of these factors (how important are they with respect to the others).
- Assume we would like to build a recommender system for ranking potential restaurants based on an individuals’ preferences.
- If we have many observations we may be able to recover the weights.

<table>
<thead>
<tr>
<th>Reviews (out of 5 stars)</th>
<th>$</th>
<th>Distance</th>
<th>Cuisine (out of 10)</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>30</td>
<td>21</td>
<td>7</td>
<td>8.5</td>
</tr>
<tr>
<td>2</td>
<td>15</td>
<td>12</td>
<td>8</td>
<td>7.8</td>
</tr>
<tr>
<td>5</td>
<td>27</td>
<td>53</td>
<td>9</td>
<td>6.7</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>5</td>
<td>6</td>
<td>5.4</td>
</tr>
</tbody>
</table>
Linear regression

• Given an input $x$ we would like to compute an output $y$

• For example:
  - Predict height from age
  - Predict Google’s price from Yahoo’s price
  - Predict distance from wall using sensor readings

Note that now $Y$ can be continuous
Linear regression

- Given an input $x$ we would like to compute an output $y$.
- In linear regression we assume that $y$ and $x$ are related with the following equation:
  \[ y = wx + \varepsilon \]
  where $w$ is a parameter and $\varepsilon$ represents measurement or other noise.

What we are trying to predict

Observed values
Our goal is to estimate \( w \) from a training data of \( \langle x_i, y_i \rangle \) pairs.

One way to find such relationship is to minimize the least squares error:

\[
\arg \min_{w} \sum_{i} (y_i - wx_i)^2
\]

Several other approaches can be used as well.

So why least squares?

- minimizes squared distance between measurements and predicted line
- has a nice probabilistic interpretation
- easy to compute

If the noise is Gaussian with mean 0 then least squares is also the maximum likelihood estimate of \( w \).
Solving linear regression using least squares minimization

• You should be familiar with this by now …

• We just take the derivative w.r.t. to \( w \) and set to 0:

\[
\frac{\partial}{\partial w} \sum_i (y_i - wx_i)^2 = 2 \sum_i -x_i(y_i - wx_i) \Rightarrow \\
2 \sum_i x_i(y_i - wx_i) = 0 \Rightarrow \\
\sum_i x_i y_i = \sum_i wx_i^2 \Rightarrow \\
w = \frac{\sum_i x_i y_i}{\sum_i x_i^2}
\]
Regression example

- Generated: $w=2$
- Recovered: $w=2.03$
- Noise: std=1
Regression example

- Generated: \( w=2 \)
- Recovered: \( w=2.05 \)
- Noise: std=2

![Graph showing linear regression with generated and recovered weights and noise level.](image)
Regression example

- Generated: $w=2$
- Recovered: $w=2.08$
- Noise: std=4
Bias term

- So far we assumed that the line passes through the origin.
- What if the line does not?
- No problem, simply change the model to:
  \[ y = w_0 + w_1 x + \varepsilon \]

- Can use least squares to determine \( w_0 \), \( w_1 \):

\[
    w_0 = \frac{\sum y_i - w_1 x_i}{n} \\
    w_1 = \frac{\sum x_i (y_i - w_0)}{\sum x_i^2}
\]
Bias term

- So far we assumed that the line passes through the origin.
- What if the line does not?
- No problem, simply change the model to
  \[ y = w_0 + w_1 x + \epsilon \]
- Can use least squares to determine \( w_0, w_1 \)

\[
    w_0 = \frac{\sum_{i} y_i - w_1 x_i}{n}
\]

\[
    w_1 = \frac{\sum_{i} x_i (y_i - w_0)}{\sum_{i} x_i^2}
\]

Just a second, we will soon give a simpler solution.
Multivariate regression

• What if we have several inputs?
  - Stock prices for Yahoo, Microsoft and Ebay for the Google prediction task
• This becomes a multivariate linear regression problem
• Again, its easy to model:

\[ y = w_0 + w_1x_1 + \ldots + w_kx_k + \varepsilon \]

- Google’s stock price
- Microsoft’s stock price
- Yahoo’s stock price
Multivariate regression

• What if we have several inputs?
  - Stock prices for Yahoo, Microsoft and Ebay for the Google prediction task
  
  This becomes a multivariate regression problem

• Not all functions can be approximated using the input values directly

• Again, its easy to model:

  \[ y = w_0 + w_1x_1 + \ldots + w_kx_k + \varepsilon \]
y=10+3x_1^2-2x_2^2+\varepsilon

In some cases we would like to use polynomial or other terms based on the input data, are these still linear regression problems?

Yes. As long as the coefficients are linear the equation is still a linear regression problem!
Non-Linear basis function

- So far we only used the observed values
- However, linear regression can be applied in the same way to functions of these values
- As long as these functions can be directly computed from the observed values the parameters are still linear in the data and the problem remains a linear regression problem

\[ y = w_0 + w_1 x_1^2 + \ldots + w_k x_k^2 + \epsilon \]
Non-Linear basis function

- What type of functions can we use?
- A few common examples:

- Polynomial: \( \phi_j(x) = x^j \) for \( j=0 \) … \( n \)

- Gaussian: \( \phi_j(x) = \frac{(x - \mu_j)}{2\sigma_j^2} \)

- Sigmoid: \( \phi_j(x) = \frac{1}{1 + \exp(-s_j x)} \)

Any function of the input values can be used. The solution for the parameters of the regression remains the same.
General linear regression problem

• Using our new notations for the basis function linear regression can be written as
  \[ y = \sum_{j=0}^{n} w_j \phi_j(x) \]

• Where \( \phi_j(x) \) can be either \( x_j \) for multivariate regression or one of the non linear basis we defined

• Once again we can use ‘least squares’ to find the optimal solution.
Our goal is to minimize the following loss function:

$$J(w) = \sum_{i} (y^i - \sum_{j} w_j \phi_j(x^i))^2$$

Moving to vector notations we get:

$$J(w) = \sum_{i} (y^i - w^T \phi(x^i))^2$$

We take the derivative w.r.t \( w \)

$$\frac{\partial}{\partial w} \sum_{i} (y^i - w^T \phi(x^i))^2 = 2 \sum_{i} (y^i - w^T \phi(x^i))\phi(x^i)^T$$

Equating to 0 we get

$$2 \sum_{i} (y^i - w^T \phi(x^i))\phi(x^i)^T = 0 \implies$$

$$\sum_{i} y^i \phi(x^i)^T = w^T \left[ \sum_{i} \phi(x^i)\phi(x^i)^T \right]$$
LMS for general linear regression problem

We take the derivative w.r.t $\mathbf{w}$

$$
\frac{\partial}{\partial \mathbf{w}} \sum_i (y^i - \mathbf{w}^T \phi(x^i))^2 = 2 \sum_i (y^i - \mathbf{w}^T \phi(x^i)) \phi(x^i)^T
$$

Equating to 0 we get

$$
2 \sum_i (y^i - \mathbf{w}^T \phi(x^i)) \phi(x^i)^T = 0 \implies 
\sum y^i \phi(x^i)^T = \mathbf{w}^T \left[ \sum_i \phi(x^i) \phi(x^i)^T \right]
$$

Define:

$$
\Phi = \begin{pmatrix}
\phi_0(x^1) & \phi_1(x^1) & \cdots & \phi_k(x^1) \\
\phi_0(x^2) & \phi_1(x^2) & \cdots & \phi_k(x^2) \\
\vdots & \vdots & \ddots & \vdots \\
\phi_0(x^n) & \phi_1(x^n) & \cdots & \phi_k(x^n)
\end{pmatrix}
$$

Then deriving $\mathbf{w}$ we get:

$$
\mathbf{w} = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{y}
$$
LMS for general linear regression problem

\[ J(w) = \sum_{i} (y^i - w^T \phi(x^i))^2 \]

Deriving \( w \) we get:

\[ w = \Phi^T \Phi^{-1} \Phi^T y \]

This solution is also known as ‘psuedo inverse’
Example: Polynomial regression

degree = 1, CV = 0.6

degree = 3, CV = 1.5

degree = 5, CV = 6.0

degree = 7, CV = 15.6
A probabilistic interpretation

Our least squares minimization solution can also be motivated by a probabilistic interpretation of the regression problem:

$$y = w^T \phi(x) + \varepsilon$$

The MLE for $w$ in this model is the same as the solution we derived for least squares criteria:

$$w = (\Phi^T \Phi)^{-1} \Phi^T y$$
Other types of linear regression

- Linear regression is a useful model for many problems
- However, the parameters we learn for this model are **global**; they are the same regardless of the value of the input $x$
- Extension to linear regression adjust their parameters based on the region of the input we are dealing with
Splines

• Instead of fitting one function for the entire region, fit a set of piecewise (usually cubic) polynomials satisfying continuity and smoothness constraints.

• Results in smooth and flexible functions without too many parameters

• Need to define the regions in advance (usually uniform)

\[ y = a_1 x^3 + b_1 x^2 + c_1 x + d_1 \]
\[ y = a_2 x^3 + b_2 x^2 + c_2 x + d_2 \]
\[ y = a_3 x^3 + b_3 x^2 + c_3 x + d_3 \]
LOCAL, KERNEL REGRESSION
Local Kernel Regression

• What is the temperature in the room?

\[ \hat{T} = \frac{1}{n} \sum_{i=1}^{n} Y_i \]

At location x?

\[ \hat{T}(x) = \frac{\sum_{i=1}^{n} Y_i 1_{||X_i-x|| \leq h}}{\sum_{i=1}^{n} 1_{||X_i-x|| \leq h}} \]

Average

“Local” Average
Local Average Regression

\[ \Rightarrow \hat{f}_n(X) = \hat{\beta} = \sum_{i=1}^{n} w_i Y_i \]

\[ \frac{1}{n_X^h} \sum_{i=1}^{n} Y_i 1_{|X - X_i| \leq h} \]

Sum of Ys in h ball around X

Recall: NN classifier with majority vote

Here we use Average instead
Nadaraya-Watson Kernel Regression

\[ \Rightarrow \hat{f}_n(X) = \hat{\beta} = \sum_{i=1}^{n} w_i Y_i \]

\[ = \frac{1}{n_X^n} \sum_{i=1}^{n} Y_i \ 1_{\mid X - X_i \mid \leq h} \]

Sum of Ys in h ball around X

\[ w_i(X) = \frac{K \left( \frac{X - X_i}{h} \right)}{\sum_{i=1}^{n} K \left( \frac{X - X_i}{h} \right)} \]

boxcar kernel:

\[ K \left( \frac{X - X_i}{h} \right) = 1_{\mid X - X_i \mid \leq h} \]
Local Kernel Regression

- Nonparametric estimator akin to kNN
- Nadaraya-Watson Kernel Estimator

\[ \hat{f}_n(X) = \sum_{i=1}^{n} w_i Y_i \]
\[ w_i(X) = \frac{K \left( \frac{X - X_i}{h} \right)}{\sum_{i=1}^{n} K \left( \frac{X - X_i}{h} \right)} \]

- Weight each training point based on distance to test point
- Boxcar kernel yields local average
Kernels

\[ K(x) \geq 0, \]

\[ \int K(x) \, dx = 1 \]

**boxcar kernel:**

\[ K(x) = \frac{1}{2} I(x), \]

**Gaussian kernel:**

\[ K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \]
Kernel Regression as Weighted Least Squares

\[
\min_{f} \frac{1}{n} \sum_{i=1}^{n} w_i (f(X_i) - Y_i)^2 \quad \text{subject to} \quad \frac{1}{n} \sum_{i=1}^{n} w_i = 1
\]

Weighted Least Squares

Weigh each training point based on distance to test point

\[
w_i(X) = \frac{K \left( \frac{X - X_i}{h} \right)}{\sum_{i=1}^{n} K \left( \frac{X - X_i}{h} \right)}
\]

\(K\) – Kernel
\(h\) – Bandwidth of kernel

Boxcar kernel:

\[K(x) = \frac{1}{2} I(x),\]

Gaussian kernel:

\[K(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}\]
Kernel Regression as Weighted Least Squares

set \( f(X_i) = \beta \) (a constant)

\[
\min_{\beta} \sum_{i=1}^{n} w_i(\beta - Y_i)^2 \quad \text{constant}
\]

\[
w_i(X) = \frac{K\left(\frac{X-X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X-X_i}{h}\right)}
\]

\[
\frac{\partial J(\beta)}{\partial \beta} = 2 \sum_{i=1}^{n} w_i(\beta - Y_i) = 0 \quad \text{Notice that} \quad \sum_{i=1}^{n} w_i = 1
\]

\[
\Rightarrow \hat{f}_n(X) = \hat{\beta} = \sum_{i=1}^{n} w_i Y_i
\]
Choice of kernel bandwidth $h$

- Too small
- $h=1$
- $h=10$ Too small
- $h=50$ Just right
- $h=200$ Too large

Image Source: Larry's book – All of Nonparametric Statistics
Choice of Bandwidth

Large Bandwidth – average more data points, reduce noise  
(Lower variance)

Small Bandwidth – less smoothing, more accurate fit  
(Lower bias)

Bias – Variance tradeoff

Should depend on n, # training data  
(determines variance)

Should depend on smoothness of function  
(determines bias)
Spatially adaptive regression

If function smoothness varies spatially, we want to allow bandwidth $h$ to depend on $X$

Local polynomials, splines, wavelets, regression trees …
Local prediction

Histogram Classifier
Local Adaptive prediction

Let neighborhood size adapt to data – small neighborhoods near decision boundary (small bias), large neighborhoods elsewhere (small variance)

Decision Tree Classifier

Majority vote at each leaf
Important points

• Linear regression
  - basic model
  - as a function of the input
• Solving linear regression
• Error in linear regression
• Advanced regression models
Splines

- The polynomials are not independent
- For cubic splines we require that they agree in the border point on the value, the values of the first derivative and the value of the second derivative
- How many free parameters do we actually have?

\[
\begin{align*}
y &= a_1 x^3 + b_1 x^2 + c_1 x + d_1 \\ y &= a_2 x^3 + b_2 x^2 + c_2 x + d_2 \\ y &= a_3 x^3 + b_3 x^2 + c_3 x + d_3
\end{align*}
\]
Splines

- Splines sometimes contain additional requirements for the first and last polynomial (for example, having them start at 0)

- Once Splines are fitted to the data they can be used to predict new values in the same way as regular linear regression, though they are limited to the support regions for which they have been defined

- Note the range of functions that can be displayed with relatively small number of polynomials (in the example I am using 5)