COMPSCI 389
Introduction to Machine Learning

Days: Tu/Th.  Time: 2:30 – 3:45  Building: Morrill 2  Room: 222

Topic 5.6: Linear Regression and the Optimization Perspective
Prof. Philip S. Thomas (pthomas@cs.umass.edu)
Review: Regression

• **$X$: Input** (also called features, attributes, covariates, or predictors)
  - Typically, $X$ is a vector, array, or list of numbers or strings.

• **$Y$: Output** (also called labels or targets)
  - In regression, $Y$ is a real number.

• An input-output pair is $(X, Y)$.

• Let $n$, called the **data set size**, be the number of input-output pairs in the data set.

• Let $(X_i, Y_i)$ denote the $i^{th}$ input-output pair.

• The complete data set is 
  $$ (X_i, Y_i)_{i=1}^n = (X_1, Y_1), (X_2, Y_2), ..., (X_n, Y_n) $$.
Review: Nearest Neighbor (Variants)

- Given a query input $x_{\text{query}}$, find the $k$ nearest points in the training data.
- Return a weighted average of their labels.
  - $k = 1$ is nearest neighbor
  - $k > 1$ with all $w_i$ equal is k-nearest neighbor
  - $k > 1$ with not all $w_i$ equal is weighted k-nearest neighbor
- These algorithms don’t pre-process the training data much.
  - They can build data structures like KD-Trees for efficiency.
Linear Regression

• Search for the **line** that is a best fit to the data.

• Different performance measures correspond to different ways of measuring the quality of a fit.

• Sample mean squared error, or the sum of the squared errors is particularly common:

\[
\overline{\text{MSE}}_n: \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \text{ and } \text{SSE}: \sum_{i=1}^{n} (y_i - \hat{y}_i)^2
\]

• Although not identical, the line that minimizes one also minimizes the other.

• Using sample MSE, this method is called “least squares linear regression.”
Linear Regression: What is a line?

\[ y = mx + b \]

- **Prediction**, \( \hat{y}_i \)
- **Slope**, \( m \)
- **Input**, \( x_i \)
- **y-intercept**, \( b \)
- "weights," or "parameters" , \( w = (w_1, w_2) \)

\[ \hat{y} = w_1 x_i + w_2 \]
A model is a mechanism that maps input data to predictions.

ML algorithms take data sets as input and produce models as output.

### Models (Review)

- **Data Set**

<table>
<thead>
<tr>
<th>pinyin</th>
<th>biology</th>
<th>history</th>
<th>English</th>
<th>geography</th>
<th>literature</th>
<th>Portuguese</th>
<th>maths</th>
<th>chemistry</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>422.60</td>
<td>491.56</td>
<td>439.93</td>
<td>707.54</td>
<td>550.65</td>
<td>557.09</td>
<td>711.97</td>
<td>721.31</td>
<td>509.60</td>
</tr>
<tr>
<td>1</td>
<td>538.00</td>
<td>490.56</td>
<td>469.93</td>
<td>729.06</td>
<td>532.28</td>
<td>647.33</td>
<td>727.61</td>
<td>779.18</td>
<td>489.64</td>
</tr>
<tr>
<td>2</td>
<td>455.16</td>
<td>440.00</td>
<td>570.86</td>
<td>417.54</td>
<td>453.53</td>
<td>425.67</td>
<td>475.63</td>
<td>476.91</td>
<td>407.15</td>
</tr>
<tr>
<td>3</td>
<td>756.91</td>
<td>476.63</td>
<td>592.63</td>
<td>532.42</td>
<td>521.80</td>
<td>550.41</td>
<td>703.76</td>
<td>598.36</td>
<td>2.3333</td>
</tr>
<tr>
<td>4</td>
<td>504.54</td>
<td>459.56</td>
<td>637.43</td>
<td>609.06</td>
<td>570.40</td>
<td>615.98</td>
<td>572.50</td>
<td>571.25</td>
<td>529.04</td>
</tr>
</tbody>
</table>

- **ML Algorithm**

A query can be one or more feature vectors.

- **Model**

Predictions are given for each feature vector in the query.

<table>
<thead>
<tr>
<th>Query</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>798.75</td>
<td>3.75000</td>
</tr>
<tr>
<td>817.58</td>
<td>2.50000</td>
</tr>
<tr>
<td>731.98</td>
<td></td>
</tr>
<tr>
<td>648.42</td>
<td></td>
</tr>
<tr>
<td>527.66</td>
<td></td>
</tr>
<tr>
<td>443.82</td>
<td></td>
</tr>
<tr>
<td>545.88</td>
<td></td>
</tr>
<tr>
<td>624.18</td>
<td></td>
</tr>
<tr>
<td>751.30</td>
<td></td>
</tr>
<tr>
<td>648.67</td>
<td></td>
</tr>
<tr>
<td>662.05</td>
<td></td>
</tr>
<tr>
<td>773.15</td>
<td></td>
</tr>
<tr>
<td>835.25</td>
<td></td>
</tr>
</tbody>
</table>

| 583.41 |          |
| 395.46 |          |
| 509.80 |          |
Parametric Model

• A model “parameterized” by a weight vector $w$.
• Different settings of $w$ result in different predictions.
• Let $\hat{y} = f_w(x)$
  • 1-dimensional linear case:
    $$f_w(x) = w_1 x + w_2$$
**Linear Regression: Hyperplanes**

• What if we have more than one input feature?
• Let $x_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,d})$ be a $d$-dimensional input.
  • We include the $i$ subscript to make it clear that $1,2,\ldots$ aren’t referencing different input vectors, but different elements of one input vector.
• We use a hyperplane:
  $$f_w(x_i) = w_1 x_{i,1} + w_2 x_{i,2} + \ldots + w_d x_{i,d} + w_{d+1}.$$
Linear Regression (cont.)

\[ f_w(x_i) = w_1 x_{i,1} + w_2 x_{i,2} + \ldots + w_d x_{i,d} + w_{d+1}. \]

- **Thought**: We don’t want to have to keep remembering a special “intercept” term.

- **Idea**: Drop the intercept term!
  - If you want to include the intercept term, add one more feature to your data set, \( x_{d+1} = 1 \).
  - If \( d \) is the dimension of the input with this additional feature, we then have:
    \[ f_w(x_i) = w_1 x_{i,1} + w_2 x_{i,2} + \ldots + w_d x_{i,d} \]
  - We can write this as:
    \[ f_w(x_i) = \sum_{j=1}^{d} w_j x_{i,j}. \]
  - This is called a **dot product** and can be written as \( w \cdot x_i \) or \( w^T x_i \).
Linear Regression (cont.)

\[ \hat{y}_i = f_w(x_i) = \sum_{j=1}^{d} w_j x_{i,j} \]

• How many weights (parameters) does the model have?
  • \( d \), the dimension of any one input vector \( x_i \).
  • Not \( n \), the number of training data points.
Linear Regression: Optimization Perspective

• Given a parametric model $f_w$ of any form how can we find the weights $w$ that result in the “best fit”?

• Let $L$ be a function called a loss function.
  • It takes as input a model (or model weights $w$)
  • It also takes as input data $D$
  • It produces as output a real-number describing how bad of a fit the model is to the provided data.

• The evaluation metrics we have discussed can be viewed as loss functions. For example, the sample MSE loss function is:

$$L(w, D) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \frac{1}{n} \sum_{i=1}^{n} (y_i - f_w(x_i))^2$$

• We phrase this as an optimization problem:
  $$\arg\min_w L(w, D)$$

For the sample MSE loss function, this can be any parametric model, not just a linear one!
Linear Regression: Optimization Perspective

\[ \text{argmin}_w \, L(w, D) \]

• **Recall**: \( \text{argmin} \) returns the \( w \) that achieves the minimum value of \( L(w, D) \), not the minimum value of \( L(w, D) \) itself.

• This expression describes a massive range of ML methods.
  • Supervised, unsupervised, (batch/offline) RL
  • Deep neural networks
  • Large language models and generative AI

• Different problem settings and algorithms in ML correspond to:
  • Different loss functions
  • Different parametric models.
  • Different algorithms for approximating the best weight vector \( w \).
Least Squares Linear Regression (cont.)

• Find the weights $w$ that minimize

$$L(w, D) = \frac{1}{n} \sum_{i=1}^{n} (y_i - f_w(x_i))^2$$

Number of training data points

Dimension of each input vector
(number of features per row)

$$L(w, D) = \frac{1}{n} \sum_{i=1}^{n} \left( y_i - \sum_{j=1}^{d} w_j x_{i,j} \right)^2$$
Linear Regression: Least Squares Solvers

• How should one solve this problem?

\[ \text{argmin}_w \frac{1}{n} \sum_{i=1}^{n} \left( y_i - \sum_{j=1}^{d} w_j x_{i,j} \right)^2 \]

• Answer: “Least squares solvers”
  • Algorithms based on concepts from linear algebra.
  • Extremely effective for solving problems of precisely this form.
  • Beyond the scope of this class.
  • **Only useful for this exact problem.**
    • Not effective when using other parametric models (e.g., not linear)
    • Not effective when using other loss functions / performance metrics.
Linear Regression

• How do we solve this problem?

$$\arg\min_w \frac{1}{n} \sum_{i=1}^{n} \left( y_i - \sum_{j=1}^{d} w_j x_{i,j} \right)^2$$

• We will study a different approach for solving this problem.
• It is less efficient.
• It applies to almost all loss functions and parametric models of interest.

• Method: Gradient descent.
  • Soon we will discuss gradient descent.
  • For now, assume we have some way of finding the $$\arg\min_w L(w, D)$$.
Least Squares Linear Regression
Linear Regression vs Weighted k-NN for GPA Prediction

Weighted KNN Model:
- Average MSE: 0.571
- MSE Standard Error: 0.004

Linear Regression Model:
- Average MSE: 0.582
- MSE Standard Error: 0.004

Very simple method achieves nearly the same performance as a tuned-version of weighted k-NN!

Soon, we will consider more complex parametric models that can be even more effective.
Linear Regression Limitation

• What if the relationship between the inputs and outputs is not linear (or affine)?
  • Linear: $A_1 x_{i,1} + A_2 x_{i,2} + \cdots + A_n x_{i,n}$
  • Affine: $A_1 x_{i,1} + A_2 x_{i,2} + \cdots + A_n x_{i,n} + b$
    • Equivalent to linear with an additional feature $x_{i,n+1} = 1$.

• **Idea:** Have parametric functions that can represent more than linear functions!
Linear Parametric Model ≠ Linear Functions

- **Linear parametric functions** are functions $f_w(x_i)$ that are **linear functions** of the weights $w$.
- They need not be linear functions of the input $x_i$.

Input $x_i$ → Feature generator $\phi$ → Feature 1: $\phi_1(x_i)$ → Feature 2: $\phi_2(x_i)$ → Feature m: $\phi_m(x_i)$ → Linear Regression: $f_w(x_i) = w_1\phi_1(x_i) + w_2\phi_2(x_i) + \cdots$

**Note:** Each feature can depend on more than one element of $x_i$. So, this is $\phi_1(x_i)$ not $\phi_1(x_{i,1})$.

**Note:** This is equivalent to pre-processing the data, converting $x_i$ (length $d$) into $\phi(x_i)$ (length $m$).

Note: The input $x_i$ is a vector – an array of values.
Linear Parametric Model \( \neq \) Linear Functions

• **Linear parametric functions** are functions \( f_w(x_i) \) that are *linear* functions of the weights \( w \).

• They need not be linear functions of the input \( x_i \).

• That is, a linear parametric model has the form:

\[
f_w(x_i) = \sum_{j=1}^{m} w_j \phi_j(x_i),
\]

where \( \phi \) takes the input vector \( x_i \) as input and produces a vector of \( m \) features as output. That is, \( \phi_j(x_i) \) is the \( j \)th feature output by \( \phi \).

• \( \phi \) is called the **basis function**, **feature generator**, or **feature mapping function**.
Linear Parametric Model

\[ f_w(x_i) = \sum_{j=1}^{m} w_j \phi_j(x_i) \]

- Polynomial basis
  - If \( x_i \in \mathbb{R} \) then \( \phi_j(x_i) = x_i^{j-1} \) so that:
    \[ \phi(x_i) = [1, x_i, x_i^2, x_i^3, \ldots, x_i^{m-1}] \]
  - Here \( m - 1 \) is the **degree** or **order** of the polynomial basis.
  - \( f_w(x_i) = w_1 + w_2 x_i + w_3 x_i^2 + w_4 x_i^3 + \cdots + w_m x_i^{m-1} \)
  - We are fitting a polynomial to the data!
  - This is a non-linear function of the input \( x_i \)
  - This can represent *any* smooth function (if \( m \) is big enough).
  - This is a linear function of \( w \).
Linear Parametric Models (cont.)

• What does it mean for a function $g(x, y)$ to be **linear** with respect to an input, $x$?
  • The slope is constant as $x$ changes.
  • The derivative with respect to $x$ is a constant (does not vary with $x$)

• Is $g(x, y) = x^2 y^2$ linear with respect to (w.r.t.) $x$?
  • $\frac{\partial g(x,y)}{\partial x} = 2xy^2$, which changes with $x$, so no.

• Is $g(x, y) = x \sin(y)$ linear w.r.t. $x$?
  • $\frac{\partial g(x,y)}{\partial x} = \sin(y)$, which does not change with $x$, so yes!

• Is $f_w(x_i) = \sum_{j=1}^{m} w_j \phi_j(x_i)$ linear w.r.t. $w$?
  • $\frac{\partial f_w(x_i)}{\partial w_j} = \phi_j(x_i)$, for all $j$, which does not change with $w$, so yes!
Linear Parametric Models (cont.)

• Is \( f_w(x_i) = \sum_{j=1}^{m} w_j \phi_j(x_i) \) linear w.r.t. \( x \)?
  • \( \frac{\partial f_w(x_i)}{\partial x_{i,j}} = w_j \frac{\partial \phi_j(x_i)}{\partial x_{i,j}} \), for all \( j \).
  • If \( \phi \) is linear w.r.t. \( x \) then yes, otherwise no.

• Is \( f_w(x_i) = w_1 w_2 x_{i,1}^2 \) linear w.r.t. \( w \)?
  • \( \frac{\partial f_w(x_i)}{\partial w_1} = w_2 x_{i,1}^2 \)
  • **No.** It is linear w.r.t. \( w_1 \) but not linear w.r.t. \( w \).
  • Linear w.r.t. \( w \) means that the derivative w.r.t. \( w \) (a vector) does not depend on \( w \) (a vector).
    • Note: The derivative w.r.t. \( w \) is
      \[
      \left[ \frac{\partial f_w(x_i)}{\partial w_1}, \frac{\partial f_w(x_i)}{\partial w_2} \right]^T
      \]
      This T means “transpose,” which just means that this should be viewed as a column not a row (the elements stacked vertically rather than horizontally). This isn’t important for this course.
Linear Parametric Models
Linear Parametric Model vs Linear Regression vs Weighted k-NN for GPA Prediction (20-fold cross-validation)

- **Weighted KNN Model:**
  - Average MSE: 0.571
  - MSE Standard Error: 0.004

- **Linear Regression Model:**
  - Average MSE: 0.582
  - MSE Standard Error: 0.004

- **Polynomial Regression Model (Degree 4):**
  - Average MSE: 0.576
  - MSE Standard Error: 0.004

Recall k-NN results:

<table>
<thead>
<tr>
<th>k</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.152084</td>
</tr>
<tr>
<td>1</td>
<td>0.853430</td>
</tr>
<tr>
<td>2</td>
<td>0.764468</td>
</tr>
<tr>
<td>3</td>
<td>0.688330</td>
</tr>
<tr>
<td>4</td>
<td>0.631001</td>
</tr>
<tr>
<td>5</td>
<td>0.579404</td>
</tr>
<tr>
<td>6</td>
<td>0.581676</td>
</tr>
<tr>
<td>7</td>
<td>0.600544</td>
</tr>
</tbody>
</table>

A simple linear model outperforms k-NN (not quite a well-tuned weighted k-NN)!
Linear Parametric Models

• Pros:
  • Relatively simple.
  • Can represent any smooth function (given the right / enough features).
  • Can use hand-crafted features.
  • Quite efficient to solve for optimal $w$.
    • Can still use least squares solvers – need not use gradient descent.
  • Extremely fast to generate predictions for new inputs
    • Compute features, take the dot-product with the weights (take the weighted sum)

• Cons:
  • Can be hard to find good features.
  • People often think linear parametric models can only represent lines, and so they think negatively of them.
Parametric vs Nonparametric

• ML algorithms are often categorized into **parametric** and **nonparametric**.
  • In general:
    • Parametric methods use parameterized functions with weights $w$.
    • Nonparametric methods store the training data or statistics of the training data.
  • More precisely
    • Parametric:
      • Have a fixed number of weights $w$.
      • Tend to make specific assumptions about the form of the function.
    • Nonparametric:
      • Do not make explicit assumptions about the form of the function.
      • Number of values stored tends to vary with the amount of training data (e.g., storing data).
  • There is some debate about whether some methods are parametric or nonparametric.
    • Linear regression and regression with linear parametric are canonical examples of parametric.
    • Nearest neighbor algorithms are canonical examples of nonparametric.
How does the polynomial basis, $\phi$, work if $x$ is multidimensional (an array rather than a number?)

- Multivariate polynomial on inputs $x, y$:
  $$a + bx + cy + dxy + ex^2 + fy^2 + gxy^2 + hx^2y + ix^3 + \cdots$$

- Multivariate polynomial on input $x_{i,1}, x_{i,2}$:
  $$w_1 + w_2x_{i,1} + w_3x_{i,2} + w_4x_{i,1}x_{i,2} + w_5x_{i,1}^2 + w_6x_{i,2}^2 + w_7x_{i,1}x_{i,2}^2 + w_8x_{i,1}^2x_{i,2} + w_9x_{i,1}^3 + \cdots$$

The expression above is $f_w(x_i)$ for a linear parametric model using the multivariate polynomial basis.

- Notice that some $\phi_j(x_i)$ terms depend on more than one element of $x_i$!
  - This term is $w_8\phi_8(x_i)$
Fourier Basis

• Each $\phi_j$ is a cosine function with a different period.
  • Can optionally include both sine and cosine functions.
• Univariate:
  • $\phi_j(x_i) = \cos(j\pi x)$
• Approximation of a step function (from Wikipedia “Fourier series” page)
Fourier Basis (Multivariate)

Figure 3: A few example Fourier basis functions defined over two state variables. Lighter colors indicate a value closer to 1, darker colors indicate a value closer to $-1$. 
Feature Engineering

• In some cases, you can hand-craft features
• Examples:
  • Average STEM score
  • Average non-STEM score
• Question: Why might these not be good features?
• Answer: They do not change the functions that can be represented!
  • A weight of $w_j$ on STEM score equates to $\frac{w_j}{9}$ being added to the weights on each of the STEM exams.
• Effective features are not linear combinations of existing features.