CSE/STAT 416
Introduction + Regression

Vinitra Swamy
University of Washington
June 22, 2020

Slides and materials for this course courtesy of Hunter Schafer and Emily Fox.
Machine Learning is changing the world.
machine learning
Search term

Interest over time

United States  2004 - present  All categories  Web Search
It’s Everywhere!

Disruptive companies differentiated by INTELLIGENT APPLICATIONS using Machine Learning.
It’s Everywhere...
It’s terrifying that both of these things are true at the same time in this world:

• computers drive cars around

• the state of the art test to check that you’re not a computer is whether you can successful identify stop signs in pictures
Generically (and vaguely)

Machine Learning is the study of algorithms that improve their performance at some task with experience.
This course is broken up into 5 main case studies to explore ML in various contexts/applications.

1. Regression  
   - Predicting housing prices
2. Classification  
   - Positive/Negative reviews (Sentiment analysis)
3. Document Retrieval + Clustering  
   - Find similar news articles
4. Recommender Systems  
   - Given past purchases, what do we recommend to you?
5. Deep Learning  
   - Recognizing objects in images
Course Topics

Models
- Linear regression, regularized approaches (ridge, LASSO)
- Linear classifiers: logistic regression
- Non-linear models: decision trees
- Nearest neighbors, clustering
- Recommender systems
- Deep learning

Algorithms
- Gradient descent
- Boosting
- K-means

Concepts
- Point estimation, MLE
- Loss functions, bias-variance tradeoff, cross-validation
- Sparsity, overfitting, model selection
- Decision boundaries
ML Course Landscape

CSE 446
- CSE majors
- Very technical course

STAT 435
- STAT majors
- Very technical course

CSE/STAT 416
- Everyone else!
  - This is a super broad audience!
- Give everyone a strong foundational understanding of ML
  - More breadth than other courses, a little less depth
Our Motto

Everyone should be able to learn machine learning, so our job is to make tough concepts intuitive and applicable.

This means...

- Minimize pre-requisite knowledge
- Focus on important ideas, avoid getting bogged down by math
- Maximize ability to develop and deploy
- Use pre-written libraries to do many tasks
- Learn concepts in case studies

Does not mean course isn’t fast paced! There are a lot of concepts to cover!
Course
Logistics
Who am I?

- Vinitra Swamy
  - Lecturer, Paul G. Allen School for Computer Science & Engineering (CSE)
  - AI Software Engineer at Microsoft
    - AI Framework Interoperability
    - Open Neural Network eXchange (ONNX)
- Office Hours
  - Time: 4:00 pm - 5:00 pm, Fridays, or by appointment
  - Location: Zoom
- Contact
  - Personal Matters: vinitra@cs.washington.edu
  - Course Content + Logistics: Piazza
Who are the TAs?

**Anne Wagner**  
amwag@uw

**Jack Wu**  
hongjun@uw

**Ben Evans**  
bevans97@cs

**Svetoslav Kolev**  
swetko@cs
- We happen to not record attendance in lectures and section, but attending these sessions is expected.
- Participation component (5% of your grade)
- Zoom Recordings for Lecture (on Canvas)
Lectures
Introduced to material for the first time. Mixed with activities and demos to give you a chance to learn by doing.

Sections
Practice material covered in 1 in a context where a TA can help you. The emphasis is still on you learning by doing.

Homework
With the scaffolding from 1 and 2, you are probably now capable to tackle the homework. These will be complex and challenging, but you’ll continue to learn by doing.

No where near mastery yet!
Assessment

- **Weekly Homework Assignments**
  - **Weight**: 65%
  - **Number**: Approximately 8
  - Each Assignment has two parts that contribute to your grade separately:
    - Programming (50%)
    - Conceptual (15%)

- **Participation**
  - **Weight**: 5%
  - Answering PollEverywhere questions during lecture or up to 24 hours after (for students in different timezones)

- **Final Exam**
  - **Weight**: 30%
  - **Date**: Wednesday, August 19
  - **Location**: Online
Homework Logistics

- **Late Days**
  - 4 Free Late Days for the whole quarter.
  - Can use up to 2 Late Days on any assignment.
  - Each Late Day used after the 4 Free Late Days results in a -10% on that assignment.

- **Collaboration**
  - You are encouraged to discuss assignments and concepts at a high level:
    - If you have code in front of you in your discussion, probably NOT high level
    - Discuss process, not answers
  - All code and answers submitted must be your own

- **Turn In**
  - Everything completed and turned in on Gradescope
  - Multiple “assignments” on Gradescope per assignment
Case Study 1

Regression:
Housing Prices
Fitting Data

**Goal:** Predict how much my house is worth

Have data from my neighborhood

\[
\begin{align*}
(x_1, y_1) &= (2318 \text{ sq.ft.}, \$315k) \\
(x_2, y_2) &= (1985 \text{ sq.ft.}, \$295k) \\
(x_3, y_3) &= (2861 \text{ sq.ft.}, \$370k) \\
&\vdots &\vdots \\
(x_n, y_n) &= (2055 \text{ sq.ft.}, \$320k)
\end{align*}
\]

**Assumption:**

There is a relationship between \( y \in \mathbb{R} \) and \( x \in \mathbb{R}^d \)

\[ y \approx f(x) \]

\( x \) is the **input data.** Can potentially have many inputs

\( y \) is the **outcome/response/target/label/dependent variable**
A **model** is how we assume the world works.

Regression model:

\[ y_i = f(x_i) + \varepsilon_i \quad \text{with} \quad E[\varepsilon_i] = 0 \]

“Essentially, all models are wrong, but some are useful.”
- George Box, 1987
Predictor

We don’t know \( f \)! We need to learn it from the data!

Use machine learning to learn a predictor \( \hat{f} \) from the data

For a given input \( x \), predict: \( \hat{y} = \hat{f}(x) \)
Linear Regression

LINEAR REGRESSION EVERYWHERE
Training Data → Feature extraction → ML model → ML algorithm → Quality metric
Linear Regression Model

Assume the data is produced by a line.

\[ y_i = w_0 + w_1 x_i + \epsilon_i \]

\( w_0, w_1 \) are the parameters of our model that need to be learned

- \( w_0 \) is the intercept (\$/the land with no house)
- \( w_1 \) is the slope (\$/increase per increase in sq. ft)

Learn estimates of these parameters \( \hat{w}_0, \hat{w}_1 \) and use them to predict new value for any input \( x \! \\

\[ \hat{y} = \hat{w}_0 + \hat{w}_1 x \]
Basic Idea

Try a bunch of different lines and see which one is best!

What does best even mean here?
Training Data

Feature extraction

ML model

ML algorithm

Quality metric

linear regression
Define a “cost” for a particular setting of parameters

- Low cost $\rightarrow$ Better fit
- Find settings that minimize the cost
- For regression, we will use the error as the cost.
  - Low error = Low cost = Better predictor (hopefully)

Note: There are other ways we can define cost which will result in different “best” predictors. We will see what these other costs are and how they affect the result.
How to define error? **Residual sum of squares (RSS)**

\[
\text{RSS}(w_0, w_1) = (y_1 - \hat{y}_1)^2 + (y_2 - \hat{y}_2)^2 + \ldots + (y_n - \hat{y}_n)^2
\]

\[
= \sum_{i=1}^{n} (y_i - \hat{y}_i)^2
\]

\[
= \sum_{i=1}^{n} (y_i - (w_0 + w_1 x_i))^2
\]

\[
\text{MSE} = \frac{1}{n} \text{RSS}(w_0, w_1)
\]
Goal: Get you actively participating in your learning
- Think (1 min): Think about the question on your own
  - Why is your answer choice correct?
  - Why are the other answer choices wrong?
- Answer (1 min): Submit your answers on PollEverywhere
- Share (1 min): We discuss the conclusions as a class

During the Answer stage, you will respond to the question via a Poll Everywhere poll
- Participation grade (5% of your grade)
  - Note: you’re not penalized for answering incorrectly
  - After lecture, these will be open for 24 hours
Sort the following lines by their RSS on the data, from smallest to largest. (estimate, don’t actually compute)
Minimizing Cost

RSS is a function with inputs $w_0, w_1$, different settings have different RSS for a dataset

$$\hat{w}_0, \hat{w}_1 = \min_{w_0, w_1} RSS(w_0, w_1)$$

$$= \min_{w_0, w_1} \sum_{i=1}^{n} (y_i - (w_0 + w_1x_i)^2)$$

Unfortunately, we can’t try it out on all possible settings 😞
Gradient Descent

Instead of computing all possible points to find the minimum, just start at one point and “roll” down the hill. Use the gradient (slope) to determine which direction is down.

Start at some (random) point $w^{(0)}$ when $t = 0$

While we haven’t converged:

$$w^{(t+1)} = w^{(t)} - \eta \nabla RSS(w^{(t)})$$
Disclaimer: This is for your comedic entertainment. Please don’t actually erase outliers 😊
This data doesn’t look exactly linear, why are we fitting a line instead of some higher-degree polynomial?

We can! We just have to use a slightly different model!

\[ y_i = w_0 + w_1 x_i + w_2 x_i^2 + w_3 x_i^3 + \epsilon_i \]
Polynomial Regression

Model

\[ y_i = w_0 + w_1 x_i + w_2 x_i + \ldots + w_p x_i^p + \epsilon_i \]

Just like linear regression, but uses more features!

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>( w_0 )</td>
</tr>
<tr>
<td>1</td>
<td>( x )</td>
<td>( w_1 )</td>
</tr>
<tr>
<td>2</td>
<td>( x^2 )</td>
<td>( w_2 )</td>
</tr>
<tr>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>0</td>
<td>( x^p )</td>
<td>( w_p )</td>
</tr>
</tbody>
</table>
Polynomial Regression

How to decide what the right degree? Come back Wednesday!
**Features** are the values we select or compute from the data inputs to put into our model. **Feature extraction** is the process of turning the data into features.

**Model**

\[
y_i = w_0 h_0(x_i) + w_1 h_1(x_i) + \ldots + w_D h_D(x_i) + \epsilon_i
\]

\[
= \sum_{j=0}^{D} w_j h_j(x_i) + \epsilon_i
\]

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(h_0(x))</td>
<td>(w_0)</td>
</tr>
<tr>
<td>1</td>
<td>(h_1(x))</td>
<td>(w_1)</td>
</tr>
<tr>
<td>2</td>
<td>(h_2(x))</td>
<td>(w_2)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>D</td>
<td>(h_D(x))</td>
<td>(w_D)</td>
</tr>
</tbody>
</table>
Adding Other Inputs

Generally we are given a data table of values we might look at that include more than one value per house.

- Each row is a single house.
- Each column (except Value) is a data input.

<table>
<thead>
<tr>
<th>sq. ft.</th>
<th># bathrooms</th>
<th>owner’s age</th>
<th>...</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1400</td>
<td>3</td>
<td>47</td>
<td>...</td>
<td>70,800</td>
</tr>
<tr>
<td>700</td>
<td>3</td>
<td>19</td>
<td>...</td>
<td>65,000</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1250</td>
<td>2</td>
<td>36</td>
<td>...</td>
<td>100,000</td>
</tr>
</tbody>
</table>
Adding more features to the model allows for more complex relationships to be learned:

$$y_i = w_0 + w_1 (\text{sq. ft.}) + w_2 (\# \text{bathrooms}) + \epsilon_i$$

Coefficients tell us the rate of change if all other features are constant.
**Notation**

**Important:** Distinction is the difference between a data input and a feature.

- Data inputs are columns of the raw data
- Features are the values (possibly transformed) for the model (done after our feature extraction $h(x)$)

Data Input: $x_i = (x_i[1], x_i[2], ..., x_i[d])$

Output: $y_i$

- $x_i$ is the $i^{th}$ row
- $x_i[j]$ is the $i^{th}$ row’s $j^{th}$ data input
- $h_j(x_i)$ is the $j^{th}$ feature of the $i^{th}$ row
You can use anything you want as features and include as many of them as you want!

Generally, more features means a more complex model. This might not always be a good thing!

Choosing good features is a bit of an art.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1 (constant)</td>
<td>$w_0$</td>
</tr>
<tr>
<td>1</td>
<td>$h_1(x) \approx x[1] = \text{sq. ft.}$</td>
<td>$w_1$</td>
</tr>
<tr>
<td>2</td>
<td>$h_2(x) \ldots x[2] = # \text{ bath}$</td>
<td>$w_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>D</td>
<td>$h_D(x) \ldots \log(x[7]) \times x[2]$</td>
<td>$w_D$</td>
</tr>
</tbody>
</table>
Linear Regression Recap

Dataset
\[
\{(x_i, y_i)\}_{i=1}^n \text{ where } x \in \mathbb{R}^d, y \in \mathbb{R}
\]

Feature Extraction
\[
h(x): \mathbb{R}^d \rightarrow \mathbb{R}^D
\]
\[
h(x) = (h_0(x), h_1(x), \ldots, h_D(x))
\]

Regression Model
\[
y = f(x) + \epsilon
\]
\[
= \sum_{j=0}^{D} w_j h_j(x) + \epsilon
\]
\[
= \mathbf{w}^T h(x) + \epsilon
\]

Quality Metric
\[
\text{RSS}(\mathbf{w}) = \sum_{i=1}^{n} (y_i - \mathbf{w}^T x_i)^2
\]

Predictor
\[
\hat{\mathbf{w}} = \min_{\mathbf{w}} \text{RSS}(\mathbf{w})
\]

ML Algorithm
Optimized using Gradient Descent

Prediction
\[
\hat{y} = \hat{\mathbf{w}}^T h(x)
\]