CSE/STAT 416
Introduction + Regression

Hunter Schafer
University of Washington
March 29, 2021
Machine Learning is changing the world.
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
What is Machine Learning?

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?

- Hunter Schafer
  - Assistant Teaching Professor
  - Paul G. Allen School for Computer Science & Engineering (CSE)

- Office Hours
  - Time: 10:00 am - 12:00 pm, Tuesdays
  - Location: Zoom/Discord

- Contact
  - Course Content + Logistics: EdStem
  - Personal Matters: hschafer@cs.washington.edu
Who are the TAs?

Andrey Risukhin
he/him
risuka@uw

Gang Cheng
he/him
gang@uw

Leona Kazi
she/her
lkazi@uw

Rahul Biswas
he/him
rbiswas1@uw

Santino Iannone
he/him
iannos@uw

Svet Kolev
swetko@uw
Lectures
Introduced to material for the first time. Mixed with activities and demos to give you a chance to learn by doing. No where near mastery yet!

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

Concept Checks
Test your understanding of the last concept.

Learning Reflections
Recap broader context of the past week.

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.

Lectures
Introduced to material for the first time. Mixed with activities and demos to give you a chance to learn by doing.

No where near mastery yet!
- We happen to not record attendance in lectures and section, but attending these sessions is expected
- Panopto for Lecture (on Canvas)
Assessment

- **Weekly Homework Assignments**
  - **Weight**: 65%
  - **Number**: Approximately 9
  - Each Assignment has two parts that contribute to your grade separately:
    - Programming (50%)
    - Conceptual (15%)

- **Checkpoints**
  - **Weight**: 10%
  - **Number**: Approximately 20 (each lecture, drop 3)

- **Learning Reflections**
  - **Weight**: 10%
  - **Number**: Approximately 10 (each week, drop 1)

- **Final Exam**
  - **Weight**: 15%
  - **Date**: Monday 6/8 – Wednesday 6/9
Homework Logistics

- **Late Days**
  - 6 Free Late Days for the whole quarter.
  - Can use up to 2 Late Days on any assignment.
  - Each Late Day used after the 6 Free Late Days results in a -10% on that assignment.
  - Learning reflections and checkpoints can be turned in up to a week later for 50% credit.

- **Collaboration**
  - You are encouraged to discuss assignments and concepts at a high level.
    - If you are reading off parts of your solution, it’s likely not high level.
    - Discuss process, not answers!
  - All code and answers submitted must be your own.

- **Turn In**
  - Concept portions and Learning reflections are turned in on Gradescope.
  - Everything else (Programming portion and checkpoints) are turned in on EdStem.
Getting Help

The best place to get **asynchronous help** is EdStem. You can post questions (publicly or privately) to get help from peers or members of the course staff.
- You’re encouraged to respond with your ideas to other posts!

The best place to get **synchronous help** is office hours or to form a study group.
- Office hours will be run on Discord! See course website for more info.
- Will try to help you meet peers this quarter to form study groups. More on this next time!
On your phone / laptop

If you could only have one pet, would you rather have a dog or cat?
Case Study 1

Regression:
Housing Prices
Goal: Predict how much my house is worth

Have data from my neighborhood

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

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:

“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: $\hat{y} = \hat{f}(x)$

Small error on an example, means we had a good fit for that point
ML Pipeline

Training Data → Feature extraction → ML model → Quality metric

ML algorithm

Training Data (x, y) → Feature extraction (x) → ML model (ŷ) → Quality metric (f)

ML Pipeline

Training Data

Feature extraction

ML model

Quality metric

ML algorithm

y

ŷ

f
Linear Regression
Training Data \rightarrow \text{Feature extraction} \rightarrow x \rightarrow \text{ML model} \rightarrow \hat{y} \rightarrow \text{ML algorithm} \rightarrow \hat{f} \rightarrow \text{Quality metric} \rightarrow y
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 (\$ of 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 \( x \)!

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

Why don’t we add \( \epsilon \)?
Basic Idea

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

What does best even mean here?
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.
Residual Sum of Squares (RSS)

How to define error? Residual sum of squares (RSS)
Goal: Get you actively participating in your learning

Typical Activity
- Question is posed
- **Think** (1 min): Think about the question on your own
- **Pair** (2 min): Talk with your neighbor to discuss question
  - If you arrive at different conclusions, discuss your logic and figure out why you differ!
  - If you arrived at the same conclusion, discuss why the other answers might be wrong!
- **Share** (1 min): We discuss the conclusions as a class

During each of the **Think** and **Pair** stages, you will respond to the question via a Poll Everywhere poll
- Not worth any points, just here to help you learn!
Sort the following lines by their RSS on the data, from smallest to largest. (estimate, don’t actually compute)
Sort the following lines by their RSS on the data, from smallest to largest. (estimate, don’t actually compute)
Training Data → Feature extraction → ML model → ML algorithm → Quality metric → Training Data

Symbols: $y$, $\hat{y}$, $x$, $\hat{f}$
Minimizing Cost

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

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

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

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

\[ w^{(t+1)} = w^{(t)} - \eta \nabla RSS(w^{(t)}) \]
Brain Break
Higher Order Features

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
\[ y_i = w_0 + w_1 x_i + w_2 x_i^2 + \ldots + w_p x_i^p + \epsilon_i \]

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 (constant)</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>( p )</td>
<td>( x^p )</td>
<td>( w_p )</td>
</tr>
</tbody>
</table>

How do you train it? Gradient descent (with more parameters)
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) ) often 1 (constant)</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
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>$h_1(x) \ldots x[1] = \text{sq. ft.}$</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>$h_D(x) \ldots \text{like log}(x[7]) * x[2]$</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
\[ h(x): \mathbb{R}^d \rightarrow \mathbb{R}^D \]
\[ h(x) = (h_0(x), h_1(x), ..., h_D(x)) \]

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

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

Predictor
\[ \hat{w} = \min_w RSS(w) \]

ML Algorithm
Optimized using Gradient Descent

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