Machine Learning is changing the world.
It’s Everywhere!

Disruptive companies differentiated by INTELLIGENT APPLICATIONS using Machine Learning
It’s Everywhere…
It’sterrifying 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

12:26 AM - 13 May 2018
5,644 Retweets 12,727 Likes
What is Machine Learning?

Generically (and vaguely)

Machine Learning (ML) is the study of algorithms that improve their performance at some task with experience.

Tom Mitchell (1998): a computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.
Taxonomy of Machine Learning (Based on tasks)

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning
1. **Supervised Learning**
   - Training data is labeled, where inputs are paired with correct outputs
   - Infers a mapping function from the inputs to outputs
   - *Examples:* image classification, stock price predictions

2. **Unsupervised Learning**
   - Analyze and cluster unlabeled datasets
   - Discover patterns or data categorization without the need for human intervention
   - *Examples:* DNA clustering, anomaly detection

3. **Reinforcement Learning**
   - *Not covered in this class (you can learn this in CSE 415 / 473 (Introduction to Artificial Intelligence))*
   - Agents learn the optimal behaviors to obtain maximum reward through interactions with the environment and observations of how they responds.
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 / underfitting, model selection
- Decision boundaries
CSE 446
- CSE majors
- Very technically demanding course *(which Pemi has taught as a TA for 4 quarters)*

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
Level of Course

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
- Allow you to understand the ML concepts in an intuitive way.

*Fun fact: ML is a very practical field, and intuitive thinking plays an important role for ML practitioners. There has not been a fully rigorous proof for the accuracy of neural networks, one of the main architecture of modern ML, but people still use intuitively understand how powerful they are.*

- Focus on important ideas, avoid getting bogged down by math
- Exposed to Python, libraries and infrastructure to program ML problems
- 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?

- **Background**
  - UW CSE graduate
  - Former Teaching Assistant and Content Development Contributor for CSE 311 (Discrete Math), CSE 312 (Probability & Stats for CS), CSE 446/546 (Machine Learning) for 7 quarters
  - Former NLP Researcher at UWNLP
  - Software Engineer at Facebook (starting June 2022)
  - Disability & Accessibility Advocate

- **Contact**
  - Course Content + Logistics: [EdStem](#)
  - Personal Matters: peming@cs.washington.edu

*Pemi is not available for the first week. Hunter will take place as the substitute lecturer and Amal will help answer questions about logistics and stuff.*
Who are the TAs?

Amal Nanavati  
Head TA  
he/they  
amaln@cs

Sahil Verma  
he/him  
vsahil@cs

Wuwei Zhang  
she/her  
wz86@cs

Jack Zhou  
he/him  
zhoujack@uw

Rahul Biswas  
he/him  
rbiswas1@uw

Jerry Wei  
he/him  
zwei5@uw

Pranav Kamath  
he/him  
pranavpk@cs
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.

Homeworks
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.

Checkpoints
Test your understanding of the last concept

Learning Reflections
Recap broader context of the past week.

Extra credit opportunities
I believe that people deserve second chance if they don’t perform well on some homeworks or exams. There are many opportunities for you to earn extra credit, which will be announced later in the quarter.
- **It's not required** to attend in lectures and section, but attending these sessions is **highly encouraged**
- Panopto for live lecture recordings and weekly section recordings by Rahul
Assessment

- **Weekly Homework Assignments (35%)**
  - **Number:** ~ 9 (drop 1)
  - You can submit in **pairs** (meaning, you can work individually or find a partner to work with)
  - Each Assignment has two parts that contribute to your grade
    - Programming (50%) – autograded, you receive scores right away
    - Conceptual (50%) – 1/2 autograded (won’t receive scores until after deadlines), 1/2 manually graded

- **Checkpoints (10%)**
  - Designed to be doable (30 mins) if you follow each previous lecture
  - **Number:** Approximately 20 (each lecture, drop 3)

- **Learning Reflections (10%)**
  - Full credit is expected (unless you don’t show any effort)
  - You have to submit them **individually**.
  - **Number:** Approximately 10 (each week, drop 1)

- **Midterm + Final Exams (45%)**
  - 15% for midterm and 20% for final
  - Open handwritten notes **allowed**. No limit on length.
  - **Dates:** TBD
Extra credit opportunities

I don’t believe in faulting people if they don’t perform well in homeworks or exams occasionally. These are a number of ways to earn extra credit:

▪ Respond to other students’ questions on EdStem as actively as possible. I won’t give you a definite answer on how active you should be to be eligible. Just do your best and contribute to class discussions within your capacity.

▪ Write original, thoughtful analysis of interesting machine learning topics on EdStem, such as linking external sources and providing your opinions. I will endorse the well-written ones.

▪ Submit answers to extra credit problems from homeworks.

▪ Towards the end of the quarter, I will invite some researchers or ML practitioners to give talks on specific applications of ML. You can attend those and write a reflection on each event. More information on this later.
Homework Logistics

- **Late Days**
  - 6 Free Late Days for the whole quarter.
  - Can use up to 2 Late Days on homework assignments only.
  - 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 with anyone not in your group.
    - 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 yours or your homework partner’s.

- **Turn In**
  - Homework submissions (both coding + conceptual) and Learning reflections are turned in on Gradescope.
  - Checkpoints are turned in on EdStem.
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 Zoom or on-campus (in CSE rooms). We will try to provide a balanced mix of virtual and in-person OHs to allow people to enjoy the benefits of both.

- We provide an unmoderated **Discord** channel for students in the class. Staff members won’t monitor Discord, so please be civil and do not engage in any academic misconduct.

- We will try to help you meet peers this quarter to form study groups. More on this later!
On your phone / laptop
What are the factors of determining the price of a house?
**Goal:** Predict how much my house is worth

Have data from my neighborhood

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

**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:

\[ y_i \approx f(x_i) \]

Essentially all models are wrong, but some are useful.

- George Box, 1987
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
Regression

- Is a supervised learning algorithm

- Given a set of training data examples $X^{(1)}, X^{(2)}, \ldots, X^{(n)}$ associated to with set of continuous values $y^{(1)}, y^{(2)}, \ldots, y^{(n)}$ we want to build a predictor function that learns how to map $x^{(i)}$ to $y^{(i)}$.

- Each example $x^{(i)}$ can have from 1 to many features $X_1^{(i)}, X_2^{(i)}, \ldots, X_d^{(i)}$. We want to establish the relationships between different features of our data in order to make a good prediction.

- A typical regression problem is house price prediction.
Linear Regression
Assume we have a simple model with **one feature**, where we establish a linear relationship between **the area of a house** \( i \) and **its price**:

\[
y^{(i)} = w_1 X^{(i)} + w_0
\]

- \( w_0 \) is the intercept / **bias**, representing the starting price of a house
- \( w_1 \) is the slope / **weight** associated with feature "area of a house"

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

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

\( \hat{f}(x) \) is the **unknown function**

\( w, b \) are the **parameters** of our model that need to be learned
Basic Idea

Try a bunch of different lines and see which one is best!
What does best even mean here?

Area of a house

Price
Training Data → Feature extraction → ML model → Optimization algorithm → Quality metric

\( y \) → \( x \) → \( \hat{y} \) → \( \hat{f} \)
Cost / Loss of predictor

Mean-Squared Error (MSE)

Define a cost / loss for a particular set of parameters
- Low cost / loss → Better fit
- Find settings that minimize the cost
- For regression, we will use MSE (mean-squared errors) as the default loss function.
  - Low error = Low loss = Better predictor (hopefully)

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

\( y \): actual value
\( \hat{y} \): predicted value

Note: There are a variety of loss functions in Machine Learning, such as Mean Absolute Error, Huber Loss for the regression task. However, MSE is usually the to-go loss function because of its easy implementation and has nicer mathematical properties (continuously differentiable, a statistic for Gaussian distribution ...
Goal: Get you actively participating in your learning

Typical Activity
- Question is posted
- **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 MSE (mean-squared errors) on the data, from smallest to largest. (estimate, don’t actually compute)
Sort the following lines by their MSE on the data, from smallest to largest. (estimate, don’t actually compute)
Training Data → Feature extraction → ML model → \(\hat{y}\) → Optimization algorithm → Quality metric → \(y\)
Minimizing Cost

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

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

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

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) weights $\mathbf{w}$
While we haven’t converged:

$$w \leftarrow \alpha \nabla L(w)$$

- $\alpha$: learning rate
- $\nabla L(w)$: the gradients of loss function $L$ on a set of weights $\mathbf{w}$
Brain Break
Training Data → Feature extraction → ML model → Optimization algorithm → Quality metric → Training Data

\( y \rightarrow x \rightarrow \hat{y} \rightarrow \hat{f} \rightarrow y \)
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!

\[ y = w_0 + w_1 x + w_2 x^2 + w_3 x^3 \]
Polynomial Regression

Model

\[ y = w_0 + w_1 x + w_2 x^2 + \ldots + w_d x^d \]

To capture a non-linear relationship in the model, we can transform the original features into 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^d )</td>
<td>( w_d )</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 reduce the number of features in a dataset by creating new features from the existing ones (and then discarding the original features).

Model

\[ y = w_0 h_0(x) + w_1 h_1(x) + \ldots + w_D h_D(x) \]

\[ = \sum_{j=0}^{D} w_j h_j(x) \]

<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 Features

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

- Each row is a data point.
- Each column (except Value) represents a feature.
- The last column (Price) contains the actual output values.

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

\[ y = w_0 + w_1(sq. ft.) + w_2(# bathrooms) \]

Coefficients tell us the rate of change if all other features are constant
Features

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] = sq. ft.</td>
<td>$w_1$</td>
</tr>
<tr>
<td>2</td>
<td>$h_2(x)$ \ldots x[2] = # bath</td>
<td>$w_2$</td>
</tr>
<tr>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>D</td>
<td>$h_D(x)$ \ldots like log(x[7]) \ast x[2]</td>
<td>$w_D$</td>
</tr>
</tbody>
</table>
Term recap

- **Supervised learning**: The machine learning task of learning a function that maps an input to an output based on example input-output pairs.

- **Regression**: A supervised learning task where the outputs are continuous values.

- **Feature**: An attribute that we’re selecting for our model. Can come from the original dataset, or through some transformations *(feature extraction)*.

- **Parameter**: The weight or bias associated with a feature. The goal of machine learning is to adjust the weights to optimize the loss functions on training data.

- **Loss function**: A function that computes the distance between the predicted output from a machine learning model and the actual output.

- **Machine learning model**: An algorithm that combs through an amount of data to find patterns, make predictions, or generate insights.

- **Optimization algorithm**: An algorithm used to minimize the loss during training. The most common one is *Gradient Descent*. 
**Linear Regression Recap**

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

**Feature Extraction**
\[
h(x): \mathbb{R}^d \to \mathbb{R}^d
\]
\[
h(x) = (h_0(x), h_1(x), ..., h_D(x))
\]

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

**Quality Metric / Loss function**
\[
MSE = \frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2
\]

**Predictor**
\[
\hat{w} = \arg\min_w MSE(w)
\]

**Optimization Algorithm**
Optimized using Gradient Descent

**Prediction**
\[
\hat{y} = \hat{w}^T h(x)
\]
Deadlines & Other Logistics

- Homework 0: (weight: near 0%)
  - Aim to test your readiness for the course
  - Coding portion on EdStem, at the Assessments tab
    (no submission required)
  - Conceptual portion due on Gradescope (due Friday 11:59 pm)
- Checkpoint 1: **Due Wednesday 2 pm**
- Special quiz about Syllabus (on EdStem): **Due Friday 11:59 pm**
- Reflection 1: **Due Friday 11:59 pm**
- Homework 1 released on Friday
- We will provide opportunities for you to find your homework partner later this week if you’re interested.
- Please check EdStem regularly for the latest updates on course logistics.