CPSC 340:
Machine Learning and Data Mining

Gradient Descent
Fall 2018
Last Time: Change of Basis

- Last time we discussed change of basis:
  - E.g., polynomial basis:
    
    \[
    \begin{bmatrix}
    x_1 \\
    x_2 \\
    \vdots \\
    x_n
    \end{bmatrix}
    \]
    
    Replace \( X \) with \( Z = \begin{bmatrix} 1 & x_1 & (x_1)^2 & \cdots & (x_1)^p \\
    1 & x_2 & (x_2)^2 & \cdots & (x_2)^p \\
    \vdots & \vdots & \vdots & \ddots & \vdots \\
    1 & x_n & (x_n)^2 & \cdots & (x_n)^p
    \end{bmatrix} \)
    
    - You can fit non-linear models with linear regression.
    
    \[
    \hat{y}_i = \mathbf{v}^T z_i = w_0 + w_1 x_i + w_2 x_i^2 + w_3 x_i^3 + \cdots + w_p x_i^p
    \]
    
    - Just treat ‘\( Z \)’ as your data, then fit linear model.
Optimization Terminology

• When we **minimize** or **maximize** a function we call it “optimization”.
  – In least squares, we want to solve the “optimization problem”:
    $$\min_{w \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^{n} (w^T x_i - y_i)^2$$
  – The function being optimized is called the “**objective**”.
    • Also sometimes called “loss” or “cost”, but these have different meanings in ML.
  – The set over which we search for an optimum is called the **domain**.

• Often, instead of the minimum objective value, you want a **minimizer**.
  • A set of **parameters** ‘w’ that achieves the minimum value.
Discrete vs. Continuous Optimization

• We have seen examples of **continuous optimization**:
  – Least squares:
    • Domain is the real-valued set of parameters ‘w’.
    • Objective is the sum of the squared training errors.

• We have seen examples of **discrete optimization**:  
  – Fitting decision stumps:
    • Domain is the finite set of unique rules.
    • Objective is the number of classification errors (or infogain).

• We have also seen a **mixture** of discrete and continuous:  
  – K-means: clusters are discrete and means are continuous.
Stationary/Critical Points

• A ‘w’ with $\nabla f(w) = 0$ is called a stationary point or critical point. The slope is zero so the tangent plane is “flat”.

Critical points
A ‘w’ with $\nabla f(w) = 0$ is called a stationary point or critical point.

- The slope is zero so the tangent plane is “flat”.

- If we’re minimizing, we would ideally like to find a global minimum.

- But for some problems the best we can do is find a stationary point where $\nabla f(w)=0$. 
Motivation: Large-Scale Least Squares

• Normal equations find ‘w’ with $\nabla f(w) = 0$ in $O(nd^2 + d^3)$ time.

$$ (\mathbf{X}^\top \mathbf{X})w = \mathbf{X}^\top y $$

- Very slow if ‘d’ is large.

• Alternative when ‘d’ is large is gradient descent methods.
  - Probably the most important class of algorithms in machine learning.
Gradient Descent for Finding a Local Minimum

• **Gradient descent** is an iterative optimization algorithm:
  – It starts with a “guess” $w^0$.
  – It uses the gradient $\nabla f(w^0)$ to generate a better guess $w^1$.
  – It uses the gradient $\nabla f(w^1)$ to generate a better guess $w^2$.
  – It uses the gradient $\nabla f(w^2)$ to generate a better guess $w^3$.
  ...
  – The limit of $w^t$ as ‘$t$’ goes to $\infty$ has $\nabla f(w^t) = 0$.

• It converges to the global optimum if ‘$f$’ is convex.
Gradient Descent for Finding a Local Minimum

- **Gradient descent** is based on a simple observation:
  - Give parameters ‘$w$’, the direction of largest decrease is $-\nabla f(w)$. 

![Diagram of gradient descent](image-url)
Gradient Descent for Finding a Local Minimum

- Gradient descent is based on a simple observation:
  - Give parameters ‘w’, the direction of largest decrease is \(-\nabla f(w)\).
Gradient Descent for Finding a Local Minimum

- **Gradient descent** is based on a simple observation:
  - Give parameters ‘w’, the direction of largest decrease is $-\nabla f(w)$. 

![Diagram of gradient descent](image)
Gradient Descent for Finding a Local Minimum

- Gradient descent is based on a simple observation:
  - Give parameters ‘w’, the direction of largest decrease is \(-\nabla f(w)\).
Gradient Descent for Finding a Local Minimum

- Gradient descent is based on a simple observation:
  - Give parameters ‘w’, the direction of largest decrease is $-\nabla f(w)$.
Gradient Descent for Finding a Local Minimum

– We start with some initial guess, $w^0$.
– Generate new guess by moving in the negative gradient direction:

$$w^{l+1} = w^l - \alpha^l \nabla f(w^l)$$

• This decreases ‘f’ if the “step size” $\alpha^0$ is small enough.
• Usually, we decrease $\alpha^0$ if it increases ‘f’ (see “findMin”).

– Repeat to successively refine the guess:

$$w^{l+1} = w^l - \alpha^l \nabla f(w^l) \quad \text{for } t = 1, 2, 3, ...$$

– Stop if not making progress or $\|\nabla f(w^t)\| \leq \varepsilon$

$\varepsilon$ is Some small scalar.

Approximate local minimum
Data Space vs. Parameter Space

• Usual regression plot is in the “x vs. y” data space (left):

• On the right is plot of the “intecerpt vs. slope” parameter space.
  – Points in parameter space correspond to models (* is least squares parameters).
Gradient Descent in Data Space vs. Parameter Space

• Gradient descent starts with an initial guess in parameter space:

– And each iteration tries to move guess closer to solution.
Gradient Descent in Data Space vs. Parameter Space

- Gradient descent starts with an initial guess in parameter space:

  - And each iteration tries to move guess closer to solution.
Gradient Descent in Data Space vs. Parameter Space

- Gradient descent starts with an initial guess in parameter space:

  > And each iteration tries to move guess closer to solution.
Gradient Descent in Data Space vs. Parameter Space

- Gradient descent starts with an initial guess in parameter space:

  - And each iteration tries to move guess closer to solution.
• Under weak conditions, algorithm converges to a ‘w’ with $\nabla f(w) = 0$.
  – ‘f’ is bounded below, $\nabla f$ can’t change arbitrarily fast, small-enough constant $\alpha^t$. 
Gradient Descent for Least Squares

• The least squares objective and gradient:

\[ f(w) = \frac{1}{2} \|Xw - y\|^2 \quad \nabla f(w) = X^T(Xw - y) \]

• Gradient descent iterations for least squares:

\[ w^{t+1} = w^t - \alpha^t X^T(Xw^t - y) \]

• Cost of gradient descent iteration is \( O(nd) \) (no need to form \( X^TX \)).
Normal Equations vs. Gradient Descent

• Least squares via normal equations vs. gradient descent:
  – Normal equations cost $O(nd^2 + d^3)$.
  – Gradient descent costs $O(ndt)$ to run for ‘t’ iterations.
    • Each of the ‘t’ iterations costs $O(nd)$.

  – Gradient descent can be faster when ‘d’ is very large:
    • If solution is “good enough” for a ‘t’ less than $\min(d, d^2/n)$.
    • CPSC 540: ‘t’ proportional to “condition number” of $X^TX$ (no direct ‘d’ dependence).

  – Normal equations only solve linear least squares problems.
    • Gradient descent solves many other problems.
Beyond Gradient Descent

• There are many variations on gradient descent.
  – Methods employing a “line search” to choose the step-size.
  – “Conjugate” gradient and “accelerated” gradient methods.
  – Newton’s method (which uses second derivatives).
  – Quasi-Newton and Hessian-free Newton methods.
  – Stochastic gradient (later in course).

• This course focuses on gradient descent and stochastic gradient:
  – They’re simple and give reasonable solutions to most ML problems.
  – But the above can be faster for some applications.
(pause)
Convex Functions

• Is finding a ‘w’ with $\nabla f(w) = 0$ good enough?
  – Yes, for convex functions.

• A function is convex if the area above the function is a convex set.
  – All values between any two points above function stay above function.
Convex Functions

- All ‘w’ with $\nabla f(w) = 0$ for convex functions are global minima.

Proof by contradiction:

Consider a local minimum.

If this is not global minimum, there must a smaller value.

By convexity we can move along line to global minimum and decrease objective.

- Normal equations find a global minimum because of convexity.
How do we know if a function is convex?

• Some useful tricks for showing a function is convex:
  – 1-variable, twice-differentiable function is convex iff \( f''(w) \geq 0 \) for all ‘\( w \)’.

  Consider \( f(w) = \frac{1}{2} aw^2 \) for \( a > 0 \). We have \( f'(w) = aw \) and \( f''(w) = a > 0 \) by assumption.

  Consider \( f(w) = e^w \). We have \( f'(w) = e^w \) and \( f''(w) = e^w > 0 \) by definition of exponential function.
How do we know if a function is convex?

• Some useful tricks for showing a function is convex:
  – 1-variable, twice-differentiable function is convex iff \( f''(w) \geq 0 \) for all ‘\( w \)’.
  – A convex function multiplied by non-negative constant is convex.

We showed that \( f(w) = e^w \) is convex, so \( f(w) = 10e^w \) is convex.
How do we know if a function is convex?

• Some useful tricks for showing a function is convex:
  – 1-variable, twice-differentiable function is convex iff \( f''(w) \geq 0 \) for all ‘\( w \)’.
  – A convex function multiplied by non-negative constant is convex.
  – Norms and squared norms are convex.
    \[
    \|w\|, \|w\|^2, \|w\|_1, \|w\|_\infty, \|w\|^2, \text{ and so on are all convex.}
    \]
How do we know if a function is convex?

• Some useful tricks for showing a function is convex:
  – 1-variable, twice-differentiable function is convex iff $f''(w) \geq 0$ for all ‘w’.
  – A convex function multiplied by non-negative constant is convex.
  – Norms and squared norms are convex.
  – The sum of convex functions is a convex function.

\[
f(w) = 10e^w + \frac{1}{2} \|w\|^2 \quad \text{is convex}
\]

From earlier constant norm squared
How do we know if a function is convex?

• Some useful tricks for showing a function is convex:
  – 1-variable, twice-differentiable function is convex iff $f''(w) \geq 0$ for all ‘w’.
  – A convex function multiplied by non-negative constant is convex.
  – Norms and squared norms are convex.
  – The sum of convex functions is a convex function.

$$f(w) = w^T x_i = w_1 x_{i1} + w_2 x_{i2} + \cdots + w_d x_{id}$$

Convex, convex, convex

Second derivative of each term is 0.
How do we know if a function is convex?

• Some useful tricks for showing a function is convex:
  – 1-variable, twice-differentiable function is convex iff \( f''(w) \geq 0 \) for all ‘w’.
  – A convex function multiplied by non-negative constant is convex.
  – Norms and squared norms are convex.
  – The sum of convex functions is a convex function.
  – The max of convex functions is a convex function.

\[
\hat{f}(w) = \max \left\{ 2w, w^2 \right\} \text{ is convex.}
\]
How do we know if a function is convex?

• Some useful tricks for showing a function is convex:
  – 1-variable, twice-differentiable function is convex iff \( f''(w) \geq 0 \) for all ‘w’.
  – A convex function multiplied by non-negative constant is convex.
  – Norms and squared norms are convex.
  – The sum of convex functions is a convex function.
  – The max of convex functions is a convex function.
  – Composition of a convex function and a linear function is convex.

\[
\text{If } f(w) = g(xw - y) \text{ then } \text{if } g \text{ is convex then } f \text{ is convex if g is convex.}
\]
How do we know if a function is convex?

• Some useful tricks for showing a function is convex:
  – 1-variable, twice-differentiable function is convex iff \( f''(w) \geq 0 \) for all ‘\( w \)’.
  – A convex function multiplied by non-negative constant is convex.
  – Norms and squared norms are convex.
  – The sum of convex functions is a convex function.
  – The max of convex functions is a convex function.
  – Composition of a convex function and a linear function is convex.

• But: not true that composition of convex with convex is convex:
  
  Even if ‘\( f \)’ is convex and ‘\( g \)’ is convex, \( f(g(w)) \) might not be convex.
  
  E.g. \( x^2 \) is convex and \( -\log(x) \) is convex but \( -\log(x^2) \) is not convex.
Example: Convexity of Linear Regression

• Consider linear regression objective with squared error:
  \[ f(w) = \|Xw - y\|^2 \]

• We can use that this is a convex function composed with linear:
  
  Let \( g(r) = \|r\|^3 \), which is convex because it's a squared norm.

  Then \( f(w) = g(Xw - y) \), which is convex because it's a convex function composed with a linear function.
Convexity in Higher Dimensions

• Twice-differentiable ‘d’-variable function is convex iff:
  – Eigenvalues of Hessian $\nabla^2 f(w)$ are non-negative for all ‘w’.

• True for least squares where $\nabla^2 f(w) = X^T X$ for all ‘w’.
  – It may not be obvious that this matrix has non-negative eigenvalues.

• Unfortunately, sometimes hard to show convexity this way.
  – Usually easier to just use some of the rules as we did on the last slide.
(pause)
Least Squares with Outliers

- Consider least squares problem with outliers in ‘y’:

  \[ x \leftarrow \text{“outlier” that doesn’t follow trend} \]

  \( x \) This is what we \underline{\text{want}} least squares to do.

http://setosa.io/ev/ordinary-least-squares-regression
Least Squares with Outliers

- Consider least squares problem with outliers in ‘y’:

\[ x \leftarrow \text{“outlier” that doesn’t follow trend} \]

- Least squares is very sensitive to outliers.
Least Squares with Outliers

• Squaring error shrinks small errors, and **magnifies large errors**: Least squares minimizes vertical distance squared.

• Outliers (large error) influence ‘w’ much more than other points.

http://students.brown.edu/seeing-theory/regression/index.html
Least Squares with Outliers

- Squaring error shrinks small errors, and **magnifies large errors**:

  - Outliers (large error) influence ‘w’ much more than other points.
    - Good if outlier means ‘plane crashes’, bad if it means ‘data entry error’.
Summary

• **Gradient descent** finds critical point of differentiable function.
  – Finds global optimum if function is convex.

• **Convex functions:**
  – Set of functions with property that $\nabla f(w) = 0$ implies ‘$w$’ is a global min.
  – Can (usually) be identified using a few simple rules.

• **Outliers in ‘$y$’** can cause problem for least squares.

• Next time:
  – Linear regression without the outlier sensitivity...
Constraints, Continuity, Smoothness

• Sometimes we need to optimize with *constraints*:
  – Later we’ll see “non-negative least squares”.

\[
\min_{w \geq 0} \frac{1}{2} \sum_{i=1}^{n} \left( w^T x_i - y_i \right)^2
\]

  – A vector ‘w’ satisfying \( w \geq 0 \) (element-wise) is said to be “feasible”.

• Two factors affecting difficulty are *continuity* and *smoothness*.
  – Continuous functions tend to be easier than discontinuous functions.
  – Smooth/differentiable functions tend to be easier than non-smooth.
  – See the calculus review [here](#) if you haven’t heard these words in a while.
Convexity, min, and argmin

• If a function is convex, then all critical points are global optima.

• However, **convex functions don’t necessarily have critical points:**
  – For example, $f(x) = ax$, $f(x) = \exp(x)$, etc.

• Also, **more than one ‘x’ can achieve the global optimum:**
  – For example, $f(x) = c$ is minimized by any ‘x’.
Why use the negative gradient direction?

• For a twice-differentiable ‘f’, multivariable Taylor expansion gives:

\[ f(w^{t+1}) = f(w^t) + \nabla f(w^t)^T (w^{t+1} - w^t) + \frac{1}{2} (w^{t+1} - w^t)^T \nabla^2 f(w)(w^{t+1} - w^t) \]

• If gradient can’t change arbitrarily quickly, Hessian is bounded and:

\[ f(w^{t+1}) = f(w^t) + \nabla f(w^t)^T (w^{t+1} - w^t) + O(||w^{t+1} - w^t||^2) \]

– But which choice of \(w^{t+1}\) decreases ‘f’ the most?

• As \(||w^{t+1} - w^t||\) gets close to zero, the value of \(w^{t+1}\) minimizing \(f(w^{t+1})\) in this formula converges to \((w^{t+1} - w^t) = -\alpha^t \nabla f(w^t)\) for some scalar \(\alpha^t\).

• So if we’re moving a small amount, the optimal \(w^{t+1}\) is:

\[ w^{t+1} = w^t - \alpha^t \nabla f(w^t) \text{ for some scalar } \alpha^t \]
Question from class: "Can we use $w^{t+1} = w^t - \frac{1}{||\nabla f(w^t)||} \nabla f(w^t)$?"

This will work for a while, but notice that

$$||w^{t+1} - w^t|| = ||\frac{1}{||\nabla f(w^t)||} \nabla f(w^t)||$$

$$= \frac{1}{||\nabla f(w^t)||} ||\nabla f(w^t)||$$

$$= 1$$

So the algorithm never converges
Random Sample Consensus (RANSAC)

• In computer vision, a widely-used generic framework for robust fitting is random sample consensus (RANSAC).

• This is designed for the scenario where:
  – You have a large number of outliers.
  – Majority of points are “inliers”: it’s really easy to get low error on them.

Random Sample Consensus (RANSAC)

• **RANSAC:**
  – Sample a small number of training examples.
    • Minimum number needed to fit the model.
    • For linear regression with 1 feature, just 2 examples.
  – Fit the model based on the samples.
    • Fit a line to these 2 points.
    • With ‘d’ features, you’ll need ‘d’ examples.
  – Test how many points are fit well based on the model.
  – Repeat until we find a model that fits at least the expected number of “inliers”.

• You might then re-fit based on the estimated “inliers”.