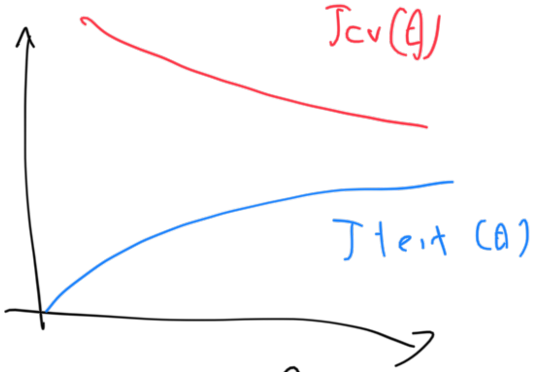
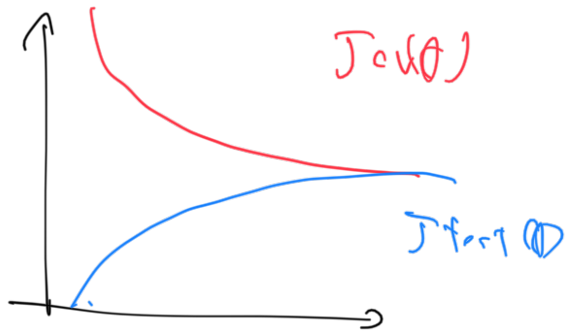
ML w10 learning with large datasets

- high bias dataset ← large # data works.

- high variance dataset ← small # data works.
  (with m is small)



need large
number of
dataset.



already high bias ..

small # data works.

Stocastic Gradient Descent → Iterate 1 example.

1. Randomly shuffle (reorder) training examples.

2. Repeat { (← 1 – 10 x)
   for i := 1 .... m {
   $\theta_j := \theta_j - \alpha(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$
   (for every j = 0 .. n).
   }
   }

Mini Batch Gradient Descent

Say b = 10, m = 1000

Repeat {

for $i = 1, 11, 21, 31, \ldots, 991$ {

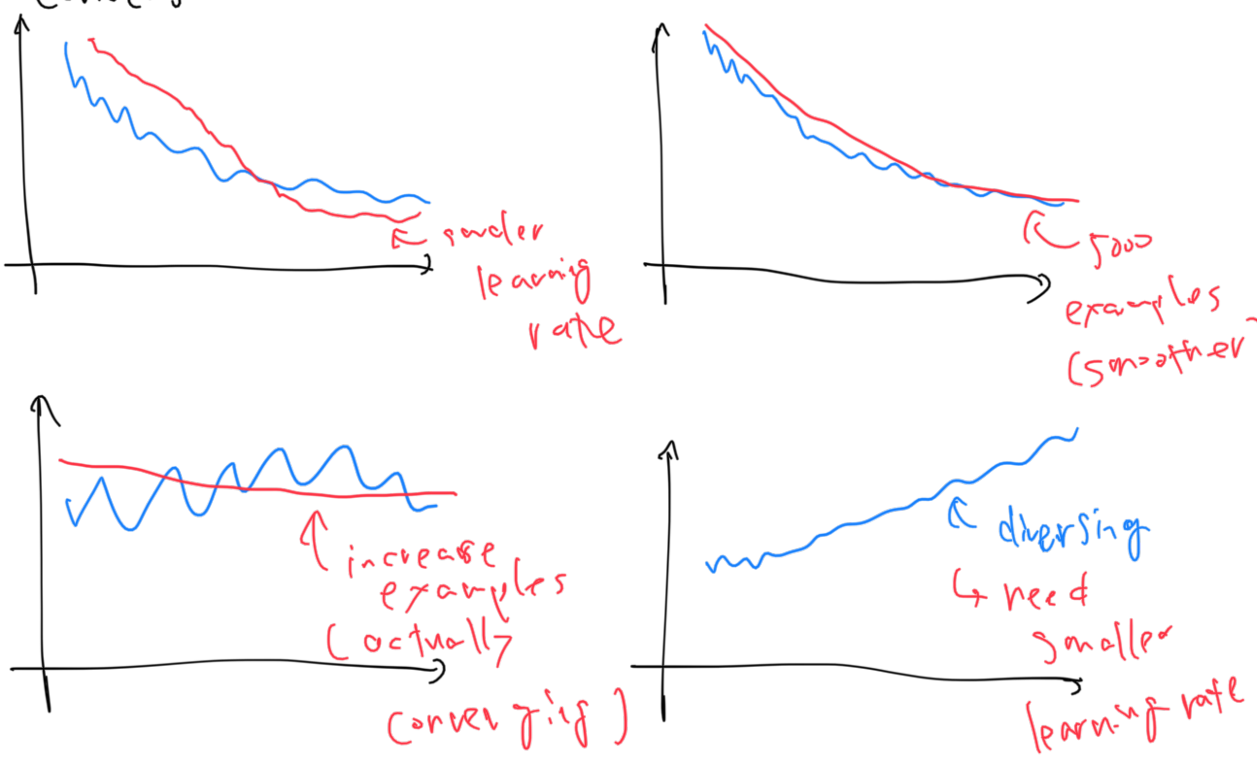
$$\theta_j := \theta_j - \alpha \frac{1}{10} \sum_{k=i}^{i+9} \left(h_\theta(x^{(k)}) - y^{(k)}\right) x_j^{(k)}$$

(for every $j = 0, \ldots, n$)

}

$\mathrm{Cost}(\theta, (x^{(i)}, y^{(i)}))$ average over the last 1000 (say) examples

Conversion



Learning rate α is typically held constant.

Can slowly decrease α over time if we want θ to converge. (E.g. $\alpha = \dfrac{\text{const1}}{\text{iterationNumber} + \text{const2}}$

Online Learning.

Product Search (learning to search)

Product search (learning to search)

User searches for "Android phone 1080p camera".
Have 100 phones in store. Will return 10 results.

x = features of phone.
how many words in user query match name / description of phone, etc.

$y = 1$ if user clicks on link, $y = 0$ otherwise.

Learn $p(y=1 \mid x; \theta)$
Use to show user the 10 phones they are most likely to click on.

(changing user pool, etc)

Map-reduce and Data parallelism

(Suppose $m = 400$)

x^1, y^1
$\vdots$
x^m, y^m

$$temp_j^{(1)} = \sum_{i=1}^{100} \left(h_\theta(x^{(i)}) - y^{(i)}\right) \cdot x_j^{(i)}$$

$$temp_j^{(2)} = \sum_{i=101}^{200} \dots$$

$$temp_j^{(3)} = \sum_{i=201}^{300} \dots$$

$$temp_j^{(4)}$$

Combine result

$$\theta_j := \theta_j - \alpha \frac{1}{400} \left(temp_j^{(1)} + temp_j^{(2)} + \dots \right.$$

1. Suppose you are training a logistic regression classifier using stochastic gradient descent. You find that the cost (say, $cost(\theta, (x^{(i)}, y^{(i)}))$, averaged over the last 500 examples), plotted as a function of the number of iterations, is slowly increasing over time. Which of the following changes are likely to help?

- ( ) Use fewer examples from your training set.
- ( ) Try averaging the cost over a smaller number of examples (say 250 examples instead of 500) in the plot.
- (x) Try halving (decreasing) the learning rate α, and see if that causes the cost to now consistently go down; and if not, keep halving it until it does.
- ( ) This is not possible with stochastic gradient descent, as it is guaranteed to converge to the optimal parameters θ.

2. Which of the following statements about stochastic gradient

descent are true? Check all that apply.

- [x] If you have a huge training set, then stochastic gradient descent may be much faster than batch gradient descent.
- [x] In order to make sure stochastic gradient descent is converging, we typically compute $J_{\text{train}}(\theta)$ after each iteration (and plot it) in order to make sure that the cost function is generally decreasing.
- [ ] One of the advantages of stochastic gradient descent is that it uses parallelization and thus runs much faster than batch gradient descent.
- [x] Before running stochastic gradient descent, you should randomly shuffle (reorder) the training set.

不正解

3. Which of the following statements about online learning are true? Check all that apply.

- [x] Online learning algorithms are usually best suited to problems were we have a continuous/non-stop stream of data that we want to learn from.
- [x] One of the advantages of online learning is that if the function we're modeling changes over time (such as if we are modeling the probability of users clicking on different URLs, and user tastes/preferences are changing over time), the online learning algorithm will automatically adapt to these changes.
- [ ] When using online learning, you must save every new training example you get, as you will need to reuse past examples to re-train the model even after you get new training examples in the future.
- [ ] Online learning algorithms are most appropriate when we have a fixed training set of size m that we want to train on.

4. Assuming that you have a very large training set, which of the following algorithms do you think can be parallelized using map-reduce and splitting the training set across different machines? Check all that apply.

- [ ] An online learning setting, where you repeatedly get a single example (x, y), and want to learn from that single example before moving on.
- [x] Linear regression trained using batch gradient descent.
- [x] A neural network trained using batch gradient descent.
- [ ] Logistic regression trained using stochastic gradient descent.

5. Which of the following statements about map-reduce are true? Check all that apply.

- [ ] Linear regression and logistic regression can be parallelized using map-reduce, but not neural network training.
- [x] Because of network latency and other overhead associated with map-reduce, if we run map-reduce using N computers, we might get less than an N-fold speedup compared to using 1 computer.
- [ ] When using map-reduce with gradient descent, we usually use a single machine that accumulates the gradients from each of the map-reduce machines, in order to compute the parameter update for that iteration.
- [x] If you have only 1 computer with 1 computing core, then map-reduce is unlikely to help.

不正解

2nd. 100%

1. Suppose you are training a logistic regression classifier using stochastic gradient descent. You find that the cost (say, $cost(\theta,(x^{(i)},y^{(i)}))$, averaged over the last 500 examples), plotted as a function of the number of iterations, is slowly increasing over time. Which of the following changes are likely to help?

- ( ) Try averaging the cost over a larger number of examples (say 1000 examples instead of 500) in the plot.
- (•) Try using a smaller learning rate α.
- ( ) Try using a larger learning rate α.
- ( ) This is not an issue, as we expect this to occur with stochastic gradient descent.

2. Which of the following statements about stochastic gradient

descent are true? Check all that apply.

- [x] Before running stochastic gradient descent, you should randomly shuffle (reorder) the training set.
- [ ] In order to make sure stochastic gradient descent is converging, we typically compute $J_{\text{train}}(\theta)$ after each iteration (and plot it) in order to make sure that the cost function is generally decreasing.
- [x] You can use the method of numerical gradient checking to verify that your stochastic gradient descent implementation is bug-free. (One step of stochastic gradient descent computes the partial derivative $\frac{\partial}{\partial\theta_j}cost(\theta,(x^{(i)},y^{(i)}))$.)
- [ ] Suppose you are using stochastic gradient descent to train a linear regression classifier. The cost function $J(\theta)=\frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)})-y^{(i)})^2$ is guaranteed to decrease after every iteration of the stochastic gradient descent algorithm.

3. Which of the following statements about online learning are true? Check all that apply.

- [x] One of the advantages of online learning is that if the function we're modeling changes over time (such as if we are modeling the probability of users clicking on different URLs, and user tastes/preferences are changing over time), the online learning algorithm will automatically adapt to these changes.
- [ ] When using online learning, you must save every new training example you get, as you will need to reuse past examples to re-train the model even after you get new training examples in the future.
- [ ] Online learning algorithms are most appropriate when we have a fixed training set of size m that we want to train on.
- [x] Online learning algorithms are usually best suited to problems were we have a continuous/non-stop stream of data that we want to learn from.

4. Assuming that you have a very large training set, which of the

following algorithms do you think can be parallelized using

map-reduce and splitting the training set across different

machines? Check all that apply.

- [ ] Logistic regression trained using stochastic gradient descent.
- [ ] An online learning setting, where you repeatedly get a single example (x,y), and want to learn from that single example before moving on.
- [x] A neural network trained using batch gradient descent.
- [x] Linear regression trained using batch gradient descent.

5. Which of the following statements about map-reduce are true? Check all that apply.

- [ ] Linear regression and logistic regression can be parallelized using map-reduce, but not neural network training.
- [x] If you have only 1 computer with 1 computing core, then map-reduce is unlikely to help.
- [x] Because of network latency and other overhead associated with map-reduce, if we run map-reduce using N computers, we might get less than an N-fold speedup compared to using 1 computer.
- [x] When using map-reduce with gradient descent, we usually use a single machine that accumulates the gradients from each of the map-reduce machines, in order to compute the parameter update for that iteration.