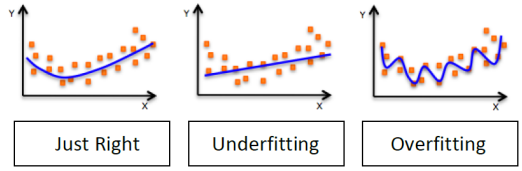
Machine Learning Cheat Sheet 1:

Saturday, October 27, 2018 7:00 PM

2017:

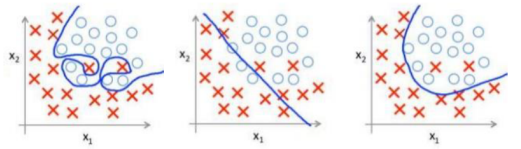
1. The ultimate goal of every machine learning algorithm is: **GENERALIZATION**
2. The top two dangers of treating machine learning as a black box:
   a. An algorithm may be applied to data which **do not fulfill the assumptions** of the algorithm, leading to **incorrect conclusions**.
   b. **Difficult to discern** if the machine learning **outcomes** are due to **statistically-significant** relationships in the data, **or** if they are just **consequences of the algorithmic design.**
3. The top two roles of unsupervised learning in supervised learning:
   a. To find **better features for supervised learning**.
   b. To **reduce** the **dimensionality of the supervised learning problem**.
4. For each row, match the words 'overfitting', 'underfitting' and 'just right' to the three pictures, and write your answers in the boxes below the pictures.

Regression



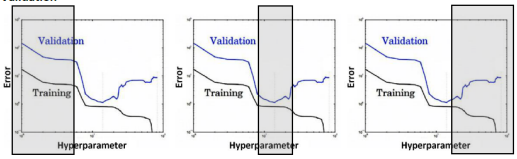
Just Right	Underfitting	Overfitting

Classification



Overfitting	Underfitting	Just Right

Validation



Underfitting	Just Right	Overfitting

5.

Point Loss $\mathcal{L}1(\theta,\theta_0;x,y)$	Predictor	Technique	Algorithm*
$\mathcal{L}_S(y-(\theta^\top x+\theta_0))$	$f(x;\theta,\theta_0)=\theta^\top x+\theta_0$	Linear Regression	Exact Solution, Gradient Descent
$\mathcal{L}_S(y-(\theta^\top x+\theta_0))+\frac{\lambda}{2}\|\theta\|^2$	$f(x;\theta,\theta_0)=\theta^\top x+\theta_0$	Ridge Regression	Exact Solution, Gradient Descent
$\mathcal{L}_H(y(\theta^\top x+\theta_0))$	$h(x;\theta,\theta_0)=\text{sign}(\theta^\top x+\theta_0)$	Linear Classification using Hinge Loss	Gradient Descent
$\mathcal{L}_H(y(\theta^\top x+\theta_0))+\frac{\lambda}{2}\|\theta\|^2$	$h(x;\theta,\theta_0)=\text{sign}(\theta^\top x+\theta_0)$	Support Vector Machine with Slack Variables	Gradient Descent
$\mathcal{L}_Z(y(\theta^\top x+\theta_0))$	$h(x;\theta,\theta_0)=\text{sign}(\theta^\top x+\theta_0)$	Perceptron (with Offset)	Perceptron Algorithm
$\mathcal{L}_L(y(\theta^\top x+\theta_0))$	$p(y\|x,\theta,\theta_0)=\text{sigmoid}(y(\theta^\top x+\theta_0))$	Logistic Regression	Gradient Descent

Learning Cost
$\mathcal{L}_n(\theta,\theta_0)=\frac{1}{n}\sum_{\text{data }(x,y)}\frac{1}{2}(y-(\theta^\top x+\theta$
$\mathcal{L}_n(\theta,\theta_0)=\frac{1}{n}\sum_{\text{data }(x,y)}\frac{1}{2}(y-(\theta^\top x+\theta_0$
$\mathcal{L}_n(\theta,\theta_0)=\frac{1}{n}\sum_{\text{data }(x,y)}\max\{1-y(\theta^\top x$
$\mathcal{L}_n(\theta,\theta_0)=\frac{1}{n}\sum_{\text{data }(x,y)}\max\{1-y(\theta$
$\mathcal{L}_n(\theta,\theta_0)=\frac{1}{n}\sum_{\text{data }(x,y)}[\![y(\theta^\top x+\theta_0$

Gradient for Linear Regression	$\nabla_\theta\mathcal{L}_n(\theta;\mathcal{S}_n)=\frac{1}{n}\sum_{(x,y)\in\mathcal{S}_n}x(\theta^\top x-y)$

	without the offset θ_0:	
6.	**Gradient for Logistic Regression without the offset θ_0:**	$\nabla_\theta \mathcal{L}_n(\theta; \mathcal{S}_n) = \frac{1}{n}\sum_{(x,y)\in\mathcal{S}_n} x(\text{sigmoid}(\theta^\top x) - [\![y = 1]\!])$

7. **[Clustering | Calculating Centroid and boundary of Voronoi Regions]**
The k-means algorithm iteratively computes the set of centroids given a clustering of the data points, and the clustering of the data points given a set of centroids. In this question, you will analyze the performance of the k-means algorithm on a one-dimensional problem.
Suppose that we have six data points 1,2,3,50,100,150 ∈ $\mathbb{R}$.

   a. **If k=1 in your k-means algorithm, where would the centroid of the single cluster be?**
   Ans: centroid = (1+2+3+50+100+150)/6
   = 51

   b. **If k=2 in your k-means algorithm, there will be two Voronoi regions corresponding to the two clusters. The boundary between the Voronoi regions will be:**
   Ans: Exactly halfway between the two centroids.

   c. If k=2 in your k-means algorithm, which of the following are possible clusters when the algorithm has converged? Circle 'Yes' if it is a possible clustering, and 'No' otherwise.

   Ans:

   - {1, 2} and {3, 50, 100, 150} Yes / (No)
   - {1, 2, 3} and {50, 100, 150} Yes / (No)
   - {1, 2, 3, 50} and {100, 150} (Yes) / No
   - {1, 2, 3, 50, 100} and {150} Yes / (No)

   a.

8. **[Collaborative Filtering]**
Suppose that you are given a data set from Amazon in the form of a partially-observed matrix Y whose entry Yai represents the rating of customer a for a product i. You decide to try both the k-nearest-neighbors algorithm and matrix factorization to predict the values of unknown ratings Yai.
In k-nearest-neighbors, to predict the unbiased rating $Yai - Ya$ where Ya is the average of all observed ratings by customer a, we use the weighted sum of the unbiased ratings of neighbors b which are nearest to a. These neighbors are ranked according to a cosine similarity function sim(a,b).

9. **[Alternating Least Squares]** $\bar{r}_a$ **using** $\sum_b w_b r_b$, **which weights** w_b **and values** r_b **should we use?**

In matrix factorization, we choose vectors $U_a \in \mathbb{R}^k$ for each customer a, and vectors $V_i \in \mathbb{R}^k$ for each product i, such that $U_a^\top V_i$ is a good prediction for the rating Y_{ai}. To optimize for these vectors, we use the alternating least-squares algorithm which takes the following form:

1. Initialize vectors $V_i \in \mathbb{R}^k$ randomly.
2. Repeat until convergence:
   a. While fixing the V_i, for each customer a, find the optimal U_a minimizing

$$\sum_{(a,i)\text{ observed}} \frac{1}{2}(Y_{ai} - U_a^\top V_i)^2 + \frac{\lambda}{2}\|U_a\|^2$$

   b. While fixing the U_a, for each product i, find the optimal V_i minimizing

$$\sum_{(a,i)\text{ observed}} \frac{1}{2}(Y_{ai} - U_a^\top V_i)^2 + \frac{\lambda}{2}\|V_i\|^2$$

For step (2a) of the algorithm, for a given customer a, let Z be the vector of all product ratings observed from customer a. Let X be the matrix whose j-th row is V_i if the entry Z_j is a rating for product i.

   a. What is the exact solution for step (2a) of the algorithm?
   Ans:
   $(X^\top X + \lambda I)^{-1} X^\top Z$

   b. What is the reason for using a **non-zero value for** λ?
   Ans: **There are infinitely many solutions for the original training loss of the matrix factorization problem, so we use** $\lambda > 0$ **to ensure that we get a unique solution.**

   C. Improving alternating least squares algo

To improve the prediction accuracies, we now introduce additional parameters and approximate

$$Y_{ai} \approx U_a^\top V_i + \beta_a + \gamma_i + \mu$$

where β_a, γ_i are parameters that represent the bias in the ratings from customer a and from product i respectively, and μ is the average of all the observed ratings. We will pre-compute μ from the data, but the parameters β_a, γ_i will be learned by optimization. The training loss of this new model is

$$\sum \frac{1}{2}(Y_{..} - U_.^\top V_. - \beta_. - \gamma_. - \mu)^2 + \frac{\lambda}{2}\left(\sum \|U_.\|^2 + \sum \|V_.\|^2 + \sum \beta_.^2 + \sum \gamma_.^2\right).$$

$\sum_{(a,i) \text{ observed}} 2($ [illegible] $)$

By applying coordinate descent to this problem, we get the following algorithm.

1. Initialize V_i, β_a, γ_i randomly.
2. Repeat until convergence:
   a. While fixing the V_i, β_a, γ_i, find the optimal U_a.
   b. While fixing the U_a, β_a, γ_i, find the optimal V_i.
   c. While fixing the U_a, V_i, find the optimal β_a, γ_i.

In step (2c) of the algorithm, what is the exact solution for β_a?

Ans:

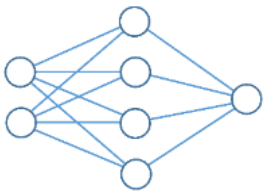
$$\frac{1}{1+\lambda}\sum_{(a,i) \text{ observed}}(Y_{ai} - U_a^\top V_i - \gamma_i - \mu)$$

10. SUPPORT VECTOR MACHINE (SVM)

If a data point (x,y) is a support vector, then the corresponding multiplier $\alpha_{x,y}$ must be equal to zero.	**False**
The margin of the SVM classifier is given by $1/2\|\theta\|^2$.	**False**
If the hyperparameter λ in the objective function $\frac{1}{n}\sum_{\text{data }(x,y)} \max\{1 - y(\theta^\top x + \theta_0),\ 0\} + \frac{\lambda}{2}\|\theta\|^2$ decreases, then the margin of the classifier increases.	**False**
The SVM without slack variables applies to data that is not linearly separable.	**False**
Consider the SVM with slack variables, whose training loss is given by $\frac{1}{2}\|\theta\|^2 + \frac{C}{n}\sum_{(x,y)\in\delta_n} \max\{1 - y(\theta^\top x + \theta_0),\ 0\}.$ To increase the margin, one must increase the hyperparameter C.	**False**
For the SVM with slack variables, if the multiplier $\alpha_{x,y}$ for a data point (x, y) is equal to C, then the point (x, y) must lie on the edge of the margin.	**False**
For the SVM without slack variables, if a data point (x, y) lies on the edge of the margin, then it must be a support vector.	**True**
Computing the SVM classifier with slack variables involves solving a convex optimization problem.	**True**
The size of the margin of the SVM classifier is proportional to $\|\theta\|^{-1}$.	**True**

11. DEEP LEARNING [BACKPROPAGATION]

Consider the three-layer neural network shown below, where $x = (x_1, x_2) \in \mathbb{R}^2$ are input neurons, $y = (y_1, y_2, y_3, y_4) \in \mathbb{R}^4$ are ReLU neurons, and $z \in \mathbb{R}$ is a linear neuron. Let $\tilde{z}$ be the desired output, as given by the training data.



The forward propagation of the neural network may be written as:

$$u = W^{(1)}x + b^{(1)}, \qquad y = \text{ReLU}(u), \qquad z = W^{(2)}y + b^{(2)}$$

where $\text{ReLU}(u) = \max\{0, u\}$. The point loss $\mathcal{L}_1$ is the squared loss $\frac{1}{2}(\tilde{z} - z)^2$. Using backpropagation, the gradients of the point loss are given by

$$\nabla_{W^{(1)}}\mathcal{L}_1 = \delta^{(2)}x^\top, \qquad \nabla_{W^{(2)}}\mathcal{L}_1 = \delta^{(3)}y^\top$$
$$\nabla_{b^{(1)}}\mathcal{L}_1 = \delta^{(2)}, \qquad \nabla_{b^{(2)}}\mathcal{L}_1 = \delta^{(3)}$$

where $\delta^{(3)}, \delta^{(2)}$ are the backpropagating error signals. Let $*$ represent the element-wise multiplication of two vectors, and let H be the function where $H(u) = 1$ if $u > 0$, and $H(u) = 0$ if $u \le 0$.

a. Given that $\delta^{(3)} = z - \tilde{z}$, what is the formula that computes the error signal $\delta^{(2)}$?

Ans: $\delta^{(2)} = (W^{(2)\top}\delta^{(3)}) * H(u)$

b. One can prove that ReLU networks give rise to continuous piecewise-linear functions. In other words, the space of inputs can be cut up into regions where the output of the network is a linear function within each region, and where neighboring regions have different linear functions. For our three-layer ReLU network, what is the maximum number of regions that can be obtained?

Ans: 11

12. [Generative Methods, MLE]

Suppose that you are a myrmecologist (a person who studies ants), and you are currently exploring two large ant nests which are near each other in a forest.
You label the two nests '+' and '-', and let their locations be $\mu^+, \mu^- \in \mathbb{R}^2$ respectively. You observe that the positions of the ants around their own nest seem to follow spherical Gaussian distributions,

$$\mathcal{N}(\mu^+, \sigma^2 I), \qquad \mathcal{N}(\mu^-, \sigma^2 I),$$

with the same variance $\sigma 2$ for both nests because all the ants are of the same species. Let $p+$ be the probability that any given ant is from the '+' nest, and let $p- = 1 - p+$ be the probability for the '-' nest.

a. Using a video (and with help from friends who are computer-vision experts), you painstakingly tracked 1000 ants, and determined which nest they are from. At some snapshot of the video, the positions and nests of these 1000 ants are given by

$$\left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(1000)}, y^{(1000)}\right)$$

where each $x(i) \in \mathbb{R}^2$ and each $y^{(i)} \in \{+, -\}$. Let n^+, n^- be the number of ants observed from each nest respectively. Let $[\![\cdot]\!]$ be the indicator function, and let

b.
$$\hat{p}^+ = \frac{n^+}{1000}, \qquad \hat{p}^- = \frac{n^-}{1000},$$
$$\hat{\mu}^+ = \frac{1}{n^+}\sum_i [\![y^{(i)} = +]\!] x^{(i)}, \quad \mu^- = \frac{1}{n^-}\sum_i [\![y^{(i)} = -]\!] x^{(i)}$$

be the maximum likelihood estimates (MLEs) of the nest probabilities and locations. What is the correct MLE of σ^2 from the data?

Ans:

$$\sigma^2 = \frac{1}{1000} \sum_i \left\| x^{(i)} - \hat{\mu}^{y^{(i)}} \right\|^2$$

c.

You find a new ant wandering around at position $x \in \mathbb{R}^2$. By **substituting the MLEs** above into the **log likelihood ratio**, you determine that the ant is from the '+' nest if $\alpha^\top x + \alpha_0 > 0$ for some α, α_0. What is the value of α and α_0?

Ans:

$$\alpha = \frac{1}{\sigma^2}(\mu^+ - \mu^-), \quad \alpha_0 = \frac{1}{2\sigma^2}(\|\mu^-\|^2 - \|\mu^+\|^2) + \log\frac{p^+}{p^-}$$

You find a new ant wandering around at position $x \in \mathbb{R}2$. The probability that the ant is from the '+' nest may be given by a sigmoid function

$$\mathbb{P}(+|x) = \text{sigmoid}(\theta^\top x + \theta_0) = \frac{1}{1 + e^{-(\theta^\top x + \theta_0)}}.$$

This shows that our model is a special case of logistic regression. What is the value of θ and θ_0?

Ans:

$$\theta = \frac{1}{\sigma^2}(\mu^+ - \mu^-), \quad \theta_0 = \frac{1}{2\sigma^2}(\|\mu^-\|^2 - \|\mu^+\|^2) + \log\frac{p^+}{p^-}$$