# Machine Learning Cheatsheet

Janet Matsen's Machine Learning (ML) notes from CSE 446, Winter 2016. http://courses.cs.washington.edu/courses/cse446/16wi/

Used LaTeX template from an existing Statistics cheat sheet: https://github.com/wzchen/probability_cheatsheet, by William Chen (http://wzchen.com) and Joe Blitzstein.



Last Updated January 26, 2016

## Essential ML ideas

- Never ever ever touch the test set
- You know you are overfitting when there is a big test between train and test results. E.g. metric of percent wrong.
- Need to be comfortable taking a hit on fitting accuracy if you can get a benefit on the result.

## Math/Stat Review

**Random Variable X** belongs to set $\Omega$

**Conditional Probability *is* Probability** $P(A|B)$ is a probability function for any fixed $B$. Any theorem that holds for probability also holds for conditional probability. $P(A|B) = P(A \cap B)/P(B)$

**Bayes' Rule** - Bayes' Rule unites marginal, joint, and conditional probabilities. We use this as the definition of conditional probability.

$$P(\mathbf{A}|\mathbf{B}) = \frac{P(\mathbf{A} \cap \mathbf{B})}{P(\mathbf{B})} = \frac{P(\mathbf{B}|\mathbf{A})P(\mathbf{A})}{P(\mathbf{B})}$$

$$P(A = a \mid B) = \frac{P(A = a)P(B \mid A = a)}{\sum_{a'} P(A = a)P(B \mid A = a)}$$

**Law of Total Probability** : $\sum_x P(X = x) = 1$

**Product Rule** : $P(A, B) = P(A \mid B) \cdot P(B)$

**Sum Rule** : $P(A) = \sum_{x \in \Omega} P(A, B = b)$

**i.i.d** : $D = \{x_i | i = 1 \dots n\}, P(D|\theta) = \prod_i P(x_i \mid \theta)$

Vocab:

- **likelihood function** $L(\theta|O)$ is called as the likelihood function. $\theta$ = unknown parameters, $O$ is the observed outcomes. The likelihood function is conditioned on the observed $O$ and that it is a function of the unknown parameters $\theta$. Not a probability density function.
- **"likelihood" vs "probability"**: if discrete, $L(\theta|O) = P(O|\theta)$. If continuous, $P(O|\theta) = 0$ so instead we estimate $\theta$ given $O$ by maximizing $L(\theta|O) = f(O|\theta)$ where $f$ is the pdf associated with the outcomes $O$.
- **hypothesis space**

## Law of Total Probability (LOTP)

Let $B_1, B_2, B_3, ...B_n$ be a *partition* of the sample space (i.e., they are disjoint and their union is the entire sample space).

$$P(A) = P(A|B_1)P(B_1) + P(A|B_2)P(B_2) + \cdots + P(A|B_n)P(B_n)$$
$$P(A) = P(A \cap B_1) + P(A \cap B_2) + \cdots + P(A \cap B_n)$$

For **LOTP with extra conditioning**, just add in another event $C$!

$$P(A|C) = P(A|B_1, C)P(B_1|C) + \cdots + P(A|B_n, C)P(B_n|C)$$
$$P(A|C) = P(A \cap B_1|C) + P(A \cap B_2|C) + \cdots + P(A \cap B_n|C)$$

Special case of LOTP with $B$ and $B^c$ as partition:

$$P(A) = P(A|B)P(B) + P(A|B^c)P(B^c)$$
$$P(A) = P(A \cap B) + P(A \cap B^c)$$

## Bayes' Rule

**Bayes' Rule, and with extra conditioning (just add in $C$!)**

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$$P(A|B, C) = \frac{P(B|A, C)P(A|C)}{P(B|C)}$$

We can also write

$$P(A|B, C) = \frac{P(A, B, C)}{P(B, C)} = \frac{P(B, C|A)P(A)}{P(B, C)}$$

**Odds Form of Bayes' Rule**

$$\frac{P(A|B)}{P(A^c|B)} = \frac{P(B|A)}{P(B|A^c)} \frac{P(A)}{P(A^c)}$$

The *posterior odds* of $A$ are the *likelihood ratio* times the *prior odds*.

Practice: What is $P(disease \mid +test)$ if P(disease) = 0.01, P(+ | disease) = 0.99, P(+ | no disease) = 0.01?

## Expectation

**f(X)** probability distribution function of X

**X ~ P** : X is distributed according to P.

**Expected value of f under P** : $E_P[f(x)] = \sum_x p(x)f(x)$

E.g. unbiased coin. x = 1, 2, 3, 4, 5, 6. p(X=x) = 1/6 for all x.
$E(X) = \sum_x p(x) \cdot x = (1/6) \cdot [1 + 2 + 3 + 4 + 5 + 6] = 3.5$

## Entropy

Always greater than or equal to 0. Zero when outcome is certain. 1 for uniform distribution.
Entropy is based on a pdf, not a list of labels. E.g.
H[1,1,0] → H[2/2, 1/3].
$X \sim P$, $x \in \Omega$

First define **Surprise**: $S(x) = -\log_2 p(x)$
$S(X = \text{heads}) = -\log_2(1/2) = 1$.

**Axiom 1** : S(1) = 0. (If an event with probability 1 occurs, it is not surprising at all.)

**Axiom 2** : S(q) > S(p) if q < p. (When more unlikely outcomes occur, it is more surprising.)

**Axiom 3** : S(p) is a continuous function of p. (If an outcomes probability changes by a tiny amount, the corresponding surprise should not change by a big amount.)

**Axiom 4** : S(pq) = S(p) + S(q). (Surprise is additive for independent outcomes.)

Surprise of 7 = pretty surprised. Probability of $1/2^7$ of happening (Shannon) **Entropy**:

$$\begin{aligned} H[X] &= -\sum_x p(x) \cdot \log_2 p(x) \\ &= -\sum_x p(x)S(x) \\ &= E[S(x)] \end{aligned}$$

The entropy is the expectation of the surprise. Throw out x for $p(x) = 0$ because log(0) is $\infty$.

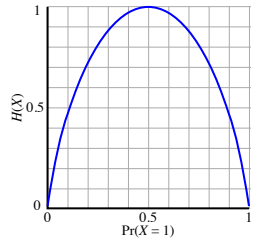
Binary Entropy Function: $p(X = 1) = \theta$ and $p(X = 0) = 1 - \theta$

$$\begin{aligned} H(X) &= -[p(X = 1) \log_2 p(X = 1) + p(X = 0) \log_2 p(X = 0)] \\ &= -[\theta \log_2 \theta + (1 - \theta) \log_2 (1 - \theta)] \end{aligned}$$

Entropy of an unbiased coin flip:
X is a coin flip. $P(X = \text{heads}) = 1/2$, $P(X = \text{tails}) = 1/2$
Note: $\log_2(1/2) = -1$, $-\log_2(1/2) = \log_2(2) = 1$
$H[X] = -[1/2 \log_2(1/2) + 1/2 \log_2(1/2)] = 1$

Entropy of a coin that always flips to heads:
$P(X = \text{heads}) = 1$, $P(X = \text{tails}) = 0$
Note: $\log_x(0) = 0$
$H[X] = -[1 \log_2(1) + 0] = 0$
No surprise: you are sure what you are going to get.

Binary entropy plot.




Canonical example:

| X | Y |
|---|---|
| 0 | 1 |
| 1 | 0 |
| 1 | 1 |

If you want to estimate entropy of X, you can use P(X=0).

$$\begin{aligned} H[X] &= -[\frac{1}{3} \log_2 \frac{1}{3} + \frac{2}{3} \log_2 \frac{2}{3}] \\ &= \frac{1}{3} \log_2 3 + \frac{2}{3} \log_2 3 - \frac{2}{3} \log_2 2 \\ &= \log_2 3 - \frac{2}{3} \approx 0.91 \end{aligned}$$

This time H[X] = H[Y] because of symmetry.
The discrete distribution with maximum entropy is the uniform distribution. For K values of X, $H(X) = \log_2 K$
Conversely, the distribution with minimum entropy (which is zero) is any delta-function that puts all its mass on one state. Such a distribution has no uncertainty.

## Conditional Entropy

If you don't know x: (this is kind of an average).
$H[Y \mid X = x] = -\sum_y P(Y = y \mid X = x) \cdot \log_2 P(y \mid X = x)$
$H[Y \mid X = x] = E[S(Y \mid X = x)]$
Note that we are summing over y because we are specifying x.

For a particular value of X:
$H[Y \mid X] = \sum_x p(x) H[Y \mid X = x]$

Back to table above:

$$H[Y \mid X = 0] = ?$$
look only at X=0 in table.
$$= -[0 + 1 \log_2]$$

Now that you know X=0, entropy goes to 0.

$H[Y \mid X = 1] = 1$: You know *less* if you know X=1.

Now use $H[Y \mid X] = \frac{1}{3}(0) + \frac{2}{3}(1) = 2/3$
Given X, you know more. Average our the more certain case and the less certain case.

Note: $H[Y \mid X] \leq H[Y]$: knowing something can't make you know less.

## Entropy and Information Gain

**Information Gain** - $IG(X) = H(Y) - H(Y \mid X)$
Y is the node on top. X are the nodes below. He might have used lower case.
**Class Example**: If $X_1$ is a node for a split, and you want to know the information gain for that node, you:

- calculate entropy of the split. Find Entropy of each branch of the split, and the fraction of points that were channeled to each split. E.g.

$$\{T, T, T, T, T, F\} \rightarrow \{T, T, T, T\} \text{ (for } X_1 = T), \{T, F\} \text{ (for } X_1 = F)$$
$$\rightarrow P(X_1 = T) = 4/6, P(X_1 = F) = 2/6$$
$$\rightarrow H(X_1 = T) = (1 * \log_2 1 + 0 * \log_2 0) = 0$$
$$\rightarrow H(X_1 = F) = \frac{2}{6}(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2})$$
$$= 1 \text{ (uniform distribution)}$$
$$H(Y|X_1) = -\frac{4}{6}(1 * \log_2 1 + 0 * \log_2 0) - \frac{2}{6}(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2})$$
$$= 2/6$$

- find the entropy of the unsplit data:
  $\{T, T, T, T, T, F\} \rightarrow -(5/6)\log_2(5/6) - (1/6)\log_2(1/6) = 0.65$
- subtract the weighted average of the split entropies from the original: $IG(X_1) = H(Y) - H(Y|X_1) = 0.65 - 0.33$

- Low uncertainty $\leftrightarrow$ Low entropy.
- Lowering entropy $\leftrightarrow$ More information gain.

## Bits

If you use log base 2 for entropy, the resulting units are called bits (short for binary digits).
How many things can you encode in 15 bits? $2^{25}$.

# Decision Trees

Summary:

- One of the most popular ML tools. Easy to understand, implement, and use. Computationally cheap (to solve heurisRcally).
- Uses informaton gain to select attributes (ID3, C4.5,)
- Presented for classification, but can be used for regression and density estimation too
- Decision trees will overfit!!!
- Must use tricks to find simple trees, e.g., (a) Fixed depth/Early stopping, (b) Pruning, (c) Hypothesis testing
- Tree-based methods partition the feature space into a set of rectangles.
- Interpretability is a key advantage of the recursive binary tree.

Pros:

- easy to explain to people
- more closely mirror human decision-making than do the regression and classification approaches
- can be displayed graphically, and are easily interpreted even by a non-expert
- can easily handle qualitative predictors without the need to create dummy variables

Cons:

- trees generally do not have the same level of predictive accuracy as some of the other regression and classification approaches
- can be very non-robust. A small change in the data can cause a large change in the final estimated tree

Vocab:

- **classification tree** - used to predict a qualitative response rather than a quantitative one
- **regression tree** - predicts a quantitative (continuous) variable.
- **depth of tree** -the maximum number of queries that can happen before a leaf is reached and a result obtained
- **split** -
- **node** - synonymous with split. A place where you split the data.
- **node purity** -
- **univariate split** - A split is called univariate if it uses only a single variable, otherwise multivariate.
- **multivariate decision tree** - can split on things like A + B or Petal.Width / Petal.Length ¡ 1. If the multivariate split is a conjunction of univariate splits (e.g. A and B), you probably want to put that in the tree structure instead.
- **univariate decision tree** - a tree with all univariate splits/nodes. E.g. only split on one attribute at a time.
- **binary decision tree**
- **argmax** - the input that leads to the maximum output
- **greedy** - at each step of the tree-building process, the best split is made at that particular step, rather than looking ahead and picking a split that will lead to a better tree in some future step.
- **threshold splits** -
- **random forest**: an ensemble of decision trees which will output a prediction value. Each decision tree is constructed by using a random subset of the training data.

Protocol:

1. Start from empty decision tree
2. Split on next best attribute (feature).
   - Use something like information gain to select next attribute. $\arg\max_i IG(X_i) = \arg\max_i H(Y) - H(Y|X_i)$
3. Recurse

When do we stop decision trees?

- Dont split a node if all matching records have the same output value
- Only split if all of your bins will have data in them. His words: "none of the attributes can create multiple nonempty children." He also said "no attributes can distinguish", and showed that for the remaining training data, each category only had data for one label. And third, "If all records have exactly the same set of input attributes then dont recurse"

He noted that you might not want to stop splitting just because all of your information nodes have zero information gain. You would miss out on things like XOR.
Decision trees will overfit. If your labels have no noise, the training set error is always zero. To prevent overfitting, we must introduce some bias towards simpler trees. Methods available:

- Many strategies for picking simpler trees
- Fixed depth
- Fixed number of leaves
- Or something smarter

One definition of overfitting: If your data is generated from a distribution $D(X, Y)$ and you have a hypothesis space $H$:
Define errors for hypothesis $h \in H$: training error = $error_{train}(h)$, Data (true) error = $error_D(h)$. The hypothesis $h$ overfits the training data if there exists an $h'$ such that $error_{train}(h) < error_{train}(h')$ and $error_D(h) > error_{train}(D)$. In plain english, if there is an alternative hypothesis that gives you more error on the training data but less error in the test data then you have overfit your data.

How to Build Small Trees
Two reasonable approaches:

- Optimize on the held-out (development) set. If growing the tree larger hurts performance, then stop growing. But this requires a larger amount of data
- Use statistical significance testing. Test if the improvement for any split it likely due to noise. If so, dont do the split. Chi Square test w/ MaxPchance = something like 0.05.

Pruning Trees
Start at the bottom, not the top. The top is most likely to have your best splits. In this way, you only cut high branches if all the branches below were cut.
Don't use the validation set for pruning. **Your code should never use the validation set.** The validation set is for **you** to learn from; the code will always learn from the training set.
Classification vs. Regression Trees
In class we mostly discussed nodes with categorical attributes. You can have continuous attributes (see HW1). You can also have either discrete or continuous output. When output is discrete, you can chose your splits based on entropy. If it is continuous, you need to do something more like least squares. For regression trees, see pg 306 from ISL or pg 307 of ESLII.
For discrete data:
"For discrete data, you can't split twice on the same feature. Once you've moved down a branch, you know that all data in that branch has the same value for the splitting feature."
For continuous data:

More computationally expensive than discrete data. Often can try to change continuos data to categorical. Might lose some smoothness for real numbers, but might be worth it
K-fold validation versus using a held-out data set:
If you have enough data to pull out a held-out set, that is preferable to K-fold validation.

# Maximum Likelihood & Maximum a Posteriori

Vocab

- **likelihood**: the probability of the data given a parameter. E.g. $P(D|\theta)$ (for discrete like Binomial). Need not a pdf; need not be normalized.
- **log-likelihood**: lower-case: $l(\theta|x) = \log L(\theta|x)$
- **MLE**: Maximum Likelihood Estimation.
- **PAC**: Probability Approximately Correct.

ML: Maximum Likelihood

MLE: Maximum Likelihood Estimation
Choose $\theta$ to maximize probability of D.
Set derivative of __ to zero and solve. If function is multivariate, set each partial derivative to zero and solve.
$\hat{\theta} = \arg\max_\theta P(D|\theta) = \arg\max_\theta \ln P(D|\theta)$
Note we are using ln, not $\log_2$ as we did for entropy above. Want it to cancel exponents now.

Binomial Distribution
Assumes i.i.d: $D = \{x_i | i = 1 \dots n\}, P(D|\theta) = \prod_i P(x_i \mid \theta)$.
Likelihood function: $P(D|\theta) = \theta^{\alpha_H}(1-\theta)^{\alpha_T}$
P(heads) = $\theta$, P(tails) = $1-\theta$

$$\begin{aligned}\hat{\theta} &= \arg\max_\theta \ln P(D|\theta) \\ &= \arg\max_\theta \ln \theta * \alpha H (1-\theta)^{\alpha T}\end{aligned}$$

Find optimal theta by setting the derivative to zero:

$$\begin{aligned}\frac{d}{d\theta}\ln P(D|\theta) &= \frac{d}{d\theta}\ln \theta * \alpha H(1-\theta)^{\alpha T} \\ &= \arg\max_\theta \ln \theta * \alpha H(1-\theta)^{\alpha T} \\ &= \dots = \frac{\alpha_H}{\alpha_H + \alpha_T}\end{aligned}$$

For Binomial, there is exponential decay in uncertainty with # of observations. You can also find the probability that you are approximately correct (see notes).
$P(|\widehat{\theta} = \theta * | \geq \epsilon) \leq 2e^{-2N\epsilon^2}$. Can calculate N (# of flips) to have error less than $\epsilon$ with probability of being incorrect $\delta$. Your sensitivity depends on your problem; error on stock market data might cost billions.
What if you had prior beliefs? Use MAP instead of MLE.

# Bayesian Learning

Inferring the probability of the parameters themselves, not the probability of the data. Whenever you see $P(\theta|D)$ you know that is some posterior distribution. That is a tidy way of representing your knowledge about $\theta$ and your uncertainty about that knowledge. (The uncertainty is held in the PDF; narrow = certain and flat = uncertain).

Rather than estimating a single $\theta$, we obtain a distribution over possible values of $\theta$.

For small sample size, prior is important!
Use Bayes' Rule: $P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)}$

- **Posterior**: $P(\theta|D)$. Note $P(\theta|D) \propto P(D|\theta)P(\theta)$
- **Data Likelihood**: $P(D|\theta)$
- **Prior**: $P(\theta)$
- **normalization**: $P(D)$. Just a constant so it doesn't matter. Hard to calculate anyway.

Or equivalently, $P(\theta|D) \propto P(D|\theta)P(\theta)$. **Always use this form**, not the one with $P(D)$ in the denominator.

Note: you are multiplying two PDFs here. When you plug in particular data, your two terms become numbers.

As you get more and more data, $P(\theta|D)$ grows more and more narrow. Like with more cannon ball holes, you are more certain about your angle $\theta$.

About the $P(D)$. It is the "marginal probability", which is basically the probability of D when you integrate out $\theta$.

**For uniform priors, MAP reduces to MLE objective**. $P(\theta) \propto 1$ leads to $P(\theta|D) \propto P(D|theta)$

If you have a uniform prior, you just do MLE.
$P(\theta) \propto 1 \rightarrow P(\theta|D) \propto P(D|\theta)$

Note: if you have D first it is Likelihood, and if you have $\theta$ first it is the Posterior. ($P(D|\theta)$ $P(\theta|D)$).
Vocab

- **prior**:
- **prior distribution**: (same as "prior")
- **posterior**:
- **posterior distribution**: (same as "posterior")
- **Maximum likelihood**: Find the parameter that makes the probability highest. E.g. $\theta$ for coin toss. (A famous "point estimator")
- **MAP**: Maximum a posteriori (estimation). Maximize the posterior instead of the likelihood. Take the value that causes the highest point in the posterior distribution.

Just take the peak of your posterior. Forget about the uncertainty. Pretty much like MLE, but you also have some influence of a prior.

Thumbtack Problem, Bayesian style (MAP)
Start as usual with Bayes' without $P(D)$: $P(\theta|D) \propto P(D|\theta)P(\theta)$.
Define parameters: $\theta$ is the probability of one side up. $\alpha_H$ and $\alpha_T$ are the number of heads and tails tossed. $\beta_H$ and $\beta_T$ are the parameters of the prior.

- use Binomial as the likelihood: $P(D|\theta) = \theta^{\alpha_H}(1-\theta)^{\alpha_T}$
- the prior is $P(\theta) = \frac{\theta^{\beta_H}(1-\theta)^{\beta_T - 1}}{B(\beta_H, \beta_T)} \sim \text{Beta}(\beta_H, \beta_T)$. The B in the denominator is for the beta function (not same as beta distribution).
- To get a simple posterior form, use a conjugate prior. Conjugate prior of Binomial is the Beta Distribution. See slides for math: $P(\theta|D) \sim \text{Beta}(\beta_H + \alpha_H, \beta_T, \alpha_T)$
- note that there are similar terms in the prior and likelihood functions. Some will cancel out when you multiply them.
- $P(\theta|D) = \frac{\theta^{\beta_H + \alpha_H - 1} \cdot (1-\theta)^{\beta_T + \alpha_T - 1}}{B(\beta_H + \alpha_H, \beta_T + \alpha_T)} \sim$ Beta$(\beta_H + \alpha_H, \beta_T + \alpha_T)$.
- The Beta prior is equivalent to extra thumbtack flips. As $N \rightarrow \infty$, the prior is forgotten. But for small sample size, prior is important.

MAP (point) estimation:

1. Chose a distribution to fit the data to. Your choice determines the form of the likelihood ($P(\theta|D)$).
2. Chose a prior (distribution). Can use a table that shows conjugate priors for various distributions. Prior is over the parameters you are guessing.
3. Now you have a posterior (multiply prior by likelihood).
4. Plug in your particular data values under many values of $\theta$ to get the likelihood ($P(D|\theta)$). Recall the likelihood need not be a PDF (need not be normalized).
5. Pick the value that causes the highest point on the peak.

MAP estimation
Closely related to Fisher's method of maximum likelihood (ML), but employs an augmented optimization objective which incorporates a prior distribution over the quantity one wants to estimate. You get to pick the distribution to represent the prior. MAP estimation can therefore be seen as a regularization of ML estimation. (Another famous "point estimator")
Chosing between MLE and MAP:
Chose ML if you don't know enough about the domain to impose a new prior.
If you are measuring a continuous variable, Gaussians are your friend.

# Gaussians

Properties of Gaussians:

- Affine transformation (multiplying by a scalar and adding a constant) are Gaussian. If X $\sim$ N($\mu$,$\sigma^2$) and Y = aX + b, then Y $\sim$ N($a\mu + b, a^2\sigma^2$)
- Sum of Gaussians is Gaussian. If X $\sim$ N($\mu_X, \sigma_X^2$), Y $\sim$ N($\mu_Y, \sigma_Y^2$), and Z = X+Y, then Z $\sim$ N($\mu_X + \mu_Y, \sigma_X^2 + \sigma_Y^2$)
- Easy to differentiate.

Learn a Gaussian: $P(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$.
MLE for Gaussian: Prob of i.i.d. samples D = $\{x_1, \dots, x_N\}$:
$P(D|\mu,\sigma) = (\frac{1}{\sigma\sqrt{2\pi}})^N \prod_{i=1}^N e^{\frac{-(x_i-\mu)^2}{2\sigma^2}}$.
Note: it is not $P(\mu,\sigma|D)$, like I thought in class.
Find $\mu_{MLE}, \sigma_{MLE} = \arg\max_{\mu,\sigma} P(D|\mu,\sigma)$.
Log-likelihood:

$$\ln P(D|\mu,\sigma) = \ln[\text{thing above}] = -N\ln\sigma\sqrt{2\pi} - \sum_{i=1}^N \frac{(x_i-\mu)^2}{2\sigma^2}.$$

Differentiate w.r.t. $\mu$ and set = 0. End up with $\widehat{\mu} = \frac{1}{N}\sum_{i=1}^N x_i$.
Differentiate w.r.t. $\sigma$ and set = 0. End up with

$$\widehat{\sigma}^2_{MLE} = \frac{1}{N}\sum_{i=1}^N (x_i - \widehat{\mu})^2.$$

But actually, that leads to a biased estimate, so people actually use

$$\widehat{\sigma}^2_{unbiased} = \frac{1}{N-1}\sum_{i=1}^N (x_i - \widehat{\mu})^2$$

The conjugate priors: mean: use Gaussian prior:
$P(\mu|\nu,\lambda) = \frac{1}{\lambda\sqrt{2\pi}} e^{\frac{-(\mu-\nu)^2}{2\sigma^2}}$. (Instead of $\sigma$, use $\lambda$ and replace the $(x-\mu)^2$ with $(\mu-\nu)^2$).
For variance: use Wishard Distribution:

# Linear Regression

Ordinary Least Squares
Notation:

- $x_i$: an input data point. __ rows by __ columns.
- $y_i$: a predicted output
- $\widehat{y_i}$: a predicted output
- $\widehat{y}$:
- $w_k$: weight k
- $\boldsymbol{w}*$:
- $f_k(x_i)$
- $t_j$: the output variable that you either have data for or are predicting.
- $t(\boldsymbol{x})$: Data. "Mapping from x to t(x)"
- $H$: $H = \{h_1, \ldots, h_K\}$. Basis functions. In the simplest case, they can just be the value of an input variable/feature or a constant (for bias).

Vocab:

- **basis function**
- **bias** - like the intercept in a linear equation. The part that doesn't depend on the features.
- **hyperplane** - a plane, usually with more than 2 dimensions.
- **input variable** - a.k.a. feature. E.g. a column like CEO salary for rows of data corresponding to different companies.
- **response variable** - synonyms: "dependent variable", "regressand", "predicted variable", "measured variable", "explained variable", "experimental variable", "responding variable", "outcome variable", and "output variable". E.g. a predicted stock price.

Ordinary Least Squares:
total error $= \sum_i (y_i - \hat{y_i})^2 = \sum_i (y_i - \sum_k w_k f_k(x_i))^2$
Under the additional assumption that the errors be normally distributed, OLS is the maximum likelihood estimator.
?? Use words to describe what subset of regression in general this is. What is ordinary? What are we limiting?

The regression problem:
Given basis functions $\{h_1, \ldots, h_K\}$ with $h_i(\mathbf{x}) \in \mathbb{R}$,
find coefficients $\boldsymbol{w} = \{w_1, \ldots, w_k\}$.
$t(\boldsymbol{x}) \approx \widehat{f}(\boldsymbol{x}) = \sum_i w_i h_i(\boldsymbol{x})$
This is called linear regression b/c it is linear in the parameters. We can still fit to nonlinear functions by using nonlinear basis functions.
Minimize the **residual squared error**:
$\boldsymbol{w}* = \arg\min_{\boldsymbol{w}} \sum_j (t(\boldsymbol{x}_j) - \sum_i w_i h_i(\boldsymbol{x}_j))^2$

For fitting a line in 2D space, your basis functions are
$\{h_1(x) = x, h_2(x) = 1\}$

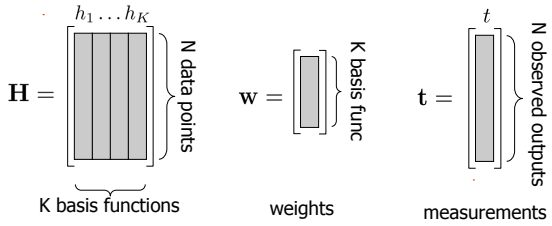
To fit a parabola, your basis functions could be
$\{h_1(x) = x^2, h_2(x) = x, h_3(x) = 1\}$.
Want a 2D parabola? Use
$\{h_1(x) = x_1^2, h_2(x) = x_2^2, h_3(x) = x_1 x_2, \ldots\}$.
Can define any basis functions $h_i(\boldsymbol{x})$ for n-dimensional input $\boldsymbol{x} =< x_1, \ldots, x_n >$

Regression: matrix notation:

$$\boldsymbol{w}* = \arg\min_w \sum_j (t(\boldsymbol{x}_j - \sum_i w_i h_i(\boldsymbol{x}_j))^2$$

$$\boldsymbol{w}* = \arg\min_w (\boldsymbol{Hw} - \boldsymbol{t})^T (\boldsymbol{Hw} - \boldsymbol{t})$$

$(\boldsymbol{Hw} - \boldsymbol{t})^T (\boldsymbol{Hw} - \boldsymbol{t})$ is the residual error.




Regression: closed form solution:

$$\boldsymbol{w}* = \arg\min_w (\boldsymbol{Hw} - \boldsymbol{t})^T (\boldsymbol{Hw} - \boldsymbol{t})$$
$$\boldsymbol{F}(\boldsymbol{w}) = \arg\min_w (\boldsymbol{Hw} - \boldsymbol{t})^T (\boldsymbol{Hw} - \boldsymbol{t})$$
$$\nabla_{\boldsymbol{w}} \boldsymbol{F}(\boldsymbol{w}) = 0$$
$$2\boldsymbol{H}^T(\boldsymbol{Hw} - \boldsymbol{t}) = 0$$
$$(\boldsymbol{H}^T\boldsymbol{Hw}) - \boldsymbol{H}^T\boldsymbol{t} = 0$$
$$\boldsymbol{w}* = (\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T\boldsymbol{t}$$

## Regression solution: simple matrix math

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \underbrace{(\mathbf{Hw} - \mathbf{t})^T(\mathbf{Hw} - \mathbf{t})}_{\text{residual error}}$$

$$\text{solution: } \mathbf{w}^* = \underbrace{(\mathbf{H}^{\mathrm{T}}\mathbf{H})^{-1}}_{\mathbf{A}^{-1}}\underbrace{\mathbf{H}^{\mathrm{T}}\mathbf{t}}_{\mathbf{b}} = \mathbf{A}^{-1}\mathbf{b}$$

$$\text{where } \mathbf{A} = \mathbf{H}^{\mathrm{T}}\mathbf{H} = \underbrace{\quad}_{\text{k×k matrix for k basis functions}} \quad \mathbf{b} = \mathbf{H}^{\mathrm{T}}\mathbf{t} = \underbrace{\quad}_{\text{k×1 vector}}$$

Linear regression prediction is a linear function plus Gaussian noise:
$t(\boldsymbol{x}) = \sum_i w_i h_i(\boldsymbol{x}) + \epsilon$

We can learn **w** using MLE: $P(t|x, w, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-[t - \sum_i w_j h_i(x)]^2}{2\sigma^2}}$
Take the log and maximize with respect to w: (maximizing log-likelihood with respect to w)

$$\ln P(D|\boldsymbol{w}, \sigma) = \ln(\frac{1}{\sigma\sqrt{2\pi}})^N \prod_{j=1}^{N} e^{\frac{-[t_j - \sum_i w_i h_i(x_j)]^2}{2\sigma^2}}$$

Now find the w that maximizes this:
$\arg\max_w \ln(\frac{1}{\sigma\sqrt{2\pi}})^N + \sum_{j=1}^N \frac{-[t_j - \sum_i w_i h_i(x_j)]^2}{2\sigma^2}$
the first term isn't impacted by $w$ so
$= \arg\max_w \sum_{j=1}^N \frac{-[t_j - \sum_i w_i h_i(x_j)]^2}{2\sigma^2}$
switch to $\arg\min_w$ when we divide by -1. The numerator is constant.:
$= \arg\min_w [t_j - \sum_i w_i h_i(x_j)]^2$
**Least-squares Linear Regression is MLE for Gaussians!!!**

# General Vocab

- **held-out data**: synonymous with validation data (?)
- **hypothesis space**: ? E.g. binomial distribution for coin flip.
- **prediction error**: measure of fit (?)
- **regularization**: a process of introducing additional information in order to solve an ill-posed problem or to prevent overfitting