KNN Find k examples $\{\mathbf{x}^{(i)}, t^{(i)}\}$ closest to the test instance $\mathbf{x}$ and then output majority $\arg\max_{t^z} \sum_{r=1}^{k} \delta(t^{(z)}, t^{(r)})$. Define $\delta(a,b) = 1$ if $a = b$, 0 otw. **Choice of k:** Rule is $k < \sqrt{n}$, small k may overfit, while large may underfit. **Curse of Dim:** In high dimensions, "most" points are approximately the same distance. **Computation Cost:** 0 (minimal) at trianing/ no learning involved. Query time find N distances in D dimension $\mathcal{O}(ND)$ and $\mathcal{O}(N \log N)$ sorting time.

Entropy $H(X) = -\mathbb{E}_{X\sim p}[\log_2 p(X)] = -\sum_{x\in X} p(x)\log_2 p(x)$ **Multi-class:** $H(X,Y) = -\sum_{x\in X}\sum_{y\in Y} p(x,y)\log_2 p(x,y)$ **Properties:** H is non-negative, $H(Y|X) \leq H(Y)$, $X \perp Y \implies H(Y|X) = H(Y)$, $H(Y|Y) = 0$, and $H(X,Y) = H(X|Y) + H(Y) = H(Y|X) + H(X)$

Expected Conditional Entropy $H(Y|X) = \mathbb{E}_{X\sim p(x)}[H(Y|X)] = \sum_{x\in X} p(x) H(Y|X=x) = -\sum_{x\in X}\sum_{y\in Y} p(x,y)\log_2 p(y|x) = -\mathbb{E}_{(X,Y)\sim p(x,y)}[\log_2 p(Y|X)]$

Information Gain $IG(Y|X) = H(Y) - H(Y|X)$

Bias Variance Decomposition Using the square error loss $L(y,t) = \frac{1}{2}(y-t)^2$, **Bias ($\uparrow \implies$ underfitting):** How close is our classifier to true target. **Variance ($\uparrow \implies$ overfitting):** How widely dispersed are out predictions as we generate new datasets

$$\begin{aligned}
\mathbb{E}_{\mathbf{x},\mathcal{D}}\left[(h_\mathcal{D}(\mathbf{x}) - t)^2\right] =& \mathbb{E}_{\mathbf{x},\mathcal{D}}\left[(h_\mathcal{D}(\mathbf{x}) - \mathbb{E}_\mathcal{D}[h_\mathcal{D}(\mathbf{x})] + \mathbb{E}_\mathcal{D}[h_\mathcal{D}(\mathbf{x})] - t)^2\right] \\
=& \mathbb{E}_{\mathbf{x},\mathcal{D}}\left[(h_\mathcal{D}(\mathbf{x}) - \mathbb{E}_\mathcal{D}[h_\mathcal{D}(\mathbf{x})])^2 + (\mathbb{E}_\mathcal{D}[h_\mathcal{D}(\mathbf{x})] - t)^2 + 2(h_\mathcal{D}(\mathbf{x}) - \mathbb{E}_\mathcal{D}[h_\mathcal{D}(\mathbf{x})])(\mathbb{E}_\mathcal{D}[h_\mathcal{D}(\mathbf{x})] - t)\right] \\
=& \underbrace{\mathbb{E}_{\mathbf{x},\mathcal{D}}\left[(h_\mathcal{D}(\mathbf{x}) - \mathbb{E}_\mathcal{D}[h_\mathcal{D}(\mathbf{x})])^2\right]}_{\text{variance}} + \underbrace{\mathbb{E}_{\mathbf{x}}\left[(\mathbb{E}_\mathcal{D}[h_\mathcal{D}(\mathbf{x})] - t)^2\right]}_{\text{bias}}
\end{aligned}$$

Bagging with Generating Distribution Suppose we could sample m independent trianing sets $\{\mathcal{D}_i\}_{i=1}^m$ from $p_{dataset}$. Learn $h_i := h_{\mathcal{D}_i}$ and out final predictor is $h = 1/m \sum_{i=1}^m h_i$. **Bias Unchanged:** $\mathbb{E}_{\mathcal{D}_1,\dots,\mathcal{D}_m \overset{iid}{\sim} p_{\text{dataset}}}[h(\mathbf{x})] = \frac{1}{m}\sum_{i=1}^m \mathbb{E}_{\mathcal{D}_i\sim p_{\text{dataset}}}[h_i(\mathbf{x})] = \mathbb{E}_{\mathcal{D}\sim p_{\text{dataset}}}[h_\mathcal{D}(\mathbf{x})]$ **Variance Reduced:** $\text{Var}_{\mathcal{D}_1,\dots,\mathcal{D}_m}[h(\mathbf{x})] = \frac{1}{m^2}\sum_{i=1}^m \text{Var}[h_i(\mathbf{x})] = \frac{1}{m}\text{Var}[h_\mathcal{D}(\mathbf{x})]$

Bootstrap Aggregation Take a single dataset $\mathcal{D}$ with n sample and generate m new datasets, each by sampling n training examples from $\mathcal{D}$, with replacement. We then the average the predictions. We have the reduction in variance to be $\text{Var}\left(\frac{1}{m}\sum_{i=1}^m h_i(\mathbf{x})\right) = \frac{1}{m}(1-\rho)\sigma^2 + \rho\sigma^2$

Random Forest Upon bootstrap aggregation, for each bag we choose a random set of features to make the trees grow on (decorrelates predictions, lower ρ).

Bayes Optimality $\mathbb{E}_{\mathbf{x},\mathcal{D},t|\mathbf{x}}\left[(h_\mathcal{D}(\mathbf{x}) - t)^2\right] = \underbrace{\mathbb{E}_{\mathbf{x}}\left[(\mathbb{E}_\mathcal{D}[h_\mathcal{D}(\mathbf{x})] - y_*(\mathbf{x}))^2\right]}_{\text{bias}} + \underbrace{\mathbb{E}_{\mathbf{x},\mathcal{D}}\left[(h_\mathcal{D}(\mathbf{x}) - \mathbb{E}_\mathcal{D}[h_\mathcal{D}(\mathbf{x})])^2\right]}_{\text{variance}} + \underbrace{\mathbb{E}_{\mathbf{x}}[\text{Var}[t|\mathbf{x}]]}_{\text{Bayes}}$

Vectorized Cost $\mathbf{y} = \mathbf{X}\mathbf{w} + b\mathbf{1}$ and $\mathcal{J} = \frac{1}{2}\|\mathbf{y} - \mathbf{t}\|^2$

Feature Mapping Some time we want fit a polynomial curve, we can do this using a feature map $y = \mathbf{w}^\top \boldsymbol{\psi}(x)$ where $\boldsymbol{\psi}(x) = [1, x, x^2, \dots]^\top$. In general the feature map could be anything.

Ridge Regression $\mathbf{w}_\lambda^{Ridge} = \underset{\mathbf{w}}{\text{argmin}}\mathcal{J}_{\text{reg}}(\mathbf{w}) = \underset{\mathbf{w}}{\text{argmin}}\frac{1}{2}\|\mathbf{X}\mathbf{w} - \mathbf{t}\|_2^2 + \frac{\lambda}{2}\|\mathbf{w}\|_2^2 = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{t}$ Notice that when $\lambda = 0$ this is just OLS solution.

Gradient Descent Consider the some cost function $\mathcal{J}$ and we want to optimize it. **GD:** $\mathbf{w} \leftarrow \mathbf{w} - \alpha\frac{\partial\mathcal{J}}{\partial\mathbf{w}}$ **GD w/ Reg** $\mathbf{w} \leftarrow \mathbf{w} - \alpha\left(\frac{\partial\mathcal{J}}{\partial\mathbf{w}} + \lambda\frac{\partial\mathcal{R}}{\partial\mathbf{w}}\right) = (1-\alpha\lambda)\mathbf{w} - \alpha\frac{\partial\mathcal{J}}{\partial\mathbf{w}}$ **mSGD: SGD:**

a