Math

Probability

- $CDF: F(x) = P(X \leq x)$
  $PDF: \sum f(x)dx = 1$
- **Bayes Theorem**: $P(Y = y_i|X) = \frac{P(X|Y=y_i)P(Y=y_i)}{\Sigma_j P(X|Y=y_j)P(Y=y_j)}$
  $\mathbf{X}$ and $\mathbf{Y}$ are **independent** iff $P(\mathbf{X}, \mathbf{Y}) = P(\mathbf{X})P(\mathbf{Y})$
  $\mathbf{X}$ and $\mathbf{Y}$ are **uncorrelated** iff $\mathbb{E}(\mathbf{X}, \mathbf{Y}) = \mathbb{E}(\mathbf{X})\mathbb{E}(\mathbf{Y})$
- $Var(X) = E[XX^T] - E[X]E[X]^T$

Linear Algebra
$Range: R(A) = w \in W | w = Av, v \in V$
$Nullity: N(A) = v \in V | Av = 0$ $N(A)^\perp = R(A^\perp)$
Matrix calculus
$f(\mathbf{x}) = \mathbf{Ax}+\mathbf{x'A}+\mathbf{x'x}+\mathbf{x'Ax} \Rightarrow \frac{df(\mathbf{x})}{d\mathbf{x}} = \mathbf{A'}+\mathbf{A}+2\mathbf{x}+(\mathbf{Ax}+\mathbf{A'x})$
$\frac{d(x^TAy)}{dA} = xy^T$, $\frac{d(a^TxbA^Tx^Tb)}{dx} = (ab^T + ba^T)$
$\mathbf{H}_{i,j} = \frac{\partial^2 f}{\partial x_i \partial x_j}$; $\nabla_x(a\mathbf{x}) = a\mathbf{I}$; $\mathbf{J} = |\frac{\partial \mathbf{y}}{\partial \mathbf{y}}| \Leftrightarrow \mathbf{J}^{-1} = |\frac{\partial \mathbf{y}}{\partial \mathbf{x}}|$

Perceptron

$f(\mathbf{x}) = \boldsymbol{\theta} \cdot \mathbf{x} + \theta_0 = \sum_{i=1}^d \theta_i x_i + \theta_0$, $\hat{y} = \begin{cases} 1, & \text{if } f(x) \geq 0 \\ -1, & \text{if } f(x) < 0 \end{cases}$

loss for perceptron: $L(z, y) = \begin{cases} 0, & \text{if } zy > 0 \\ -zy, & \text{otherwise.} \end{cases}$,

Decision boundary, a **hyperplane** in $\mathbb{R}^d$:
$H = \{\mathbf{x} \in \mathbb{R}^d : f(\mathbf{x}) = 0\} = \{\mathbf{x} \in \mathbb{R}^d : w \cdot \mathbf{x} + \mathbf{b} = 0\}$

w is the **normal** of the hyperplane,
$\mathbf{b}$ is the **offset** of the hyperplane from origin,
$-\frac{\mathbf{b}}{\|w\|}$ is the **signed distance** from the origin to hyperplane.

Perceptron algorithm,
Input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n) \in \mathbb{R}^d \times \{\pm 1\}$
while some $y_i \neq \text{sign}(\boldsymbol{\theta} \cdot \mathbf{x}_i)$
pick some misclassified $(\mathbf{x}_i, y_i)$
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + y_i \mathbf{x}_i$

Given a **linearly separable data**, perceptron algorithm will take no more than $\frac{R^2}{\gamma^2}$ updates to **converge**, where $R = \max_i \|\mathbf{x}_i\|$ is the radius of the data, $\gamma = \min_i \frac{y_i(\boldsymbol{\theta} \cdot \mathbf{x}_i)}{\|\boldsymbol{\theta}\|}$ is the margin.
Also, $\frac{\boldsymbol{\theta} \cdot \mathbf{x}}{\|\boldsymbol{\theta}\|}$ is the signed distance from H to $\mathbf{x}$ in the direction $\boldsymbol{\theta}$.
$\boldsymbol{\theta} = \sum_i \alpha_i y_i \mathbf{x}_i$, thus any inner product space will work, this is a **kernel**.

Gradient descent view of perceptron, minimize margin cost function $J(\boldsymbol{\theta}) = \sum_i (-y_i(\boldsymbol{\theta} \cdot \mathbf{x}_i))_+$ with $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla J(\boldsymbol{\theta})$

Support Vector Machine

Hard margin SVM,
$\min_{\boldsymbol{\theta}} \|\boldsymbol{\theta}\|^2$, such that $y_i \boldsymbol{\theta} \cdot \mathbf{x}_i \geq 1 (i = 1, \ldots, n)$
Soft margin SVM,
$\min_{\boldsymbol{\theta}} \|\boldsymbol{\theta}\|^2 + C \sum_{i=1}^n (1 - y_i \boldsymbol{\theta} \cdot \mathbf{x}_i)_+$

Regularization and SVMs: Simulated data with many features $\phi(\mathbf{x})$;
C controls trade-off between margin $1/\|\boldsymbol{\theta}\|$ and fit to data;
Large C: Fit to data, more overfitting, smaller margin.
Less data, more features → overfit

$\boldsymbol{\theta} = \sum_j \alpha_j y_j \mathbf{x}_j$, $\alpha_j \neq 0$ only for support vectors.
Margin is $\frac{1}{\|\vec{\boldsymbol{\theta}}\|}$. **Slab** is $\frac{2}{\|\vec{\boldsymbol{\theta}}\|}$.

Decision Theory

Loss function: $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, and $l(\hat{y}, y)$ is the cost of predicting $\hat{y}$ when the outcome is y.

Risk for a given class: $R(\alpha_i|x) = \sum_{i=1}^C \lambda_{ij} P(w = j|x)$

Assume $(\mathbf{X}, \mathbf{Y})$ are chosen i.i.d according to some probability distribution on $\mathcal{X} \times \mathcal{Y}$. **Risk** is misclassification probability:
$R(f) = \mathbb{E}l(f(\mathbf{X}), \mathbf{Y}) = Pr(f(\mathbf{X}) \neq \mathbf{Y})$

Bayes Decision Rule is
$f^*(x) = \begin{cases} 1, & \text{if } P(\mathbf{Y} = 1|x) > P(\mathbf{Y} = -1|x) \\ -1, & \text{otherwise.} \end{cases}$,
and the optimal risk (Bayes risk) $R^* = \inf_f R(f) = R(f^*)$

Excess risk is for any $f : \mathcal{X} \to \{-1, +1\}$,
$R(f) - R^* = \mathbb{E}(1[f(x) \neq f^*(x)]|2P(\mathbf{Y} = +1|\mathbf{X}) - 1|)$
Risk in Regression is expected squared error:
$R(f) = \mathbb{E}l(f(\mathbf{X}), \mathbf{Y}) = \mathbb{EE}[f(\mathbf{X}) - \mathbf{Y}^2|\mathbf{X}]$

Generative and Discriminative

Decision T.H. aka Risk minimization:
Risk for r (classifier) is the expected loss over all values of x,y
$R(r) = E[L(r(x), Y)]$

Bayes decision rule/Bayes classifier: r* minimize R(r)
If 2 class, 0-1 loss, $\{x : P(Y = i|X = x) = 0.5\}$

Three ways to build classifiers
1.Generative model(LDA): $P(Y|x) \Rightarrow P(x|Y)P(Y)$
2.Discriminative model(logistic regression): model P(Y—x) directly.
3.Find decision boundary(SVM): model r(x) directly(no posterior).

Gaussian Discriminant Ana
Assump: each class follows normal dist.
$f(x) = \frac{1}{(\sqrt{2\pi}\sigma)^d} e^{-\frac{\|x-\mu\|^2}{2\sigma^2}}$
Be careful with the sigma! It's one dim, a scalar!
$argmax(P(Y = C|X = x)) = argmax(P(X = x|Y = C)\pi_C)$
$= argmax(log(P(X = x|Y = C)) + log(\pi_C))$
$= argmax(-\frac{\|x-\mu_C\|^2}{2\sigma_C^2} - dlog\sigma_C + log(\pi_C))$

QDA:
For binary classification:
$P(Y = C|X = x) = \frac{e^{Q_C(x)}}{e^{Q_C(x)} + e^{Q_D(x)}} = s(e^{Q_C(x)} - e^{Q_D(x)})$
General form(could be **anisotropic**):
$argmax(-\frac{(x-\mu_C)^T\Sigma^{-1}(x-\mu_C)}{2\sigma_C^2} - log|\Sigma_C| + log(\pi_C))$

LDA: Assump: same variance σ for all.
Find C maximize **linear discriminant fn**:
$f(x, C) = \frac{\mu_C x}{\sigma^2} - \frac{\|\mu_C\|^2}{\sigma^2} + log\pi_C$
General form(could be **anisotropic**):
$argmax(-\frac{(x-\mu_C)^T\Sigma^{-1}(x-\mu_C)}{2\sigma_C^2} - log|\Sigma_C| + log(\pi_C))$

Likelihood of Gaussian: $\hat{\sigma} = \frac{\sum \|x_i - \mu\|^2}{nd}$

Estimation

Maximum likelihood(MLE): Choose parameter so that the distribution it defines gives the obverved data the highest probability (likelihood).

- **Maximum log likelihood**: Log of maximum likelihood, equilvalent to maximum likelihood since log is monotonically increase; it is useful since it can change $\prod$ to $\sum$.
- **Penalized maximum likelihood**: Add a penalty term in the maximum (log) likelihood equation; treat the penalty term as some imaginary data points crafted for desired probability.

Maximum a posterior probability(MAP): the mode of the posterior. If uniform prior, MAP is MLE; if not uniform prior, MAP is Penalized MLE.

Multivariate Normal Distribution

Could be anisotropic:
$\mathbf{x} \in \mathbb{R}^d : p(x) = \frac{1}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}|}} e^{(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}))}$

Covariance matrix: $\boldsymbol{\Sigma} = \mathbb{E}[(\mathbf{X} - \boldsymbol{\mu})(\mathbf{X} - \boldsymbol{\mu})^T]$

- Symmetric: $\boldsymbol{\Sigma}_{i,j} = \boldsymbol{\Sigma}_{j_i}$
- Non-negative diagonal entries: $\boldsymbol{\Sigma} i, i \geq 0$
- Positive semidefinite: $\forall \mathbf{v} \in \mathbb{R}^d, \mathbf{v'}\boldsymbol{\Sigma}\mathbf{v} \geq 0$

Spectral Theorem for **non-diagonal covariance**:
$U = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n]$, $\boldsymbol{\Lambda} = \text{diag}([\lambda_1, \lambda_2, \ldots, \lambda_n]')$
We can eigen decompose $\boldsymbol{\Sigma}^{-1} = U\boldsymbol{\Lambda}^{-1}U'$, this is like to change to a different eigen spaces, where covariances ($\boldsymbol{\Lambda}$) diagonal axis-alianed.
The **eigenvectors** of the sample covariance matrix tell us some orthogonal directions (alternative coordinate axes) along which the points are not correlated.

Given a d-dimensaional Gaussian $\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$,

- write $\mathbf{X} = \begin{bmatrix} \mathbf{Y} \\ \mathbf{Z} \end{bmatrix}$, $\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_\mathbf{Y} \\ \boldsymbol{\mu}_\mathbf{Z} \end{bmatrix}$, $\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{\mathbf{YY}} & \boldsymbol{\Sigma}_{\mathbf{YZ}} \\ \boldsymbol{\Sigma}_{\mathbf{ZY}} & \boldsymbol{\Sigma}_{\mathbf{ZZ}} \end{bmatrix}$,
  where $\mathbf{Y} \in \mathbb{R}^m$, and $\mathbf{Z} \in \mathbb{R}^{d-m}$. Then $\mathbf{Y} \sim \mathcal{N}(\mu_\mathbf{Y}, \boldsymbol{\Sigma}_{\mathbf{YY}})$
- matrix $\mathbf{A} \in \mathbb{R}^{m \times d}$ and vector $\mathbf{b} \in \mathbb{R}^m$, define $\mathbf{Y} = \mathbf{AX} + \mathbf{b}$. Then $\mathbf{Y} \sim \mathcal{N}(\mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}')$
- with $\boldsymbol{\Sigma}$ positive definite,
  $\mathbf{Y} = \boldsymbol{\Sigma}^{-\frac{1}{2}}(\mathbf{X} - \boldsymbol{\mu}) \sim \mathcal{N}(\mathbf{0}, \mathbf{I},)$

Gaussian maximum likelihood estimation:

Sample mean: $\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$;
Sample covariance: $\hat{\boldsymbol{\Sigma}} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \hat{\boldsymbol{\mu}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}})'$

Some terms:
Let each row: a sample pt:

- **centering** X: $x_i - = \mu$ for all i $\Rightarrow \dot{X}$
- **decorrelating** X: $Z = \dot{X}V$, where $Var(R) = \frac{1}{n}\dot{X}^T\dot{X} = V\Sigma V^T$
- **Sphering** X: $Z = \dot{X}Var(R)^{\frac{1}{2}}$
- **Whitening** X: centering + sphering

Regression aka Fitting curves to data

Regression fns:
(1) linear: $h(x; w, \alpha) = wx + \alpha$
(2) polynomial [equivalent to linear regression with added polynomial features]
(3) logistic: $h(x; w, \alpha) = s(wx + \alpha)$

Loss fns(y:true label):
(A) **squared error**: $L(z, y) = (z - y)^2$
(B) **absolute error**: $L(z, y) = |z - y|$
(C) **logistic error** $L(z, y) = -ylnz - (1 - y)ln(1 - z)$

Risk fns:
(a) **mean loss**: $J(h) = \frac{1}{n}\sum_i L(h(X_i), y_i)$
(b) **maximum loss**: $J(h) = max(L(h(X_i), y_i))$
(c) **weighted sum**: $J(h) = \sum_i w_i L(h(X_i), y_i)$
(d) l_1 **penalized**: $J(h) = (a), (b), (c) + \lambda\|w\|_l 1$
(f) l_2 **penalized**: $J(h) = (a), (b), (c) + \lambda\|w\|^2$

Least squares linear regression
(1) + (A) + (a)
Task: Find w, α to minimize $J(w, \alpha) = \sum(X_i \cdot w + \alpha - y_i)^2$
$= \|Xw - y\|^2 = RSS(w)$, for **residual sum of squares**
Solu: $w = (X^TX)^{-1}X^Ty$
- Sensitive to outliers (Errs are squared.)
- Fails if X^TX singular.

Logistic regression
(3) + (C) + (a)
Task: Find w to minimize $J(w) = \sum L(s(X_i \cdot w), y_i)$
Solu: J(w) convex, solved by GD. - Linear regression always separates linearly separable pts.

Least squares polynomial regression: switched to linear by adding polynomial features (Do not forget fictitious dim "1".)

Weighted least-squares polynomial regression: (1) + (A) + (c)
Task: Find w to minimize
$J(w) = (Xw - y)^T\Omega(Xw - y) = \sum w_i(X_i \cdot w - y_i)^2$
Solu: $w = (X^T\Omega X)^{-1}X^T\Omega y$

Newton's method
Often much faster than gradient descent if fn smooth enough.
If convex, reach optimum in one step.(e,g logistic regression with $w_0 = 0$)
Taylor series about v:
$\nabla J(w) = \nabla J(v) + (\nabla^2 J(v))(w - v) + O(\|w - v\|^2)$
Set J(w)=0, we get
$w = v - (\nabla^2 J(v))^{-1}\nabla J(v)$

Steps:

- pick start pt w
- repeat: $e \leftarrow solu\ to\ (\nabla^2 J(v))^{-1}e = \nabla J(v)$,
  $w \leftarrow w + e$

Example: use Newton's method to solve logistic regression faster.(Iteratively reweighted least squares)

- w=0
- $e \leftarrow solu\ to\ (Xw - y)^T\Omega(Xw - y) = X^T(y - s)$
  (Ω, s are fns of w, change through iters)
  $w \leftarrow w + e$

LDA vs logistic regression
Advantages of LDA:

- For well-separated classes, LDA stable; log. reg. surprisingly unstable
- more than 2 classes easy & elegant; log. reg. needs modifying (softmax regression)
- LDA slightly more accurate when classes nearly normal, especially if n is small

Advantages of log. reg.:

- More emphasis on decision boundary; always separates linearly separable pts
- More robust on some non-Gaussian distributions (e.g., dists. w/large skew)
- Naturally fits labels between 0 and 1 [usually probabilities]

ROC curve(for test sets) ROC curve(receiver operating characteristics),

- x-axis: **false positive rate** = % of $-ve$ classified as $+ve$
- y-axis: **true positive rate** = % of $+ve$ classified as $+ve$ aka **sensitivity**
- **false negative rate**: vertical distance from curve to top [1- sensitivity]
- **specificity**: horizontal distance from curve to right [1-false positive rate; true negative rate]

Statistical justifications for regression

Reality: $y_i = g(X_i) + \epsilon_i$, where ϵ_i ~ $N(0, \sigma)$
Goal of regression: find h that estimates g.
Ideal approach: find $h(x) = E_Y[Y|X = x] = g(x) + E[\epsilon] = g(x)$

Empirical Risk
Empirical distribution: the discrete uniform distribution over the sample pts
Empirical risk: expected loss under empirical distribution
$\hat{R(h)} = \frac{1}{n}\sum L(h(X_i), y_i)$ (approximation)

The bias-variance trade-off
2 sources of error in a hypothesis h:

- bias: error due to inability of hypothesis h to fit g perfectly
- variance: error due to fitting random noise in data

$R(h) = E[L(h(z), \gamma)] = E[(h(z) - \gamma)^2]$
$R(h) = \mathbb{E}[\underbrace{(h(z) - g(z))^2}_{\text{bias}^2\text{ of method}}] + \underbrace{Var(h(z))}_{\text{variance of method}} + \underbrace{Var(\epsilon)}_{\text{irreducible error}}$

Some intuitions:

- Underfitting: too much bias
- Most of overfitting: too much variance
- Noise in test set: affect $Var(\epsilon)$,
  Noise in training set: affect bias and $Var(h)$

Regression with penalty

Ridge regression aka Tikhonov Regularization
(1) + (A) + (f)
solu: $(X^TX + \lambda I)w = X^Ty$
Bayesian Justification for Ridge Reg.

Feature subset selection:
Use acc on val set.
All features increase variance, but not all features reduce bias.
1. Forward stepwise selection: Start with null model (0 features);
2. Backward stepwise selection: Start with all d features;
3. Only try to remove features with small weights.
Note: These methods don't guarantee the optimal model.
LASSO
(1) + (A) + (d)
Task: Find w to minimize $\|Xw - y\|^2 + \lambda|w'|$,
where $|w'| = \sum_{i=0}^{d}|w_i|$ (not penalize α)
- The isosurfaces of $|w'|$ are cross-polytopes.
- Normalize the features first before applying Lasso.

Some notes from past exam

- To get Bayes optimal decision boundary, you need both its prior knowledge and(P(Y)) its distribution (P(X—Y)), and always goes for the $argmax(P(Y|X))$.
- For 0,1 loss, Bayes risk is the area under the minimum of curve $P(X|Y)P(Y)$. (Nothing to do with your training data. If $P(X|Y)$ not overlap, the risk would be 0.)
- If the sample cov matrix is not of full rank, the columns of the design matrix are linearly dependent.
- Regression with varying noise is equivalent to weighted least-squares regression, and we are penalized less for deviation from sample points with high variance, cuz we know our measurement is noisy, and we shouldn't try to overfit to it.
- ROC curve is always increasing, not necessarily concave.
- Multiply data matrix by an invertible mat may change its scale hence changing how they are classified., but mul by an orthonormal one won't.
- For 0,1 loss, the LDA decision boundary is
$\{x : Q_C(x) - Q_D(x) = 0\}$
- Logistic regression makes no assumption about linearity, normality, etc.
- LDA finds what the Bayes decision rule would be under the assumption the class conditionals have normal distributions, parameterized by the sample means and covariance.