CS540 Machine learning
L7: MVN
* Lots of students don't realize that when $X$ is rectangular, $\text{inv}(X)$ doesn't exist. This is the most serious problem I saw with Q1, as most of students simplify by taking $\text{inv}(X'X) = \text{inv}(X) \cdot \text{inv}(X')$.

* Also most students still don't realize that $a'b = b'a$. They also have problems seeing the matrix dimensions, so sometime they will equate scalars to vectors without even realizing, or multiply matrices of incompatible dimensions. Some also didn't realize that $AB$ not equal to $BA$. So a common mistake was $X \cdot \text{inv}(X'X) \cdot X' = I$, as it can be rearranged. Anyways, I think some more stress on linear algebra will help.
* Very few students understood that $J(W)$ can be split into $J(w_i)$ and because of this separation of function into $q$ number of functions depending only on $w_i$, it is possible to convert the problem into a bunch of OLS problems.

* Many used the word "independent" ambiguously. Many students had the right idea but failed to clearly express it (and I didn't deduct marks whenever I saw even a glimpse of the right idea).

* Few students also wrote that because norm is positive so that they can change "min of sum" to "sum of min" (which is wrong).
Q3

- * Many students didn't realize that \( J(w) \) is a scalar and when you
- differentiate wrt a scalar, you
- get a scalar. Most of these students got a vector at the end which
- was equated to zero, and then
- they tried to "magically" take the average and get the answer. I
- could see that only few students
- knew how to do matrix differentiation properly.
**Q4**

- * This question had least problems. It was easiest of all (although I thought it may be hard). I found that few students have very similar answers. I even found a bunch of answers with the same (less likely) mistake. I don't know if it was a coincidence or not. But anyway,
- this question was very easy for most of them.
- Only those people lost marks, who didn't attempt the last part.
Last time

- Logistic regression $p(y|x,\theta)$
- Perceptron algorithm
- IRLS (Newton’s algorithm)
- Multinomial logistic regression
- Why probabilistic classifiers?
This time

- Multivariate Gaussians
- Definition
- Eigenanalysis
- MLE
- Plug into a classifier
Correlated features

- Height and weight are not independent
Multivariate Gaussian

• Multivariate Normal (MVN)

\[ \mathcal{N}(x | \mu, \Sigma) \overset{\text{def}}{=} \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right] \]

• Exponent is the Mahalanobis distance between \( x \) and \( \mu \)

\[ \Delta = (x - \mu)^T \Sigma^{-1} (x - \mu) \]

\( \Sigma \) is the covariance matrix (positive definite)

\[ x^T \Sigma x > 0 \ \forall x \]
Bivariate Gaussian

• Covariance matrix is

\[ \Sigma = \begin{pmatrix} \sigma_x^2 & \rho \sigma_x \sigma_y \\ \rho \sigma_x \sigma_y & \sigma_y^2 \end{pmatrix} \]

where the correlation coefficient is

\[ \rho = \frac{Cov(X, Y)}{\sqrt{Var(X)Var(Y)}} \]

and satisfies \(-1 \leq \rho \leq 1\)

• Density is

\[ p(x, y) = \frac{1}{2\pi \sigma_x \sigma_y \sqrt{1 - \rho^2}} \exp \left( -\frac{1}{2(1 - \rho^2)} \left( \frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} - \frac{2\rho xy}{\sigma_x \sigma_y} \right) \right) \]
Spherical, diagonal, full covariance

\[ \Sigma = \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix} \quad \Sigma = \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix} \quad \Sigma = \begin{pmatrix} \sigma_x^2 & \rho \sigma_x \sigma_y \\ \rho \sigma_x \sigma_y & \sigma_y^2 \end{pmatrix} \]
Surface plots

![Surface plots](image)
• We can show analytically that the contours of constant density will be ellipses by studying the eigenvectors / values of $\Sigma$.
• This analysis will prove useful for other things, too.
Eigenvectors and eigenvalues

• We can compute the evects $u_i$ and evals $\lambda_i$ of any square $m \times m$ matrix $A$; these satisfy

\[ A u_i = \lambda_i u_i \]

• In matrix form, this becomes

\[ AU = U \Lambda \]

\[ (A - \Lambda)U = 0 \]

where $\Lambda$ is a diagonal matrix of evals.

• For this set of eqns to have a soln, we require

\[ |A - \Lambda| = 0 \]

• This is a polynomial of order $m$, so it has $m$ solutions (though these need not all be distinct).

• In Matlab, just type

\[ [U,Lam] = \text{eig}(A); \]
Real, symmetric matrices

- If $A_{ij} \in \mathbb{R}$, then $A$ is called real.
- If $A^T = A$, then $A$ is called symmetric.
- Examples include: covariance matrices, kernel matrices and Hessian matrices.
- $A^{-1}$ is also symmetric, since

\[
A^{-1}A = I \\
A^T(A^{-1})^T = I^T \\
AA^{-T} = I \\
A^{-T} = A^{-1}
\]
Orthogonal matrices

- If \( A \) is real and symmetric (so \( A^T = A \)), then one can show that the evals are real and the evecs are orthonormal, i.e. \( u_i^T u_j = \delta(i - j) \)
- In matrix form this becomes \( U^T U = I \)
- We say \( U \) is an orthogonal matrix.
- The rows are also orthonormal since
  \[
  U^T U = I \\
  U^T U U^{-1} = U^{-1} \\
  U U^{-1} = U U^T = I
  \]
Diagonalization

• If $A$ is real and symmetric, then $U$ is orthogonal.
• Hence we can express $A$ as a sum of outer products of the evecs weighted by the evals

$$AU = U\Lambda$$

$$A = U\Lambda U^T = \sum_{i=1}^{p} \lambda_i u_i u_i^T$$

\[
A = \begin{pmatrix} u_1 & \cdots & u_p \end{pmatrix} \begin{pmatrix} \lambda_1 & \cdots & 0 \\ 0 & \ddots & 0 \\ \vdots & \ddots & \lambda_p \end{pmatrix} \begin{pmatrix} u_1^T \\ \vdots \\ u_p^T \end{pmatrix} = \lambda_1(u_1 u_1^T) + \cdots + \lambda_p(u_p u_p^T)
\]
Transformation by an orthogonal matrix

- Consider a vector $\mathbf{x}$ transformed by the orthogonal matrix $\mathbf{U}$ to give

$$\tilde{\mathbf{x}} = \mathbf{U} \mathbf{x}$$

- The length of the vector is preserved since

$$||\tilde{\mathbf{x}}||^2 = \tilde{\mathbf{x}}^T \tilde{\mathbf{x}} = \mathbf{x}^T \mathbf{U}^T \mathbf{U}^T \mathbf{x} = \mathbf{x}^T \mathbf{x} = ||\mathbf{x}||^2$$

- The angle between vectors is preserved

$$\tilde{\mathbf{x}}^T \tilde{\mathbf{y}} = \mathbf{x}^T \mathbf{U} \mathbf{U}^T \mathbf{y} = \mathbf{x}^T \mathbf{y}$$

- Thus multiplication by $\mathbf{U}$ can be interpreted as a rigid rotation of the coordinate system.
Geometry of diagonalization

• Let \( A \) be a linear transformation. We can always decompose this into a rotation \( U \), a scaling \( \Lambda \), and a reverse rotation \( U^T = U^{-1} \).

• Hence \( A = U \Lambda U^T \).

• The inverse mapping is given by \( A^{-1} = U \Lambda^{-1} U^T \)

\[
A = \sum_{i=1}^{m} \lambda_i u_i u_i^T \\
A^{-1} = \sum_{i=1}^{m} \frac{1}{\lambda_i} u_i u_i^T
\]
Positive definite matrices

- A matrix $A$ is pd if $x^T A x > 0$ for any non-zero vector $x$.
- Hence all the evecs of a pd matrix are positive
  
  
  \[
  A u_i = \lambda_i u_i \\
  u_i^T A u_i = \lambda_i u_i^T u_i = \lambda_i > 0
  \]

- A matrix is positive semi definite (psd) if $\lambda_i \geq 0$.
- A matrix of all positive entries is not necessarily pd; conversely, a pd matrix can have negative entries

\[
\begin{array}{c}
\text{> [u,v] = eig([1 2; 3 4])} \\
u = \\
-0.8246 & -0.4160 \\
0.5658 & -0.9094
\end{array}
\begin{array}{c}
\text{[u,v]=eig([2 -1; -1 2])} \\
u = \\
-0.7071 & -0.7071 \\
-0.7071 & 0.7071
\end{array}
\begin{array}{c}
v = \\
-0.3723 & 0 \\
0 & 5.3723
\end{array}
\begin{array}{c}
v = \\
1 & 0 \\
0 & 3
\end{array}
\]
The rank of a matrix is the number of non zero values.

If the matrix is not full rank, it is not invertible, since

\[ A = U \Lambda U^T \]

\[ |A| = |U||\Lambda||U^T| = \prod_{i=1}^{m} \lambda_i \]
Visualizing a covariance matrix

- Let $\Sigma = U \Lambda U^T$. Hence:
  
  $$ \Sigma^{-1} = U^{-T} \Lambda^{-1} U^{-1} = U \Lambda^{-1} U = \sum_{i=1}^{p} \frac{1}{\lambda_i} u_i u_i^T $$

- Let $y = U(x - \mu)$ be a transformed coordinate system, translated by $\mu$ and rotated by $U$. Then:

  $$ (x - \mu)^T \Sigma^{-1} (x - \mu) = (x - \mu)^T \left( \sum_{i=1}^{p} \frac{1}{\lambda_i} u_i u_i^T \right) (x - \mu) $$

  $$ = \sum_{i=1}^{p} \frac{1}{\lambda_i} (x - \mu)^T u_i u_i^T (x - \mu) = \sum_{i=1}^{p} \frac{y_i^2}{\lambda_i} $$
Visualizing a covariance matrix

- From the previous slide
  \[(x - \mu)^T \Sigma^{-1} (x - \mu) = \sum_{i=1}^{p} \frac{y_i^2}{\lambda_i}\]
- Recall that the equation for an ellipse in 2D is
  \[
  \frac{y_1^2}{\lambda_1} + \frac{y_2^2}{\lambda_2} = 1
  \]
- Hence the contours of equiprobability are elliptical, with axes given by the evecs and scales given by the evals of \( \Sigma \)
Standardizing the data

• We can subtract off the mean and divide by the standard deviation of each dimension to get the following (for case $i=1:n$ and dimension $j=1:d$)

$$y_{ij} = \frac{x_{ij} - \bar{x}_j}{\sigma_j}$$

• Then $E[Y]=0$ and $\text{Var}[Y_j]=1$.
• However, $\text{Cov}[Y]$ might still be elliptical due to correlation amongst the dimensions.
Whitening the data

- Let $X \sim N(\mu, \Sigma)$ and $\Sigma = U \Lambda U^T$.
- To remove any correlation, we can apply the following linear transformation

$$Y = \Lambda^{-\frac{1}{2}} U^T X$$

$$\Lambda^{-\frac{1}{2}} = \text{diag}(1/\sqrt{\Lambda_{ii}})$$

- In Matlab

```matlab
[U,D] = eig(cov(X));
Y = sqrt(inv(D)) * U’ * X;
```
Whitening: example
Whitening: proof

- Let
  \[ Y = \Lambda^{-\frac{1}{2}} U^T X \]
  \[ \Lambda^{-\frac{1}{2}} = \text{diag}(1/\sqrt{\Lambda_{ii}}) \]

- Using
  \[ \text{Cov}[AX] = A\text{Cov}[X]A^T \]

we have

\[ \text{Cov}[Y] = \Lambda^{-\frac{1}{2}} U^T \Sigma U \Lambda^{-\frac{1}{2}} \]
\[ = \Lambda^{-\frac{1}{2}} U^T (U \Lambda U^T) U \Lambda^{-\frac{1}{2}} \]
\[ = \Lambda^{-\frac{1}{2}} \Lambda \Lambda^{-\frac{1}{2}} = I \]

and

\[ EY = \Lambda^{-\frac{1}{2}} U^T E[X] \]
Linear transformations of Gaussian RVs

- If $X \sim N(\mu, \Sigma)$ and $Y = AX$, then one can show that $Y \sim N(A\mu, A\Sigma A^T)$
- Hence the whitened data is also Gaussian $Y \sim \mathcal{N}(\Lambda^{-\frac{1}{2}}U^T\mu, I)$
Maximum likelihood estimation

- Log likelihood

\[ \log p(X|\mu, \Sigma) = -\frac{Np}{2} \log(2\pi) - \frac{N}{2} \log |\Sigma| - \frac{1}{2} \sum_{i=1}^{N} (x_i - \mu)^T \Sigma^{-1} (x_i - \mu) \]

- Let \( \Lambda = \Sigma^{-1} \) be the precision matrix

\[ \log p(X|\mu, \Sigma) = -\frac{Np}{2} \log(2\pi) \frac{N}{2} \log |\Lambda| - \frac{1}{2} \sum_{i=1}^{N} (x_i - \mu)^T \Lambda (x_i - \mu) \]

- Just solve

\[ \frac{\partial}{\partial \mu} \log p(X|\mu, \Lambda) = 0, \quad \frac{\partial}{\partial \Lambda} \log p(X|\mu, \Lambda) = 0 \]
MLE for mean

- Log likelihood
  \[
  \log p(X|\mu, \Sigma) = -\frac{Np}{2} \log(2\pi) \frac{N}{2} \log |\Lambda| - \frac{1}{2} \sum_{i=1}^{N} (x_i - \mu)^T \Lambda (x_i - \mu)
  \]

- Taking derivatives wrt a vector
  \[
  \frac{\partial (a^T y)}{\partial y} = a
  \]
  \[
  \frac{\partial (y^T A y)}{\partial y} = (A + A^T)y
  \]

- Let \( y_i = x_i - \mu \) Then
  \[
  \frac{\partial}{\partial \mu}(x_i - \mu)^T \Sigma^{-1} (x_i - \mu) = \frac{\partial}{\partial y_i} \frac{\partial y_i}{\partial \mu} y_i^T \Sigma^{-1} y_i
  \]
  \[
  = -1(\Sigma^{-1} + \Sigma^{-T})y_i
  \]
MLE for mean

• Log likelihood

\[
\log p(X|\mu, \Sigma) = -\frac{Np}{2} \log(2\pi) - \frac{N}{2} \log |\Lambda| - \frac{1}{2} \sum_{i=1}^{N} (x_i - \mu)^T \Lambda (x_i - \mu)
\]

• From before

\[
\frac{\partial}{\partial \mu} (x_i - \mu)^T \Sigma^{-1} (x_i - \mu) = \frac{\partial}{\partial y_i} \frac{\partial y_i}{\partial \mu} y_i^T \Sigma^{-1} y_i = -1(\Sigma^{-1} + \Sigma^{-T})y_i
\]

• Hence

\[
\frac{\partial}{\partial \mu} \log p(X|\mu, \Sigma) = -\frac{1}{2} \sum_{i=1}^{N} -2\Sigma^{-1} (x_i - \mu) = \Sigma^{-1} \sum_{i=1}^{N} (x_i - \mu) = 0
\]

• So finally

\[
\mu_{ML} = \frac{1}{N} \sum_{i} x_i
\]
MLE for $\Sigma$

- **Log likelihood**
  \[
  \log p(X|\mu, \Sigma) = -\frac{Np}{2} \log(2\pi) \frac{N}{2} \log |\Lambda| - \frac{1}{2} \sum_{i=1}^{N} (x_i - \mu)^T \Lambda (x_i - \mu)
  \]

- **Trace of matrix is sum of diagonal entries**
  \[
  \text{tr}(A) = \sum_{i} A_{ii}
  \]

- **Cyclic permutation property of trace**
  \[
  \text{tr}(ABC) = \text{tr}(CAB) = \text{tr}(BCA)
  \]

- **“Trace trick”**
  \[
  x^T Ax = \text{tr}(x^T Ax) = \text{tr}(xx^T A)
  \]

- **Derivatives wrt a matrix**
  \[
  \frac{\partial}{\partial A} \text{tr}(BA) = B^T
  \]
  \[
  \frac{\partial}{\partial A} \log |A| = A^{-T}
  \]
MLE for $\Sigma$

- **Log likelihood**

$$
\ell(\mathcal{D}|\Lambda, \hat{\mu}) = \frac{N}{2} \log |\Lambda| - \frac{1}{2} \sum_i (x_i - \mu)^T \Lambda (x_i - \mu)
$$

$$
= \frac{N}{2} \log |\Lambda| - \frac{1}{2} \sum_i \text{tr}[(x_i - \mu)(x_i - \mu)^T \Lambda]
$$

$$
= \frac{N}{2} \log |\Lambda| - \frac{1}{2} \sum_i \text{tr}[S\Lambda]
$$

$$
S \overset{\text{def}}{=} \sum_i (x_i - \bar{x})(x_i - \bar{x})^T = (\sum_i x_i x_i^T) - N\bar{x}\bar{x}^T
$$

- **Derivative**

$$
\frac{\partial \ell(\mathcal{D}|\Sigma, \hat{\mu})}{\partial \Lambda} = \frac{N}{2} \Lambda^{-T} - \frac{1}{2} S^T = 0
$$

$$
\Lambda^{-T} = \Sigma = \frac{1}{N} S
$$
MLE for $\Sigma$

- **MLE**
  \[
  \Sigma_{ML} = \frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})(x_i - \bar{x})^T
  \]

- **In Matlab**
  \[
  \text{Sigma} = \text{cov}(X, 1)
  \]

- **In 1d**
  \[
  \sigma_{ML}^2 = \frac{1}{N} \sum_{i=1}^{N} (x_i - \mu)^2
  \]

- **Unbiased estimate**
  \[
  \Sigma_{unb} = \frac{1}{N - 1} \sum_{i=1}^{N} (x_i - \bar{x})(x_i - \bar{x})^T
  \]

- **In Matlab**
  \[
  \text{Sigma} = \text{cov}(X)
  \]
Bayes rule for classifiers

\[ p(y = c | x) = \frac{p(x | y = c)p(y = c)}{\sum_{c'} p(x | y = c')p(y = c')} \]

Class posterior

Class-conditional density

Class prior

Normalization constant
Gaussian classifiers

- Class posterior (using plug-in rule)

\[
p(Y = c | x) = \frac{p(x | Y = c)p(Y = c)}{\sum_{c'}^C p(x | Y = c')p(Y = c')}
= \frac{\pi_c |2\pi \Sigma_c|^{-\frac{1}{2}} \exp \left[-\frac{1}{2}(x - \mu_c)^T \Sigma_c^{-1}(x - \mu_c)\right]}{\sum_{c'} \pi_{c'} |2\pi \Sigma_{c'}|^{-\frac{1}{2}} \exp \left[-\frac{1}{2}(x - \mu_{c'})^T \Sigma_{c'}^{-1}(x - \mu_{c'})\right]}
\]

- We will consider the form of this equation for various special cases:
  - \(\Sigma_1 = \Sigma_0\),
  - \(\Sigma_c\) tied, many classes
  - General case

Linear/ quadratic discriminant analysis
$\Sigma_1 = \Sigma_0$

Class posterior simplifies to

$$p(Y = 1|x) = \frac{p(x|Y = 1)p(Y = 1)}{p(x|Y = 1)p(Y = 1) + p(x|Y = 0)p(Y = 0)}$$

$$= \frac{\pi_1 \exp \left[-\frac{1}{2}(x - \mu_1)^T \Sigma^{-1}(x - \mu_1)\right]}{\pi_1 \exp \left[-\frac{1}{2}(x - \mu_1)^T \Sigma^{-1}(x - \mu_1)\right] + \pi_0 \exp \left[-\frac{1}{2}(x - \mu_0)^T \Sigma^{-1}(x - \mu_0)\right]}$$

$$= \frac{\pi_1 e^{a_1}}{\pi_1 e^{a_1} + \pi_0 e^{a_0}} = \frac{1}{1 + \frac{\pi_0}{\pi_1} e^{a_0 - a_1}}$$

$$a_c \overset{\text{def}}{=} -\frac{1}{2}(x - \mu_c)^T \Sigma(x - \mu_c)$$
\[ \Sigma_1 = \Sigma_0 \]

- Class posterior simplifies to

\[
p(Y = 1|x) = \frac{1}{1 + \exp \left[ - \log \frac{\pi_1}{\pi_0} + a_0 - a_1 \right]}
\]

\[
a_0 - a_1 = -\frac{1}{2}(x - \mu_0)^T \Sigma^{-1}(x - \mu_0) + \frac{1}{2}(x - \mu_1)^T \Sigma^{-1}(x - \mu_1)
\]

\[
= -(\mu_1 - \mu_0)^T \Sigma^{-1}x + \frac{1}{2}(\mu_1 - \mu_0)^T \Sigma^{-1}(\mu_1 + \mu_0)
\]

so

\[
p(Y = 1|x) = \frac{1}{1 + \exp \left[ -\beta^T x - \gamma \right]} = \sigma(\beta^T x + \gamma)
\]

\[
\beta \overset{\text{def}}{=} \Sigma^{-1}(\mu_1 - \mu_0)
\]

\[
\gamma \overset{\text{def}}{=} -\frac{1}{2}(\mu_1 - \mu_0)^T \Sigma^{-1}(\mu_1 + \mu_0) + \log \frac{\pi_1}{\pi_0}
\]

\[
\sigma(\eta) \overset{\text{def}}{=} \frac{1}{1 + e^{-\eta}} = \frac{e^{\eta}}{e^{\eta} + 1}
\]

Linear function of x
Decision boundary

- Rewrite class posterior as
  \[ p(Y = 1|x) = \sigma(\beta^T x + \gamma) = \sigma(w^T(x - x_0)) \]
  \[ w = \beta = \Sigma^{-1}(\mu_1 - \mu_0) \]
  \[ x_0 = -\frac{\gamma}{\beta} = \frac{1}{2}(\mu_1 + \mu_0) - \frac{\log(\pi_1/\pi_0)}{(\mu_1 - \mu_0)^T\Sigma^{-1}(\mu_1 - \mu_0)}(\mu_1 - \mu_0) \]

- If \( \Sigma=I \), then \( w=(\mu_1-\mu_0) \) is in the direction of \( \mu_1-\mu_0 \), so the hyperplane is orthogonal to the line between the two means, and intersects it at \( x_0 \)

- If \( \pi_1=\pi_0 \), then \( x_0 = 0.5(\mu_1+\mu_0) \) is midway between the two means

- If \( \pi_1 \) increases, \( x_0 \) decreases, so the boundary shifts toward \( \mu_0 \) (so more space gets mapped to class 1)
Decision boundary in 1d

\[ P(Y=1|X) \neq P(Y=0|X) \]

Discontinuous decision region
Decision boundary in 2d

\[ p(y = 1 | x) = \frac{p(y = 0 | x)}{\lambda} \]
Tied $\Sigma$, many classes

- Similarly to before

\[
p(Y = c|x) = \frac{\pi_c \exp \left[ -\frac{1}{2} (x - \mu_c)^T \Sigma^{-1} (x - \mu_c) \right]}{\sum_{c'} \pi_{c'} \exp \left[ -\frac{1}{2} (x - \mu_{c'})^T \Sigma^{-1} (x - \mu_{c'}) \right]}
\]

\[
= \frac{\exp \left[ \mu_c^T \Sigma^{-1} x - \frac{1}{2} \mu_c^T \Sigma^{-1} \mu_c + \log \pi_c \right]}{\sum_{c'} \exp \left[ \mu_{c'}^T \Sigma^{-1} x - \frac{1}{2} \mu_{c'}^T \Sigma^{-1} \mu_{c'} + \log \pi_{c'} \right]}
\]

\[
\theta_c \overset{\text{def}}{=} \begin{pmatrix} -\mu_c^T \Sigma^{-1} \mu_c + \log \pi_c \\ \Sigma^{-1} \mu_c \end{pmatrix} = \begin{pmatrix} \gamma_c \\ \beta_c \end{pmatrix}
\]

\[
p(Y = c|x) = \frac{e^{\theta_c^T x}}{\sum_{c'} e^{\theta_{c'}^T x}} = \frac{e^{\beta_c^T x + \gamma_c}}{\sum_{c'} e^{\beta_{c'}^T x + \gamma_{c'}}}
\]

- This is the multinomial logit or softmax function
Tied \( \Sigma, \) many classes

- **Discriminant function**

  \[
g_c(x) = -\frac{1}{2} (x - \mu_c)^T \Sigma^{-1} (x - \mu_c) + \log p(Y = c) = \beta_c^T x + \beta_{c0}
\]

  \[
  \beta_c = \Sigma^{-1} \mu_c
  \]

  \[
  \beta_{c0} = -\frac{1}{2} \mu_c^T \Sigma^{-1} \mu_c + \log \pi_c
  \]

- **Decision boundary is again linear, since** \( x^T \Sigma x \) **terms cancel**

- **If** \( \Sigma = I, \) **then the decision boundaries are orthogonal to** \( \mu_i - \mu_j, \) **otherwise skewed**
Decision boundaries

\[ g_1(x) - \max (g_2(x), g_3(x)) = 0 \]

\[
[x, y] = \text{meshgrid(linspace(-10,10,100), linspace(-10,10,100))}; \\
g1 = \text{reshape(mvnpdf(X, \text{mu1(:)'}, S1), [m n])}; ... \\
\text{contour(x,y,g2*p2-max(g1*p1, g3*p3),[0 0],'-k');}
\]
\( \Sigma_0, \Sigma_1 \) arbitrary

- If the \( \Sigma \) are unconstrained, we end up with cross product terms, leading to quadratic decision boundaries
General case

\[ \mu_1 = (0, 0), \mu_2 = (0, 5), \mu_3 = (5, 5), \pi = (1/3, 1/3, 1/3) \]

\[ \Sigma_c = I \]

\[ \Sigma_1 = \begin{pmatrix} 1 & 0 \\ 0 & 3 \end{pmatrix}, \Sigma_2 = \begin{pmatrix} 4 & 0 \\ 0 & 1 \end{pmatrix}, \Sigma_3 = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix} \]

\[ \Pi = (\sigma, \nu_1, \nu_2) \]

\[ \Sigma_1 = \begin{pmatrix} 4 & 0 \\ 0 & 1 \end{pmatrix}, \Sigma_2 = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix} \]