Probabilistic Graphical Models

Homework 1

Bastien PONCHON

1 Learning in Discrete Graphical Models

Let $x = (x_1, ..., x_N)^T$ and $z = (z_1, ..., z_N)^T$ our i.i.d. sample of N observations. We then have:

$$\begin{aligned} p(z, x|\pi, \theta) &= \prod_{i=1}^{N} p(z_i, x_i|\pi, \theta) \\ &= \prod_{i=1}^{N} p(z_i|\pi) p(x_i|z_i, \pi, \theta) \\ &= \prod_{i=1}^{N} \pi_{z_i} \theta_{z_i x_i} \\ &= \prod_{m=1}^{M} \pi_m^{N_m} \prod_{k=1}^{K} \theta_{mk}^{N_{mk}} \end{aligned}$$

Where:

$$\begin{aligned} N_m &= \#\{z_i = m, i = 1..N\} \\ N_{mk} &= \#\{z_i = m \ \& \ x_i = k, i = 1..N\} \end{aligned}$$

We can then write the log-likelihood of our observations as the following:

$$l(\pi, \theta) = \sum_{m=1}^{M} \left(N_m log(\pi_m) + \sum_{k=1}^{K} N_{mk} log(\theta_{mk}) \right)$$

We want to maximize this quantity with regards to π and θ, with the following constraints:

$$\begin{aligned} \sum_m \pi_m &= 1 \\ \forall m \in [1, M], \sum_k \theta_{mk} &= 1 \end{aligned}$$

The Lagrangian associated with this optimization problem is:

$$L(\pi, \theta, \lambda) = -l(\pi, \theta) + \lambda_0 \left(\sum_m \pi_m - 1 \right) + \sum_{m=1}^{M} \lambda_m \left(\sum_k \theta_{mk} - 1 \right)$$

$$\begin{aligned}
\frac{\partial L(\pi,\theta,\lambda)}{\partial \pi_m} &= 0 \\
\Leftrightarrow -\frac{N_m}{\pi_m} + \lambda_0 &= 0 \\
\Leftrightarrow \pi_m &= \frac{N_m}{\lambda_0}
\end{aligned}$$

And

$$\begin{aligned}
\frac{\partial L(\pi,\theta,\lambda)}{\partial \theta_{mk}} &= 0 \\
\Leftrightarrow -\frac{N_{mk}}{\theta_{mk}} + \lambda_m &= 0 \\
\Leftrightarrow \theta_{mk} &= \frac{N_{mk}}{\lambda_m}
\end{aligned}$$

By replacing π and θ in the constraint equations by their expression in λ, we get:

$$\begin{aligned}
\sum_m \pi_m = \sum_m \frac{N_m}{\lambda_0} &= 1 \\
\Leftrightarrow \lambda_0 &= \sum_m N_m = N \\
\Leftrightarrow \forall m \in [1, M], \pi_m &= \frac{N_m}{N}
\end{aligned}$$

and:

$$\begin{aligned}
\forall m \in [1, M], \sum_k \theta_{mk} = \sum_k \frac{N_{mk}}{\lambda_m} &= 1 \\
\Leftrightarrow \forall m \in [1, M], \lambda_m &= \sum_k N_{mk} = N_m \\
\Leftrightarrow \forall m \in [1, M], k \in [1, K], \theta_{mk} &= \frac{N_{mk}}{N_m}
\end{aligned}$$

Where:

$$N_m = \#\{z_i = m, i = 1..N\}$$
$$N_{mk} = \#\{z_i = m \ \& \ x_i = k, i = 1..N\}$$

2 Linear Classification

2.1 Generative Model (LDA)

Let $(x_n, y_n), n \in [1, N]$ a set of N i.i.d. observations assumed to be sampled from the following distributions:

$$y_n \sim Bernoulli(\pi), x_n | y_n = i \sim Normal(\mu_i, \Sigma)$$

2.1.1 (a)

Let us estimate the parameters π, μ_i, Σ based on Maximum Likelihood Estimation.

The Likelihood of our data in this model is:

$$\begin{aligned} L(\pi, \mu, \Sigma) = p(X, Y | \pi, \mu, \Sigma) &= \prod_{n=1}^{N} p(x_n, y_n | \pi, \mu, \Sigma) \\ &= \prod_n p(y_n | \pi) p(x_n | y_n, \mu, \Sigma) \\ &= \prod_n \pi^{y_n} (1-\pi)^{1-y_n} \frac{1}{\sqrt{2\Pi|\Sigma|}} e^{-\frac{1}{2}(x_n - \mu_{y_n})^T \Sigma^{-1} (x_n - \mu_{y_n})} \end{aligned}$$

The log-likelihood is thus:

$$\begin{aligned} l(\pi, \mu, \Sigma) =& log(L(\pi, \mu, \Sigma)) \\ =& \sum_n \left(y_n log(\pi) + (1 - y_n) log(1 - \pi) - \frac{1}{2} log(2\Pi) - \frac{1}{2} log(|\Sigma|) - \frac{1}{2} (x_n - \mu_{y_n})^T \Sigma^{-1} (x_n - \mu_{y_n}) \right) \\ =& N_1 log(\pi) + (N - N_1) log(1 - \pi) - \frac{N}{2} log(2\Pi) - \frac{N}{2} log(|\Sigma|) \\ & - \frac{1}{2} \sum_n (1 - y_n)(x_n - \mu_0)^T \Sigma^{-1} (x_n - \mu_0) - \frac{1}{2} \sum_n y_n (x_n - \mu_1)^T \Sigma^{-1} (x_n - \mu_1) \end{aligned}$$

Where: $N_1 = \#\{n \in [1, N], y_n = 1\}$

l is concave, so let us equal to 0 its partial derivative in order to find the maximum likelihood estimators of π, μ, and Σ:

$$\begin{aligned}
\frac{\partial l(\pi,\mu,\Sigma)}{\partial \pi} &= 0 \\
\Leftrightarrow \frac{N_1}{\pi} - \frac{N-N_1}{1-\pi} &= 0 \\
\Leftrightarrow \pi &= \frac{N_1}{N}
\end{aligned}$$

And:

$$\begin{aligned}
\frac{\partial l(\pi,\mu,\Sigma)}{\partial \mu_1} &= 0 \\
\Leftrightarrow -\frac{1}{2}\sum_n y_n \left(2\Sigma^{-1}x_n - 2\Sigma^{-1}\mu_1\right) &= 0 \\
\Leftrightarrow \mu_1 &= \frac{1}{N_1}\sum_n y_n x_n \\
&= \frac{1}{N_1}\sum_{y_n=1} x_n
\end{aligned}$$

Similarly, we get:

$$\begin{aligned}
\mu_0 &= \frac{1}{N-N_1}\sum_n (1-y_n)x_n \\
&= \frac{1}{N-N_1}\sum_{y_n=0} x_n
\end{aligned}$$

And finally, let us write $A = \Sigma^{-1}$, we then have:

$$\begin{aligned}
l(\pi,\mu,\Sigma) &= f(\pi) + cste - \frac{N}{2}log(|\Sigma|) - \frac{1}{2}\sum_n (x_n-\mu_{y_n})^T\Sigma^{-1}(x_n-\mu_{y_n}) \\
&= f(\pi) + cste + \frac{N}{2}log(|A|) - \frac{1}{2}\sum_n (x_n-\mu_{y_n})^T A(x_n-\mu_{y_n}) \\
&= f(\pi) + cste + \frac{N}{2}log(|A|) - \frac{N}{2}Tr(\tilde{\Sigma}A) \\
&= l(\pi,\mu,A)
\end{aligned}$$

Where: $\tilde{\Sigma} = \frac{1}{N}\sum_n (x_n-\mu_{y_n})(x_n-\mu_{y_n})^T$
By taking the partial derivative with regards to A (as maximizing with

regards to Σ is the same as maximizing with regards to A), we get:

$$\begin{aligned} \frac{\partial l(\pi, \mu, A)}{\partial A} &= 0 \\ \Leftrightarrow \frac{N}{2} A^{-1} - \frac{N}{2} \tilde{\Sigma} &= 0 \\ \Leftrightarrow A^{-1} &= \tilde{\Sigma} \\ \Leftrightarrow \Sigma &= \tilde{\Sigma} \end{aligned}$$

2.1.2 (b)

Now, for given parameters μ, π, Σ, let us express the conditional distribution $p(y = 1|x)$:

$$\begin{aligned} p(y=1|x) &= \frac{p(x|y=1)p(y=1)}{p(x|y=1)p(y=1) + p(x|y=0)p(y=0)} \\ &= \frac{1}{1 + \frac{p(x|y=0)}{p(x|y=1)} \frac{p(y=0)}{p(y=1)}} \\ &= \frac{1}{1 + \frac{1-\pi}{\pi} e^{-\frac{1}{2}(x-\mu_0)^T \Sigma^{-1}(x-\mu_0) + \frac{1}{2}(x-\mu_1)^T \Sigma^{-1}(x-\mu_1)}} \\ &= \frac{1}{1 + \frac{1-\pi}{\pi} e^{-\frac{1}{2}\left(-2\mu_0^T \Sigma^{-1} x + 2\mu_1^T \Sigma^{-1} x + \mu_0 \Sigma^{-1} \mu_0 - \mu_1 \Sigma^{-1} \mu_1\right)}} \end{aligned}$$

This can be seen as a logistic regression $p(y = 1|x) = \sigma(w^T x + b)$ with:

$$\begin{aligned} w &= \Sigma^{-1}(\mu_1 - \mu_0) \\ b &= \frac{1}{2}(\mu_0 \Sigma^{-1} \mu_0 - \mu_1 \Sigma^{-1} \mu_1) - log(\frac{1-\pi}{\pi}) \\ \sigma &: z \mapsto \frac{1}{1+e^{-z}} \end{aligned}$$

2.1.3 (c)

I implemented the Maximum Likelihood Estimation for this model and applied it to the 3 data sets classificationA.train, classification.train and classificationC.train.

I got the following results:

As we can see in Figure 1, the LDA model with Fisher's assumption $\Sigma_1 = \Sigma_0 = \Sigma$ is quite relevant for train set A, as both classes seems to follow the same distribution and have the quite the same intra-class co-variance.

This assumption no longer holds for train set B, where the two classes seems to have completely different intra-class covariances. However, the assumption

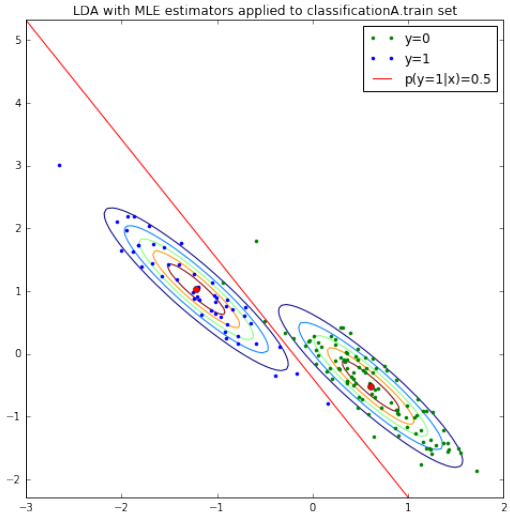

Figure 1: LDA model applied to classificationA.train set, with parameters estimated by MLE

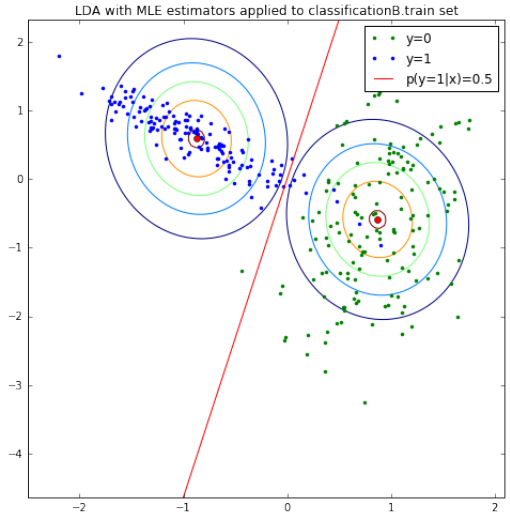

Figure 2: LDA model applied to classificationB.train set, with parameters estimated by MLE

of multivariate gaussian distribution for each class remains relevant, as it can be seen in Figure 2.

Finally, as it can be seen in Figure 3,none of these assumptions holds for the train set C, as the class $y = 1$ seems to have two different modes and therefore

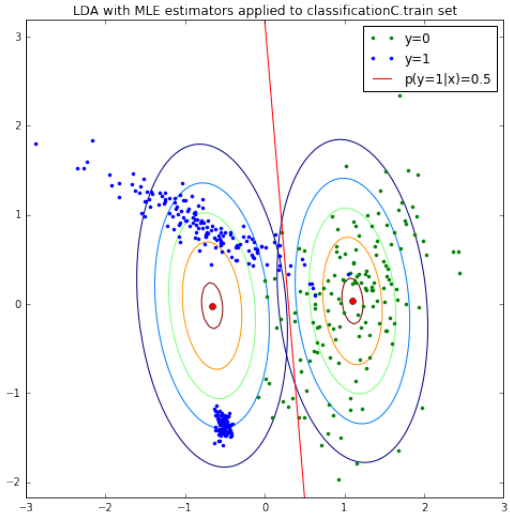

Figure 3: LDA model applied to classificationC.train set, with parameters estimated by MLE

be ill-modeled by a normal distribution. A Gaussian Mixture Model may be more suitable for this data set.

2.2 Logistic Regression

We implemented logistic regression for an affine function, using the IRLS algorithm to estimate the parameters.

To simplify the algorithm, we applied it to the vector $w' = (w_0, w_1, b)^T$ and $x' = (x_0, x_1, 1)^T$. The IRLS algorithm then boils down to initializing w' randomly and then for a number of iterations, iterate its value:

$$w^{(t+1)} \leftarrow w^{(t)} + (X'^T D_{\eta^{(t)}} X -)^{-1} X'^T (y - \eta^{(t)})$$

with:

$$D_{\eta^{(t)}} = Diag((\eta_i^{(t)}(1 - \eta_i^{(t)}))_i)$$

and

$$\eta^{(t)} = \sigma(w'^T x)$$

For a 10 iterations of the algorithm, we get the following parameters estimated values (rounded):

	w_0	w_1	b
A	-316.35	-203.53	-69.77
B	-6.40	1.67	1.37
C	-7.17	1.62	2.06

Just as for the LDA generative model, the equation of the line $p(y = 1|x) = 0.5$ is $w^T + b = 0$ (as in both cases, this conditional distribution can be represented as the image of $w^T + b$ by a sigmoid function).

Here how this model performed when trained on A, B and C train data sets:

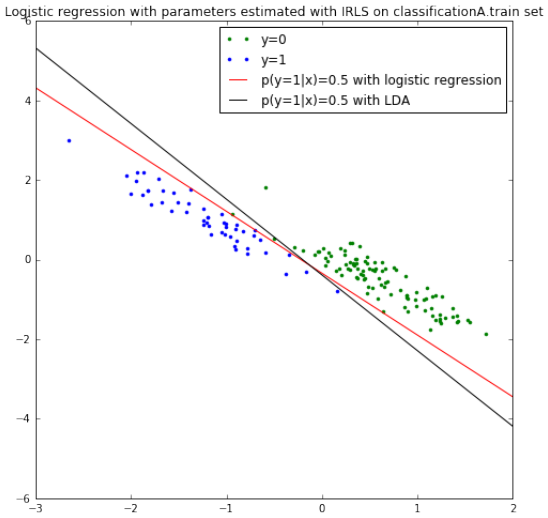

Figure 4: Logistic regression trained on classificationA.train set compared with LDA classification

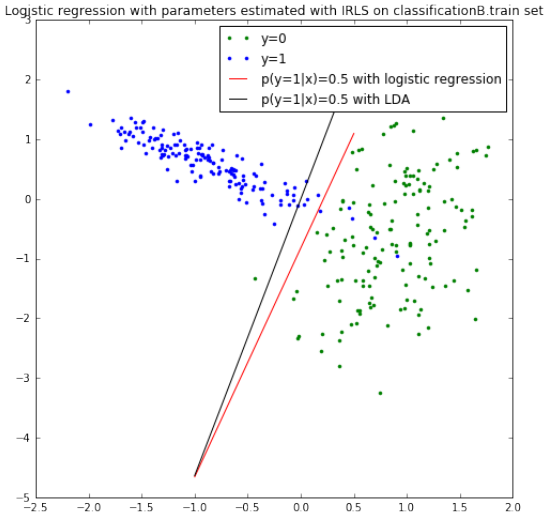

Figure 5: Logistic regression trained on classificationB.train set compared with LDA classification

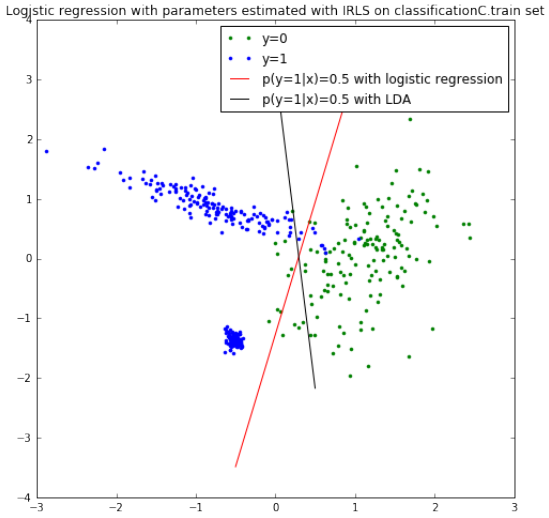

Figure 6: Logistic regression trained on classificationC.train set compared with LDA classification

We can note that it is the case where LDA assumptions do not hold (C dataset, that is represented on Figure 6) that the lines representing the conditional distribution of $y = 1$ given x differ the most between the two models.

2.3 Linear regression

We then implemented linear regression, that is to say a model where:

$$y|x \sim Normal(w^T x + b, \sigma^2)$$

But with value of y replaced by 1 if > 0.5 and by 0 else.

The values of w, b, and σ^2 are estimated by solving the normal equations (we replaced w by $w' = (w^T, b)^T$, just like we did with the logistic regression). We get the following estimated values (rounded):

	w_0	w_1	b	σ^2
A	-0.809	-0.425	0.333	0.0399
B	-0.391	0.084	0.5	0.054
C	-0.416	-0.0388	0.625	0.0622

The equation of the line $p(y = 1|x) = 0.5$ is $w^T + b = 0.5$ (as $w^T + b$ is the mean of the Gaussian from which is sampled $y|x$, which will be then compared with 0.5 to determine whether to predict a 0 or a 1).

Figures 7, 8 and 9 shows the Linear regression trained on A, B and C (resp.) train data sets, compared with LDA and logistic regression classifications.

Note that the line $p(y = 1|x) = 0.5$ is the same for LDA and linear regression.

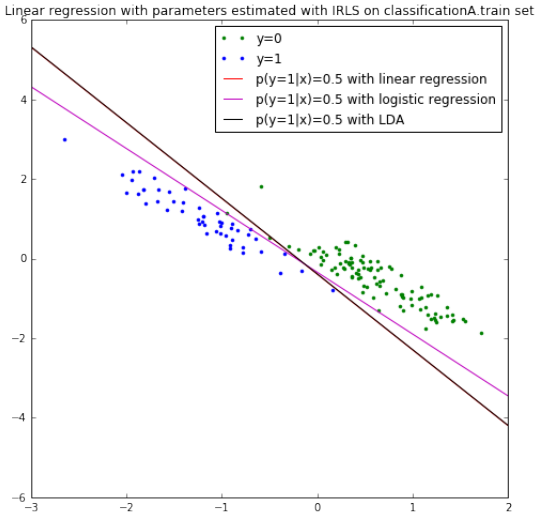

Figure 7: Linear regression trained on classificationA.train set compared with LDA and logistic regression classifications

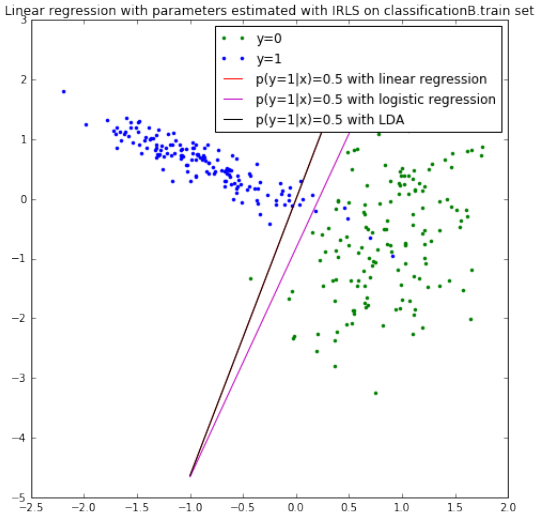

Figure 8: Linear regression trained on classificationB.train set compared with LDA and logistic regression classifications

2.4 Comparing Models

In order to compare three models, we computed the misclassification error of each of them both on A, B and C train (used to train the models) and test datasets.

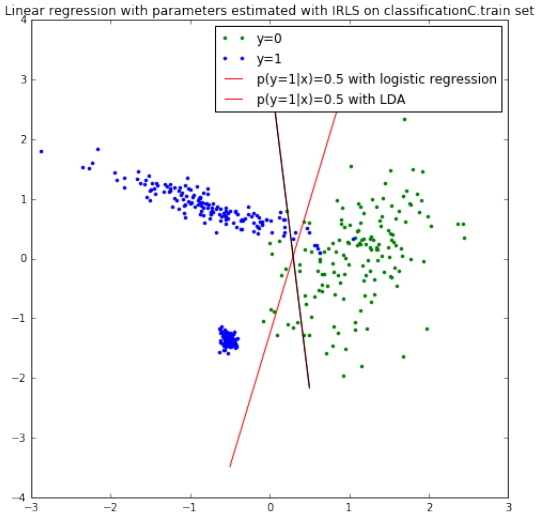

Figure 9: Linear regression trained on classificationC.train set compared with LDA and logistic regression classifications

We get the following results:

Data set A	Train Set	Test Set
LDA	1.33%	1.73%
Logistic regression	0%	4.07%
Linear regression	1.33%	2.33%

Data set B	Train Set	Test Set
LDA	3%	4.45%
Logistic regression	2%	4.2%
Linear regression	3%	4.2%

Data set C	Train Set	Test Set
LDA	5.5%	4.33%
Logistic regression	4%	2.47%
Linear regression	5.5%	4.5%

First of all, we note that the misclassification error is larger on the test sets that on the train sets, even though both are usually close. This seems logical as the train sets are the one that has actually been used to train the models. However, as in our cases the test sets closely follow the same distribution as the train sets, it comes with no surprise that the misclassification errors remain close.

We also note that the error on the train set of the LDA and the Linear regression are always close, if not the same. This can be related to the fact

that the lines of equation $p(y = 1|x) = 0.5$ are the same for both models. On average, the errors of both models are always very close, which can accounted on the fact that both take the assumption of the data from each cluster being sampled from a Gaussian with the same covariance.

We note that all models perform better on data set A, than on data sets B and C. This is probably due to the strong regularity of dataset A (two clusters of equal covariances). On data set A, we note that LDA is the model that performed the best on the test set, which can be explained by the fact that this data set verifies all the assumptions of this model. However LDA performs more poorly on the two other data sets that does not verify these assumptions.

On data sets B and C, Logistic regression yields the best results. This accounts for the fact that these two data sets are not generated from Normal distributions with equal co-variances between both classes.

2.5 QDA model

Now we relax the assumptions from the LDA of the co-variances of both classes having to be equal. In the MLE algorithm, this only changes the values of Σ_0 and Σ_1, which, instead of being equal to $\frac{1}{N}\sum_n (x_n - \mu_{y_n})(x_n - \mu_{y_n})^T$, are now equal to $\frac{1}{N-N_1}\sum_{n/y_n=0}(x_n-\mu_0)(x_n-\mu_0)^T$ and $\frac{1}{N_1}\sum_{n/y_n=1}(x_n-\mu_1)(x_n-\mu_1)^T$ respectively.

We get the following learned values (rounded):

	pi	μ_0	μ_1	Σ_0	Σ_1
A	0.333	$(0.608, -0.514)^T$	$(-1.22, 1.03)^T$	$\begin{pmatrix} 0.246 & -0.300 \\ -0.300 & 0.442 \end{pmatrix}$	$\begin{pmatrix} 0.288 & -0.372 \\ -0.372 & 0.530 \end{pmatrix}$
B	0.5	$(0.873, -0.588)^T$	$(-0.873, 0.588)^T$	$\begin{pmatrix} 0.180 & 0.174 \\ 0.174 & 1.114 \end{pmatrix}$	$\begin{pmatrix} 0.295 & -0.218 \\ -0.218 & 0.194 \end{pmatrix}$
C	0.625	$(1.10, 0.0327)^T$	$(-0.661, -0.0196)^T$	$\begin{pmatrix} 0.274 & 0.168 \\ 0.168 & 0.562 \end{pmatrix}$	$\begin{pmatrix} 0.271 & -0.237 \\ -0.237 & 1.26 \end{pmatrix}$

Figures 10, 11, and 12 show the train set A, B and C respectively, with the conic defined by the equation $p(y = 1|x) = 0.5$.

Here is the misclassification error of QDA model on A, B and C train and test data sets:

QDA Model	Train Set	Test Set
Data Set A	0.667%	2.2%
Data Set B	1.33%	1.9%
Data Set C	5.25%	3.4%

We can note that the QDA performs better than the other three models on data set B, and performs as well on data set A, but as poorly on data set C (more poorly than Logistic regression on data set C). This accounts for the fact that QDA model makes the assumption of unimodal normal data distribution

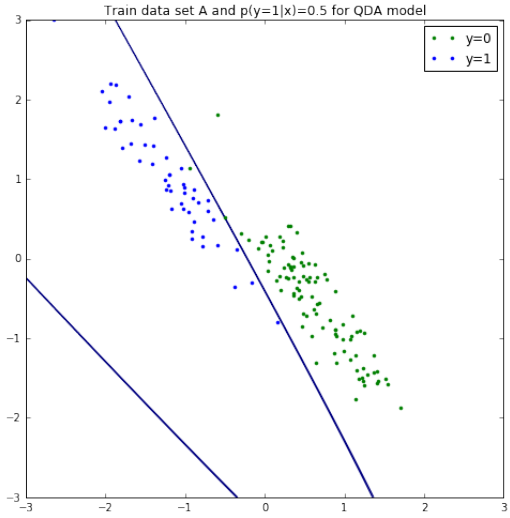

Figure 10: QDA trained on classificationA.train set

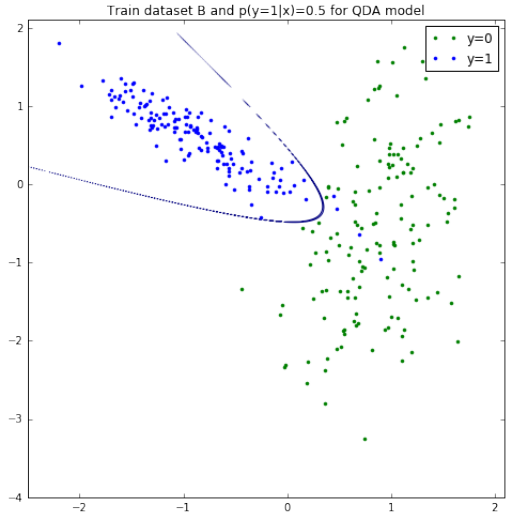

Figure 11: QDA model trained on classificationB.train set

for each cluster. This assumption is verified both by data set A and B. However data set C has a class whose distribution seems to have two modes.

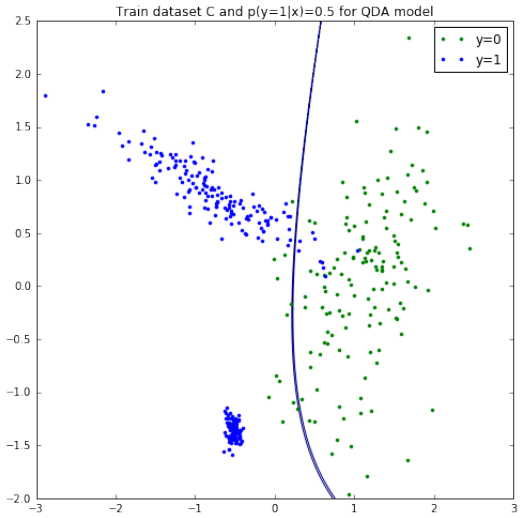

Figure 12: QDA model trained on classificationC.train set