Machine Learning - I (CS/DS 864)
Aug–Dec
***Instructor*: Prof. G.Srinivasaraghavan**

Lecture Notes

Linear Models

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications. They may be distributed outside this class only with the permission of the Instructor.*

Contents

1 Introduction **5**

2 Transformations Using Non-Linear Basis Functions **6**

3 Linear Regression **7**

3.1 Least Squares Regression 7

3.2 Equivalence of MLE with Gaussian Errors and Least Squares 9

3.3 Robust Linear Regression 10

3.3.1 Laplace Regression 10

3.3.2 Ridge Regression 11

3.3.3 Lasso Regression 12

4 Logistic Regression **12**

4.1 Decision Making Using Logistic Regression 16

4.2 Multinomial Logistic Regression 17

5 Decision Theory - Introduction **19**

6 Fischer's Linear Discriminant **19**

6.1 Decision Making from Fisher's Linear Discriminant 21

6.2 Least Squares Interpretation of the Fisher's Discriminant 22

6.3 Multiclass-Multidimensional FLDA 22

7 Generalized Linear Models **24**

7.1 Structure of a GLM 25

7.2 Exponential Family of Distributions 26

7.2.1 Examples of Exponential Family Distributions 27

7.3 Properties of the Log-Partition Function 28

7.4 GLM Formulation 29

7.5 Maximum Likelihood with GLM 30

List of Figures

1 labelInTOC 6
2 labelInTOC 8
3 labelInTOC 13
4 labelInTOC 14
5 labelInTOC 21

List of Tables

1 Payoff Matrix for Secure Entry 17

2 Some Common Link Functions and Their Inverses 30

1 Introduction

Linear Models are among the most widely used hypotheses in Machine Learning. Their usefulness comes from several characteristics of linear models:

1. Linear models are the simplest to implement and interpret. They are more often than not extremely efficient.
2. Linear models often give rise to closed form solutions which minimizes the need for numerical and other approximations.
3. Linear Models invariably generalize very well owing to their simplicity.

Assume the data domain is $\mathcal{R}^d$. Let $\{\phi_1(\boldsymbol{x}),\ldots,\phi_k(\boldsymbol{x})\}$ be a set of basis functions (of a vector space of functions) that together map raw data vectors $\boldsymbol{x} \in \mathcal{R}^d$ to homogeneous *feature vectors* $\boldsymbol{y} \in \mathcal{R}^k$ as

$$\boldsymbol{x} \to \boldsymbol{y} \equiv (1, \phi_1(\boldsymbol{x}),\ldots,\phi_k(\boldsymbol{x})) \in \mathbb{R}^k$$

Note that here (unlike our discussion on Dimensionality Reduction) k can even be much larger than d.

A linear model $h_{\boldsymbol{w}}(\boldsymbol{x}) : \mathcal{R}^d \to \mathcal{R}$ is one that is parametrized by a parameter vector $\boldsymbol{w} \in \mathcal{R}^{k+1}$ and computes the weighted sum

$$\boldsymbol{w}^T.\boldsymbol{y} = \sum_{i=1}^{k+1}(w_i y_i) = w_1 + \sum_{i=1}^{k+1}(w_i \phi_i(\boldsymbol{x}))$$

for any given data point $\boldsymbol{x}$. Notice that the 'linearity' here is on the parameters $\boldsymbol{w}$ and not on the attributes of the data point — in other words these models are such that for any two parameter vectors $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$

$$h_{(\boldsymbol{w}_1+\boldsymbol{w}_2)}(\boldsymbol{x}) = h_{\boldsymbol{w}_1}(\boldsymbol{x}) + h_{\boldsymbol{w}_2}(\boldsymbol{x})$$

The functions $\phi_i(\boldsymbol{x})$ can be potentially non-linear functions in the components of $\boldsymbol{x}$ but what matters for all the linear methods to go through is that after a transformation, the transformed 'features' are combined linearly using the model parameters as the weights. This has the advantages of being able to arrive at non-linear models using linear methods and consequently exploiting all the advantages of a linear model, but in a possibly more complex (higher dimensional) domain.

Given a set of training data point, value pairs $\{(\boldsymbol{x}_i, v_i),\ldots,(\boldsymbol{x}_n, v_n)\}$, $\boldsymbol{x}_i \subset \mathcal{R}^d$ and a loss function $\mathcal{L}((\boldsymbol{x}, v), h_{\boldsymbol{w}}(\boldsymbol{x}))$ we try to find the weight vector $\boldsymbol{w}$ to minimize the risk

$$\int_{\boldsymbol{x}} \mathcal{L}((\boldsymbol{x}, v), h_{\boldsymbol{w}}(\boldsymbol{x}))\, p_{\mathcal{D}}(\boldsymbol{x})\, d\boldsymbol{x}$$

The ERM Induction principle is often used to find the hypothesis that minimizes this risk, where the hypothesis $\boldsymbol{w}^*$ chosen is

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}((\boldsymbol{x}_i, v_i), h_{\boldsymbol{w}}(\boldsymbol{x}_i))$$

This setting is that of a generic linear regression problem. Linear Classification problems are those where the values from a fixed set of labels (in particular ±1 for binary classification). We will explore some of the implications and uses of non-transformations prior to applying a linear model for learning, in the section below. The subsequent sections will explore specific linear regression / classification algorithms.

2 Transformations Using Non-Linear Basis Functions

We explore the non-linear feature transformations a some detail here to highlight the usefulness of such transformations. As an example see Figure 1 where there are two sets of data points — green and red. The separator happens to be a circle with the equation $x^2 + y^2 = 9$. Cleary the red points are the ones for which $x^2 + y^2 > 9$ and the green points are those with $x^2 + y^2 < 9$. We can transform all the points to a feature space where the axes are $t = x^2$ and $s = y^2$ — t, s are the features. Then all the red points will map to one side of the line $t + s = 9$ and the green ones will map to the other side. Therefore in the feature space the separator is the line $t + s = 9$. For instance we can now use the perceptron algorithm to compute a linear separator in the feature space.

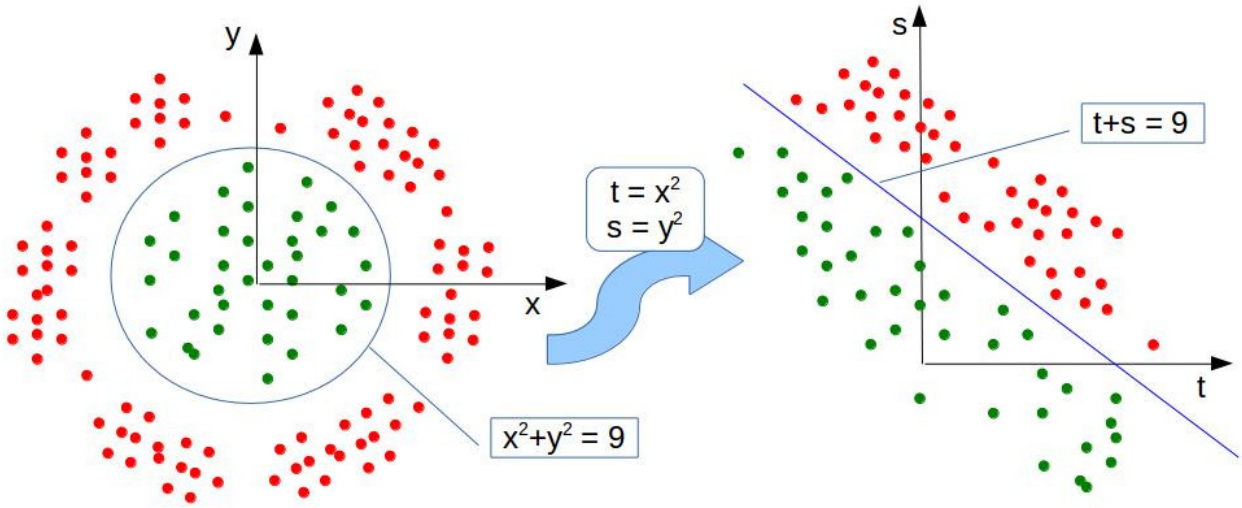

Figure 1: Circular Separator Transformed to a Linear Separator

For this discussion let us restrict ourselves to polynomial transformations. Suppose we want to accommodate all possible quadratic functions of the attributes of the original data points then the basis functions would include all pairwise combinations of the original attributes. Suppose the original data points are $\boldsymbol{x} \in \mathcal{R}^d$ then the basis functions form the set

$$\{1, x_1, \ldots, x_d, x_i x_j \; 1 \leq i, j \leq d\}$$

When $d = 2$, the basis functions are $\{1, x_1.x_2, x_1^2, x_1 x_2, x_2^2\}$. For the example in Figure 1 every raw data point (x, y) gets transformed to the 6-dimensional homogeneous feature vector $(1, 0, 0, x^2, 0, y^2)$. So a point in the quadratic feature space $\mathcal{F}^2$ is represented as a 7-tuple

$(\vartheta_1, \vartheta_2, \vartheta_3, \vartheta_4, \vartheta_5, \vartheta_6, \vartheta_7)$. Therefore the circle $x^2+y^2=9$ in the original raw attribute space would get transformed to the line $\vartheta_5+\vartheta_7=9$ in the feature space. Figure 1 shows just the ϑ_5-ϑ_7 plane on which the separator in the feature space lies.

Exercise 1 *Specify the equation of the line $\mathcal{F}^2$ that the following curves in the data attribute space will transform to:*

1. *The circle* $(x-3)^2+(y-2)^2=16$
2. *The ellipse* $\frac{(x-3)^2}{5}+\frac{(y-2)^2}{6}=10$

Exercise 2 *Derive the dimensionality of the feature space $\mathcal{F}^k$ of a transform that consists of degree k polynomials of attributes of points in $\mathcal{R}^d$. The dimensionality of the feature space will obviously be a function $\mathrm{d}(d,k)$ of d and k.*

In the rest of this document where we discuss specific linear models, we assume that the original data points have already been transformed into an appropriate feature space. The data vectors that we consider henceforth are vectors in the feature space.

3 Linear Regression

Linear Regression is the problem of finding the 'best fit' hyperplane $\boldsymbol{w}^* \in \mathcal{R}^{d+1}$ through a set of point-value pairs $\mathcal{D}_n = \{(\boldsymbol{x}_i, v_i), \ldots, (\boldsymbol{x}_n, v_n)\}$, $\boldsymbol{x}_i \subset \mathbb{R}^d$ in a d-dimensional homogenized feature space. In this case the hypotheses are of the form $h_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{w}^T.\boldsymbol{x}$. The regression problem is therefore that of finding the vector $\boldsymbol{w}^*$ such that

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(v_i, \boldsymbol{w}^T.\boldsymbol{x}_i)$$

for an appropriately chosen loss function $\mathcal{L}$. The loss function characterizes how far away from the model hyperplane does a point lie. The given data set $\mathcal{D}_n$ as defined above is assumed for all versions of linear regression discussed below.

3.1 Least Squares Regression

Least Squares Linear Regression is one that minimizes the square-error loss function. The best hypothesis generated by least-squares linear regression is therefore given by

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \frac{1}{n}\sum_{i=1}^{n} |v_i - \boldsymbol{w}^T.\boldsymbol{x}_i|^2 = \arg\min_{\boldsymbol{w}} \frac{1}{n}\sum_{i=1}^{n} (v_i - \boldsymbol{w}^T.\boldsymbol{x}_i)^2$$

Imagine an extra dimension for the value at each of the points. The value given for the data point $\boldsymbol{x}_1$ is v_i. The i^{th} point is plotted with $\boldsymbol{x}_i$ on the first $(d+1)$ dimensions and v_i along the

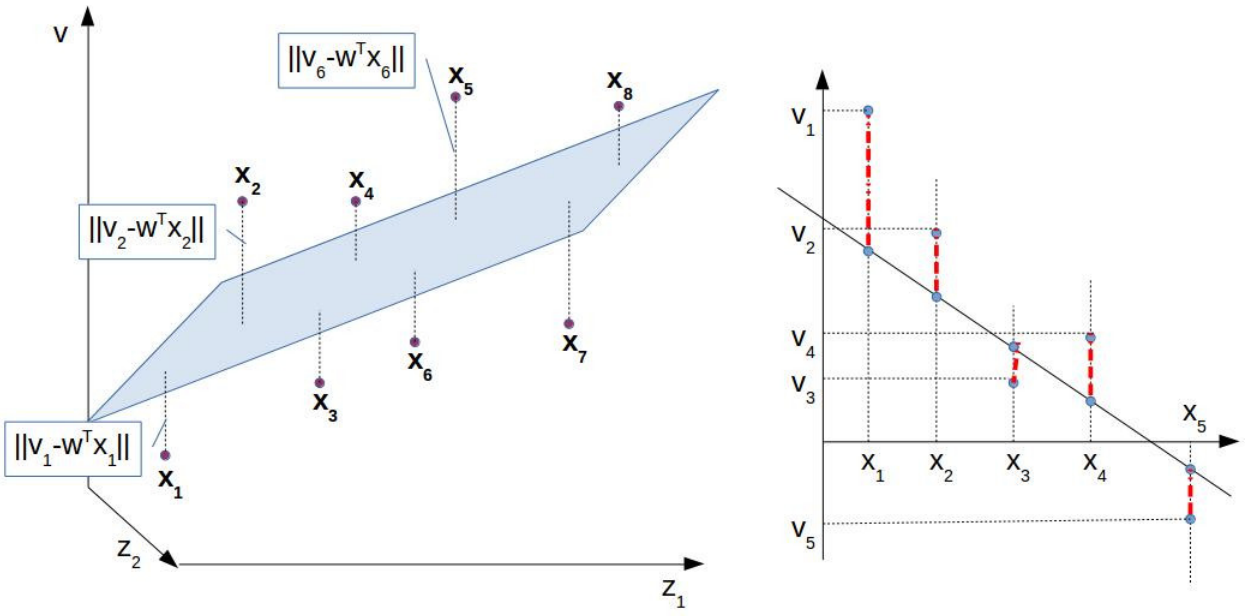

Figure 2: Linear Regression — (i) General (ii) $d = 1$

line from $\boldsymbol{x}_i$ orthogonal to the first $(d+1)$ dimensions. The model predicts the value $\boldsymbol{w}^T.\boldsymbol{x}_i$ along this line. The square of the absolute difference between the two is the value of the loss function at $\boldsymbol{x}_i$. See Figure 2 for an illustration.

Let $\boldsymbol{v}$ be the vector with the v_is as its components and let X be the matrix with the vectors $\boldsymbol{x}_i$s as its rows. Now $\sum_{i=1}^{n}(v_i - \boldsymbol{w}^T.\boldsymbol{x}_i)^2$ can be thought of as the Euclidean norm of the vector $\boldsymbol{z}$ where the components of $\boldsymbol{z}$ are $z_i = (v_i - \boldsymbol{w}^T.\boldsymbol{x}_i)$. So observing that being a scalar $\boldsymbol{w}^T.\boldsymbol{x}_i = \boldsymbol{x}_i^T.\boldsymbol{w}$, it is easy to verify that $\boldsymbol{z} = (\boldsymbol{v} - X\boldsymbol{w})$. Continuing from the earlier expression for the regression hypothesis we get

$$\begin{aligned}
\boldsymbol{w}^* &= \arg\min_{\boldsymbol{w}} \frac{1}{n}\sum_{i=1}^{n}(v_i - \boldsymbol{w}^T.\boldsymbol{x}_i)^2 = \arg\min_{\boldsymbol{w}} \frac{1}{n}||(\boldsymbol{v} - X\boldsymbol{w})||^2 \\
&= \arg\min_{\boldsymbol{w}} \frac{1}{n}(\boldsymbol{v} - X\boldsymbol{w})^T(\boldsymbol{v} - X\boldsymbol{w}) \\
&= \arg\min_{\boldsymbol{w}} \frac{1}{n}\left(\boldsymbol{v}^T\boldsymbol{v} + \boldsymbol{w}^T X^T X\boldsymbol{w} - 2\boldsymbol{w}^T X^T\boldsymbol{v}\right) \quad \left(\boldsymbol{w}^T X^T\boldsymbol{v} \text{ is a scalar}\right)
\end{aligned}$$

Equating the $\boldsymbol{w}$ gradient of the last expression in brackets to zero we get

$$X^T X\boldsymbol{w} = X^T\boldsymbol{v} \tag{1}$$

Note that for any vector $\boldsymbol{w} \in \mathcal{R}^d$ and matrix $A^{(d\times d)}$

$$\frac{d}{d\boldsymbol{w}}(\boldsymbol{w}^T A\boldsymbol{w}) = \frac{d}{d\boldsymbol{w}}\left(\sum_{i=1}^{d} a_{ij}w_i w_j\right) = \boldsymbol{u}$$

where $u_i = \sum_{j=1}^{d}(a_{ij} + a_{ji})w_j$ — u_i is obtained by differentiating the expression in brackets with respect to w_i. Therefore

$$\frac{d}{d\boldsymbol{w}}(\boldsymbol{w}^T A\boldsymbol{w}) = (A + A^T)\boldsymbol{w}$$

In our case $A = A^T = X^T X$. Also the matrix $(X^T X)$ is more-often-than-not in practice, invertible — invertibility of $X^T X$ requires d linearly independent rows, formed from .

Assuming $X^T X$ is invertible we get

$$\boldsymbol{w}^* = (X^T X)^{-1} X^T \boldsymbol{v} \tag{2}$$

This is the closed form solution for the best fit regression line given by the coefficient vector $\boldsymbol{w}^*$. It is expected that $\boldsymbol{x}_i^T . \boldsymbol{w}^* \approx v_i$. Therefore we expect that

$$\hat{\boldsymbol{v}} = X(X^T X)^{-1} X^T \boldsymbol{v} = X\boldsymbol{w}^* \approx \boldsymbol{v}$$

This implies that the *Hat Matrix* $\hat{H}^{(n\times n)} = X(X^T X)^{-1} X^T \approx I^{(n\times n)}$. The matrix $(X^T X)^{-1} X^T$ is therefore often referred to as the *pseudo-inverse* of X. The Hat matrix in fact behaves remarkably like an identity matrix.

Exercise 3 *Show the following properties of the Hat matrix*

1. $\hat{H}$ *is symmetric.*
2. $\hat{H}^k = \hat{H}$ *and* $(I - \hat{H})^k = (I - \hat{H})$ *for any positive integer* k *where* I *is the identity matrix of order* $(n \times n)$.
3. *Show that for any two matrices* $A^{(p\times q)}, B^{(q\times p)}$, $trace(AB) = trace(BA)$. *Hence show that* $trace(\hat{H}) = d$.

3.2 Equivalence of MLE with Gaussian Errors and Least Squares

Recall the discussion on Maximum Likelihood Estimation in the document on 'Ml Problem Formulation' (Sections 3.3, 3.4 and 4.5.1). We carry out a similar exercise here and surprisingly it turns out that the MLE estimates assuming Gaussian errors are indeed identical to those arrived at above using least squares. We assume a linear model but recognize the fact that the data points given are noisy. Assuming a Gaussian noise we therefore consider the following hypothesis class for the probability density of the underlying input space

$$p_{\mathcal{D}}(v|\boldsymbol{x}) = h_{(\boldsymbol{w},\sigma)}(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{w}^T . \boldsymbol{x}, \sigma^2)$$

The hypotheses are Gaussian distributions with $\boldsymbol{w}^T . \boldsymbol{x}$ as the mean. We determine $\boldsymbol{w}$ to maximize the likelihood (probability) that the hypothesis generates the given values for the data points. As is usually the case, we minimize the negative log-likelihood instead of maximizing the likelihood. Therefore denoting the probability of generating $\mathcal{D}_n$ given the hypothesis $h_{(\boldsymbol{w},\sigma)}$

as $p(\mathcal{D}_n \mid \boldsymbol{w}, \sigma)$ we have

$$\begin{aligned}
\boldsymbol{w}^* &= \arg\max_{\boldsymbol{w}} \ln p(\mathcal{D}_n \mid \boldsymbol{w}, \sigma) = \arg\max_{\boldsymbol{w}} \ln \left(p\left(\bigwedge_{i=1}^{n} ((\boldsymbol{x}_i, v_i) \mid \boldsymbol{w}, \sigma) \right) \right) \\
&= \arg\max_{\boldsymbol{w}} \ln \prod_{i=1}^{n} p((\boldsymbol{x}_i, v_i) \mid \boldsymbol{w}, \sigma) \\
&= \arg\max_{\boldsymbol{w}} \sum_{i=1}^{n} \ln p((\boldsymbol{x}_i, v_i) \mid \boldsymbol{w}, \sigma) \\
&= \arg\max_{\boldsymbol{w}} \sum_{i=1}^{n} \ln p(h_{(\boldsymbol{w},\sigma)}(\boldsymbol{x}_i) = v_i) \\
&= \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} -\ln \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(v_i - \boldsymbol{w}^T.\boldsymbol{x}_i)^2}{2\sigma^2} \right) \\
&= \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} (v_i - \boldsymbol{w}^T.\boldsymbol{x}_i)^2
\end{aligned}$$

which is the same as the minimum square error regression criterion.

3.3 Robust Linear Regression

Square Error minimization penalizes deviations quadratically. Therefore points far from the model hypothesis will have a larger effect on the choice of the model. So it is natural that least squares regression is very sensitive to outliers. One way to interpret this is that least squares minimization tries 'hard' to be as close to all the points as possible (overfitting) and has a tendency to become more 'complicated' than it needs to be in the presence of outliers. Going 'out of the way' to take care of the outliers will result in loss of generalizability. We will explore a few ways to control this phenomenon by imposing some regularization strategies that keep the model complexity under check. These are referred to as *Robust Regression* strategies.

3.3.1 Laplace Regression

One way to achieve robustness is to assume a *heavy tail* distribution as opposed to a thin tail distribution such as the Gaussian. Heavy tail distributions allow outliers with higher probability and hence consider them more 'normal' than the thin tail distributions do. Hence the tendency to modify the mean to accommodate the outliers is less with the use of heavy tail distributions. A Laplace Distribution (with parameters μ, b) is a well known heavy tail distribution and is given by

$$L(v \mid \mu, b) = \frac{1}{2b} \exp\left(-\frac{|v - \mu|}{b} \right)$$

We therefore take $L(v \mid \boldsymbol{w}^T.\boldsymbol{x}, b)$, the Laplace Distribution with mean at $\boldsymbol{w}^T.\boldsymbol{x}$ as the hypothesis distribution and carry out a negative log likelihood minimization to arrive at the model $\boldsymbol{w}^*$. The negative log likelihood, for a fixed b in this case is

$$n \ln(2b) + \frac{1}{b} \sum_{i=1}^{n} |v_i - \boldsymbol{w}^T.\boldsymbol{x}_i|$$

Therefore

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} |v_i - \boldsymbol{w}^T.\boldsymbol{x}_i|$$

Absolute value $|.|$ is not a linear operator. So minimizing the above objective function directly is not easy. However in this case one can employ a neat trick (called the *split variable trick*) that can reduce the above to a linear programming problem. Let us denote $(v_i - \boldsymbol{w}^T.\boldsymbol{x}_i)$ as the residue r_i. We introduce two other variables $r_i^+ \geq 0$ and $r_i^- \geq 0$ for each r_i such that $r_i = r_i^+ - r_i^-$. Now the problem of minimizing $\sum_{i=1}^{n} |r_i|$ is equivalent to the linear program

$$\begin{aligned} \min \sum_{i=1}^{n} (r_i^+ + r_i^-) \\ r_i^+, r_i^- \geq 0, \quad 1 \leq i \leq n \\ r_i^+ - r_i^- + \boldsymbol{w}^T.\boldsymbol{x}_i = v_i, \quad 1 \leq i \leq n \end{aligned}$$

To see why the linear program actually solves the problem of minimizing the sum of absolute values, we first observe that any optimal solution to the linear program must be such that for any $1 \leq i \leq n$ either $r_i^+ = 0$ or $r_i^- = 0$. Otherwise suppose for some i, it is true that $r_i^+ > 0$ and $r_i^- > 0$. Let $\delta = \min(r_i^+, r_i^-) > 0$. Let $s_i^+ = r_i^+ - \delta$ and $s_i^- = r_i^i - \delta$. Then $(s_i^+ - s_i^-) = (r_i^+ - r_i^-)$. Therefore replacing r_i^+ by s_i^+ and r_i^- by s_i^- will give us another feasible solution to the linear program. However $(s_i^+ + s_i^-) = (r_i^+ + r_i^- - 2\delta)$, implying that $(s_i^+ + s_i^-) < (r_i^+ + r_i^-)$. In this case the original solution couldn't have been optimal. Now for the optimal solution if $r_i > 0$ then $r_i^+ = r_i$, $r_i^- = 0$ and in turn we have $|r_i| = r_i^+ + r_i^-$, Similarly if $r_i < 0$ then $r_i^+ = 0$, $r_i^- = -r_i$ and again we have that $|r_i| = r_i^+ + r_i^-$.

3.3.2 Ridge Regression

Another way to control the model complexity is to add a penalization term accounting for the model complexity. The complexity of the linear hypothesis is measured as $||\boldsymbol{w}||^2$. The loss function therefore is taken as

$$\mathcal{L}(v_i, \boldsymbol{w}^T.\boldsymbol{x}_i) = (v_i - \boldsymbol{w}^T.\boldsymbol{x}_i)^2 + \frac{\lambda}{n}||\boldsymbol{w}||^2$$

for some learning rate λ that controls that degree of penalization for the complexity of the model. The ridge regression problem can thus be formulated as that of finding the $\boldsymbol{w}^*$, for an appropriately chosen λ such that

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \left(\sum_{i=1}^{n} (v_i - \boldsymbol{w}^T.\boldsymbol{x}_i)^2 + \lambda.||\boldsymbol{w}||^2 \right)$$

As was done for the least squares regression we can rewrite this as

$$\begin{aligned} \boldsymbol{w}^* &= \arg\min_{\boldsymbol{w}} ((\boldsymbol{v} - X\boldsymbol{w})^T(\boldsymbol{v} - X\boldsymbol{w}) + \lambda.\boldsymbol{w}^T.\boldsymbol{w}) \\ &= \arg\min_{\boldsymbol{w}} (\boldsymbol{v}^T.\boldsymbol{v} + \boldsymbol{w}^T(\lambda I + X^T X)\boldsymbol{w} - 2\boldsymbol{w}^T X^T \boldsymbol{v})) \end{aligned}$$

Again differentiating the RHS with respect to $\boldsymbol{w}$, equating it to zero and assuming $(\lambda I + X^T X)$ is invertible we get

$$\boldsymbol{w}^* = (\lambda I + X^T X)^{-1} X^T \boldsymbol{v}$$

In practice it is also the case that for suitable choices of λ, $(\lambda I + X^T X)$ is more likely to be invertible than $X^T X$ and that makes ridge regression also more computationally stable than least squares regression.

Another way to interpret ridge regression is that to control the complexity of the coefficient vector $\boldsymbol{w}$ we keep the probability of such unwieldy coefficients low. We can therefore adopt the minimum negative log-likelihood method where the coefficient vector $\boldsymbol{w}$ is itself normally distributed at mean $\mathbf{0}$ making the probability of large vectors negligibly small. The formulation therefore is that we need to find a $\boldsymbol{w}^*$ such that

$$\begin{aligned}\boldsymbol{w}^* &= \arg\min_{\boldsymbol{w}} \left(\sum_{i=1}^{n} \log \mathcal{N}(v_i \mid \boldsymbol{w}^T.\boldsymbol{x}_i, \sigma^2) + \sum_{i=1}^{d} \log \mathcal{N}(w_i \mid 0, \tau^2) \right) \\ &= \arg\min_{\boldsymbol{w}} \frac{1}{2\sigma^2} \left(\sum_{i=1}^{n} (v_i - \boldsymbol{w}^T.\boldsymbol{x}_i)^2 + \frac{\sigma^2}{\tau^2} \sum_{i=1}^{d} w_i^2 \right) \\ &= \arg\min_{\boldsymbol{w}} \left(\sum_{i=1}^{n} (v_i - \boldsymbol{w}^T.\boldsymbol{x}_i)^2 + \lambda.||\boldsymbol{w}||^2 \right)\end{aligned}$$

where $\lambda = \sigma^2/\lambda^2$. Note that this is simply the ridge regression formulation that we stated at the beginning of this section.

3.3.3 Lasso Regression

4 Logistic Regression

Strictly speaking this is a binary classification method, though it resembles regression in its approach (hence the name Logistic Regression). We are given n points and their binary (± 1) labels $\{(\boldsymbol{x}_1, v_1), \ldots, (\boldsymbol{x}_n, v_n)\}$ where $\boldsymbol{x}_1 \in \mathcal{R}^d$, $v_i \in \pm 1$. We seek a linear model $\boldsymbol{w}^T.\boldsymbol{x}$ with parameters $\boldsymbol{w}$ along with a thresholding function $\theta(y)$ that returns the probability with which a given value y will get the label +1. The loss function would be the negative log-likelihood of the model returning the correct observed value v at any given point $\boldsymbol{x}$ after thresholding. We want to minimize the expected loss.

The loss at a point $\boldsymbol{x}$ with label v, given the model $\boldsymbol{w}$ is

$$\mathcal{L}((\boldsymbol{x}, v), (\boldsymbol{w}, \theta)) = \begin{cases} -\ln \theta(\boldsymbol{w}^T.\boldsymbol{x}) & \text{if } v = +1 \\ -\ln(1 - \theta(\boldsymbol{w}^T.\boldsymbol{x})) & \text{Otherwise} \end{cases}$$

The idea is to have a probabilistic classifier that provides a smooth transition from the points with label +1 to those with label −1. This is particularly useful when the data attributes available are not complete and/or the classification results in a safety-critical decision (for instance suspicion of an impending heart failure). In such cases the system gives a probabilistic estimate of the classification label letting the final categorization to be carried out with an expert intervention to interpret the analysis in an appropriate context. The thresholding function typically provides the smooth transition from +1 to −1. The following are two common examples of thresholding functions:

1. **Logistic Function**: The Logistic function is defined as

$$\lg(x) = \frac{e^x}{e^x + 1} = \frac{1}{1 + e^{-x}}$$

It is easy to show also that $\lg(-x) = 1 - \lg(x)$, $\lg(-\infty) = 0$, $\lg(0) = 1/2$ and $\lg(\infty) = 1$. It is the inverse of the *Logit* function defined as

$$\text{logit}(x) = \ln\left(\frac{x}{1-x}\right)$$

2. **Tanh Function**: The tanh function is defined as

$$\tau(x) = \frac{e^x - e^{-x}}{e^x + e^x}$$

The class of thresholding functions like the Logistic function and the Tanh function is called the class of *Sigmoid* functions. Sigmoid function is in general a function that varies smoothly (differentiable) from l to h, $l < h$ as the argument varies from $-\infty$ to $+\infty$. The slope of a sigmoid function resembles a bell curve with the peak at $x = 0$. The horizontal lines at $y = l$ and $y = h$ are asymptotic to the sigmoid curve at $x = -\infty$ and $x = +\infty$ respectively. See Figures 3, 4 for illustrations of several sigmoid functions.

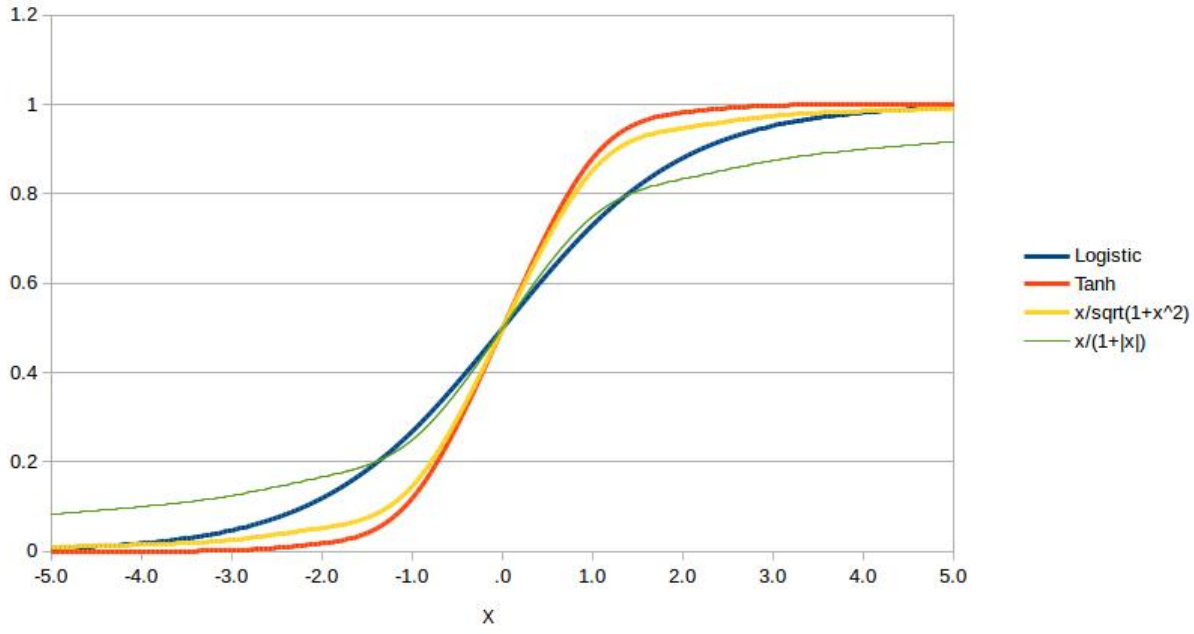

Figure 3: Sigmoid Functions. Functions other than the Logistic Function, have been scaled appropriately to the range $[0,1]$

Exercise 4 *Show that with an appropriate transformation the logistic function can be transformed into the tanh function and vice versa.*

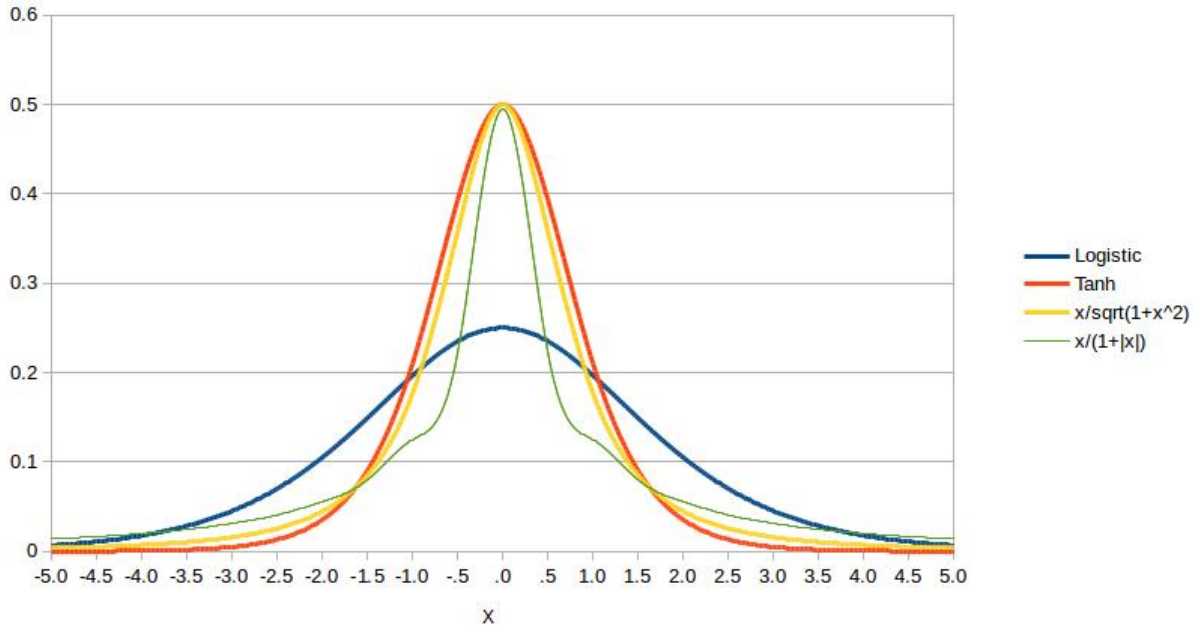

Figure 4: Slopes of Sigmoid Functions

Using the logistic function $\lg(x)$ as the thresholding function $\theta(x)$, it is easy to verify that the loss function is simply $-\ln\lg(v_i \boldsymbol{w}^T.\boldsymbol{x}_i) = -\ln\lg(\zeta_i)$ at any point $\boldsymbol{x}_i$ with label v_i where $\zeta_i = v_i \boldsymbol{w}^T.\boldsymbol{x}_i$ is the signed distance of the point $\boldsymbol{x}_i$ from the model hyperplane with coefficients $\boldsymbol{w}$. Again resorting to ERM we try to minimize the empirical error to arrive at the best fit hypothesis $\boldsymbol{w}^*$ as below

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \frac{1}{n}\sum_{i=1}^{n} -\ln\left(\frac{e^{\zeta_i}}{1+e^{\zeta_i}}\right) = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} \ln\left(1+e^{-\zeta_i}\right)$$

where the empirical error is given by

$$E^n_{\boldsymbol{w}} = \frac{1}{n}\sum_{i=1}^{n} -\ln\left(\frac{e^{\zeta_i}}{1+e^{\zeta_i}}\right) = \frac{1}{n}\sum_{i=1}^{n} \ln\left(1+e^{-\zeta_i}\right)$$

In a way we are minimizing the empirical error with the loss function at each point given by $\ln\left(1+e^{-\zeta_i}\right)$. As we would expect, the error gets smaller as the term ζ_i becomes a large positive number — these are precisely the points for which the classification is more-or-less unambiguous, points far away from the separator and on the correct side. There is no direct way to carry out this minimization other than resorting to one of the known non-linear optimization methods such as gradient descent. The gradient vector in this case will be

$$\begin{aligned}\nabla E^n_{\boldsymbol{w}} &= \frac{1}{n}\sum_{i=1}^{n} \frac{-v_i \boldsymbol{x}_i.e^{-\zeta_i}}{1+e^{-\zeta_i}} = \frac{1}{n}\sum_{i=1}^{n} \frac{-v_i \boldsymbol{x}_i}{1+e^{\zeta_i}} = \frac{1}{n}\sum_{i=1}^{n} -v_i \lg(-\zeta_i)\boldsymbol{x}_i \\ &= \frac{1}{n}\sum_{i=1}^{n} X^T.\boldsymbol{\mu}\end{aligned}$$

where $\boldsymbol{\mu}$ is the column vector $(-(v_1 \lg(-\zeta_1)),\ldots,-(v_n \lg(-\zeta_n))$. We can also compute the Hessian Matrix for the $E^n_{\boldsymbol{w}}$ — this is the multivariate second order gradient. The Hessian will be defined as

$$H(E^n_{\boldsymbol{w}}) = \begin{bmatrix} \frac{\partial^2 E^n_{\boldsymbol{w}}}{\partial w_1^2} & \frac{\partial^2 E^n_{\boldsymbol{w}}}{\partial w_1 \partial w_2} & \cdots & \frac{\partial^2 E^n_{\boldsymbol{w}}}{\partial w_1 \partial w_d} \\ \frac{\partial^2 E^n_{\boldsymbol{w}}}{\partial w_2 \partial w_1} & \frac{\partial^2 E^n_{\boldsymbol{w}}}{\partial w_2^2} & \cdots & \frac{\partial^2 E^n_{\boldsymbol{w}}}{\partial w_2 \partial w_d} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 E^n_{\boldsymbol{w}}}{\partial w_d \partial w_1} & \frac{\partial^2 E^n_{\boldsymbol{w}}}{\partial w_d \partial w_2} & \cdots & \frac{\partial^2 E^n_{\boldsymbol{w}}}{\partial w_d^2} \end{bmatrix}$$

We can compute the elements of the Hessian $H(E^n_{\boldsymbol{w}})$ as follows:

$$\begin{aligned} \frac{\partial^2 E^n_{\boldsymbol{w}}}{\partial w_i \partial w_j} &= \frac{1}{n}\sum_{k=1}^{n} -v_k x_{ki} \frac{\partial \lg(-\zeta_k)}{\partial w_j} = \frac{1}{n}\sum_{k=1}^{n} -v_k x_{ki} \frac{\partial (1+e^{\zeta_k})^{-1}}{\partial w_j} \\ &= \frac{1}{n}\sum_{k=1}^{n} \frac{v_k x_{ki} e^{\zeta_k}}{(1+e^{\zeta_k})^2} \frac{\partial \zeta_k}{\partial w_j} \\ &= \frac{1}{n}\sum_{k=1}^{n} v_k^2 x_{ki} x_{kj} \frac{e^{\zeta_k}}{1+e^{\zeta_k}} \frac{1}{1+e^{\zeta_k}} \\ &= \frac{1}{n}\sum_{k=1}^{n} \lg(\zeta_k)(1-\lg(\zeta_k)) x_{ki} x_{kj} \end{aligned}$$

Therefore the Hessian matrix can now be written as

$$\begin{aligned} H(E^n_{\boldsymbol{w}}) &= \frac{1}{n}\sum_{k=1}^{n} \lg(\zeta_k)(1-\lg(\zeta_k)) \boldsymbol{x}_k \boldsymbol{x}_k^T \\ &= X^T D X \end{aligned}$$

where $D^{(n\times n)}$ is the diagonal matrix whose k^{th} element is $\lg(\zeta_k)(1-\lg(\zeta_k))$. The matrix $\boldsymbol{x}\boldsymbol{x}^T$ for any non-zero vector $\boldsymbol{x}$ is positive definite since

$$\forall\, \boldsymbol{y} \neq \boldsymbol{0}:\ \boldsymbol{y}^T \boldsymbol{x}\boldsymbol{x}^T \boldsymbol{y} = (\boldsymbol{x}^T \boldsymbol{y})^T (\boldsymbol{x}^T \boldsymbol{y}) = \|\boldsymbol{x}^T \boldsymbol{y}\|^2 > 0$$

Also any positive weighted sum of positive definite matrices is also positive definite. It is now easy to see that $H(E^n_{\boldsymbol{w}})$ is positive definite — note that $\lg(\zeta_k) \in [0,1]$. Therefore $E^n_{\boldsymbol{w}}$ is a convex objective function, with a unique global minimum and no other local minima. The gradient descent algorithm in this case would work as shown in the pseudocode given below.

Logistic Regression can also suffer from overfitting issues. For instance if the data is separable and a separator $\boldsymbol{w}$ is found then for every data point $\boldsymbol{x}_i$, $\zeta_i = v_i \boldsymbol{w}^T.\boldsymbol{x}_i > 0$ and since the objective function for Logistic Regression is

$$E^n_{\boldsymbol{w}} = \frac{1}{n}\sum_{i=1}^{n} \ln\left(1+e^{-\zeta_i}\right)$$

Assuming the coefficients of all the data points are positive (a simple origin-shift can ensure this), we can make the error arbitrarily small by simply 'scaling' $\boldsymbol{w}$ — $e^{-\zeta_i}$ can be made arbitrarily

Algorithm 1 Logistic Regression

1: **Initialize**

1. An appropriately chosen learning rate η
2. Termination thresholds for the
   (a) minimum norm ϵ_g of the gradient $\nabla E^n_{\boldsymbol{w}_t}$
   (b) minimum change ϵ_v in the empirical error between two iterations
   (c) maximum number of iterations T
3. A random vector $\boldsymbol{w}_0$ — each coefficient chosen from a Gaussian distribution with zero mean and a small variance.
4. $t = 0$

2: **while** $t = 0$ or $(\|\nabla E^n_{\boldsymbol{w}_t}\|^2 > \epsilon_g$ and $|E^n_{\boldsymbol{w}_t} - E^n_{\boldsymbol{w}_{t-1}}| > \epsilon_v$ and $t < T)$ **do**

3: With $\zeta^t_i = v_i \boldsymbol{w}^T_t . \boldsymbol{x}_i$ compute the negative gradient vector $\boldsymbol{g}_t$ at $\boldsymbol{w}_t$

$$\boldsymbol{g}_t = -\frac{\nabla E^n_{\boldsymbol{w}_t}}{\|\nabla E^n_{\boldsymbol{w}_t}\|} = \frac{\sum_{i=1}^n v_i \, \mathsf{s}(-\zeta^t_i)\boldsymbol{x}_i}{\|\sum_{i=1}^n v_i \, \mathsf{s}(-\zeta^t_i)\boldsymbol{x}_i\|}$$

4: Update $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \eta \boldsymbol{g}_t$

5: $t \leftarrow t + 1$

6: **end while**

7: Return $\boldsymbol{w}^* \leftarrow \boldsymbol{w}_t$

small by replacing $\boldsymbol{w}$ by $c\boldsymbol{w}$ for larger and larger values of $c > 0$. As the model gets scaled it is in a way becoming increasingly sure of itself in classifying the data points — note that the scaling results in the sigmoid function becoming increasingly close to zero or one and nothing in between. This is a form of overfitting.

Regularization strategy similar to the one we discussed for regression will also work in this case. The modified empirical risk will now be:

$$E^n_{\boldsymbol{w}} = \frac{1}{n}\sum_{i=1}^{n} \ln\left(1 + e^{-\zeta_i}\right) + \lambda \|\boldsymbol{w}\|^2$$

which we need to minimize.

Exercise 5 *Show that the Hessian for the modified empirical risk function continues to be positive semi-definite. Use this to derive an appropriate optimization strategy for minimizing* $E^n_{\boldsymbol{w}}$.

4.1 Decision Making Using Logistic Regression

We discuss one way to take decisions based on Logistic Regression. Logistic regression models will give out a probability that a given test point belongs to class +1. The question we are trying to address here is how does one convert the probability into a firm decision (we need to convert the probability into an unambiguous classification). We illustrate the idea through an example.

Decision	*Actual Status* **Genuine**	**Intruder**
Allow	0	c_{10}
Restrict	c_{01}	0

Table 1: Payoff Matrix for Secure Entry

Consider a security system that regulates authorized entry into a restricted area with some biometric identification. We construct what is called a *payoff matrix* to make the tradeoffs involved in the different scenarios that can arise. The leftmost column in Table 1 shows the possible decisions that could be taken by the system and the top row shows the status of the visitor. The table captures the cost of every combination of the actual status of the visitor vs the decision taken by the authentication system. There is no cost if the decision matches the actual status of the visitor. However if there is a mismatch then there is a cost involved. Often the mismatch costs are highly asymmetric. The cost of allowing an intruder for instance is typically much more than the cost of not allowing a genuine visitor — $c_{10} \gg c_{01}$.

In this framework we decide on a threshold probability ρ based on the cost matrix and decide to allow the visitor if the output of the system (the probability of the visitor being genuine according to the system) is more than ρ. The visitor is not allowed otherwise (may be the visitor will have to get additional clearances). The question of how we should be setting ρ based on the payoff matrix remains. We set ρ in such a way that if the probability output by the regression model is at least ρ then the cost of accepting a visitor is less than the cost of rejecting the visitor. Suppose for a visitor $\boldsymbol{v}$ the system outputs a probability $\theta(\boldsymbol{v})$. Note that $\theta(\boldsymbol{v})$ is the probability that the visitor is genuine.

$$\text{Cost of Accepting}(\boldsymbol{v}) = \theta(\boldsymbol{v}).0 + (1-\theta(\boldsymbol{v}))c_{10}$$
$$\text{Cost of Rejecting}(\boldsymbol{v}) = \theta(\boldsymbol{v}).c_{01} + (1-\theta(\boldsymbol{v})).0$$

We require that Cost of Accepting$(\boldsymbol{v})$ < Cost of Rejecting$(\boldsymbol{v})$. This translates to

$$(1-\theta(\boldsymbol{v}))c_{10} < \theta(\boldsymbol{v}).c_{01} \Rightarrow \theta(\boldsymbol{v}) > \frac{c_{10}}{c_{10}+c_{01}}$$

We therefore set the threshold probability as

$$\rho = \frac{c_{10}}{c_{10}+c_{01}}$$

. This is an example of a more elaborate theory of decision making, called *Decision Theory* from probabilistic inferences. We examine Decision Theory in a little more detail in Section 5.

4.2 Multinomial Logistic Regression

We can easily extend the above discussion to find a set of probabilistic discriminants to distinguish between multiple classes. This is also often referred to as *Softmax Regression*. Suppose each object in the domain belongs to one of K classes $\{C_1, \ldots, C_K\}$. As with the binary

logistic regression, we are given n points and their labels $\mathcal{D}_n = \{(\boldsymbol{x}_1, c_1), \ldots, (\boldsymbol{x}_n, c_n)\}$ where $\boldsymbol{x}_1 \in \mathcal{R}^d, c_i \in \{1, \ldots, K\}$. We seek a set of linear models $\boldsymbol{w}_i^T.\boldsymbol{x}, 1 \leq i \leq K$ with parameters $\boldsymbol{w}_i$ along with a thresholding function $\theta_i(y)$ that returns the probability with which a given value $y = \boldsymbol{w}_i^T.\boldsymbol{x}$ will get the label i. The loss function would be the negative log-likelihood of the model returning the correct observed value c at any given point $\boldsymbol{x}$ after thresholding. We want to minimize the expected loss (maximize likelihood). The generalization of the Logistic function for multiple classes would be as follows:

$$p(C_i \mid \boldsymbol{x}) = \theta(\boldsymbol{w}_i^T.\boldsymbol{x}) = \frac{e^{\boldsymbol{w}_i^T.\boldsymbol{x}}}{\sum_{j=1}^{K} e^{\boldsymbol{w}_j^T.\boldsymbol{x}}} = \frac{e^{\boldsymbol{w}_i^T.\boldsymbol{x}}}{N_f}$$

where for convenience we have denoted the normalization factor $\sum_{j=1}^{K} e^{\boldsymbol{w}_j^T.\boldsymbol{x}}$ as N_f. Let the matrix $V^{(n\times K)}$ be such that $v_{ij} = \mathbb{I}(c_i = j)$ and let $\boldsymbol{W}$ represent the collection of parameter vectors $\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K\}$. The negative log-likelihood (empirical error) for the dataset $\mathcal{D}_n$ is then

$$\begin{aligned}
E_{\boldsymbol{W}}^n &= -\frac{1}{n} \ln \prod_{i=1}^{n} \prod_{j=1}^{K} \left(\theta(\boldsymbol{w}_j^T.\boldsymbol{x}_i)\right)^{v_{ij}} \\
&= -\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{K} v_{ij}.\ln\left(\theta(\boldsymbol{w}_j^T.\boldsymbol{x}_i)\right) \\
&= -\frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{K} v_{ij} \boldsymbol{w}_j^T.\boldsymbol{x}_i - \left(\sum_{j=1}^{K} v_{ij}\right) \ln\left(N_f\right) \right) \\
&= -\frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{K} v_{ij} \boldsymbol{w}_j^T.\boldsymbol{x}_i - \ln\left(N_f\right) \right)
\end{aligned}$$

Note that $\sum_{j=1}^{K} v_{ij} = 1$ — there is exactly one label given for each data point. This is also called the *Cross-Entropy* error function. As before we can now compute the gradient of $E_{\boldsymbol{W}}^n$.

$$\begin{aligned}
\nabla_{\boldsymbol{w}_k} E_{\boldsymbol{W}}^n = \frac{\partial E_{\boldsymbol{W}}^n}{\partial \boldsymbol{w}_k} &= -\frac{1}{n} \sum_{i=1}^{n} \left(v_{ik}\boldsymbol{x}_i - \frac{\frac{\partial N_f}{\partial \boldsymbol{w}_k}}{N_f} \right) = -\frac{1}{n} \sum_{i=1}^{n} \left(v_{ik}\boldsymbol{x}_i - \frac{e^{\boldsymbol{w}_k^T \boldsymbol{x}_i}}{N_f} \boldsymbol{x}_i \right) \\
&= \frac{1}{n} \sum_{i=1}^{n} \left(\theta\left(\boldsymbol{w}_k^T \boldsymbol{x}_i\right) - v_{ik} \right) \boldsymbol{x}_i
\end{aligned}$$

Note that this has a structure similar to that we obtained for the binary logistic regression case, where the gradient of the empirical error with respect to some $\boldsymbol{w}_k$ is of the form $\frac{1}{n}\sum_{i=1}^{n} \epsilon_{ki}\boldsymbol{x}_i$ for some error terms ϵ_{ki}. A compact way to represent the gradient $\nabla E_{\boldsymbol{W}}^n$ (with respect to all the parameter vectors $\boldsymbol{w}_j$) would be to use the notation $A \otimes B$ for the *Krönecker Product* of matrices $A^{(p\times q)}$ and $B^{(s\times t)}$, defined as the block matrix of order $(ps \times qt)$

$$A \otimes B \equiv \begin{bmatrix} a_{11}B & \ldots & a_{1q}B \\ \vdots & \ddots & \vdots \\ a_{p1}B & \ldots & a_{pq}B \end{bmatrix}$$

Suppose $\boldsymbol{\theta}_i$ and $\boldsymbol{v}_i$ are K-dimensional vectors for each $1 \leq i \leq n$, whose j^{th} components are $\theta_{ij} = \theta(\boldsymbol{w}_j^T.\boldsymbol{x}_i)$ and v_{ij} respectively for $1 \leq j \leq K$, then we can write the gradient as

$$\nabla E_{\boldsymbol{W}}^n = \frac{1}{n} \sum_{i=1}^{n} (\boldsymbol{\theta}_i - \boldsymbol{v}_i) \otimes \boldsymbol{x}_i$$

Note that the gradient is a (Kd)-dimensional vector. We can now also compute the Hessian for $E^n_{\boldsymbol{W}}$ as follows:

$$\nabla_{\boldsymbol{w}_l}\nabla_{\boldsymbol{w}_k}E^n_{\boldsymbol{W}} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\partial\left(\theta\left(\boldsymbol{w}_k^T\boldsymbol{x}_i\right)-v_{ik}\right)}{\partial\boldsymbol{w}_l}\boldsymbol{x}_i\right) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\partial\theta\left(\boldsymbol{w}_k^T\boldsymbol{x}_i\right)}{\partial\boldsymbol{w}_l}\boldsymbol{x}_i\right)$$

Exercise 6 *Show that the $\nabla_{\boldsymbol{w}_l}\nabla_{\boldsymbol{w}_k}E^n_{\boldsymbol{W}}$ part of the Hessian of $E^n_{\boldsymbol{W}}$ can be reduced to*

$$\nabla_{\boldsymbol{w}_l}\nabla_{\boldsymbol{w}_k}E^n_{\boldsymbol{W}} = \frac{1}{n}\sum_{i=1}^{n}\theta_{il}(\delta_{kl}-\theta_{ik})\boldsymbol{x}_i\boldsymbol{x}_i^T$$

where δ_{kl} is the Krönecker delta which is 1 iff $k=l$ and zero otherwise. Extend this to show that the Hessian $H(E^n_{\boldsymbol{W}}) \equiv \nabla^2 E^n_{\boldsymbol{W}}$ is equal to the block matrix of order $(K\times K)\otimes(d\times d)$ defined as

$$\nabla^2 E^n_{\boldsymbol{W}} = \frac{1}{n}\left[\ \sum_{i=1}^{n}\left(\boldsymbol{\theta}_i^T I^{(K\times K)} - \boldsymbol{\theta}_i\boldsymbol{\theta}_i^T\right)\otimes\left(\boldsymbol{x}_i\boldsymbol{x}_i^T\right)\ \right]$$

Finally show that just the way it is for the binary logistic regression case, the Hessian as defined above is also positive definite.

5 Decision Theory - Introduction

Decision Theory concerns itself with methodologies for taking decisions and/or actions from probabilistic inferences.

6 Fischer's Linear Discriminant

This, also known as *Fisher's Linear Discriminant Analysis (FLDA)*, is a linear classification algorithm that maximizes a certain measure of separation between the classes that it seeks to separate. The method, unlike the Perceptron algorithm works even for non-separable training sets. The idea is to find the one-dimensional projection of the training data that minimizes the 'intra-class' variance and maximizes the 'inter-class' variance. Let's first consider the two-class separation case. Given a direction $\boldsymbol{w}$ we can project all the data points on the hyperplane with $\boldsymbol{w}$ as the normal. We then look at the spread of the distances of the projections from the line defined by $\boldsymbol{w}$. We can now look for a direction $\boldsymbol{w}$ such that the projections of the points in one class are as far separated from those of the other class, as possible.

Suppose $\mathcal{D}_n = \{(\boldsymbol{x}_1, v_1),\ldots,(\boldsymbol{x}_n, v_n)\}$ is the training data where $\boldsymbol{x}_i \in \mathcal{R}^d$ are the attribute vectors and $v_i \in \pm 1$ are the corresponding labels. We are looking for a linear discriminant $\boldsymbol{w}$ such that $\text{sign}(\boldsymbol{w}^T.\boldsymbol{x})$ predicts the label for any data point $\boldsymbol{x} \in \mathcal{R}^d$ as accurately as possible. We again carry out an empirical risk minimization to arrive at the model $\boldsymbol{w}$. To define the loss function for the Fisher's discriminant, we introduce a few notions. Let $I_+ = \{i \mid v_i = +1\}$ and $I_- = \{i \mid v_i = -1\}$ denote the sets of points with labels $+1$ and -1 respectively. We define the following for a given model hypothesis $\boldsymbol{w}$.

The class mean for each class of points is defined as follows

$$\boldsymbol{\mu}_+ = \frac{1}{|I_+|}\sum_{i\in I_+}\boldsymbol{x}_i, \quad \boldsymbol{\mu}_- = \frac{1}{|I_-|}\sum_{i\in I_-}\boldsymbol{x}_i$$

The *inter-class variance* is then defined as

$$\sigma_\pm^2 = \left(\boldsymbol{w}^T.(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)\right)^2 = \boldsymbol{w}^T.(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^T\boldsymbol{w} = \boldsymbol{w}^T\Sigma_\mu\boldsymbol{w}$$

Note that this is the just the square of the orthogonal (to $\boldsymbol{w}$) distance between the projections of the two class mean points. Now we can define the variances within each class (*intra-class variance*) as

$$\sigma_+^2 = \sum_{i\in I_+}\left(\boldsymbol{w}^T.(\boldsymbol{x}_i - \boldsymbol{\mu}_+)\right)^2 = \boldsymbol{w}^T.\left(\sum_{i\in I_+}(\boldsymbol{x}_i - \boldsymbol{\mu}_+)(\boldsymbol{x}_i - \boldsymbol{\mu}_+)^T\right)\boldsymbol{w} = \boldsymbol{w}^T\Sigma_+\boldsymbol{w}$$

$$\sigma_-^2 = \sum_{i\in I_-}\left(\boldsymbol{w}^T.(\boldsymbol{x}_i - \boldsymbol{\mu}_-)\right)^2 = \boldsymbol{w}^T.\left(\sum_{i\in I_-}(\boldsymbol{x}_i - \boldsymbol{\mu}_-)(\boldsymbol{x}_i - \boldsymbol{\mu}_-)^T\right)\boldsymbol{w} = \boldsymbol{w}^T\Sigma_-\boldsymbol{w}$$

See Figure 5 for a pictorial illustration of the all these notions. The matrices $\Sigma_\mu, \Sigma_+, \Sigma_-$ are also referred to as Scatter Matrices in the literature. We are now ready the define the loss function for the Fisher's Linear Discriminant. The loss function is defined as the squared-ratio of the projected distance of the test point from its class mean to the projected distance between the two class means. The loss function and the empirical error and the output model $\boldsymbol{w}^*$ are defined formally as

$$\mathcal{L}((\boldsymbol{x}, v), \boldsymbol{w}^T.\boldsymbol{x}) = \frac{\left(\boldsymbol{w}^T.(\boldsymbol{x} - \boldsymbol{\mu})\right)^2}{\boldsymbol{w}^T\Sigma_\mu\boldsymbol{w}}$$

$$E_{\boldsymbol{w}}^n = \frac{\boldsymbol{w}^T(\Sigma_+ + \Sigma_-)\boldsymbol{w}}{\boldsymbol{w}^T\Sigma_\mu\boldsymbol{w}} = \frac{\boldsymbol{w}^T\Sigma_D\boldsymbol{w}}{\boldsymbol{w}^T\Sigma_\mu\boldsymbol{w}}$$

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} E_{\boldsymbol{w}}^n$$

where $\mu = \mu_+$ if $v = +1$ and $\mu = \mu_-$ otherwise. Using the necessary condition for $E_{\boldsymbol{w}}^n$ to be minimum we get

$$\frac{dE_{\boldsymbol{w}}^n}{d\boldsymbol{w}} = \frac{2\Sigma_D\boldsymbol{w}}{\boldsymbol{w}^T\Sigma_\mu\boldsymbol{w}} - \frac{2\Sigma_\mu\boldsymbol{w}\left(\boldsymbol{w}^T\Sigma_D\boldsymbol{w}\right)}{\left(\boldsymbol{w}^T\Sigma_\mu\boldsymbol{w}\right)^2} = 0$$

$$\boldsymbol{w} = \left(\frac{\boldsymbol{w}^T\Sigma_D\boldsymbol{w}}{\boldsymbol{w}^T\Sigma_\mu\boldsymbol{w}}\right)\Sigma_D^{-1}\Sigma_\mu\boldsymbol{w}$$

$$= \left(\frac{\boldsymbol{w}^T\Sigma_D\boldsymbol{w}}{\boldsymbol{w}^T\Sigma_\mu\boldsymbol{w}}\right)\left((\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^T\boldsymbol{w}\right)\Sigma_D^{-1}(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)$$

$$\boldsymbol{w}^* \propto \Sigma_D^{-1}(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-) \tag{3}$$

Note: We have used the fact that the expressions $\boldsymbol{w}^T\Sigma_D\boldsymbol{w}$, $\boldsymbol{w}^T\Sigma_\mu\boldsymbol{w}$ and $(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^T\boldsymbol{w}$ are all scalars. So the direction of $\boldsymbol{w}^*$ is determined only by $\Sigma_D^{-1}(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)$.

Note that if $\Sigma_D \propto I$ (as would happen if for instance each class was generated by a Gaussian with an identity covariance matrix) then the direction of the discriminant coincides with the vector difference between the two class means. This is clearly what we would expect.

Exercise 7 *Show that the covariance matrix of a spherically symmetric distribution is of the form αI for some scalar constant α.*

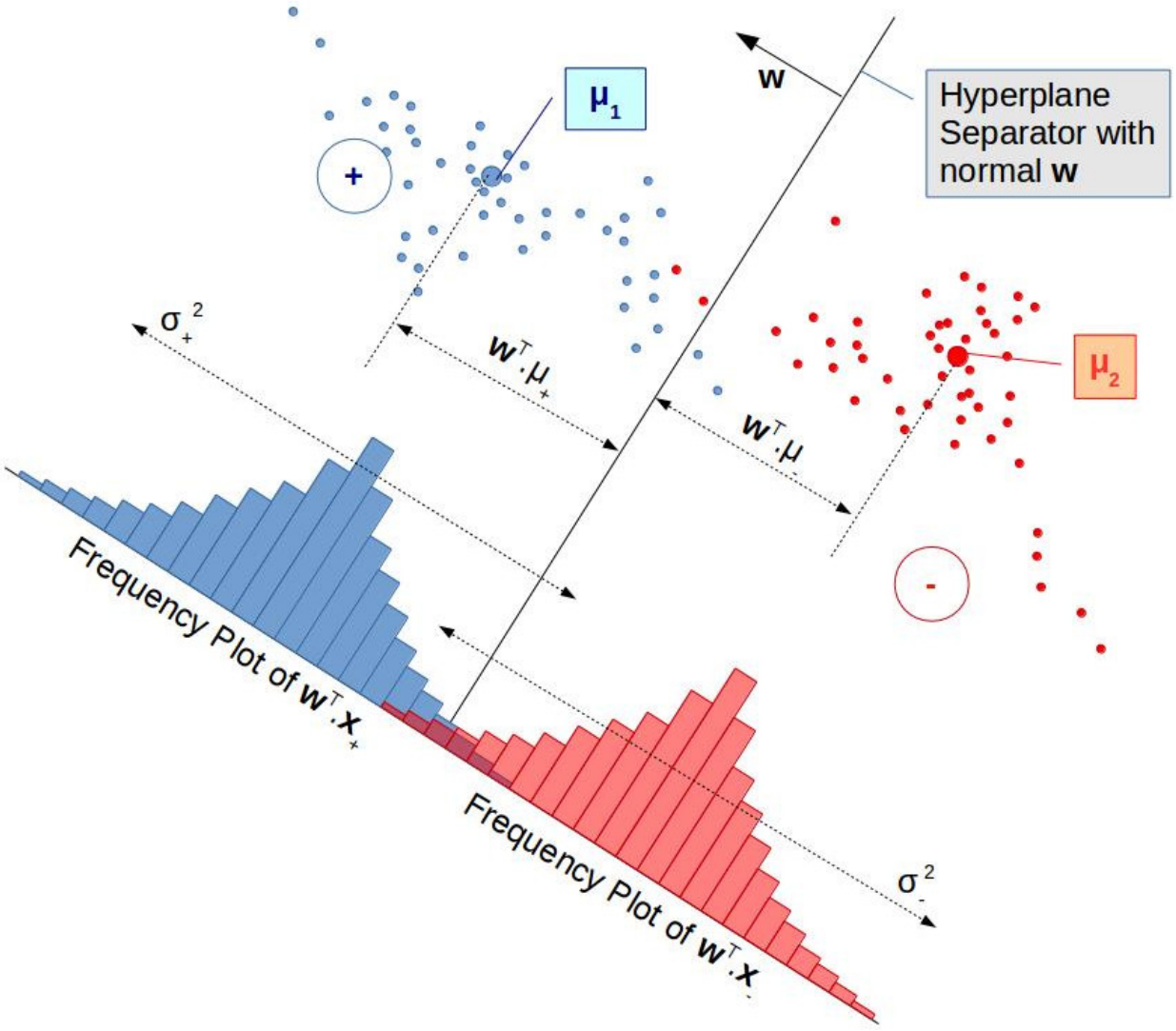

Figure 5: Fisher's Linear Discriminant

6.1 Decision Making from Fisher's Linear Discriminant

Given some training data, we can arrive at a linear discriminant $\boldsymbol{w}$ using Fisher's technique as described above. How does one use this to classify any new test point $\boldsymbol{x}$? One way is to first compute the projections $y_i = \boldsymbol{w}^T.\boldsymbol{x}_i$ of each of the training data points $\boldsymbol{x}_i, 1 \leq i \leq n$ with respect to $\boldsymbol{w}$. Consider the points labelled +1. We would expect the $y_i, i \in I_+$s to be approximately normally distributed around $\boldsymbol{w}^T.\mu_+$. This is because the projections $\boldsymbol{w}^T.\boldsymbol{x}_i$ are weighted sums of random variables (taking the components of the $\boldsymbol{x}_i$s as random variables with some arbitrary distribution). The central limit theorem assures us that the sum of a large number of random variables is approximately normal. Therefore given all $y_i, i \in I_+$ we can try fitting a Gaussian for these values. We resort to the Gaussian Density Estimation discussion in Section 4.5.1 of the Problem Formulation document. We use the formalism described there to estimate the mean and the variance of the Gaussian. In fact it turns out that the mean and variance are simply μ_+ and σ_+^2 that we defined earlier. Identical observations hold for the points with the indices in I_-

as well. Let the Gaussian densities for the two classes arrived at as described above be $p_+(y)$ and $p_-(y)$. Now given a test point $\boldsymbol{x}$ we classify it as follows. Compute the probabilites $p_+(\boldsymbol{w}^T.\boldsymbol{x})$ and $p_-(\boldsymbol{w}^T.\boldsymbol{x})$ — these are (posterior) probabilities that the test point $\boldsymbol{x}$ belongs classes + and − respectively. We can now use the Decision Theory as described in Section 5 to then take a decision on the class that we will assign the point $\boldsymbol{x}$ to.

6.2 Least Squares Interpretation of the Fisher's Discriminant

Fisher's Discriminant can in fact be seen as a special case of the Least Squares regression solution. Let $\mathcal{D}_n = \{(\boldsymbol{x}_1, v_1), \ldots, (\boldsymbol{x}_n, v_n)\}$ be the training data where $\boldsymbol{x}_i \in \mathcal{R}^d$ are the attribute vectors and $v_i \in \pm 1$ are the corresponding labels. As earlier let $I_+ = \{i \mid v_i = +1\}$ and $I_- = \{i \mid v_i = -1\}$ denote the sets of points with labels +1 and −1 respectively. We create a transformed dataset that is centered around $\boldsymbol{\mu}_- = \frac{1}{|I_-|}\sum_{i \in I_-} \boldsymbol{x}_i$ and with one value $y_+ \in \mathcal{R}$ for all the points in I_+ and another value $y_- \in \mathcal{R}$ for all the points in I_-. We will specify appropriate constants for y_+ and y_- later in this discussion. We now have a new transformed dataset $D'_n = \{((\boldsymbol{x}_1 - \boldsymbol{\mu}_-), y_1), \ldots, ((\boldsymbol{x}_n - \boldsymbol{\mu}_-), y_n)\}$ where $y_i \in \{y_+, y_-\}$. We now run a least squares regression on D'_n. Using the basic least squares solution of Equation 1 and denoting the vector $(y_1, \ldots, y_n)$ as $\boldsymbol{y}$ we get

$$\begin{aligned}\left(\sum_{i=1}^{n}(\boldsymbol{x}_i - \boldsymbol{\mu}_-)(\boldsymbol{x}_i - \boldsymbol{\mu}_-)^T\right)\boldsymbol{w} &= ((\boldsymbol{x}_1 - \boldsymbol{\mu}_-), \ldots, (\boldsymbol{x}_n - \boldsymbol{\mu}_-)).\boldsymbol{y} \\ &= y_+ \sum_{i \in I_+}(\boldsymbol{x}_i - \boldsymbol{\mu}_-) + y_- \sum_{i \in I_-}(\boldsymbol{x}_i - \boldsymbol{\mu}_-) \\ &= y_+|I_+|(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-) \end{aligned} \tag{4}$$

Note that $\sum_{i \in I_+}(\boldsymbol{x}_i - \boldsymbol{\mu}_+) = \sum_{i \in I_-}(\boldsymbol{x}_i - \boldsymbol{\mu}_-) = 0$. Similary

$$\begin{aligned}\left(\sum_{i=1}^{n}(\boldsymbol{x}_i - \boldsymbol{\mu}_-)(\boldsymbol{x}_i - \boldsymbol{\mu}_-)^T\right) &= \left(\Sigma_- + \sum_{i \in I_+}(\boldsymbol{x}_i - \boldsymbol{\mu}_-)(\boldsymbol{x}_i - \boldsymbol{\mu}_-)^T\right) \\ &= \left(\Sigma_- + \sum_{i \in I_+}(\boldsymbol{x}_i - \boldsymbol{\mu}_+ + \Delta\boldsymbol{\mu})(\boldsymbol{x}_i - \boldsymbol{\mu}_+ + \Delta\boldsymbol{\mu})^T\right) \\ &= \left(\Sigma_- + \Sigma_+ + |I_+|\Delta\boldsymbol{\mu}.\Delta\boldsymbol{\mu}^T\right) \\ &= (\Sigma_D + |I_+|\Sigma_\mu) \end{aligned} \tag{5}$$

where $\Delta\boldsymbol{\mu} = (\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)$. We have again used the fact that $\sum_{i \in I_+}(\boldsymbol{x}_i - \mu_+) = 0$. Now putting Equations 4 and 5 together we get

$$\begin{aligned}(\Sigma_D + |I_+|\Sigma_\mu)\boldsymbol{w} &= y_+|I_+|(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-) \\ \Sigma_D\boldsymbol{w} &= |I_+|\left(y_+ - (\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^T\boldsymbol{w}\right)(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-) \\ \boldsymbol{w} &= \alpha\Sigma_D^{-1}(\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)\end{aligned}$$

for some scalar $\alpha = |I_+|\left(y_+ - (\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^T\boldsymbol{w}\right)$. This is exactly what we obtained in Equation 3.

6.3 Multiclass-Multidimensional FLDA

We now examine the extension of LDA to multiple classes and also projection spaces that are multidimensional. Notice that the FLDA that we discussed above projects the data points onto a

line (the normal to the separator) and optimizes the between class variance to in-class variance ratio, along this projection line. We now explore the generalization of FLDA to multiple classes and projections onto a subspace of dimension $k > 1$. So instead of just a projection vector $\boldsymbol{w}$ we have a projection matrix $W^{(d\times k)}$ that maps a point $\boldsymbol{x} \in \mathcal{R}^d$ to $\mathcal{R}^k$, i.e., $W^T.\boldsymbol{x} = \boldsymbol{y} \in \mathcal{R}^k$. As before we are given a dataset $\mathcal{D}_n = \{(\boldsymbol{x}_1, c_1), \ldots, (\boldsymbol{x}_n, c_n)\}$ where $\boldsymbol{x}_i \in \mathcal{R}^d$ are the data attributes and $c_i \in \{1, \ldots, C\}$ are the class indices for any $1 \le i \le n$. We need to find the projection matrix W that maximizes the between-class variance to the in-class variance ratio. Let $I_c = \{i \mid c_i = c\}$ be the set of indices of points that belong to class c and let $n_c = |I_c|$ be the number of points belonging to class c. Note that $\sum_{c=1}^{C} n_c = n$. Also let $W^T\boldsymbol{x}_i = \boldsymbol{y}_i$. We try to find the W that will maximize the between-class variance to the in-class variance ratio after projection using W.

Again we define the mean of the points in each class and the total mean of the entire dataset

$$\boldsymbol{\mu}_c = \frac{1}{n_c}\sum_{i\in I_c}\boldsymbol{x}_i \quad 1 \le c \le C, \quad \text{and} \quad \boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i$$

It may also be useful to define the mean of the projections of each of the data points.

$$\boldsymbol{\mu}'_c = \frac{1}{n_c}\sum_{i\in I_c}\boldsymbol{y}_i = W^T\boldsymbol{\mu}_c \quad 1 \le c \le C, \quad \text{and} \quad \boldsymbol{\mu}' = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{y}_i = W^T\boldsymbol{\mu}$$

We can now construct the between-class and in-class covariance matrices of the projections

$$\Sigma'_c = \frac{1}{n_c}\sum_{i\in I_c}(\boldsymbol{y}_i - \boldsymbol{\mu}'_c)(\boldsymbol{y}_i - \boldsymbol{\mu}'_c)^T = W^T\Sigma_c W \quad 1 \le c \le C, \quad \text{and}$$
$$\Sigma'_D = \frac{1}{n}\sum_{i\in I_c} n_c\Sigma'_c = W^T\Sigma_D W$$

where the Σ_cs are within-class scatter matrices and Σ_D is the cumulative in-class scatter matrix as defined below.

$$\Sigma_c = \frac{1}{n_c}\sum_{i\in I_c}(\boldsymbol{x}_i - \boldsymbol{\mu}_c)(\boldsymbol{x}_i - \boldsymbol{\mu}_c)^T \quad 1 \le c \le C, \quad \text{and} \quad \Sigma_D = \frac{1}{n}\sum_{i\in I_c} n_c\Sigma_c$$

The inter-class (between-class) covariance can be defined with respect to the mean for the entire dataset.

$$\Sigma'_\mu = \frac{1}{n}\sum_{c=1}^{C} n_c(\boldsymbol{\mu}'_c - \boldsymbol{\mu}')(\boldsymbol{\mu}'_c - \boldsymbol{\mu}')^T = W^T\Sigma_\mu W$$

where Σ_μ is the between-class scatter matrix

$$\Sigma_\mu = \frac{1}{n}\sum_{c=1}^{C} n_c(\boldsymbol{\mu}_c - \boldsymbol{\mu})(\boldsymbol{\mu}_c - \boldsymbol{\mu})^T$$

We now need to define the Empirical Error E_W^n as a scalar measure that when optimized will 'maximize' the between-class covariance and 'minimize' the in-class covariance. In the multiclass-

multidimensional case there could be several choices for the scalar measure, some of which are:

$$E_W^n = \frac{|\Sigma'_D|}{|\Sigma'_\mu|} = \frac{|W^T\Sigma_D W|}{|W^T\Sigma_\mu W|} = \frac{|W\Sigma_D W^T|}{|W\Sigma_\mu W^T|} \tag{6}$$

$$E_W^n = \frac{Tr(\Sigma'_D)}{Tr(\Sigma'_\mu)} = \frac{Tr(W^T\Sigma_D W)}{Tr(W^T\Sigma_\mu W)} = \frac{Tr(W\Sigma_D W^T)}{Tr(W\Sigma_\mu W^T)} \tag{7}$$

$$\begin{aligned} E_W^n &= Tr\left(\left(\Sigma'_\mu\right)^{-1}\Sigma'_D\right) = Tr\left((W^T\Sigma_D W)^{-1}(W^T\Sigma_\mu W)\right) \\ &= Tr\left((W\Sigma_D W^T)^{-1}(W\Sigma_\mu W^T)\right) \end{aligned} \tag{8}$$

The discriminant matrix W^* is the one that minimizes the empirical error E_W^n.

$$W^* = \arg\min_W E_W^n$$

For empirical error as defined in 6 it can be shown that $W^* = \Sigma_D^{-\frac{1}{2}}U$ where U is the matrix that has the top k eigenvectors of $\Sigma_D^{-\frac{1}{2}}\Sigma_\mu\Sigma_D^{-\frac{1}{2}}$ as its columns. In other cases we would have to run a non-linear optimization algorithm to find W^* corresponding to a local minimum. Note that in these cases often E_W^n is not convex.

Note that we can also view FLDA as a dimensionality reduction technique — the method results in a projection matrix W^* which then can be used to map any given vector $\boldsymbol{x} \in \mathcal{R}^d$ to a lower dimensional vector $\boldsymbol{y} = W^T.\boldsymbol{x}$. The vector $\boldsymbol{y}$ is the 'feature vector' extracted from $\boldsymbol{x}$. However note that every Σ_c is of rank 1 — it is obtained from a single vector. Therefore Σ_D, Σ_μ can both be of rank at most $(C-1)$. Note that $\boldsymbol{\mu}$ is a linear combination of the class means $\boldsymbol{\mu}_c$. This constraint brings the upper bound on the rank of Σ_μ from C to $(C-1)$. The FLDA therefore is restricted to feature spaces of at most $(C-1)$ features for any dataset with C classes irrespective of how large the original dimensionality of the dataset is.

7 Generalized Linear Models

Setting is again that of a dataset $\mathcal{D}_n = \{(\boldsymbol{x}_1, v_1), \ldots, (\boldsymbol{x}_n, v_n)\}$ consisting of n independently sampled data points. Linear Regression that we have discussed in Section 3 rests on the assumption that the dataset that we are trying to 'explain' using the model vector $\boldsymbol{w}$ is essentially of the form

$$\boldsymbol{v} = X.\boldsymbol{w} + \boldsymbol{\epsilon}$$

where

- $\boldsymbol{v} = (v_1, \ldots, v_n)$ is the vector of 'outcome' values of points in the dataset,
- each row of the matrix X is a vector of explanatory variables $\boldsymbol{x}_i$, and
- the conditional-mean of the outcome v_i given the explanatory variables $\boldsymbol{x}_i$ subject to a random noise ϵ_i is $\boldsymbol{x}_i^T.\boldsymbol{w}$. Formally we assume that $E[v_i|\boldsymbol{x}_i] = \boldsymbol{w}^T.\boldsymbol{x}_i$.

The term $X\boldsymbol{w}$ in the above equation is often called the *systematic component* and the 'noise' term $\boldsymbol{\epsilon}$ is called the *stochastic component*. The term $\boldsymbol{\epsilon}$ represents the *residues* — the differences between the systematic model outcomes $X.\boldsymbol{w}$ and the actual observed outcomes $\boldsymbol{v}$. The least squares solution that we derived in Equation 2 assumes certain conditions under which the standard least squares solution of Equation is 2 is applicable. These conditions, listed below, are often termed the **Gauss-Markov Conditions**.

1. The relationship between each explanatory variable and the outcome variable is linear, subject to random variations (noise).
2. The error term has zero mean i.e., $E[\boldsymbol{\epsilon}|\boldsymbol{x}] = 0$.
3. The residuals (error terms) are independent with no correlation between observations. Formally
$$Corr[\epsilon] = E[\epsilon_i, \epsilon_j] = \mathbb{I}(i = j)$$
4. **Homoskedasticity**: The conditional variance of the error term (given the explanatory variable vector $\boldsymbol{x}$) is constant for all $\boldsymbol{x}$ and over time. The error variance is a measure of model uncertainty. Homoskedasticity implies that the model uncertainty is identical across observations. Formally:
$$Var[\epsilon_i|\boldsymbol{x}] = E[\epsilon_i^2|\boldsymbol{x}] = \sigma^2 \text{ (constant)}$$
5. The error terms are independent of the explanatory variables i.e., the error term is not influenced by the actual explanatory variable vector $\boldsymbol{x}$. Formally, the covariance between the explanatory variables and the error term is zero.
$$Cov[\boldsymbol{x}, \boldsymbol{\epsilon}] = 0$$

The neat closed form solution of Equation 2 is strictly applicable only under the Gauss-Markov conditions. However in practice that rarely happens. One would often need a broader framework that can accomodate data sets that do not necessarily satisfy all of the Gauss-Markov conditions and yet are amenable to regression with linear models.

Many of the models discussed above can be thought of as special cases of a generic family of linear predictors called the *Generalized Linear Models (GLM)*. The GLM was introduced by Nelder and Wedderburn in the 1970's as a generic framework for linear statistical models for data. Given their wide applicability it is important to get a broad understanding of the GLM and its implications to Machine Learning.

7.1 Structure of a GLM

The generalized linear models have a generic structure that makes their analysis easy. We seek a linear model to fit the data into. A GLM has three parts to it:

1. A **random component** that specifies the conditional distribution of the values $v_i \in \mathcal{R}$ (which in general could be a vector) given the attribute vectors $\boldsymbol{x}_i$ — usually specified as the probability density function $p(v_i|\boldsymbol{x}_i)$. The components of $\boldsymbol{x}_i$ are sometimes called the 'explanatory variables' and v_i the 'response' variable. The most common (and those in the original formulation of GLM) choices for the random component are those from the exponential family of distributions. Examples are Gaussian (normal), binomial, Poisson, gamma, or inverse-Gaussian families of distributions. Exponential family of distributions have many other advantages. For instance these are the only family (under certain natural 'regularity' conditions) of distributions that have a finite sized sufficient statistics — the distribution can be completely specified by a just a finite set of numbers (for instance the mean and the variance completely specifies a Gaussian). Exponential family of distributions are also the only ones that have conjugate priors, a notion that comes in extremely handy in the treatment of Bayesian techniques.

2. A **linear predictor** that combines the explanatory variables $\boldsymbol{x}$ in a linear manner as below

$$\eta_i = \boldsymbol{w}^T.\boldsymbol{x}_i$$

   Note that as we mentioned earlier in this document, the explanatory variables are not necessarily the 'raw' attributes. These could be features that we obtain after applying a set of (potentially non-linear) basis functions.

3. A smooth and invertible linearizing **link function** $g(.)$, which transforms the expectation of the response variable, $\mu_i \equiv E[v_i|\boldsymbol{x}_i]$, to the linear predictor:

$$g(\eta_i) = \mu_i, \quad g^{-1}(\mu_i) = \boldsymbol{w}^T.\boldsymbol{x}_i$$

   The inverse of the link function g^{-1} is also sometimes referred to as the *mean function*.

7.2 Exponential Family of Distributions

An exponential family of distributions with *canonical parameter vector* $\boldsymbol{\theta}$, *mean parameters* $\boldsymbol{\eta}(\boldsymbol{\theta})$, vector of *sufficient statistics* $\boldsymbol{\phi}(\boldsymbol{v})$, *dispersion parameter* $\tau > 0$, *dispersion function* $\psi(\tau)$ and *scaling function* $h(\boldsymbol{v})$ is defined as one that has following generic form:

$$\begin{aligned} p(\boldsymbol{v}|\boldsymbol{\theta},\tau) &= \frac{1}{N(\boldsymbol{\theta},\tau)} h(\boldsymbol{v},\tau) \exp\left(\frac{\boldsymbol{\eta}(\boldsymbol{\theta})^T.\boldsymbol{\phi}(\boldsymbol{v})}{\psi(\tau)}\right) \\ &= h(\boldsymbol{v},\tau) \exp\left(\frac{\boldsymbol{\eta}(\boldsymbol{\theta})^T.\boldsymbol{\phi}(\boldsymbol{v}) - L(\boldsymbol{\theta},\tau)}{\psi(\tau)}\right), \quad \text{or} \qquad (9) \\ &= \exp\left(\frac{\boldsymbol{\eta}(\boldsymbol{\theta})^T.\boldsymbol{\phi}(\boldsymbol{v}) - L(\boldsymbol{\theta},\tau)}{\psi(\tau)} + c(\boldsymbol{v},\tau)\right) \qquad (10) \end{aligned}$$

where $N(\boldsymbol{\theta},\tau)$, called the *Partition Function* marginalizes the distribution with

$$\begin{aligned} N(\boldsymbol{\theta},\tau) &= \int_{\boldsymbol{v}} h(\boldsymbol{v},\tau) \exp\left(\frac{\boldsymbol{\eta}(\boldsymbol{\theta})^T.\boldsymbol{\phi}(\boldsymbol{v})}{\psi(\tau)}\right) d\boldsymbol{v} \\ h(\boldsymbol{v},\tau) &= e^{c(\boldsymbol{v},\tau)} \\ L(\boldsymbol{\theta},\tau) &= \psi(\tau) \ln(N(\boldsymbol{\theta},\tau)) \end{aligned}$$

The distribution is said to be in the *canonical form* if $\boldsymbol{\eta}(\boldsymbol{\theta}) = \boldsymbol{\theta}$. We actually do not lose any generality by assuming the canonical form — to convert an exponential distribution to its canonical form we only need to substibute another parameter vector $\boldsymbol{\beta} = \boldsymbol{\eta}(\boldsymbol{\theta})$ and treat the $\boldsymbol{\beta}$ as the vector of the model parameters. We assume henceforth that the distributions we deal with are in the canonical form unless stated otherwise.

7.2.1 Examples of Exponential Family Distributions

- **Bernoulli Distribution**: A distribution over $\{0,1\}$ where $p(1|\mu) = \mu$ and $p(0|\mu) = (1-\mu)$ for some $0 \leq \mu \leq 1$. We show that this belongs to the exponential distribution family as defined by 9. We rewrite the distribution $p(v|\mu)$ alternatively as (note that $v \in \{0,1\}$):

$$\begin{aligned} B(v|\mu) &= \mu^v (1-\mu)^{(1-v)} = e^{(v\ln\mu + (1-v)\ln(1-\mu))} \\ &= \exp\left(v\ln\left(\frac{\mu}{1-\mu}\right) + \ln(1-\mu)\right) \\ &= (1-\mu)\exp\left(v\ln\left(\frac{\mu}{1-\mu}\right)\right) \end{aligned}$$

  This indeed fits into the pattern of 9 with $\theta = \ln\left(\frac{\mu}{1-\mu}\right)$, $\psi(\tau) = \tau = 1$, $N(\theta,\tau) = (1+e^{\theta})$, $c(\boldsymbol{v},\tau) = 0$, $h(\boldsymbol{v},\tau) = 1$ and $\boldsymbol{\phi}(\boldsymbol{v}) = v$. Interestingly we can recover the mean parameter μ from the canonical paramter θ as follows

$$\theta = \ln\left(\frac{\mu}{1-\mu}\right) \Rightarrow \mu = \frac{1}{1+e^{-\theta}}$$

  the last one being exactly the logistic function that we have explored earlier.

- **Multinomial Distribution**: A distribution over a finite set of choices $\{1,\ldots,K\}$ where $\mu_i \geq 0$ for $1 \leq k \leq K$ is the probability of observing choice k. The mean parameter vector is $\boldsymbol{\mu} = (\mu_1,\ldots,\mu_K)$ with $\sum_{k=1}^{K}\mu_k = 1$ and the response variable vector is the vector of indicator variables $\boldsymbol{v} = (v_1,\ldots,v_K)$ where $v_k \in \{0,1\}$ with $\sum_{k=1}^{K} v_k = 1$. The multinomial distribution can now be expressed as an exponential distribution as shown below.

$$\begin{aligned} M(\boldsymbol{v}|\boldsymbol{\mu}) &= \prod_{k=1}^{K}\mu_k^{v_k} = \exp\left(\sum_{k=1}^{K-1} v_k\ln\mu_k + v_K\ln\mu_K\right) \\ &= \exp\left(\sum_{k=1}^{K-1} v_k\ln\mu_k + \left(1-\sum_{k=1}^{K-1}v_k\right)\ln\mu_K\right) \\ &= \exp\left(\sum_{k=1}^{K-1} v_k\ln\left(\frac{\mu_k}{\mu_K}\right) + \ln\mu_K\right) \end{aligned}$$

  This is exponential with $\psi(\tau) = \tau = 1$, $c(\boldsymbol{v},\tau) = 0$, $h(\boldsymbol{v},\tau) = 1$, $\boldsymbol{\phi}(\boldsymbol{v}) = \boldsymbol{v}$ and

$$\boldsymbol{\theta} = \begin{pmatrix} \ln\left(\frac{\mu_1}{\mu_K}\right) \\ \vdots \\ \ln\left(\frac{\mu_{K-1}}{\mu_K}\right) \end{pmatrix}, \quad N(\boldsymbol{\theta},\tau) = \frac{1}{\mu_K}$$

Trying to recover the mean parameters $\boldsymbol{\mu}$ from the canonical parameters $\boldsymbol{\theta}$ we get

$$\theta_k = \ln\left(\frac{\mu_k}{\mu_K}\right) \Rightarrow \mu_K e^{\theta_k} = \mu_k \Rightarrow \mu_K \sum_{k=1}^{K} e^{\theta_k} = \sum_{k=1}^{K} \mu_k = 1$$

Therefore

$$\mu_k = \frac{e^{\theta_k}}{\sum_{j=1}^{K} e^{\theta_j}}$$

which is exactly the class conditional distribution we had used for the multiclass logistic regression (softmax regression) formulation we had discussed earlier.

- **Univariate Gaussian**: The standard Gaussian is also exponential as one would expect. This is how it fits into the structure of an exponential distribution.

$$\begin{aligned}\mathcal{N}(v|\mu,\sigma^2) &= \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(v-\mu)^2}{2\sigma^2}\right)\\ &= \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(v\frac{\mu}{\sigma^2} - v^2\frac{1}{2\sigma^2} - \frac{\mu^2}{2\sigma^2}\right)\end{aligned}$$

is an exponential distribution with $h(\boldsymbol{v}) = 1$ and

$$\begin{aligned}\boldsymbol{\theta} &= \begin{pmatrix} \mu/\sigma^2 \\ -1/2\sigma^2 \end{pmatrix}\\ \boldsymbol{\phi}(v) &= \begin{pmatrix} v \\ v^2 \end{pmatrix}\\ N(\boldsymbol{\theta}) &= \sqrt{2\pi\sigma^2}\exp\left(\frac{\mu^2}{2\sigma^2}\right) = \sqrt{2\pi}(-2\theta_2)^{-1/2}\exp\left(\frac{-\theta_1^2}{4\theta_2}\right)\end{aligned}$$

7.3 Properties of the Log-Partition Function

The Log-Partition function $L(\boldsymbol{\theta})$ that we used in the definition of an exponential family of distributions has several interesting properties that may of independent interest. We show in particular that the *cumulants* (such as expectation and variance with respect to the canonical parameters $\boldsymbol{\theta}$) of the sufficient statistics of the distribution, given by $\boldsymbol{\phi}(\boldsymbol{v})$, can actually be represented as derivatives of the log-partition function.

$$\begin{aligned}\frac{dL(\boldsymbol{\theta})}{d\boldsymbol{\theta}} &= \frac{d}{d\boldsymbol{\theta}}\ln\int_{\boldsymbol{v}} h(\boldsymbol{v})\exp\left(\boldsymbol{\theta}^T.\boldsymbol{\phi}(\boldsymbol{v})\right)d\boldsymbol{v}\\ &= \frac{\frac{d}{d\boldsymbol{\theta}}\int_{\boldsymbol{v}} h(\boldsymbol{v})\exp\left(\boldsymbol{\theta}^T.\boldsymbol{\phi}(\boldsymbol{v})\right)d\boldsymbol{v}}{\int_{\boldsymbol{v}} h(\boldsymbol{v})\exp\left(\boldsymbol{\theta}^T.\boldsymbol{\phi}(\boldsymbol{v})\right)d\boldsymbol{v}}\\ &= \frac{\int_{\boldsymbol{v}} \boldsymbol{\phi}(\boldsymbol{v})h(\boldsymbol{v})\exp\left(\boldsymbol{\theta}^T.\boldsymbol{\phi}(\boldsymbol{v})\right)d\boldsymbol{v}}{N(\boldsymbol{\theta})}\\ &= \int_{\boldsymbol{v}} \boldsymbol{\phi}(\boldsymbol{v})p(\boldsymbol{v}|\boldsymbol{\theta})d\boldsymbol{v}\\ &= E[\boldsymbol{\phi}(\boldsymbol{v})]\end{aligned}$$

Differentiating this again we get

$$\begin{aligned}\frac{d^2L(\boldsymbol{\theta})}{d\boldsymbol{\theta}^2} &= \frac{d}{d\boldsymbol{\theta}}\int_{\boldsymbol{v}}\boldsymbol{\phi}(\boldsymbol{v})h(\boldsymbol{v})\exp\left(\boldsymbol{\theta}^T.\boldsymbol{\phi}(\boldsymbol{v})-L(\boldsymbol{\theta})\right)d\boldsymbol{v} \quad \text{(using 9)}\\ &= \int_{\boldsymbol{v}}\boldsymbol{\phi}(\boldsymbol{v})h(\boldsymbol{v})\exp\left(\boldsymbol{\theta}^T.\boldsymbol{\phi}(\boldsymbol{v})-L(\boldsymbol{\theta})\right)(\boldsymbol{\phi}(\boldsymbol{v})-L'(\boldsymbol{\theta}))d\boldsymbol{v}\\ &= \int_{\boldsymbol{v}}\boldsymbol{\phi}(\boldsymbol{v})^2 p(\boldsymbol{v}|\boldsymbol{\theta})d\boldsymbol{v}-L'(\boldsymbol{\theta})\int_{\boldsymbol{v}}\boldsymbol{\phi}(\boldsymbol{v})p(\boldsymbol{v}|\boldsymbol{\theta})d\boldsymbol{v}\\ &= E[\boldsymbol{\phi}^2(\boldsymbol{v})]-(E[\boldsymbol{\phi}(\boldsymbol{v})])^2\\ &= Var[\boldsymbol{\phi}(\boldsymbol{v})]\end{aligned}$$

Notice that for the binomial distribution $\boldsymbol{\phi}(\boldsymbol{v}) = v$. Therefore

$$\frac{dL(\boldsymbol{\theta})}{d\boldsymbol{\theta}} = \frac{d\ln(1+e^{\theta})}{d\theta} = \frac{e^{\theta}}{1+e^{\theta}} = \mu = E[v] = E[\boldsymbol{\phi}(\boldsymbol{v})]$$

which is the mean and

$$\frac{d^2L(\boldsymbol{\theta})}{d\boldsymbol{\theta}^2} = \frac{d}{d\theta}\left(\frac{e^{\theta}}{1+e^{\theta}}\right) = \frac{e^{\theta}}{1+e^{\theta}}\frac{1}{1+e^{\theta}} = \mu(1-\mu)$$

which is just the variance of a binomial distribution.

Similarly for a Gaussian distribution, we have

$$\frac{\partial L(\boldsymbol{\theta})}{d\theta_1} = \frac{\partial}{\partial\theta_1}\ln\left(\sqrt{2\pi}(-2\theta_2)^{-1/2}\exp\left(\frac{-\theta_1^2}{4\theta_2}\right)\right) = \frac{-\theta_1}{2\theta_2} = \mu = E[v]$$

and

$$\frac{\partial L(\boldsymbol{\theta})}{d\theta_2} = \frac{\partial}{\partial\theta_2}\ln\left(\sqrt{2\pi}(-2\theta_2)^{-1/2}\exp\left(\frac{-\theta_1^2}{4\theta_2}\right)\right) = \frac{-1}{2\theta_2}+\frac{\theta_1^2}{4\theta_2^2} = \sigma^2+\mu^2 = E[v^2]$$

Therefore

$$\frac{dL(\boldsymbol{\theta})}{d\boldsymbol{\theta}} = E\left[\begin{pmatrix} v \\ v^2 \end{pmatrix}\right] = E[\boldsymbol{\phi}(v)]$$

7.4 GLM Formulation

In this section we use the exponential distributions derived above construct generalized linear models out of Binomials, Multinomials and Gaussians. We also use the GLM as a framework to explore other linear models such as Poisson Regression. Letting $\boldsymbol{w}$ represent the weight vector for the linear predictor, we only need to specify the link function apart from the exponential family distribution. We complete the formulation below for the classes of linear models we have already looked at.

1. **Logistic Regression**: Logistic Regression fits into the GLM structure with the following:
   - *Random Component*: Bernoulli Distribution
   - *Link Function*:

Link	$\eta_i = g(\mu_i)$	$\mu_i = g^{-1}(\eta_i)$
Identity	μ_i	η_i
Log	$\ln \mu_i$	e^{η_i}
Inverse	μ^{-1}	η^{-1}
Inverse-Square	μ^{-2}	$\eta^{-1/2}$
Square-Root	$\sqrt{\mu_i}$	η_i^2
Logit	$\ln\left(\frac{\mu_i}{1-\mu_i}\right)$	$\frac{1}{1+e^{-\eta_i}}$
Probit		
Log-Log	$-\ln(-\ln \mu_i)$	$e^{-e^{-\eta_i}}$
Complimentary Log-Log	$\ln(-\ln(1-\mu_i))$	$1-e^{-e^{\eta_i}}$

Table 2: Some Common Link Functions and Their Inverses

7.5 Maximum Likelihood with GLM

Much of the what we did earlier with Logistic Regression, Least Squares fit etc. can indeed be carried over to any GLM. We