Matlab for Finance Course: Session 4

Dr. Peter A. Bebbington

Brainpool AI

peter@brainpool.ai

bebbington @brainpoolai @brainpoolai

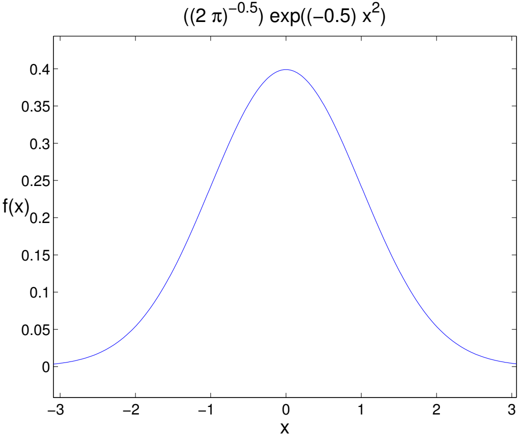
March 12, 2023

OBJECTIVE

- Functional Handles
- Portfolio Optimisation
- Coefficient of Determination
- Sharpe Ratio
- Supervised Learning
- Logistic Regression

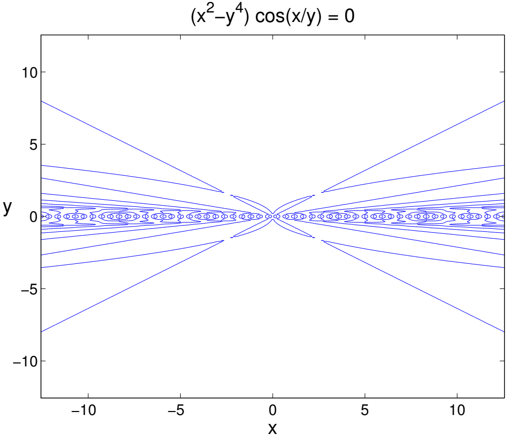
FUNCTION HANDLE

- Given $\mathcal{N}(x|0,1) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$
- `fun = @(x)((2.*pi).^(-0.5)).*exp((-0.5).*x.^2);`
- `ezplot(fun);`



FUNCTION STRING

- Given $f(x,y) = (x^2 - y^4)\cos(x/y)$
- Plot the implicit function $(x^2 - y^4)\cos(x/y) = 0$ by `ezplot('(x^2-y^4).*cos(x./y)',[-4*pi,4*pi])`



PORTFOLIO OPTIMISATION

- Given a portfolio which is defined as follows:

$$\pi_i \begin{cases} > 0, & \text{long position (buying assets)} \\ = 0, & \text{no position} \\ < 0, & \text{short position (selling asset)} \end{cases}$$

- We can define the portfolios variance as

$$\sigma_p^2 = \boldsymbol{\pi}'\boldsymbol{\Sigma}\boldsymbol{\pi}$$

- Our goal in portfolio optimisation is to

$$\boldsymbol{\pi}^* = \arg\min_{\boldsymbol{\pi}}\{\sigma_p^2\}$$

with the following constraints

$$\boldsymbol{\pi}'\mathbf{1} = 1$$
$$\boldsymbol{\pi}'\boldsymbol{\mu} = \mu_p.$$

OPTIMAL SOLUTION

- The solution to $\boldsymbol{\pi}^*$ is found by FOC giving

$$\pi_i^* = \frac{\lambda_1}{2}\sum_{j=1}^{N}\Sigma_{ij}^{-1} + \frac{\lambda_2}{2}\sum_{j=1}^{N}\Sigma_{ij}^{-1}\mu_j$$

- This problem is written linear as

$$\underbrace{\begin{pmatrix} \tilde{\Sigma}_{11} & \tilde{\Sigma}_{12} & \cdots & \tilde{\Sigma}_{1N} & 1 & \tilde{\mu}_1 \\ \tilde{\Sigma}_{21} & \tilde{\Sigma}_{22} & & \vdots & 1 & \tilde{\mu}_2 \\ \vdots & & \ddots & & \vdots & \vdots \\ \tilde{\Sigma}_{N1} & \cdots & & \tilde{\Sigma}_{NN} & 1 & \tilde{\mu}_N \\ 1 & 1 & \dots & 1 & 0 & 0 \\ \tilde{\mu}_1 & \tilde{\mu}_2 & \dots & \tilde{\mu}_N & 0 & 0 \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} \pi_1^* \\ \pi_2^* \\ \vdots \\ \pi_N^* \\ -\lambda_1/2 \\ -\lambda_2/2 \end{pmatrix}}_{\mathbf{a}} = \underbrace{\begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ \mu_p \end{pmatrix}}_{\mathbf{b}}$$

where the the mutual fund strategy is found by

$$\mathbf{a} = \mathbf{A}^{-1}\mathbf{b}.$$

COEFFICIENT OF DETERMINATION

- In the session script we see that solution to a is found `a=A\b` (equivalent to a linear regression) which quicker and more accurate than `inv(A)` or $\mathtt{A}^{-1}$.
- Assuming linear correlations $X_j = a + bX_i$ we can see how well model fits with

$$\varepsilon_{i,j} = X_j - (a + bX_i)$$

which is known as a residual. We can define the "Coefficient of Determination" as the square of the elements of the correlation matrix

$$\rho_{i,j}^2 = 1 - \frac{\mathbb{V}[\varepsilon_{i,j}]}{\mathbb{V}[X_j]}$$

where $\rho_{i,j}^2 = 1 \ \forall i, j$ means the linear model fits perfectly and large $\mathbb{V}[\varepsilon_{i,j}]$ means a poor linear fit.

SUPERVISED LEARNING METHOD

- One Can define a training set as

$$\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$$

- The goal is to infer a function

$$\mathcal{D} \mapsto f_{\mathcal{D}}(\mathbf{x}_i) \approx y_i$$

  then apply $f_{\mathcal{D}}$ help predict future data set

$$\mathcal{D}' = \{(\mathbf{x}_{m+1}, y_{m+1}), (\mathbf{x}_{m+2}, y_{m+2}), \dots\}$$

- examples; Classification $y \in \{-1, +1\}$, Regression : $y \in \mathbb{R}$

LINEAR REGRESSION

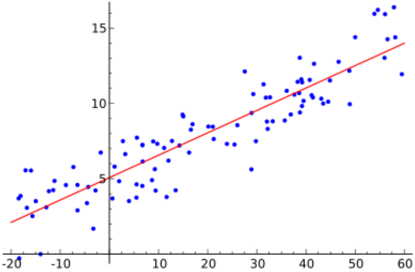
- This approach tries to fit the linear line

$$y_i \approx w_1 x_{i1} + w_2 x_{i2} + \ldots + w_m x_{im} + b_i$$

where $i = 1, 2, \ldots, n$ which can be written in matrix notation

$$y_i \approx \mathbf{x}_i' \mathbf{w} + b_i = \mathbf{X}' \mathbf{w} + \bar{b}$$

where $\bar{b} \in \mathbb{R}^n$ is a constant and represents the error/residuals, $\mathbf{w} \in \mathbb{R}^m$, $\mathbf{x}_i \in \mathbb{R}^n$ are column vectors and $\mathbf{X} \in \mathbb{R}^{n \times m}$ is a rectangular matrix.



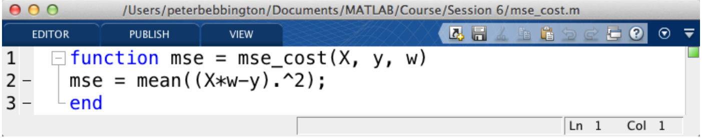
MEAN SQUARE ERROR (MSE)

- MSE is defined as

$$\mathrm{MSE}(\mathcal{D}, \mathbf{w}) = \frac{1}{m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2$$

  where $\hat{y}_i$ is our linear predictor $\hat{y}_i = \mathbf{w}'\mathbf{x} = \sum_{j=1}^{n} \mathrm{w}_j x_{ij}$

- We can write this in Matlab as the function `mse_cost.m`



LEAST SQUARE REGRESSION (LSR)

- This model looks for the weight vector that minimises the mean square errors on all **training samples** and is defined as

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \frac{1}{m} \sum_{i=1}^{m} (\mathbf{x}_i' \mathbf{w} - y_i)^2$$

- To solve the LSR equation we use matrix notation

$$\mathcal{L} = \frac{1}{m} \sum_{i=1}^{m} (\mathbf{x}_i' \mathbf{w} - y_i)^2 = \frac{1}{m} (\mathbf{X}' \mathbf{w} - \mathbf{y})' (\mathbf{X}' \mathbf{w} - \mathbf{y})$$

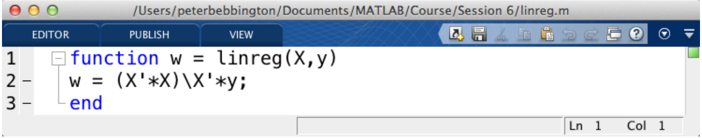
- Then apply FOC ($\frac{\partial \mathcal{L}}{\partial \mathbf{W}} = 0$), we find

$$\mathbf{w}^* = (\mathbf{X}' \mathbf{X})^{-1} \mathbf{X}' \mathbf{y}$$

- Solution exists if $\mathbf{X}$ is non-singular.

LSR FUNCTION

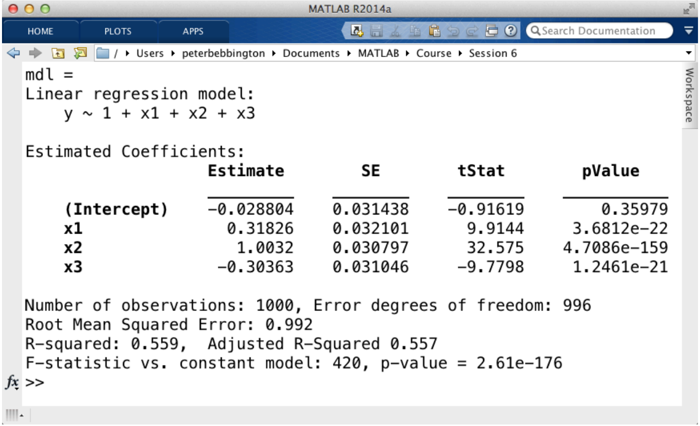
- One can write the function `linreg.m` to perform LSR



- As an example create a 3×3 matrix $\mathbf{X}$ and an output vector $\mathbf{y}$ in Matlab command line.
- Then `w = linreg(X,y)` and `mse = mse_cost(X,y,w)`

fitlm

- Alternative, one can use the function `fitlm()` which has the benefit of calculating various statistics



BASIS FUNCTIONS

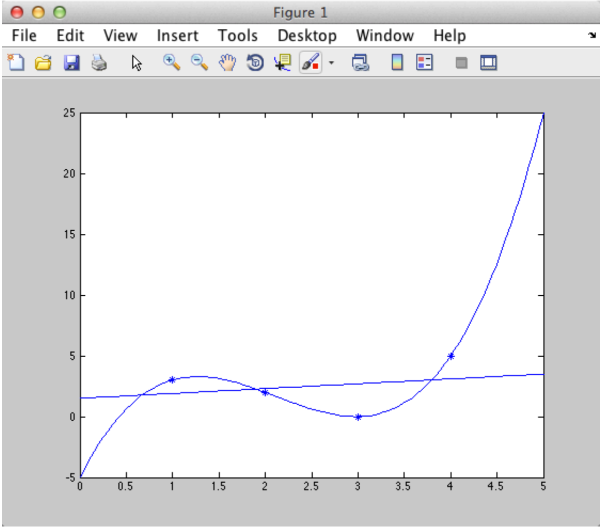
- If the function $f_{\mathcal{D}}$ is non-linear try introducing a polynomial vector as your basis function $\boldsymbol{\phi}(\mathbf{x}_i) = \phi_j(\mathbf{x}_i) = (1, x_i, x_i^2, \ldots, x_i^k)$ where $i = 1, 2, ..., m$ and $j = 1, 2, ..., k$. We now make the following change of basis $\mathbf{x}_i'\mathbf{w} \rightarrow \boldsymbol{\phi}(\mathbf{x}_i)'\mathbf{w}$.
- The LSR problem know becomes

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \frac{1}{m} \sum_{i=1}^{m} (y_i - \sum_{j=1}^{k} \phi_j(\mathbf{x}_i) w_j)^2 = (\boldsymbol{\Phi}'\boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}'\mathbf{y}$$

where the matrix

$$\boldsymbol{\Phi} = \begin{pmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^k \\ 1 & x_2 & x_2^2 & \cdots & x_2^k \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_m & x_m^2 & \cdots & x_m^k \end{pmatrix}$$

POLYFIT EXAMPLE



Ridge Regression (RR)

- A rule of thumb if $m << n$ LSR may find a function $f_{\mathcal{D}}$ that will overfit your data, that is you are just fitting noise.
- A way avoid this is RR defined as

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \left\{ \lambda \mathbf{w}'\mathbf{w} + \frac{1}{m} \sum_{i=1}^{m} (\mathbf{x}_i'\mathbf{w} - y_i)^2 \right\}$$

solve as before impose the FOC and we find

$$\begin{aligned} \mathbf{X}'\mathbf{X}\mathbf{w}^* + \lambda m \mathbf{w}^* &= \mathbf{X}'\mathbf{y} \\ \Rightarrow \mathbf{w}^* &= (\mathbf{X}'\mathbf{X} + \lambda m \mathbb{I}_n)^{-1} \mathbf{X}'\mathbf{y} \end{aligned}$$

where $\mathbb{I}_n$ is the $n \times n$ identity matrix.

RR & POLYNOMIAL BASIS

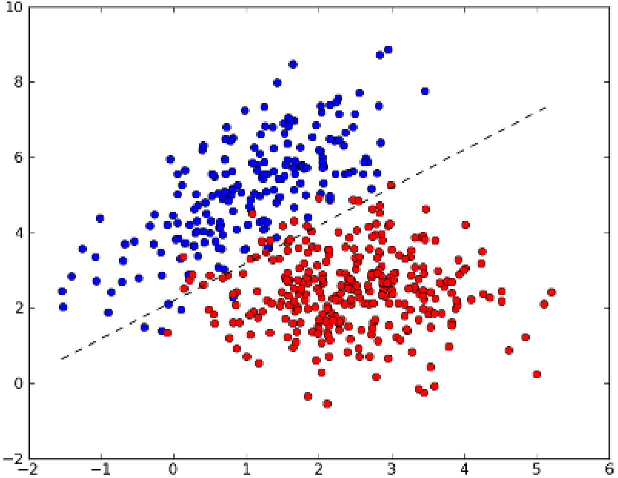
- Same Logic as before, replace $\mathbf{x}_i'\mathbf{w} \to \boldsymbol{\phi}(\mathbf{x}_i)'\mathbf{w}$ picking a polynomial basis of the form $\boldsymbol{\phi}(\mathbf{x}_i) = \phi_j(\mathbf{x}_i) = (1, x_i, x_i^2, \ldots, x_i^k)$ where $i = 1, 2, ..., m$ and $j = 1, 2, ..., k$.
- The RR problem is now defined as

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \left\{ \lambda \mathbf{w}'\mathbf{w} + \frac{1}{m} \sum_{i=1}^{m} (y_i - \sum_{j=1}^{k} \phi_j(\mathbf{x}_i) w_j)^2 \right\}$$

solve as before impose the FOC and we find

$$\begin{aligned} \boldsymbol{\Phi}'\boldsymbol{\Phi}\mathbf{w}^* + \lambda k \mathbf{w}^* &= \boldsymbol{\Phi}'\mathbf{y} \\ \Rightarrow \mathbf{w}^* &= (\boldsymbol{\Phi}'\boldsymbol{\Phi} + \lambda k \mathbb{I}_k)^{-1} \boldsymbol{\Phi}'\mathbf{y} \end{aligned}$$

where $\mathbb{I}_k$ is the $k \times k$ identity matrix.


10
8
6
4
2
0
−2
−2
−1
0
1
2
3
4
5
6

BINARY CLASSIFICATION

- Used Everywhere!
- A few projects last year used this (Credit Risk).
- Not a Regression!
- Its Actual Classification Method.
- Think back if we have a training set then

$$\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$$

- Classification $\Rightarrow y_i \in \{0, 1\}$, $y_i \in \{-1, +1\}$ or $y \in \{C_1, C_2\}$
  $\rightarrow$ known as binary classification

LOGISTIC REGRESSION

- Our objective is to find some linear relationship (a hyperplane) in our new basis function space that divides the two classes $\{C_1, C_2\}$. The hyperplane is defined as before $\mathbf{w} \in \mathbb{R}^n$ such that

$$p(C_1|\mathbf{w}, \mathbf{x}) = \sigma(\mathbf{w}'\phi(\mathbf{x}))$$

  where is the $\sigma(.)$ is a sigmoid function defined as

$$\sigma(u) = \frac{1}{1 + e^{-u}}$$

- You can see what the function looks by the command `ezplot('1/(1+exp(-u))')`.
- **Please note that this is not a sigma-algebra**

MLE SOLUTION

- For the likelihood of observing outputs $\mathbf{y} \in \{C_1, C_2\}_{i=1}^m$ given inputs $\mathbf{X}$ and a hyperplane parametrised by $\mathbf{w}$ will be given by

$$p(\mathbf{y}|\mathbf{w}, \mathbf{X}) = \prod_{i=1}^{m} [\sigma(\mathbf{w}'\phi(\mathbf{x}_i))]^{y_i} [1 - \sigma(\mathbf{w}'\phi(\mathbf{x}_i))]^{1-y_i}$$

- Objective $\arg\max_{\mathbf{w}} \{\log(p(\mathbf{t}|\mathbf{w}, \mathbf{X}))\}$ a useful relationship to find this $\sigma'(u) = \sigma(u)(1 - \sigma(u))$.
- As exercise prove the following result

$$\frac{\partial}{\partial \mathbf{w}} \log(p(\mathbf{y}|\mathbf{w}, \mathbf{X})) = \sum_{i=1}^{m} (y_i - \underbrace{\sigma(\mathbf{w}'\phi(\mathbf{x}_i))}_{\hat{y}_i})\phi(\mathbf{x}_i) = 0$$

- We now have something which is of a similar form to a LSR and the original reason why this method was called a regression.

BINARY LOGISTIC REGRESSION

- Consider $\mathbf{w} = (\mathrm{w}_0, \mathrm{w}_1, \mathrm{w}_2, \mathrm{w}_3)$ and $\boldsymbol{\phi}(\mathbf{x}_i) = (1, x_1, x_2, x_3)$, so our

$$\hat{y}_i = \sigma(\mathbf{w}'\boldsymbol{\phi}(\mathbf{x}_i)) = \sigma(\mathrm{w}_0 + \mathrm{w}_1 x_1 + \mathrm{w}_2 x_2 + \mathrm{w}_3 x_3)$$

- Credit Risk

$$p(\text{default}|\text{data}) = p(y = 1|\mathbf{w}, \mathbf{X}) = \sigma(\mathbf{w}'\mathbf{X})$$

- Lets look at an example in the session script