Project on Machine Learning

Alfred Alocias Mariadason

June 27, 2018

- Ising Model
- Metha et al, arXiv 1803.08823 accompanied by a Jupyter notebook.

- Ising Model
- Metha et al, arXiv 1803.08823 accompanied by a Jupyter notebook.
- Coupling constant

- Ising Model
- Metha et al, arXiv 1803.08823 accompanied by a Jupyter notebook.
- Coupling constant
- Phases
  - Ordered
  - Critical
  - Disordered

Linear Regression

- $\{y_i, \boldsymbol{x}_i\}_{i=1}^{n}$, $i = 1, \ldots, n$
- $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$
- minimize the L_2-norm $\min_{\boldsymbol{\beta}} \left| \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y} \right|^2$
- Solution: $\boldsymbol{\beta}_{\mathrm{LS}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \left| \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y} \right|^2 \Rightarrow \boldsymbol{\beta}_{\mathrm{LS}} = \left(\boldsymbol{X}^T \boldsymbol{X} \right)^{-1} \boldsymbol{X}^T \boldsymbol{y}$

Linear Regression

- $\{y_i, \boldsymbol{x}_i\}_{i=1}^n, \; i = 1, \ldots, n$
- $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$
- minimize the L_2-norm $\min_{\boldsymbol{\beta}} \left| \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y} \right|^2$
- Solution: $\boldsymbol{\beta}_{\text{LS}} = \operatorname{argmin}_{\boldsymbol{\beta}} \left| \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y} \right|^2 \Rightarrow \boldsymbol{\beta}_{\text{LS}} = \left(\boldsymbol{X}^T \boldsymbol{X} \right)^{-1} \boldsymbol{X}^T \boldsymbol{y}$

Ridge Regression

- L_2-regularization: $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \alpha \sum_{i=1}^{m} \beta_i^2$
- $\boldsymbol{\beta}_{\text{Ridge}} = \operatorname{argmin}_{\boldsymbol{\beta}} \left(\left| \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y} \right|^2 + \alpha \left| \boldsymbol{\beta} \right|^2 \right) \Rightarrow \boldsymbol{\beta}_{\text{Ridge}} = \left(\boldsymbol{X}^T \boldsymbol{X} + \alpha \boldsymbol{I} \right)^{-1} \boldsymbol{X}^T \boldsymbol{y}$

Linear Regression

- $\{y_i, \boldsymbol{x}_i\}_{i=1}^{n}$, $i = 1, \ldots, n$
- $\boldsymbol{y} = \boldsymbol{X\beta} + \boldsymbol{\varepsilon}$
- minimize the L_2-norm $\min\limits_{\boldsymbol{\beta}} \left|\boldsymbol{X\beta} - \boldsymbol{y}\right|^2$
- Solution: $\boldsymbol{\beta}_{\mathrm{LS}} = \operatorname*{argmin}\limits_{\boldsymbol{\beta}} \left|\boldsymbol{X\beta} - \boldsymbol{y}\right|^2 \Rightarrow \boldsymbol{\beta}_{\mathrm{LS}} = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$

Ridge Regression

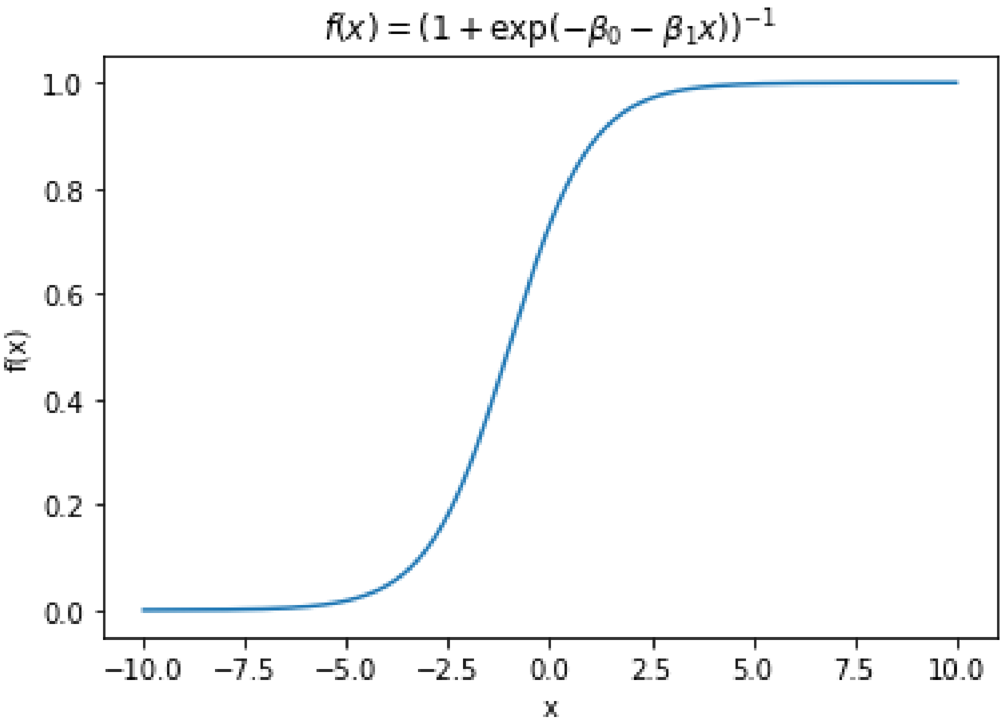
- L_2-regularization: $\boldsymbol{y} = \boldsymbol{X\beta} + \alpha \sum\limits_{i=1}^{m} \beta_i^2$
- $\boldsymbol{\beta}_{\mathrm{Ridge}} = \operatorname*{argmin}\limits_{\boldsymbol{\beta}} \left(\left|\boldsymbol{X\beta} - \boldsymbol{y}\right|^2 + \alpha\left|\boldsymbol{\beta}\right|^2\right) \Rightarrow \boldsymbol{\beta}_{\mathrm{Ridge}} = \left(\boldsymbol{X}^T\boldsymbol{X} + \alpha\boldsymbol{I}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$

Lasso Regression

- L_1-regularization: $\boldsymbol{y} = \boldsymbol{X\beta} + \alpha \sum\limits_{i=1}^{m} \left|\beta_i\right|$
- Constrained ($|\boldsymbol{\beta}| \leq t$): $\boldsymbol{\beta}_{\mathrm{Lasso}} = \operatorname*{argmin}\limits_{\boldsymbol{\beta}} \left(\left|\boldsymbol{X\beta} - \boldsymbol{y}\right|^2 + \alpha\left|\boldsymbol{\beta}\right|\right)$
- Solution: $\beta_j^{\mathrm{Lasso}} = \operatorname{sign}\left(\beta_j^{\mathrm{LS}}\right)\left(\left|\beta_j^{\mathrm{LS}}\right| - \alpha\right)_+$

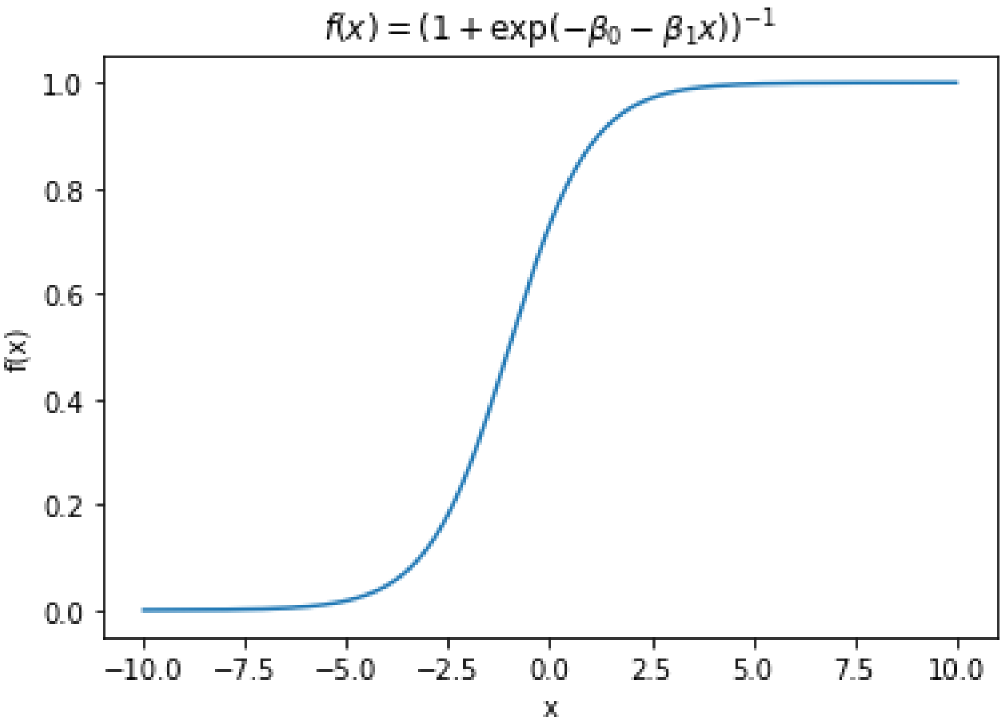
Logistic Regression

- Fit to sigmoid $f(x) = \frac{1}{1+\exp(-\boldsymbol{x}^T \boldsymbol{w})}$



Logistic Regression

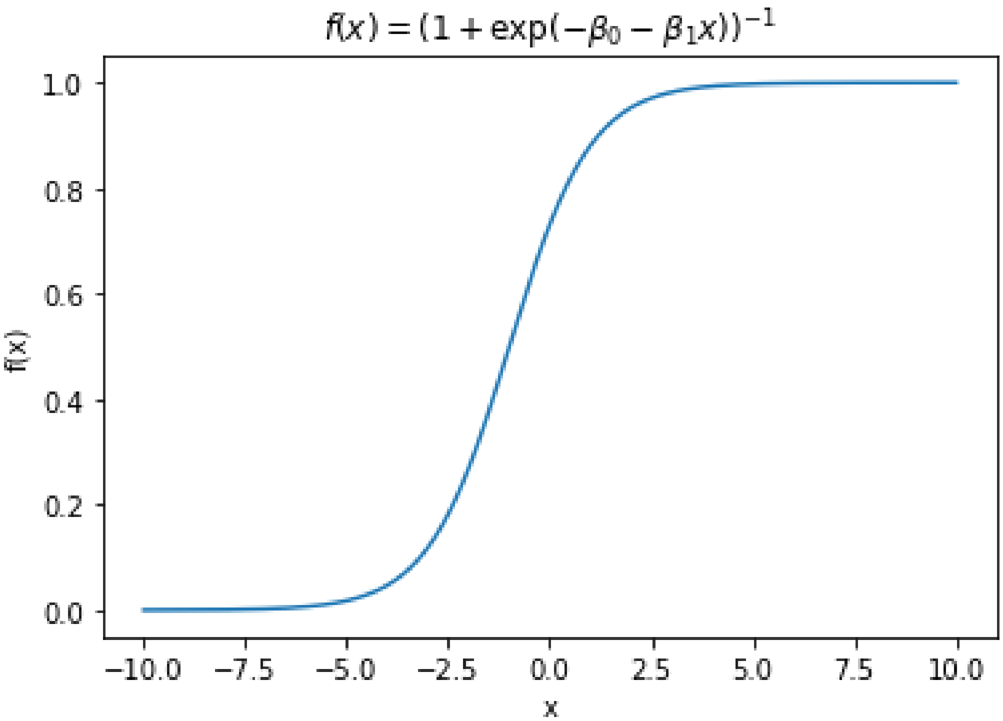
- Fit to sigmoid $f(x) = \frac{1}{1+\exp(-\boldsymbol{x}^T \boldsymbol{w})}$



- Categorisation

Logistic Regression

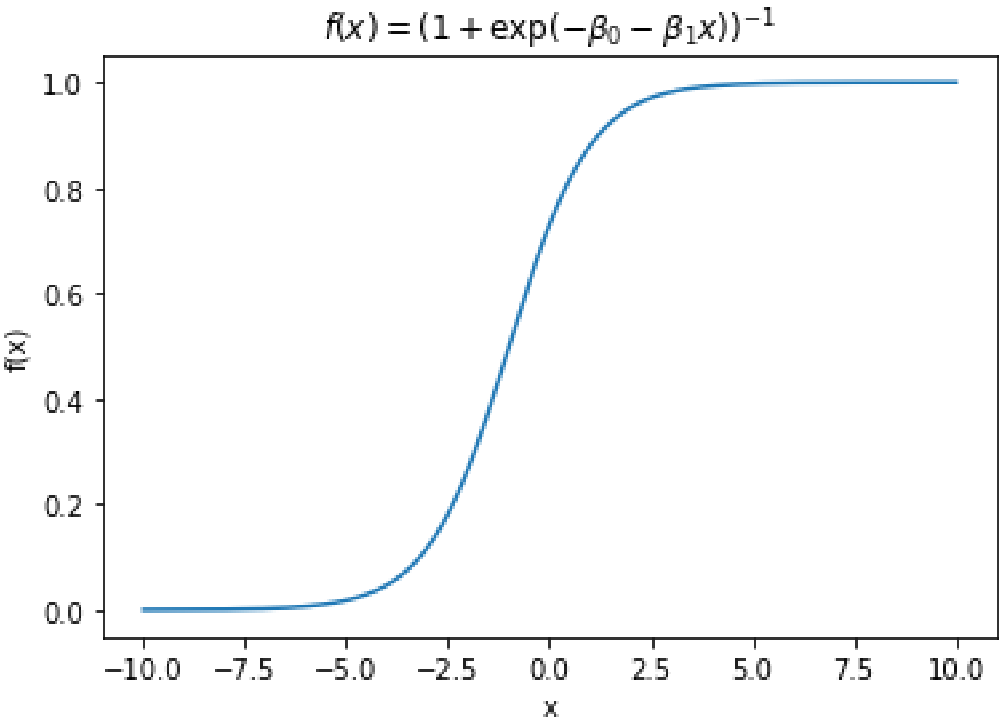
- Fit to sigmoid $f(x) = \frac{1}{1+\exp(-\boldsymbol{x}^T\boldsymbol{w})}$



- Categorisation
  - Binary: $P(Y|X)$

Logistic Regression

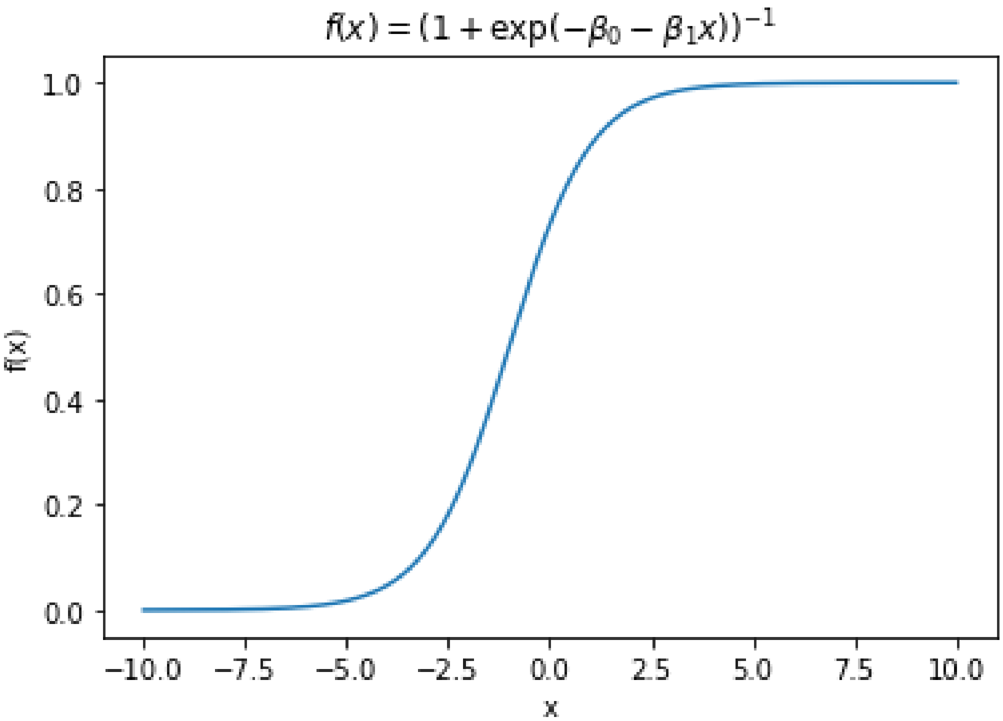
- Fit to sigmoid $f(x) = \frac{1}{1+\exp(-\boldsymbol{x}^T\boldsymbol{w})}$



- Categorisation
  - Binary: $P(Y|X)$
- Cost: $C(\boldsymbol{w}) = -\sum_{i=1}^{n} \left[Y_i \log\left(f\left(\boldsymbol{X}_i^T\boldsymbol{w}\right)\right) + (1 - Y_i)\log\left(1 - f\left(\boldsymbol{X}_i^T\boldsymbol{w}\right)\right)\right]$

Logistic Regression

- Fit to sigmoid $f(x) = \frac{1}{1+\exp(-\boldsymbol{x}^T \boldsymbol{w})}$



- Categorisation
  - Binary: $P(Y|X)$
- Cost: $C(\boldsymbol{w}) = -\sum_{i=1}^{n} \left[Y_i \log \left(f \left(\boldsymbol{X}_i^T \boldsymbol{w} \right) \right) + (1 - Y_i) \log \left(1 - f \left(\boldsymbol{X}_i^T \boldsymbol{w} \right) \right) \right]$
- Numerical minimization: $\mathbf{0} = \nabla C(\boldsymbol{w}) = \sum_{i=1}^{n} \left[f \left(\boldsymbol{X}_i^T \boldsymbol{w} \right) - Y_i \right] \boldsymbol{X}_i$

Random Forest Algorithm

- Consists of Decision Trees

Random Forest Algorithm

- Consists of Decision Trees
- Bagging: Create bootstrap ensemble from data

Random Forest Algorithm

- Consists of Decision Trees
- Bagging: Create bootstrap ensemble from data
- Feature Bagging: Bagging on data and labels

Random Forest Algorithm

- Consists of Decision Trees
- Bagging: Create bootstrap ensemble from data
- Feature Bagging: Bagging on data and labels
- Extremized Random Forest: Split decision trees at random

Neural Networks

- input: $\boldsymbol{x} = \begin{pmatrix} x_{11} & \dots & x_{1d} \\ \vdots & \ddots & \vdots \\ x_{d1} & \dots & x_{nd} \end{pmatrix}$

Neural Networks

- input: $\boldsymbol{x} = \begin{pmatrix} x_{11} & \dots & x_{1d} \\ \vdots & \ddots & \vdots \\ x_{d1} & \dots & x_{nd} \end{pmatrix}$
- $\{\boldsymbol{w}\}_{l=1}^{m} = \left(\boldsymbol{w}^{(1)}, \dots, \boldsymbol{w}^{(m)}\right)$

Neural Networks

- input: $\boldsymbol{x} = \begin{pmatrix} x_{11} & \dots & x_{1d} \\ \vdots & \ddots & \vdots \\ x_{d1} & \dots & x_{nd} \end{pmatrix}$
- $\{\boldsymbol{w}\}_{l=1}^{m} = \left(\boldsymbol{w}^{(1)}, \dots, \boldsymbol{w}^{(m)}\right)$
- Bias: $z^{(l)} = \boldsymbol{w}^{(l)} \cdot \boldsymbol{x} + b^{(l)}$

Neural Networks

- input: $\boldsymbol{x} = \begin{pmatrix} x_{11} & \dots & x_{1d} \\ \vdots & \ddots & \vdots \\ x_{d1} & \dots & x_{nd} \end{pmatrix}$
- $\{\boldsymbol{w}\}_{l=1}^{m} = \left(\boldsymbol{w}^{(1)}, \dots, \boldsymbol{w}^{(m)}\right)$
- Bias: $z^{(l)} = \boldsymbol{w}^{(l)} \cdot \boldsymbol{x} + b^{(l)}$
- Backpropagation:
  - **Activation at input layer**: Calculate activations $a_j^l = \sigma\left(\sum_k w_{kj}^l a_k^{l-1}\right)$
  - **Feedforward**: Compute z^l and a^l for subsequent layers
  - **Error at output**: Calculate error in output
  - **Backpropagate**: Calculate error for all layers
  - **Calculate Gradient**: Calculate gradient and update weights

Neural Networks

- input: $\boldsymbol{x} = \begin{pmatrix} x_{11} & \dots & x_{1d} \\ \vdots & \ddots & \vdots \\ x_{d1} & \dots & x_{nd} \end{pmatrix}$
- $\{\boldsymbol{w}\}_{l=1}^{m} = \left(\boldsymbol{w}^{(1)}, \dots, \boldsymbol{w}^{(m)}\right)$
- Bias: $z^{(l)} = \boldsymbol{w}^{(l)} \cdot \boldsymbol{x} + b^{(l)}$
- Backpropagation:
  - **Activation at input layer**: Calculate activations $a_j^l = \sigma\left(\sum_k w_{kj}^l a_k^{l-1}\right)$
  - **Feedforward**: Compute z^l and a^l for subsequent layers
  - **Error at output**: Calculate error in output
  - **Backpropagate**: Calculate error for all layers
  - **Calculate Gradient**: Calculate gradient and update weights
- Cost function

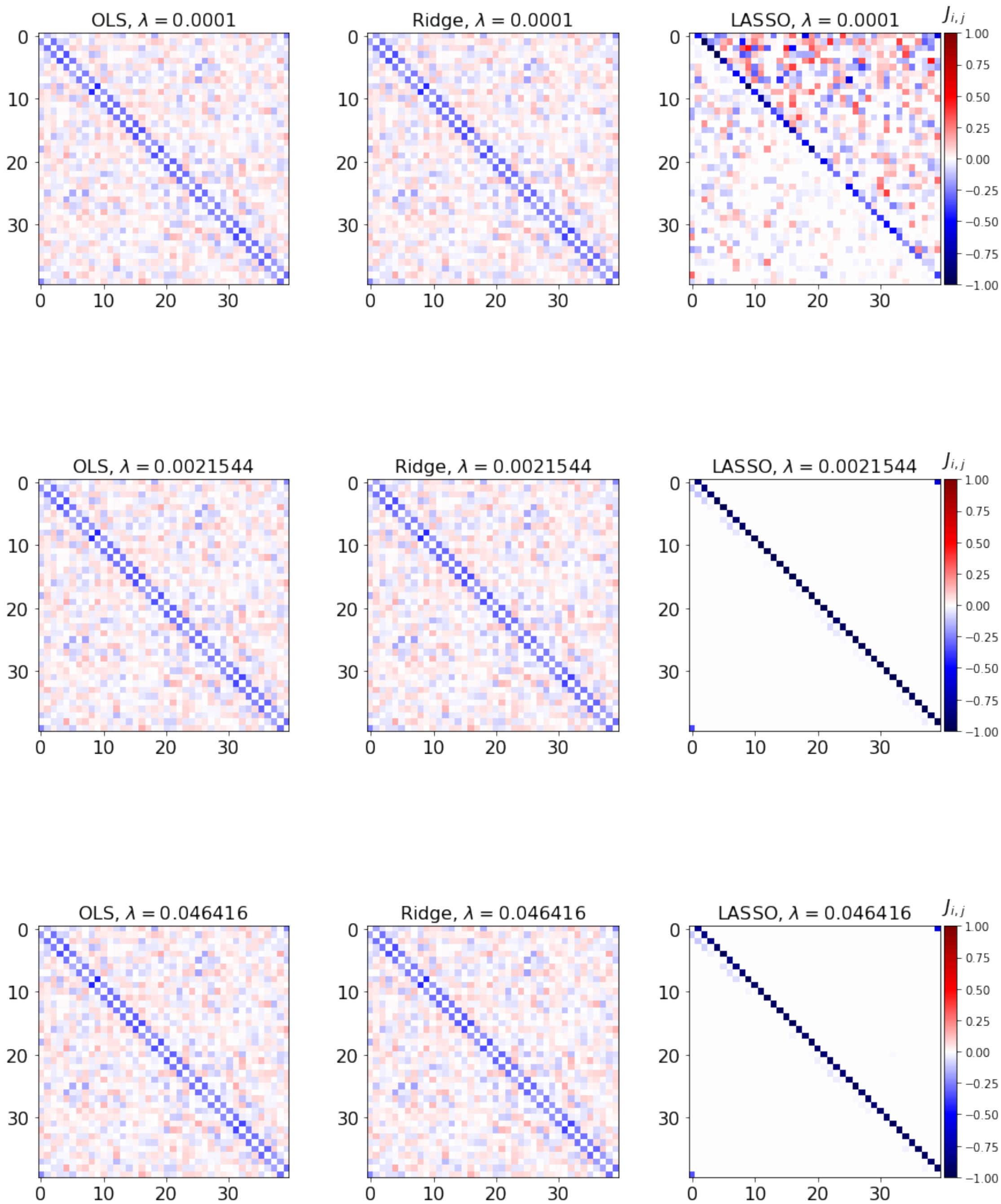
Estimating the Coupling Constant of the 1D Ising Model

- Data with $J = 1$

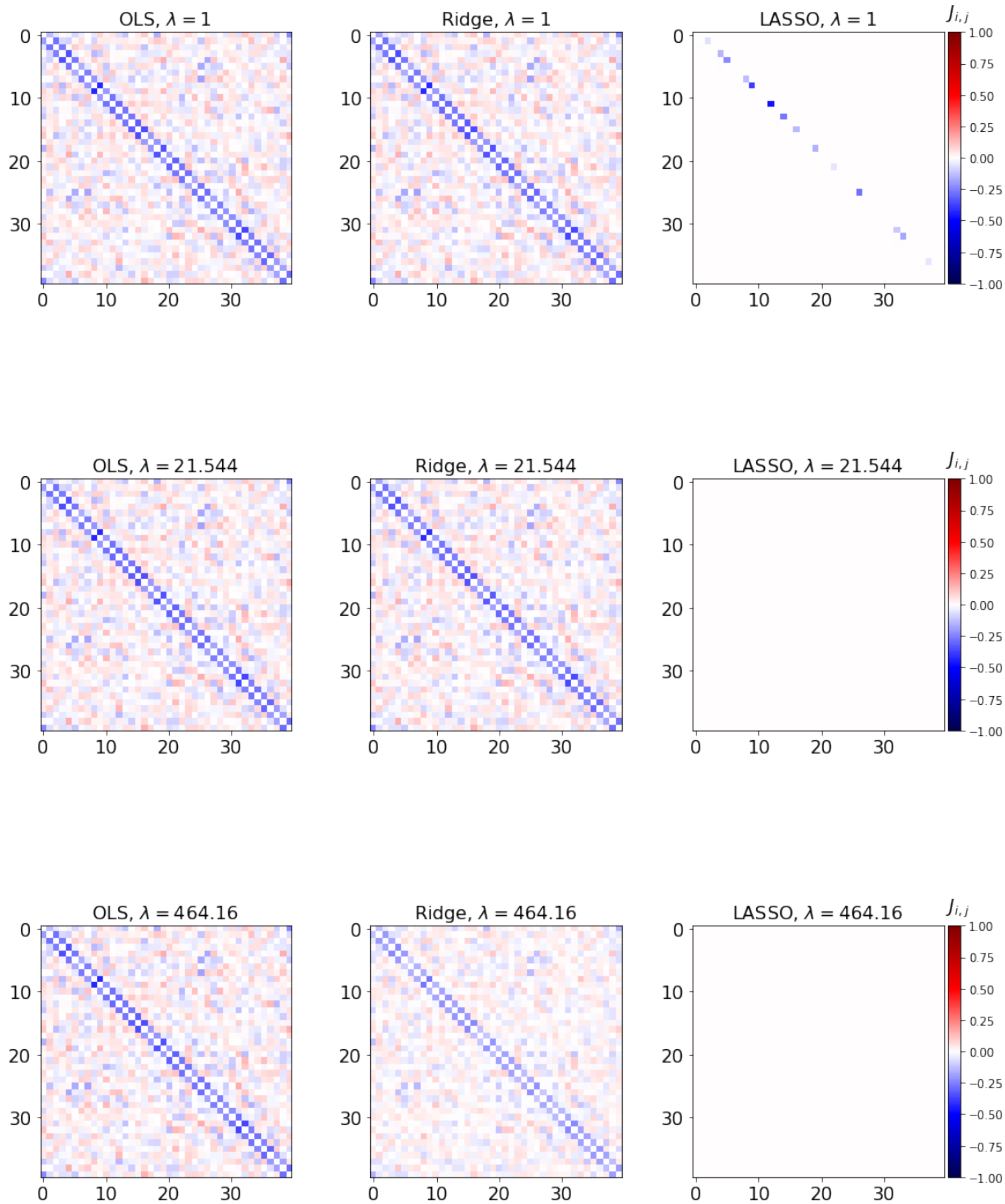
Estimating the Coupling Constant of the 1D Ising Model

- Data with $J = 1$
- $E^{(i)} = -\boldsymbol{X}^{(i)} \cdot \boldsymbol{J}$

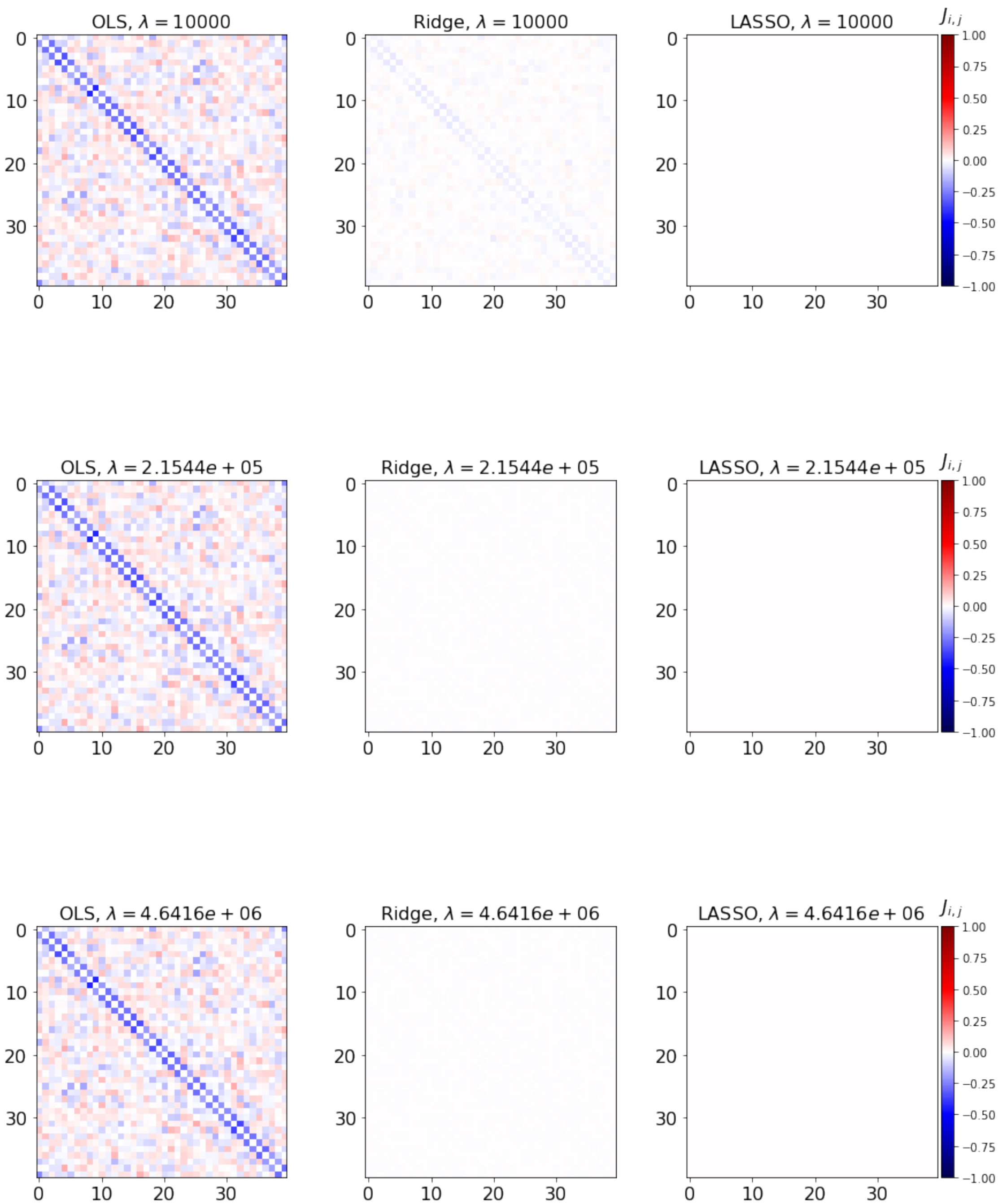
Estimating the Coupling Constant of the 1D Ising Model



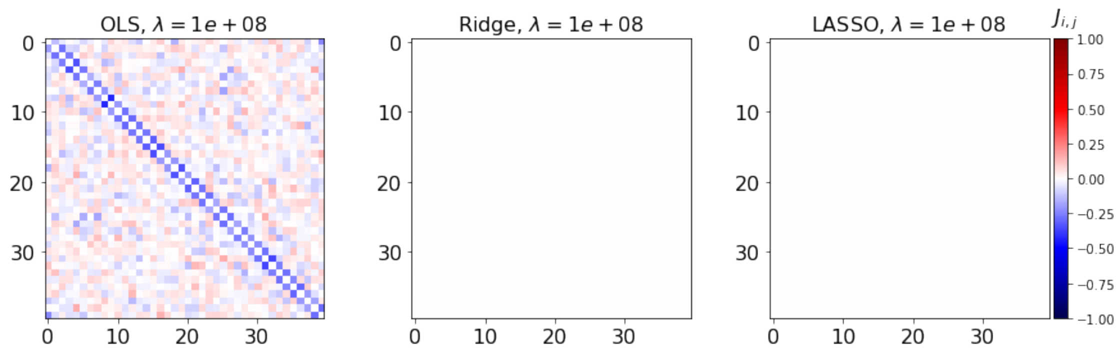
Estimating the Coupling Constant of the 1D Ising Model



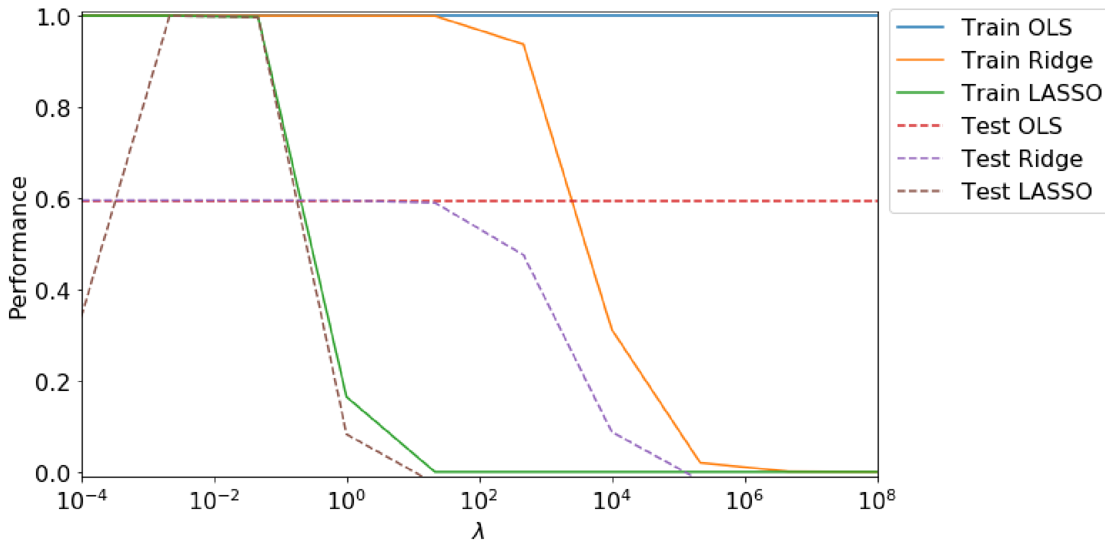
Estimating the Coupling Constant of the 1D Ising Model



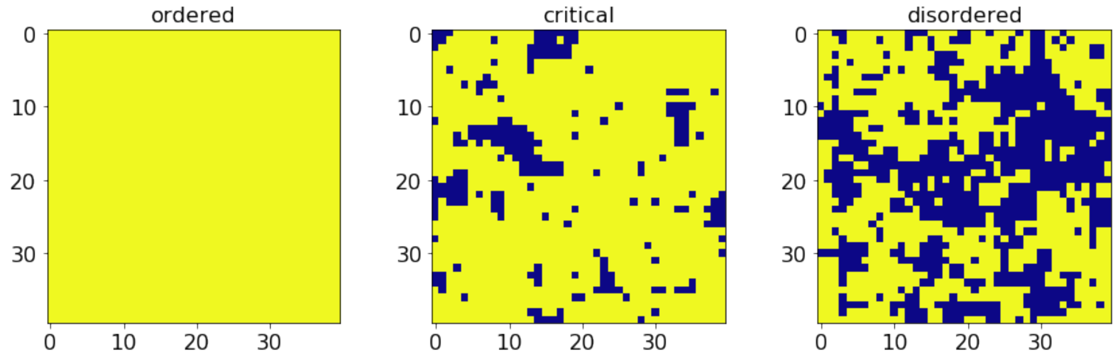
Estimating the Coupling Constant of the 1D Ising Model



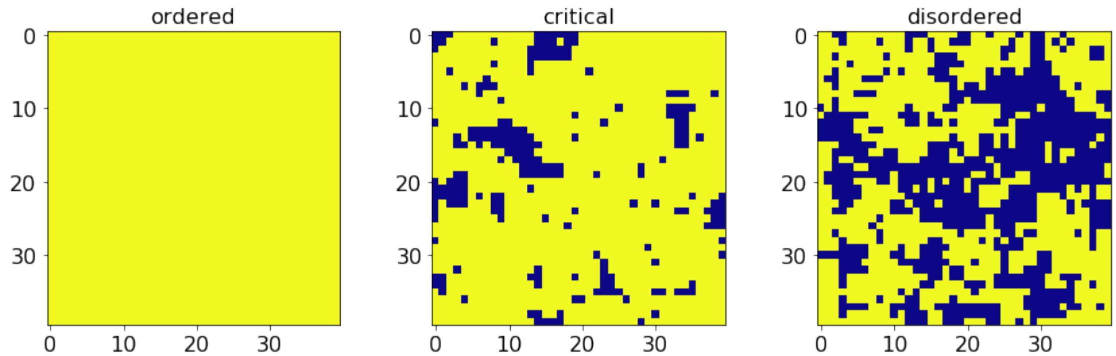
Estimating the Coupling Constant of the 1D Ising Model

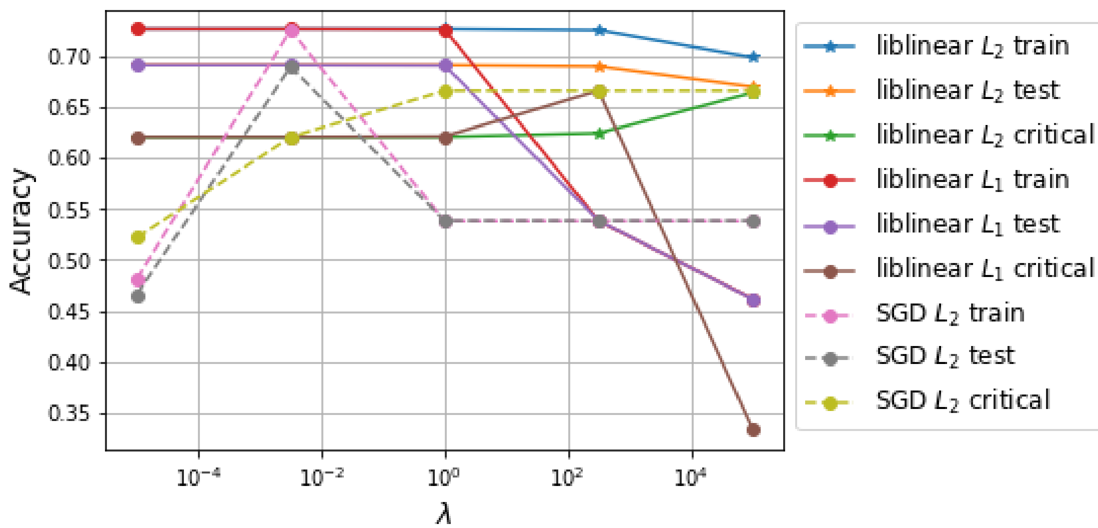


Determine the Phase of the Two-Dimensional Ising Model

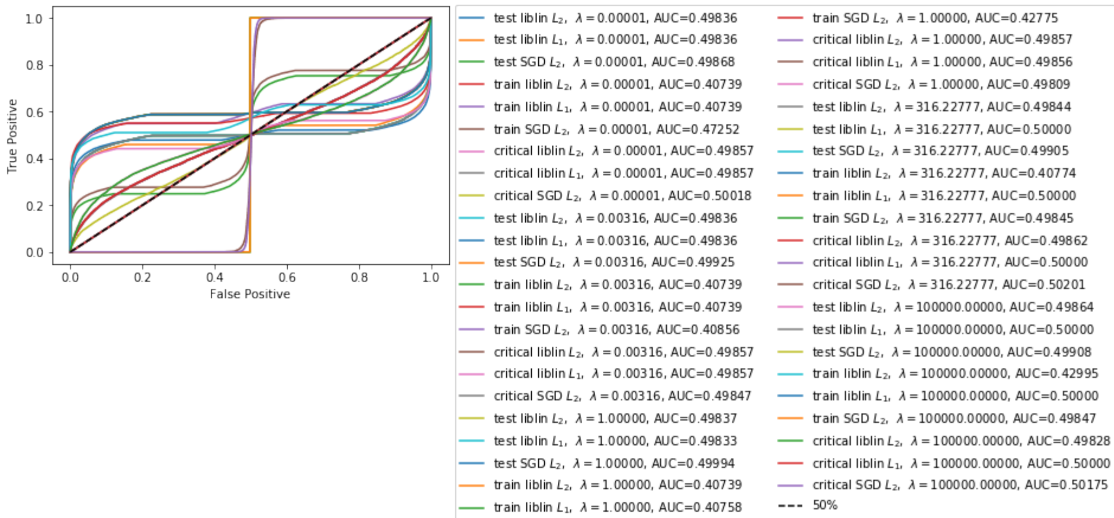


Determine the Phase of the Two-Dimensional Ising Model

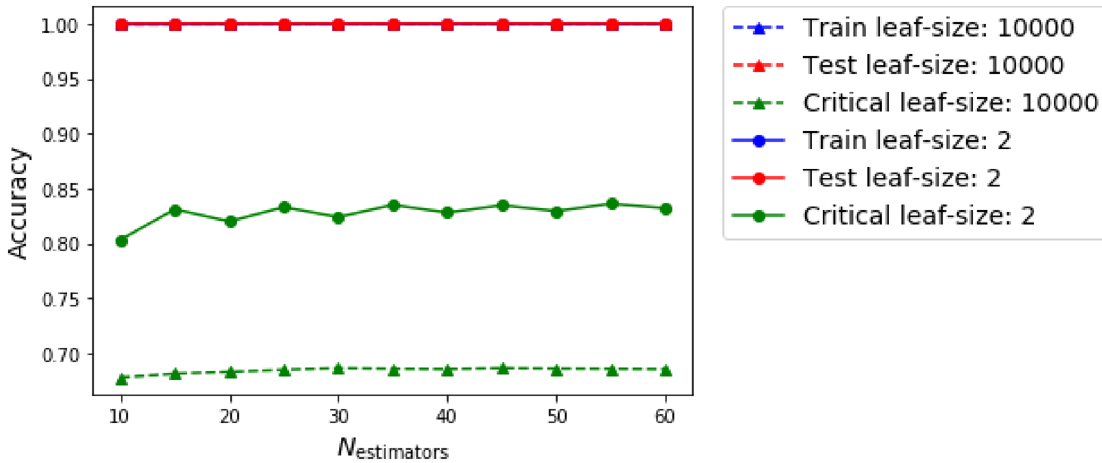




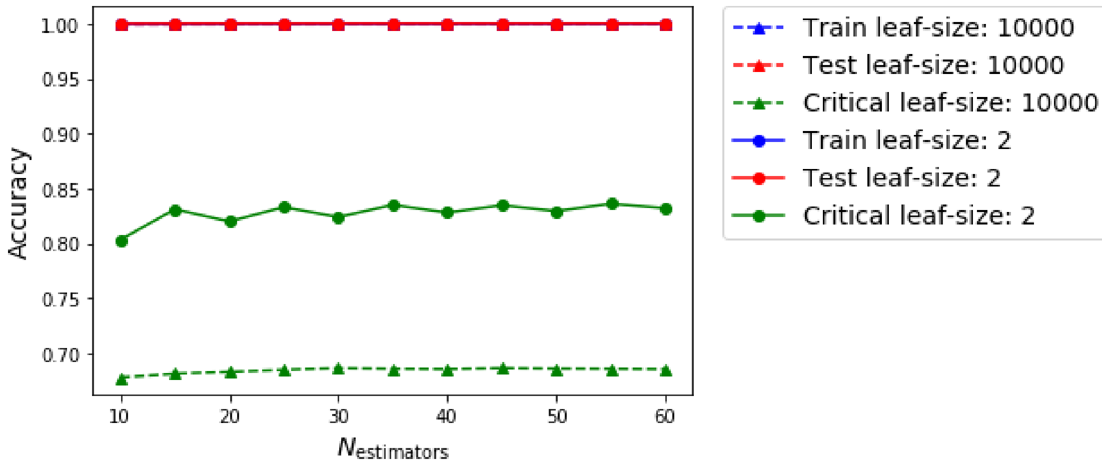
Determine the Phase of the Two-Dimensional Ising Model

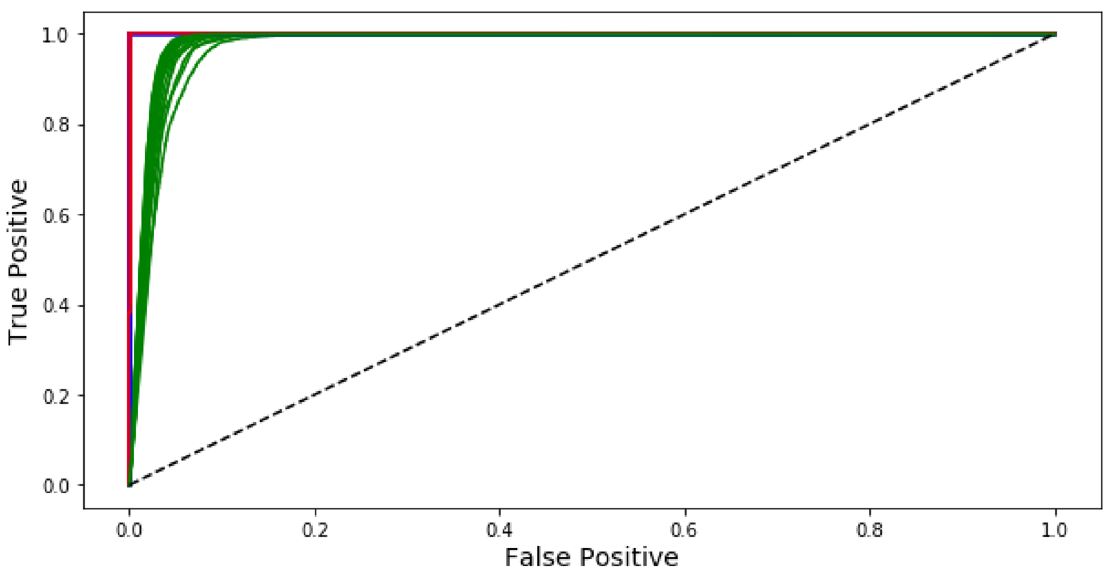


0.0.1 Using Random Forest to Classify Phases in the Ising Model

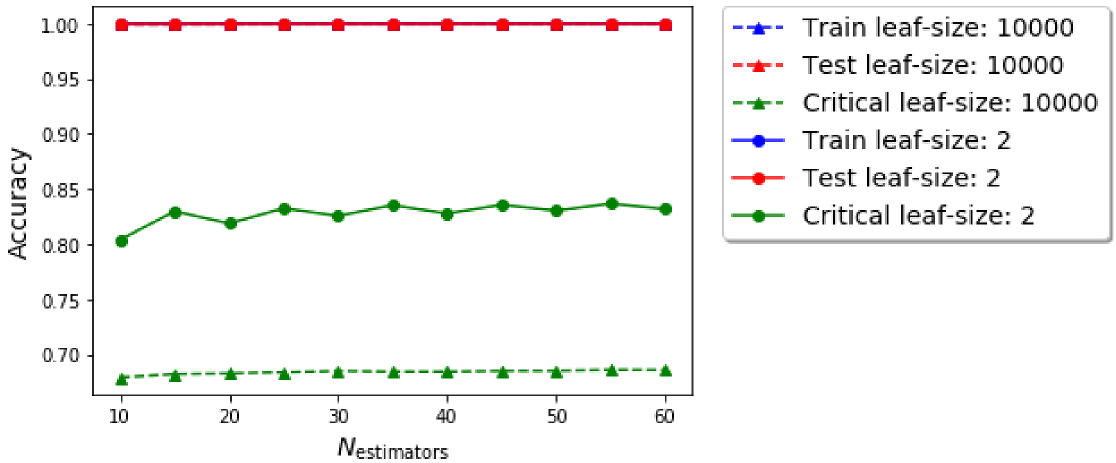


0.0.2 Using Random Forest to Classify Phases in the Ising Model

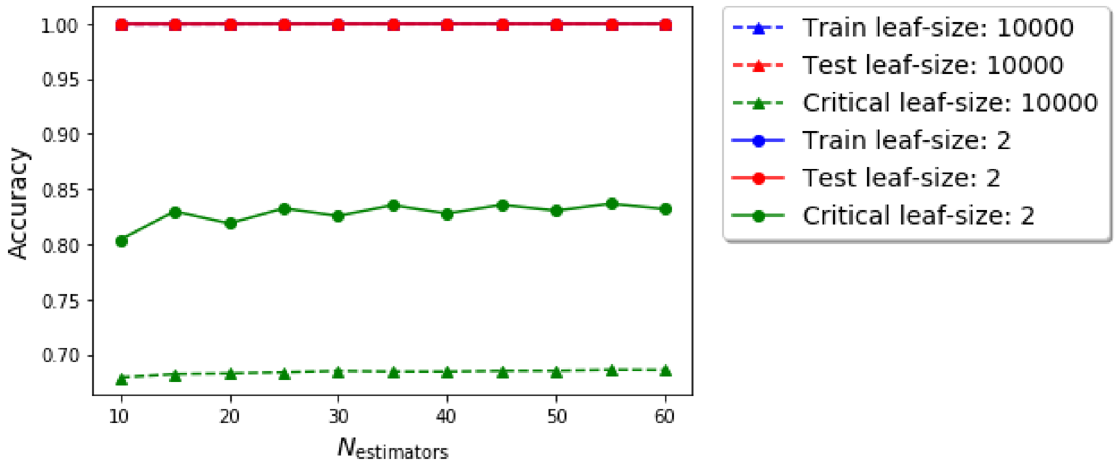


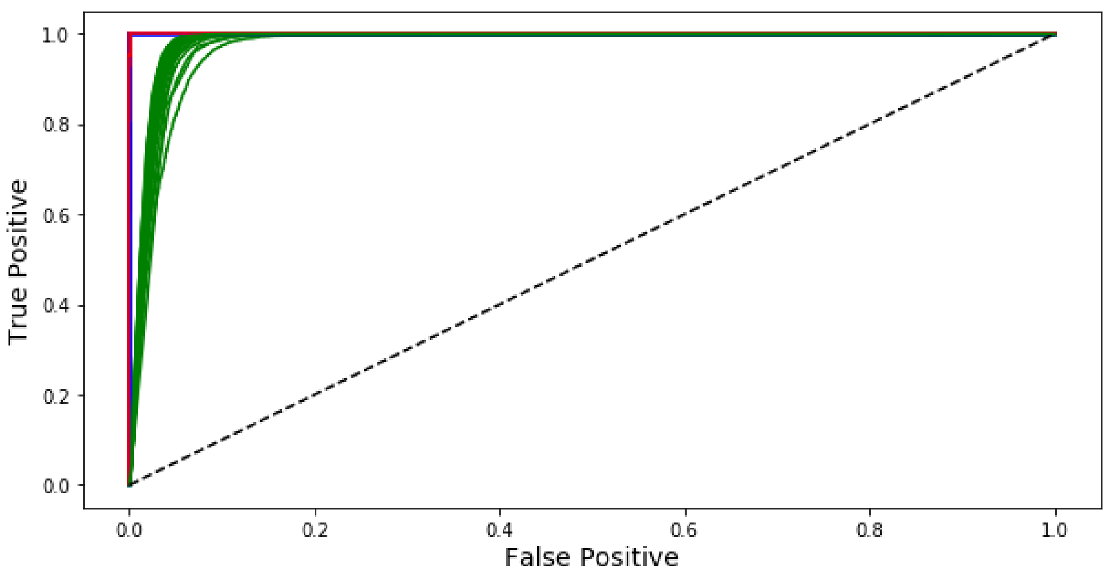


0.0.3 Using Random Forest to Classify Phases in the Ising Model

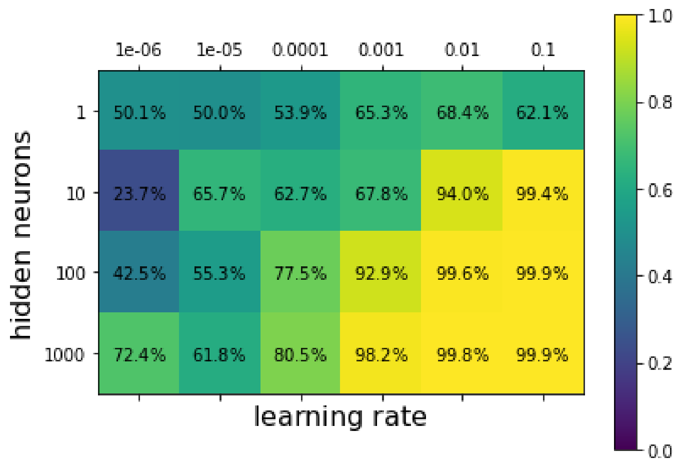


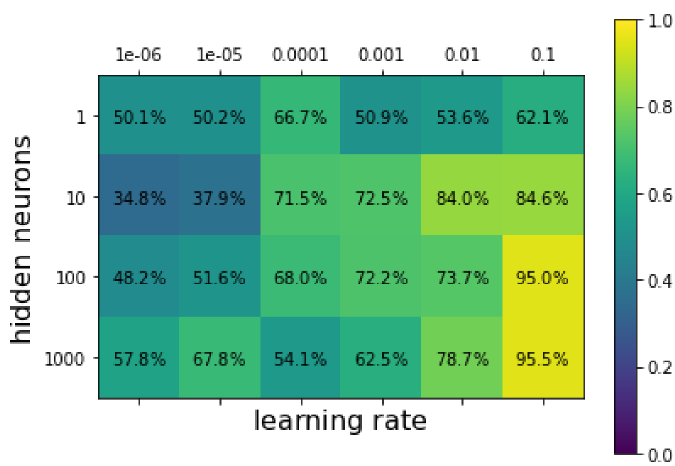
0.0.4 Using Random Forest to Classify Phases in the Ising Model





0.0.5 Classifying the Ising Model Phase Using Neural Networks





0.0.6 Classifying the Ising Model Phase Using Neural Networks