DS-UA 301
Advanced Topics in Data Science
Advanced Techniques in ML and Deep Learning

LECTURE 2

Parijat Dube

Today's Agenda

- How to choose the right model?
  - Performance vs complexity of model
  - TSS, RSS, R^2, MSE
  - Underfitting, overfitting
  - Bias-variance tradeoff
- ML performance improvement techniques
  - Regularization techniques
  - Cross-validation

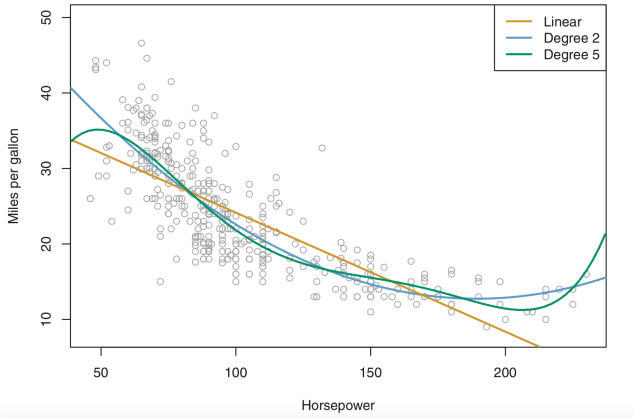
ML Tasks: Classification and Regression

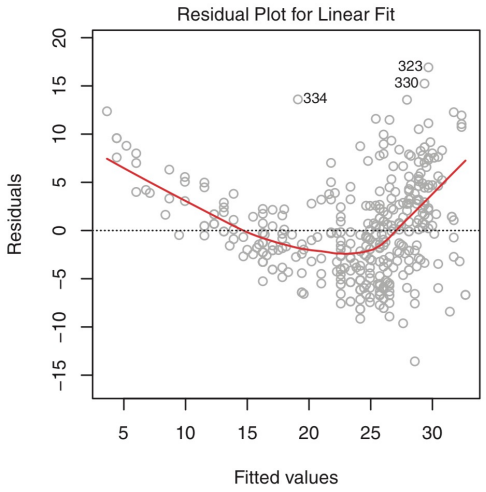
- **Regression**: Output is a continuous valued real variable
  - Stock price prediction
  - House value price prediction

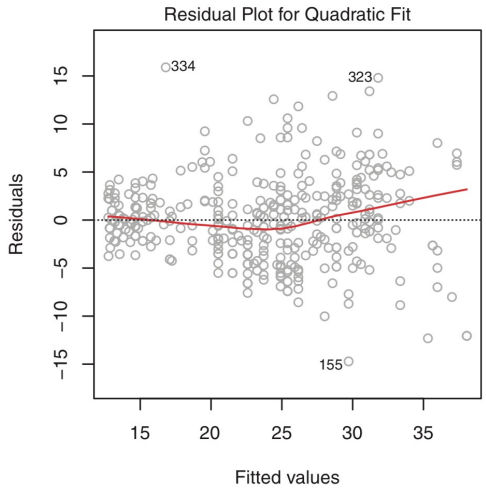
- **Classification**: Output is a categorical, unordered variable
  - Binary classification
    - Tumor: benign vs malignant
    - Food image : Greek vs Non-Greek
  - Multiclass classification
    - Tumor: benign, stage-1, stage-2, stage-3
    - Food image: falafel, salad, pita, …

Linear Regression

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p$$







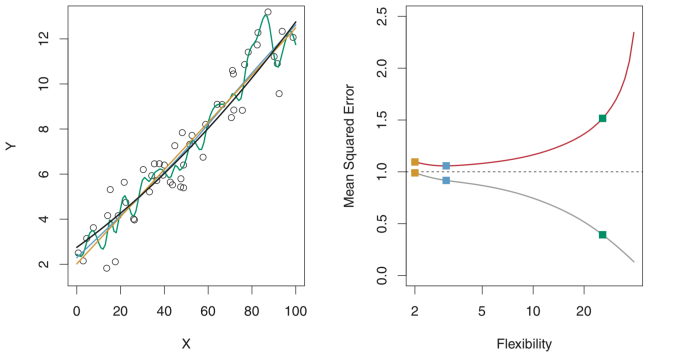
$$\text{TSS} = \sum (y_i - \bar{y})^2$$

$$\text{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

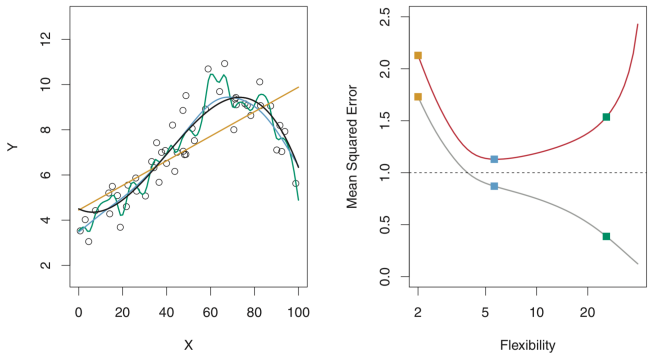
$$R^2 = \frac{\text{TSS} - \text{RSS}}{\text{TSS}} = 1 - \frac{\text{RSS}}{\text{TSS}}$$

Mean Square Error (MSE)

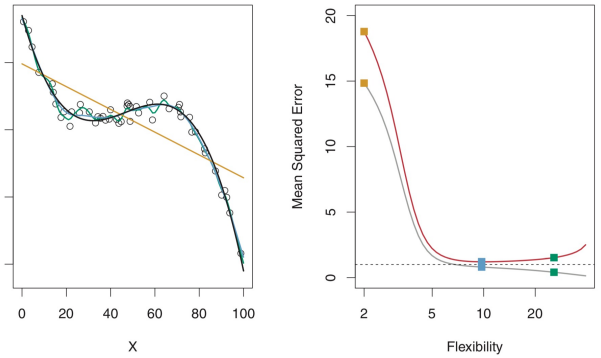
CASE 1



CASE 2



CASE 3



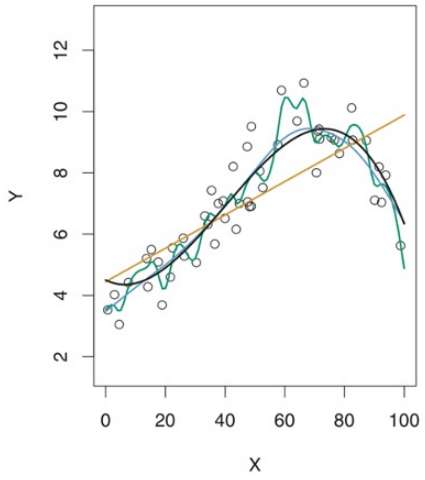
true value $Y = f(X) + \epsilon.$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

predicted $\hat{Y} = \hat{f}(X)$

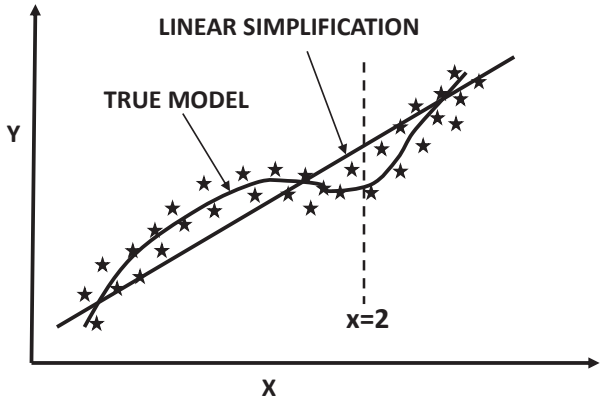
Overfitting and Underfitting

- Overfitting: model performs well on training data but does not generalize well to unseen data (test data)
- Underfitting: model is not complex enough to capture pattern in the training data well and therefore suffers from low performance on unseen data



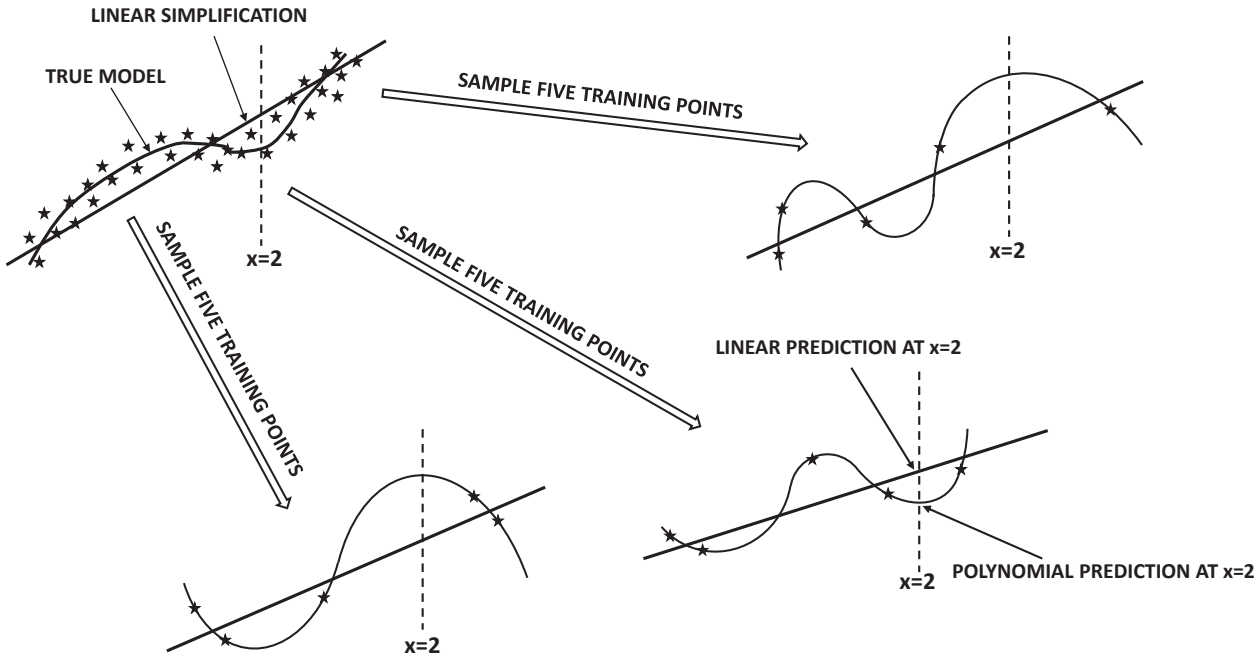
Bias of a model

Example: Predict y from x



- **First impression:** Polynomial model such as $y = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4$ is "better" than linear model $y = w_0 + w_1 x$.

Different Training Data Sets with Five Points



- Zero error on training data but wildly varying predictions of $x = 2$

Observations

- The higher-order model is more complex than the linear model and has less *bias*.
  - But it has more parameters.
  - For a small training data set, the learned parameters will be more sensitive to the nuances of that data set.
  - Different training data sets will provide different predictions for y at a particular x.
  - This variation is referred to as model *variance*.

Noise Component

- Unlike bias and variance, noise is a property of the *data* rather than the model.

- Noise refers to unexplained variations ϵ_i of data from true model $y_i = f(x_i) + \epsilon_i$.

- Real-world examples:

  – Human mislabeling of test instance $\Rightarrow$ Ideal model will never predict it accurately.

  – Error during collection of temperature due to sensor malfunctioning.

- Cannot do anything about it even if seeded with knowledge about true model.

Bias-Variance Trade-off: Setup

- Imagine you are given the true distribution $\mathcal{B}$ of training data (including labels).
- You have a principled way of sampling data sets $\mathcal{D} \sim \mathcal{B}$ from the training distribution.
- Imagine you create an infinite number of training data sets (and trained models) by repeated sampling.
- You have a *fixed* set $\mathcal{T}$ of unlabeled test instances.
  - The test set $\mathcal{T}$ does not change over different training data sets.
  - Compute prediction of each instance in $\mathcal{T}$ for each trained model.

Informal Definition of Bias

- Compute averaged prediction of each test instance x over different training models $g(x, \mathcal{D})$.

- Averaged prediction of test instance will be different from true (unknown) model $f(x)$.

- Difference between (averaged) $g(x, \mathcal{D})$ and $f(x)$ caused by erroneous assumptions/simplifications in modeling $\Rightarrow$ Bias

  - **Example:** Linear simplification to polynomial model causes bias.

  - If the true (unknown) model $f(x)$ were an order-4 polynomial, and we used any polynomial of order-4 or greater in $g(x, \mathcal{D})$, bias would be 0.

Informal Definition of Variance

- The value $g(x, \mathcal{D})$ will vary with $\mathcal{D}$ for fixed x.
  - The prediction of the same test instance will be different over different trained models.
- All these predictions cannot be simultaneously correct $\Rightarrow$ Variation contributes to error
- Variance of $g(x, \mathcal{D})$ over different training data sets $\Rightarrow$ Model Variance
  - **Example:** Linear model will have low variance.
  - Higher-order model will have high variance.

Bias-Variance Equation

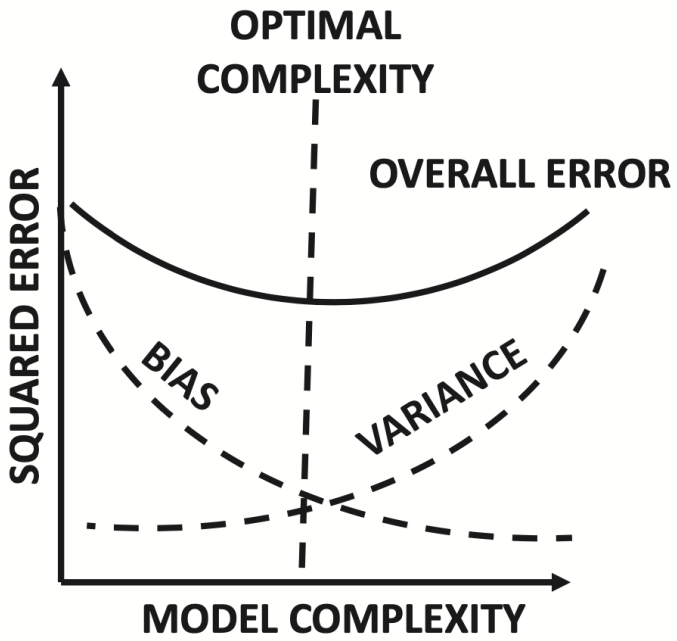
- Let $E[MSE]$ be the expected mean-squared error of the fixed set of test instances over different samples of training data sets.

$$E[MSE] = \text{Bias}^2 + \text{Variance} + \text{Noise} \quad (1)$$

  - In linear models, the bias component will contribute more to $E[MSE]$.
  - In polynomial models, the variance component will contribute more to $E[MSE]$.

- We have a trade-off, when it comes to choosing model complexity!

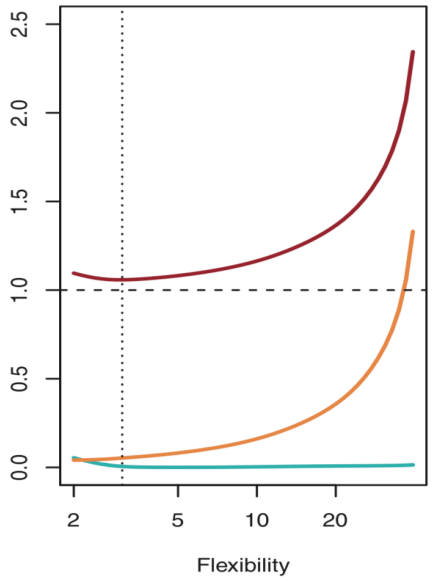
The Bias-Variance Trade-Off



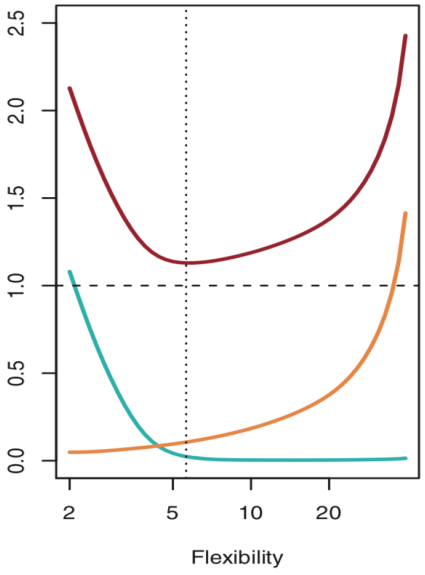
- Optimal point of model complexity is somewhere in middle.

Model Complexity Tradeofffs

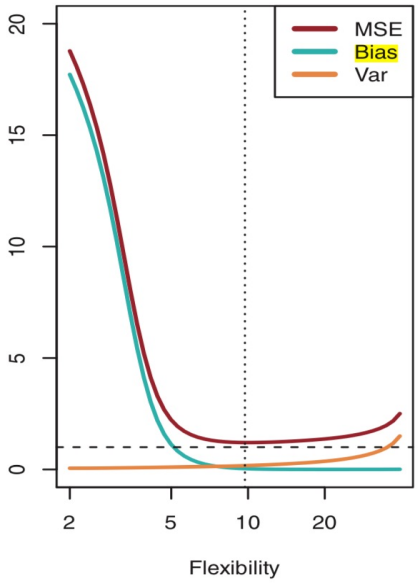
CASE 1



CASE 2



CASE 3

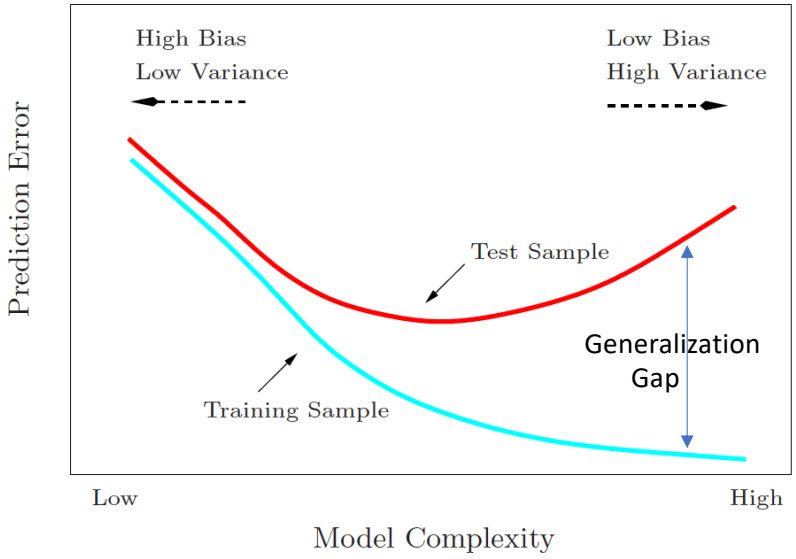


$$E\left(y_0 - \hat{f}(x_0)\right)^2 = \mathrm{Var}(\hat{f}(x_0)) + [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\epsilon)$$

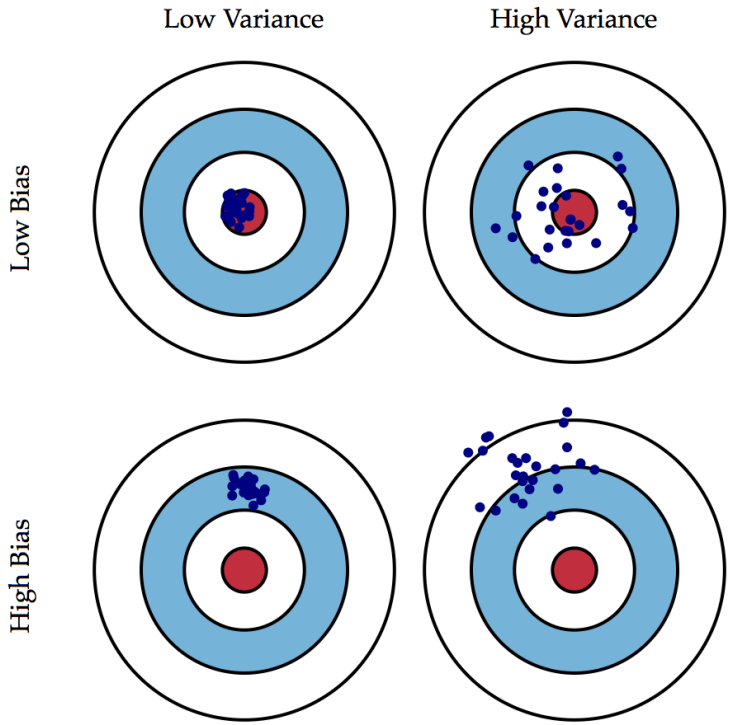
Irreducible

Model Complexity Tradeoffs

- Simple model
  - Fail to completely capture the relationship between features
  - Introduce bias: Consistent test error across different choices of training data
  - Low variance
  - Increasing training data does not help in reducing bias
- Complex model captures nuances in training data causing Overfitting
  - Low bias
  - Train error << Test error
  - With different training instances, the model prediction for same test instance will be very different – High Variance
- Variance does not depend on the true value of the test data



Bias-Variance Tradeoff



$$MSE_{test} = Bias^2 + Variance + Irreducible\ error$$

High Bias

Increase model complexity

Low Bias

High Variance

Increase training data

Low Variance

Key Takeaway of Bias-Variance Trade-Off

- A model with greater complexity might be *theoretically* more accurate (i.e., low bias).
  - But you have less control on what it might predict on a tiny training data set.
  - Different training data sets will result in widely *varying* predictions of same test instance.
  - Some of these must be wrong $\Rightarrow$ Contribution of model variance.
- *A more accurate model for infinite data is not a more accurate model for finite data.*
  - Do not use a sledgehammer to swat a fly!

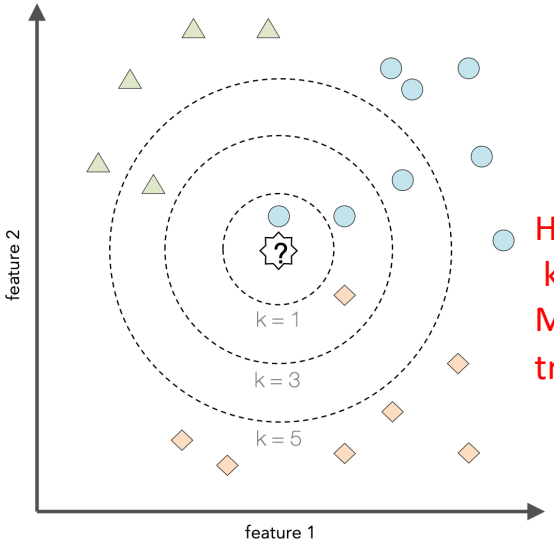
Bias-variance tradeoff example

- https://dustinstansbury.github.io/theclevermachine/bias-variance-tradeoff

Hyperparameters vs Model parameters

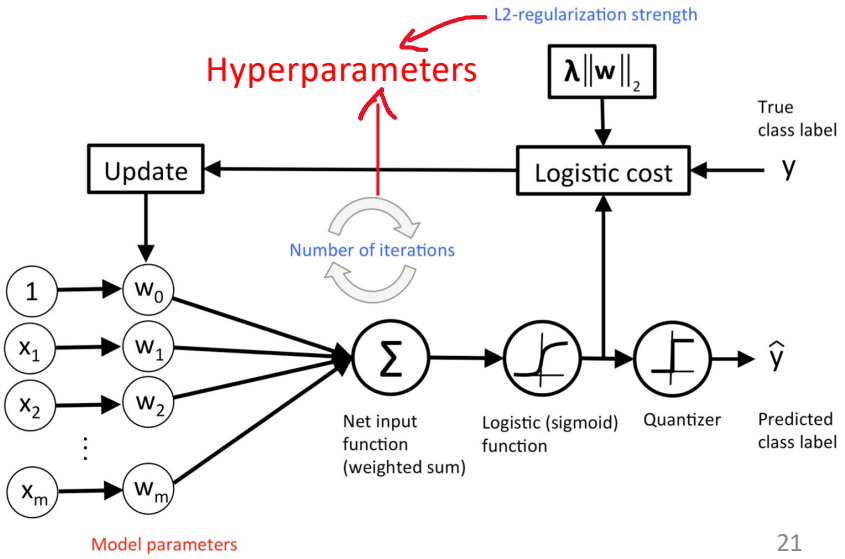
- Hyperparameters: parameters of the learning algorithm, which needs to specified a priori – before model fitting
- Model parameters: learned parameters of the model

k-nearest neighbor



Hyperparameters:
k, distance metric
Model parameters:
training instance

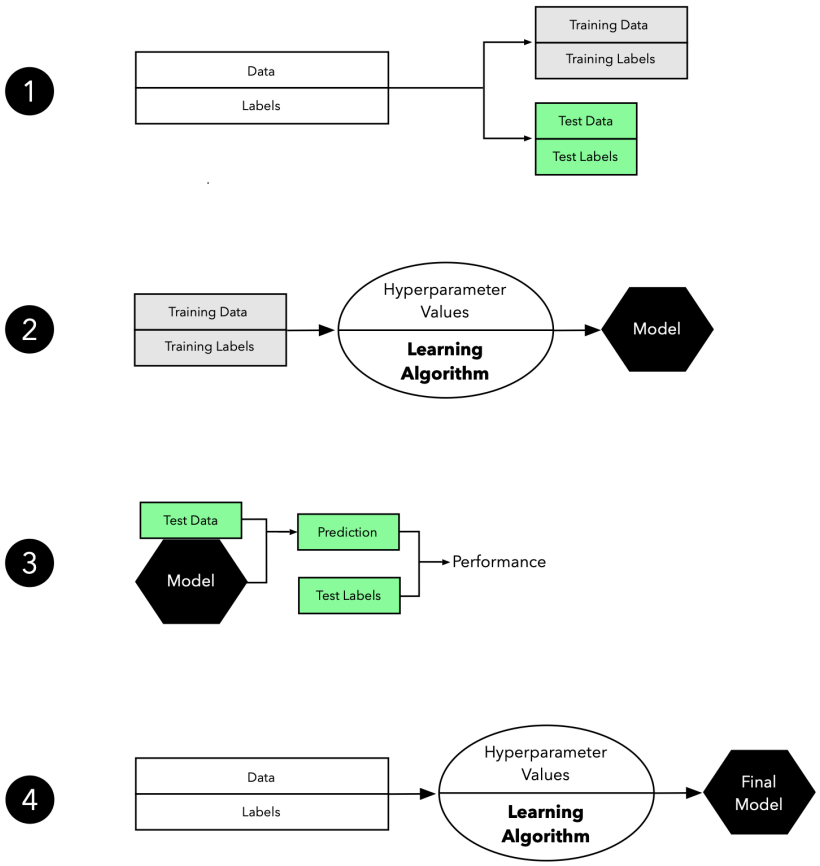
Logistic regression



Model Evaluation

- Estimate the generalization performance
  - What is the performance of my model on unseen dataset
- Improve predictive performance by tweaking the learning algorithm and selecting the best performing model (Hyperparameter Tuning and Model Selection)
  - What hyperparameter values gives the best performance
  - Computational requirements can be considered
- Identify the best machine learning algorithm (Algorithm Selection)
  - What is the best algorithm

Holdout Validation



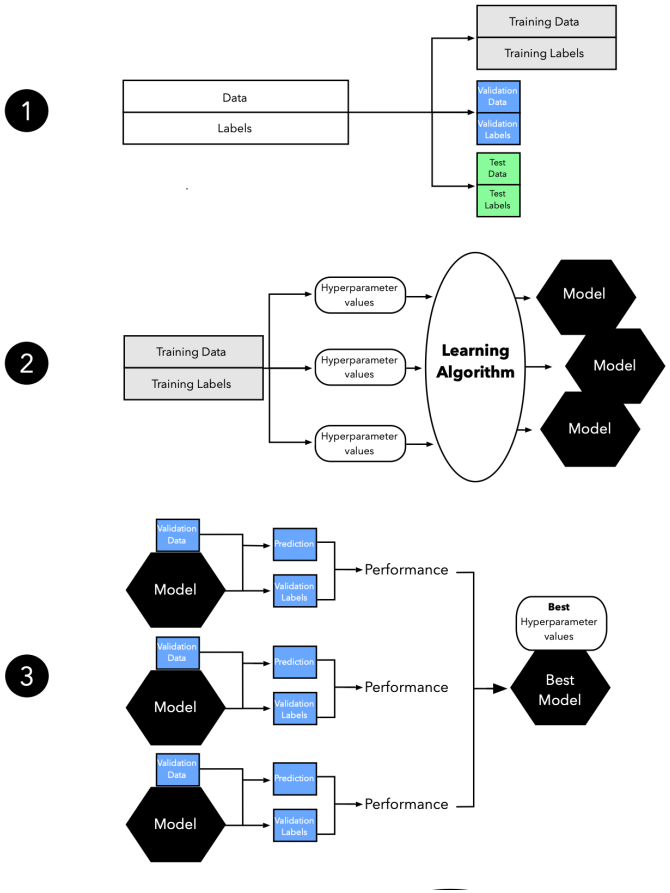
Training set	**Test set**
model training	final performance

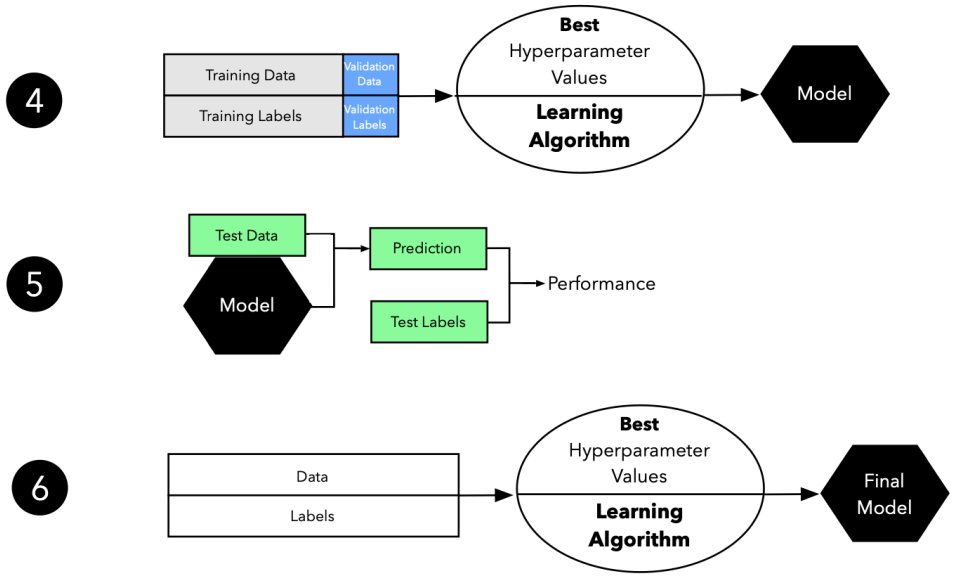
Model Evaluation using Cross-Validation

- Training data: model training, i.e., to learn model parameters
- Validation data: model evaluation for hyperparameter tuning/model selection
- Test data: final performance of the tuned and trained model
- **3-way Holdout technique**
  - Data training divide into 2 subsets: training set and validation set
  - Train on training set and evaluate model performance on validation set

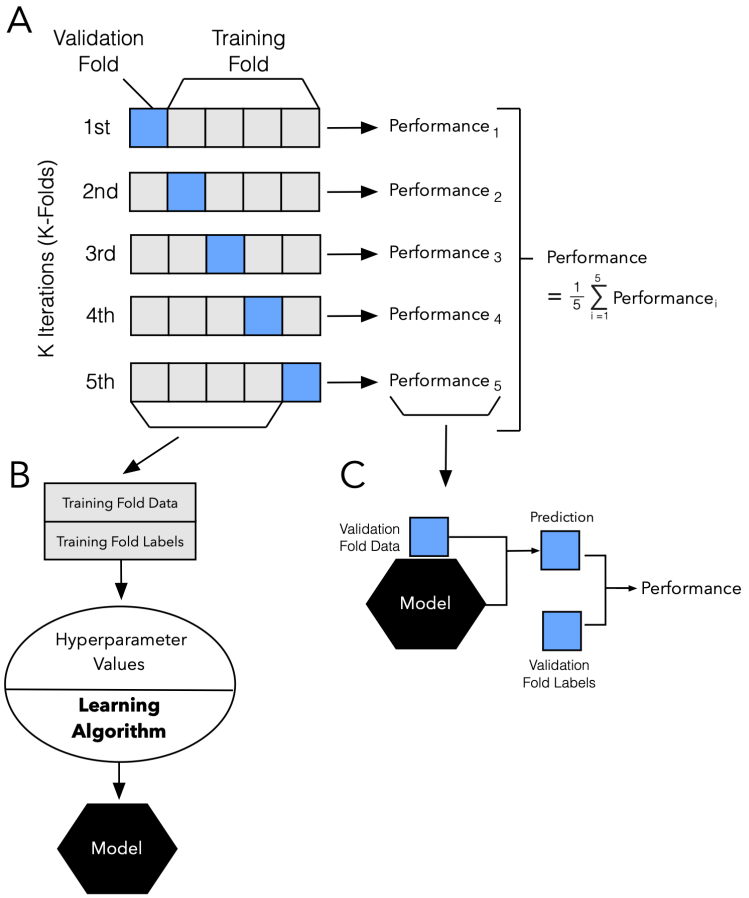
Training set	**Validation set**	**Test set**
model training	hyperparameter tuning/model selection	final performance

3-way Holdout for Model Selection





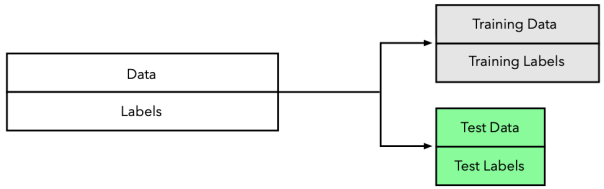
K-fold cross validation



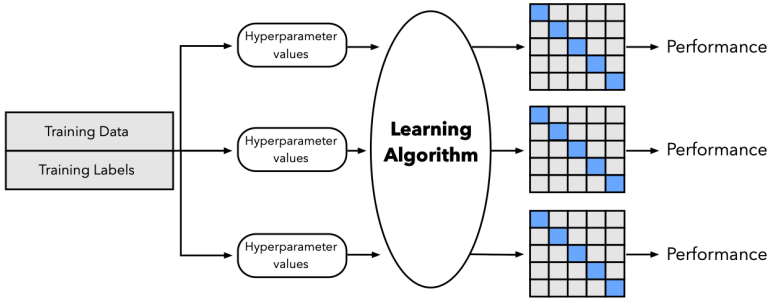
- Data divided into K subsets
- Repeat hold K times, each time with *(K-1)* subsets as training set and 1 subset as validation set.
- Every data point gets to be in test set once
- Model performance is average across *K* validation sets
- Variance of the model performance estimate is reduced as *K* is increased.
- When K=data set size its *leave-one-out cross-validation (LOOCV)*

K-fold Cross-validation for Model Selection

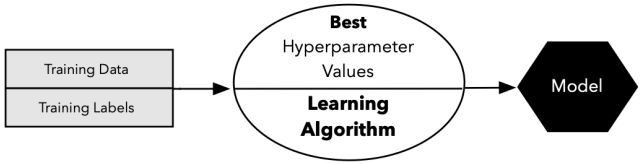
1



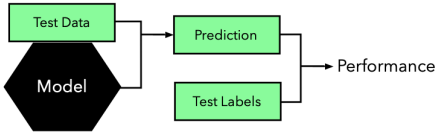
2



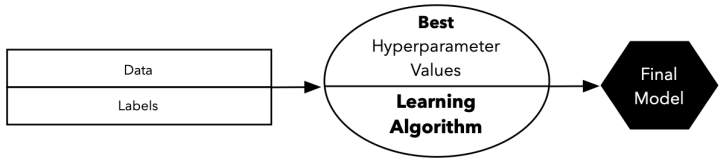
3



4



5



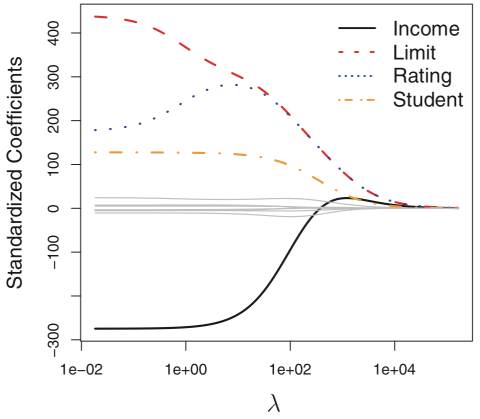
Regularization

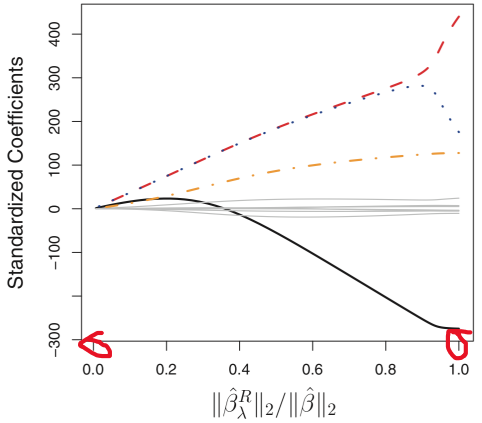
- Techniques used to improve generalization of a model by reducing its complexity
- Techniques to make a model perform well on test data often at expense of its performance on training data
- Avoid overfitting, reduce variance
- Simpler models are preferable: low memory, increase interpretability
- However simpler models may reduce the expressive power of models

Regularization in Regression

L_2 Regularization Loss (Ridge Regression)

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}\beta_j^2$$

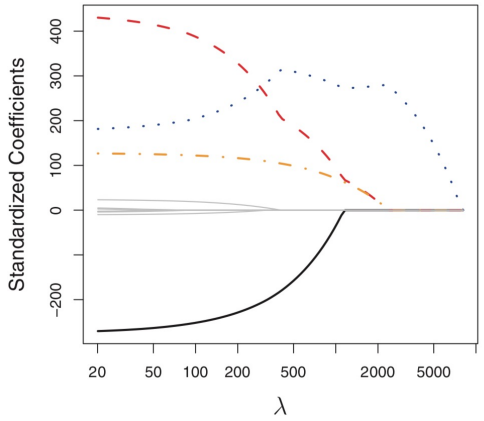


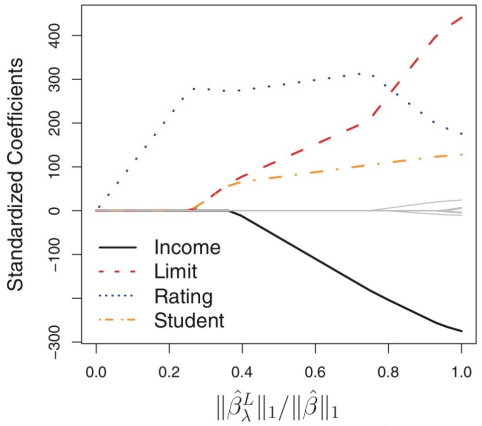


L_1 Regularization Loss (LASSO)

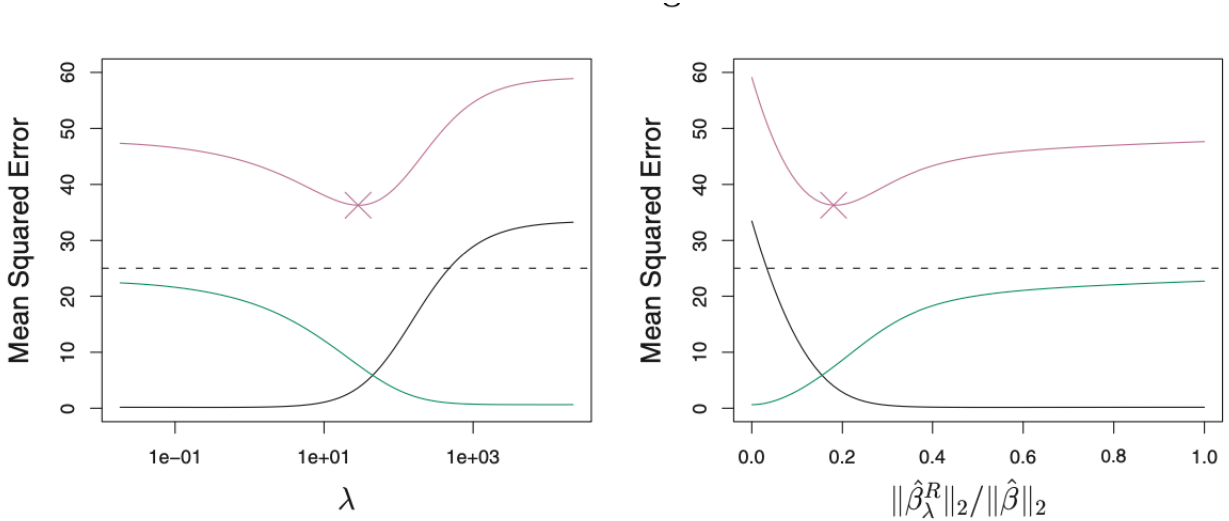
$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}|\beta_j|$$

What value of lambda to choose ?





Bias-variance tradeoff with Lambda



Logistic Regression

- Classification algorithm
- Works for binary classification problems with linearly separable data
- Can be generalized to multiclass settings
- Uses a probabilistic model for binary classification
- Let p be the probability of a positive event (that we are predicting)
- Odds in favor of this positive event: p/(1-p)
- Logit function: log of the odds

$$logit(p) = log\ \frac{p}{(1-p)}$$

$$logit\big(p(y=1|\boldsymbol{x})\big) = w_0x_0 + w_1x_1 + \ldots + w_m\,x_m = \sum_{i=0}^{m} w_ix_i = \boldsymbol{w^T x}$$

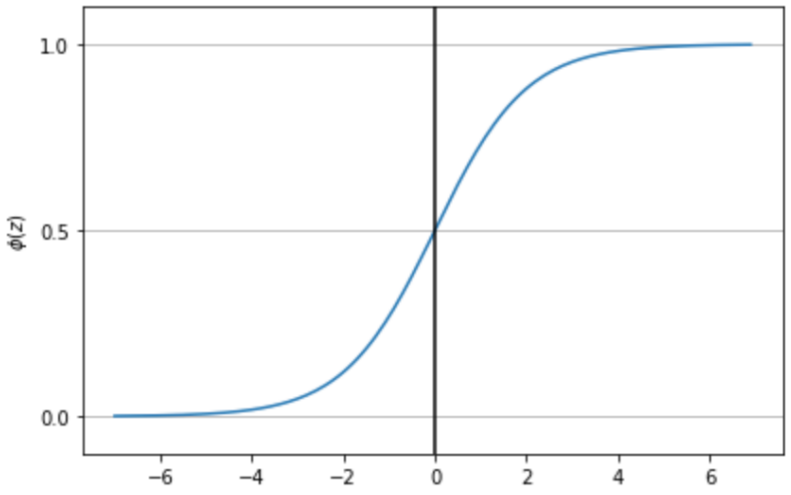
$p(y=1|\boldsymbol{x})$: conditional probability that an input with features $\boldsymbol{x}$ belongs to positive class

- Linear regression model for log of the odds.

Sigmoid function

- We are interested in knowing $p(y = 1|\boldsymbol{x})$
- Let $z = \boldsymbol{w}^T\boldsymbol{x}$ then the inverse of logit function is:

$$\phi(z) = \frac{1}{1+e^{-z}} \qquad \text{(Sigmoid function)}$$



Sigmoid function takes real values as input and transforms /squashes them to values in the range [0,1]

Logistic regression cost function

- Likelihood function

$$\mathrm{L}(\boldsymbol{w}) = P(\boldsymbol{y} \mid \boldsymbol{x}; \boldsymbol{w}) = \prod_{i=1}^{i=n} P\left(y^{(i)} \mid x^{(i)}; \boldsymbol{w}\right) = \prod_{i=1}^{n} (\phi(z^{(i)}))^{y^{(i)}} (1 - \phi(z^{(i)}))^{1-y^{(i)}}$$

- Log Likelihood function

$$l(\boldsymbol{w}) = \log L(\boldsymbol{w}) = \sum_{i=1}^{n} [y^{(i)} \log\left(\phi\left(z^{(i)}\right)\right) + (1 - y^{(i)}) \log(1 - \phi(z^{(i)}))]$$

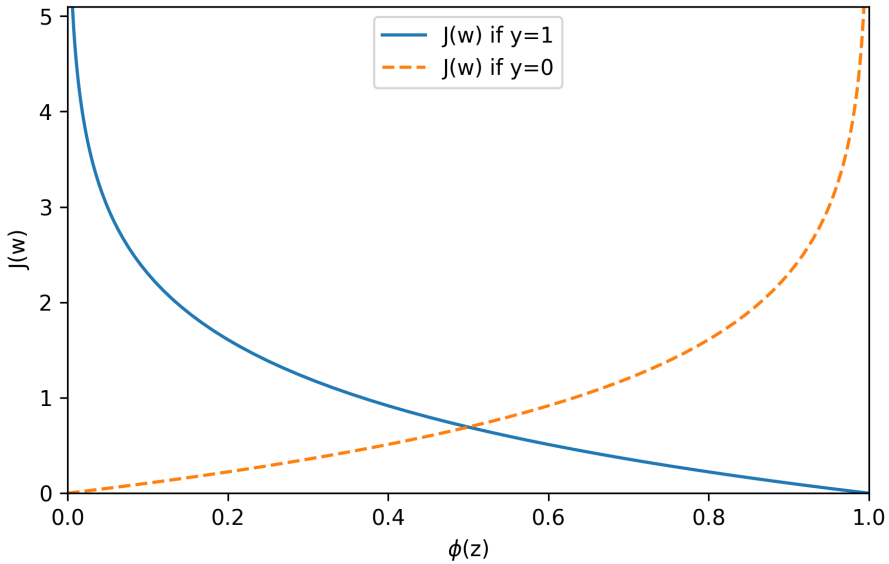
- Loss/Cost function: negative of log likelihood

$$J(\boldsymbol{w}) = -l(\boldsymbol{w}) = \sum_{i=1}^{n} [-y^{(i)} \log(\phi\left(z^{(i)}\right)) - (1 - y^{(i)}) \log(1 - \phi(z^{(i)}))]$$

Logistic regression cost function behavior

- For a single training sample the cost is:

$$J(\boldsymbol{w}) = -y \log\big(\phi(z)\big) - (1-y) \log\big(1 - \phi(z)\big)$$



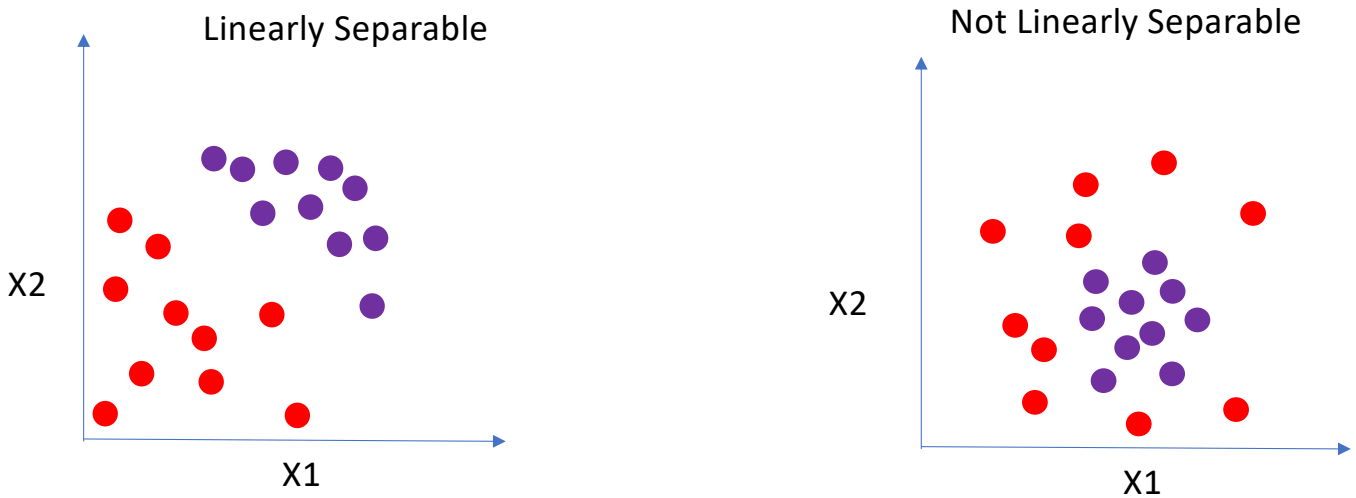
Preventing overfitting in logistic regression

- Cost function for logistic regression regularized by adding $\boldsymbol{L_2}$ regularization term

$$J(\boldsymbol{w}) = -l(\boldsymbol{w}) = \sum_{i=1}^{n}[-y^{(i)}\log(\phi(z^{(i)})) - (1\ -y^{(i)})\log(1-\phi(z^{(i)}))] + \frac{\lambda}{2}\|\boldsymbol{w}\|^2$$

- λ is the regularization parameter; low λ leads to complex models whereas high λ leads to simple models.
- In sci-kit learn LogisticRegression class the parameter C controls the regularization, with C being inverse of λ.
  - High C leads to complex models while low C leads to simple models

Linear Separability



- Measure of data complexity for classification problems
- For binary classification, linear separability implies the existence of a hyperplane completely separating the two classes
- Tests for Linear Separability:
  - Convex Hull: intersection of convex hulls of classes is empty
  - 100% SVM accuracy with linear kernel
  - Review example using IRIS dataset

Reading List

- **Model Selection**
  - Sebastian Raschka. Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning. 2018
- **Practical ML**
  - Pedro Domingos, A few useful things to Know about machine Learning
  - Jeff Dale, *Best Deals in Deep Learning Cloud Providers*, *medium article*

Code Links

- Model Selection: Underfitting, Overfitting, and the Bias-Variance Tradeoff

https://dustinstansbury.github.io/theclevermachine/bias-variance-tradeoff

- Methods for testing linear separability in Python

  http://www.tarekatwan.com/index.php/2017/12/methods-for-testing-linear-separability-in-python/#fnref-102-6