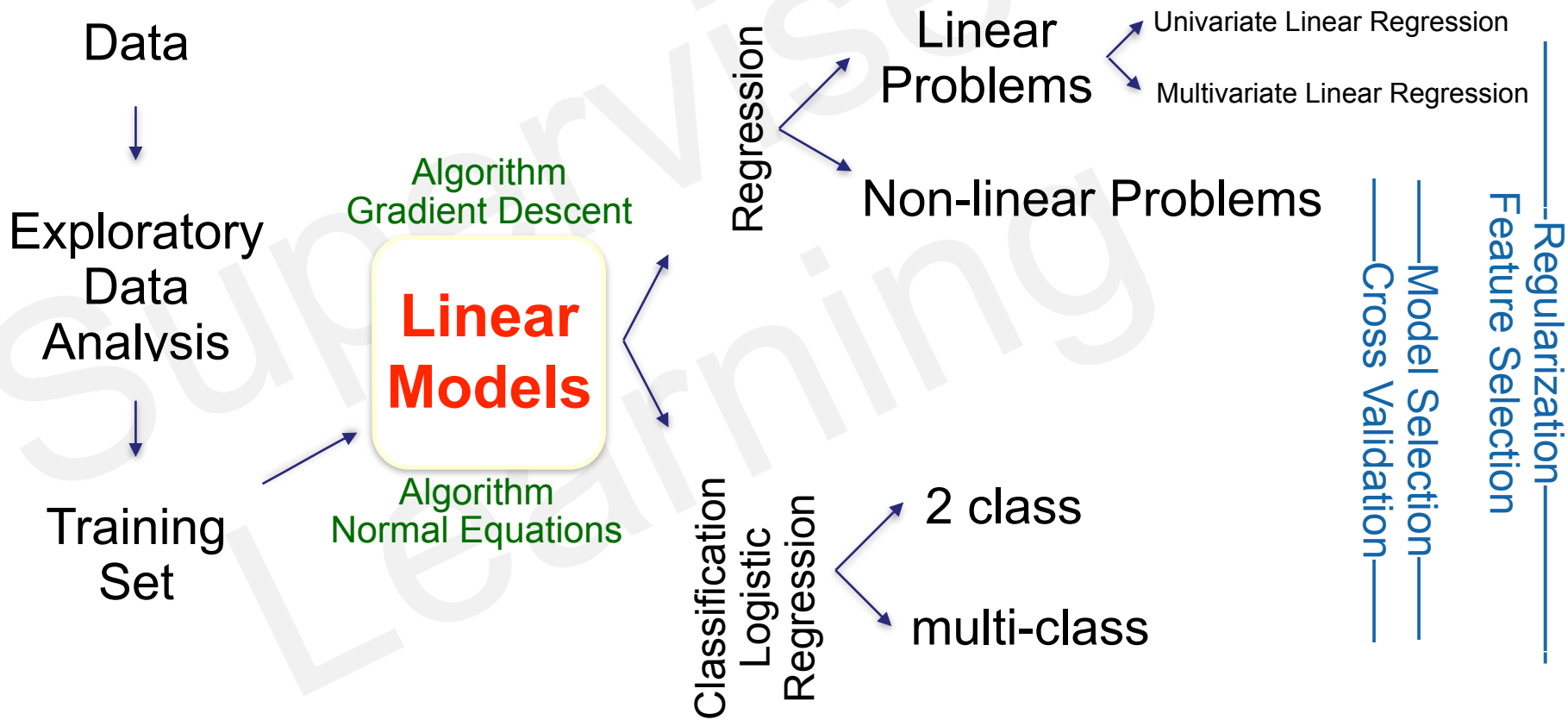
GENERAL ASSEMBLY

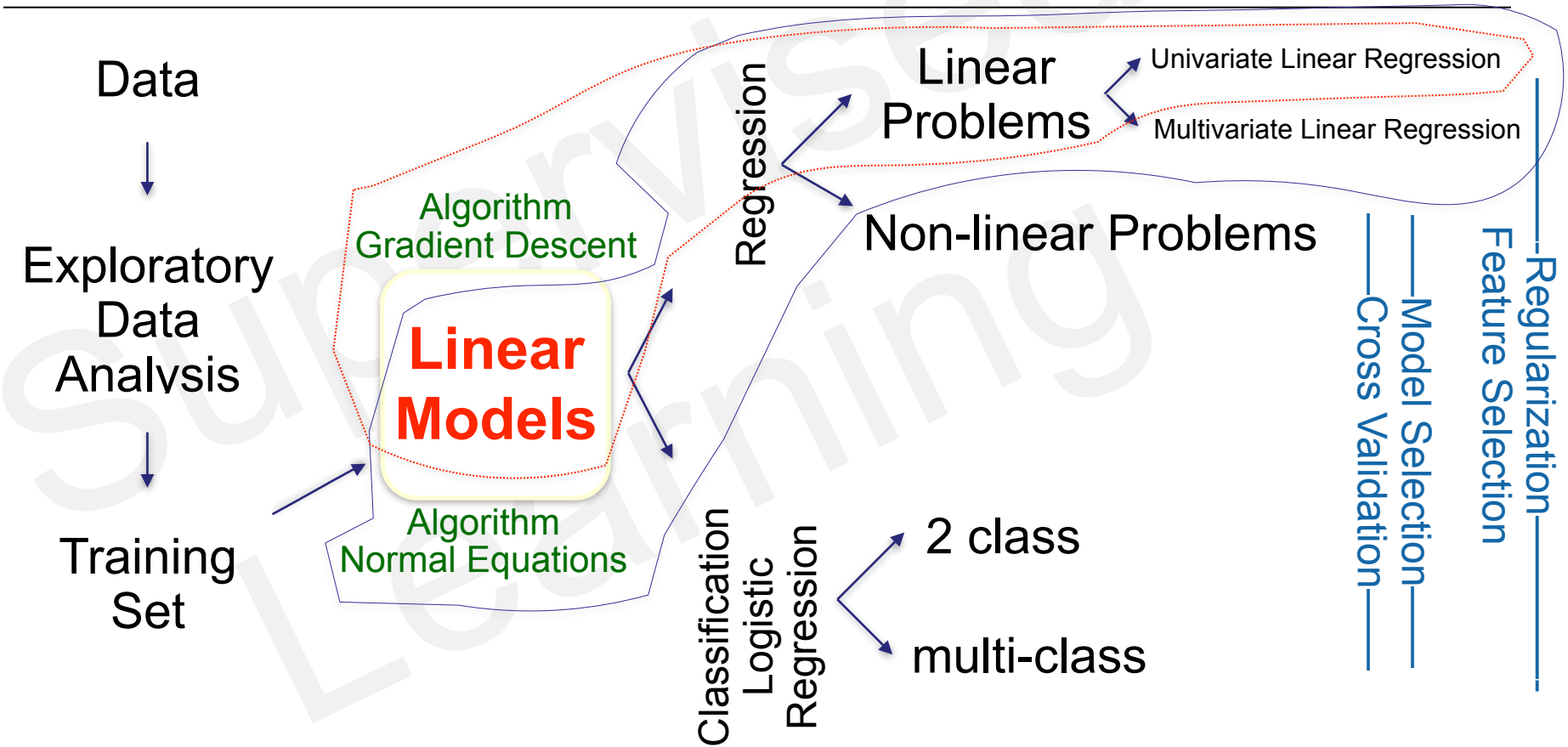
INTRO TO DATA SCIENCE
LECTURE 5: LINEAR MODELS & LINEAR REGRESSION

WHERE ARE WE ON THE DATA SCIENCE ROAD-MAP?



Supervised Learning

WHERE ARE WE ON THE DATA SCIENCE ROAD-MAP?



Supervised Learning

Finding the minimum of the cost function determines values for θ that optimally (in the sum of squared errors sense) model the training set

A second algorithm for finding the minimum of the cost function are via the Normal Equations

- Standard minimization problem
- Take the derivative and set equal to zero
- Using the training set can derive a group of simultaneous equations
- This is not an iterative algorithm, rather it is purely analytical and solves for the final values of theta directly
- The matrix containing the training data is called ‘The Design Matrix’
- Although there is no strict need to feature scale, by force of habit I generally always scale my features

$$\vec{\theta} = (\vec{X}^T \vec{X})^{-1} \vec{X}^T \vec{y}$$

where $\vec{\theta}$ is the vector of model parameters, $\vec{X}$ is an augmented matrix containing the training data (The Design Matrix). It is augmented in the sense that it contains the intercept parameter. y are the outputs of the training data

Step 1: Form the Design Matrix (X) from the data
Step 2: Solve directly for theta using:

$$\vec{\theta} = (\vec{X}^T \vec{X})^{-1} \vec{X}^T \vec{y}$$

Sklearn - Ordinary Least Squares

1. Gradient Descent

i. An iterative algorithm
ii. Need to know the cost function
iii. Start with an initial random guess for the model parameters
iv. At each iteration we alter the parameters a little to reduce the cost function
v. Stop when a “minimum” of the cost function is reached
vi. The training data must be scaled before we used this algorithm
vii. We need to determine a hyper-parameter, called the learning rate
viii. Why do you need to know about this algorithm??

BIG DATA

2. Normal Equations

i. It is possible to solve for the model parameters directly!!
ii. Enter the training data (inputs and outputs) into a set of equations and determine the values of theta
iii. No need to scale the inputs
iv. No hyper-parameter to estimate
v. Involves the calculation of the inverse of a matrix
vi. As the dataset gets large this becomes very computationally expensive
vii. Equations can become unstable if linear regression assumptions are violated

- So far, we have only considered the simplest of linear models
- A single value of x
- Let’s now consider linear models that comprise multiple features

$$y = \theta_0 + \theta_1 x$$ Univariate Linear Regression

Extend the number of features beyond 1.

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \ldots \theta_n x_n$$ Multivariate Linear Regression

- Example: Continuing on the theme of house prices
- Univariate Case: Using a single feature of a house to predict price

  house price = θ_0 + square footage * θ_1

- Multivariate Case: Use multiple features of a house to predict price

  house price = θ_0 + (sqft * θ_1) + (# baths * θ_2) + (lot size * θ_3) +…

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \ldots \theta_n x_n$$

- Makes visualization more difficult/impossible
- For a single feature (univariate linear regression) we fit a line to points in 1-dimension
- For two features we fit a plane to points in 2-dimensions
- For multiple features we fit a hyper-plane to points in N-dimensions
- Begs the question - what features to use as inputs into the linear model? This problem is known as Feature Selection

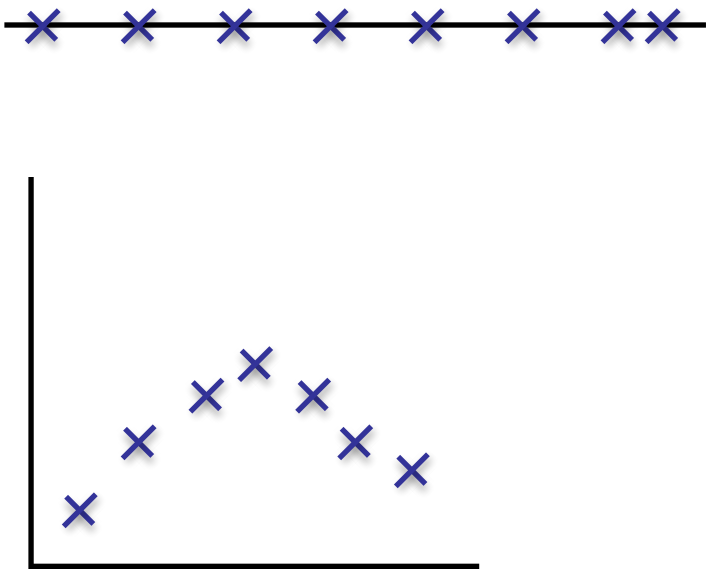
- You are aiming to find the 'best' set of features that will maximize your predictive capability in a robust manner.

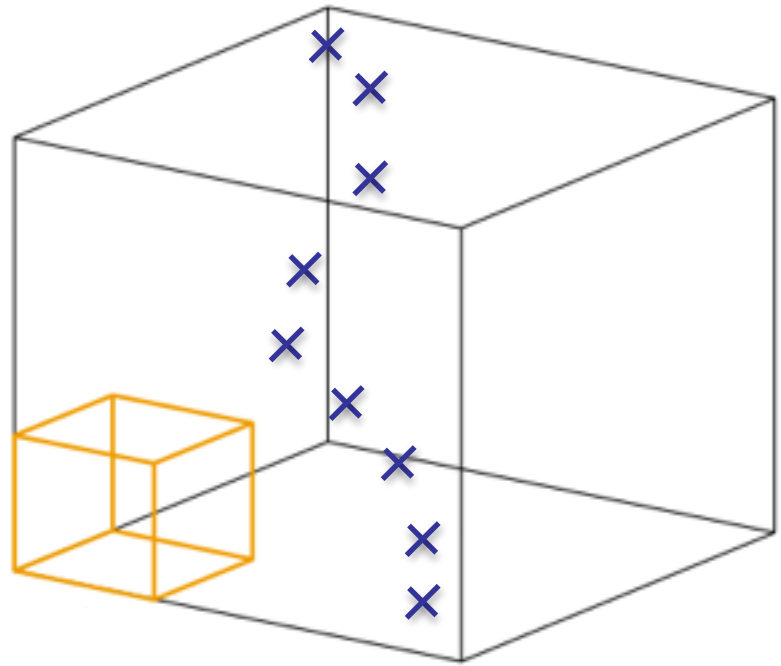
- In addition, you are trying to do this with the fewest number of features you can!

Considerations:

- Domain knowledge - human intuition about what is important
- Interpolation vs extrapolation - predictive models should only interpolate
- Curse of dimensionality - successful interpolation requires a certain density of points in the space

For every additional dimension you need exponentially more data





Considerations:

- Correlation - it is useful to look at correlation between input and output
- Regularization - at the end of the day the model can be trained to determine the ‘useful’ inputs
- Dimensionality reduction - a useful transformation of your data to assist with all of the above
  - De-correlate features
  - Reduce the number of features
  - Linear transformation

1. Linear relationship - the relationship between features and output is linear
2. Multivariate normality - the features are identically distributed
3. No or little multicollinearity - the features are independent
4. No auto-correlation - the residuals are independent of each other
5. Homoscedasticity - the variance in the residuals is ‘constant'

If the features are not independent and exhibit multi-collinearity, the normal equations can become ill-posed

Adding a regularizer solves this problem