

CS102: Big Data

Tools and Techniques, Discoveries and Pitfalls

Spring 2017

Ethan Chan, Lisa Wang

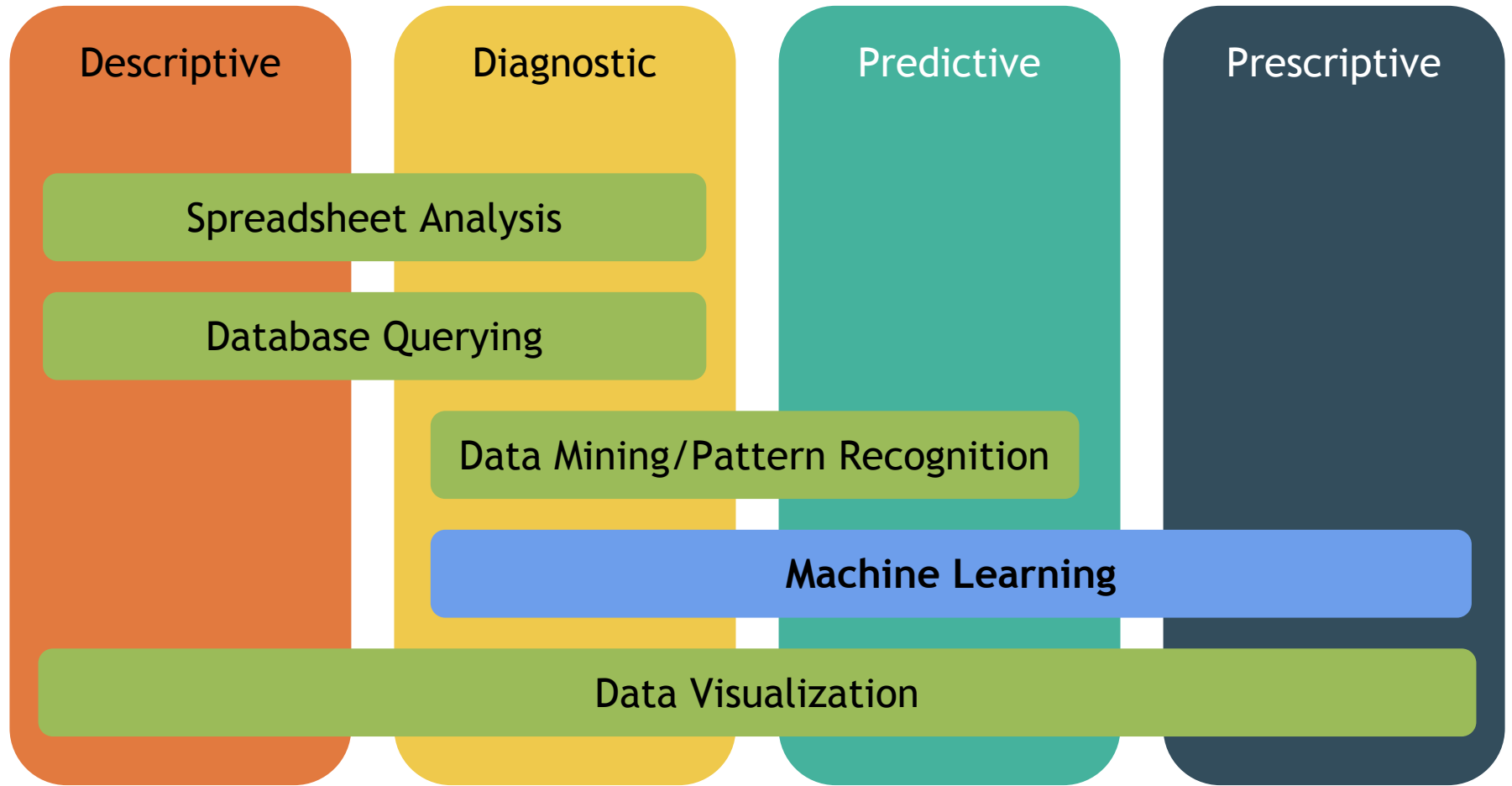
Lecture 13:

Metrics + Building ML Models

Announcements

- Thanks for feedback!
 - Slower pacing through coding parts
 - Will code side by side
 - Exams will still be on paper, 90 minutes
 - Conceptual
 - Practice midterm
 - We will have practice questions for the final
 - Recordings
 - We don't have infrastructure to record, but we can have useful links/resources to learn concepts outside of class
 - Speaking louder
 - Office Hours
 - Email us for office hour appointment, or use piazza

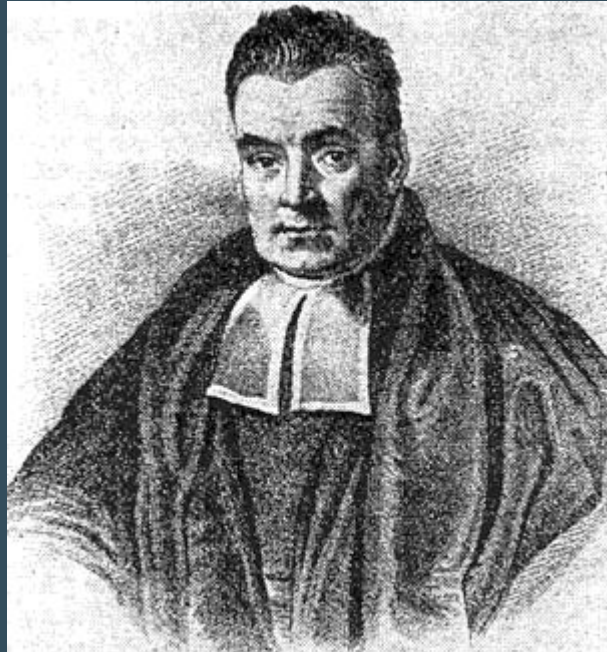
Tools & Techniques



Learning Goals

- Summary of Supervised Learning
- Guide to building Machine Learning models
- Evaluation Metrics
- Unsupervised Learning (Clustering)

Naive Bayes (recap)



Naive Bayes Summary (recap)

This is all you need to know for purposes of this class

Given training data that follows this format..

Feature X1	Feature X2	Feature X3	...	Feature N	Y (Label) [A or B]
..
..

And you are given new data without labels that you want to classify

Feature X1	Feature X2	Feature X3	...	Feature N	Y (Label) [A or B]
..	???
..	???

Determine if

$P(X1 | Y = A) * P(X2 | Y = A) * P(X3 | Y = A) * .. * P(X_N | Y = A) * P(Y = A)$

OR

$P(X1 | Y = B) * P(X2 | Y = B) * P(X3 | Y = B) * .. * P(X_N | Y = B) * P(Y = B)$

Is greater

Now, your turn!

Person	X1 Age [young / old]	X2 Tumor Size [none / S / L]	Y (cancer or not)
A	old	none	Cancer
B	old	small	Cancer
C	young	none	No Cancer
D	young	large	Cancer
E	old	none	No Cancer
F	old	none	No Cancer
G	young	small	No Cancer

Given a new person who is X1 = young and X2 = small, what will the model predict?

Remember! Determine if

$P(X1 | Y = A) * P(X2 | Y = A) * P(X3 | Y = A) * .. * P(X999 | Y = A) * P(Y = A)$

OR

$P(X1 | Y = B) * P(X2 | Y = B) * P(X3 | Y = B) * .. * P(X999 | Y = B) * P(Y = B)$

Is greater

Now, your turn!

Person	X1 Age [young / old]	X2 Tumor Size [none / S / L]	Y (cancer or not)
A	old	none	Cancer
B	old	small	Cancer
C	young	none	No Cancer
D	young	large	Cancer
E	old	none	No Cancer
F	old	none	No Cancer
G	young	small	No Cancer

Given a new person who is X1 = young and X2 = small, what will the model predict?

$$P(X1 = \text{young} \mid Y = \text{Cancer}) * P(X2 = \text{small} \mid Y = \text{Cancer}) * P(Y = \text{Cancer})$$

$$= \frac{1}{3} * \frac{1}{3} * \frac{3}{7} = 0.0476$$

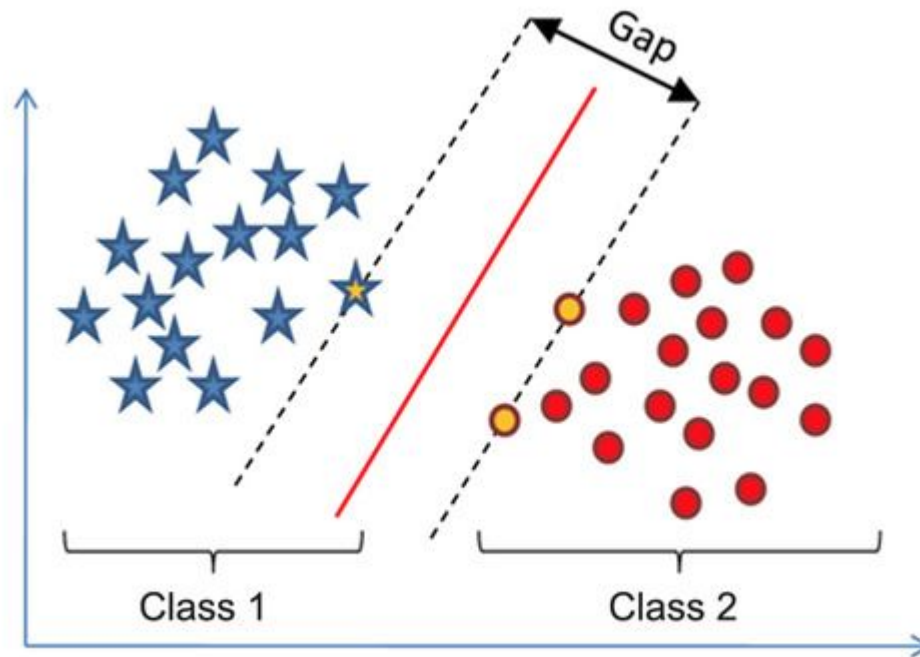
$$P(X1 = \text{young} \mid Y = \text{No Cancer}) * P(X2 = \text{small} \mid Y = \text{No Cancer}) * P(Y = \text{No Cancer})$$

$$= \frac{2}{4} * \frac{1}{4} * \frac{4}{7} = 0.0714$$

0.0476 < 0.0714, NO CANCER!! :)

Support Vector Machines

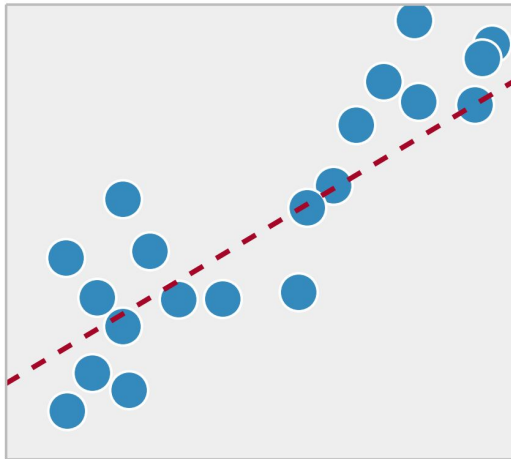
Finds a line that best separates 2 classes of points.



Supervised Learning Summary

Regression vs. Classification

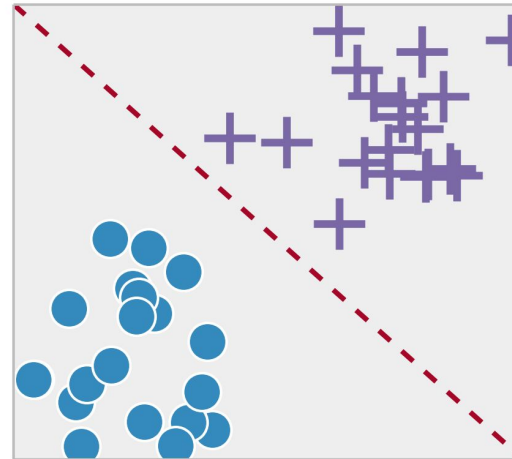
Regression



Output values are real numbers (continuous)

Regression fits a curve to the data, so you can use the curve to predict the real-valued output

Classification



Output values in two or more classes, e.g. cats and dogs (discrete)

Classification tries to predict the class based on features by learning *decision boundaries (the red line)*

Supervised Learning Recap:

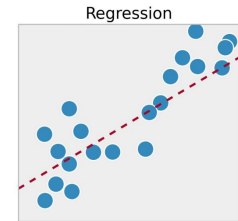
Regression

- Linear Regression
 - Draw a line of best fit through a set of points
- Polynomial Regression
 - Draw a curve of best fit through a set of points

Classification

- Logistic Regression
 - Draw a line (logit curve) to separate ≥ 2 set of points.
- Decision Trees
 - Draw boundaries on axes to separate data points
- Random Forests
 - Group multiple decision trees and take most common vote
- K-Nearest Neighbors
 - Boundary determined by nearest points
- Naive Bayes!
 - Boundary can be determined from probability functions
- Support Vector Machines
 - Draw a line/plane to separate 2 sets of points

Supervised Learning Recap:



Regression

- Linear Regression
 - Draw a line of best fit through a set of points
- Polynomial Regression
 - Draw a curve of best fit through a set of points

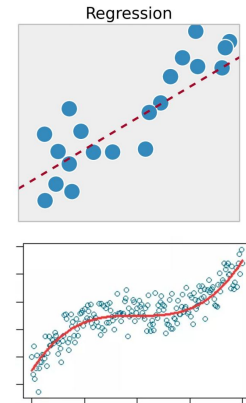
Classification

- Logistic Regression
 - Draw a line (logit curve) to separate ≥ 2 set of points.
- Decision Trees
 - Draw boundaries on axes to separate data points
- Random Forests
 - Group multiple decision trees and take most common vote
- K-Nearest Neighbors
 - Boundary determined by nearest points
- Naive Bayes!
 - Boundary can be determined from probability functions
- Support Vector Machines
 - Draw a line/plane to separate 2 sets of points

Supervised Learning Recap:

Regression

- Linear Regression
 - Draw a line of best fit through a set of points
- Polynomial Regression
 - Draw a curve of best fit through a set of points



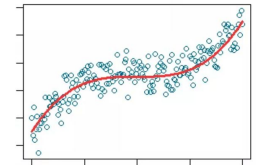
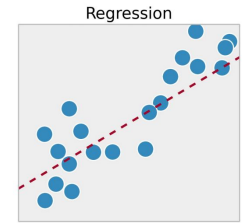
Classification

- Logistic Regression
 - Draw a line (logit curve) to separate ≥ 2 set of points.
- Decision Trees
 - Draw boundaries on axes to separate data points
- Random Forests
 - Group multiple decision trees and take most common vote
- K-Nearest Neighbors
 - Boundary determined by nearest points
- Naive Bayes!
 - Boundary can be determined from probability functions
- Support Vector Machines
 - Draw a line/plane to separate 2 sets of points

Supervised Learning Recap:

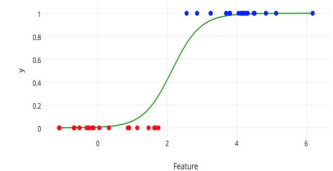
Regression

- Linear Regression
 - Draw a line of best fit through a set of points
- Polynomial Regression
 - Draw a curve of best fit through a set of points



Classification

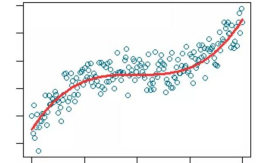
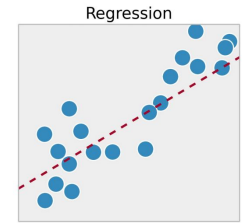
- Logistic Regression
 - Draw a line (logit curve) to separate ≥ 2 set of points.
- Decision Trees
 - Draw boundaries on axes to separate data points
- Random Forests
 - Group multiple decision trees and take most common vote
- K-Nearest Neighbors
 - Boundary determined by nearest points
- Naive Bayes!
 - Boundary can be determined from probability functions
- Support Vector Machines
 - Draw a line/plane to separate 2 sets of points



Supervised Learning Recap:

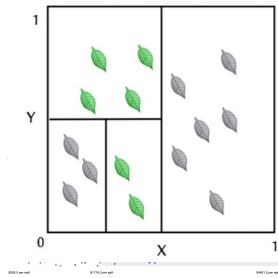
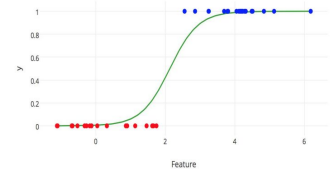
Regression

- Linear Regression
 - Draw a line of best fit through a set of points
- Polynomial Regression
 - Draw a curve of best fit through a set of points



Classification

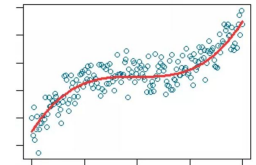
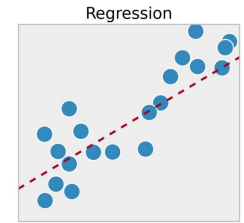
- Logistic Regression
 - Draw a line (logit curve) to separate ≥ 2 set of points.
- Decision Trees
 - Draw boundaries on axes to separate data points
- Random Forests
 - Group multiple decision trees and take most common vote
- K-Nearest Neighbors
 - Boundary determined by nearest points
- Naive Bayes!
 - Boundary can be determined from probability functions
- Support Vector Machines
 - Draw a line/plane to separate 2 sets of points



Supervised Learning Recap:

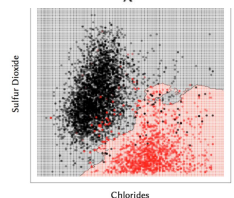
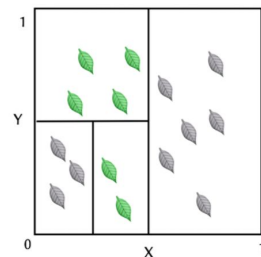
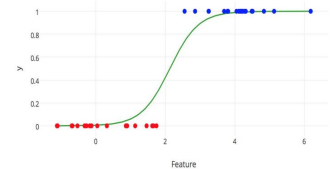
Regression

- Linear Regression
 - Draw a line of best fit through a set of points
- Polynomial Regression
 - Draw a curve of best fit through a set of points



Classification

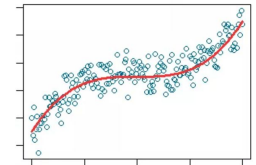
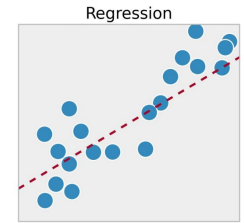
- Logistic Regression
 - Draw a line (logit curve) to separate ≥ 2 set of points.
- Decision Trees
 - Draw boundaries on axes to separate data points
- Random Forests
 - Group multiple decision trees and take most common vote
- **K-Nearest Neighbors**
 - **Boundary determined by nearest points**
- Naive Bayes!
 - Boundary can be determined from probability functions
- Support Vector Machines
 - Draw a line/plane to separate 2 sets of points



Supervised Learning Recap:

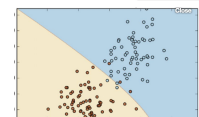
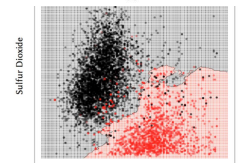
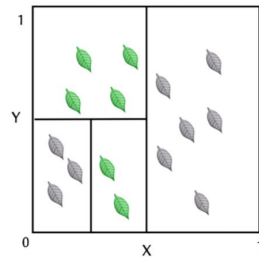
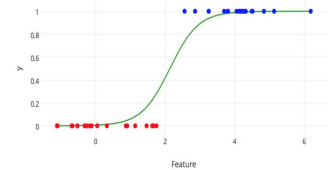
Regression

- Linear Regression
 - Draw a line of best fit through a set of points
- Polynomial Regression
 - Draw a curve of best fit through a set of points



Classification

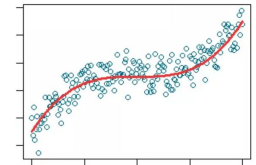
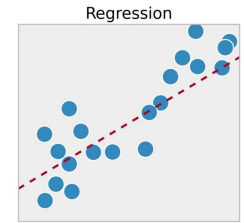
- Logistic Regression
 - Draw a line (logit curve) to separate ≥ 2 set of points.
- Decision Trees
 - Draw boundaries on axes to separate data points
- Random Forests
 - Group multiple decision trees and take most common vote
- K-Nearest Neighbors
 - Boundary determined by nearest points
- Naive Bayes!
 - Boundary can be determined from probability functions
- Support Vector Machines
 - Draw a line/plane to separate 2 sets of points



Supervised Learning Recap:

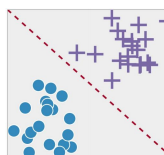
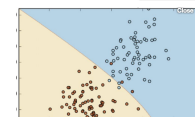
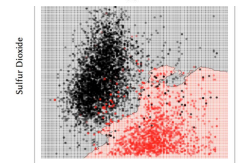
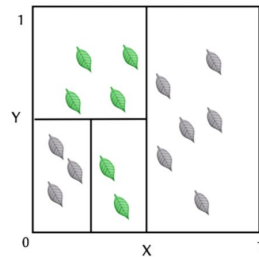
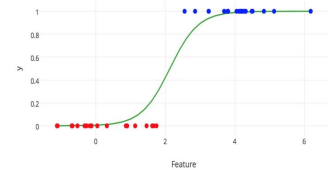
Regression

- Linear Regression
 - Draw a line of best fit through a set of points
- Polynomial Regression
 - Draw a curve of best fit through a set of points



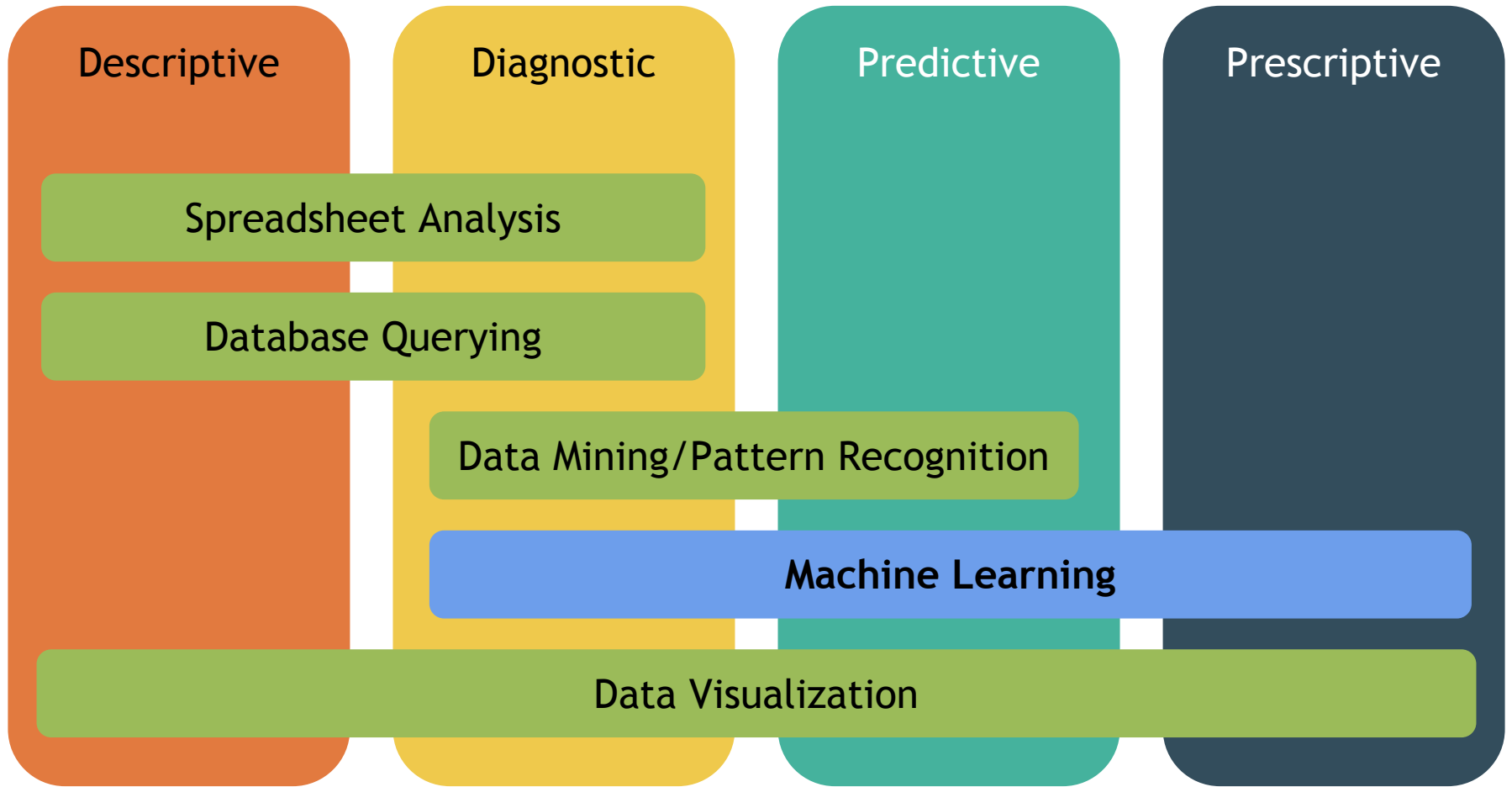
Classification

- Logistic Regression
 - Draw a line (logit curve) to separate ≥ 2 set of points.
- Decision Trees
 - Draw boundaries on axes to separate data points
- Random Forests
 - Group multiple decision trees and take most common vote
- K-Nearest Neighbors
 - Boundary determined by nearest points
- Naive Bayes!
 - Boundary can be determined from probability functions
- Support Vector Machines
 - Draw a line (in 2D) / plane (in 3D) to separate 2 sets of points



Yay! We finished
supervised learning!!

Tools & Techniques



Where are we now?

1. Descriptive

- a. What happened?
 - i. Piecharts / histograms etc

2. Diagnostic analysis

- a. Why did something happen?
 - i. Correlation coefficients of Y and X to see patterns
 - ii. Frequent itemset mining

3. Predictive analysis

- a. What will happen?
 - i. Regression and Classification

4. Prescriptive

- a. What decision should I make?
 - i. Classification (Labels are decisions, features are data)
 - a. Ex. What treatment should I give a patient based on these symptoms?

Building Machine Learning Models

Useful for Final Projects, your research projects, and any data analysis you do in the future

Data Analysis Process

ask the right questions

“Which customers are most likely to leave my website?”

collect the data

Talks to various teams within the firm to get the data

clean and process data

Most of time the data is always incomplete / messy

make sense of it


Build a machine learning model to predict who will leave

tells a story

Tells the sales/marketing team who to focus on



~80% of
your time

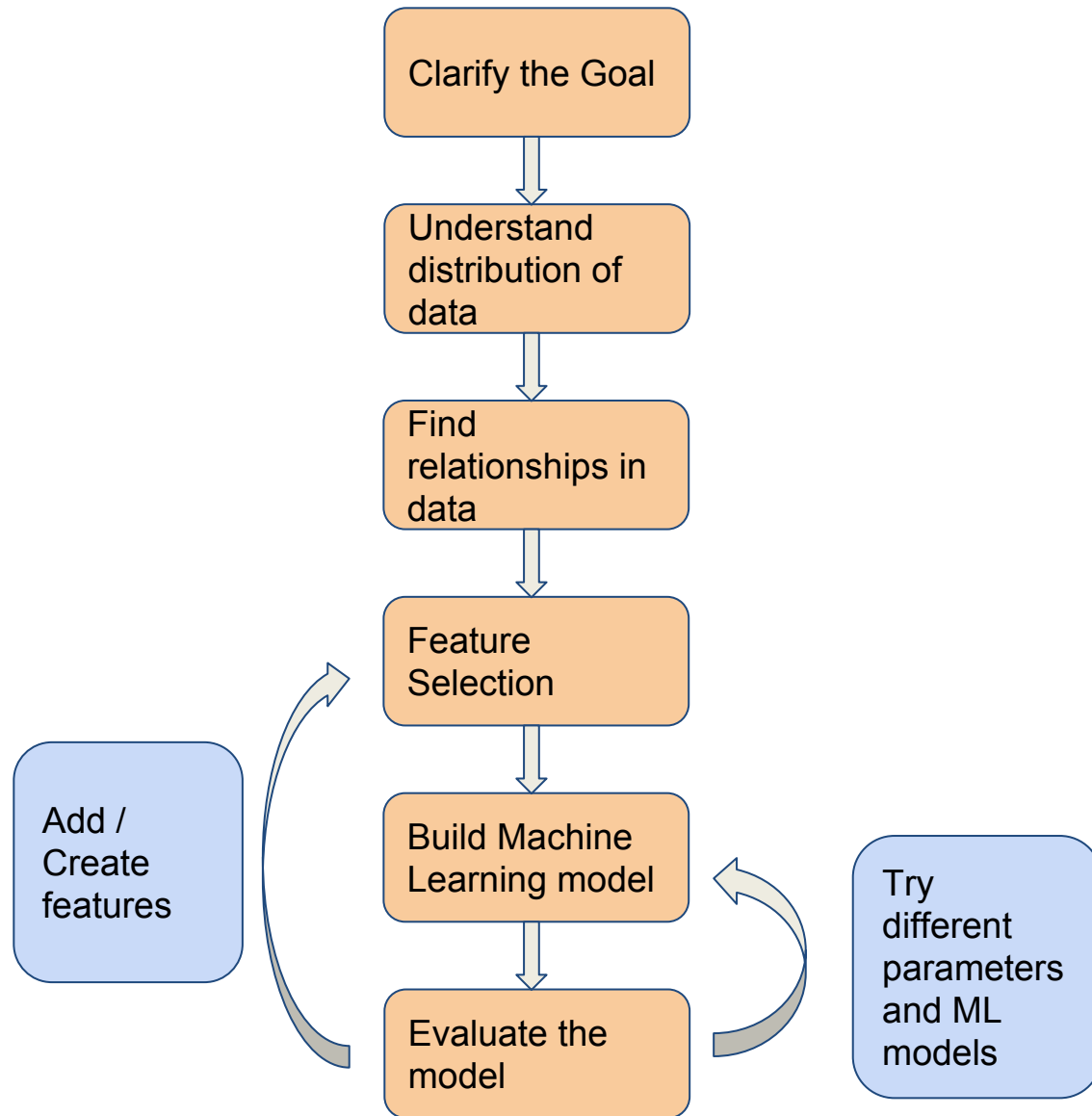


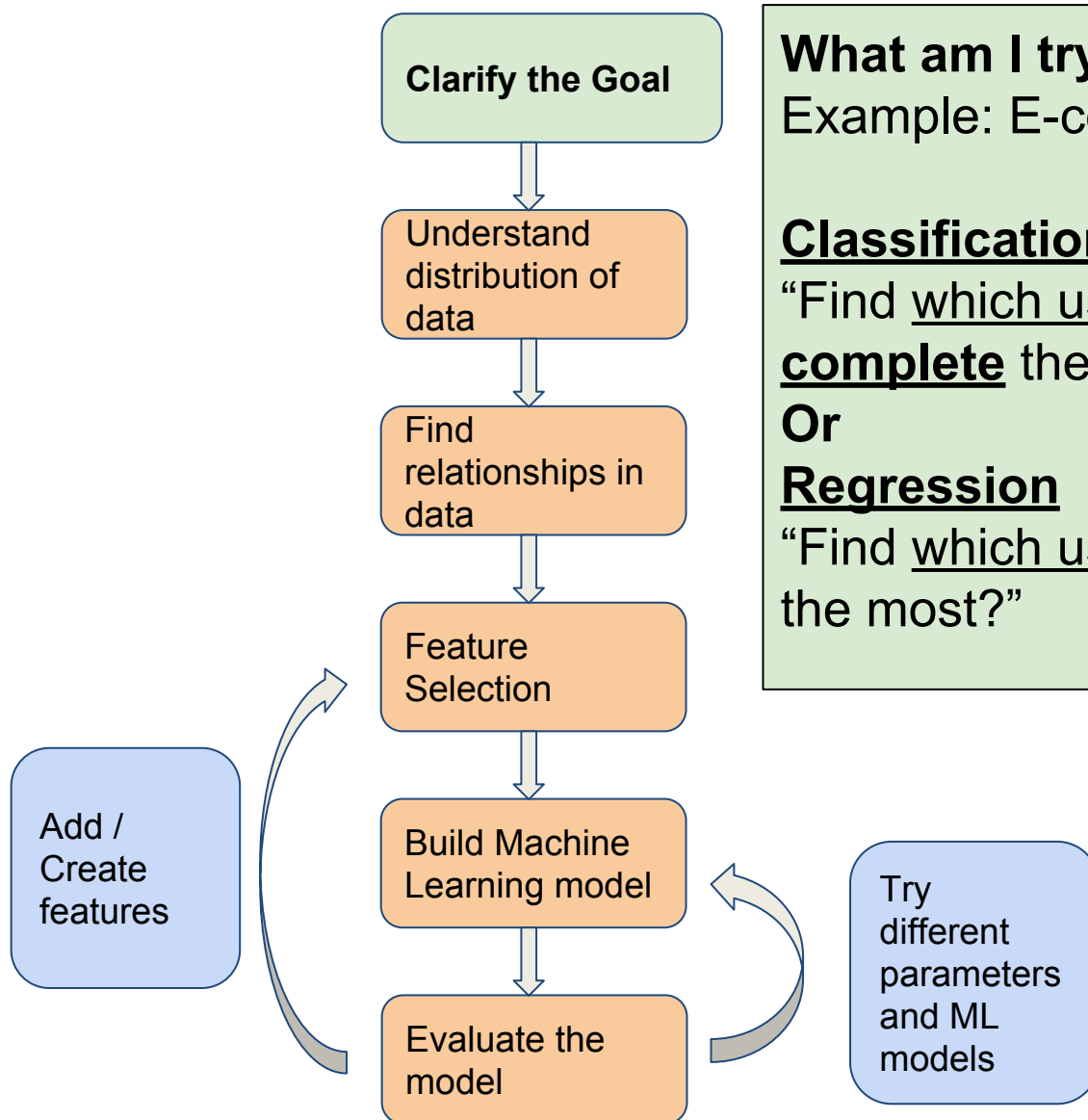
Learn how to
build a good
model

Example: data for an ecommerce company

Device Used	Age	Gender	Number of Clicks	Completed Registration (1 = yes, 0 = no)	Money Spent
..	1	\$999
..	0	\$100
..	1	\$0

Building an ML Model





What am I trying to investigate?

Example: E-commerce company

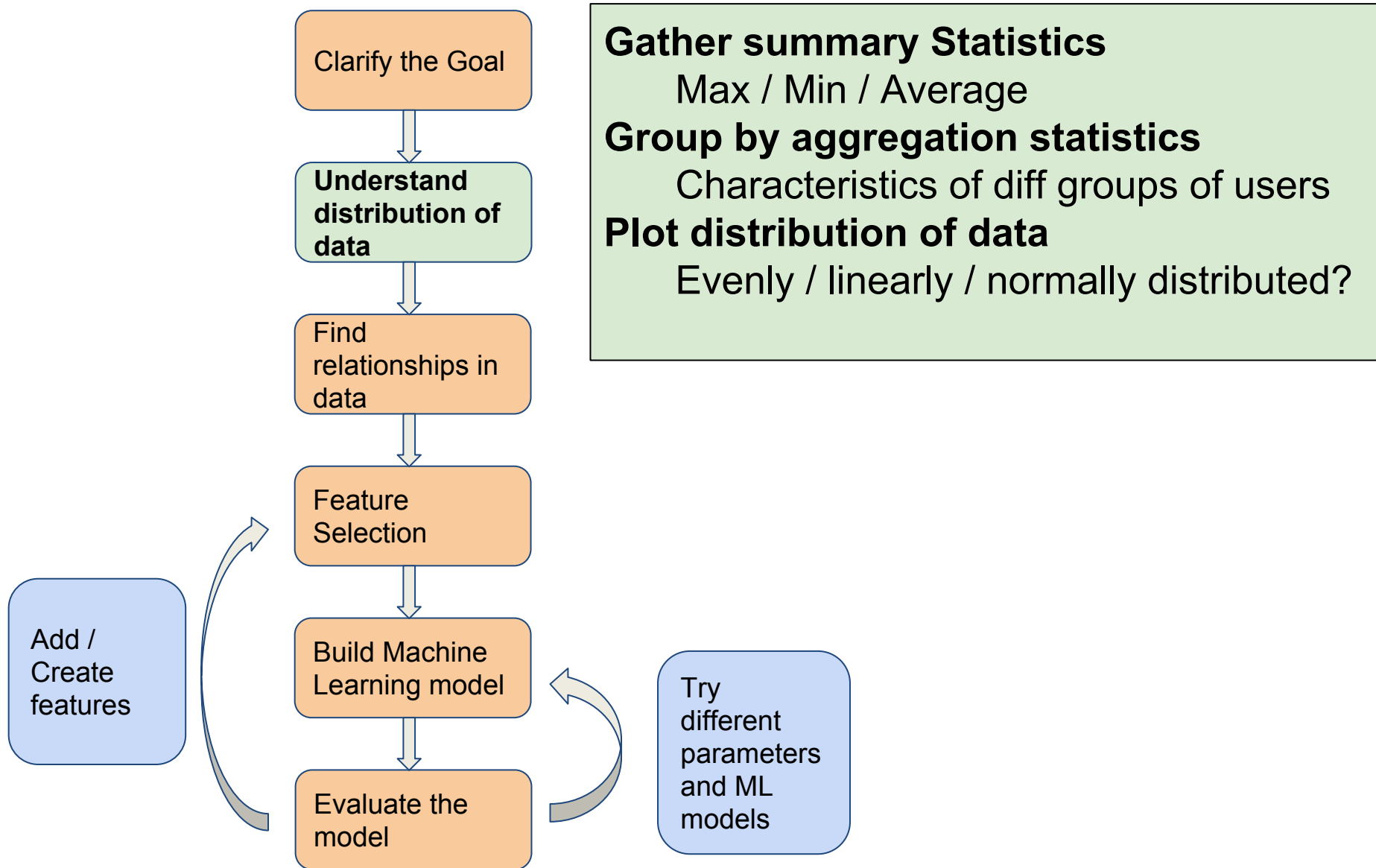
Classification

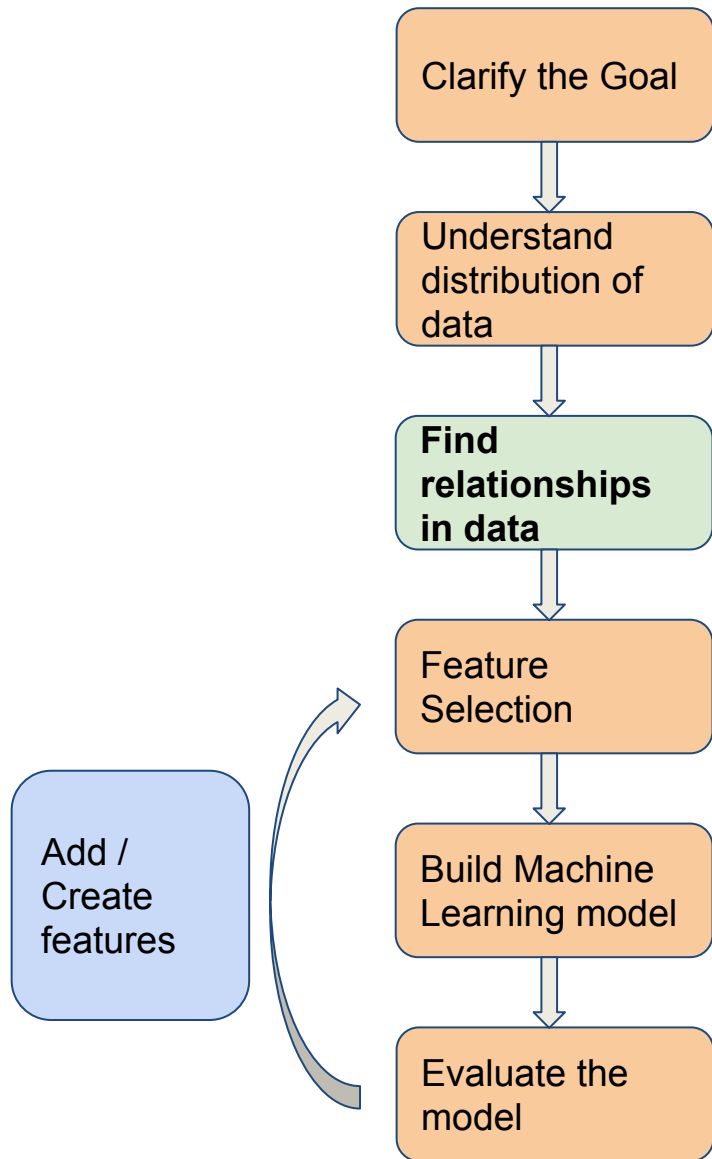
“Find which users are most likely to **complete** the registration process?”

Or

Regression

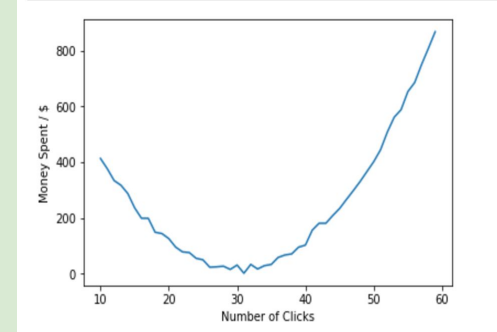
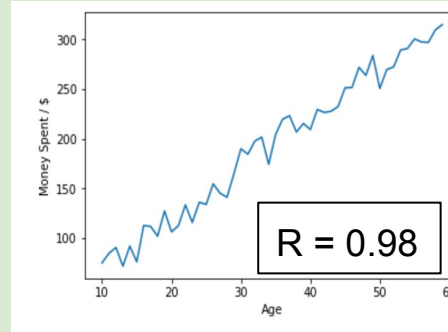
“Find which users are most likely to **spend** the most?”





For Regression type questions

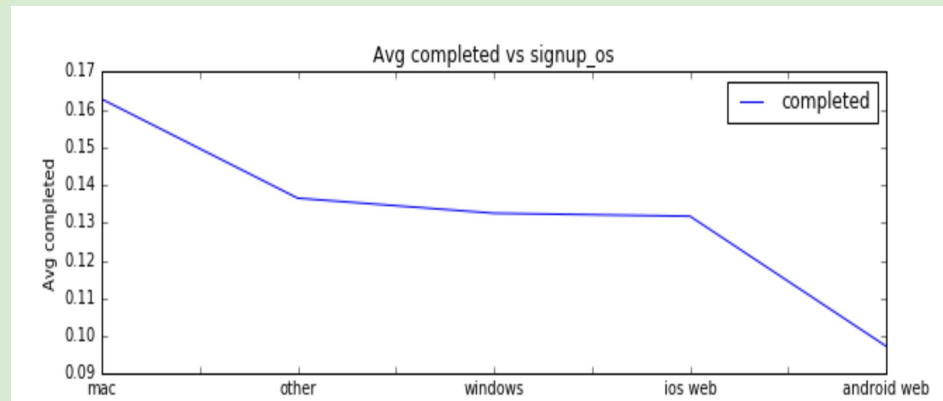
1. Plot Y against each variable X

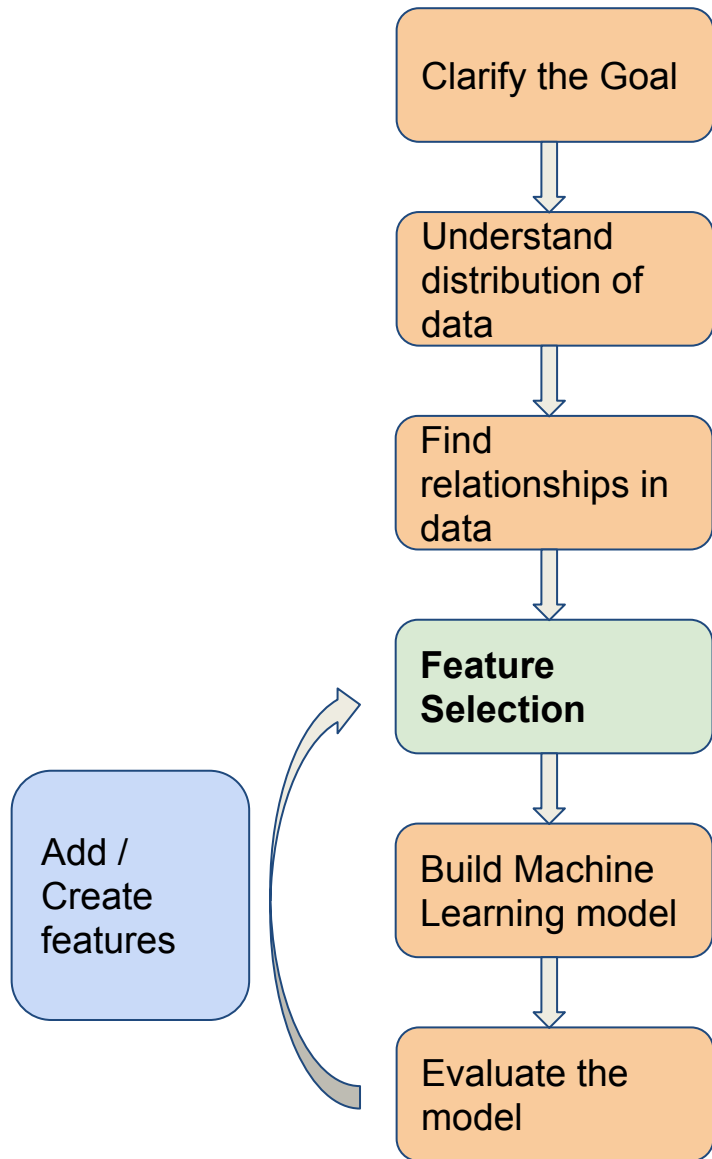


For Classification type questions

1. Plot the average Y against each variable X

a. Y: 1 = completed, 0 = not completed.





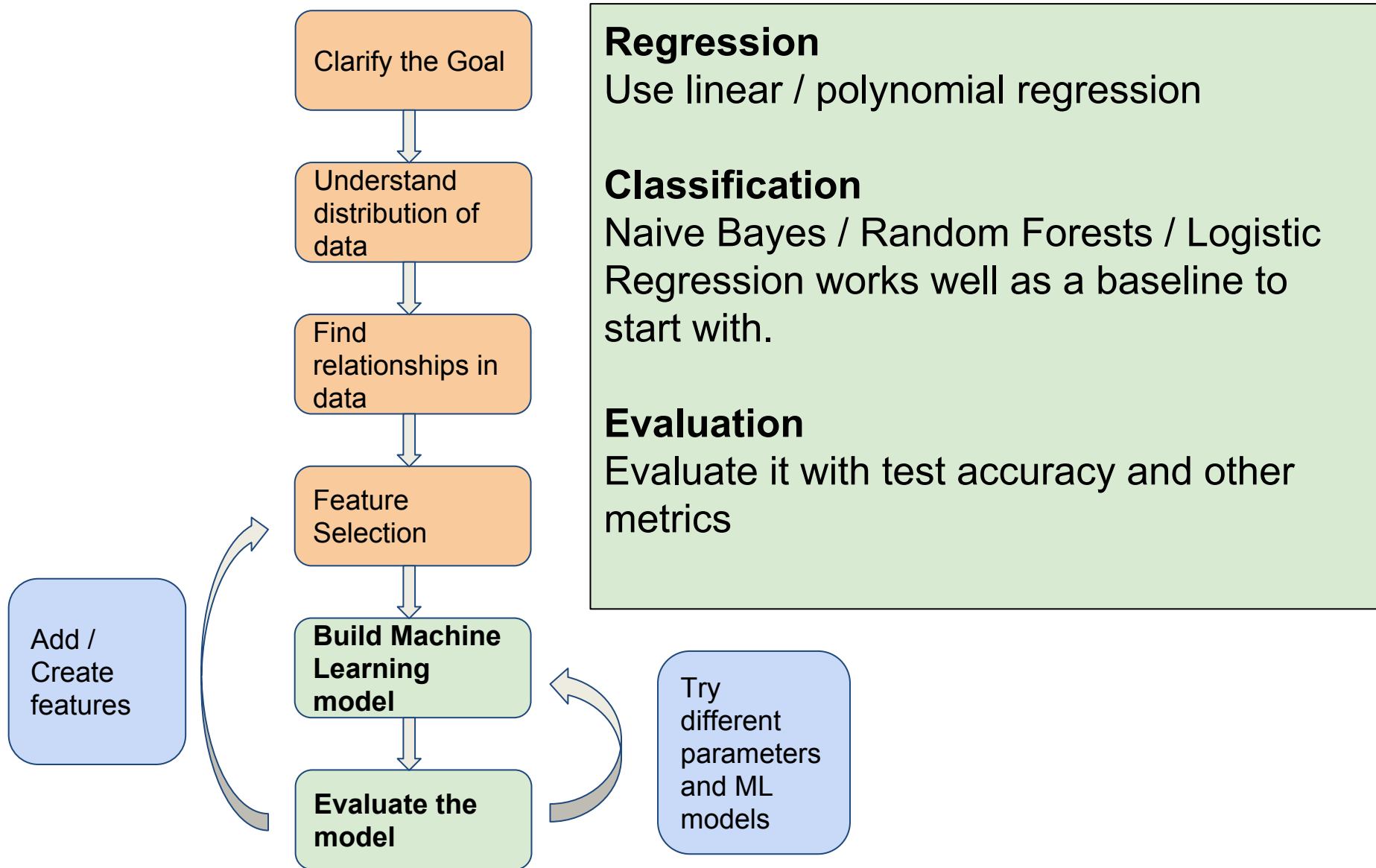
Start with all the features, and remove later
Some features have very clear relationships to the predicted variable, some might just add noise, I will suggest trying putting them all in first and remove noisy ones later.

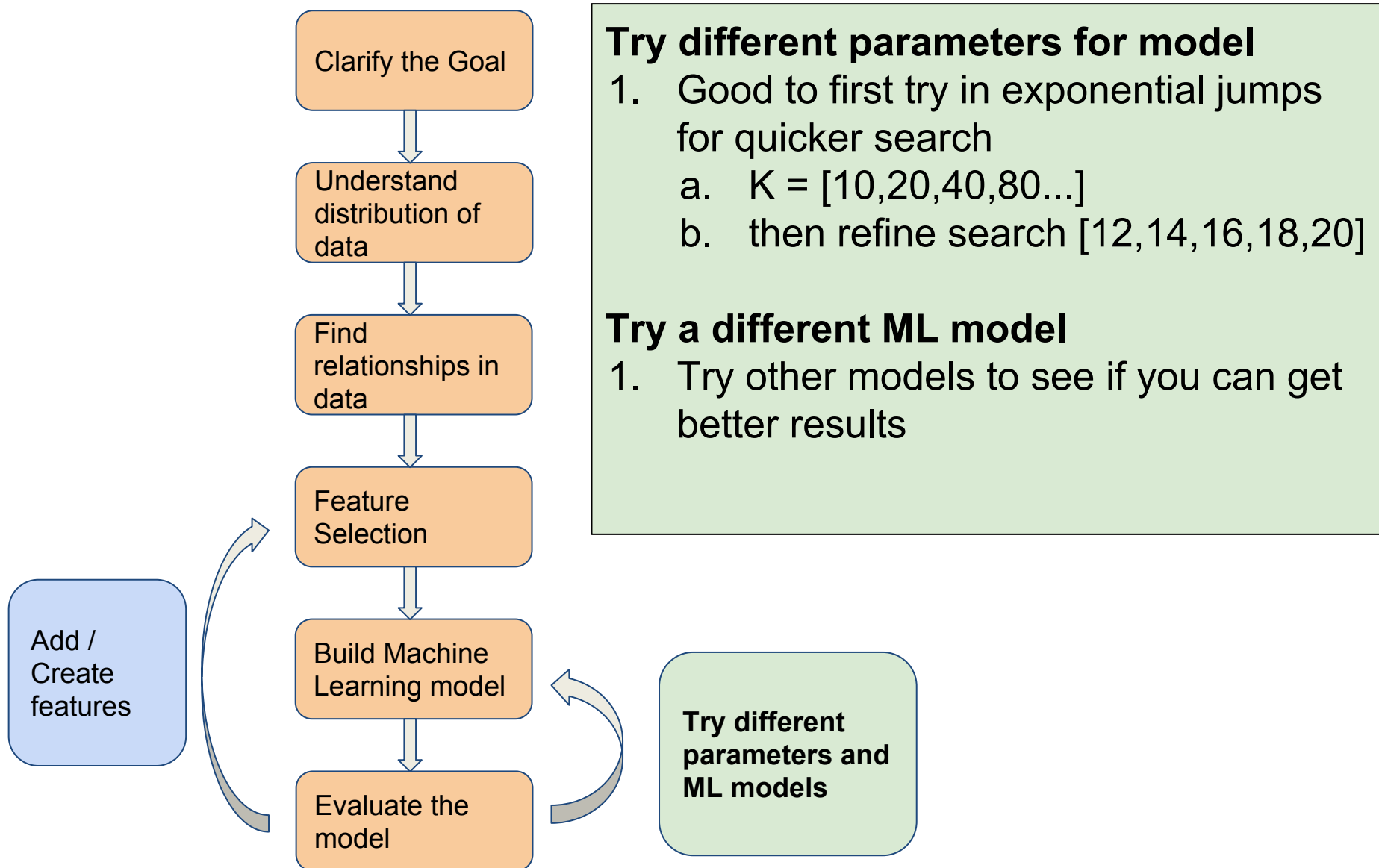
Careful of feature bias!

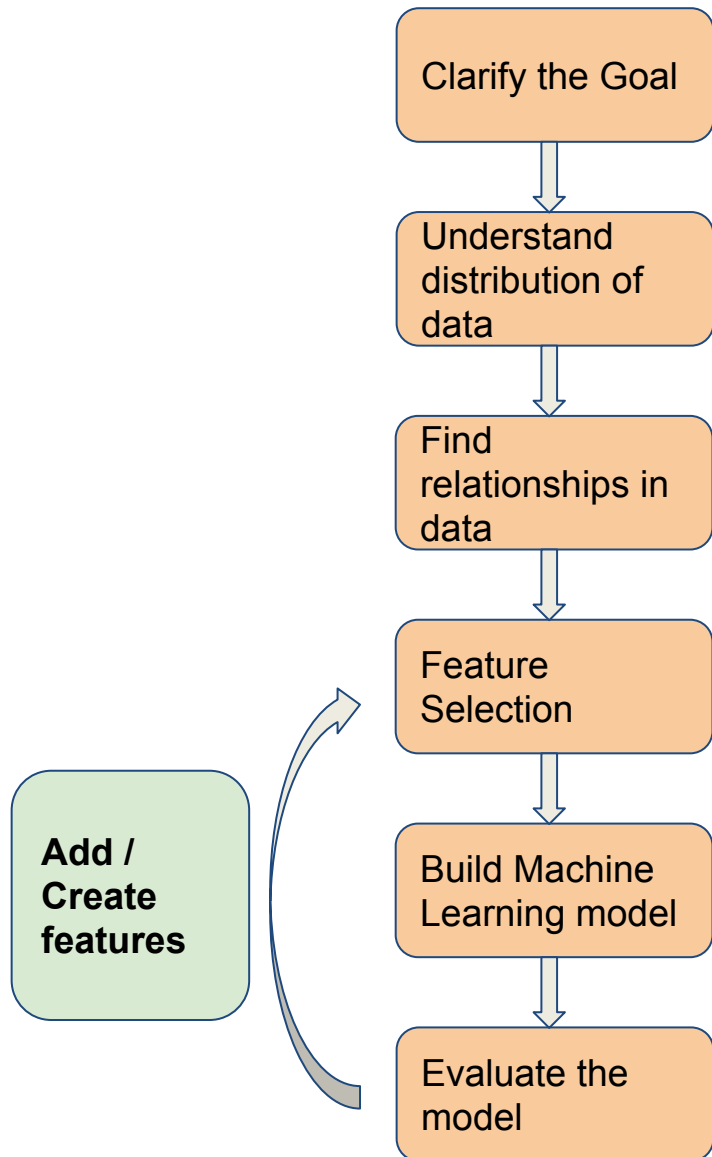
Some researchers initially build ML models to predict cancer where the most predictive features used was the Hospital Name.

It performed very well on the training/test data but very poorly when launched on the real world. Why?

There was one hospital that all the other hospitals redirected their patients to because that hospital specialized in cancer treatment. Hence, when model used on new diagnoses from other hospitals, it failed miserably.







Drop Feature

1. Some features might be adding more noise to the model than helping
 - a. Revisit relationships you've determined in the data to decide which ones to drop
2. Some features might be redundant, reducing model's effectiveness
 - a. Example: distance in miles and distance in km. Drop one of them.

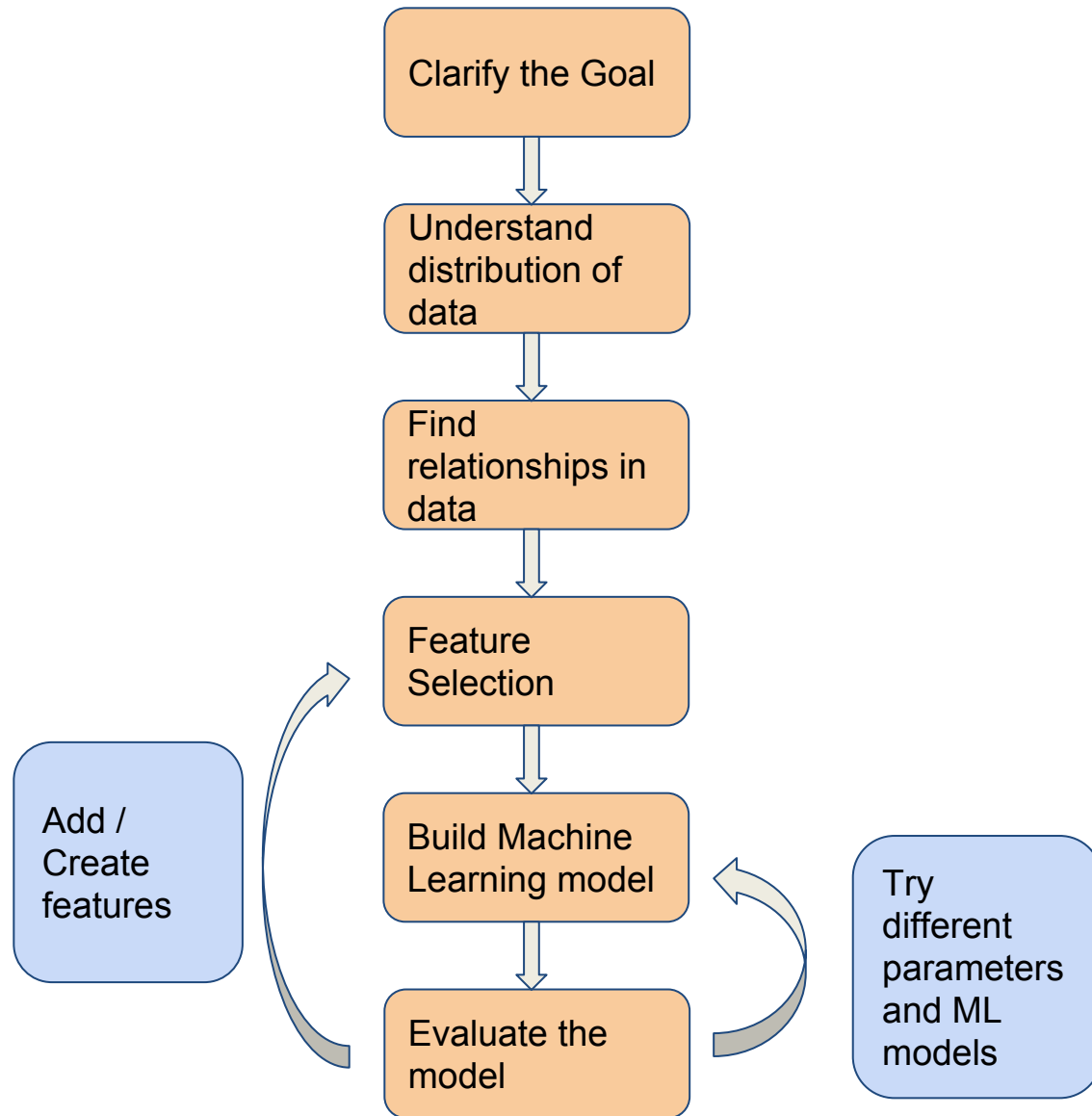
Create new features

1. Add new features that might give model more information
 - a. Example: You have total time spent per user, and total number of visits
 - b. Create a new feature that is average time spent per visit

Obtain new data

1. Limitations of data
 - a. None of the features are that helpful at all, consider augmenting with new dataset

Building an ML Model



Evaluation Metrics

How do we measure a successful model?

Evaluation Metrics

Accuracy

Concept of False Positives / False Negatives

Precision

Recall

F1 Score

Accuracy

Actual Label	Predicted Label
0	0
1	1
0	1
0	1
1	1
0	0
1	1
0	1
1	0
0	1

$$\frac{\text{Number of Correct Predictions}}{\text{Number of Total Predictions}}$$

Accuracy

Actual Label	Predicted Label
0	0
1	1
0	1
0	1
1	1
0	0
1	1
0	1
1	0
0	1

$$\frac{\text{Number of Correct Predictions}}{\text{Number of Total Predictions}}$$

$$\text{Accuracy} = 5/10 = 0.5$$

Limitations

Example: For a cancer dataset, imagine if we had a dataset that had 900 negative labels and only 100 positive labels.

We can simply build a model to predict all negative labels and still achieve 90% accuracy. Accuracy is not helpful in this case, we need other metrics.

False Positives / False Negatives

Actual Label	Predicted Label
0	0
1	1
0	1
0	1
1	1
0	0
1	1
0	1
1	0
0	1

		Actual Label	
		Positive [1]	Negative [0]
Predicted Label	Positive [1]	True Positive (TP)	False Positive (FP)
	Negative [0]	False Negative (FN)	True Negative (TN)

False Positives / False Negatives

Actual Label	Predicted Label
0	0
1	1
0	1
0	1
1	1
0	0
1	1
0	1
1	0
0	1

		Actual Label	
		Positive [1]	Negative [0]
Predicted Label	Positive [1]	True Positive (TP) 3	False Positive (FP) 4
	Negative [0]	False Negative (FN) 1	True Negative (TN) 2

Precision

Actual Label	Predicted Label
0	0
1	1
0	1
0	1
1	1
0	0
1	1
0	1
1	0
0	1

Intuition: What percent of the positive predictions were correct?

$$\begin{aligned}\text{Precision} &= \frac{TP}{TP + FP} \\ &= 3 / (3 + 4) \\ &= 3 / 7 \\ &= 0.42\end{aligned}$$

Recall

Actual Label	Predicted Label
0	0
1	1
0	1
0	1
1	1
0	0
1	1
0	1
1	0
0	1

Intuition: What percent of positive cases were caught?

Recall =

$$\frac{TP}{TP + FN}$$

$$= 3 / (3 + 1)$$

$$= 3 / 4$$

$$= 0.75$$

F1 Score

Actual Label	Predicted Label
0	0
1	1
0	1
0	1
1	1
0	0
1	1
0	1
1	0
0	1

Intuition: Combines precision and recall to give an overall score.

$$\text{F1 Score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

$$= 2 * (0.42 * 0.75) / (0.42 + 0.75)$$

$$= 0.5384$$

Standard evaluation metric

Recap

Example: For a cancer dataset, imagine if we had a dataset that had 900 negative labels and only 100 positive labels.

Accuracy

What percent of your predictions were correct? **90%**

This means that 100 of your patients who have cancer will go undiagnosed!

Precision

What percent of the positive predictions were correct? **0%**

If you are the doctor, you want precision to be as high as possible, but having a low precision is still “fine”. Better to be safe than sorry.

Recall

What percent of positive cases were caught? **0%**

If you are a doctor, you want your recall to be very high else someone who has cancer might go undiagnosed.

Evaluation Metrics Summary

We can see that in different contexts might emphasize accuracy / precision / recall differently.

It boils down to the real cost of a False Positive and False Negative.

- Some people might weight false positives more than false negatives
 - (e.g. marketing pregant content to expectant mothers, false positive might lead to disastrous PR)
- While some might weight false negatives more
 - (e.g. doctor failing to correctly diagnose patients with cancer)

Unsupervised Learning

Finding inherent patterns and structures in data

ML Task Families

Supervised Learning

“Use labeled data to train a model that makes output predictions from the input data”

- Regression
- Classification
- ...

Unsupervised Learning

“Find inherent patterns and structures in unlabeled data”

- Frequent Itemsets
/ Association Rules
- Clustering
- ...

Other ML Families (not covered in CS102)

Semi-supervised Learning (*data partially labelled*)

Reinforcement learning (*using feedback from the environment to update decision making strategy/policy*)

Supervised vs Unsupervised

Labeled vs unlabeled data

Supervised Learning

*“Use **labeled data** to train a model that makes output predictions from the input data”*

- Regression
- Classification
- ...

Unsupervised Learning

*“Find inherent patterns and structures in **unlabeled data**”*

- Frequent Itemsets
/ Association Rules
- Clustering
- ...

Data Mining

- Frequent Itemsets and Association Rules fall under unsupervised learning






















Transaction 1	   
Transaction 2	  
Transaction 3	 
Transaction 4	 
Transaction 5	   
Transaction 6	  
Transaction 7	 
Transaction 8	

Table 1. Example transactions.

Clustering

Like classification, data items consist of **values** for a set of **features** (numeric or categorical)

- Medical patients

Feature values: age, gender, symptom1-severity, symptom2-severity, test-result1, test-result2

- Web pages

Feature values: URL domain, length, #images, heading₁, heading₂, ..., heading_n

- Products

Feature values: category, name, size, weight, price

Clustering

Like classification, data items consist of **values** for a set of **features** (numeric or categorical)

- Medical patients

Feature values: age, gender, symptom1-severity, symptom2-severity, test-result1, test-result2, ...

Unlike classification,
there is no **label**

- Web pages

Feature values: URL domain, length, #images, heading₁, heading₂, ..., heading_n

- Products

Feature values: category, name, size, weight, price

Clustering

Like K-nearest neighbors, for any pair of data items i_1 and i_2 , from their feature values can compute distance function: $distance(i_1, i_2)$

Example:

Features - gender, profession, age, income, postal-code

person₁ = (male, teacher, 47, \$25K, 94305)

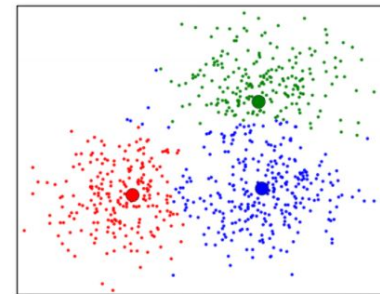
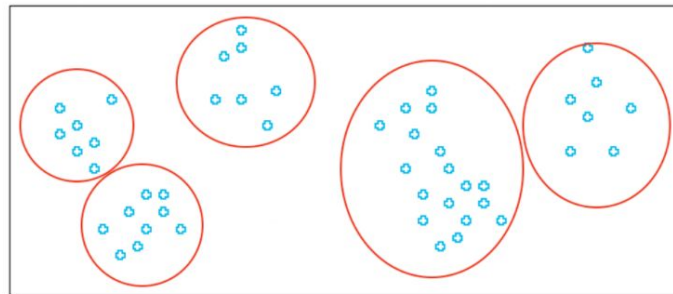
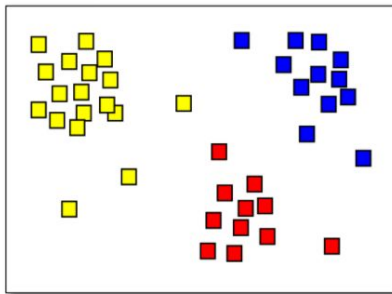
person₂ = (female, teacher, 43, \$28K, 94309)

$distance(\text{person}_1, \text{person}_2)$

Clustering

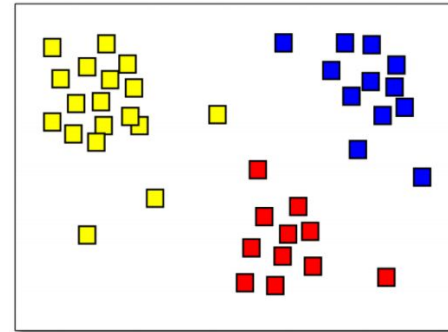
GOAL: Given a set of data items, partition them into groups (= clusters) so that items within groups are close to each other based on distance function

- Sometimes number of clusters is pre-specified
- Typically clusters need not be same size



Some Uses for Clustering

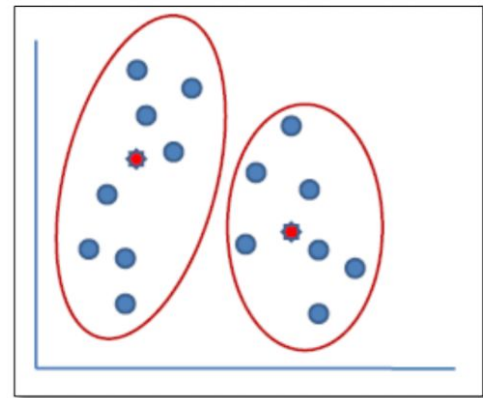
- Classification!
 - Assign labels to clusters
 - New data items get the label of their cluster
- Identify similar items
 - For substitutes or recommendations
 - For de-duplication
- Anomaly (outlier) detection
 - Items that are far from any cluster



K-Means Clustering

Reminder: for any pair of data items i_1 and i_2 have $distance(i_1, i_2)$

For a group of items, the **mean value (centroid)** of the group is the item i (in the group or not) that minimizes the sum of $distance(i, i')$ for all i' in the group

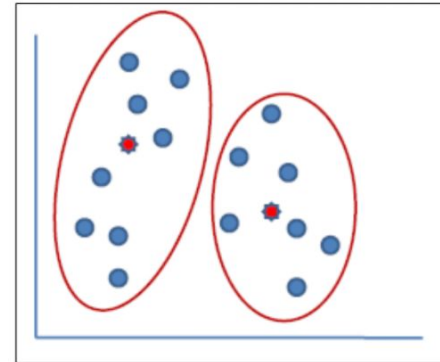


K-Means Clustering

For a group of items, the **mean value (centroid)** of the group is the item i (in the group or not) that minimizes the sum of $distance(i, i')$ for all i' in the group

- **Error** for each item: distance d from the mean for its group; **squared error** is d^2
- **Error** for the entire clustering: **sum of squared errors (SSE)**

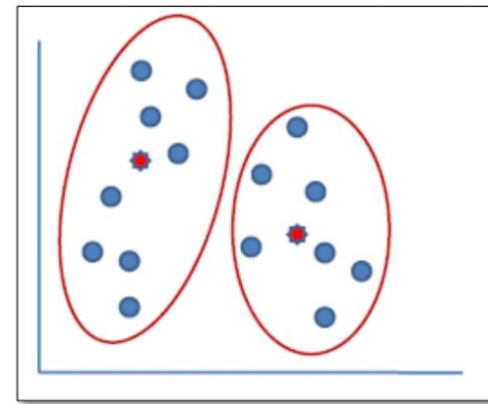
Remind you of anything?



K-Means Clustering

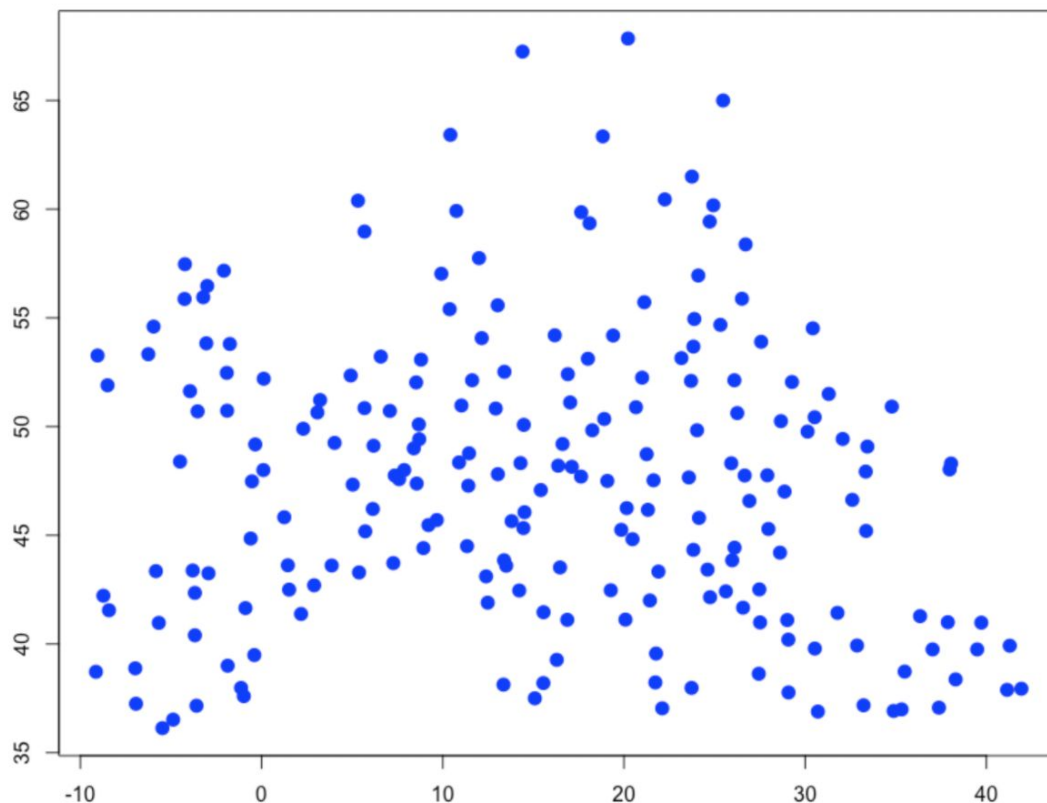
Given set of data items and desired number of clusters k , K-means groups the items into k clusters minimizing the SSE

- Extremely difficult to compute efficiently
 - In fact, impossible
- Most algorithms compute an **approximate** solution (might not be absolute lowest SSE)



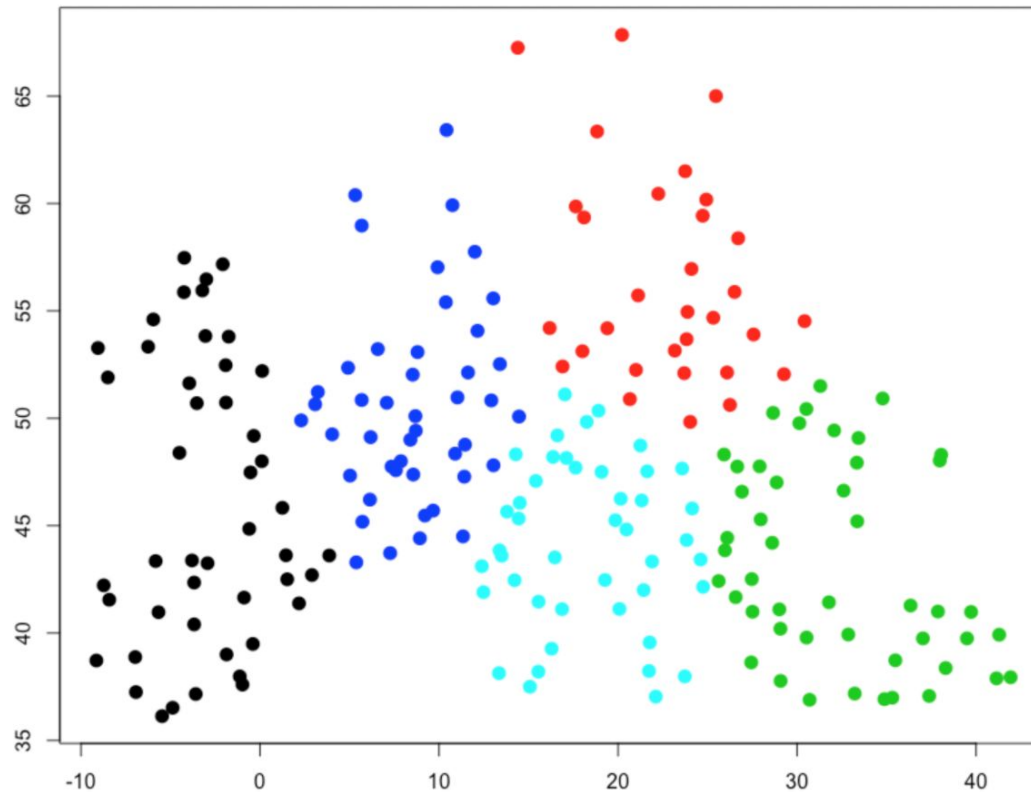
Clustering European Cities

By geographic distance, then by temperature



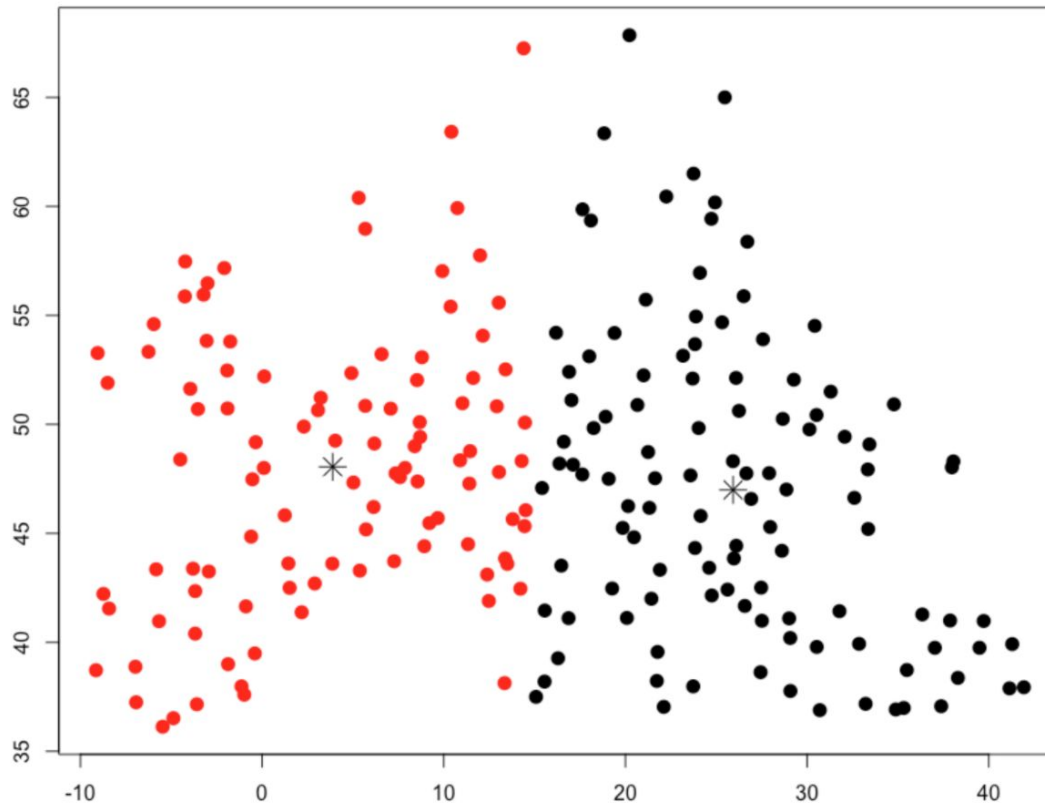
Clustering European Cities

Distance = actual distance, $k = 5$



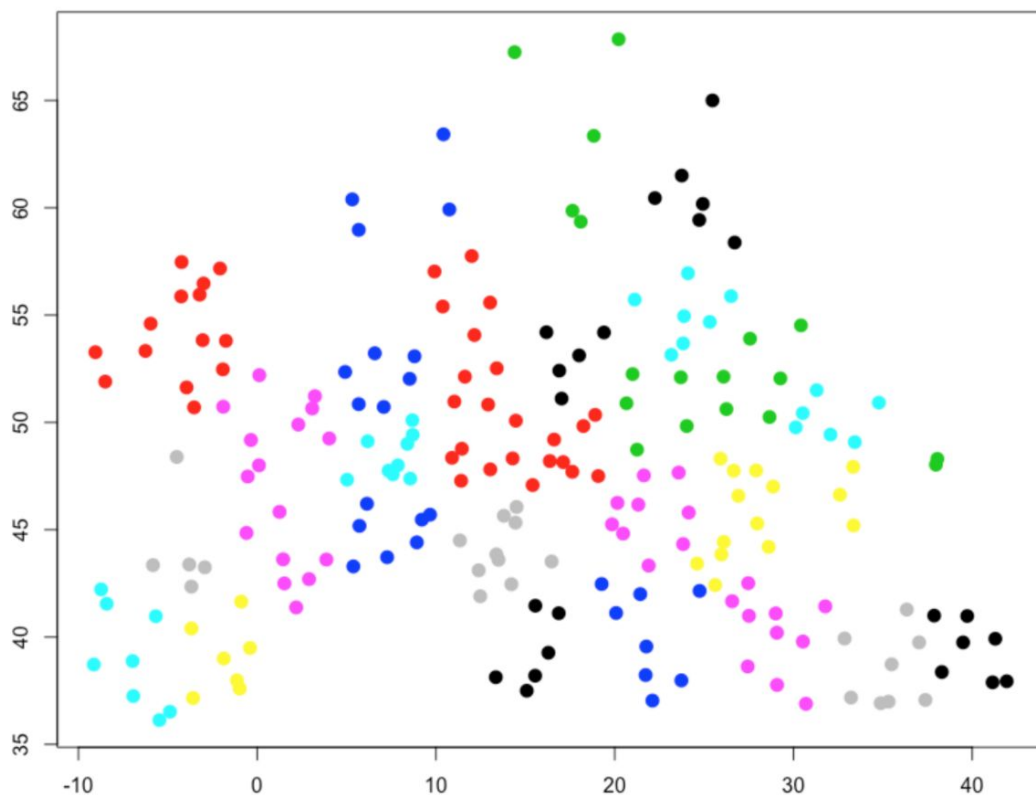
Clustering European Cities

Distance = actual distance, $k = 2$, with cluster means



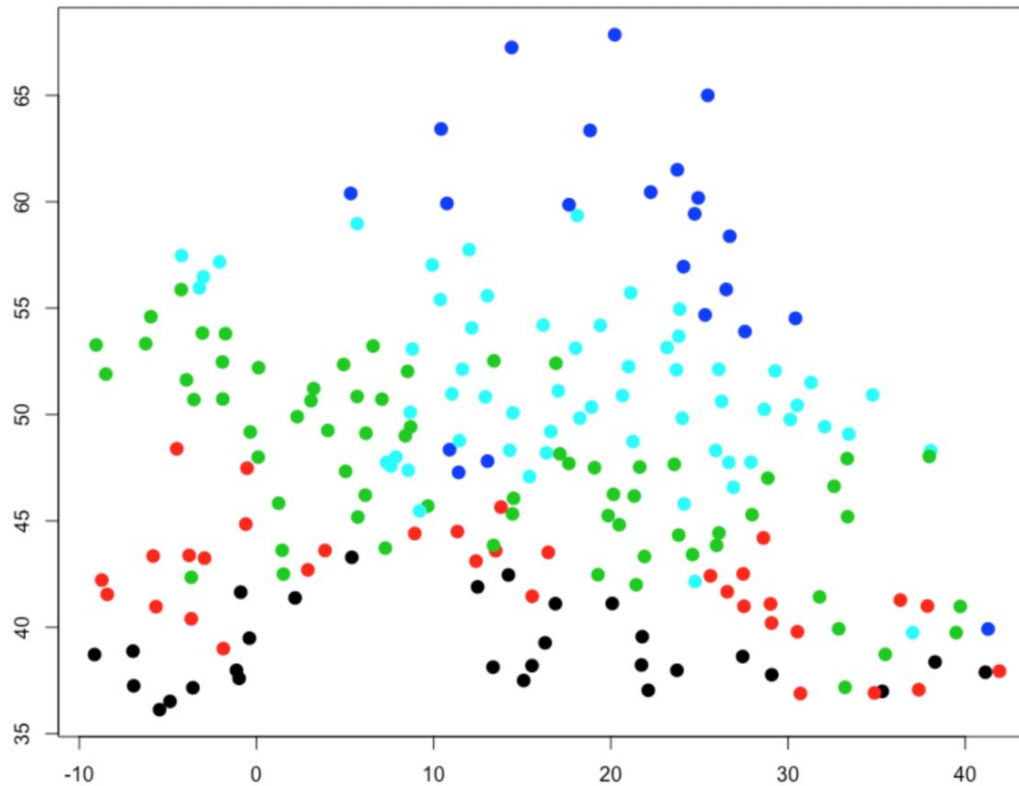
Clustering European Cities

Distance = actual distance, $k = 30$



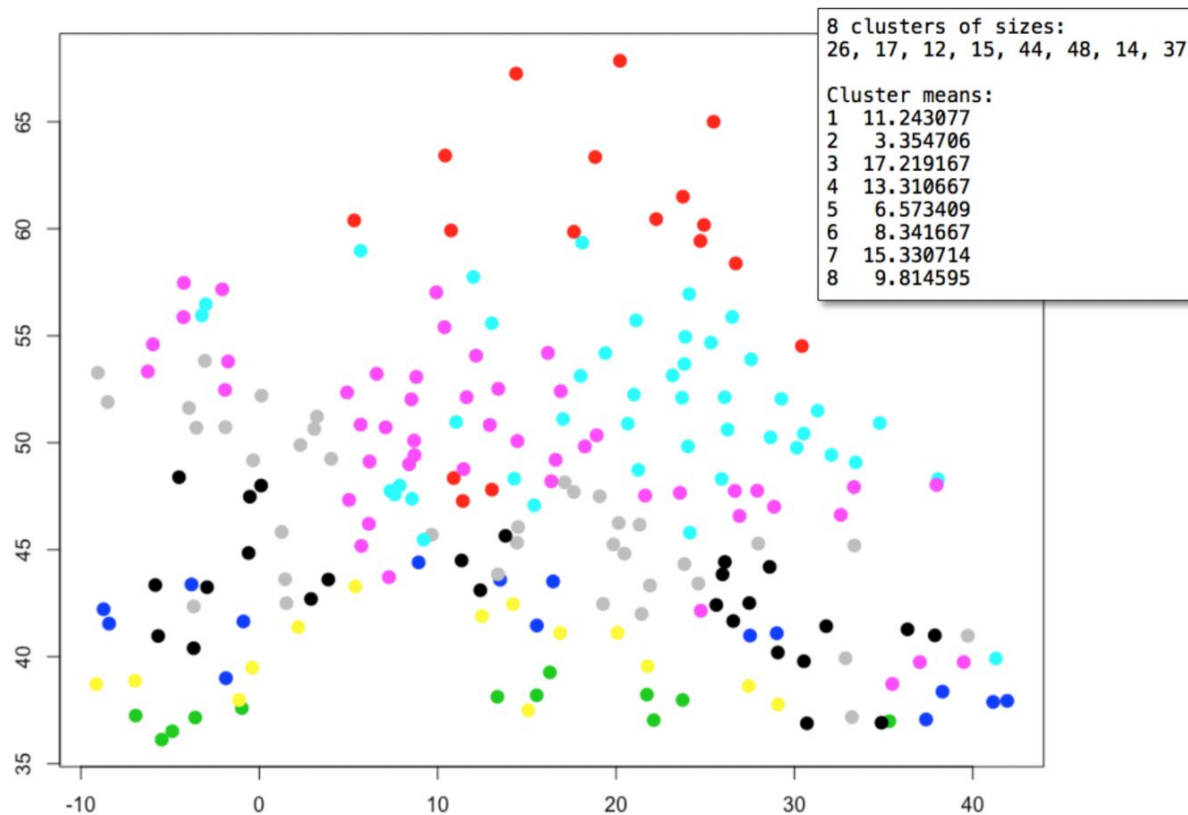
Clustering European Cities

Distance = temperature, $k = 5$



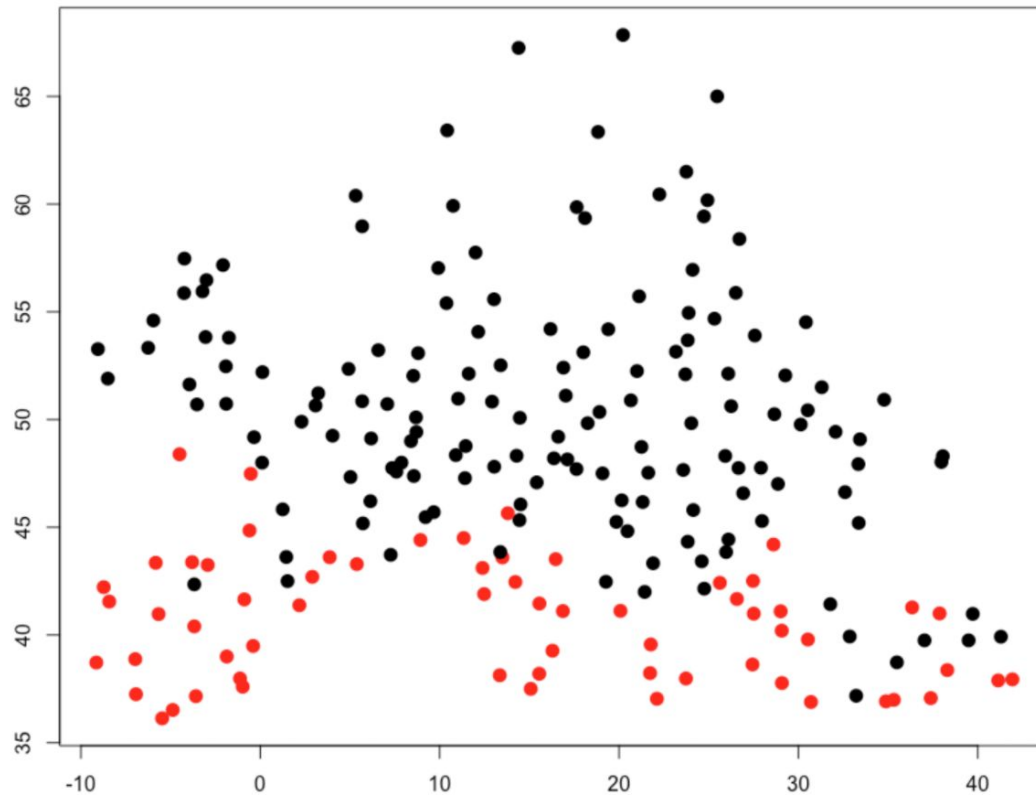
Clustering European Cities

Distance = temperature, $k = 8$, with means



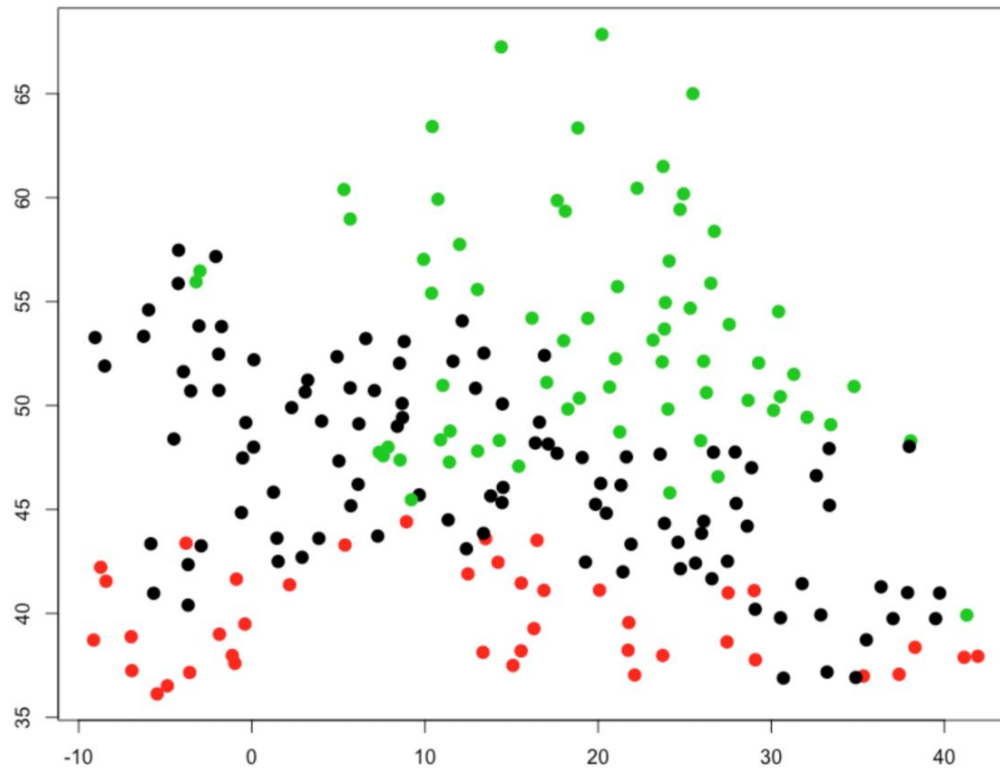
Clustering European Cities

Distance = temperature, $k = 2$



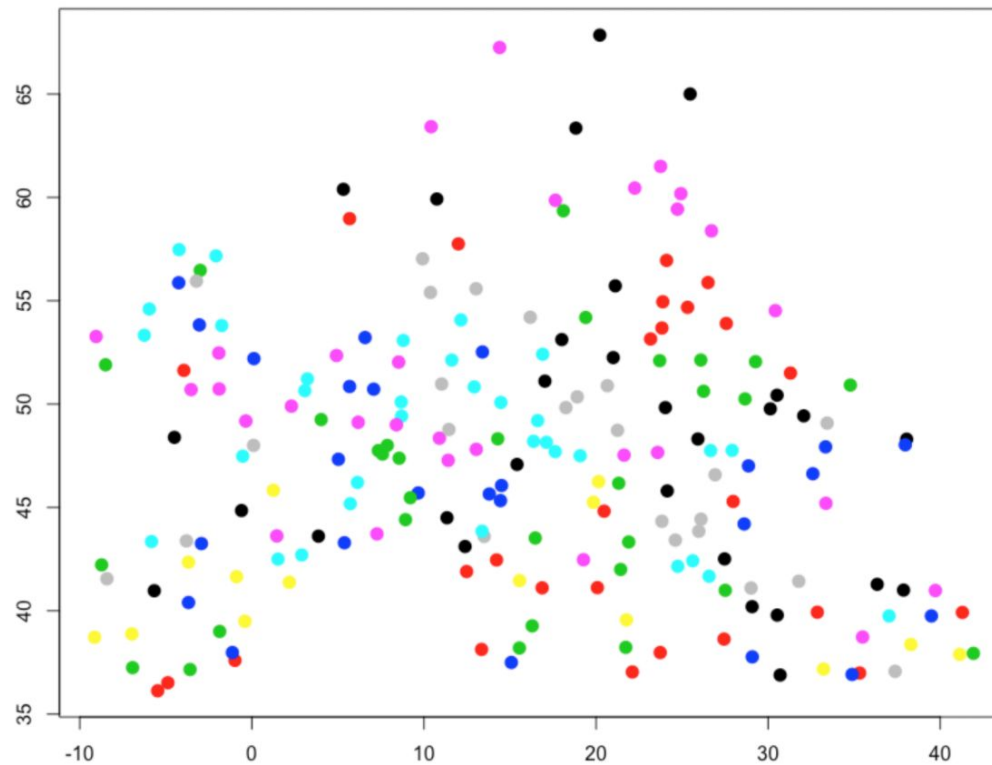
Clustering European Cities

Distance = temperature, $k = 3$



Clustering European Cities

Distance = temperature, $k = 30$



Use Cases of Clustering

- Classification
 - Assign labels to clusters
 - New data items get the label of their cluster
- Identify similar items
 - For substitutes or recommendations
 - For de-duplication
- Anomaly (outlier) detection
 - Items that are far from any cluster

Unsupervised Learning Summary

- Input data is unlabeled
- Goal is to discover patterns and structures present in the data itself
- Data Mining
- Clustering can be used for:
 - Categorization
 - You don't know what categories are, algorithm helps you determine it
 - Anomaly detection
 - Data points that are far away from any cluster can be counted as outliers