GENERAL ASSEMBLY

INTRO TO CLASSIFICATION

Adam Jones, PhD

Data Scientist @ Critical Juncture

LEARNING OBJECTIVES

- Define class label and classification
- Build a K-Nearest Neighbors using the sci-kit-learn library
- Evaluate and tune model by using metrics such as classification accuracy/error

COURSE

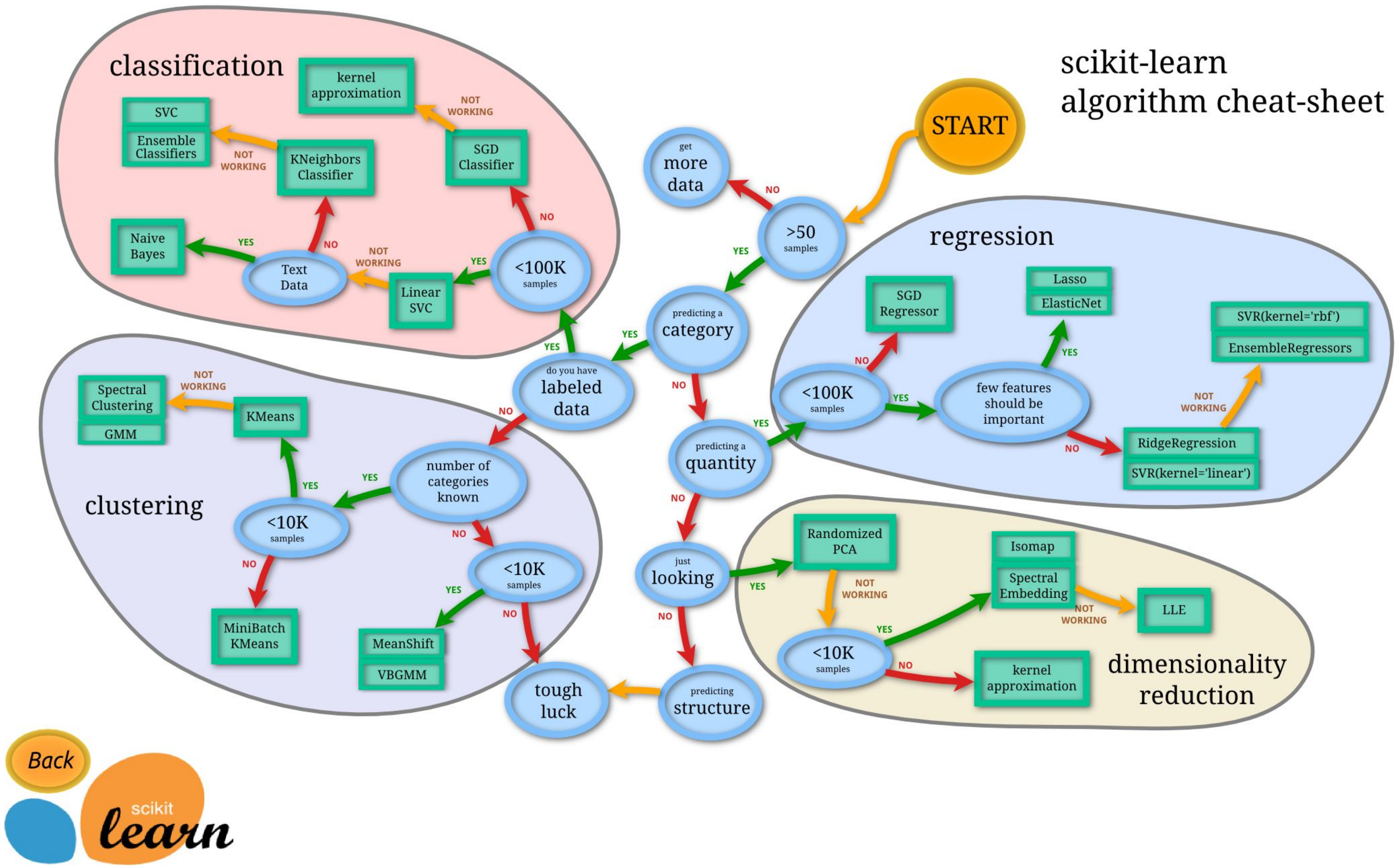
PRE-WORK

PRE-WORK REVIEW

- Understand how to optimize for error in a model
- Understand the concept of iteration to solve problems
- Measure basic probability

OPENING

INTRO TO CLASSIFICATION


scikit-learn
algorithm cheat-sheet
START
>50 samples
get more data
predicting a category
do you have labeled data
predicting a quantity
just looking
predicting structure
tough luck
classification
<100K samples
SGD Classifier
kernel approximation
Linear SVC
Text Data
Naive Bayes
KNeighbors Classifier
SVC
Ensemble Classifiers
clustering
number of categories known
<10K samples
KMeans
Spectral Clustering
GMM
MiniBatch KMeans
MeanShift
VBGMM
regression
SGD Regressor
few features should be important
Lasso
ElasticNet
RidgeRegression
SVR(kernel='linear')
SVR(kernel='rbf')
EnsembleRegressors
dimensionality reduction
Randomized PCA
Isomap
Spectral Embedding
LLE
YES
NO
NOT WORKING
Back
scikit learn

INTRO TO CLASSIFICATION

- So far, we’ve worked primarily with regression problems
    - We’ve focused on predicting a continuous set of values

- That means we’ve been able to use distance to measure how accurate our prediction is

- However, for other problems, we need to predict binary responses
    - E.g.: A loan will default or it won’t. An email is spam or isn’t spam.

ACTIVITY: KNOWLEDGE CHECK

EXERCISE

ANSWER THE FOLLOWING QUESTIONS

1. What if we want to build a model to predict a set of values, like a photo color or the gender of a baby?
2. Can we use regression for binary values?
3. Do the same principles apply?

DELIVERABLE

Answers to the above questions

INTRODUCTION

WHAT IS CLASSIFICATION?

WHAT IS CLASSIFICATION?

- **Classification** is a machine learning problem for solving a set value given the knowledge we have about that value
- Many classification problems are trying to predict *binary* values
- For example, we may be using patient data (medical history) to predict whether the patient is a smoker or not

WHAT IS CLASSIFICATION?

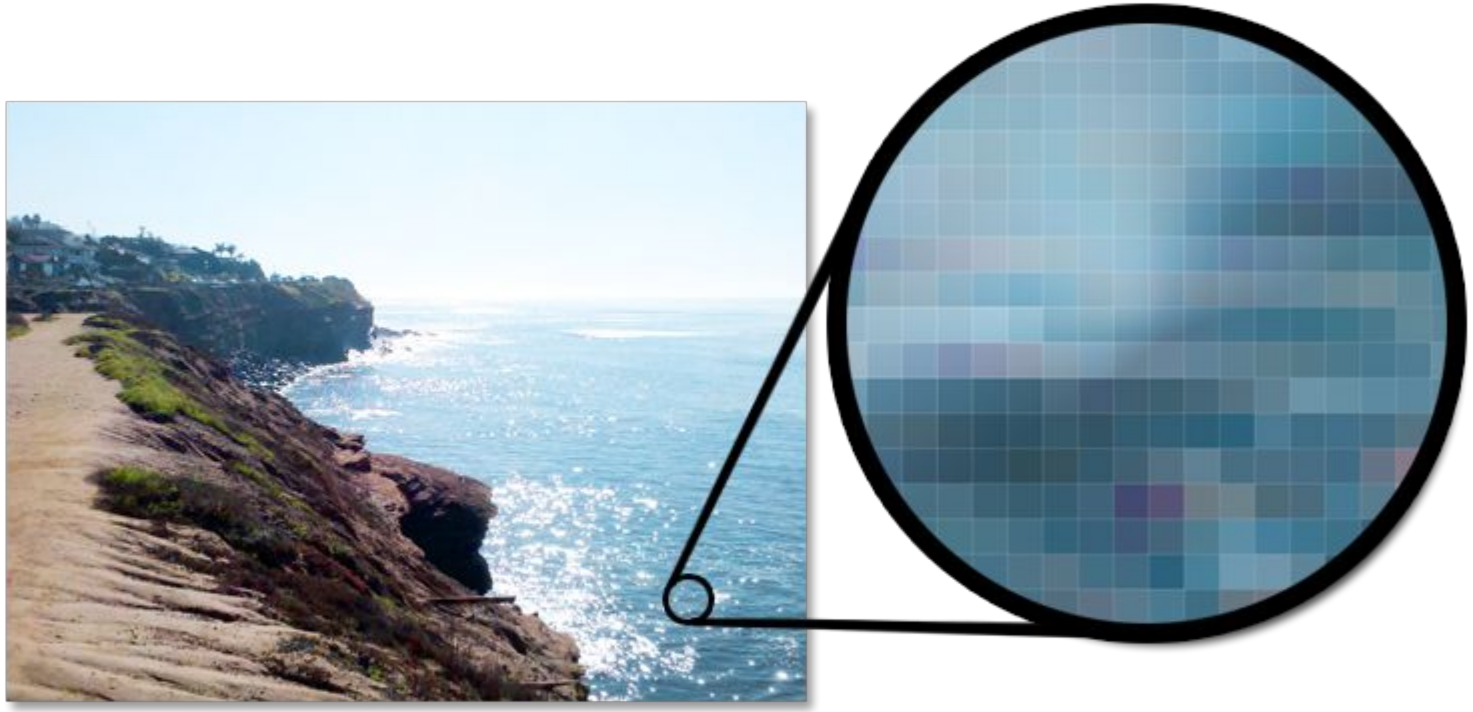
- Some problems don’t appear to be binary at first glance

- However, you can boil down the response to a *boolean* (true/false) value

- What if you are predicting whether an image pixel will be `red` or `blue`?

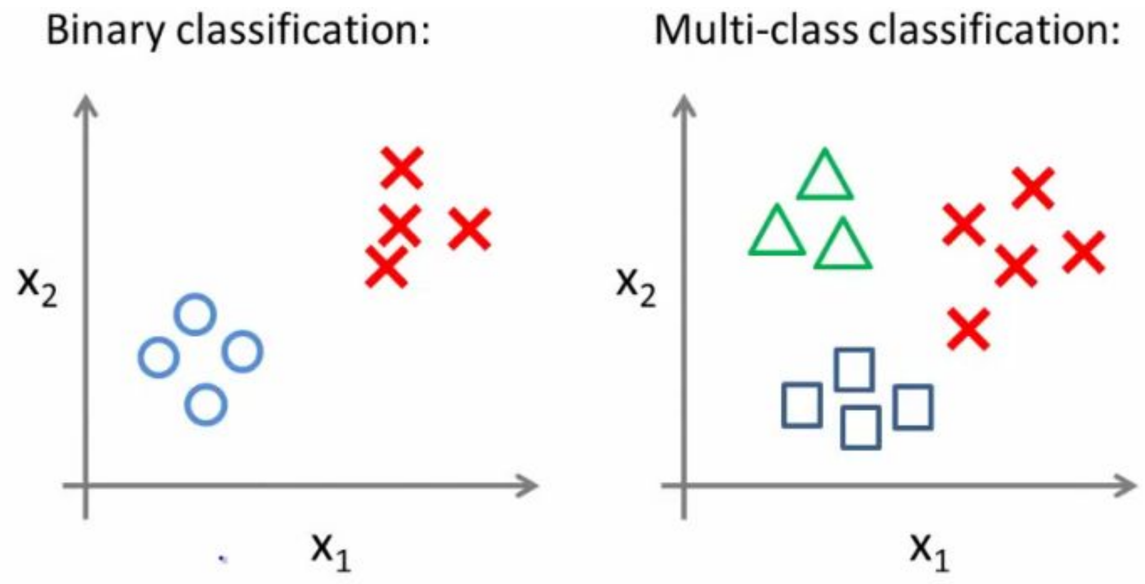
- We don’t need to predict that a pixel is `blue`, just that it is not `red`

- This is similar to the concept of dummy variables



WHAT IS CLASSIFICATION?

- Binary classification is the simplest form of classification
- However, classification problems can have multiple *class labels*
- Instead of predicting whether the pixel is red or blue, you could predict whether the pixel is red, blue, or green



WHAT IS A CLASS LABEL?

- A **class label** is a representation of what we are trying to predict: our *target*
- Examples of class labels from before are:

Data Problem	**Class Labels**
Patient data problem	is smoker, is not smoker
pixel color	red, blue, green

DETERMINING REGRESSION OR CLASSIFICATION

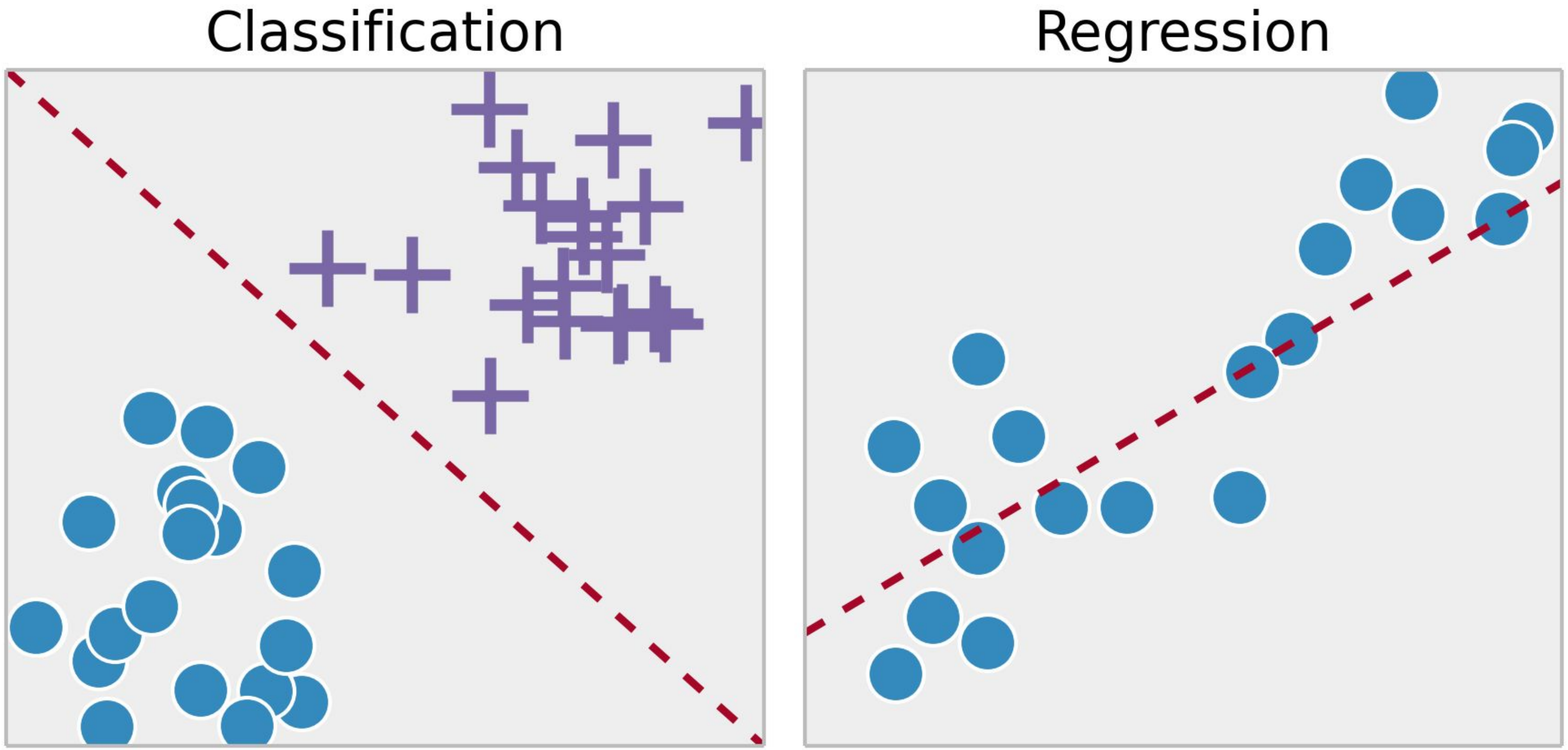
- To determine if a problem is regression or classification
  - determine if our *target* variable can be ordered mathematically

- For example, if predicting company revenue, `$100MM` is greater than `$90MM`
  - This is a *regression* problem because the target can be ordered

- However, if predicting pixel color, `red` is not inherently greater than `blue`
  - Therefore, this is a *classification* problem

DETERMINING REGRESSION OR CLASSIFICATION

- Classification and regression differ in what you are trying to predict



GUIDED PRACTICE

REGRESSION OR CLASSIFICATION?

ACTIVITY: REGRESSION OR CLASSIFICATION?

EXERCISE

DIRECTIONS (20 minutes)

Review the following situations and decide if each one is a regression problem, classification problem, or neither:

1. Using the total number of explosions in a movie, predict if the movie is by JJ Abrams or Michael Bay
2. Determine how many tickets will be sold for a concert given who is performing, where, and the date and time
3. Given the temperature over the last year by day, predict tomorrow's temperature outside
4. Using data from four cell phone microphones, reduce the noisy sounds so the voice is crystal clear to the receiving phone
5. With customer data, determine if a user will return or not in the next 7 days to an e-commerce website

DELIVERABLE

Answers to the above questions

INDEPENDENT PRACTICE

BUILD A CLASSIFIER!

ACTIVITY: BUILD A CLASSIFIER!

EXERCISE

DIRECTIONS (20 minutes)

1. Re-explore the iris dataset and build a program that classifies each data point
    - Use if-else statements and some Pandas functions
2. Measure the *accuracy* of your classifier using the math of “total correct” over “total samples”
3. Your classifier should be able to:
    a. Get one class label 100% correct (one type of iris is easily distinguishable from the other two)
    b. Accurately predict the majority of the other two classes with some error (hint: make sure you *generalize*)

DELIVERABLE

Classification program for the iris dataset

ACTIVITY: BUILD A CLASSIFIER!

EXERCISE

DIRECTIONS (20 minutes)

1. Re-explore the iris dataset and build a program that classifies each data point
   - Use if-else statements and some Pandas functions
2. Measure the *accuracy* of your classifier using the math of “total correct” over “total samples”
3. Your classifier should be able to:
   a. Get one class label 100% correct (one type of iris is easily distinguishable from the other two)
   b. Accurately predict the majority of the other two classes with some error (hint: make sure you *generalize*)

DELIVERABLE

Classification program for the iris dataset

ACTIVITY: BUILD A CLASSIFIER!

EXERCISE

STARTER CODE

```
from sklearn import datasets, neighbors, metrics
import pandas as pd

iris = datasets.load_iris()
irisdf =
pd.DataFrame(iris.data,columns=iris.feature_names)
irisdf['target'] = iris.target
cmap = {0: 'r', 1: 'g', 2: 'b' }
irisdf['ctarget'] =
irisdf.target.apply(lambda x: cmap[x])
```

ACTIVITY: BUILD A CLASSIFIER!

EXERCISE

STARTER CODE

```
irisdf.plot('petal length (cm)', 'petal width (cm)',
kind='scatter', c=irisdf.ctarget)

print(irisdf.describe())
```

ACTIVITY: BUILD A CLASSIFIER!

EXERCISE

STARTER CODE

```
# starter code
def my_classifier(row):
    if row['petal length (cm)'] < 2:
        return 0
    else:
        return 1

predictions = irisdf.apply(my_classifier, axis=1)
```

ACTIVITY: BUILD A CLASSIFIER!

EXERCISE

DIRECTIONS

Answer the following questions.

1. How simple could the if-else classifier be while remaining *relatively* accurate?
2. How complicated could our if-else classifier be and remain *completely* accurate? How many if-else statements would you need, or nested if-else statements, in order to get the classifier 100% accurate?
3. Which if-else classifier would work better against iris data that it hasn't seen? Why is that the case?

DELIVERABLE

Answers to the above questions

INTRODUCTION

WHAT IS K NEAREST NEIGHBORS?

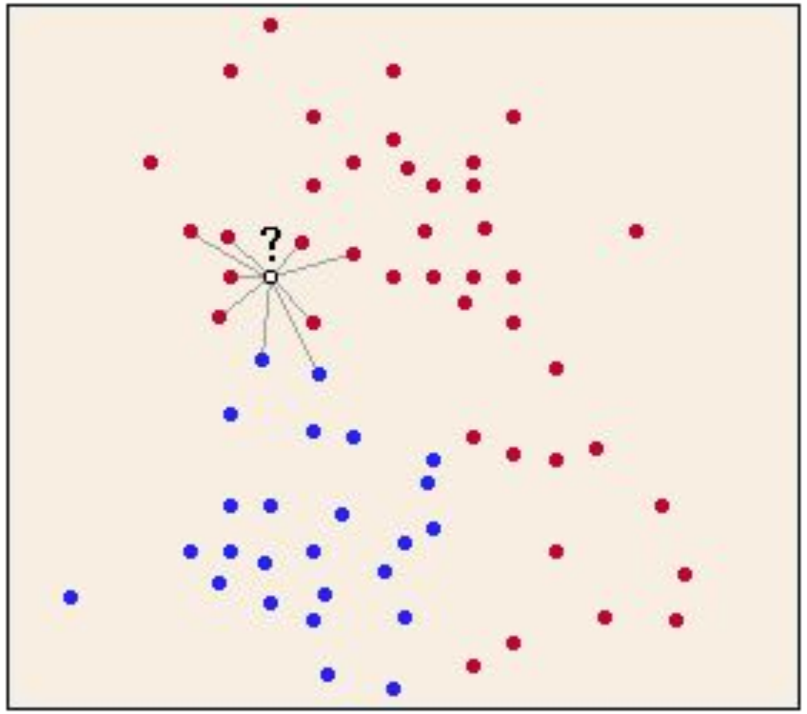
WHAT IS K NEAREST NEIGHBORS?

- **K Nearest Neighbors (KNN)** is a classification algorithm that makes a prediction based upon the closest data points

- The KNN algorithm:
  - For a given point, calculate the distance to all other points
  - Given those distances, pick the *k* closest points
  - Calculate the probability of each class label given those points
  - The original point is classified as the class label with the largest probability ("votes")

WHAT IS K NEAREST NEIGHBORS?

- KNN uses distance to predict a class label
  - This application of distance is used as a measure of similarity between classifications
- We’re using shared traits to identify the most likely class label



WHAT IS K NEAREST NEIGHBORS?

- Suppose we want to determine your favorite type of music
  - How might we determine this without directly asking you?
- Generally, friends share similar traits and interests (e.g. music, sports teams, hobbies, etc)
  - We could ask your five closest friends what their favorite type of music is and take the majority vote
- This is the idea behind KNN: we look for things similar to (or close to) our new observation and identify shared traits
  - We can use this information to make an educated guess about a trait of our new observation

ACTIVITY: KNOWLEDGE CHECK

EXERCISE

ANSWER THE FOLLOWING QUESTIONS

1. What is K Nearest Neighbors?

DELIVERABLE

Answers to the above questions

DEMO

KNN IN ACTION

KNN IN ACTION

- The following code demonstrates using KNN via sklearn

```
from sklearn import datasets, neighbors, metrics
import pandas as pd

iris = datasets.load_iris()
# n_neighbors is our option in KNN. We'll tune this value to attempt
to improve our prediction.
knn = neighbors.KNeighborsClassifier(n_neighbors=5, weights='uniform')
knn.fit(iris.data[:,2:], iris.target)
print knn.predict(iris.data[:,2:])
print iris.target
print knn.score(iris.data[:,2:], iris.target)
```

WHAT HAPPENS IN TIES?

- What happens if two classes get the same number of votes?

- This could happen in binary classification if we use an even number for *k*
  - This could also happen if there are multiple class labels

- In sklearn, it will choose the class that it first saw in the *training set*

WHAT HAPPENS IN TIES?

- We could implement a *weight*
  - taking into account the distance between the point and its neighbors

- This can be done in sklearn by changing the `weights` parameter to "`distance`"

- We can change the `weights` parameter

  - To see how it affects accuracy

WHAT HAPPENS IN HIGH DIMENSIONALITY?

- Since KNN works with distance, higher dimensionality of data (i.e. more features) requires *significantly* more samples in order to have the same predictive power

- Because, with more dimensions, all points slowly start averaging out to be equally distant
  - This causes significant issues for KNN

- Keep the feature space limited and KNN will do well
  - Exclude extraneous features when using KNN

WHAT HAPPENS IN HIGH DIMENSIONALITY?

- Consider two different examples:
  a. classifying users of a newspaper
  b. users of a particular toothpaste

- The features of the *newspapers* are very *broad* and there are many:
  - sections, topics, types of stories, writers, online vs print, etc.

- However, the features of a *toothpaste* are more *narrow*:
  - has fluoride, controls tartar, etc.

- For which problem would KNN work better?

WHAT HAPPENS IN HIGH DIMENSIONALITY?

- KNN would work better on classifying users of a particular toothpaste since the feature set is more narrow and distinct

INTRODUCTION

CLASSIFICATION METRICS

INTRODUCTION TO CLASSIFICATION METRICS

- Metrics for regression do **not** apply to classification

- We *could* measure the distance between the probability of a given class and an item being in that class
  - Guessing 0.6 for a 1 is a 0.5 error

- But this overcomplicates our goal: understanding binary classification, whether something is black or white, right or wrong

- To do this, we'll measure "correctness" or "incorrectness"

INTRODUCTION TO CLASSIFICATION METRICS

- We'll use two primary metrics: *accuracy* and *misclassification rate*
- **Accuracy** is the number of *correct* predictions out of all predictions in the sample
  - This is a value we want to *maximize*
- **Misclassification rate** is the number of *incorrect* predictions out of all predictions in the sample
  - This is a value we want to *minimize*
- These two metrics are directly opposite of each other
- 1 - misclassification rate = accuracy

INTRODUCTION TO CLASSIFICATION METRICS

- **WARNING**: You cannot use regression evaluation metrics for a classification problem, or vice versa
  - This is a *common mistake*

- sklearn will not intuitively understand if you are doing regression or classification,
  - so make sure to manually review your metrics

INDEPENDENT PRACTICE

SOLVING FOR K

ACTIVITY: SOLVING FOR K

EXERCISE

DIRECTIONS (35 minutes)

One of the primary challenges of KNN is solving for k - how many neighbors do we use?

The **smallest** k we can use is 1. However, using only one neighbor will probably perform poorly.

The largest k we can use is n-1 (every other point in the data set). This would also perform poorly.

Use the lesson 8 starter code and the iris data set to answer the following questions:

1. What is the accuracy for k=1?
2. What is the accuracy for k=n-1?
3. Using cross validation, what value of k optimizes model accuracy
    - Create a plot with *k* as the x-axis and *accuracy* as the y-axis (called a “fit chart”) to help find the answer

DELIVERABLE

Answers to the above questions

ACTIVITY: SOLVING FOR K

EXERCISE

STARTER CODE

```
from sklearn import grid_search

params = {'n_neighbors': }

gs = grid_search.GridSearchCV(
    estimator=,
    param_grid=,
    cv=,
)
gs.fit(iris.data, iris.target)
gs.grid_scores_
```

ACTIVITY: SOLVING FOR K

EXERCISE

DIRECTIONS

Bonus Questions:

1. Explore the distance metrics that are available to use with the KNN classifier in sklearn
   a. What *type* of data does this metric work best for?
   b. What *type* of data does this distance metric not work for?

- You can read about distance metrics in the sklearn documentation

DELIVERABLE

Answers to the above questions

CONCLUSION

TOPIC REVIEW

REVIEW

- What are class labels? What does it mean to classify?
- How is a classification problem different from a regression problem? How are they similar?
- How does the KNN algorithm work?
- What primary parameters are available for tuning a KNN estimator?
- How do you define: accuracy, misclassification?

COURSE

BEFORE NEXT CLASS

BEFORE NEXT CLASS

DUE DATE

- Unit 3 Project due: Thurs - 4/19

LESSON

Q & A

LESSON

EXIT TICKET

DON’T FORGET TO FILL OUT YOUR EXIT TICKET