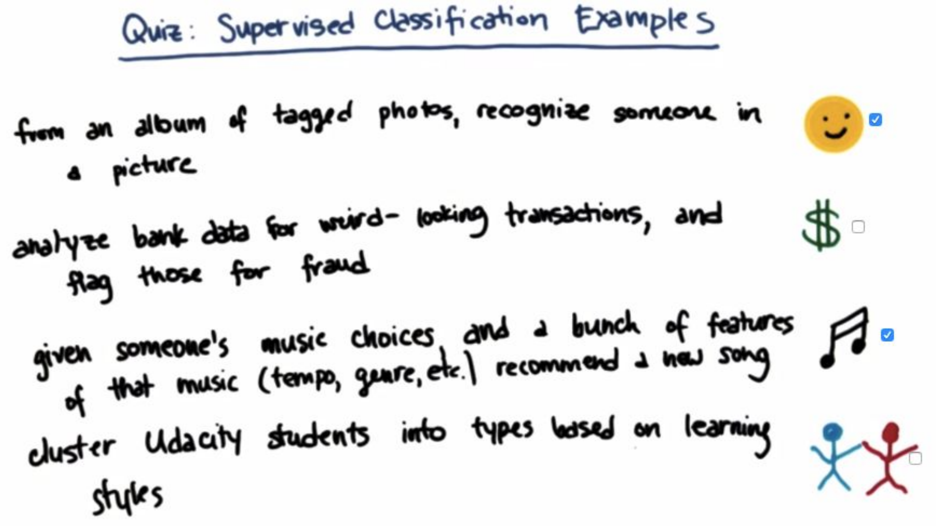
Naive Bayes Notes

Lesson 3



Quiz: Supervised Classification Examples

from an album of tagged photos, recognize someone in a picture ☑

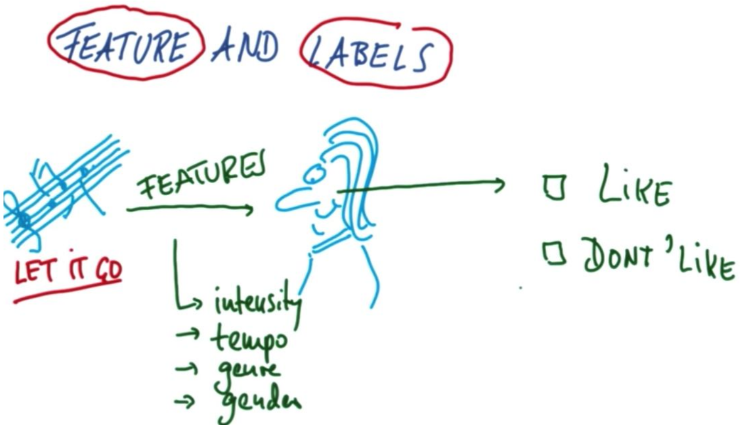
analyze bank data for weird-looking transactions, and flag those for fraud □

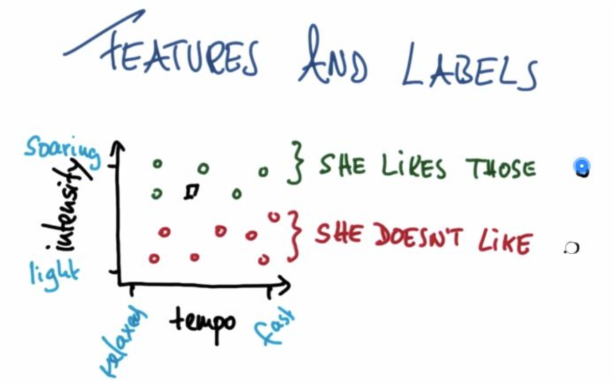
given someone's music choices and a bunch of features of that music (tempo, genre, etc.) recommend a new song ☑

cluster Udacity students into types based on learning styles □

Lesson 4-5

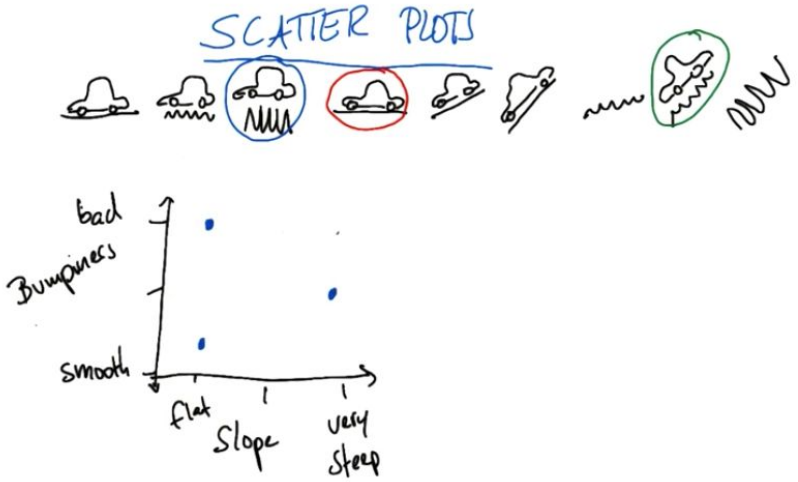
- In Machine Learning, we take in **Features** as inputs and we use them to produce **Labels.**





Lesson 8-10

- Instead of looking at raw data, we plot it in a scatter graph where all data points are converted to points on a graph.

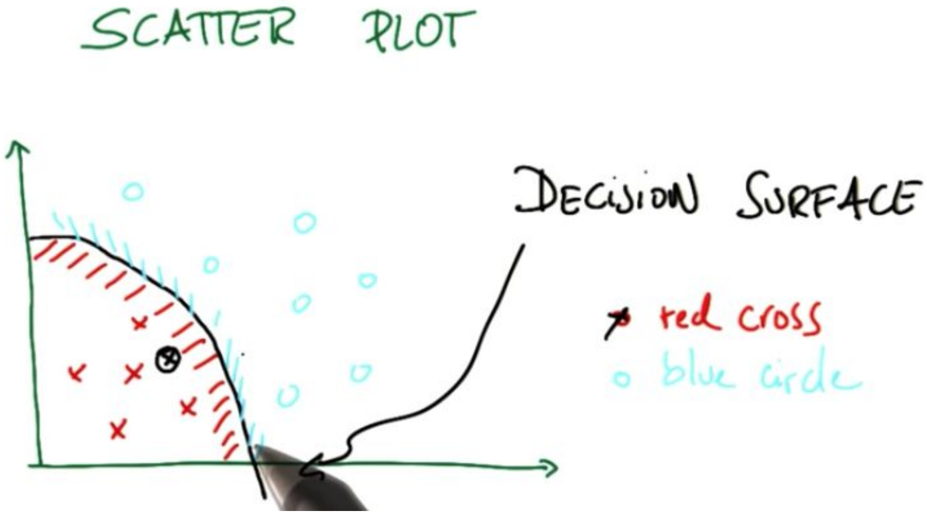


Lesson 11

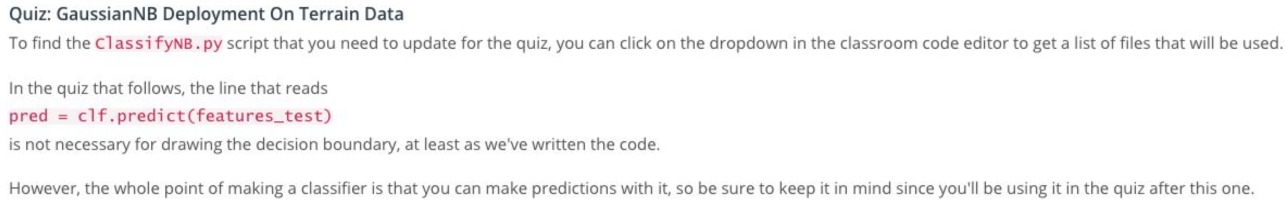
- Important question in ML: **What can you predict of a new data point you've never seen before, given the past data?**

Lesson 13-14

- **Decision Surface** separates one class from another class in a way that you can generalise to new never before seen data points.
- Machine learning algorithm takes in data and transforms it into a decision surface that for all future cases can enable you to make a determination of whether it's a red cross or blue cross.



Lesson 19



Quiz: GaussianNB Deployment On Terrain Data

To find the `ClassifyNB.py` script that you need to update for the quiz, you can click on the dropdown in the classroom code editor to get a list of files that will be used.

In the quiz that follows, the line that reads

```
pred = clf.predict(features_test)
```

is not necessary for drawing the decision boundary, at least as we've written the code.

However, the whole point of making a classifier is that you can make predictions with it, so be sure to keep it in mind since you'll be using it in the quiz after this one.

Lesson 20

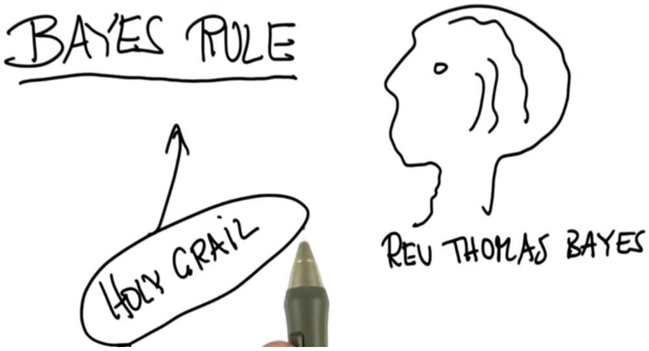
- **Accuracy** is defined as the number of test points that are classified correctly divided by the total number of test points.

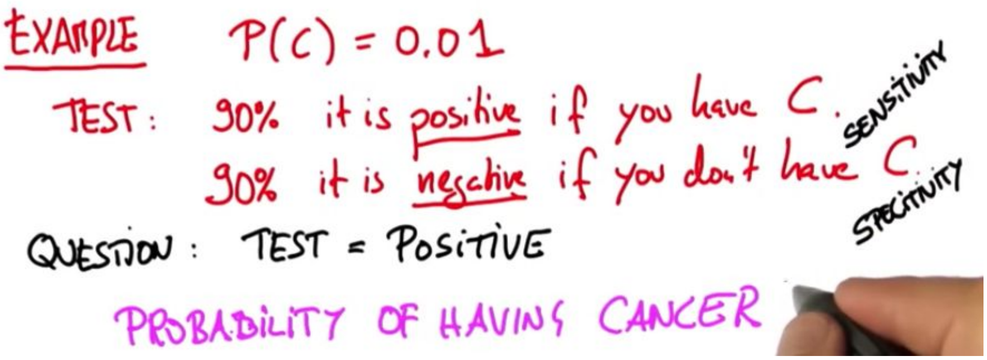
Lesson 21

- **In ML, you train and test on 2 different sets of data.**
  - If you don't do this, you overfit to your trained data therefore it won't be able to generalise to new data.
  - **Save 10% of your data and use it as your testing set.**

Lesson 23

- **Naive Bayes** is the holy grail of **Probabilistic Inference.**





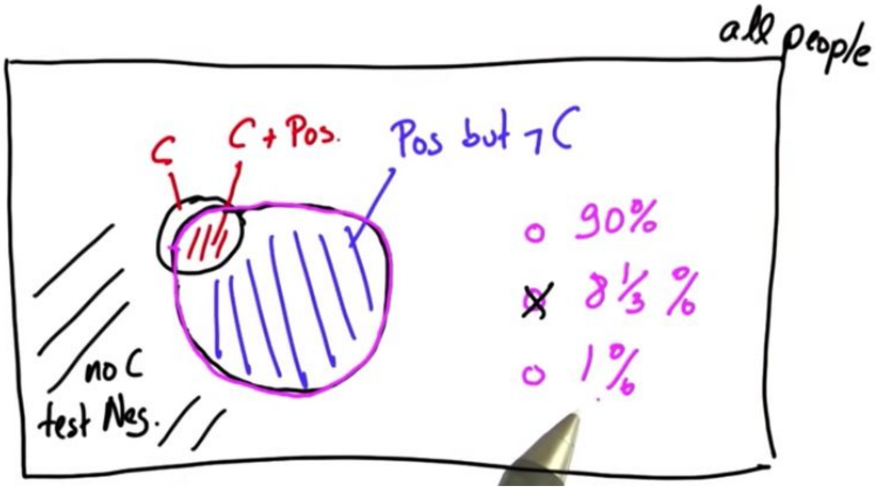
EXAMPLE $P(C) = 0.01$

TEST: 90% it is positive if you have C. SENSITIVITY

90% it is negative if you don't have C. SPECITIVITY

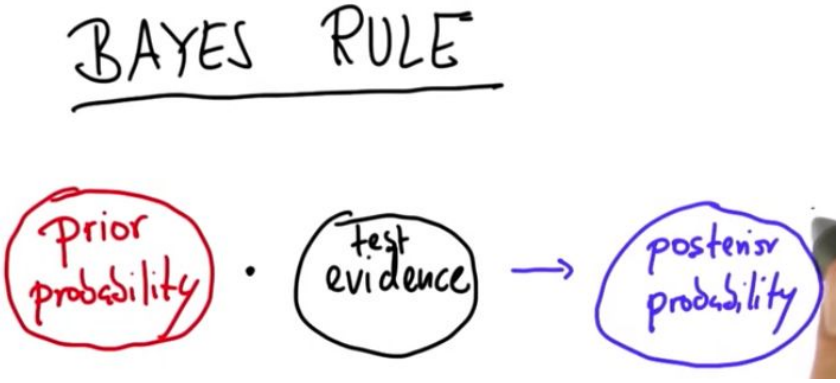
QUESTION: TEST = POSITIVE

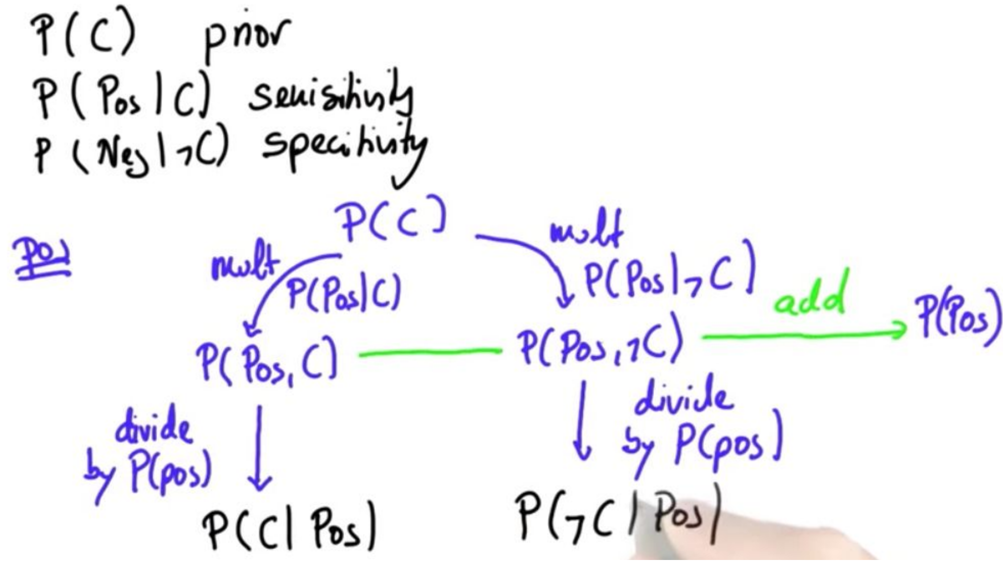
PROBABILITY OF HAVING CANCER

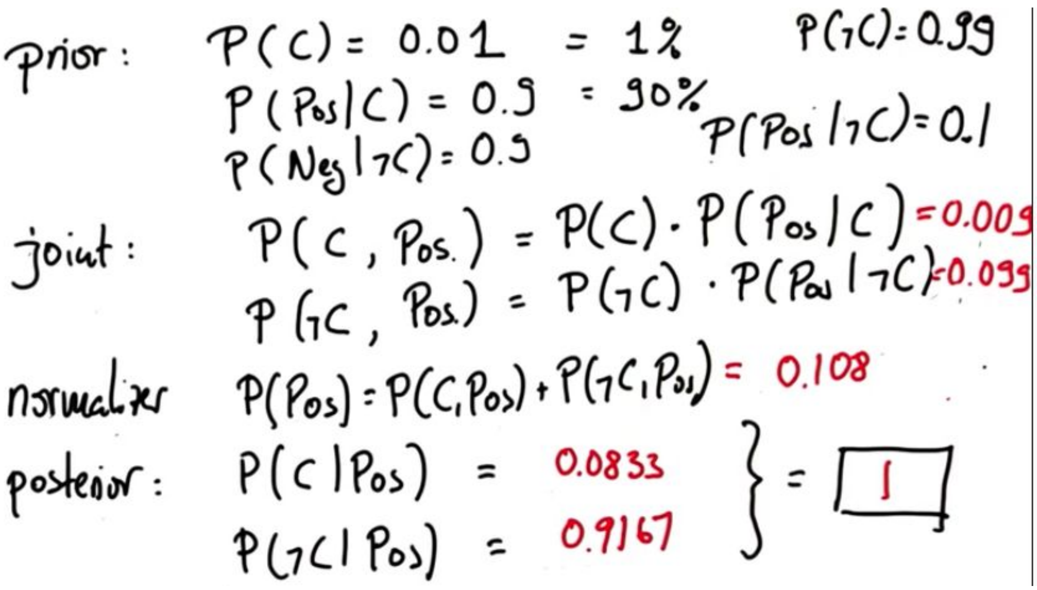


Lesson 24-29

- **Bayes Rule** incorporates some evidence from a **test** into your **prior probability** to arrive at a **posterior probability.**







prior: $P(C) = 0.01 = 1\%$ $P(\neg C) = 0.99$

$P(Pos|C) = 0.9 = 90\%$ $P(Pos|\neg C) = 0.1$

$P(Neg|\neg C) = 0.9$

joint: $P(C, Pos.) = P(C) \cdot P(Pos|C) = 0.009$

$P(\neg C, Pos.) = P(\neg C) \cdot P(Pos|\neg C) = 0.099$

normalizer $P(Pos) = P(C, Pos) + P(\neg C, Pos) = 0.108$

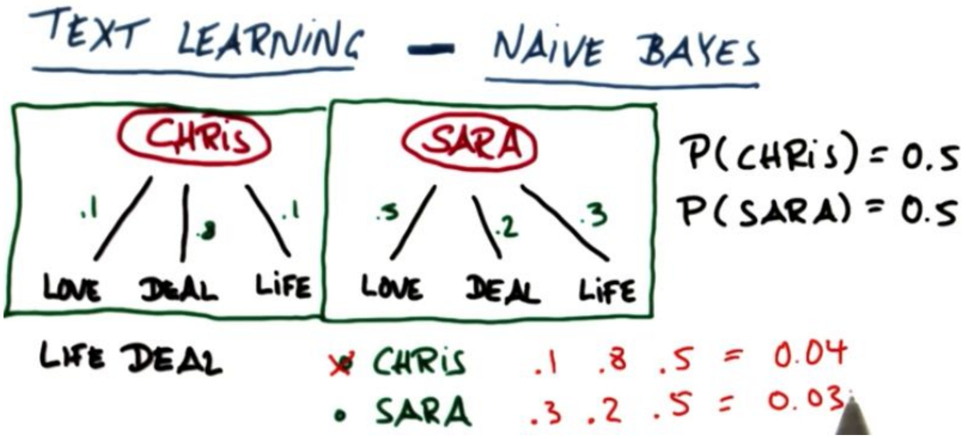
posterior: $P(C|Pos) = 0.0833$

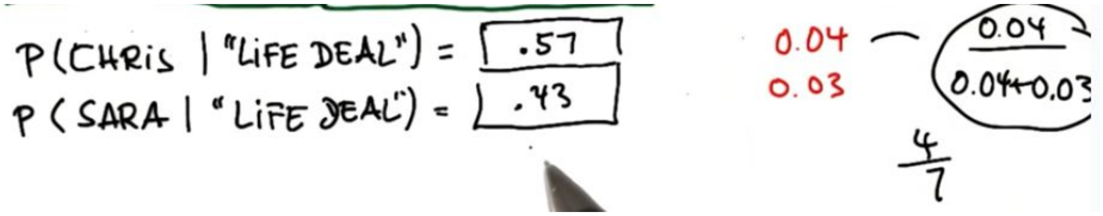
$P(\neg C|Pos) = 0.9167$

$\Big\} = \boxed{1}$

Lesson 30-33

- **Bayes rule** is often used for **text learning** (i.e. learning from documents).
- **Naive Bayes** allows you to determine based on a random document who the likely person who sent this document is.





$P(\text{CHRIS} \mid \text{"LIFE DEAL"}) = \boxed{.57}$

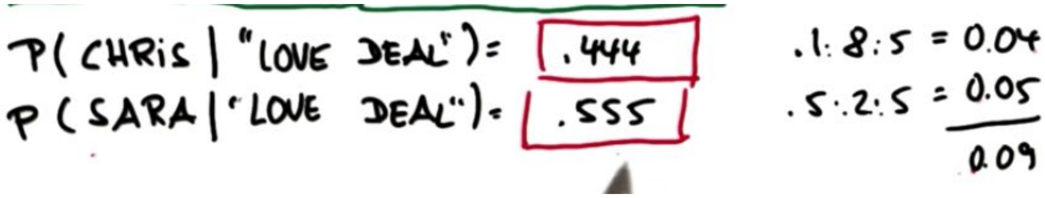
$P(\text{SARA} \mid \text{"LIFE DEAL"}) = \boxed{.43}$

0.04

0.03

$\frac{0.04}{0.04+0.03}$

$\frac{4}{7}$



$P(\text{CHRIS} \mid \text{"LOVE DEAL"}) = \boxed{.444}$

$P(\text{SARA} \mid \text{"LOVE DEAL"}) = \boxed{.555}$

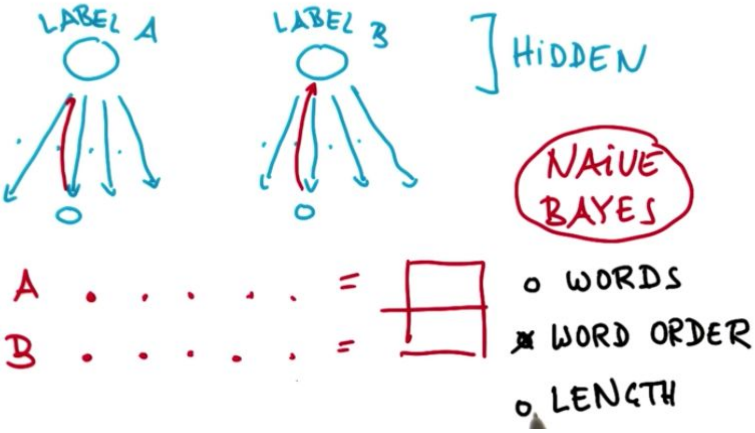
$.1 \cdot .8 \cdot .5 = 0.04$

$.5 \cdot .2 \cdot .5 = 0.05$

0.09

Lesson 34-35

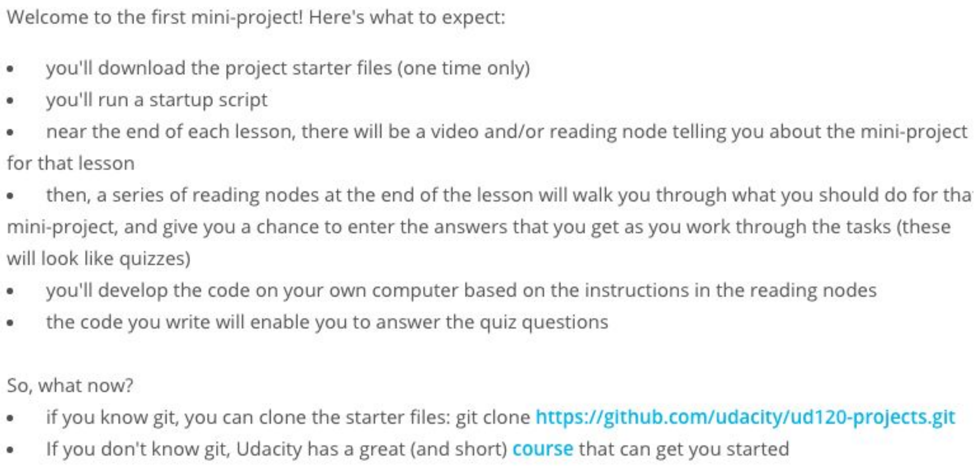
- **Naive Bayes** lets you identify from a text source which label is more likely for that source by giving you a ratio. You can do this for people, news sources, song e.g. was text written by Shakespeare.
    - It does this by looking at the different probabilities of each word, i.e. gives you evidence for whether is was person A or B.
    - Multiplies the evidences for every word you see, the product.
    - Then uses these large products in a ratio to predict which label is correct.
- **Naive** because the product doesn't consider the order in which the words occur.
    - It doesn't understand text, **just looks at word frequencies to do classification!**



- **Naive Bayes:**
    - **Advantages:**
        - Really easy to implement
        - Deals well with large features spaces
        - Simple to run and efficient
    - **Disadvantages:**
        - Phrases that encompass multiple works and have distinct meanings don't work really well in Naive Bayes.

Mini Project

Lesson 39



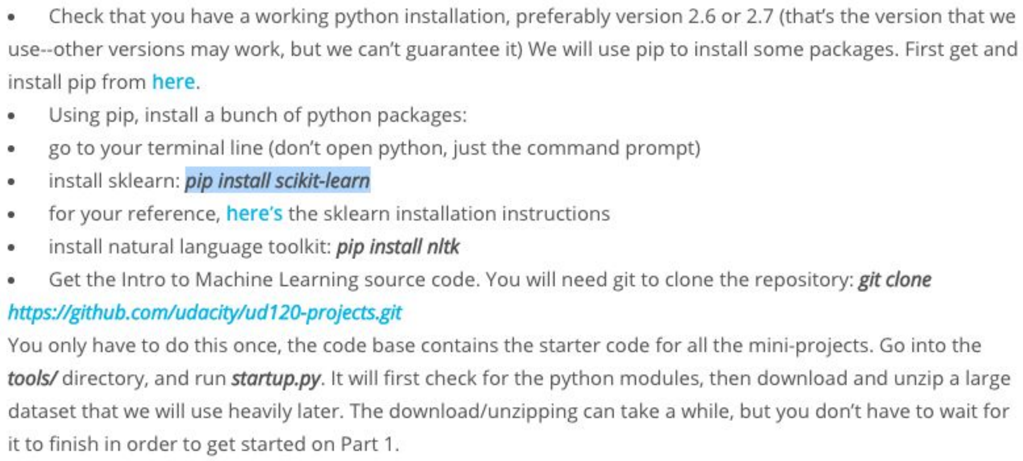
Welcome to the first mini-project! Here's what to expect:

- you'll download the project starter files (one time only)
- you'll run a startup script
- near the end of each lesson, there will be a video and/or reading node telling you about the mini-project for that lesson
- then, a series of reading nodes at the end of the lesson will walk you through what you should do for tha mini-project, and give you a chance to enter the answers that you get as you work through the tasks (these will look like quizzes)
- you'll develop the code on your own computer based on the instructions in the reading nodes
- the code you write will enable you to answer the quiz questions

So, what now?

- if you know git, you can clone the starter files: git clone https://github.com/udacity/ud120-projects.git
- If you don't know git, Udacity has a great (and short) course that can get you started

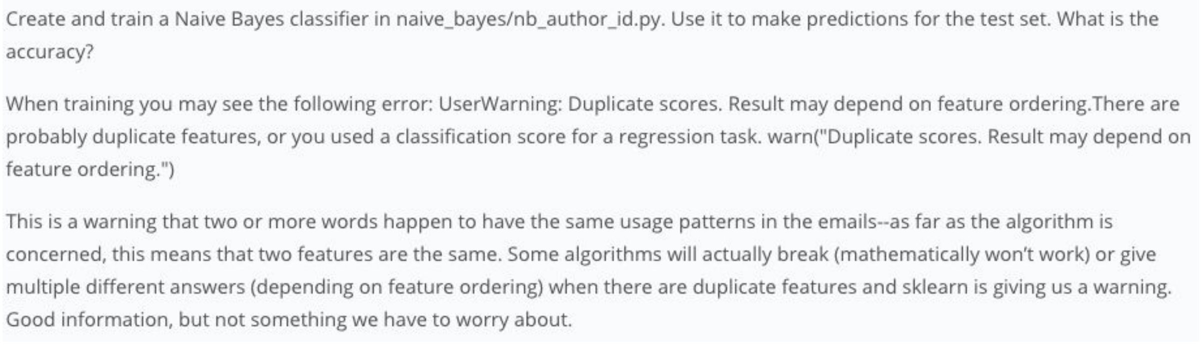
Lesson 41



- Check that you have a working python installation, preferably version 2.6 or 2.7 (that's the version that we use--other versions may work, but we can't guarantee it) We will use pip to install some packages. First get and install pip from here.
- Using pip, install a bunch of python packages:
- go to your terminal line (don't open python, just the command prompt)
- install sklearn: ***pip install scikit-learn***
- for your reference, here's the sklearn installation instructions
- install natural language toolkit: ***pip install nltk***
- Get the Intro to Machine Learning source code. You will need git to clone the repository: ***git clone*** *https://github.com/udacity/ud120-projects.git*

You only have to do this once, the code base contains the starter code for all the mini-projects. Go into the ***tools/*** directory, and run ***startup.py***. It will first check for the python modules, then download and unzip a large dataset that we will use heavily later. The download/unzipping can take a while, but you don't have to wait for it to finish in order to get started on Part 1.

Lesson 42



Create and train a Naive Bayes classifier in naive_bayes/nb_author_id.py. Use it to make predictions for the test set. What is the accuracy?

When training you may see the following error: UserWarning: Duplicate scores. Result may depend on feature ordering.There are probably duplicate features, or you used a classification score for a regression task. warn("Duplicate scores. Result may depend on feature ordering.")

This is a warning that two or more words happen to have the same usage patterns in the emails--as far as the algorithm is concerned, this means that two features are the same. Some algorithms will actually break (mathematically won't work) or give multiple different answers (depending on feature ordering) when there are duplicate features and sklearn is giving us a warning. Good information, but not something we have to worry about.