Lab -- Features; Train / Test

Instructions

1. If you haven't already, please create a group using the box below.
2. Work through the lab assignment with your group, write your answers on paper, and **ask questions while you're working! Put yourself in the help queue to do so**.
3. When you're finished, put yourself in the queue for a checkoff. **Only 1 person in the group needs to request the checkoff** (creating the group adds you all to the system together!)
4. You should try to finish and get your checkoff done before the lab ends. If you don't finish, you can get the checkoff during office hours. The lab checkoff is due as specified on the top of the page.
5. After your checkoff, you're welcome to leave or stay to work on the homework.

Group information

Create Your Group

Once you and your partner(s) are in a breakout room, you need to create a group.

Instructions:

(1) **One** of you should click here to create a group.

(2) Everyone else should then enter the given new group name (including trailing numbers) into the box below (and press enter when done).
Name of group to join:

Reload this page anytime to refresh your group status (and to get a delete button, if you want to remove yourself from a group you've joined).

Groups are limited to 3 persons max each, so that checkoff discussions can be inclusive.

You are not currently in any group

Lab 2

Welcome to Lab 2, where will will discuss the evaluation of learning algorithms, and explore how to represent input features for the model. We intend for you to begin thinking about these concepts, but **please do not stress if you don't understand everything perfectly by the end.** Use the lab session to build your understanding and clear up confusion, and explore the different ways in which we can represent the features we input to the model. Come to office hours if you want to discuss lab topics further. The staff are here to help you learn!

1) Evaluating a learning algorithm

Note the following notation and definitions, used in the problem:

- A generator $\mathcal{G}$ is a function that draws a random subset from a very large data set.
- A training dataset $\mathcal{D}_{\text{train}}$ is a set of labeled samples $\mathcal{X}, y$ generated by a generator $\mathcal{G}$, where $x^{(i)}$ represents the features of an object to be classified (vector of real and/or discrete values), and $y^{(i)}$ represents the label of $x^{(i)}$. (You can think of i as the index for point $x^{(i)}$.)
- A binary classifier h is a function that takes a data point x (a $d \times 1$ vector) as input and returns $+1$ or -1 as output.

In last week's lab, we explored the question of what makes a good hypothesis and the challenges involved with evaluating a hypothesis. This week, we explore how to evaluate the *learning algorithms*, which are used to generate hypotheses.

A learning algorithm is a function L that takes as input the data set $\mathcal{D}_{\text{train}}$ as training data and returns a hypothesis (in this particular question, our hypothesis is a *classifier*) h.

1.1)

What is the difference between a classifier and a learning algorithm? (Stuck? Check the notes on Linear Classifiers).

1.2)

Let $\mathcal{D}_{\text{train}_1}$, $\mathcal{D}_{\text{train}_2}$ be different training datasets generated by $\mathcal{G}_1$, $\mathcal{G}_2$, respectively. Would running the learning algorithm L on $\mathcal{D}_{\text{train}_1}$ and $\mathcal{D}_{\text{train}_2}$ produce the same classifier? In other words, would $h_1 = L(\mathcal{D}_{\text{train}_1})$ be the same classifier as $h_2 = L(\mathcal{D}_{\text{train}_2})$?

What if $\mathcal{D}_{\text{train}_1}$, $\mathcal{D}_{\text{train}_2}$ were generated by the same generator?

1.3)

Now, consider a situation in which someone is trying to sell you a new learning algorithm, and you want to know how good it is. There is an interesting theorem called the No Free Lunch Theorem, that says that without any assumptions about your data there is no learning algorithm that, for all data sources, is better than every other learning algorithm. So, you'll need to assess the learning algorithm's performance in the context of a particular data source.

Assume that you have a generator of labeled data, $\mathcal{G}$, which will be suitable for your application. The learning algorithm's performance on $\mathcal{G}$-generated data will be a good predictor of the learning algorithm's performance on data from your application. (You can review how to evaluate learning algorithms in the notes on Linear Classifiers.

Linnea Separatoria wants to evaluate a **learning algorithm**, and suggests the following procedure:

```
def eval_learning_alg(L, G, n):
    # draw a set of n training data points (points and labels)
    train_X, train_y = G(n)

    # run L
    h = L(train_X, train_y)

    # evaluate using your classifier scoring procedure, on some new labeled data
    test_data = G(n) # draw new set of test data

    return eval_classifier(h, test_data)
```

How well or not well does Linnea's strategy evaluate the learning algorithm?

1.4)

Next, Linnea decides to generate one classifier h but evaluate that classifier with multiple (10) test sets in her `eval_learning_alg`. More specifically, Linnea changed her code above into:

```
def eval_learning_alg(L, G, n):
    # draw a set of n training data points (points and labels)
    train_X, train_y = G(n)

    # run L
    h = L(train_X, train_y)

    # evaluate using your classifier scoring procedure, on some new labeled data
    score = 0
    for i in range(10):
        test_data = G(n) # draw new set of test data
        score += eval_classifier(h, test_data)

    return score/10
```

Is Linnea's strategy better now? Explain why or why not.

Show Hint

1.5)

Now design a better procedure, `better_eval_learning_alg` for evaluating L, that takes L, $\mathcal{G}$ and n and returns a score. What would the output score measure? What are the best and worst score values? Why is your method more desirable than Linnea's?

1.6)

In reality, it's almost never possible to have a generator of all the data you want. In fact, in some domains, data is very expensive to collect, and so, you are given a fixed, small set of samples. Now assume that you only have 100 labeled data points to use for training and testing/evaluation.

How would you implement `better_eval_learning_alg` without $\mathcal{G}$ but instead with your 100 labeled data?

Check and submit this box once you've finished this section:

☐ I've finished this section.

Save | Submit | View Answer | Ask for Help

As staff, you are always allowed to submit. If you were a student, you would see the following:

You have infinitely many submissions remaining.

2) Features Engineering

The performance of our hypotheses on the real world depends heavily on how we represent our data. In the next few questions we will introduce you to several ways of representing data (from standard numerical data to text data and image data!).

Please make sure you read the notes on Feature Representation for this lab.

2.1) Feature engineering for car data

Open and view `auto-mpg.tsv`. You can also download a local version through the google sheets link or **here**. This file is in a common format, called "tab separated values". In the first line you will find the names of the columns. Each row is a data point; **the first column is the label for that point (1 or -1).**

The original data came from the StatLib library from CMU. It was modified by Ross Quinlan to remove entries with unknown mpg (miles per gallon). We have modified it further by removing six entries with unknown horsepower. We have also binarized the mpg column to turn this into a classification problem (later in the term, we will look at predicting continuous values, i.e. regression problems). Here are the nine columns in the dataset:

```
1. mpg:           continuous (modified by us to be -1 for mpg < 23, 1 for others)
2. cylinders:     multi-valued discrete
3. displacement:  continuous
4. horsepower:    continuous
5. weight:        continuous
6. acceleration:  continuous
7. model year:    multi-valued discrete
8. origin:        multi-valued discrete
9. car name:      string (many values)
```

There are 392 entries in the dataset, evenly split between positive and negative labels. The field names should be self-explanatory except possibly `displacement` and `origin`. You can read about `displacement` here; in this data set `displacement` is in cubic inches. `origin` is 1=USA, 2=Europe, 3=Asia. We'll ignore `car name` in this assignment.

A new student, Hiper Playne, suggests that we should just use all the numeric features in their raw form to predict whether the car will get good or bad gas mileage. He will use the Perceptron algorithm to learn the classifier. Once he trains the model on this dataset, he wants to predict whether cars in 2021 will get good gas mileage.

2.1.1)

Which of the following is a problem or may lead to problems when using the raw features directly?

- [ ] Because `weight` values and `cylinders` values are on different scales, perceptron might take many iterations
- [ ] `cylinders` is a multi-valued discrete number feature
- [ ] `origin` is a multi-valued discrete number feature
- [ ] There are too many features and the classifier will overfit
- [ ] `model year` is in the 70s, so a classifier based on this data might not perform well in 2021

Save | Submit | View Answer | Ask for Help

As staff, you are always allowed to submit. If you were a student, you would see the following:
You have infinitely many submissions remaining.

2.1.2)

For each feature from the following:
`[cylinders, displacement, horsepower, weight, acceleration, model_year, origin]`
indicate how you can represent it so as to make classification easier and get good generalization on unseen data, by choosing one of:

- `'drop'` - leave the feature out,
- `'raw'` - use values as they are,
- `'standard'` - standardize values by subtracting out average value and dividing by standard deviation. Check the notes on Feature Representation (Section: Hand-constructing features for real domains) for more details
- `'one-hot'` - use a one-hot encoding.

There could be multiple answers that make sense for each feature; please mention the tradeoffs between each answer. Be prepared to justify your choices (ex. why standardize instead of using raw features? why use one-hot encoding?)

Checkoff 1:
Have a check-off conversation with a staff member, to explain your answers.

Ask for Help | Ask for Checkoff

2.2) Feature engineering for food reviews (text data)

In this part of the assignment, we are going to explore ways of defining features for textual data.

Open and view `reviews.tsv`. This file is in "tab separated values" (tsv) format, and it consists of 10,000 fine food reviews from Amazon. You can find additional information here. We will be focusing on the `text` field and use it to predict the `sentiment` field (-1 or 1).

We will convert review texts into fixed-length feature vectors using a bag-of-words (BOW) approach. We start by compiling all the words that appear in a **training** set of reviews into a list of d *unique* words.

2.2.1)

For instance, consider two simple documents "Mary is selling apples." and "Tom is buying apples to eat". If we had only these two sentences in our training set of reviews, which option corresponds to the list of unique words that we build when using the bag-of-words approach discussed above?

- ○ $(Mary,\ is,\ selling,\ apples,\ Tom,\ is,\ buying,\ apples,\ to,\ eat)$
- ○ $(Mary,\ is,\ selling,\ apples,\ Tom,\ buying,\ to,\ eat)$
- ○ $(Mary,\ is,\ selling,\ red,\ apples,\ Tom,\ buying,\ blue,\ to,\ eat)$

Save | Submit | View Answer | Ask for Help

As staff, you are always allowed to submit. If you were a student, you would see the following:
You have infinitely many submissions remaining.

2.2.2)

We can now transform each of the reviews into a feature vector of length d by setting the i^{th} coordinate of the feature vector to 1 if the i^{th} word in the list of unique words appears in the review or 0 if it does not. Hence, how would we represent the sentence "Tom is buying apples to eat" as a feature vector using the bag-of-words approach and the list of unique words we found above?

- ○ $(1, 1, 0, 0, 1, 1, 1, 1)$
- ○ $(0, 1, 0, 1, 1, 0, 1, 1)$
- ○ $(0, 1, 0, 1, 1, 1, 1, 1)$

Save | Submit | View Answer | Ask for Help

As staff, you are always allowed to submit. If you were a student, you would see the following:
You have infinitely many submissions remaining.

2.2.3)

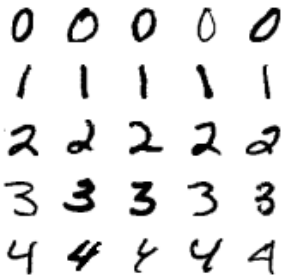
Talk with your group about the weaknesses of the bag-of-words approach seen above. For instance, how would you interpret the feature vector $(1, 1, 1, 1, 1, 0, 1, 0)$?

2.2.4)

Words like "the", "to", "a", "and" and so on occur with very high frequency in English sentences. Do they help in detecting the sentiment in a review? Talk to your group about how to deal with these words, when building your list of unique words, using the bag-of-words approach.

2.3) Image features

We will be exploring in the homework how the perceptron algorithm might be applied to classify images of handwritten digits, like those from a well-known ("MNIST") dataset, with items like these:



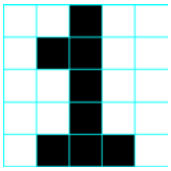
For today, assume we only have images of digits '1' and '3' and we would like to find a linear classifier which can classify these images correctly, giving label $+1$ for the digit '1' and label -1 for the digit '3'.

For simplicity, assume each image provided is a 5-pixel wide and 5-pixel tall (5x5) black and white image.

2.3.1)

Assume images from the MNIST dataset are provided to you as 5x5 matrices. An image is represented as a matrix, say A, with the entry of the i^{th} row and j^{th} column ($A_{i,j}$) representing the image pixel found at the i^{th} row and j^{th} column of pixels. Black pixels have value 0 while white pixels have value 1 in the matrix.

With the help of your group, write down the matrix corresponding to the image shown here:



2.3.2)

Imagine we want to use the perceptron algorithm to perform the task of distinguishing between images of digits '1' and '3'. Each image is a data point, which we need in **vector** form, to feed to the perceptron algorithm. One approach is to take the two-dimensional matrix representation of an image and concatenate the rows one after the other, into a one-dimensional vector of the form $[A_{1,1}, A_{1,2}, A_{1,3}, A_{1,4}, A_{1,5}, A_{2,1}, A_{2,2}, ..., A_{5,5}]$

For the image shown above, what will be the last 3 entries of the one-dimensional vector representing that image? Reminder that 0 represents black and 1 represents white. Enter a python list of length 3 for your answer:

Save | Submit | View Answer | Ask for Help

As staff, you are always allowed to submit. If you were a student, you would see the following:
You have infinitely many submissions remaining.

2.3.3)

How well would you expect the perceptron algorithm to work on classifying images if you simply represented them as 1D vectors (if you do not reliably know the width and length of the image)?

2.3.4)

What approaches might you suggest to extract more meaningful features from the images, such that the perceptron algorithm might do a good job of classification? (Hint: What makes the image of the digit '1' different compared to the image of the digit '3'?)

Discuss these questions with your group, and write down your answers. Be prepared to discuss these with the staff for your lab checkoff.

In the homework for this week we will investigate these datasets in more detail. And later in the semester, we'll explore other algorithms, including neural networks and convolutional neural networks, which can essentially do feature extraction on their own.

3) Fairness in ML

Last week, Feirna Sinamel introduced the problem of using machine learning to predict whether or not to give someone a loan. We saw how even the seemingly simple decision of choosing input features has several ethical and societal implications (and having done this lab, you can probably imagine that the way we represent features also has ethical implications).

After we have thoroughly thought through the implications of certain technical decisions (like, choosing features and their representation), we will need some way of evaluating the impact of our choices (presumably based on the performance of the model).

So this raises the question -- how do we evaluate the fairness of a model?

To get started thinking about this, let's take a look at some past data that Feirna gave you, and for the sake of concreteness, let's focus specifically on "fairness" across genders.

You notice that only 30% of the applicants for loan applications come from women, and of the applications that get approved, 10% are from women. So, you ask Feirna how gender has been used in the application assessment process in the past. She finds that though applications do contain information on gender, the application readers cannot view these fields. This brings up the interesting question: What would a "fair" outcome by an application screening model look like when it comes to the gender distribution? More generally, what does it mean for a model to be "fair"?

LIT Company's Definition of Fairness (Group Unaware): The company believes that a fair process and, therefore, a fair model, would not account for gender or race at all.

Advocacy Group's Definition (Demographic parity): An advocacy group believes that a model is fair if the distribution of outcomes for each demographic, gender, or other subgroup is the same among those that applied and those that were accepted. For example, in the example above, 30% of the applicants for loan applications come from women. In the demographic parity definition of fairness, this means 30% of the approved loan applications should come from women.

Note: The list of definitions of fairness mentioned above is not exhaustive (scroll down for some more examples). Note also that questions about a model's "fairness" are not strictly questions about statistical distribution of the model's outputs. Questions of fairness may also concern assumptions about the model's data. This also brings up the question of whether the application of machine learning itself is discriminatory or socially unjust (See here for some interesting reading on this).

3.1)

What are the merits and drawbacks of these particular definitions. What situations do those definitions account for? What do they not account for?

3.2)

How might we implement and evaluate these two fairness standards?

3.3)

Now of course, gender discrimination is not the only form of discrimination, and one could argue that race-based discrimination is far more prevalent in loan approval processes. In general, as machine learning engineers, how can we make sure fairness (across all different demographic groups) is considered when we formulate and test hypotheses?

While you're waiting for the checkoff, some further food for thought...
If you want to learn more about ML fairness in loan applications, check out this paper (Black Loans Matter) that discusses this exact problem! If you're interested in more definitions of fairness in ML, here is an article from Google about 5 common definitions (including the two discussed above).

Finally many researchers have identified major ethical limitations in attempts to formalize fairness in ML, arguing that those attempts overlook deeper patterns of inequality and injustice. See this article in The Lancet for limitations of algorithmic fairness in healthcare. For longer technical articles offering different positions on this debate, see here and here.

Checkoff 2:
Have a check-off conversation with a staff member, to explain your answers.

Ask for Help | Ask for Checkoff

RES.TLL-008 Social and Ethical Responsibilities of Computing (SERC)

Fall 2021