Introduction to Machine Learning
(Fall 2022)

Lab attendance check

Type in your section passcode to get attendance credit (within the first fifteen minutes of your scheduled section).
Passcode: Enter

Instructions

In 6.390 a large part of the learning happens by discussing lab questions with partners. Please complete this group self-partnering question then begin the lab.

Group information

Create/join a group

1. One person (and only one) should create a group.
2. Everyone else should enter the group name below (in the form `groupname_0000`).

   Join group: Submit

To join another group or leave your current group, reload the page.

You are not in a group.

Please refer to the course notes on Regression for definitions and explanations of basic concepts.

Lab 2

1) Warm up

As a warm-up, discuss the following questions with your lab partner(s) and be ready to share your reasoning with your instructor:

1.1)

What is the difference between a learning algorithm and a hypothesis? Write down one possible hypothesis for a linear regression problem in which the input dimension is $d = 3$. Name two learning algorithms that you know for linear regression.

1.2)

We often use *squared error* as a loss function in regression. Can you think of a situation in which that might not be a good idea or other loss functions would be better? (What would be a good loss function if you were trying to throw a ball of radius r into a circular hole of radius R?)

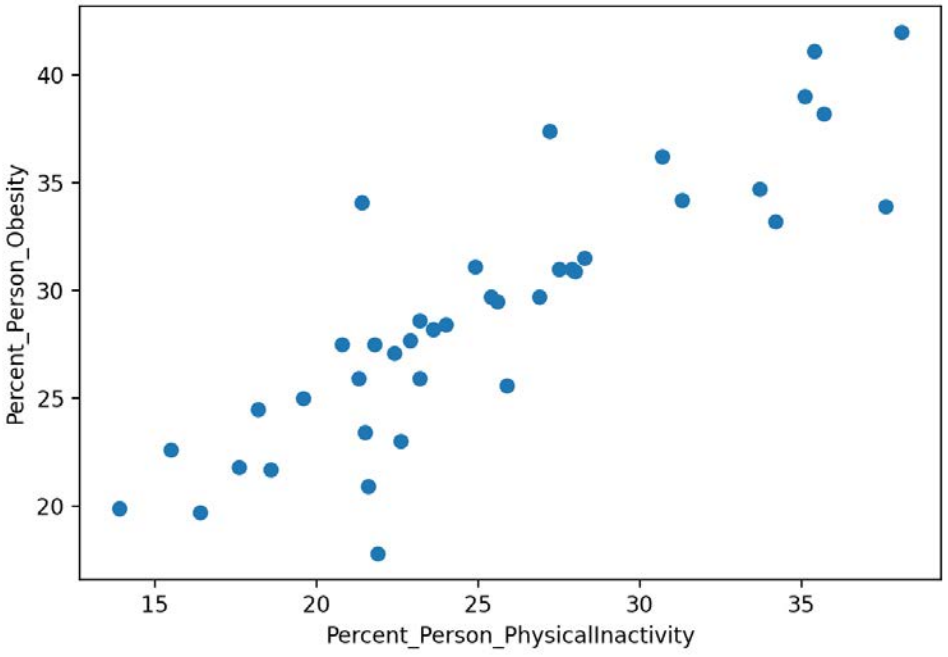
2) Least-squares regression on a real data set

In this problem we will use linear regression to study the relationship between variables in a real public health dataset. Each data-point represents a U.S. city, and it is characterized by a number of features characterizing aspects of the health of its population, each of which constitutes a dimension of the input x associated with that city. In this exercise, we will try to predict the attribute `Percent_Person_Obesity` based on the following other attributes:

- `Count_Person`
- `Median_Income_Person` (k$)
- `Percent_NoHealthInsurance` (%)
- `Percent_Person_PhysicalInactivity` (%)
- `Percent_Person_SleepLessThan7Hours` (%)
- `Commute_Time` (min)
- `Percent_Person_WithHighBloodPressure` (%)
- `Percent_Person_WithMentalHealthNotGood` (%)
- `Percent_Person_WithHighCholesterol` (%)

The data we will be using contain the above information from 500 U.S. cities and is acquired from Data Commons, a free API combining publicly available data from open sources.

As a simple illustration of the dataset, see below a plot of `Percent_Person_Obesity` vs `Percent_Person_PhysicalInactivity` using data from 50 random cities. (Does it look linear? What do you think linear regression will do on the data in this plot?)



We first separate the original 500 cities dataset into three training datasets

- `Train_small` contains data from 10 large cities (SF, NYC, Atlanta, etc.)
- `Train_big` contains data from 200 cities
- `Train_tiny` contains data from 5 cities

and one test dataset `Test_data` of 50 cities (different from those in the training sets).

Because these attributes are described in very different units (percentages, thousands of dollars, minutes, etc.) learning will work best if we pre-process them so they are all roughly in the same range. (We'll study this process more in a few weeks.) Below, we will be using a black-box function that does linear regression on the processed data, then "unprocesses" the data and computes the error as well as plots the learned regression fits alongside the original data. The processed datasets, together with the normalization coefficients used for scaling, are viewable at this online spreadsheet.

Our goal is to learn a linear regression model using the training data such that it can make good predictions on the cities in `Test_data`.

In ordinary least squares regression problems, we assume that our objective function $J(\theta, \theta_0)$ comes without a regularization term and only has a mean square error loss. In other words,

$$J(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(g^{(i)}, a^{(i)}) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_s(h(x^{(i)}; \theta, \theta_0), y^{(i)}) \ ,$$

where $g^{(i)}$ is the prediction made by the hypothesis, and $a^{(i)}$ is the actual sample output value. For our ordinary least squares case, $\mathcal{L}$ is the squared loss $\mathcal{L}_s$, where we have made explicit that the hypothesis depends on both input data $x^{(i)}$, and model parameters θ and θ_0. Recall that it is possible to solve an ordinary least-squares regression problem directly via the matrix algebra expression for the optimal parameter vector θ in terms of the input data X and desired output vector Y:

$$\theta = (X X^T)^{-1} \underbrace{X}_{(d+1)\times n} Y^T.$$

Note that above we have already added a row of all ones into X and concatenated θ and θ_0 together, making the hypothesis $Y = \theta^T X$ (so that the analytic formula gives us both θ and θ_0!). We have written the expression for θ above in terms of our X and Y matrices where data-points correspond to columns; to make it match the notes and many other descriptions of regression (in which data points correspond to rows), you can use the $\tilde{X}, \tilde{Y}$ version of the solution, which just works with the transposes of X and Y. **In the rest of the lab, we will implicitly use this "augmented" version of X and θ, unless otherwise specified**.

In this section, we will explore the θ solutions that this *analytic* strategy produces. In the homework, we will actually implement this analytic strategy.

1-D Regression

We start with a 1-dimensional regression first, so we can visualize the data by a 2D plot.

2.1)

With the `Train_small` dataset and *exactly one* of the features above, what are the sizes of X, θ, and Y?

Reminder: For labs, all members of a group should enter their answers in the answer checkers in order to get **credit** for those questions.

Enter the size of X as a list of [number_of_rows, number_of_columns]:

Check Formatting | 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.

Enter the size of θ as a list of [number_of_rows, number_of_columns]:

Check Formatting | 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.

Enter the size of Y as a list of [number_of_rows, number_of_columns]:

Check Formatting | 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.

The function `test_analytic_regress` below takes in the training dataset name (e.g., `Train_small`) and a list of feature names (from those listed above), finds the parameters θ using the analytic regression formula, then evaluates the mean squared loss on `Test_data` and returns the test set loss as well as a plot of data alongside the learned predictor.

To answer the questions below, you should modify the arguments to `test_analytic_regress` as needed and then run the codebox by hitting `Submit`.

```
# train_data can be 'Train_small', 'Train_big', 'Train_tiny'
# features should be a list of strings. E.g ['Count_Person','Commute_Time']
def run():
    return test_analytic_regress(
        train_data = "Train_small",
        features = ["Count_Person"]
    )
```

Run Code | 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)

Run the codebox above, using the `Train_small` dataset and the `Count_Person` feature. Do you think the fit or linear hypothesis is reasonable or helpful for this case?

Using the `Train_small` dataset and *exactly one* of the features listed below, find the feature that gives you the lowest test set error.

- `Count_Person`
- `Percent_Person_WithHighBloodPressure`
- `Percent_NoHealthInsurance`
- `Median_Income_Person`
- `Commute_Time`

Does this result make sense to you?

2.3)

Now use the feature you found in the previous problem (the one with the lowest test error) and train on `Train_big`. What do you observe and why? You should use the codebox in the previous problem.

2-D Regression

Now let's make it more interesting with a 2-dimensional regression (so that the data is visualized by a 3D plot)!

2.4)

With the `Train_big` dataset and *exactly two* features, what are the sizes of X, θ, and Y?

Enter the size of X as a list of [number_of_rows, number_of_columns]:

Check Formatting | 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.

Enter the size of θ as a list of [number_of_rows, number_of_columns]:

Check Formatting | 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.

Enter the size of Y as a list of [number_of_rows, number_of_columns]:

Check Formatting | 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.

2.5)

Using the `Train_big` dataset and one of the feature pairs below, find the pair that gives you the lowest test set error. We've provided another codebox below for convenience; modify arguments as needed and hit the `Submit` button to run.

- `Count_Person` and `Percent_Person_SleepLessThan7Hours`
- `Percent_NoHealthInsurance` and `Percent_Person_PhysicalInactivity`
- `Percent_Person_SleepLessThan7Hours` and `Median_Income_Person`
- `Commute_Time` and `Percent_Person_WithHighBloodPressure`

Does the result agree with your intuition?

```
# train_data can be 'Train_small', 'Train_big', 'Train_tiny'
# features should be a list of strings. E.g ['Count_Person','Commute_Time']
def run():
    return test_analytic_regress(
        train_data = "Train_big",
        features = ["Count_Person",
                    "Percent_Person_SleepLessThan7Hours"]
    )

```

Run Code | 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.

2.6)

Compare the lowest test set errors you got when using two features versus when using one feature.

9-D Regression

Let's now use all of the nine features. Unfortunately we can't easily visualize a 10D space, so we'll only see mean square error as output when running the codebox below.

2.7)

Run the codebox below, where `test_analytic_regress` queries all nine features, and compare the test set losses when using `Train_small` vs. `Train_big`.

```python
#train_data can be 'Train_small', 'Train_big', 'Train_tiny'
# features is a string: 'all_features'. (Do not change this)
def run():
    return test_analytic_regress(
        train_data = "Train_small",
        features = 'all_features'
    )
```

Run Code | 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.

2.8)

In our example, the size of the training set is at most 200 samples. Do you see any problems with our analytic regression if you were to attempt regression with a very large training set size, e.g., as big as the population of the US (about 332 million)?

What if we had a very large training set (again, perhaps 332 million samples), but now with many features per sample (e.g., 2 million features per each individual)? Any problems with analytic regression in this case?

2.9)

In the other extreme, let's try running analytic regression on the `Train_tiny`, which contains five cities, and use `'all_features'`. You should use the previous codebox. What do you see when you run the code? Can you explain why that is the case?

Checkoff1:
Have a check-off conversation with a staff member, to explain your answers.

Ask for Help | Ask for Checkoff

3) A taste of regularization

Recall the form of the regularized *ridge regression* objective:

$$J_{ridge}(\theta, \theta_0) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_s(h(x^{(i)}; \theta, \theta_0), y^{(i)}) + \lambda\|\theta\|^2$$

3.1)

What happens to the learned regression line and what would you expect to happen to $J_{ridge}(\theta, \theta_0)$, with very large and very small (e.g., 0) values of λ?

3.2)

If our goal is to solely minimize $\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_s(h(x^{(i)}; \theta, \theta_0), y^{(i)})$ on the training dataset, what would be the best value of λ?

3.3)

What purpose does adding the regularization term serve?

3.4)

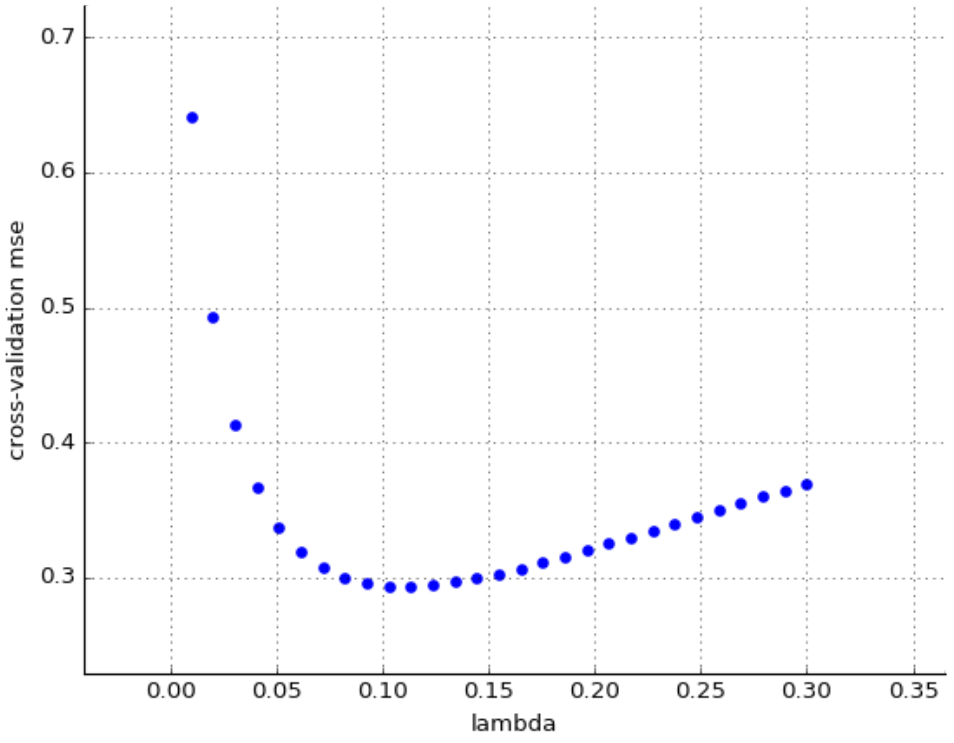
Why don't we regularize the offset θ_0 in ridge regression?

3.5)

Cross validation

Cross-validation is a method that lets us estimate the performance of a **learning algorithm** on a data set. (Not a hypothesis! Make sure you are clear on this point. Ask if not!)

You can think of changing a hyperparameter (like λ) as changing the algorithm. So, we can use cross-validation to evaluate a choice of λ. For example, shown below is a validation curve obtained by doing leave-one-out cross-validation on a simple dataset, where we vary λ from 0.01 to 0.3.



Why is this the shape of the graph? What is the best value of λ, in terms of generalization performance, based on this data? Is it the same as the best value for λ for performance on the training set, discussed in question 3.2 above?

In this week's homework, you will actually implement k-fold cross validation to select λ on the public health data!

4) Bands of Bands

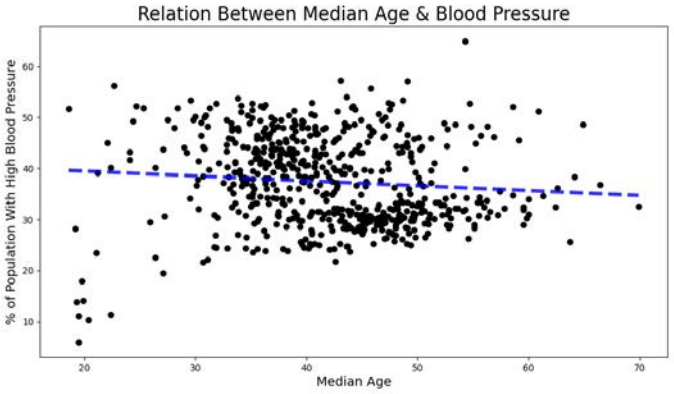
In many of our 6.390 labs, we will explore a variety of potential biases affecting machine learning. This week, we introduce *aggregation bias*.

Aggregation bias arises during model construction, when distinct populations are inappropriately combined. In many applications, the population of interest is heterogeneous (i.e., consisting of dissimilar sub-populations) and a single model is unlikely to suit all subgroups.

To illustrate this, we look at a particular phenomenon called Simpson's paradox, which arises due to a form of aggregation bias.

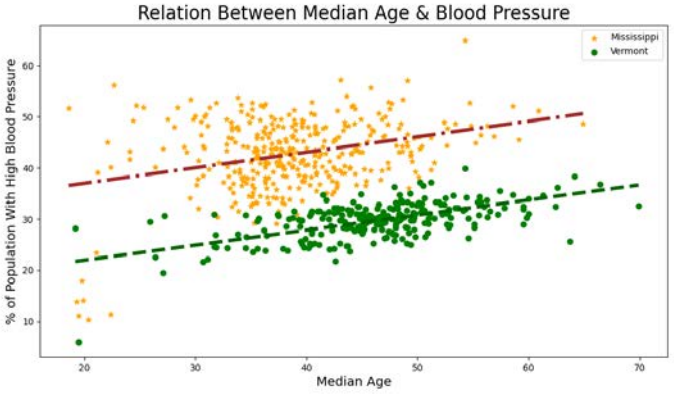
4.1)

You continue your foray into exploring Data Commons, and you make an interesting discovery about the relationship between age and blood pressure, as shown in the graph below! The blue line is produced from a linear regression on this data. What does this regression line indicate about the relationship between median age and high blood pressure?



4.2)

This data is drawn from two different states: Mississippi (orange stars) and Vermont (green circles). If we plot separate regression lines for these different groups as below, what does the graph tell us about the relationship between median age and high blood pressure? Why might aggregation of these two groups reverse the relationship between age and high blood pressure?



4.3)

Simpson's paradox is a phenomenon in which a trend appears in multiple groups of data but disappears or is reversed when data is aggregated. Can you think of a situation in which aggregating data could be harmful? Does this paradox mean we should not aggregate data, for risk of combining subgroups inappropriately? Can aggregating data ever be a good idea?

Checkoff2:
Have a check-off conversation with a staff member, to explain your answers.

Ask for Help | Ask for Checkoff

Food for thought:

Regression was first devised by Francis Galton, a Victorian scientist who is known as the "father of eugenics." Galton's discovery of regression was motivated by his interest in applying heredity to "improve the human race." He made the first regression line by plotting the diameter of sweet pea parental seeds against progeny seeds to examine whether exceptional physical traits were inheritable (see Figure). Galton subsequently introduced the idea of "regression towards mediocrity," in which he observed that extreme characteristics regress towards the mean of a distribution. Galton used these ideas to advocate for "selective breeding."

While Galton and other statisticians used their creations to advocate inhumane policies, this history does not invalidate the usefulness of regression. But, when you apply a statistical technique, consider: what are the limitations of this technique? How can your use of this technique create, perpetuate, or amplify societal harms?

RES.TLL-008 Social and Ethical Responsibilities of Computing

Spring 2023