CPSC 340 Assignment 3 (due Friday, Oct 12 at 11:55pm)

Instructions

Rubric: {mechanics:3}

The above points are allocated for following the general homework instructions on the course homepage.

1 Finding Similar Items

For this question we'll use the Amazon product data set[1] from `http://jmcauley.ucsd.edu/data/amazon/`. We will focus on the "Patio, Lawn, and Garden" section. You should start by downloading the ratings at `https://stanford.io/2Q7QTvu` and place the file in your `data` directory with the original filename. Once you do that, running `python main.py -q 1` should do the following steps:

- Load the raw ratings data set into a Pandas dataframe.
- Construct the user-product matrix as a sparse matrix (to be precise, a `scipy.sparse.csr_matrix`).
- Create bi-directional mappings from the user ID (e.g. "A2VNYWOPJ13AFP") to the integer index into the rows of `X`.
- Create bi-directional mappings from the item ID (e.g. "0981850006") to the integer index into the columns of `X`.

1.1 Exploratory data analysis

1.1.1 Most popular item

Rubric: {code:1}

Find the item with the most total stars. Submit the product name and the number of stars.

Note: once you find the ID of the item, you can look up the name by going to the url `https://www.amazon.com/dp/ITEM_ID`, where `ITEM_ID` is the ID of the item. For example, the URL for item ID "B00CFM0P7Y" is `https://www.amazon.com/dp/B00CFM0P7Y`.

- The product name is Classic Accessories 73942 Veranda Grill Cover - Durable BBQ Cover with Heavy-Duty Weather Resistant Fabric, X-Large, 70-Inch
- The number of stars is 14454.0

[1]The author of the data set has asked for the following citations: (1) Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. R. He, J. McAuley. WWW, 2016, and (2) Image-based recommendations on styles and substitutes. J. McAuley, C. Targett, J. Shi, A. van den Hengel. SIGIR, 2015.

1.1.2 User with most reviews

Rubric: {code:1}

Find the user who has rated the most items, and the number of items they rated.

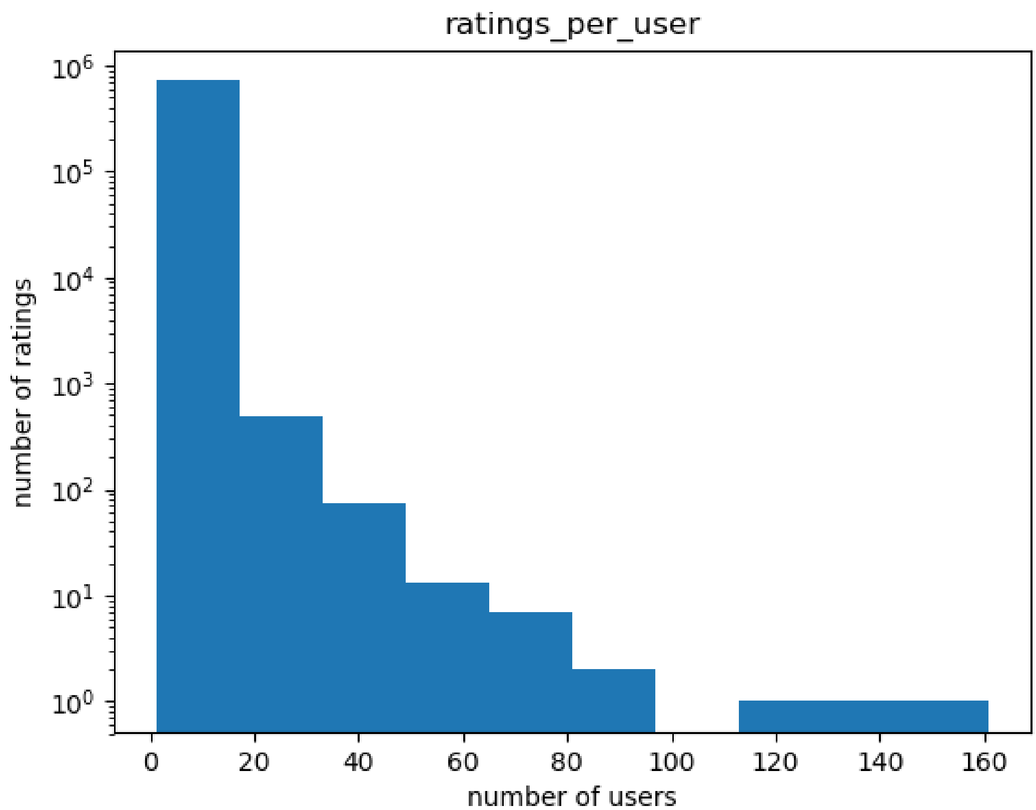
- number of ratings: 161
- user: A100WO06OQR8BQ

1.1.3 Histograms

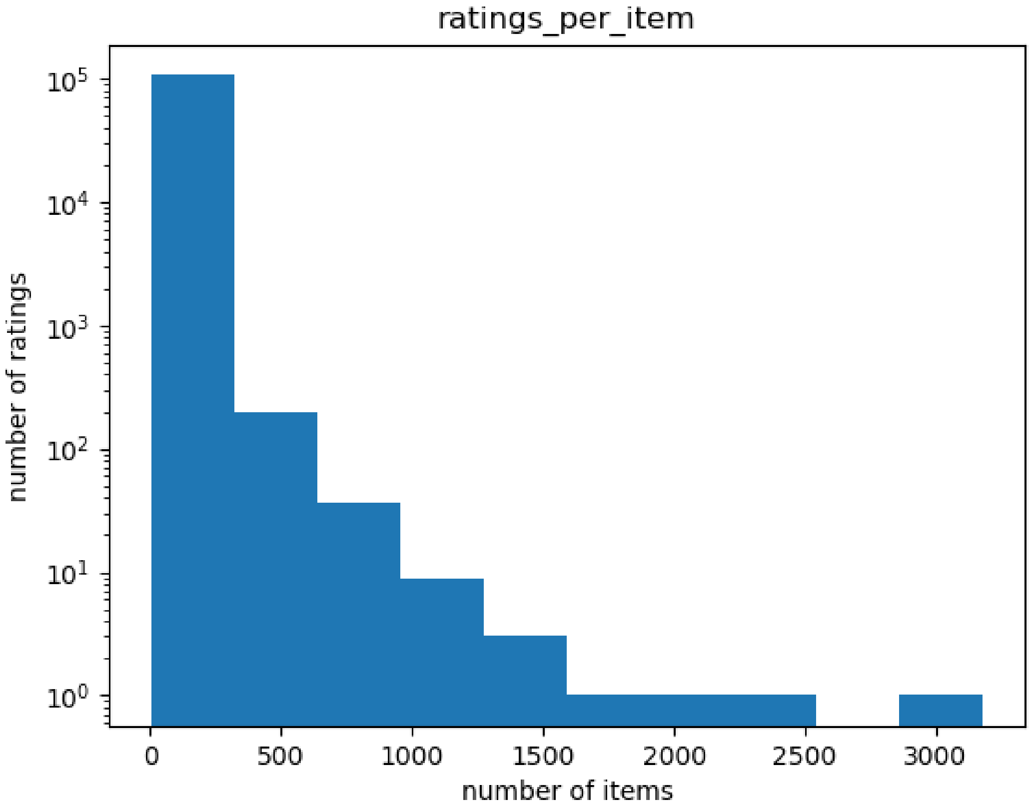
Rubric: {code:2}

Make the following histograms:

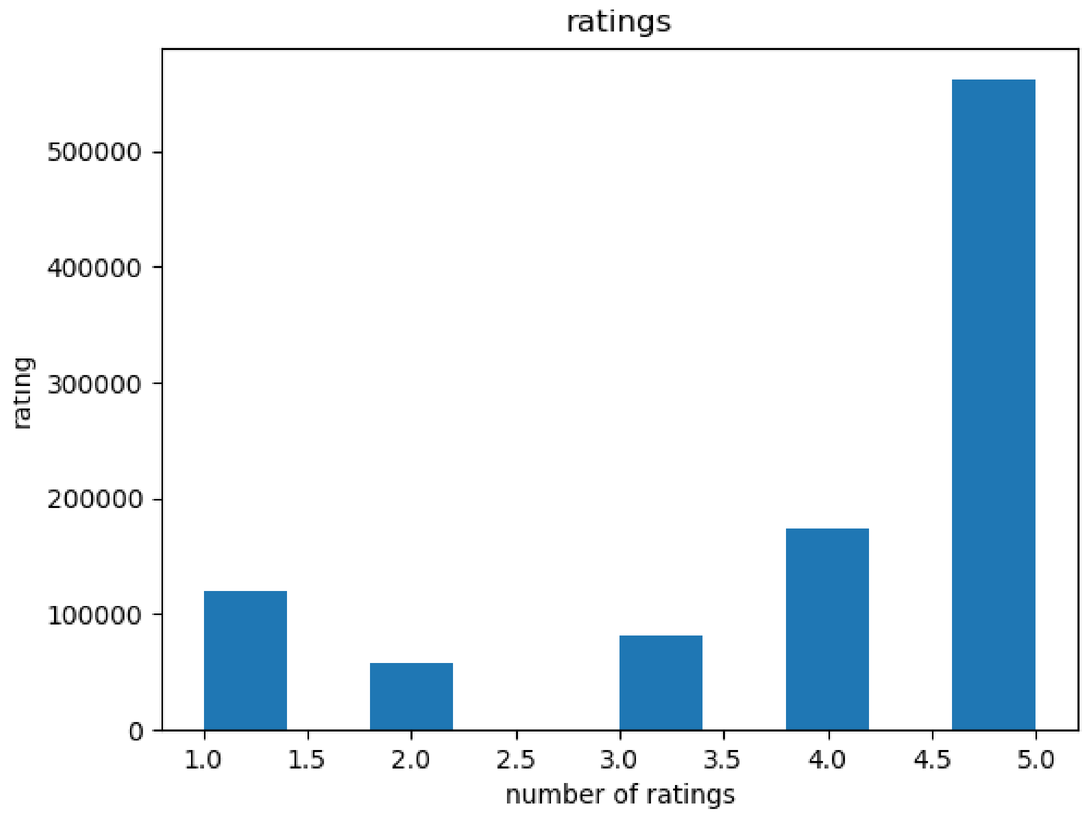
1. The number of ratings per user



2. The number of ratings per item



3. The ratings themselves



Note: for the first two, use `plt.yscale('log', nonposy='clip')` to put the histograms on a log-scale. Also, you can use `X.getnnz` to get the total number of nonzero elements along a specific axis.

1.2 Finding similar items with nearest neighbours

Rubric: {code:6}

We'll use scikit-learn's `neighbors.NearestNeighbors` object to find the items most similar to the example item above, namely the Brass Grill Brush 18 Inch Heavy Duty and Extra Strong, Solid Oak Handle, at URL `https://www.amazon.com/dp/B00CFM0P7Y`.

Find the 5 most similar items to the Grill Brush using the following metrics:

1. Euclidean distance (the `NearestNeighbors` default)
2. Normalized Euclidean distance (you'll need to do the normalization)
3. Cosine similarity (by setting `metric='cosine'`)

Some notes/hints...

- If you run `python main.py -q 1.2`, it will grab the row of `X` associated with the grill brush. The mappers take care of going back and forther between the IDs (like "B00CFM0P7Y") and the indices of the sparse array $(0, 1, 2, \ldots)$.

- Keep in mind that scikit-learn's `NearestNeighbors` is for taking neighbors across rows, but here we're working across columns.
- Keep in mind that scikit-learn's `NearestNeighbors` will include the query item itself as one of the nearest neighbours if the query item is in the "training set".
- Normalizing the columns of a matrix would usually be reasonable to implement, but because X is stored as a sparse matrix it's a bit more of a mess. Therefore, use `sklearn.preprocessing.normalize` to help you with the normalization in part 2.

Did normalized Euclidean distance and cosine similarity yields the same similar items, as expected?

- Yes, they return the same array of items, namely: [93652 103866 103867 103865 98068 98066]

1.3 Total popularity

Rubric: {reasoning:2}

For both Euclidean distance and cosine similarity, find the number of reviews for each of the 5 recommended items and report it. Do the results make sense given what we discussed in class about Euclidean distance vs. cosine similarity and popular items?

Note: in `main.py` you are welcome to combine this code with your code from the previous part, so that you don't have to copy/paste all that code in another section of `main.py`.

- The # ratings of the items chosen by non-normal euclidean are: 55 45 1 1 1
- The # ratings of the items chosen by cosine are: 55 91 45 66 110

  This looks reasonable because cosine metric returned more popular items than euclidean distance. E.g. 3 of the items chosen by the euclidean metrics have only 1 rating

2 Matrix Notation and Minimizing Quadratics

2.1 Converting to Matrix/Vector/Norm Notation

Rubric: {reasoning:3}

Using our standard supervised learning notation (X, y, w) express the following functions in terms of vectors, matrices, and norms (there should be no summations or maximums).

1. $\max_{i\in\{1,2,\ldots,n\}} |w^T x_i - y_i|$.
2. $\sum_{i=1}^n v_i(w^T x_i - y_i)^2 + \frac{\lambda}{2}\sum_{j=1}^d w_j^2$.
3. $\left(\sum_{i=1}^n |w^T x_i - y_i|\right)^2 + \frac{1}{2}\sum_{j=1}^d \lambda_j |w_j|$.

1. $|Xw - y|_\infty$
2. $V|Xw - y|_2^2 + \frac{\lambda}{2}|w|_2^2$
3. $|Xw - y|_1^2 + \frac{1}{2}|\Lambda w|_1$

Note that in part 2 we give a *weight* v_i to each training example, whereas in part 3 we are regularizing the parameters with different weights λ_j. You can use V to denote a diagonal matrix that has the values v_i along the diagonal, and Λ as a diagonal matrix that has the λ_j values along the diagonal. You can assume that all the v_i and λ_i values are non-negative.

2.2 Minimizing Quadratic Functions as Linear Systems

Rubric: {reasoning:3}

Write finding a minimizer w of the functions below as a system of linear equations (using vector/matrix notation and simplifying as much as possible). Note that all the functions below are convex so finding a w with $\nabla f(w) = 0$ is sufficient to minimize the functions (but show your work in getting to this point).

1. $f(w) = \frac{1}{2}\|w - v\|^2$ (projection of v onto real space).
2. $f(w) = \frac{1}{2}\|Xw - y\|^2 + \frac{1}{2}w^T \Lambda w$ (least squares with weighted regularization).
3. $f(w) = \frac{1}{2}\sum_{i=1}^{n} v_i(w^T x_i - y_i)^2 + \frac{\lambda}{2}\|w - w^0\|^2$ (weighted least squares shrunk towards non-zero w^0).

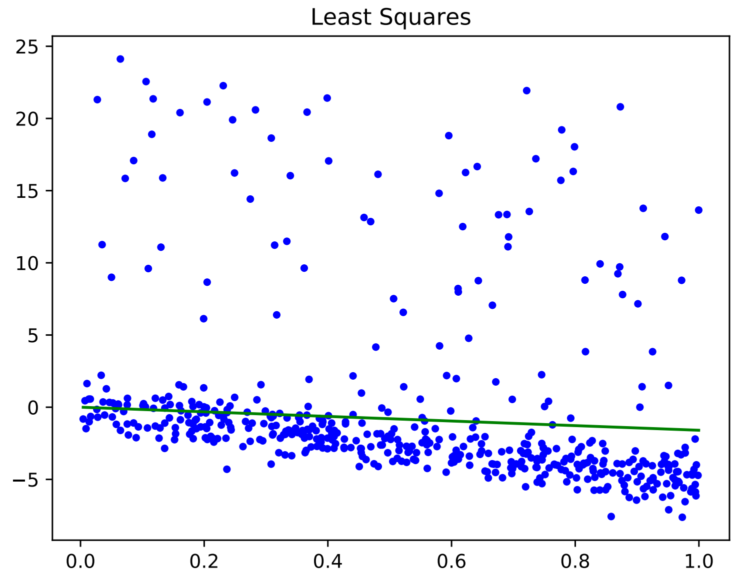
Above we assume that v and w^0 are d by 1 vectors, and Λ is a d by d diagonal matrix (with positive entries along the diagonal). You can use V as a diagonal matrix containing the v_i values along the diagonal.

Hint: Once you convert to vector/matrix notation, you can use the results from class to quickly compute these quantities term-wise. As a sanity check for your derivation, make sure that your results have the right dimensions.

1. $f(w) = \frac{1}{2}\|w - v\|^2 = \frac{1}{2}(w - v)(w - v)^T = \frac{1}{2}|w|^2 - 2vw^T$ setting the gradient to zero: $2w = 2v$ so $w = v$ is the minimizer
2. $f(w) = \frac{1}{2}\|Xw - y\|^2 + \frac{1}{2}w^T\Lambda w = \frac{1}{2}(w^T X^T Xw - Xwy^T - X^T w^T y + yy^T) + \frac{1}{2}w^T \Lambda w$ using the methods we learned in the class while taking the gradient we find that $\nabla f(w) = X^T Xw - X^T y + \Lambda w$ so $(X^T X + \Lambda I)w = X^T y$ and $w = (X^T y)(X^T x + \Lambda I)^{-1}$ if the last term is invertible
3. The first part is almost identical to the first part of the second question. So I am just going to copy the derivation and add the v_i term in the front. So, $\nabla f(w) = vX^T Xw - vX^T\ y + 2w - w^0 - w^{0T}$ where the second part comes from $\frac{\lambda}{2}(w - w^0)^T(w - w^0) = \frac{\lambda}{2}(w^T - w^{0T})(w - w^0)$ and when we take the gradient of this part it becomes $= \lambda I(2w - w^0 - w^{0T})$ then we can see that $(vX^T X + \lambda)w = vX^T y + \frac{\lambda I(w^0 + w^{0T})}{2}$ and if the coefficient of w is invertible then $w = (vX^T y + \frac{\lambda I(w^0+w^{0T})}{2})(vX^T X + \lambda I)^{-1}$

3 Robust Regression and Gradient Descent

If you run `python main.py -q 3`, it will load a one-dimensional regression dataset that has a non-trivial number of 'outlier' data points. These points do not fit the general trend of the rest of the data, and pull the least squares model away from the main downward trend that most data points exhibit:



Note: we are fitting the regression without an intercept here, just for simplicity of the homework question. In reality one would rarely do this. But here it's OK because the "true" line passes through the origin (by design). In Q4.1 we'll address this explicitly.

3.1 Weighted Least Squares in One Dimension

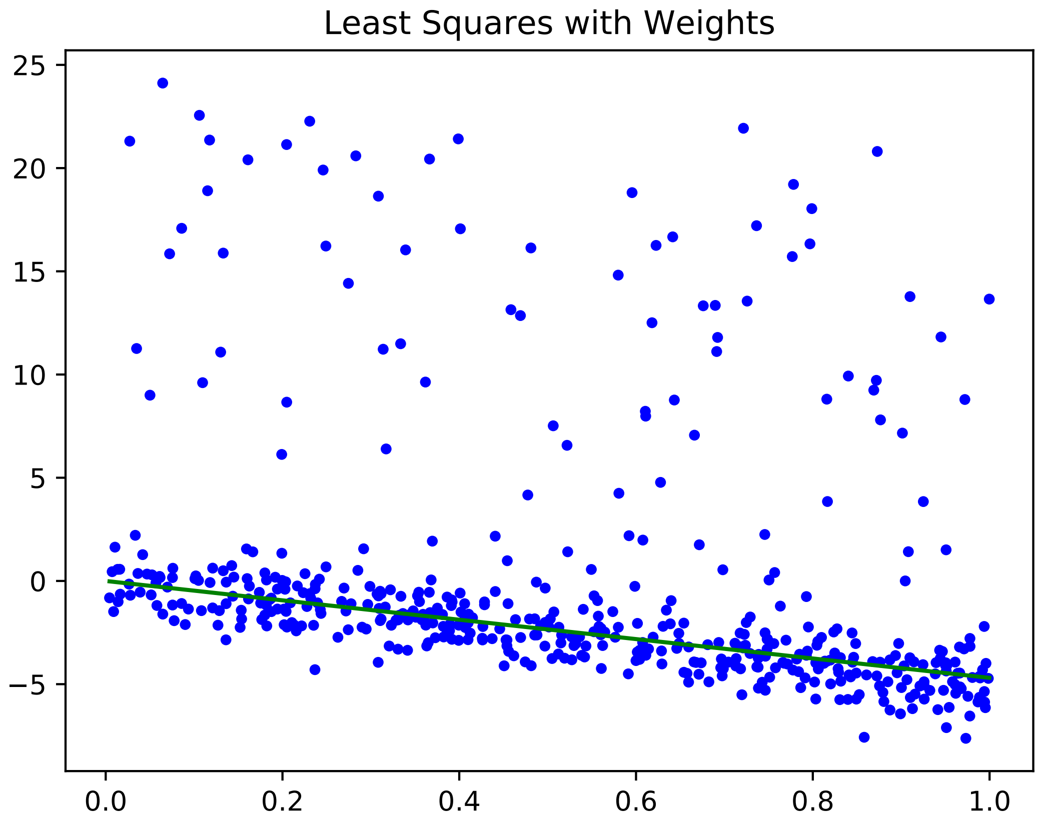
Rubric: {code:3}

One of the most common variations on least squares is *weighted* least squares. In this formulation, we have a weight v_i for every training example. To fit the model, we minimize the weighted squared error,

$$f(w) = \frac{1}{2}\sum_{i=1}^{n} v_i(w^T x_i - y_i)^2.$$

In this formulation, the model focuses on making the error small for examples i where v_i is high. Similarly, if v_i is low then the model allows a larger error. Note: these weights v_i (one per training example) are completely different from the model parameters w_j (one per feature), which, confusingly, we sometimes also call "weights".

Complete the model class, `WeightedLeastSquares`, that implements this model (note that Q2.2.3 asks you to show how a few similar formulation can be solved as a linear system). Apply this model to the data containing outliers, setting $v = 1$ for the first 400 data points and $v = 0.1$ for the last 100 data points (which are the outliers). Hand in your code and the updated plot.



3.2 Smooth Approximation to the L1-Norm

Rubric: {reasoning:3}

Unfortunately, we typically do not know the identities of the outliers. In situations where we suspect that there are outliers, but we do not know which examples are outliers, it makes sense to use a loss function that is more robust to outliers. In class, we discussed using the sum of absolute values objective,

$$f(w) = \sum_{i=1}^{n} |w^T x_i - y_i|.$$

This is less sensitive to outliers than least squares, but it is non-differentiable and harder to optimize. Nevertheless, there are various smooth approximations to the absolute value function that are easy to optimize. One possible approximation is to use the log-sum-exp approximation of the max function[2]:

$$|r| = \max\{r, -r\} \approx \log(\exp(r) + \exp(-r)).$$

Using this approximation, we obtain an objective of the form

$$f(w) = \sum_{i=1}^{n} \log\left(\exp(w^T x_i - y_i) + \exp(y_i - w^T x_i)\right).$$

[2]Other possibilities are the Huber loss, or $|r| \approx \sqrt{r^2 + \epsilon}$ for some small ϵ.

which is smooth but less sensitive to outliers than the squared error. Derive the gradient ∇f of this function with respect to w. You should show your work but you do not have to express the final result in matrix notation.

- $\frac{f(w)}{\delta w_j} = \frac{e^{w^T x_i - y_i} x_{ij} + e^{y_i - w^T x_i} x_{ij}}{e^{w^T x_i - y_i} + e^{y_i - w^T x_i}}$
- $\nabla f(w) = X^T \frac{e^{w^T x_i - y_i} + e^{y_i - w^T x_i}}{e^{w^T x_i - y_i} + e^{y_i - w^T x_i}}$
-

3.3 Robust Regression

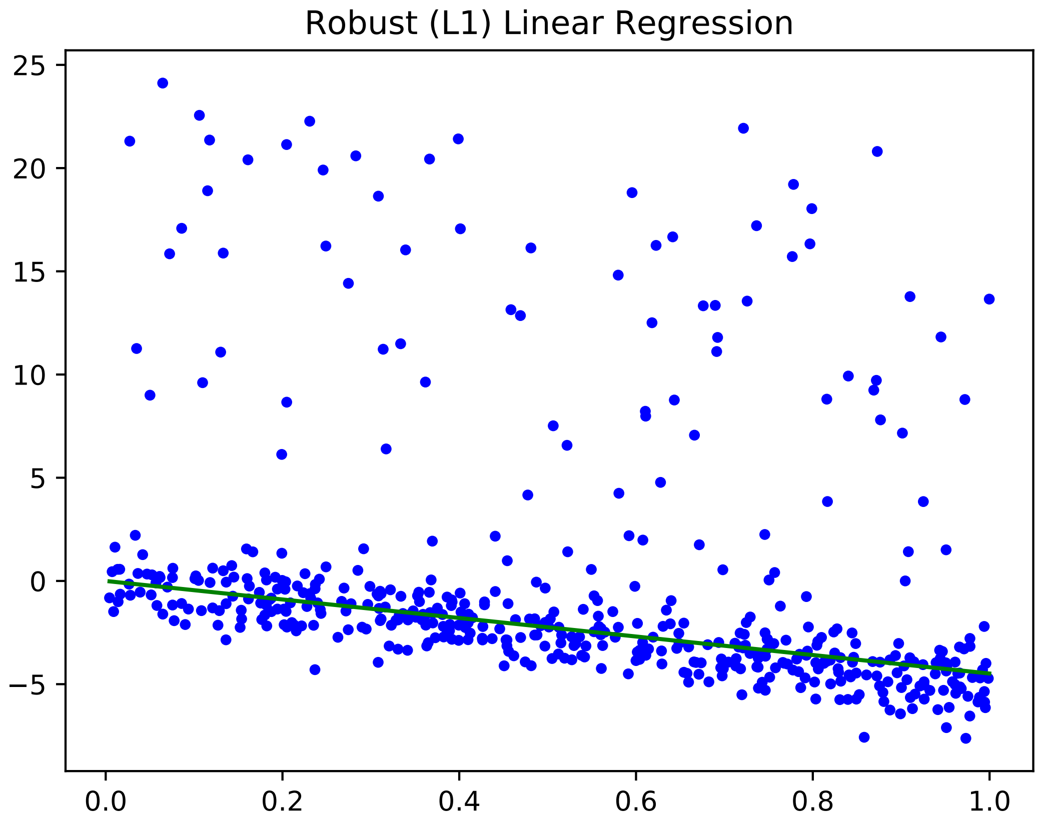
Rubric: {code:3}

The class `LinearModelGradient` is the same as `LeastSquares`, except that it fits the least squares model using a gradient descent method. If you run `python main.py -q 3.3` you'll see it produces the same fit as we obtained using the normal equations.

The typical input to a gradient method is a function that, given w, returns $f(w)$ and $\nabla f(w)$. See `funObj` in `LinearModelGradient` for an example. Note that the `fit` function of `LinearModelGradient` also has a numerical check that the gradient code is approximately correct, since implementing gradients is often error-prone.[3]

An advantage of gradient-based strategies is that they are able to solve problems that do not have closed-form solutions, such as the formulation from the previous section. The class `LinearModelGradient` has most of the implementation of a gradient-based strategy for fitting the robust regression model under the log-sum-exp approximation. The only part missing is the function and gradient calculation inside the `funObj` code. Modify `funObj` to implement the objective function and gradient based on the smooth approximation to the absolute value function (from the previous section). Hand in your code, as well as the plot obtained using this robust regression approach.

[3] Sometimes the numerical gradient checker itself can be wrong. See CPSC 303 for a lot more on numerical differentiation.



4 Linear Regression and Nonlinear Bases

In class we discussed fitting a linear regression model by minimizing the squared error. In this question, you will start with a data set where least squares performs poorly. You will then explore how adding a bias variable and using nonlinear (polynomial) bases can drastically improve the performance. You will also explore how the complexity of a basis affects both the training error and the test error. In the final part of the question, it will be up to you to design a basis with better performance than polynomial bases.

4.1 Adding a Bias Variable

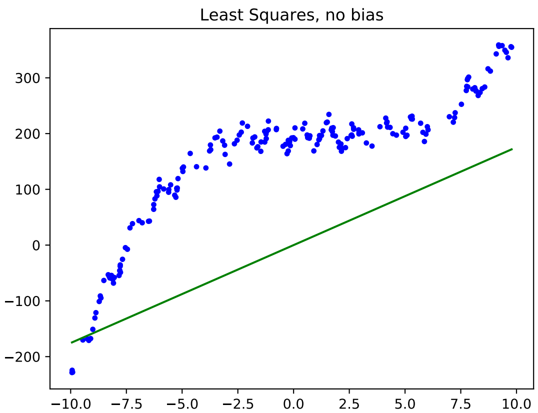
Rubric: {code:3}

If you run `python main.py -q 4`, it will:

1. Load a one-dimensional regression dataset.
2. Fit a least-squares linear regression model.
3. Report the training error.
4. Report the test error (on a dataset not used for training).

5. Draw a figure showing the training data and what the linear model looks like.

Unfortunately, this is an awful model of the data. The average squared training error on the data set is over 28000 (as is the test error), and the figure produced by the demo confirms that the predictions are usually nowhere near the training data:



The y-intercept of this data is clearly not zero (it looks like it's closer to 200), so we should expect to improve performance by adding a *bias* (a.k.a. intercept) variable, so that our model is

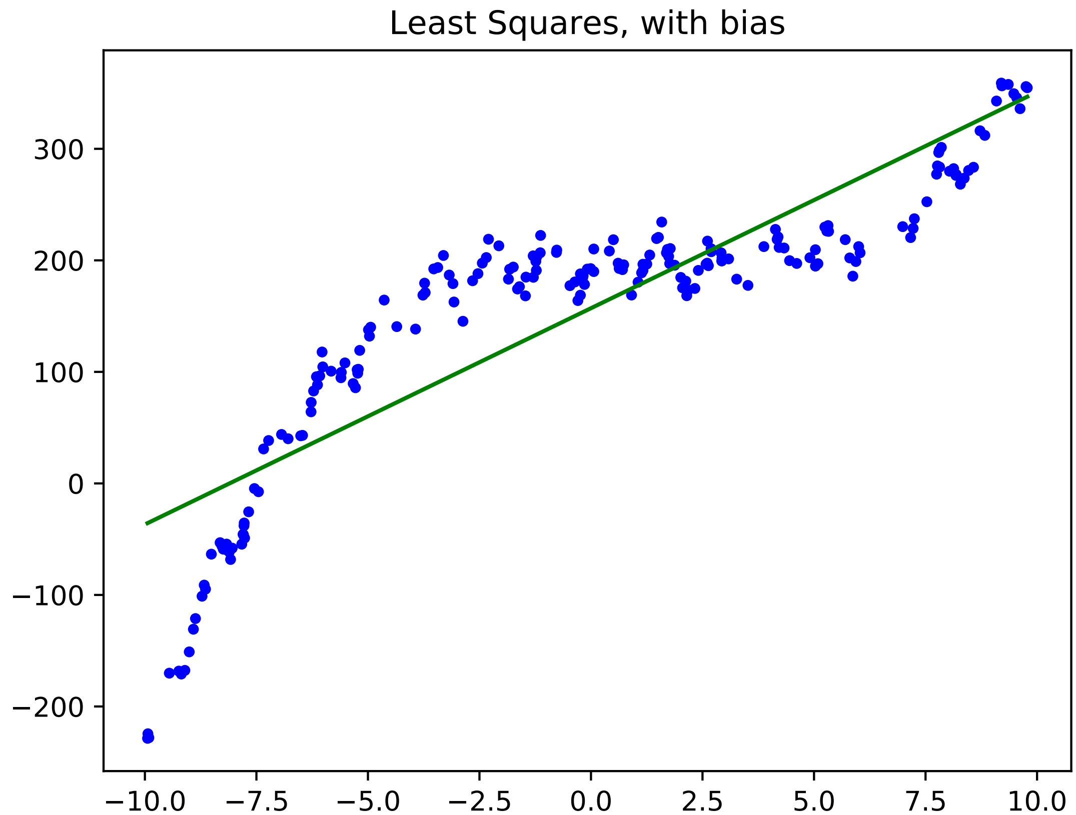
$$y_i = w^T x_i + w_0.$$

instead of

$$y_i = w^T x_i.$$

In file `linear_model.py`, complete the class, `LeastSquaresBias`, that has the same input/model/predict format as the `LeastSquares` class, but that adds a *bias* variable (also called an intercept) w_0 (also called β in lecture). Hand in your new class, the updated plot, and the updated training/test error.

Hint: recall that adding a bias w_0 is equivalent to adding a column of ones to the matrix X. Don't forget that you need to do the same transformation in the `predict` function.



- The training error:3551.3
- The test error: 3393.9

4.2 Polynomial Basis

Rubric: {code:4}

Adding a bias variable improves the prediction substantially, but the model is still problematic because the target seems to be a *non-linear* function of the input. Complete `LeastSquarePoly` class, that takes a data vector x (i.e., assuming we only have one feature) and the polynomial order p. The function should perform a least squares fit based on a matrix Z where each of its rows contains the values $(x_i)^j$ for $j = 0$ up to p. E.g., `LeastSquaresPoly.fit(x,y)` with $p = 3$ should form the matrix

$$Z = \begin{bmatrix} 1 & x_1 & (x_1)^2 & (x_1)^3 \\ 1 & x_2 & (x_2)^2 & (x_2)^3 \\ \vdots & & & \\ 1 & x_n & (x_n)^2 & (x_N)^3 \end{bmatrix},$$

and fit a least squares model based on it. Hand in the new class, and report the training and test error for $p = 0$ through $p = 10$. Explain the effect of p on the training error and on the test error.

Note: you should write the code yourself; don't use a library like sklearn's `PolynomialFeatures`.

- p=0 Training error = 15480.5 Test error = 14390.8
- p=1 Training error = 3551.3 Test error = 3393.9
- p=2 Training error = 2168.0 Test error = 2480.7
- p=3 Training error = 252.0 Test error = 242.8
- p=4 Training error = 251.5 Test error = 242.1
- p=5 Training error = 251.1 Test error = 239.5
- p=6 Training error = 248.6 Test error = 246.0
- p=7 Training error = 247.0 Test error = 242.9
- p=8 Training error = 241.3 Test error = 246.0
- p=9 Training error = 235.8 Test error = 259.3
- p=10 Training error = 235.1 Test error = 256.3

  The lowest test and training error occur at p=5. The training error and test error sharply decline before p=3 and the rate gets slower after that. Training error always decrease even though the rate of change gets lower. Test error gets lower until p=7 then starts increasing again and and decreases from p=9 to p=10

5 Very-Short Answer Questions

Rubric: {reasoning:7}

1. Suppose that a training example is global outlier, meaning it is really far from all other data points. How is the cluster assignment of this example by k-means? And how is it set by density-based clustering?

   In k-means, the outliers get assigned to the closest mean. They would also pull one of the means towards themselves - probably not significantly because of their number- since it is so far away. In density based clustering, if the epsilon and number of neighbors are not set specifically to contain outliers, they could be left out, meaning they do not belong to the groups specified by density clustering

2. Why do need random restarts for k-means but not for density-based clustering?

   Because k-means depends on the initialization of the means whereas density-based clustering depends on two parameters which determine the clusterings which is assigned by the programmer

3. Can hierarchical clustering find non-convex clusters?

   No, hierarchical clustering can only find convex regions as discussed in the class

4. For model-based outlier detection, list an example method and problem with identifying outliers using this method.

   Probabilistic method. The problems are: mean and variance are sensitive to outliers meaning they are affected by outliers. Also, it assumes uni-modality which does not necessarily hold for our data

5. For graphical-based outlier detection, list an example method and problem with identifying outliers using this method.

   We can use box plot. But since we can only look at 1-variable at a time it might take a while to identify to outliers.

6. For supervised outlier detection, list an example method and problem with identifying outliers using this method.

   We can use KNN. The downside is it is supervised so we need to know something about the dataset and outliers beforehand

7. If we want to do linear regression with 1 feature, explain why it would or would not make sense to use gradient descent to compute the least squares solution.

   It would make sense since in the one dimensional case, the gradient would just be the derivative of our function. So as long as our function is convex,continuous and the its derivative is defined. There is nothing wrong with using gradient descent with one variable.

8. Why do we typically add a column of 1 values to X when we do linear regression? Should we do this if we're using decision trees?

   We do this because we want to account for the bias(intercept). This way we do not have to write the bias vector in the equations, facilitating writing and solving equations. We should not do this if we're using decision trees because all the elements would have the same intercept then this level(bias) in the decision tree does not help us identify elements

9. If a function is convex, what does that say about stationary points of the function? Does convexity imply that a stationary points exists?

   All stationary points in a convex function are minima points. Convexity does not imply stationary points exist: e.g. f(x) = x is convex since f"(x) = 0 for all x so f(x) is convex. But f(x) has no stationary points since f'(x) = 1 for all x

10. Why do we need gradient descent for the robust regression problem, as opposed to just using the normal equations? Hint: it is NOT because of the non-differentiability. Recall that we used gradient descent even after smoothing away the non-differentiable part of the loss.

    The normal equations take $O(nd^2 + d^3$ whereas gradient descent $O(ndt)$ so gradient descent is better for large d. It is linear in d

11. What is the problem with having too small of a learning rate in gradient descent?

    Because since it take $O(ndt)$ the gradient descent is linear in n. Picking a too small step size increases n and as a consequence the algorithm takes too long

12. What is the problem with having too large of a learning rate in gradient descent?

    Since the learning rate is too fast, We can jump over the local/absolute minimum and keep searching forever and not finding any minima. Also, the loss goes up

13. What is the purpose of the log-sum-exp function and how is this related to gradient descent?

    log-sum-exp is approximately the same as absolute value loss function. As used in this homework, we can use log-sum-exp in the case where we have outliers. It is more robust to outliers and smooth enough that we can take derivatives

14. What type of non-linear transform might be suitable if we had a periodic function?

    Change the basis such that we have a basis in terms of similar non-linear periodic functions such as sin's,cos's,tan's,cot's and the variable itself.

6 Project Proposal (FOR CPSC 532M STUDENTS ONLY!)

If you are enrolled in CPSC 340, ignore this question.

If you enrolled in CPSC 532M, for the final part of this assignment you must a submit a project proposal for your course project. The proposal should be a maximum of 2 pages (and 1 page or half of a page is ok if you can describe your plan concisely). The proposal should be written for the instructors and the TAs, so you don't need to introduce any ML background but you will need to introduce non-ML topics. The projects must be done in groups of 2-3. If you are doing your assignment in a group that is different from your project group, only 1 group member should include the proposal as part of their submission (we'll do the merge across assignments, and this means that assignments could have multiple proposals). Please state clearly who is involved with each project proposal.

There is quite a bit of flexibility in terms of the type of project you do, as I believe there are many ways that people can make valuable contributions to research. However, note that ultimately the final deliverable for the project will be a report containing at most 6 pages of text (the actual document can be longer due to figures, tables, references, and proofs) that emphasizes a particular "contribution" (i.e., what doing the project has added to the world). The reason for this, even though it's strange for some possible projects, is that this is the standard way that results are communicated to the research community.

The three mains ingredients of the project proposal are:

1. What problem you are focusing on.
2. What you plan to do.
3. What will be the "contribution".

Also, note that for the course project that negative results (i.e., we tried something that we thought we would work in a particular setting but it didn't work) are acceptable (and often unavoidable).

Here are some standard project "templates" that you might want to follow:

- **Application bake-off**: you pick a specific application (from your research, personal interests, or maybe from Kaggle) or a small number of related applications, and try out a bunch of techniques (e.g., random forests vs. logistic regression vs. generative models). In this case, the contribution would be showing that some methods work better than others for this specific application (or your contribution could be that everything works equally well/badly).
- **New application**: you pick an application where people aren't using ML, and you test out whether ML methods are effective for the task. In this case, the contribution would be knowing whether ML is suitable for the task.
- **Scaling up**: you pick a specific machine learning technique, and you try to figure out how to make it run faster or on larger datasets. In this case, the contribution would be the new technique and an evaluation of its performance, or could be a comparison of different ways to address the problem.
- **Improving performance**: you pick a specific machine learning technique, and try to extend it in some way to improve its performance. In this case, the contribution would be the new technique and an evaluation of its performance.
- **Generalization to new setting**: you pick a specific machine learning technique, and try to extend it to a new setting (for example, making a multi-label version of random forests). In this case, the contribution would be the new technique and an evaluation of its performance, or could be a comparison of different ways to address the problem.
- **Perspective paper**: you pick a specific topic in ML, read at least 10 papers on the topic, then write a report summarizing what has been done on the topic and what are the most promising directions of

future work. In this case, the contribution would be your summary of the relationships between the existing works, and your insights about where the field is going.

- **Coding project**: you pick a specific method or set of methods, and build an implementation of them. In this case, the contribution could be the implementation itself or a comparison of different ways to solve the problem.
- **Theory**: you pick a theoretical topic (like the variance of cross-validation), read what has been done about it, and try to prove a new result (usually by relaxing existing assumptions or adding new assumptions). The contribution could be a new analysis of an existing method, or why some approaches to analyzing the method will not work.
- **Reproducibility Challenge**: you take part in the 2019 ICLR reproducibility challenge, where you try to reproduce the results of a recently-submitted machine learning paper. Information on the challenge is available here: `https://reproducibility-challenge.github.io/iclr_2019`

The above are just suggestions, and many projects will mix several of these templates together, but if you are having trouble getting going then it's best to stick with one of the above templates. Also note that the project can focus on topics not covered in the course (like RNNs), so there is flexibility in the topic, but the topic should be closely-related to ML.

This question is mandatory but will not be formally marked: it's just a sanity check that you have at least one project idea that fits within the scope of a 532M course project, and it's an excuse for you to allocate some time to thinking about the project. Also, there is flexibility in the choice of project topics even after the proposal: if you want to explore different topics you can ultimately choose to do a project that is unrelated to the one in your proposal (and changing groups is ok too).