CPSC 340 Assignment 3 (due Friday, Feb 6 at 11:55pm)

Zitong Wang 40883150 k5j0b, Jinyuan Hu 41465155 z3o0b

January, 2019

1 Finding Similar Items

1.1 Exploratory data analysis

1.1.1 Most popular item

The most popular item is: B000HCLLMM

Total Stars: 14454.0

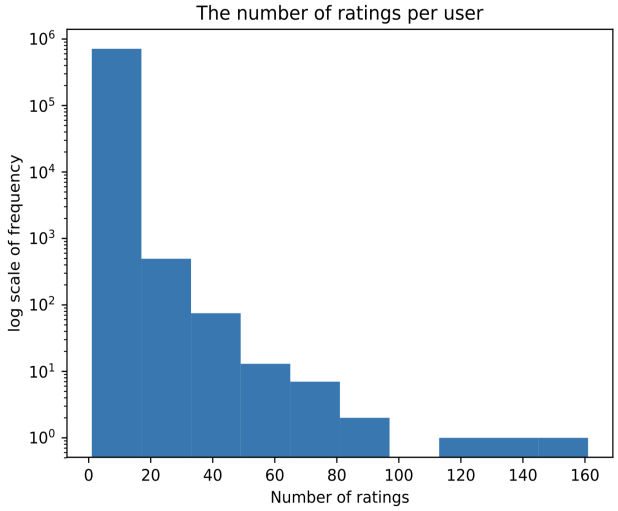
1.1.2 User with most reviews

The most reviewed user is: A100WO06OQR8BQ

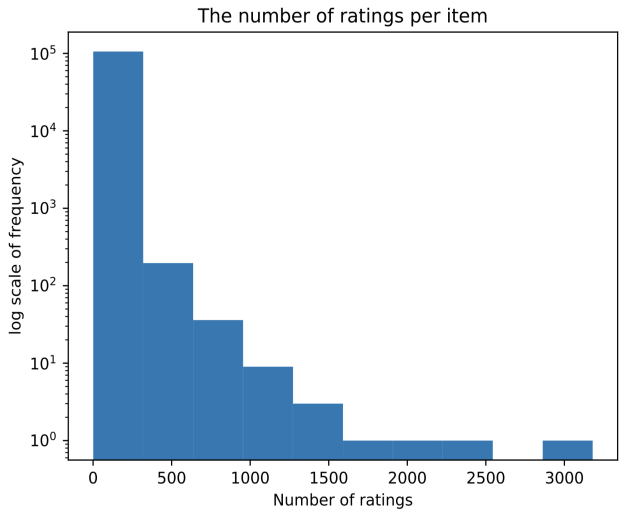
Total comments: 161

1.1.3 Histograms

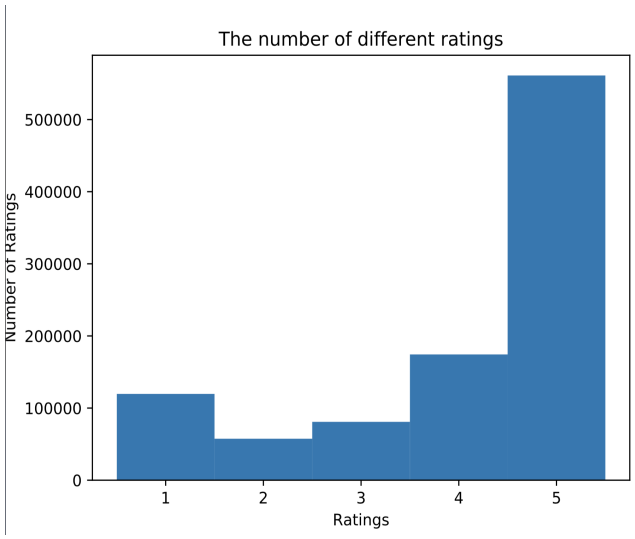
1. The number of ratings per user:



2. The number of ratings per item:



3. The ratings themselves



1.2 Finding similar items with nearest neighbours

1. **Euclidean Distance:**

   1 th nearest item is: B00IJB5MCS

   2 th nearest item is: B00IJB4MLA

   3 th nearest item is: B00EXE4O42

   4 th nearest item is: B00743MZCM

   5 th nearest item is: B00HVXQY9A

2. **Normalized Euclidean distance:**

   1 th nearest item is: B00IJB8F3G

2 th nearest item is: B00IJB5MCS

3 th nearest item is: B00IJB4MLA

4 th nearest item is: B00EF45AHU

5 th nearest item is: B00EF3YF0Y

3. **Cosine similarity:**

   1 th nearest item is: B00IJB8F3G

   2 th nearest item is: B00IJB5MCS

   3 th nearest item is: B00IJB4MLA

   4 th nearest item is: B00EF45AHU

   5 th nearest item is: B00EF3YF0Y

1.3 Total popularity

1. Euclidean Distance:

   1 th nearest item is: B00IJB5MCS has 55 reviews.

   2 th nearest item is: B00IJB4MLA has 45 reviews.

   3 th nearest item is: B00EXE4O42 has 1 reviews.

   4 th nearest item is: B00743MZCM has 1 reviews.

   5 th nearest item is: B00HVXQY9A has 1 reviews.

2. Cosine similarity:

   1 th nearest item is: B00IJB5MCS has 55 reviews.

   2 th nearest item is: B00IJB8F3G has 91 reviews.

   3 th nearest item is: B00IJB4MLA has 45 reviews.

   4 th nearest item is: B00EF45AHU has 66 reviews.

   5 th nearest item is: B00EF3YF0Y has 110 reviews.

The result seems reasonable since the last 3 result for the euclidean distance is 1 review which clearly shows that it is effected by the number of review. It is inaccurate.

2 Matrix Notation and Minimizing Quadratics

2.1 Converting to Matrix/Vector/Norm Notation

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

2.2 Minimizing Quadratic Functions as Linear Systems

2.2.1

Since $f(w) = \frac{1}{2}(w-v)^T(w-v) = \frac{1}{2}(w^T - v^T)(w-v)$
it can be simplified as: $\frac{1}{2}(w^Tw - w^Tv - v^Tw - v^Tv)$
which equals to: $\frac{1}{2}w^Tw - w^Tw - \frac{1}{2}v^Tv$
by taking gradient with respect to w we have: $\bigtriangledown f(w) = w - v = 0$, s.t $w = v$,
which can minimize the function.

2.2.2

Since $f(w) = \frac{1}{2}[((Xw)^T - y^T)(Xw-y)] + \frac{1}{2}w^T\Lambda w = \frac{1}{2}[(w^TX^T - y^T)(Xw-y)] + \frac{1}{2}w^T\Lambda w$
we can simplify this as: $\frac{1}{2}(w^TX^TXw - w^TX^Ty - y^TXw + y^Ty) + \frac{1}{2}w^T\Lambda w$
which equals to: $\frac{1}{2}w^TX^TXw - w^TX^Ty + \frac{1}{2}y^Ty + \frac{1}{2}w^T\Lambda w$
By setting gradient to zero: $X^TXw - X^Ty + \Lambda w = 0$
Such that: Solve $(X^TX + \Lambda)w = X^Ty$, **which can minimize the function.**

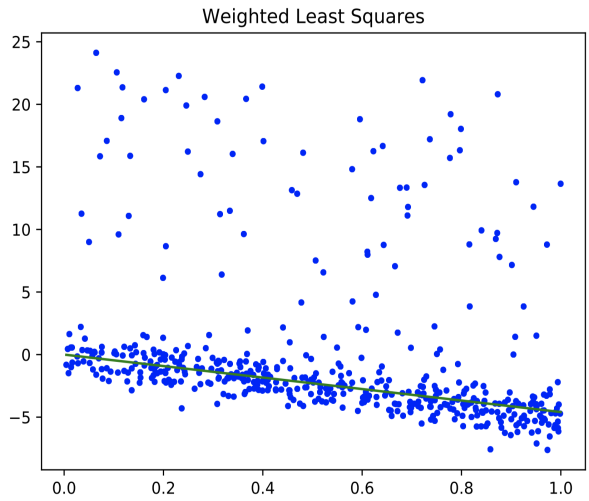
2.2.3

Since $f(w) = \frac{1}{2}(Xw-y)^TV(Xw-y) + \frac{1}{2}(w-w^o)^T\Lambda(w-w^o)$
which equals to: $\frac{1}{2}(w^TX^T - y^T)V(Xw-y) + \frac{1}{2}(w^T - w^{o^T})\Lambda(w-w^o)$
It can be simplified as: $\frac{1}{2}w^TX^TVXw - w^TX^TVy + \frac{1}{2}y^TVy + \frac{1}{2}w^T\lambda w - w^T\Lambda w^o + \frac{1}{2}w^{o^T}\Lambda w^o$
By setting gradient to zero: $\bigtriangledown f(w) = X^TVXw - X^TVy + \Lambda w - \Lambda w^o = 0$
Such that we have: Solve $(X^TV + \Lambda)w = X^TVy + \Lambda w^o$, **which can minimize the function.**

3 Robust Regression and Gradient Descent

3.1 Weighted Least Squares in One Dimension

The training error is 40.9. Please refer linear_model.py for implementation of weighted least squares.



3.2 Smooth Approximation to the L1-Norm

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

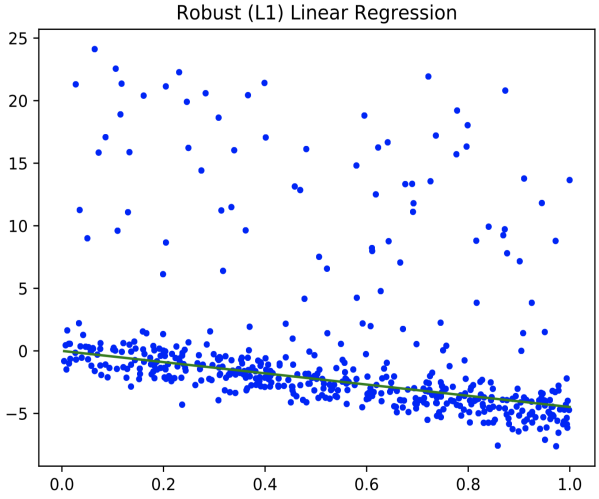
It can be simplified as: $\bigtriangledown f(w) = \sum_{i=1}^{n} \frac{x_i exp(w^T x_i - y_i) - x_i exp(y_i - w^T x_i)}{exp(w^T x_i - y_i) + exp(y_i - w^T x_i)}$

It equals to: $\sum_{i=1}^{n} x_i \frac{exp(w^T x_i - y_i) - exp(y_i - w^T x_i)}{exp(w^T x_i - y_i) + exp(y_i - w^T x_i)}$

Therefore we have: $\bigtriangledown f(w) = X^T r$

where r equals to: $r_i = \sum_{i=1}^{n} \frac{exp(w^T x_i - y_i) - exp(y_i - w^T x_i)}{exp(w^T x_i - y_i) + exp(y_i - w^T x_i)}$

3.3 Robust Regression



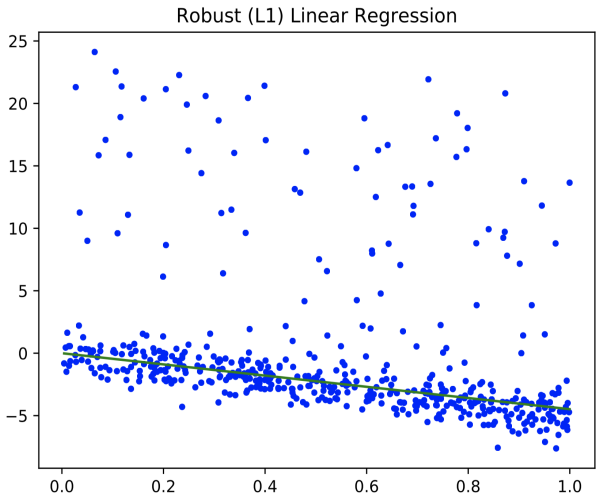
The training error is: 40.6, which is close to the weighted least squares.

Please refer to file linear_model.py for implementation.

4 Linear Regression and Nonlinear Bases

4.1 Adding a Bias Variable

For updated plot shown below, we have Training error = 3551.3, Test error = 3393.9



Please refer to file linear_model.py for implementation.

4.2 Polynomial Basis

Errors that obtained (for degree of polynomial from 0 to 10) are:

P=0:

Training error = 28122.8

Test error = 28299.0

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

We observe that, as degree of polynomial(P) increase, the training error decrease correspondingly. Noticing that test error was decreasing in the first place but increasing after P = 6, it might due to overfitting. By increasing the degree of polynomial, errors are decreasing significantly until P ¿= 3; after P = 3, errors are almost remained the same.

please refer to file linear_model.py for code submission.

5 Very-Short Answer Questions

1. If we were using K-means, the clusters that generated would be closer to the outlier while such a outlier won't affect the clusters by density-based clustering.
2. Because k-means is sensitive to initially selected centroids (means) while density based clustering does not.
3. Yes. Since we are combining 2 closet cluster each time.
4. Assuming data following normal distribution, by looking at z-score, model based detection is hard to tell the mean and variance as well as outliers.

5. Box plot, by using box plot (or scatter plot) identifying outliers, it would totally depend on user's judgment, which could be biased and inaccurate.
6. Decision Tree. we need to know what outliers look like.
7. Solving directly by calculating normal equation would be better, since gradient descent would perform better in run-time when features are more sophisticated.
8. By adding column of 1s, we could have non-zero intercept that can avoid passing through origin to cause inaccuracy. We don't need to add column 1s to decision tree since we can directly change decision stump by adding some constants.
9. If a function is convex, the stationary point of such a function would be part of the global minima. Convexity does not imply stationary point exists.
10. Because when number of features is large, using gradient descent would have better performance (ie runtime).
11. It would cause fitting the data set too slow as well as the runtime.
12. It would fit the data set too fast such that may 'pass through' (diverge) and never reach the minima again.
13. It is used for smoothing the equation.
14. Trigonometric function might be applicable.