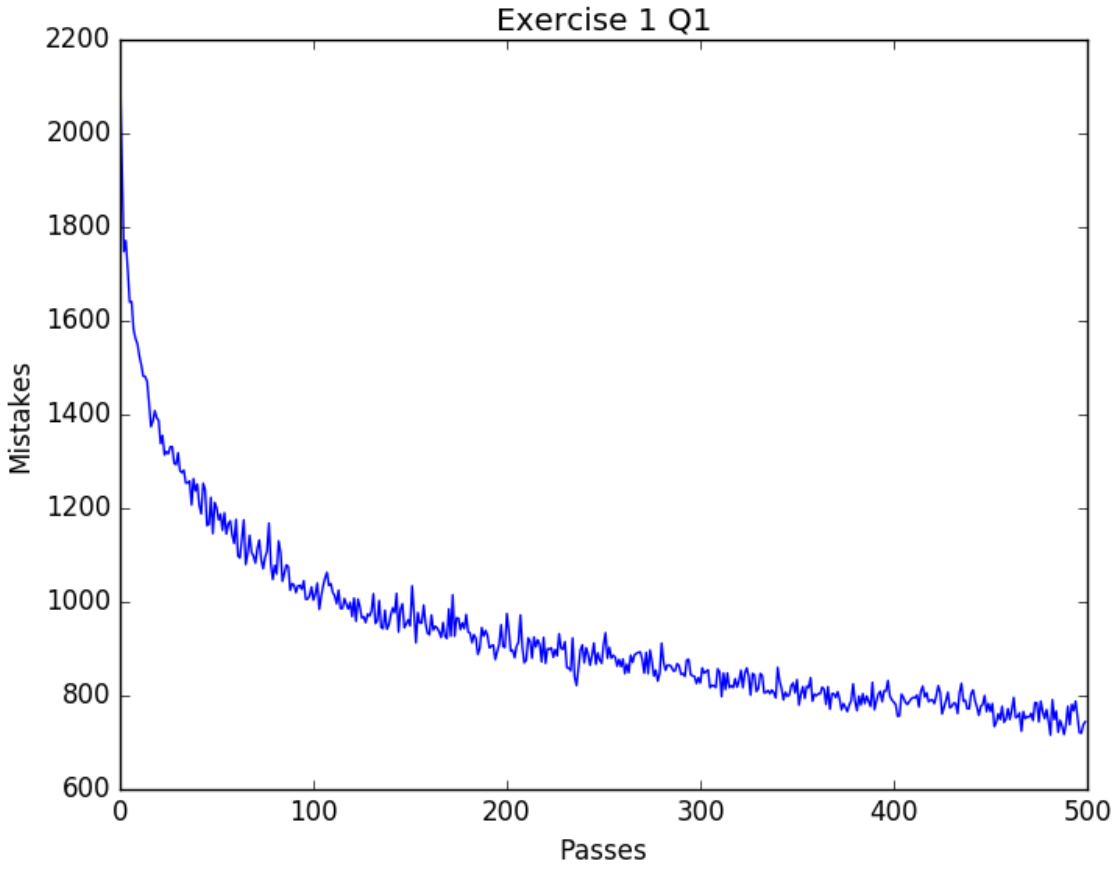
Boshen Cui
20613736
b2cui@edu.uwaterloo.ca

Please ensure that you have numpy, matplotlib, and sklearn installed. If you do not, you can simply do `pip install –r requirements.txt`
All scripts run on **python 2**

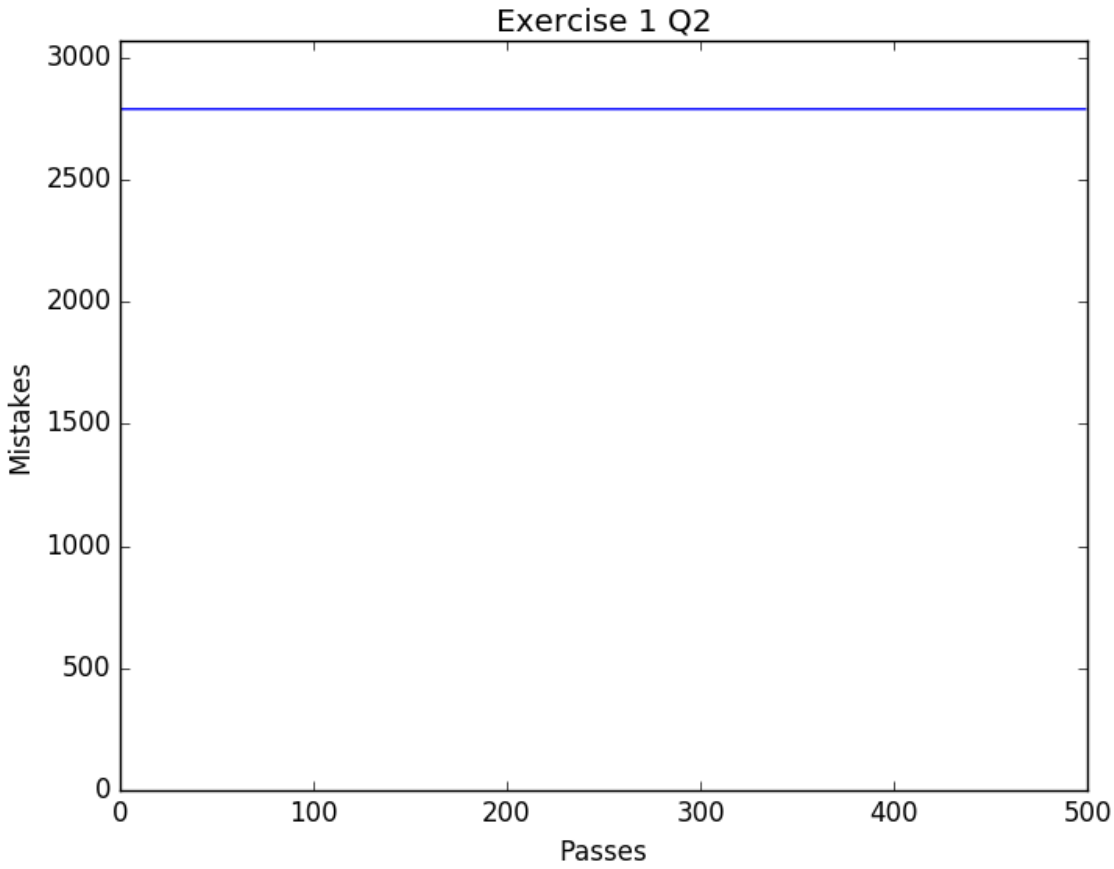
Exercise 1:

1. Plot of mistakes w.r.t. the number of passes:



To run this for yourself, run `python A1.py`, and type in `E1Q1`. The outputted plot will be in an image called Exercise1Q1.png.

2. Plot of mistakes w.r.t. number of passes, updating the weight vectors regardless of prediction result:



To run this for yourself, run `python A1.py`, and type in `E1Q2`. The outputted plot will be in an image called Exercise1Q2.png.

3. Suppose there exists a w*, b* s.t. for all i,

$$x_i \cdot w^* + b^* \geq 0 \quad if\ y_i = 1$$
$$x_i \cdot w^* + b^* < 0 \quad if\ y_i = -1$$

Let x' be the largest x_i s.t. $x' \cdot w^* + b^* < 0$ still holds (i.e. the closest point, or points to 0 that is still negative). Let $c = x' \cdot w^* + b^*$. We define $b' = b^* + \frac{c}{2}$ and $w' = w^*$

Consider the case where $x_i \cdot w^* + b^* = 0$. Now, since $b' = b^* + \frac{c}{2}$, $b' > b^*$,

so $x_i \cdot w^* + b' > 0$ for all $y_i = 1$.

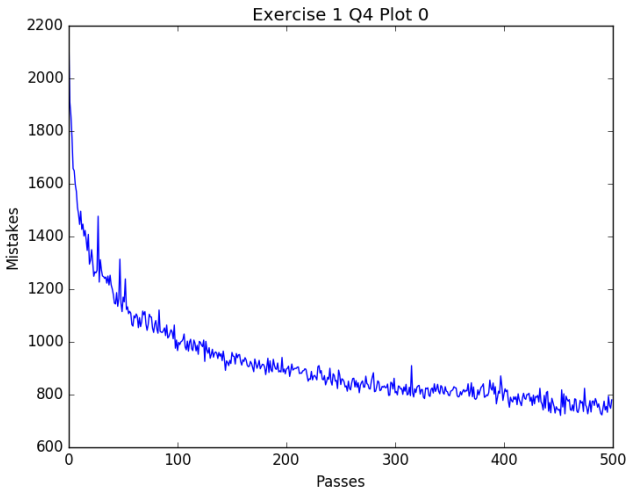
Since $b' = b^* + \frac{c}{2}$, $x' \cdot w^* + b' = x' \cdot w^* + b^* + (x' \cdot w^* + b^*)/2 = 1.5(x' \cdot w^* + b^*) < 0$.

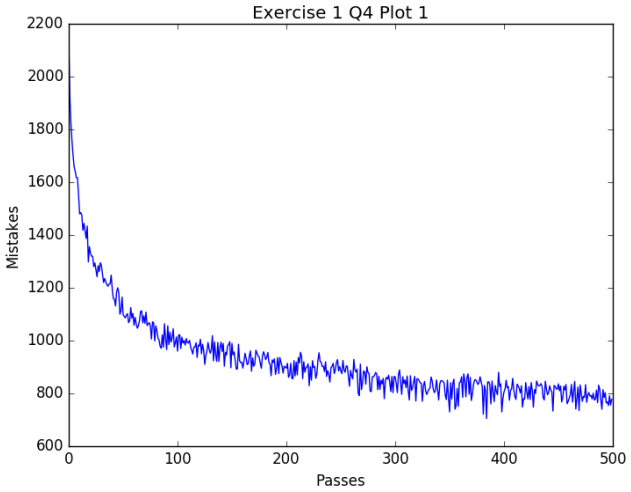
Since x' is the largest x_i s.t. $x' \cdot w^* + b^* < 0$, $x_i \cdot w^* + b^* < 0 \quad if\ y_i = -1$ still holds.

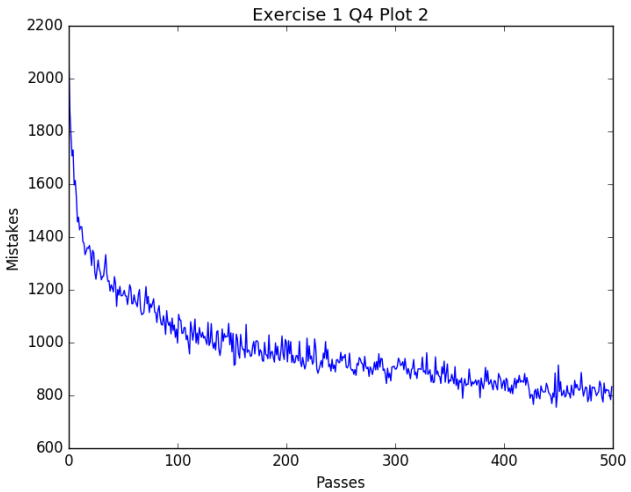
Thus, there exists a w', b', such that for all i,

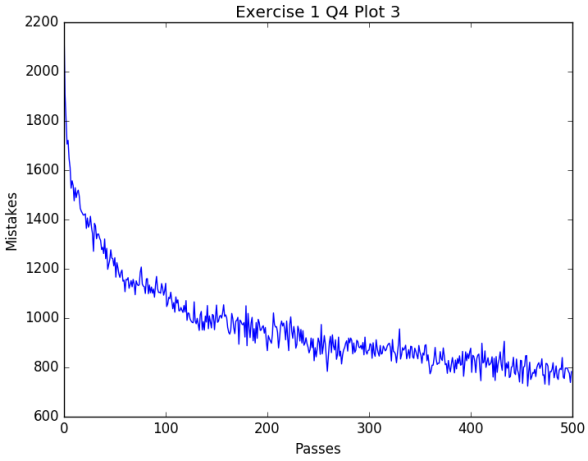
$$x_i \cdot w' + b' > 0 \quad if\ y_i = 1$$
$$x_i \cdot w' + b' < 0 \quad if\ y_i = -1$$

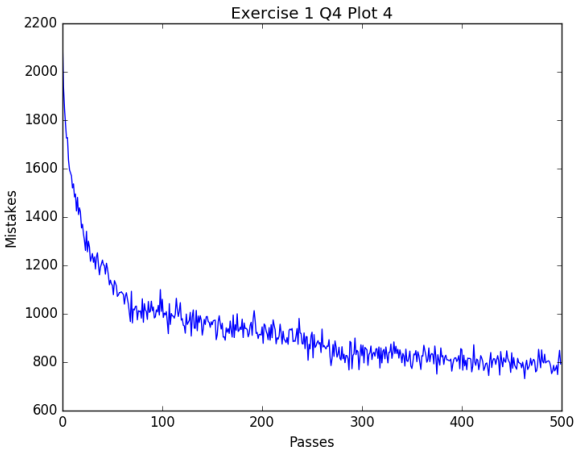
4. 5 plots of the number of mistakes w.r.t. the number of passes, with columns permuted:











To run this for yourself, run `python A1.py`, and type in `E1Q4`. The outputted plot will be in 5 separate images labeled Exercise1Q4-Plot0.png to Exercise1Q4-Plot4.png.

Exercise 2:

To run any of these questions for yourself, run `python A1.py`, and type in the appropriate question number: `E2Q1`, `E2Q2`, `E2Q3`, `E2Q4`, `E2Q5`

1.

Lambda	Training Set MSE	Average Validation Set MSE	Test Set MSE	% Nonzeros in W
0	9.694298639	14.11000839	370.2229573	100
10	9.958153355	14.47373515	190.015861	100
20	9.975538654	14.49419226	185.4553425	100
30	9.98171703	14.50138335	183.887614	100
40	9.984882847	14.5050525	183.0945681	100
50	9.986807508	14.50727806	182.615768	100
60	9.988101258	14.50877192	182.2953213	100
70	9.989030628	14.50984397	182.0658177	100
80	9.989730552	14.51065077	181.8933537	100
90	9.990276663	14.51127992	181.7590153	100
100	9.990714642	14.51178427	181.6514186	100

2.

Lambda	Training Set MSE	Average Validation Set MSE	Test Set MSE	% Nonzeros in W
0	92.76733782	2.76629E+12	527.8160729	100
10	96.41562192	2.79639E+12	784.0072391	100
20	96.66221246	2.79735E+12	812.6460923	100
30	96.75008358	2.79768E+12	821.886014	100
40	96.79518323	2.79784E+12	827.3969749	100
50	96.82253846	2.79795E+12	830.29364	100
60	96.8409779	2.79801E+12	831.6832595	100
70	96.85420067	2.79806E+12	833.4077781	100
80	96.86414702	2.7981E+12	835.6336142	100
90	96.87198525	2.79813E+12	835.3725475	100
100	96.87813174	2.79815E+12	837.5677575	100

By scaling one data point randomly, we've essentially skewed the fitted line away from where most/all of the points are, in order to compensate for the one random point we've multiplied by 10^6/10^3. This means that for literally all of the other points, the regression model will overshoot, and overestimate, which leads to a high MSE in the training set, validation set, and testing set.

3.

Lambda	Training Set MSE	Average Validation Set MSE	Test Set MSE	% Nonzeros in W
0	47618.34118	142043831.6	3269025552	100
10	47618.34118	1484557527	1.71811E+13	100
20	47618.34118	1917727340	2393181672	100
30	47618.34118	6.6891E+11	427271344.9	100
40	47618.34118	1102888225	172267692.5	100
50	47618.34118	46665921.65	1216626425	100
60	47618.34118	164427103.6	2257178954	100
70	47618.34118	1884835744	108279410.8	100
80	47618.34118	2563601726	1243964535	100
90	47618.34118	67652636.3	1.48005E+12	100
100	47618.34118	31791497934	72253310.78	100

4.

Lambda	Training Set MSE	Average Validation Set MSE	Test Set MSE	% Nonzeros in W
0	86.39934964	119.4808117	6274.898468	35.71428571
10	87.21289866	118.4897645	4217.062208	35.71428571
20	86.51140755	112.8335174	4240.881834	35.71428571
30	85.52263392	112.3110273	2623.376361	35.71428571
40	86.06401836	113.3543393	2625.254931	28.57142857
50	85.46575134	113.5227207	1401.793375	28.57142857
60	85.24094537	113.6623401	572.7114084	28.57142857
70	86.34058631	107.444157	136.8705752	28.57142857
80	80.64371254	92.54167	80.9802504	14.28571429
90	80.64371254	96.21244359	80.9802504	14.28571429
100	80.64371254	98.77074641	80.9802504	14.28571429

5.

Lambda	Training Set MSE	Average Validation Set MSE	Test Set MSE	% Nonzeros in W
0	47.12818851	148.8000759	617.9395216	3.944773176
10	57.50820721	155.1616658	2569.756039	2.465483235
20	62.96598702	142.2007557	2609.222296	1.972386588
30	67.38102392	131.974551	1399.334331	1.775147929
40	72.71469877	126.6942061	1405.19643	1.084812623
50	74.74076328	124.2110991	577.33463	0.887573964
60	75.75310712	110.7531451	140.4890989	0.887573964
70	78.19641101	100.501659	103.9853939	0.690335306
80	75.37466739	94.87195173	81.46595017	0.394477318
90	75.37466739	96.82938555	81.46595017	0.394477318
100	80.36528494	97.01215037	87.85838798	0.394477318

Comparing to Ex.2.3:

- The mean square error, in both the training, validation, and test sets are a lot smaller than in Ex.2.3 The numbers are reasonable, and do not differ much from the lasso regression run on the dataset without the irrelevant features, in Ex.2.4. This implies that lasso regression can handle irrelevant features a lot better than ridge regression, and perform a lot better even if the dataset contains meaningless data.
- Running lasso regression takes a significantly longer time than ridge regression. Although solving the system of equations for ridge regression takes O(n^3) time, vs the O(nd) time it takes to compute **w** once for lasso regression, we must repeat the O(nd) calculations for lasso regression until **w** converges, or the change drops below a certain tolerance, which can often take longer, and can potentially be unbounded.
- **w** is a lot sparser in lasso – with only between 0.4% - 4% of the 1000 entries corresponding to the irrelevant features being nonzero, compared to the full 100% in ridge regression. This implies that lasso was able to “weed out” these irrelevant features, and determine that they were irrelevant to the regression, as opposed to ridge, which treated them as meaningful features to the regression.