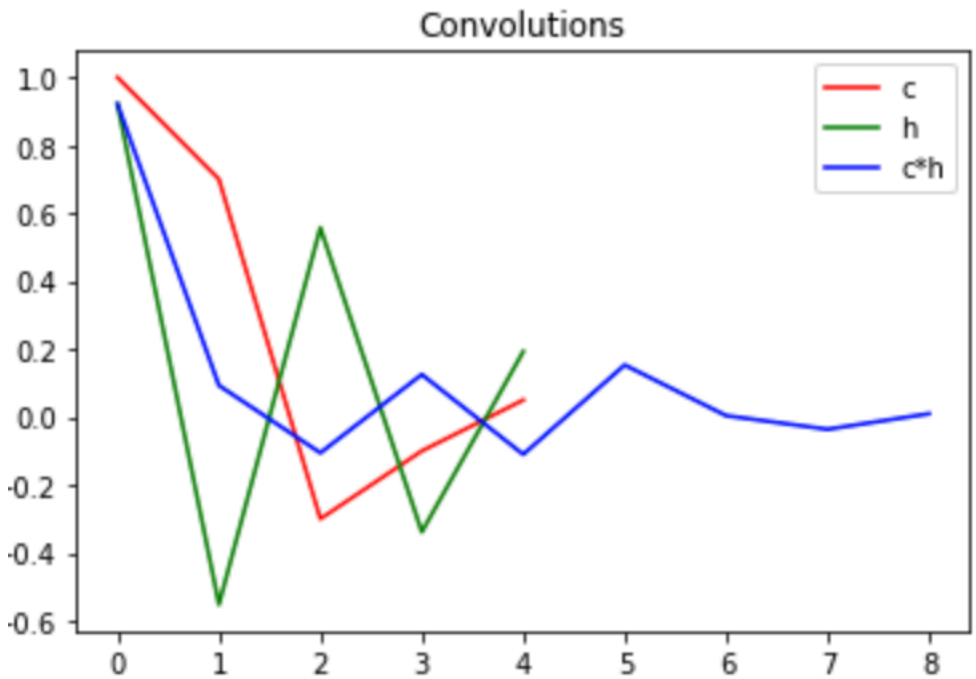
Nicholas Miller
MIS HW 6

12.6
By properties of convolutions $h * c = T(h)c = T(c)h$
Where, $T(c)_{ij} = \begin{cases} c_{i-j+1} & 1 \le i-j+1 \le n \\ 0 & otherwise \end{cases}$

Now the problem has changed to minimize, $\|T(c)h - e_1\|^2$
Thus, we need to find the least squares solution to solve $T(c)^{-1}e_1 = h$

Solution:
h = [0.921, -0.552, 0.557, -0.338, 0.192]



12.12
(a)
$D(u) = \|A^T u\|^2 = \sum_{edges(k,l)}((p_1)_l - (p_1)_k)^2$
$D(v) = \|A^T v\|^2 = \sum_{edges(k,l)}((p_2)_l - (p_2)_k)^2$

Then, $D(u) + D(v) = D(u+v) = \sum_{edges\ (k,l)}((u+v)_l - (u+v)_k)^2 =$
$\sum_{edges\ (k,l)}(u_l + v_l - u_k - v_k)^2 = \sum_{edges\ (k,l)}(p_i - p_j)^2 =$
$\|p_{i1} - p_{j1}\|^2 + \cdots + \|p_{iL} - p_{jL}\|^2$

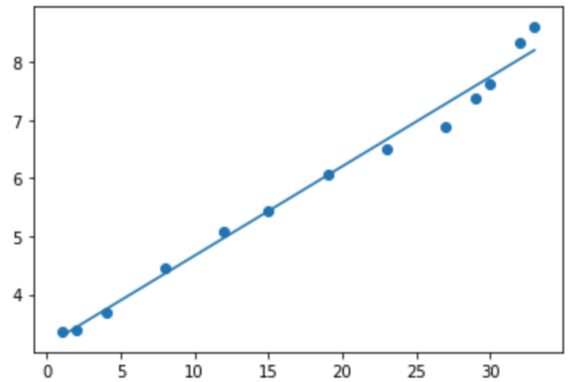
Nicholas Miller
MIS HW 6

13.3

(a)

$\Theta_1 = 3.126 \; and \; \Theta_2 = 0.154$

$rms = 0.041$



(b)

$Log(N_{f(t)}) = 3.126 + 0.154 * f(t), where \; f(t) = t - 1970$

Prediction: 10.056

Actual: $Log(4 * 10^9) = 9.602$

Percent error = 4.7%

(c)

For my model $Log(N_{f(t)})$

With $f(t) = 0$, Prediction is 3.12, where $10^{3.12} = 1318$

With $f(t) = 2.5$, Prediction is 3.51, where $10^{3.51} = 3235$

So, $\frac{N_{2.5}}{N_0} \approx 2$

13.11

Model A: slightly smaller Train RMS

- Might be a slightly over-fit model.
- Low RMS.
- <u>Might</u> be a <u>good model</u>.

Model B: Slightly bigger Train RMS

- Might be slightly underfitting the model.
- High RMS.
- <u>Bad model</u>.

Model C: Significantly bigger Train RMS

- I don't believe the results.
- <u>Results seem fishy</u>.

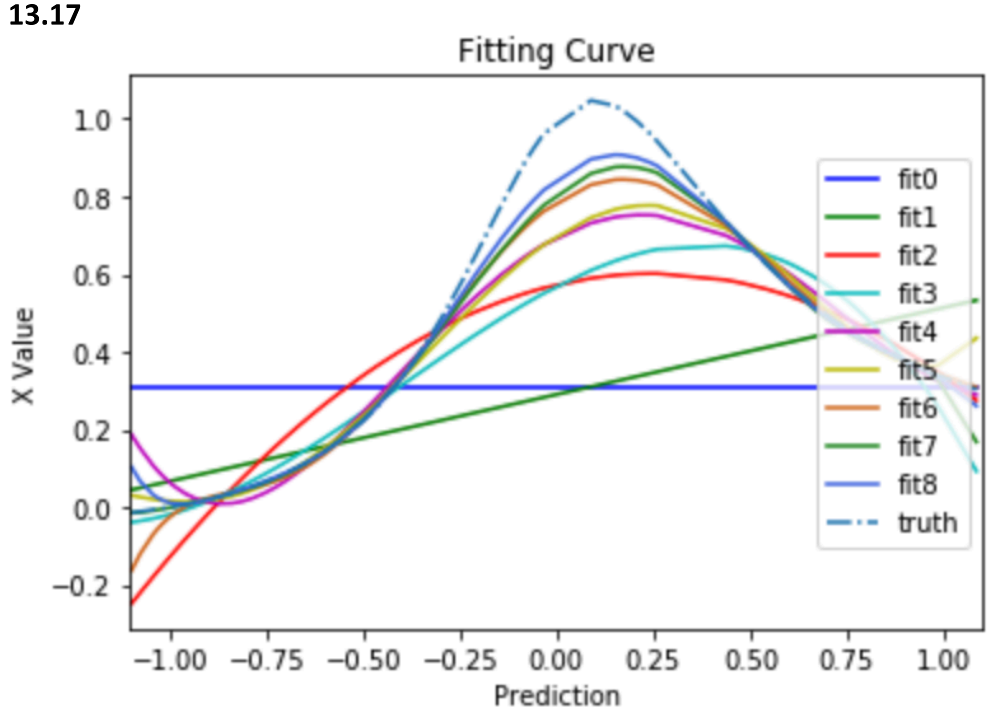
Model D: Significantly smaller Train RMS

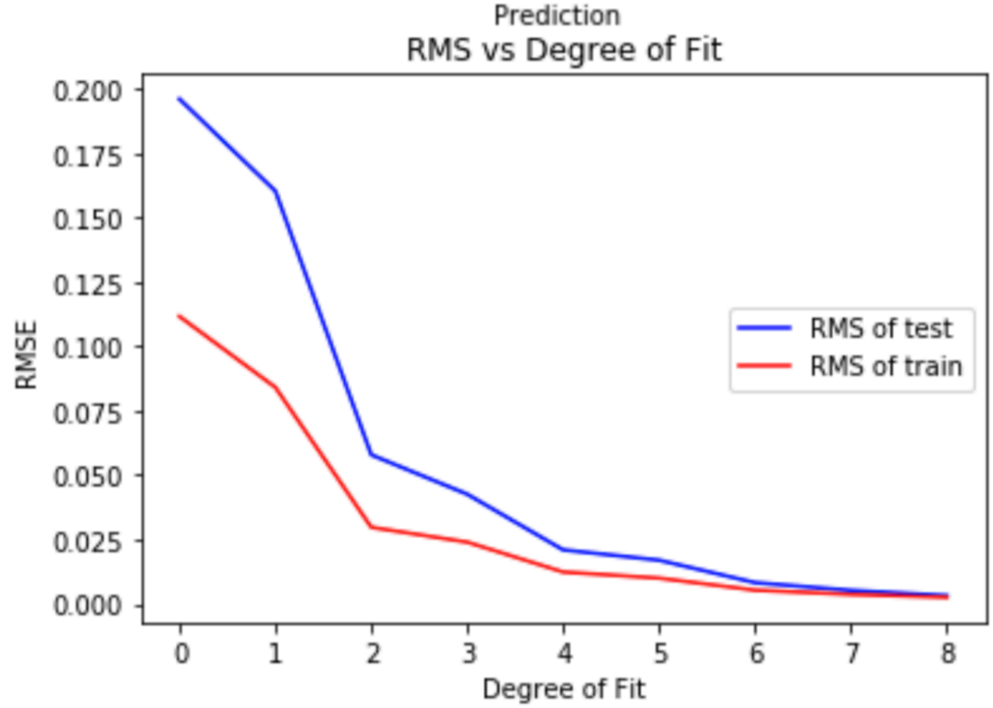
- Definitely overfitting the model.
- Generalizes unseen data.
- <u>Bad model</u>.

Nicholas Miller
MIS HW 6

Model E: Equal Train and Test RMS
- Model performs the same on unseen data.
- Small RMS.
- Great model.

13.17





Nicholas Miller
MIS HW 6

A reasonable value for the degree of the polynomial fit is 4. There is a slight increase in the RMS of the test over the train, but that value doesn't seem to decrease as you complicate the model. At degree 4 the model still extrapolates information well, while not complicating the model too much.

3.22
The point of the competition is to be able to predict unseen data. Since contestants haven't trained on the hidden test set, if their model can perform well on the unseen data then their model might be a good predictor.

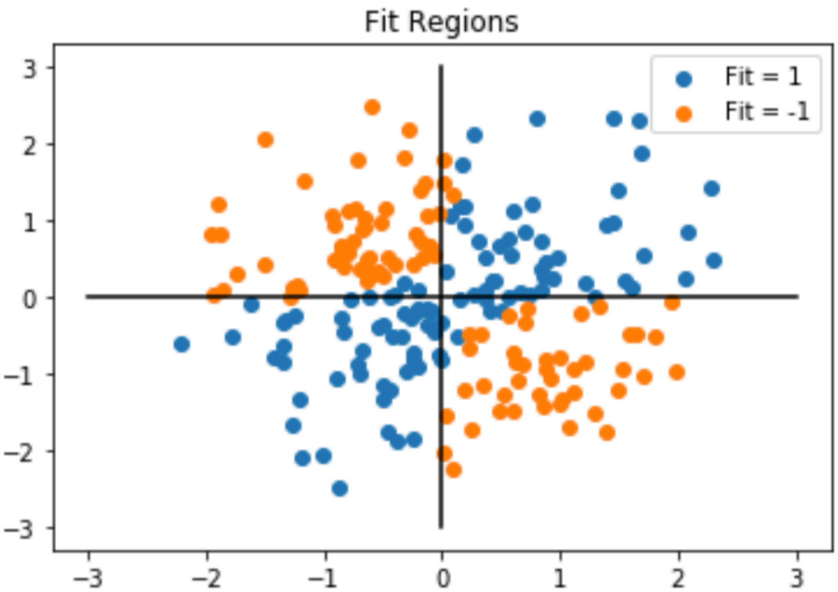
A team would want to run cross validation so that you can determine if your model is over-fitting and that it can accurately predict unseen data. Another thing teams could do is generate new training data from the distribution of the given dataset. Additionally, teams can run feature generation functions to increase accuracy, but this might have the side effect of over complicating your model and actually decreasing accuracy.

If there were no limit to the number of predictions a team could make, then one thing a team could do is submit their prediction, see how their model fairs on the hidden test set and tweak their model to increase accuracy on the test set, and resubmit until they can get the best prediction on the test set.

A simple way for a team to get around the prediction limit would to create a new team once after they have reached the prediction limit and keep submitting predictions under the new team name. They would then claim the team name that had the best results after the contest ended.

Nicholas Miller
MIS HW 6

14.7
Error Rate of Classifier: 0.075



Coefficients of first polynomial:
Θ_1 =0.06604638
Θ_2 =0.01046992
Θ_3 =-0.03628486
Θ_4 =-0.04513321
Θ_5 =0.70481271
Θ_6 =-0.06652015

Coefficient of second polynomial:
$\Theta_1 = 0.68125115$

Comparision:
As we can see from the coefficients above both classifiers place a lot of importance on the constant in front of the $x_1 * x_2$ term. The first polynomial place less weight across all of terms meaning that the classifier determined that they are not important in determining class, but since the model for the first polynomial is more complex the error rate increases.