# CMPT 419/726: Machine Learning (Fall 2018)

Assignment 1: Regression
Name: Anurag Bejju
Student ID: 301369375
Instructor: Greg Mori

5th October 2018

## 1 Question 1: Probabilistic Modeling

.

1. What are the parameters $\mu$ of this distribution? (See PRML Appendix B.)

As stated in PRML Appendix B for Multinomial Distribution, If we generalize the Bernoulli distribution to an K-dimensional binary variable x with components $x \epsilon [0, 1]$ such that $\sum_k x_k$, then we obtain the following discrete distribution.

$$\prod_{k=1}^{K} \mu_k^{x_k}$$

This distribution can be used for this model. Parameters $\mu$ for this model would be:

$$\mu_1 = [\text{Probability for NDP Winning}]$$
$$\mu_2 = [\text{Probability for Liberals Winning}]$$
$$\mu_3 = [\text{Probability for Green Party Winning}]$$

2. What would be the value of the parameters $\mu$ for an election where the outcome is an equal chance of each party winning?

Since there is an equal chance for every party to win this election (provided its fair) then each one will have a $\frac{1}{3}$ probability to win it. Therefore

$$\mu_1 = [\text{Probability for NDP Winning}] = [\tfrac{1}{3}, 0, 0]$$
$$\mu_2 = [\text{Probability for Liberals Winning}] = [0, \tfrac{1}{3}, 0]$$
$$\mu_3 = [\text{Probability for Green Party Winning}] = [0, 0, \tfrac{1}{3}]$$

3. What would be the value of the parameters $\mu$ for an election that is completely "rigged"? E.g. the party currently in power is definitely going to win.

If one party riggs the election, then the prior for that party to win will be 1 and it will be zero for all the others. Therefore

$$\mu_1 = [\text{Probability for NDP Winning if it riggs}] = [1, 0, 0]$$
$$\mu_2 = [\text{Probability for Liberals Winning if it riggs}] = [0, 1, 0]$$
$$\mu_3 = [\text{Probability for Green Party Winning if it riggs}] = [0, 0, 1]$$

4. Specify a prior P ($\mu$) that encodes a belief that one party has rigged the election, but there is an equal chance that it is any of the 3 parties.

This scenario can be represented using Dirac delta function. It is used to model the density of an idealized point mass or point charge as a function equal to zero everywhere except for zero and whose integral over the entire real line is equal to one.
In this scenario, even though we have prior stating that one party has rigged the election, our lack of knowledge about the party that rigged it would lead to $\frac{1}{3}$ probability for each party to win it. That means if we represent this as a Dirac delta function, then the function will jump at 1 if that party riggs and remain zero at all other times. Since we have three scenarios and they can be represented as three Dirac delta function, when we sum it should be one. Therefore

$$P(\mu_1) = \frac{1}{3}$$
$$P(\mu_1) = \frac{1}{3}$$
$$P(\mu_1) = \frac{1}{3}$$

5. Suppose my prior is that the Green Party has completely rigged the election. Assume I see a set of polls where the NDP has the largest share of the vote in each poll. What would be my posterior probability on ?

We are aware that the posterior probability is directly proportional to prior and likelihood. i.e

$$\text{Posterior } \alpha \text{ prior} \cdot \text{likelihood}$$

Since in this case our prior probability is so strong, it makes the likelihood redundant. That means, our posterior would take the shape of our prior we have in this situation. Even though our likelihood says that NDP is getting largest share of the votes in the given sample data set, we have a strong prior stating the elections have been rigged by Green Party. Therefore our posterior would be that Green Party rigged the election.

6. Suppose if party i is elected, they will set university tuition to be ti dollars. Write down an equation for the expected amount tuition will be, given a prior P ($\mu$).

Since Dirichlet distribution is the conjugate prior of the multinomial distribution, we can us it to write down the required equation. The Dirichlet is a multivariate distribution over K random variables (0 $<= k <=$ 1, where k = 1, . . . , K, subject to the constraints

$$0 <= \mu_k <= 1 \text{ and } Dir(\mu|\alpha) = C(\alpha) \prod_{k=1}^{K} \mu_k^{\alpha_k - 1}$$

where $\mu = (\mu_1, ...\mu_n)^T$ and $\alpha = (\alpha_1, ...\alpha_n)^T$
Here our $\mu$ are our parameters which statets the probability of party i winning and $\alpha$ is its corresponding amount of tuition fee we will get in $t_i$ dollars.

Therefore this is the equation representing equation for the amount of tution will be if the party i wins election:

$$\mu = (\mu_1, ...\mu_i)^T \text{ and } T = (t_1, ...t_i)^T$$
$$\text{then}$$
$$Dir(\mu|T) = C(T) \prod_{k=1}^{i} \mu_k^{T_k - 1}$$

## 2 Question 2: Precision Per Datapoint

The log likelihood for regression (Eqn. 3.10 in PRML) assumes an additive Gaussian noise with the same value of variance at every training data point. In some instances, we may wish to have a different value of noise variance at each training data point. This could arise if we have precision estimates at each training data point.

Consider the likelihood function with different precision values corresponding to each data point:

$$p(t|X,w,\beta)=\prod_{n=1}^{N}\mathcal{N}(t_n|w^T\phi(x_n),{\beta_n}^{-1})$$

with precision estimates $\beta_n$ for each training data point. Derive the relation between the log likelihood function $p(t|w,{\beta_n}^{-1})$ and the sum-of-squares error function in this scenario.

Solution: The given likelihood function for N input values X = $(x_1,...,x_n)^T$ and their corresponding target values t = $(t_1,...,t_n)^T$ is

$$p(t|X,w,\beta)=\prod_{n=1}^{N}\mathcal{N}(t_n|w^T\phi(x_n),{\beta_n}^{-1}) \qquad \Leftarrow (2.1)$$

Here we are expressing uncertainty over the value of the target variable t using a probability distribution. For this purpose, we shall assume that, given the value of x, the corresponding value of t has a Gaussian distribution with a mean equal to the value y(x, w) and precision value $\beta=\frac{1}{\sigma^2}$ Since the equation for Gaussian Distribution is :

$$\mathcal{N}(x|\mu,\sigma^2)=\tfrac{1}{\sqrt{2\pi\sigma^2}}e^{-(x-\mu)^2/2\sigma^2} \qquad \Leftarrow (2.2)$$

Representing Equation 2.1 in form of 2.2, we get:

$$p(t|X,w,\beta)=\prod_{n=1}^{N}\tfrac{1}{\sqrt{2\pi{\beta_n}^{-1}}}\ e^{-\frac{\beta_n}{2}\left(t_n-w^T\phi(x_n)\right)^2} \qquad \Leftarrow (2.3)$$

Taking ln on both sides:

$$\ln p(t|X,w,\beta)=\ln\prod_{n=1}^{N}\tfrac{1}{\sqrt{2\pi{\beta_n}^{-1}}}\ e^{-\frac{\beta_n}{2}\left(t_n-w^T\phi(x_n)\right)^2} \qquad \Leftarrow (2.4)$$

Upon using three rules, namely $\ln(ab)=\ln a+\ln b$, $\ln\frac{a}{b}=\ln a-\ln b$ and $\ln x^n=n\ln x$ we can simplify equation 2.4 as:

$$\begin{aligned}
\ln p(t|X,w,\beta)&=\sum_{n=1}^{N}\ln\left(\frac{1}{\sqrt{2\pi{\beta_n}^{-1}}}\ e^{-\frac{\beta_n}{2}\left(t_n-w^T\phi(x_n)\right)^2}\right)\\
&=\sum_{n=1}^{N}\ln\left(\frac{1}{\sqrt{2\pi{\beta_n}^{-1}}}\right)\ +\ln\left(e^{-\frac{\beta_n}{2}\left(t_n-w^T\phi(x_n)\right)^2}\right)\\
&=\sum_{n=1}^{N}\left[-\ln\left(\frac{2\pi}{\beta_n}\right)^{\frac{1}{2}}\ -\left(\frac{\beta_n}{2}\left(t_n-w^T\phi(x_n)\right)^2\right)\right]\\
&=\sum_{n=1}^{N}\left[-\frac{1}{2}\ln\left(\frac{2\pi}{\beta_n}\right)\ -\left(\frac{\beta_n}{2}\left(t_n-w^T\phi(x_n)\right)^2\right)\right]\\
&=\sum_{n=1}^{N}\left[-\frac{1}{2}\ln 2\pi+\frac{1}{2}\ln\beta_n\ -\left(\frac{\beta_n}{2}\left(t_n-w^T\phi(x_n)\right)^2\right)\right]\\
&=-\frac{n\ln 2\pi}{2}+\sum_{n=1}^{N}\left[\frac{1}{2}\ln\beta_n\ -\left(\frac{\beta_n}{2}\left(t_n-w^T\phi(x_n)\right)^2\right)\right]
\end{aligned} \qquad (2.5)$$

Since sum-of-squares error function is defined as:

$$\begin{aligned} E_D(W) &= \frac{1}{2}\sum_{n=1}^{N}(t_n - w^T\phi(x_n))^2 \\ &= \frac{1}{2}\Big[(t_1 - w^T\phi(x_1))^2 + (t_2 - w^T\phi(x_2))^2 + ...... + (t_n - w^T\phi(x_n))^2\Big] \\ &= \frac{1}{2}\Big[E_1(W) + E_1(W) + ... + E_n(W)\Big] \end{aligned} \tag{2.6}$$

We can represent equation 2.5 as:

$$\ln p(t|X, w, \beta) = -\frac{n\ln 2\pi}{2} + \sum_{n=1}^{N}\frac{\ln\beta_n}{2} - \sum_{n=1}^{N}\Big(\beta_n \cdot E_n(W)\Big) \tag{2.7}$$

## 3 Question 3: Training vs. Test Error

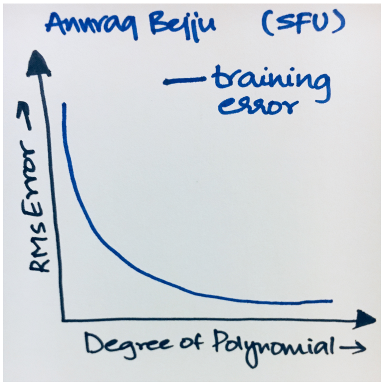
For the questions below, assume that error means RMS (root mean squared error).

1. Suppose we perform unregularized regression on a dataset. Is the validation error always higher than the training error? Explain.

False. Validation error can be lower than the training error if our model generalizes well and fits better for the selected validation set. Also it can happen when the validation data belonged to the type of data that influenced the model the most while training or maybe because the validation cases contained the type of data that the model is good at predicting.

2. Suppose we perform un-regularized regression on a dataset. Is the training error with a degree 10 polynomial always lower than or equal to that using a degree 9 polynomial? Explain.

False. This is not the case all the time. When the data is un-regularized, we cannot determine if the error will go down or up with increase in polynomial. This phenomena can only be observed if the data has been regularized. As you can see in the attached picture, the training error decreases w.r.t degree of polynomial when the data is regularized.



3. Suppose we perform both regularized and unregularized regression on a dataset. Is the testing error with a degree 20 polynomial always lower using regularized regression compared to unregularized regression? Explain.

False. It depends on the regularization lambda being used. If the lambda is too big, we might have a model with high bias which might give more testing error than the error calculated from an unregularized regression model. Flip-side, If the lambda is too small (lets assume = 0), then testing error for both unregularized and regularized regression model will be the same.

Therefore the statement that the testing error with a degree 20 polynomial is always lower using regularized regression compared to unregularized regression is incorrect.

## 4 Question 4: Basis Function Dependent Regularization

In lecture we saw a regularization technique applied to linear regression where all weights in the regression model are regularized in the same fashion (like $L_1$, or $L_2$), and with a common value for $\lambda$. Consider the case where for each weight $w_n$, we have a different trade-off parameter $\lambda_m$, and a choice from among one of $L_1$, or $L_2$ regularizer. Derive the formula of the gradient for the regularized squared error loss function in this scenario.

$$\nabla E(w) = ?$$

(Hint: Let $\mathcal{J}_1$ be the set of indices of basis functions whose weights have $L_1$ regularization, and $\mathcal{J}_2$ be the set of indices of basis functions whose weights have $L_2$ regularization.)

Solution: Regularized Square Error Loss Function can be represented as:

$$\tilde{E}(w) = \frac{1}{2}\sum_{n=1}^{N}(w^T\phi(x_n) - t_n)^2 + \frac{\lambda}{2}\sum_{j=1}^{M}(|w_j|)^q \tag{3.1}$$

For $L_1$ regression or Lasso regression, the q value will be 1 and for $L_2$ regression or Ridge regression, the q value will be 2. Modifying 3.1 equation as per the stated requirements we get:

$$\tilde{E}(w) = \frac{1}{2}\sum_{i=1}^{N}\Big(\big(\sum_{k\epsilon(\mathcal{J}_1,\mathcal{J}_2)} w_k \cdot \phi_k(x_i)\big) - t_i\Big)^2 + \frac{1}{2}\sum_{k\epsilon\mathcal{J}_1}(\lambda_k \cdot |w_k|) + \frac{1}{2}\sum_{k\epsilon\mathcal{J}_2}(\lambda_k \cdot |w_k|^2) \tag{3.2}$$

The gradient of the regularized squared error loss function (3.2) takes the form:

$$\nabla\tilde{E}(w) = -\sum_{i=1}^{N}\Big[(\phi_k(x_i) \cdot \Big(\big(\sum_{k\epsilon(\mathcal{J}_1,\mathcal{J}_2)} w_k \cdot \phi_k(x_i)\big) - t_i\Big)\Big] + \frac{1}{2}\sum_{k\epsilon\mathcal{J}_1}(\lambda_k \cdot \frac{w_k}{|w_k|}) + \sum_{k\epsilon\mathcal{J}_2}(\lambda_k \cdot \frac{w_k}{|w_k|}) \tag{3.3}$$

## 5 Question 5: Regression

In this question you will train models for regression and analyze a dataset. Start by downloading the code and dataset from the website.
The dataset is created from data provided by UNICEF's State of the World's Children 2013 report: http://www.unicef.org/sow
Child mortality rates (number of children who die before age 5, per 1000 live births) for 195 countries, and a set of other indicators are included.

### 5.1 Getting started

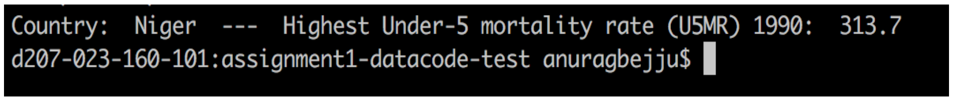
Run the provided script polynomial regression.py to load the dataset and names of coun- tries / features.
Answer the following questions about the data. Include these answers in your report.

1. Which country had the highest child mortality rate in 1990? What was the rate?

Answer: The country with highest child mortality rate in 1990 is Niger and its rate is 313.7
Code:

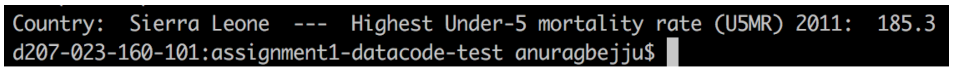
```
import pandas as pd
data = pd.read_csv('SOWC_combined_simple.csv',na_values='_',encoding='latin1')
b = data.loc[data['Under−5_mortality_rate_(U5MR)_1990'].idxmax()]
print('Country:_',b['Countries_and_areas'],'_−−−__Highest_Under−5_mortality_rate_(U5MR)_1990:_',b['
    red↪ Under−5_mortality_rate_(U5MR)_1990'])
```

Output



```
Country:  Niger  ---  Highest Under-5 mortality rate (U5MR) 1990:  313.7
d207-023-160-101:assignment1-datacode-test anuragbejju$
```

2. Which country had the highest child mortality rate in 2011? What was the rate?
Answer: The country with highest child mortality rate in 2011 is Sierra Leone and its rate is 185.3
Output



```
Country:  Sierra Leone  ---  Highest Under-5 mortality rate (U5MR) 2011:  185.3
d207-023-160-101:assignment1-datacode-test anuragbejju$
```

3. Some countries are missing some features (see original .xlsx/.csv spreadsheet). How is this handled in the function assignment1.load unicef data()?

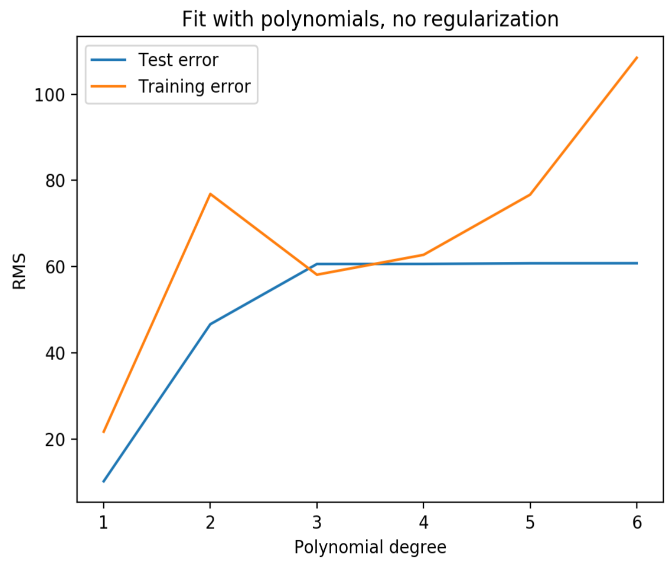
Answer: assignment1.load unicef data() function does pre-processing of data by replacing NaN values (i.e values which are null or not available) with its mean of that column.

For the rest of this question use the following data and splits for train/test and cross-validation.

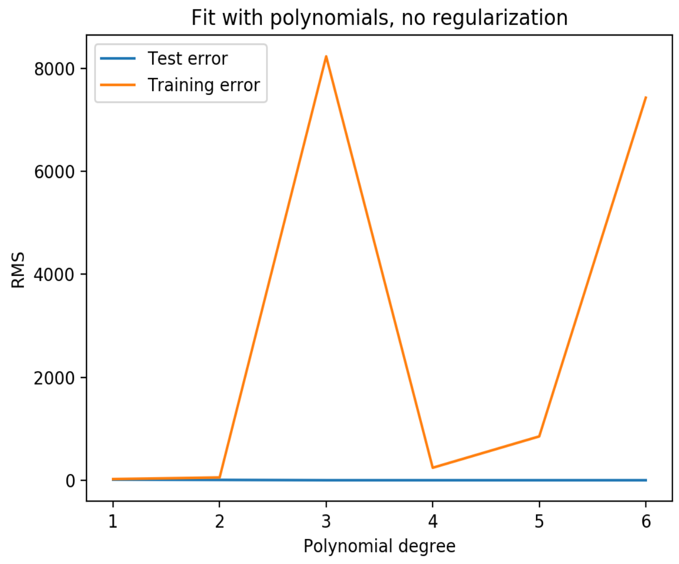
- Target value: column 2 (Under-5 mortality rate (U5MR) 2011)1.
- Input features: columns 8-40.
- Training data: countries 1-100 (Afghanistan to Luxembourg).
- Testing data: countries 101-195 (Madagascar to Zimbabwe).
- Cross-validation: subdivide training data into folds with countries 1-10 (Afghanistan to Aus- tria), 11-20 (Azerbaijan to Bhutan), ... . I.e. train on countries 11-100, validate on 1-10; train on 1-10 and 21-100, validate on 11-20, ...

## 5.2 Polynomial Regression

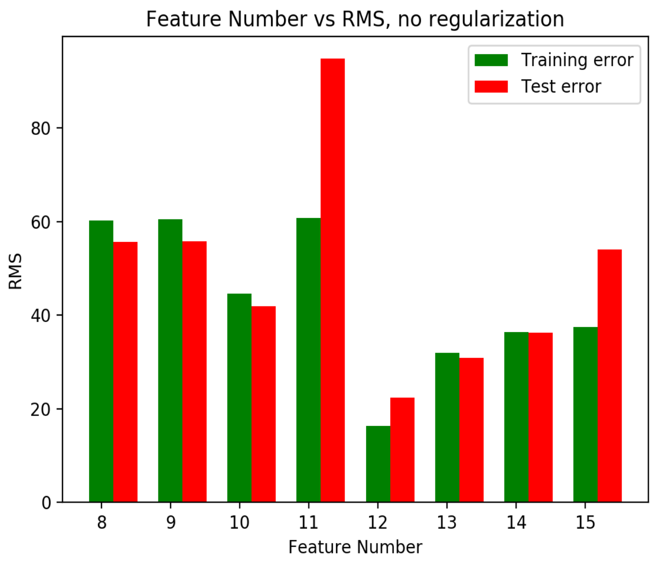
Implement linear basis function regression with polynomial basis functions. Use only monomials of a single variable $(x_1, {x_1}^2, {x_2}^2)$ and no cross-terms $(x_1.x_2)$. Perform the following experiments:
1. Create a python script polynomial regression.py for the following.
Fit a polynomial basis function regression (unregularized) for degree 1 to degree 6 polynomials. Plot training error and test error (in RMS error) versus polynomial degree.
Put this plot in your report, along with a brief comment about what is "wrong" in your report.
Answer: Since each feature has data in various ranges and is not normalized, we can expect such erratic spikes in out training and testing error.



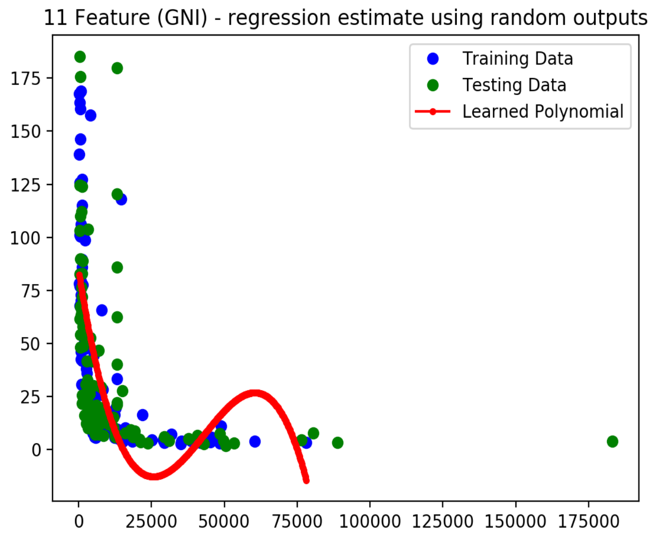
Normalize the input features before using them (not the targets, just the inputs x). Use assignment1.normalize data().
Run the code again, and put this new plot in your report.

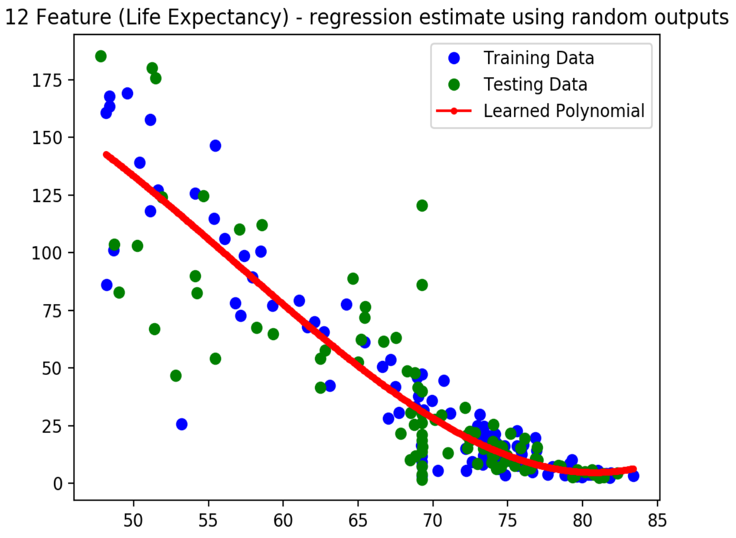


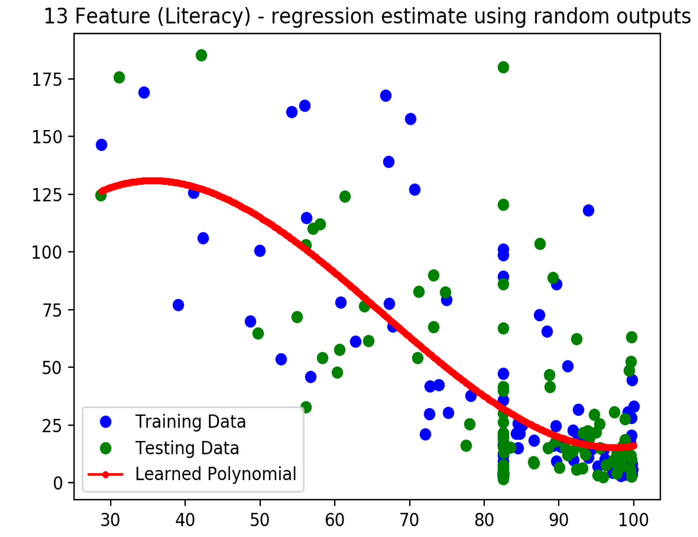
2. Create a python script polynomial regression 1d.py for the following. Perform regression using just a single input feature. Try features 8-15 (Total population - Low birthweight). For each (un-normalized) feature fit a degree 3 polynomial (unregularized).
Plot training error and test error (in RMS error) for each of the 8 features. This should be a bar chart (e.g. use matplotlib.pyplot.bar()).
Put this bar chart in your report.



The testing error for feature 11 (GNI per capita) is very high. To see what happened, pro- duce plots of the training data points, learned polynomial, and test data points. The code visualize 1d.py may be useful.
In your report, include plots of the fits for degree 3 polynomials for features 11 (GNI), 12 (Life expectancy), 13 (literacy).

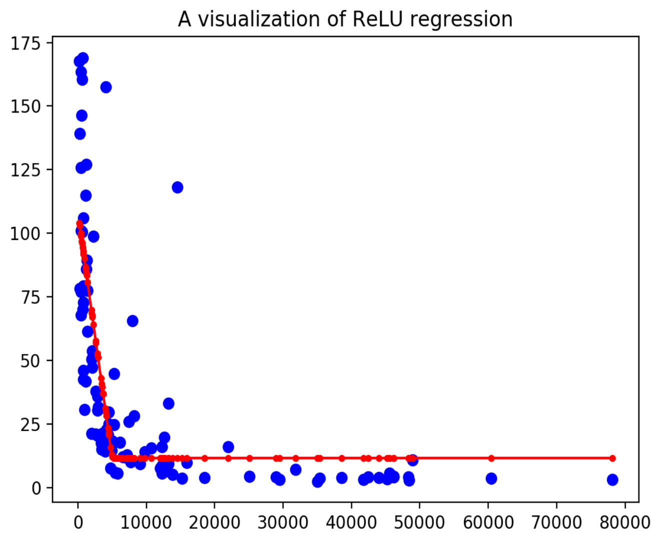




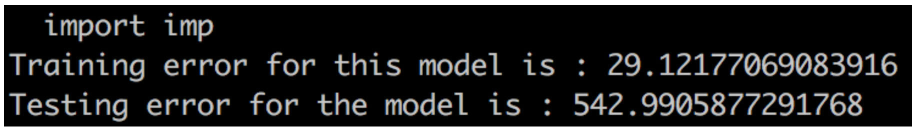


## 5.3 ReLU Basis Function

1. Create a python script relu regression.py for the following. Implement regression using a modified version of ReLU basis function for a single input feature. Mathematically, ReLU is defined as $f(x) = \max(0, x)$. For this part, use the modified ReLU defined as $f(x) = \max(0, g(x))$, where $g(x) = -x + 5000$ Include a bias term. Use unnormalized features.
Fit this regression model using feature 11 (GNI per capita).
In your report, include a plot of the fit for feature 11 (GNI).



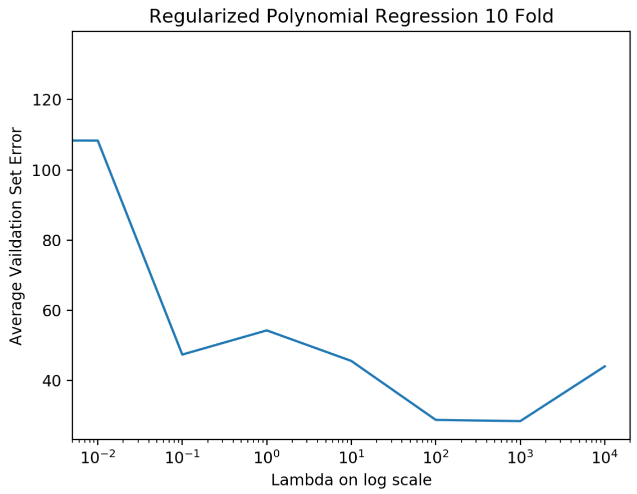
In your report, include the training and testing error for this regression model.



```
  import imp
Training error for this model is : 29.12177069083916
Testing error for the model is : 542.9905877291768
```

## 5.4 Regularized Polynomial Regression

1. Create a python script polynomial regression reg.py for the following.
Implement $L_2$-regularized regression. Fit a degree 2 polynomial using $\lambda = [0, .01, .1, 1, 10, 10^2, 10^3, 10^4]$. Use normalized features as input. Use 10-fold cross-validation to decide on the best value for
$\lambda$. Produce a plot of average validation set error versus $\lambda$. Use a matplotlib.pyplot.semilogx plot, putting $\lambda$ on a $Log_2 scale$.
Put this plot in your report, and note which $\lambda$ value you would choose from the cross- validation.
Answer: I would select a $\lambda$ value between $10^2$ and $10^3$ as the average validation set error w.r.t lamda is least in this case.


Regularized Polynomial Regression 10 Fold
Average Vaildation Set Error
10⁻²
10⁻¹
10⁰
10¹
10²
10³
10⁴
Lambda on log scale