1 Probabilistic Modeling

Solution.

1. We can define μ_1, μ_2 and μ_3 as the probability of NDP, Liberals and Green Party winning the election, respectively, where $\mu_1, \mu_2, \mu_3 \in [0,1]$ and μ_1+μ_2+μ_3=1. Hence, our parameter $\boldsymbol{\mu}$ would be $\boldsymbol{\mu} = (\mu_1, \mu_2, \mu_3)$. However, to make our model easier, we are going to use the condition $\mu_1 + \mu_2 + \mu_3 = 1$ to reduce our number of variables from 3 to 2: we only need μ_1 and μ_2 because μ_3 is automatically computed as $1 - \mu_1 - \mu_2$.

2. If the outcome is equal chance of each party winning, the probability of each party winning would be equal. Also, μ_1+μ_2+μ_3=1. Therefore, μ_1=1/3, μ_2=1/3 and μ_3=1/3.

3. The party that has rigged the election will win. For example, if party NDP has rigged the election, then μ_1 will be 1 and μ_2 and μ_3 will be 0.

4. In a completely rigged election, we want exactly one of μ_1, μ_2, μ_3 be equal to 1 and the other two must be zero. Therefore, it makes sense to definte the prior probability $P(\boldsymbol{\mu})$ as
$$P(\boldsymbol{\mu}) = \begin{cases} \delta(0)/3, & \text{if } (\mu_1, \mu_2, \mu_3) = (1,0,0) \text{ or } (0,1,0) \text{ or } (0,0,1), \\ 0, & \text{otherwise.} \end{cases}$$

5. Our prior knowledge is that the third party (Green Party) has rigged the election. So,
$$P(\boldsymbol{\mu}) = \begin{cases} \delta(0), & \text{if } (\mu_1, \mu_2, \mu_3) = (0,0,1), \\ 0, & \text{otherwise.} \end{cases}$$
Let $\mathcal{D}$ be the set of polls where NDP (party 1) has the largest share of votes. If $(\mu_1, \mu_2) \neq (0,0)$, then $P(\boldsymbol{\mu}) = 0$ and so
$$P(\boldsymbol{\mu}|\mathcal{D}) = \frac{P(\mathcal{D}|\boldsymbol{\mu}) \times \overbrace{P(\boldsymbol{\mu})}^{=0}}{P(\mathcal{D})} = 0.$$
On the other hand, if $(\mu_1, \mu_2) = 0$, we have $P(\boldsymbol{\mu}) = 1$ and hence
$$P(\boldsymbol{\mu} = (0,0,1)|\mathcal{D}) = \frac{P(\mathcal{D}|\boldsymbol{\mu} = (0,0,1)) \times \overbrace{P(\boldsymbol{\mu})}^{=1}}{P(\mathcal{D})} = \frac{P(\mathcal{D}|\boldsymbol{\mu} = (0,0,1))}{P(\mathcal{D})}.$$
Notice that
$$\begin{aligned} P(\mathcal{D}) &= \int_0^1 \int_0^1 P(\mathcal{D}|\boldsymbol{\mu}) P(\boldsymbol{\mu})\, d\mu_1\, d\mu_2 \\ &= \int_0^1 \int_0^1 P(\mathcal{D}|\boldsymbol{\mu} = (0,0,1)) P(\boldsymbol{\mu} = (0,0,1))\, d\mu_1\, d\mu_2 \qquad \text{(by definition of } P(\boldsymbol{\mu})) \\ &= P(\mathcal{D}|\boldsymbol{\mu} = (0,0,1)). \end{aligned}$$
Hence, in the case when $(\mu_1, \mu_2) = 0$, we have
$$P(\boldsymbol{\mu} = (0,0,1)|\mathcal{D}) = \frac{P(\mathcal{D}|\boldsymbol{\mu} = (0,0,1))}{P(\mathcal{D})} = 1.$$
To make the long story short, the posterior probability is 0 when $(\mu_1, \mu_2) \neq (0,0)$ and 1 otherwise.

6. Let us define tuition by T. Since the third variable (μ_3) is uniquely determined from μ_1 and μ_2 (because $1 + \mu_1 + \mu_2 = 1$), we can write
$$\mathbb{E}[T|\boldsymbol{\mu}] = \mu_1 t_1 + \mu_2 t_2 + (1 - \mu_1 - \mu_2) t_3.$$
Then,
$$\begin{aligned} \mathbb{E}_\mu[T] &= \int_0^1 \int_0^1 P(\boldsymbol{\mu}) \mathbb{E}[T|\boldsymbol{\mu}]\, d\mu_1\, d\mu_2 \\ &= \int_0^1 \int_0^1 P(\boldsymbol{\mu}) \left(\mu_1 t_1 + \mu_2 t_2 + (1 - \mu_1 - \mu_2) t_3\right) d\mu_1\, d\mu_2. \end{aligned}$$

2 Precision Per Datapoint

Solution. The likelihood of data $\boldsymbol{t} = \{t_j\}$ using the given function:

$$\boldsymbol{p}(\boldsymbol{t} \mid \boldsymbol{X}, \boldsymbol{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n \mid \boldsymbol{w}^{\boldsymbol{T}} \phi(\boldsymbol{x}_n), \beta_n^{-1}).$$

with precision estimates β_n for each training data point. So, the log-likelihood is

$$\begin{aligned} \ln \boldsymbol{p}(\boldsymbol{t} \mid \boldsymbol{X}, \boldsymbol{w}, \beta) &= \ln \prod_{n=1}^{N} \frac{\sqrt{\beta_n}}{\sqrt{2\pi}} \exp\left(-\frac{\beta_n}{2}\left[t_n - \boldsymbol{w}^{\boldsymbol{T}} \phi(\boldsymbol{x}_n)\right]^2\right) \\ &= \underbrace{-\frac{N}{2}\ln(2\pi) + \frac{1}{2}\sum_{n=1}^{N} \ln \beta_n}_{\text{Constant w.r.t. } \boldsymbol{w}} - \underbrace{\frac{1}{2}\sum_{n=1}^{N} \beta_n \left[t_n - \boldsymbol{w}^{\boldsymbol{T}} \phi(\boldsymbol{x}_n)\right]^2}_{\text{Squared Error}} \end{aligned}$$

3 Training vs. Test Error

Solution.

1. No. It depends on the data. Validation data might be very close to the regression curve and we might have a lower validation error than training error.
2. Yes. This is true because degree 10 polynomial contains degree 9 polynomial. So, accuracy of degree 10 polynomial is more or equal to accuracy of degree 9 polynomial.j
3. No. It depends on the testing data. Our testing data might be very closer to unregularized regression in degree 20 polynomial than regularized regression . So the testing error might be lower in unregularized regression.

4 Basis Function Dependent Regularization

Solution. Let $\mathcal{I}_1$ and $\mathcal{I}_2$ be

$$\begin{aligned} \tilde{E}(\boldsymbol{w}) &= \frac{1}{2}\sum_{n=1}^{N} \left[t_n - y(x_n, \boldsymbol{w})\right]^2 + \sum_{w_j \in \mathcal{I}_1} \frac{\lambda_j}{2}|w_j| + \sum_{w_j \in \mathcal{I}_2} \frac{\lambda_j}{2}|w_j|^2 \\ &= \frac{1}{2}\sum_{n=1}^{N} \left[t_n - \boldsymbol{w}^{\boldsymbol{T}} \boldsymbol{\phi}(\boldsymbol{x}_n)\right]^2 + \sum_{w_j \in \mathcal{I}_1} \frac{\lambda_j}{2}|w_j| + \sum_{w_j \in \mathcal{I}_2} \frac{\lambda_j}{2}|w_j|^2. \end{aligned}$$

Taking the derivative with respect to w_i, we find that

$$\frac{\partial \tilde{E}(\boldsymbol{w})}{\partial w_i} = \sum_{n=1}^{N} \left(t_n - \boldsymbol{w}^{\boldsymbol{T}} \boldsymbol{\phi}(\boldsymbol{x}_n)\right) \phi_i(\boldsymbol{x}_n) + \sum_{\substack{w_j \in \mathcal{I}_1 \\ w_j \neq 0}} \frac{\lambda_j}{2} \frac{|w_j|}{w_j} + \sum_{w_j \in \mathcal{I}_2} \lambda_j w_j.$$

So, in vector form:

$$\nabla \tilde{E}(\boldsymbol{w}) = \sum_{n=1}^{N} t_n \boldsymbol{w}^{\boldsymbol{T}} \boldsymbol{\phi}(\boldsymbol{x}_n)^T - \boldsymbol{w}^{\boldsymbol{T}}$$

5 Regression

5.1 Getting started

Solution.

1. "Niger" had the highest child mortality in 1990 with the rate of 313.7.
2. "Sierra Leone" had the highest child mortality rate in 2011 with the rate of 185.3.
3. This is handled by finding places with missing values (NaN values) and replace them with mean values (mean of values of the corresponding column). This line calculates the mean value:

   `mean_vals = np.nanmean(values, axis=0)`.

   This line indentifies the indices of missing values:

   `inds = np.where(np.isnan(values))`.

   This line replaces NaN values with mean value:

   `values[inds] = np.take(mean_vals, inds[1])`.

4.

5.2 polynomial regression

Solution. Part 1: Figure 1 shows traning and testing error versus polynomial degree in unregularized regression where the data is not normalized. However, the problem with this regression is that, the training error is increasing after degree 3 and the reason is using unnormalized data. We do not have this problem in figure 2 where the data is normalized.

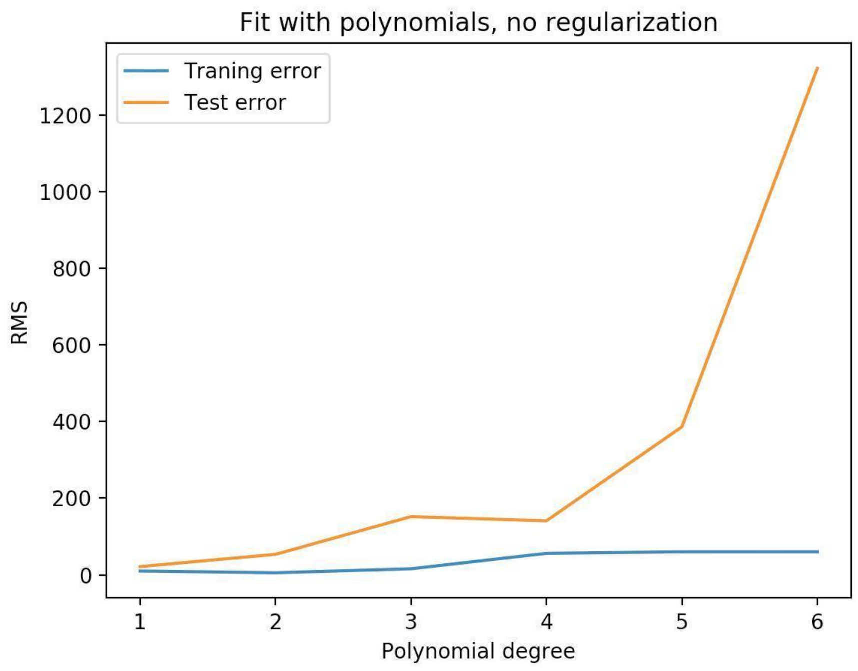

Figure 1: Plot of training error and test error (in RMS error) versus polynomial degree in unnormalized regression.

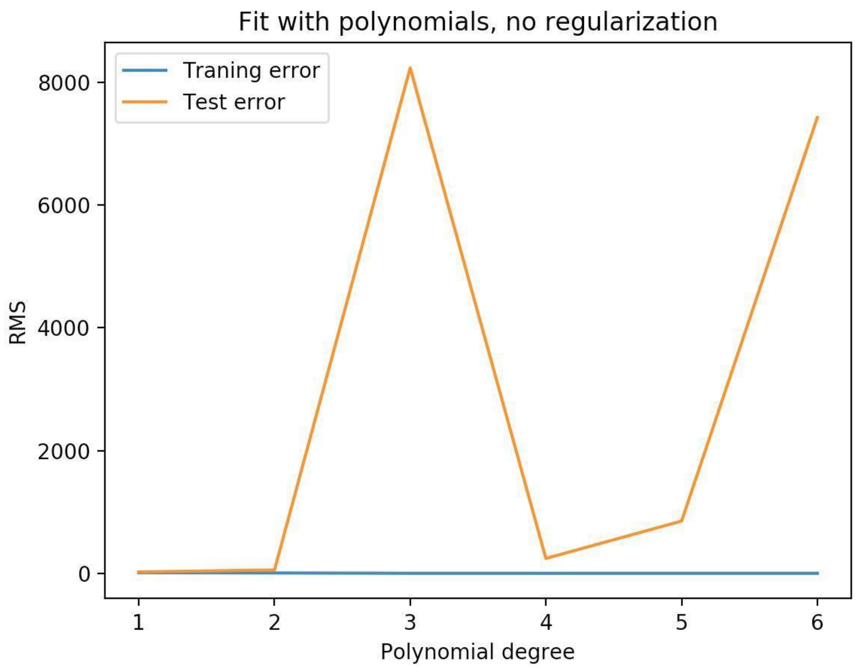

Figure 2: Plot of training error and test error (in RMS error) versus polynomial degree in normalized regression.

Part 2: In figure 3 you can see the plot of training and testing error for features $8 - 15$. The testing error for feature 11 is very high. We can observe the reason for this high testing error in Figure 4. The reason is the noisy testing point which is too far from the other data points. You can also see the predicted curve over training and testing points for life expectancy and literacy in figures 5 and 6.

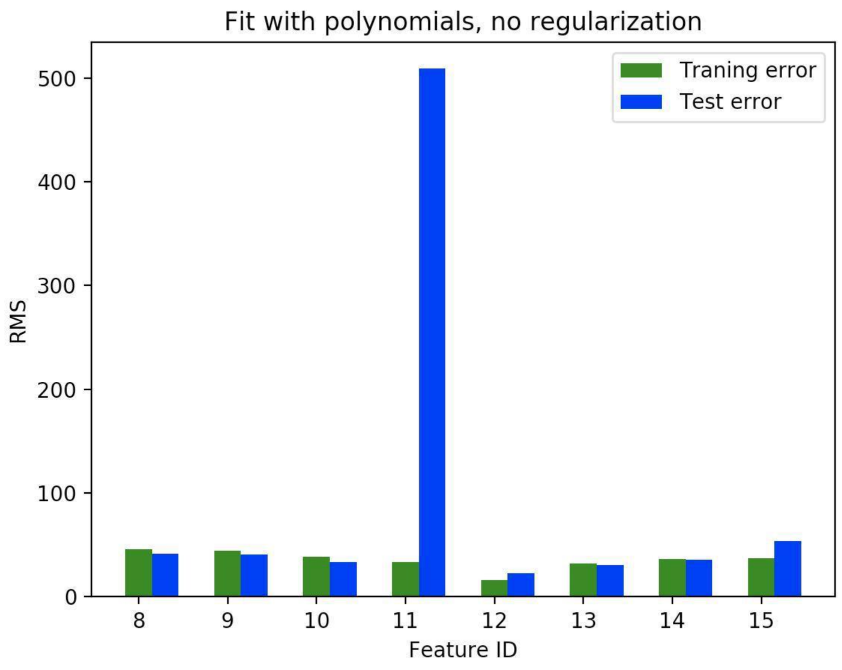

Figure 3: Plot of training and testing error for 8 features.

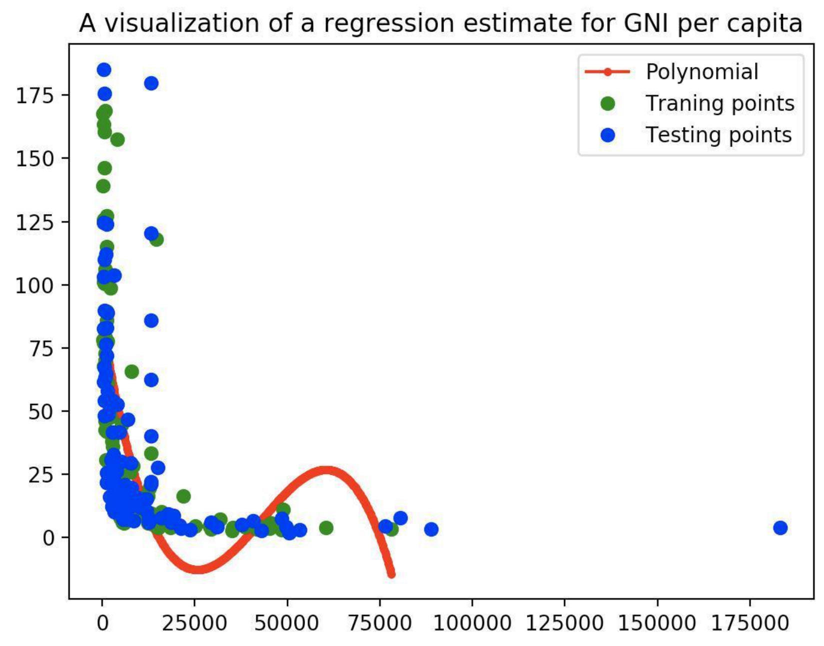

Figure 4: Plot of learned polynomiad, training data and testing data for GNI per capital.

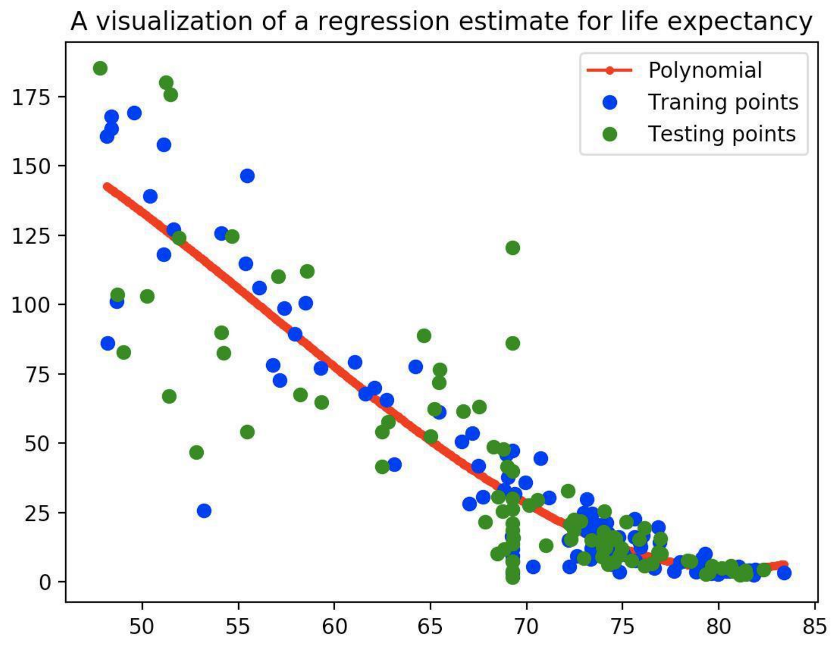

Figure 5: Plot of learned polynomiad, training data and testing data for life expectancy.

5.3 ReLU Basis Function

Figure 7 shows the ReLU regression for GNI per capita over training points. Training error is: 29.1217706908 and testing error is: 34.223585775.

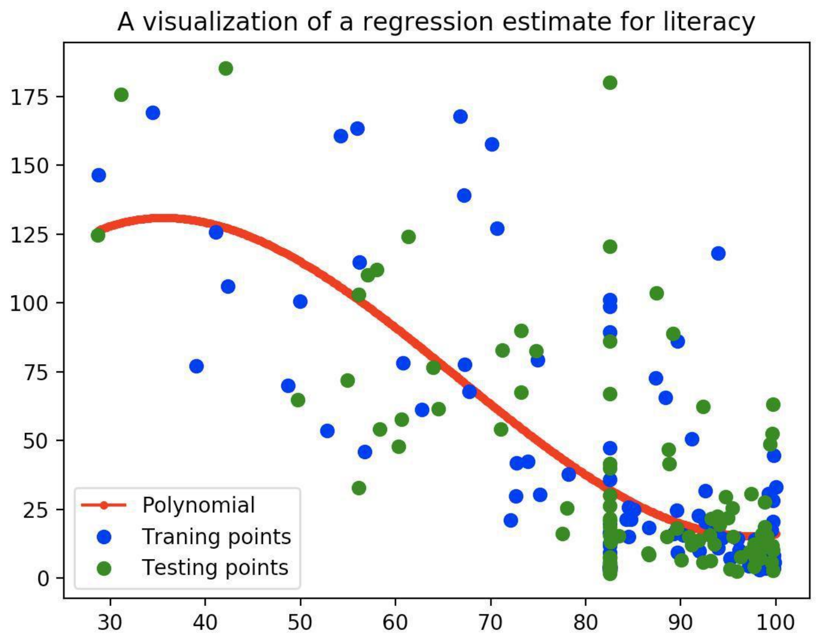

Figure 6: Plot of learned polynomiad, training data and testing data for literacy.

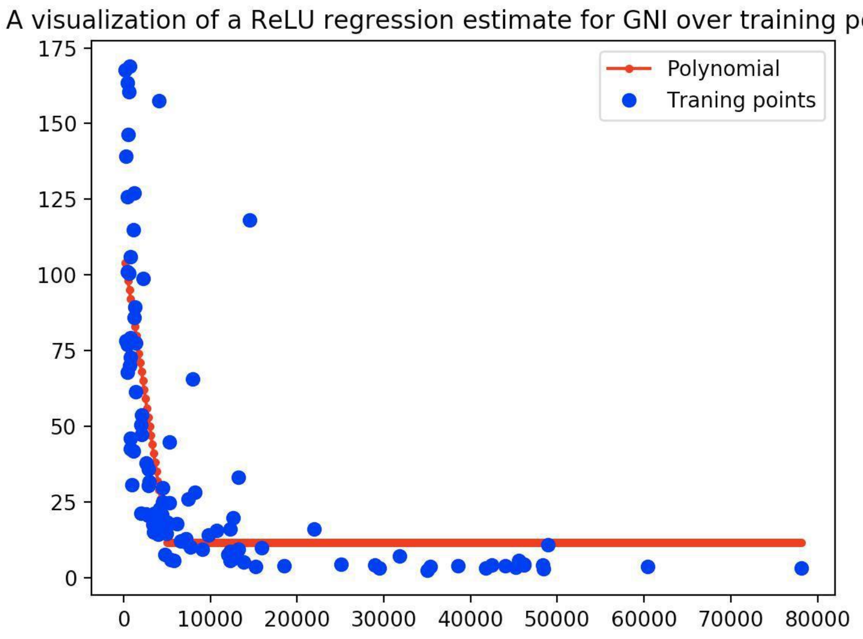

Figure 7: Plot of ReLU regression polynomial for GNI per capital

5.4 Regularized Polynomial Regression

Figure 8 shows cross validation error in regularized regression for different values of lambda. As you can see, the best value for lambda is 1000.

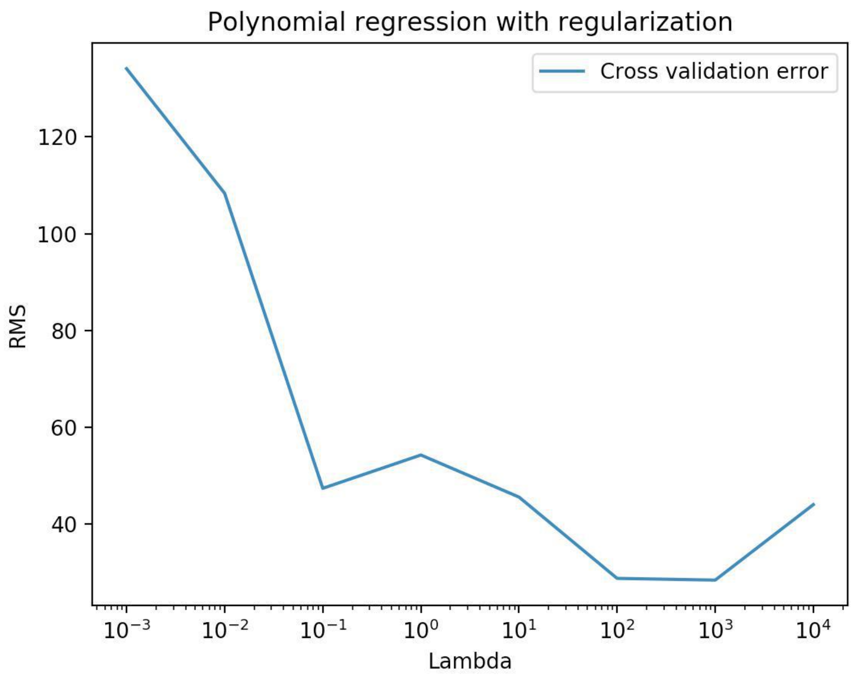

Figure 8: Plot of cross validation error versus different values for lambda