Adjustments and Regularization

Adjustments and Regularization

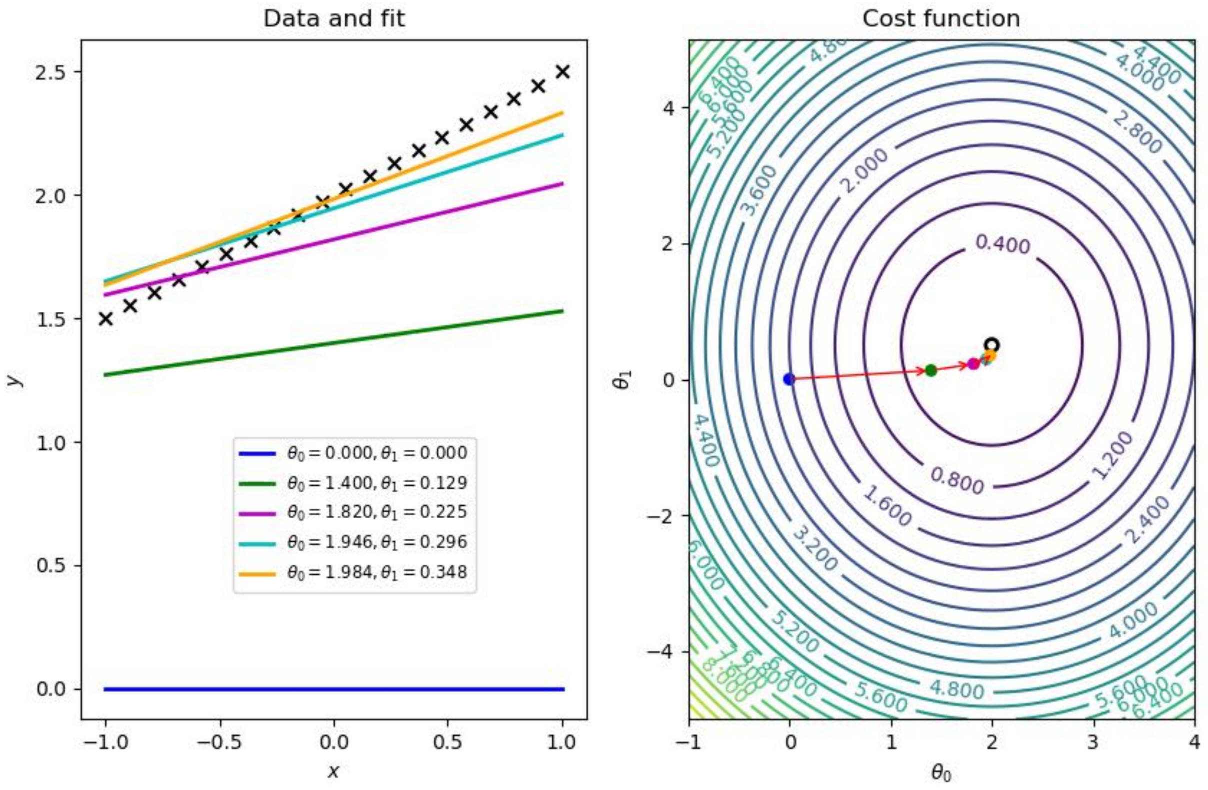
- Now that there are multiple features on a regression, the following may occur:
  - Each feature is on a different scale
    - Feature 1 ranges from 1 to 5 (# beds)
    - Feature 2 ranges from 500 to 8,000 (Square foot)
    - This doesn’t mean that square footage is a better predictor (more important when predicting the output) compared to the # of bedrooms

Adjustments

- There are two main tools to solve the scale problem:
  - Feature Scaling
  - Feature Mean Normalization

Feature Scaling

- Why is it even needed?
  - It is not, but it usually helps to converge faster
- Linear Regression cost function will always have a “bowl” shape
- Every time we do an iteration using gradient descent we get closer to the center of the “bowl”

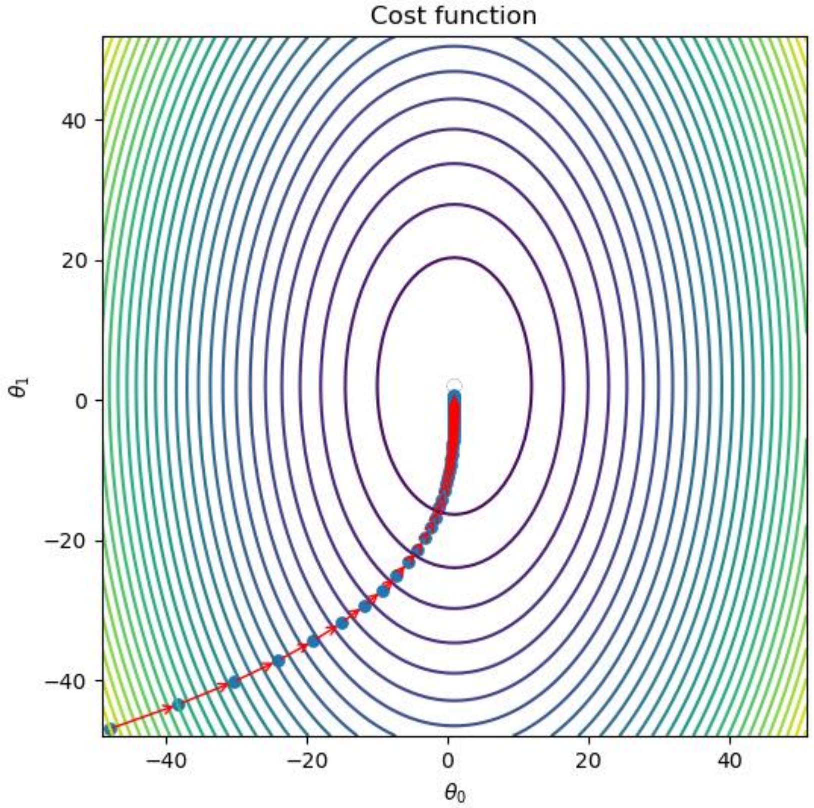


Feature Scaling

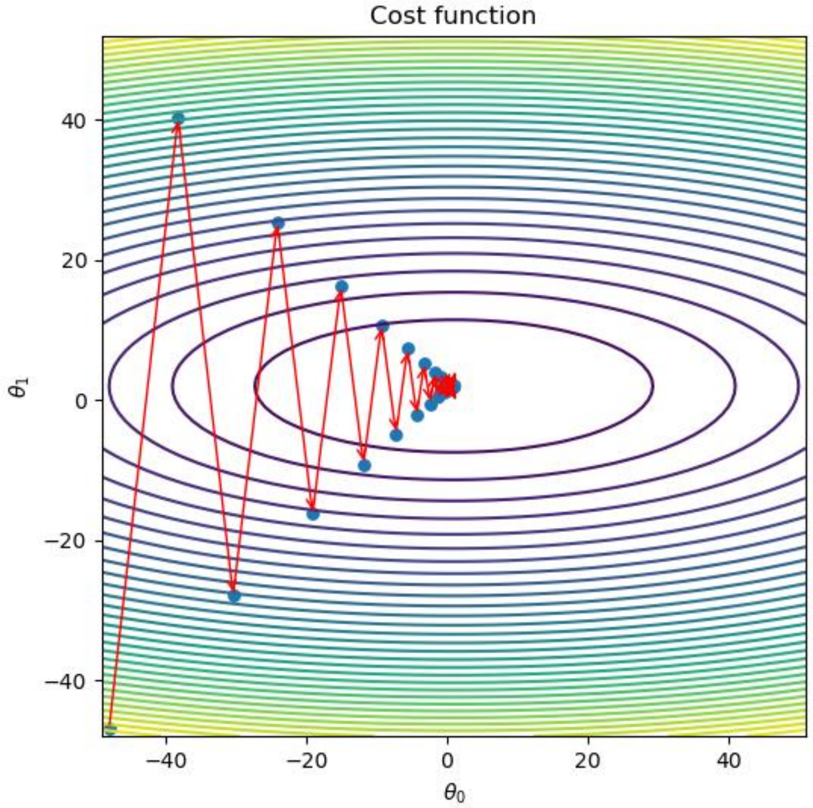
- What happens with that bowl when feature values are very different range wise?
  - It will get squashed/stretched
  - A theta will probably have a big weight and the other one will be really small
  - That may affect the learning

Feature Scaling

But it still converges…
Let is just iterate more times



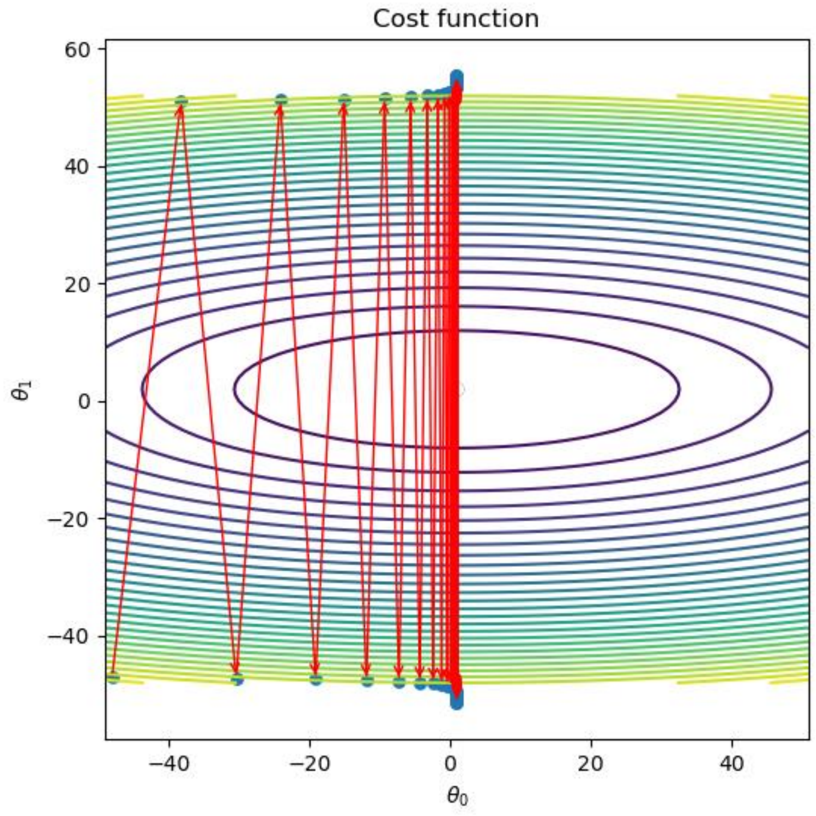
Feature range: [-1,1]
α: 0.2
$h_\theta(x) = y = \theta_0 + \theta_1 x$



Feature range: [-5,5]
α: 0.2
$h_\theta(x) = y = \theta_0 + \theta_1 x$

Feature Scaling

- Not the case always
- That gradient descent is not going downwards
- Will tend to infinite values for θs



Feature range: [-5.3,5.3]

α: 0.2

$$h_\theta(x) = y = \theta_0 + \theta_1 x$$

Adjustments

- Goal:
  - Have the values of every x (features) in one of the following ranges:

$$-1 \leq x \leq 1$$
$$-0.5 \leq x \leq 0.5$$
$$0 \leq x \leq 1$$

Min-Max Scaling

- Compute the range of values (maximum – minimum)
- Divide each value by the range and subtract the minimum

$$x = \frac{x - min_x}{max_x - min_x}$$

- What is the range of the output?

$$0 \leq x \leq 1$$

- This is applied to each feature independently!

Min-Max Scaling

Data	800	1000	830	910	980	1510	990	890	820

$$x = \frac{x - min_x}{max_x - min_x} = \frac{x - 800}{1510 - 800} = \frac{x - 800}{710}$$

Data	800	1000	830	910	980	1510	990	890	820
Min-Max	0	0.28	0.04	0.15	0.25	1	0.27	0.13	0.03

- What if we remove the value max value of the feature from the dataset?

Data	800	1000	830	910	980	X	990	890	820
Min-Max	0	1	0.15	0.55	0.9	X	0.95	0.45	0.1

Feature Mean Normalization

- Features are not normalized.
  - Mean is not 0
    - A house's square footage is 500 to 8,500
    - Mean = 4,500
  - Why normalize?
    - Normalizing allows θ_0 to be the predicted output (y) when all predictor values (x_1, x_2, x_3 etc.) are set to their means instead of 0
- Compute the mean of a feature
- Compute the range of values (maximum – minimum)

Feature Mean Normalization

- Compute the mean of a feature
  - Sum all values per features
  - Divide by the number of samples
- Compute the range of values (maximum – minimum)
- Subtract the mean from each value

$$x = \frac{x - \mu}{max_x - min_x}$$

- What is the range of the output?

$$-1 \leq x \leq 1$$

Feature Mean Normalization

Data	800	1000	830	910	980	1210	990	890	820

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)} = \frac{800 + 1000 + 830 + 910 + 980 + 1210 + 990 + 890 + 820}{9} \approx 936.66$$

$$x = \frac{x - \mu}{max_x - min_x} = \frac{x - 936.66}{1210 - 800} = \frac{x - 936.66}{410}$$

Data	800	1000	830	910	980	1210	990	890	820
FMN	-0.33	0.15	-0.26	-0.07	0.11	0.67	0.13	-0.11	-0.28

Data	800	1000	830	910	980	X	990	890	820
FMN	-0.51	0.49	-0.36	0.04	0.39	X	0.44	-0.06	-0.41

Adjustment

- Are the ranges of Feature Scaling and Mean Normalization guaranteed?
  - Yes
  - Always?
    - Only for the training dataset…
- While training we only have access to some datapoints, not every possible datapoint
  - Future test data might be out of the training data range

Adjustment Implementation

- When should we apply this scaling on the implementation?
  - Everytime data is passed?
    - Isn’t this wasteful?
  - Once for the training dataset?
    - What about the test dataset?
- Considerations
  - What data do we need to apply normalization?
    - Min and Max (Range)
    - Average (for FMN)
  - Should we know about ALL datapoints beforehand?

Recall on Underfitting

- Underfitting
  - Model has fewer parameters than needed/justified by the data
  - Curve is much simpler than what it needs to be
  - Example: Trying to fit a line into points generated from a quadratic equation
  - Won’t be able to predict future values accurately

Recall on Underfitting

- Common underfitting scenarios?
  - Happens when model does not have enough information to predict the output
  - Complexity is not enough, need to increment the complexity
  - Data is completely unrelated (cannot find a correlation)

Recall on Overfitting

- Overfitting
  - Model has more parameters than needed/justified by the data
  - The curve of a high order polynomial can twist and adjust in order to pass through more samples
    - Or at least pass closer to them
    - $J(\theta)$ will be close to 0
  - Example:
    - Training points closer to $y = ax + b$
    - Model is trying to fit a quadratic or cubic curve
  - Won't be able to predict future values accurately

Recall on Overfitting

- Common overfitting scenarios
  - Few training samples, or too many features (or both)
  - Model is actually memorizing training data, not generalizing it
  - Need to reduce the complexity or remove features
  - Regularization

Regularization

- Goal: Avoid overfitting
- Not enough samples:
  - Not an algorithm related issue, cannot do much about that
- Using too many features increases the possibility of overfitting
  - Learning algorithm might put large emphasis on unimportant features
  - Solution: Eliminate features!
  - Which ones?

Regularization

- Higher order polynomials contribute more to overfitting
  - $y = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2 + \theta_3 x_3^4 + \theta_4 x_1^2 x_3^2$
    - "Twistier" curve
    - Assume $x_1 = 5$, $x_2 = 7$, $x_3 = 9$

      $y = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2 + \theta_3 x_3^4 + \theta_4 x_1^2 x_3^2$

      Becomes

      $y = \theta_0 + 5\theta_1 + 49\theta_2 + 6561\theta_3 + 2025\theta_4$
  - Fitted curve needs to be as simple as possible
  - Still providing reliable predictions on unseen data

Regularization

- We want:
  - Keep all the features
  - Reduce the effect of higher order polynomials
    - Don’t necessarily remove them completely
  - If θ_3 and θ_4 were closer to 0, then the equation above would become less complex
  - Closer to $y = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2$
  - Simpler hypothesis

Regularization

- Regularization pushes the values of θ's to be smaller
  - θ's are adjusted by the negative of the cost function derivative
  - The higher the cost, the greater the adjustment
  - Modify the cost function, so that it takes θ's values into account

Regularization

- New cost function

$$J(\theta) = \frac{1}{2m}[\underbrace{\boxed{\sum_{i=1}^{m}\left(h_\theta\left(x^i\right) - y^i\right)^2}}_{\text{Usual Least Squares Cost function}} + \underbrace{\boxed{\lambda \sum_{j=1}^{n} \theta_j^2]}}_{\text{Regularization}}$$

Regularization

$$J(\theta) = \frac{1}{2m}[\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2 + \lambda\sum_{j=1}^{n}\theta_j^2]$$

- λ = Regularization parameter
- n: number of θ values
- Cost increases as θ’s values increase
- Basically, penalize the complexity of the equation
- Equation is now a tradeoff: Fitting the data VS reducing the complexity

Regularization

- Regularization parameter λ
  - Controls the tradeoff between the 2 goals: Converging without overfitting
  - The greater the value of θ's, the greater the cost
  - Tries to keep the θ's small to avoid overfitting

Regularization

- λ's value and its effect on regularization
  - If $\lambda = 0$, then there is no regularization
    - Cost function would be the same as before
  - If λ is too high, it could result in underfitting
    - Or even fail to converge at all
    - Example: If all θ's are very close to 0, then y would always be equal to θ_0
      - $y = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2 + \theta_3 x_2 x_3 + \theta_4 x_3^4 + \theta_5 x_1^2 x_3^2$
        becomes
        $y = \theta_0$

Regularization

$$h_\theta(x) = y = \theta x$$

- If we change the cost function…
- We need to update the cost derivative

New Cost Derivative $= \frac{\partial}{\partial_\theta} J(\theta)$

$$= \frac{\partial}{\partial_\theta} \frac{1}{2m} \left[\sum_{i=1}^{m} \left(h_\theta(x^i) - y^i\right)^2 + \lambda \sum_{j=1}^{n} \theta_j^2\right]$$

$$= \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^i) - y^i\right) . x^i + \lambda\theta_j$$

- Remember each θ value has a different cost derivative based on the power of the feature it is attached to

Regularization

- Gradient descent with regularization
  - For every feature θ_j

$$\theta_j = \theta_j - \alpha \frac{1}{m} [\sum_{i=1}^{m} (h_\theta(x^i) - y^i).x_j^i + \lambda\theta_j]$$

- θ_0
  - Common practice not to regularize it

$$\theta_0 = \theta_0 - \alpha \frac{1}{m} [\sum_{i=1}^{m} (h_\theta(x^i) - y^i)]$$

Regularization

- Added steps in the algorithm:
  - Pick a value for the regularization term λ
  - Change the cost function and its derivative

Regularization

- If we are training a model with regularization and we get an underfitting model:
  - Decrease the regularization parameter λ
  - Still can add more features
  - Make the polynomial more complex

References

- Notes by Antoine Abi Chacra, DigiPen Institute of Technology