

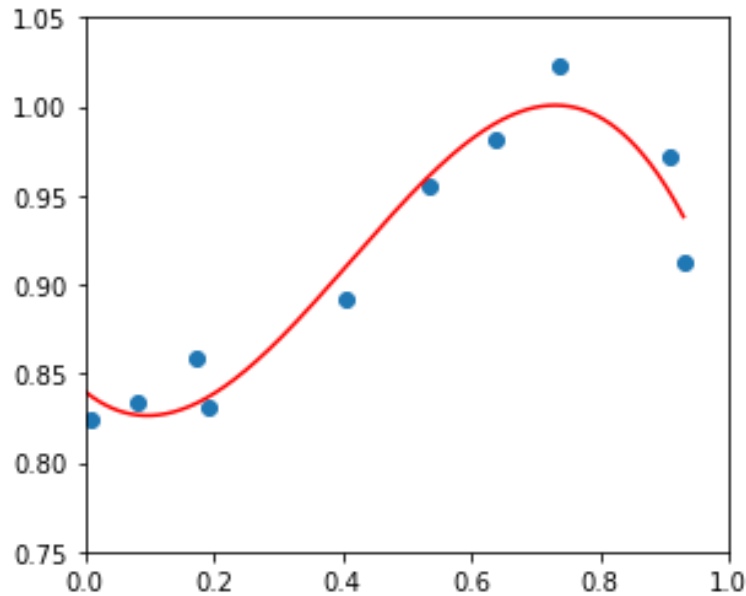


MULTI-CLASS CLASSIFICATION

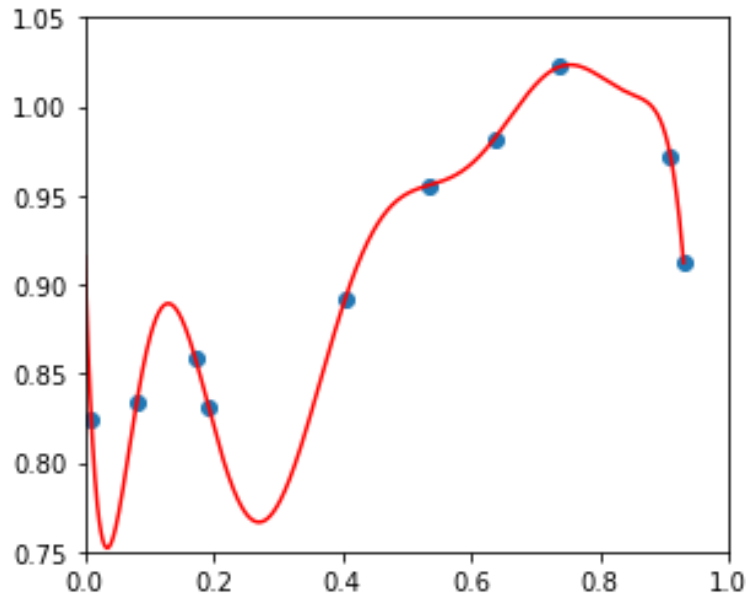
DR. FARHAD RAZAVI



SOLUTION TO OVERFITTING

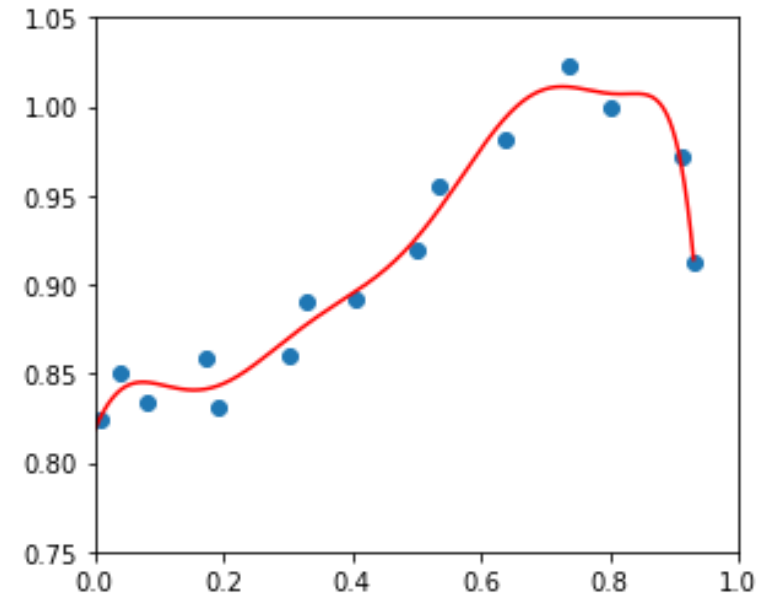


$$w_1x_1 + w_2x_1^2 + w_3x_1^3 + b$$



$$w_1x_1 + w_2x_1^2 + w_3x_1^3 + w_4x_1^4 + w_5x_1^5 + w_6x_1^6 + w_7x_1^7 + w_8x_1^8 + w_9x_1^9 + b$$

$$w_4, w_5, w_6, w_7, w_8, w_9 \neq 0$$



$$w_1x_1 + w_2x_1^2 + w_3x_1^3 + w_4x_1^4 + w_5x_1^5 + w_6x_1^6 + w_7x_1^7 + w_8x_1^8 + w_9x_1^9 + b$$

$$w_4, w_5, w_6, w_7, w_8, w_9 \rightarrow 0$$

- One solution is to collect more data.
- More data will force the model to adjust the parameters in order to get a better fit for the data.
- Sometimes there are no ways to gather more data and this method might not work.

SOLUTION TO OVERFITTING

Size	Bedrooms	Floors	Age	Avg income	...	Distance to coffee shop	Price
x_1	x_2	x_3	x_4	x_5		x_{100}	y
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots

All features
+
Insufficient data

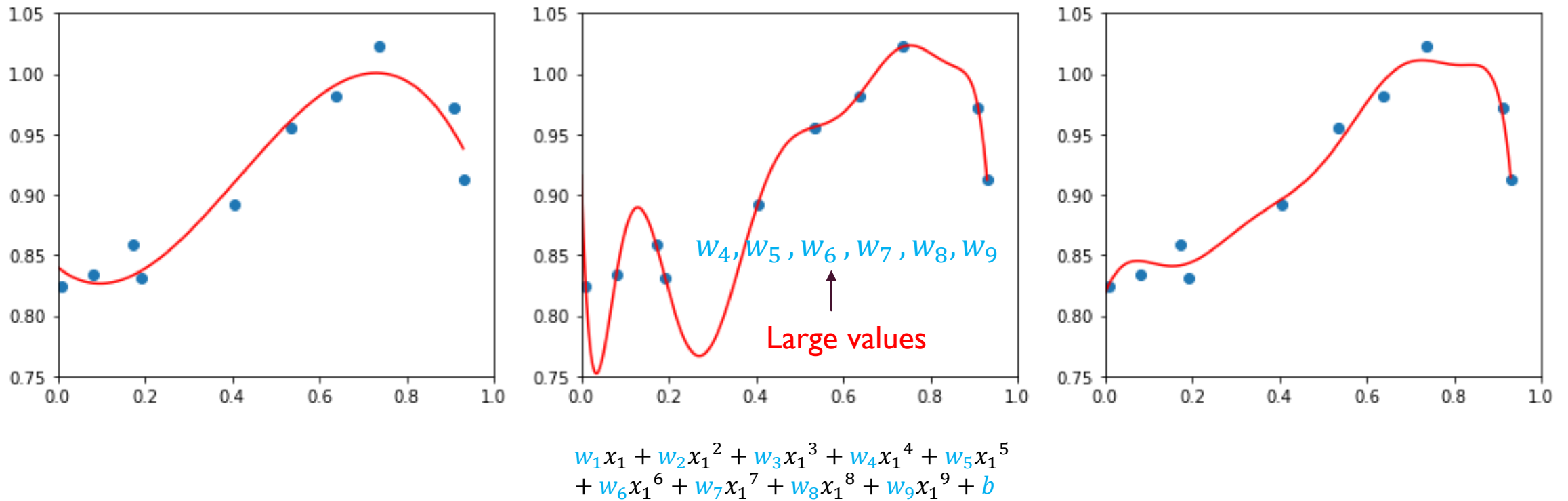
Over fit

selected features
+
(size, bedrooms, age)

Just right

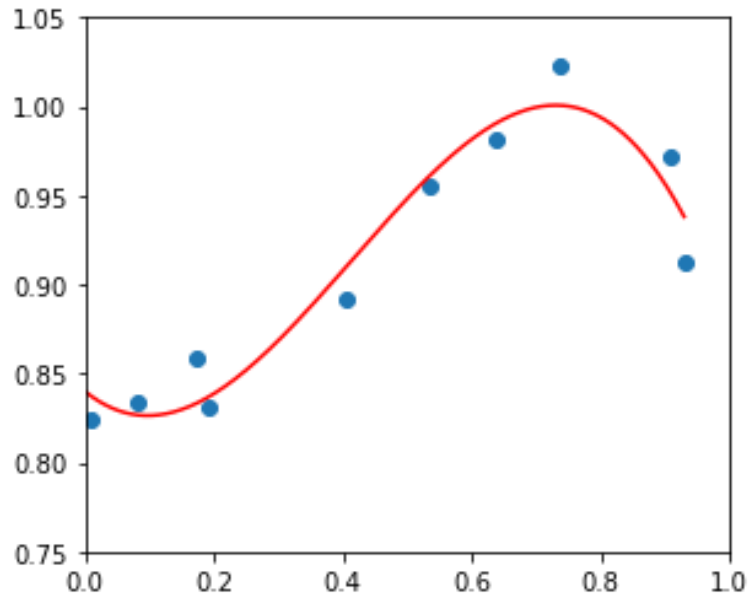
- Another solution is to select only a few relevant features and therefore reduce the dimensionality of the model.
- Feature selection is dependent on the designer intuition. One disadvantage is that some of the ignored features might have been important.

SOLUTION TO OVERFITTING

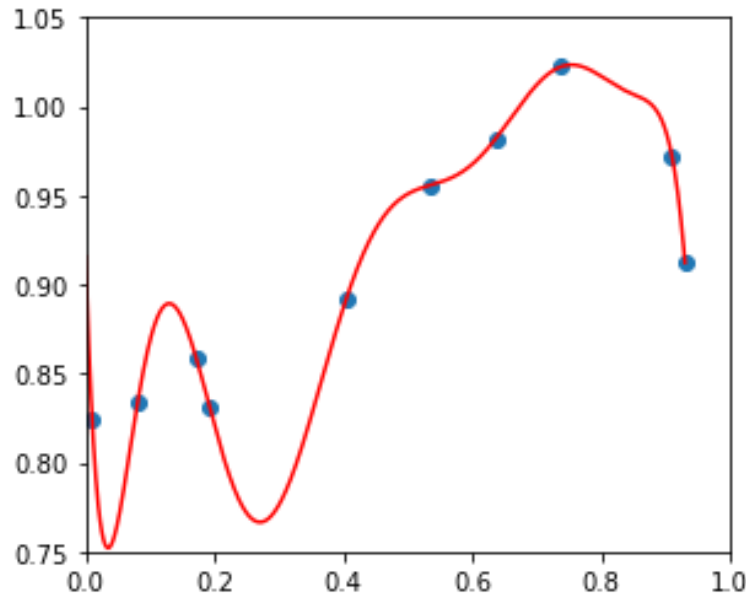


- Another solution is to regularize the parameters of the model (w_j) so that no parameters gets too large.

COST FUNCTION WITH REGULARIZATION



$$w_1x_1 + w_2x_1^2 + w_3x_1^3 + b$$



$$w_1x_1 + w_2x_1^2 + w_3x_1^3 + w_4x_1^4 + w_5x_1^5 + w_6x_1^6 + w_7x_1^7 + w_8x_1^8 + w_9x_1^9 + b$$

- $w_4, w_5, w_6, w_7, w_8, w_9$ are large numbers.
- How can we enforce the model to keep them small.
- One way is to penalize the model from fitting to large values of w_j by adding some extra terms in cost function.

$$J(\vec{w}, b) = \frac{1}{2m} \sum_{i=1}^m (f_{\vec{w}, b}(x^{(i)}) - y^{(i)})^2 + 1000 w_4^2 + 1000 w_5^2 + \dots + 1000 w_9^2$$

SOLUTION TO OVERFITTING

Size	Bedrooms	Floors	Age	Avg income	...	Distance to coffee shop	Price
x_1	x_2	x_3	x_4	x_5		x_{100}	y
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots

- In general, we do not know which parameters to penalize.
- We penalize all parameters and will let algorithm to decide which is more important.

$$J(\vec{w}, b) = \frac{1}{2m} \sum_{i=1}^m (f_{\vec{w}, b}(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m} \sum_{j=1}^n w_j^2$$

REGULARIZED REGRESSION

- $J(\vec{w}, b) = \frac{1}{2m} \sum_{i=1}^m (f_{\vec{w}, b}(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m} \sum_{j=1}^n w_j^2 = \frac{\lambda}{2m} \sum_{j=1}^n w_j^2 + \frac{1}{2m} \sum_{i=1}^m (f_{\vec{w}, b}(x^{(i)}) - y^{(i)})^2$

- Repeat for $j = 1, 2, \dots, n$ {

$$w_j = w_j - \alpha \frac{\partial}{\partial w_j} J(\vec{w}, b);$$

$$b = b - \alpha \frac{\partial}{\partial b} J(\vec{w}, b);$$

} simultaneous updates

$$\frac{\partial}{\partial w_j} J(\vec{w}, b) = \frac{\lambda}{m} w_j + \frac{1}{m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)}) x^{(i)}$$

$$\frac{\partial}{\partial b} J(\vec{w}, b) = \frac{1}{m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)})$$

REGULARIZED REGRESSION

- Repeat for $j = 1, 2, \dots, n$ {

$$w_j = w_j - \alpha \left[\frac{\lambda}{m} w_j + \frac{1}{m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)}) x^{(i)} \right]$$

$$b = b - \alpha \left[\frac{1}{m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)}) \right]$$

} simultaneous updates

$$w_j = \underbrace{w_j - \alpha \frac{\lambda}{m} w_j}_{w_j \left(1 - \alpha \frac{\lambda}{m} \right)} - \underbrace{\alpha \left[\frac{1}{m} \sum_{i=1}^m (f_{\vec{w}, b}(\vec{x}^{(i)}) - y^{(i)}) x^{(i)} \right]}_{\text{usual updates}}$$

$$\alpha = 0.01; \lambda = 10; m = 50 \rightarrow 1 - 0.01 \frac{10}{50} = 1 - 0.002 = 0.998$$

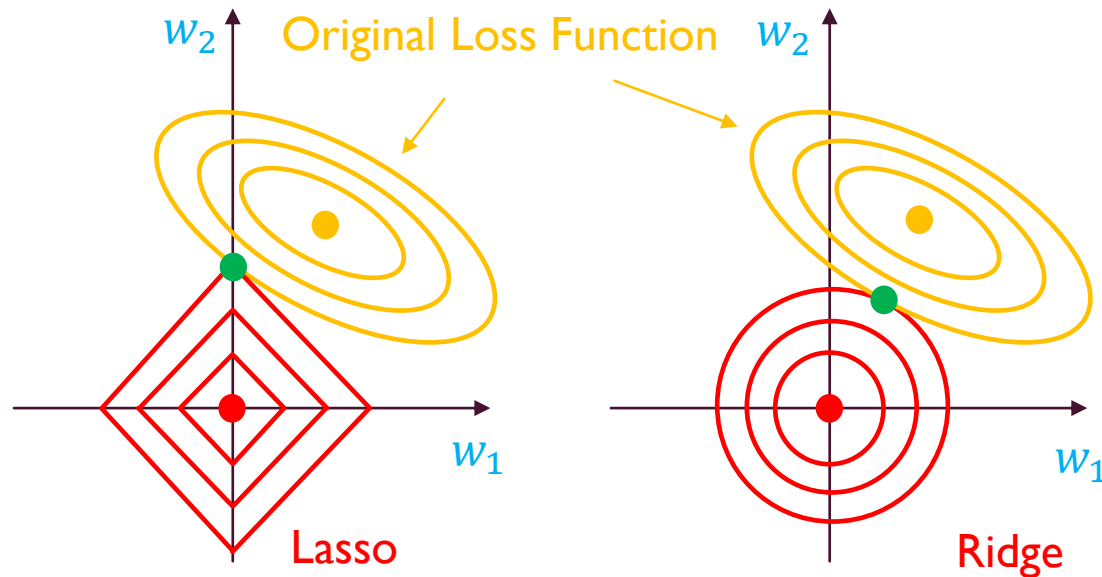
REGULARIZATION USING OTHER NORMS

L_1 norm (Lasso)

$$\frac{\lambda}{2m} \sum_{j=1}^n |w_j|$$

L_2 norm (Ridge)

$$\frac{\lambda}{2m} \sum_{j=1}^n w_j^2$$



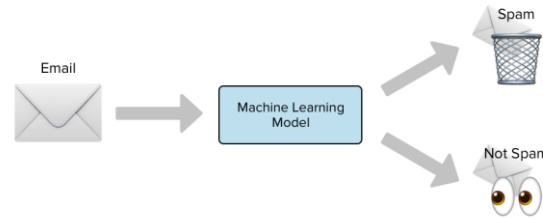
- Lasso regression :
 - Introduces more sparsity to the model.
 - Is helpful for feature selection.
 - Is more computationally efficient as a model due to the sparse solutions.
- Ridge regression:
 - Pushes weights towards 0, but not actually 0.
 - In practice, usually performs better.
- Usually, a combination of these two works best (Elastic Net).

MULTICLASS CLASSIFICATION

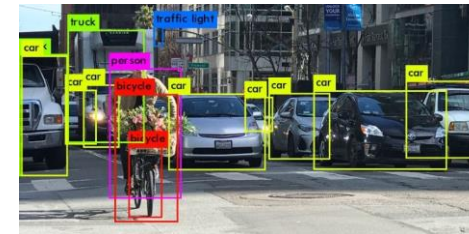
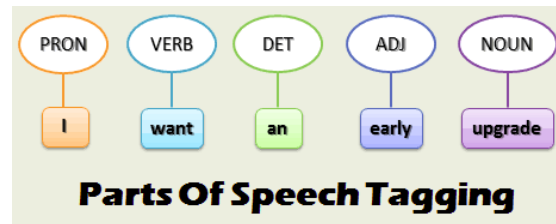
- Binary and Multi-class: problems and classifiers
- Solving Multi-class problems with binary classifiers
 - One-vs-all
 - All pairs
 - Error correcting codes

CLASSIFICATION PROBLEM

- Natural binary:
 - Spam classification (spam vs. ham)
 - Online Marketing
 - Cat vs no cat!
- However, many are multiclass
 - News Topic classification
 - Part of speech tagging
 - Object detection
 - Weather



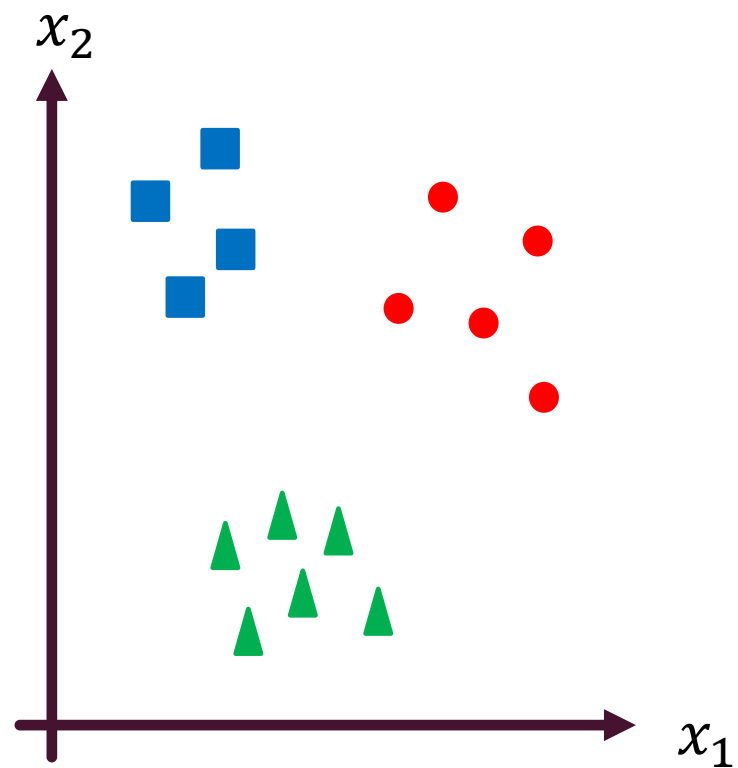
Business, Sports, Entertainment, ...






CLASSIFIERS

- Some directly are multi-class (naïve Bayes, Logistic regression, K-Nearest Neighbor)
- Other classifiers are basically binary
 - SVM (support vector machine)
 - Perceptron
 - Boosting

MULTI-CLASS DATA



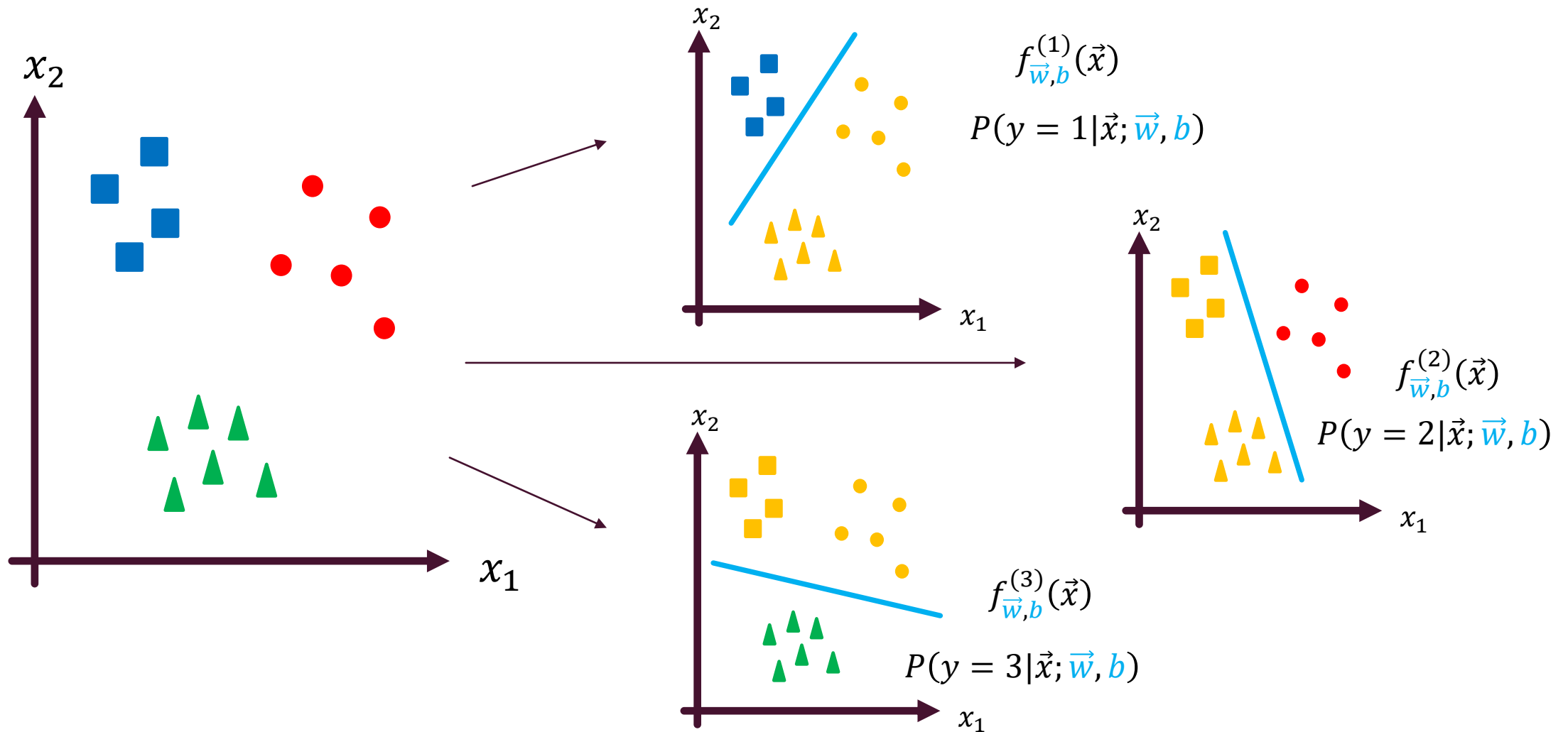
- Class 1: 
- Class 2: 
- Class 3: 

ONE-VS-ALL (LOGISTIC REGRESSION)

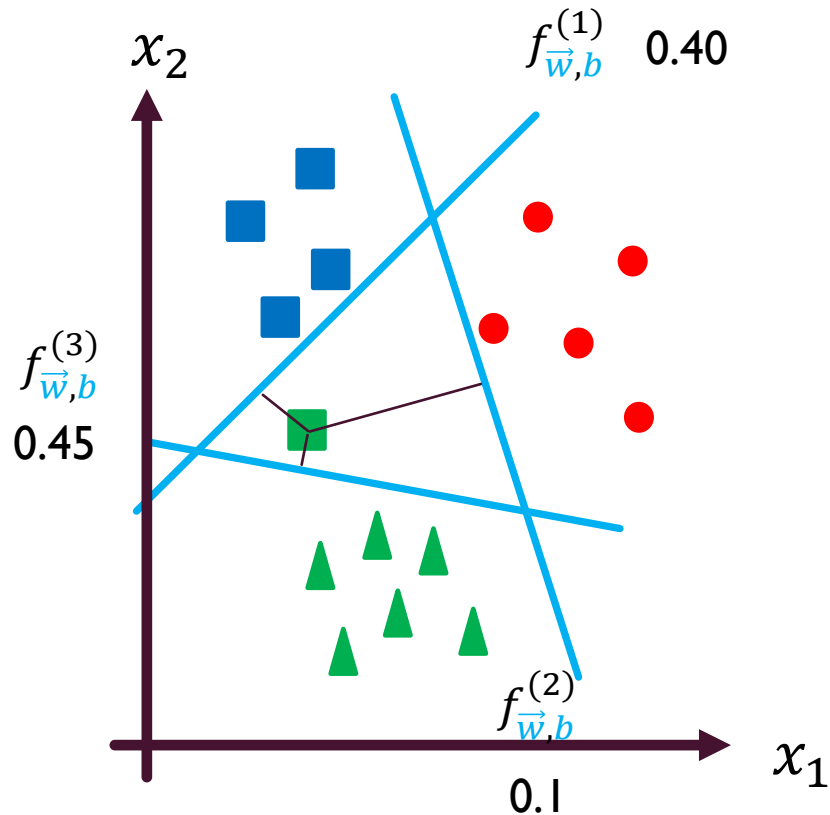
- Train a logistic regression classifier $f_{\vec{w}, b}^{(i)}(\vec{x})$ for each class i to predict the probability that $y = i$.
- On a new input \vec{x} , to make a prediction, pick the class i that maximizes

$$\max_i f_{\vec{w}, b}^{(i)}(\vec{x})$$

ONE-VS-ALL (ONE-VS-REST)

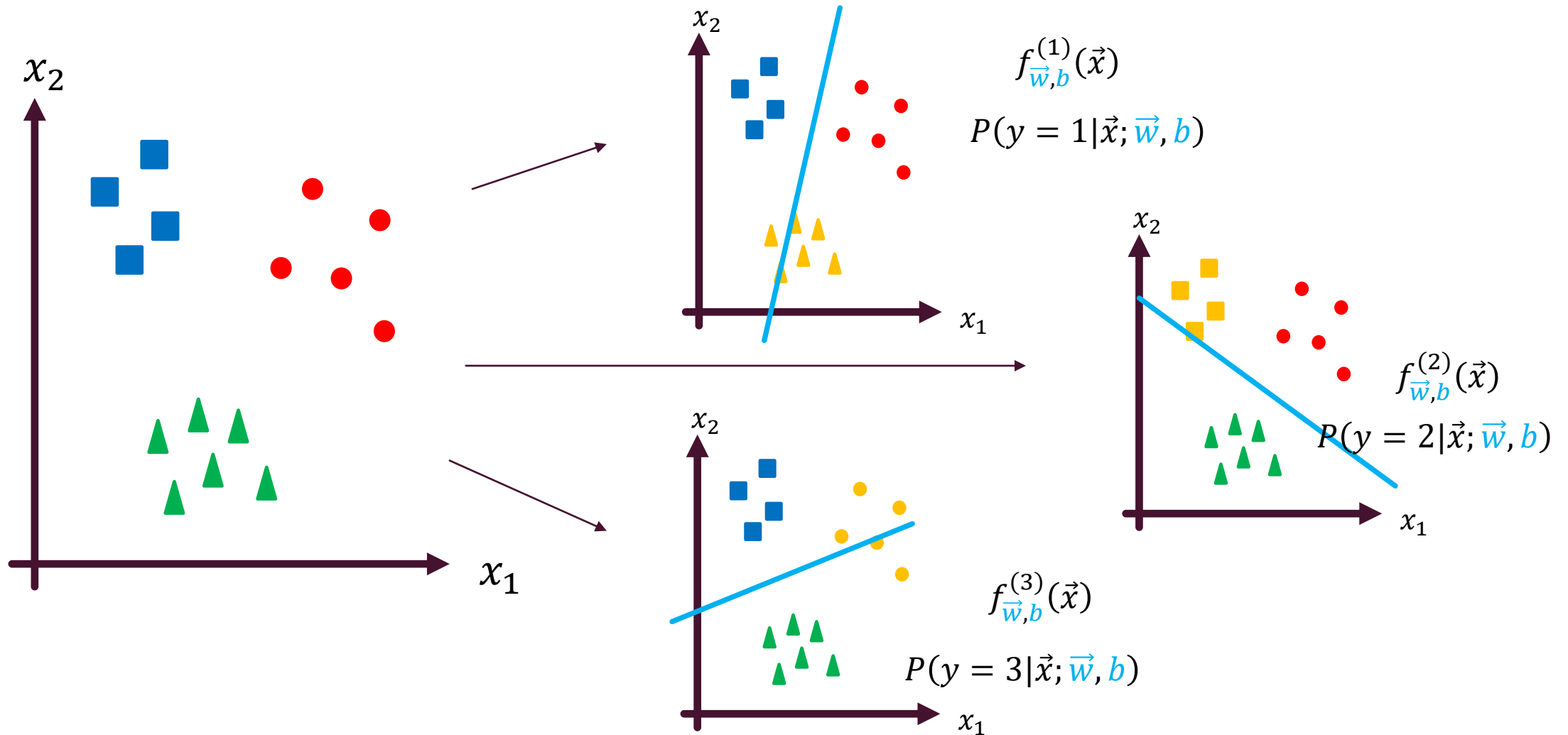


ONE-VS-ALL (ONE-VS-REST)

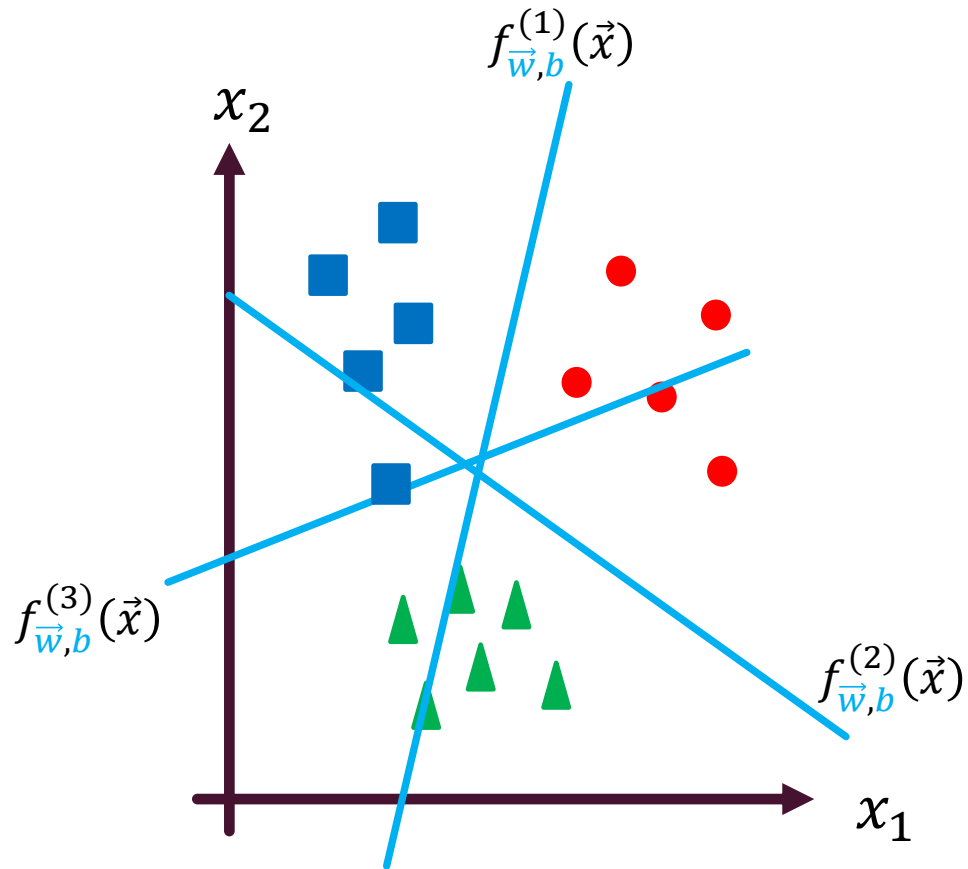





- Break k -class problem into k binary problems and solve separately.
- Combine predictions: evaluate all f 's, hope exactly one is + (otherwise take highest confidence).
 - $f_{\vec{w},b}^{(1)}(\vec{x}): 0.40$
 - $f_{\vec{w},b}^{(2)}(\vec{x}): 0.10$
 - $f_{\vec{w},b}^{(3)}(\vec{x}): 0.45$
- Incorrect predictions happens more often.
- Sensitive to class imbalance
 - Forces the decision boundaries to get too close to the class with far smaller samples.

ONE-VS-ONE (ALL-PAIRS)



ONE-VS-ONE (ALL-PAIRS)



- One binary problem for each pair of classes (total of $\binom{k}{2}$).
- Take class with most positive and least negatives
 - $f_{\vec{w},b}^{(1)}(\vec{x})$: 
 - $f_{\vec{w},b}^{(2)}(\vec{x})$: 
 - $f_{\vec{w},b}^{(3)}(\vec{x})$: 
- More accurate than one-vs-all
- Could be faster as each classifier deals with a smaller number of training data in comparison to one-vs-all.

SOFTMAX

- Recall that for logistic regression we used the sigmoid function to model the following probability of a new sample \vec{x} belonging to category $y = 1$.

$$a_1 = g(z) = g(\vec{w} \cdot \vec{x} + b) = \frac{1}{1 + e^{-z}} = P(y = 1 | \vec{x}) \quad \bullet$$

$$a_2 = 1 - a_1 = P(y = 0 | \vec{x}) \quad \bullet$$

- Let's generalize this with more than two categories.

- Category 1:

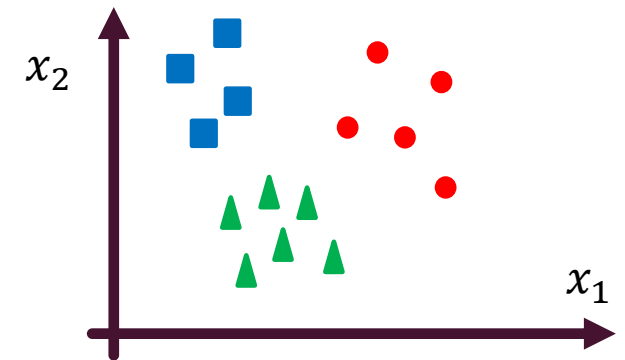
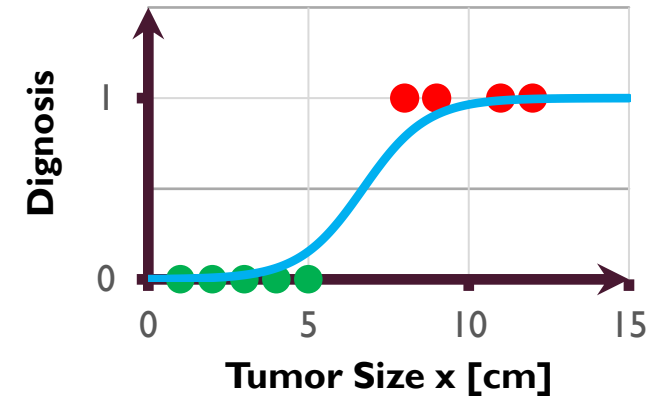
$$a_1 = P(y = 1 | \vec{x}) \quad \blacksquare$$

- Category 2:

$$a_2 = P(y = 2 | \vec{x}) \quad \blacktriangle$$

- Category 3:

$$a_3 = P(y = 3 | \vec{x}) \quad \bullet$$



SOFTMAX

- For each category we will define a different z parameter.

$$z_1 = \vec{w}_1 \cdot \vec{x} + b_1$$

$$z_2 = \vec{w}_2 \cdot \vec{x} + b_2$$

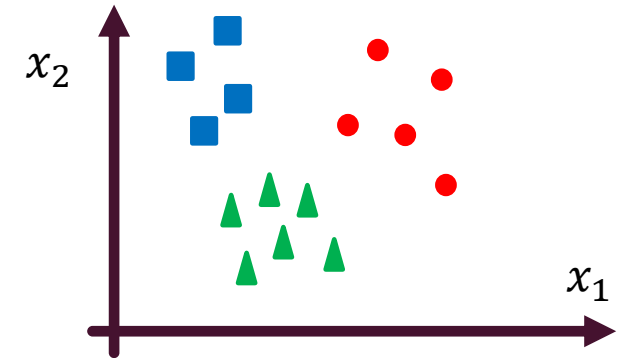
$$z_3 = \vec{w}_3 \cdot \vec{x} + b_3$$

- Then the probability of the feature belonging to each category can be modeled as.

$$a_1 = \frac{e^{z_1}}{e^{z_1} + e^{z_2} + e^{z_3}} = P(y = 1 | \vec{x}) \quad 0.5$$

$$a_2 = \frac{e^{z_2}}{e^{z_1} + e^{z_2} + e^{z_3}} = P(y = 2 | \vec{x}) \quad 0.2$$

$$a_3 = \frac{e^{z_3}}{e^{z_1} + e^{z_2} + e^{z_3}} = P(y = 3 | \vec{x}) \quad 0.3$$



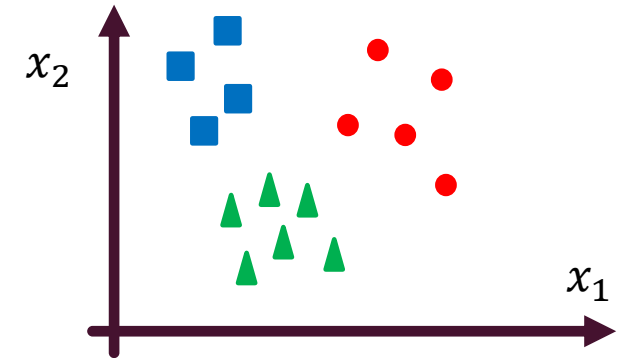
SOFTMAX GENERALIZED

- For K different category we will define a different z parameter.

$$z_j = \vec{w}_j \cdot \vec{x} + b_j \quad j = 1, \dots, K$$

- Then the probability of the feature belonging to each category can be modeled as.

$$a_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}} = \frac{e^{z_j}}{e^{z_1} + e^{z_2} + \dots + e^{z_K}} = P(y = j | \vec{x})$$
$$\sum_{j=1}^K a_j = 1$$



SOFTMAX COST

- For Logistic Regression:

$$a_1 = g(z) = \frac{1}{1 + e^{-z}} = P(y = 1|\vec{x})$$

$$a_2 = 1 - a_1 = P(y = 0|\vec{x})$$

- We defined the loss as follow:

$$Loss = \begin{cases} -\log(g(z)) & \text{if } y = 1 \\ -\log(1 - g(z)) & \text{if } y = 0 \end{cases}$$

$$J(\vec{w}, b) = \frac{1}{m} \sum_{i=1}^m [Loss]$$

SOFTMAX COST

- For Logistic Regression:

$$a_1 = g(z) = \frac{1}{1 + e^{-z}} = P(y = 1|\vec{x})$$

$$a_2 = 1 - a_1 = P(y = 0|\vec{x})$$

- We defined the loss as follow:

$$Loss = \begin{cases} -\log(a_1) & \text{if } y = 1 \\ -\log(1 - a_1) & \text{if } y = 0 \end{cases}$$

$$Loss = \begin{cases} -\log(a_1) & \text{if } y = 1 \\ -\log(a_2) & \text{if } y = 0 \end{cases}$$

- For softmax regression:

$$a_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}} = P(y = j|\vec{x}) \quad j = 1, \dots, K$$










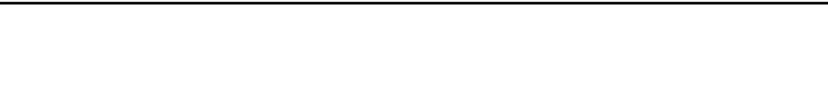
- Similarly, the loss will be defined as:

$$Loss = \begin{cases} -\log(a_1) & \text{if } y = 1 \\ -\log(a_2) & \text{if } y = 2 \\ \vdots & \\ -\log(a_K) & \text{if } y = K \end{cases}$$

- This is called Crossentropy loss.

PROJECT

- **Goal:**
 - Developing a categorization model to classify fashion MNIST dataset.
- **Groups:**
 - Minimum 5 and maximum 6 students

Label	Description	Examples
0	T-Shirt/Top	
1	Trouser	
2	Pullover	
3	Dress	
4	Coat	
5	Sandals	
6	Shirt	
7	Sneaker	
8	Bag	
9	Ankle boots	

INSTRUCTIONS

- It is ok to use others' code, but you need to clearly reference them. If you are modifying a portion of their code, that also needs to be clearly mentioned. I recommend writing comments on the body of the code together with your final report.
- Final report should summarize the steps you took to prepare your data. If you have done cleaning or feature reduction or addition on the dataset, make sure you will state your reasoning. Visualize the data for a few of the samples in the dataset.
- The role of each person must be clearly mentioned at the end of the report. Make sure every member of the group understands the model that is developed.
- Try to visualize the optimized parameters of your model.
- You can use any of the models that we have explained in the class. Nonetheless, explain your reasoning for the model you will end up choosing. The emphasis is less on picking the perfect model than applying the techniques we have learned in the class to improve the model.
- For the complexity of your model, try to show that the complexity that you have picked will give a reasonable balance between bias and variance.



REFERENCE

- Multiclass classification: Machine Learning Jordan Boyd-Graber, https://home.cs.colorado.edu/~jbg/teaching/CSCI_5622/

