

# Interpreting the coefficients in Linear Models

The dot product has been a recurring character during our Classical Machine Learning journey.

$$\Theta \cdot \mathbf{x}$$

By examining this expression more closely

- We can gain insight into what  $\Theta$  means
- Understand mathematically why transformation may be necessary
- Gain an appreciation of the "pattern matching" that it embodies

Recall the places in which dot product appears

- Linear Regression

$$\hat{\mathbf{y}} = \Theta \cdot \mathbf{x}$$

- Logistic Regression

$$\hat{s} = \Theta \cdot \mathbf{x}$$

for score  $\hat{s}$  (which becomes a probability via  $\hat{p} = \sigma(\hat{s})$ )

- Boundary equation for linearly separable classifiers, e.g., SVM

$$0 = \Theta \cdot \mathbf{x}$$

Consider one feature  $\mathbf{x}_j^{(i)}$  for example  $i$ .

- A unit increase in  $\mathbf{x}_j^{(i)}$
- Holding constant the values for all other features,
- Increases  $\Theta \cdot \mathbf{x}^{(i)}$  by  $\Theta_j$

Thus

$$\Theta_j = \frac{\partial}{\partial \mathbf{x}_j} \Theta \cdot \mathbf{x}$$

$\Theta_j$  may be interpreted as

- The sensitivity of  $\Theta \cdot \mathbf{x}$  to changes in feature  $j$

## Numeric features

Consider numeric features  $\mathbf{x}_j, \mathbf{x}_{j'}$ .

Does

$$\Theta_j > \Theta_{j'}$$

mean that feature  $j$  is "more important" than feature  $j'$  ?

- No !
- It just means it has a larger impact
- Which can *also* occur if  $\mathbf{x}_j, \mathbf{x}_{j'}$  are on different scales

For example consider the equality

$$\mathbf{y} = \Theta \cdot \mathbf{x}$$

- Replacing  $\mathbf{x}_j$
- By  $\mathbf{x}_{j''} = \mathbf{x}_j * 10$
- Mathematically results in  $\Theta_{j''} = \Theta_j / 10$

Thus, the scale of the parameter is dependent on the scale of the feature.

Unless two features are on the same scale: we can't directly compare their corresponding parameters.

# Categorical features

Consider a categorical feature with categories from

$$C = \{c_1, c_2, \dots\}$$

One Hot Encoding this feature replaces the original feature with  $|C|$  binary features

- $\text{Is}_{c_1}$
- $\text{Is}_{c_2}$
- $\vdots$
- $\text{Is}_{c_{|C|}}$

Suppose  $\mathbf{x}_j$  corresponds to the binary feature

$\text{Is}_{c_1}$

Then, by the formula for dot product

- $\Theta_j$  is the *increment* to  $\Theta \cdot \mathbf{x}$
- Arising from  $\mathbf{x}_j^{(i)} = 1$
- Compared to  $\mathbf{x}_j^{(i)} = 0$



That is:

- $\Theta_j$  is how much  $\Theta \cdot \mathbf{x}$  increases
- When example  $i$  has feature value  $c_1$  rather than any of  $\{c_2, \dots, \}$

We can use this interpretation

- To further emphasize the problem of
- Treating a categorical variables as a number rather than a collection of binary indicator variables

For example, let's revisit the Passenger Class  $Pclass \in \{1, 2, 3\}$  from the Titanic example.

- As a collection of binary indicator variables, the increment of being in each class is  $\Theta_{Is_1}, \Theta_{Is_2}, \Theta_{Is_3}$
- As a numeric variable with parameter value  $\Theta_j$ 
  - Being in Class 3 has three times the effect as being in Class 1

Thus

- As numeric, we imply a particular magnitude with each category
- As binary indicator, the magnitude is determined by the data

# Motivating a transformation

Understanding the meaning of  $\Theta$  may help us choose a transformation.

Suppose we have examples  $\langle \mathbf{X}, \mathbf{y} \rangle$  where

$$\mathbf{y} = \Theta \cdot \mathbf{x}$$

**does not** seem to hold.

Perhaps a transformation to either/both of  $\mathbf{x}$ ,  $\mathbf{y}$  **will** make the relationship linear.

For example, consider

- $\mathbf{y}$  and  $\mathbf{x}$  are time-series of prices, with different scales
- We observe that the impact on  $\mathbf{y}$  of a *unit change* in  $\mathbf{x}_j$ 
  - Is much bigger when  $\mathbf{x}_j$  is small
  - Compared to when  $\mathbf{x}_j$  is large
- So

$$\mathbf{y} \neq \Theta \cdot \mathbf{x}$$

Now suppose we re-dominate (by transforming) the timeseries  $\mathbf{y}$  and  $\mathbf{x}$  to

- Timeseries  $\mathbf{y}'$   
= daily % change in  $\mathbf{y}$
- Timeseries  $\mathbf{x}'$   
= daily % change in  $\mathbf{x}$

This transformation from Price to Return is common in Finance.

It may now turn out that

- A unit change in  $\mathbf{x}'$
- Results in a change in  $\mathbf{y}'$
- That is *independent* of the magnitude of  $\mathbf{x}'$
- So

$$\mathbf{y}' = \Theta' \cdot \mathbf{x}'$$

That is

- A 1 percentage point change in the price of  $\mathbf{x}$
- Causes  $\mathbf{y}$  to change by  $\Theta'$  percentage points

So there **was** a relationship between  $\mathbf{y}$  and  $\mathbf{x}$ , not in Price but in Return.

The Capital Asset Pricing Model of Finance postulates such a relationship and it is common to transform prices to returns.



## Transformed targets

At times we may apply transformations to target values rather than just features.

This means that  $\Theta_j$  is the sensitivity of the *transformed* target.

Recall that Logistic Regression could be formulated as

- Linear Regression of the features
- Versus the *log odds*

$$\log_e \frac{\hat{p}}{1 - \hat{p}} = \Theta^T \mathbf{x}$$

So a unit change in feature  $\mathbf{x}_j$  with parameter  $\Theta_j$  changes the odds  $\frac{\hat{p}}{1-\hat{p}}$  in a *multiplicative* way

$$\Theta^T(\mathbf{x} + 1) - \Theta^T \mathbf{x} = \Theta_j$$

unit change in  $\mathbf{x}$

$$\log_e \frac{\hat{p}}{1-\hat{p}}$$

increases to

$$\log\left(\frac{\hat{p}}{1-\hat{p}}\right) + \Theta_j$$

$$= \log\left(\frac{\hat{p}}{1-\hat{p}} * \exp \Theta_j\right) \quad \text{odds increase by factor of } \exp \Theta_j$$

-

---

# Examples

- Log transform of target:
  - $\log \mathbf{y} = \Theta_0 + \Theta_1 * \mathbf{x}_1$
  - $\theta_1 = \frac{\partial \log \mathbf{y}}{\partial \mathbf{x}_1} = \% \text{ change in } \mathbf{y} \text{ per unit change in } \mathbf{x}_1$
- Log transform of both target and feature:
  - $\log \mathbf{y} = \Theta_0 + \Theta_1 * \log \mathbf{x}_1$
  - $\Theta_1 = \frac{\partial \log \mathbf{y}}{\partial \log \mathbf{x}_1} = \% \text{ change in } \mathbf{y} \text{ per } \% \text{ change in } \mathbf{x}_1$
- Standardize feature
  - Transform  $\mathbf{x}$  into  $z_{\mathbf{x}} = \frac{\mathbf{x} - \bar{\mathbf{x}}}{\sigma_{\mathbf{x}}}$
  - $\mathbf{y} = \Theta_0 + \Theta_1 * z_{\mathbf{x}}$
  - $\Theta_1 = \frac{\partial \mathbf{y}}{\partial z_{\mathbf{x}}}$  change in  $\mathbf{y}$  per 1 standard deviation change in  $\mathbf{x}$ 
    - since  $z$  is in units of "number of standard deviations"

## Remember

- if you transform features in training, you must apply the same transformation to features in test
  - if the transformation is parameterized, the parameters are determined at **train** fit time, not test !
- if you transform the target, the prediction is in different units than the original
  - you can perform the inverse transformation to get a prediction in original units

# Bucketing/Binning re-visited

Suppose  $x_j$  is a continuous numeric feature (e.g., Age ).

Some questions to consider

- Is a 1 year increase in age equally relevant for all ages ?
  - If so: numeric
- Is a 1 year increase in age of the same relevance for a senior adult compared to an infant ?
  - If not: consider reducing discrete ages to discrete buckets
  - Is there a linear relationship between target and the center point of the bucket ?
    - If so: bucket feature can be numeric
    - If not: bucket feature categorical

# Interpreting the MNIST classifier: template matching

The  $\Theta$  produced by a linear classifier can be viewed as templates

- the strength of  $\Theta_j$  tells you how strongly feature  $\mathbf{x}_j$  influences the target

So we can interpret  $\Theta$  as a "template" for what a model is looking for.

Let's look at the template for

- The 10 separate, single-digit binary MNIST classifiers
- Or similarly: each row of  $\Theta$  for the multinomial 10 class MNIST classifier

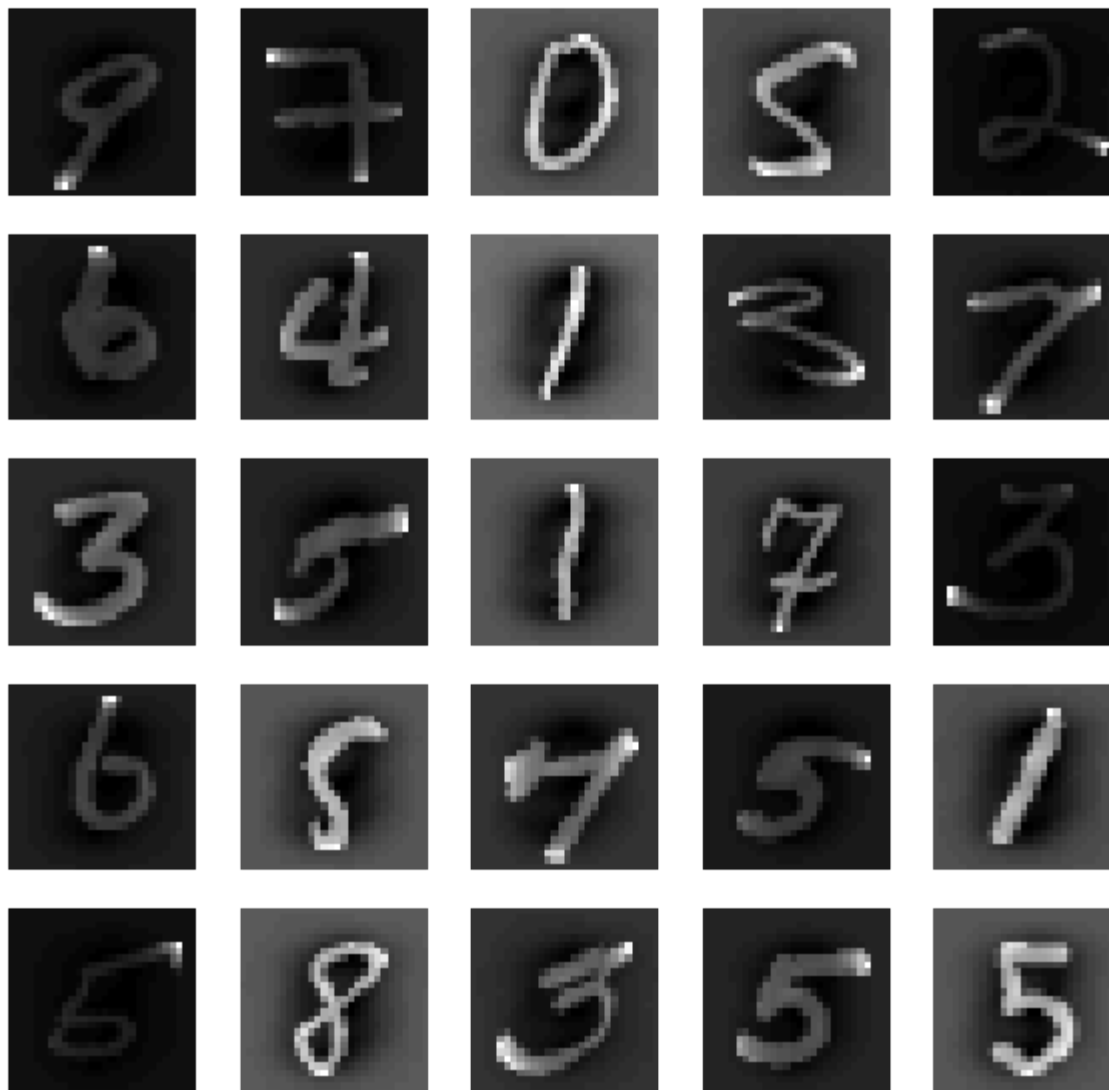
Here's the training data

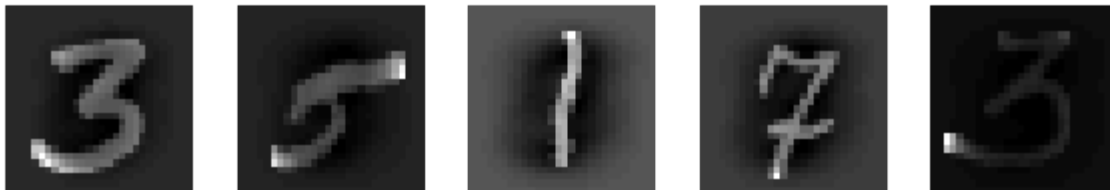
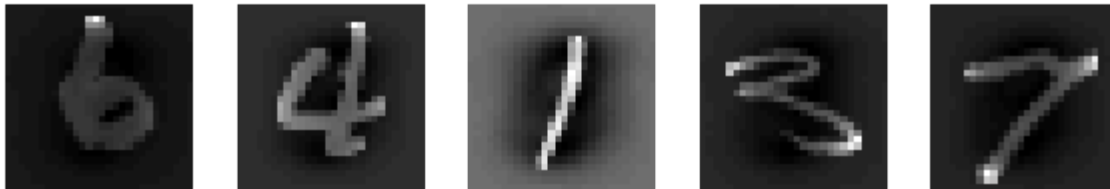
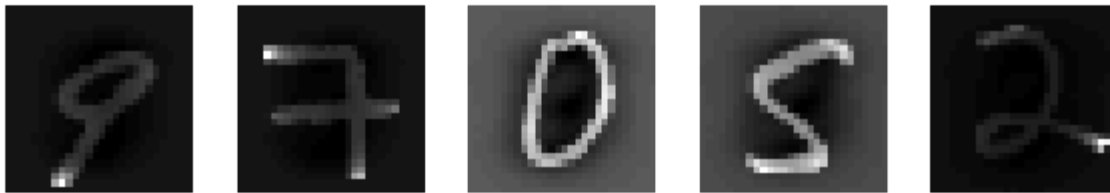


```
In [5]: mnh.setup()  
        mnh.visualize()
```

Retrieving MNIST\_784 from cache

Out[5]:



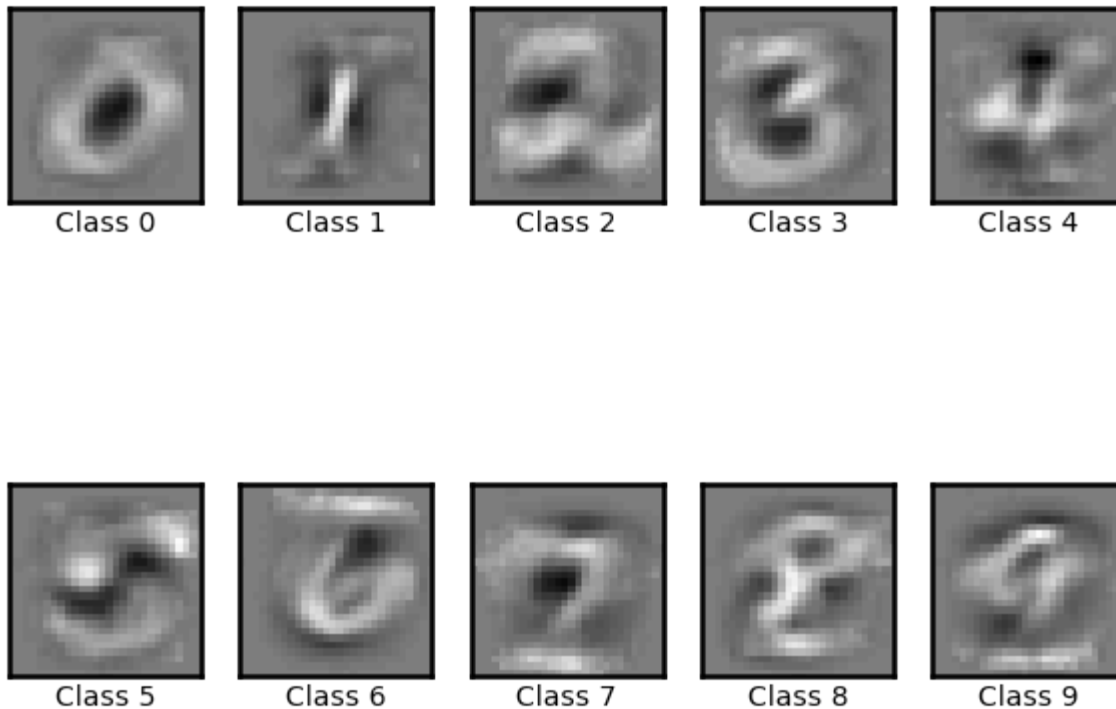


Let's fit a LogisticRegression model and examine the templates (coefficients  $\Theta$ )



```
In [6]: _ = mnh.fit()  
mnist_fig, mnist_ax = mnh.plot_coeff()
```

Parameters for...



## Recall

- There is one parameter per pixel
- The parameters are ordered in the same way as the linearization of the pixels
  - from  $(28 \times 28)$  grid to a vector of 784 numbers.
- We can display the 784 parameters in a  $(28 \times 28)$  image to show the intensity of parameter associated with a pixel
- White is high parameter value; Black is low (or negative)

- The template for 0 emphasizes small values (absence of bright pixels) in the center of the image
- The template for 1 emphasizes bright vertical pixels
- The template for 8 emphasizes the absence of bright pixels
  - in the two circles
  - in the pinched waist

You can now imagine how these templates might lead to misclassification

What is the classification of

- a "7" with a strong vertical line in the center (that's what the "1" template tries to match)
- a thin "0" (the "0" template is looking for a large donut)

So interpretation is a very powerful diagnostic tool for both understanding and improving your models.

In [7]: `print("Done")`

Done



