Statistical Learning: Feature Learning

Jesse Gronsbell
Department of Statistical Sciences, University of Toronto

Recap from yesterday

- How to compute or *estimate* m and b for the linear model with LS

- More on LS and linear regression

- Logistic regression

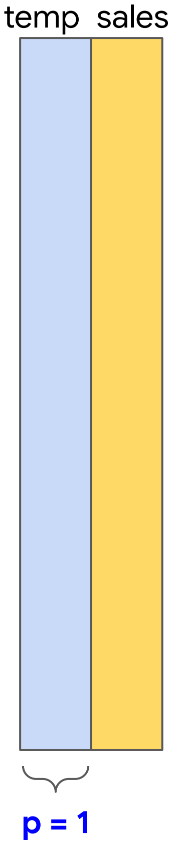
What we'll cover today

- Feature learning

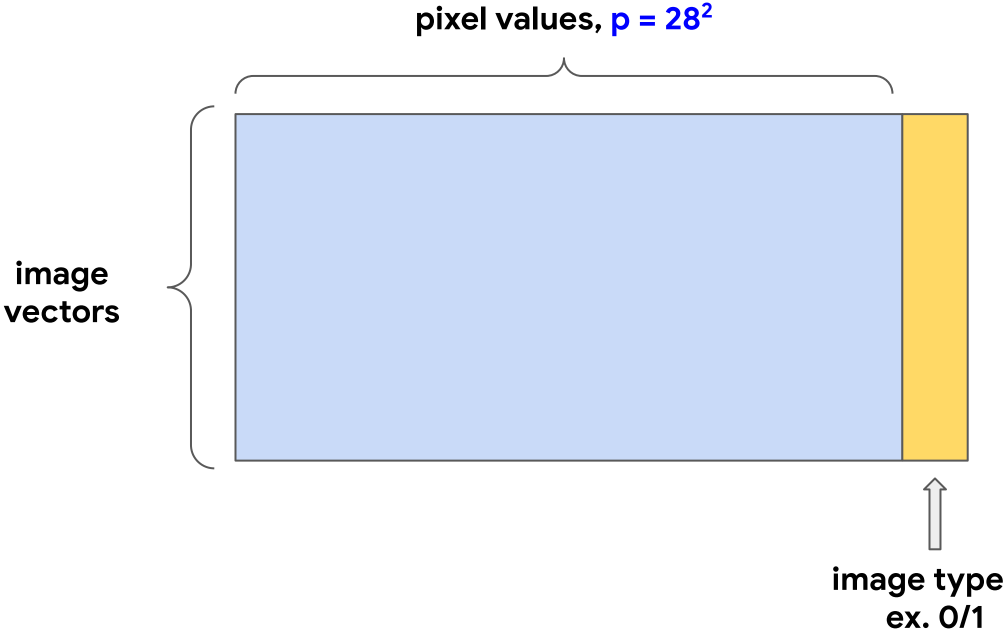
- Model evaluation

Part III: Principal Component Analysis

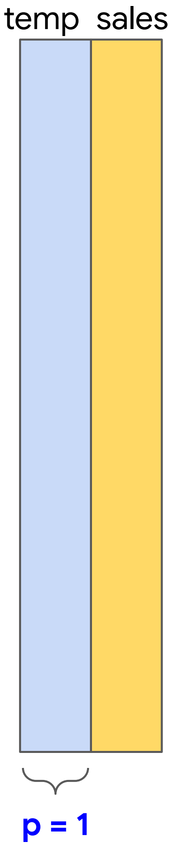
What happens when we have more covariates?



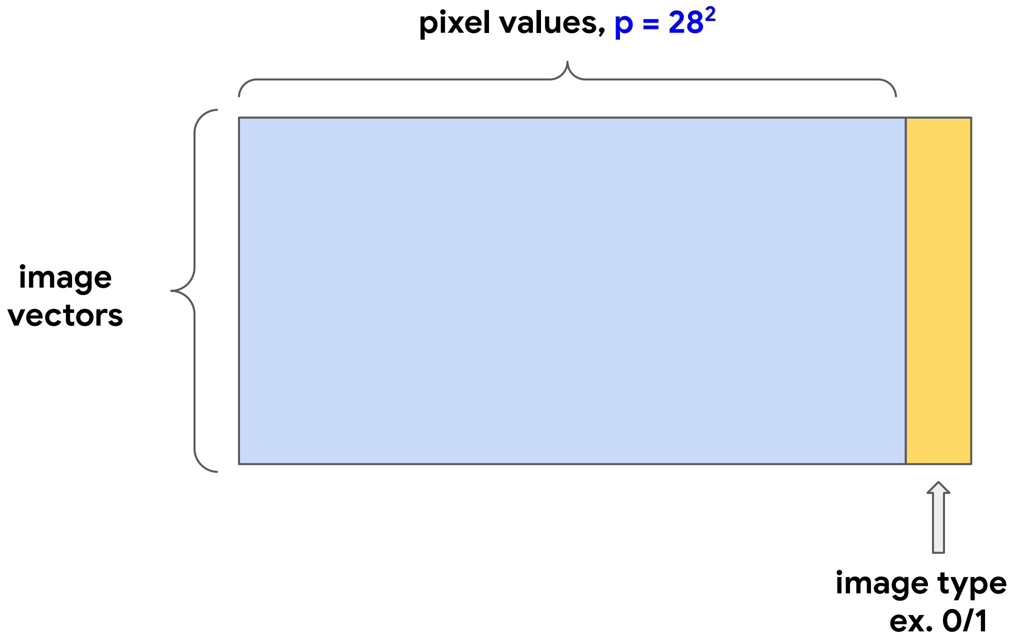
vs.



What happens when we have more covariates?



vs.



When p gets larger we often ask, is there a more compact representation of the covariates?

Why do we have to think about this?

- In the old days, we had a **small number of covariates** to build a model

- We now have **big data sets** that require a lot of cleaning and preparation

- **Cleaner and correctly prepared data** will lead to "**good**" **results**
    - Good = prediction performance, interpretation, understanding, trustworthiness, etc.

Feature (or representation) learning

- A subfield of statistical learning (or ML) focusing on methods that yield **useful and often concise representations of the data**

- Can replace **feature engineering** or the manual process of identifying covariates

- Useful when:
  - We have a **lot of covariates** and want to simplify and/or remove noise
  - The data set doesn't have a **natural representation**
    - **Ex.** Yelp dataset on kaggle has 5.2 million reviews on 174,000 businesses

There are many flavors of feature learning

<u>Supervised</u> Find a representation of **X** using **y**	**<u>Unsupervised</u>** Find a representation of **X** without using **y**
• Linear discriminant analysis ('shallow') • Neural network ('deep')	• Principal component analysis ('shallow') • Autoencoder ('deep')

There are many flavors of feature learning

<u>Supervised</u> Find a representation of **X** using **y**	**<u>Unsupervised</u>** Find a representation of **X** without using **y**
• Linear discriminant analysis ('shallow') • Neural network ('deep')	• Principal component analysis ('shallow') • Autoencoder ('deep')

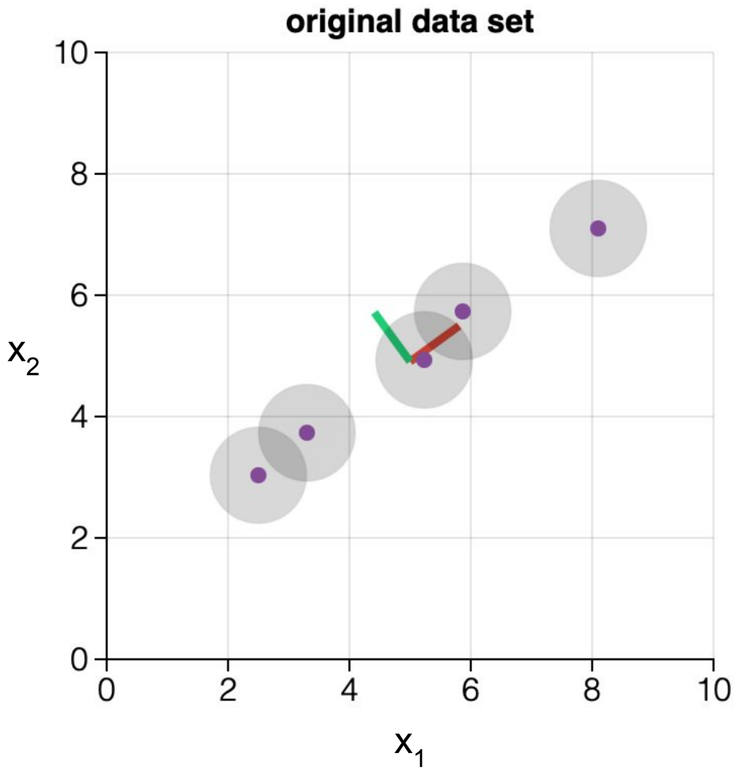
Principal component analysis (PCA)

- PCA **generates new covariates** that are weighted sums of the original covariates

- The new covariates **capture the variance** in the data and are **uncorrelated**

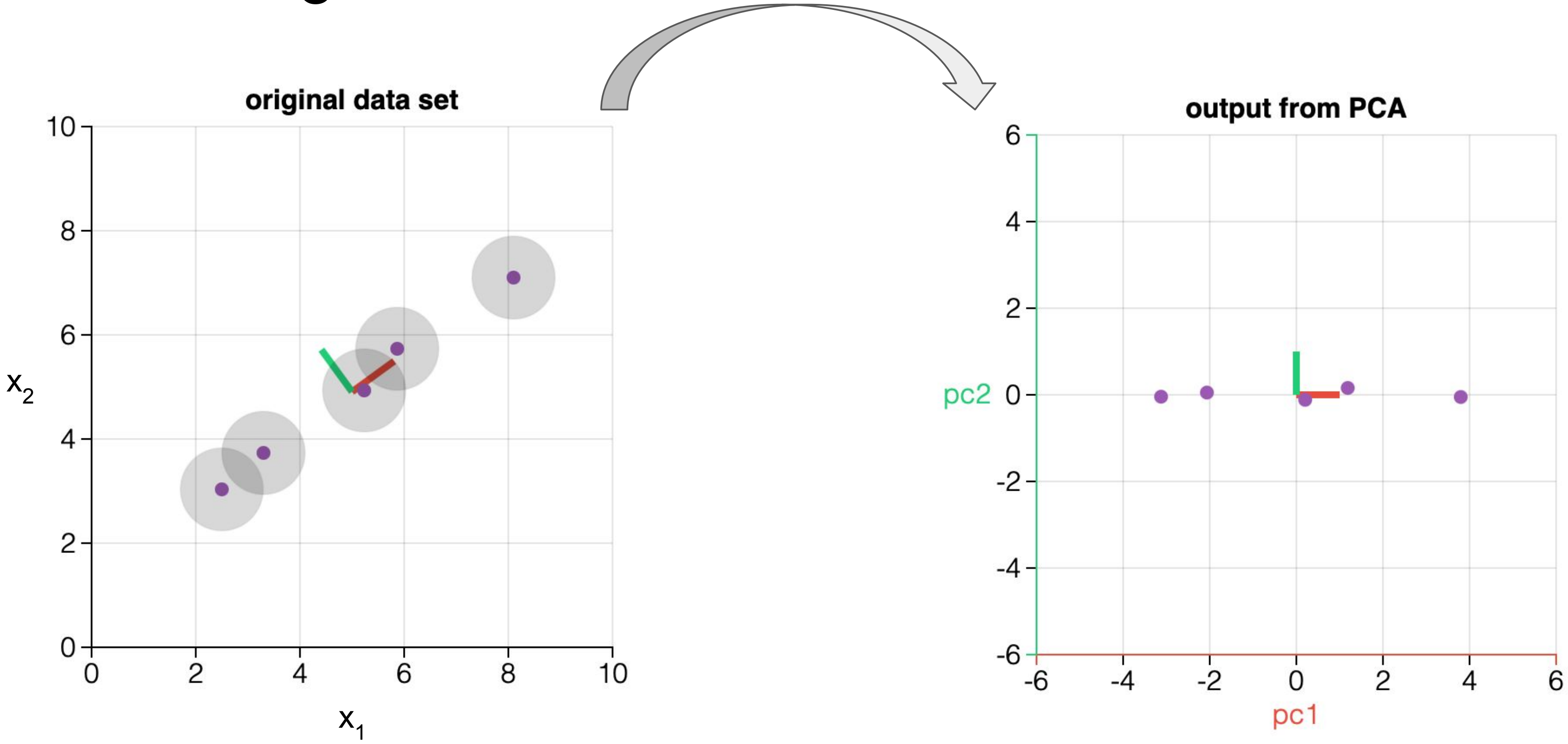
- Sometimes, only a **subset of the new covariates** contain all the variability in the original covariates
    - PCA yields a **more compact** representation of the original data set without losing information

Visualizing PCA



https://setosa.io/ev/principal-component-analysis/

Visualizing PCA



Intuition for the math behind PCA

- We obtain the principal components with the **eigendecomposition** of the **covariance matrix** of X

- The **covariance matrix** tells us how the covariates vary with one another

- The **eigendecomposition** gives us:
  - **Eigenvectors**: the directions (or components) that capture the most variance
  - **Eigenvalues**: the magnitude of the directions

- The **eigenvectors with the largest eigenvalues** are the principal components

PCA is great - what is the catch?

- We may have a more compact data set, but our new covariates are **harder to understand**

- We pay the price in what is called **model interpretability**

- More in the deep learning lectures!