Derived Inputs: Principal Components

-Applied Multivariate Analysis-

Lecturer: Darren Homrighausen, PhD

Lower dimensional embeddings

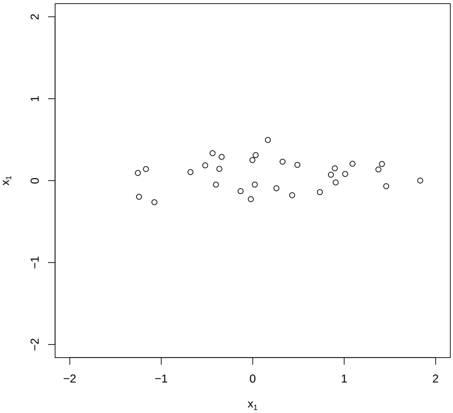
Remember: We have data $(X_1, Y_1), \ldots, (X_n, Y_n)$ where $X_i \in \mathbb{R}^p$ for each $i = 1, \ldots, n$.

The idea behind model selection is that a subset of the variables $(X_1, X_2, \ldots, X_p)$ are important for predicting the response.

This is basically like saying there is a lower dimensional space that contains most of the 'action'

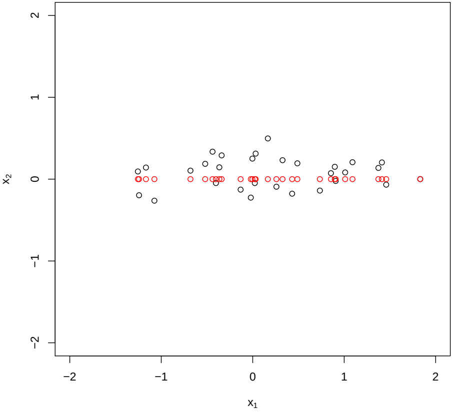
Lower dimensional embeddings: First Example

Suppose we have predictors X_1 and X_2 (there is no response, yet):



Lower dimensional embeddings: First Example

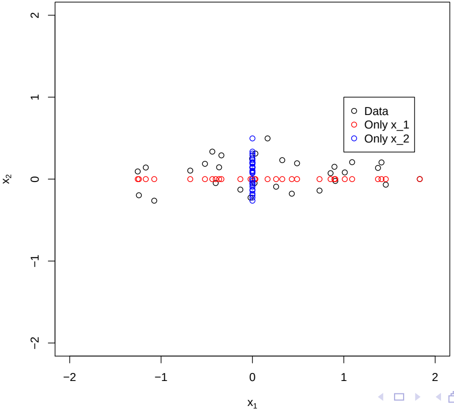
When we are doing variable selection, we are implicitly using the red dots (in this case, setting X_2 to zero):



Lower dimensional embeddings: First Example

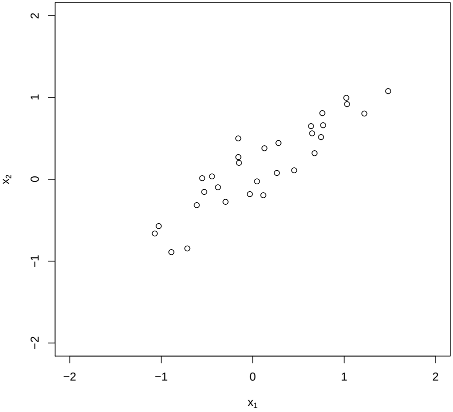
Looking at these alternatives at the same time, we can see that

- We more faithfully preserve the structure of the data by keeping X_1 and setting X_2 to zero than the opposite
- We don't lose that much structure by setting X_2 to zero.



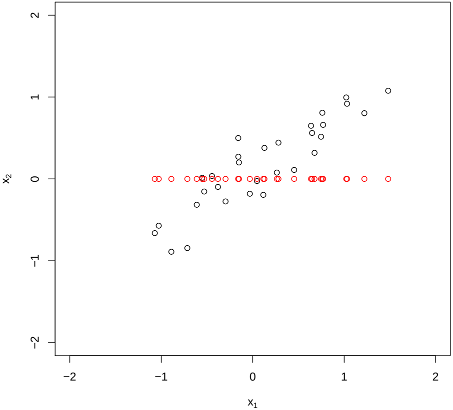
Lower dimensional embeddings: Second Example

An important feature of the First Example is that X_1 and X_2 aren't correlated with each other. What if they are correlated?



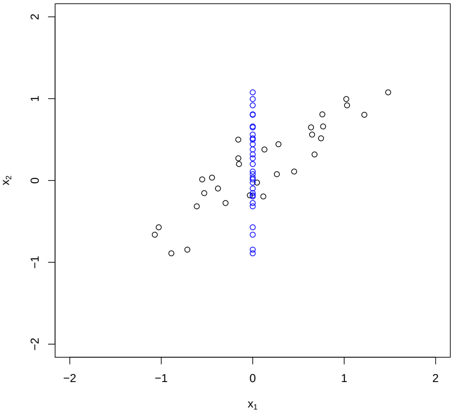
Lower dimensional embeddings: Second Example

When we are doing variable selection, we are implicitly using the red dots (in this case, setting X_2 to zero):



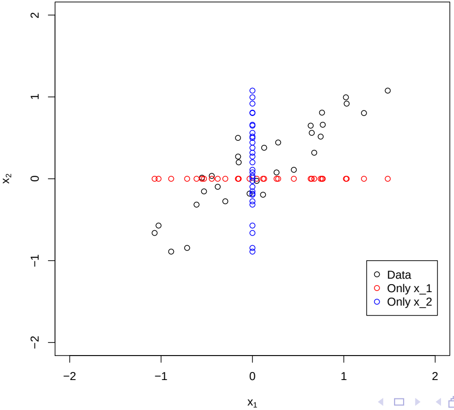
Lower dimensional embeddings: Second Example

Alternatively, we can select X_2 only, in which case we are setting X_1 to zero:



Lower dimensional embeddings: Second Example

We do lose a lot of structure by setting X_2 to zero.



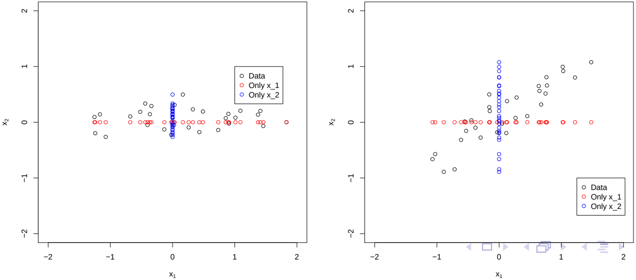
Lower dimensional embeddings: Comparison of Examples

Correlation complicates the model selection problem

Eliminating variables tcan significantly change the structure

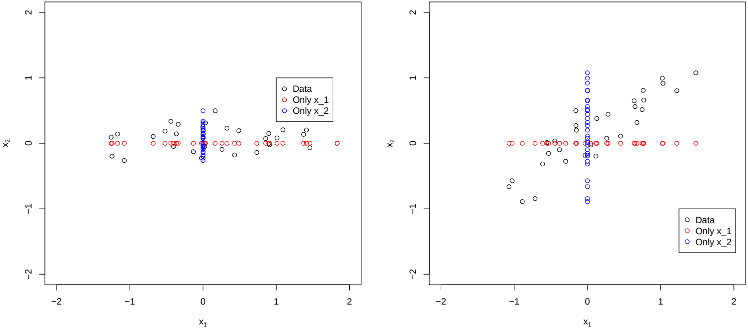
There isn't that much structurally different between the First and Second Examples

In fact, the Second Example is just a rotation of the First Example.



Lower dimensional embeddings: Comparison of Examples

If we knew how to rotate our data so that the Second Example looked like the First Example, we would be able to preserve more structure when doing model selection.



Lower dimensional embeddings: We Can!

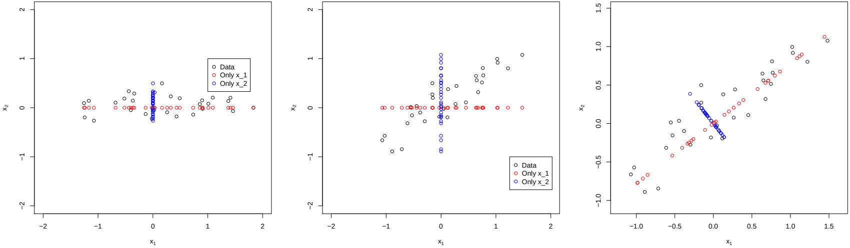
It turns out that Principal Components Analysis (PCA) gives us exactly this rotation.

I don't want to overwhelm you with definitions, so this is all I'll say about formally defining PCA

- PCA finds the rotation that maximizes the variance explained
- PCA finds the rotation that minimizes the squared error
- PCA can be computed by getting the SVD of $X - \overline{X} = UDV^\top$ (The UD are the principal components) and V is the rotation.

Lower dimensional embeddings: Comparison of Examples

Now, using the Principal components, we can again see that by setting PC2 to zero doesn't lose too much structure.



Caveat: Both X_1 and X_2 are mixed together inside both PC1 and PC2. So, this approach doesn't do variable selection, it does dimension reduction

PCA in R

```
PCA.out = prcomp(X,scale=TRUE)
```

or

```
PCA.out = princomp(X,scale=TRUE)
```

Only use prcomp, not princomp. Much more numerically stable!

We can also get the objects ourselves:

```
svd.out = svd(scale(X,scale=TRUE))
```

PCA in R

```
PCA.out = prcomp(X,scale=TRUE)
> names(PCA.out)
[1] "sdev"     "rotation" "center"   "scale"    "x"
> dim(X)
[1] 100  10
> dim(PCA.out$rotation)
[1] 10 10
> dim(PCA.out$x)
[1] 100  10
```

The coordinates of...

- the observations are in PCA.out$x

  (Known as scores)

- the covariates are in PCA.out$rotation

  (Known as loadings)

PCA in R

```
> PCA.out$rotation[1:2,1:3]
            PC1         PC2        PC3
[1,] -0.3797434 0.007642462 -0.3559232
[2,] -0.2505855 0.479266913 -0.1575462
> PCA.out$x[1:2,1:3]
            PC1        PC2        PC3
[1,] -1.3056426 -0.5296034 -0.9157294
[2,] -0.3535175  0.5285959  1.4482028

> svd.out$v[1:2,1:3]
            [,1]        [,2]       [,3]
[1,] -0.37974339 0.007642462 -0.3559232
[2,] -0.25058548 0.479266913 -0.1575462
> UD = svd.out$u %*% diag(svd.out$d)
> UD[1:2,1:3]
           [,1]       [,2]       [,3]
[1,] -1.3056426 -0.5296034 -0.9157294
[2,] -0.3535175  0.5285959  1.4482028
```

To scale or not scale?

If we do either

```
PCA.out = prcomp(X,scale=TRUE)
```

or

```
svd.out = svd(scale(X,scale=TRUE))
```

we need to decide whether to scale the covariates

(Important: always center the covariates)

As a general rule, scale if the covariates are measured in different units

The next set if lecture notes provide examples of when to scale