feature engineering

creating new features out of existing variables

- deal with categorical or missing data
- remove or mitigate outliers
- reduce dimensionality of data
- transformations for distribution fit of a model
- augment data with an additional related data set

missing data

- remove it
- replace it with an arbitrary value
- replace it with the mean, median, mode of column
- fit a linear (or random forest) model and predict missing values

categorical data

- treat each category as an indicator variable (presence or absence of category)
- create new column with interaction of two or more categories
- compute statistic of float column over each category

categorical data

- treat each category as an indicator variable (presence or absence of category)
- **create new column with interaction of two or more categories**
- compute statistic of float column over each category

Col 1	Col 2	New Col
Coke	M	CM
Pepsi	F	PF
Pepsi	M	PM
Coke	F	CF

categorical data

- treat each category as an indicator variable (presence or absence of category)
- create new column with interaction of two or more categories
- **compute statistic of float column over each category**

Col 1	Col 2	New Col
32	M	28
11	F	9.5
24	M	28
8	F	9.5

transformations

- whiten data (subtract mean, divide by standard deviation of column) - - more important for GLM

- log transform (log of data may be more Gaussian)

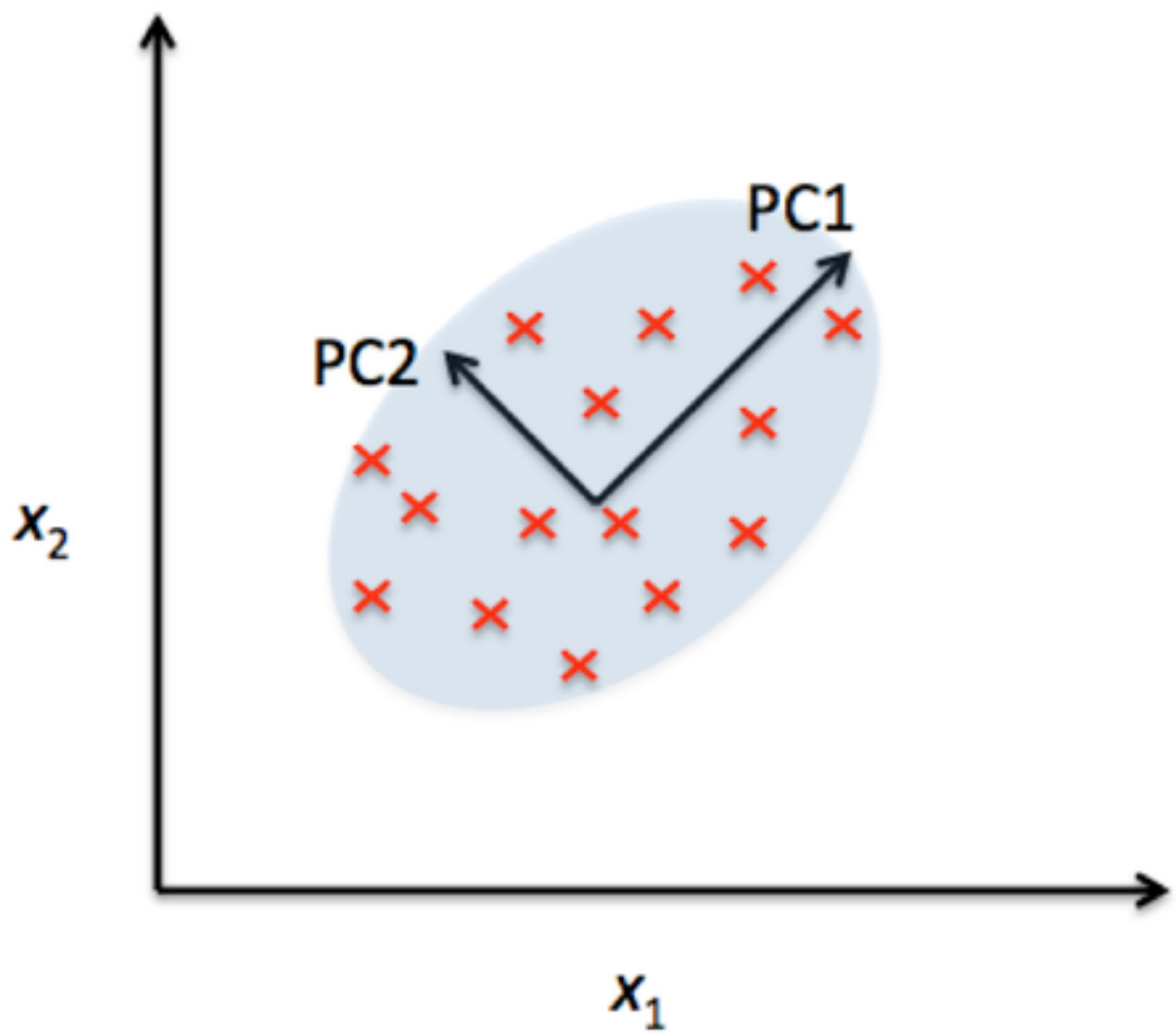
- remove rows that have outlier, or threshold them

- other transforms? maybe dividing two columns makes sense, etc

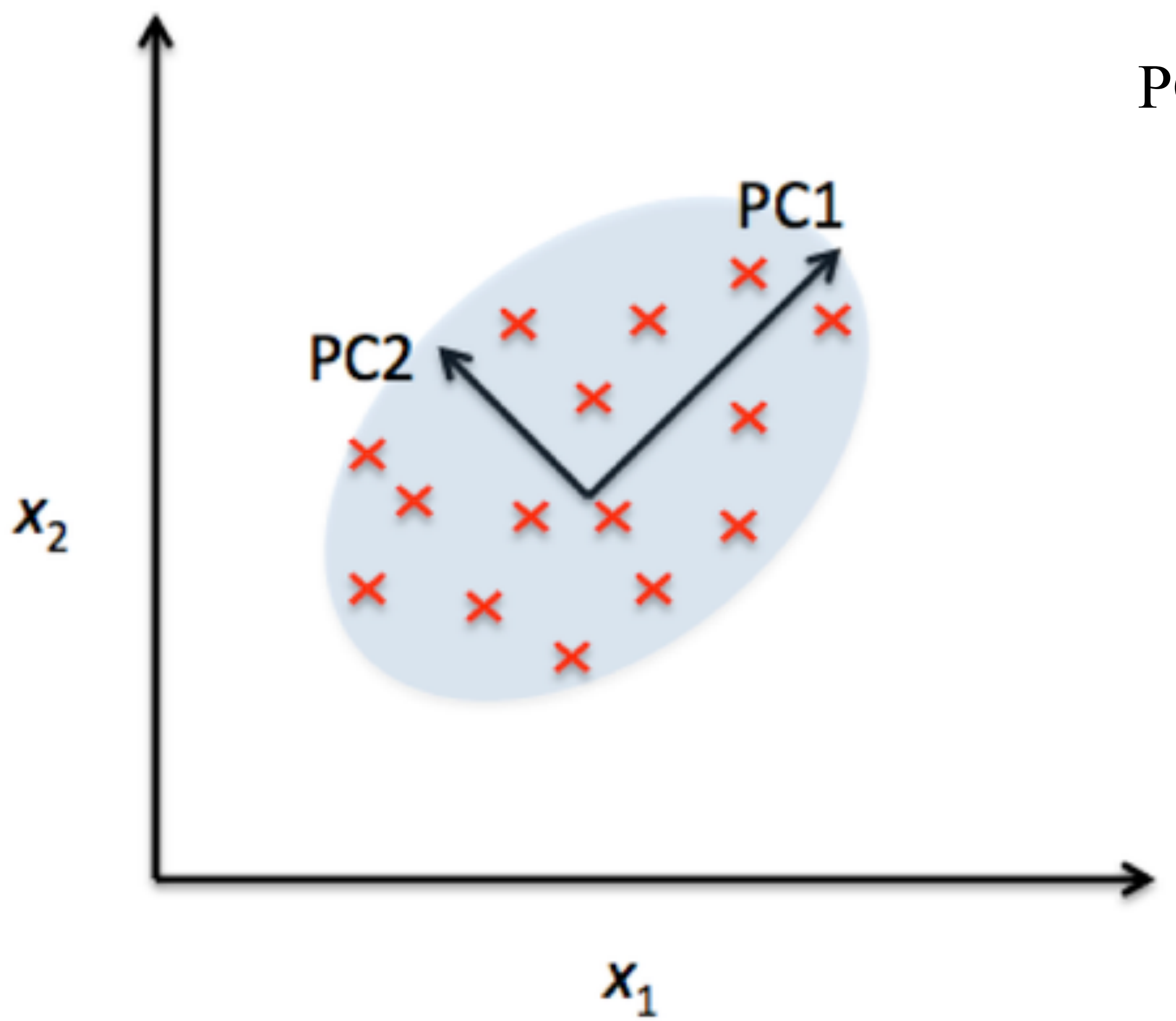
high dimensional data

- too many variables may cause over-fitting (GLM)
- too many variables may be expensive (decision trees)
- we may want to understand the data in a lower dimension
- we may want to group data together

principal component analysis (PCA)



principal component analysis (PCA)

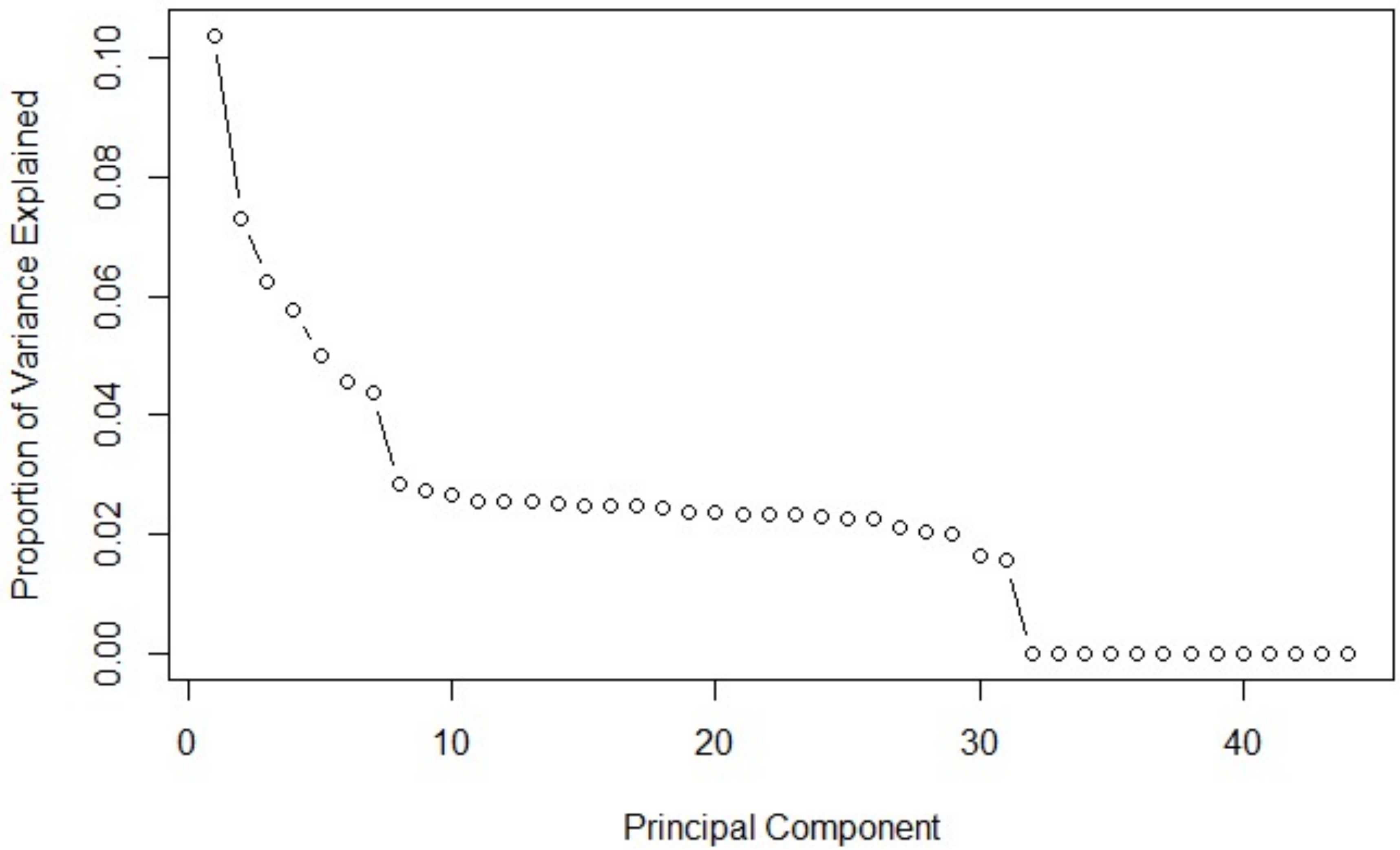


PCA is an orthogonal linear transformation (rotation) that transforms data to a new coordinate system. Greatest variance by lies on the first coordinate (*first principal component*), the second greatest variance lies on the second coordinate (*second principal component*), etc

computing PCA

- whiten data (subtract mean and divide by standard deviation for each column)
- compute covariance matrix tr(X)*X
- compute eigenvectors v_1,…,v_m and eigen values lam_1,…,lam_m
- first principal component is X*v1, second is X*v2, etc

select cutoff to reduce dimension



strengths/weaknesses

- simple to compute, method is easy to explain
- components are linear combination of original variables, this is just a data rotation and projection
- scale dependent
- R: prcomp package

sparse selection of features

- Forward selection: add features one at a time and keep them if they improve accuracy

- Backward selection: remove features and leave them out if accuracy improves

- These are both computationally expensive (must try all combinations)

- Sparse regression automatically selects features

generalized linear model (GLM)

$$\Theta = \theta_1 x_1 + \ldots + \theta_n x_n$$

$$y \sim f(\Theta)$$

- x are independent variables or “predictors”
- theta are model coefficients (linear)
- f is link function (for Gaussian linear regression f is the identity)

add regularization

$$\max_{\vec{\theta}} \sum_k \log(L(y_k, f(X_k\vec{\theta})) - \lambda\|\vec{\theta}\|$$

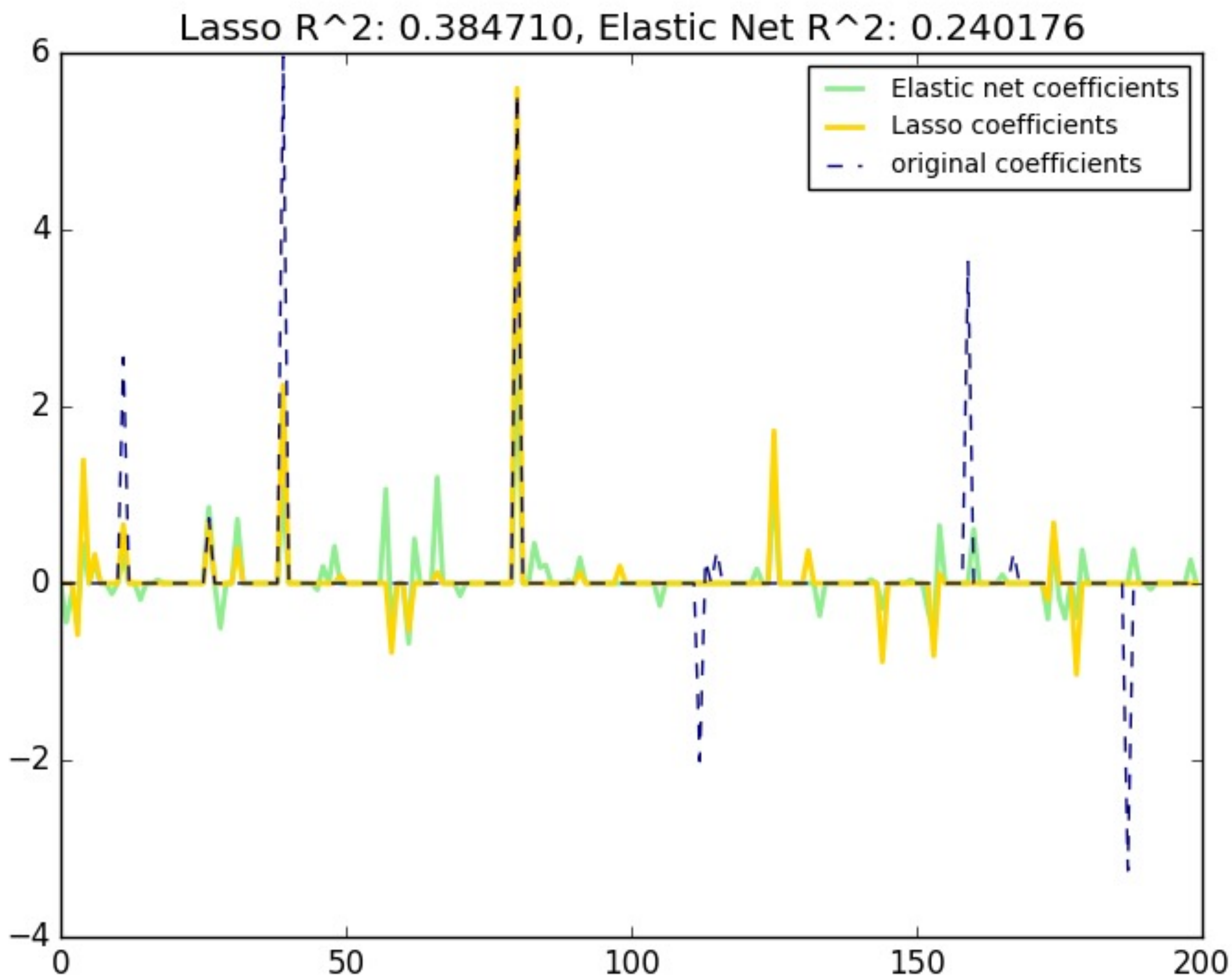
log likelihood

penalty
enforcing simpler
model

sparse regression

$$\|\theta\|_1 = \sum_{j=1}^{M} |\theta_j|$$

sparse regression



glmnet in R

```
model <- glmnet(x.train, y.train, family="gaussian", alpha=1)
```

alpha controls balance between l1 and l2 penalty

lambda controls amount of regularization (larger lambda, simpler model)

glmnet can find best lambda using cv.glmnet