REGRESSION PART 2: LINEAR MODELS

Hsin-Min Lu
盧信銘
台大資管系

Multiple Linear Regression Model

$$\boxed{Y_i = b_0 + b_1 X_1 + b_2 X_2 + \cdots + b_p X_p + e}$$

- The parameters in the linear regression model are easy to interpret.

- β_0 is the intercept (i.e. the average value for Y if all the X's are zero), β_j is the slope for the jth variable X_j

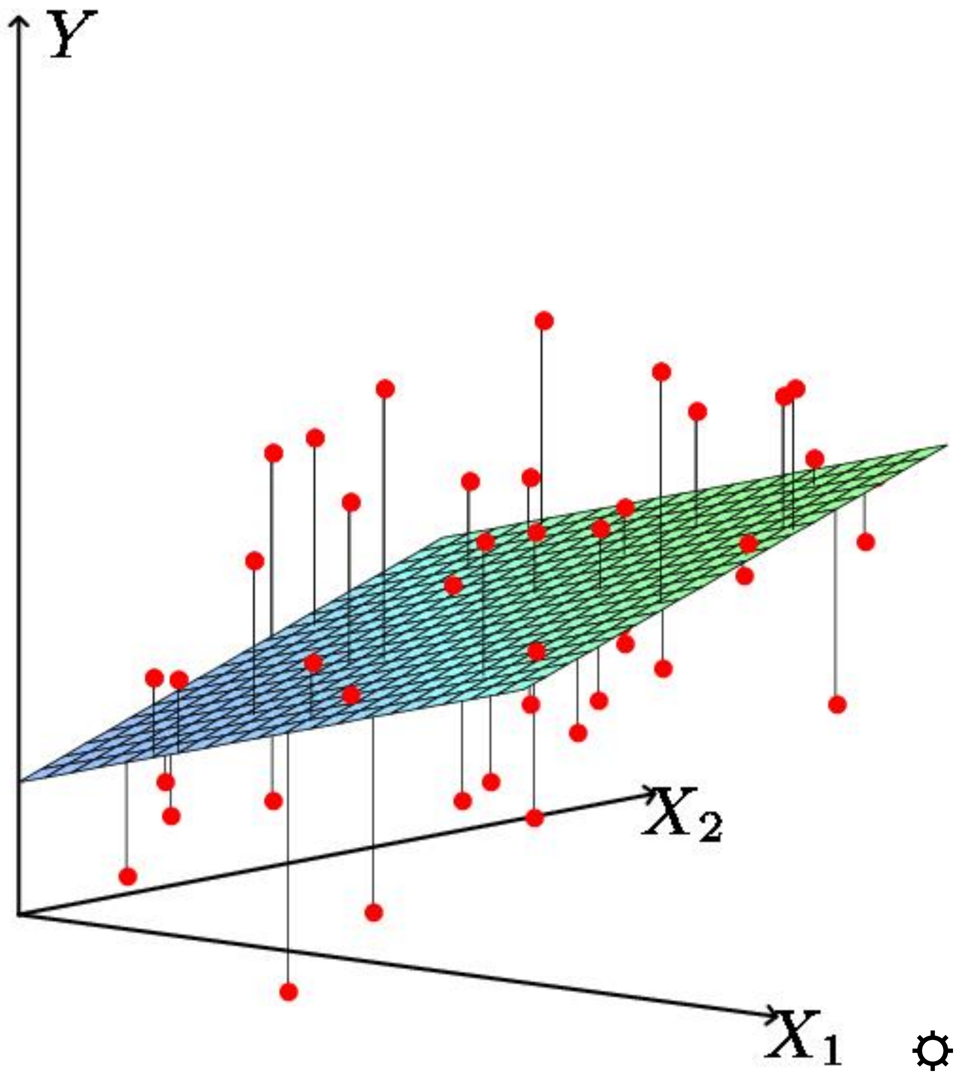
- β_j is the average increase in Y when X_j is increased by one and **all other X's are held constant.**

Least Squares Fit

- We estimate the parameters using least squares i.e. minimize

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2$$

$$= \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{\beta}_0 - \hat{\beta}_1 X_1 - \cdots - \hat{\beta}_p X_p\right)^2$$



Relationship Between Population and Least Squares Lines (Assuming we have the right model!)

Population line

$$\boxed{Y_i = b_0 + b_1 X_1 + b_2 X_2 + \cdots + b_p X_p + e}$$

Least Squares line

$$\boxed{\hat{Y}_i = \hat{b}_0 + \hat{b}_1 X_1 + \hat{b}_2 X_2 + \cdots + \hat{b}_p X_p}$$

- We would like to know β_0 through β_p i.e. the population line. Instead we know $\hat{\beta}_0$ through $\hat{\beta}_p$ i.e. the least squares line.
- Hence we use $\hat{\beta}_0$ through $\hat{\beta}_p$ as guesses for β_0 through β_p and $\hat{Y}_i$ as a guess for Y_i. The guesses will not be perfect just as $\bar{x}$ is not a perfect guess for μ.

Measures of Fit: R^2

- Some of the variation in Y can be explained by variation in the X's and some cannot.

- R^2 tells you the fraction of variance that can be explained by X.

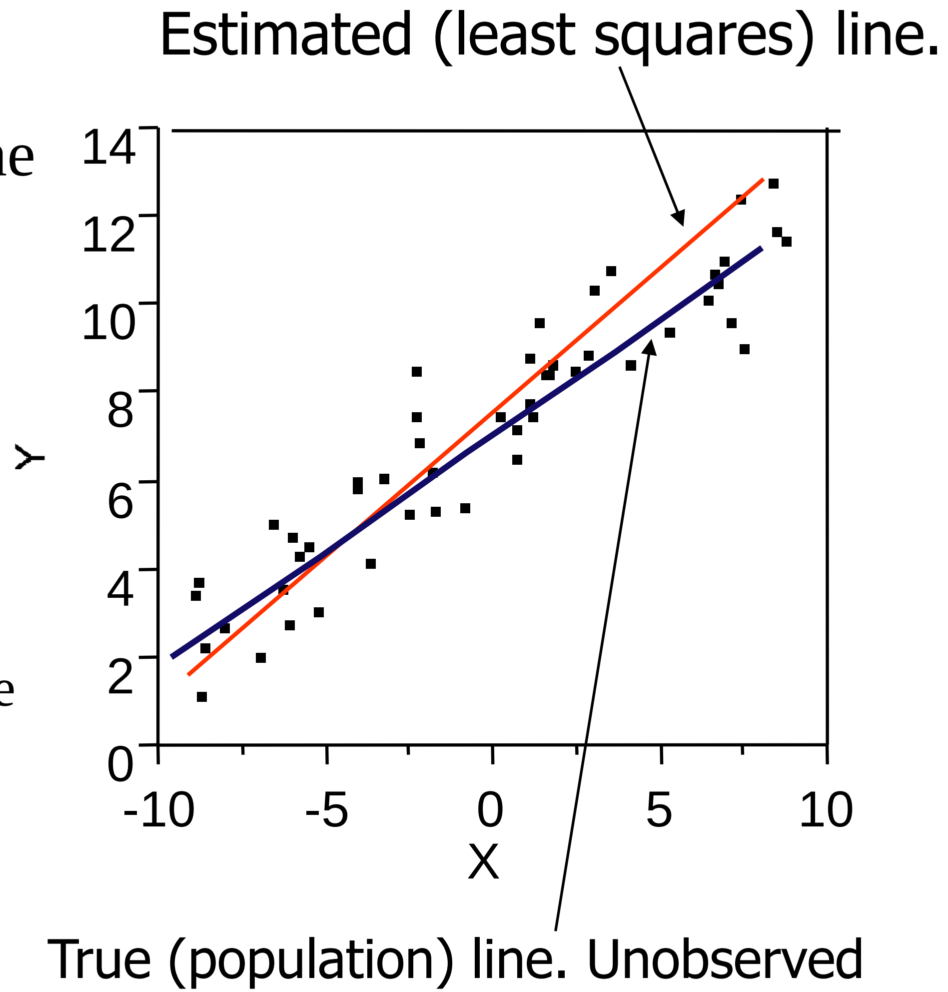
$$R^2 = 1 - \frac{RSS}{\sum (Y_i - \bar{Y})^2} \approx 1 - \frac{\text{Ending Variance}}{\text{Starting Variance}}$$

R^2 is always between 0 and 1. Zero means no variance has been explained. One means it has all been explained (perfect fit to the data).

Note: R^2 can be computed on training or testing data. The meaning is different.

Inference in Regression

- The regression line from the sample is not the regression line from the population.
- What we want to do:
  - Assess how well the line describes the plot.
  - Guess the slope of the population line.
  - Guess what value Y would take for a given X value



Some Relevant Questions

- Is $\beta_j=0$ or not? We can use a hypothesis test to answer this question. If we can't be sure that $\beta_j\neq 0$ then there is no point in using X_j as one of our predictors.

- Can we be sure that at least one of our X variables is a useful predictor i.e. is it the case that $\beta_1= \beta_2=\cdots= \beta_p=0$?

Linear Models and Least Squares

- N pairs of (x_i, y_i), $i = 1, 2, \dots, N$.
- x_i: features, y_i: outcome
- $x_i \in R^p$, $y_i \in R$
- Assume $N > p$.
- Linear model: $y_i = \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j + \epsilon_i$
  - with ϵ_i (white noise) IID, $E(\epsilon_i) = 0$, $Var(\epsilon_i) = \sigma^2$.

- We either assume the linear model is correct, or more realistically think of it as a linear approximation to the regression model $E(y_i|x_i) = f(x_i)$.

Minimizing RSS

- Residual Sum of Square (RSS)
- $RSS(\beta_0, \beta_1, \ldots, \beta_p) = \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2$
- Note: Given x_i, the predicted value $\hat{y}_i = \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j$
- The prediction error is $y_i - \hat{y}_i$.
- Thus RSS is the sum of squared prediction errors.

- Want: find $\beta_0, \beta_1, \ldots, \beta_p$ such that RSS is minimized.

Vector Notation

$$\mathbf{RSS}(\beta_0, \beta_1, \ldots, \beta_p) = \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 \quad (2)$$

- Absorb β_0 into β and augment the vector x_i with a 1.
- Write $Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}_{N\times 1}$, $X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{bmatrix}_{N\times(p+1)}$
- The coefficient vector becomes $\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix}$
- For observation i, $x_i^T = [1 \quad x_{i1} \quad x_{i2} \quad \cdots \quad x_{ip}]$
- $x_i^T\beta = \beta_0 + x_{i1}\beta_1 + x_{i2}\beta_2 + \cdots + x_{ip}\beta_p$
- The residual $e_i = y_i - x_i^T\beta$

RSS Revisited

- Now we can rewrite

$$RSS(\beta) = \sum_{i=1}^{N} \left(y_i - x_i^T \beta\right)^2 = \sum_{i=1}^{N} e_i^2$$

- Define $e = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_N \end{bmatrix}$, ➔ $RSS(\beta) = e^T e$

- Let $Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$ and note that $X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{bmatrix}$, so $X\beta = \begin{bmatrix} x_1^T \beta \\ x_2^T \beta \\ \vdots \\ x_n^T \beta \end{bmatrix}$

RSS Revisited (Cont'd.)

- Since $e_i = y_i - x_i^T\beta$, it is clear that
- $e = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_N \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} - \begin{bmatrix} x_1^T\beta \\ x_2^T\beta \\ \vdots \\ x_n^T\beta \end{bmatrix} = Y - X\beta$
- Recall that $RSS(\beta) = e^T e$

$$
\begin{aligned}
&= (Y - X\beta)^T(Y - X\beta) \\
&= (Y^T - \beta^T X^T)(Y - X\beta) \\
&= Y^T Y - Y^T X\beta - \beta^T X^T Y + \beta^T X^T X\beta \\
&= Y^T Y - 2Y^T X\beta + \beta^T X^T X\beta
\end{aligned}
$$

- Note the dimensions of each terms!

Minimizing $RSS(\beta)$

- We want to minimize $RSS(\beta)$ by selecting a good β
- This can be achieved by selecting a β such that

$$\frac{\partial RSS(\beta)}{\partial \beta} = 0$$

where $RSS(\beta) = Y^T Y - 2Y^T X\beta + \beta^T X^T X\beta$

- Here we need to differentiate $RSS(\beta)$ with respect to a matrix β

Review of Matrix Calculus

- $y = f(x)$, x and y are scalars, then $f'(x) = \frac{\partial y}{\partial x}$ and $f''(x) = \partial^2 y/\partial^2 x$

- Consider $y = f(x_1, x_2, \dots, x_n)$, $x_1, x_2, \dots, x_n, and\ y$ are scalars

- Let $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ and denote $y = f(x)$

Review of Matrix Calculus (cont'd.)

- Then $\frac{\partial f(x)}{\partial x} = \begin{bmatrix} \partial f/\partial x_1 \\ \partial f/\partial x_2 \\ \vdots \\ \partial f/\partial x_n \end{bmatrix}$

- $\frac{\partial f(x)}{\partial x^T} = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \cdots & \frac{\partial f}{\partial x_n} \end{bmatrix}$

Review of Matrix Calculus (cont'd.)

- Consider $x_i^T = [1 \quad x_{i1} \quad \cdots \quad x_{ip}]$ and $\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}$

$$g = x_i^T \beta = \beta_0 + \sum_{k=1}^{p} x_{ik} \beta_k$$

- What is $\frac{\partial g}{\partial \beta}$?

$$\frac{\partial g}{\partial \beta} = \begin{bmatrix} \partial g / \partial \beta_0 \\ \partial g / \partial \beta_1 \\ \vdots \\ \partial g / \partial \beta_p \end{bmatrix} = \begin{bmatrix} 1 \\ x_{i1} \\ \vdots \\ x_{ip} \end{bmatrix} = x_i$$

- To summarize: $\frac{\partial x_i^T \beta}{\partial \beta} = {x_i^T}^T = x_i$
- Also, $\frac{\partial \beta^T x_i}{\partial \beta} = x_i$ (check by yourself!)

Review of Matrix Calculus (cont'd.)

- How about $\frac{\partial(\beta^T X^T X\beta)}{\partial\beta}$?
- Recall the product rule: $\frac{\partial(f\ g)}{\partial x} = f'g + f g'$
- Apply the product rule in this case:
  - $\frac{\partial(\beta^T X^T X\beta)}{\partial\beta} = \mathrm{X}^{\mathrm{T}}\mathrm{X}\beta + (\beta^T X^T X)^T = 2X^T X\beta$

Minimizing $RSS(\beta)$ Revisited

- We want to minimize $RSS(\beta)$ by selecting a good β
- This can be achieved by selecting a β such that

$$\frac{\partial RSS(\beta)}{\partial \beta} = 0$$

where $RSS(\beta) = Y^T Y - 2Y^T X\beta + \beta^T X^T X\beta$

- $\frac{\partial RSS(\beta)}{\partial \beta} = \frac{Y^T Y - 2Y^T X\beta + \beta^T X^T X\beta}{\partial \beta} = 0 - 2\mathrm{X}^\mathrm{T}\mathrm{Y} + 2\mathrm{X}^\mathrm{T}\mathrm{X}\beta = 0$
  - ➔ $X^T X\beta = X^T Y$ ➔ $\beta = (X^T X)^{-1} X^T Y$, if $X^T X$ is nonsingular (this is true as long as the columns of X are linearly independent).
- We often call this solution $\hat{\beta}$. That is

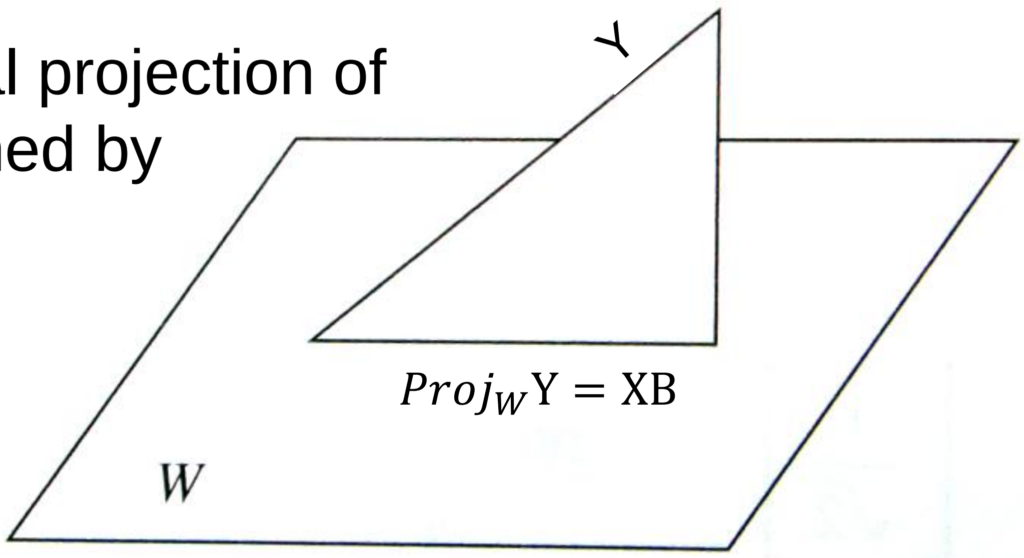
$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

Geometry of Least Squares

- Least Square Problem:

$$Y = X\beta + \epsilon$$

- $Proj_W$Y is the orthogonal projection of Y on to the space spanned by columns of X



W = Column space of X

Solving for $\hat{\beta}$

- $\hat{\beta} = (X^T X)^{-1} X^T Y$
- While we can compute $(X^T X)^{-1}$ directly in theory, most packages do not do this due to potential numerical unstable problem if columns of X are close to linearly dependent.
- To achieve a stable numerical solution, a standard practice is to use QR decomposition.
  - The computations are efficient and numerically stable.

Covariance of $\hat{\beta}$

- $\hat{\beta} = (X^T X)^{-1} X^T Y$
- $Var[\hat{\beta}] = Var[(X^T X)^{-1} X^T Y\ ]$
- $= Var[(X^T X)^{-1} X^T (X\beta + \epsilon)]$
- $= Var[\beta + (X^T X)^{-1} X^T \epsilon]$
- $= Var[(X^T X)^{-1} X^T \epsilon] = (X^T X)^{-1} X^T Var[\epsilon] X (X^T X)^{-1}$
- $= (X^T X)^{-1} X^T \sigma^2 I X (X^T X)^{-1}$
- $= (X^T X)^{-1} \sigma^2$
- That is, if $\epsilon \sim N(0, \sigma^2 I)$, then $\hat{\beta} \sim N(\beta, (X^T X)^{-1} \sigma^2)$
- This result gives us the way to conduct t-test for individual parameters.

t-test for $\hat{\beta}$

- $\beta = \left(\beta_0\ \beta_1\ \ldots \beta_p\right)^T$
- $\hat{\beta} = (X^T X)^{-1} X^T Y = \left(\hat{\beta}_0\ \hat{\beta}_1\ \ldots \hat{\beta}_p\right)^T$
- $\hat{\beta} \sim N(\beta, (X^T X)^{-1}\sigma^2)$, note that $(X^T X)^{-1}\sigma^2 \equiv \hat{\Sigma}$ is a square matrix $(p+1) \times (p+1)$.
- Substitute σ^2 with $\hat{\sigma}^2 = \frac{1}{N-p-1} \sum_{i=1}^{N} \left(y_i - x_i^T \hat{\beta}\right)^2$, an unbiased estimator for σ^2.
- H_0: $\beta_i = 0$; H_1: $\beta_i \neq 0$
- t-statistics: $\frac{\widehat{\beta}_i - 0}{\widehat{\Sigma}_{ii}} \sim t_{N-p-1}$

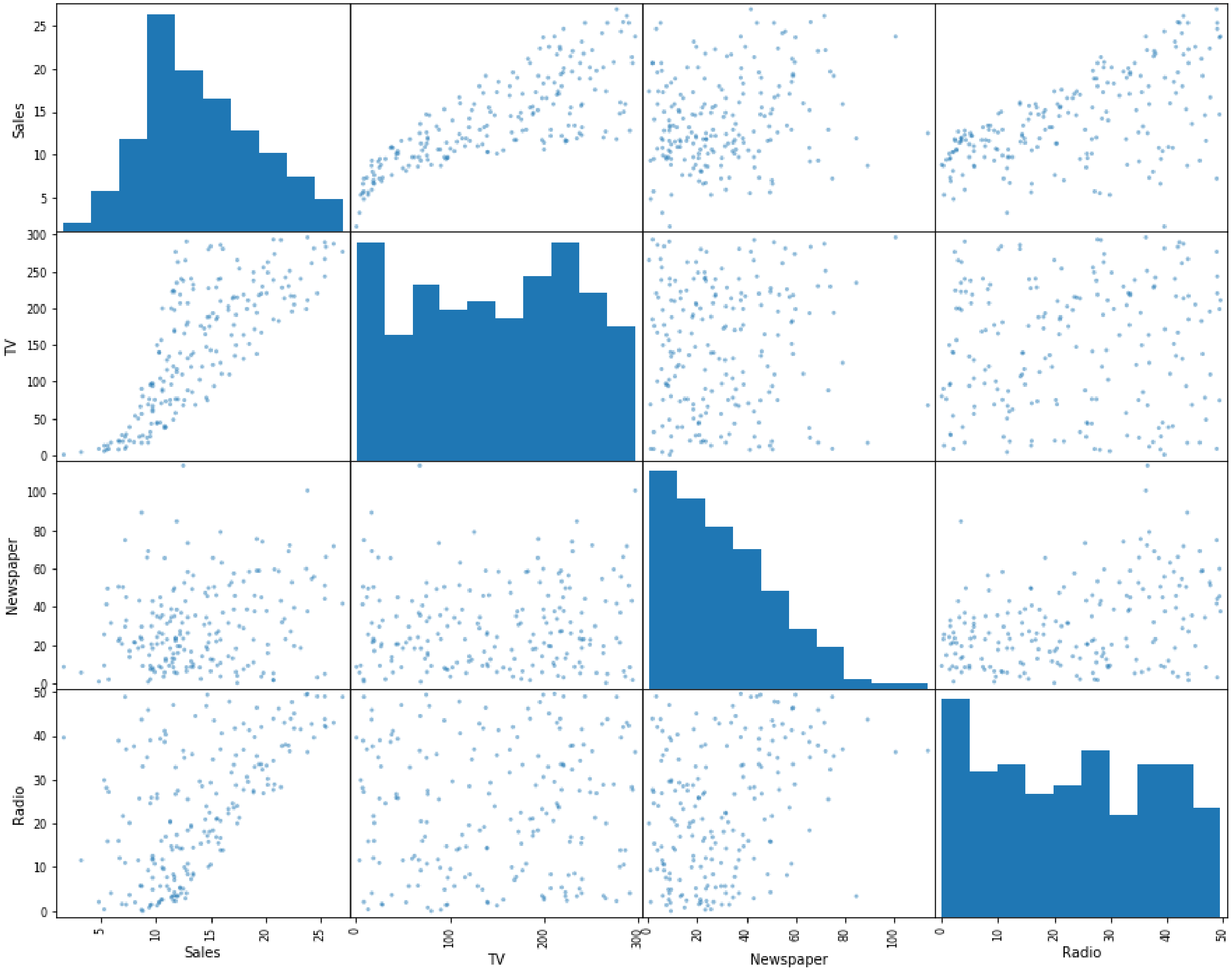
Example: Advertising Dataset

- 200 data points of sales given different combination of budgets on TV, Radio, and Newspaper
- We usually include a constant term in regression model.
- Thus, the X matrix looks like this:
- Note the first column is all ones

const	TV	Radio	Newspape
1	230.1	37.8	69.2
1	44.5	39.3	45.1
1	17.2	45.9	69.3
1	151.5	41.3	58.5
1	180.8	10.8	58.4
1	8.7	48.9	75
1	57.5	32.8	23.5
1	120.2	19.6	11.6
1	8.6	2.1	1

```python
import pandas as pd
df1 = pd.read_csv('Advertising.csv') df1.head()
#======
from pandas.plotting import scatter_matrix
attributes = ['Sales', 'TV', 'Newspaper', 'Radio']
_ = scatter_matrix(df1[attributes], figsize = (15, 12))
```

Plotting the Data



Linear Regression Results

```python
import statsmodels.api as sm
model = sm.OLS(df1['Sales'], sm.tools.add_constant(df1[['TV', 'Newspaper', 'Radio']])).fit()
model.summary()
```

Dep. Variable:	Sales	**R-squared:**	0.897
Model:	OLS	**Adj. R-squared:**	0.896
Method:	Least Squares	**F-statistic:**	570.3
Date:	Fri, 01 Feb 2019	**Prob (F-statistic):**	1.58e-96
Time:	12:20:22	**Log-Likelihood:**	-386.18
No. Observations:	200	**AIC:**	780.4
Df Residuals:	196	**BIC:**	793.6
Df Model:	3		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	2.9389	0.312	9.422	0.000	2.324	3.554
TV	0.0458	0.001	32.809	0.000	0.043	0.049
Newspaper	-0.0010	0.006	-0.177	0.860	-0.013	0.011
Radio	0.1885	0.009	21.893	0.000	0.172	0.206

Testing Individual Variables

Is there a (statistically detectable) linear relationship between Newspapers and Sales after all the other variables have been accounted for?

Regression coefficients

	Coefficient	Std Err	t-value	p-value
Constant	2.9389	0.3119	9.4223	0.0000
TV	0.0458	0.0014	32.8086	0.0000
Radio	0.1885	0.0086	21.8935	0.0000
Newspaper	-0.0010	0.0059	-0.1767	0.8599

No: big p-value

Regression coefficients

	Coefficient	Std Err	t-value	p-value
Constant	12.3514	0.6214	19.8761	0.0000
Newspaper	0.0547	0.0166	3.2996	0.0011

Small p-value in simple regression

Almost all the explaining that Newspapers could do in simple regression has already been done by TV and Radio in multiple regression!

2. Is the whole regression explaining anything at all?

$$F = \frac{(\mathrm{TSS} - \mathrm{RSS})/p}{\mathrm{RSS}/(n-p-1)} \sim F_{p,n-p-1}$$

- Test for:
  - H_0: all slopes = 0 $(\beta_1=\beta_2=\cdots=\beta_p=0)$,
  - H_a: at least one slope $\neq$ 0

ANOVA Table

Source	df	SS	MS	F	p-value
Explained	2	4860.2347	2430.1174	859.6177	0.0000
Unexplained	197	556.9140	2.8270		

Answer comes from the F test in the ANOVA (ANalysis Of VAriance) table.
The ANOVA table has many pieces of information. What we care about is the F Ratio and the corresponding p-value.

Users Beware

- You should not claim any causality relations between Y and X.
- Usual interpretation of coefficients: “other things being equal,” a unit change in x_i is associated with β_i changes in y_i.
- This interpretation is not always reasonable.

- If features among x_i are highly correlated, then the natural of training data did not allow us to have “other things being equal” interpretation.

Two Famous Quotes

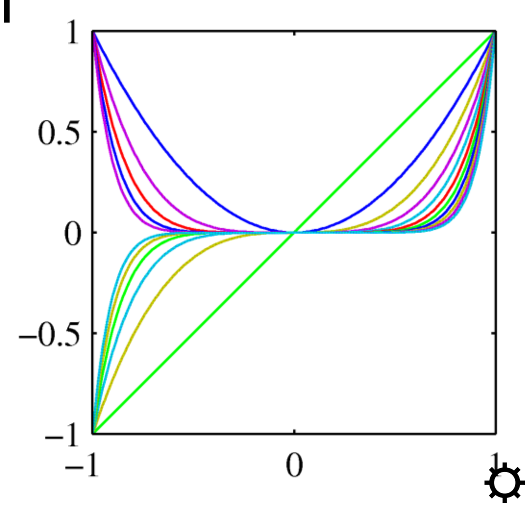
- Essentially, all models are wrong, but some are useful.
  - George Box

- The only way to find out what will happen when a complex system is disturbed is to disturb the system, not merely to observe it passively.
  - Fred Mosteller and John Tucky, paraphrasing Geroge Box

Enriching the input features

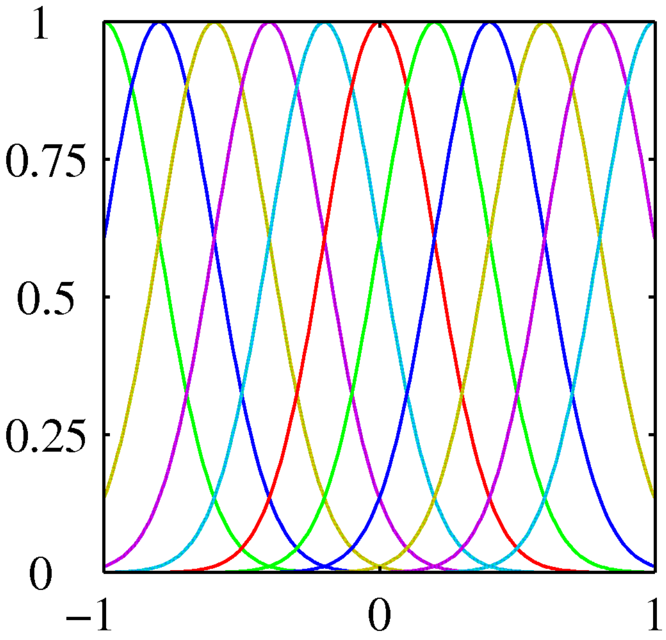
- One way to do “feature engineering.”
- We can incorporate non-linear features through several different types of basis functions
- A common example is polynomial functions
- $y = \mathrm{w}_0 + \sum w_i x_i + \sum w_{2,i} x_i^2 + \sum w_{3,i} x_i^3 + \cdots$
- Can also add cross-product terms: $x_i^a x_j^b$
- We adopted a general notation for this setting:
- $y(x, w) = \sum_{j=0}^{M-1} w_j \phi_j(x) = w^T \phi(x)$
- $\phi(x)$ is called the basis function.
  - Usually $\phi_0(x) = 1$
  - $x = (x_1, x_2, \dots, x_p)^T$

$$\phi(\mathbf{x}) = \begin{pmatrix} \phi_0(\mathbf{x}) \\ \phi_1(\mathbf{x}) \\ \vdots \\ \phi_{M-1}(\mathbf{x}) \end{pmatrix}$$



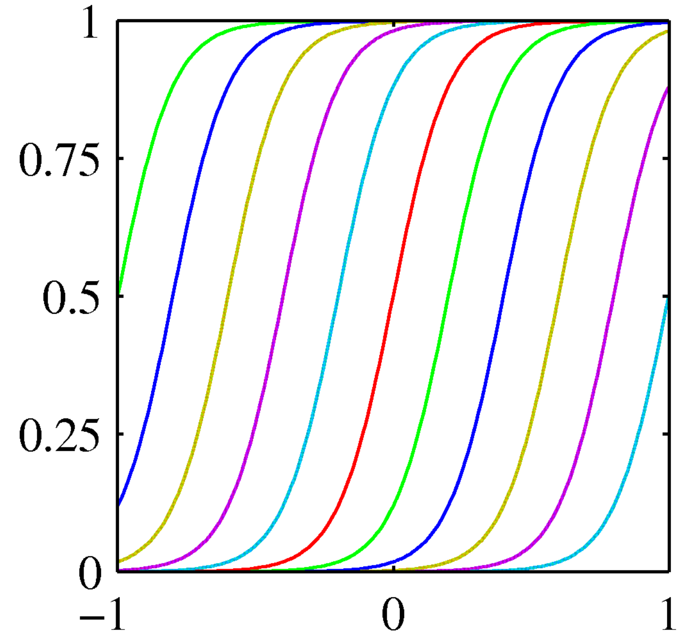
Gaussian Basis function

- Gaussian basis functions:
- $\phi_{a,j}(x) = \exp\left\{-\frac{(x_a-\mu_j)^2}{2s^2}\right\}$
- These are "local features"
- A small change in x_a only affect nearby basis functions.
- μ_j and s control location and scale (width).



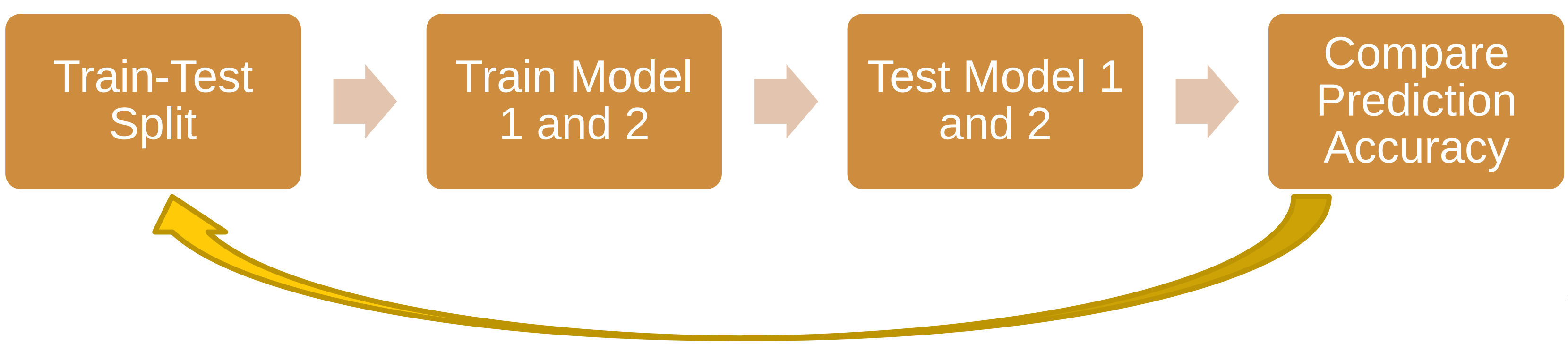
Sigmoidal Basis Function

- Sigmoidal basis functions:
- $\phi_{a,j} = \sigma\left(\frac{x_a - \mu_j}{s}\right)$
- where $\sigma(a) = \frac{1}{1+\exp(-a)}$
- Also these are local; a small change in x only affect nearby basis functions.
- μ_j and s control location and scale (slope).



Example: How useful is the Gaussian Basis Functions?

- We want to know how useful is the Gaussian basis function for sales prediction using the previous dataset (TV, Newspaper, and Radio advertisement).
- We are going to focus on prediction improvement.
- Model one: Sales~TV+Newspaper+Radio
- Model two: Sales~TV+Newspaper+Radio+Features from Gaussian Basis Functions
- Overall design for prediction performance evaluation:



Train-Test Split

- Need to reserve testing dataset that is not used for model training.
- E.g.: 80% training, 20% testing.
- Each model will be trained and tested using the same split.
- Reduce the noise of sampling variation.
- Compute the performance difference of different models.
- Repeat the process for several times (e.g. 10 times)
- Using t-test to see whether the difference is statistically meaningful
- Performance measure: Root Mean Squared Error (RMSE)

Using Gaussian Basis Function

- Recall: $\phi_{a,j}(x) = \exp\left\{-\frac{(x_a-\mu_j)^2}{2s^2}\right\}$
- Need to determine μ_j for each x_a.
- Need to determine how many nodes (i.e. # of μ_j) to use
- A tuning parameter that need to be selected using data driving approach.
- Will set it to a predefined value (4), more about parameter tuning later.
- Need to select values of μ_j.
- Simply setting μ_j to equal percentile values, but skipping the extreme values

Using Gaussian Basis Function (Cont’d.)

- For example, using 4 nodes, set values to 1%, 33.67%, 66.33%, 99% percentiles.
- Set s to $(x_{99\%} - x_{1\%})/4$
- For each node, generate additional feature value for each observation: $\phi_{a,j}(x) = \exp\left\{-\frac{(x_a - \mu_j)^2}{2s^2}\right\}$

Using Gaussian Basis Function (Cont'd.)

```
allfeatures = ['TV', 'Newspaper', 'Radio']
allfeatures2 = allfeatures.copy()
for focal_x in allfeatures:
    nnode = 4
    node1 = np.linspace(0.01, 0.99, num = nnode)
    gauss_mean = df1[focal_x].quantile(node1)
    #width
    s1 = (gauss_mean.max() - gauss_mean.min()) / nnode
    print("%s s1 = %f" % (focal_x, s1))

    for ii in range(nnode):
        am = gauss_mean.iloc[ii]
        newf = np.exp(-(df1[focal_x] - am)**2/(2*s1**2))
        newname = "%s_%d" % (focal_x, ii)
        df1[newname] = newf
        allfeatures2.append(newname)
```

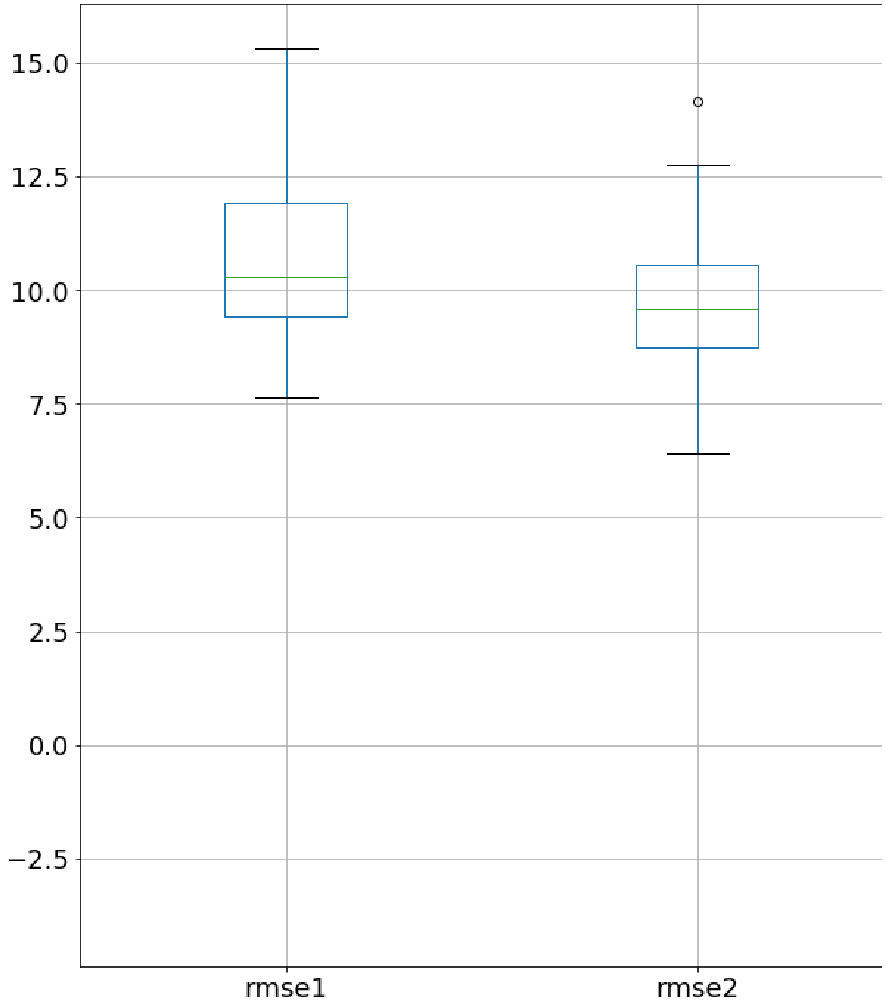
Running the Experiments

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

nrepeat = 100
rmse1all = [] #using original features
rmse2all = [] #using augmented features
for runid in range(nrepeat):
    train_set, test_set = train_test_split(df1, test_size=0.2,
                                           random_state=55 + runid)
    lin_reg = LinearRegression()
    lin_reg.fit(train_set[allfeatures], train_set['Sales'])
    ypred = lin_reg.predict(test_set[allfeatures])
    ytrue = test_set['Sales']
    rmse1 = np.sqrt(np.sum((ytrue - ypred)**2))
    rmse1all.append(rmse1)

    lin_reg2 = LinearRegression()
    lin_reg2.fit(train_set[allfeatures2], train_set['Sales'])
    ypred2 = lin_reg2.predict(test_set[allfeatures2])
    rmse2 = np.sqrt(np.sum((ytrue - ypred2)**2))
    rmse2all.append(rmse2)
```

Results



- rmse2 – rmse1 has a t-value of -7.81, which is statistically significant at a 95% confidence level
- Augmented features can reduce testing RMSE

Should We Always Prefer More Features?

- In the previous example, we have seen that additional features allow us to capture additional variations of the outcome, and thus provides better prediction.
- The key question is should we always prefer more features when constructing a model?
- Is there any drawback when we add large amounts of features?