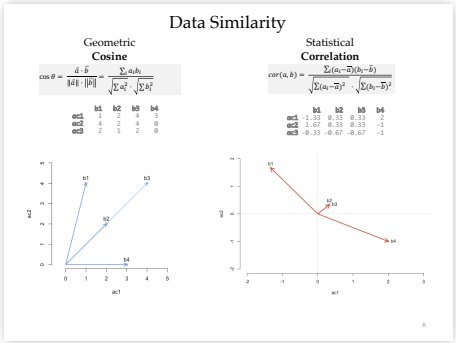
Business Analytics Using
Computational Statistics

Week 7
Data Similarity

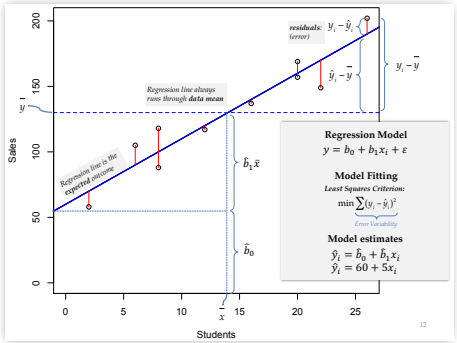
Week 8
Linear Regression

Week 9
Applied Regression

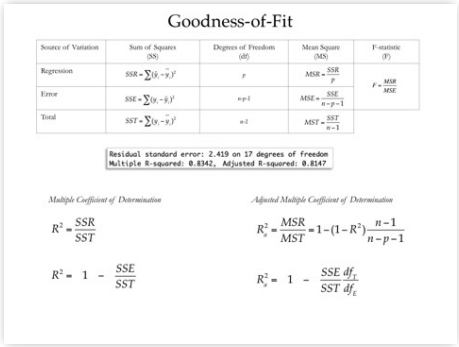
Correlation



Regression Statistics



Regression Geometry



Big Data and Performance

Loading and processing data with many rows x columns is memory and processor intensive

Loading Big Data

Look for high-performance packages written in C when using large datasets in R.

data.frame: *base R library for data tables*

```
system.time( read.csv("piccollage_accounts_bundles.csv", header=TRUE) )
   user  system elapsed
  0.501   0.015   0.516
```

data.table: *high performance package for data tables*

```
library(data.table)
system.time( fread("piccollage_accounts_bundles.csv") )
   user  system elapsed
  0.053   0.002   0.055
```

data.table is substantially faster than data.frame for loading and processing big tables

```
ac_bundles_dt <- fread("piccollage_accounts_bundles.csv")
ac_bundles_matrix <- as.matrix(ac_bundles_dt[, -1])
```

matrix *data structure required for cosine formula*

Sparse Data

sparse *- data that is mostly empty (0 or NA)*

```
library(lsa)
system.time( cosine(ac_bundles_matrix) )
   user  system elapsed
  4.414   1.075   5.491
```

Most of our vector multiplications will yield 0 – it would be faster to just skip them!

	Maroon5V	between	pellington	StickerLite	saintvalentine
[1,]	0	0	0	173	0
[2,]	0	0	0	1	0
[3,]	11	1	7	79	2
[4,]	0	0	0	4	0
[5,]	0	0	0	2	0
[6,]	0	0	0	9	0
[7,]	0	0	0	0	0
[8,]	0	0	0	6	0
[9,]	0	0	0	2	0
[10,]	0	0	0	0	0

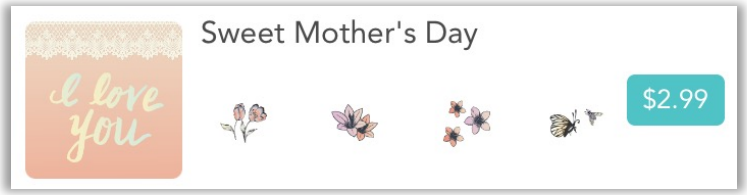
qlcMatrix: *processing and analysis of sparse matrices*

```
remotes::install_github("cysouw/qlcMatrix")
library(qlcMatrix)
system.time( qlcMatrix::cosSparse(ac_bundles_matrix) )
   user  system elapsed
  0.009   0.002   0.011
```

qlcMatrix::cosSparse(…) optimizes performance by taking advantage of sparsity in data

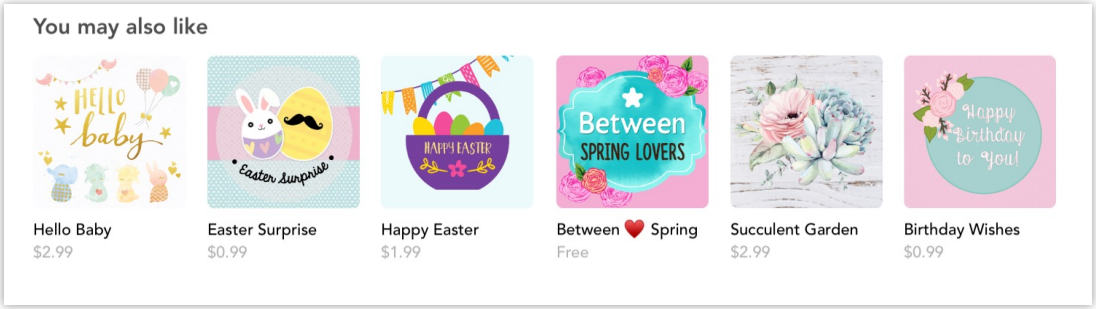
Recommendations

Our pick of sticker bundle



Let's pick "***sweetmothersday***"

Designer recommendations



🤔

What are designers' criteria for recommendations?

Our Intuitive guesses

Find similar themes and keywords
- ***grep**(…) searches for partial matches of text*

Pipe symbol '|' means 'or'

```
terms <- "mom|mother|dad|father|baba|mama"

bundle_names <- colnames(ac_bundles_matrix)

grep(terms, bundle_names, ignore.case=TRUE, value=TRUE)
```

```
 [1] "Emome"           "toMomwithLove"   "Mom2013"         "Dad2013"
 [5] "bestdaddy"       "sweetmothersday" "superherodad2"
```

🤔

Why are we "recommending" other bundles?

Developing a **Cosine Recommendation** Function

Cosine Recommendations

```
cosine_recos <- function(items_matrix) {

  cos_sim_matrix <- qlcMatrix::cosSparse(items_matrix)
  bundle_names <- colnames(items_matrix)
  dimnames(cos_sim_matrix) <- list(bundle_names, bundle_names)

    Make diagonals 2.0 – an impossible value for cosine!
     diag(…) – get or set the diagonals of a matrix
  diag(cos_sim_matrix) <- 2

    Sort recommendations for each row
    Remove self-recommendations (2.0 values)
    Return only five recommendations
  row_recos <- function(cos_sim_row) {
    names(sort(cos_sim_row, decreasing = TRUE))
  }

  all_recos <- t(apply(cos_sim_matrix, 2, row_recos))

    Remove first column (2.0 values) from reco matrix
  final_recos <- all_recos[, -1]

  return(final_recos[, 1:5])
}
```

```
     Maroon5V between pellington StickerLite saintvalentine
[1,]        0       0          0         173              0
[2,]        0       0          0           1              0
[3,]       11       1          7          79              2
[4,]        0       0          0           4              0
[5,]        0       0          0           2              0
```

```
             Maroon5V   between pellington StickerLite saintvalentine
Maroon5V       1.0000000 1.0000000  1.0000000   0.4152664      1.0000000
between        1.0000000 1.0000000  1.0000000   0.4152664      1.0000000
pellington     1.0000000 1.0000000  1.0000000   0.4152664      1.0000000
StickerLite    0.4152664 0.4152664  0.4152664   1.0000000      0.4152664
saintvalentine 1.0000000 1.0000000  1.0000000   0.4152664      1.0000000
```

```
             Maroon5V   between pellington StickerLite saintvalentine
Maroon5V       2.0000000 1.0000000  1.0000000   0.4152664      1.0000000
between        1.0000000 2.0000000  1.0000000   0.4152664      1.0000000
pellington     1.0000000 1.0000000  2.0000000   0.4152664      1.0000000
StickerLite    0.4152664 0.4152664  0.4152664   2.0000000      0.4152664
saintvalentine 1.0000000 1.0000000  1.0000000   0.4152664      2.0000000
```

```
               [,1]           [,2]        [,3]          [,4]             [,5]
Maroon5V       "Maroon5V"     "between"   "pellington"  "saintvalentine" "StickerLite"
between        "between"      "Maroon5V"  "pellington"  "saintvalentine" "StickerLite"
pellington     "pellington"   "Maroon5V"  "between"     "saintvalentine" "StickerLite"
StickerLite    "StickerLite"  "Maroon5V"  "between"     "pellington"     "saintvalentine"
saintvalentine "saintvalentine" "Maroon5V" "between"     "pellington"     "StickerLite"
```

```
               [,1]        [,2]          [,3]             [,4]
Maroon5V       "between"   "pellington"  "saintvalentine" "StickerLite"
between        "Maroon5V"  "pellington"  "saintvalentine" "StickerLite"
pellington     "Maroon5V"  "between"     "saintvalentine" "StickerLite"
StickerLite    "Maroon5V"  "between"     "pellington"     "saintvalentine"
saintvalentine "Maroon5V"  "between"     "pellington"     "StickerLite"
```

Similarity-based Recommendations

Cosine Similarity Recommendations

```
recos_cos <- cosine_recos(ac_bundles_matrix)
recos_cos["sweetmothersday", ]
  [1] "mmlm"            "julyfourth"      "tropicalparadise"
  [4] "bestdaddy"       "justmytype"
```

Correlation Recommendations

```
bundle_means <- apply(ac_bundles_matrix, 2, mean)
bundle_means_matrix <- t(replicate(nrow(ac_bundles_matrix), bundle_means))
ac_bundles_mc_b <- ac_bundles_matrix - bundle_means_matrix

recos_cor <- cosine_recos(ac_bundles_mc_b)
recos_cor["sweetmothersday", ]
  [1] "tropicalparadise" "mmlm"            "julyfourth"
  [4] "bestdaddy"        "justmytype"
```

Adjusts for differences between bundles

- *mean-centers <u>columns</u> before cosine*
- *each number becomes relative to the average usage of a given bundle*

Adjusted-Cosine Recommendations

```
account_means <- apply(ac_bundles_matrix, 1, mean)
account_means_matrix <- replicate(ncol(ac_bundles_matrix), account_means)
ac_bundles_mc_ac <- ac_bundles_matrix - account_means_matrix

recos_adj_cos <- cosine_recos(ac_bundles_mc_ac)
recos_adj_cos["sweetmothersday", ]
  [1] "justmytype" "julyfourth" "gudetama"   "mmlm"       "bestdaddy"
```

Adjusts for differences between accounts

- *mean-centers <u>rows</u> before cosine*
- *each number becomes relative to the average usage of a particular account*

The **Duality** of Geometry and Statistics

Two approaches that produce the same results, but use different methods/approaches

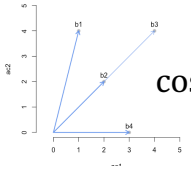
Computational methods can be viewed from either

Statistical Perspective

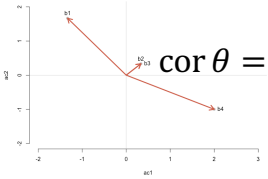
$$r_{xy} = \frac{\text{cov}_{xy}}{s_x s_y} = \frac{SS_{xy}}{\sqrt{SS_x SS_y}} = \frac{\sum_i (x - \bar{x})(y - \bar{y})}{\sqrt{\sum (x - \bar{x})^2} \cdot \sqrt{\sum (y - \bar{y})^2}}$$

Geometric Perspective

(linear algebra)



$$\cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \cdot \|\vec{b}\|} = \frac{\sum_i a_i b_i}{\sqrt{\sum a_i^2} \cdot \sqrt{\sum b_i^2}}$$



$$\text{cor}\,\theta = \frac{\overrightarrow{(a - \bar{a})} \cdot \overrightarrow{(b - \bar{b})}}{\left\|\overrightarrow{(a - \bar{a})}\right\| \cdot \left\|\overrightarrow{(b - \bar{b})}\right\|} = \frac{\sum_i (a_i - \bar{a})(b_i - \bar{b})}{\sqrt{\sum (a_i - \bar{a})^2} \cdot \sqrt{\sum (b - \bar{b})^2}}$$

*Which one do **you** find more intuitive?*

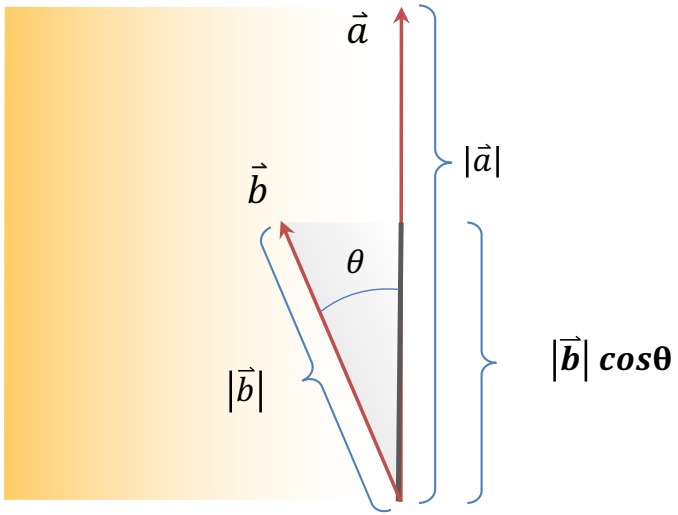
*Which one does the **computer** use?*

The **Geometry** of Similarity

The Dot Product $\vec{a} \cdot \vec{b} = |\vec{a}||\vec{b}|cos\theta$

Recall:

$$\cos\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\cdot\|\vec{b}\|}$$



Projection of a vector onto another

geometrically *or* ***statistically***

(e.g., in correlation)

$|\vec{a}||\vec{b}|cos\theta$

The product of length of $\vec{a}$ *in the direction of* $\vec{b}$

$\sum(a_i - \bar{a})(b_i - \bar{b})$

The product of deviations of a and b from their means

captures agreement in all their component directions

captures agreement in dispersion between columns

Cosine vs. Correlation

Zero value columns

*Cosine **cannot compare** zero columns*

```
cosine(c(0,0), c(3,2))

     [,1]
[1,]  NaN
```

*Correlation **cannot compare** zero columns*

```
cor(c(0,0), c(3,2))

[1] NA
Warning message: the standard deviation is zero
```

Identically rated items (constant columns)

```
scores <- rbind(c(1,5,3,2), c(2,4,3,9), c(4,7,3,2))
     [,1] [,2] [,3] [,4]
[1,]    1    5    3    2
[2,]    2    4    3    9
[3,]    4    7    3    2
```

*Cosine **can use** identically rated items*

```
cosine(scores)

          [,1] [,2] [,3] [,4]
[1,] 1.00 0.94 0.88 0.65
[2,] 0.94 1.00 0.97 0.67
[3,] 0.88 0.97 1.00 0.80
[4,] 0.65 0.67 0.80 1.00
```

*Correlation **cannot use** identically rated items*

```
cor(scores)

      [,1]  [,2] [,3]  [,4]
[1,]  1.00  0.79   NA -0.19
[2,]  0.79  1.00   NA -0.76
[3,]    NA    NA    1    NA
[4,] -0.19 -0.76   NA  1.00

Warning message:
 the standard deviation is zero
```

Cosine vs. Correlation

Shift Invariance

*Cosine **varies with shift** in data!*

```
cosine(b1, b2)

       [,1]
  [1,] 0.87

cosine(b1 + 2, b2)
       [,1]
  [1,] 0.94
```

*Correlation is **invariant to shift** in data*

```
cor(b1, b2)

  [1] 0.19

cor(b1 + 2, b2)

  [1] 0.19
```

Scaling Invariance

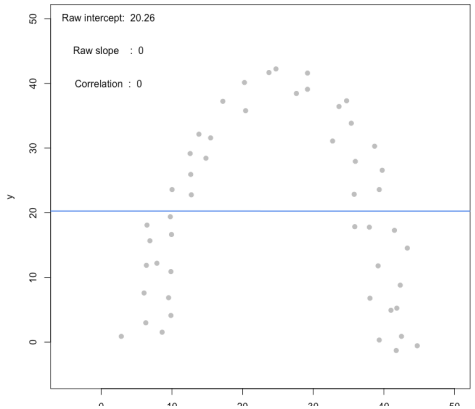
*Cosine is **invariant to scaling** in data!*

```
cosine(b1, b2)

       [,1]
  [1,] 0.87

cosine(b1 * 2, b2)

       [,1]
  [1,] 0.87
```

*Correlation is **invariant to scaling** in data*

```
cor(b1, b2)

  [1] 0.19

cor(b1 * 2, b2)

  [1] 0.19
```

Misleading Correlations

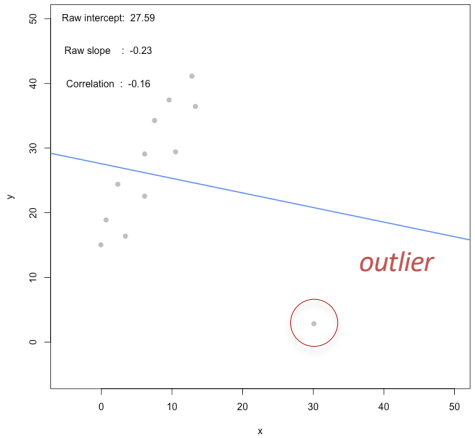


Misleading zero correlation

Correlation is a linear measure of similarity between variables (e.g., x and y)

Non-linear relationships require transforming the variables (e.g., $\mathbf{x^2}$ and y)

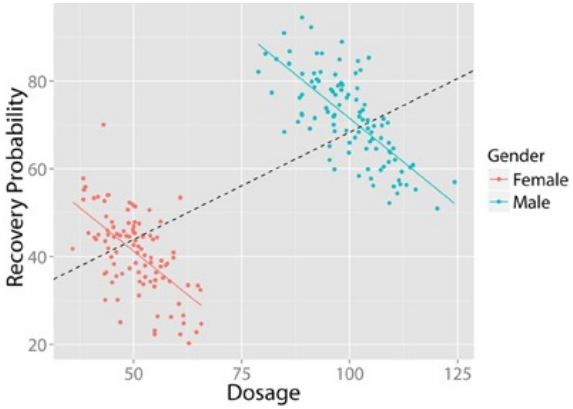
🤔 *How can know whether our correlation analysis is misfit with a data set?*



Outliers

Influential points in the data can produce unexpected correlations

Example
A single outlier can significantly change the correlation of a data set



Simpson's Paradox

Local sub-groups may have different correlations than a combined dataset

Example: How dosage level of medicine is related to recovery probability.
***Altogether**, patients given a higher dosage have higher recovery probability.*
*But separately, **male** and **female** patients face lower recovery probability with higher dosage*

🤔 *How can we be prevent misdiagnosing from a correlation?*

Simple Regression

Relationship between only two variables: X and Y
*One dependent variable regressed over **one independent variable***

Pizza example

Raw Data

Store	*Student Population (1000s)*	*Sales (millions NTD)*
1	2	58
2	6	105
3	8	88
4	8	118
5	12	117
6	16	137
7	20	157
8	20	169
9	22	149
10	26	202

Covariance

```
var(pizza)
          students   sales
students    63.11  315.56
sales      315.56 1747.78
```

Correlation

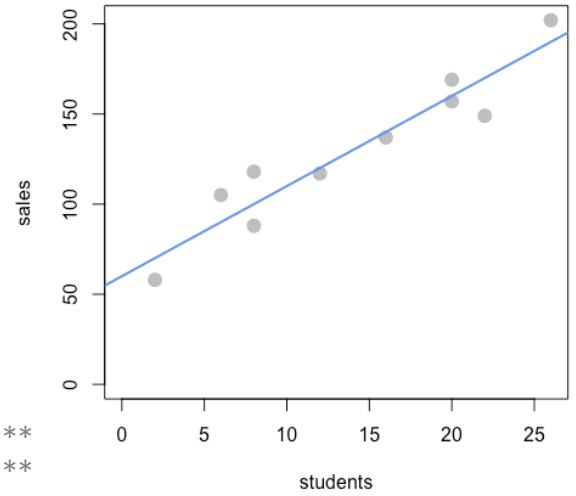
```
cor(pizza)
          students sales
students      1.00  0.95
sales         0.95  1.00
```

Regression

```
summary( lm(sales ~ students, data=pizza) )
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  60.0000     9.2260   6.503 0.000187 ***
students      5.0000     0.5803   8.617 2.55e-05 ***
```

***How would you describe** this regression line relative to the points?*

```
plot(pizza, ...)
abline(pizza_regr)
```



Standardized Simple Regression

All variables rescaled to
mean = 0, SD = 1

```
pizza_std <- data.frame( scale(pizza) )
```

scale() *mean-centers and standardizes vectors in data frame*

Standardized Data

```
      students sales
 [1,]    -1.51 -1.72
 [2,]    -1.01 -0.60
 [3,]    -0.76 -1.00
 [4,]    -0.76 -0.29
 [5,]    -0.25 -0.31
 [6,]     0.25  0.17
 [7,]     0.76  0.65
 [8,]     0.76  0.93
 [9,]     1.01  0.45
[10,]     1.51  1.72
```

Covariance

```
var(pizza_std)
          students sales
students      1.00  0.95
sales         0.95  1.00
```

Correlation

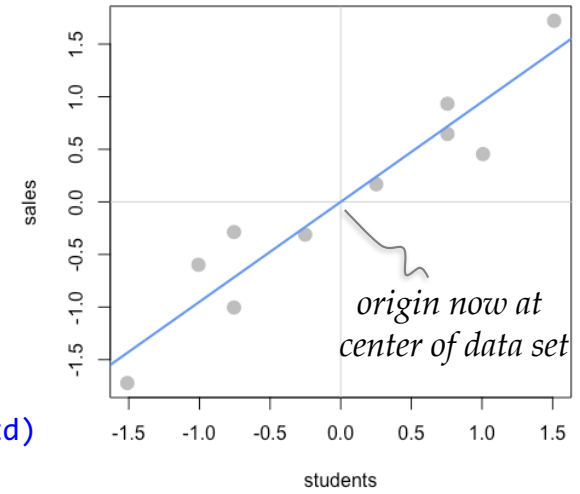
```
cor(pizza_std)
          students sales
students      1.00  0.95
sales         0.95  1.00
```

$$r_{xy} = \frac{cov_{xy}}{s_x s_y}$$

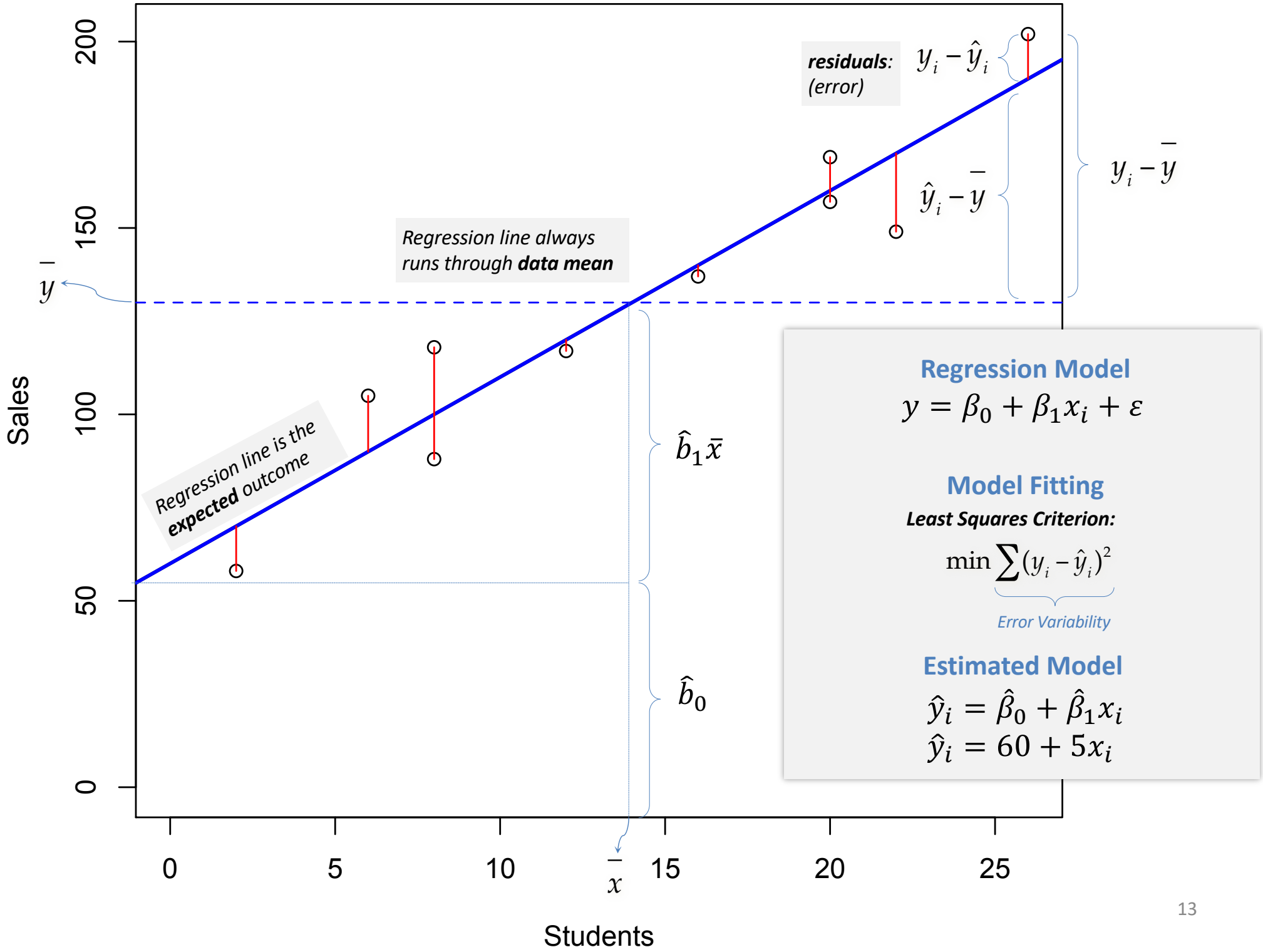
Regression

```
pizza_regr_std <- lm(sales ~ students, data=pizza_std)
summary(pizz_regr_std)
 Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 2.468e-16  1.046e-01   0.000        1
students    9.501e-01  1.103e-01   8.617 2.55e-05 ***
```

```
plot(pizza_std)
abline(pizza_regr_std)
```



Standardized simple regression *coefficient, and* ***standardized covariance*** *are both the same as* ***correlation****!*



Regression Model

$$y = \beta_0 + \beta_1 x_i + \varepsilon$$

Model Fitting

Least Squares Criterion:

$$\min \underbrace{\sum (y_i - \hat{y}_i)^2}_{\text{Error Variability}}$$

Estimated Model

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$$

$$\hat{y}_i = 60 + 5x_i$$

Fitting a Simple Regression

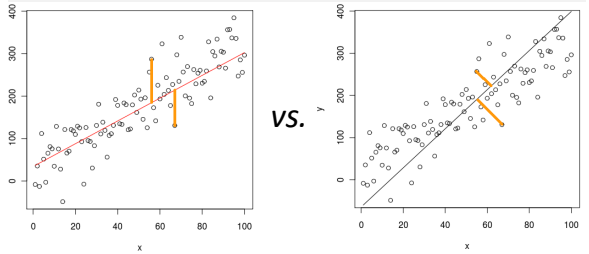
$$y = \beta_0 + \beta_1 x_i + \varepsilon$$

Least Squares Criterion: $\min \sum (y_i - \hat{y}_i)^2$

SSE: Sum of Squares due to Error (Error Variability)

β: *beta refers to **population** regression coefficient*

Why are we minimizing vertical error?
Why not make it the "best fitting" line?



$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$$

$$\hat{\boldsymbol{\beta}}_1 = \frac{\boldsymbol{cov_{xy}}}{\boldsymbol{s_x^2}} = \frac{SS_{xy}/df}{SS_x/df} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

Slope coefficient: *how much of the covariance of sales and students, is explained by the variance of students*

Intercept: *how much sales would we have, if there were no students*

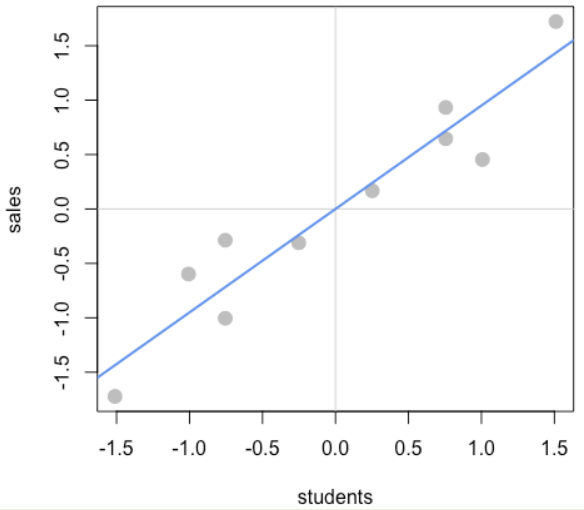
β̂ : *"beta hat" is our **estimate** of the population coefficient*

Raw data regression

```
b1 <- cov(pizza$Students, pizza$Sales) / var(pizza$Students)
  [1] 5

b0 <- mean(pizza$Sales) - b1*mean(pizza$Students)
  [1] 60
```

Standardized data regression

```
pizza_std <- data.frame(scale(pizza))

b1_std <- cov(pizza_std$students, pizza_std$sales) / var(pizza_std$students)
# [1] 0.95

b0_std <- mean(pizza_std$sales) - b1_std*mean(pizza_std$students)
# [1] 0
```



For standardized, simple regression:
*The **regression coefficient** is the **correlation***
*The **intercept** is **zero***

Fitted Values: $\hat{y}$

*The values of y **expected** by the estimated regression model*

Computing fitted values

```
pizza_regr <- lm(sales ~ students, data=pizza)

b0 + b1 * pizza$students
[1]  70  90 100 100 120 140 160 160 170 190

fitted(pizza_regr)
pizza_regr$fitted.values
  1   2   3   4   5   6   7   8   9  10
 70  90 100 100 120 140 160 160 170 190
```

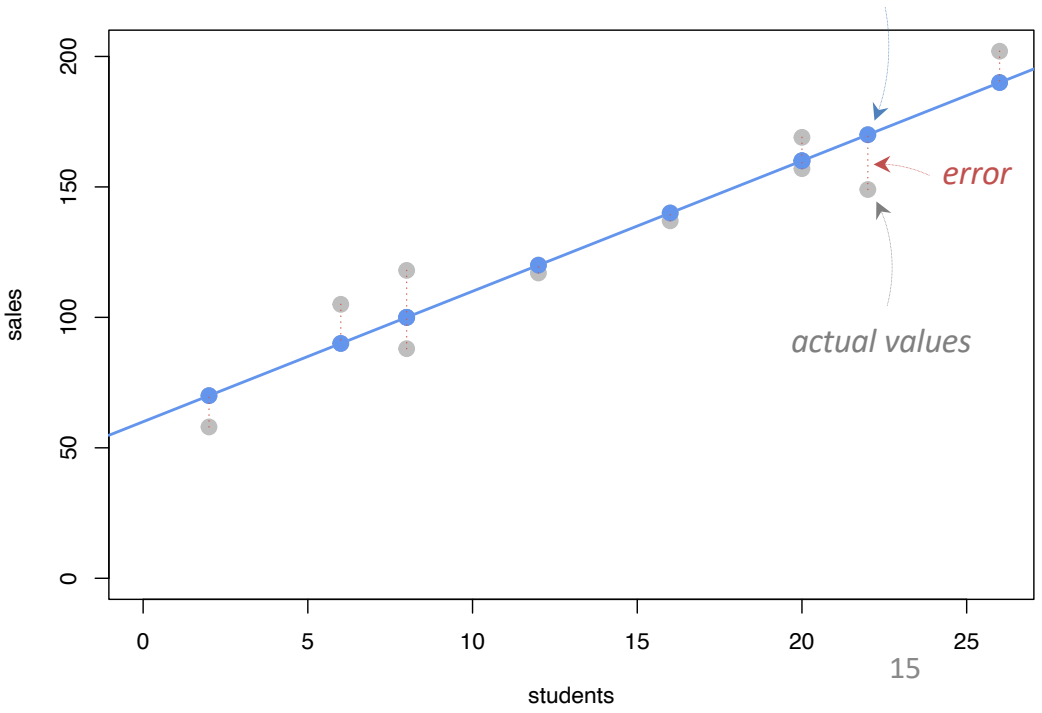
These are the values of sales (y) the regression model expects ($\hat{y}$)

```
data.frame(pizza, y_hat = pizza_regr$fitted.values)
```

	students	sales	y_hat
1	2	58	70
2	6	105	90
3	8	88	100
4	8	118	100
5	12	117	120
6	16	137	140
7	20	157	160
8	20	169	160
9	22	149	170
10	26	202	190

Visualizing fitted values

```
plot(pizza, …)

abline(pizza_regr, lwd=2, col="cornflowerblue")

points(pizza$students, pizza_regr$fitted.values, …)

segments(pizza$students, pizza$sales,
         pizza$students, pizza_regr$fitted.values, …)
```



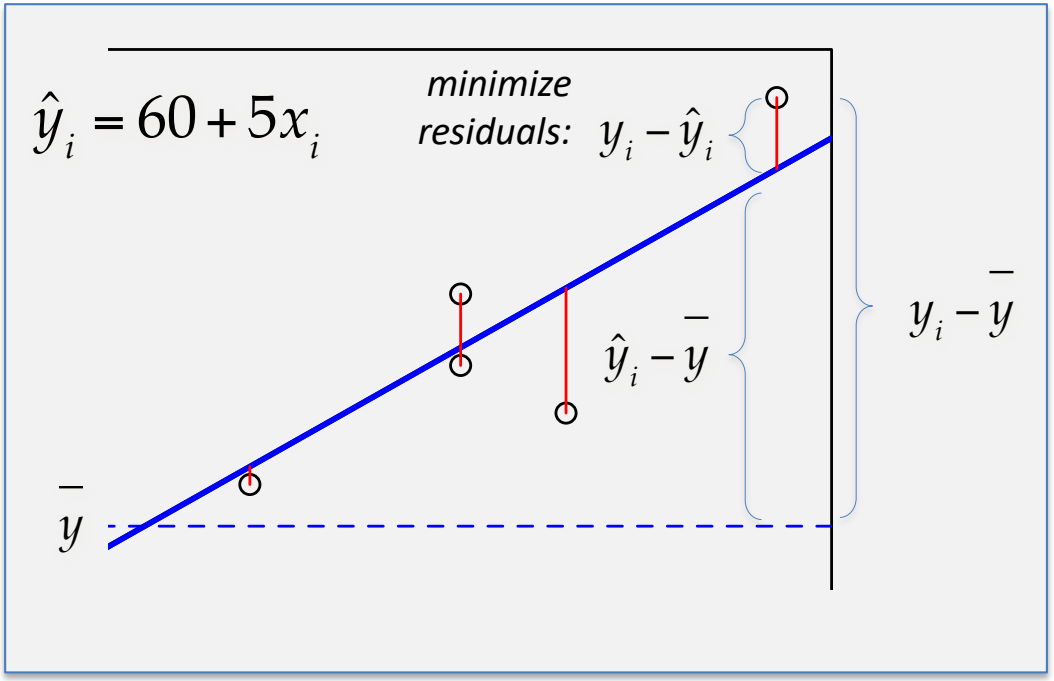
Simple Regression: **Model Fit**

How well does the model match the data?

$$SSR = \sum(\hat{y}_i - \bar{y}_i)^2$$ **Regression** Variability

$$+ \quad SSE = \sum(y_i - \hat{y}_i)^2$$ **Error** Variability

$$SST = \sum(y_i - \bar{y}_i)^2$$ **Total** Variability



Coefficient of determination
(Variance explained)

$$r^2 = \frac{SSR}{SST}$$

Notice: $r^2 = r_{xy}^2$ where $r_{xy} = \frac{\text{cov}_{xy}}{s_x s_y} = \frac{SS_{xy}}{\sqrt{SS_x SS_y}}$

Model fit can be seen as correlation between *independent variables and the dependent variable*

Multiple Regression

Multiple Regression Model:

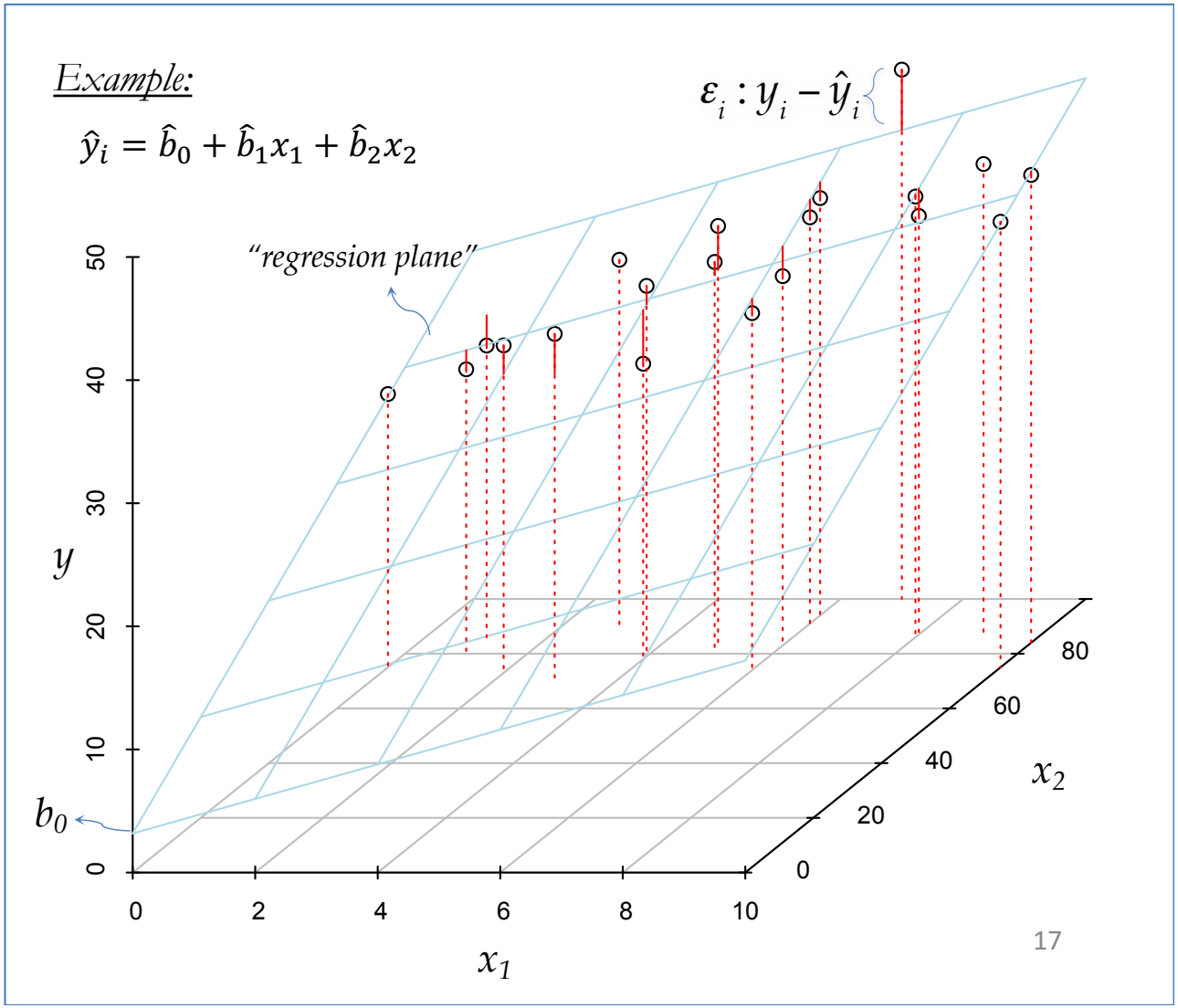
$$y = \beta_0 + \underbrace{\beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p}_{\text{multiple independent variables}} + \varepsilon$$

Estimated Multiple Regression:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \cdots + \hat{\beta}_p x_p$$

Least Squares Criterion:

$$\min \sum (y_i - \hat{y}_i)^2$$



Example: Software Developer Salaries

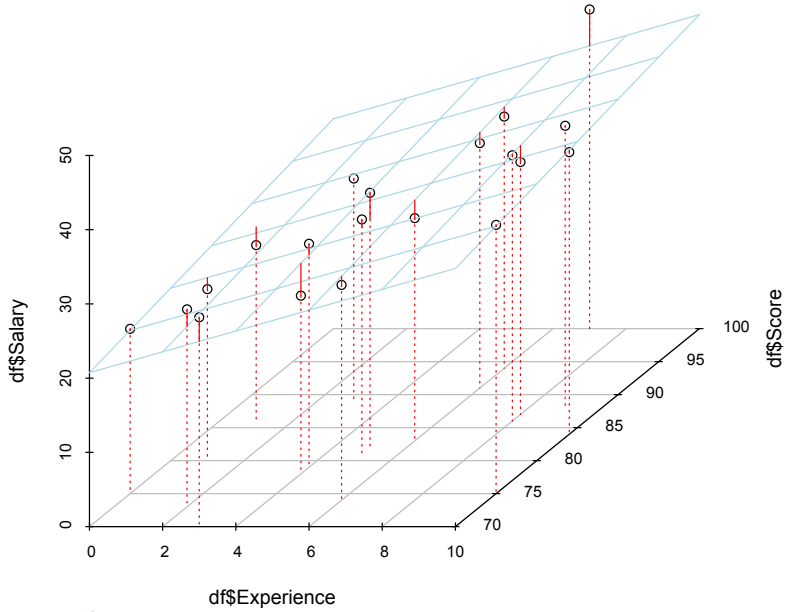
```
prog <- read.table("programmer_salaries.txt",
                   header=TRUE)
```

Independent Variables *Dependent Variable*

Experience	Score	Salary
4	78	24
7	100	43
1	86	23.7
5	82	34.3
8	86	35.8
10	84	38
0	75	22.2
1	80	23.1
6	83	30
6	91	33
9	88	38
2	73	26.6
10	75	36.2
5	81	31.6
6	74	29
8	87	34
4	79	30.1
6	94	33.9
3	70	28.2
3	89	30

Standardized coefficients are between -1 to 1

What are the (dis)advantages of standardizing?



Raw data regression

```
prog_regr <- lm(Salary ~ Experience + Score, data=prog)
summary(prog_regr)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.17394    6.15607   0.516  0.61279
Experience   1.40390    0.19857   7.070 1.88e-06 ***
Score        0.25089    0.07735   3.243  0.00478 **

Multiple R-squared: 0.8342, Adjusted R-squared: 0.8147
F-statistic: 42.76 on 2 and 17 DF,  p-value: 2.328e-07
```

Standardized regression

```
prog_std <- data.frame(scale(prog))
prog_regr_std <- lm(Salary ~ Experience + Score, data=prog_std)
summary(prog_regr_std)
```

```
Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       6.365e-17  9.626e-02   0.000  1.00000
scale(Experience) 7.412e-01  1.048e-01   7.070 1.88e-06 ***
scale(Score)      3.400e-01  1.048e-01   3.243  0.00478 **

Multiple R-squared: 0.8342, Adjusted R-squared: 0.8147
F-statistic: 42.76 on 2 and 17 DF,  p-value: 2.328e-07
```

Dummy Variables

Independent Variables *Dependent Variable*

Dummy Variable

Experience	Score	Degree	Salary
4	78	0	24
7	100	1	43
1	86	0	23.7
5	82	1	34.3
8	86	1	35.8
10	84	1	38
0	75	0	22.2
1	80	0	23.1
6	83	0	30
6	91	1	33
9	88	1	38
2	73	0	26.6
10	75	1	36.2
5	81	0	31.6
6	74	0	29
8	87	1	34
4	79	0	30.1
6	94	1	33.9
3	70	0	28.2
3	89	0	30

Dummy variable:
binary: absence or presence of a criteria

```
prog_regr_full <- lm(Salary ~ Experience + Score + Degree, prog)
summary(prog_regr_full)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   7.9448     7.3808   1.076   0.2977
Experience    1.1476     0.2976   3.856   0.0014 **
Score         0.1969     0.0899   2.191   0.0436 *
Degree        2.2804     1.9866   1.148   0.2679

Residual standard error: 2.396 on 16 degrees of freedom
Multiple R-squared: 0.8468, Adjusted R-squared: 0.8181
F-statistic: 29.48 on 3 and 16 DF,  p-value: 9.417e-07
```

Model Fit of Multiple Regression

Source of Variation		Sum of Squares (SS)
Regression		$SSR = \sum(\hat{y}_i - \bar{y}_i)^2$
Error	+	$SSE = \sum(y_i - \hat{y}_i)^2$
Total	=	$SST = \sum(y_i - \bar{y}_i)^2$

Multiple Coefficient of Determination

$$R^2 = \frac{SSR}{SST}$$

$$R^2 = 1 - \frac{SSE}{SST}$$

*R is now a **multiple correlation** between many independent variables and a dependent variable*

Adjusted Multiple Coefficient of Determination

$$R_a^2 = \frac{MSR}{MST} = 1 - (1 - R^2)\frac{n-1}{n-p-1}$$

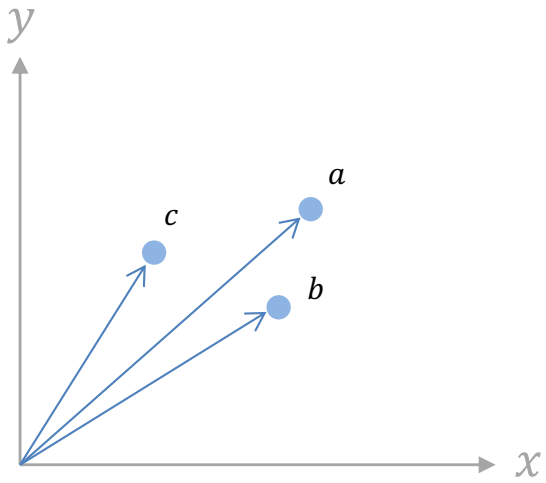
$$R_a^2 = 1 - \frac{SSE}{SST}\frac{df_T}{df_E}$$

Adjusted R^2 includes information about the *parsimony*, or simplicity/complexity of the model.

Geometric Perspective: Simple Regression

Column Space (2D)

	x	y
a	5.5	24
b	4.7	22
c	4.8	21

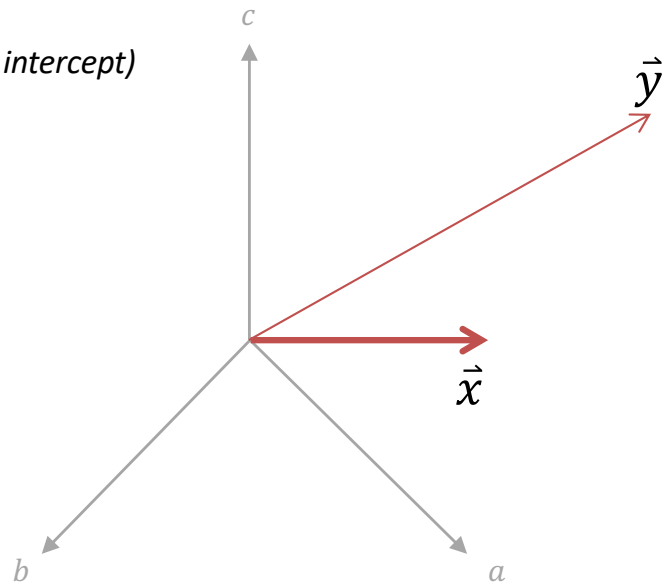


*This is the **statistical view** of regression, where **x and y are the axes** of the space while **rows are points** in this space*

Row Space (3D)

(simplified representation without intercept)

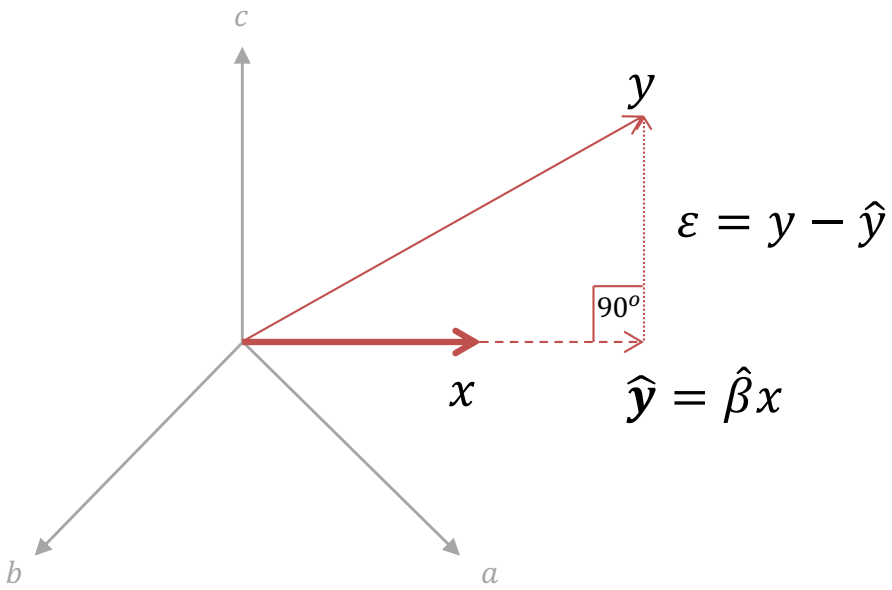
	x	y
a	5.5	24
b	4.7	22
c	4.8	21



*This is the **geometric view** of regression, where the **rows are the axes** of the space while **x and y are vectors** in this space*

Simple Regression as **Orthogonal Projection**

Greek: "upright" "angle"



Regression Model: $y = bx + \varepsilon$

ε captures the portion of $\vec{y}$ that is not captured by $b\vec{x}$ (residual)

$\hat{y}$ is a scaling of $\vec{x}$ such that it is the projection of $\vec{y}$ onto $\vec{x}$

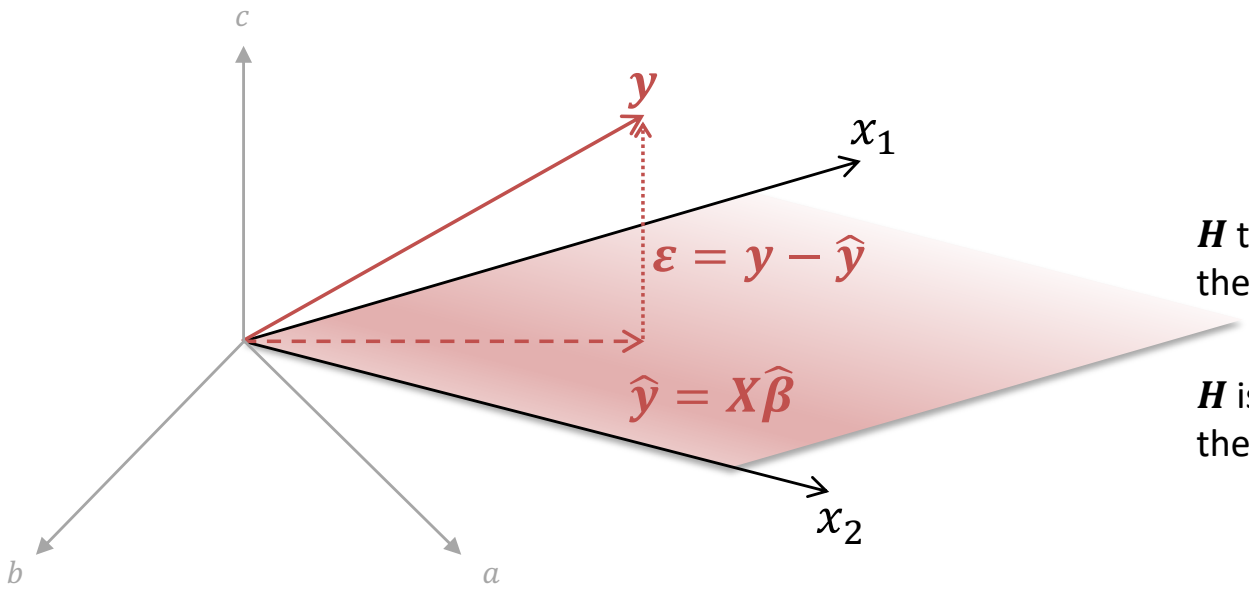
The orthogonal projection is the ***closest fitting distance*** *from the tip of $\vec{y}$ onto $\vec{x}$.*

Geometric Interpretation: Multiple Regression

(simplified representation without intercept)

$$y = (X \times \hat{\beta}) + \varepsilon$$

$$\begin{pmatrix} 24 \\ 22 \\ 21 \\ \vdots \end{pmatrix} = \begin{pmatrix} x_1 & x_2 \\ 5.5 & 11 \\ 4.7 & 9 \\ 4.8 & 10 \\ \vdots & \vdots \end{pmatrix} \times \begin{pmatrix} \widehat{\beta_1} \\ \widehat{\beta_1} \end{pmatrix} \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \end{pmatrix}$$



transpose *inverse*

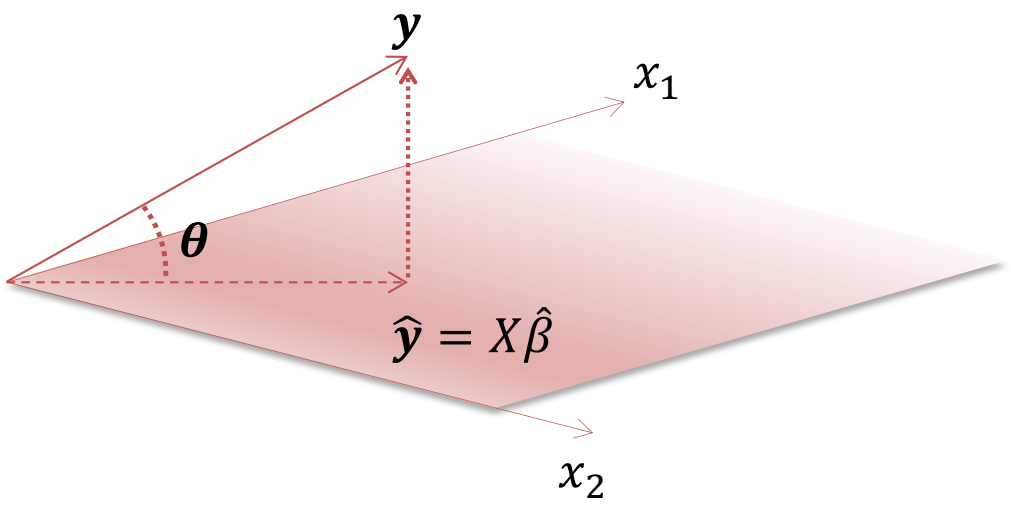
$$X\hat{\beta} = \underbrace{X(X^T X)^{-1} X^T}_{H} y$$

$$\hat{y} = Hy$$

$\boldsymbol{H}$ transforms vectors by *orthogonally projecting* them onto the space *spanned* by $\boldsymbol{X}$

$\boldsymbol{H}$ is known as the "**Hat Matrix**" because it puts the "hat" on y to make it $\hat{y}$ (y-hat)

Geometric Interpretation of R^2



If all vectors are mean-centered, then the angle θ between y and $\hat{y}$ tells us *how similar y is to $\hat{y}$*

*one possible metric of **model fit***

Coefficient of Multiple Correlation

$$R = cos(\theta)$$

*cosine of angle between **mean centered** vectors*

```
cosV <- function(a, b) sum(a*b) /
        (sqrt(sum(a^2)) * sqrt(sum(b^2)))

cosV(pts$y - mean(pts$y), y_hat - mean(y_hat))
# [1] 0.848422
```

$$R = cor(y, \hat{y})$$

```
cor(pts$y, y_hat)
# [1] 0.848422
```

R is coefficient of **multiple correlation** between dependent y and ALL independent X

Squared Multiple Correlation

$$R^2 = cos(\theta)^2$$

*cosine of angle between **mean centered** vectors*

```
cosV(pts$y - mean(pts$y), y_hat - mean(y_hat))^2
# [1] 0.7198199
```

$$R^2 = cor(y, \hat{y})^2$$

```
cor(pts$y, y_hat)^2
# [1] 0.7198199
```

R^2 is the **coefficient of determination**

Linear Algebra: Solving Regression Geometrically

Put constant values of 1 in first column to capture the intercept

$$X = \begin{bmatrix} 1 & 4 & 78 & 0 \\ 1 & 7 & 100 & 1 \\ 1 & 1 & 86 & 0 \\ . & . & . & . \\ . & . & . & . \end{bmatrix} \qquad y = \begin{bmatrix} 24.0 \\ 43.0 \\ 23.7 \\ . \\ . \end{bmatrix}$$

Intercept term

$$\hat{\beta} = \begin{bmatrix} \widehat{\beta_0} \\ \widehat{\beta_1} \\ \widehat{\beta_2} \\ . \\ . \end{bmatrix}$$

transpose *inverse*

$$\hat{\beta} = \left(\boldsymbol{X^T X}\right)^{-1} \boldsymbol{X^T} y$$

$$\hat{y} = X\hat{\beta}$$

$$\varepsilon = y - \hat{y}$$

$$R^2 = cor(y, \hat{y})^2$$

Common Linear Algebra Operations in R

Matrix-matrix or Matrix-vector Multiplication:

```
A %*% B        A %*% b
```

Matrix Transpose:

```
t(A)
```

Matrix Inverse:

```
solve(A)
```