Lecture 4 – Linear regression

ME494 – Data Science and Machine Learning for Mech Engg

Instructor – Subramanian Sankaranarayanan

Teaching Assistants

Aditya Koneru (akoner3@uic.edu)

Suvo Banik (sbanik2@uic.edu)

Machine Learning Pipeline

Example dataset

Material	Property Value
Material 1	P_1
Material 2	P_2
⋮	⋮
Material N	P_N

Fingerprinting (descriptors) and learning

Material		Fingerprint		Property Value
Material 1	→	$x_{11}, x_{12}, \ldots x_{1M}$	→	P_1
Material 2	→	$x_{21}, x_{22}, \ldots x_{2M}$	→	P_2
⋮		⋮		⋮
Material N	→	$x_{N1}, x_{N2}, \ldots x_{NM}$	→	P_N
	Fingerprinting		Learning	

ML pipeline in practice

1. Collect Data → 2. Fingerprint Materials → 3. Train Model → 4. Make Prediction

Let us look at Step 3

An ML model:

- maps materials fingerprint to target property (more popular)
- can accomplish other mappings in case of unsupervised and reinforcement learning (more advanced)

Many possible algorithms:

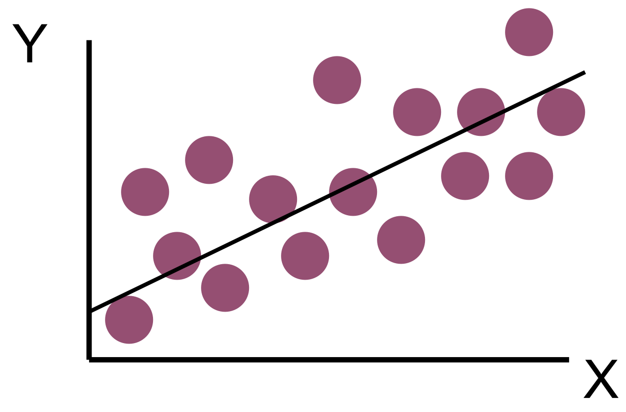
- linear regression
- polynomial regression
- Gaussian process regression
- random forest
- deep neural networks
- gradient boosting
- genetic algorithm
- many more…

Overview

- In this lecture, we will provide a brief overview of regression and specifically linear regression.
- Most of you probably have seen linear models in some form, but we will start from scratch to further illustrate key concepts such as bias and variance.
- Various ways of assessing the quality of the dataset
- Python implantation of the regression models

What is regression?

- In correlation, the two variables are treated as equals. In regression, one variable is considered independent (=predictor) variable (*X*) and the other the dependent (=outcome) variable *Y*.

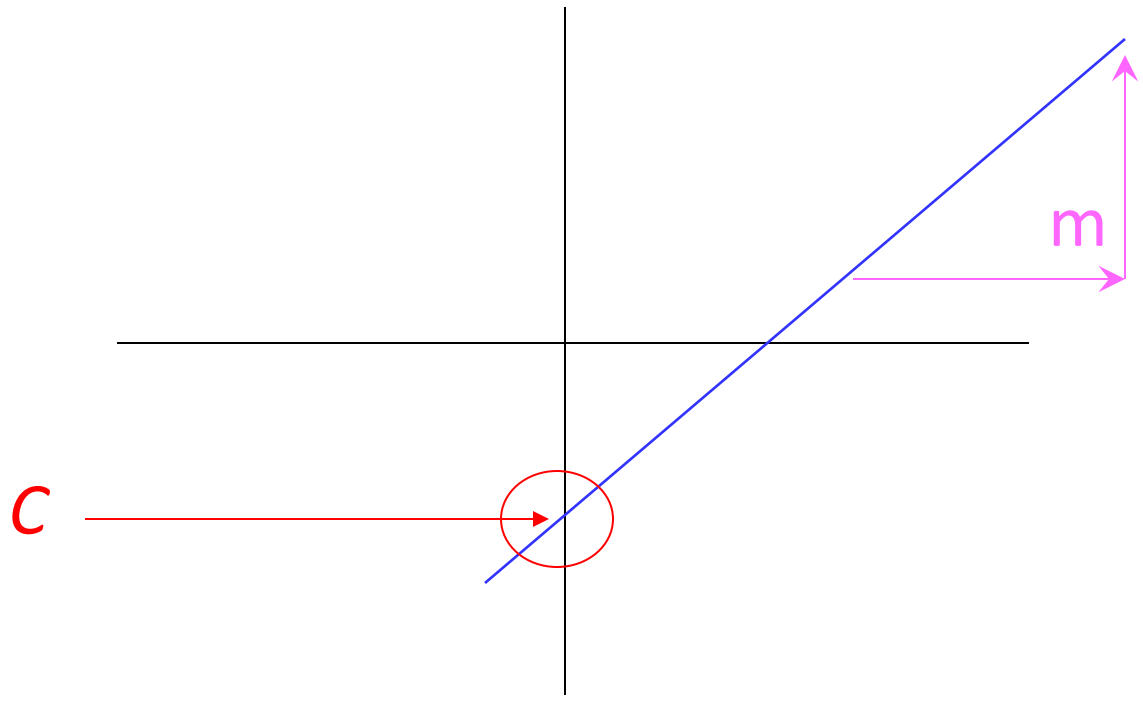


$$Y=f(x)$$

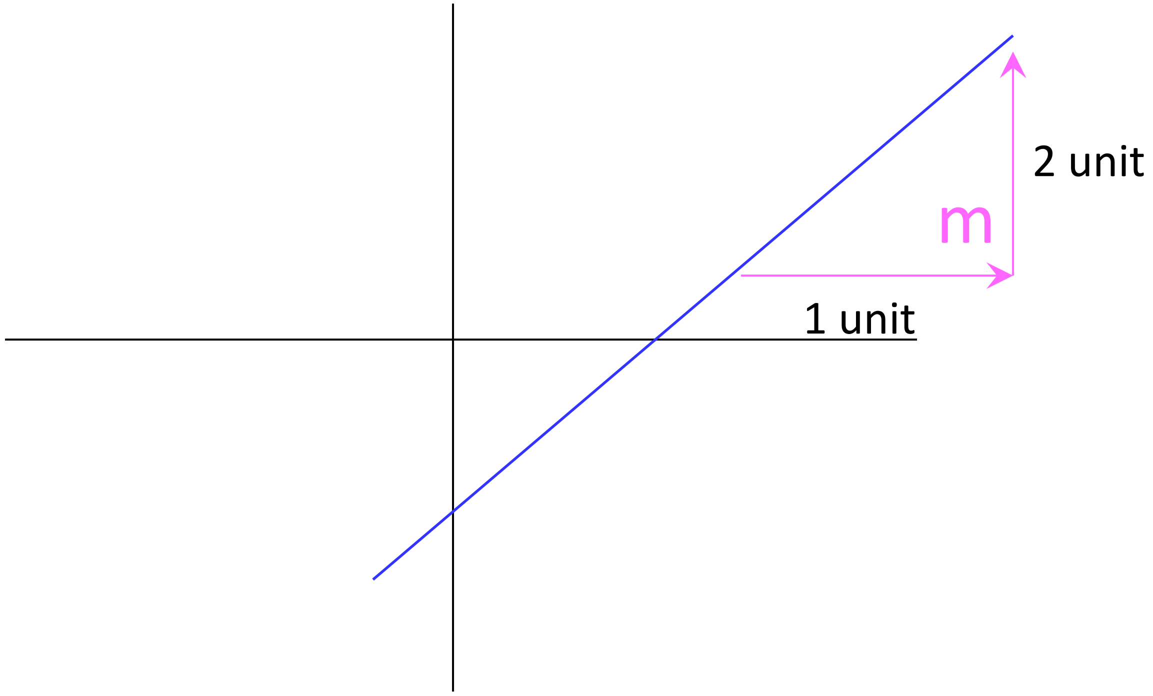
- Simplest possible model between target and feature

What is “Linear”?

- Remember this:
- *Y=mX+C?*



What’s Slope?



A slope of 2 means that every 1-unit change in X yields a 2-unit change in Y.

Prediction

If you know something about X, this knowledge helps you predict something about Y. (Sound familiar?…sound like conditional probabilities?)

Regression equation…

Expected or (predicted) value of y at a given level of *x*=

$$E(y_i / x_i) = \alpha + \beta x_i$$

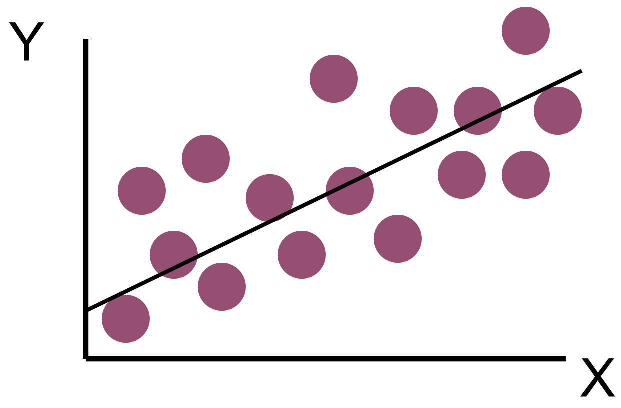
Predicted value for an individual…

$$y_i = \underbrace{\alpha + \beta * x_i}_{\text{Fixed – exactly on the line}} + \boxed{\text{Random Error}_i}$$

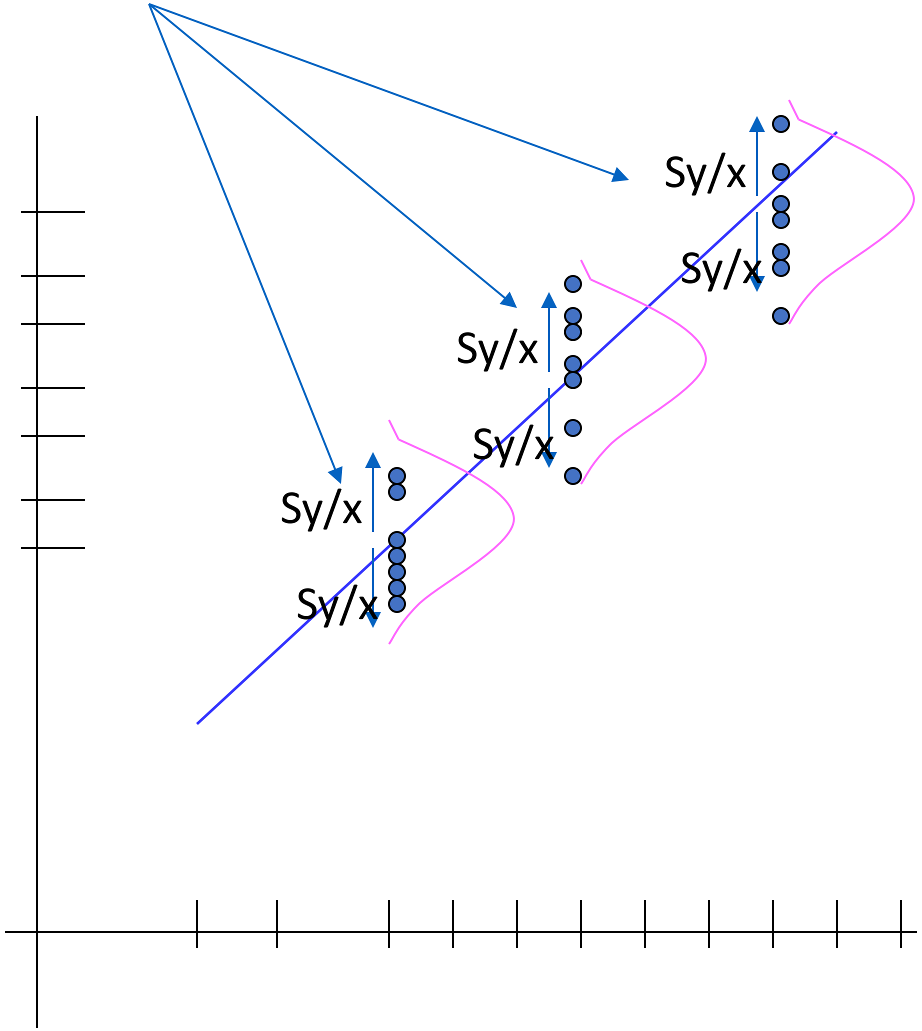
Follows a normal distribution

Assumptions

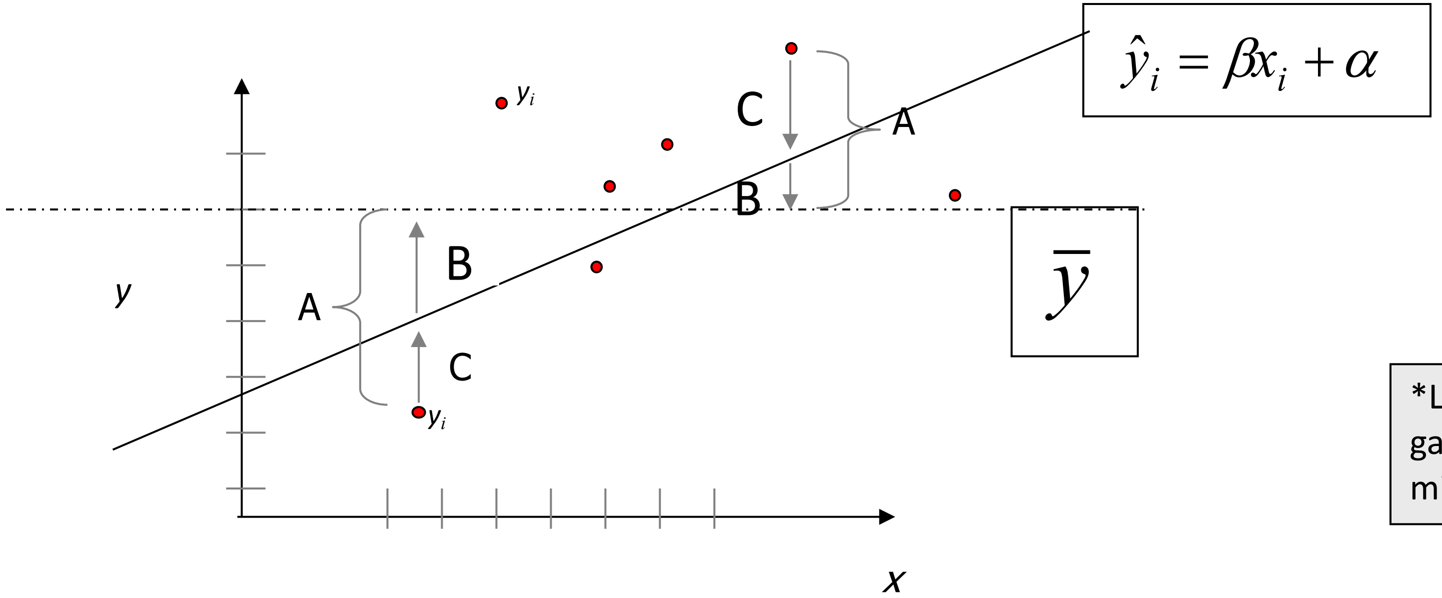
- Linear regression assumes that…
  - The relationship between X and Y is linear
  - Y is distributed normally at each value of X
  - The variance of Y at every value of X is the same (homogeneity of variances)
  - The observations are independent



The standard error of Y given X is the average variability around the regression line at any given value of X. It is assumed to be equal at all values of X.



Overview of Linear Regression



*Least squares estimation gave us the line (β) that minimized C^2

$$\sum_{i=1}^{n}(y_i-\bar{y})^2=\sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2+\sum_{i=1}^{n}(\hat{y}_i-y_i)^2$$

$R^2 = SS_{reg}/SS_{total}$

A^2

SS_{total}

Total squared distance of observations from naïve mean of y

Total variation

B^2

SS_{reg}

Distance from regression line to naïve mean of y

Variability due to x (regression)

C^2

$SS_{residual}$

Variance around the regression line

Additional variability not explained by x—what least squares method aims to minimize

Let us generalize this concept

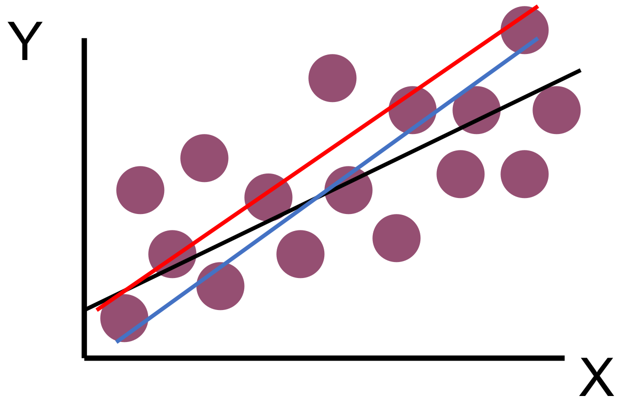
$$Y = f(X_1, X_2, ..., X_p) = \beta_0 + \sum_{j=1}^{p} \beta_j X_j$$

X_j can be:

- Quantitative inputs
- Transformations of quantitative inputs, e.g., log, exp, powers, etc. Basis expansions, e.g., $X_2 = X_1^2$, $X_3 = X_1^3$
- Interactions between variables, e.g., $X_1 X_2$
- Encoding of levels of inputs

Supervised Learning

- Given a set of paired observations $\{x_{ij}, y_i\}$, what are the model parameters (in this case, the coefficients β_j) that are "optimal"?
- "Optimal" is typically defined as minimization of some **loss function** (also known as **cost function**) that measures the error of the model.



How do you now pick which line fits the best?

Now some basic math .....

Consider the simple case of

$$Y = \beta_0 + \beta_1 X_1$$

In least squares regression, the loss function is defined as the sum squared error given the N observations:

$$\begin{aligned} L(Y, \hat{f}(X)) &= \sum_{i=1}^{N} (y_i - f(x_i))^2 \\ &= \sum_{i=1}^{N} (y_i - \beta_0 - \beta_1 x_{i1})^2 \end{aligned}$$

Remember that loss function is the difference between true observation and predicted outcomes

What are the optimal parameters β_0 and β_1?

$$
\begin{aligned}
\frac{\partial L}{\partial \beta_0} &= \sum_{i=1}^{N} 2(y_i - \beta_0 - \beta_1 x_{i1})(-1) = 0 \\
\implies & \sum_{i=1}^{N} y_i = N\beta_0 + \sum_{i=1}^{N} \beta_1 x_{i1} \\
\implies & \beta_0 = \bar{y} - \beta_1 \bar{x_1} \\
\frac{\partial L}{\partial \beta_1} &= \sum_{i=1}^{N} 2(y_i - \beta_0 - \beta_1 x_{i1})(-x_{i1}) = 0 \\
\implies & \beta_1 = \frac{\sum_{i=1}^{N} x_{i1} y_i - N\bar{x_1}\bar{y}}{\sum_{i=1}^{N} x_{i1}^2 - N\bar{x_1}^2}
\end{aligned}
$$

What are the optimal parameters β_0 and β_1?

$$\hat{\beta}_1 = \frac{n\sum_{i=1}^{n} x_i y_i - n^2 \bar{x}\bar{y}}{n\sum_{i=1}^{n} x_i^2 - n^2 \bar{x}^2} = \frac{\frac{\sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y}}{n-1}}{\frac{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2}{n-1}} = \frac{cov(x,y)}{var(x)},$$

where var and cov represent their sample counterparts.

What if you have 2 dependent variables?

Graphic representation of Multiple Linear Regression with two dependent variables

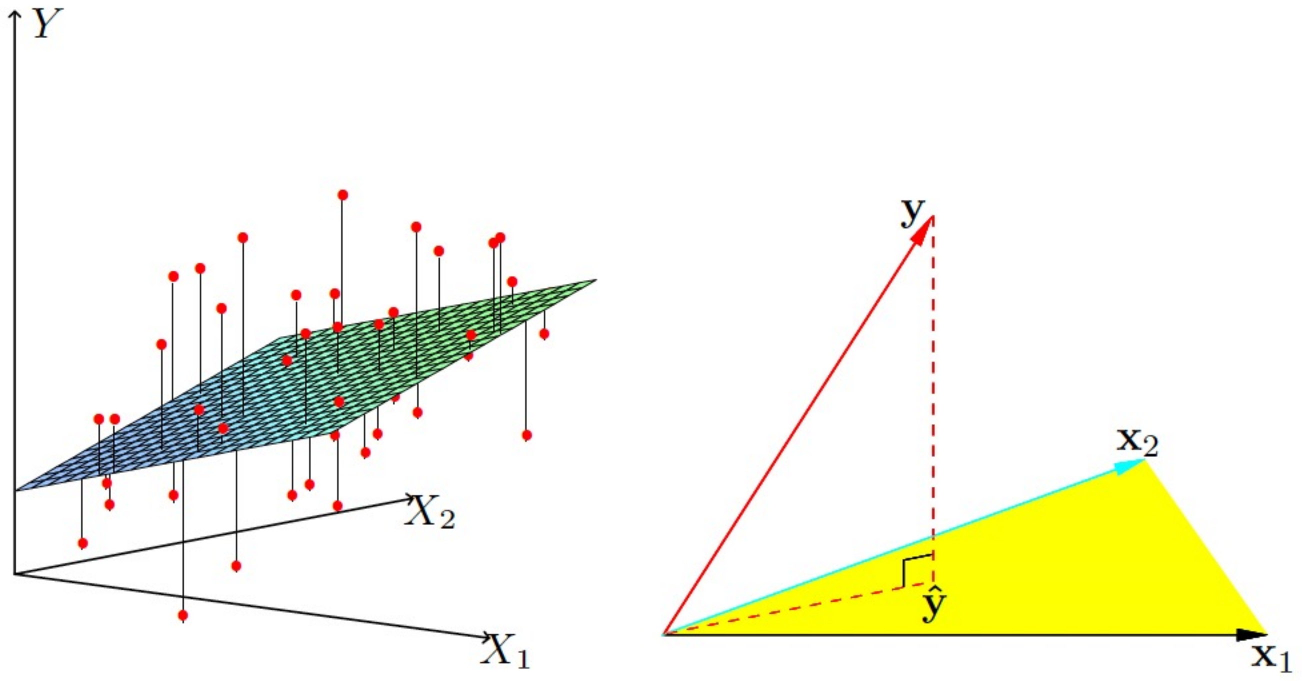


Figure: MLR minimizes sum square of residuals. The projection $\hat{\mathbf{y}}$ represents the vector of the least squares predictions onto the hyperplane spanned by the input vectors $\mathbf{x_1}$ and $\mathbf{x_2}$.

What if you have M dependent variables?

M-D Linear Regression model

Example M = 6

	ID	HV	C.al	C.co	C.cr	C.cu	C.fe	C.ni
0	25	170	0.056604	0.000000	0.188679	0.188679	0.188679	0.377358
1	60	380	0.200000	0.266667	0.000000	0.000000	0.266667	0.266667
2	155	775	0.400000	0.200000	0.200000	0.000000	0.200000	0.000000
3	88	486	0.208333	0.000000	0.208333	0.208333	0.208333	0.166667
4	2	118	0.024390	0.243902	0.243902	0.000000	0.243902	0.243902
...	...	...	...	...	...	...	...	...
115	59	371	0.166667	0.000000	0.555556	0.000000	0.000000	0.277778
116	104	537	0.166667	0.166667	0.166667	0.083333	0.250000	0.166667
117	116	558	0.264706	0.147059	0.147059	0.147059	0.147059	0.147059
118	154	768	0.400000	0.133333	0.066667	0.133333	0.200000	0.066667
119	123	584	0.222222	0.222222	0.000000	0.111111	0.222222	0.222222

120 rows × 8 columns

Materials Data for hardness of high entropy alloys

Difficult to visualize data. Needs a 7-dim plot

We assume that a linear model can explain the data

$$\hat{y} = f(\boldsymbol{x}) = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3 \dots + w_M x_M$$

$$= \sum_{i=0}^{M} w_i x_i$$

$$\boldsymbol{x} = (x_1, x_2, x_3, \dots, x_M)$$

Solution that minimizes the training error

$$X^T X \widehat{\boldsymbol{w}} = X^T \boldsymbol{y} \qquad \boldsymbol{x}_*^T \widehat{\boldsymbol{w}} = \boldsymbol{x}_*^T (X^T X)^{-1} X^T \boldsymbol{y}$$

HOW? We need to learn Linear Algebra!

Reformulating the general multiple linear regression as a vector equation...

Considering N observations of

$$y_i = \beta_0 + \beta_1 x_{i1} + +\beta_2 x_{i2} + \ldots + \beta_p x_{ip}$$

Intercept Coefficients

Let

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \ldots \\ y_n \end{pmatrix}, \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \ldots \\ \beta_p \end{pmatrix}, \mathbf{X} = \begin{pmatrix} 1 & x_{11} & x_{12} & \ldots & x_{1p} \\ 1 & x_{21} & x_{22} & \ldots & x_{2p} \\ \vdots & & & & \\ 1 & x_{N1} & x_{N2} & \ldots & x_{Np} \end{pmatrix},$$

So,

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta}$$

Note that $\mathbf{y}$ is a $N \times 1$ vector, $\boldsymbol{\beta}$ is a $(p+1) \times 1$ vector, and $\mathbf{X}$ is a $N \times (p+1)$ matrix.

Reformulating the general multiple linear regression as a vector equation...

$$L = RSS = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta)$$

Assuming (for the moment) that $\mathbf{X}$ has full column rank, and hence $\mathbf{X}^T\mathbf{X}$ is positive definite, It can be shown using the same principles that the following unique solution for β is:

Coefficients $$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

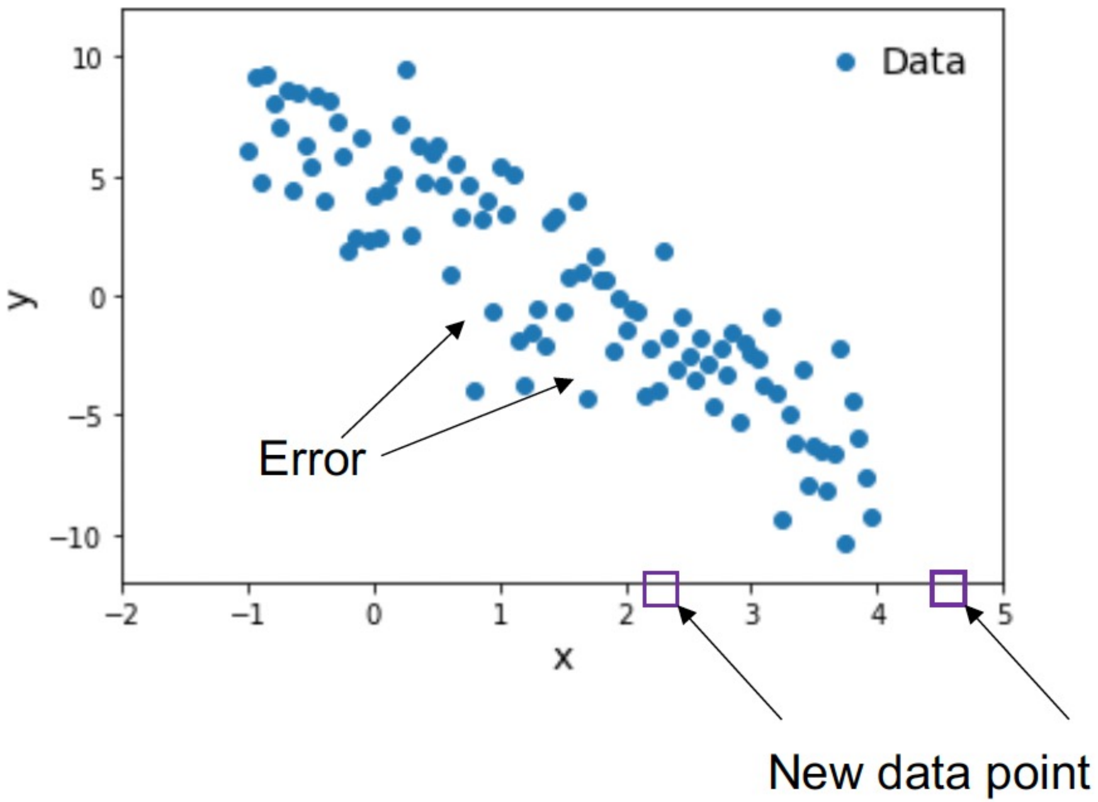
Predictions $$\hat{\mathbf{y}} = \mathbf{X}\hat{\beta} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

What is the first coefficient $\hat{\beta}$?

$$\hat{\beta}_1 = \frac{cov(x, y)}{var(x)},$$

where var and cov represent their sample counterparts.

Error Metrics



Error $e_i = y_i - \hat{y}_i$

Mean square error $\frac{1}{N}\sum_{i=1}^{N}(e_i)^2$

Coefficient of determination (R^2) $1 - \frac{\sum_{i=1}^{N}(e_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$

Mean of $\boldsymbol{y}$

Our aim is to minimize the mean square error

How do we evaluate the quality of the regression model?

Covariance vs Variance

Covariance is a statistical measure of how two variables change together. It measures the direction of the relationship between two variables. Covariance is similar to variance, but while variance measures how a single variable varies, covariance measures how two variables vary together.

Recall: Covariance

$$\operatorname{cov}(x, y) = \frac{\sum_{i=1}^{n}(x_i - \bar{X})(y_i - \bar{Y})}{n-1}$$

Cov(x, y) = covariance between x and y
xi = value of x
yi = value of y

$\bar{X}$ = mean of x
$\bar{Y}$ = mean of y
n = total number of values

Interpreting Covariance

$\text{cov}(X,Y) > 0 \rightarrow$ X and Y are positively correlated

$\text{cov}(X,Y) < 0 \rightarrow$ X and Y are inversely correlated

$\text{cov}(X,Y) = 0 \rightarrow$ X and Y are independent

Python Implementation – calculate covariance

```python
import pandas as pd
from sklearn import datasets
#
# Load IRIS dataset
#
iris = datasets.load_iris()
#
# Create dataframe from IRIS dataset
#
df = pd.DataFrame(iris.data, columns=["sepal_length",
"sepal_width", "petal_length", "petal_width"])
df["class"] = iris.target
#
# Calculate covariance between different columns
#
df.iloc[:, 0:4].cov()
```

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	0.685694	-0.042434	1.274315	0.516271
sepal_width	-0.042434	0.189979	-0.329656	-0.121639
petal_length	1.274315	-0.329656	3.116278	1.295609
petal_width	0.516271	-0.121639	1.295609	0.581006

Variance

$$\sigma^2 = \frac{\sum_{i=1}^{N} (x_i - \bar{x})^2}{N - 1}$$

Variance measures how much the value of the random variable varies from its mean. A high variance indicates that the data points are spread out; a low variance indicates that they are close to the mean.

Note that variance is the average squared deviations from the mean, while standard deviation is the square root of this number

Python Implementation - Variance

```python
import pandas as pd
from sklearn import datasets
#
# Load IRIS dataset
#
iris = datasets.load_iris()
#
# Create dataframe from IRIS dataset
#
df = pd.DataFrame(iris.data, columns=["sepal_length",
"sepal_width", "petal_length", "petal_width"])
df["class"] = iris.target
#
# Calculate variance for different columns
#
df.iloc[:, 0:4].var()
```

```
sepal_length    0.685694
sepal_width     0.189979
petal_length    3.116278
petal_width     0.581006
dtype: float64
```

Correlation

$$\hat{r} = \frac{\operatorname{cov} ariance(x, y)}{\sqrt{\operatorname{var} x}\sqrt{\operatorname{var} y}} = \frac{\dfrac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{n-1}}{\sqrt{\dfrac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}}\sqrt{\dfrac{\sum_{i=1}^{n}(y_i - \bar{y})^2}{n-1}}}$$

Simpler calculation formula…

$$\hat{r} = \frac{\dfrac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\cancel{n-1}}}{\sqrt{\dfrac{\sum_{i=1}^{n}(x_i-\bar{x})^2}{\cancel{n-1}}}\sqrt{\dfrac{\sum_{i=1}^{n}(y_i-\bar{y})^2}{\cancel{n-1}}}} =$$

$$\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}} = \frac{SS_{xy}}{\sqrt{SS_x SS_y}}$$

$$\hat{r} = \frac{SS_{xy}}{\sqrt{SS_x SS_y}}$$

Numerator of covariance

Numerators of variance

Correlation coefficient

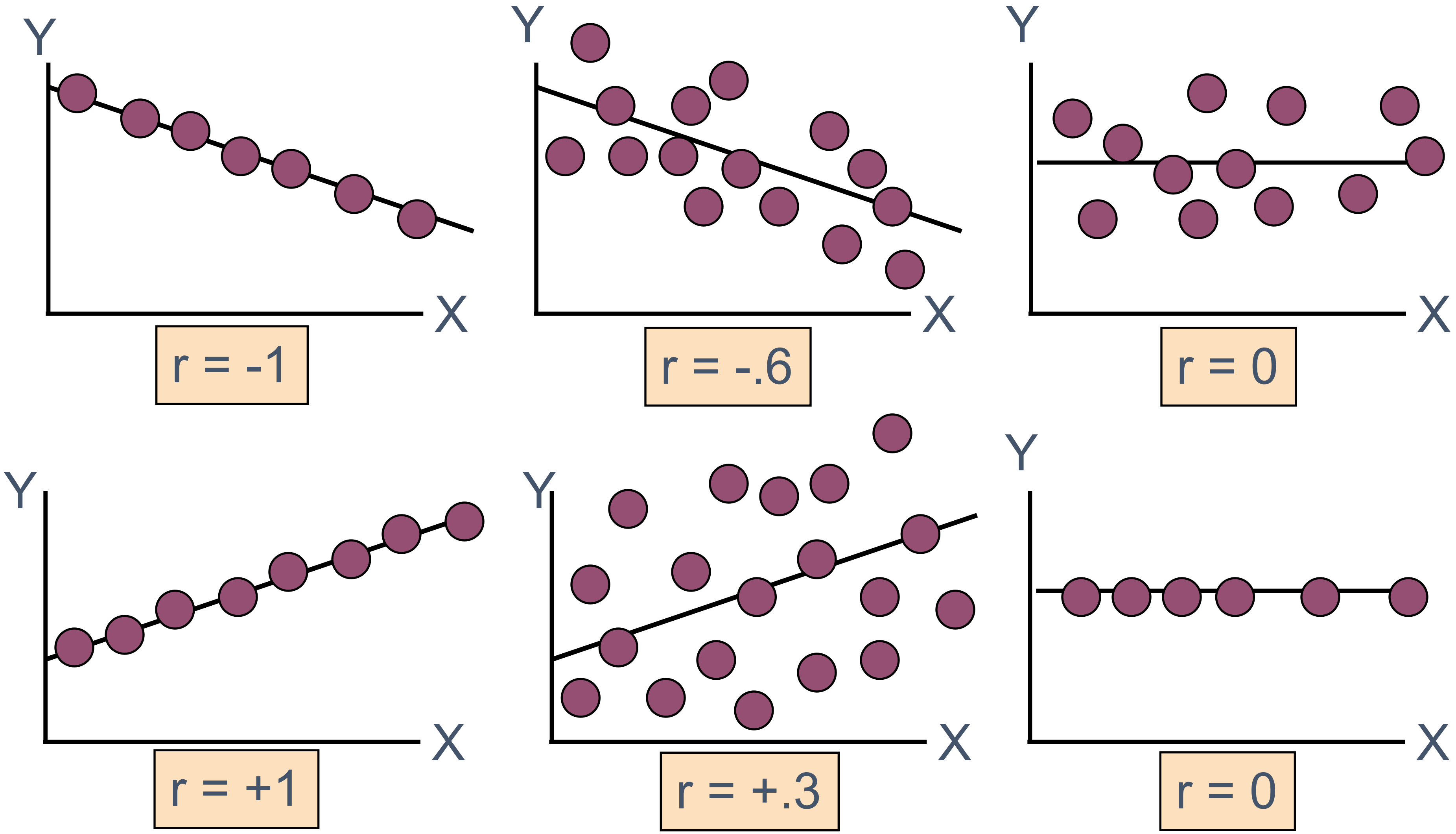
- Pearson's Correlation Coefficient is standardized covariance (unitless):

$$r = \frac{\operatorname{cov}ariance(x, y)}{\sqrt{\operatorname{var} x}\sqrt{\operatorname{var} y}}$$

Correlation

- Measures the relative strength of the *linear* relationship between two variables
- Unit-less
- Ranges between –1 and 1
- The closer to –1, the stronger the negative linear relationship
- The closer to 1, the stronger the positive linear relationship
- The closer to 0, the weaker any positive linear relationship

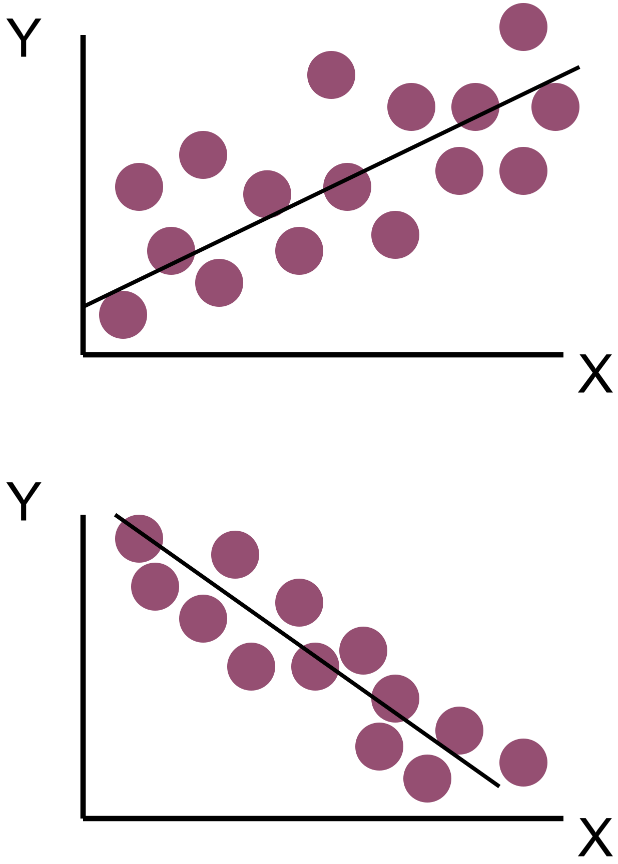
Scatter Plots of Data with Various Correlation Coefficients



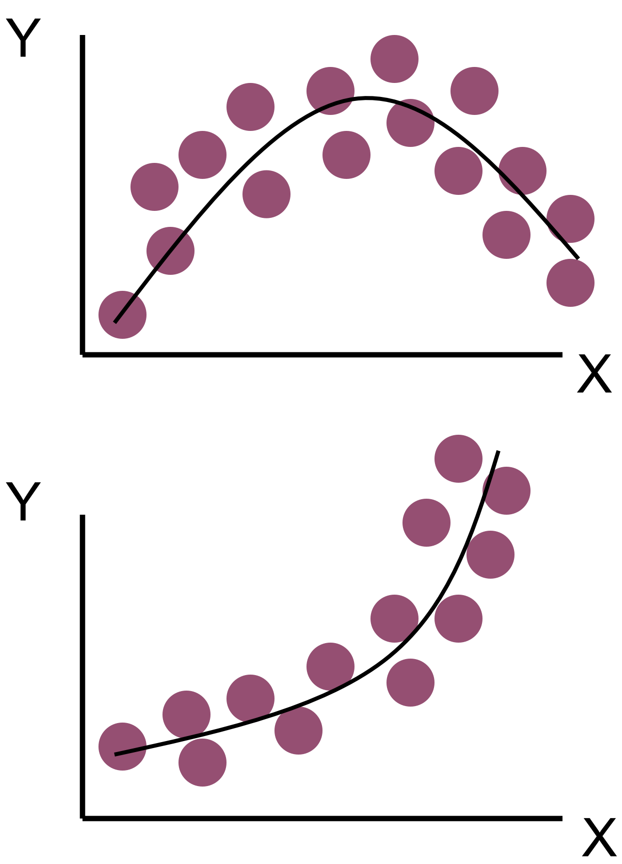
▪Slide from: Statistics for Managers Using Microsoft® Excel 4th Edition, 2004 Prentice-Hall

Linear Correlation

Linear relationships

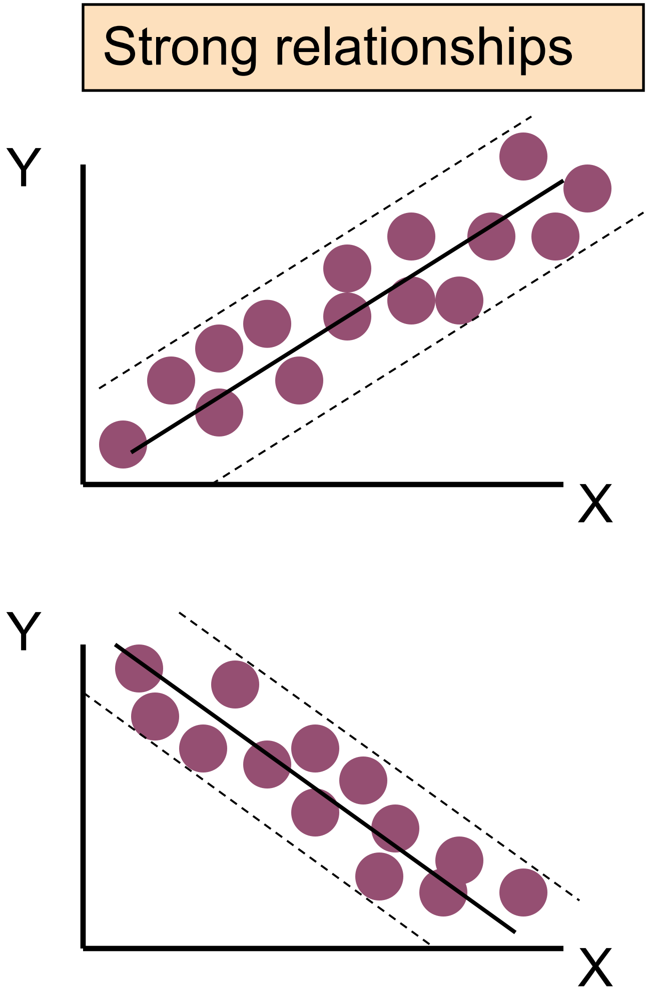


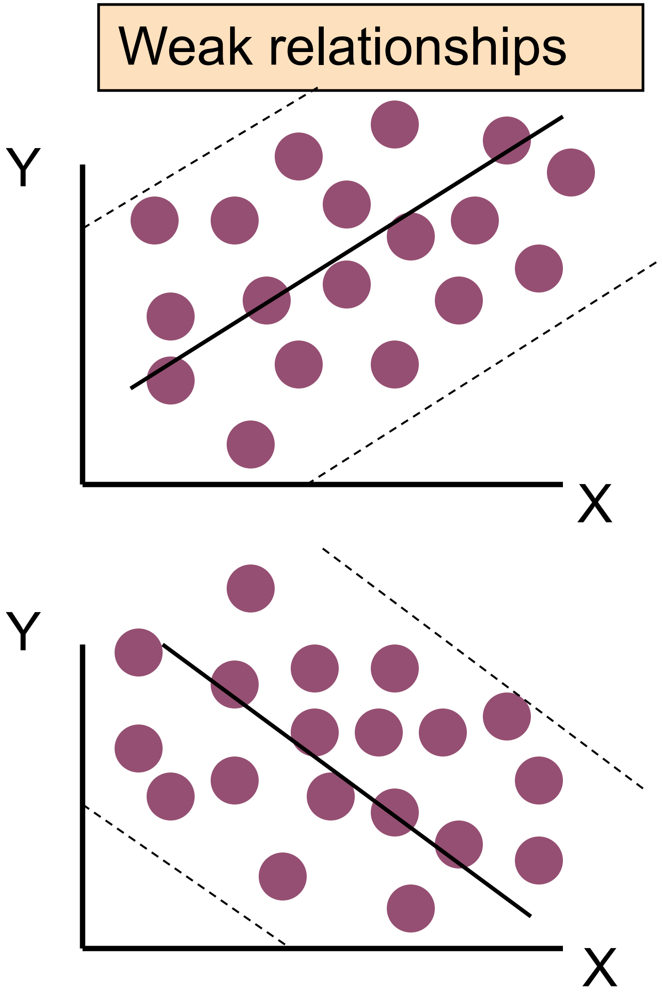
Curvilinear relationships



▪Slide from: Statistics for Managers Using Microsoft® Excel 4th Edition, 2004 Prentice-Hall

Linear Correlation

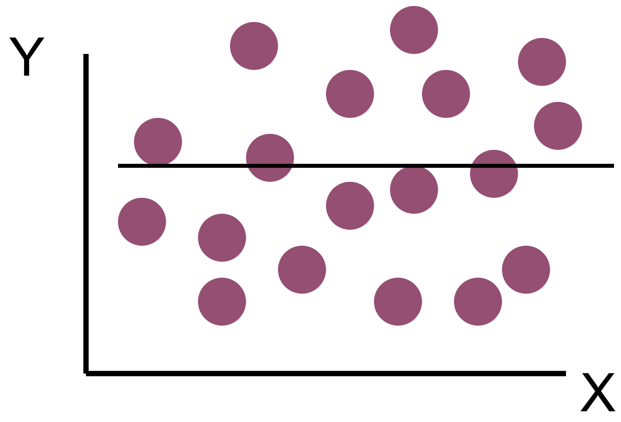


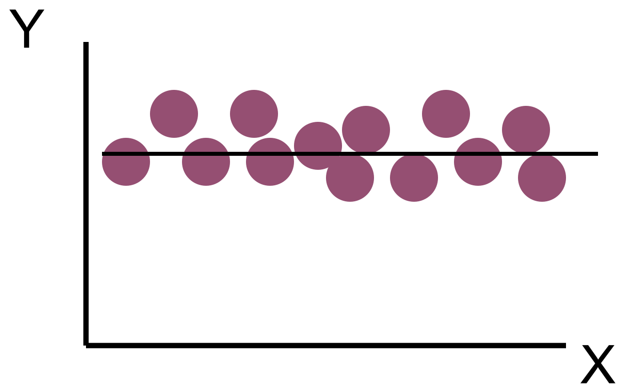


▪Slide from: Statistics for Managers Using Microsoft® Excel 4th Edition, 2004 Prentice-Hall

Linear Correlation

No relationship





- Slide from: Statistics for Managers Using Microsoft® Excel 4th Edition, 2004 Prentice-Hall

Python Implementation – Calculate Correlation

```
import pandas as pd
from sklearn import datasets
#
# Load IRIS dataset
#
iris = datasets.load_iris()
#
# Create dataframe from IRIS dataset
#
df = pd.DataFrame(iris.data, columns=["sepal_length",
"sepal_width", "petal_length", "petal_width"])
df["class"] = iris.target
#
# Calculate pairwise correlation between different columns
#
df.iloc[:, 0:4].corr()
```

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	1.000000	-0.117570	0.871754	0.817941
sepal_width	-0.117570	1.000000	-0.428440	-0.366126
petal_length	0.871754	-0.428440	1.000000	0.962865
petal_width	0.817941	-0.366126	0.962865	1.000000

Next Lecture

Python implementation of linear regression and multiple linear regression