Introduction to Data Science with Python

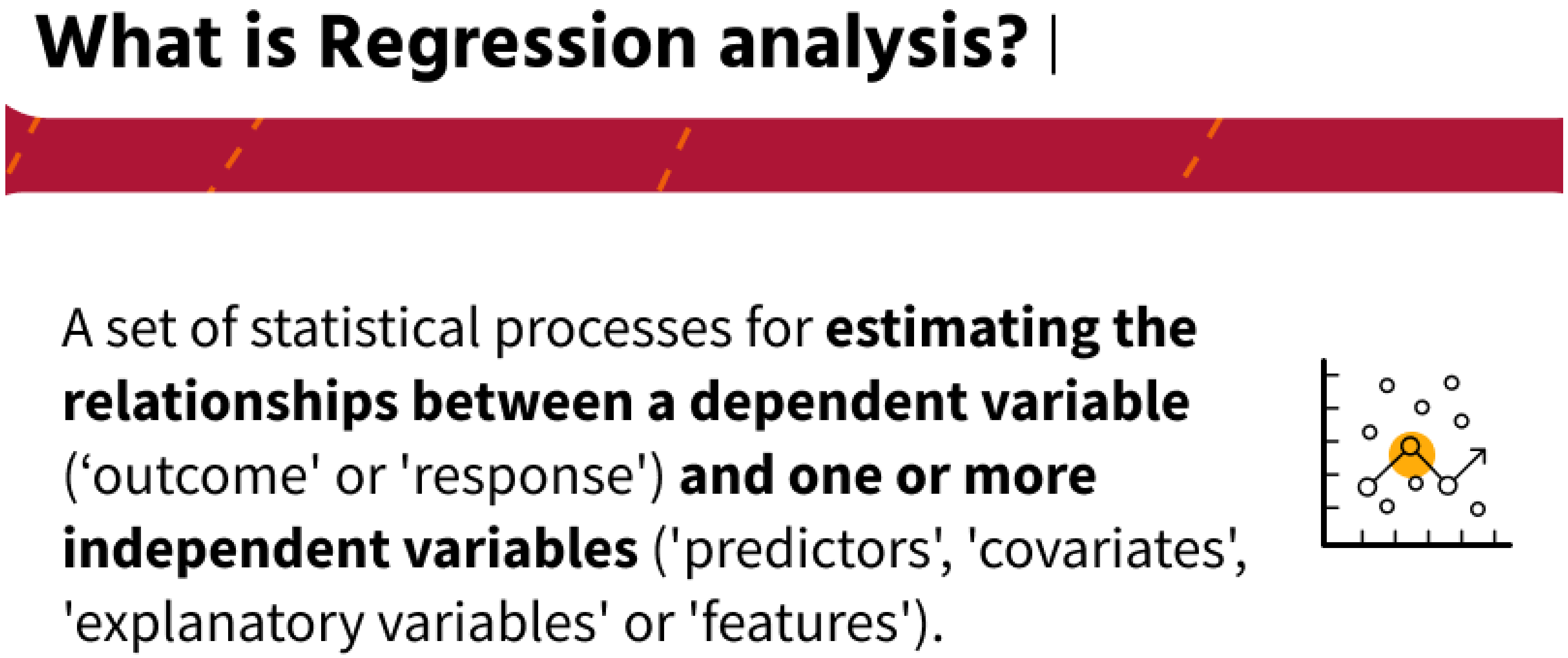
WMASDS04

Week 06: Regression Analysis with Python

Outlines

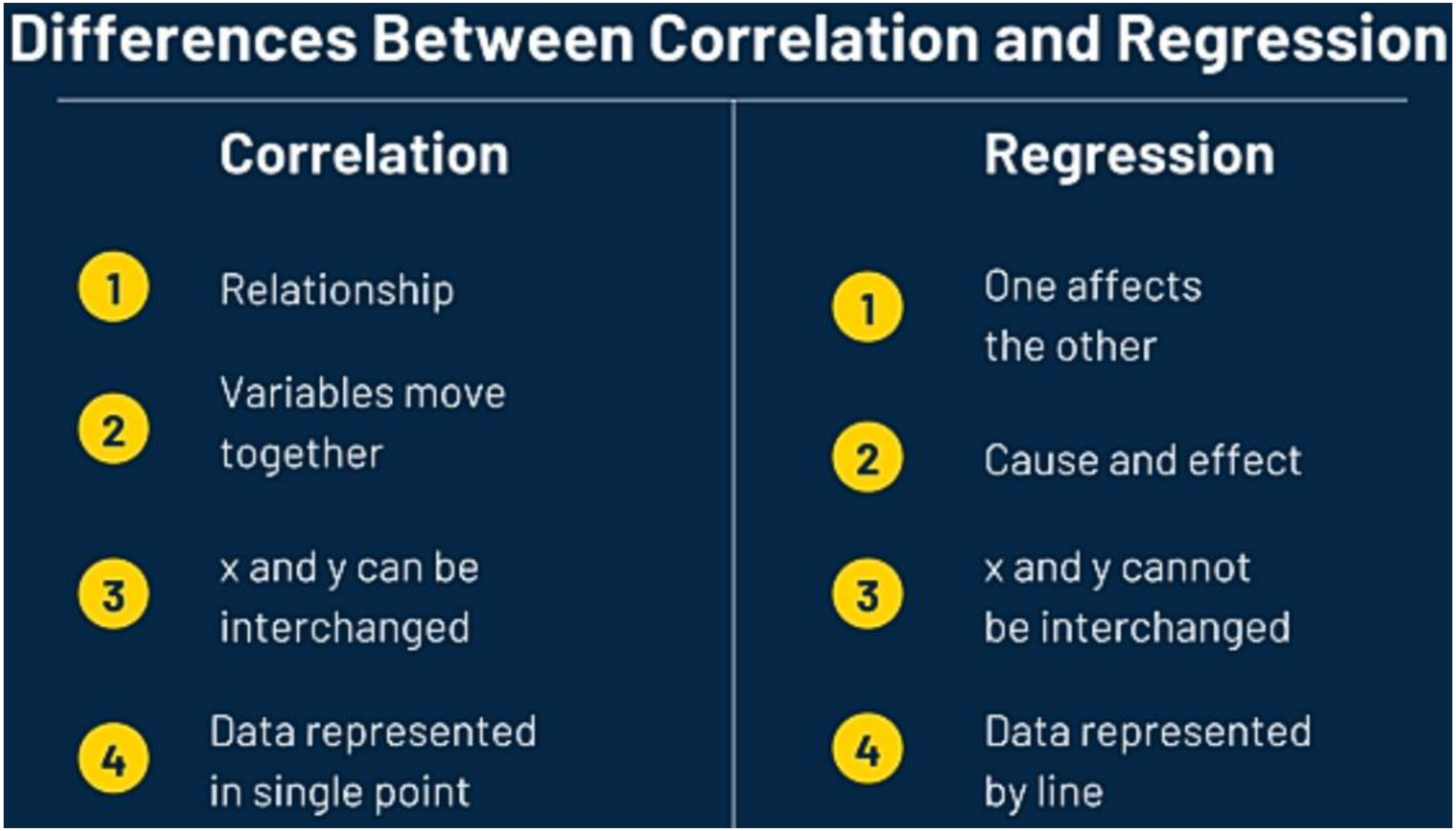
- Linear Regression,
  - Simple Linear Regression Model
  - Multiple Linear Regression Model
- Nonlinear Regression
  - Polynomial Regression
- Sparse Model
  - Least Absolute Shrinkage and Selection Operator (LASSO Regression)
- Logistic Regression.
- Python tools for regression analysis

-



What is Regression analysis?

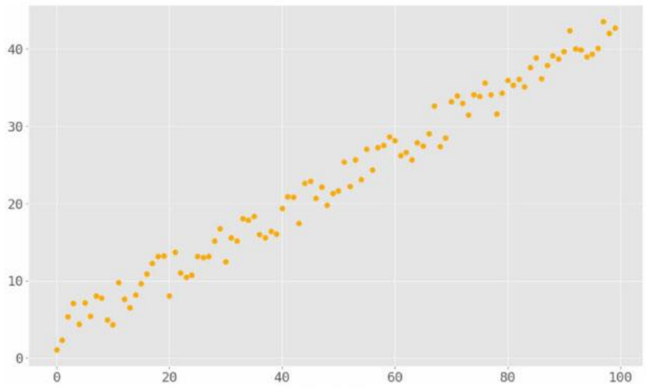
A set of statistical processes for **estimating the relationships between a dependent variable** (‘outcome' or 'response') **and one or more independent variables** ('predictors', 'covariates', 'explanatory variables' or 'features').



Differences Between Correlation and Regression

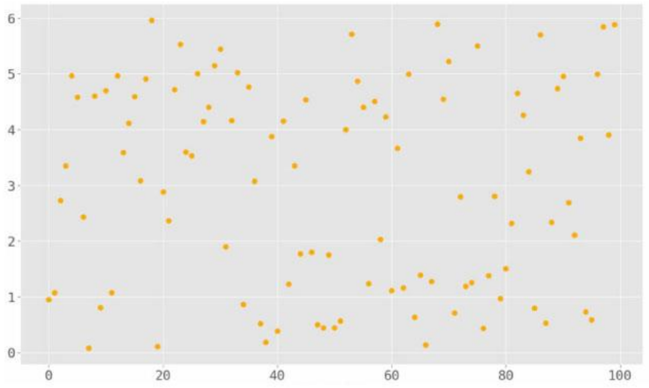
Correlation	Regression
1 Relationship	1 One affects the other
2 Variables move together	2 Cause and effect
3 x and y can be interchanged	3 x and y cannot be interchanged
4 Data represented in single point	4 Data represented by line

•



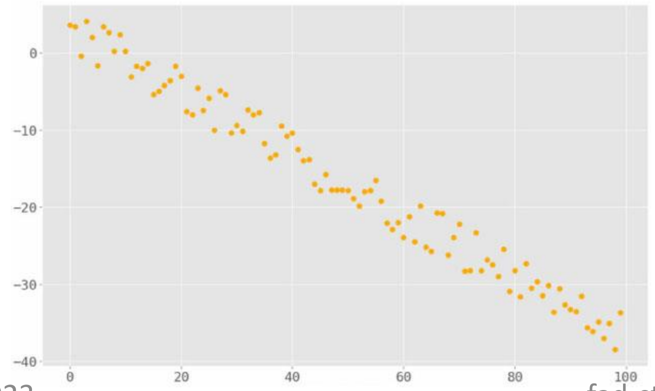
Strong Positive Correlation

$r^2 \approx 0.9$



Zero Correlation

$r^2 \approx 0.0$



Strong Negative Correlation

$r^2 \approx -0.9$

α

Regression	Dependent	Independent
Simple	One	One
Multiple	One	More than one
Multivariate	More than one	One or more

Research Questions

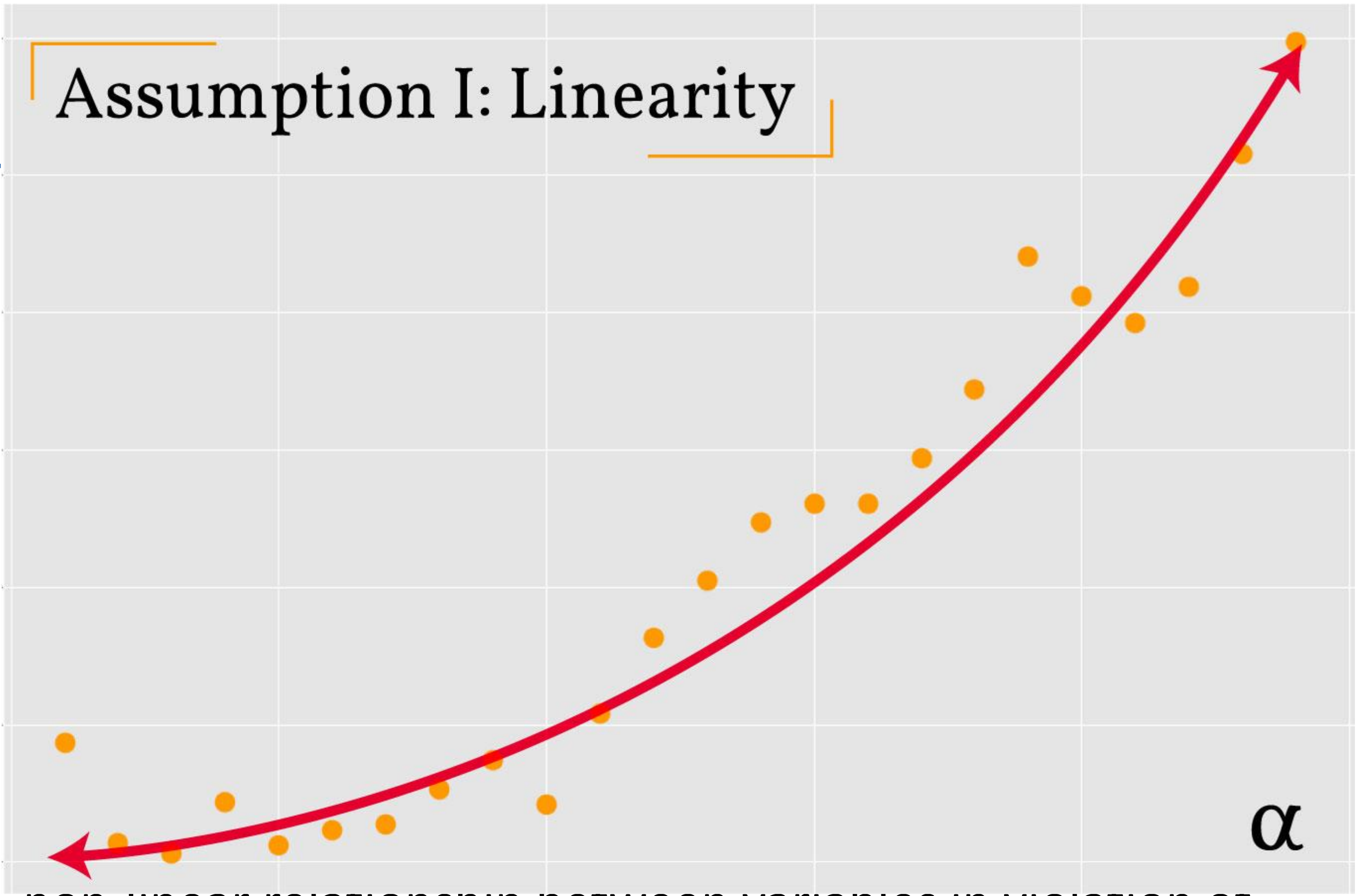
- How does sales volume change with changes in price?
- How is sales volume affected by the weather?
- How does the title of a book affect its sales?
- How does the amount of a drug absorbed vary with the patient's body weight; and does this relationship depend on blood pressure?
- **How** many customers can I expect today?
- At **what** time should I go home to avoid traffic jams?
- **What** is the chance of rain on the next two Mondays; and what is the expected temperature?

Regression Model

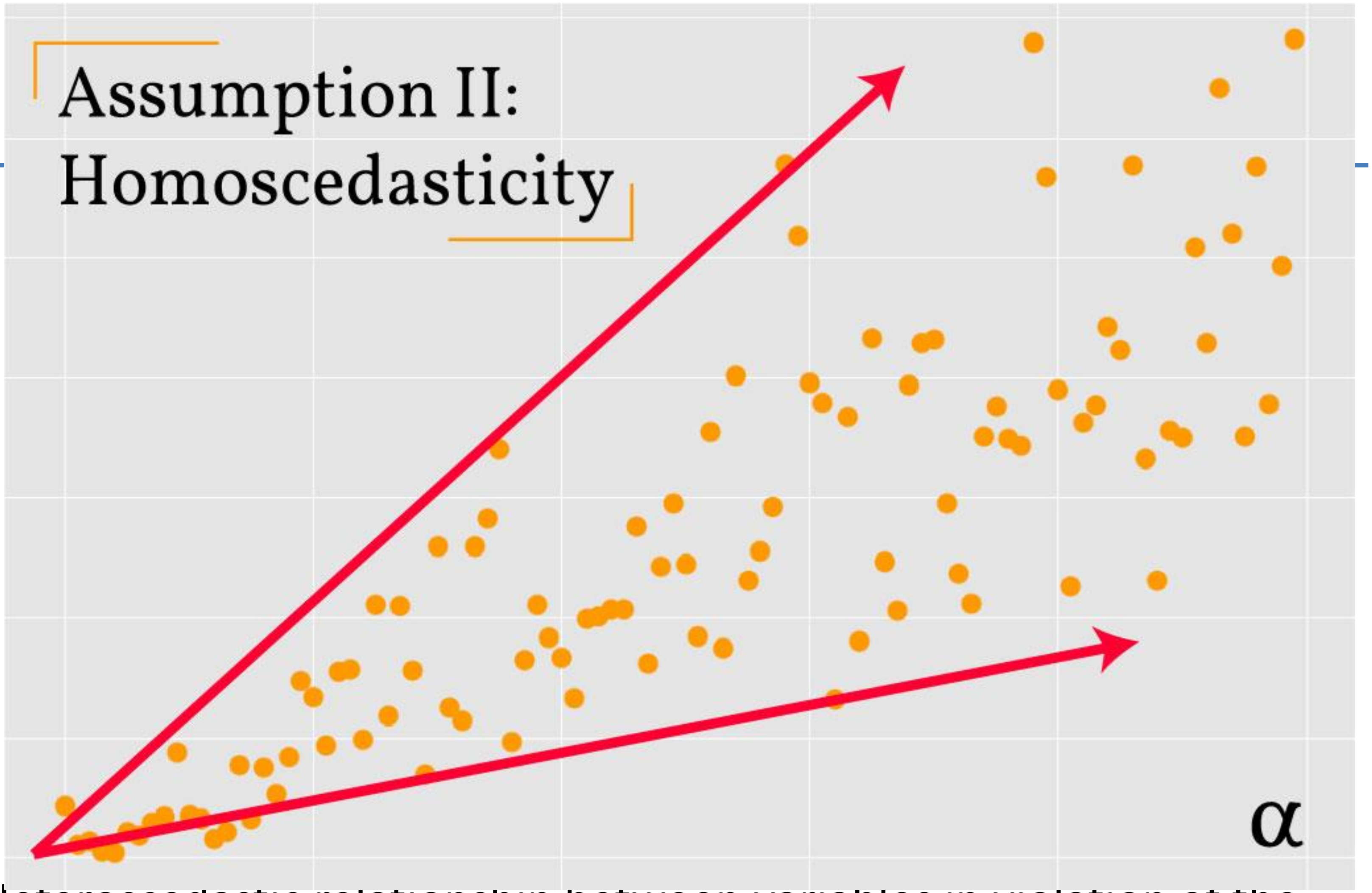
- *response* can be expressed as a combination of one or more (independent) *variables*.
- This process involves the transition *from* ***data to model***.
- Regression model can be useful in different tasks:
  - (1) **analyzing the behavior** of data (the relation between the response and the variables),
  - (2) **predicting data values** (whether continuous or discrete), and
  - (3) **finding important variables** for the model.

Assumptions of Linear Regression

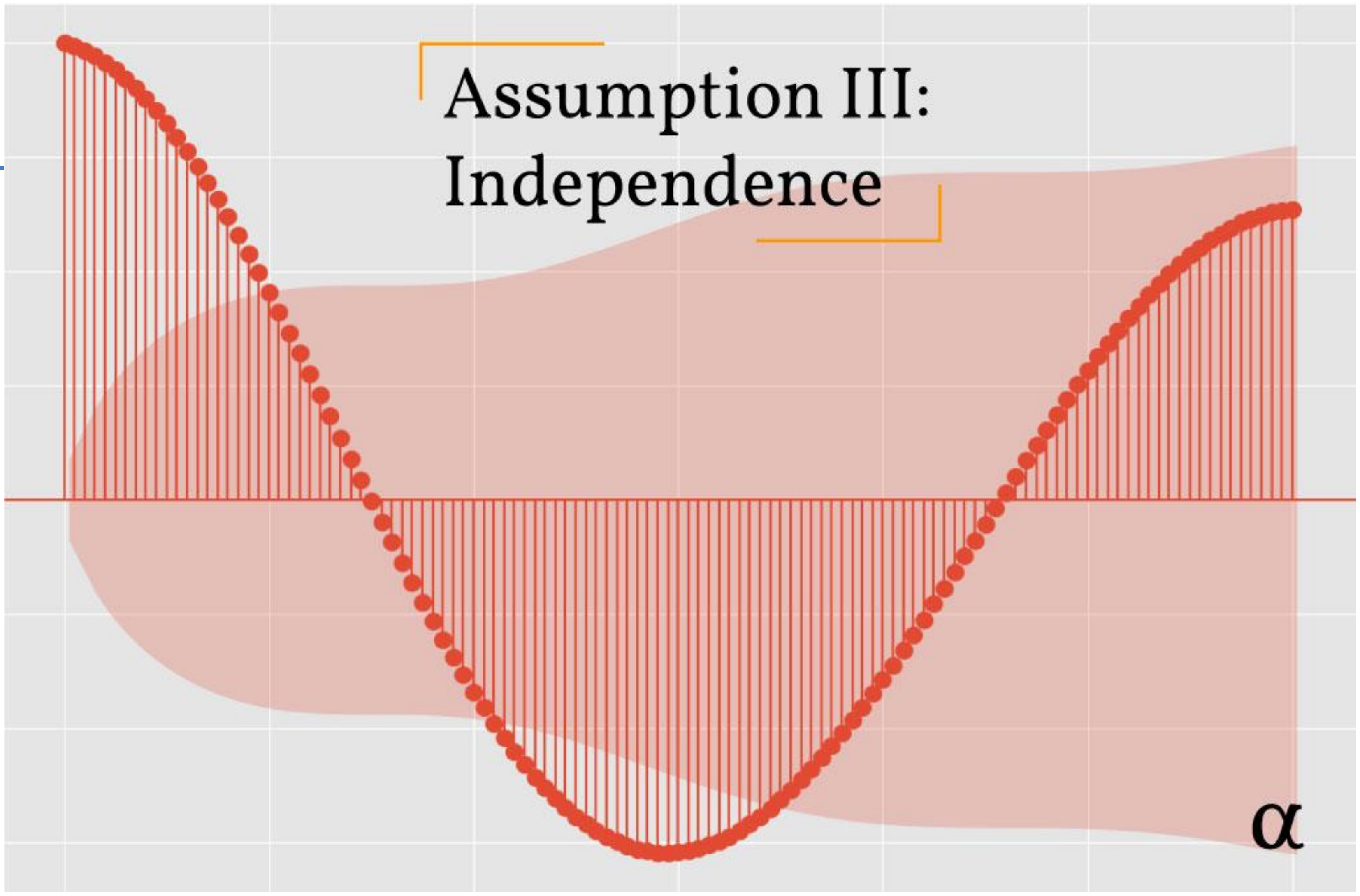
- Linearity
- Homoscedasticity
- Independence
- Normality
- No Multicollinearity



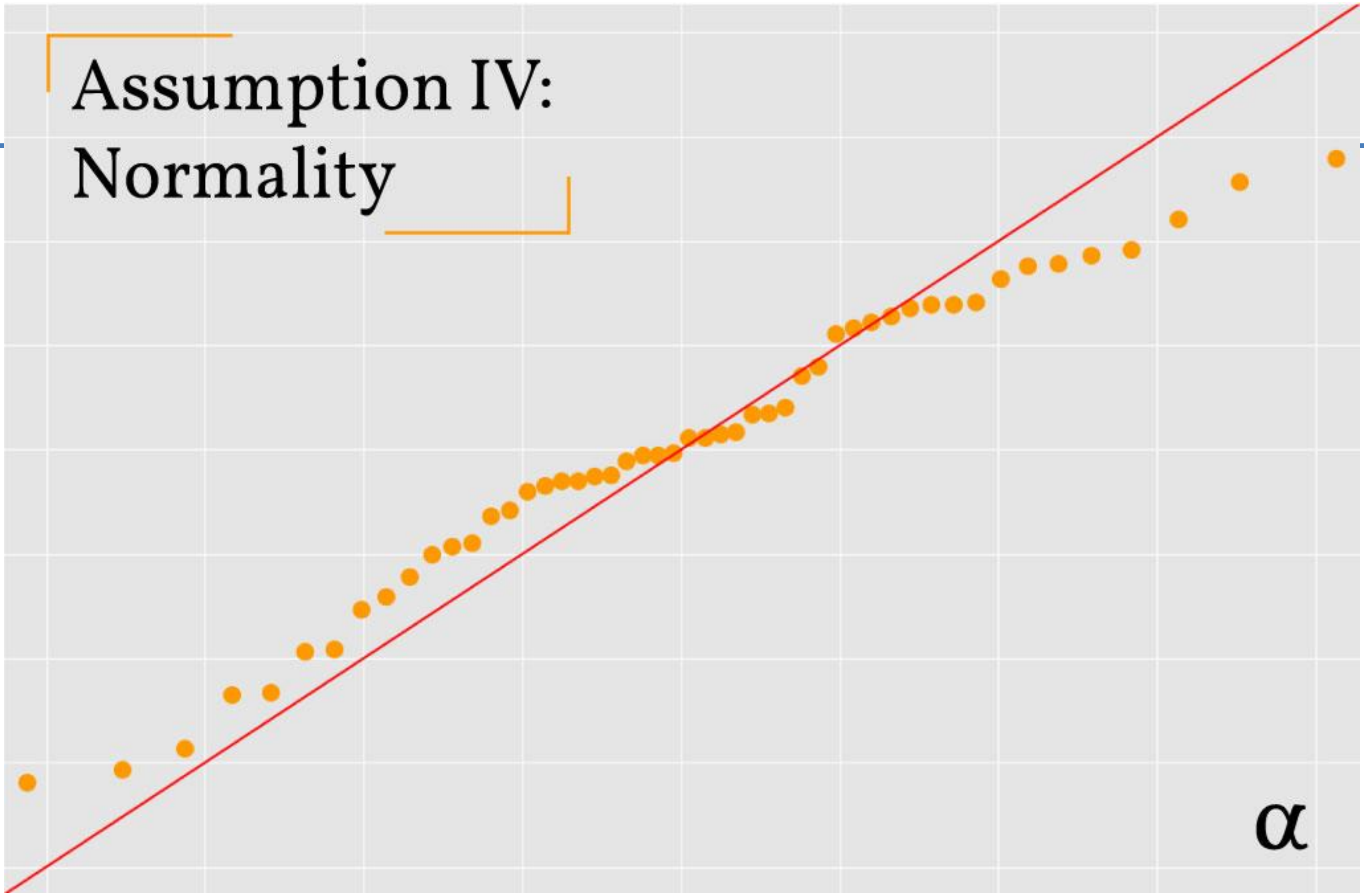
- non-linear relationship between variables in violation of the linearity assumption.



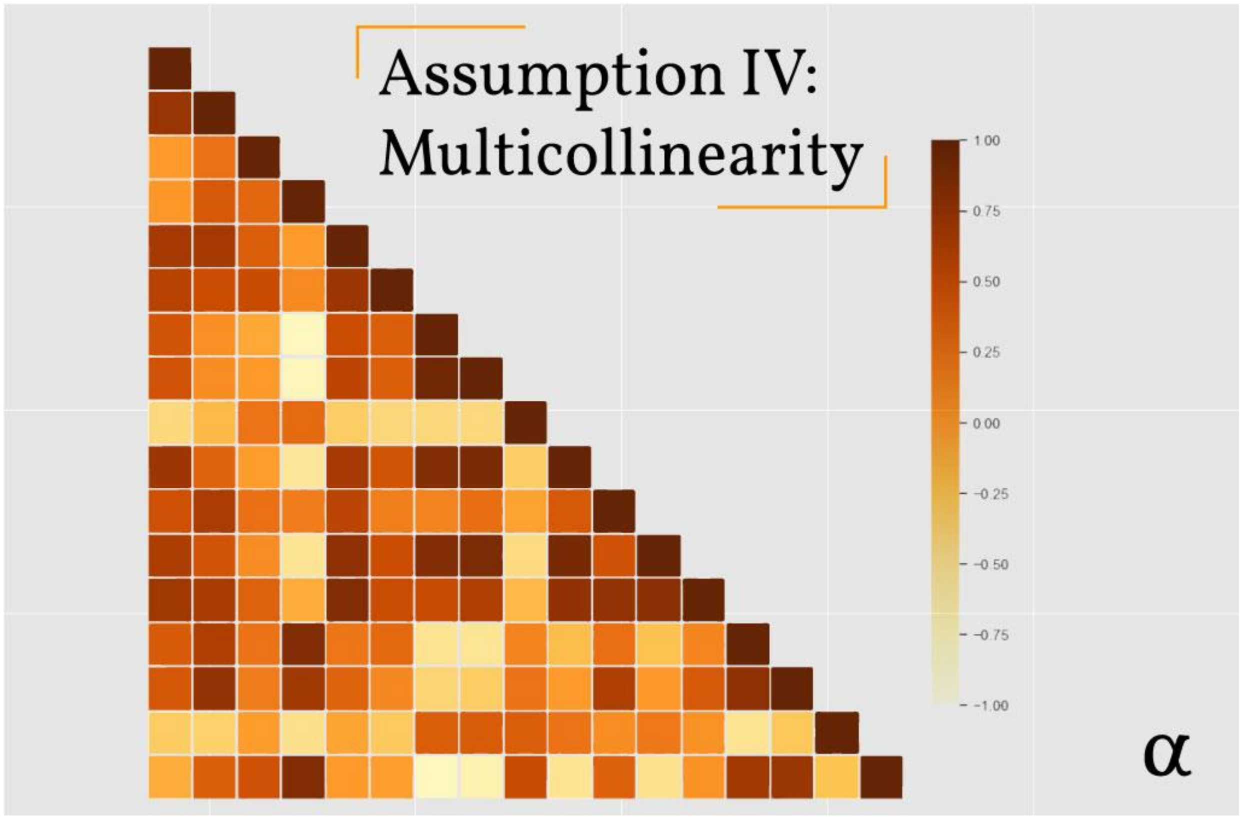
- Heteroscedastic relationship between variables in violation of the homoscedasticity assumption.



- An autocorrelation plot showing a dependency among predictor variables in violation of the independence assumption



- Q-Q Plot of residuals illustrating an approximately-normal distribution in accordance with the normality assumption.



- **A correlation matrix showing several strong linear relationships among predictor variables in violation of the assumption of no Multicollinearity**

Simple Linear Regression

- *Simple linear regression* describes the relationship between a dependent variable and an independent variable with a linear model of y on x:

$$\boldsymbol{y} = a_0 + a_1\boldsymbol{x} + \mathbf{e},$$

where the parameter a_0 is called the *intercept* or the *constant term*

a_1 is known as the regression coefficient.

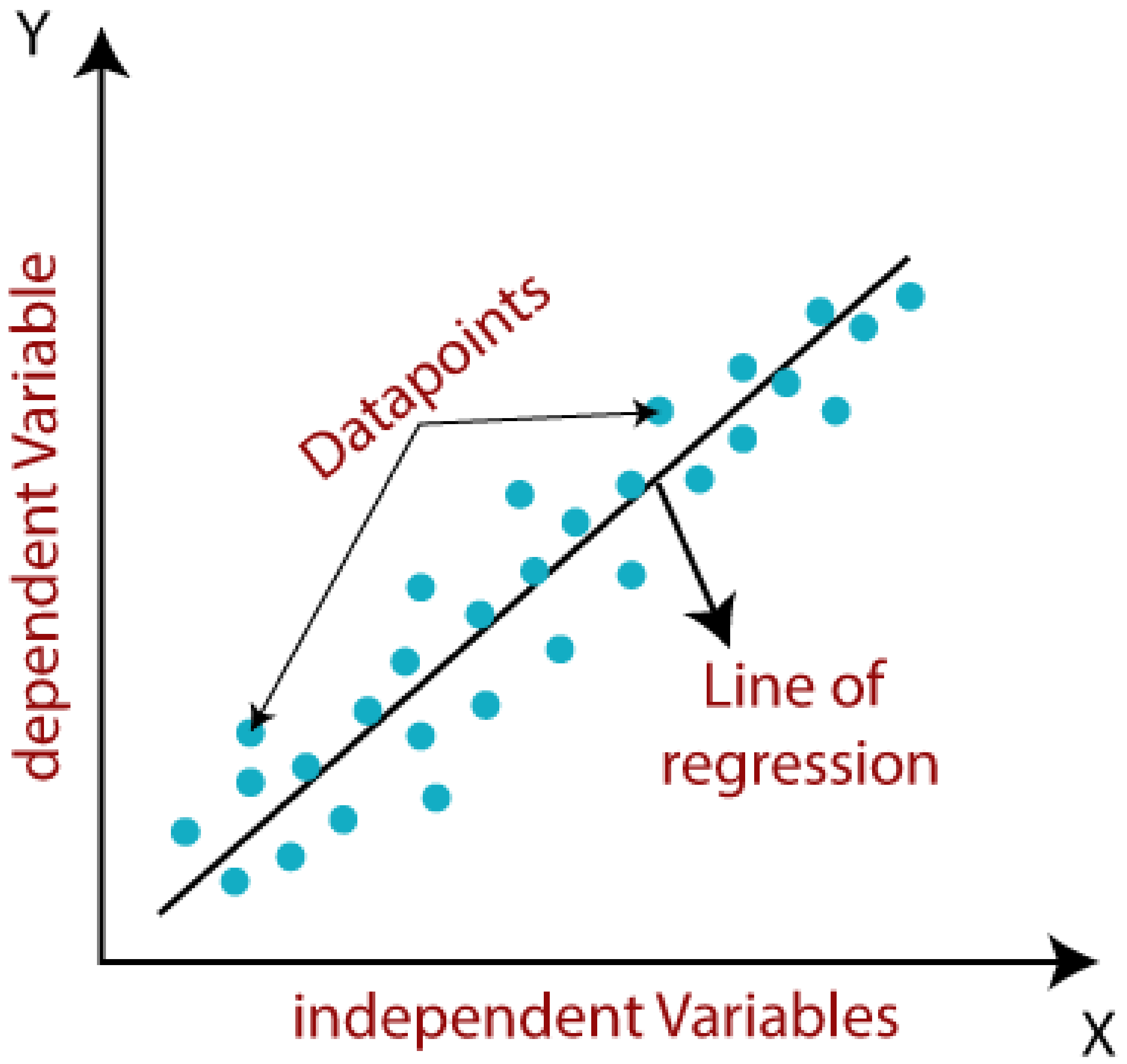
- We can only establish that change in the value of the outcome variable (*Y*) is associated with change in the value of feature *X*,
  - regression technique cannot be used for establishing causal relationship between two variables.

- By rearranging:

$$\boxed{y = a_0 + a_1 x + e} \longrightarrow \boxed{e = y - a_0 - a_1 x}$$

- e is the error or residual, the discrepancy between the true value of y
- a_0+a_1x is the approximate value

- .


Y
dependent Variable
Datapoints
Line of regression
independent Variables
X

- Assumptions of Linear Regression
  - Linearity
  - Homoscedasticity (constant variance)
  - Independence
  - Normality
  - Multicollinearity

Best Fitted Line?

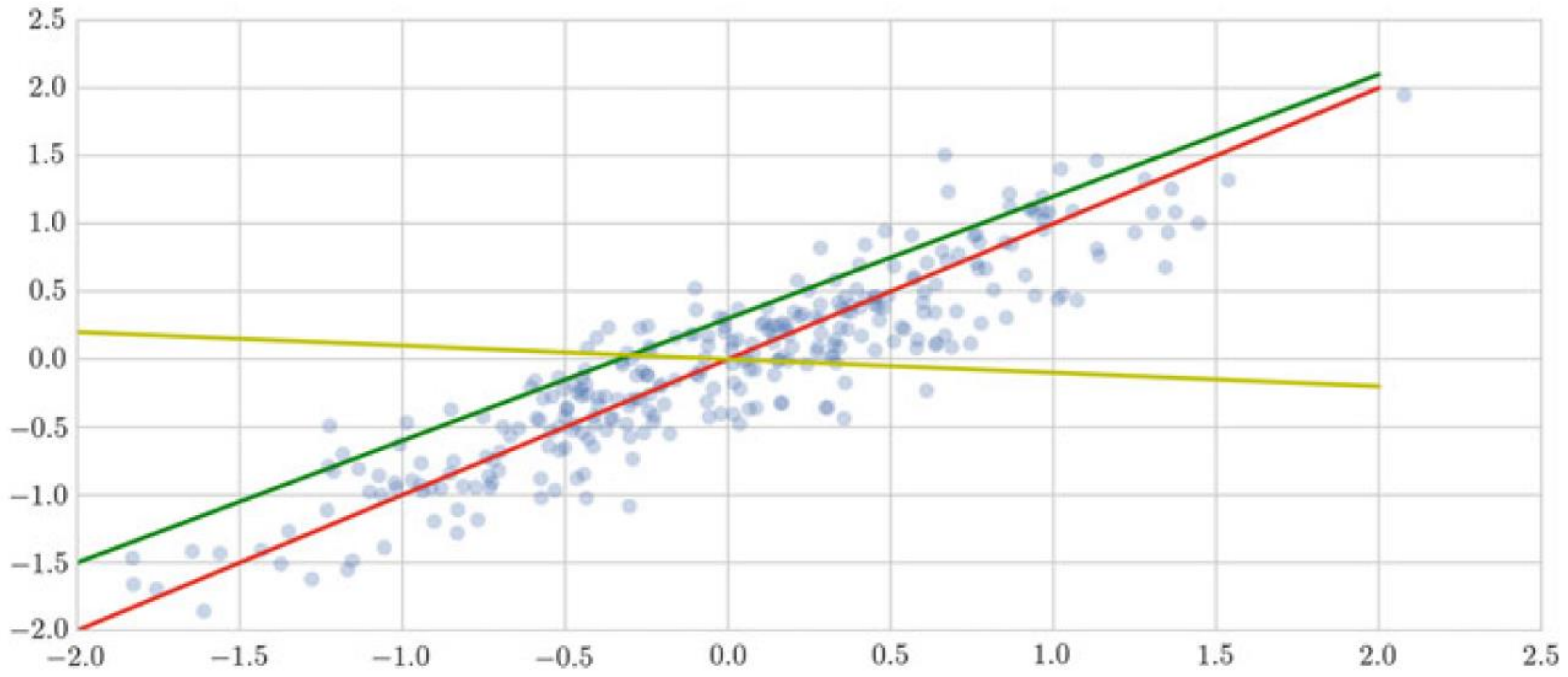


Fig. 6.1 Illustration of different simple linear regression models. *Blue points* correspond to a set of random points sampled from a univariate normal (Gaussian) distribution. *Red*, *green* and *yellow lines* are three different simple linear regression models

Ordinary Least Square Method

- By minimizing the sum of the squares of the residuals between the measured y and the y calculated with the linear mode

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(y_{i,measured} - y_{i,\mathrm{model}}\right)^2$$

$$\boxed{S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i\right)^2}$$

Coefficients of SLR Model

- After several mathematical steps, a_0 and a_1 will yields:

$$\boxed{a_1 = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2}}$$

$$\boxed{a_0 = \bar{y} - a_1\bar{x}}$$

- Where $\bar{y}$ and $\bar{x}$ are the means of y and x, respectively

OLS to SLR Model

- Best straight line: $y = a_0 + a_1x$
- Determine a_0 and a_1: Apply the least-square criterion
- Minimize $S = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - a_0 - a_1x_i)^2$ with respect to a_0 and a_1
- Set $\frac{\partial S}{\partial a_0} = -2\sum_{i=1}^{n}(y_i - a_0 - a_1x_i) = 0$ and $\frac{\partial S}{\partial a_1} = -2\sum_{i=1}^{n}(y_i - a_0 - a_1x_i)x_i = 0.$
- Solve the simultaneous linear equations for a_0 and a_1:

$$a_1 = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2} \text{ and } a_0 = \frac{1}{n}\sum_{i=1}^{n} y_i - \left(\frac{1}{n}\sum_{i=1}^{n} x_i\right)a_1$$

Practical Case: Sea Ice Data and Climate Change

- Research question: Is the climate really changing?
- we want to show the effect of the climate change by determining whether the *sea ice area* (or *extent*) has decreased over the years.
  - Sea ice area refers to the total area covered by ice, whereas sea ice extent is the area of ocean with at least 15% sea ice. Reliable measurement of sea ice edges began with the satellite era in the late 1970s. Before then, sea ice area and extent were monitored less precisely by a combination of ships, buoys, and aircraft.
- In order to check whether there is an anomaly in the evolution of sea ice extent over recent years, we want to build a simple linear regression model and analyze the fitting; but before we need to perform several processing steps.

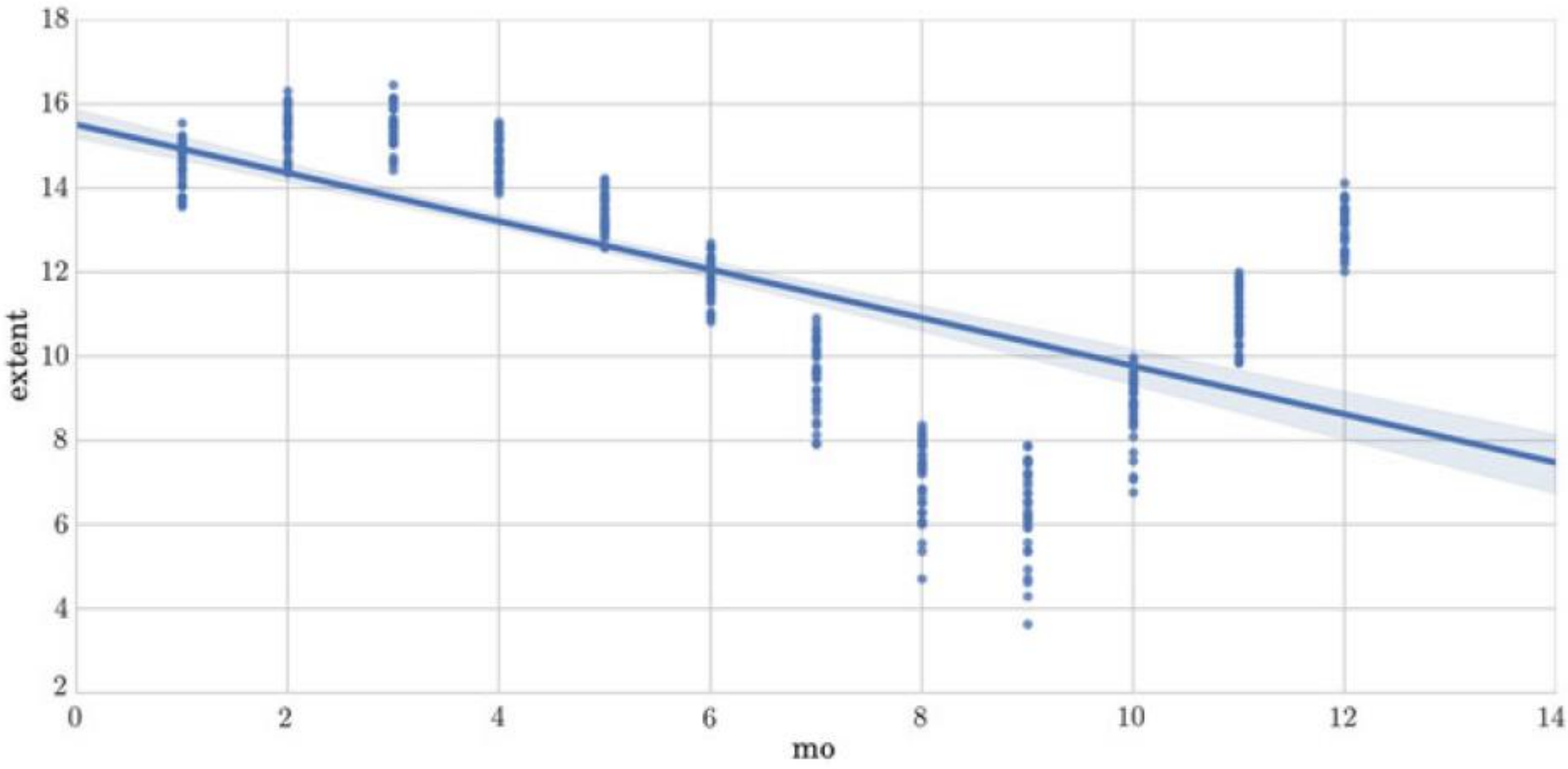


Fig. 6.2 Ice extent data by month

Different Types of Regression Model

- Simple linear regression describes the relationship between the variable and the response with a straight line. $\mathbf{y} = a_0 + a_1\mathbf{x}$
- In the case of multiple linear regression, we extend this idea by fitting a *d*-dimensional hyperplane to our *d* variables.

  $y = a_0 + a_1x_1 + \cdot\cdot\cdot + a_dx_d$
- When the response depends on the variables in nonlinear ways, this model then consider nonlinear transformations $\phi(\cdot)$ of the variables:

  $y = a_1\varphi(x_1)+\cdot\cdot\cdot+a_d\varphi(x_d)$
- This model is called polynomial regression and it is a popular nonlinear regression technique which models the relationship between the response and the feature variables as an *p*-th order polynomial.
- However, using higher-order polynomial can involve computational complexity and overfitting.
- Overfitting occurs when a model fits the characteristics of the training data and loses the capacity to generalize from the seen to predict the unseen.

Sparse Model

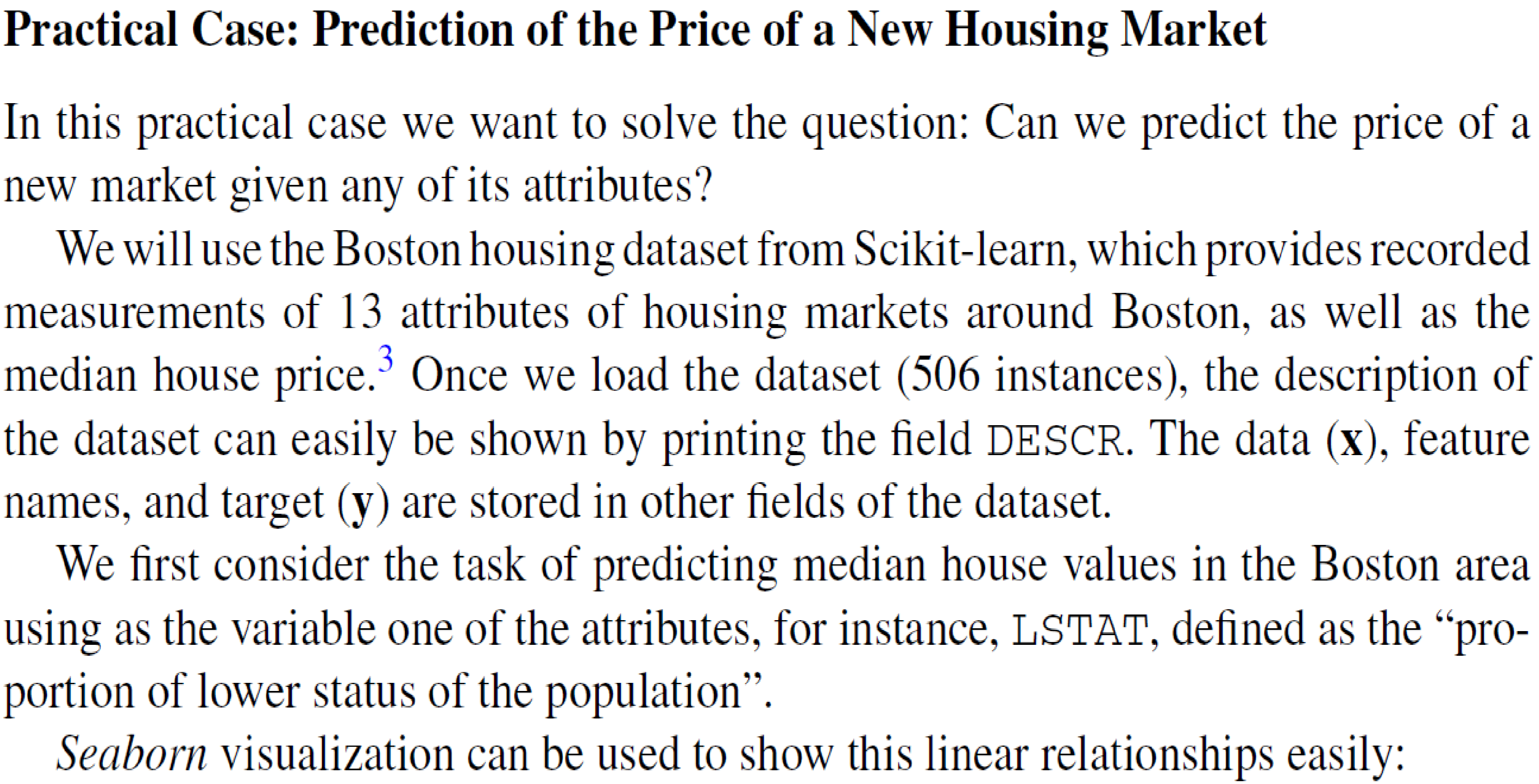
Often, in real problems, there are uninformative variables in the data which prevent proper modeling of the problem and thus, the building of a correct regression model. In such cases, a feature selection process is crucial to select only the informative features and discard non-informative ones. This can be achieved by *sparse methods* which use a penalization approach, such as *LASSO* (least absolute shrinkage and selection operator) to set some model coefficients to zero (thereby discarding those variables). Sparsity can be seen as an application of Occam's razor: prefer simpler models to complex ones.

Given the set of samples $(\mathbf{X}, \mathbf{y})$, the objective of a sparse model is to minimize the SSE through a restriction (or penalty):

$$\frac{1}{2n}||\mathbf{X}\mathbf{w} - \mathbf{y}||_2^2 + \alpha||\mathbf{w}||_1,$$

where $||\mathbf{w}||_1$ is the $L1$-norm of the parameter vector $\mathbf{w} = (a_0, \ldots, a_d)$.

Example



Practical Case: Prediction of the Price of a New Housing Market

In this practical case we want to solve the question: Can we predict the price of a new market given any of its attributes?

We will use the Boston housing dataset from Scikit-learn, which provides recorded measurements of 13 attributes of housing markets around Boston, as well as the median house price.[3] Once we load the dataset (506 instances), the description of the dataset can easily be shown by printing the field `DESCR`. The data (**x**), feature names, and target (**y**) are stored in other fields of the dataset.

We first consider the task of predicting median house values in the Boston area using as the variable one of the attributes, for instance, `LSTAT`, defined as the "proportion of lower status of the population".

Seaborn visualization can be used to show this linear relationships easily:

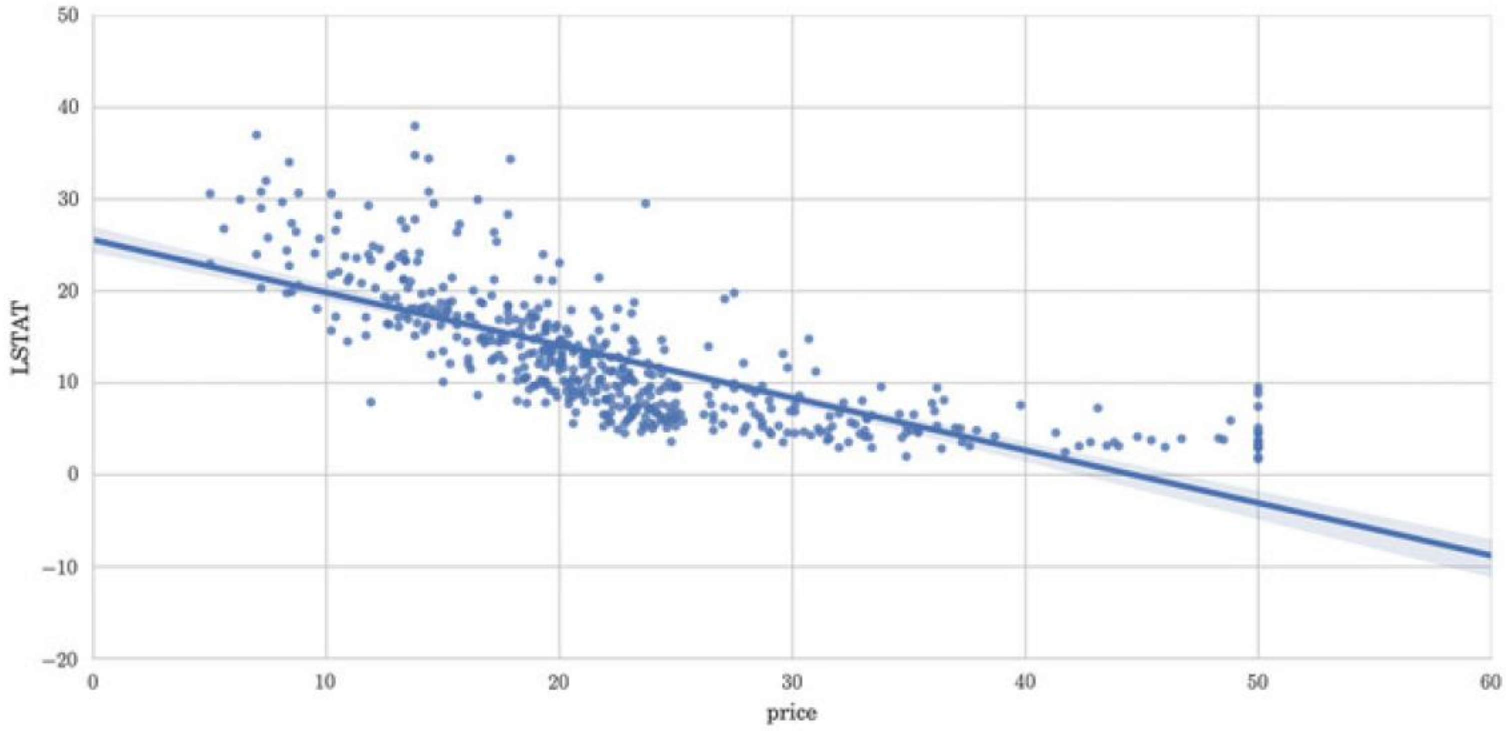


Fig. 6.5 Scatter plot of Boston data (`LSTAT` versus `price`) and their linear relationship (using `lmplot`)

In Fig. 6.5, we can clearly see that the relationship between `price` and `LSTAT` is nonlinear, since the straight line is a poor fit. We can examine whether a better fit can be obtained by including higher-order terms. For example, a quadratic model:

$$\mathbf{y}_i \approx a_0 + a_1\mathbf{x}_i + a_2\mathbf{x}_i^2$$

The `lmplot` function allows to easily change the order of the model as is done in the next code, which outputs Fig. 6.6, where we observe a better fit.

```
sns.lmplot("price", "LSTAT", df_boston, order = 2)
```

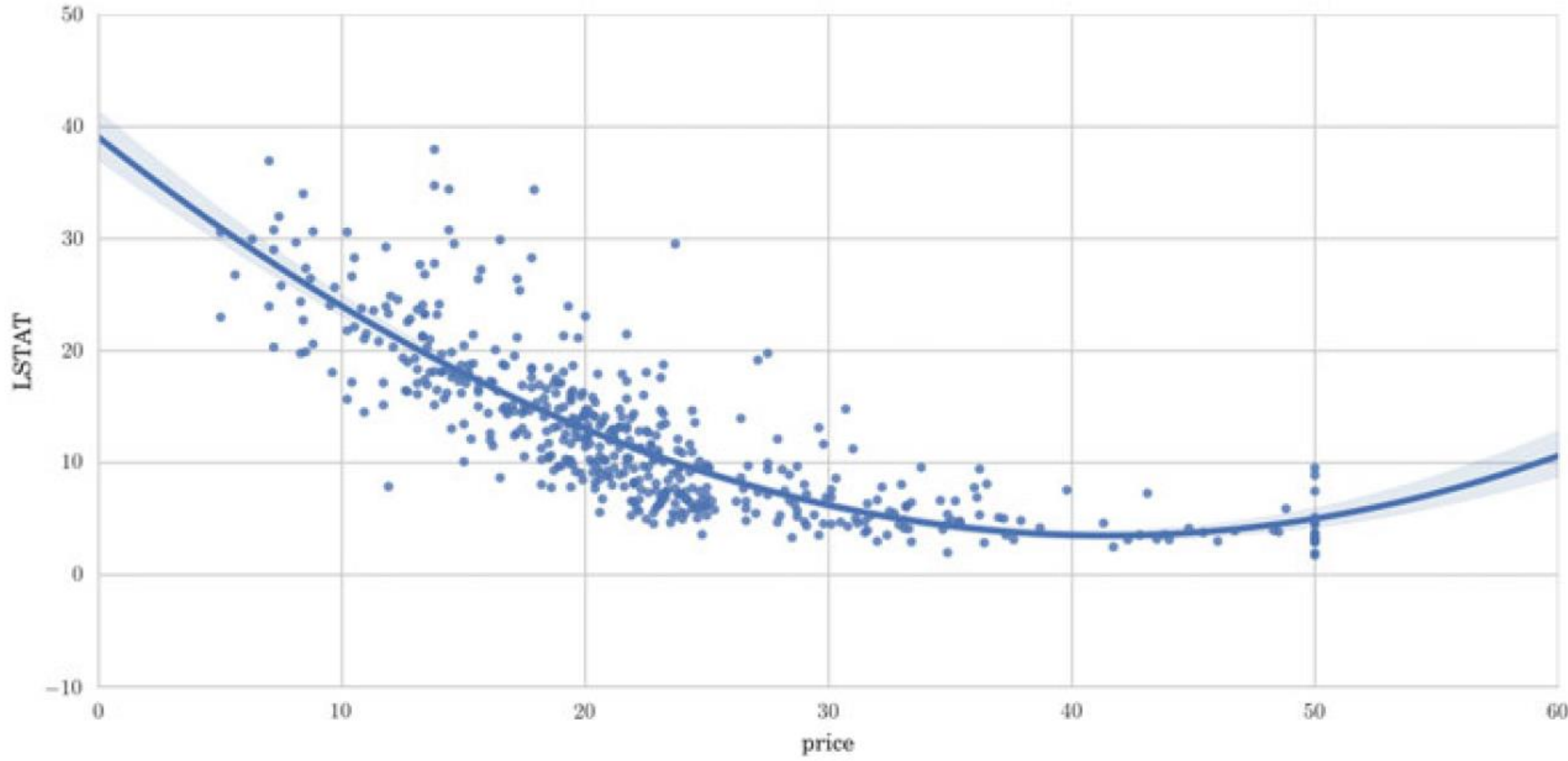
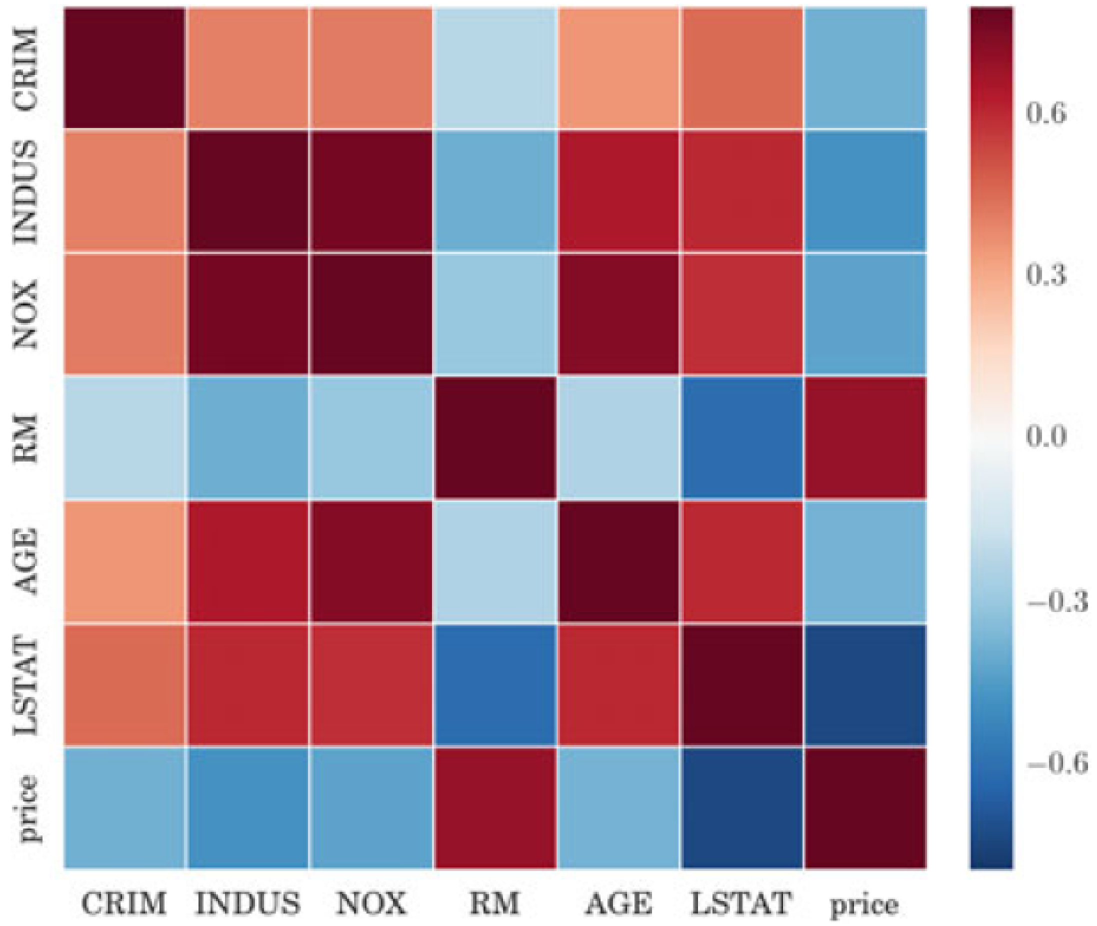


Fig. 6.6 Scatter plot of Boston data (`LSTAT` versus `price`) and their polynomial relationship (using `lmplot` with order 2)

- To study the relation among multiple variables in a dataset, there are different options.
- We can study the relationship between several variables in a dataset by using the functions corr and heatmap which allow to calculate a correlation matrix for a dataset and draws a *heat map* with the correlation values.
- The heat map is a matricial image which helps to interpret the correlations among variables.
- For the visualization, we consider six variables in the Boston housing data,
  - CRIM, per capita crime rate by town;
  - INDUS, proportion of non-retail business acres per town;
  - NOX, nitric oxide concentrations (parts per 10 million);
  - RM, average number of rooms per dwelling;
  - AGE, proportion of owner-occupied units built prior to 1940; and
  - LSTAT.

Fig. 6.7 Correlation plot: heat map representing the correlation between seven pairs of variables in the Boston housing dataset



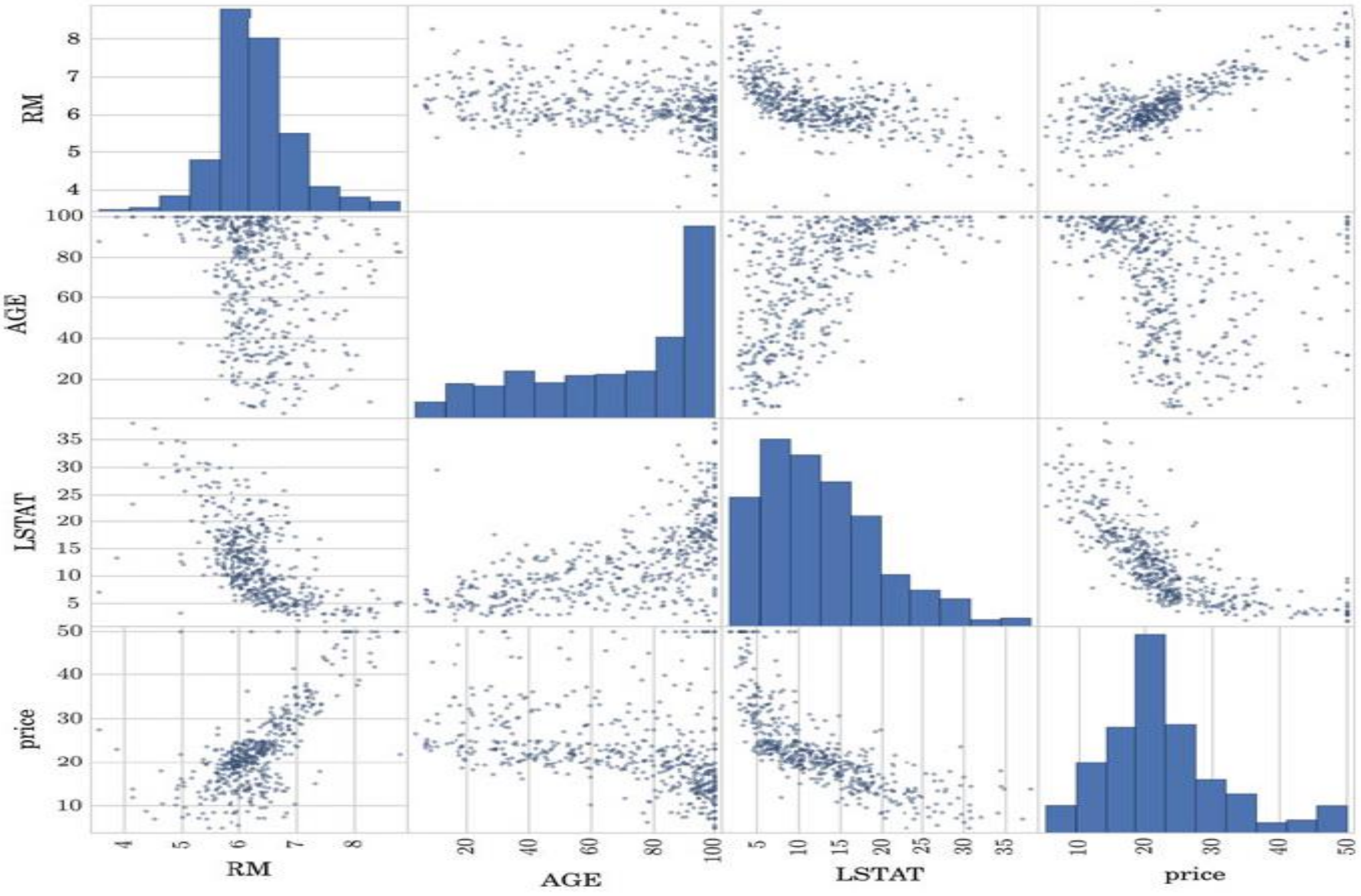


Fig. 6.8 Scatter plot of Boston housing dataset

- For the evaluation of the prediction power of the model with new samples, we split the data into a training set and a testing set, and we compute the linear regression score, which returns the coefficient of determination R^2 of the prediction.
- We can also calculate the MSE.

```
from sklearn import linear_model
train_size = X_boston.shape[0]/2
X_train = X_boston[:train_size]
X_test = X_boston[train_size:]
y_train = y_boston[:train_size]
y_test = y_boston[train_size:]
print 'Training and testing set sizes',
       X_train.shape, X_test.shape
regr = LinearRegression()
regr.fit(X_train, y_train)
print 'Coeff and intercept:',
      regr.coef_, regr.intercept_
print 'Testing Score:', regr.score(X_test, y_test) print '
    Training
MSE: ',
      np.mean((regr.predict(X_train) - y_train)**2)
print 'Testing MSE: ',
      np.mean((regr.predict(X_test) - y_test)**2)
```

- We can see that all the coefficients obtained are different from zero, meaning that no variable is discarded.
- Next, we try to build a sparse model to predict the price using the most important factors and discarding the non-informative ones.
- To do this, we can create a LASSO regressor, forcing zero coefficients.

```
regr_lasso = linear_model.Lasso(alpha = .3)
regr_lasso.fit(X_train, y_train) print 'Coeff and intercept:
    ',regr_lasso.coef_
print 'Tesing Score:', regr_lasso.score(X_test,
y_test) print 'Training MSE: ',
      np.mean((regr_lasso.predict(X_train)  - y_train)**2)
print 'Testing MSE: ',
      np.mean((regr_lasso.predict(X_test) - y_test)**2)
```

```
Coeff and intercept: [ 0. 0.01996512 -0. 0. -0. 7.69894744
-0.03444803 -0.79380636 0.0735163 -0.0143421 -0.66768539
0.01547437 -0.22181817] -6.18324183615
Testing Score: 0.501127529021
Training MSE: 10.7343110095
Testing MSE: 46.5381680949
```

- It can now be seen that the result of the model fitting for a set of sparse coefficients is much better than before (using all the variables), with the score increasing from −2.24 to 0.5.
- This demonstrates that four of the initial variables are not important for the prediction and in fact they confuse the regressor.

With the LASSO result, we can also emphasize the most important factors for determining the price of a new market, based on the coefficient values:

```
ind = np.argsort(np.abs(regr_lasso.coef_))
print 'Ordered variable (from less to more important):',
      boston.feature_names[ind]
```

```
Ordered variable (from less to more important): ['CRIM' 'INDUS'
'CHAS' 'NOX' 'TAX' 'B' 'ZN' 'AGE' 'RAD' 'LSTAT' 'PTRATIO' 'DIS'
'RM']
```

There are also other strategies for feature selection. For instance, we can select the k=5 best features, according to the k highest scores, using the function `SelectKBest` from Scikit-learn:

```
import sklearn.feature_selection as fs
selector = fs.SelectKBest(score_func = fs.f_regression,
                          k = 5)
selector.fit_transform(X_train, y_train) per
selector.fit(X_train,y_train)
print 'Selected features:',
      zip(selector.get_support(), boston.feature_names)
```

True Target versus predicted target

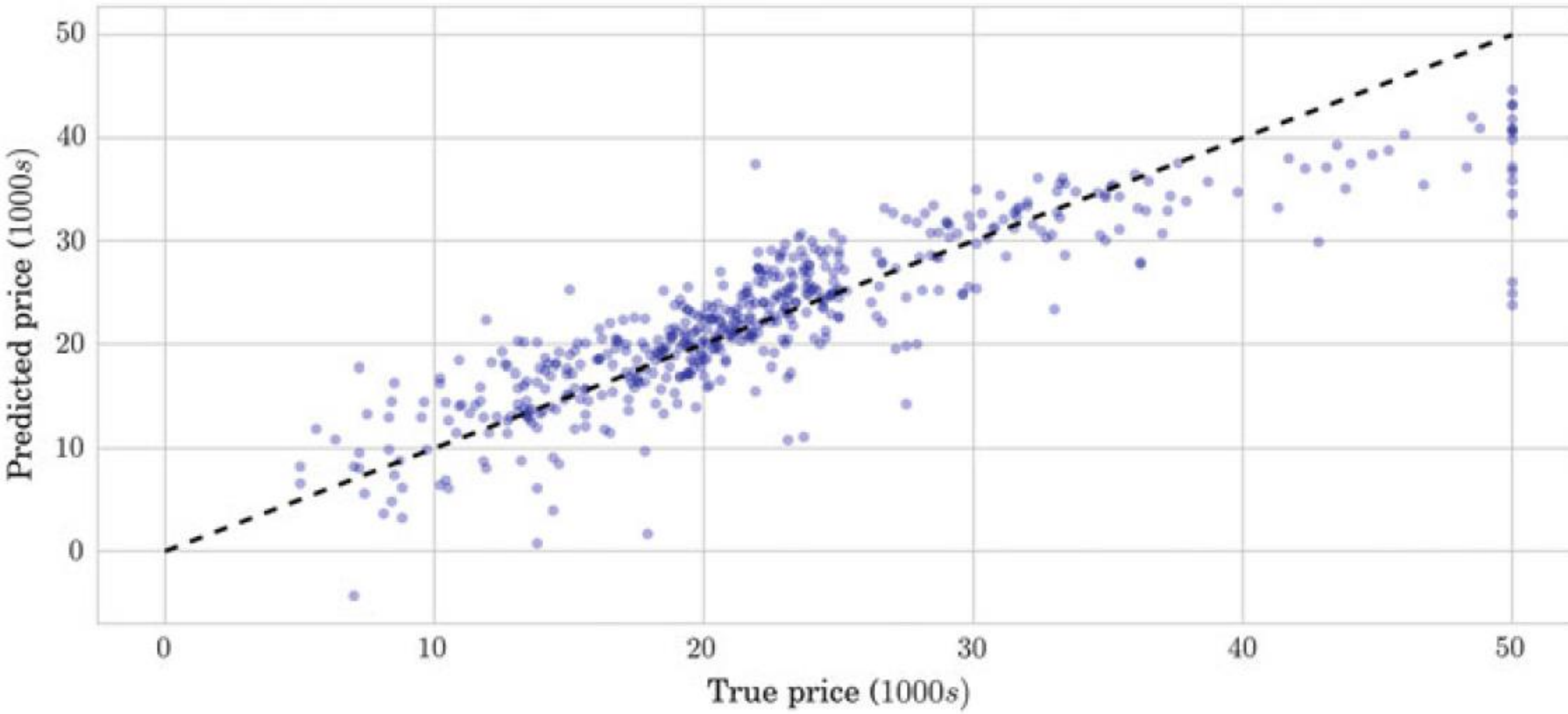


Fig. 6.9 Relation between true (x-axis) and predicted (y-axis) prices

```
clf = LinearRegression()
clf.fit(boston.data, boston.target)
predicted = clf.predict(boston.data)
plt.scatter(boston.target, predicted, alpha = 0.3)
plt.plot([0, 50], [0, 50], '--k')
plt.axis('tight')
plt.xlabel('True price ($1000s)')
plt.ylabel('Predicted price ($1000s)')
```

Logistic Regression

- *Logistic regression* is a type of model of probabilistic statistical classification.
- It is used as a binary model to predict a binary response, the outcome of a categorical dependent variable (i.e., a class label), based on one or more variables.
- The form of the logistic function or sigmoid function is:
- .

$$f(x) = \frac{1}{1 + e^{-\lambda x}}$$

- .
- .

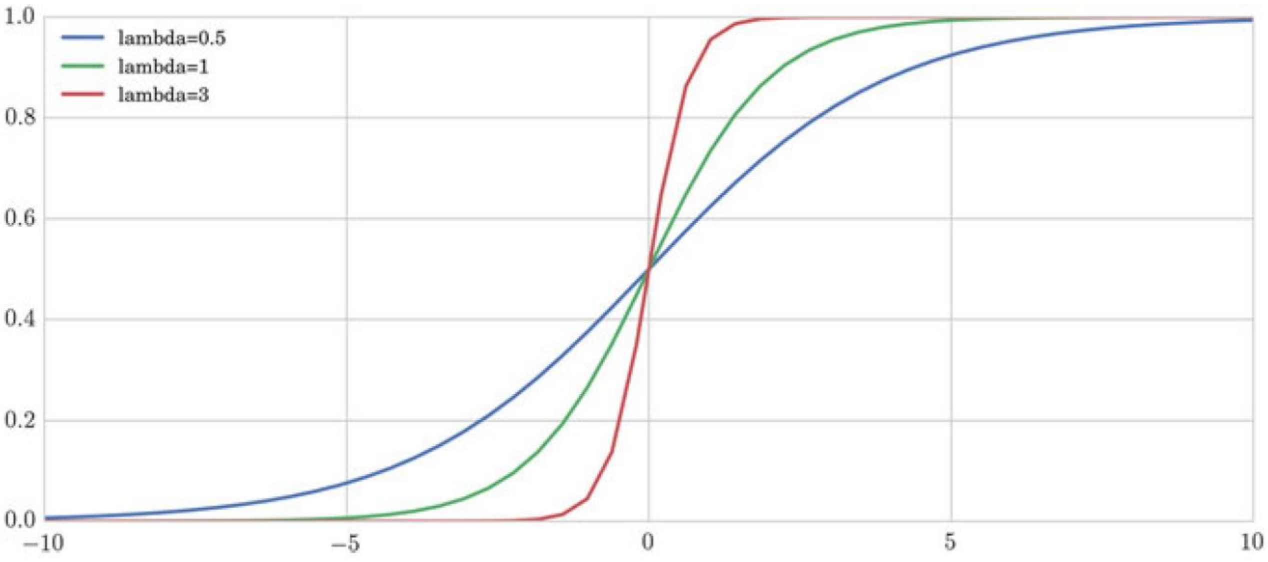


Fig. 6.10 Logistic function for different lambda values

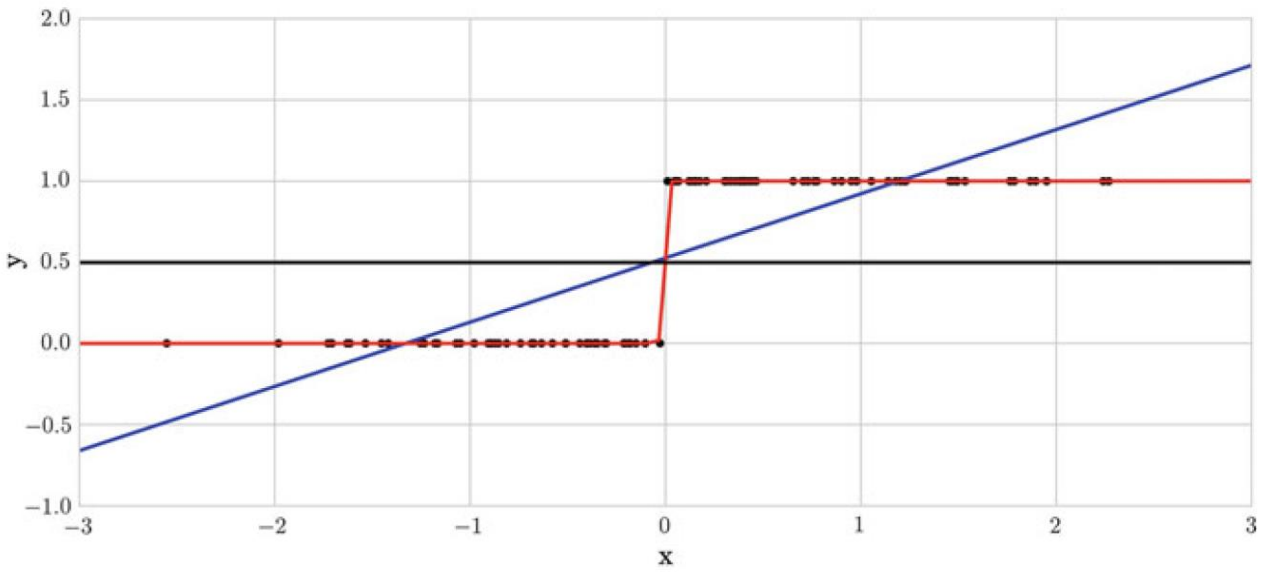


Fig. 6.11 Linear regression (*blue*) versus logistic regression (*red*) for fitting a set of data (*black points*) normally distributed across the 0 and 1 y-values

Logistic Regression: Example

- Practical Case: Winning or Losing Football Team
- What number of goals makes a football team the winner or the loser?
- More concretely, we want to predict victory or defeat in a football match when we are given the number of goals a team scores.
- To do this we consider the set of results of the football matches from the Spanish league5 and we build a classification model with it.

- We first read the data file in a DataFrame and select the following columns in a new DataFrame:
  - HomeTeam,
  - AwayTeam,
  - FTHG (home team goals),
  - FTAG (away team goals), and
  - FTR (H=home win, D=draw, A=away win).

X: a *d*-dimensional vector of variables with all the scores, **x**, and

y: a binary response indicating victory or defeat, **y**.

- For that, we create two extra columns containing W the number of goals of the winning team and L the number of goals of the losing team and we concatenate these data.
- Finally, we can compute and visualize a logistic regression model to predict the discrete value (victory or defeat) using these data.

```python
from sklearn.linear_model import LogisticRegression
data = pd.read_csv('files/ch06/SP1.csv')
s = data[['HomeTeam','AwayTeam', 'FTHG', 'FTAG', 'FTR']]
def my_f1(row):
    return max(row['FTHG'], row['FTAG'])
def my_f2(row):
    return min(row['FTHG'], row['FTAG'])
s['W'] = s.apply(my_f1, axis = 1)
s['L'] = s.apply(my_f2, axis = 1)
x1 = s['W'].values
y1 = np.ones(len(x1), dtype = np.int)
x2 = s['L'].values
y2 = np.zeros(len(x2), dtype = np.int)
x = np.concatenate([x1, x2])
x = x[:, np.newaxis]
y = np.concatenate([y1, y2])
logreg = LogisticRegression()
logreg.fit(x, y)
X_test = np.linspace(-5, 10, 300)
def lr_model(x):
    return 1 / (1+np.exp(-x))
loss = lr_model(X_test*logreg.coef_ + logreg.intercept_)
       .ravel()
X_test2 = X_test[:,np.newaxis]
losspred = logreg.predict(X_test2)
plt.scatter(x.ravel(), y,
            color = 'black',
            s = 100, zorder = 20,
            alpha = 0.03)
plt.plot(X_test, loss, color = 'blue', linewidth = 3)
plt.plot(X_test, losspred, color = 'red', linewidth = 3)
```

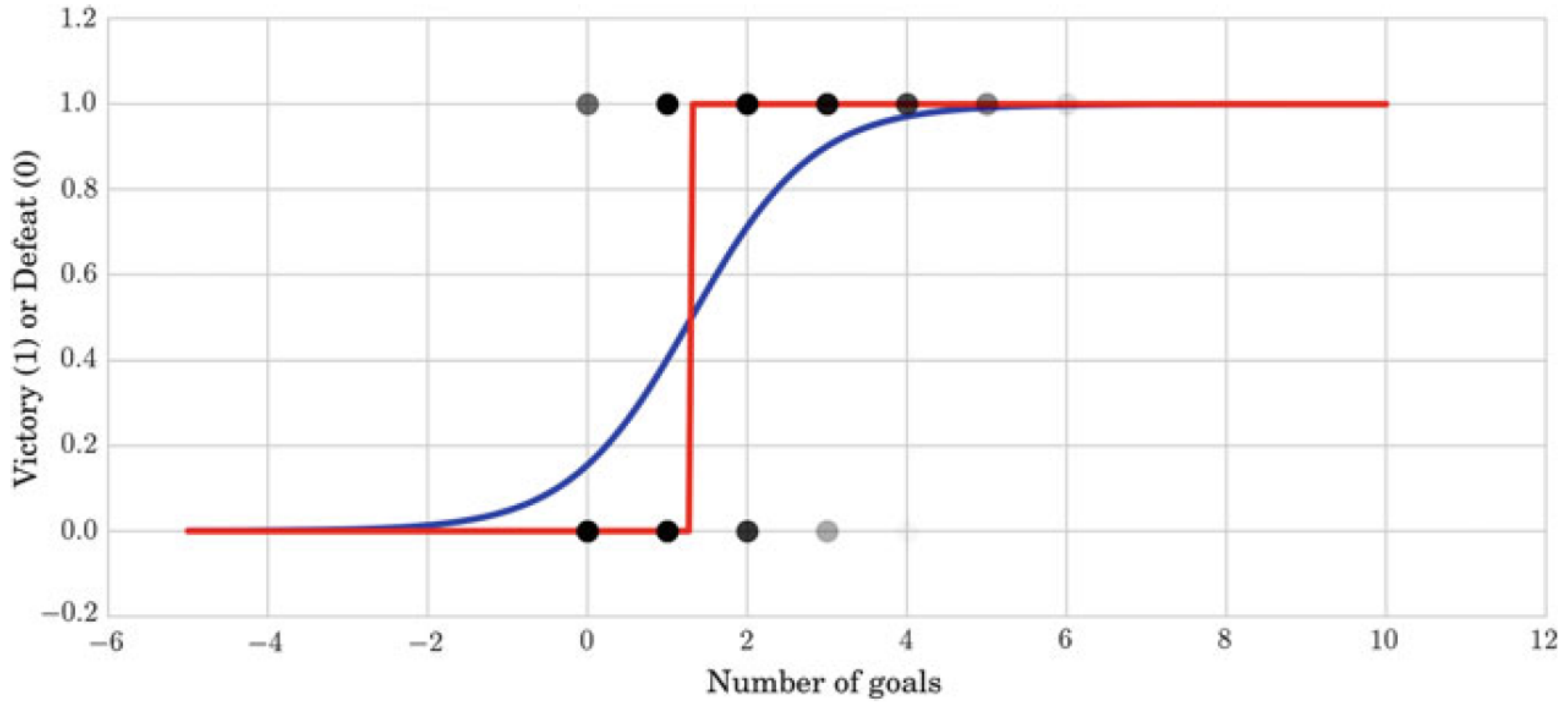


Fig. 6.12 Fitting of the logistic regression model (*blue*) and prediction of the logistic regression model (*red*) for the Spanish football league results

Synonyms

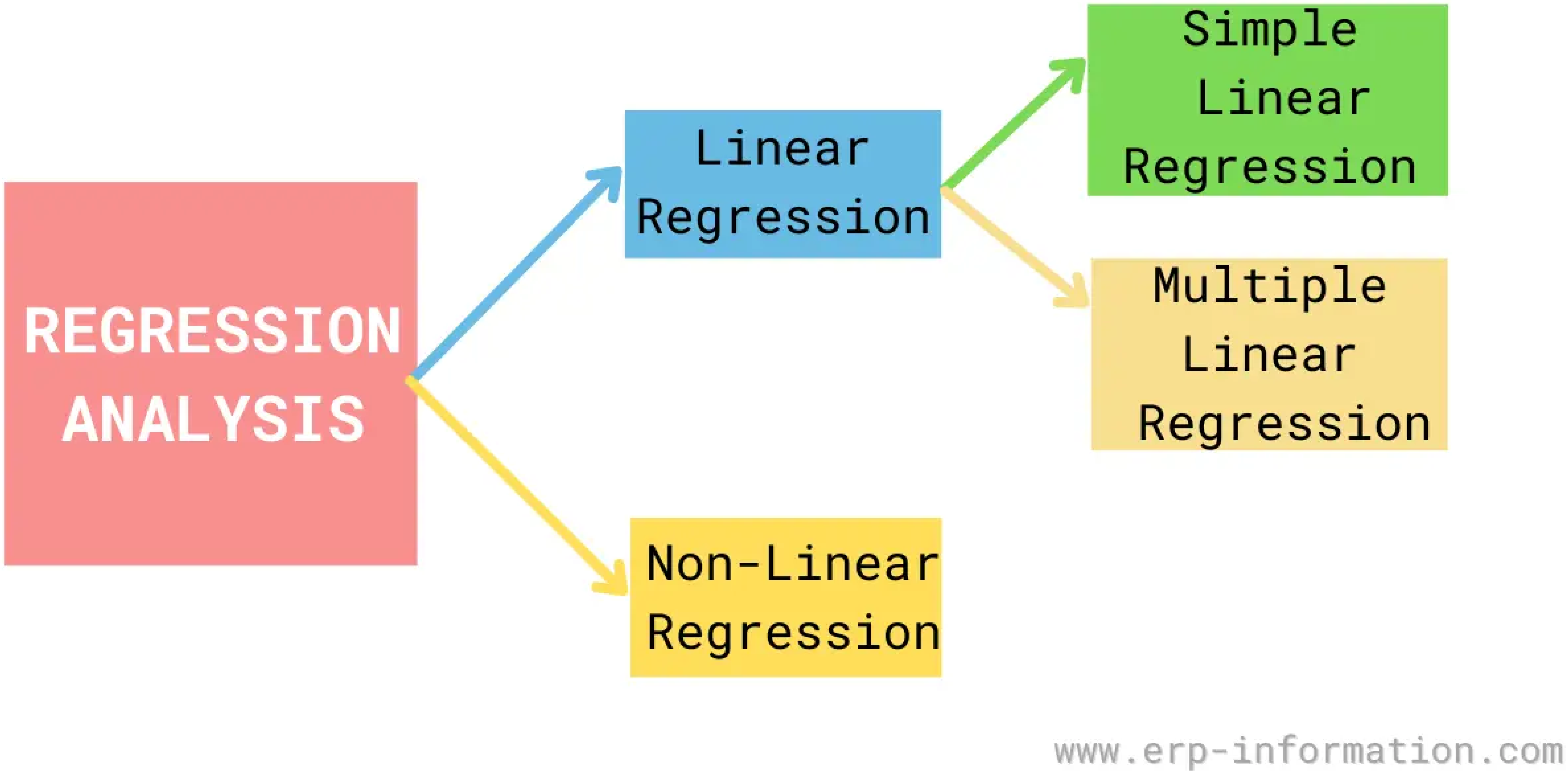
- Independent variable
  - “regressors,“
  - “controlled variable,”
  - “manipulated variable,”
  - “explanatory variable,”
  - “exposure variable,”
  - “input variable.”

- Dependent variable
  - “response variable,”
  - “regressand,”
  - “measured variable,”
  - “observed variable,”
  - “responding variable,”
  - “explained variable,”
  - “outcome variable,”
  - “experimental variable,”
  - “output variable.”

Three Practical Cases

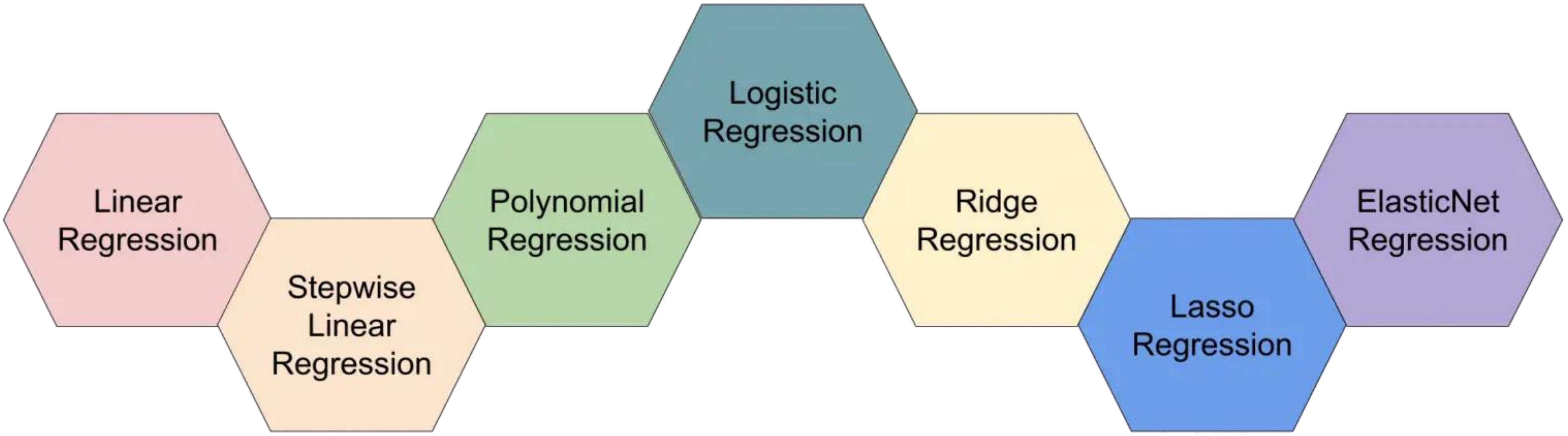
- In these practical cases, we solve different questions regarding
  - the behavior of the data,
  - the prediction of data values (continuous or discrete), and the importance of variables for the model.
- Case_01: there is a decreasing tendency in the sea ice extent over the years, and we also predicted the amount of ice for the next 20 years.
- Case_02: we predicted the price of a market given a set of attributes and distinguished which of the attributes were more important in the prediction.
  - we presented a useful way to show the correlation between pairs of variables, as well as
  - a way to plot the relationship between pairs of variables.
- Case_03: we faced the problem of predicting victory or defeat in a football match given the score of a team.
  - We posed this problem as a classification problem and solved it using a logistic regression model; and
  - we estimated the minimum number of goals a team has to score to win.

Types of regression model based on number of variables



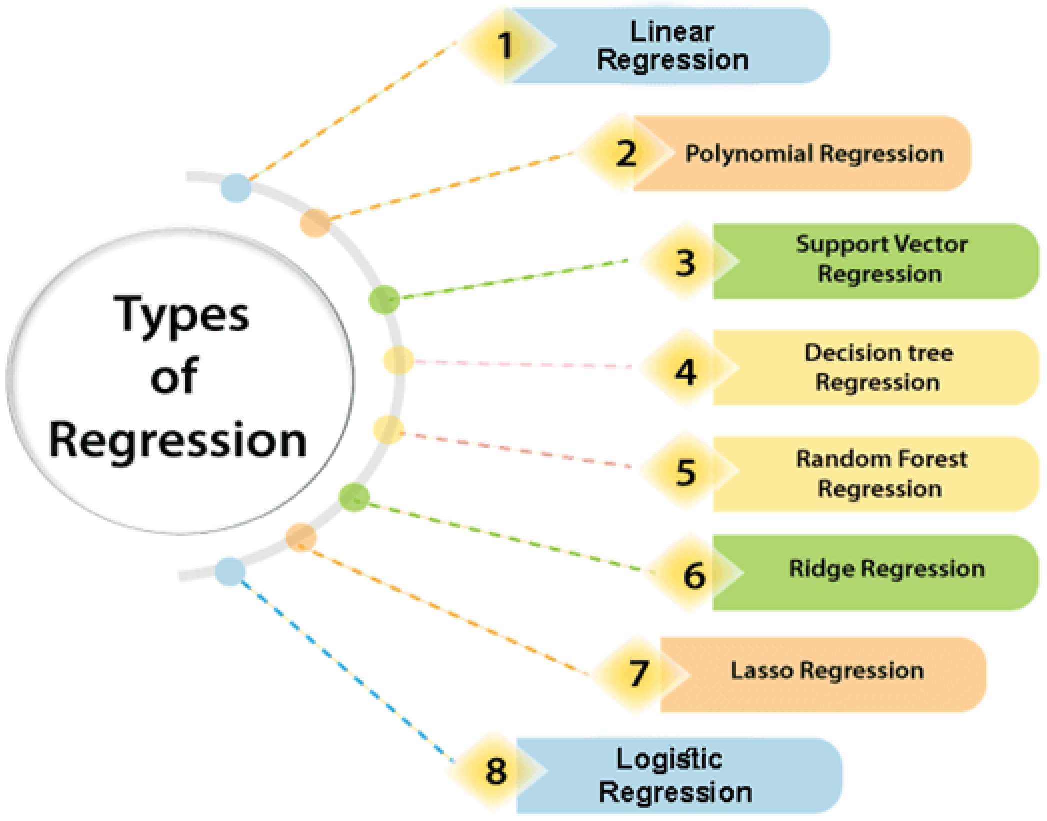
Different types of regression model

- .

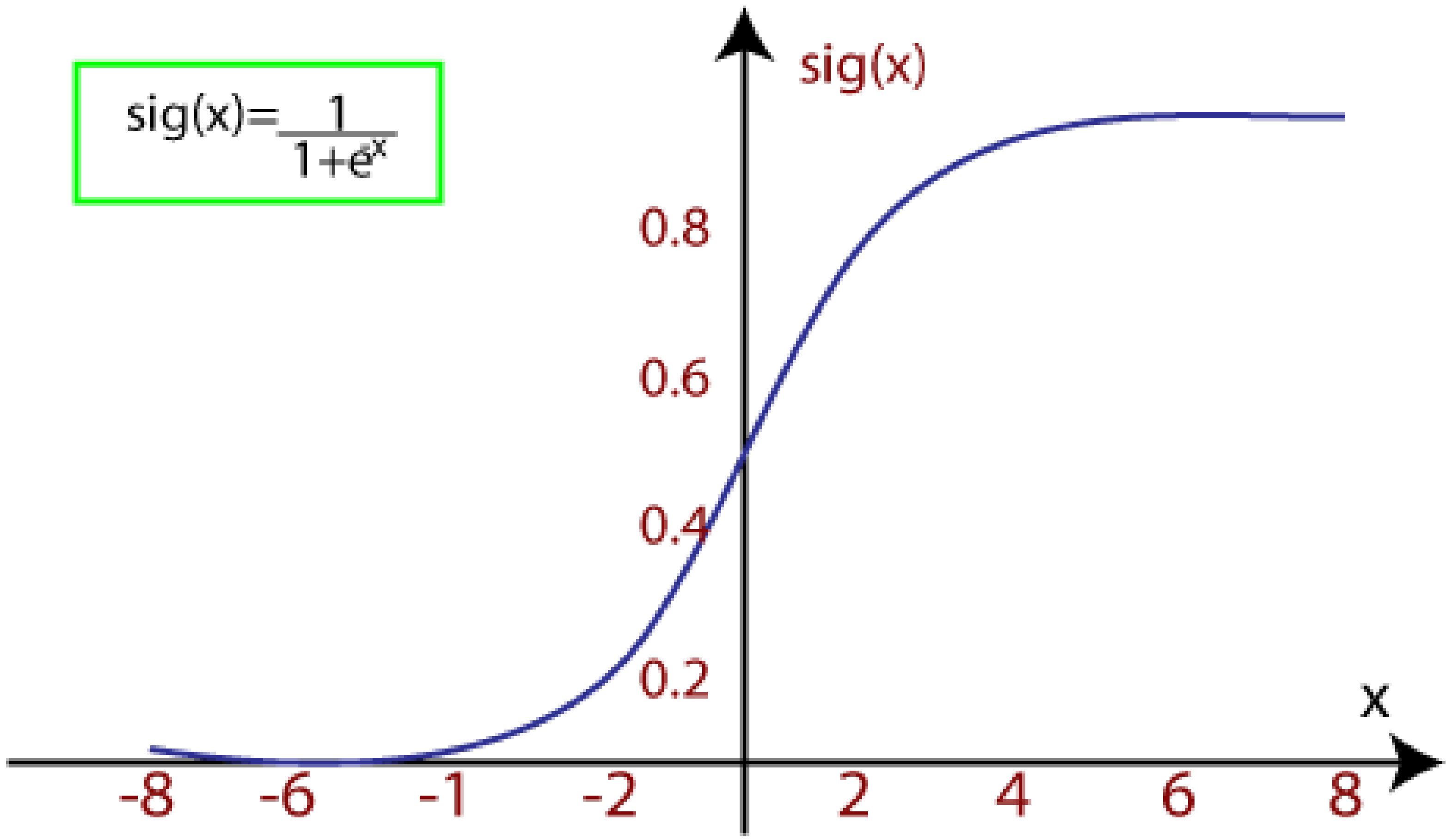


Regression algorithms

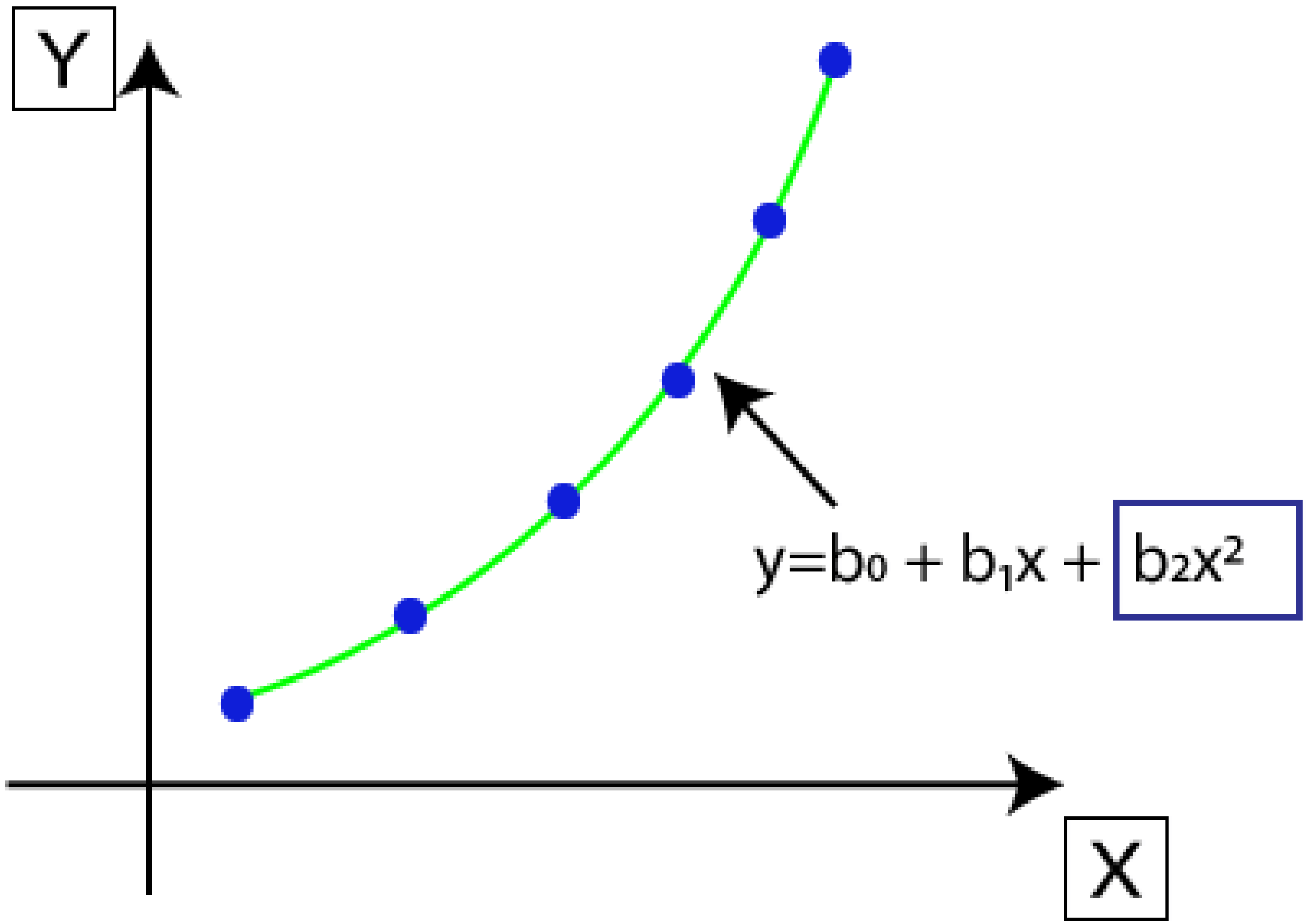
- .



Logistic Regression

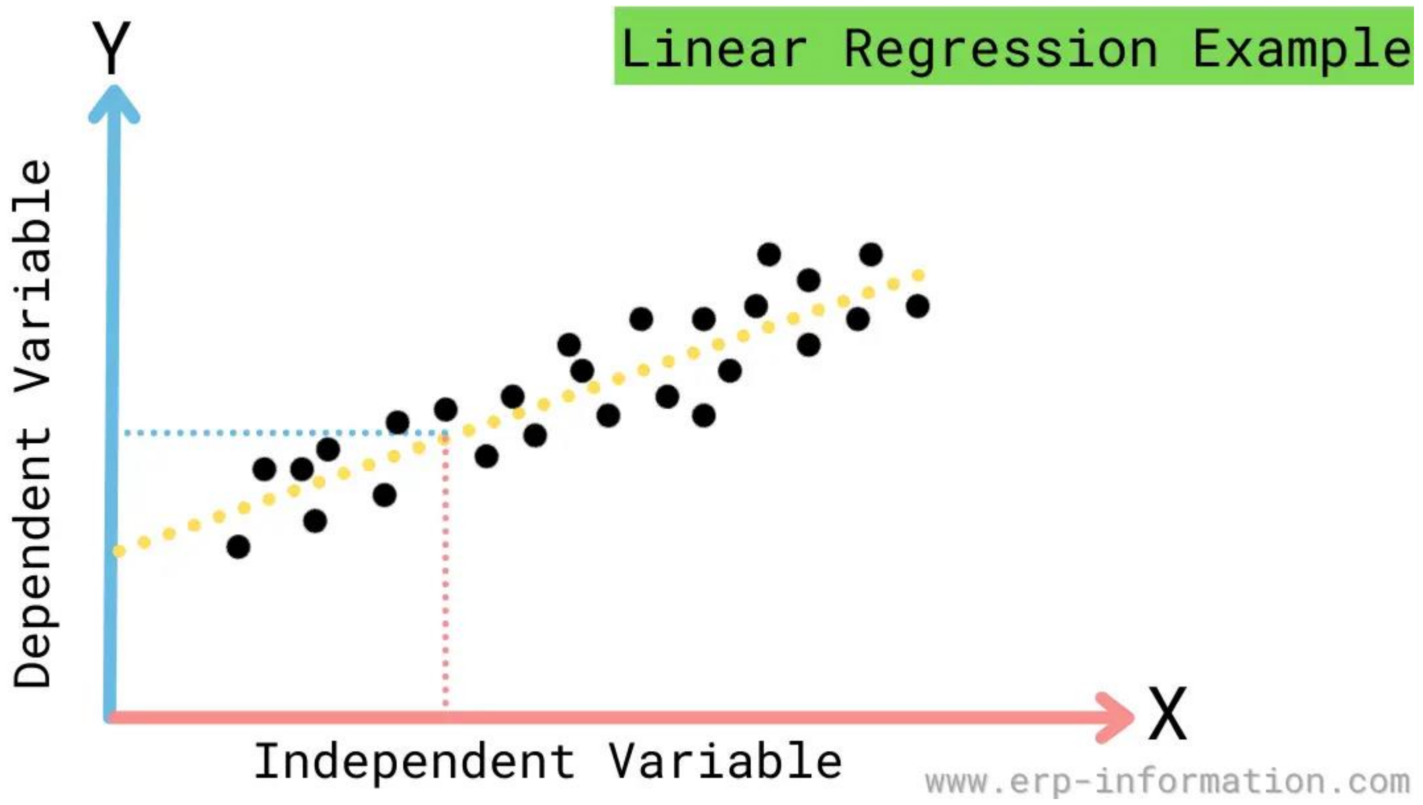


Polynomial Regression



Linear Regression

•



summary

- Regression quantifies the relationship among variable.
- There are at least one dependent and at least one independent variable.
- Effects on dependent variable due to change in independent variable are measured.

THANK YOU