Unit II - REGRESSION ANALYSIS AND FORECASTING

Unit outcome

- 2. Illustrate the regression analysis and forecasting in time series analysis

What is regression?

- Regression Analysis is a supervised learning analysis where supervised learning is the analyzing or predicting the data based on the previously available data or past data.
- Regression analysis is a statistical technique for modeling and investigating the relationships between an outcome or response variable and one or more predictor or regressor variables.

- **Purpose of regression analysis:**
- The end result of a regression analysis study is often to generate a model that can be used to forecast or predict future values of the response variable, given specified values of the predictor variables.

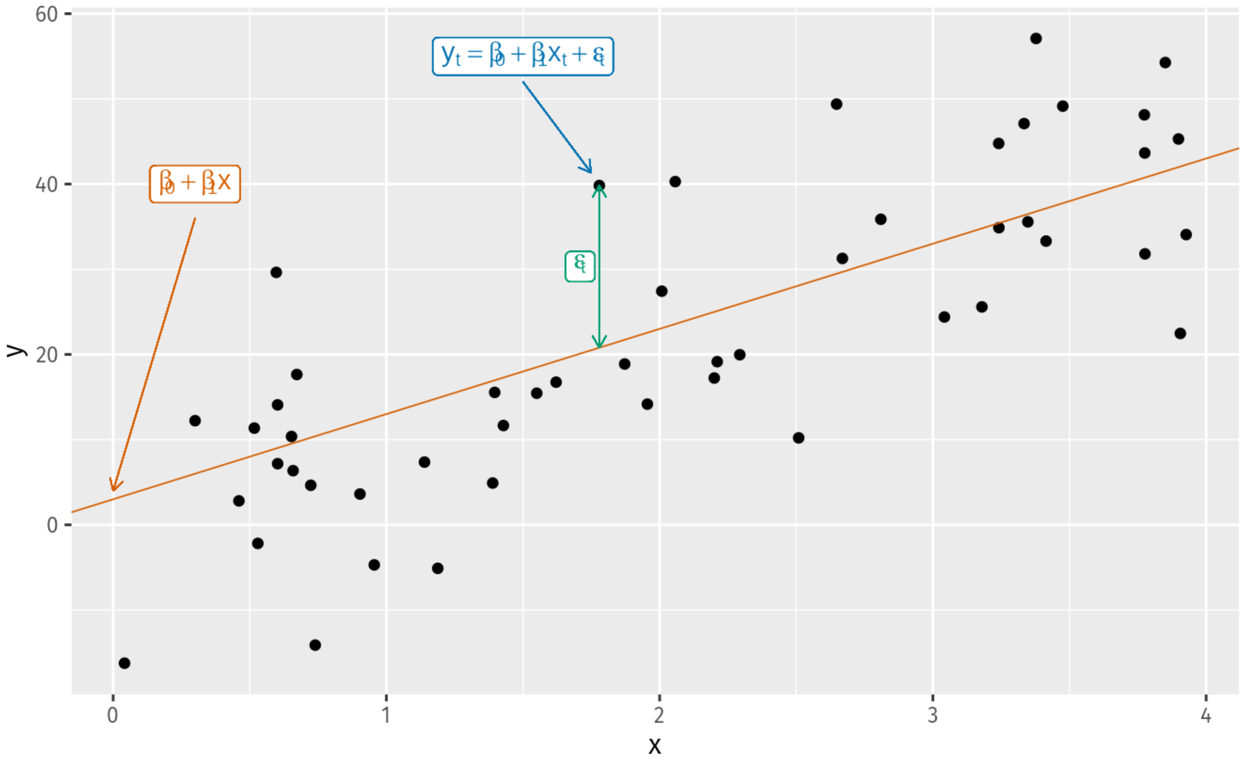
Types of regression analysis

- **Simple Linear Regression:**
- In this simple linear regression there is **only one dependent and one independent variable.**
- This linear regression model **only one predictor**. This linear regression model gives the linear **relationship between the dependent and independent variables.**
- This simple linear regression analysis is mostly used in weather forecasting, financial analysis , market analysis .
- It can be used for the **predicting outcomes , increasing the efficiency of the models , make necessary measures to prevent the mistakes of the model.**

- the regression model allows for a linear relationship between the forecast variable y and a single predictor variable x

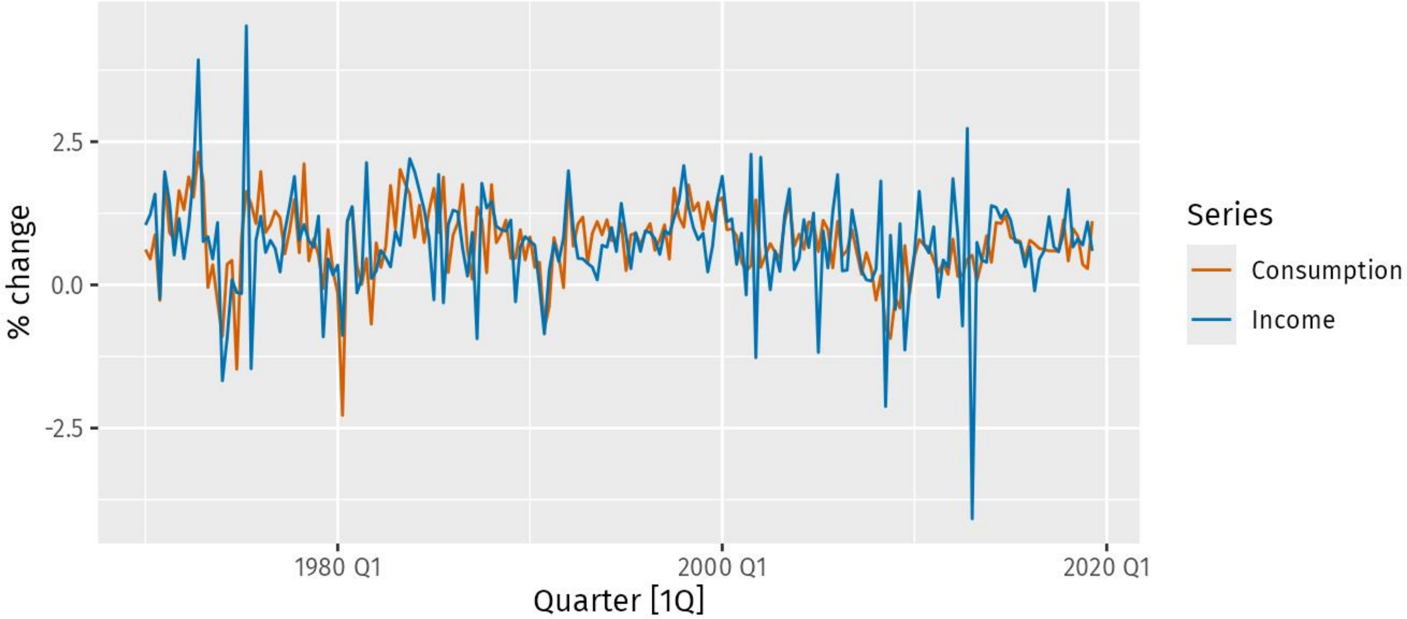
$$y_t = \beta_0 + \beta_1 x_t + \varepsilon_t.$$

The coefficients **β0** and **β1** denote the **intercept** and the **slope** of the line respectively.
The intercept **β0** represents the **predicted value** of **y** when **x=0**.
The slope **β1** represents the average predicted change in **y** resulting from a one unit increase in **x**.



- **Example: US consumption expenditure**
- Figure shows time series of quarterly percentage changes (growth rates) of real **personal consumption expenditure**, **y,** and **real personal disposable income, x,** for the US from 1970 Q1 to 2019 Q2.

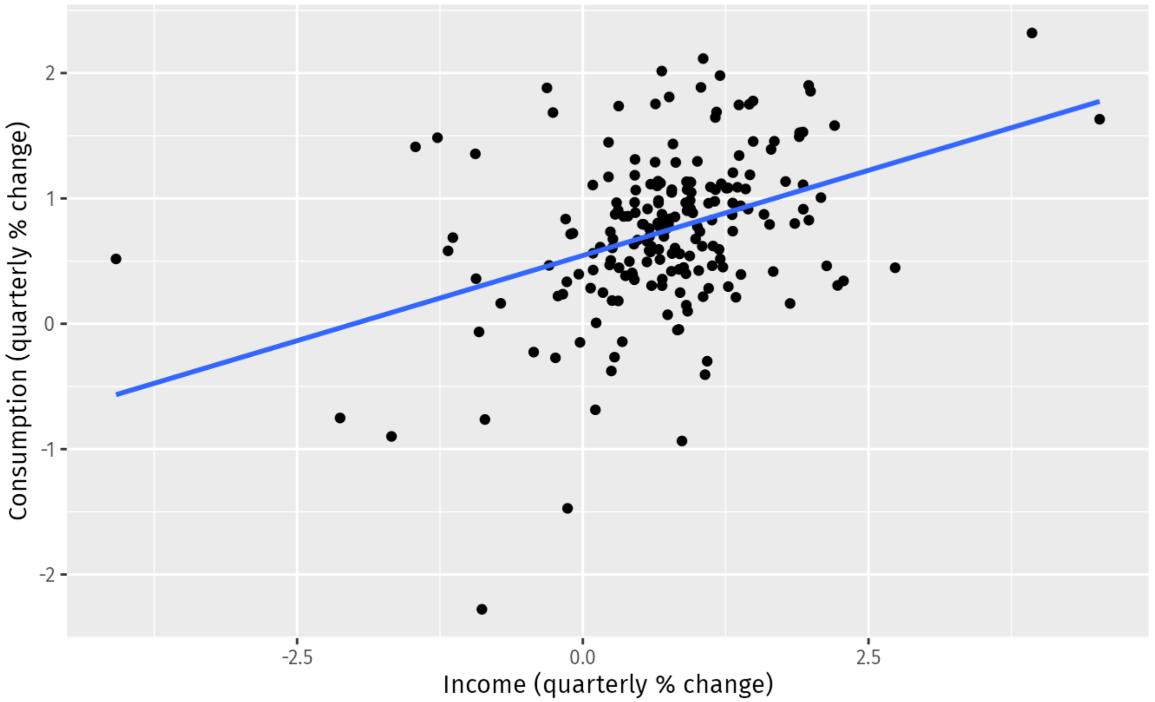
```
us_change |>
  pivot_longer(c(Consumption, Income), names_to="Series") |>
  autoplot(value) +
  labs(y = "% change")
```



- **A scatter plot of consumption changes against income changes** is shown in Figure along with the estimated regression line

$$\hat{y}_t = 0.54 + 0.27x_t.$$

```
us_change |>
  ggplot(aes(x = Income, y =
Consumption)) +
  labs(y = "Consumption (quarterly %
change)",
       x = "Income (quarterly %
change)") +
  geom_point() +
  geom_smooth(method = "lm", se =
FALSE)
```



- The mathematical equation for the **simple linear regression model** is shown below.

$$y=ax+b$$

- where y is a dependent variable
- x is a independent variable
- a, b are the regression coefficients

- Normal equations of the linear regression equation y= b+ax is.
- ∑ y = n*b + a ∑ x
- ∑ x*y = b ∑ x + a ∑ x^2
- where n is the total number of observations of the provided data/information for the above given information n=10

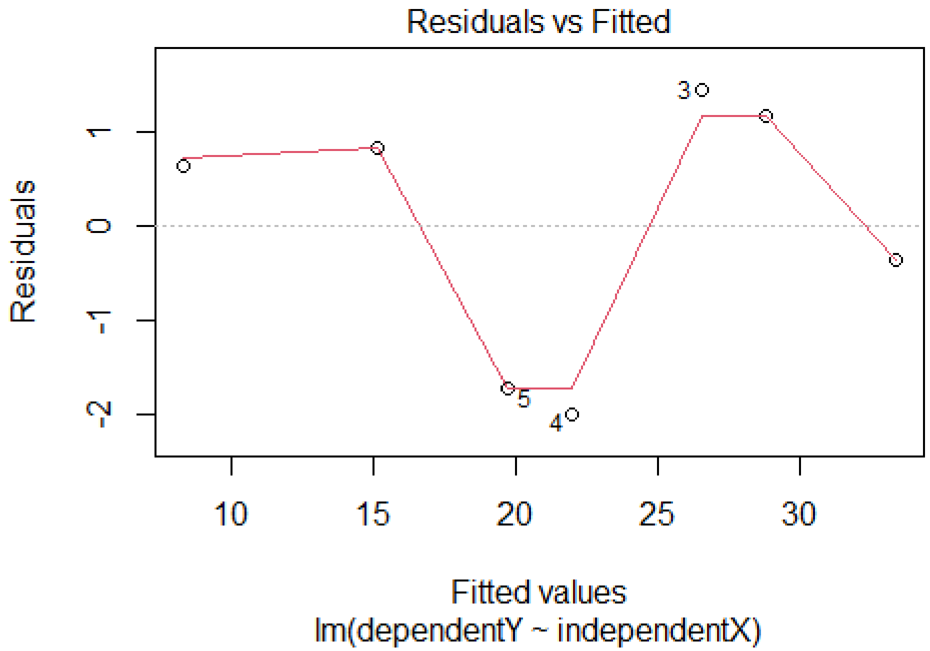
Let us now calculate the value of a and b by solving the normal equations of the linear regression curve.

x	y	x^2	xy
8	11	64	88
5	10	25	50
4	4	16	16
6	8	36	48
7	9	49	63
9	13	81	117
10	15	100	150
3	6	9	18
2	12	4	24
12	7	144	84

- From the above table
- n=10 , ∑ x = 66 , ∑ y = 95 , ∑ xy =1186 , ∑ x^2 = 528
- Now the normal equations become :
- 95 = 10*b + 66a
- 1186 = 66*b + 528a
- By solving the above two euations we get a = 6.05 and b = -30.429
- The linear regression equation is y = -30.429 + 6.05 x.
- Let us now discuss the implementation of the linear regression curve in R

- R Code

```
#Storing the independent value X
independentX<-c(5,7,8,10,11,13,16)
#soring the dependent value
dependentY<-c(33,30,28,20,18,16,9)
#performing the regression analysis using the function lm
linearregression<-lm(dependentY,independentX)
#printing the summary of the result
summary(linearregression)
#ploting the model
plot(linearregression)
plot(independentX,dependentY)
```

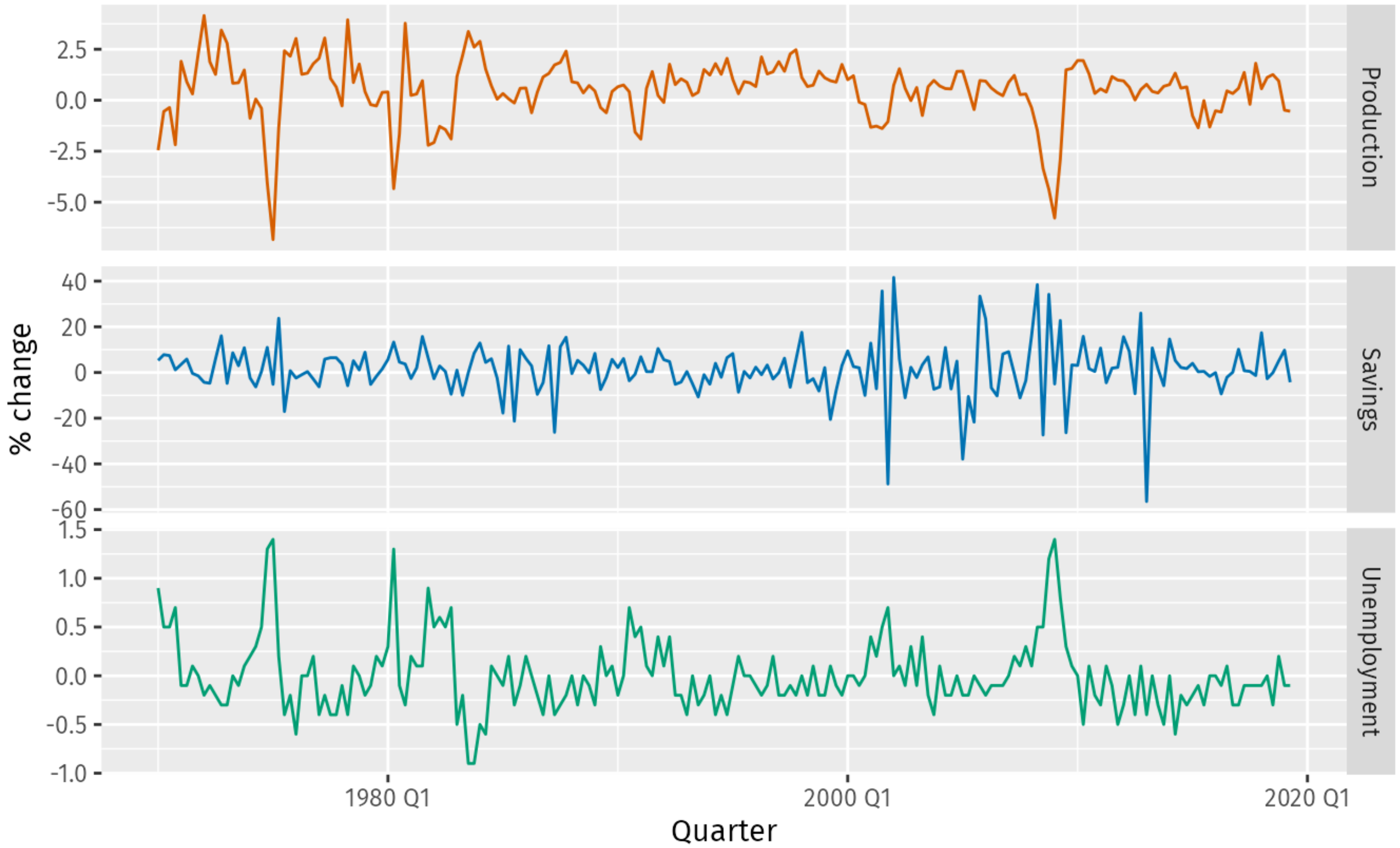


- **Multiple Linear Regression**
- Multiple linear regression analysis gives the relationship between the two or more independent varibales and a dependent variable.
- Multiple linear regression can be represented as the hyper plane in multidimensional space . It is also a linear type regression analysis .
- When there are two or more predictor variables, the model is called a **multiple regression model**. The general form of a multiple regression model is

$$y_t = \beta_0 + \beta_1 x_{1,t} + \beta_2 x_{2,t} + \cdots + \beta_k x_{k,t} + \varepsilon_t,$$

- where y is the variable to be forecast and x1,…,xk are the k predictor variables.
- Each of the predictor variables must be numerical.
- The coefficients β1,…,βk measure the effect of each predictor after taking into account the effects of all the other predictors in the model.
- Thus, the coefficients measure the *marginal effects* of the predictor variables.

```
us_change |>
  select(-Consumption, -Income) |>
  pivot_longer(-Quarter) |>
  ggplot(aes(Quarter, value, colour = name)) +
  geom_line() +
  facet_grid(name ~ ., scales = "free_y") +
  guides(colour = "none") +   labs(y="% change")
```



x1	1	2	3	4	5
x2	8	6	4	2	10
y	3	7	5	9	11

x1	x2	y	x1^2	x2^2	x1*x2	x1*y	x2*y
1	8	3	1	64	8	3	24
2	6	7	4	36	12	14	42
3	4	5	9	16	12	15	20
4	2	9	16	4	8	36	18
5	10	11	25	100	50	55	110

From the above table

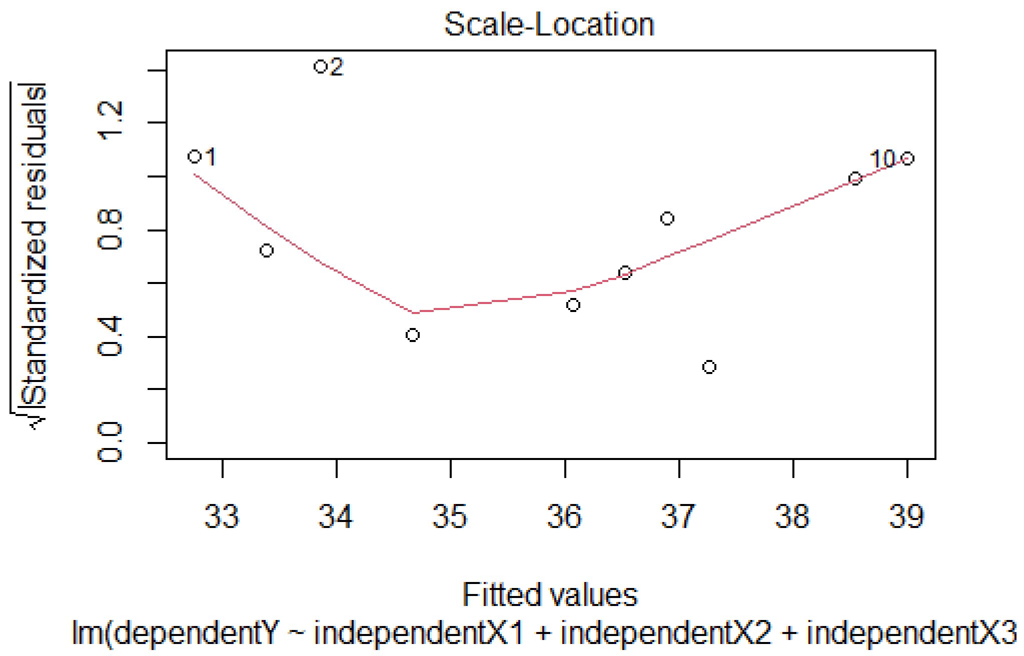
- n=5 , ∑ x1 = 15 , ∑ x2 = 30 , ∑ y = 35 , ∑ x1^2 = 55 , ∑ x2^2 = 220 , ∑ x1*x2 = 90 , ∑ x1*y = 123 , ∑ x2 *y = 214
- Then the normal equations become:

35 = 5b + 15a0 + 30a1

- 123 = 15b + 55a0 + 90a1
- 214 = 30b + 90a0 + 220a1
- By solving the above three normal equations we get the values of a0 , a1 and b .
- a0 = 1.8 , a1 = 0.1 , b = 1.666
- The multilinear regression analysis curve can be fit as y = 1.666 + 1.8*x1 + 0.1 * x2 .

R Code

```
#in this we are performing the multi linear regression using the three independent
#storing the three independent variables
independentX1<-c(8,10,15,19,20,11,16,13,6,18)
independentX2<-c(22,26,24,32,38,39,29,13,15,25)
independentX3<-c(28,26,24,22,29,25,27,23,20,21)

#storing the dependent variable y
dependentY<-c(43,12,45,48,33,37,39,38,36,28)

#performing the multilinear regression analysis
multilinear<-lm(dependentY~independentX1+independentX2+independentX3)

#printing the summary of the result
summary(multilinear)
plot(multilinear)
```



- **Polynomial Regression**
- Polynomial regression analysis is a non linear regression analysis .
- Polynomial regression analysis helps for the flexible curve fitting of the data , involves the fitting of polynomial equation of the data.
- Polynomial regression analysis is the extension of the simple linear regression analysis by adding the extra independent variables obtained by raising the power .
- *y=a0+a1x+a2x^2+………..+anx^n*
- *where y is dependent variable*
- *x is independent variable*
- *a0,a1,a2 are the coefficeients of independent variable.*

- **Exponential Regression**
- Expenential regression is a non linear type of regression .
- Exponential regression can be expressed in two ways .
- Exponential regression can be used in finance , biology , physics etc fields . Let us look the mathematical expression for the exponential regression with example.

- y=ae^(bx)

- where y is dependent variable
- x is independent variable
- a , b are the regression coefficients.

What is the Least Squares Regression method and why use it?

- Least squares is a method to apply linear regression.
- It helps us predict results based on an existing set of data as well as clear anomalies in our data.
- Anomalies are values that are too good, or bad, to be true or that represent rare cases.

we have a collection of observations but we do not know the values of the coefficients β0,β1,…,βk. These need to be estimated from the data.

The least squares principle provides a way of choosing the coefficients effectively by minimizing the sum of the squared errors.
That is, we choose the values of β0,β1,…,βk that minimize

$$\sum_{t=1}^{T} \varepsilon_t^2 = \sum_{t=1}^{T} (y_t - \beta_0 - \beta_1 x_{1,t} - \beta_2 x_{2,t} - \cdots - \beta_k x_{k,t})^2.$$

- This is called **least squares** estimation because it gives the least value for the sum of squared errors.
- Finding the best estimates of the coefficients is often called “fitting” the model to the data, or sometimes “learning” or “training” the model.

Example: US consumption expenditure

A multiple linear regression model for US consumption is

$$y_t = \beta_0 + \beta_1 x_{1,t} + \beta_2 x_{2,t} + \beta_3 x_{3,t} + \beta_4 x_{4,t} + \varepsilon_t,$$

- where y is the percentage change in real personal consumption expenditure,
- x1 is the percentage change in real personal disposable income,
- x2 is the percentage change in industrial production,
- x3 is the percentage change in personal savings and
- x4 is the change in the unemployment rate.

- The TSLM() function fits a linear regression model to time series data.
- It is similar to the lm() function which is widely used for linear models, but TSLM() provides additional facilities for handling time series.

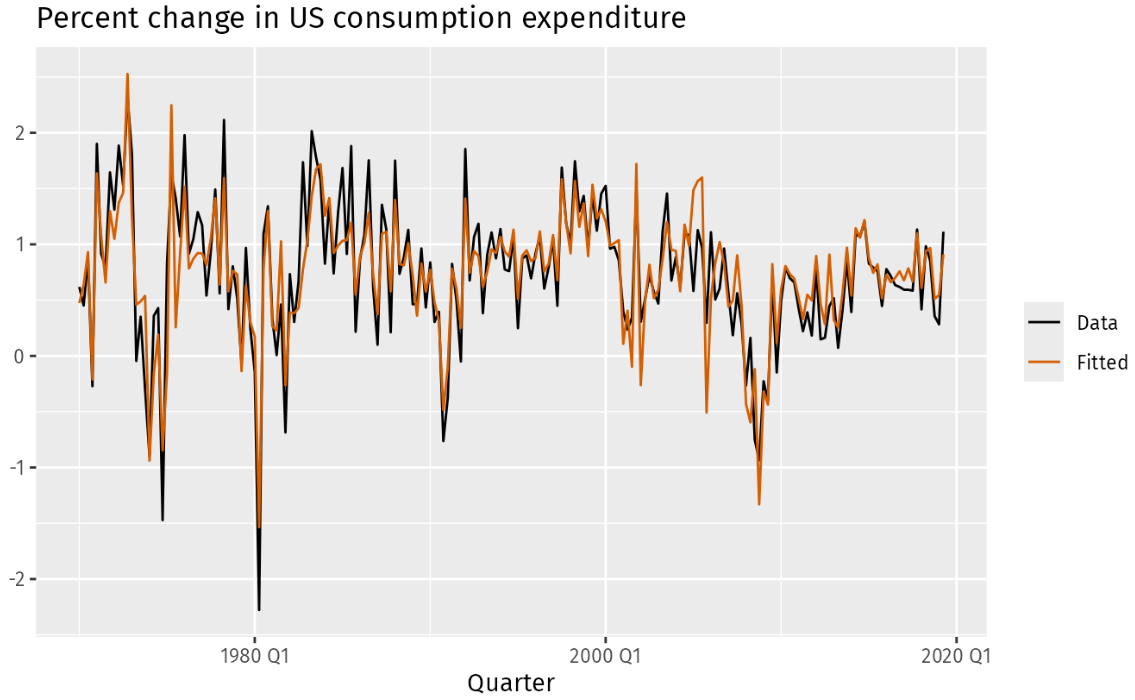
```
fit_consMR <- us_change |>
  model(tslm = TSLM(Consumption ~ Income + Production +
                                  Unemployment + Savings))
report(fit_consMR)
#> Series: Consumption
#> Model: TSLM
#>
#> Residuals:
#>     Min      1Q  Median      3Q     Max
#> -0.9055 -0.1582 -0.0361  0.1362  1.1547
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept)   0.25311    0.03447    7.34  5.7e-12 ***
#> Income       0.74058    0.04012   18.46  < 2e-16 ***
#> Production    0.04717    0.02314    2.04    0.043 *
#> Unemployment -0.17469    0.09551   -1.83    0.069 .
#> Savings      -0.05289    0.00292  -18.09  < 2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 0.31 on 193 degrees of freedom
#> Multiple R-squared: 0.768,   Adjusted R-squared: 0.763
#> F-statistic:  160 on 4 and 193 DF, p-value: <2e-16
```

- **Fitted values**
- Predictions of y can be obtained by using the estimated coefficients in the regression equation and setting the error term to zero. In general we write,

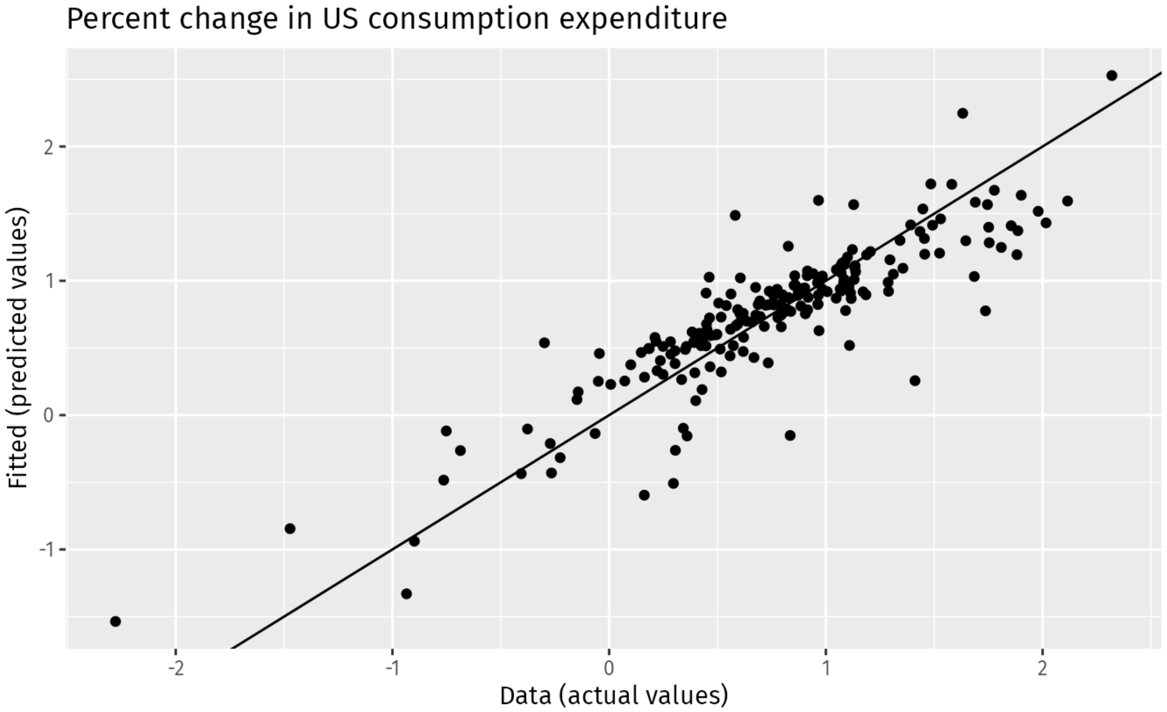
$$\hat{y}_t = \hat{\beta}_0 + \hat{\beta}_1 x_{1,t} + \hat{\beta}_2 x_{2,t} + \cdots + \hat{\beta}_k x_{k,t}.$$

- Plugging in the values of x1,t,…,xk,t for t=1,…,T returns predictions of yt within the training set, referred to as *fitted values*.

```
augment(fit_consMR) |>
  ggplot(aes(x = Quarter)) +
  geom_line(aes(y = Consumption, colour = "Data")) +
  geom_line(aes(y = .fitted, colour = "Fitted")) +
  labs(y = NULL,
    title = "Percent change in US consumption expenditure"
  ) +

scale_colour_manual(values=c(Data="black",Fitted="#D55E00"
)) +
  guides(colour = guide_legend(title = NULL))
```



```
augment(fit_consMR) |>
  ggplot(aes(x = Consumption, y = .fitted)) +
  geom_point() +
  labs(
    y = "Fitted (predicted values)",
    x = "Data (actual values)",
    title = "Percent change in US consumption expenditure"
  ) +
  geom_abline(intercept = 0, slope = 1)
```



- **Standard error of the regression**
- Another measure of how well the model has fitted the data is the standard deviation of the residuals, which is often known as the “residual standard error”
- The standard error will be used when generating prediction intervals,

$$\hat{\sigma}_e = \sqrt{\frac{1}{T-k-1}\sum_{t=1}^{T} e_t^2},$$

- where k is the number of predictors in the model.
- Notice that we divide by T−k−1 because we have estimated k+1 parameters (the intercept and a coefficient for each predictor variable) in computing the residuals.
- The standard error is related to the size of the average error that the model produces.

Unit II - REGRESSION ANALYSIS AND FORECASTING

Unit outcome

- 2. Illustrate the regression analysis and forecasting in time series analysis

Statistical Inference in Linear Regression

- In this section, we describe several important hypothesis-testing procedures and a confidence interval estimation procedure.
- These procedures require that the errors $\boldsymbol{\varepsilon i}$ in the model are normally and independently distributed with **mean zero** and **variance $\boldsymbol{\sigma}$2**, abbreviated NID(0, σ2).
- **What is the significance test for linear regression?**
- Significance tests for linear regression are **used to determine if the relationship between the dependent variable and one or more independent variables** is statistically significant.

Test for Significance of Regression

- The test for significance of regression is a test to determine whether there is a linear relationship between the response **variable y** and a subset of the predictor or regressor variables **x1, x2,…, xk**.
- The appropriate hypotheses are
- $H0 : \beta 1 = \beta 2 = \cdots = \beta k = 0$
- $H1$: at least one $\beta j \neq 0$ Eq.
- Rejection of the null hypothesis *H*0 in Eq. implies that at least one of the predictor variables *x*1, *x*2,…, *xk* contributes significantly to the model

- **INTRODUCTION TO HYPOTHESIS TESTING**
- A statistician will make a decision about whether these claims are true or false. This process is called **hypothesis testing**.
- A hypothesis test involves collecting data from a sample and evaluating the data.

- The test procedure involves an analysis of variance partitioning of the total **sum of squares**

$$SS_{\mathrm{T}} = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

- into a sum of squares due to the **model** (or to **regression**) and a sum of squares due to **residual** (or **error**), say,

$$SS\mathrm{T} = SS\mathrm{R} + SS\mathrm{E}$$

- Now if the null hypothesis in Eq.($H1$: at least one $\beta j \neq 0$) is true and the model errors are normally and independently distributed with constant variance as assumed, then the test statistic for significance of regression is

$$F_0 = \frac{SS_{\mathrm{R}}/k}{SS_{\mathrm{E}}/(n-p)}$$

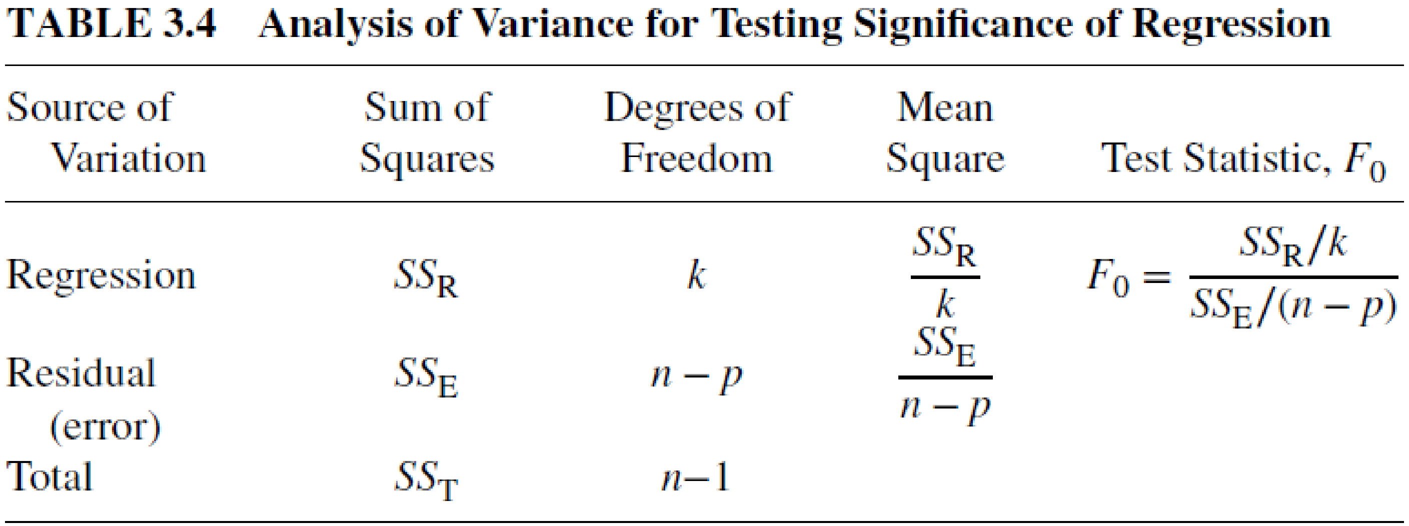
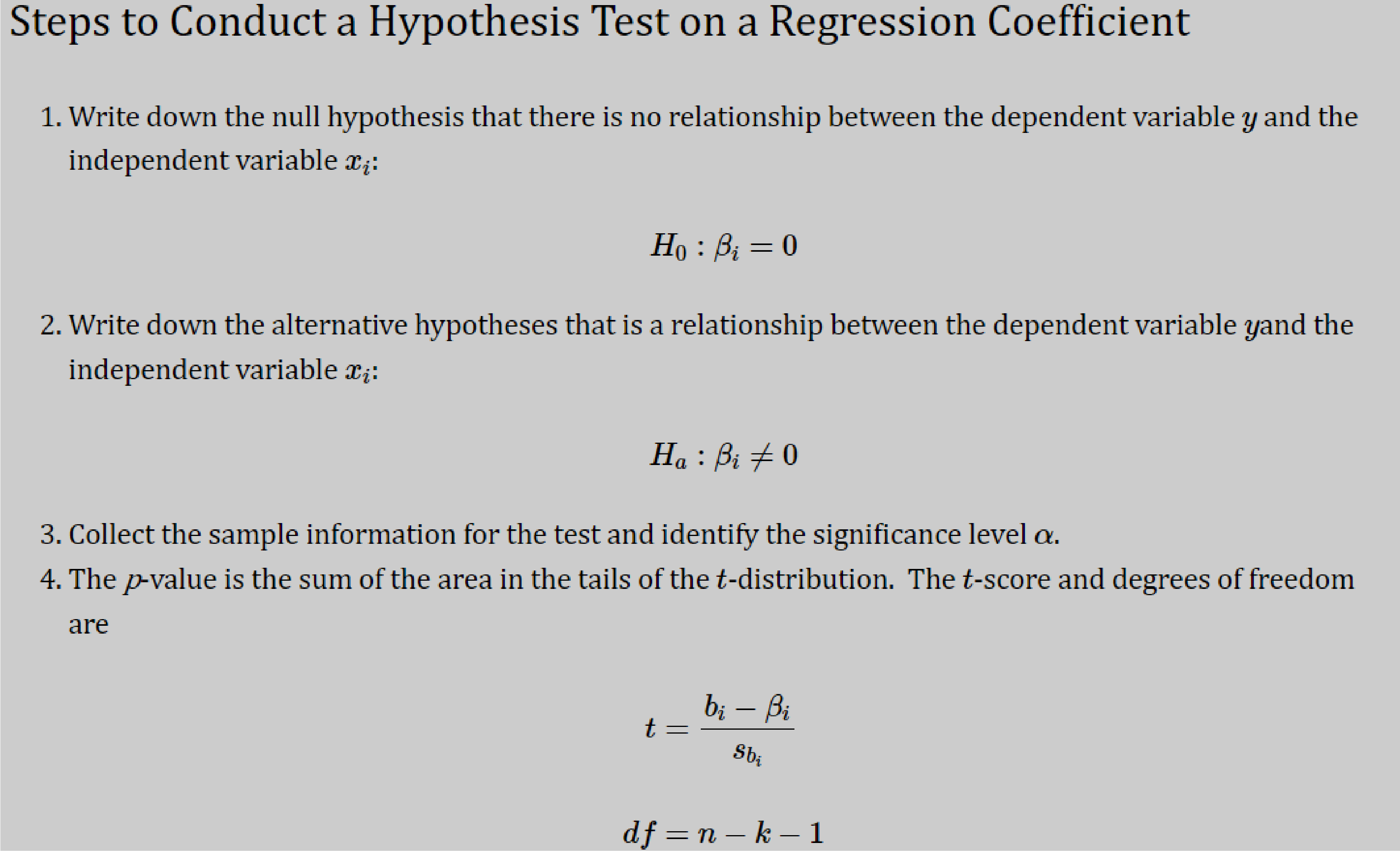


TABLE 3.4 Analysis of Variance for Testing Significance of Regression

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Square	Test Statistic, F_0
Regression	SS_R	k	$\frac{SS_R}{k}$	$F_0 = \frac{SS_R/k}{SS_E/(n-p)}$
Residual (error)	SS_E	$n-p$	$\frac{SS_E}{n-p}$	
Total	SS_T	$n-1$		

- **Calculate the Sum of Squares Total (SST):** SST represents the total variation in the dependent variable. It's calculated as the sum of the squared differences between each data point and the mean of the dependent variable.
- **Calculate the Sum of Squares Regression (SSR):** SSR represents the variation in the dependent variable that your model explains. It's calculated as the sum of the squared differences between the predicted values from your model and the mean of the dependent variable.
- **Calculate the Residual Sum of Squares (SSE):** SSE represents the unexplained variation or error in your model. It's calculated as the sum of the squared differences between the actual data points and the predicted values from your model.
- **Compute R-squared (R^2):** R-squared is calculated as the ratio of SSR to SST. In other words, it tells you what proportion of the total variation in the dependent variable is explained by your model. The formula for R-squared is: $R^2 = 1 - (SSE / SST)$
- **Degrees of Freedom** Where **n** is the sample size:

- ***Tests on Individual Regression Coefficients*** We are frequently interested in testing hypotheses on the individual regression coefficients.
- These tests would be useful in determining the value or contribution of each predictor variable in the regression model.



Steps to Conduct a Hypothesis Test on a Regression Coefficient

1. Write down the null hypothesis that there is no relationship between the dependent variable y and the independent variable x_i:

$$H_0 : \beta_i = 0$$

2. Write down the alternative hypotheses that is a relationship between the dependent variable $y$and the independent variable x_i:

$$H_a : \beta_i \neq 0$$

3. Collect the sample information for the test and identify the significance level α.
4. The p-value is the sum of the area in the tails of the t-distribution. The t-score and degrees of freedom are

$$t = \frac{b_i - \beta_i}{s_{b_i}}$$

$$df = n - k - 1$$

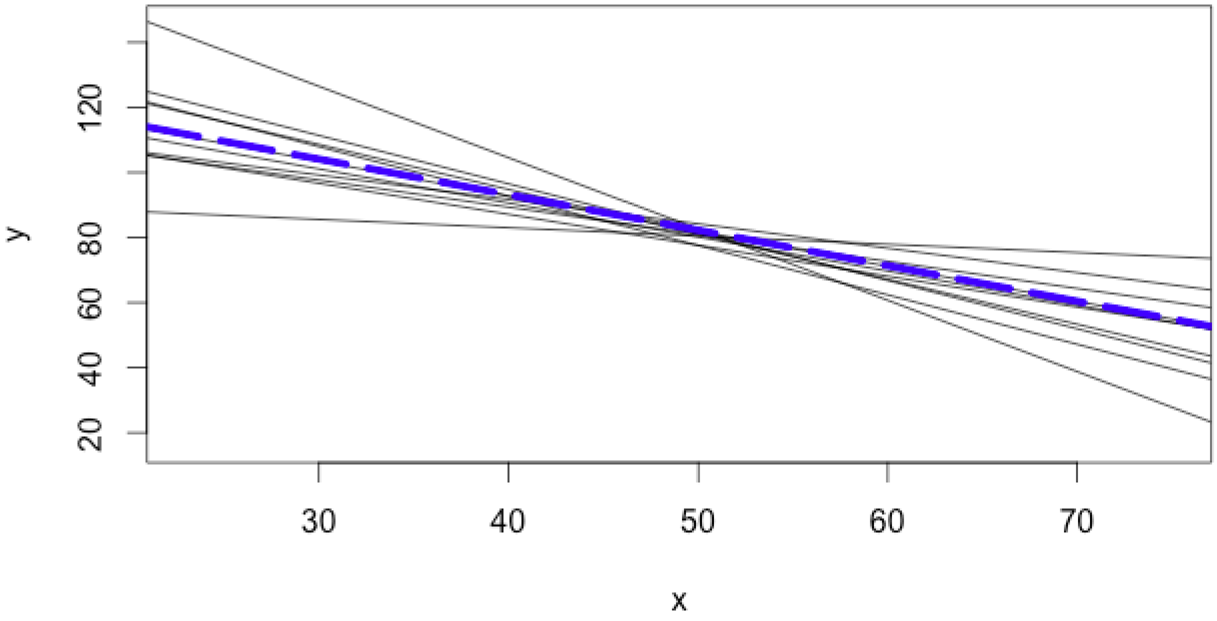
5. Compare the p-value to the significance level and state the outcome of the test:

    - If p-value$\leq \alpha$, reject H_0 in favour of H_a.
        - The results of the sample data are significant. There is sufficient evidence to conclude that the null hypothesis H_0 is an incorrect belief and that the alternative hypothesis H_a is most likely correct.
    - If p-value$> \alpha$, do not reject H_0.
        - The results of the sample data are not significant. There is not sufficient evidence to conclude that the alternative hypothesis H_a may be correct.

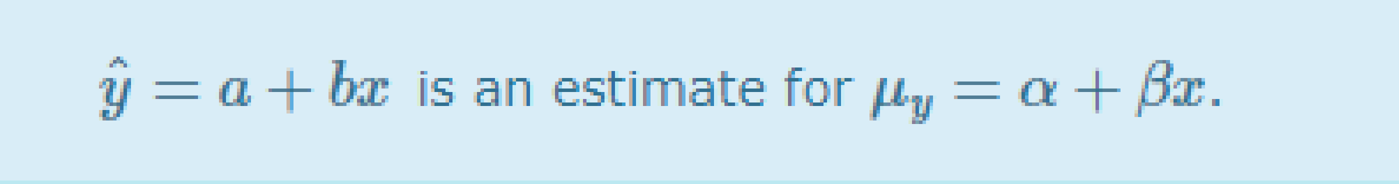
6. Write down a concluding sentence specific to the context of the question.

The required t-score and p-value for the test can be found on the regression summary table, which we learned how to generate in Excel in a previous section.

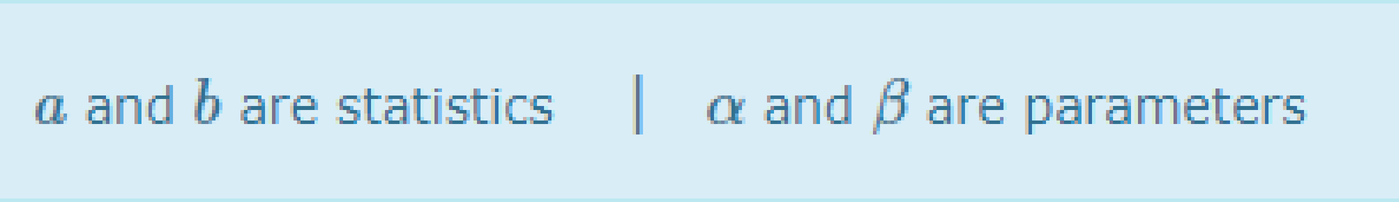
- The image below shows these least square regression lines generated from different random samples as solid **grey lines**, and
- the true population regression line as a dotted **blue line.**



- The least squares regression line (yˆ=a+bx), which runs through a sample of data points, is really an estimate of the true population regression line (μy=α+βx).



$\hat{y} = a + bx$ is an estimate for $\mu_y = \alpha + \beta x$.

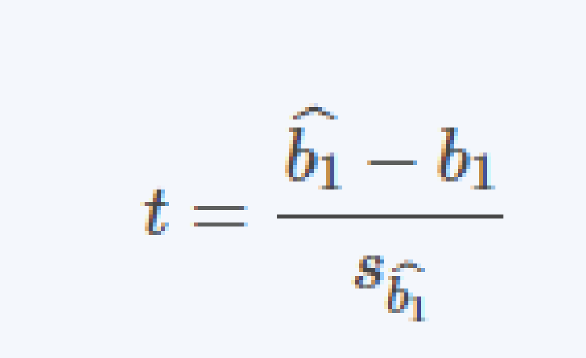


a and b are statistics | α and β are parameters

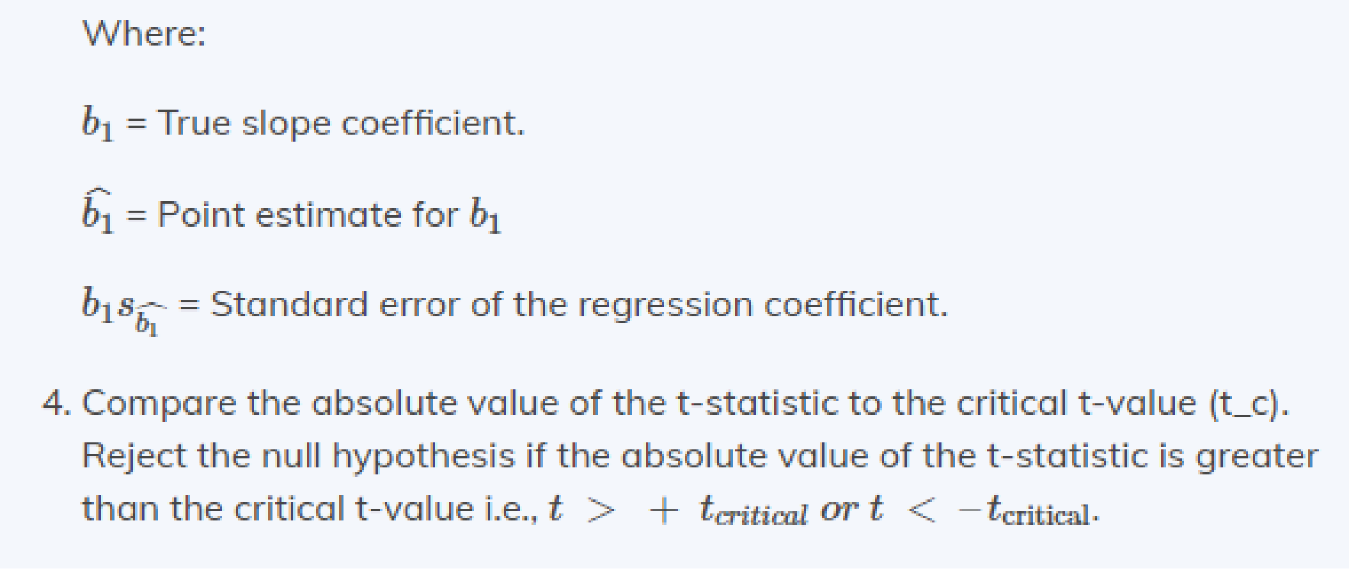
- We conduct *statistical inference* in linear regression when we find a sample slope and
- then use it to make a confidence interval or perform a hypothesis test about the true population slope.

Hypothesis testing approach

- **The t-test Approach**
- The following are the steps followed in the performance of the t-test:
  - Set the significance level for the test.
  - Formulate the null and the alternative hypotheses.
  - Calculate the t-statistic using the formula below



$$t = \frac{\widehat{b_1} - b_1}{s_{\widehat{b_1}}}$$



Where:

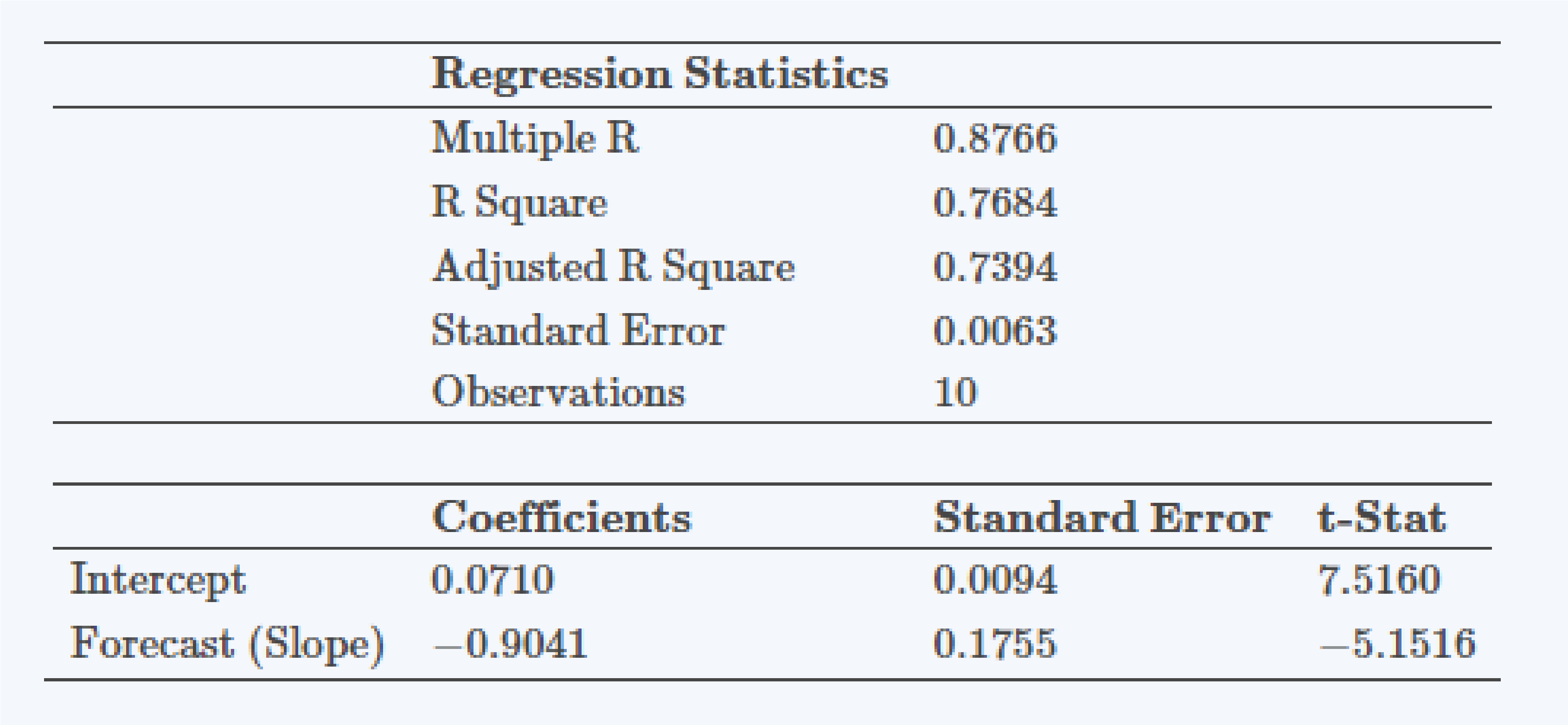
b_1 = True slope coefficient.

$\widehat{b_1}$ = Point estimate for b_1

$b_1 s_{\widehat{b_1}}$ = Standard error of the regression coefficient.

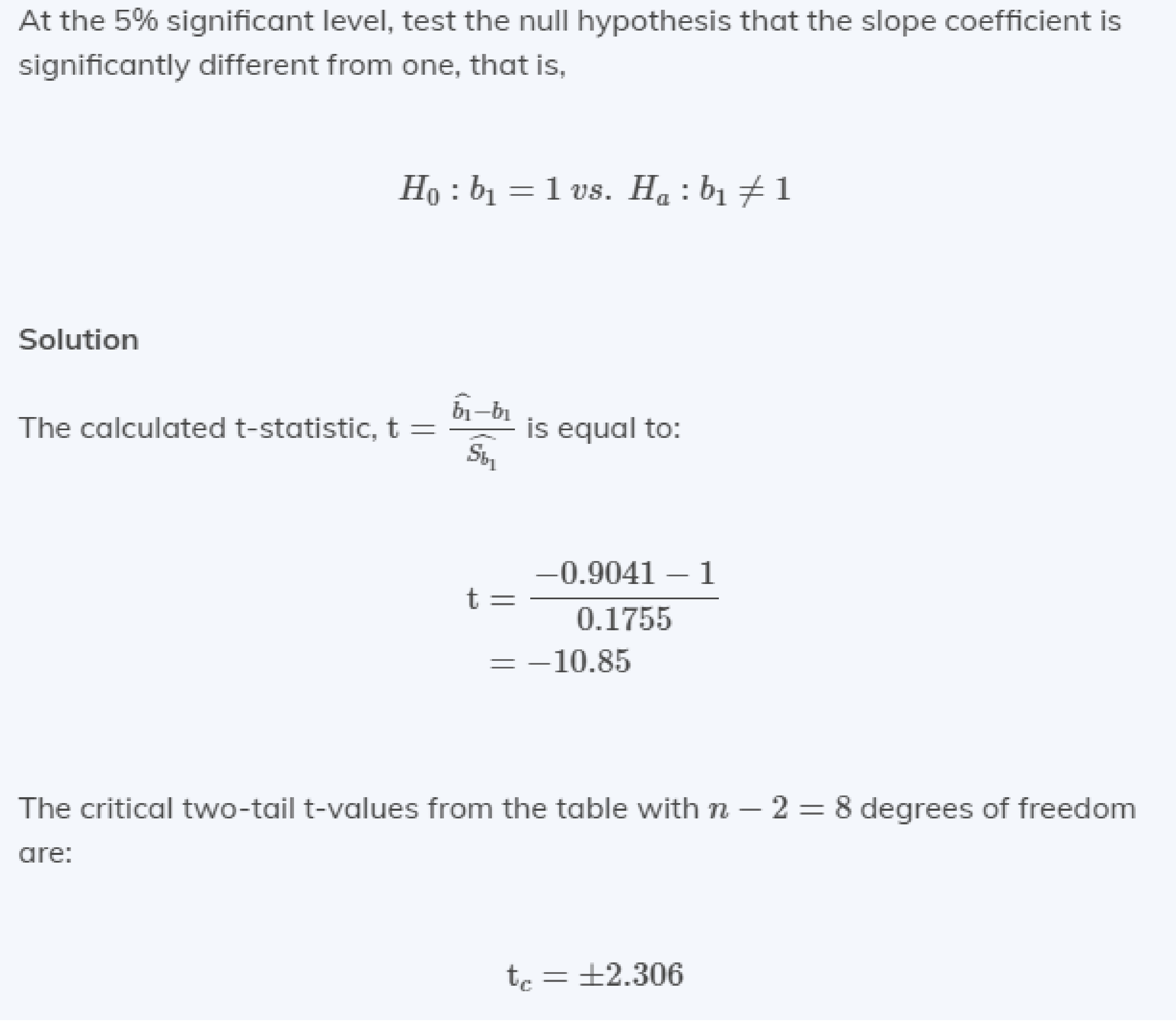
4. Compare the absolute value of the t-statistic to the critical t-value (t_c). Reject the null hypothesis if the absolute value of the t-statistic is greater than the critical t-value i.e., $t\ >\ +\ t_{critical}\ or\ t\ <\ -t_{\text{critical}}$.

Example: Hypothesis Testing of the Significance of Regression Coefficients



Regression Statistics	
Multiple R	0.8766
R Square	0.7684
Adjusted R Square	0.7394
Standard Error	0.0063
Observations	10

	Coefficients	Standard Error	t-Stat
Intercept	0.0710	0.0094	7.5160
Forecast (Slope)	–0.9041	0.1755	–5.1516



At the 5% significant level, test the null hypothesis that the slope coefficient is significantly different from one, that is,

$$H_0 : b_1 = 1 \; vs. \; H_a : b_1 \neq 1$$

Solution

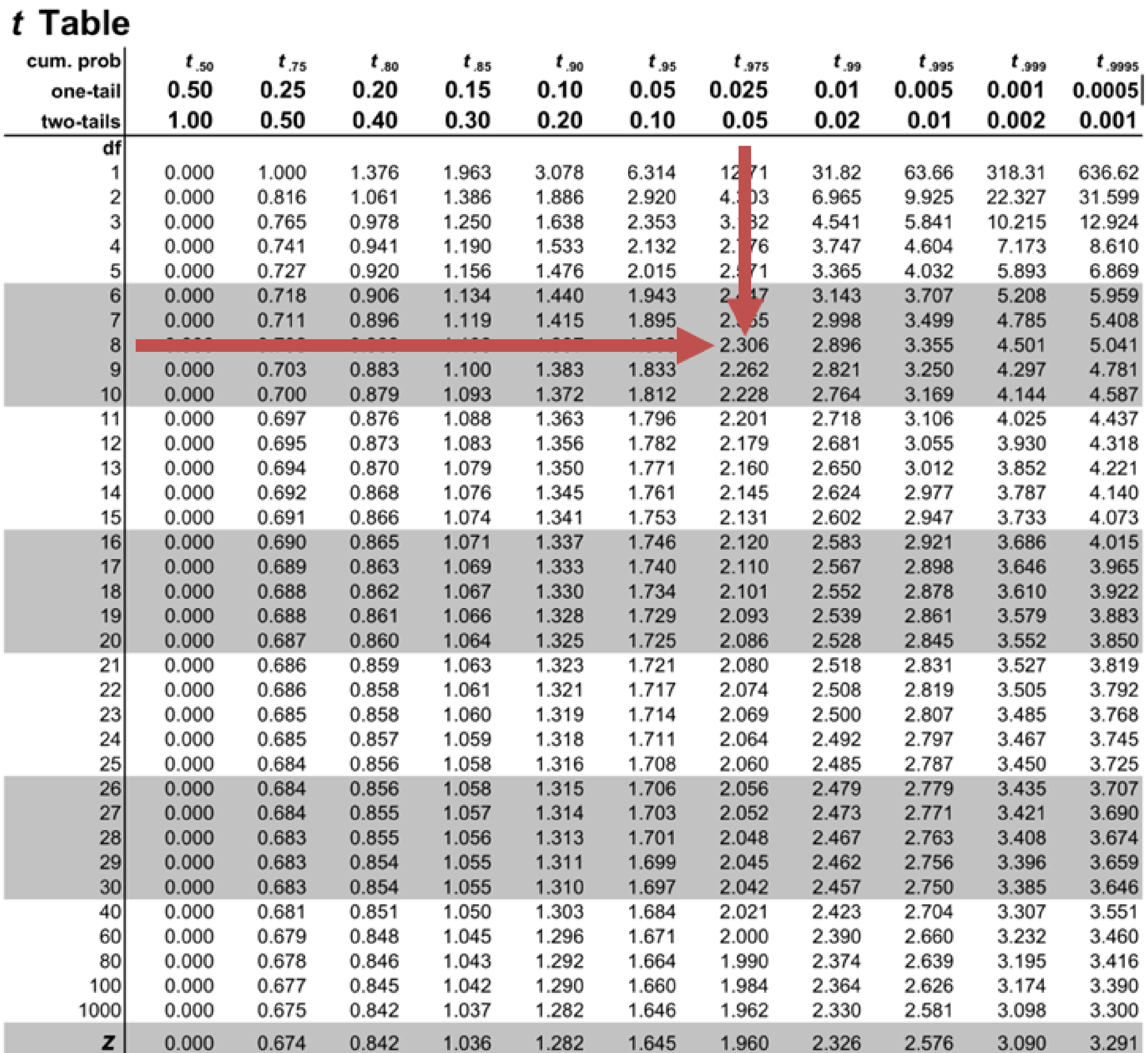
The calculated t-statistic, $t = \frac{\widehat{b_1} - b_1}{\widehat{S_{b_1}}}$ is equal to:

$$\begin{aligned} t &= \frac{-0.9041 - 1}{0.1755} \\ &= -10.85 \end{aligned}$$

The critical two-tail t-values from the table with $n - 2 = 8$ degrees of freedom are:

$$t_c = \pm 2.306$$

Notice that |t| > tc|t| > tc i.e., (10.85>2.30610.85>2.306) Therefore, **we reject the null hypothesis and conclude that the estimated slope coefficient is statistically different from one**.



t Table

cum. prob	$t_{.50}$	$t_{.75}$	$t_{.80}$	$t_{.85}$	$t_{.90}$	$t_{.95}$	$t_{.975}$	$t_{.99}$	$t_{.995}$	$t_{.999}$	$t_{.9995}$
one-tail	**0.50**	**0.25**	**0.20**	**0.15**	**0.10**	**0.05**	**0.025**	**0.01**	**0.005**	**0.001**	**0.0005**
two-tails	**1.00**	**0.50**	**0.40**	**0.30**	**0.20**	**0.10**	**0.05**	**0.02**	**0.01**	**0.002**	**0.001**
df											
1	0.000	1.000	1.376	1.963	3.078	6.314	12.71	31.82	63.66	318.31	636.62
2	0.000	0.816	1.061	1.386	1.886	2.920	4.303	6.965	9.925	22.327	31.599
3	0.000	0.765	0.978	1.250	1.638	2.353	3.182	4.541	5.841	10.215	12.924
4	0.000	0.741	0.941	1.190	1.533	2.132	2.776	3.747	4.604	7.173	8.610
5	0.000	0.727	0.920	1.156	1.476	2.015	2.571	3.365	4.032	5.893	6.869
6	0.000	0.718	0.906	1.134	1.440	1.943	2.447	3.143	3.707	5.208	5.959
7	0.000	0.711	0.896	1.119	1.415	1.895	2.365	2.998	3.499	4.785	5.408
8	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	2.306	2.896	3.355	4.501	5.041
9	0.000	0.703	0.883	1.100	1.383	1.833	2.262	2.821	3.250	4.297	4.781
10	0.000	0.700	0.879	1.093	1.372	1.812	2.228	2.764	3.169	4.144	4.587
11	0.000	0.697	0.876	1.088	1.363	1.796	2.201	2.718	3.106	4.025	4.437
12	0.000	0.695	0.873	1.083	1.356	1.782	2.179	2.681	3.055	3.930	4.318
13	0.000	0.694	0.870	1.079	1.350	1.771	2.160	2.650	3.012	3.852	4.221
14	0.000	0.692	0.868	1.076	1.345	1.761	2.145	2.624	2.977	3.787	4.140
15	0.000	0.691	0.866	1.074	1.341	1.753	2.131	2.602	2.947	3.733	4.073
16	0.000	0.690	0.865	1.071	1.337	1.746	2.120	2.583	2.921	3.686	4.015
17	0.000	0.689	0.863	1.069	1.333	1.740	2.110	2.567	2.898	3.646	3.965
18	0.000	0.688	0.862	1.067	1.330	1.734	2.101	2.552	2.878	3.610	3.922
19	0.000	0.688	0.861	1.066	1.328	1.729	2.093	2.539	2.861	3.579	3.883
20	0.000	0.687	0.860	1.064	1.325	1.725	2.086	2.528	2.845	3.552	3.850
21	0.000	0.686	0.859	1.063	1.323	1.721	2.080	2.518	2.831	3.527	3.819
22	0.000	0.686	0.858	1.061	1.321	1.717	2.074	2.508	2.819	3.505	3.792
23	0.000	0.685	0.858	1.060	1.319	1.714	2.069	2.500	2.807	3.485	3.768
24	0.000	0.685	0.857	1.059	1.318	1.711	2.064	2.492	2.797	3.467	3.745
25	0.000	0.684	0.856	1.058	1.316	1.708	2.060	2.485	2.787	3.450	3.725
26	0.000	0.684	0.856	1.058	1.315	1.706	2.056	2.479	2.779	3.435	3.707
27	0.000	0.684	0.855	1.057	1.314	1.703	2.052	2.473	2.771	3.421	3.690
28	0.000	0.683	0.855	1.056	1.313	1.701	2.048	2.467	2.763	3.408	3.674
29	0.000	0.683	0.854	1.055	1.311	1.699	2.045	2.462	2.756	3.396	3.659
30	0.000	0.683	0.854	1.055	1.310	1.697	2.042	2.457	2.750	3.385	3.646
40	0.000	0.681	0.851	1.050	1.303	1.684	2.021	2.423	2.704	3.307	3.551
60	0.000	0.679	0.848	1.045	1.296	1.671	2.000	2.390	2.660	3.232	3.460
80	0.000	0.678	0.846	1.043	1.292	1.664	1.990	2.374	2.639	3.195	3.416
100	0.000	0.677	0.845	1.042	1.290	1.660	1.984	2.364	2.626	3.174	3.390
1000	0.000	0.675	0.842	1.037	1.282	1.646	1.962	2.330	2.581	3.098	3.300
z	0.000	0.674	0.842	1.036	1.282	1.645	1.960	2.326	2.576	3.090	3.291

Tests on Groups of Coefficients

- The procedure for doing this is the general regression significance test or, as it is more often called, the extra sum of squares method.
- This procedure can also be used to investigate the contribution of a subset involving several regressor or predictor variables to the model.

$$y = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \qquad \text{eq.}$$

- where y is (n × 1),
- X is (n × p),
- $\boldsymbol{\beta}$ is (p × 1),
- $\boldsymbol{\varepsilon}$ is (n × 1), and p = k + 1.
- We would like to determine if a subset of the predictor variables x1, x2, … , xr

Regression: Creating models to predict future observations

- The Regression Approach for Predictions
- Using regression to make predictions doesn’t necessarily involve predicting the future. Instead, you predict the mean of the dependent variable given specific values of the independent variable(s).

- The general procedure for using regression to make good predictions is the following:

1. Research the subject-area so you can build on the work of others. This research helps with the subsequent steps.
2. Collect data for the relevant variables.
3. Specify and assess your regression model.
4. If you have a model that adequately fits the data, use it to make predictions.

- In R programming, predictive models are extremely useful for forecasting future outcomes and estimating metrics that are impractical to measure.

Steps for prediction

1. Collect some data relevant to the problem (more is almost always better).
2. Clean, augment, and preprocess the data into a convenient form, if needed.
3. Conduct an exploratory analysis of the data to get a better sense of it.
4. Using what you find as a guide, construct a model of some aspect of the data.
5. Use the model to answer the question you started with, and validate your results.

Example

- Linear regression is one of the simplest and most common supervised machine learning algorithms that data scientists use for predictive modeling
- Example: regression to build a model that predicts cherry tree volume from metrics that are much easier for folks who study trees to measure.

Example continue

- If you want to practice building the models and visualizations yourself, we'll be using the following R packages:
- data sets This package contains a wide variety of practice data sets. We'll be using one of them, “trees”, to learn about building linear regression models.
- ggplot2 We'll use this popular data visualization package to build plots of our models.
- GGally This package extends the functionality of ggplot2. We'll be using it to create a plot matrix as part of our initial exploratory data visualization.
- scatterplot3d We'll use this package for visualizing more complex linear regression models with multiple predictors.

Example continue

- **How do they measure tree volume, anyway?**
- data(trees) ## access the data from R’s datasets package
- head(trees) ## look at the first several rows of the data

Girth	Height	Volume
8.3	70	10.3
8.6	65	10.3
8.8	63	10.2
10.5	72	16.4
10.7	81	18.8
10.8	83	19.7

- str(trees) ## look at the structure of the variables

Example continue

$ Girth : num	8.3 8.6 8.8 10.5 10.7 10.8 11 11 11.1 11.2 …
$ Height: num	70 65 63 72 81 83 66 75 80 75 …
$ Volume: num	10.3 10.3 10.2 16.4 18.8 19.7 15.6 18.2 22.6 19.9 …

- The trunk girth (in)
- height (ft)
- volume (ft^3)

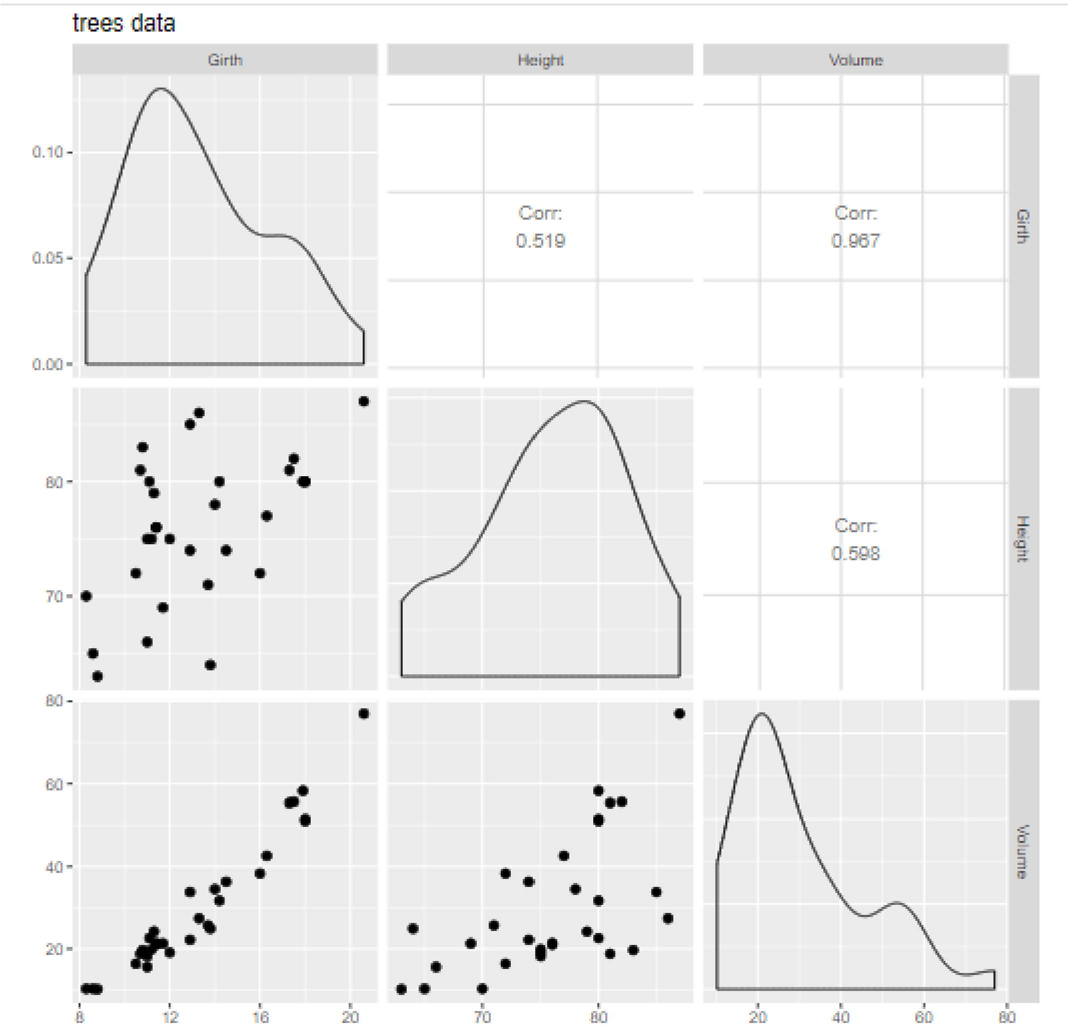
To decide whether we can make a predictive model, the first step is to see if there appears to be a relationship between our predictor and response variables (in this case girth, height, and volume).

```
ggpairs(data=trees, columns=1:3, title="trees data")
```

Let’s do some exploratory data visualization. We’ll use the ggpairs() function from the GGally package to create a plot matrix to see how the variables relate to one another.

Example continue

- The ggpairs() function gives us scatter plots for each variable combination, as well as density plots for each variable and the strength of correlations between variables.

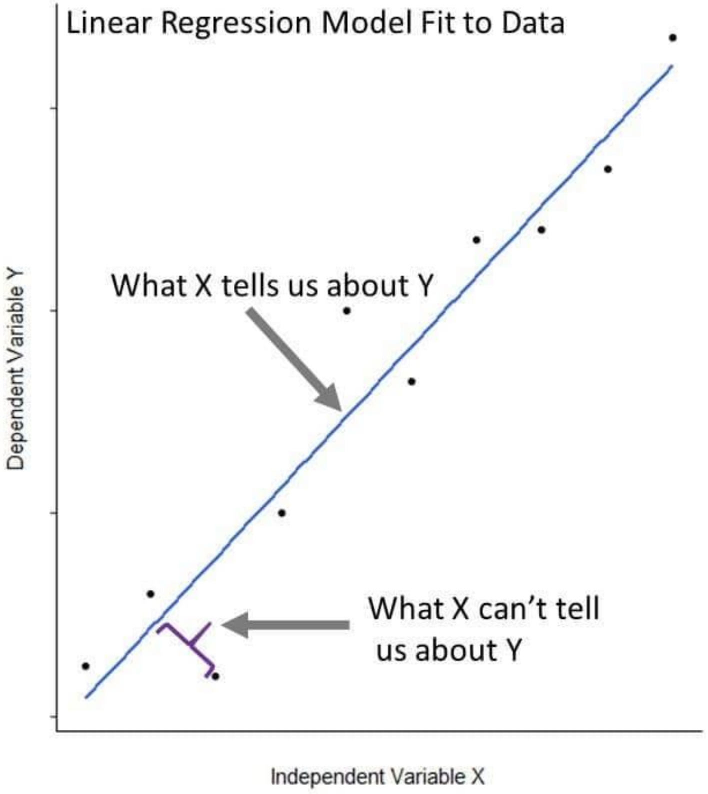


Example continue

- **Forming a hypothesis and use**
- A hypothesis is an educated guess about what we think is going on with our data.
- In this case, let’s hypothesize that cherry tree girth and volume are related.
- Every hypothesis we form has an opposite: the “null hypothesis” (H0). Here, our null hypothesis is that girth and volume aren’t related.
- In statistics, the null hypothesis is the one we use our data to support or reject; we can’t ever say that we “prove” a hypothesis.
- We call the hypothesis that girth and volume are related our “alternative” hypothesis (Ha).
- To summarize: H0 : There is no relationship between girth and volume
- Ha: There is some relationship between girth and volume Our linear regression model is what we will use to test our hypothesis.
- If we find strong enough evidence to reject H0, we can then use the model to predict cherry tree volume from girth.

Example continue

- **Building blocks of a linear regression model**
- Let’s dive right in and build a linear model relating tree volume to girth. R makes this straight forward with the base function lm().
- fit_1 <- lm(Volume ~ Girth, data = trees)
- The lm() function fits a line to our data that is as close as possible to all 31 of our observations.



Example continue

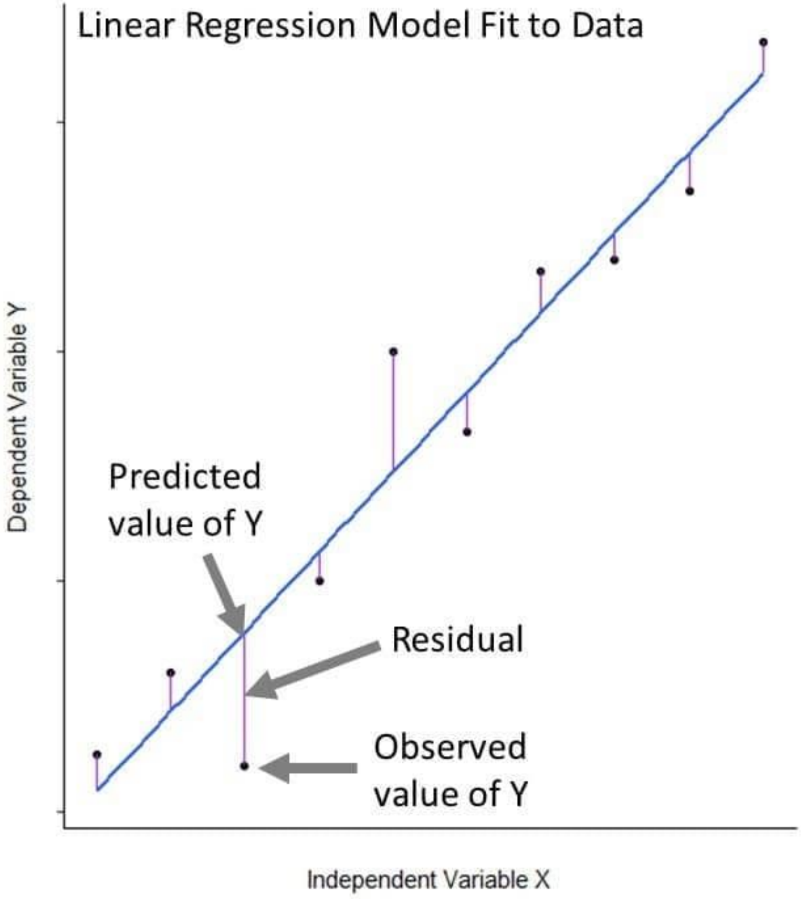
Mathematically, can we write the equation for linear regression as: **Y ≈ β0 + β1X + ε**

•The **Y** and **X** variables are the response and predictor variables from our data that we are relating to eachother

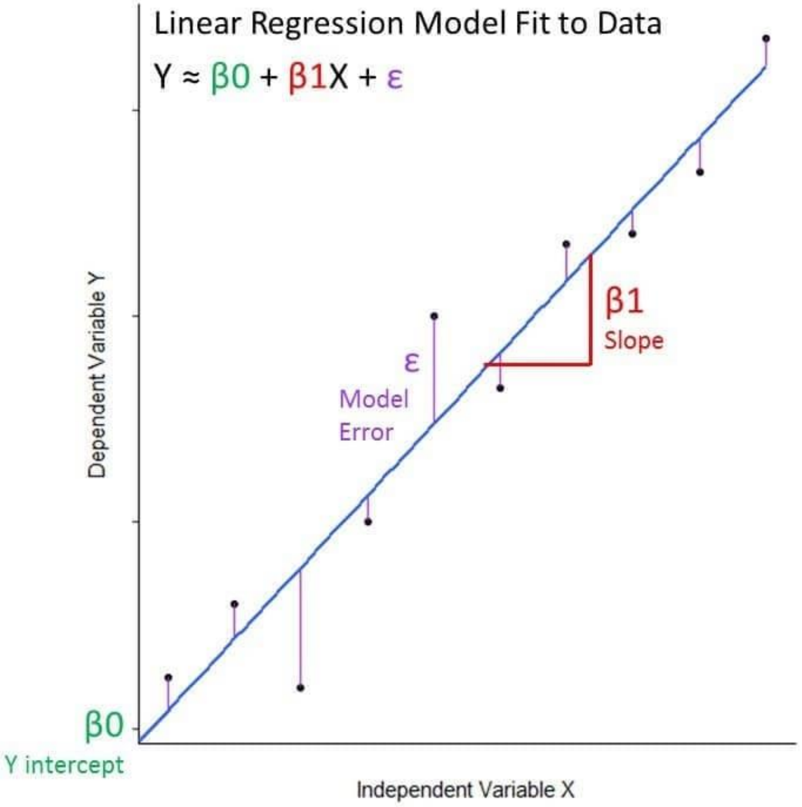
•**β0** is the model coefficient that represents the model intercept, or where it crosses the y axis

•**β1** is the model coefficient that represents the model slope, the number that gives information about the steepness of the line and its direction (positive or negative)

•**ε** is the error term that encompasses variability we cannot capture in the model (what X cannot tell us about Y)



In the case of our example: **Tree Volume ≈ Intercept + Slope(Tree Girth) + Error**



- **Can we use this model to make predictions?**
- Whether we can use our model to make predictions will depend on:

  1. Whether we can reject the null hypothesis that there is no relationship between our variables.
  2. Whether the model is a good fit for our data.

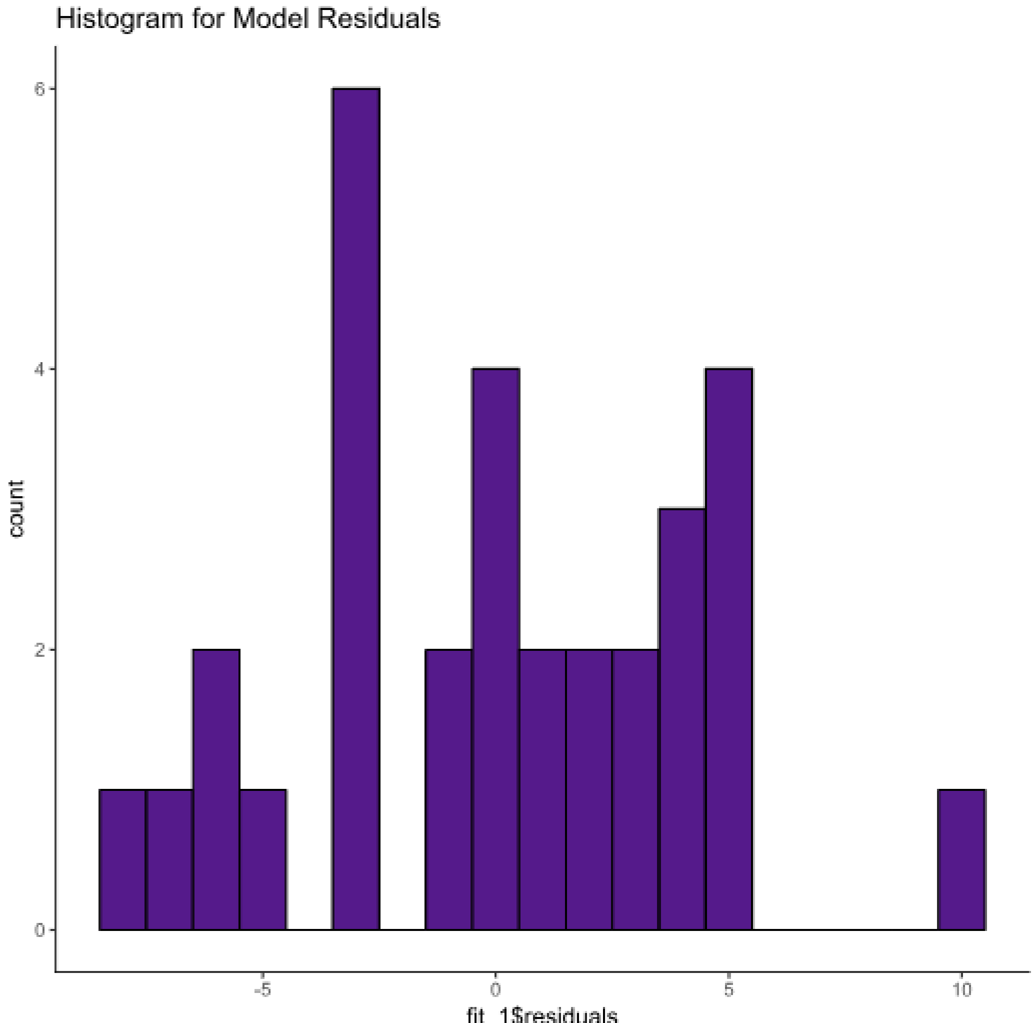
- Let’s call the output of our model using summary().

```
summary(fit_1)
```

- **Is the hypothesis supported?**
- *Coefficients: Estimate and Std. Error*:
- The intercept in our example is the expected tree volume if the value of girth was zero. Of course we cannot have a tree with negative volume, but more on that later.
- The slope in our example is the effect of tree girth on tree volume. We see that for each additional inch of girth, the tree volume increases by 5.0659 ft$^{3.}$
- The coefficient standard errors tell us the average variation of the estimated coefficients from the actual average of our response variable.

- *t value*:
- This is a p-value, defined as the probability of observing any value equal or larger than t if H_0 is true. The larger the t statistic, the smaller the p-value. Generally, we use 0.05 as the cutoff for significance; when p-values are smaller than 0.05, we reject H_0.
- We can reject the null hypothesis in favor of believing there to be a relationship between tree width and volume.

- **How well does the model fit the data?**
- *Residuals*:
- We can make a histogram to visualize this using ggplot2.
- ggplot(data=trees, aes(fit_1$residuals)) +
- geom_histogram(binwidth = 1, color = "black", fill = "purple4") +
- theme(panel.background = element_rect(fill = "white"),
- axis.line.x=element_line(),
- axis.line.y=element_line()) +
- ggtitle("Histogram for Model Residuals")



- **Using our simple linear model to make predictions**

Girth	Height	Volume
18.2 in	72 ft	46.2 ft^3

- **predict(fit_1, data.frame(Girth = 18.2))**

- **Adding more predictors: multiple linear regression**

Unit II - REGRESSION ANALYSIS AND FORECASTING

Unit outcome

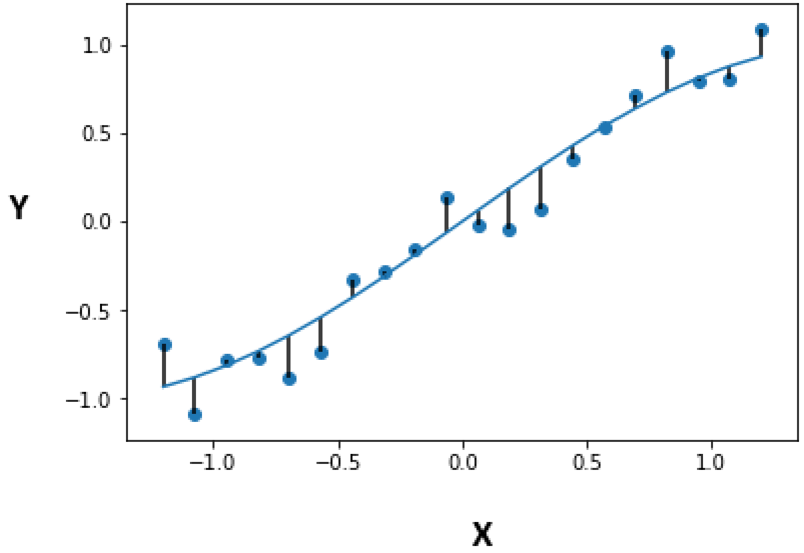
- 2. Illustrate the regression analysis and forecasting in time series analysis

MODEL ADEQUACY CHECKING

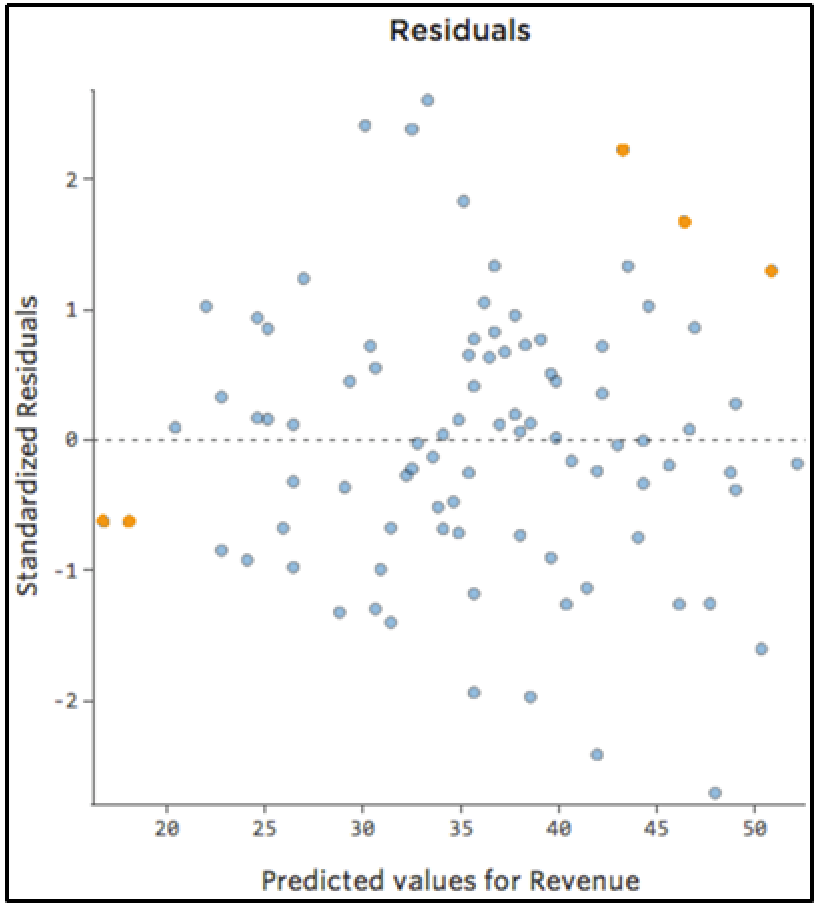
- **What is Residuals?**
- A residual is a measure of how far away a point is vertically from the regression line. Simply, it is the **error between a predicted value and the observed actual value.**

$$Residual\ (\epsilon) = \ y - \ \hat{y}$$

- Figure 1 is an example of how to visualize residuals against the line of best fit. The vertical lines are the residuals.



- **Residual Plots**
- A typical residual plot has the **residual values on the Y-axis and the independent variable on the x-axis.**
- Figure 2 below is a good example of how a typical residual plot looks like.
- **Residual plots are the primary approach to model adequacy checking**



- **Residual Plot Analysis**
- The most important assumption of a linear regression model is that the errors are independent and normally distributed.

- **Characteristics of Good Residual Plots**

- It has a high density of points close to the origin and a low density of points away from the origin
- It is symmetric about the origin

Residuals Use

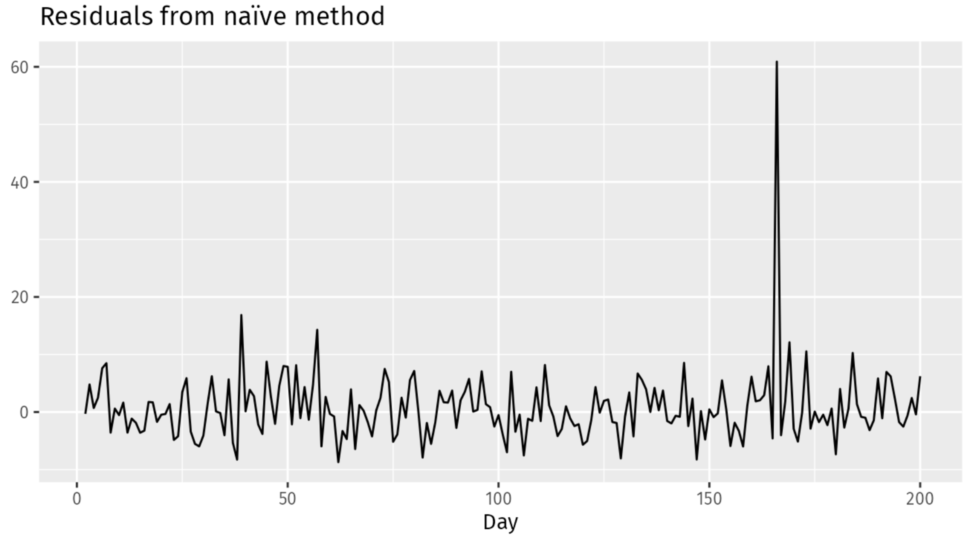
- Residuals are useful in checking whether **a model has adequately captured the information in the data**.
- A good forecasting method will yield residuals with the following properties:

1. The residuals are uncorrelated. If there are correlations between residuals, then there is information left in the residuals which should be used in computing forecasts.
2. The residuals have zero mean. If the residuals have a mean other than zero, then the forecasts are biased.

Example: Forecasting the Google daily closing stock price

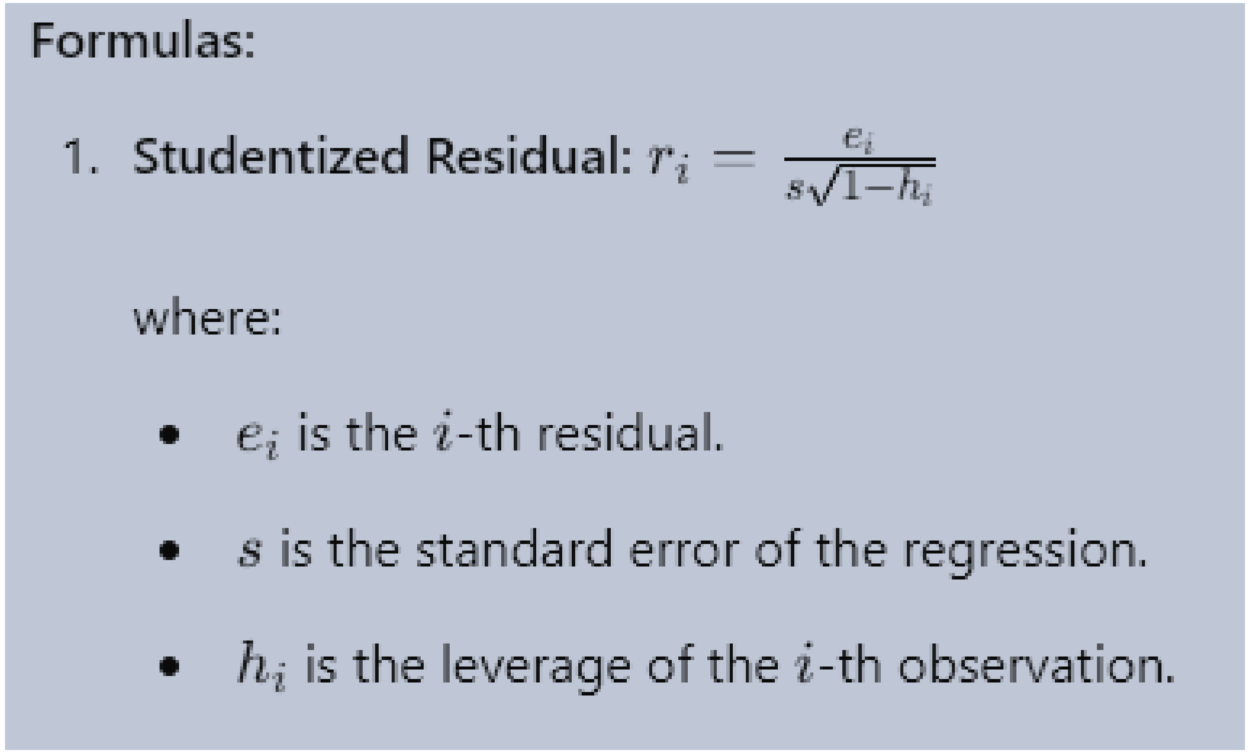
- For stock market prices and indexes, the best forecasting method is often the naïve method. That is, each forecast is simply equal to the last observed value.
- The following graph shows the Google daily closing stock price (GOOG).

```
autoplot(goog200) +
  xlab("Day") + ylab("Closing Price (US$)") +
  ggtitle("Google Stock (daily ending 6 December
2013)")

res <- residuals(naive(goog200))
autoplot(res) + xlab("Day") + ylab("") +
  ggtitle("Residuals from naïve method")
```


Residuals from naïve method

Scaled Residuals and PRESS

- Residuals are the differences between **observed and predicted values from a model**.
- Scaled residuals are adjusted versions of these residuals, often used to standardize them, making it easier to identify outliers or assess the overall fit of the model.
- Scaled residuals are residuals from a statistical model that have been adjusted to account for variability, making them more comparable across different observations.
- They are used to **standardize residuals** so that **their distribution** can be **more easily analyzed**, particularly when checking model assumptions like homoscedasticity (constant variance) and normality.



Formulas:

1. **Studentized Residual:** $r_i = \frac{e_i}{s\sqrt{1-h_i}}$

   where:

   - e_i is the i-th residual.
   - s is the standard error of the regression.
   - h_i is the leverage of the i-th observation.

Scaled Residuals Methods

- There are a few common methods to scale residuals:

1. Standardized Residuals: These are the residuals divided by an estimate of their standard deviation. For a linear regression model, this is often the standard error of the residuals. They help identify outliers and are useful in diagnostic plots.

$$r_i = \frac{e_i}{\hat{\sigma}\sqrt{1 - h_{ii}}}$$

- where ei is the residual for observation i,
- σ^ is the estimated standard deviation of the residuals,
- and hii is the leverage of observation ii.

2. Studentized Residuals: These are similar to standardized residuals but are more accurate for small sample sizes. They use a more precise estimate of the variance.

$$t_i = \frac{e_i}{\hat{\sigma}_{(i)}\sqrt{1 - h_{ii}}}$$

- where σ^(i)is the estimated standard deviation of the residuals when the i-th observation is excluded.

- **Pearson Residuals**: Commonly used in the context of generalized linear models (GLMs), these residuals are the difference between the observed and fitted values divided by the standard deviation of the observed values.

$$p_i = \frac{y_i - \hat{y}_i}{\sqrt{\hat{\sigma}_i^2}}$$

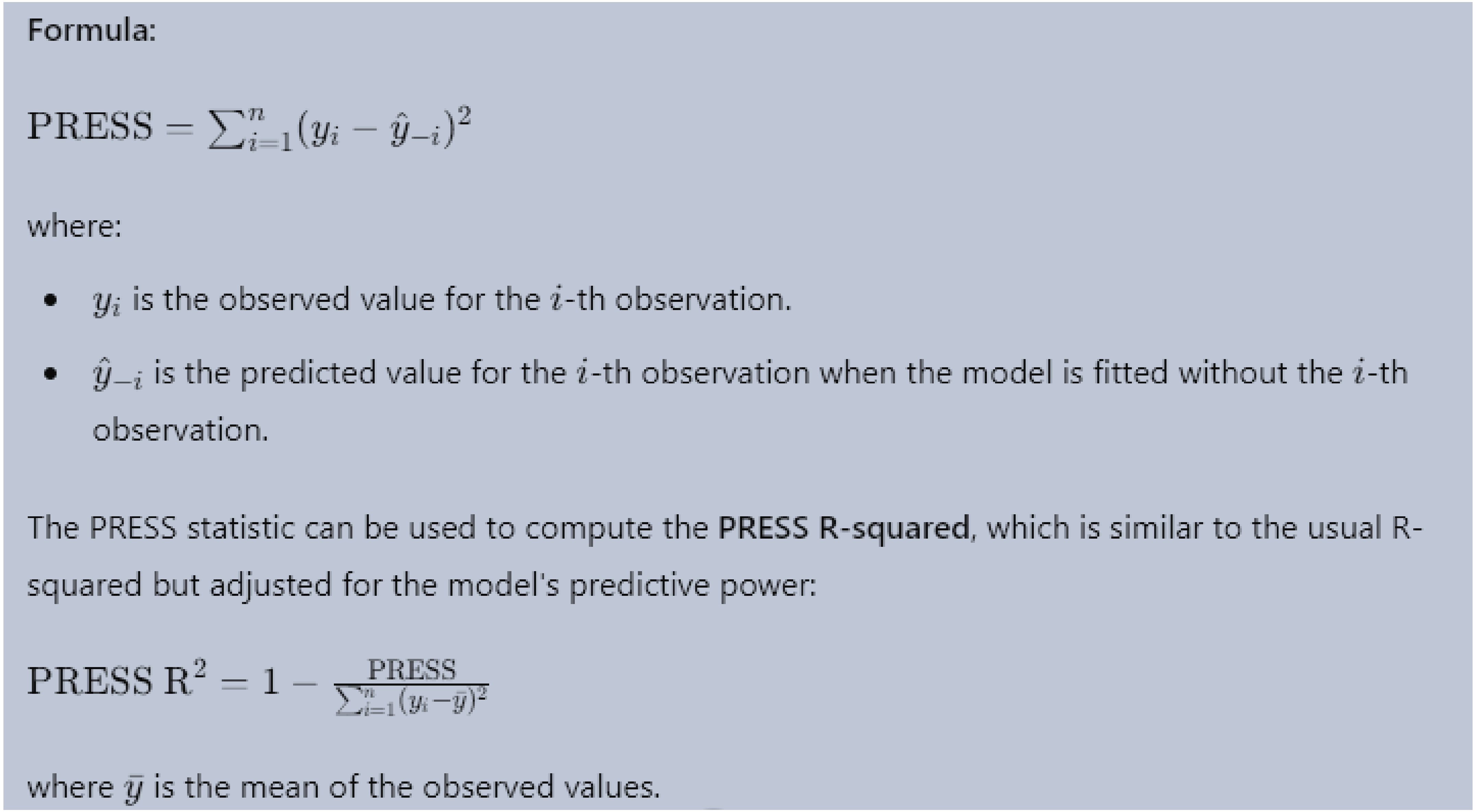
- where yi is the observed value, $y^{\wedge}i$ is the fitted value, and $\sigma^{\wedge 2}i$ is the standard deviation of the observed value.

- **4. Deviance Residuals**: Used in generalized linear models (GLMs), these are the signed square root of the contribution to the model's deviance for each observation. They are helpful for identifying poorly fitted data points.

$$d_i = \text{sign}(y_i - \hat{y}_i)\sqrt{2\left(y_i \log\left(\frac{y_i}{\hat{y}_i}\right) - (y_i - \hat{y}_i)\right)}$$

PRESS (Predicted Residual Sum of Squares)

- PRESS is a measure used to evaluate the predictive capability of a regression model.
- It is particularly useful in assessing how well the model generalizes to new data.
- The PRESS statistic is calculated by leaving out one observation from the dataset, fitting the model to the remaining data, predicting the left-out observation, and
- then computing the squared difference between the observed and predicted values.
- This process is repeated for each observation in the dataset, and the squared differences are summed.



Formula:

$$\mathrm{PRESS} = \sum_{i=1}^{n}(y_i - \hat{y}_{-i})^2$$

where:

- y_i is the observed value for the i-th observation.
- $\hat{y}_{-i}$ is the predicted value for the i-th observation when the model is fitted without the i-th observation.

The PRESS statistic can be used to compute the **PRESS R-squared**, which is similar to the usual R-squared but adjusted for the model's predictive power:

$$\mathrm{PRESS\ R}^2 = 1 - \frac{\mathrm{PRESS}}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

where $\bar{y}$ is the mean of the observed values.

- **Usage and Interpretation**
- **Scaled Residuals** are useful for diagnosing potential issues with the model, such as outliers or influential points.
- They help to identify observations that may disproportionately affect the model's fit.
- **PRESS** and **PRESS R-squared** provide insights into the model's ability to predict new data.
- A lower PRESS value indicates a model with better predictive accuracy, while a higher PRESS R-squared indicates a model that generalizes well.

- **Interpretation**
- Studentized residuals can help identify potential outliers. Typically, residuals greater than 2 or less than -2 may indicate outliers, depending on the context and sample size.
- You can also use other packages like MASS for additional diagnostics, including calculating other types of residuals or using robust regression methods.

Example: Calculating Studentized Residuals in R

Explanation

1. **Loading Necessary Packages:** The car package is useful for various regression diagnostics, including calculating studentized residuals.
2. **Creating Example Data:** We create a simple dataset with x as the predictor and y as the response variable.
3. **Fitting a Linear Model:** The lm() function is used to fit a linear model, with y as the response and x as the predictor.
4. **Calculating Studentized Residuals:** The rstudent() function from the car package calculates the studentized residuals, which are printed to the console.

```
# Load necessary package
install.packages("car") # if not already installed
library(car)

# Example data
data <- data.frame(
  x = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10),
  y = c(2.3, 2.9, 3.2, 4.5, 5.1, 6.8, 7.3, 8.0, 9.5, 9.8)
)
# Fit a linear model
model <- lm(y ~ x, data = data)
# Summary of the model
summary(model)
# Calculate studentized residuals
studentized_residuals <- rstudent(model)
# Output studentized residuals
print(studentized_residuals)
```

Variable Selection Methods in Regression

- Variable selection is a crucial aspect of building regression models, as it helps in **identifying** the most relevant **predictors** and improving model **interpretability**, **accuracy**, and **generalizability**.

1. Stepwise Selection

- Stepwise selection methods iteratively add or remove variables based on specific criteria. There are three main types:
- **Forward Selection:** Starts with no predictors and adds variables one by one, based on a chosen criterion (e.g., p-value, AIC, BIC). The process stops when no additional significant predictors can be added.
- **Backward Elimination:** Starts with all candidate variables and removes the least significant variable at each step, again based on a criterion, until no insignificant predictors remain.
- **Stepwise Selection (Both Directions):** Combines forward selection and backward elimination by adding or removing variables at each step based on the criterion.

- **How Forward Selection Works:**
- **Start with No Predictors:**
  - Begin with an empty model that contains no predictors (independent variables).
- **Add Variables One by One:**
  - Evaluate each predictor variable individually to see how well it improves the model when added. This is typically done by fitting the model with one predictor at a time and assessing the performance using a chosen criterion.
- **Choose the Best Predictor:**
  - Select the predictor that improves the model the most according to the chosen criterion. This criterion can be a statistical measure like the p-value, AIC (Akaike Information Criterion), BIC (Bayesian Information Criterion), or adjusted R-squared.
- **Repeat the Process:**
  - Add the chosen predictor to the model and then repeat the process by evaluating the remaining predictors. The model is refitted with the new predictor included, and the evaluation criterion is used to determine the next best predictor to add.
- **Stop When Criteria Are Met:**
  - Continue adding predictors until no additional variables meet the criterion for inclusion, or until a specified stopping rule is met (such as a predetermined number of predictors or the criterion no longer improving).

- **Criteria for Selecting Predictors:**
- **p-Value:** Often used in hypothesis testing to determine the statistical significance of a predictor. Predictors with lower p-values are considered more significant.
- **AIC (Akaike Information Criterion):** A measure of the relative quality of a model. It balances model fit and complexity, with lower AIC values indicating better models.
- **BIC (Bayesian Information Criterion):** Similar to AIC, but penalizes model complexity more heavily. Lower BIC values are preferred.
- **Adjusted R-squared:** Indicates the proportion of variance explained by the predictors, adjusted for the number of predictors. Higher values suggest better model fit.

2. Lasso Regression (Least Absolute Shrinkage and Selection Operator)

- Lasso regression is a regularization technique that includes a penalty proportional to the absolute value of the coefficients. It can shrink some coefficients to zero, effectively performing variable selection. The penalty term is controlled by a tuning parameter (λ), which can be selected using cross-validation.
- **Benefits of Lasso Regression**
- **Simplicity and Interpretability:** By shrinking some coefficients to zero, lasso regression helps in creating simpler models with fewer predictors, which are easier to interpret.
- **Prevents Overfitting:** The regularization term reduces the risk of overfitting by discouraging overly complex models. This can lead to better generalization on new, unseen data.
- **Feature Selection:** Lasso can automatically select a subset of features and discard others, which is particularly useful when dealing with high-dimensional data.

3. Ridge Regression

- Ridge regression is another regularization technique that includes a penalty proportional to the square of the coefficients. Unlike lasso, it does not perform variable selection but can reduce multi-collinearity and improve model stability.
- **Benefits of Ridge Regression**
- **Handles Multicollinearity**: By adding a penalty term, ridge regression can stabilize the coefficient estimates and reduce their variance.
- **Improves Prediction Accuracy**: The regularization term helps to prevent overfitting, especially in models with many predictors or when the predictors are highly correlated.
- **Coefficients Shrinkage**: It tends to shrink the coefficients towards zero but does not set them exactly to zero. This means it can keep all features in the model, unlike Lasso regression which can perform variable selection by setting some coefficients to zero.

4. Elastic Net

- Elastic Net combines the penalties of both lasso and ridge regression. It can handle situations where there are highly correlated variables and can select groups of correlated variables.

5. Principal Component Analysis (PCA)

- PCA is a dimensionality reduction technique that transforms the predictors into a set of uncorrelated components. The first few components, which capture most of the variance, can be used in the regression model. PCA is helpful when dealing with multicollinearity.

Principal Component Analysis

- **How PCA Works**
- **Standardization**: If the variables have different units or scales, standardize the data (i.e., subtract the mean and divide by the standard deviation) so that each variable contributes equally to the analysis.
- **Covariance Matrix**: Compute the covariance matrix of the standardized data. This matrix captures the relationships between different variables.
- **Eigenvalues and Eigenvectors**: Calculate the eigenvalues and eigenvectors of the covariance matrix. The eigenvectors represent the directions of maximum variance (principal components), and the eigenvalues represent the magnitude of these variances.
- **Sort and Select**: Sort the eigenvectors by their corresponding eigenvalues in descending order. Select the top kkk eigenvectors to form a new matrix that will be used to transform the original data into the principal component space.
- **Transform Data**: Multiply the original standardized data by the selected eigenvectors (principal components) to get the reduced representation of the data.

6. Information Criterion-Based Methods

- **Akaike Information Criterion (AIC):** AIC is a measure of model quality that balances goodness of fit and model complexity. Lower AIC values indicate better models. It can be used to compare models with different sets of variables.
- **Bayesian Information Criterion (BIC):** Similar to AIC, BIC includes a penalty term that increases with the number of parameters, favoring simpler models. It tends to select fewer variables than AIC.

- **7. Recursive Feature Elimination (RFE)**
- RFE is an iterative process that fits a model and removes the least significant features one by one. The model is re-fitted at each step, and features are ranked based on their importance.
- **8. Cross-Validation**
- Cross-validation techniques, like k-fold cross-validation, are used in conjunction with other methods to assess model performance and select the best set of variables. Cross-validation helps prevent overfitting and provides a more robust measure of a model's predictive capability.
- **9. Domain Knowledge**
- Incorporating domain knowledge can be crucial for variable selection. Experts may know which variables are most relevant, leading to more parsimonious and interpretable models.

Generalized and Weighted Least Squares

- **Generalized Least Squares (GLS)** and **Weighted Least Squares (WLS)** are both extensions of the **ordinary least squares (OLS)** method, which are used when the assumptions of OLS are violated, particularly concerning the homoscedasticity and independence of errors.
- **Generalized Least Squares (GLS)**
- **Purpose:** GLS is used when the error terms in a regression model are not **homoscedastic** (i.e., they do not have constant variance) or are **correlated**.
- This violation can lead to **inefficient** and **biased estimates** in OLS.
- **GLS addresses these issues by transforming the model so that the transformed errors are homoscedastic and uncorrelated.**

Generalized Least Squares (GLS)

- **Method:**
- GLS modifies the standard OLS approach by transforming the model to ensure that the error terms have constant variance and are uncorrelated.
- **Steps:**

1. **Estimate the covariance matrix Σ\SigmaΣ**: This can be done through various methods, such as using residuals from an OLS fit.
2. **Transform the model**: Multiply both sides of the regression equation by $\sum^{-1/2}$
3. **Apply OLS to the transformed model**: This results in unbiased and efficient estimators for the regression coefficients.

Generalized Least Squares (GLS)

- **Application:** GLS is used when there is a known form of heteroscedasticity or correlation structure among the errors, such as time series data with auto correlated errors or panel data with cross-sectional correlation.

Weighted Least Squares (WLS)

- **Purpose:** WLS is a special case of GLS used when the error terms have non-constant variance (heteroscedasticity), but the errors are uncorrelated.
- WLS assigns weights to each observation, with the weights being inversely proportional to the variance of the error term for that observation.

- **Method:**
- In WLS, weights are applied to each observation to account for the differing variances.
- The weights are typically the inverse of the variances of the error terms.

- **Steps:**
- **Determine the weights**: These are often estimated as the inverse of the variances of the errors.
- **Transform the model**: Multiply both sides of the regression equation by the square root of the weights $\sqrt{wi}$
- **Apply OLS to the weighted model**: This results in unbiased and efficient estimators for the regression coefficients.

- **Key Differences**
- **Assumptions about Errors:**
  - **GLS:** Assumes heteroscedasticity and/or correlation in error terms.
  - **WLS:** Assumes heteroscedasticity but no correlation among errors.
- **Weights and Transformation:**
  - **GLS:** Uses the covariance matrix Ω\OmegaΩ to transform the data.
  - **WLS:** Uses weights that are inversely proportional to the variance of the error terms.
- **Complexity:**
  - **GLS:** More complex due to the need for estimating or knowing the covariance structure Ω\OmegaΩ.
  - **WLS:** Simpler, as it only requires knowledge or estimation of variances for weighting.
- Both GLS and WLS improve the efficiency of estimates compared to OLS when the assumptions of OLS are not met. They help to achieve better estimates by appropriately accounting for the structure in the errors.

REGRESSION MODELS FOR GENERAL TIME SERIES DATA

Step 1: Data Preparation

1.1 Collect and Clean Data

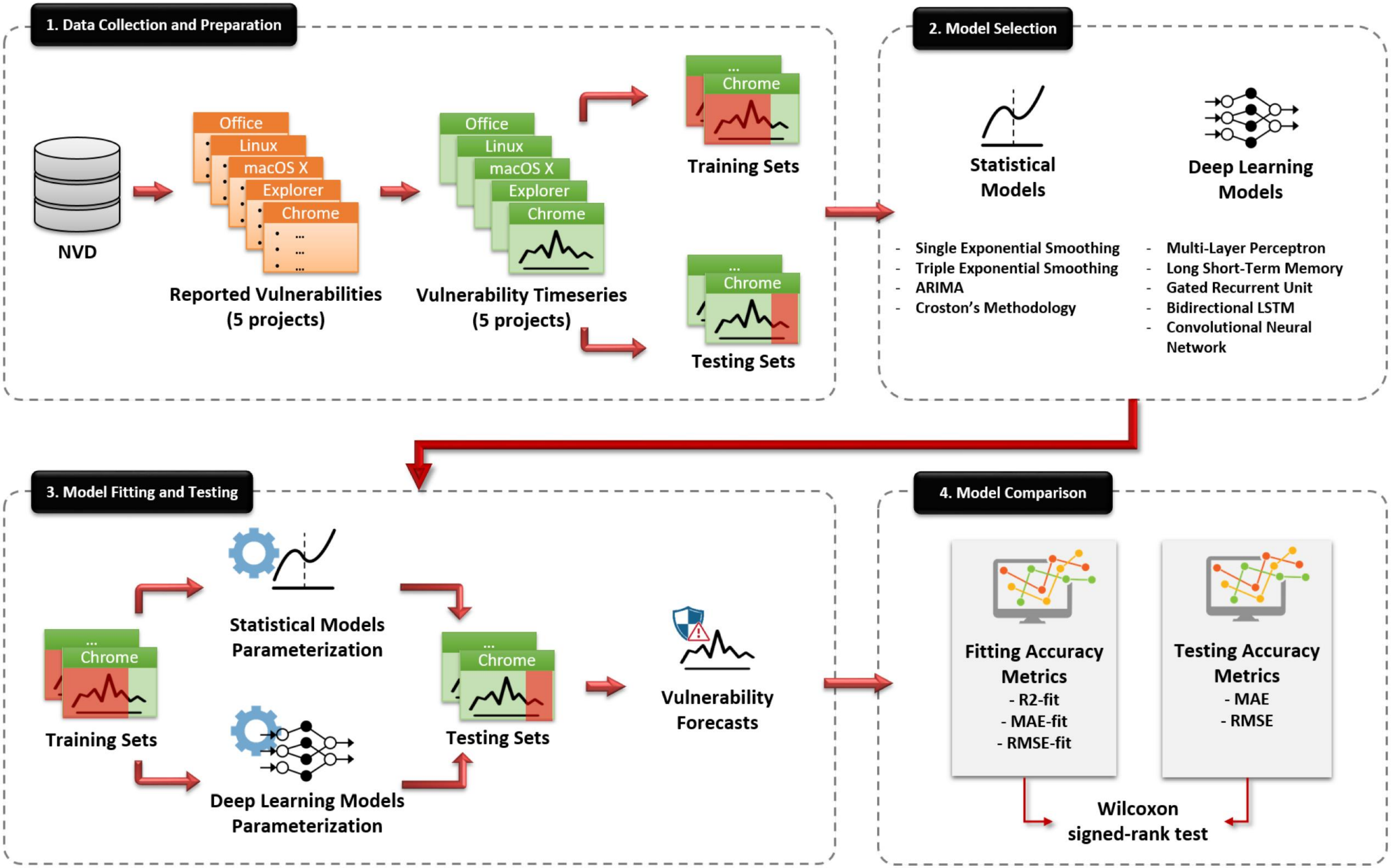
- **Collect Data**: Gather the time series data.
- **Clean Data**: Handle missing values (impute or remove), remove outliers, and ensure data is in chronological order.

1.2 Exploratory Data Analysis (EDA)

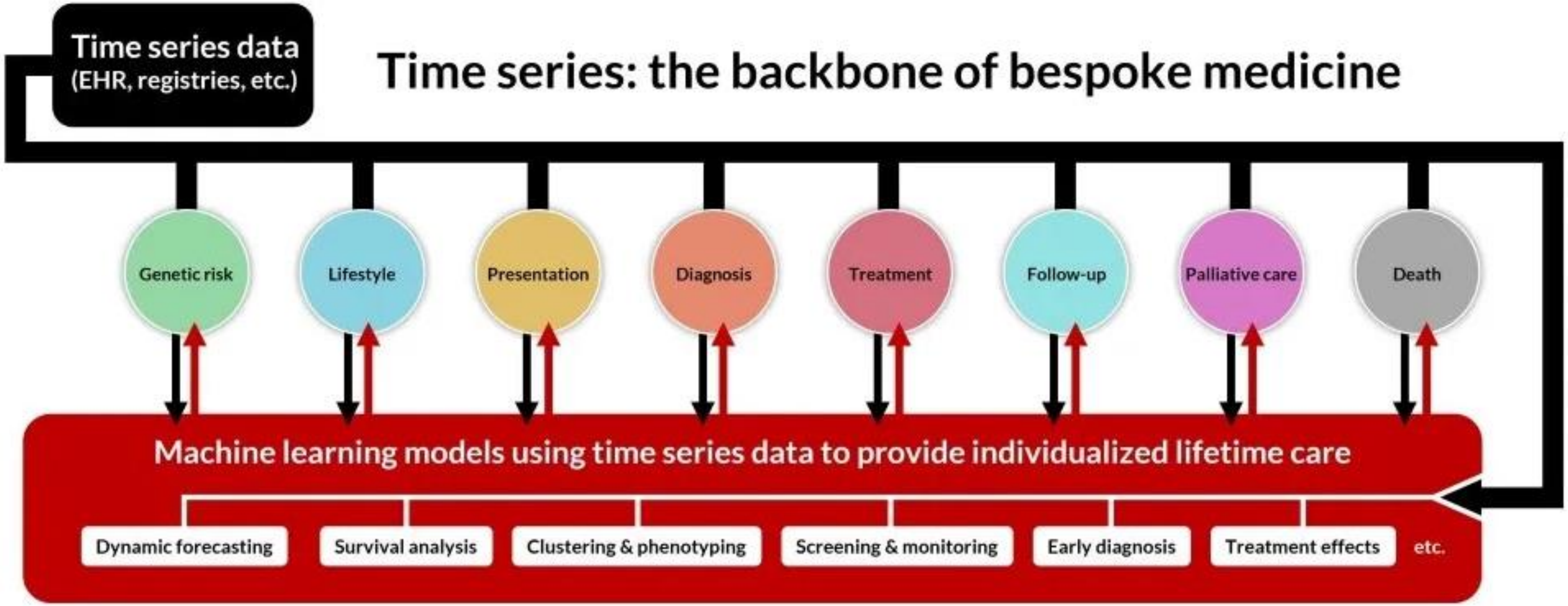
- **Plot the Time Series**: Visualize the data to identify trends, seasonality, and any anomalies.
- **Summary Statistics**: Calculate mean, median, variance, etc.
- **Check for Stationarity**: Use plots and statistical tests like the Augmented Dickey-Fuller (ADF) test.

1.3 Transform Data

- **Detrend**: Remove long-term trends (e.g., differencing).
- **Deseasonalize**: Remove seasonal effects.
- **Stabilize Variance**: Apply transformations like log or Box-Cox if necessary.


1. Data Collection and Preparation
NVD
Office
Linux
macOS X
Explorer
Chrome
...
Reported Vulnerabilities (5 projects)
Vulnerability Timeseries (5 projects)
Training Sets
Testing Sets
2. Model Selection
Statistical Models
Deep Learning Models
- Single Exponential Smoothing
- Triple Exponential Smoothing
- ARIMA
- Croston's Methodology
- Multi-Layer Perceptron
- Long Short-Term Memory
- Gated Recurrent Unit
- Bidirectional LSTM
- Convolutional Neural Network
3. Model Fitting and Testing
Training Sets
Statistical Models Parameterization
Deep Learning Models Parameterization
Testing Sets
Vulnerability Forecasts
4. Model Comparison
Fitting Accuracy Metrics
- R2-fit
- MAE-fit
- RMSE-fit
Testing Accuracy Metrics
- MAE
- RMSE
Wilcoxon signed-rank test

Time series: the backbone of bespoke medicine



Time series data
(EHR, registries, etc.)

Genetic risk

Lifestyle

Presentation

Diagnosis

Treatment

Follow-up

Palliative care

Death

Machine learning models using time series data to provide individualized lifetime care

Dynamic forecasting

Survival analysis

Clustering & phenotyping

Screening & monitoring

Early diagnosis

Treatment effects

etc.

Step 2: Model Selection

2.1 Choose a Model

Depending on the characteristics of your time series data, select an appropriate model:

- **ARIMA**: AutoRegressive Integrated Moving Average for univariate time series.
- **SARIMA**: Seasonal ARIMA for time series with seasonality.
- **VAR**: Vector AutoRegression for multivariate time series.
- **SARIMAX**: Seasonal ARIMA with eXogenous regressors.
- **Holt-Winters**: Exponential smoothing for trend and seasonality.
- **Machine Learning Models**: Random Forest, XGBoost, etc., for capturing non-linear relationships.

2.2 Split Data

- **Training and Testing Sets**: Split the data into training and testing sets to validate the model.

Step 3: Model Training

3.1 Define the Model

- Use appropriate libraries and methods to define the model. For example, using statsmodels for ARIMA:

```
import statsmodels.api as sm

# ARIMA model example
model = sm.tsa.ARIMA(train_data, order=(p,d,q))

# SARIMAX model example
model = sm.tsa.SARIMAX(train_data, order=(p,d,q),
seasonal_order=(P,D,Q,s), exog=exog_train)
```

3.2 Fit the Model

- Fit the model to the training data:

```
model_fit = model.fit(disp=False)
print(model_fit.summary())
```

3.3 Diagnose the Model

- Check residuals to ensure they are white noise:
  - **Plot Residuals**: Residual plots should show no obvious patterns.
  - **Ljung-Box Test**: Test for autocorrelation in residuals.
  - **ACF and PACF**: Autocorrelation Function and Partial Autocorrelation Function plots of residuals.

Step 4: Forecasting and Validation

4.1 Make Predictions

- Predict on the training set and forecast future values:

```
# In-sample prediction
in_sample_pred = model_fit.predict(start, end)

# Out-of-sample forecast
forecast = model_fit.forecast(steps=len(test_data),
exog=exog_test)
```

4.2 Evaluate Model Performance

Calculate performance metrics and visualize the results:

- **Metrics**: Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), etc.
- **Plot Actual vs. Predicted**: Compare actual values with predictions.

```
from sklearn.metrics import mean_squared_error

rmse = mean_squared_error(test_data, forecast,
squared=False)
print(f'RMSE: {rmse}')

import matplotlib.pyplot as plt

plt.figure(figsize=(10,6))
plt.plot(train_data, label='Train')
plt.plot(test_data, label='Test')
plt.plot(test_data.index, forecast, label='Forecast')
plt.legend()
plt.show()
```

4.3 Model Refinement

- **Parameter Tuning**: Adjust model parameters to improve performance.
- **Cross-Validation**: Validate the model with different subsets of the data.

Step 5: Implementation

5.1 Final Model Training

- Train the model on the entire dataset for final implementation.

5.2 Deployment

- **Deploy the Model**: For real-time or batch prediction.
- **Monitor Performance**: Continuously monitor model performance and retrain as necessary.

Step 6: Documentation and Reporting

6.1 Documentation

- **Document Process**: Detail each step, including assumptions and decisions made.
- **Create Visualizations**: Use plots to illustrate findings and model performance.

6.2 Reporting

- **Summary Report**: Present findings, model performance, and any recommendations.
- **Stakeholder Communication**: Clearly communicate results to stakeholders.

- **Example Code**
- Here's an example code snippet using ARIMA:

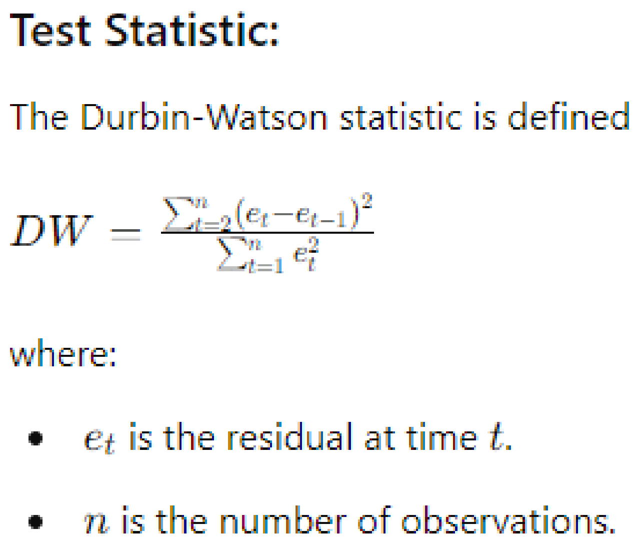
```
import pandas as pd
import statsmodels.api as sm
from sklearn.metrics import mean_squared_error

# Load data
data = pd.read_csv('time_series_data.csv', index_col='date',
parse_dates=True)

# Split data
train_data = data[:'2022']
test_data = data['2023':]

# Fit ARIMA model
model = sm.tsa.ARIMA(train_data, order=(1,1,1))
model_fit = model.fit(disp=False)
# In-sample prediction
in_sample_pred = model_fit.predict(start=len(train_data),
end=len(train_data)+len(test_data)-1, dynamic=False)
# Forecast

forecast = model_fit.forecast(steps=len(test_data))[0]

# Evaluate
rmse = mean_squared_error(test_data, forecast,
squared=False)
print(f'RMSE: {rmse}')

# Plot
import matplotlib.pyplot as plt
plt.figure(figsize=(10,6))
plt.plot(train_data, label='Train')
plt.plot(test_data, label='Test')
plt.plot(test_data.index, forecast, label='Forecast')
plt.legend()
plt.show()
```

Detecting Autocorrelation: The Durbin–Watson Test

- The Durbin-Watson (DW) test is a statistical test used to detect the presence of autocorrelation at lag 1 in the residuals from a regression analysis.
- Autocorrelation occurs when the residuals (errors) of a model are not independent of each other, which can lead to inefficient estimates and invalid statistical inferences.
- The test developed by Durbin andWatson (1950, 1951, 1971) is a very widely used procedure. This test is based on the assumption that the errors in the regression model are generated by a **first-order autoregressive process** observed at equally spaced time periods; that is,

$$\varepsilon_t = \phi\varepsilon_{t-1} + a_t,$$

- **Durbin-Watson Test Overview**
- **Purpose:**
- The DW test assesses whether the residuals from a linear regression model are serially correlated. This is crucial because the presence of autocorrelation violates the assumption of independent errors, which is fundamental to the validity of many regression techniques.
- **Test Statistic:**
- The Durbin-Watson statistic is defined as:



Test Statistic:

The Durbin-Watson statistic is defined as:

$$DW = \frac{\sum_{t=2}^{n}(e_t - e_{t-1})^2}{\sum_{t=1}^{n} e_t^2}$$

where:

- e_t is the residual at time t.
- n is the number of observations.

Interpretation:

- **DW ≈ 2**: No autocorrelation.
- **DW < 2**: Positive autocorrelation.
- **DW > 2**: Negative autocorrelation.

The DW statistic ranges from 0 to 4:

- A value close to 0 indicates strong positive autocorrelation.
- A value close to 4 indicates strong negative autocorrelation.
- A value around 2 suggests no autocorrelation.

Hypotheses:

- **Null Hypothesis (H0)**: There is no autocorrelation (rho = 0 or $\rho=0$).
- **Alternative Hypothesis (H1)**: There is autocorrelation (rho ≠0 or $\rho\neq0$).

Conducting the Durbin-Watson Test

Step-by-Step Procedure:

1. **Fit the Regression Model**: Fit your linear regression model to obtain residuals.
2. **Calculate Residuals**: Obtain the residuals from the regression model.
3. **Compute the Durbin-Watson Statistic**: Use the formula provided above or utilize statistical software to calculate the DW statistic.
4. **Interpret the Results**: Compare the DW statistic to the critical values or use the rule of thumb (around 2 indicates no autocorrelation).

Example Using Python

- Here is an example using Python with the stats models library to perform the Durbin-Watson test:

```
import numpy as np
import pandas as pd
import statsmodels.api as sm
from statsmodels.stats.stattools import durbin_watson

# Example data
np.random.seed(0)
X = np.random.randn(100, 2)
X = sm.add_constant(X)  # Adds a constant term to the predictor
y = X @ np.array([1, 0.5, -0.2]) + np.random.randn(100)  # Linear relation with noise

# Fit the linear regression model
model = sm.OLS(y, X).fit()

# Get the residuals
residuals = model.resid

# Perform the Durbin-Watson test
dw_statistic = durbin_watson(residuals)

print(f'Durbin-Watson statistic: {dw_statistic}')
```

Estimating the Parameters in Time Series Regression Models

- **Definition:**
- Estimating parameters in time series regression models involves determining the coefficients that quantify the relationship between the dependent variable and one or more independent variables, while also accounting for temporal dependencies.
- This process typically involves fitting a model to historical data and using statistical methods to derive the best estimates of the model parameters.

- **Purpose:**
- The primary purpose of estimating parameters in time series regression models is to:
- **Understand Relationships**: Quantify how predictor variables influence the response variable over time.
- **Forecasting**: Predict future values of the time series based on the historical data.
- **Control and Optimization**: Make informed decisions in fields like finance, economics, environmental studies, and engineering.

Steps in Estimating Parameters:

1. Model Selection

Choose an appropriate model based on the data characteristics:

- **ARIMA (AutoRegressive Integrated Moving Average)**: Suitable for univariate time series.
- **SARIMA (Seasonal ARIMA)**: Extends ARIMA to handle seasonality.
- **VAR (Vector AutoRegression)**: For multivariate time series.
- **SARIMAX (Seasonal ARIMA with eXogenous factors)**: ARIMA with additional explanatory variables.
- **Holt-Winters**: For capturing trends and seasonality in a time series.

2. Model Specification

Define the structure of the model, including the order of autoregression (p), differencing (d), and moving average (q) components for ARIMA models.

3. Parameter Estimation

Use historical data to estimate the parameters. This typically involves:

- **Maximum Likelihood Estimation (MLE)**: Commonly used for ARIMA models.
- **Least Squares**: Often used for models like regression and VAR.
- **Gradient Descent**: Used in more complex models and machine learning approaches.

4. Model Fitting

Fit the model to the data to find the parameter values that best explain the observed time series. This involves optimizing the chosen estimation criterion (e.g., minimizing the sum of squared errors).

5. Model Validation

Validate the model using diagnostic checks:

- **Residual Analysis**: Check if residuals are white noise.
- **Goodness-of-Fit**: Use metrics like AIC, BIC, and R-squared.
- **Cross-Validation**: Split the data into training and testing sets to evaluate model performance.

The Cochrane–Orcutt Method

Definition:

- The Cochrane–Orcutt method is an iterative procedure used to correct for serial correlation (autocorrelation) in the residuals of a linear regression model.
- This method is specifically designed to handle the first-order autoregressive process, denoted as AR(1), where the residuals from the regression model are correlated with their own previous values.

Purpose:

- The main purpose of the Cochrane–Orcutt method is to improve the efficiency of the parameter estimates in a linear regression model when the residuals exhibit autocorrelation.
- Autocorrelation in residuals violates the classical assumption of ordinary least squares (OLS) regression, which assumes that the residuals are independent and identically distributed.
- When autocorrelation is present, the OLS estimates are still unbiased but not efficient, meaning they do not have the smallest possible variance.

- **How to Use the Cochrane–Orcutt Method:**

1. **Initial Regression**:
2. **Estimate the Autoregressive Parameter (ρ/rho)**:
3. **Transform the Variables**:
4. **Iterate**:
5. **Final Model**:

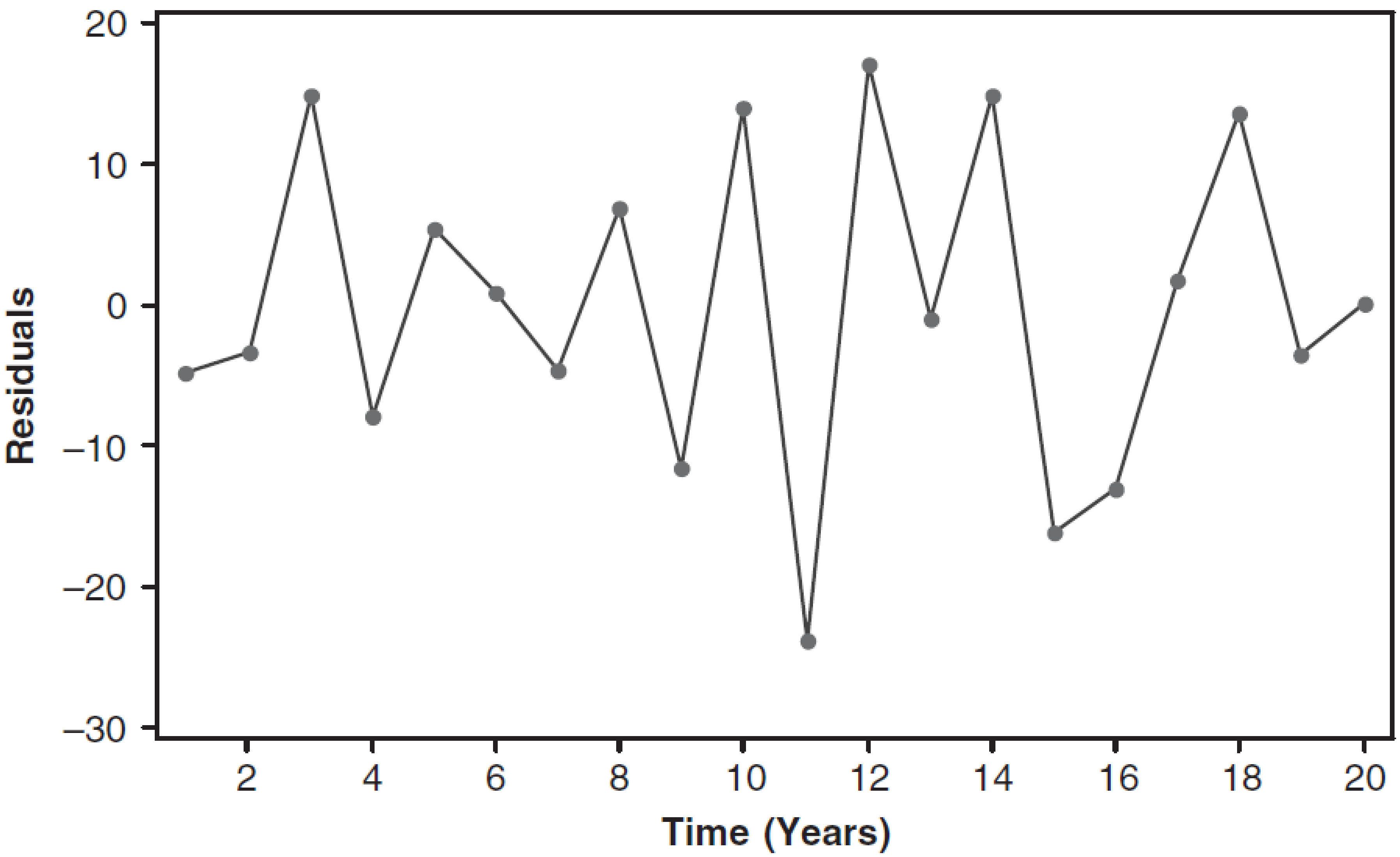
- **Key Points:**
- **Initial OLS Regression**: Perform OLS regression and obtain residuals.
- **Estimate ρ/rho**: Regress residuals on their lagged values to estimate ρ/rho.
- **Transform Variables**: Adjust original variables based on ρ/rho.
- **Iterate**: Refit the model and update ρ/rho until convergence.
- **Final Model**: Use the transformed model for interpretation and prediction.
- The Cochrane–Orcutt method is particularly useful when dealing with time series data where residuals are likely to be autocorrelated, thus improving the efficiency and reliability of the regression estimates.

Example

- Below is an example of how to implement and plot the Cochrane-Orcutt method for a sales model using Python.

Step-by-Step Implementation

1. **Simulate Sales Data**: We'll generate some synthetic sales data with a linear trend and autocorrelated residuals.
2. **Initial OLS Regression**: Fit an ordinary least squares (OLS) regression model to the sales data.
3. **Cochrane-Orcutt Transformation**: Apply the Cochrane-Orcutt transformation to correct for autocorrelation.
4. **Plot the Results**: Plot the original sales data, the initial OLS fit, and the Cochrane-Orcutt corrected fit.


Residuals
20
10
0
−10
−20
−30
2
4
6
8
10
12
14
16
18
20
Time (Years)

The Maximum Likelihood Approach in Regression Analysis

- **Definition:**
- The Maximum Likelihood (ML) approach is a method of estimating the parameters of a statistical model. In the context of regression analysis, it involves finding the parameter values that maximize the likelihood function, which measures how likely it is that the observed data were generated by the model with those parameters.

Purpose:

The Maximum Likelihood approach is used for several reasons:

1. **Efficiency**: ML estimators have desirable statistical properties, such as being asymptotically efficient, meaning they achieve the lowest possible variance among unbiased estimators as the sample size grows.
2. **Flexibility**: ML can be applied to a wide range of models, including those with non-normal error distributions and complex relationships.
3. **Inference**: ML allows for straightforward hypothesis testing and construction of confidence intervals.

Benefits of Maximum Likelihood:

1. **Asymptotic Properties**: ML estimators are consistent, asymptotically normal, and efficient.
2. **Adaptability**: It can be adapted to various types of data and models, including those with non-normal distributions.
3. **Inference**: Facilitates hypothesis testing and confidence interval construction through the likelihood ratio test, Wald test, and score test.

Example

- **How to Implement:** Here's an example using Python and the statsmodels library to implement ML estimation for a linear regression model:

Explanation:

1. Generate Data:

Simulate a dataset with a linear relationship between the predictors and the response variable.

2. Specify the Model:

•Use sm.add_constant(X) to add an intercept to the predictors.

•Use the OLS class from statsmodels to specify the linear regression model.

3. Fit the Model Using ML:

•The fit method with method='mle' fits the model using Maximum Likelihood Estimation.

4. Summary:

•The summary method provides detailed results, including parameter estimates, standard errors, and goodness-of-fit measures.

```
import numpy as np
import pandas as pd
import statsmodels.api as sm

# Generate example data
np.random.seed(0)
n = 100
X = np.random.randn(n, 2)
X = sm.add_constant(X)  # Adds a constant term to the predictor
beta = np.array([1, 0.5, -0.2])
y = X @ beta + np.random.randn(n)

# Fit the linear regression model using Maximum Likelihood
model = sm.OLS(y, X).fit(method='mle')

# Print the summary of the model
print(model.summary())
```

Concept of Forecasting and Prediction Intervals

- **Forecasting:**
- Forecasting involves making predictions about future values of a time series based on observed historical data. In the context of regression and time series analysis, forecasting is used to estimate future outcomes by applying the fitted model to new data points.

- **Prediction Intervals:**
- A prediction interval provides a range within which future observations are expected to fall, with a specified level of confidence. Unlike confidence intervals, which estimate the range for the mean response, prediction intervals account for both the uncertainty in the parameter estimates and the variability of the future observations.

Importance of Prediction Intervals:

Uncertainty Quantification: Prediction intervals provide a measure of the uncertainty associated with predictions, helping to understand the range of possible future values.

Risk Management: In fields like finance, economics, and engineering, prediction intervals help in assessing potential risks and making informed decisions.

Reliability: They offer a more reliable understanding of the expected variability in future outcomes compared to point forecasts alone.

ECONOMETRIC MODELS

- **Key Concepts in Econometric Models**
- Econometric models are statistical models that use economic theory and data to estimate relationships between economic variables.
- They play a crucial role in analyzing and interpreting economic phenomena, forecasting future trends, and guiding economic policy decisions.
- **Economic Theory**:
  - **Foundation**: Econometric models are grounded in economic theory, which provides the hypotheses about relationships between variables. For example, economic theory might suggest that higher education leads to higher income.
  - **Model Specification**: Econometricians translate theoretical relationships into mathematical models that can be estimated using data.

Key Types of Econometric Models

1. Linear Regression Models:

- **Simple Linear Regression**: Examines the relationship between two variables (one dependent and one independent) using a straight line.

$$y = \beta_0 + \beta_1 x + \epsilon$$

- **Multiple Linear Regression**: Extends the simple model to include multiple independent variables.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k + \epsilon$$

- **Usage**: Estimating relationships between economic indicators, such as how education affects wages.

2. Time Series Models:

- **ARIMA (AutoRegressive Integrated Moving Average)**: Combines autoregressive, differencing, and moving average components to model time series data.

$$y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 \epsilon_{t-1} + \cdots + \theta_q \epsilon_{t-q} + \epsilon_t$$

GARCH (Generalized Autoregressive Conditional Heteroskedasticity): Models volatility clustering in time series data.

Usage: Forecasting GDP, inflation rates, stock prices.

3. Panel Data Models:

- **Fixed Effects Model**: Controls for time-invariant characteristics by allowing each entity to have its own intercept.

$$y_{it} = \alpha_i + \beta x_{it} + \epsilon_{it}$$

- **Random Effects Model:** Assumes that individual-specific effects are random and uncorrelated with the independent variables.
- **Usage:** Analyzing data across countries or firms over time.

4. Instrumental Variables (IV) Models:

- Used when there is endogeneity (correlation between the regressors and the error term) to obtain unbiased estimates.
- **Two-Stage Least Squares (2SLS)**: A common method for estimating IV models.
- **Usage**: Estimating the effect of education on earnings when education may be endogenous.

5. Simultaneous Equations Models:

- **Structural Equation Models**: Contain multiple interdependent equations representing different economic relationships.

$$y_1 = \beta_{10} + \beta_{11} y_2 + \epsilon_1$$

$$y_2 = \beta_{20} + \beta_{21} y_1 + \epsilon_2$$

- **Usage**: Modeling systems where variables simultaneously influence each other, such as supply and demand models.

6. Logit and Probit Models:

- **Logit Model**: Used for binary dependent variables where the outcome is a probability

$$P(y=1) = \frac{1}{1+\exp(-(\beta_0+\beta_1 x))}$$

- **Probit Model:** Similar to logit but uses the cumulative normal distribution function.
- **Usage:** Modeling binary choices such as purchasing decisions or labor force participation.

7. Dynamic Models:

- **Vector Autoregression (VAR)**: Models the interdependencies among multiple time series.

$$\mathbf{y}_t = A_1\mathbf{y}_{t-1} + \cdots + A_p\mathbf{y}_{t-p} + \mathbf{u}_t$$

- **Usage**: Analyzing the joint behavior of multiple economic variables over time.

Applications of Econometric Models

- **Policy Analysis**: Assessing the impact of fiscal and monetary policies on economic growth, inflation, and employment.
- **Economic Forecasting**: Predicting future economic indicators such as GDP, inflation rates, and unemployment.
- **Market Analysis**: Understanding and predicting consumer behavior, market demand, and pricing strategies.
- **Risk Management**: Quantifying and managing financial risks using models of asset returns and volatility.

Example of Econometric Model Application

Scenario: Estimating the Impact of Education on Income

Objective: Determine how an additional year of education affects annual income.

Model Specification:

$$\text{Income}_i = \beta_0 + \beta_1 \text{Education}_i + \epsilon_i$$

Data:

Dependent Variable: Income (annual income of individuals)

Independent Variable: Education (years of education)

Steps:

Collect Data: Gather data on income and education from a survey or dataset.

Fit Model: Use OLS regression to estimate β0\beta_0β0 and β1\beta_1β1.

Interpret Results: Analyze the estimated coefficient β1\beta_1β1 to understand the impact of education on income.

Predict and Forecast: Use the model to predict income for different levels of education.

Estimation Techniques:

•**Ordinary Least Squares (OLS)**: Minimizes the sum of squared residuals to estimate parameters in linear regression models.

•**Maximum Likelihood Estimation (MLE)**: Estimates parameters by maximizing the likelihood function, often used in more complex models.

•**Two-Stage Least Squares (2SLS)**: Used in instrumental variables estimation to handle endogeneity.

Model Diagnostics:

•**Residual Analysis**: Examines the residuals (errors) to check for patterns that might indicate problems with the model, such as autocorrelation or heteroskedasticity.

•**Goodness-of-Fit**: Measures how well the model explains the variability in the dependent variable, often assessed using R-squared or adjusted R-squared.

•**Hypothesis Testing**: Tests hypotheses about model parameters using t-tests, F-tests, and likelihood ratio tests.

Forecasting and Prediction:

- **Forecasting**: Uses historical data and the fitted model to make predictions about future values.
- **Prediction Intervals**: Provides a range within which future observations are expected to fall, considering both model uncertainty and inherent variability.

Application and Interpretation:

- **Policy Analysis**: Assesses the impact of economic policies on various outcomes, such as how a tax change affects consumer spending.
- **Risk Management**: Models financial risk and volatility to manage exposure and make informed decisions.
- **Market Analysis**: Understands consumer behavior, market trends, and pricing strategies.

R Commands

1. Linear Regression To perform linear regression, you use the lm() function, which fits a linear model.

- Example

```
# Load necessary library
library(tidyverse)

# Load example dataset
data(mtcars)

# Fit a linear model
model <- lm(mpg ~ wt + hp, data = mtcars)

# Summary of the model
summary(model)
```

2. Multiple Linear Regression

- Similar to simple linear regression but with multiple predictors.
- **Example:**

```
# Fit a multiple linear regression model
model_multi <- lm(mpg ~ wt + hp + qsec, data = mtcars)

# Summary of the model
summary(model_multi)
```

3. Logistic Regression For binary outcomes, use the **glm()** function with **family = binomial.**

- Example:

```
# Simulate binary outcome variable
mtcars$am <- as.factor(mtcars$am)

# Fit a logistic regression model
logit_model <- glm(am ~ wt + hp, data = mtcars, family =
binomial)

# Summary of the model
summary(logit_model)
```

4. Time Series Analysis For time series data, you can use the ts() function to create a time series object and auto.arima() from the forecast package for ARIMA modeling.

- Example:

```
# Load the forecast package
library(forecast)
# Create a time series object
ts_data <- ts(mtcars$mpg, start = c(1970, 1), frequency = 12)
# Fit an ARIMA model
arima_model <- auto.arima(ts_data)
# Summary of the model
summary(arima_model)
# Forecasting
forecast_values <- forecast(arima_model, h = 10)
plot(forecast_values)
```

5. Panel Data Models For panel data, you might use the plm package. It allows for fixed effects or random effects models.

- **Example**:

```
# Load the plm package
library(plm)
# Simulate panel data
data("Grunfeld", package = "plm")
# Fit a fixed effects model
fe_model <- plm(inv ~ value + capital, data = Grunfeld, model = "within")
# Summary of the model
summary(fe_model)
# Fit a random effects model
re_model <- plm(inv ~ value + capital, data = Grunfeld, model = "random")
# Summary of the model
summary(re_model)
```

6. Handling Multicollinearity: To check for multicollinearity, you can use the **vif()** function from the **car** package.

- Example:

```
# Load the car package
library(car)

# Check for multicollinearity
vif(model_multi)
```

7. Checking Model Diagnostics: You can use diagnostic plots with the **plot()** function on the model object.

- Example:

```
# Diagnostic plots for a linear model
par(mfrow = c(2, 2))
plot(model_multi)
```

8. Model Refinement

- Refine models by adding or removing variables, and comparing models using metrics like AIC or BIC.
- **Example:**

```
# Compare models using AIC
model1 <- lm(mpg ~ wt + hp, data = mtcars)
model2 <- lm(mpg ~ wt + hp + qsec, data = mtcars)

AIC(model1, model2)
```

9. Generalized Linear Models (GLM)

Definition: Generalized linear models extend linear models to allow for response variables that have error distribution models other than a normal distribution. It includes logistic, Poisson, and other models.

Example:

```
# Fit a generalized linear model (e.g., Poisson)
glm_model <- glm(counts ~ wt + hp, data = mtcars, family =
poisson)

# Summary of the model
summary(glm_model)
```

10. Negative Binomial Regression

- **Definition:** Negative binomial regression is used for count data with over dispersion (when variance exceeds the mean). It extends Poisson regression by adding a parameter to account for this over dispersion.

```
# Load the MASS package for negative binomial regression
library(MASS)

# Fit a negative binomial regression model
nb_model <- glm.nb(count ~ x1 + x2, data = dataset)

# Summary of the model
summary(nb_model)
```

- **Additional Resources**

tidyverse: A collection of R packages for data manipulation and visualization.

forecast: For time series forecasting.

plm: For panel data analysis.

car: For regression diagnostics.
plm: For panel data models like fixed effects and random effects.
AER: For instrumental variables and other econometric methods.
survival: For survival analysis models like the Cox model.
quantreg: For quantile regression.
MASS: For negative binomial regression.