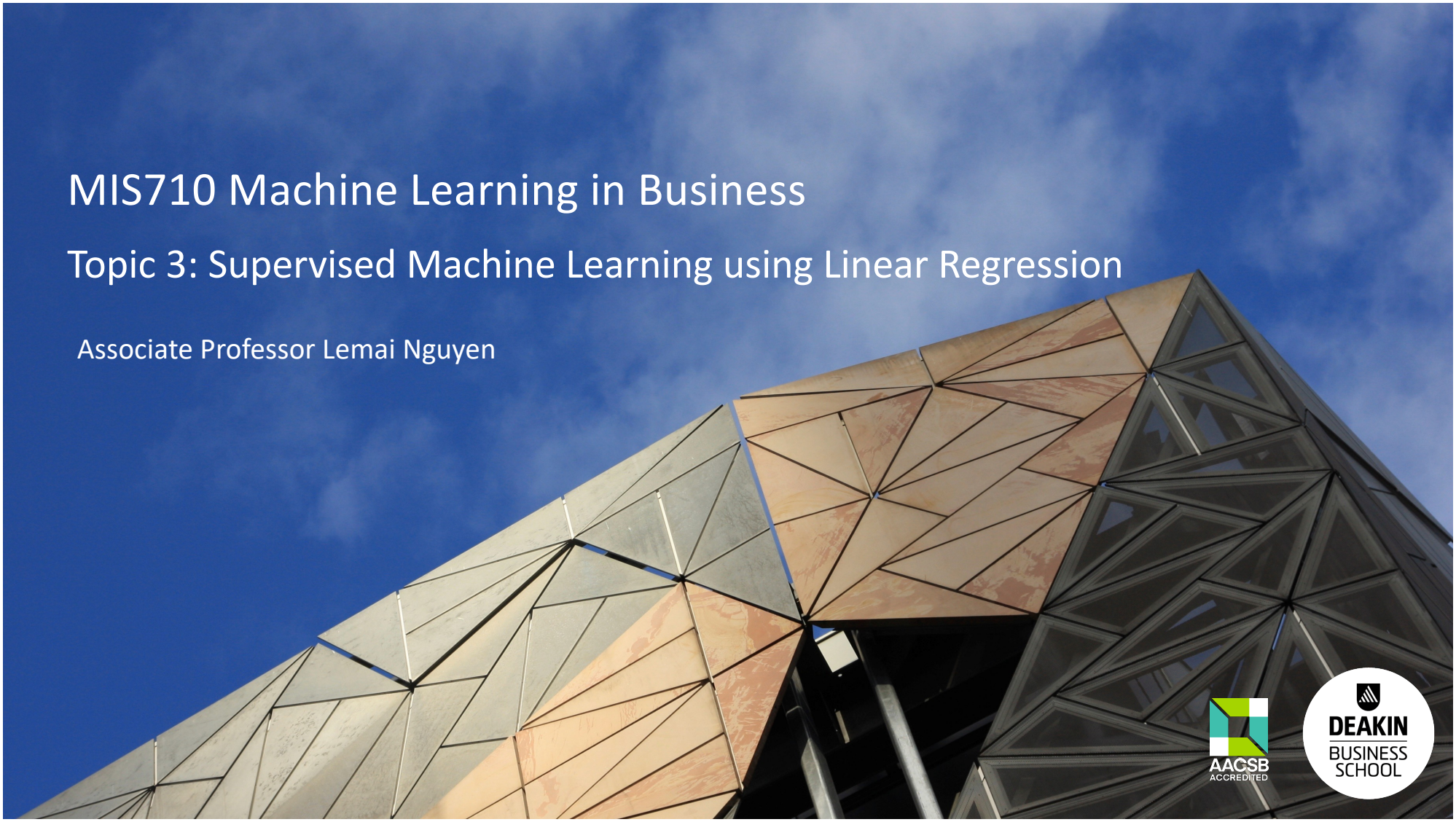
MIS710 Machine Learning in Business

Topic 3: Supervised Machine Learning using Linear Regression

Associate Professor Lemai Nguyen



Topic 3: Supervised Machine Learning – Linear Regression

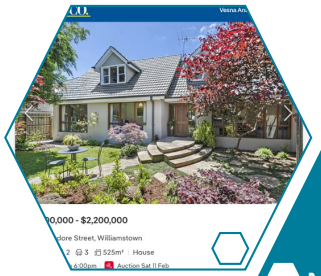
Check the unit site for REGULAR updates and upcoming consultation sessions:

- Week 3 Friday 26th July 3pm AEST (10.30am IST)

Lecture 3 and Lab 3 are critical to complete A1

Check A1 rubric

Assignment extension requests are processed centrally, beyond the UNIT TEAM



Estimation

ML in Business Framing

Revisiting the business case using BACCM

Need: a problem or opportunity to address;

e.g., to bring vendors and buyers to sign contracts

Solution: a way to satisfy one or more needs;

e.g., estimate prices based on previous executed contracts

Value: the worth, importance, or usefulness of something to a stakeholder within a context;

e.g., increased #listed properties, #potential buyers, #new contracts, improved vendor and buyer experience

Change: act of transformation in response to needs;

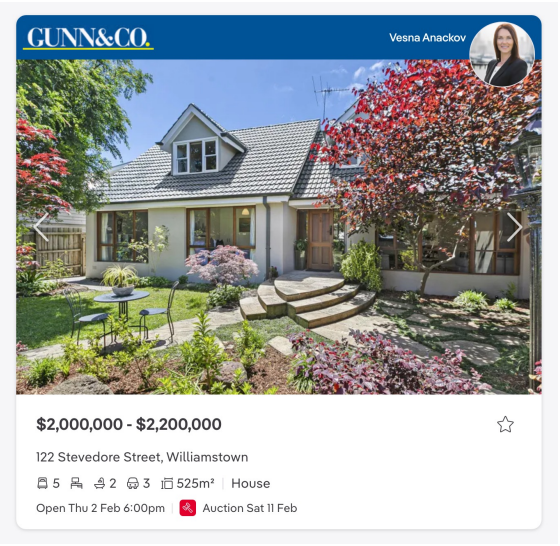
e.g., augmentation of house listing and price estimation decision process

Stakeholder: a group or individual with a relationship to the change, the need, or the solution;

e.g., agents, vendors, buyers

Context: the circumstances that influence, are influenced by, and provide understanding of the change;

e.g., high competition and dynamics of real estate market and interest rates, new developments in the areas, current data and systems, decision culture and processes in the company



https://www.realestate.com.au/property-house-vic-williamstown-141027400

ML in Business Framing

Business needs

- High competition in the real estate market
- Estimate a house price accurately will help satisfy the vendors and attract buyers

Data

- Sources: Records of past executed contracts
  - External data?
- **Data content: House attributes (predictors) and sold price (target)**

Analytics (ML)

- **Supervised ML, estimation**
- Build and test a **regression model** that best fit the house (and market) attributes and sold prices

Deployment

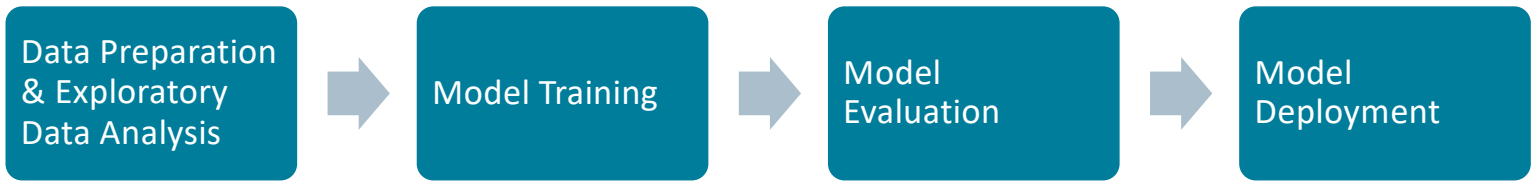
- List and advertise new properties for sale
- Information for contract negotiation

ML Type and Problem Framing

- Business Context: House selling
- Business Problem: To estimate the house price accurately to facilitate contract negotiations
- Business Data: Historical datasets of previous contracts: house attributes and sold prices (label)

ML approach:

- Machine Learning type: Supervised
- Problem: Estimation (how much – predict a continuous numeric value)



Let's use a Kaggle dataset…

independent variables, Attributes, Features, Predictors (X)

dependent variable, Target, label (y)

Data points
Observations
Examples

area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterhe	aircondition	parking	prefarea	furnishingstatus	price
7420	4	2	3	yes	no	no	no	yes	2	yes	furnished	13300000
8960	4	4	4	yes	no	no	no	yes	3	no	furnished	12250000
9960	3	2	2	yes	no	yes	no	no	2	yes	semi-furnished	12250000
7500	4	2	2	yes	no	yes	no	yes	3	yes	furnished	12215000
7420	4	1	2	yes	yes	yes	no	yes	2	no	furnished	11410000
7500	3	3	1	yes	no	yes	no	yes	2	yes		10850000
8580	4	3	4	yes	no	no	no	yes	2	yes	semi-furnished	10150000
16200	5	3	2	yes	no	no	no	no	0	no	unfurnished	10150000
8100	4	1	2	yes	yes	yes	no	yes	2	yes	furnished	9870000
	3	2	4	yes	yes	no	no	yes	1	yes	unfurnished	9800000
13200	3	1	2	yes	no	yes	no	yes	2	yes	furnished	9800000
	4	3	2	yes	yes	yes	yes	no	2	no	semi-furnished	9681000
6550	4	2	2	yes	no	no	no	yes	1	yes	semi-furnished	9310000
3500	4	2	2	yes	no	no	yes	no	2	no	furnished	9240000
	3	2	2	yes	no	no	no	no	0	yes	semi-furnished	9240000
	4	1	2		no		no		2			9100000
6600	4	2	2	yes	yes	yes	no	yes	1	yes	unfurnished	9100000
8500	3	2	4	yes	no	no	no	yes	2	no	furnished	8960000
4600	3	2	2	yes	yes	no	no	yes	2	no	furnished	8890000
6420	3	2	2		no	no	no	yes	1	yes	semi-furnished	8855000
	3	1	2	yes	no	yes	yes	no	2	no	semi-furnished	8750000
7155	3	2	1	yes	yes	yes	no	yes	2	no	unfurnished	8680000
8050	3	1	1	yes	yes	yes	no	yes	1	no	furnished	8645000
	3	2	2	yes	yes	yes	no	yes	1	no	furnished	8645000
8800	3	2	2	yes	no	no	no	yes	2	no	furnished	8575000
6540	4	2	2	yes	yes	yes	no	yes	2	yes	furnished	8540000

https://www.kaggle.com/datasets/yasserh/housing-prices-dataset

Understanding data

13 columns * 454 records:

1. area (535) floating point number
2. bedrooms integer number
3. bathrooms integer number
4. stories integer number
5. parking integer number

1. mainroad (539) boolean
2. guestroom boolean
3. basement boolean
4. hotwaterheating boolean
5. airconditioning boolean
6. prefarea boolean

1. furnishingstatus (538) string

Target/label

1. price floating point number

Pre-process and EDA

- Size
- Data types
- Detect missing data
- Summarise numerical and categorical data
- Visualise data distributions
- Detect outliers
- Handle missing data
- Explore relationships among variables
- Scale, normalise data
- Convert data types
- Transform or compute data

```
# Replace erroneous release years with NaN
records['Released_Year'] = records['Released_Year'].apply(lambda x: x if x > 0 else pd.NA)
```

👉 Univariate, bivariable and multi-variate analyses

Low outliers < Q_1 - 1.5 * IQR
High outliers > Q_3 + 1.5 * IQR

Imagine a simple data split

Attributes (X)

Label (y)

Data points

area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterhe	aircondition	parking	prefarea	furnishingstatus	price
7420	4	2	3	yes	no	no	no	yes	2	yes	furnished	13300000
8960	4	4	4	yes	no	no	no	yes	3	no	furnished	12250000
9960	3	2	2	yes	no	yes	no	no	2	yes	semi-furnished	12250000
7500	4	2	2	yes	no	yes	no	yes	3	yes	furnished	12215000
7420	4	1	2	yes	yes	yes	no	yes	2	no	furnished	11410000
7500	3	3	1	yes	no	yes	no	yes	2	yes		10850000
8580	4	3	4	yes	no	no	no	yes	2	yes	semi-furnished	10150000
16200	5	3	2	yes	no			no	0	no	unfurnished	10150000
8100	4	1	2	yes	yes			yes	2	yes	furnished	
	3	2	4	yes	yes			yes	1	yes	unfurnished	
13200	3	1	2	yes	no			yes	2	yes	furnished	9800000
	4	3	2	yes	yes	yes	yes	no	2	no	semi-furnished	9681000
6550	4	2	2	yes	no	no	no	yes	1	yes	semi-furnished	9310000
3500	4	2	2	yes	no	no	yes	no	2	no	furnished	9240000
	3	2	2	yes	no	no	no	no	0	yes	semi-furnished	9240000
	4	1	2		no		no		2			9100000
6600	4	2	2	yes	yes	yes	no	yes	1	yes	unfurnished	9100000
8500	3	2	4	yes	no	no	no	yes	2	no	furnished	8960000
4600	3	2	2	yes	yes	no	no	yes	2	no	furnished	8890000
6420	3	2	2		no	no	no	yes	1	yes	semi-furnished	8855000
	3	1	2	yes	no	yes	yes	no	2	no	semi-furnished	8750000
7155	3	2	1	yes	yes	yes	no	yes	2	no	unfurnished	8680000
8050	3	1	1	yes	yes	yes	no	yes	1	no	furnished	8645000
	3	2	2	yes	yes	yes	no	yes	1	no	furnished	8645000
8800	3	2	2	yes	no	no	no	yes	2	no	furnished	8575000
6540	4	2	2	yes	yes	yes	no	yes	2	yes	furnished	8540000

X_train

y_train

X_test

y_test

Training:
Train f to minimize
(y_train – y_pred)

where
y_pred=f(X_train)

Random sampling

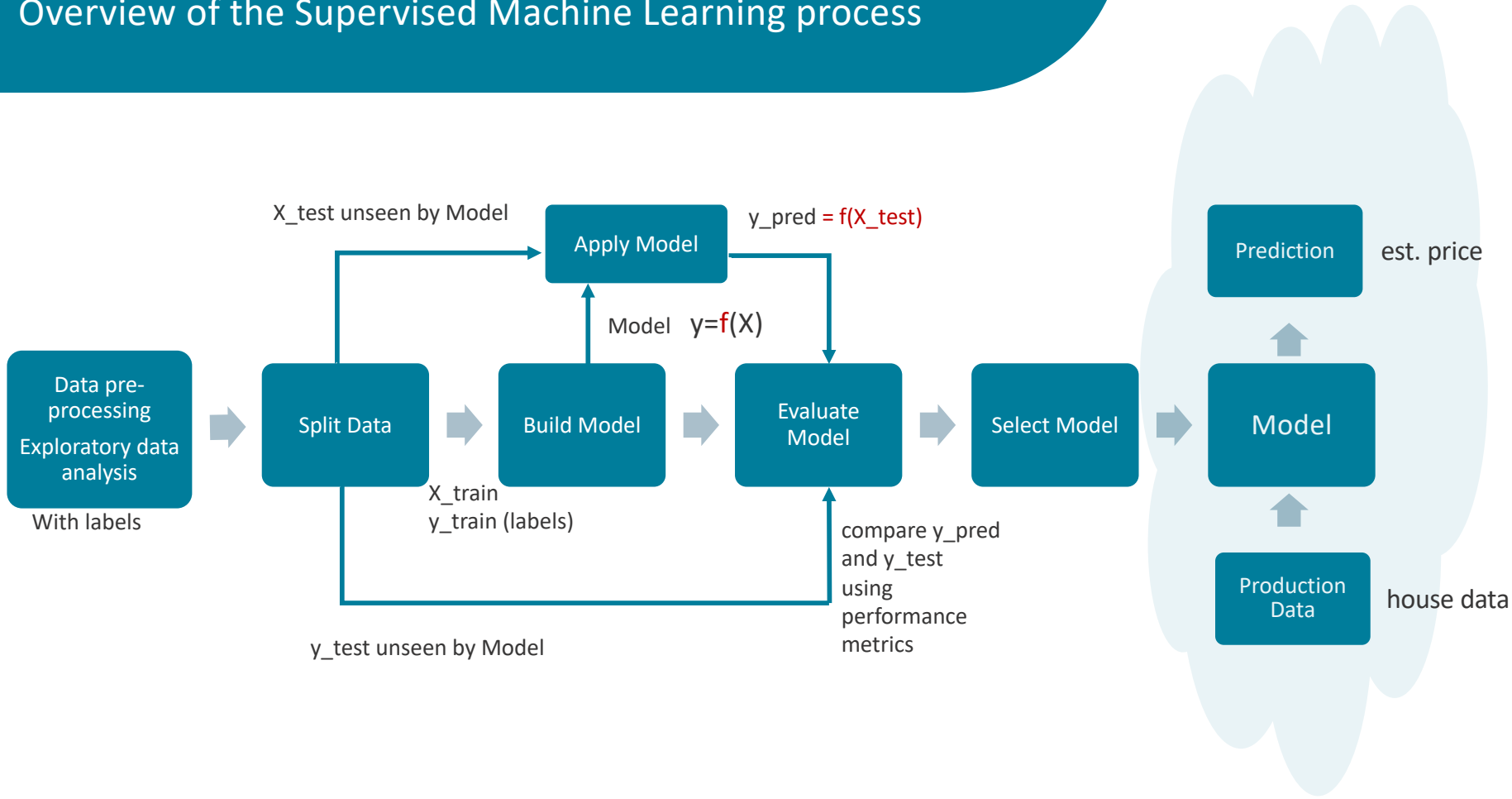
Evaluation:
is f good enough that
(y_test – y_pred) is acceptable,
where
y_pred=f(X_test)

```
#feature selection
features=['area']
X=records[features]

#specify the label
y=records['price']
```

- Large datasets: 70% and 30% works well;
- Small datasets: 80% or 8% for training may be needed;

```
# Import train_test_split function
from sklearn.model_selection import train_test_split

# Split dataset into a training dataset 70% and a test dataset 30%
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=2024)
```

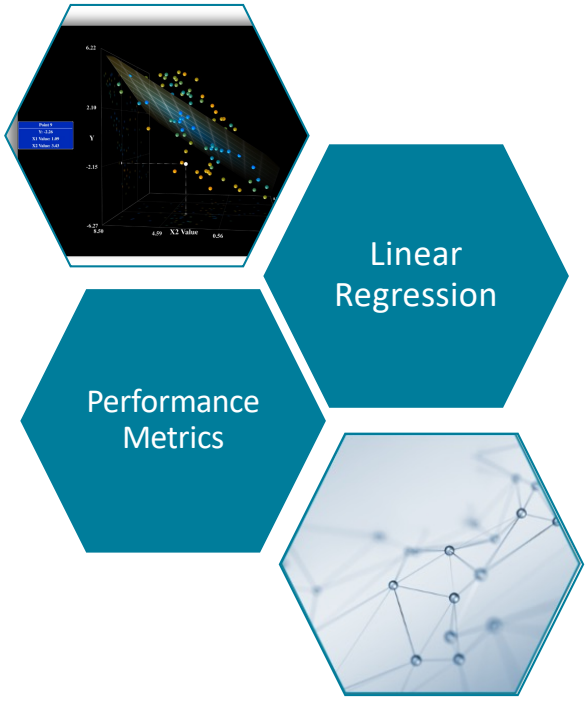
Overview of the Supervised Machine Learning process



Are you keeping pace?

https://www.mentimeter.com

Code: 18 11 35 5



Prerequisite: repeat Lab 2 steps to cleanse, handle missing data, convert data types, explore and visualise data on this new dataset.

- Simple linear regression
- Multiple linear regression
- Performance metrics

Supervised Machine Learning: Regression

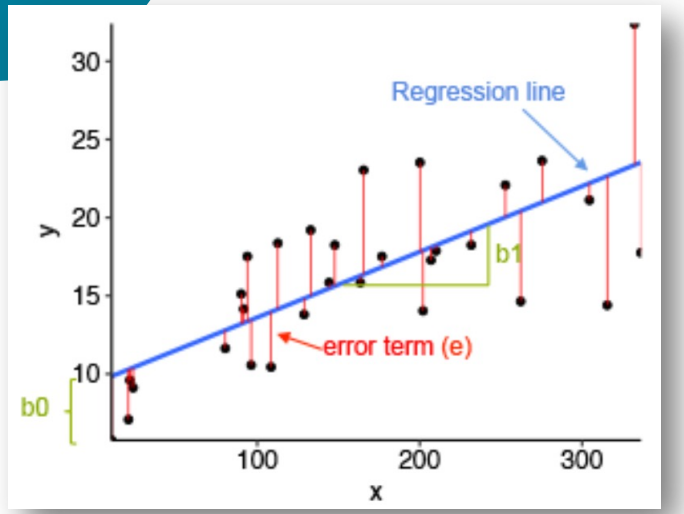
Basic understandings

- Purpose: to predict a real number (continuous number)
- Regression analysis models the relationship between a dependent variable and one or more independent variables
- The dependent variable – label - is available
- Examples:
  - Predict people's income; business revenue, sale, price; hospital length of stay; etc

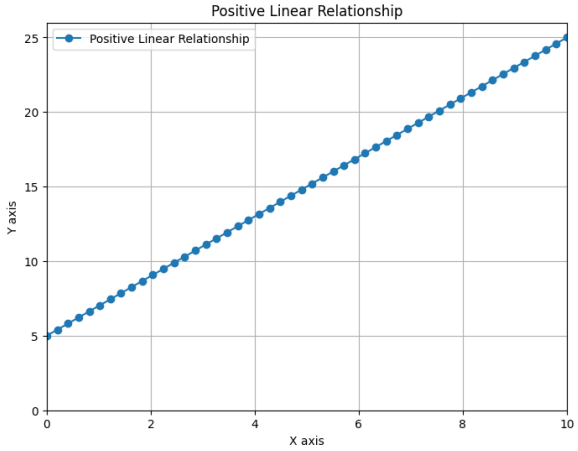
Supervised Machine Learning: Regression

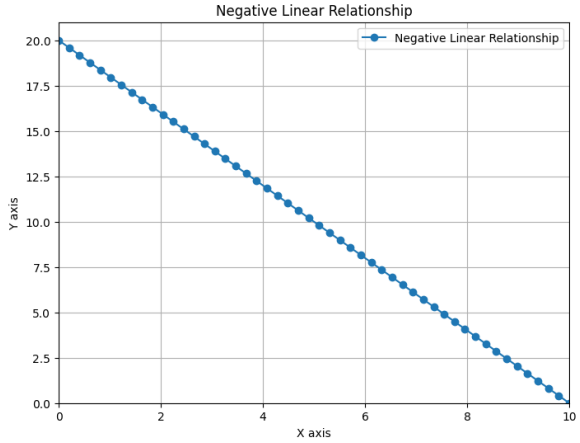
Types

- Simple Regression: the linear relationship between one independent variable and one dependent variable.



source: http://www.sthda.com/english/articles/40-regression-analysis/167-simple-linear-regression-in-r/



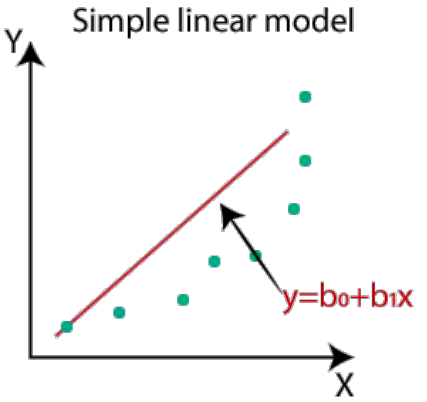


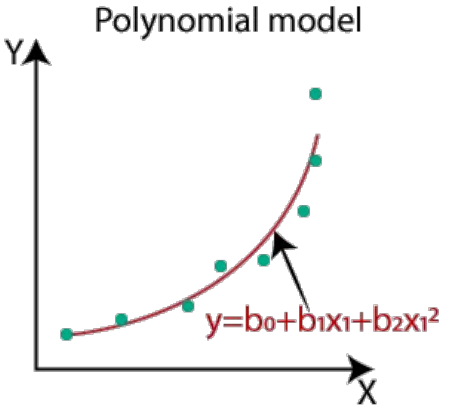
Supervised Machine Learning: Regression

Types

- Simple Regression: the linear relationship between one independent variable and one dependent variable.
- Multiple Regression: the linear relationship between two or more independent variables and one dependent variable.
- Polynomial Regression: the relationship between one independent variable and one dependent variable using an n^{th} degree polynomial function.
- Polynomial Multiple Regression: the relationship between two or more independent variables and one dependent variable using an n^{th} degree polynomial function.

Lee, 2014, ch 6



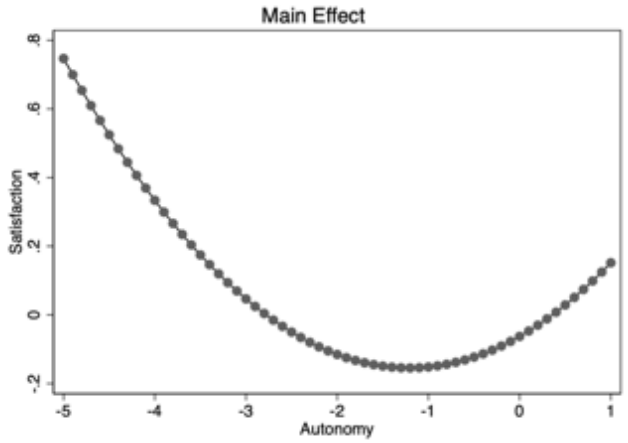


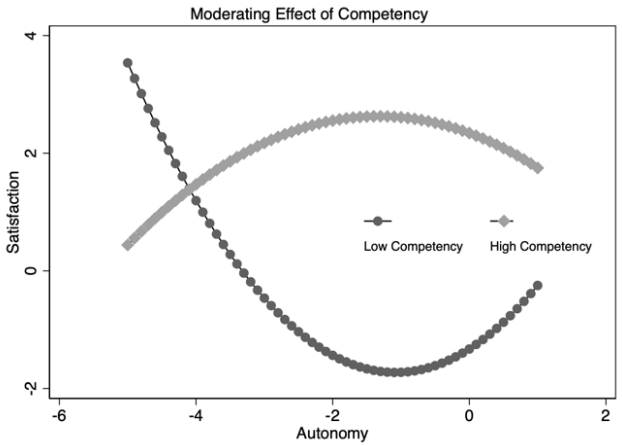
https://medium.com/analytics-vidhya/understanding-polynomial-regression-5ac25b970e18

Supervised Machine Learning: Regression

Types

- Quadratic Regression - Polynomial Multiple Regression: the relationship between two or more independent variables and one dependent variable using an n^{th} degree polynomial function when n=2.





Nguyen et al., 2024

https://aisel.aisnet.org/pacis2024/track13_hcinteract/track13_hcinteract/4/

Simple Linear Regression

The formula and visualisation

$$\hat{y} = f(x) = \boldsymbol{b_0} + \boldsymbol{b_1 x}$$

$\boldsymbol{b_0}$ is the intercept of the line

$\boldsymbol{b_1}$ is the slope of the line

x=independent variable/predictor

y=dependent variable/label

$\hat{y}$ = predicted/estimated value

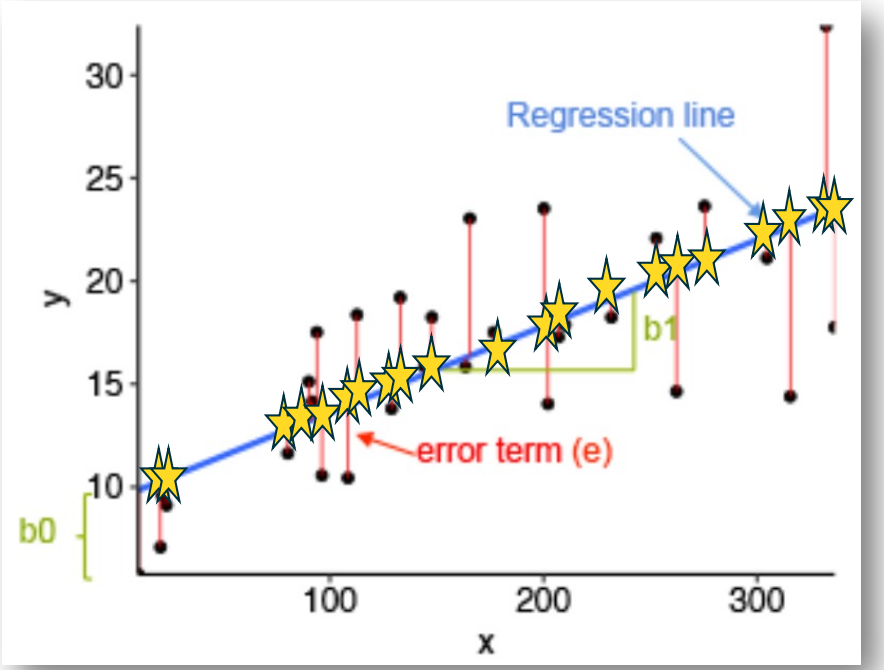


Image source: http://www.sthda.com/english/articles/40-regression-analysis/167-simple-linear-regression-in-r/

Simple Linear Regression

$\hat{y} = b_0 + b_1 x$

An example of linear models

(estimated) Sales $= \$10,000 + \$31,565$ **(\$1K_Ad_spend)**

For each additional \$1000 dollar spent on Advertising, Sales will increase approximately by \$31,565.

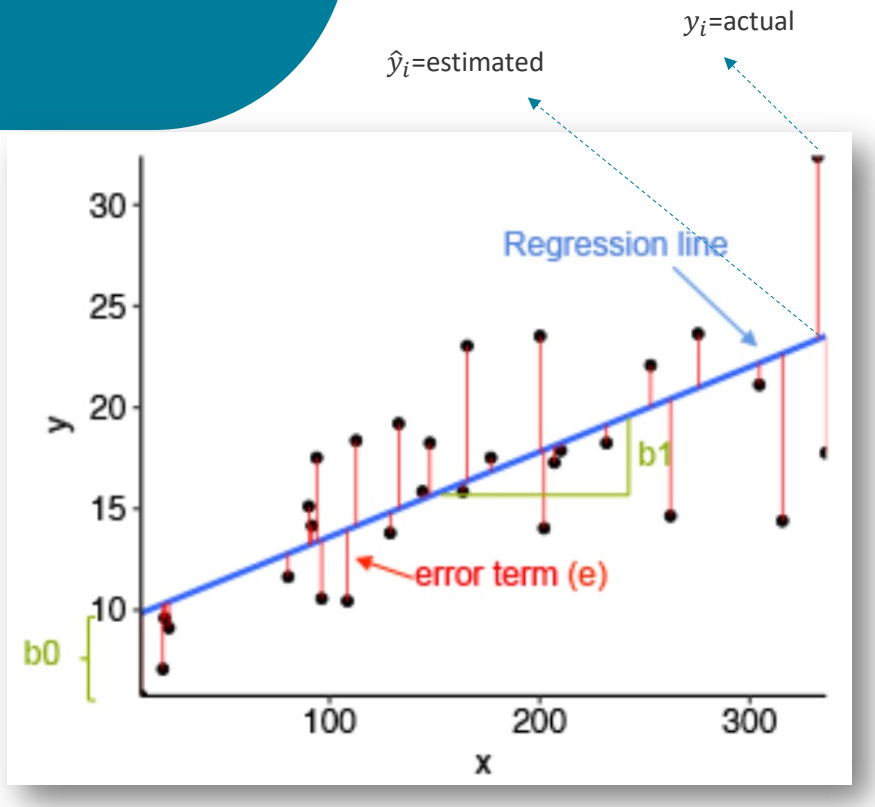


Image source: http://www.sthda.com/english/articles/40-regression-analysis/167-simple-linear-regression-in-r/

Model evaluation metrics: goodness of fit

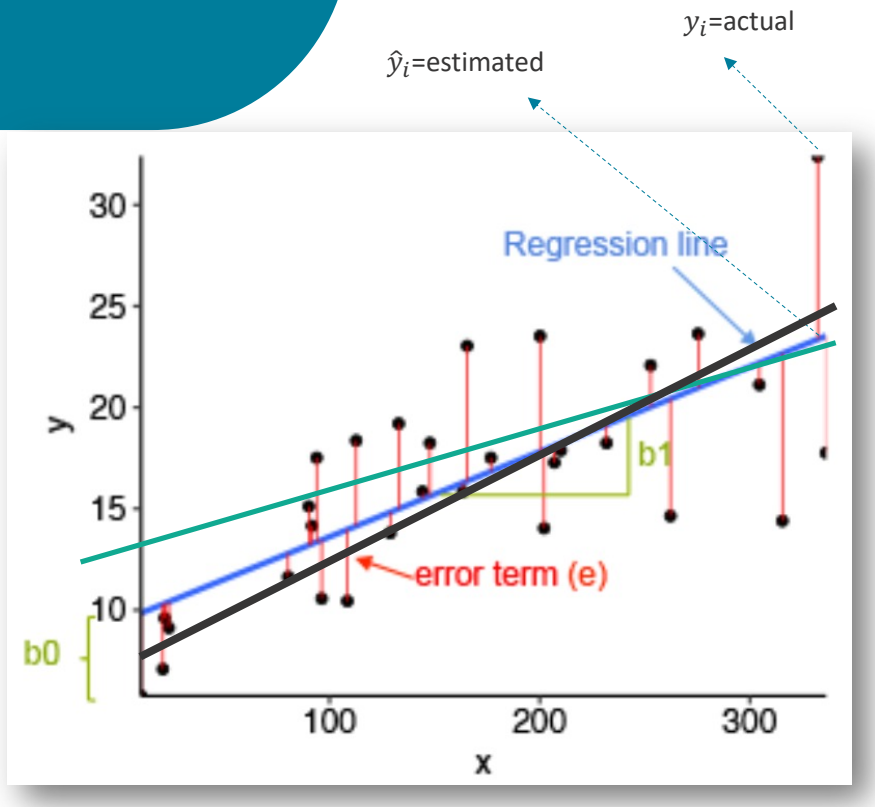
$$\hat{y} = \boldsymbol{b_0} + \boldsymbol{b_1 x}$$

Goodness of fit

Coefficient of determination R^2 summarises the proportion of the variance for a dependent variable that can be explained by independent variables in a regression model.

$$R^2 = 1 - \frac{SS_{res}}{SS_{tol}}$$

The ratio of the explained variance to the total variance.



Adapted from: http://www.sthda.com/english/articles/40-regression-analysis/167-simple-linear-regression-in-r/

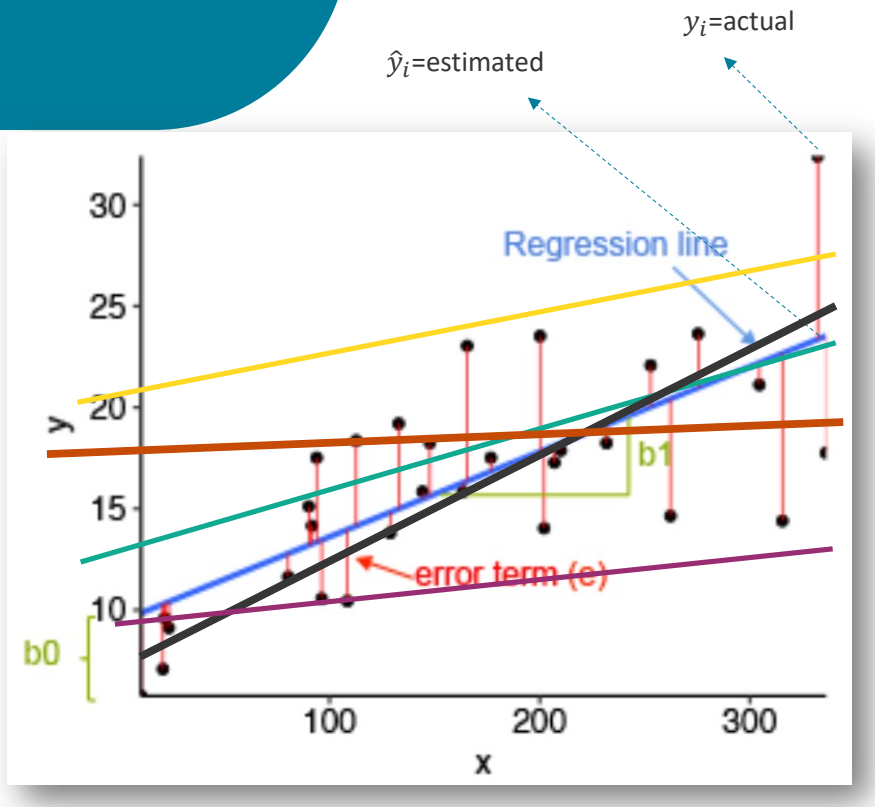
Model evaluation metrics: goodness of fit

$$\hat{y} = \boldsymbol{b_0} + \boldsymbol{b_1 x}$$

Goodness of fit

Coefficient of determination R^2 summarises the proportion of the variance for a dependent variable that can be explained by independent variables in a regression model.

$$R^2 = 1 - \frac{SS_{res}}{SS_{tol}}$$



Adapted from: http://www.sthda.com/english/articles/40-regression-analysis/167-simple-linear-regression-in-r/

SSres = sum of squared residuals - difference between the actual values and the predicted values of the dependent variable(s).
SStot = total sum of squares, which is the sum of the squared distances between the actual values and the mean of the dependent variable.

Model evaluation metrics: loss functions

A loss function measures of the error between the estimated value of the dependent variable using a model and the actual value:

at the datapoint i: $y_i - \hat{y}_i$

MAE and RMSE/RMSD functions aggregate the estimation errors for all data points:

Mean absolute error (MAE)

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n}$$

Mean square error

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

Root mean square error/deviation (RMSE/RMSD)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n}}$$

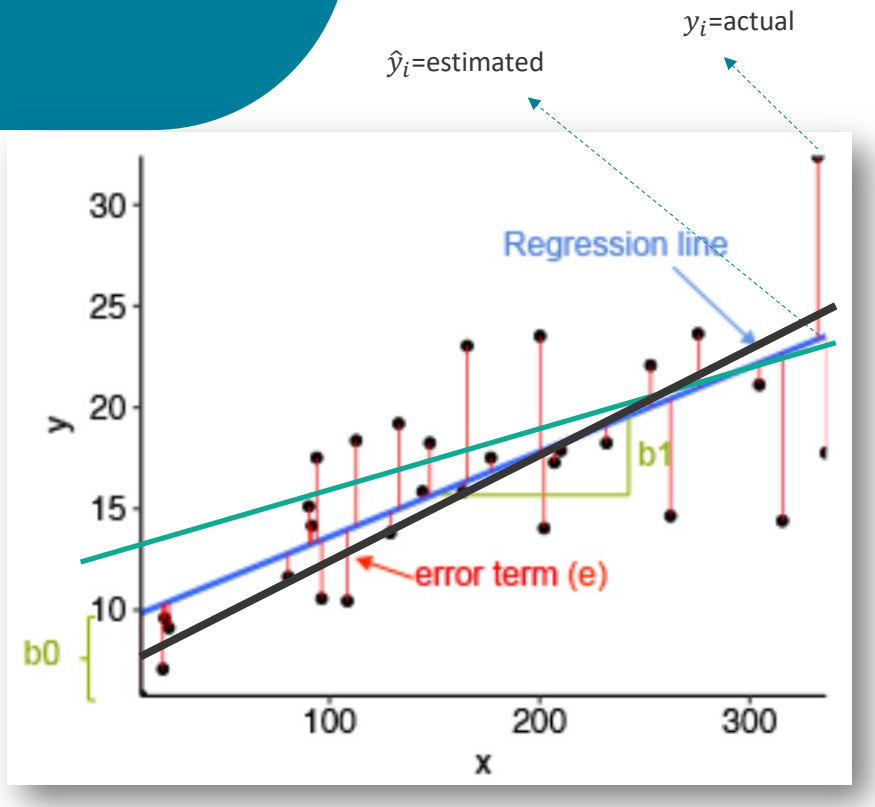


Image source: http://www.sthda.com/english/articles/40-regression-analysis/167-simple-linear-regression-in-r/

Model evaluation metrics: loss functions

A loss function measures of the error between the estimated value of the dependent variable using a model and the actual value:

at the datapoint i: $y_i - \hat{y}_i$

squared error loss: $(\boldsymbol{y_i - \hat{y}_i})^2$

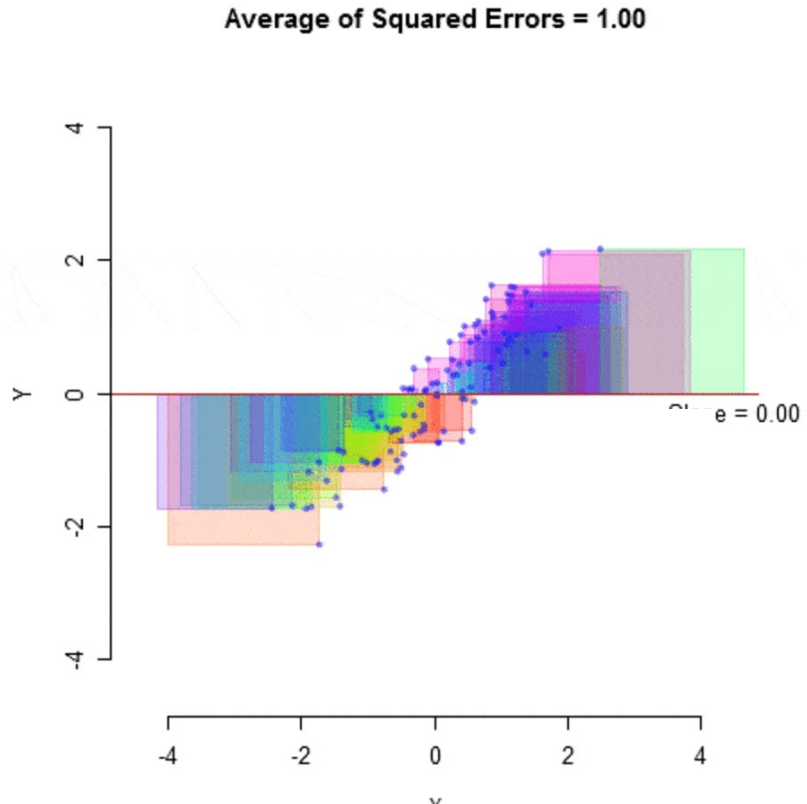
A cost function aggregates the loss over the entire training dataset.

Mean square error:

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

In scikit learn implementation: Ordinary Least Squares (OLS) method is used to minimise - residual squared errors (RSS), $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

i.e. effectively minimising MSE

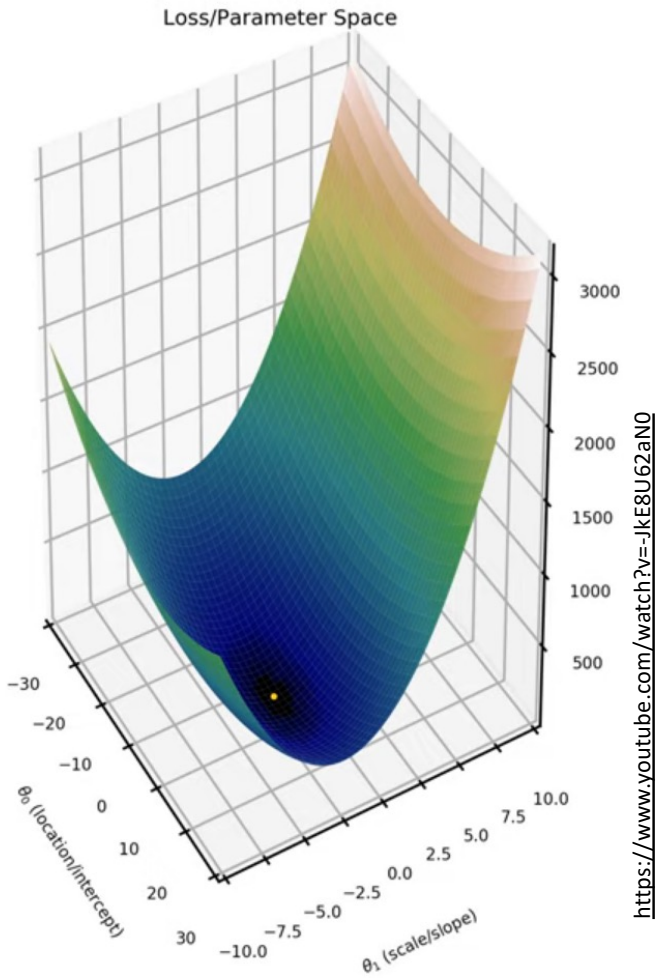


https://www.linkedin.com/feed/update/urn:li:activity:7087998400339587074?utm_source=share&utm_medium=member_desktop

The learning process

$$\hat{y} = b_0 + b_1 x$$

- The goal is to find the optimal values of the coefficients b_0 and b_1 that minimizes the error between the actual and estimated values of the dependent variable.
- Gradient descent is a learning technique which iteratively updates the coefficients in the direction of steepest descent of the cost function.



https://www.youtube.com/watch?v=-JkE8U62aN0

```
#feature selection
features=['area']
X=records[features]

#specify the label
y=records['price']
```

```
# Import train_test_split function
from sklearn.model_selection import train_test_split

# Split dataset into a training dataset 70% and a test dataset 30%
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=2024)
```

```
records[['area','price']]
```

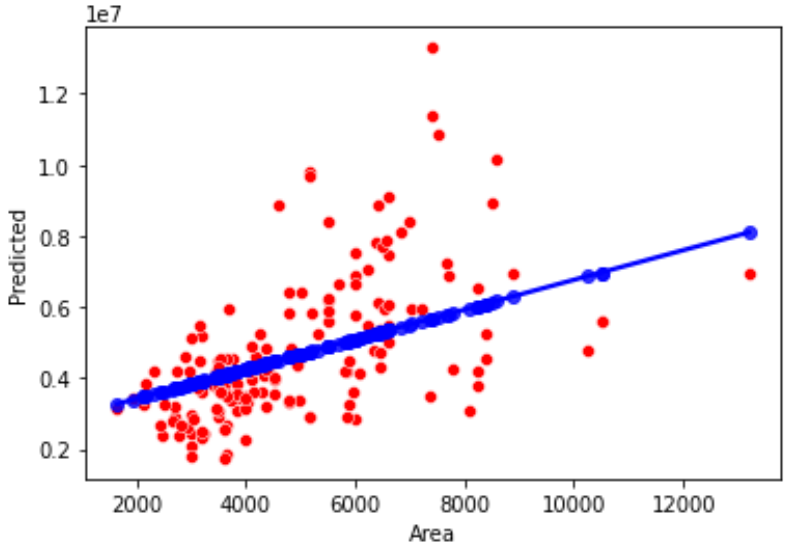
	area	price
0	7420.0	13300000
1	8960.0	12250000
2	9960.0	12250000
3	7500.0	12215000
4	7420.0	11410000
...	...	...
540	3000.0	1820000
541	2400.0	1767150
542	3620.0	1750000
543	2910.0	1750000
544	3850.0	1750000

545 rows × 2 columns

Simple Linear Regression: example

Predicting house price based on area

```
#import linear_model
from sklearn import linear_model

#create a linear_model object
reg = linear_model.LinearRegression()

# Train the model with the training dataset
reg=reg.fit(X_train, y_train)

#Make predictions for the test dataset
y_pred = reg.predict(X_test)
```




```
#Visualise the model
inspection=pd.DataFrame({'Area':area, 'Actual':y_test, 'Predicted':y_pred})
sns.scatterplot(data=inspection, x='Area', y='Actual', color='red')
sns.regplot(data=inspection, x='Area', y='Predicted', color='blue')
```

Simple Linear Regression: example

Predicting house price based on area

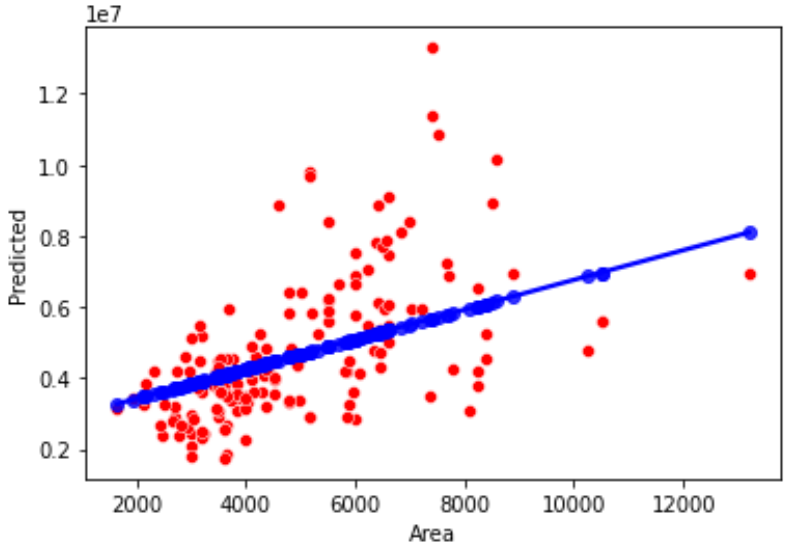
$$\hat{y} = b_0 + b_1 x$$

Price = $2,590,929 + $417 * Area

R-squared: 0.29

Mean Absolute Error: 1,221,100

Root Mean Square Error: 1,694,385



mean 4,766,729
std 1,870,440
min 1,750,000
max 13,300,000

Are you keeping pace?

https://www.mentimeter.com

Code: 18 11 35 5

Multiple Linear Regression

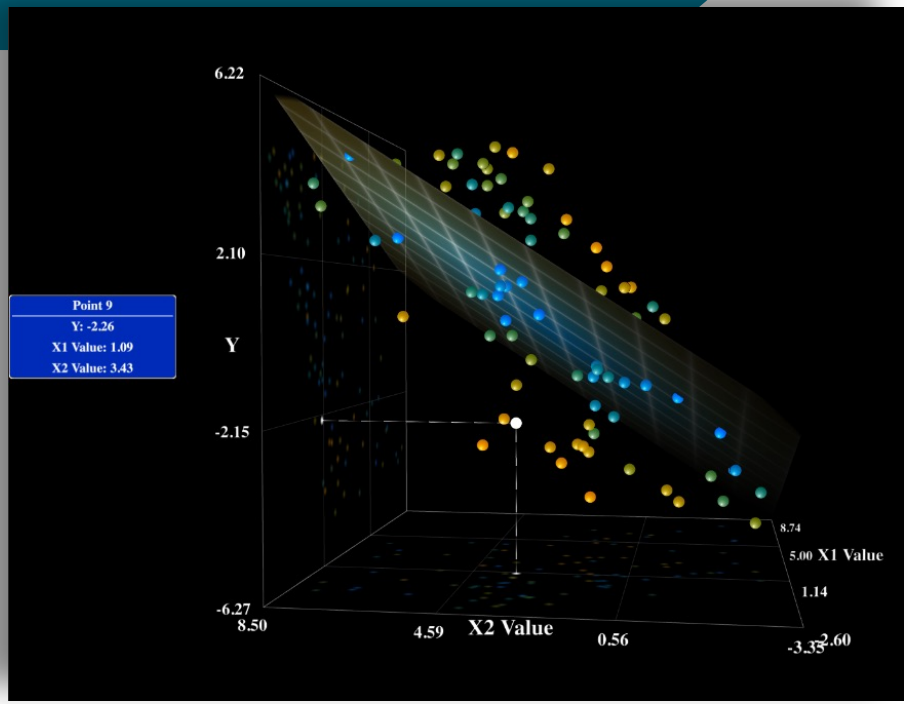


Image source: https://www.miabellaai.net/regression.html

If we have more than one independent variables:

$$\hat{y} = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n$$

Multiple Linear Regression: example

Predicting house price based on multiple predictors

```
#feature selection
features=['area','bedrooms', 'bathrooms','stories',
'parking','mainroad_N','guestroom_N','basement_N', 'hotwaterheating_N',
'airconditioning_N', 'prefarea_N', 'furnishingstatus_N']
X=records[features]


#specify the label
y=records['price']

# Import train_test_split function
from sklearn.model_selection import train_test_split

# Split dataset into a training dataset 70% and a test dataset 30%
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=1)
```

Multiple Linear Regression: example

Predicting house price based on multiple predictors

```
#import linear_model
from sklearn import linear_model

#create a linear_model object
reg = linear_model.LinearRegression()

# Train the model with the training dataset
reg=reg.fit(X_train, y_train)

#Make predictions for the test dataset
y_pred = reg.predict(X_test)
```

Multiple Linear Regression: example

Predicting house price based on multiple predictors

Regression equation:

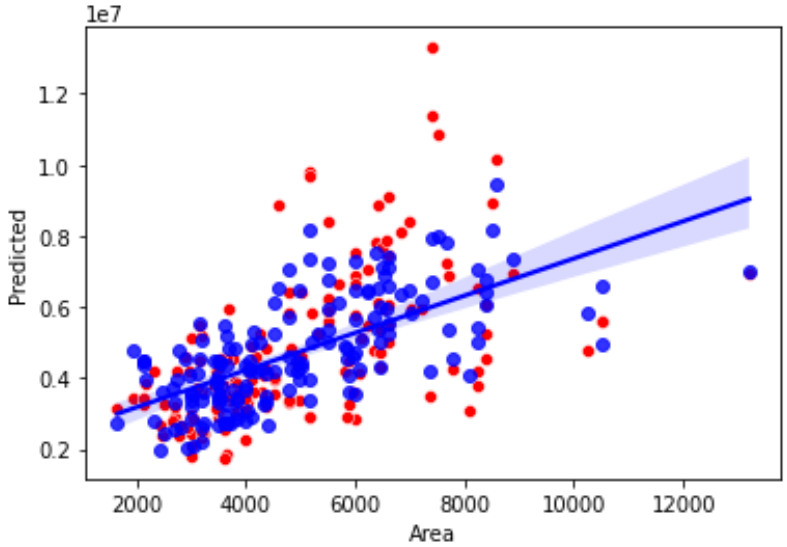
price = -880438.34 + 240.79 * area + 73833.90 * bedrooms + 1024597.99 * bathrooms + 511444.54 * stories + 280657.06 * parking + 504075.98 * mainroad + 346602.65 * guestroom + 404649.31 * prefarea + 754354.98 * furnishingstatus

R-squared: 0.66

Mean Absolute Error: 856,599

Root Mean Square Error: 1,176,619

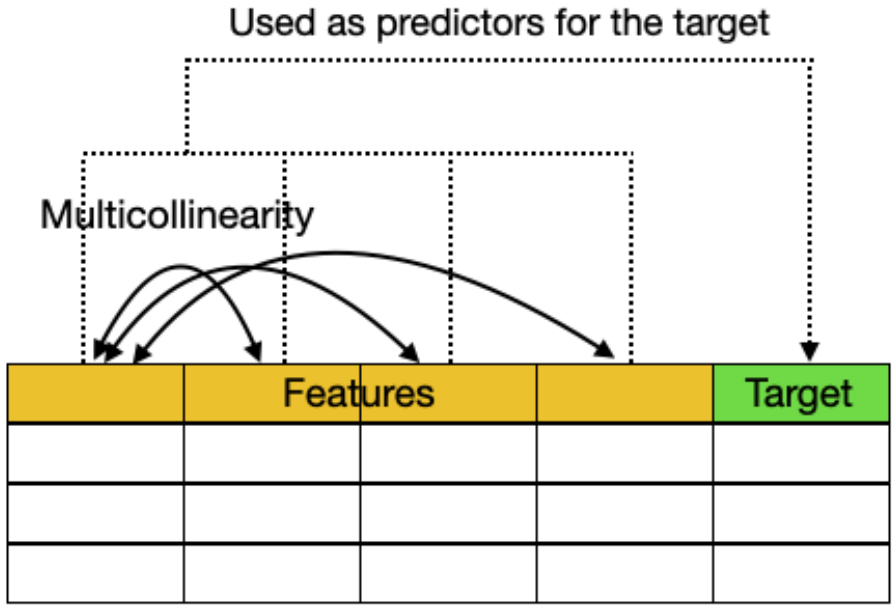
mean 4,766,729
std 1,870,440
min 1,750,000
max 13,300,000



Multicollinearity

Multicollinearity is a common problem in linear regression, where two or more independent variables are highly correlated with each other.

Example: currency exchange rate and price, BMI and weight



https://towardsdatascience.com/statistics-in-python-collinearity-and-multicollinearity-4cc4dcd82b3f

Multicollinearity

Multicollinearity is a common problem in linear regression, where two or more independent variables are highly correlated with each other.

Problems:

- Interpretability
- Difficulty in determining the relative importance of variables
- Overfitting with noises

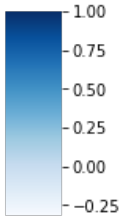
Will NOT make the model invalid; but can result in erratic changes in coefficient estimates in response to small changes in data

Can be removed for model simplicity

Use Lasso regularisation to identify most important predictors, or principal component analysis or factor analysis, to combine correlated predictors into new variables that capture the most important information.

Multicollinearity

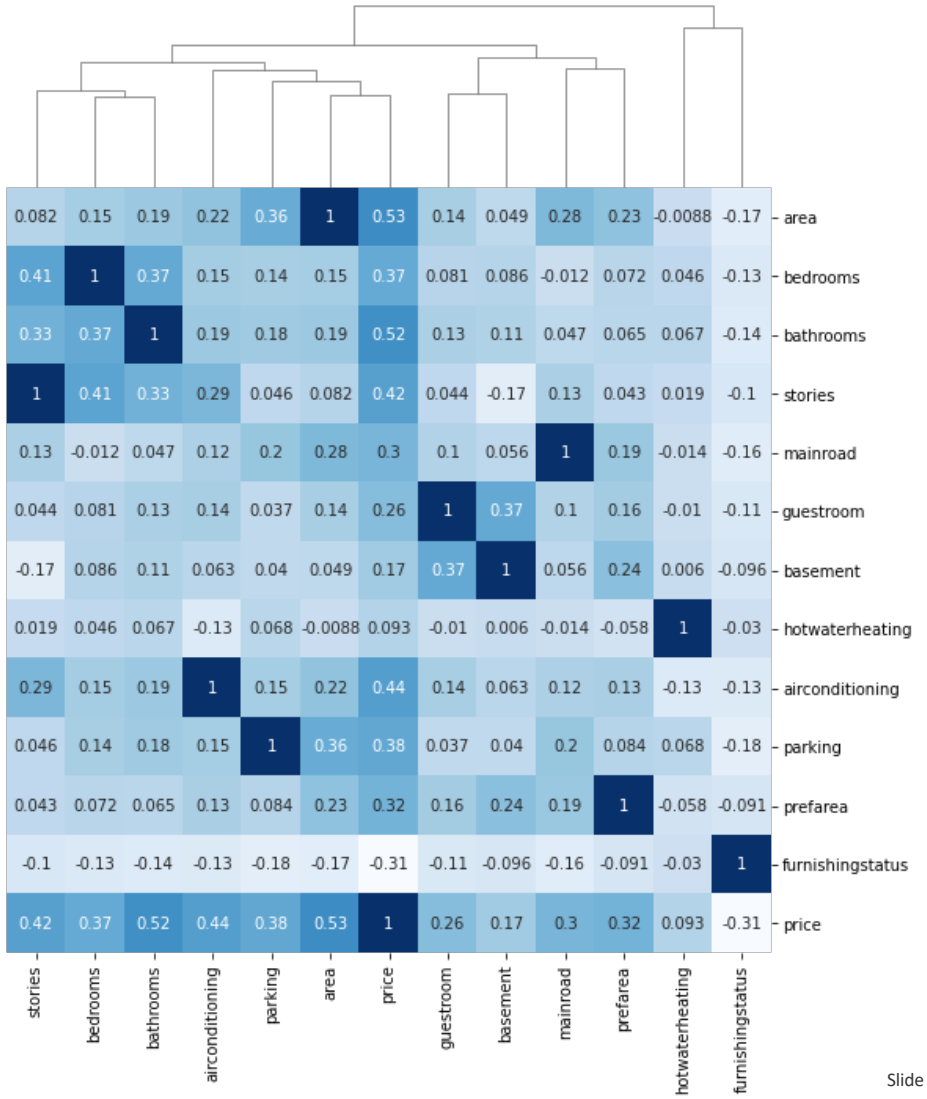
Predicting house price based on area



price = -880438.34 + 240.79 * area + 73833.90 * bedrooms + 1024597.99 * bathrooms + 511444.54 * stories + 280657.06 * parking + 504075.98 * mainroad + 346602.65 * guestroom + 404649.31 * prefarea + 754354.98 * furnishingstatus

Interpretability: e.g. with each additional bedroom, the price may not increase by $73,833.90.

Overfitting and difficulty in determining the relative importance of variables



```python
from sklearn.linear_model import Lasso

# Create a Lasso model
lasso = Lasso(alpha=0.1)

# Fit the model using the training data
lasso.fit(X_train, y_train)

# Get the coefficients
coefficients = lasso.coef_

# Print the features and their coefficients
for feature, coef in zip(features,
coefficients):
print(f'{feature}: {coef}')
```

Least Absolute Shrinkage and Selection Operator: Features with small coefficients do not contribute to the predictive power of the model. Shrink them.

Multicollinearity

The tolerance coefficient measures how well a predictor variable can be predicted by the other predictor variables in the model and can be used to diagnose multicollinearity.

- The tolerance coefficient ranges from 0 to 1; values close to 1 indicate low multicollinearity.
- For each predictor, tolerance is calculated as 1-r^2 where r^2 is model fit **for that predictor** and all the other attributes as predictors

Variance Inflation Factor (VIF) measures the degree to which the variance of the estimated regression coefficient for a predictor is inflated due to its correlation with other predictors in the model.

- VIF = 1 no multicollinearity with other predictors
- VIF > 1 some degree of collinearity
- VIF > 5 or >10 high or very high multicollinearity, and the predictor should be removed from the model.

Note…!

$$y' = b_0 + b_1x_1 + b_2x_2 + \cdots + b_nx_n$$

- **Numeric** predictors and target
- No missing data
- No outliers
- Check multi-collinearity
- Normal distribution and constant variance of residuals
- Not good for non-linear relationships, complex relationships
- Variations: ordinary least squares (OLS), Ridge regression…

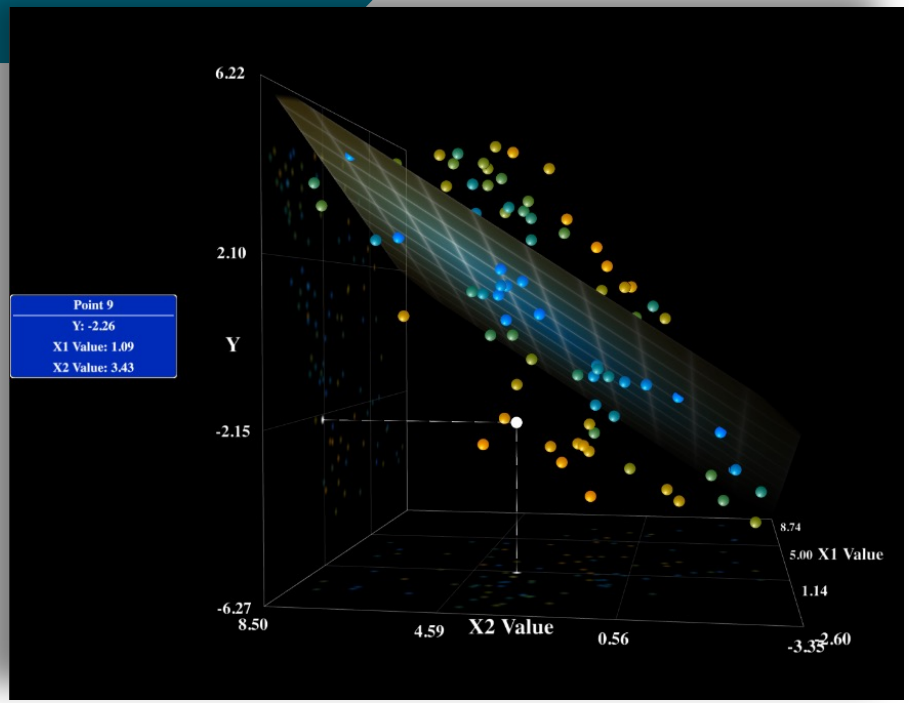


Image source: https://www.miabellaai.net/regression.html

Are you keeping pace?

https://www.mentimeter.com

Code: 18 11 35 5

Supervised Machine Learning: Regression

Types

- Simple Regression: the linear relationship between one independent variable and one dependent variable.

$$\hat{y} = b_0 + b_1 x$$

- Multiple Regression: the linear relationship between two or more independent variables and one dependent variable.

$$\hat{y} = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n$$

- Polynomial Regression: the relationship between one independent variable and one dependent variable using an n^{th} degree polynomial function.

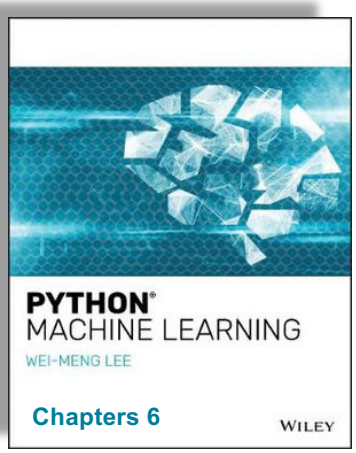
$$\hat{y} = b_0 + b_1 x + b_2 x^2 + \ldots + b_d x^d$$

- Polynomial Multiple Regression: the relationship between two or more independent variables and one dependent variable using an n^{th} degree polynomial function.

$$\hat{y} = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_1^2 + b_4 x_2^2 + b_5 x_1 x_2$$

In summary

- Business and data understandings are important
- Supervised ML, estimation: target is continuous
- Learning takes place to fit data into Linear Regression equations by optimising coefficients
- Assume linear relationships between predictors and target
- Performance metrics: r2, and loss functions: Root mean squared error, Mean absolute error



https://ebookcentral-proquest-com.ezproxyb.deakin.edu.au/lib/deakin/detail.action?docID=5747364

AND Sites cited on the Lecture slides

See useful sites in Lab 3