Data Mining and Machine Learning

Comp 3027J

Dr Catherine Mooney
Assistant Professor

catherine.mooney@ucd.ie

Lectures and Text

- **Core Text:**
  Fundamentals of Machine Learning for Predictive Data Analytics
  By John D. Kelleher, Brian Mac Namee and Aoife D'Arcy
- Last week Chapter 4, sections 4.4.2, 4.4.4 and 4.4.5 (Handling Continuous Descriptive Features, Tree Pruning and Model Ensembles).
- This week we will cover Chapter 7, sections 7.2 and 7.3 (Error-based Learning).
- Please read these sections of the book.

1. Error-based Learning
2. Simple Linear Regression
3. Measuring Error
4. Error Surfaces
5. Standard Approach: Multivariate Linear Regression with Gradient Descent
6. Gradient Descent
7. Interpreting Multivariable Linear Regression Models
8. Preview of Lab 7

Error-based Learning

- A **paramaterised** prediction model is initialised with a set of random parameters and an error function is used to judge how well this initial model performs when making predictions for instances in a training dataset.
- Based on the value of the error function the parameters are iteratively adjusted to create a more and more accurate model.

Simple Linear Regression

Table: The **office rentals dataset**: a dataset that includes office rental prices and a number of descriptive features for 10 Dublin city-centre offices.

ID	Size	Floor	Broadband Rate	Energy Rating	Rental Price
1	500	4	8	C	320
2	550	7	50	A	380
3	620	9	7	A	400
4	630	5	24	B	390
5	665	8	100	C	385
6	700	4	8	B	410
7	770	10	7	B	480
8	880	12	50	A	600
9	920	14	8	C	570
10	1,000	9	24	B	620

The office rentals dataset

- the SIZE of the office (in square feet)
- the FLOOR in the building in which the office space is located
- the BROADBAND rate available at the office
- the ENERGY RATING of the building in which the office space is located (ratings range from A to C, where A is the most efficient)
- the RENTAL PRICE (in Euro per month)

Table: Simplified version of the **office rentals dataset** showing office rental prices and office size.

ID	Size	Rental Price
1	500	320
2	550	380
3	620	400
4	630	390
5	665	385
6	700	410
7	770	480
8	880	600
9	920	570
10	1,000	620

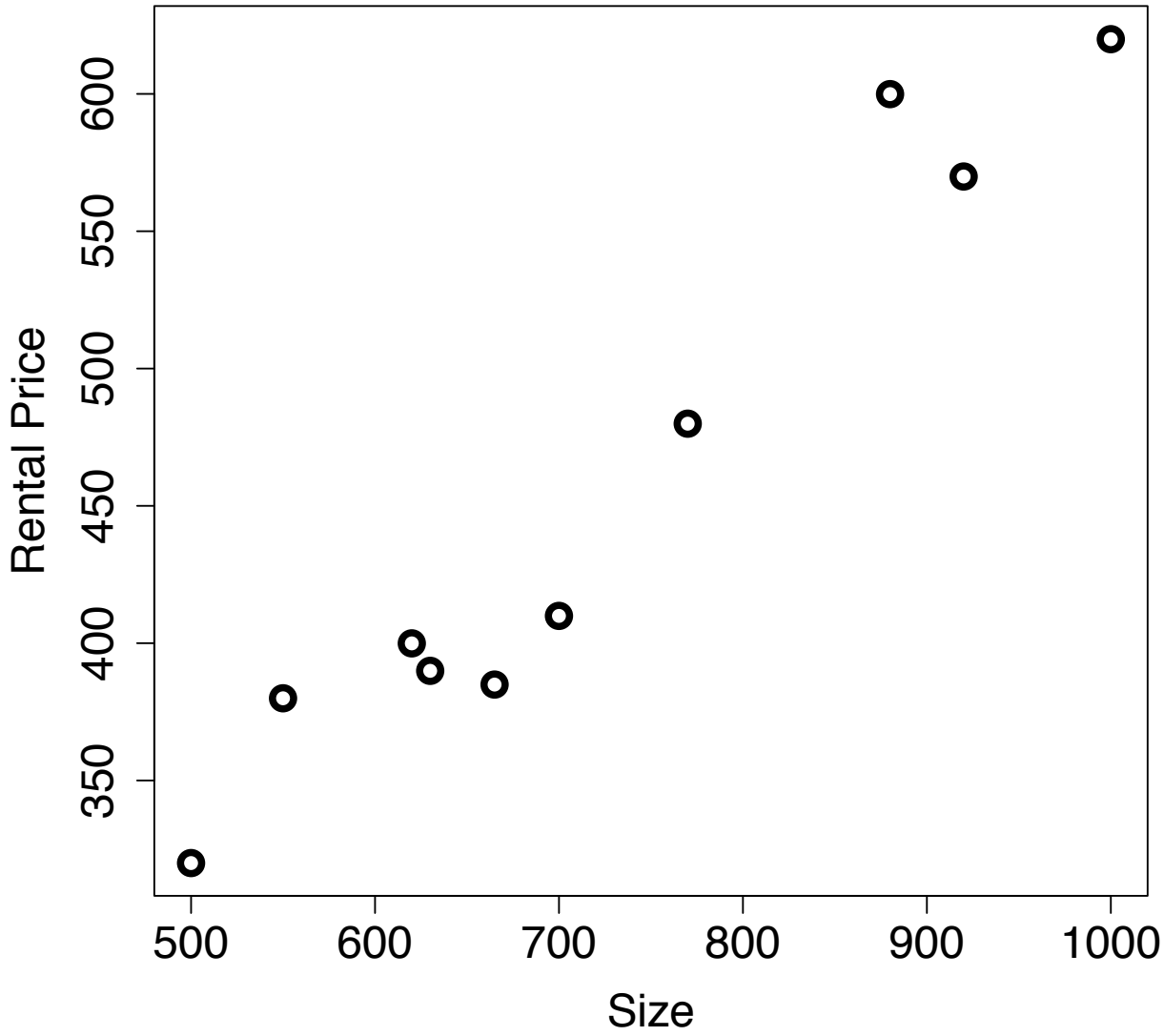


Figure: A scatter plot of the SIZE and RENTAL PRICE features from the office rentals dataset.

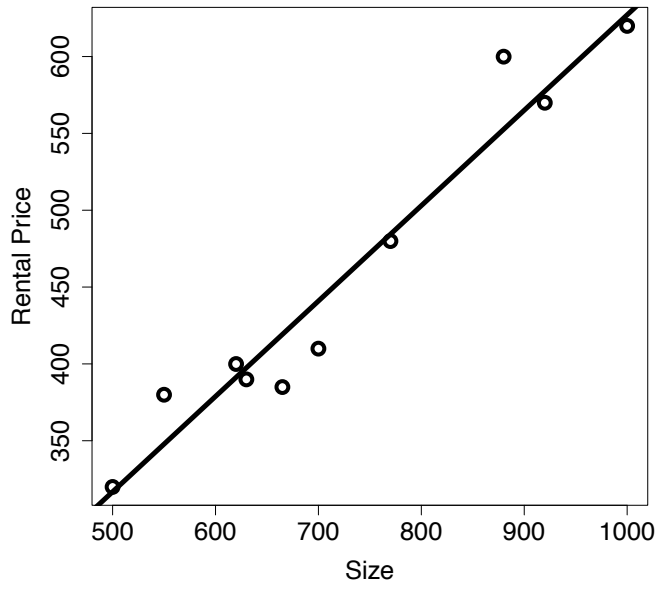
- From the scatter plot it appears that there is a linear relationship between the SIZE and RENTAL PRICE.
- If we could capture this relationship in a model, we would be able to do two important things.
  1. We would be able to understand how office size affects office rental price.
  2. We would be able to fill in the gaps in the dataset to predict office rental prices for office sizes that we have never actually seen
- Both of these things would be of great use to real estate agents trying to make decisions about the rental prices they should set for new rental properties.
- For example, how much would we expect a 730 square foot office to rent for?

There is a simple, well-known mathematical model that can capture the relationship between two continuous features like those in our dataset – The equation of a line, which can be written as:

$$y = mx + b$$

- m is the slope of the line
- b is known as the y-intercept of the line (i.e., the position at which the line meets the vertical axis when the value of x is set to zero).

The equation of a line predicts a y value for every x value given the slope and the y-intercept, and we can use this simple model to capture the relationship between two features such as SIZE and PRICE.



- A scatter plot of the SIZE and RENTAL PRICE features from the office rentals dataset with a simple linear model added to capture the relationship.
- This model is:

$$\begin{aligned} \text{RENTAL PRICE} &= 0.62 \times \text{SIZE} + 6.47 \\ y &= mx + b \end{aligned}$$

Exercise

$$\text{RENTAL PRICE} = 6.47 + 0.62 \times \text{SIZE}$$

- Using this model determine the expected rental price of the 730 square foot office

$$\text{RENTAL PRICE} = 6.47 + 0.62 \times \text{SIZE}$$

- Using this model determine the expected rental price of the 730 square foot office:

$$\begin{aligned} \text{RENTAL PRICE} &= 6.47 + 0.62 \times 730 \\ &= 459.07 \end{aligned}$$

This kind of model is known as a **simple linear regression model**. This approach to modeling the relationships between features is extremely common in both machine learning and statistics.

We can rewrite the simple linear regression model as

$$\mathbb{M}_{\mathbf{w}}(d) = \mathbf{w}[0] + \mathbf{w}[1] \times \mathbf{d}[1]$$

- the parameters $\mathbf{w}[0]$ and $\mathbf{w}[1]$ are referred to as weights
- $\mathbf{d}$ is an instance defined by a single descriptive feature $\mathbf{d}[1]$
- $\mathbb{M}_{\mathbf{w}}(d)$ is the prediction output by the model for the instance $\mathbf{d}$

Measuring Error

The error function

- The key to using simple linear regression models is determining the optimal values for the weights in the model.
- The optimal weights allow the model to capture the relationship between the descriptive features and a target feature.
- They are said to fit the training data.
- We need some way to measure how well a model fits a training dataset.
- We do this by defining an error function.
- An **error function** captures the error between the predictions made by a model and the actual values in a training dataset.

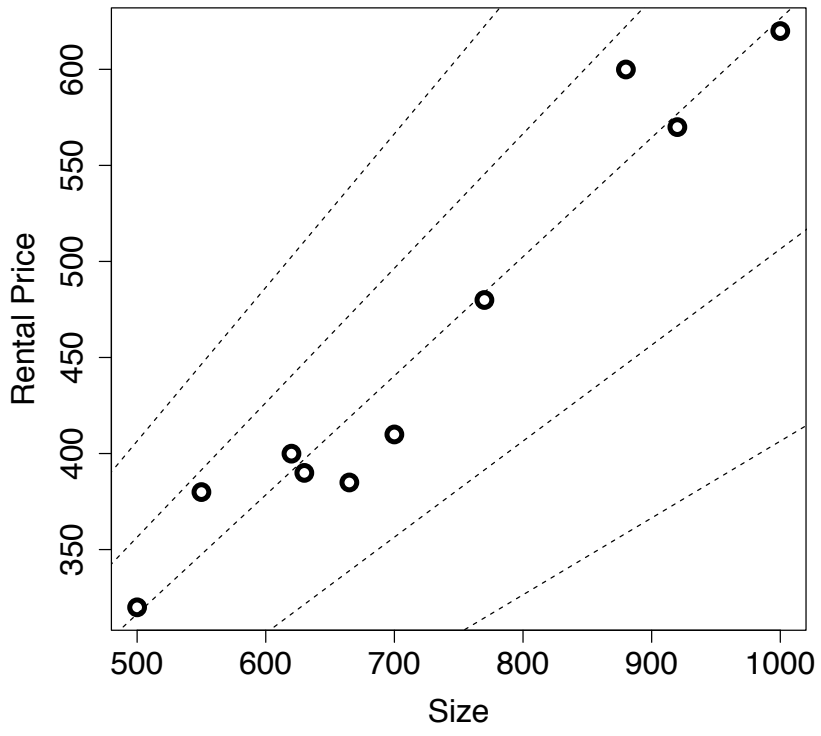


Figure: A collection of possible simple linear regression models capturing the relationship between SIZE and RENTAL PRICE. For all models **w**[0] is set to 6.47. From top to bottom the models use 0.4, 0.5, 0.62, 0.7 and 0.8 respectively for **w**[1]. The model with **w**[1] set to 0.62 most accurately fits the relationship between office sizes and office rental prices, but how do we measure this formally?

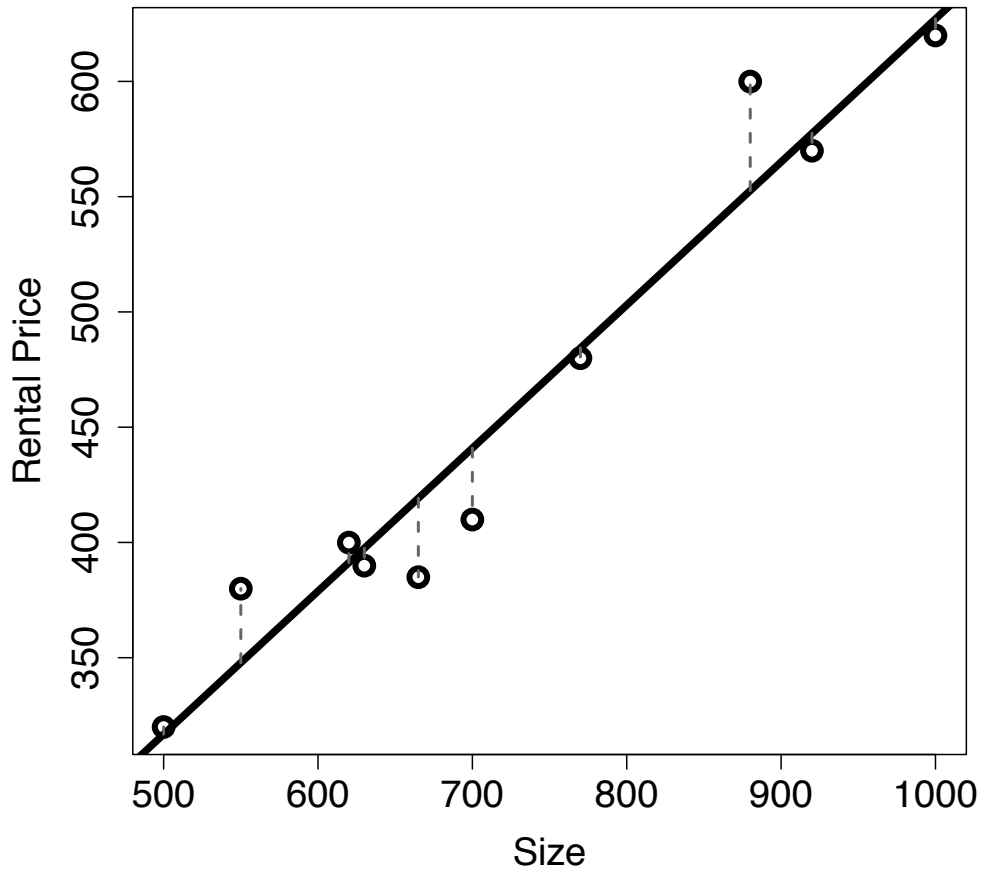


Figure: A scatter plot of the SIZE and RENTAL PRICE features from the office rentals dataset showing a candidate prediction model (with $\mathbf{w}[0] = 6.47$ and $\mathbf{w}[1] = 0.62$) and the resulting errors.

The sum of squared errors error function

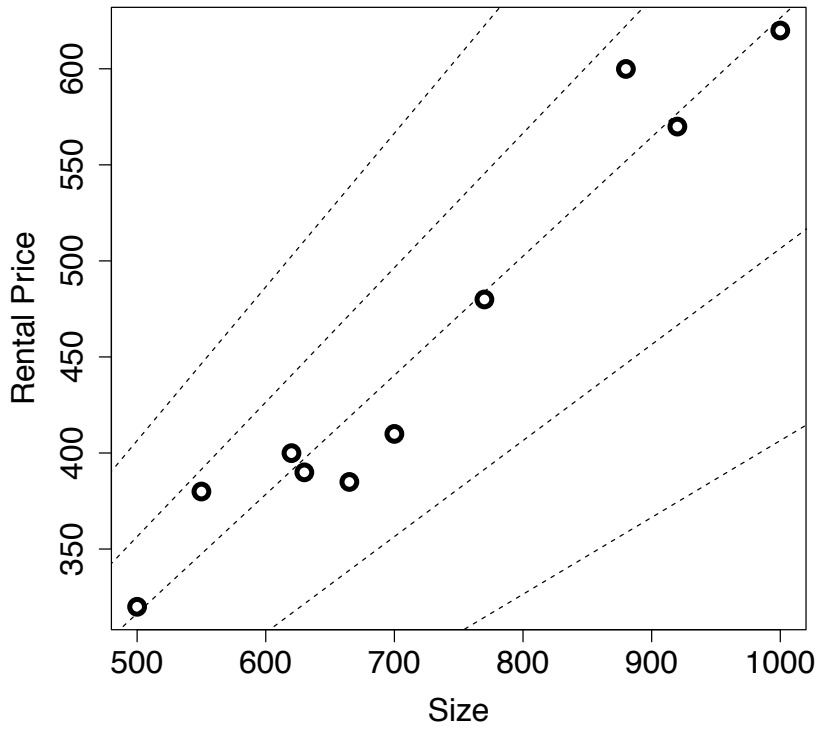
- There are many different kinds of error functions
- For measuring the fit of simple linear regression models, the most commonly used is the sum of squared errors error function, or SSE.
- To calculate SSE we make a prediction for each member of the training dataset and then calculate the error between these predictions and the actual target feature values in the training set.

The sum of squared errors error function, SSE, is formally defined as

$$\begin{aligned} SSE(\mathbb{M}_{\mathbf{w}}, \mathcal{D}) &= \frac{1}{2}\sum_{i=1}^{n}(t_i - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_i[1]))^2 \\ &= \frac{1}{2}\sum_{i=1}^{n}(t_i - (\mathbf{w}[0] + \mathbf{w}[1] \times \mathbf{d}_i[1]))^2 \end{aligned}$$

Table: Calculating the sum of squared errors for the candidate model (with $\mathbf{w}[0] = 6.47$ and $\mathbf{w}[1] = 0.62$) making predictions for the the office rentals dataset.

ID	Rental Price	Model Prediction	Error Error	Squared Error
1	320	316.79	3.21	10.32
2	380	347.82	32.18	1,035.62
3	400	391.26	8.74	76.32
4	390	397.47	-7.47	55.80
5	385	419.19	-34.19	1,169.13
6	410	440.91	-30.91	955.73
7	480	484.36	-4.36	19.01
8	600	552.63	47.37	2,243.90
9	570	577.46	-7.46	55.59
10	620	627.11	-7.11	50.51
			Sum	**5,671.64**
		Sum of squared errors (Sum/2)		**2,835.82**



If we perform the same calculation for the other candidate models we find that with **w**[1] set to 0.4, 0.5, 0.7, and 0.8, the sums of squared errors are 136,218, 42,712, 20,092, and 90,978 respectively.

Error Surfaces

How can the values of an error function for many different potential models be combined to form an error surface across which we can search for the optimal weights with the minimum error?

- For every possible combination of weights, $\mathbf{w}[0]$ and $\mathbf{w}[1]$, there is a corresponding sum of squared errors value that can be joined together to make a surface.
- We can think about all these error values joined to make a surface defined by the weight combinations

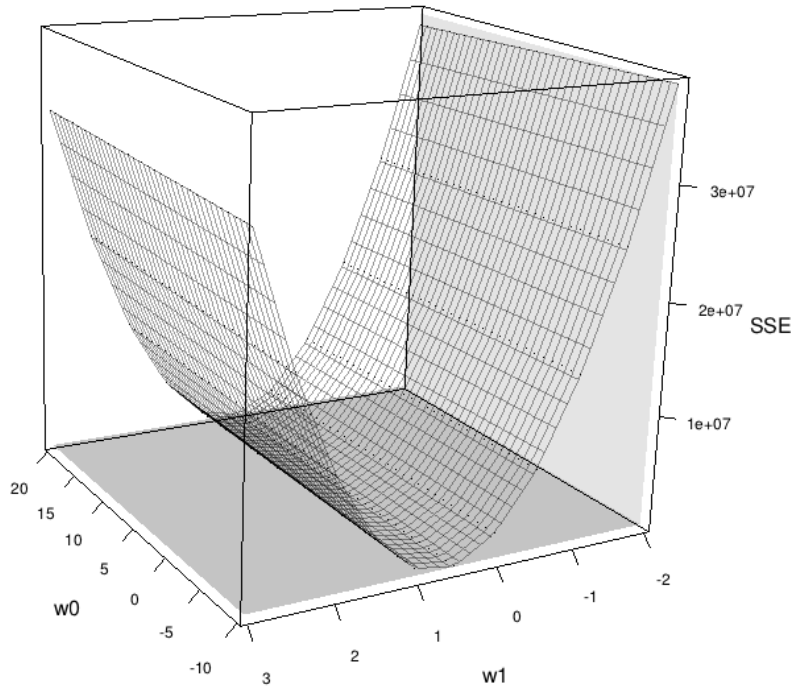


Figure: A 3D surface plot of the error surface generated by plotting the sum of squared errors value for the office rentals training set for each possible combination of values for $\mathbf{w}[0]$ (from the range $[-10, 20]$) and $\mathbf{w}[1]$ (from the range $[-2, 3]$).

- The *x*-*y* plane is known as a **weight space** and the surface is known as an **error surface**.
- The model that best fits the training data is the model corresponding to the lowest point on the error surface.

- For some simple problems it is possible to try out every reasonable combination of weights find the best combination (**brute-force search**).
- For most real-world problems this is not feasible – the computation required would take far too long.
- We need a more efficient way to find the best combination of weights.

- Fortunately, these error surfaces have two properties that help us find the optimal combination of weights
  1. they are convex
  2. they have a global minimum
- If we can find the global minimum of the error surface, we can find the set of weights defining the model that best fits the training dataset.
- This approach to finding weights is known as least squares optimization.

- We can find the optimal weights at the point where the partial derivatives of the error surface with respect to **w**[0] and **w**[1] are equal to 0.
- This is the point at the very bottom of the bowl defined by the error surface – there is no slope at the bottom of the bowl.
- This point is at the **global minimum** of the error surface and the coordinates of this point define the weights for the prediction model with the lowest sum of squared errors on the dataset.
- If you need to refresh your understanding of differentiation and derivatives see Appendix C of your book.

- We can formally define this point on the error surface as the point at which:

$$\frac{\partial}{\partial \mathbf{w}[0]} \frac{1}{2} \sum_{i=1}^{n} (t_i - (\mathbf{w}[0] + \mathbf{w}[1] \times \mathbf{d}_i[1]))^2 = 0$$

and

$$\frac{\partial}{\partial \mathbf{w}[1]} \frac{1}{2} \sum_{i=1}^{n} (t_i - (\mathbf{w}[0] + \mathbf{w}[1] \times \mathbf{d}_i[1]))^2 = 0$$

- There are a number of different ways to find this point.
- The most common approach is known as the **gradient descent** algorithm.

5 minute break?

Standard Approach: Multivariate Linear Regression with Gradient Descent

- The most common approach to error-based machine learning for predictive analytics is to use **multivariable linear regression with gradient descent** to train a best-fit model for a given training dataset.
- Now we are going to look at how to extend the simple linear regression model described in the previous section to handle multiple descriptive features.

- Fortunately, extending the simple linear regression model to a multivariable linear regression model is straightforward.
- We can define a multivariate linear regression model as:

$$\begin{aligned} \mathbb{M}_{\mathbf{w}}(\mathbf{d}) &= \mathbf{w}[0] + \mathbf{w}[1] \times \mathbf{d}[1] + \cdots + \mathbf{w}[m] \times \mathbf{d}[m] \\ &= \mathbf{w}[0] + \sum_{j=1}^{m} \mathbf{w}[j] \times \mathbf{d}[j] \end{aligned}$$

- We can make the equation look a little neater by inventing a dummy descriptive feature, $\mathbf{d}[0]$, that is always equal to 1:

$$\begin{aligned}\mathbb{M}_{\mathbf{w}}(\mathbf{d}) &= \mathbf{w}[0] \times \mathbf{d}[0] + \mathbf{w}[1] \times \mathbf{d}[1] + \ldots + \mathbf{w}[m] \times \mathbf{d}[m] \\ &= \sum_{j=0}^{m} \mathbf{w}[j] \times \mathbf{d}[j] \\ &= \mathbf{w} \cdot \mathbf{d}\end{aligned}$$

- $\mathbf{w} \cdot \mathbf{d}$ is the dot product of the vectors $\mathbf{w}$ and $\mathbf{d}$.
- The dot product of two vectors is the sum of the products of their corresponding elements.

- The sum of squared errors loss function, L_2, changes only very slightly to reflect the new regression equation:

$$\begin{aligned} L_2(\mathbb{M}_{\mathbf{w}}, \mathcal{D}) &= \frac{1}{2}\sum_{i=1}^{n}(t_i - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_i))^2 \\ &= \frac{1}{2}\sum_{i=1}^{n}(t_i - (\mathbf{w}\cdot\mathbf{d}_i))^2 \end{aligned}$$

Table: A dataset that includes office rental prices and a number of descriptive features for 10 Dublin city-center offices.

ID	Size	Floor	Broadband Rate	Energy Rating	Rental Price
1	500	4	8	C	320
2	550	7	50	A	380
3	620	9	7	A	400
4	630	5	24	B	390
5	665	8	100	C	385
6	700	4	8	B	410
7	770	10	7	B	480
8	880	12	50	A	600
9	920	14	8	C	570
10	1,000	9	24	B	620

- This multivariate model allows us to include all but one of the descriptive features in a regression model to predict office rental prices.
- The resulting multivariate regression model equation is:

$$\begin{aligned} \text{RENTAL PRICE} = \mathbf{w}[0] &+ \mathbf{w}[1] \times \text{SIZE} + \mathbf{w}[2] \times \text{FLOOR} \\ &+ \mathbf{w}[3] \times \text{BROADBAND RATE} \end{aligned}$$

- We will see in the next section how the best-fit set of weights for this equation are found, but for now we will set:
  - $\mathbf{w}[0] = -0.1513$,
  - $\mathbf{w}[1] = 0.6270$,
  - $\mathbf{w}[2] = -0.1781$,
  - $\mathbf{w}[3] = 0.0714$.
- This means that the model is rewritten as:

$$\begin{aligned}\text{RENTAL PRICE} = -0.1513 &+ 0.6270 \times \text{SIZE} \\ &- 0.1781 \times \text{FLOOR} \\ &+ 0.0714 \times \text{BROADBAND RATE}\end{aligned}$$

Exercise

- Using this model:

$$\begin{aligned}\text{RENTAL PRICE} = -0.1513 &+ 0.6270 \times \text{SIZE} \\ &- 0.1781 \times \text{FLOOR} \\ &+ 0.0714 \times \text{BROADBAND}\end{aligned}$$

- we can, for example, predict the expected rental price of a 690 square foot office on the 11^{th} floor of a building with a broadband rate of 50 Mb per second as:

$$\text{RENTAL PRICE} = ?$$

- Using this model:

$$\begin{aligned}\text{RENTAL PRICE} = -0.1513 &+ 0.6270 \times \text{SIZE} \\ &- 0.1781 \times \text{FLOOR} \\ &+ 0.0714 \times \text{BROADBAND RATE}\end{aligned}$$

- we can, for example, predict the expected rental price of a 690 square foot office on the 11^{th} floor of a building with a broadband rate of 50 Mb per second as:

$$\begin{aligned}\text{RENTAL PRICE} &= -0.1513 + 0.6270 \times 690 \\ &\quad -0.1781 \times 11 + 0.0714 \times 50 \\ &= 434.0896\end{aligned}$$

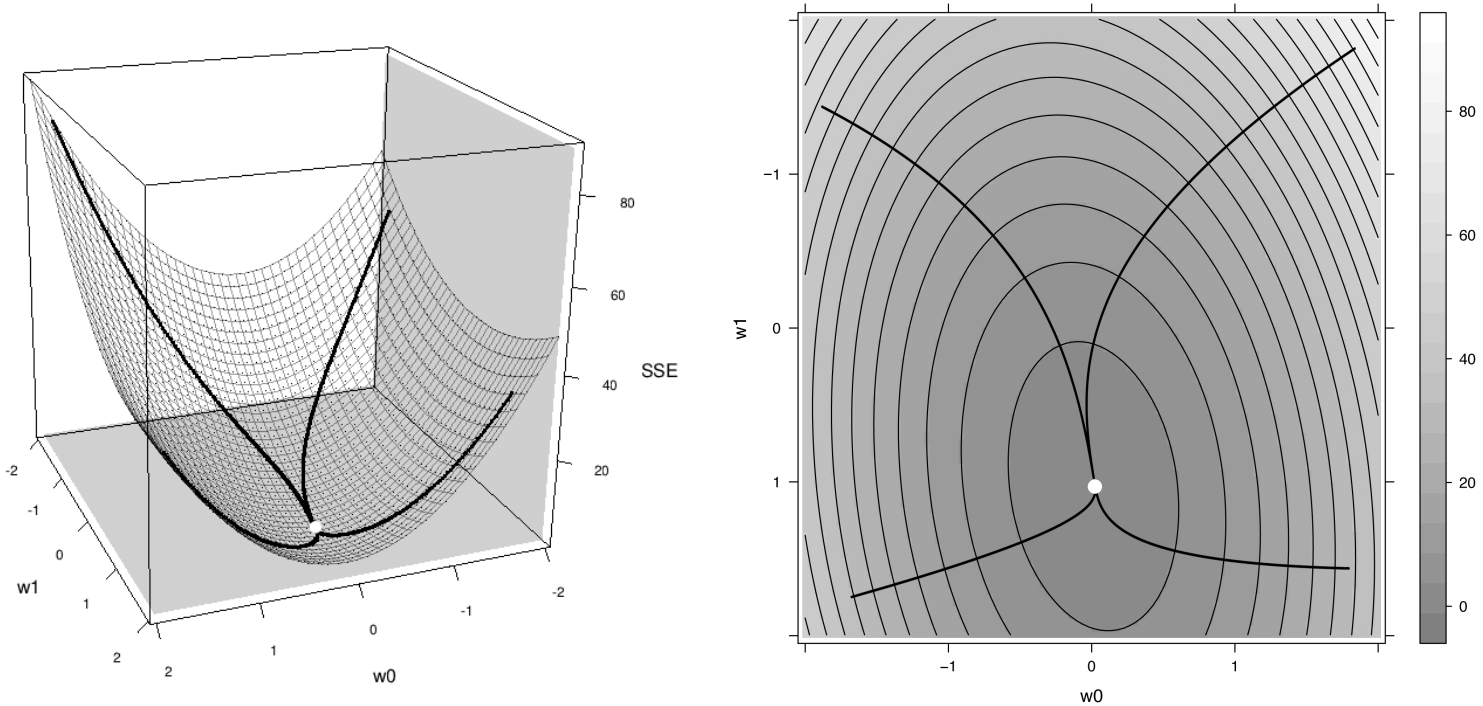
Gradient Descent

Gradient Descent

A simple approach to learning weights based on the facts that

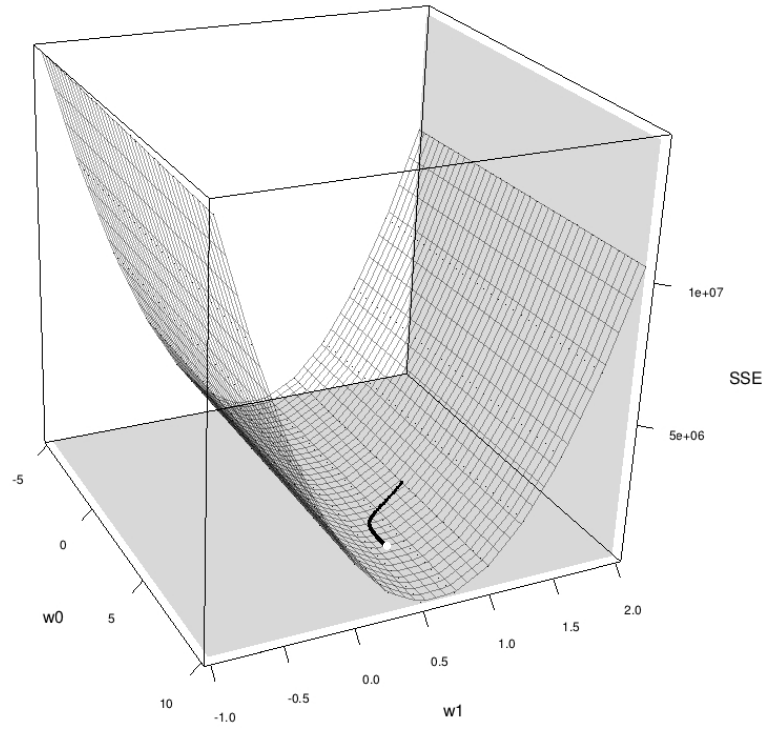
1. even though they are hard to visualize, the error surfaces that correspond to high-dimensional weight spaces still have a convex shape (albeit in multiple dimensions)
2. that a single global minimum exists.

This approach uses a guided search from a random starting position and is known as **gradient descent**.

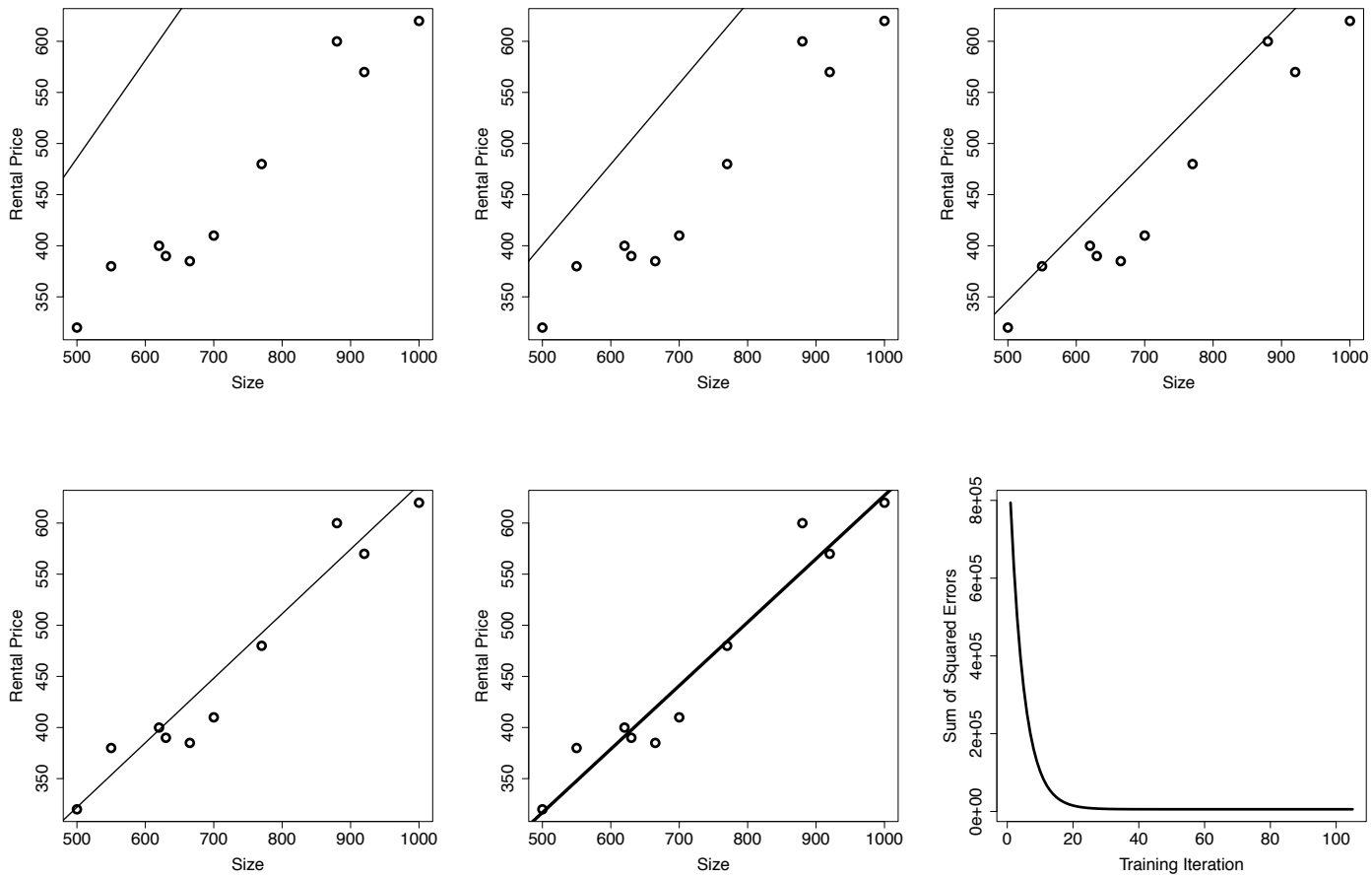


(a) (b)

Figure: (a) A 3D surface plot and (b) a contour plot of the same error surface. The lines indicate the path that the gradient decent algorithm would take across this error surface from different starting positions to the global minimum - marked as the white dot in the centre.



A 3D surface plot of the error surface for the office rentals dataset – showing the journey across the error surface that is taken by the gradient descent algorithm when training the simple version of the office rentals example - involving just SIZE and RENTAL PRICE.



A selection of the simple linear regression models developed during the gradient descent process for the simple version of the office rentals example - involving just SIZE and RENTAL PRICE. The final panel shows the sum of squared error (SSE) values generated during the gradient descent process.

The gradient descent algorithm for training multivariate linear regression models.

Require: a set of training instances $\mathcal{D}$
Require: a learning rate α that controls how quickly the algorithm converges
Require: a function, **errorDelta**, that determines the direction in which to adjust a given weight, **w**[*j*], so as to move down the slope of an error surface determined by the dataset, $\mathcal{D}$
Require: a convergence criterion that indicates that the algorithm has completed

1: **w** $\leftarrow$ random starting point in the weight space
2: **repeat**
3: **for** each **w**[*j*] in **w** **do**
4: **w**[*j*] $\leftarrow$ **w**[*j*] $+ \alpha \times$ **errorDelta**($\mathcal{D}$, **w**[*j*])
5: **end for**
6: **until** convergence occurs

- The most important part to the gradient descent algorithm is Line 4 on which the weights are updated:

$$\mathbf{w}[j] \leftarrow \mathbf{w}[j] + \alpha \times \textbf{errorDelta}(\mathcal{D}, \mathbf{w}[j])$$

- Each weight is considered independently and for each one a small adjustment is made by adding a small **delta** value to the current weight, $\mathbf{w}[j]$.
- This adjustment should ensure that the change in the weight leads to a move *downwards* on the error surface.

- Adjusting the calculation to take into account multiple training instances:

$$\frac{\partial}{\partial \mathbf{w}[j]} L_2(\mathbb{M}_{\mathbf{w}}, \mathcal{D}) = \sum_{i=1}^{n} ((t_i - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_i)) \times \mathbf{d}_i[j])$$

- We use this equation to define the **errorDelta** in our gradient descent algorithm.

$$\mathbf{w}[j] \leftarrow \mathbf{w}[j] + \alpha \underbrace{\sum_{i=1}^{n} ((t_i - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_i)) \times \mathbf{d}_i[j])}_{errorDelta(\mathcal{D},\mathbf{w}[j])}$$

- The **learning rate**, α, determines the size of the adjustment made to each weight at each step in the process.
- Unfortunately, choosing learning rates is not a well defined science.
- Most practitioners use rules of thumb and trial and error.

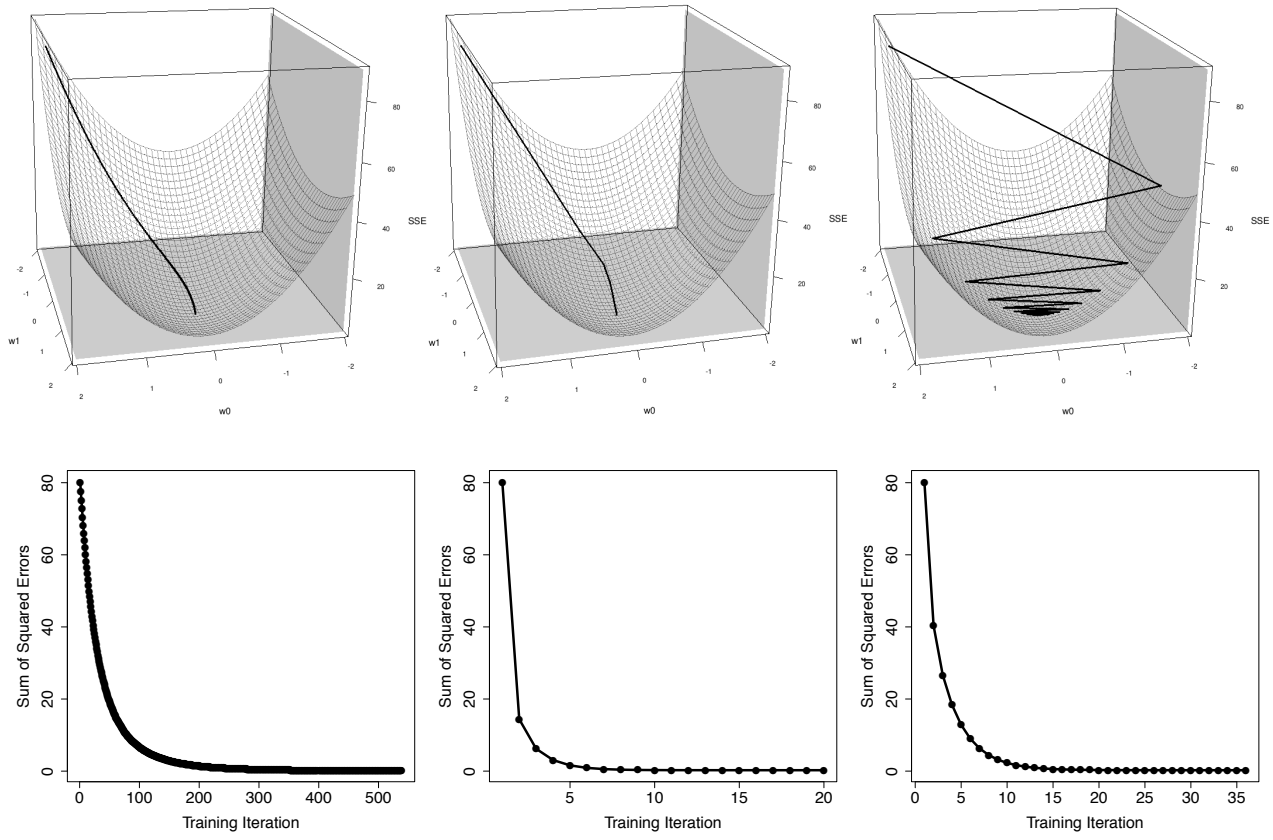


Figure: Plots of the journeys made across the error surface for (a) a very small learning rate (0.002), (b) a medium learning rate (0.08) and (c) a very large learning rate (0.18).

- A typical range for learning rates is $[0.00001, 10]$
- Based on empirical evidence, choosing random initial weights uniformly from the range $[-0.2, 0.2]$ tends to work well.

Interpreting Multivariable Linear Regression Models

- The weights used by linear regression models indicate the effect of each descriptive feature on the predictions returned by the model.
- Both the **sign** and the **magnitude** of the weight provide information on how the descriptive feature effects the predictions of the model.

- It is tempting to infer the relative importance of the different descriptive features in the model from the magnitude of the weights
- However, direct comparison of the weights tells us little about their relative importance.
- A better way to determine the importance of each descriptive feature in the model is to perform a **statistical significance test**.

- The statistical significance test we use to analyze the importance of a descriptive feature $\mathbf{d}[j]$ in a linear regression model is the ***t*-test**.
- The null hypothesis for this test is that the feature does not have a significant impact on the model. The test statistic we calculate is called the t-statistic.

- The standard error for the overall model is calculated as

$$se = \sqrt{\frac{\sum_{i=1}^{n} (t_i - \mathbb{M}_{\mathbf{w}}(\mathbf{d}_i))^2}{n-2}} \tag{1}$$

- A standard error calculation is then done for a descriptive feature as follows:

$$se(\mathbf{d}[j]) = \frac{se}{\sqrt{\sum_{i=1}^{n} \left(\mathbf{d}_i[j] - \overline{\mathbf{d}[j]}\right)^2}} \tag{2}$$

- The t-statistic for this test is calculated as follows:

$$t = \frac{\mathbf{w}[j]}{se(\mathbf{d}[j])} \tag{3}$$

- Using a standard *t*-statistic look-up table, we can then determine the *p*-value associated with this test (this is a two tailed *t*-test with degrees of freedom set to the number of instances in the training set minus 2).
- If the *p*-value is less than the required significance level, typically 0.05, we reject the null hypothesis and say that the descriptive feature has a significant impact on the model; otherwise we say that it does not.

Table: Weights and standard errors for each feature in the office rentals model.

Descriptive Feature	Weight	Standard Error	*t*-statistic	*p*-value
SIZE	0.6270	0.0545	11.504	$<$0.0001
FLOOR	-0.1781	2.7042	-0.066	0.949
BROADBAND RATE	0.071396	0.2969	0.240	0.816

1. Error-based Learning
2. Simple Linear Regression
3. Measuring Error
4. Error Surfaces
5. Standard Approach: Multivariate Linear Regression with Gradient Descent
6. Gradient Descent
7. Interpreting Multivariable Linear Regression Models
8. Preview of Lab 7

Recommended Reading

- **Core Text:**
  Fundamentals of Machine Learning for Predictive Data Analytics
  By John D. Kelleher, Brian Mac Namee and Aoife D'Arcy
- This week we covered Chapter 7, sections 7.2 and 7.3 (Error-based Learning – Linear Regression and Gradient Descent).
- I would suggest that you would read over these sections again.
- Email me if you have any questions and I will cover them at the beginning of our next lecture.
- Next week we will cover Chapter 7 – Logistic Regression and Support Vector Machines.

Data Mining and Machine Learning Assignment Part 2

Questions?

Preview of Lab 7

Preview of Lab 7

- In Lab 7 we will be using R for Error-based Learning (Linear Regression)
- install.packages("car")
- install.packages("ggplot2")

Preview of Lab 7

- The labs are not graded HOWEVER
- The labs are compulsory and you MUST submit what you have done during the lab before the end of the lab
- If I have to make a decision, for example, between pass/fail, I may use them

Please come and talk to me:

- Wang Guoxin
- Du Xin
- Han Peiqi
- Sun Haoyang
- Zhang Jingcheng