SIT 112 - Data Science Concepts

Lecturer: Truyen Tran | truyen.tran@deakin.edu.au

School of Information Technology,
Deakin University, VIC 3216, Australia.

Practical Session 2.1: Markdown Language

The purpose of this session is:

1. introducing you the syntax of Markdown language
2. using Markdown in Jupyter Notebook cells

Source: The following material has been prepared based on Jupyter Notebook documentation.

1. Using Markdown Cells for Writing

Why aren't there font and font size selection drop down menus, buttons I can press to bold and italicize my text, or other advanced style options in my Notebook? When you use Microsoft Word, Google Docs, Apple Pages, Open Office, or any other word processing software, you generally use your mouse to select various style options, like line spacing, font size, font color, paragraph format etc. This kind of system is often describes as a WYSIWYG (What You See Is What You Get) interface. This means that the input (what you tell the computer) exactly matches the output (what the computer gives back to you). If you type the letter G, highlight it, select the color green and up the font size to 64 pt, your word processor will show you a fairly large green colored letter G. And if you print out that document you will print out a fairly large green colored letter G.

This Notebook, however, does not use a WYSIWYG interface. Instead it uses something called a "markup Language (http://en.wikipedia.org/wiki/Markup_language)". When you use a a markup language, your input does not necessarily exactly equal your output.

For example, if I type "#Header 1" at the beginning of a cell, but then press Shift-Enter (or click the play button at the top of the window), this notebook will turn my input into a somewhat different output in the following way:

```
#Header 1
```

#Header 1

And if I type "##Header 2" (at the beginning of a cell), this notebook will turn that input into another output:

```
##Header 2
```

##Header 2

In these examples, the hashtags are markers which tell the Notebook how to typeset the text. There are many markup languages, but one family, or perhaps guiding philosophy, of markup languages is called "Markdown," named somewhat jokingly for its simplicity. Your Notebook uses "marked," a Markdown library of typeset and other formatting instructions, like the hashtags in the examples above.

Markdown is a markup language that generates HTML, which the cell can interpret and render. This means that Markdown Cells can also render plain HTML code. If you're interested in learning HTML, check out this helpful online tutorial (http://www.w3schools.com/html/).

Why Use Markdown (and not a WYSIWYG)?

Why is Markdown better? Well, it's worth saying that maybe it isn't. Mainly, it's not actually a question of better or worse, but of what's in front of you and of who you are. A definitive answer depends on the user and on that user's goals and experience. These Notebooks don't use Markdown because it's definitely better, but rather because it's different and thus encourages users to think about their work differently.

It is very important for computer science students to learn how to conceptualize input and output as dependent, but also distinct. One good reason to use Markdown is that it encourages this kind of thinking. Relatedly, it might also promote focus on substance over surface aesthetic. Markdown is somewhat limited in its style options, which means that there are inherently fewer non-subject-specific concerns to agonize over while working. It is the conceit of this philosophy that you would, by using Markdown and this Notebook, begin to think of the specific stylistic rendering of your cells as distinct from what you type into those same cells, and thus also think of the content of your writing as necessarily separate from its formating and appearance.

1.1 Block Elements

1.1.1 Headers

```
#Header 1
```

Header 1

```
##Header 2
```

Header 2

```
###Header 3
```

Header 3

```
####Header 4
```

Header 4

```
#####Header 5
```

Header 5

```
######Header 6
```

Header 6

1.1.2 Block Quotes

1.1.2.1 Standard Block Quoting

```
>blah blah block quote blah blah block quote blah blah block
```

```
quote blah blah block quote blah blah block
quote blah blah block quote blah blah block quote blah blah block quote
```

> blah blah block quote blah blah block quote blah blah block
>
> quote blah blah block quote blah blah block
>
> quote blah blah block quote blah blah block quote blah blah block quote

Note: Block quotes work best if you intentionally hard-wrap the lines.

1.1.2.2 Nested Block Quoting

```
>blah blah block quote blah blah block quote blah blah block
block quote blah blah block block quote blah blah block
>>quote blah blah block quote blah blah
block block quote blah blah block
>>>quote blah blah block quote blah blah block quote blah blah block quote
```

> blah blah block quote blah blah block quote blah blah block
>
> block quote blah blah block block quote blah blah block
>
> > quote blah blah block quote blah blah
> >
> > block block quote blah blah block
> >
> > > quote blah blah block quote blah blah block quote blah blah block quote

1.1.3 Lists

1.1.3.1 Ordered Lists

In Markdown, you can list items using numbers, a **+**, a **-**, or a *****. However, if the first item in a list or sublist is numbered, Markdown will interpret the entire list as ordered and will automatically number the items linearly, no matter what character you use to denote any given separate item.

```
####Groceries:

0. Fruit:
    6. Pears
    0. Peaches
    3. Plums
    4. Apples
        2. Granny Smith
        7. Gala
    * Oranges
    - Berries
        8. Strawberries
        + Blueberries
        * Raspberries
    - Bananas
9. Bread:
    9. Whole Wheat
        0. With oats on crust
        0. Without oats on crust
    0. Rye
    0. White
0. Dairy:
    0. Milk
        0. Whole
        0. Skim
    0. Cheese
        0. Wisconsin Cheddar
        0. Pepper Jack
```

Groceries:

1. Fruit:
    A. Pears
    B. Peaches

         C. Plums
         D. Apples
             a. Granny Smith
             b. Gala
         E. Oranges
         F. Berries
             a. Strawberries
             b. Blueberries
             c. Raspberries
         G. Bananas
   2. Bread:
         A. Whole Wheat
             a. With oats on crust
             b. Without oats on crust
         B. Rye
         C. White
   3. Dairy:
         A. Milk
             a. Whole
             b. Skim
         B. Cheese
             a. Wisconsin Cheddar
             b. Pepper Jack

1.1.3.2 Bulleted Lists

If you begin your list or sublist with a **+**, a **-**, or a *****, then Markdown will interpret the whole list as unordered and will use bullets regardless of the characters you type before any individual list item.

```
####Groceries:

* Fruit:
    * Pears
    0. Peaches
    3. Plums
```

```
        4. Apples
            - Granny Smith
            7. Gala
        * Oranges
        - Berries
            - Strawberries
            + Blueberries
            * Raspberries
        - Bananas
9. Bread:
        * Whole Wheat
            * With oats on crust
            0. Without oats on crust
        + Rye
        0. White
0. Dairy:
        * Milk
            + Whole
            0. Skim
        - Cheese
            - Wisconsin Cheddar
            0. Pepper Jack
```

Groceries:

- Fruit:
    - Pears
    - Peaches
    - Plums
    - Apples
        - Granny Smith
        - Gala
    - Oranges
    - Berries
        - Strawberries
        - Blueberries
        - Raspberries
    - Bananas

- Bread:
    - Whole Wheat
        - With oats on crust
        - Without oats on crust
    - Rye
    - White
- Dairy:
    - Milk
        - Whole
        - Skim
    - Cheese
        - Wisconsin Cheddar
        - Pepper Jack

1.1.4 Section Breaks

———

***

------

* * *

```
_ _ _
```

```
- - -
```

1.2 Backslash Escape

What happens if you want to include a literal character, like a #, that usually has a specific function in Markdown? Backslash Escape is a function that prevents Markdown from interpreting a character as an instruction, rather than as the character itself. It works like this:

```
\ Wow, this isn't a header.
# This is definitely a header.
```

Wow, this isn't a header.

This is definitely a header.

Markdown allows you to use a backslash to escape from the functions of the following characters:

- \ backslash
- ` backtick
- * asterisk
- _ underscore
- {} curly braces

- [] square brackets
- () parentheses
- # hashtag
- + plus sign|
- - minus sign (hyphen)
- . dot
- ! exclamation mark

1.3 Hyperlinks

1.3.1 Automatic Links

```
http://en.wikipedia.org
```

http://en.wikipedia.org (http://en.wikipedia.org)

1.3.2 Standard Links

```
[click this link](http://en.wikipedia.org)
```

click this link (http://en.wikipedia.org)

1.3.3 Reference Links

Suppose you are writing a document in which you intend to include many links. The format above is a little arduous and if you have to do it repeatedly *while* you're trying to focus on the content of what you're writing, it's going to be a really big pain.

Fortunately, there is an alternative way to insert hyperlinks into your text, one where you indicate that there is a link, name that link, and then use the name to provide the actually URL later on when you're less in the writing zone. This method can be thought of as a "reference-style" link because it is similar to using in-text citations and then defining those citations later in a more detailed reference section or bibliography.

```
This is [a reference] [identification tag for link]

[identification tag for link]: http://en.wikipedia.org/wiki/Chile "Wikipedia Article About Chile"
```

This is [a reference (http://en.wikipedia.org/wiki/Chile)](http://en.wikipedia.org/wiki/Chile)

Note: The "identification tag for link" can be anything. For example:

```
This is [a reference] [lfskdhflhslgfh333676]

[lfskdhflhslgfh333676]: http://en.wikipedia.org/wiki/Chile "Wikipedia Article About Chile"
```

This is [a reference (http://en.wikipedia.org/wiki/Chile)](http://en.wikipedia.org/wiki/Chile)

This means you can give your link an intuitive, easy to remember, and relevant ID:

```
This is [a reference][Chile]

[chile]: http://en.wikipedia.org/wiki/Chile "Wikipedia Article About Chile"
```

This is [a reference (http://en.wikipedia.org/wiki/Chile)](http://en.wikipedia.org/wiki/Chile)

Note: Link IDs are not case-sensitive.

If you don't want to give your link an ID, you don't have to. As a short cut, Markdown will understand if you just use the words in the first set of brackets to define the link later on. This works in the following way:

```
This is [a reference][]

[a reference]: http://en.wikipedia.org/wiki/Chile "Wikipedia Article About Chile"
```

This is [a reference (http://en.wikipedia.org/wiki/Chile)](http://en.wikipedia.org/wiki/Chile)

Another really helpful feature of a reference-style link is that you can define the link anywhere in the cell. (must be in the cell) For example:

```
This is [a reference] [ref] blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah
blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah
blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah
blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah
blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah
blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah
blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah
blah blah blah blah blah blah blah blah <br/><br/>

[ref]: http://en.wikipedia.org/wiki/Chile "Wikipedia Article About Chile"
```

This is a reference (http://en.wikipedia.org/wiki/Chile) blah

Note: Providing a mouse-over title for any link, regardless of whether it is a standard or reference-stlye type, is optional. With reference-style links, you can include the mouse-over title by placing it in quotes, single quotes, or parentheses. For standard links, you can only define a mouse-over title in quotes.

1.3.4 Notebook-Internal Links

When you create a Header you also create a discrete location within your Notebook. This means that, just like you can link to a specific location on the web, you can also link to a Header Cell inside your Notebook. Internal links have very similar Markdown formatting to regular links. The only difference is that the name of the link, which is the URL in the case of external links, is just a hashtag plus the name of the Header Cell you are linking to (case-sensitive) with dashes in between every word. If you hover your mouse over a Header Cell, a blue Greek pi letter will appear next to your title. If you click on it, the URL at the top of your window will change and the internal link to that section will appear last in the address. You can copy and paste it in order to make an internal link inside a Markdown Cell.

1.3.4.1 Standard Notebook-Internal Links

```
[Here's a link to the section of Lists](#1.1.3-Lists)
```

Here's a link to the section of Lists

1.3.4.2 Reference-Style Notebook-Internal Links

```
[Here's a link to the section of Lists][ref_link]
[ref_link]: #1.1.3-Lists
```

Here's a link to the section of Lists

1.4 Tables

In Markdown, you can make a table by using vertical bars and dashes to define the cell and header borders:

```
|Header|Header|Header|Header|
|------|------|------|------|
|Cell  |Cell  |Cell  | Cell |
|Cell  |Cell  |Cell  | Cell |
|Cell  |Cell  |Cell  | Cell |
|Cell  |Cell  |Cell  | Cell |
```

Header	Header	Header	Header
Cell	Cell	Cell	Cell
Cell	Cell	Cell	Cell
Cell	Cell	Cell	Cell
Cell	Cell	Cell	Cell

Making a table this way might be especially useful if you want your document to be legible both rendered and unrendered. However, you don't *need* to include all of those dashes, vertical bars, and spaces for Markdown to understand that you're making a table. Here's the bare minimum you would need to create the table above:

```
Header|Header|Header|Header
-|-|-|-
Cell|Cell|Cell|Cell
Cell|Cell|Cell|Cell
Cell|Cell|Cell|Cell
Cell|Cell|Cell|Cell
```

Header	Header	Header	Header
Cell	Cell	Cell	Cell
Cell	Cell	Cell	Cell
Cell	Cell	Cell	Cell
Cell	Cell	Cell	Cell

It's important to note that the second line of dashes and vertical bars is essential. If you have just the line of headers and the second line of dashes and vertical bars, that's enough for Markdown to make a table.

Another important formatting issue has to do with the vertical bars that define the left and right edges of the table. If you include all the vertical bars on the far left and right of the table, like in the first example above, Markdown will ignore them completely. *But*, if you leave out some and include others, Markdown will interpret any extra vertical bar as an additional cell on the side that the bar appears in the unrendered version of the text. This also means that if you include the far left or right vertical bar in the second line of bars and dashes, you must include all of the otherwise optional vertical bars (like in the first example above).

1.4.1 Cell Justification

If not otherwise specified the text in each header and cell of a table will justify to the left. If, however, you wish to specify either right justification or centering, you may do so like this:

```
**Centered, Right-Justified, and Regular Cells and Headers**:

centered header  |  regular header  |  right-justified header  |  centered header  |  regular header
:-:|-|-:|:-:|-
centered cell|regular cell|right-justified cell|centered cell|regular cell
centered cell|regular cell|right-justified cell|centered cell|regular cell
```

Centered, Right-Justified, and Regular Cells and Headers:

centered header	regular header	right-justified header	centered header	regular header
centered cell	regular cell	right-justified cell	centered cell	regular cell
centered cell	regular cell	right-justified cell	centered cell	regular cell

While it is difficult to see that the headers are differently justified from one another, this is just because the longest line of characters in any column defines the width of the headers and cells in that column.

Note: You cannot make tables directly beneath a line of text. You must put a blank line between the end of a paragraph and the beginning of a table.

1.5 Style and Emphasis

```
*Italics*
```

Italics

```
_Italics_
```

Italics

```
**Bold**
```

Bold

```
__Bold__
```

Bold

Note: If you want actual asterisks or underscores to appear in your text, you can use the backslash escape function like this:

```
\*awesome asterisks\* and \_incredible under scores\_
```

awesome asterisks and _incredible under scores_

1.6 Other Characters

```
Ampersand &amp;amp; Ampersand
```

Ampersand & Ampersand

```
&lt; angle brackets &gt;
```

< angle brackets >

```
&amp;quot; quotes &amp;quot;
```

" quotes "

1.7 Including Code Examples

If you want to signify that a particular section of text is actually an example of code, you can use backquotes to surround the code example. These will switch the font to monospace, which creates a clear visual formatting difference between the text that is meant to be code and the text that isn't.

Code can either in the middle of a paragraph, or as a block. Use a single backquote to start and stop code in the middle of a paragraph. Here's an example:

```
The word `monospace` will appear in a code-like form.
```

The word `monospace` will appear in a code-like form.

Note: If you want to include a literal backquote in your code example you must suround the whole text block in double backquotes like this:

```
`` Look at this literal backquote ` ``
```

Look at this literal backquote `

To include a complete code-block inside a Markdown cell, use triple backquotes. Optionally, you can put the name of the language that you are quoting after the starting triple backquotes, like this:

```
```python
def function(n):
    return n + 1
```
```

That will format the code-block (sometimes called "fenced code") with syntax coloring. The above code block will be rendered like this:

```
def function(n):
    return n + 1
```

The language formatting names that you can currently use after the triple backquote are:

```
apl            django   go             jinja2       ntriples    q        smalltalk     toml
asterisk       dtd      groovy         julia        octave      r        smarty        turtle
clike          dylan    haml           less         pascal      rpm      smartymixed   vb
clojure        ecl      haskell        livescript   pegjs       rst      solr          vbscript
cobol          eiffel   haxe           lua          perl        ruby     sparql        velocity
coffeescript   erlang   htmlembedded   markdown     php         rust     sql           verilog
commonlisp     fortran  htmlmixed                   pig         sass     stex          xml
css            gas      http           mirc         properties  scheme   tcl           xquery
d              gfm      jade           mllike       puppet      shell    tiddlywiki    yaml
diff           gherkin  javascript     nginx        python      sieve    tiki          z80
```
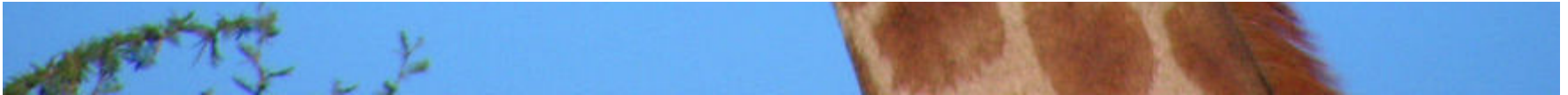
1.8 Images

1.8.1 Images from the Internet

Inserting an image from the internet is almost identical to inserting a link. You just also type a ! before the first set of brackets:

```
![It doesn't matter what you write here]
(http://upload.wikimedia.org/wikipedia/commons/thumb/b/b7/South_African_Giraffe,_head.jpg/877px-
South_African_Giraffe,_head.jpg "Picture of a Giraffe)
```

Note: Unlike with a link, the words that you type in the first set of brackets do not appear when they are rendered into html by Markdown.