

LECTURE 5

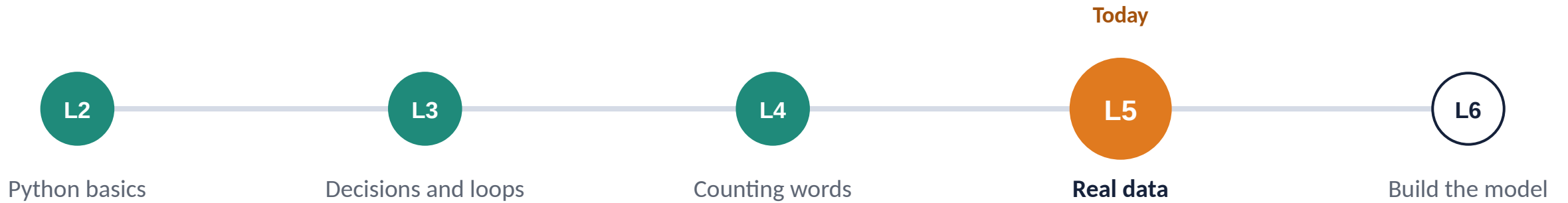
Real data with pandas

Exploring 5,572 real text messages

Today's key line of code

```
df = pd.read_csv(url, sep="\t")  
df["label"].value_counts()
```

Where we are on the road to a spam detector



Today you will

- 1 Load a real dataset from the internet with pandas
- 2 Look inside a table: rows, columns, single messages
- 3 Count spam and normal messages
- 4 Add new columns and compare groups
- 5 Clean out duplicates, and see why they matter

Meet the dataset: SMS Spam Collection

1

5,572 real messages

Real SMS messages, each labeled by people as spam or ham (normal).

2

Where from

Published in 2011 by Almeida and Hidalgo, and shared on the UCI Machine Learning Repository.

3

Its limits

Mostly UK and Singapore messages from around 2011. Today's scams and our local slang are not in it.

Load the data in one line

- `import pandas as pd` loads the library, nicknamed `pd`
- `read_csv` reads a table from a file or web link
- `sep="\t"`: columns are separated by tabs
- `df` (DataFrame) is a table; `.shape` is (rows, columns)

Code

```
import pandas as pd

url = ("https://raw.githubusercontent.com/"
      "justmarkham/pycon-2016-tutorial/"
      "master/data/sms.tsv")
df = pd.read_csv(url, sep="\t", header=None,
                 names=["label", "message"])
print(df.shape)
```

Output

```
(5572, 2)
```

In the lab (offline): load from a file

- Lab PCs have no internet, so the web link cannot work
- Copy `sms.tsv` into the same folder as your notebook
- Then give `read_csv` the file name instead of the link
- Same data, same result. At home in Colab, either way works

Code

```
df = pd.read_csv("sms.tsv", sep="\t", header=None,
                 names=["label", "message"])
print(df.shape)
```

Output

```
(5572, 2)
```

Look before you leap

`df.head()` shows the first 5 rows. Long messages are cut short with `...` on screen, but the full text is still there.

Code

```
print(df.head())
```

Output

	label	message
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

Read one message in full

- `df["message"]` selects one column
- `.loc[2, "message"]` selects row 2, column message
- Read a few messages yourself before modeling
- What clues would you use to spot spam?

Code

```
full = df.loc[2, "message"]  
print(full)  
print(len(full))
```

Output

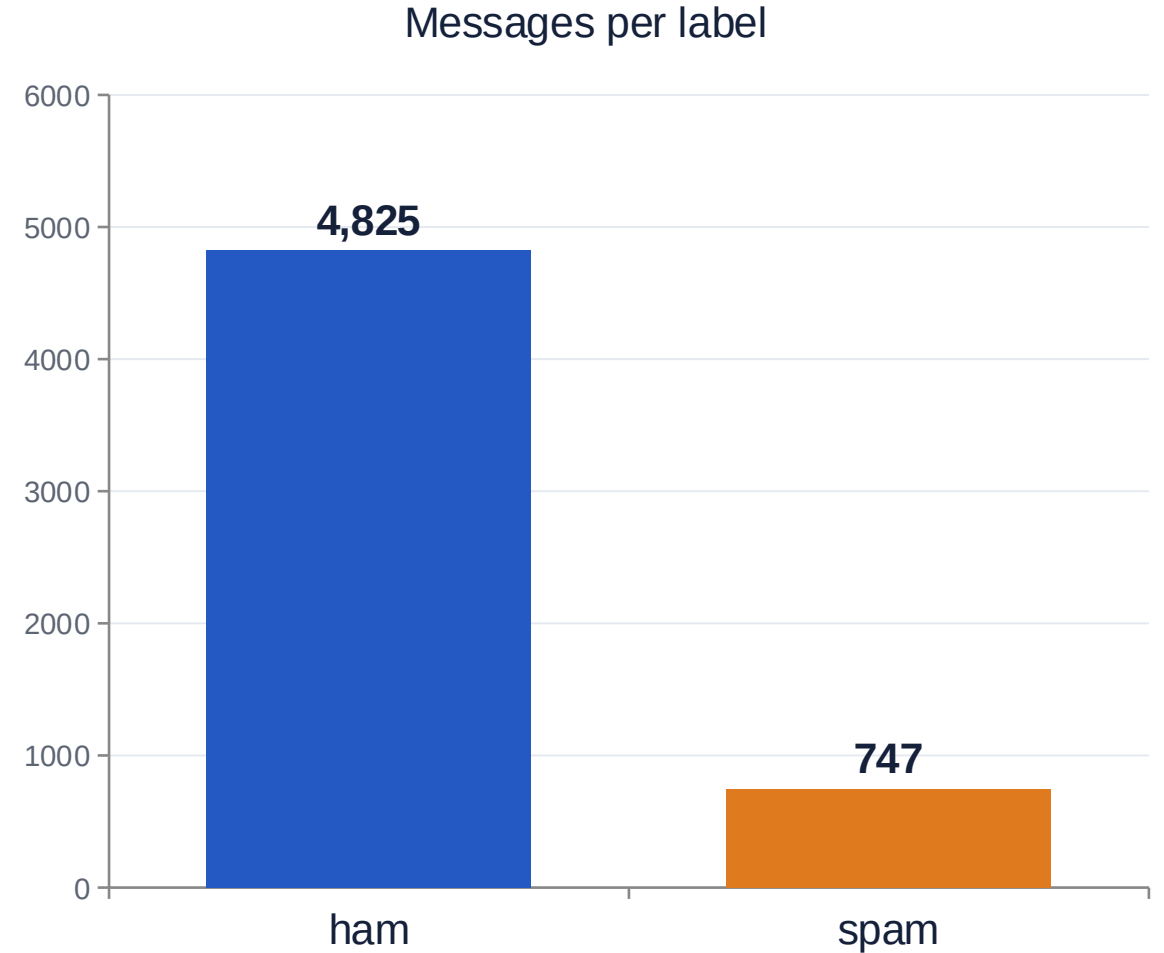
```
Free entry in 2 a wkly comp to win FA Cup final tkts 21st  
May 2005. Text FA to 87121 to receive entry question(std  
txt rate)T&C's apply 08452810075over18's  
155
```

How much spam is there?

- `value_counts()` counts each label
- Only about 13% of messages are spam
- A lazy model that always says "ham" would be about 87% right
- Remember this number: it is the score to beat

Code

```
print(df["label"].value_counts())
```

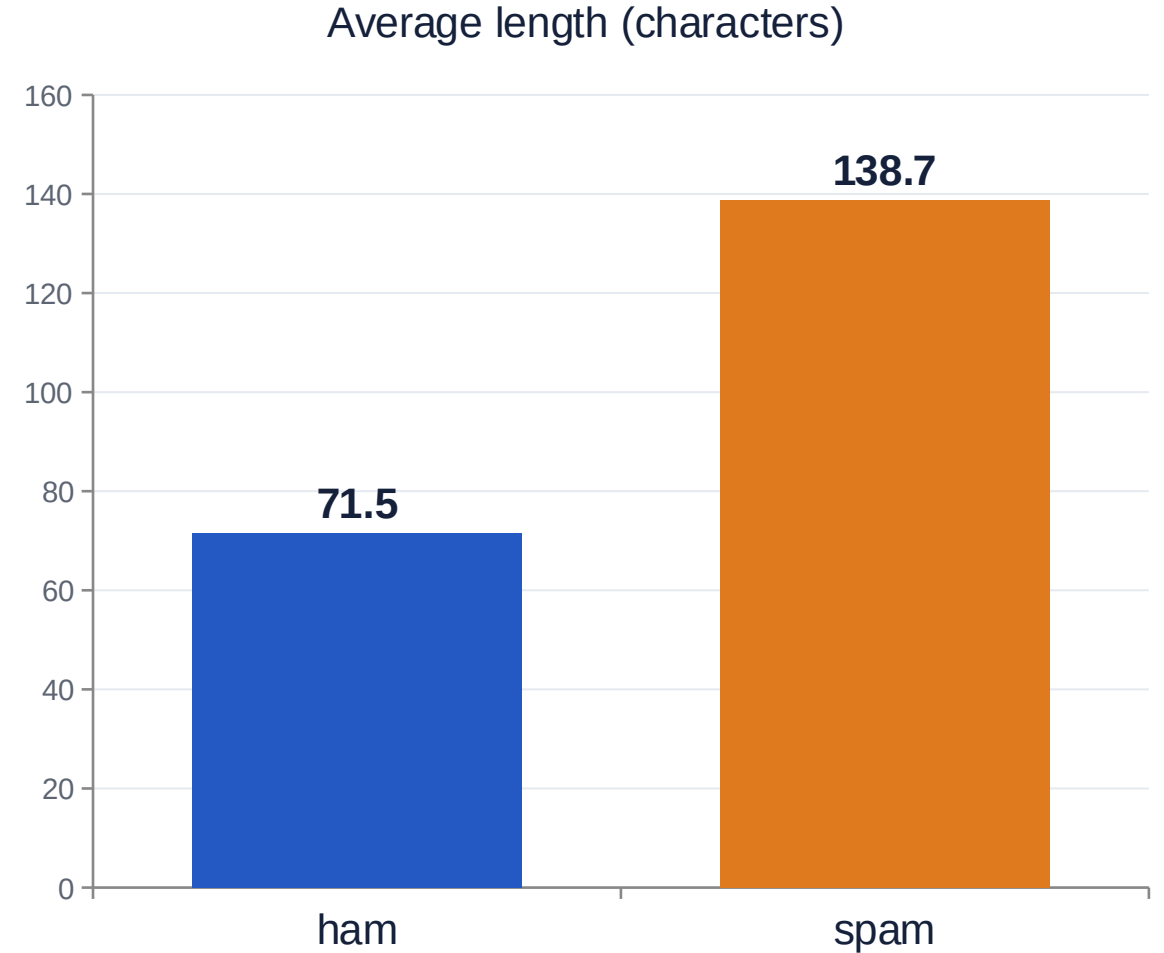


Spam messages are longer

- `.str.len()` measures every message at once
- A new column: `df["length"]`
- `groupby("label")` splits the table by label
- Spam averages about twice the length of ham

Code

```
df["length"] = df["message"].str.len()  
avg = df.groupby("label")["length"].mean()  
print(avg.round(1))
```



Watch it: filtering and grouping

WATCH FOR

- The filter asks a True/False question of every row
- Rows answering False are dropped
- groupby splits rows into one group per label
- mean() is computed inside each group

PANDAS, STEP BY STEP

What filtering and grouping do to a table

A small piece of the real SMS table: 8 rows.

Watch rows being kept, removed and grouped.

Click the video to play, click again to pause. Silent: talk the class through it.

Clean out duplicate messages

- `df.duplicated()` marks rows seen before
- 403 messages appear more than once
- Spammers send the same text to many people
- Duplicates could leak into both training and testing, so we drop them

Code

```
print(df.duplicated().sum())  
df = df.drop_duplicates()  
print(df.shape)  
print(df["label"].value_counts())
```

Output

```
403  
(5169, 3)  
label  
ham      4516  
spam     653  
Name: count, dtype: int64
```

Test a rule on real data

- `str.contains("free")` checks every message at once
- Mean of True/False = the share that is True
- "free" appears in far more spam than ham
- Yet 3 in 4 spam messages never say "free"

Code

```
text = df["message"].str.lower()
df["has_free"] = text.str.contains("free")
share = df.groupby("label")["has_free"].mean()
print(share.round(3))
```

Output

```
label
ham      0.014
spam     0.259
Name: has_free, dtype: float64
```

Your turn

7 minutes

- 1 Pick a word you think signals spam (e.g. "call", "txt", "win")
- 2 Compute the share of spam and ham containing it
- 3 Is it a better clue than "free"?
- 4 Bonus: print 3 random spam messages with `.sample(3)`

Code

```
word = "call"
df["has_word"] = text.str.contains(word)
print(df.groupby("label")["has_word"].mean())

spam = df[df["label"] == "spam"]
print(spam["message"].sample(3))
```

Common pandas mistakes

Read the last line of an error first. It names the problem.

What you typed	What Python says	What it means
<pre>print(df["Label"])</pre>	<pre>KeyError: 'Label'</pre>	Column names are case-sensitive. The column is "label".
<pre>print(df.lenght)</pre>	<pre>AttributeError: 'DataFrame' object has no attribute 'lenght'</pre>	Typo in a name. Check spelling, or <code>print(df.columns)</code> .
<pre>print(df["message"].lower())</pre>	<pre>AttributeError: 'Series' object has no attribute 'lower'</pre>	A column is not a single string. Use <code>.str.lower()</code> for text columns.

Interactive lab: Real Data Lab

In the lab you can

1 Look inside

head, tail, filters and df.loc on all 5,572 real messages.

2 Word clues

Share of spam and ham containing any word you choose.

3 Group and compare

Lengths by label, and can length alone find spam?

4 Duplicates

See what repeats and how duplicates leak into a test.

LECTURE 5 - INTERACTIVE LAB

Real Data Lab

Explore all 5,572 real text messages: look inside, test word clues, compare groups and find duplicates

Introduction to Data Science and Machine Learning
Md Mahbub E Noor - University of Barishal
[All Labs](#)

1 Look inside

2 Word clues

3 Group and compare

4 Duplicates

Word clues: is this word a good sign of spam?

For any word, see what share of spam and of normal (ham) messages contain it, exactly as `str.contains` and `groupby` compute it.

word =

Add to comparison

free

call

txt

claim

prize

now

you

u

ok

£

a.c

(

☒ after drop_duplicates() ☐ regex=False (search the text literally)

```
1 import pandas as pd
2 df = pd.read_csv("sms.tsv", sep="\t", header=None,
3                 names=["label", "message"])
4 df = df.drop_duplicates()
5 text = df["message"].str.lower()
6 df["has_word"] = text.str.contains("free")
7 share = df.groupby("label")["has_word"].mean()
8 print(share.round(3))
```

label

ham 0.014

spam 0.259

Name: has_word, dtype: float64

SHARE OF MESSAGES CONTAINING IT

ham 1.4%

spam 25.9%

YOUR COMPARISON (BEST SPAM CLUE FIRST)

word	ham	spam	spam share ÷ ham share
free	1.4%	25.9%	18.9×

Open the lab ▶

Works offline in any browser

Labs_open_in_browser/
L5_Real_Data_Lab.html

Today's toolkit

<code>import pandas as pd</code>	Load the pandas library
<code>pd.read_csv(url)</code>	Read a table from a file or link
<code>df.shape, df.head()</code>	Size, and a first look
<code>df["col"]</code>	Select one column
<code>df.loc[row, "col"]</code>	Select a single cell

<code>value_counts()</code>	Count each value
<code>df["new"] = ...</code>	Add a column
<code>df.groupby("label")</code>	Split by label to compare groups
<code>drop_duplicates()</code>	Remove repeated rows
<code>df[df["label"]=="spam"]</code>	Keep only matching rows

Homework 5

Due before Lecture 6 • [date]

1 Load it yourself

In a new notebook, load the dataset, print its shape, and count the labels.

2 Read like a detective

Print 5 spam messages with `spam["message"].sample(5, random_state=<your roll number>)`. List 3 patterns you notice.

3 Best single clue

Test 3 words of your choice. For each, print the share of spam and ham containing it. Which is the best clue, and why?

4 Make a chart

Plot average length by label with `.plot(kind="bar", title="...")`. Give it a clear title.

5 Think

A model always predicts "ham". What is its accuracy after removing duplicates? Why would it still be useless?

Submit on [LMS / Google Classroom]: your notebook file `HW5_<your ID>.ipynb`, or a Colab link set to "Anyone with the link can view".

Recap: Lecture 5 in 90 seconds

RECAP

Lecture 5 in 90 seconds

A short recap of today's lecture. Pause any time.

Click to play or pause. Watch it again at home to revise.