

# Machine Learning

## Features and Modeling

Jeff Abrahamson

9, 16 décembre 2016    13 janvier, 9 mars 2017

# Vector spaces

# Vector spaces

Features are dimensions

**Feature extraction**

**Feature engineering**









**Feature extraction**

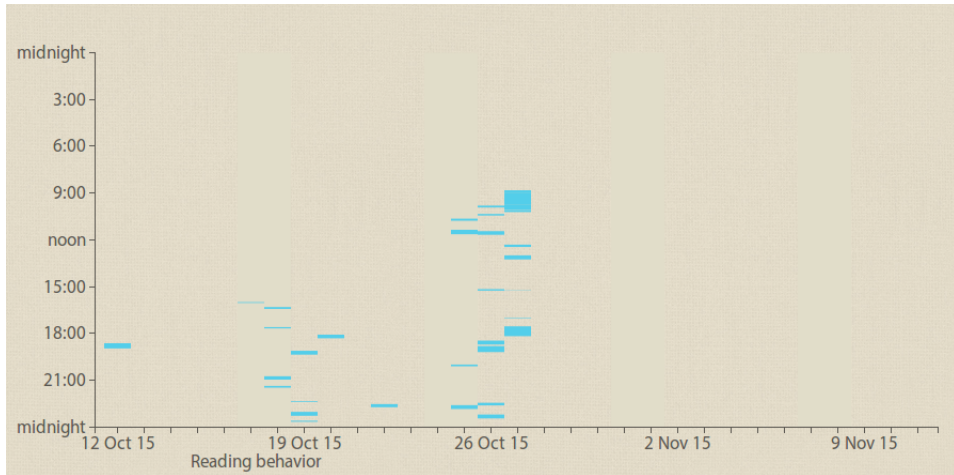
**Feature engineering**

Synthetic features

# Feature Engineering

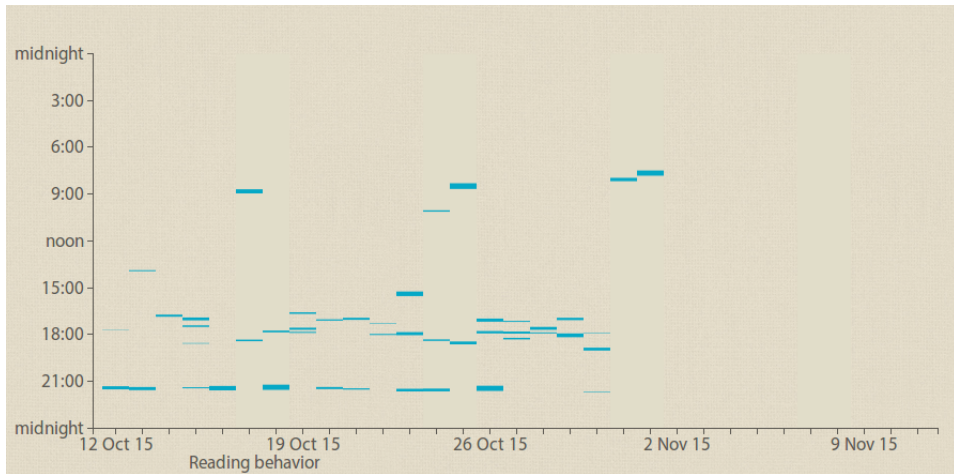
- 1 Brainstorm
- 2 Pick some
- 3 Make them
- 4 Evaluate
- 5 Repeat

Value	Count	Percent	
Mr.	517	58.025%	
Miss.	185	20.763%	
Mrs.	125	14.029%	
Master.	40	4.489%	
Dr.	7	0.786%	
Rev.	6	0.673%	
Sir.	5	0.561%	
Col.	2	0.224%	
Jonkheer.	1	0.112%	
Lady.	1	0.112%	
the Countess.	1	0.112%	
Ms.	1	0.112%	



*Jellybooks*





*Jellybooks*

**One of  $K$  = one-hot encoding**

# Text features

## Bag of words

- Corpus (documents)
- Vocabulary (set of unique words)
- Words

# Text features

## Bag of words

- Order doesn't matter
- Stop words
- Stemming (*racinisation, désuffixation*)
- Lemmatisation (*transformer en lemme*)

# Image features

- Corners, edges (rotation invariant, but scaling can hide)
- More complex: scale space or RNN
- Point matching is easy

# Image features

## Problems

- Illumination
- Scale
- Rotation
- Skew (perspective)
- Data size (matrices not sparse)

## Useful tools

- virtualenv
- pip
- ipython
- ipython notebook
- `conda.pydata.org`

## Notes

- `pip install -r requirements.txt`
- `ipython` offers tab completion (vs `python`)
- `ipython notebook` opens in a browser, caches cell output but not cell state



# pandas

```
import pandas as pd
import numpy as np
import scipy
import matplotlib.pyplot as plt
```

Dataframe has many constructors. For example,

```
In [5]: pd.DataFrame({ 'A' : 1.,
                        'B' : pd.Timestamp('20161209'),
                        'C' : pd.Series(1,index=list(range(4)),dtype='float32'),
                        'D' : np.array([3] * 4, dtype='int32'),
                        'E' : pd.Categorical(["test","train","test","train"]),
                        'F' : 'hello' })
```

Out [5]:

	A	B	C	D	E	F
0	1	2016-12-09	1	3	test	hello
1	1	2016-12-09	1	3	train	hello
2	1	2016-12-09	1	3	test	hello
3	1	2016-12-09	1	3	train	hello

In [6]:

## Viewing data

```
In [16]: dates = pd.date_range('20161209', periods=4, freq='1w')
```

```
In [17]: df = pd.DataFrame(np.random.randn(4,5), index=dates,  
                           columns=list('ABCDE'))
```

```
In [18]: df.head()
```

```
Out[18]:
```

	A	B	C	D	E
2016-12-11	-1.303610	-1.235823	0.621914	0.379340	-0.326934
2016-12-18	-1.218197	-1.113826	0.546314	-0.255001	-0.135573
2016-12-25	-0.124625	0.337268	-0.406295	0.587049	-0.904906
2017-01-01	-0.283182	-0.866213	0.051509	0.693037	-0.661055

```
In [19]:
```

## Basic data exploration

```
In [19]: df.describe()
```

```
Out [19]:
```

	A	B	C	D	E
count	4.000000	4.000000	4.000000	4.000000	4.000000
mean	-0.732403	-0.719648	0.203361	0.351106	-0.507117
std	0.614672	0.721194	0.478728	0.424558	0.342755
min	-1.303610	-1.235823	-0.406295	-0.255001	-0.904906
25%	-1.239550	-1.144325	-0.062942	0.220755	-0.722018
50%	-0.750689	-0.990019	0.298912	0.483195	-0.493995
75%	-0.243543	-0.565343	0.565214	0.613546	-0.279094
max	-0.124625	0.337268	0.621914	0.693037	-0.135573

```
In [20]:
```

## Select a column (series)

```
In [20]: df.loc[dates[1]]
```

```
Out[20]:
```

```
A    -1.218197
```

```
B    -1.113826
```

```
C     0.546314
```

```
D    -0.255001
```

```
E    -0.135573
```

```
Name: 2016-12-18 00:00:00, dtype: float64
```

```
In [21]:
```

## Select a range

```
In [21]: df.loc[:, ['A', 'C']]
```

```
Out[21]:
```

	A	C
2016-12-11	-1.303610	0.621914
2016-12-18	-1.218197	0.546314
2016-12-25	-0.124625	-0.406295
2017-01-01	-0.283182	0.051509

```
In [22]:
```

## Boolean selection criteria

```
In [23]: df[df.D > 0]
```

```
Out [23]:
```

	A	B	C	D	E
2016-12-11	-1.303610	-1.235823	0.621914	0.379340	-0.326934
2016-12-25	-0.124625	0.337268	-0.406295	0.587049	-0.904906
2017-01-01	-0.283182	-0.866213	0.051509	0.693037	-0.661055

```
In [24]:
```

## Recommended

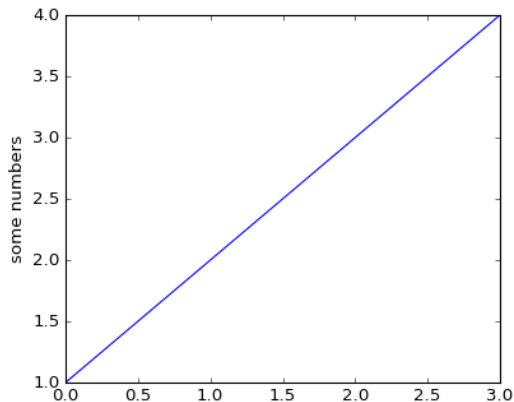
[http://www.gregreda.com/2013/10/26/  
intro-to-pandas-data-structures/](http://www.gregreda.com/2013/10/26/intro-to-pandas-data-structures/)



# Plotting

## Draw a line

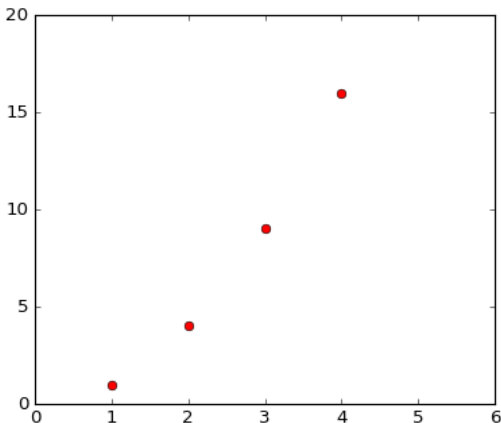
```
import matplotlib.pyplot as plt  
plt.plot([1,2,3,4])  
plt.ylabel('some numbers')  
plt.show()
```



# Plotting

## Draw a line

```
import matplotlib.pyplot as plt  
plt.plot([1, 2, 3, 4], [1, 4, 9, 16])  
plt.ylabel('some numbers')  
plt.show()
```



# Plotting

## Draw a line

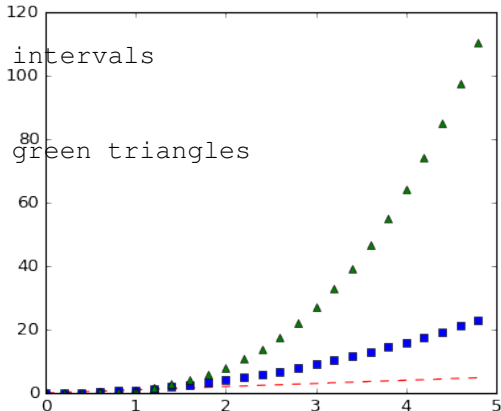
```
import numpy as np
import matplotlib.pyplot as plt
```

```
# evenly sampled time at 200ms intervals
t = np.arange(0., 5., 0.2)
```

```
# red dashes, blue squares and green triangles
```

```
plt.plot(t, t,
         'r--', t,
         t**2, 'bs',
         t, t**3, 'g^')
```

```
plt.show()
```



# Plotting

## Draw two curves

```
import numpy as np
import matplotlib.pyplot as plt

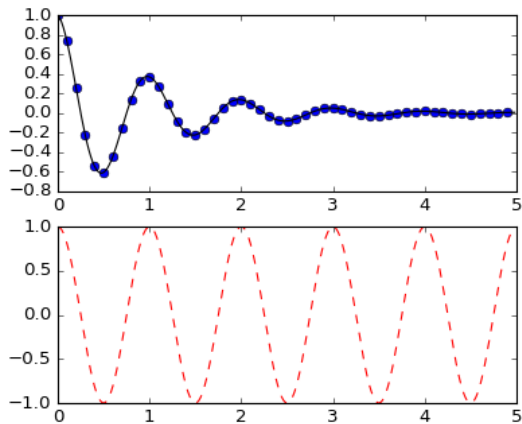
def f(t):
    return np.exp(-t) * np.cos(2*np.pi*t)

t1 = np.arange(0.0, 5.0, 0.1)
t2 = np.arange(0.0, 5.0, 0.02)

plt.figure(1)
plt.subplot(211)
plt.plot(t1, f(t1), 'bo', t2, f(t2), 'k')

plt.subplot(212)
plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
plt.show()
```

# Plotting



# Plotting

## Draw two curves

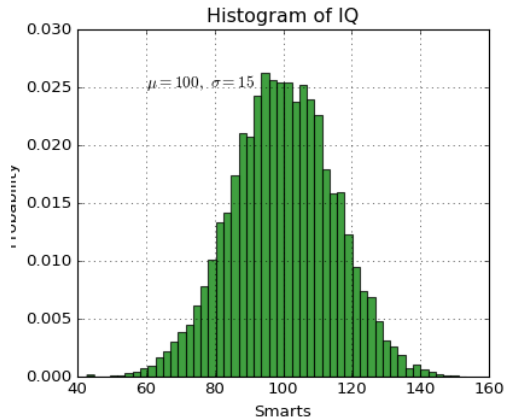
```
import numpy as np
import matplotlib.pyplot as plt

mu, sigma = 100, 15
x = mu + sigma * np.random.randn(10000)

n, bins, patches = plt.hist(x, 50, normed=1, facecolor='g', alpha=0.75)

plt.xlabel('Smarts')
plt.ylabel('Probability')
plt.title('Histogram of IQ')
plt.text(60, .025, r'$\mu=100,\ \sigma=15$')
plt.axis([40, 160, 0, 0.03])
plt.grid(True)
plt.show()
```

# Plotting



# Plotting

## Scatter plot

[http://matplotlib.org/mpl\\_examples/pylab\\_examples/scatter\\_demo2.py](http://matplotlib.org/mpl_examples/pylab_examples/scatter_demo2.py)

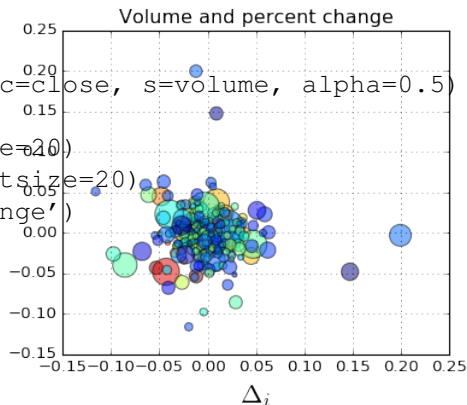
```
import numpy as np
import matplotlib.pyplot as plt
```

```
fig, ax = plt.subplots()
ax.scatter(delta1[:-1], delta1[1:], c=close, s=volume, alpha=0.5)

ax.set_xlabel(r'$\Delta_i$', fontsize=20)
ax.set_ylabel(r'$\Delta_{i+1}$', fontsize=20)
ax.set_title('Volume and percent change')

ax.grid(True)
fig.tight_layout()

plt.show()
```





# Plotting

*[http://matplotlib.org/users/pyplot\\_tutorial.html](http://matplotlib.org/users/pyplot_tutorial.html)*

*<http://matplotlib.org/users/beginner.html>*



**Questions?**