# Confidence and lift

## MARKET BASKET ANALYSIS IN PYTHON

**Isaiah Hull**

Visiting Associate Professor of Finance,
BI Norwegian Business School

# When support is misleading

| TID | Transaction |
|-----|-------------|
| 1 | Coffee, Milk |
| 2 | Bread, Milk, Orange |
| 3 | Bread, Milk |
| 4 | Bread, Milk, Sugar |
| 5 | Bread, Jam, Milk |
| ... | ... |

1. **Milk and bread frequently purchased together.**
   - Support: {Milk} → {Bread}

2. **Rule is not informative for marketing.**
   - Milk and bread are both popular items.

# The confidence metric

1. Can improve over **support** with additional metrics.

2. Adding **confidence** provides a more complete picture.

$$\frac{Support(X\&Y)}{Support(X)}$$

# Interpreting the confidence metric

| TID | Transaction |
|-----|-------------|
| 1 | Coffee, Milk |
| 2 | Bread, Milk, Orange |
| 3 | Bread, Milk |
| 4 | Bread, Milk, Sugar |
| 5 | Bread, Jam, Milk |

| TID | Transaction |
|-----|-------------|
| 1 | Coffee, Milk |
| 2 | Bread, Milk, Orange |
| 3 | Bread, Milk |
| 4 | Bread, Milk, Sugar |
| 5 | Bread, Jam, Milk |

$$Support(Milk \& Coffee) = 0.20$$

$$Support(Milk) = 1.00$$

# Interpreting the confidence metric

$$\frac{Support(Milk\&Coffee)}{Support(Milk)} = \frac{0.20}{1.00} = 0.20$$

$$\frac{Support(Milk\&Coffee)}{Support(Milk)} = \frac{0.20}{0.80} = 0.25$$

$$\frac{Support(Milk\&Coffee)}{Support(Milk)} = \frac{0.20}{0.20} = 1.00$$

# The lift metric

- **Lift provides another metric for evaluating the relationship between items.**
  - **Numerator:** Proportion of transactions that contain X and Y.

  - **Denominator:** Proportion if X and Y assigned randomly and independently.

$$\frac{Support(X\&Y)}{Support(X)Support(Y)}$$

# Preparing the data

```python
from mlxtend.preprocessing import TransactionEncoder
import pandas as pd

# Split library strings into lists
libraries = data['Library'].apply(lambda t: t.split(','))

# Convert to list of lists
libraries = list(libraries)

# One-hot encode books
books = TransactionEncoder().fit(libraries).transform(libraries)

# Convert one-hot encoded data to DataFrame
books = pd.DataFrame(books, columns = encoder.columns_)
```

# Computing confidence and lift

```python
# Print first five items
print(books.head())
```

```
       Hunger          Gatsby
0      False           True
1      False           True
2      False           False
3      False           True
4      False           True
```

**Dataset: GoodBooks-10K.**

# Computing confidence and lift

```python
# Computing support.
supportHG = np.logical_and(books['Hunger'],books['Gatsby']).mean()

supportH = books['Hunger'].mean()

supportG = books['Gatsby'].mean()
```

```python
# Compute and print confidence and lift.
confidence = supportHG / supportH

lift = supportHG / (supportH * supportG)
```

```python
# Print results.
print(supportG, confidence, lift)
```

```
(0.30, 0.16, 0.53)
```

# Let's practice!

## MARKET BASKET ANALYSIS IN PYTHON

# Leverage and conviction

## MARKET BASKET ANALYSIS IN PYTHON

**Isaiah Hull**

Visiting Associate Professor of Finance,
BI Norwegian Business School

# Building on simpler metrics

$$Support(X) = \frac{Frequency(X)}{N}$$

$$Support(X \rightarrow Y) = \frac{Frequency(X \& Y)}{N}$$

$$Confidence(X \rightarrow Y) = \frac{Support(X \rightarrow Y)}{Support(X)}$$

$$Lift(X \rightarrow Y) = \frac{Support(X \rightarrow Y)}{Support(X)Support(Y)}$$

# The leverage metric

- Leverage also builds on **support**.

$$Leverage(X \rightarrow Y) =$$

$$Support(X\&Y) - Support(X)Support(Y)$$

- Leverage is similar to lift, but **easier to interpret.**

- Leverage **lies in -1 and +1 range.**
  - **Lift** ranges from 0 to infinity.

# Computing leverage

```python
# Compute support for Twilight and Harry Potter
supportTP = np.logical_and(books['Twilight'], books['Potter']).mean()


# Compute support for Twilight
supportT = books['Twilight'].mean()


# Compute support for Harry Potter
supportP = books['Potter'].mean()
```

```python
# Compute and print leverage
leverage = supportTP - supportP * supportT
print(leverage)
```
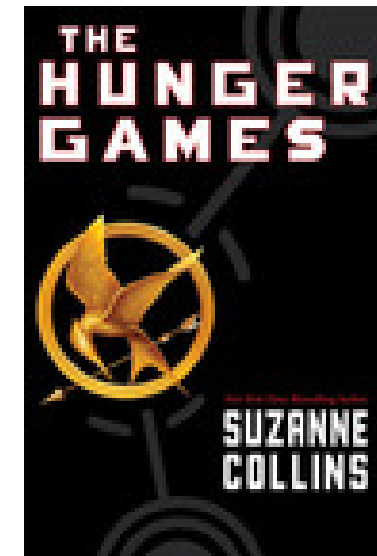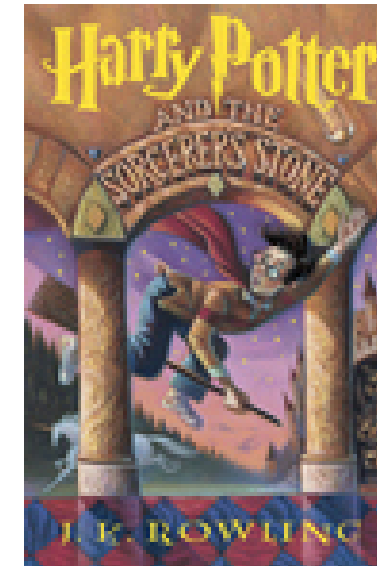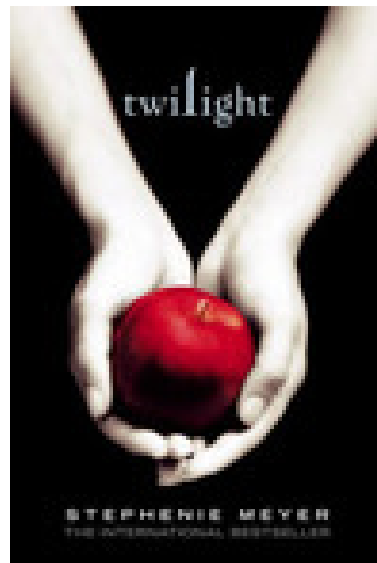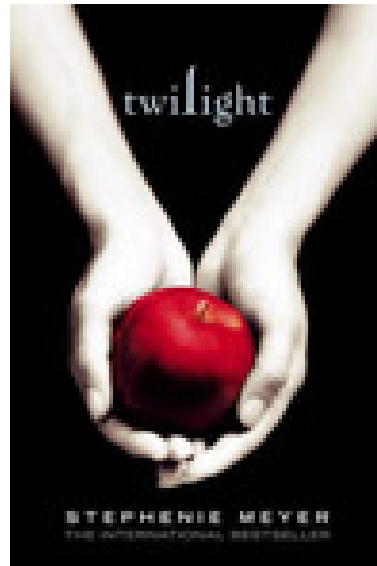
```
0.018
```

# The conviction metric

1. **Conviction** is also built using support.

2. More complicated and less intuitive than **leverage**.

$$Conviction(X \rightarrow Y) =$$

$$\frac{Support(X)Support(\bar{Y})}{Support(X\&\bar{Y})}$$

# Interpreting conviction

# Computing conviction

```python
# Compute support for Twilight and Harry Potter and Twilight
supportTP = np.logical_and(books['Twilight'], books['Potter']).mean()
supportT = books['Twilight'].mean()
```

```python
# Compute support for NOT Harry Potter
supportnP = 1.0 - books['Potter'].mean()
```

```python
# Compute support for Twilight and NOT Harry Potter
supportTnP = supportT - supportPT
```

```python
# Compute conviction
conviction = supportT*supportnP / supportTnP
print(conviction)
```

```
1.16
```

# Let's practice!

MARKET BASKET ANALYSIS IN PYTHON
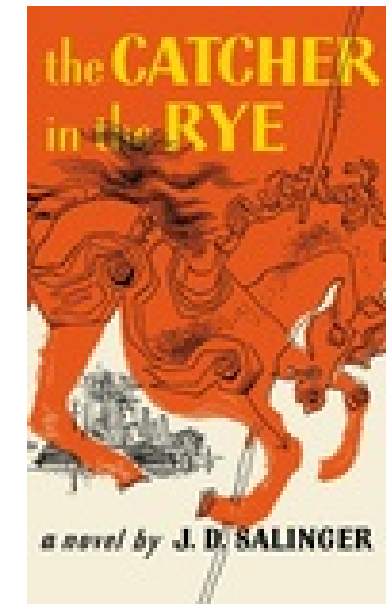
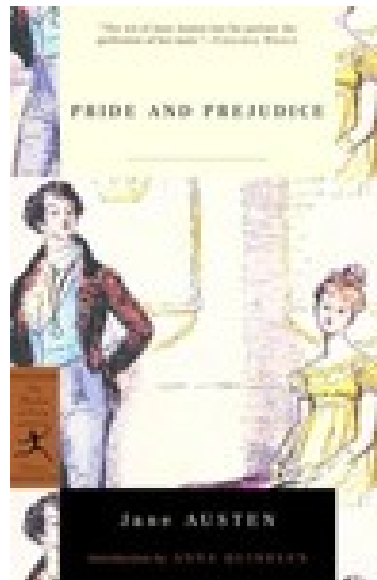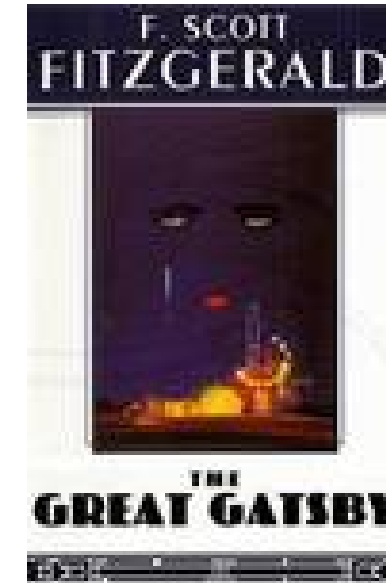# Association and dissociation

## MARKET BASKET ANALYSIS IN PYTHON

**Isaiah Hull**

Visiting Associate Professor of Finance,
BI Norwegian Business School

# Using dissociation to pair ebooks
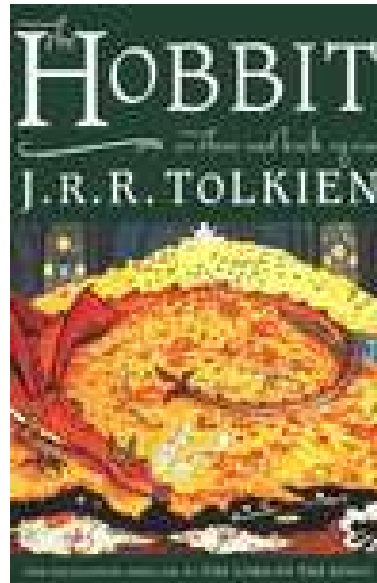
# Introduction to Zhang's metric

1. **Introduced by Zhang (2000)**
   - Takes values between -1 and +1
   - Value of +1 indicates perfect association
   - Value of -1 indicates perfect dissociation

2. **Comprehensive and interpretable**

3. **Constructed using support**

[1] Zhang, T. (2000). Association Rules. Proceedings of the 4th Pacific-Asia conference, PADKK, pp.245-256. Kyoto, Japan.

# Defining Zhang's metric

$$Zhang(A \rightarrow B) =$$

$$\frac{Confidence(A \rightarrow B) - Confidence(\bar{A} \rightarrow B)}{Max[Confidence(A \rightarrow B), Confidence(\bar{A} \rightarrow B)]}$$

$$Confidence = \frac{Support(A\&B)}{Support(A)}$$

# Constructing Zhang's metric using support

$$Zhang(A \to B) =$$

$$\frac{Support(A\&B) - Support(A)Support(B)}{Max[Support(AB)(1 - Support(A)), Support(A)(Support(B) - Support(AB))]}$$

# Computing Zhang's metric

```python
# Compute the support of each book
supportH = hobbit.mean()

supportP = pride.mean()
```

```python
# Compute the support of both books
supportHP = np.logical_and(hobbit, pride).mean()
```

# Computing Zhang's metric

```python
# Compute the numerator
num = supportHP - supportH*supportP
```

```python
# Compute the denominator
denom = max(supportHP*(1-supportH), supportH*(supportP-supportHP))
```

```python
# Compute Zhang's metric
zhang = num / denom
print(zhang)
```

```
0.08903
```

# Let's practice!

## MARKET BASKET ANALYSIS IN PYTHON

# Advanced rules

## MARKET BASKET ANALYSIS IN PYTHON

**Isaiah Hull**

Visiting Associate Professor of Finance,
BI Norwegian Business School

# Overview of market basket analysis

- **Standard procedure for market basket analysis.**
  1. Generate large set of rules.

  2. Filter rules using metrics.

  3. Apply intuition and common sense.

# Generating rules

- **Number of rules grows exponentially in number of items.**
  - Most rules are not useful.

- **Must apply initial round of filtering.**
  - Covered in chapter 3 using Apriori algorithm

# How does filtering work?

| ID | antecedents | consequents | support | Lift |
|----|-------------|-------------|---------|------|
| 1 | Harry Potter | The Hunger Games | **0.001** | 1.01 |
| 2 | Hunger Games | Twilight | 0.020 | 1.23 |
| 3 | Pride and Prejudice | The Hobbit | 0.030 | 1.05 |
| 4 | The Hobbit | Twilight | 0.015 | **0.85** |
| 5 | Harry Potter | The Hobbit | 0.010 | 1.07 |

# Multi-metric filtering

| ID | antecedents | consequents | support | zhang |
|---|---|---|---|---|
| 1 | Harry Potter | The Hunger Games | 0.001 | -0.05 |
| 2 | Hunger Games | Twilight | 0.020 | 0.17 |
| 3 | Pride and Prejudice | The Hobbit | 0.030 | 0.06 |
| 4 | The Hobbit | Twilight | 0.015 | -0.04 |
| 5 | Harry Potter | The Hobbit | 0.010 | 0.34 |

# Performing multi-metric filtering

```
print(rules.head())
```

```
  antecedents consequents  antecedent support  ...  support  confidence  ...  conviction
0 Potter      Hunger       0.48                ...  0.12  ... 0.26       ...  0.92
1 Potter      Hunger       0.32                ...  0.12  ... 0.26       ...  0.92
2 Twilight    Hunger       0.26                ...  0.09  ... 0.35       ...  1.04
3 Hunger      Twilight     0.32                ...  0.09  ... 0.28       ...  1.03
4 Mockingbird Hunger       0.48                ...  0.10  ... 0.20       ...  0.85
```

# Performing multi-metric filtering

```python
# Select subset of rules with low consequent support.
rules = rules[rules['consequent support'] < 0.05]
print(len(rules))
```

```
12
```

```python
# Select subset of rules with lift > 1.5.
rules = rules[rules['lift'] > 1.5]
print(len(rules))
```

```
2
```

# Let's practice!

## MARKET BASKET ANALYSIS IN PYTHON