150B/355B
Introduction to Machine Learning for Social Science
TA Section 6

Haemin Jee and Tongtong Zhang

February 16, 2018

Road Map

1. K Nearest Neighbor

Road Map

1. K Nearest Neighbor
2. Text Preprocessing

Road Map

1. K Nearest Neighbor
2. Text Preprocessing
3. Dictionary method

Road Map

1. K Nearest Neighbor
2. Text Preprocessing
3. Dictionary method
4. R Exercise

K Nearest Neighbors

We assume some relationship between label Y and predictors $X = (X_1, X_2, ..., X_p)$, such that:

$$Y = f(X) + \epsilon$$

where f is fixed but unknown function of $X_1, ..., X_p$, and ϵ is a random error term that is is independent of X and has mean zero. .

K Nearest Neighbors

We assume some relationship between label Y and predictors $X = (X_1, X_2, ..., X_p)$, such that:

$$Y = f(X) + \epsilon$$

where f is fixed but unknown function of $X_1, ..., X_p$, and ϵ is a random error term that is is independent of X and has mean zero. .

Machine learning: estimating f with $\hat{f}$.

K Nearest Neighbors

In the first half of the course, we have assumed the functional forms of f as:

- Linear probability regression
- Logit regression
- LASSO

K Nearest Neighbors

In the first half of the course, we have assumed the functional forms of f as:

- Linear probability regression
- Logit regression
- LASSO

$\rightsquigarrow$ parametric models

- Advantage: simplifies estimation of f, interpretability of the relationship between X and Y
- Disadvantage: what if our assumption of f is wrong?

K Nearest Neighbors

In the first half of the course, we have assumed the functional forms of f as:

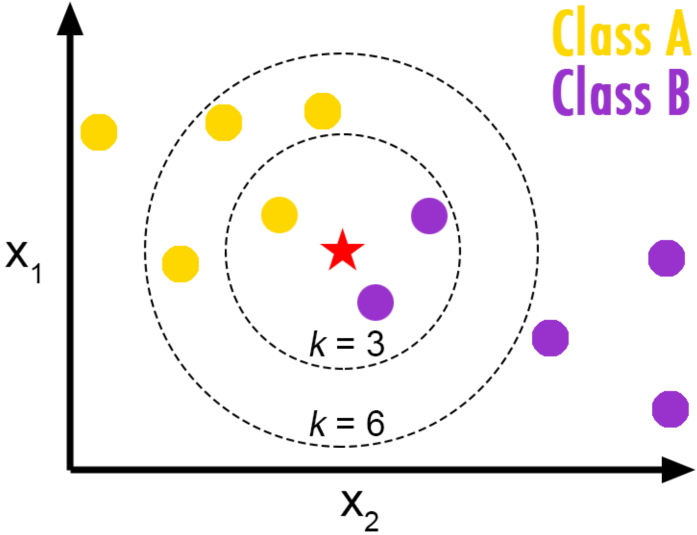
- Linear probability regression
- Logit regression
- LASSO

$\rightsquigarrow$ parametric models

- Advantage: simplifies estimation of f, interpretability of the relationship between X and Y
- Disadvantage: what if our assumption of f is wrong? $\rightsquigarrow$ nonparametric models

K Nearest Neighbors

KNN: a nonparametric model



K Nearest Neighbors

Nonparametric model

- No explicit assumption about the functional form of f

K Nearest Neighbors

Nonparametric model

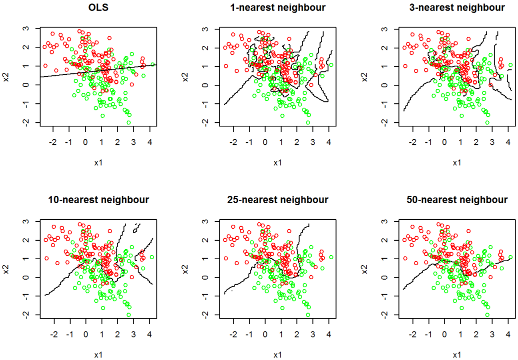
- No explicit assumption about the functional form of f
- No fixed parameters (no β coefficients)

K Nearest Neighbors

Nonparametric model

- No explicit assumption about the functional form of f
- No fixed parameters (no β coefficients)
- No or little training: all computation delayed until prediction

K Nearest Neighbors



Choice of K has a drastic effect on the KNN classifier:

As K increases, the decision boundary is less and less flexible (higher bias, lower variance)

Text Preprocessing

Task: Convert raw documents into a document-term matrix

Text Preprocessing

Task: Convert raw documents into a document-term matrix

- Step 1: Convert raw documents (html, pdf) into a corpus

Text Preprocessing

Task: Convert raw documents into a document-term matrix

- Step 1: Convert raw documents (html, pdf) into a corpus (a csv file in which each document is a row, one column is text, other columns are metadata like author, date)

Text Preprocessing

Task: Convert raw documents into a document-term matrix

- Step 1: Convert raw documents (html, pdf) into a corpus (a csv file in which each document is a row, one column is text, other columns are metadata like author, date)
- Step 2: Pre-processing

Text Preprocessing

Task: Convert raw documents into a document-term matrix

- Step 1: Convert raw documents (html, pdf) into a corpus (a csv file in which each document is a row, one column is text, other columns are metadata like author, date)
- Step 2: Pre-processing
    1) Remove capitalization, punctuation
    2) Discard Word Order: (Bag of Words Assumption)
    3) Discard stop words
    4) Combine similar terms: Stem, Lemmatize
    5) Discard less useful features $\rightsquigarrow$ depends on application
    6) Other reduction, weighting
    7) Output: a Count vector for each document, each element counts occurrence of terms

Text Preprocessing

```
Four score and seven years ago our fathers brought forth on
this continent a new nation, conceived in liberty, and
dedicated to the proposition that all men are created
equal.
```

Text Preprocessing

`Four score and seven years ago our fathers brought forth on this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal.`

Step 1: Remove capitalization and punctuation:

Text Preprocessing

Four score and seven years ago our fathers brought forth on this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal.

Step 1: Remove capitalization and punctuation:

four score and seven years ago our fathers brought forth on this continent a new nation conceived in liberty and dedicated to the proposition that all men are created equal

Text Preprocessing

Step 1: Remove capitalization and punctuation:

`four score and seven years ago our fathers brought forth on this continent a new nation conceived in liberty and dedicated to the proposition that all men are created equal`

Step 2: Discard word order - Tokenize:

Text Preprocessing

Step 1: Remove capitalization and punctuation:

`four score and seven years ago our fathers brought forth on this continent a new nation conceived in liberty and dedicated to the proposition that all men are created equal`

Step 2: Discard word order - Tokenize:

- Words often imply the theme of the text

Text Preprocessing

Step 1: Remove capitalization and punctuation:

`four score and seven years ago our fathers brought forth on this continent a new nation conceived in liberty and dedicated to the proposition that all men are created equal`

Step 2: Discard word order - Tokenize:

- Words often imply the theme of the text
- Word order could be critical for classification

Text Preprocessing

Step 1: Remove capitalization and punctuation:

```
four score and seven years ago our fathers brought forth on
this continent a new nation conceived in liberty and
dedicated to the proposition that all men are created equal
```

Step 2: Discard word order - Tokenize:

- Words often imply the theme of the text
- Word order could be critical for classification
  - Unigram, Bigram, Trigram

Text Preprocessing

Step 1: Remove capitalization and punctuation:

`four score and seven years ago our fathers brought forth on this continent a new nation conceived in liberty and dedicated to the proposition that all men are created equal`

Step 2: Discard word order - Tokenize:

- Words often imply the theme of the text
- Word order could be critical for classification
    - Unigram, Bigram, Trigram
    - Validation by human coders

Text Preprocessing

Step 1: Remove capitalization and punctuation:
Step 2: Discard word order - Tokenize:

`four, score, and, seven, years, ago, our, fathers, brought, forth, on, this, continent, a, new, nation, conceived, in, liberty, and, dedicated, to, the, proposition, that, all, men, are, created, equal`

$\vdots$

Text Preprocessing

Step 1: Remove capitalization and punctuation:
Step 2: Discard word order - Tokenize:

`four, score, and, seven, years, ago, our, fathers, brought, forth, on, this, continent, a, new, nation, conceived, in, liberty, and, dedicated, to, the, proposition, that, all, men, are, created, equal`

Step 3: Remove stop words:

Text Preprocessing

Step 1: Remove capitalization and punctuation:

Step 2: Discard word order - Tokenize:

`four, score, and, seven, years, ago, our, fathers, brought, forth, on, this, continent, a, new, nation, conceived, in, liberty, and, dedicated, to, the, proposition, that, all, men, are, created, equal`

Step 3: Remove stop words:

- Stop words: `the, it, if, a, able, at, be, because...`

Text Preprocessing

Step 1: Remove capitalization and punctuation:

Step 2: Discard word order - Tokenize:

Step 3: Remove stop words:

`four, score, seven, years, ago, fathers, brought, forth, continent, new, nation, conceived, liberty, dedicated, proposition, men, created, equal`

Step 4: Applying Stemming/Lemmatizing

Text Preprocessing

Step 1: Remove capitalization and punctuation:

Step 2: Discard word order - Tokenize:

Step 3: Remove stop words:

`four, score, seven, years, ago, fathers, brought, forth, continent, new, nation, conceived, liberty, dedicated, proposition, men, created, equal`

Step 4: Applying Stemming/Lemmatizing

- Combine words with the same root

  `family, families, familial` $\rightarrow$ `famili`

Text Preprocessing

Step 1: Remove capitalization and punctuation:

Step 2: Discard word order - Tokenize:

Step 3: Remove stop words:

`four, score, seven, years, ago, fathers, brought, forth, continent, new, nation, conceived, liberty, dedicated, proposition, men, created, equal`

Step 4: Applying Stemming/Lemmatizing

- Combine words with the same root

  `family, families, familial` → `famili`

- Different ways to chop off end of word: Porter/Snowball/Lancaster stemmer

Text Preprocessing

Step 1: Remove capitalization and punctuation:
Step 2: Discard word order - Tokenize:
Step 3: Remove stop words:
Step 4: Applying Stemming/Lemmatizing

`four, score, seven, year, ago, father, brought, forth, contin, new, nation, conceiv, liberti, dedic, proposit, men, creat, equal`

Text Preprocessing

Step 1: Remove capitalization and punctuation:
Step 2: Discard word order - Tokenize:
Step 3: Remove stop words:
Step 4: Applying Stemming/Lemmatizing

`four, score, seven, year, ago, father, brought, forth, contin, new, nation, conceiv, liberti, dedic, proposit, men, creat, equal`

Step 5: Create Count Vector

Text Preprocessing

Step 1: Remove capitalization and punctuation:

Step 2: Discard word order - Tokenize:

Step 3: Remove stop words:

Step 4: Applying Stemming/Lemmatizing

`four, score, seven, year, ago, father, brought, forth, contin, new, nation, conceiv, liberti, dedic, proposit, men, creat, equal`

Step 5: Create Count Vector

Doc no.	ago	brought	seven	creat	conceiv	men	father
Doc 1	1	1	1	1	1	1	1
⋮	⋮						

Text Preprocessing

Step 1: Remove capitalization and punctuation:

Step 2: Discard word order - Tokenize:

Step 3: Remove stop words:

Step 4: Applying Stemming/Lemmatizing

Step 5: Create Count Vector

Doc no.	ago	brought	seven	creat	conceiv	men	father
Doc 1	1	1	1	1	1	1	1
⋮	⋮						

Step 6: Repeat steps 1-5 for each document

Step 1: Remove capitalization and punctuation:

Step 2: Discard word order - Tokenize:

Step 3: Remove stop words:

Step 4: Applying Stemming/Lemmatizing

Step 5: Create Count Vector

Doc no.	ago	brought	seven	creat	conceiv	men	father
Doc 1	1	1	1	1	1	1	1
⋮	⋮						

Step 6: Repeat steps 1-5 for each document

`R Code, Section 1!`

Dictionary Method

Task: classify (measure) sentiment in texts (positive vs. negative; anger, fear, joy,...)

Method: Dictionary methods

Dictionary Method

Task: classify (measure) sentiment in texts (positive vs. negative; anger, fear, joy,...)
Method: Dictionary methods

Logic: Counting words – the higher proportion of words fall into an emotion category, the more likely that the document expresses that emotion.

Dictionaries

```
## 1      2-faced  negative
## 2      2-faces  negative
## 3           a+  positive
## 4     abnormal  negative
## 5      abolish  negative
## 6   abominable  negative
## 7   abominably  negative
## 8    abominate  negative
## 9  abomination  negative
## 10       abort  negative
```

- Dictionaries are dataframes where,
  - Each row is a word

Dictionaries

```
## 1       2-faced  negative
## 2       2-faces  negative
## 3            a+  positive
## 4      abnormal  negative
## 5       abolish  negative
## 6    abominable  negative
## 7    abominably  negative
## 8     abominate  negative
## 9   abomination  negative
## 10        abort  negative
```

- Dictionaries are dataframes where,
  - Each row is a word
  - One column is the cateogry (positive vs. negative, sports vs. food vs. others)

Dictionaries

```
## 1      2-faced  negative
## 2      2-faces  negative
## 3           a+  positive
## 4     abnormal  negative
## 5      abolish  negative
## 6   abominable  negative
## 7   abominably  negative
## 8    abominate  negative
## 9  abomination  negative
## 10       abort  negative
```

- Dictionaries are dataframes where,
  - Each row is a word
  - One column is the cateogry (positive vs. negative, sports vs. food vs. others)
- Sentiment dictionaries:
  - Broad Sentiment (Positive, Negative)

Dictionaries

```
## 1      2-faced  negative
## 2      2-faces  negative
## 3           a+  positive
## 4     abnormal  negative
## 5      abolish  negative
## 6   abominable  negative
## 7   abominably  negative
## 8    abominate  negative
## 9  abomination  negative
## 10       abort  negative
```

- Dictionaries are dataframes where,
  - Each row is a word
  - One column is the cateogry (positive vs. negative, sports vs. food vs. others)
- Sentiment dictionaries:
  - Broad Sentiment (Positive, Negative)
  - Nuanced Emotion (Anger, Joy, Sadness)

Dictionaries

```
## 1      2-faced  negative
## 2      2-faces  negative
## 3           a+  positive
## 4     abnormal  negative
## 5      abolish  negative
## 6   abominable  negative
## 7   abominably  negative
## 8    abominate  negative
## 9  abomination  negative
## 10       abort  negative
```

- Dictionaries are dataframes where,
  - Each row is a word
  - One column is the cateogry (positive vs. negative, sports vs. food vs. others)
- Sentiment dictionaries:
  - Broad Sentiment (Positive, Negative)
  - Nuanced Emotion (Anger, Joy, Sadness)
  - Not all English words included (many are neutral)

Dictionaries

```
## 1      2-faced  negative
## 2      2-faces  negative
## 3           a+  positive
## 4     abnormal  negative
## 5      abolish  negative
## 6   abominable  negative
## 7   abominably  negative
## 8    abominate  negative
## 9  abomination  negative
## 10       abort  negative
```

- Dictionaries are dataframes where,
  - Each row is a word
  - One column is the cateogry (positive vs. negative, sports vs. food vs. others)
- Sentiment dictionaries:
  - Broad Sentiment (Positive, Negative)
  - Nuanced Emotion (Anger, Joy, Sadness)
  - Not all English words included (many are neutral)
- Word weights / scores
  - Binary: $\{\text{Positive } (+1), \text{Negative } (\text{-}1)\}$

Dictionaries

```
## 1     abandon    -2
## 2   abandoned    -2
## 3    abandons    -2
## 4    abducted    -2
## 5   abduction    -2
## 6 abductions    -2
## 7       abhor    -3
## 8    abhorred    -3
## 9   abhorrent    -3
## 10     abhors    -3
```

- Dictionaries are dataframes where,
  - Each row is a word
  - One column is the cateogry (positive vs. negative, sports vs. food vs. others)
- Sentiment dictionaries:
  - Broad Sentiment (Positive, Negative)
  - Nuanced Emotion (Anger, Joy, Sadness)
  - Not all English words included (many are neutral)
- Word weights / scores
  - Binary: {Positive (+1), Negative (-1)}
  - Numerical: $\{-2, -1, 1, 2\}$

Dictionaries

```
## 1     abandon    -2
## 2   abandoned    -2
## 3    abandons    -2
## 4    abducted    -2
## 5   abduction    -2
## 6 abductions    -2
## 7       abhor    -3
## 8    abhorred    -3
## 9   abhorrent    -3
## 10     abhors    -3
```

- Dictionaries are dataframes where,
  - Each row is a word
  - One column is the cateogry (positive vs. negative, sports vs. food vs. others)
- Sentiment dictionaries:
  - Broad Sentiment (Positive, Negative)
  - Nuanced Emotion (Anger, Joy, Sadness)
  - Not all English words included (many are neutral)
- Word weights / scores
  - Binary: $\{$Positive $(+1)$, Negative (-1)$\}$
  - Numerical: $\{-2, -1, 1, 2\}$
- Non-sentiment dictionaries: Words about sports, food, places...

Applying Dictionaries to Documents

- Vector of word counts: $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{iP})$, $i = (1, \ldots, N)$

Applying Dictionaries to Documents

- Vector of word counts: $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{iP})$, $i = (1, \ldots, N)$
- Weights attached to words $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_P)$

Applying Dictionaries to Documents

- Vector of word counts: $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \dots, X_{iP})$, $i = (1, \dots, N)$
- Weights attached to words $\boldsymbol{\theta} = (\theta_1, \theta_2, \dots, \theta_P)$
  - $\theta_p \in \{0, 1\}$: 1=sports, 0=others

Applying Dictionaries to Documents

- Vector of word counts: $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{iP})$, $i = (1, \ldots, N)$
- Weights attached to words $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_P)$
  - $\theta_p \in \{0, 1\}$: 1=sports, 0=others
  - $\theta_p \in \{-1, 0, 1\}$: 1=positive, 0=neutral, -1=negative

Applying Dictionaries to Documents

- Vector of word counts: $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{iP})$, $i = (1, \ldots, N)$
- Weights attached to words $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_P)$
  - $\theta_p \in \{0, 1\}$: 1=sports, 0=others
  - $\theta_p \in \{-1, 0, 1\}$: 1=positive, 0=neutral, -1=negative
  - $\theta_p \in \{-2, -1, 0, 1, 2\}$: more nuanced sentiment scores

Applying Dictionaries to Documents

- Vector of word counts: $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{iP})$, $i = (1, \ldots, N)$
- Weights attached to words $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_P)$
  - $\theta_p \in \{0, 1\}$: 1=sports, 0=others
  - $\theta_p \in \{-1, 0, 1\}$: 1=positive, 0=neutral, -1=negative
  - $\theta_p \in \{-2, -1, 0, 1, 2\}$: more nuanced sentiment scores
  - $\theta_p \in \Re$

Applying Dictionaries to Documents

- Vector of word counts: $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{iP})$, $i = (1, \ldots, N)$
- Weights attached to words $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_P)$
  - $\theta_p \in \{0, 1\}$: 1=sports, 0=others
  - $\theta_p \in \{-1, 0, 1\}$: 1=positive, 0=neutral, -1=negative
  - $\theta_p \in \{-2, -1, 0, 1, 2\}$: more nuanced sentiment scores
  - $\theta_p \in \Re$

For each document i calculate score for document

Applying Dictionaries to Documents

- Vector of word counts: $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{iP})$, $i = (1, \ldots, N)$
- Weights attached to words $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_P)$
  - $\theta_p \in \{0, 1\}$: 1=sports, 0=others
  - $\theta_p \in \{-1, 0, 1\}$: 1=positive, 0=neutral, -1=negative
  - $\theta_p \in \{-2, -1, 0, 1, 2\}$: more nuanced sentiment scores
  - $\theta_p \in \Re$

For each document i calculate score for document

$$Y_i = \frac{\sum_{p=1}^{P} \theta_p X_{ip}}{\sum_{p=1}^{P} X_{ip}}$$

Applying Dictionaries to Documents

- Vector of word counts: $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{iP})$, $i = (1, \ldots, N)$
- Weights attached to words $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_P)$
  - $\theta_p \in \{0, 1\}$: 1=sports, 0=others
  - $\theta_p \in \{-1, 0, 1\}$: 1=positive, 0=neutral, -1=negative
  - $\theta_p \in \{-2, -1, 0, 1, 2\}$: more nuanced sentiment scores
  - $\theta_p \in \Re$

For each document i calculate score for document

$$Y_i = \frac{\sum_{p=1}^{P} \theta_p X_{ip}}{\sum_{p=1}^{P} X_{ip}}$$

$Y_i \approx$ continuous $\rightsquigarrow$ Classification using some threshold $\alpha \in (0, 1/2...)$

Applying Dictionaries to Documents

- Vector of word counts: $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{iP})$, $i = (1, \ldots, N)$
- Weights attached to words $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_P)$
  - $\theta_p \in \{0, 1\}$: 1=sports, 0=others
  - $\theta_p \in \{-1, 0, 1\}$: 1=positive, 0=neutral, -1=negative
  - $\theta_p \in \{-2, -1, 0, 1, 2\}$: more nuanced sentiment scores
  - $\theta_p \in \Re$

For each document i calculate score for document

$$Y_i = \frac{\sum_{p=1}^{P} \theta_p X_{ip}}{\sum_{p=1}^{P} X_{ip}}$$

$Y_i \approx$ continuous $\rightsquigarrow$ Classification using some threshold $\alpha \in (0, 1/2...)$

$Y_i > \alpha \Rightarrow$ Positive Category

Applying Dictionaries to Documents

- Vector of word counts: $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{iP})$, $i = (1, \ldots, N)$
- Weights attached to words $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_P)$
  - $\theta_p \in \{0, 1\}$: 1=sports, 0=others
  - $\theta_p \in \{-1, 0, 1\}$: 1=positive, 0=neutral, -1=negative
  - $\theta_p \in \{-2, -1, 0, 1, 2\}$: more nuanced sentiment scores
  - $\theta_p \in \Re$

For each document i calculate score for document

$$Y_i = \frac{\sum_{p=1}^{P} \theta_p X_{ip}}{\sum_{p=1}^{P} X_{ip}}$$

$Y_i \approx$ continuous $\rightsquigarrow$ Classification using some threshold $\alpha \in (0, 1/2...)$

$Y_i > \alpha \Rightarrow$ Positive Category

$Y_i < \alpha \Rightarrow$ Negative Category

Applying Dictionaries to Documents

- Vector of word counts: $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{iP})$, $i = (1, \ldots, N)$
- Weights attached to words $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_P)$
  - $\theta_p \in \{0, 1\}$: 1=sports, 0=others
  - $\theta_p \in \{-1, 0, 1\}$: 1=positive, 0=neutral, -1=negative
  - $\theta_p \in \{-2, -1, 0, 1, 2\}$: more nuanced sentiment scores
  - $\theta_p \in \Re$

For each document i calculate score for document

$$Y_i = \frac{\sum_{p=1}^{P} \theta_p X_{ip}}{\sum_{p=1}^{P} X_{ip}}$$

$Y_i \approx$ continuous $\rightsquigarrow$ Classification using some threshold $\alpha \in (0, 1/2...)$

$Y_i > \alpha \Rightarrow$ Positive Category

$Y_i < \alpha \Rightarrow$ Negative Category

$Y_i \approx \alpha$ Ambiguous

Dictionary Methods

Use with Caution!

- Most dictionary methods do not take into account qualifiers (e.g. “no good”).

Dictionary Methods

Use with Caution!

- Most dictionary methods do not take into account qualifiers (e.g. "no good").
- Ignores sarcasm, irony, nuance.

Dictionary Methods

Use with Caution!

- Most dictionary methods do not take into account qualifiers (e.g. "no good").
- Ignores sarcasm, irony, nuance.
- Dictionaries are context dependent.

Dictionary Methods

Use with Caution!

- Most dictionary methods do not take into account qualifiers (e.g. "no good").
- Ignores sarcasm, irony, nuance.
- Dictionaries are context dependent.
  - Ideally, we can create a dictionary specific to the context of our application (e.g. accounting)

Dictionary Methods

Use with Caution!

- Most dictionary methods do not take into account qualifiers (e.g. "no good").
- Ignores sarcasm, irony, nuance.
- Dictionaries are context dependent.
  - Ideally, we can create a dictionary specific to the context of our application (e.g. accounting)
  - Face validity

Dictionary Methods

Use with Caution!

- Most dictionary methods do not take into account qualifiers (e.g. "no good").
- Ignores sarcasm, irony, nuance.
- Dictionaries are context dependent.
    - Ideally, we can create a dictionary specific to the context of our application (e.g. accounting)
    - Face validity
- `R Code, Section 2!`