MACHINE LEARNING
OVERVIEW
What if your model has a high error?

- Try getting more training examples
- Try smaller sets of features
- Try getting additional features
- Try creating features from existing features (kernels)
- Try decrease regularization
- Try increase regularization
Bias and Variance

Error

Training set M
Bias and Variance

Training set M

Error

Training Set (m)

Test-Set (ts)

Training Set

Validation Set
Bias and Variance

Training Set M

Training Error

Training Set (m)

Training Set

Validation Set
Bias and Variance

Error

Training Error

Training Set M

Training Set (m)

Training Set

Validation Set
Bias and Variance

Error

Training set M

Training Error

Training Set (m)

Training Set

Validation Set
Bias and Variance

Training set M

Training Error

Error

Training Set (m)

Training Set

Validation Set
Bias and Variance

Error

Training set M

Training Error

Training Set (m)

Training Set

Validation Set
Bias and Variance

Error

Training Error

Training Set M

Training Set (m)

Training Set

Validation Set
Bias and Variance

Training set M

Error

Training Error

Training Set (m)

Training Set

Validation Set
Bias and Variance

Error

Training Error
Test Error

Training Set M

Training Set (m)

Clicker:
Test error
a) decreases with M
b) increases with M
c) stays constant

Training Set
Validation Set

12
Bias and Variance

Error

Training Error

Test Error

Training set M

Training Set (m)

Training Set

Validation Set
High Bias

![Graph showing training and test error with respect to training set size.](image)
Clicker: If you have high-bias, does more data help?

a) No
b) Yes
Clicker: If you have high-variance, does more data help?

a) No
b) Yes
1. Get more training examples
2. Try smaller sets of features
3. Try getting additional features
4. Try adding polynomial features (kernels)
5. Try increase regularization
6. Try decrease regularization

Helps with
A. High Variance
B. High Bias
C. Both
D. None
Clicker

1. Get more training examples
2. Try smaller sets of features
3. Try getting additional features
4. Try adding polynomial features (kernels)
5. Try increase regularization
6. Try decrease regularization

Helps with
A. High Variance
B. High Bias
C. Both
D. None
Clicker

1. Get more training examples
2. Try smaller sets of features
3. **Try getting additional features**
4. Try adding polynomial features (kernels)
5. Try increase regularization
6. Try decrease regularization

Helps with
A. High Variance
B. High Bias
C. Both
D. None
1. Get more training examples
2. Try smaller sets of features
3. Try getting additional features
4. Try adding polynomial features (kernels)
5. Try increase regularization
6. Try decrease regularization

Helps with
A. High Variance
B. High Bias
C. Both
D. None
Clicker

1. Get more training examples
2. Try smaller sets of features
3. Try getting additional features
4. Try adding polynomial features (kernels)
5. Try increase regularization
6. Try decrease regularization

Helps with
A. High Variance
B. High Bias
C. Both
D. None
Clicker

1. Get more training examples
2. Try smaller sets of features
3. Try getting additional features
4. Try adding polynomial features (kernels)
5. Try increase regularization
6. Try decrease regularization

Helps with
A. High Variance
B. High Bias
C. Both
D. None
Cross-validation

k-fold: split the data into k groups, train on every group except for one, which you test on.

Repeat for all groups
Parameter Tuning

Grid Search
How to speed-up tuning?

Can we use sampling?

**Algorithm 1:**
- Training
- Validation

**Algorithm 2:**
- Training
- Validation
How to speed-up tuning?

Can we use sampling?

Algorithm 1:
- Training
- Validation

Algorithm 2:
- Training
- Validation
How to speed-up tuning?

Can we use sampling?

**Algorithm 1:**
- Training
- Validation

**Algorithm 2:**
- Training
- Validation
How to speed-up tuning?

Can we use sampling?

**Algorithm 1:**
- Training
- Validation

**Algorithm 2:**
- Training
- Validation
How to speed-up tuning?

Can we use sampling?

**Algorithm 1:**
- Training
- Validation

**Algorithm 2:**
- Training
- Validation
How to speed-up tuning?

Can we use sampling?

Algorithm 1:
- Training
- Validation

Algorithm 2:
- Training
- Validation

Can we prune now?
How to speed-up tuning?

Can we use sampling?

**Algorithm 1:**
- Training
- Validation

**Algorithm 2:**
- Training
- Validation

Algorithm 1 training error > Algorithm 2 validation error
Northstar’s (now Einblick) AutoML

Built for interactive results, unlike all other Auto-ML tools, which can take hours to produce results.

What modeling options do I have?
Rule-based Search Space Expansion

What should I try first?
Use past experience to optimize for expected quality per time-unit

How can I get some quick results?
Adaptive sampling-based pruning and transfer learning

Improve based on results

Zeyuan Shang, Emanuel Zgraggen, Benedetto Buratti, Ferdinand Kossmann, Philipp Eichmann, Yeounoh Chung, Carsten Binnig, Eli Upfal, Tim Kraska: Democratizing Data Science through Interactive Curation of ML Pipelines. SIGMOD Conference 2019: 1171-1188

Free to use for MIT students: https://einblick.ai/
AutoML Tools

- Auto-WEKA
- AutoKeras
- H2O AutoML
- einblick
Feature Engineering

(a) Training
- **label**
- **input**
- **feature extractor**
- **features**
- **machine learning algorithm**

(b) Prediction
- **input**
- **feature extractor**
- **features**
- **classifier model**
- **label**
How would you predict the unemployment rate before the official numbers come out?

Feature engineering

- Dropping features
  - Remove duplicates
  - Highly correlated values (Zip code, Lon/Lat)

- Feature creation
  - Feature crosses: Cost per square feet
  - Creating special features (“I lost my job”)
  - Row statistics
    - Number of 0, nulls, negative value, mean, max, min,…
  - Projection to circle
    - Turn a single feature (like day_of_week) into two coordinates on a circle
    - Ensures that distance between Monday and Sunday etc is the same
  - Spatial
    - GPS encoding
    - Categorized locations (e.g., close to city, rural, nearby hospital, etc.)
  - Use embeddings from other models (more on that later)
  - Discretization (date → weekend/weekday)
  - ...

37
Transformations

• Rounding
  – Lossy
  – Precision can just be noise -> might improve training
  – Log transform before rounding often useful
• Binning
  – Removes information
  – Can work gracefully with variables outside of ranges seen in the train set
• Scaling
  – Standard (Z) Scaling
  – MinMax Scaling
  – Root Scaling
  – Log Scaling
• Outlier removal
• Imputation (mean, median, …)
• Interaction encoding: Specifically encodes the interaction between two numerical variables
  – Subtraction, Addition, Multiplication….
  – **Polynomial encoding**
    • Linear algorithms can not solve XOR problem
    • A polynomial kernel can solve XOR
Encodings

- **One-hot**
- **NaN, null, etc** → create explicit encoding
- **Hash-encoding** (careful might introduce collisions)
- **Count encoding**: replace categorical value with their count
  - Useful for both linear and non-linear algorithms
  - Sensitive to outliers
  - Might create collisions
- **Rank encoding**: Rank categorical variables by count in train set
  - Useful for both linear and non-linear algorithms
  - Not sensitive to outliers
  - Won’t give same encoding to different variables
- **Target encoding**: Encode categorical variables by their ratio of target (binary classification) in train set
  - Be careful to avoid overfit
  - Add smoothing to avoid setting variable encoding to 0
  - Add random noise?
  - Can work extremely well when done right
- **Consolidation/expansion encoding**: map different categorical variables to the same
  - Spelling errors, slightly different job descriptions, abbreviations
Example: Customer Conversion

Questions:
- What sequence of emails, phone calls, showing ads in the platform, etc. leads to the highest conversion rate (e.g., open an account)?
- How many emails/phone calls are too much?
- What should the timing be?
- How can we annotate the data with custom information we have available (notes from 1on1 conversations, portal logins, current accounts, etc.)?
- How do we quickly adjust to changing conditions (e.g., covid happens, increasing interest rates, inflation,...)?

→ This is not a standardized process. Rather it requires to quickly experiment with new models.

The data

<table>
<thead>
<tr>
<th>Customer</th>
<th>Date</th>
<th>Event</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tim</td>
<td>2022/01/03</td>
<td>E11</td>
<td>Promotion email to savings bank account</td>
</tr>
<tr>
<td>Paul</td>
<td>2022/01/03</td>
<td>E11</td>
<td>Promotion email to savings bank account</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/03</td>
<td>P10</td>
<td>Phone call.</td>
</tr>
<tr>
<td>Mark</td>
<td>2022/01/05</td>
<td>E11</td>
<td>Promotion email to savings bank account</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/05</td>
<td>E11</td>
<td>Promotion email to start saving</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/06</td>
<td>D1</td>
<td>Display ad regarding savings account</td>
</tr>
<tr>
<td>Paul</td>
<td>2022/01/07</td>
<td>D1</td>
<td>Display ad regarding savings account</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/08</td>
<td>B1</td>
<td>User opens saving account</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Customer</th>
<th>Event</th>
<th>Log</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tim</td>
<td>1on1 Meeting</td>
<td>Had a good meeting. Expressed some interest in savings account but was worried about inflation</td>
</tr>
<tr>
<td>Paul</td>
<td>1on1 Meeting</td>
<td>Good conversation, but expressed no interest in savings accounts</td>
</tr>
<tr>
<td>Mark</td>
<td>Phone call</td>
<td>Talked about insurance and mortgages</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Customer</th>
<th>Account</th>
<th>Balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tim</td>
<td>Checking</td>
<td>10k</td>
</tr>
<tr>
<td>Paul</td>
<td>Checking</td>
<td>20k</td>
</tr>
<tr>
<td>Tim</td>
<td>Trading</td>
<td>40k</td>
</tr>
<tr>
<td>Mark</td>
<td>Checking</td>
<td>15k</td>
</tr>
</tbody>
</table>

...
Example: Customer Conversion

How would you build a model over this data to predict if a user opens a new savings account?

<table>
<thead>
<tr>
<th>Customer</th>
<th>Date</th>
<th>Event</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tim</td>
<td>2022/01/03</td>
<td>E11</td>
<td>Promotion email to savings bank account</td>
</tr>
<tr>
<td>Paul</td>
<td>2022/01/03</td>
<td>E11</td>
<td>Promotion email to savings bank account</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/04</td>
<td>P10</td>
<td>Phone call.</td>
</tr>
<tr>
<td>Mark</td>
<td>2022/01/05</td>
<td>E11</td>
<td>Promotion email to savings bank account</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/05</td>
<td>E11</td>
<td>Promotion email to start saving</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/06</td>
<td>D1</td>
<td>Display ad regarding savings account</td>
</tr>
<tr>
<td>Paul</td>
<td>2022/01/07</td>
<td>D1</td>
<td>Display ad regarding savings account</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/08</td>
<td>B1</td>
<td>User opens saving account</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Customer</th>
<th>Event</th>
<th>Log</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tim</td>
<td>1on1 Meeting</td>
<td>Had a good meeting. Expressed some interest in savings account but was worried about inflation</td>
</tr>
<tr>
<td>Paul</td>
<td>1on1 Meeting</td>
<td>Good conversation, but expressed no interest in savings accounts</td>
</tr>
<tr>
<td>Mark</td>
<td>Phone call</td>
<td>Talked about insurance and mortgages</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Customer</th>
<th>Account</th>
<th>Balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tim</td>
<td>Checking</td>
<td>10k</td>
</tr>
<tr>
<td>Paul</td>
<td>Checking</td>
<td>20k</td>
</tr>
<tr>
<td>Tim</td>
<td>Trading</td>
<td>40k</td>
</tr>
<tr>
<td>Mark</td>
<td>Checking</td>
<td>15k</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
## Example: Customer Conversion

<table>
<thead>
<tr>
<th>Customer</th>
<th>Date</th>
<th>Event</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tim</td>
<td>2022/01/03</td>
<td>E11</td>
<td>Promotion email to savings bank account</td>
</tr>
<tr>
<td>Paul</td>
<td>2022/01/03</td>
<td>E11</td>
<td>Promotion email to savings bank account</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/04</td>
<td>P10</td>
<td>Phone call</td>
</tr>
<tr>
<td>Mark</td>
<td>2022/01/05</td>
<td>E11</td>
<td>Promotion email to savings bank account</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/05</td>
<td>L1</td>
<td>User logged into platform</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/06</td>
<td>D1</td>
<td>Display ad regarding savings account</td>
</tr>
<tr>
<td>Paul</td>
<td>2022/01/07</td>
<td>L1</td>
<td>Display ad regarding savings account</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/08</td>
<td>B1</td>
<td>User opens saving account</td>
</tr>
</tbody>
</table>

### Customer Event Log

<table>
<thead>
<tr>
<th>Customer</th>
<th>Date</th>
<th>Event</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tim</td>
<td>2022/01/03</td>
<td>E11</td>
<td>Promotion email to savings bank account</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/04</td>
<td>L1</td>
<td>User logged into platform</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/05</td>
<td>L1</td>
<td>Display ad regarding savings account</td>
</tr>
<tr>
<td>Paul</td>
<td>2022/01/06</td>
<td>E11</td>
<td>Promotion email to savings bank account</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/07</td>
<td>E11</td>
<td>Promotion email to savings bank account</td>
</tr>
<tr>
<td>Tim</td>
<td>2022/01/08</td>
<td>B1</td>
<td>User opens saving account</td>
</tr>
</tbody>
</table>

### Customer Account Balance

<table>
<thead>
<tr>
<th>Customer</th>
<th>Account</th>
<th>Balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tim</td>
<td>CK1</td>
<td>10k</td>
</tr>
<tr>
<td>Paul</td>
<td>CK2</td>
<td>20k</td>
</tr>
<tr>
<td>Tim</td>
<td>TD</td>
<td>40k</td>
</tr>
<tr>
<td>Mark</td>
<td>CK1</td>
<td>15k</td>
</tr>
</tbody>
</table>

### Customer Next Action

<table>
<thead>
<tr>
<th>Customer</th>
<th>Account</th>
<th>Balance</th>
<th>Next Action</th>
<th>Result open account</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tim</td>
<td>CK1</td>
<td>10k</td>
<td>$100 Promotion Email</td>
<td>80%</td>
</tr>
<tr>
<td>Paul</td>
<td>CK2</td>
<td>20k</td>
<td>Email promotion</td>
<td>2%</td>
</tr>
<tr>
<td>Mark</td>
<td>CK1</td>
<td>15k</td>
<td>Phone call</td>
<td>3%</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Other time-related feature engineering tricks

- Trendlines
  - Instead of encoding: total spend, encode things like: spend in last week, spend last month, spend in last year
  - Gives a trend to the algorithm

- Closeness to major events
  - Hardcode categorical features like date_3_days_before_holidays
  - Try national holidays, major sport events, weekends, end of quarter, etc. → All can have impact on spending behavior

- Projection to circle
Word embeddings

- **Idea:** learn an embedding from words into vectors
  
  \( \text{Cat} = \{0.002, 0.244, 0.546, \ldots, 0.345\} \)

- Need to have a function \( W(\text{word}) \) that returns a vector encoding that word.

- **Applications:** ???
Word embeddings: properties

Relationships between words correspond to difference between vectors.

\[
W(\text{“woman”}) - W(\text{“man”}) \approx W(\text{“aunt”}) - W(\text{“uncle”})
\]

\[
W(\text{“woman”}) - W(\text{“man”}) \approx W(\text{“queen”}) - W(\text{“king”})
\]

Word embeddings: questions

• How big should the embedding space be?
  – Trade-offs like any other machine learning problem – greater capacity versus efficiency and overfitting.

• How do we find W?
  – Often as part of a prediction or classification task involving neighboring words.
Learning word embeddings

First attempt:
- Input data is sets of 5 words from a meaningful sentence. E.g., “one of the best places”. Modify half of them by replacing middle word with a random word. “one of function best places”
- W is a map (depending on parameters, Q) from words to 50 dim’l vectors. E.g., a look-up table or an RNN.
- Feed 5 embeddings into a module R to determine ‘valid’ or ‘invalid’
- Optimize over Q to predict better


Word-Embeddings: word2vec

- Predict words using context
- Two versions: CBOW (continuous bag of words) and Skip-gram
Continuous Bag of words (CBOS)

- Bag of words (BOW)
  - Gets rid of word order. Used in discrete case using counts of words that appear.

- CBOW
  - Takes vector embeddings of $n$ words before target and $n$ words after and adds them (as vectors).
  - Also removes word order, but the vector sum is meaningful enough to deduce missing word.
Jay was hit by a ______ bus in...
Continuous Bag of Word

- E.g. “The cat sat on floor”
  - Window size = 2
COBW

Input

Matrix $W$

Hidden

Matrix $W'$

Output softmax

$N = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix}$

$V = \begin{bmatrix} v_1 \\ \vdots \\ v_j \\ \vdots \\ v_N \end{bmatrix}$

$X$

$V = \begin{bmatrix} \text{avg} \\ \vdots \\ h_3 \\ \vdots \\ h_N \end{bmatrix}$

$N$-dimension vector (Average of vectors of all input words)
We must learn $W$ and $W'$

$W_{V \times N}$

N will be the size of word vector
\[ W_{V \times N}^T \times x_{on} = v_{on} \]

\[ v_{cat} = W_{V \times N}^T \times x_{cat} \]

\[ \hat{v} = \frac{v_{cat} + v_{on}}{2} \]
\[
\sigma(\tilde{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}
\]

Softmax turns the vector into probabilities

\[
\hat{y} = \text{softmax}(z)
\]

\[
\begin{align*}
\mathbf{W}_{V \times N} \times \mathbf{\hat{v}} &= \mathbf{z} \\
\mathbf{W}'_{V \times N} \times \mathbf{\hat{v}} &= \mathbf{z} \\
\hat{y} &= \text{softmax}(\mathbf{z})
\end{align*}
\]

N will be the size of word vector
We would prefer $\hat{y}$ close to $\hat{y}_{sat}$

$W_{V \times N} \times \hat{v} = z$

$\hat{y} = \text{softmax}(z)$

$V$ will be the size of word vector

$N$ will be the size of word vector

www.cs.ucr.edu/~vagelis/classes/CS242/slides/word2vec.pptx
We can consider either $W$ or $W'$ as the word's representation. Or even take the average.

www.cs.ucr.edu/~vagelis/classes/CS242/slides/word2vec.pptx
Some interesting results

Word Analogies

Test for linear relationships, examined by Mikolov et al. (2014)

\[ a:b :: c:? \]

man:woman :: king:? 

\[ + \quad \text{king} \quad [0.30 \ 0.70] \]
\[ - \quad \text{man} \quad [0.20 \ 0.20] \]
\[ + \quad \text{woman} \quad [0.60 \ 0.30] \]
\[ \quad \text{queen} \quad [0.70 \ 0.80] \]
Word analogies
Skip gram

- Map from center word to probability on surrounding words. One input/output unit below.
- Start with a single word embedding and try to predict the surrounding words.
- Much less well-defined problem, but works better in practice (scales better).

http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/
Skip Gram (window 2)

Sam  likes  Celine  Dion  and  biking

Sam  likes  Celine  Dion  and  biking

Sam  likes  Celine  Dion  and  biking

Sam  likes  Celine  Dion  and  biking

Sam  likes  Celine  Dion  and  biking

Sam  likes  Celine  Dion  and  biking

Sam  likes  Celine  Dion  and  biking

Sam  likes  Celine  Dion  and  biking

Sam  likes  Celine  Dion  and  biking
Skip gram

Input

\[ \begin{align*}
&x_1 \\
&x_2 \\
&\vdots \\
&x_i \\
&x_V
\end{align*} \]

Vector of word i

Embedding matrix

Matrix \( W \)

Hidden

\[ h_1, h_2, h_3, \ldots, h_N \]

\[ N \]

\[ V = X \]

\[ h_N \]

Output softmax

\[ \begin{align*}
&0 \\
&0 \\
&0 \\
&y_1 \\
&y_2 \\
&\vdots \\
&y_j \\
&\vdots \\
&y_V
\end{align*} \]

N-dimensional vector

Context matrix

Matrix \( W' \)

Vector of word j.
Skip gram example

- Vocabulary of 10,000 words.
- Embedding vectors with 300 features.
- So the hidden layer is going to be represented by a weight matrix of size 300 with 10,000 rows.

\[
\begin{bmatrix}
0 & 0 & 0 & 1 & 0
\end{bmatrix}
\times
\begin{bmatrix}
17 & 24 & 1 & 23 & 5 & 7 & 4 & 6 & 13 & 10 & 12 & 19 & 11 & 18 & 25
\end{bmatrix}
= \begin{bmatrix}
10 & 12 & 19
\end{bmatrix}
\]
Word2vec shortcomings

• **Problem:** 10,000 words and 300 dim embedding gives a large parameter space to learn. And 10K words is minimal for real applications.

• Slow to train, and need lots of data, particularly to learn uncommon words.

Any ideas how to make the approach more scalable?
Word2vec improvements: word pairs and phrases

- **Idea:** Treat common word pairs or phrases as single “words.”
  - E.g., Boston Globe (newspaper) is different from Boston and Globe separately. Embed Boston Globe as a single word/phrase.

- **Method:** make phrases out of words which occur together often relative to the number of individual occurrences. Prefer phrases made of infrequent words in order to avoid making phrases out of common words like “and the” or “this is”.

- **Pros/cons:** Increases vocabulary size but decreases training expense.
Word2vec improvements: subsample frequent words

- **Idea:** Subsample frequent words to decrease the number of training examples.
  - The probability that we cut the word is related to the word’s frequency. More common words are cut more.
  - Uncommon words (anything < 0.26% of total words) are kept
  - E.g., remove some occurrences of “the.”

- **Method:** For each word, cut the word with probability related to the word’s frequency.

- **Benefits:** If we have a window size of 10, and we remove a specific instance of “the” from our text:
  - As we train on the remaining words, “the” will not appear in any of their context windows.
Word2vec improvements: selective updates

- **Idea:** Use “Negative Sampling”, which causes each training sample to update only a small percentage of the model’s weights.

- **Observation:** A “correct output” of the network is a one-hot vector. That is, one neuron should output a 1, and all of the other thousands of output neurons to output a 0.

- **Method:** With negative sampling, randomly select just a small number of “negative” words (let’s say 5) to update the weights for. (In this context, a “negative” word is one for which we want the network to output a 0 for). We will also still update the weights for our “positive” word.
Applications

- Clustering
- Next word prediction
- Translation

**Other: Images ....**
- Can apply to get a joint embedding of words and images or other multi-modal data sets.
- New classes map near similar existing classes: e.g., if 'cat' is unknown, cat images map near dog.
WHAT IS THE PROBLEM WITH WORD EMBEDDINGS?

The mountain has a lot of **grass**

You should never smoke **grass**

same word embedding [0.99, 0.8, ...]

Solution: Train contextual representations on text corpus
LITTLE HISTORY

Semi-Supervised Sequence Learning, Google, 2015

ELMo: Deep Contextual Word Embeddings, AI2 & University of Washington, 2017
GPT

Improving Language Understanding by Generative Pre-Training, OpenAI, 2018 – Based on transformers/attention from "Attention is All You Need" Vaswani et al
BERT

Use the output of the masked word’s position to predict the masked word

Possible classes:
- All English words
- Aardvark
- Improvisation
- Zzyzyva

Randomly mask 15% of tokens

Input

[CLS] Let's stick to [MASK] in this skit
BERT VS OPENAI GPT VS ELMo

See also http://jalammar.github.io/illustrated-gpt2/
TASKS

(a) Sentence Pair Classification Tasks: MNLI, QQP, QNL, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA

(c) Question Answering Tasks: SQuAD v1.1

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

http://www.msmarco.org/leaders.aspx
### KeyPhrase Extraction (10/18/2019) ranked by F1 @3 on Eval

<table>
<thead>
<tr>
<th>Rank</th>
<th>Model</th>
<th>Submission Date</th>
<th>Precision @1, @3, @5</th>
<th>Recall @1, @3, @5</th>
<th>F1 @1, @3, @5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>BERT (Base) Sequence Tagging</strong> Si Sun (Tsinghua University), Cheryan Xiong (MSR AI), Zhiyuan Lu (Tsinghua University) [Code]</td>
<td>November 5th, 2019</td>
<td>0.464, 0.312, 0.227</td>
<td>0.255, 0.469, 0.563</td>
<td>0.321, 0.361, 0.314</td>
</tr>
<tr>
<td>2</td>
<td><strong>Baseline finetuned on Bing Queries</strong> MSMARCO Team</td>
<td>October 19th, 2019</td>
<td>0.397, 0.248, 0.149</td>
<td>0.215, 0.391, 0.391</td>
<td>0.267, 0.292, 0.209</td>
</tr>
<tr>
<td>3</td>
<td><strong>Baseline MSMARCO Team</strong></td>
<td>October 19th, 2019</td>
<td>0.365, 0.237, 0.142</td>
<td>0.196, 0.367, 0.367</td>
<td>0.244, 0.277, 0.198</td>
</tr>
</tbody>
</table>

### Passage Retrieval (10/26/2018-Present) ranked by MRR on Eval

<table>
<thead>
<tr>
<th>Rank</th>
<th>Model</th>
<th>Ranking Style</th>
<th>Submission Date</th>
<th>MRR@10 On Eval</th>
<th>MRR@10 On Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>Enriched BERT base + AOA index + CAS</strong> Ming Yan of Alibaba Damo NLP</td>
<td>Full Ranking</td>
<td>August 20th, 2019</td>
<td>0.393</td>
<td>0.408</td>
</tr>
<tr>
<td>2</td>
<td><strong>W-Index retrieval + BERT-F re-rank</strong> Zhuyun Dai of Carnegie Mellon University</td>
<td>Full Ranking</td>
<td>September 12th, 2019</td>
<td>0.388</td>
<td>0.394</td>
</tr>
<tr>
<td>3</td>
<td><strong>Enriched BERT base + AOA Index V1</strong> Ming Yan of Alibaba Damo NLP</td>
<td>Full Ranking</td>
<td>May 13th, 2019</td>
<td>0.383</td>
<td>0.397</td>
</tr>
</tbody>
</table>

### Q&A Task (03/01/2018-Present)

<table>
<thead>
<tr>
<th>Rank</th>
<th>Model</th>
<th>Submission Date</th>
<th>Rouge-L</th>
<th>Bleu-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>Multi-doc Enriched BERT</strong> Ming Yan of Alibaba Damo NLP</td>
<td>June 20th, 2019</td>
<td>0.540</td>
<td>0.585</td>
</tr>
<tr>
<td>2</td>
<td><strong>Human Performance</strong></td>
<td>April 23th, 2018</td>
<td>0.539</td>
<td>0.485</td>
</tr>
<tr>
<td>3</td>
<td><strong>BERT Encoded T-Net</strong> Y. Zhang, C. Wang, X.L. Chan</td>
<td>August 5th, 2019</td>
<td>0.526</td>
<td>0.539</td>
</tr>
</tbody>
</table>

### Q&A + Natural Language Generation Task (03/01/2018-Present)

<table>
<thead>
<tr>
<th>Rank</th>
<th>Model</th>
<th>Submission Date</th>
<th>Rouge-L</th>
<th>Bleu-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>Human Performance</strong></td>
<td>April 23th, 2018</td>
<td>0.632</td>
<td>0.530</td>
</tr>
<tr>
<td>2</td>
<td><strong>Masque NLGEN Style</strong> NTT Media Intelligence Laboratories [Nishida et al., '19]</td>
<td>January 3rd, 2019</td>
<td>0.496</td>
<td>0.501</td>
</tr>
<tr>
<td>3</td>
<td><strong>BERT+ Multi-Pointer-Generator</strong> Tongjun Li of the ColorfulClouds Tech and BUPT</td>
<td>June 11th, 2019</td>
<td>0.495</td>
<td>0.476</td>
</tr>
</tbody>
</table>
Google is now using BERT

Before

Parking on a Hill. Uphill: When headed uphill at a curb, turn the front wheels away from the curb and let your vehicle roll backwards slowly until the rear part of the front wheel rests against the curb using it as a block. Downhill: When you stop your car headed downhill, turn your front wheels toward the curb.

Parking on a Hill - DriversEd.com

After

For either uphill or downhill parking, if there is no curb, turn the wheels toward the side of the road so the car will roll away from the center of the road if the brakes fail. When you park on a sloping driveway, turn the wheels so that the car will not roll into the street if the brakes fail.
GOOGLE IS NOW USING BERT

Can you get medicine for someone pharmacy

**BEFORE**

9:00

google.com

MedlinePlus (.gov) > ency > article

Getting a prescription filled: MedlinePlus Medical Encyclopedia

Aug 26, 2017 · Your health care provider may give you a prescription in ... Writing a paper prescription that you take to a local pharmacy ... Some people and insurance companies choose to use ...

**AFTER**

9:00

google.com

HHS.gov > hipaa > for-professionals

Can a patient have a friend or family member pick up a prescription ...

Dec 19, 2002 · A pharmacist may use professional judgment and experience with common practice to ... the patient's best interest in allowing a person, other that the patient, to pick up a prescription.
GOOGLE IS NOW USING BERT
ADABOOST - CORE IDEA

Take a set of weak classifiers (normally they should do better than guessing)

Classify Result

Combine to form the Final strong classifier

Weight the result of each classify with

\[ H(x) = \text{sign} \sum_{i=1}^{n} q_i h_i(x) \]
ADABOOST - CORE IDEA

Take a set of weak classifiers (normally they should do better than guessing)

Combine to form the Final strong classifier

Weight the result of each classify with

\[ H(x) = \text{sign} \sum_{i=1}^{n} q_i h_i(x) \]
**ADABOOST - CORE IDEA**

**Take a set of weak classifiers** (normally they should do better than guessing)

\[ H(x) = \text{sign} \sum_{i=1}^{n} q_i h_i(x) \]

**Classification Result**

Combine to form the Final strong classifier

XGBoost follows the same idea

Weight the result of each classify with
IDEA BEHIND GRADIENT BOOSTING

<table>
<thead>
<tr>
<th>Desc</th>
<th>Protein</th>
<th>Carb</th>
<th>Sugar</th>
<th>Iron</th>
<th>Kcal</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHEESE, GRUYERE</td>
<td>30</td>
<td>0</td>
<td>0.36</td>
<td>0.17</td>
<td>413</td>
</tr>
<tr>
<td>ICE CRM SNDWCH</td>
<td>4</td>
<td>37</td>
<td>18.57</td>
<td>0.26</td>
<td>237</td>
</tr>
<tr>
<td>PORK, LOIN</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>0.84</td>
<td>143</td>
</tr>
<tr>
<td>CARROTS, RAW</td>
<td>1</td>
<td>10</td>
<td>4.74</td>
<td>0.3</td>
<td>41</td>
</tr>
<tr>
<td>APPLES, RAW, WITH SKIN</td>
<td>0</td>
<td>14</td>
<td>10.39</td>
<td>0.12</td>
<td>52</td>
</tr>
<tr>
<td>BEEF, STRIP STEAKS</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>1.85</td>
<td>117</td>
</tr>
</tbody>
</table>

Step 1: Build baseline model
IDEA BEHIND GRADIENT BOOSTING

<table>
<thead>
<tr>
<th>Desc</th>
<th>Protein</th>
<th>Carb</th>
<th>Sugar</th>
<th>Iron</th>
<th>Kcal</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHEESE, GRUYERE</td>
<td>30</td>
<td>0</td>
<td>0.36</td>
<td>0.17</td>
<td>413</td>
</tr>
<tr>
<td>ICE CRM SNDWCH</td>
<td>4</td>
<td>37</td>
<td>18.57</td>
<td>0.26</td>
<td>237</td>
</tr>
<tr>
<td>PORK, LOIN</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>0.84</td>
<td>143</td>
</tr>
<tr>
<td>CARROTS, RAW</td>
<td>1</td>
<td>10</td>
<td>4.74</td>
<td>0.3</td>
<td>41</td>
</tr>
<tr>
<td>APPLES, RAW, WITH SKIN</td>
<td>0</td>
<td>14</td>
<td>10.39</td>
<td>0.12</td>
<td>52</td>
</tr>
<tr>
<td>BEEF, STRIP STEAKS</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>1.85</td>
<td>117</td>
</tr>
</tbody>
</table>

Step 1: Build baseline model
1a) Naïve baseline: average \((413 + 237 + \ldots + 117)/6 = 167\)
### IDEA BEHIND GRADIENT BOOSTING

<table>
<thead>
<tr>
<th>Desc</th>
<th>Protein</th>
<th>Carb</th>
<th>Sugar</th>
<th>Iron</th>
<th>Kcal</th>
<th>Residual</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHEESE,GRUYERE</td>
<td>30</td>
<td>0</td>
<td>0.36</td>
<td>0.17</td>
<td>413</td>
<td>246</td>
</tr>
<tr>
<td>ICE CRM SNDWCH</td>
<td>4</td>
<td>37</td>
<td>18.57</td>
<td>0.26</td>
<td>237</td>
<td>70</td>
</tr>
<tr>
<td>PORK,LOIN</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>0.84</td>
<td>143</td>
<td>-24</td>
</tr>
<tr>
<td>CARROTS,RAW</td>
<td>1</td>
<td>10</td>
<td>4.74</td>
<td>0.3</td>
<td>41</td>
<td>-126</td>
</tr>
<tr>
<td>APPLES,RAW,WITH SKIN</td>
<td>0</td>
<td>14</td>
<td>10.39</td>
<td>0.12</td>
<td>52</td>
<td>-115</td>
</tr>
<tr>
<td>BEEF,STRIP STEAKS</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>1.85</td>
<td>117</td>
<td>-50</td>
</tr>
</tbody>
</table>

**Step 1: Build baseline model**

1a) Naïve baseline: average \((413+237+\ldots+117)/6 = 167\)

1b) Calculate **residuals**: actual value – predicted value

For example, for Brie: \(413 - 167 = 246\)
### IDEA BEHIND GRADIENT BOOSTING

<table>
<thead>
<tr>
<th>Desc</th>
<th>Protein</th>
<th>Carb</th>
<th>Sugar</th>
<th>Iron</th>
<th>Kcal</th>
<th>Residual</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHEESE, GRUYERE</td>
<td>30</td>
<td>0</td>
<td>0.36</td>
<td>0.17</td>
<td>413</td>
<td>246</td>
</tr>
<tr>
<td>ICE CRM SNDWCH</td>
<td>4</td>
<td>37</td>
<td>18.57</td>
<td>0.26</td>
<td>237</td>
<td>70</td>
</tr>
<tr>
<td>PORK, LOIN</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>0.84</td>
<td>143</td>
<td>-24</td>
</tr>
<tr>
<td>CARROTS, RAW</td>
<td>1</td>
<td>10</td>
<td>4.74</td>
<td>0.3</td>
<td>41</td>
<td>-126</td>
</tr>
<tr>
<td>APPLES, RAW, WITH SKIN</td>
<td>0</td>
<td>14</td>
<td>10.39</td>
<td>0.12</td>
<td>52</td>
<td>-115</td>
</tr>
<tr>
<td>BEEF, STRIP STEAKS</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>1.85</td>
<td>117</td>
<td>-50</td>
</tr>
</tbody>
</table>

**Step 1:** Build baseline model

**Step 2:** Build tree over residuals (not labels)
### IDEA BEHIND GRADIENT BOOSTING

<table>
<thead>
<tr>
<th>Desc</th>
<th>Protein</th>
<th>Carb</th>
<th>Sugar</th>
<th>Iron</th>
<th>Kcal</th>
<th>Residual</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHEESE, GRUYERE</td>
<td>30</td>
<td>0</td>
<td>0.36</td>
<td>0.17</td>
<td>413</td>
<td>246</td>
</tr>
<tr>
<td>ICE CRM SNDWCH</td>
<td>4</td>
<td>37</td>
<td>18.57</td>
<td>0.26</td>
<td>237</td>
<td>70</td>
</tr>
<tr>
<td>PORK, LOIN</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>0.84</td>
<td>143</td>
<td>-24</td>
</tr>
<tr>
<td>CARROTS, RAW</td>
<td>1</td>
<td>10</td>
<td>4.74</td>
<td>0.3</td>
<td>41</td>
<td>-126</td>
</tr>
<tr>
<td>APPLES, RAW, WITH SKIN</td>
<td>0</td>
<td>14</td>
<td>10.39</td>
<td>0.12</td>
<td>52</td>
<td>-115</td>
</tr>
<tr>
<td>BEEF, STRIP STEAKS</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>1.85</td>
<td>117</td>
<td>-50</td>
</tr>
</tbody>
</table>

**Step 1:** Build baseline model

**Step 2:** Build tree over residuals (not labels)

Compute average for residuals in the same leaf

- Protein < 20
  - Sugar < 18: -120.5
  - Iron < 0.5: 70
- Iron < 0.5: 246
- Protein < 20: -37

IDEA BEHIND GRADIENT BOOSTING

<table>
<thead>
<tr>
<th>Desc</th>
<th>Protein</th>
<th>Carb</th>
<th>Sugar</th>
<th>Iron</th>
<th>Kcal</th>
<th>Residual</th>
<th>Predictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHEESE, GRUYERE</td>
<td>30</td>
<td>0</td>
<td>0.36</td>
<td>0.17</td>
<td>413</td>
<td>246</td>
<td>413</td>
</tr>
<tr>
<td>ICE CRM SNDWCH</td>
<td>4</td>
<td>37</td>
<td>18.57</td>
<td>0.26</td>
<td>237</td>
<td>70</td>
<td>237</td>
</tr>
<tr>
<td>PORK, LOIN</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>0.84</td>
<td>143</td>
<td>-24</td>
<td>130</td>
</tr>
<tr>
<td>CARROTS, RAW</td>
<td>1</td>
<td>10</td>
<td>4.74</td>
<td>0.3</td>
<td>41</td>
<td>-126</td>
<td>47</td>
</tr>
<tr>
<td>APPLES, RAW, WITH SKIN</td>
<td>0</td>
<td>14</td>
<td>10.39</td>
<td>0.12</td>
<td>52</td>
<td>-115</td>
<td>47</td>
</tr>
<tr>
<td>BEEF, STRIP STEAKS</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>1.85</td>
<td>117</td>
<td>-50</td>
<td>130</td>
</tr>
</tbody>
</table>

Step 1: Build baseline model
Step 2: Build tree over residuals (not labels)
Step 3': Predict the target labels using all trees (currently just one)
Prediction: avg + residual predicted by decision tree
For example for Gruyere: 167 + 246 = 413

### Idea Behind Gradient Boosting

<table>
<thead>
<tr>
<th>Desc</th>
<th>Protein</th>
<th>Carb</th>
<th>Sugar</th>
<th>Iron</th>
<th>Kcal</th>
<th>Residual</th>
<th>Predictions</th>
<th>New Residuals</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHEESE, GRUYERE</td>
<td>30</td>
<td>0</td>
<td>0.36</td>
<td>0.17</td>
<td>413</td>
<td>246</td>
<td>413</td>
<td>0</td>
</tr>
<tr>
<td>ICE CRM SNDWCH</td>
<td>4</td>
<td>37</td>
<td>18.57</td>
<td>0.26</td>
<td>237</td>
<td>70</td>
<td>237</td>
<td>0</td>
</tr>
<tr>
<td>PORK, LOIN</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>0.84</td>
<td>143</td>
<td>-24</td>
<td>130</td>
<td>13</td>
</tr>
<tr>
<td>CARROTS, RAW</td>
<td>1</td>
<td>10</td>
<td>4.74</td>
<td>0.3</td>
<td>41</td>
<td>-126</td>
<td>47</td>
<td>-6</td>
</tr>
<tr>
<td>APPLES, RAW, WITH SKIN</td>
<td>0</td>
<td>14</td>
<td>10.39</td>
<td>0.12</td>
<td>52</td>
<td>-115</td>
<td>47</td>
<td>6</td>
</tr>
<tr>
<td>BEEF, STRIP STEAKS</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>1.85</td>
<td>117</td>
<td>-50</td>
<td>130</td>
<td>-13</td>
</tr>
</tbody>
</table>

**Step 1:** Build baseline model  
**Step 2:** Build tree over residuals (not labels)  
**Step 3':** Predict the target labels using all trees (currently just one)  
**Step 4':** Compute the new residuals

![Decision tree diagram]
IDEA BEHIND GRADIENT BOOSTING

<table>
<thead>
<tr>
<th>Desc</th>
<th>Protein</th>
<th>Carb</th>
<th>Sugar</th>
<th>Iron</th>
<th>Kcal</th>
<th>Residual</th>
<th>Predictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHEESE,GRUYERE</td>
<td>30</td>
<td>0</td>
<td>0.36</td>
<td>0.17</td>
<td>413</td>
<td>246</td>
<td>192</td>
</tr>
<tr>
<td>ICE CRM SNDWCH</td>
<td>4</td>
<td>37</td>
<td>18.57</td>
<td>0.26</td>
<td>237</td>
<td>70</td>
<td>174</td>
</tr>
<tr>
<td>PORK,LOIN</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>0.84</td>
<td>143</td>
<td>-24</td>
<td>163</td>
</tr>
<tr>
<td>CARROTS,RAW</td>
<td>1</td>
<td>10</td>
<td>4.74</td>
<td>0.3</td>
<td>41</td>
<td>-126</td>
<td>155</td>
</tr>
<tr>
<td>APPLES,RAW,WITH SKIN</td>
<td>0</td>
<td>14</td>
<td>10.39</td>
<td>0.12</td>
<td>52</td>
<td>-115</td>
<td>155</td>
</tr>
<tr>
<td>BEEF,STRIP STEAKS</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>1.85</td>
<td>117</td>
<td>-50</td>
<td>163</td>
</tr>
</tbody>
</table>

Step 1: Build baseline model
Step 2: Build tree over residuals (not labels)
Step 3: Predict the target labels using all trees with **Learning Rate**

Prediction: \( \text{avg} + \text{LR} \times \text{residual predicted by decision tree} \)

For example for Gruyere: \( 167 + 0.1 \times 246 = 191.6 \)
### IDEA BEHIND GRADIENT BOOSTING

<table>
<thead>
<tr>
<th>Desc</th>
<th>Protein</th>
<th>Carb</th>
<th>Sugar</th>
<th>Iron</th>
<th>Kcal</th>
<th>Residual1</th>
<th>Predictions1</th>
<th>Residual2</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHEESE, GRUYERE</td>
<td>30</td>
<td>0</td>
<td>0.36</td>
<td>0.17</td>
<td>413</td>
<td>246</td>
<td>192</td>
<td>221</td>
</tr>
<tr>
<td>ICE CRM SNDWCH</td>
<td>4</td>
<td>37</td>
<td>18.57</td>
<td>0.26</td>
<td>237</td>
<td>70</td>
<td>174</td>
<td>63</td>
</tr>
<tr>
<td>PORK, LOIN</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>0.84</td>
<td>143</td>
<td>-24</td>
<td>163</td>
<td>-20</td>
</tr>
<tr>
<td>CARROTS, RAW</td>
<td>1</td>
<td>10</td>
<td>4.74</td>
<td>0.3</td>
<td>41</td>
<td>-126</td>
<td>155</td>
<td>-114</td>
</tr>
<tr>
<td>APPLES, RAW, WITH SKIN</td>
<td>0</td>
<td>14</td>
<td>10.39</td>
<td>0.12</td>
<td>52</td>
<td>-115</td>
<td>155</td>
<td>-103</td>
</tr>
<tr>
<td>BEEF, STRIP STEAKS</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>1.85</td>
<td>117</td>
<td>-50</td>
<td>163</td>
<td>-46</td>
</tr>
</tbody>
</table>

**Step 1:**  Build baseline model  
**Step 2:**  Build tree over residuals (not labels)  
**Step 3:**  Predict the target labels using all trees  
**Step 4:**  Compute the new residuals  
**Step 5:**  Repeat steps 3 to 5 until breaking criteria (test/validation error, iterations, etc.)

Once trained, use all of the trees in the ensemble to make a final prediction as to the value of the target variable:

\[
\text{AVG} + \text{LR} \times \text{Predicted Residual1} + \text{LR} \times \text{Predicted Residual2} + \ldots
\]

---

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.

https://colab.research.google.com/github/jakevdp/PythonDataScienceHandbook/blob/master/notebooks/05.09-Principal-Component-Analysis.ipynb#scrollTo=V7ssdJKEi9O1
Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.
Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.

https://colab.research.google.com/github/jakevdp/PythonDataScienceHandbook/blob/master/notebooks/05.09-Principal-Component-Analysis.ipynb#scrollTo=V7ssdJKEi9O1
FIRST PRINCIPLE COMPONENT
CLICKER: CAN WE USE LR TO FIND THE FIRST PRINCIPLE COMPONENT?

a) Yes
b) No

(if you say yes, argue why. If not, create a toy example to explain why not)
## PCA FOR NUTRITION DATA

<table>
<thead>
<tr>
<th>Shrt_Desc</th>
<th>Protein_(g)</th>
<th>Fiber_TD_(g)</th>
<th>Vit_C_(mg)</th>
<th>FA_Sat_(g)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHICKEN, GIZZARD, ALL CLASSES, RAW</td>
<td>17.66</td>
<td>0</td>
<td>3.7</td>
<td>0.529</td>
</tr>
<tr>
<td>PORK, FRSH, LEG (HAM), RUMP HALF, LN&amp;FAT, CKD, RSTD</td>
<td>27.03</td>
<td>0</td>
<td>0</td>
<td>3.369</td>
</tr>
<tr>
<td>BABYFOOD, DINNER, VEG&amp;TURKEY, STR</td>
<td>2.32</td>
<td>1.5</td>
<td>0.7</td>
<td>0.236</td>
</tr>
<tr>
<td>CEREALS RTE, FRSTD OAT CRL W/MARSHMALLOWS</td>
<td>7.1</td>
<td>4.3</td>
<td>20</td>
<td>0.62</td>
</tr>
<tr>
<td>CANDIES, FRUIT SNACKS, W/ HI VIT C</td>
<td>0.08</td>
<td>0</td>
<td>136.4</td>
<td>0</td>
</tr>
<tr>
<td>PORK, NORTHERN, CKD, DRY HEAT</td>
<td>24.69</td>
<td>0</td>
<td>3.8</td>
<td>0.151</td>
</tr>
<tr>
<td>APPLEBEE'S, CHICK TENDERS, FROM KIDS' MENU</td>
<td>19.25</td>
<td>1.2</td>
<td>2.852</td>
<td></td>
</tr>
<tr>
<td>FRUIT JUC SMOOTHIE, BOLTHOUSE FARMS, BERRY BOOST</td>
<td>0.63</td>
<td>0</td>
<td>108.6</td>
<td>0.003</td>
</tr>
<tr>
<td>NOODLES, EGG, DRY, UNENR</td>
<td>14.16</td>
<td>3.3</td>
<td>0</td>
<td>1.18</td>
</tr>
<tr>
<td>BABYFOOD, FRUIT, BANANAS W/ TAPIOCA, STR</td>
<td>0.4</td>
<td>1.6</td>
<td>16.7</td>
<td>0.035</td>
</tr>
<tr>
<td>EGG, WHOLE, COOKED, OMELET</td>
<td>10.57</td>
<td>0</td>
<td>0</td>
<td>3.319</td>
</tr>
<tr>
<td>FAST FOODS, BISCUIT, W/HAM</td>
<td>11.85</td>
<td>0.7</td>
<td>0.1</td>
<td>10.096</td>
</tr>
<tr>
<td>CEREALS RTE, GENERAL MILLS, COOKIE CRISP</td>
<td>5.2</td>
<td>5.1</td>
<td>23.1</td>
<td>0.8</td>
</tr>
<tr>
<td>CANDIES, MARS SNACKFOOD US, MILKY WAY MIDNIGHT BAR</td>
<td>3.2</td>
<td>2.9</td>
<td>0.2</td>
<td>11.474</td>
</tr>
<tr>
<td>BEEF, RND, BTM RND RST, LN, 1/8&quot; FAT, SEL, CKD, RSTD</td>
<td>28.45</td>
<td>0</td>
<td>0</td>
<td>1.608</td>
</tr>
<tr>
<td>BEEF, NZ, IMP, SUBCUTANEOUS FAT, CKD</td>
<td>6.5</td>
<td>0</td>
<td>0</td>
<td>30.256</td>
</tr>
<tr>
<td>WHEAT GERM, CRUDE</td>
<td>23.15</td>
<td>13.2</td>
<td>0</td>
<td>1.665</td>
</tr>
<tr>
<td>APPLEBEE'S, FRENCH FR</td>
<td>3.31</td>
<td>3.9</td>
<td>0.7</td>
<td>2.333</td>
</tr>
<tr>
<td>TSTR PSTRS, KELLOGG, KELLOGG'S LF POP TARTS, FRSTD</td>
<td>4.2</td>
<td>5.6</td>
<td>1.7</td>
<td></td>
</tr>
<tr>
<td>STRAWBERRY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHICKEN, BROILERS OR FRYERS, WING, MEAT ONLY, CKD, RSTD</td>
<td>30.46</td>
<td>0</td>
<td>0</td>
<td>2.26</td>
</tr>
<tr>
<td>DILL WEED, DRIED</td>
<td>19.96</td>
<td>13.6</td>
<td>50</td>
<td>0.234</td>
</tr>
<tr>
<td>FAT, BEEF TALLOW</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>49.8</td>
</tr>
<tr>
<td>TUNA, LT, CND IN H2O, WO/SALT, DRND SOL</td>
<td>25.51</td>
<td>0</td>
<td>0</td>
<td>0.234</td>
</tr>
<tr>
<td>PORK SAUSAGE, LINK/PATSY, UNPREP</td>
<td>15.39</td>
<td>0</td>
<td>0</td>
<td>7.57</td>
</tr>
<tr>
<td>INF FORMULA, NESTLE, GOOD START SUPREME, W/ IRON, PDR</td>
<td>11.3</td>
<td>0</td>
<td>46.1</td>
<td>11.66</td>
</tr>
<tr>
<td>BEEF, RIB, BACK RIBS, BONE-IN, LN, 0&quot; FAT, CHOIC, RAW</td>
<td>18.72</td>
<td>0</td>
<td>0</td>
<td>7.812</td>
</tr>
<tr>
<td>BREAD, PUMPERNICKEL</td>
<td>8.7</td>
<td>6.5</td>
<td>0</td>
<td>0.437</td>
</tr>
<tr>
<td>PIE, BANANA CRM, PREP FROM RECIPE</td>
<td>4.4</td>
<td>0.7</td>
<td>1.6</td>
<td>3.758</td>
</tr>
<tr>
<td>Digiorno Pizza, CHS TOPPING, CHS STUFFED CRUST, FRZ, BKD</td>
<td>13.48</td>
<td>1.9</td>
<td>0.5</td>
<td>5.63</td>
</tr>
<tr>
<td>PORK, FRSH, LOIN, BLADE (ROASTS), BNLESS, LN, CKD, RSTD</td>
<td>27.58</td>
<td>0</td>
<td>0</td>
<td>2.56</td>
</tr>
</tbody>
</table>
TRYING TO MAP FOOD INTO 1 DIM

https://annalyzin.wordpress.com/2016/06/15/principal-component-analysis-tutorial/
TRYING TO MAP FOOD INTO 1 DIM

Vitamin C

- Parsley
- Kale
- Broccoli
- Cauliflower
- Soybeans
- Yam
- Guinea Hen

(Vitamin C) - (Fat)

- Parsley
- Kale
- Broccoli
- Cauliflower
- Cabbage
- Spinach
- Yam
- Sweet Corn
- Guinea Hen
- Bluefish
- Mackerel
- Chicken

- Beef
- Pork
- Lamb
TRYING TO MAP FOOD INTO 1 DIM
Vitamin C
- Parsley
- Kale
- Broccoli
- Cauliflower
- Soybeans
- Yam
- Guinea Hen

(Vitamin C) - (Fat)
- Parsley
- Kale
- Broccoli
- Cauliflower
- Cabbage
- Spinach
- Yam
- Sweet Corn
- Guinea Hen
- Bluefish
- Mackerel
- Chicken
- Beef
- Pork
- Lamb

(Vitamin C + Fiber) - (Fat)
- Parsley
- Peas
- Lotus Root
- Chives
- Cauliflower
- Soybeans
- Eggplant
- Sweet Corn
- Mushrooms
- Haddock
- Guinea Hen
- Bluefish
- Mackerel
- Chicken
- Beef
- Pork
- Lamb

<table>
<thead>
<tr>
<th></th>
<th>PC1</th>
<th>PC2</th>
<th>PC3</th>
<th>PC4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fat</td>
<td>-0.45</td>
<td>0.66</td>
<td>0.58</td>
<td>0.18</td>
</tr>
<tr>
<td>Protein</td>
<td>-0.55</td>
<td>0.21</td>
<td>-0.46</td>
<td>-0.67</td>
</tr>
<tr>
<td>Fiber</td>
<td>0.55</td>
<td>0.19</td>
<td>0.43</td>
<td>-0.69</td>
</tr>
<tr>
<td>Vitamin C</td>
<td>0.44</td>
<td>0.70</td>
<td>-0.52</td>
<td>0.22</td>
</tr>
</tbody>
</table>
https://annalyzin.wordpress.com/2016/06/15/principal-component-analysis-tutorial/
HOW MANY COMPONENTS TO USE

https://annalyzin.wordpress.com/2016/06/15/principal-component-analysis-tutorial/
Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.