Web/App Intelligence Part I: Supervised Machine Learning

Shan-Hung Wu
shwu@cs.nthu.edu.tw

Department of Computer Science,
National Tsing Hua University, Taiwan

Software Design & Studio

Outline

1. **Web/App Intelligence**

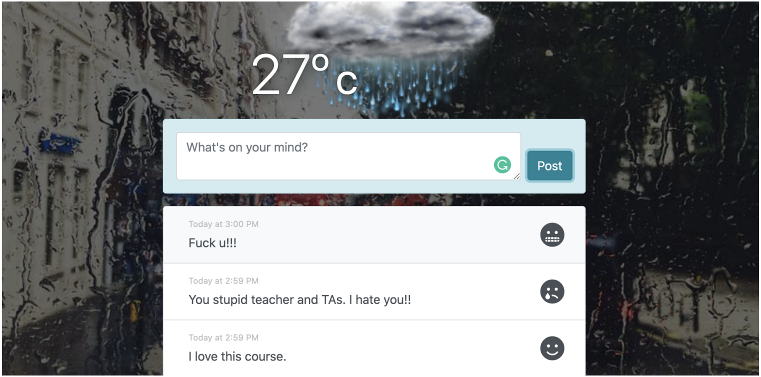
2. **What's Machine Learning?**

3. **Post Toxicity Detection**

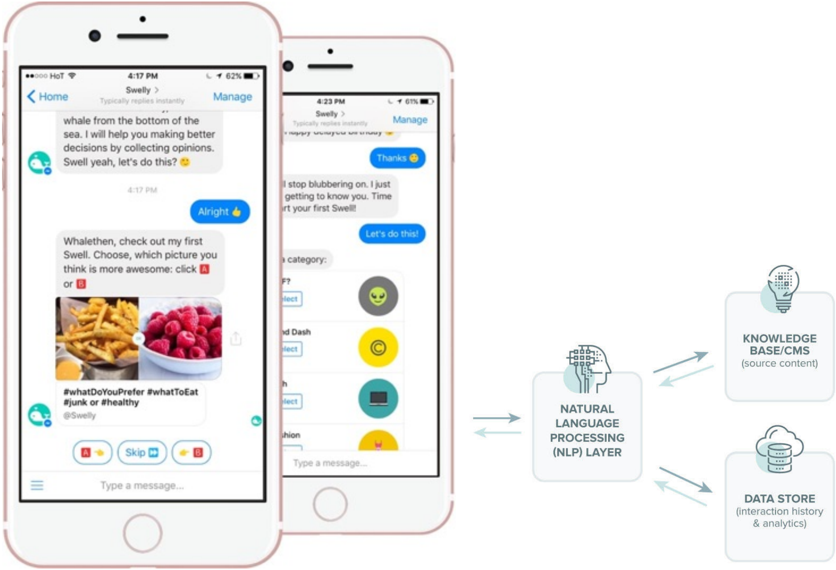
Outline

Let's Make WeatherMood More Intelligent...

```
$ git clone weathermood-toxicity-detection
$ npm install
$ npm run start
```

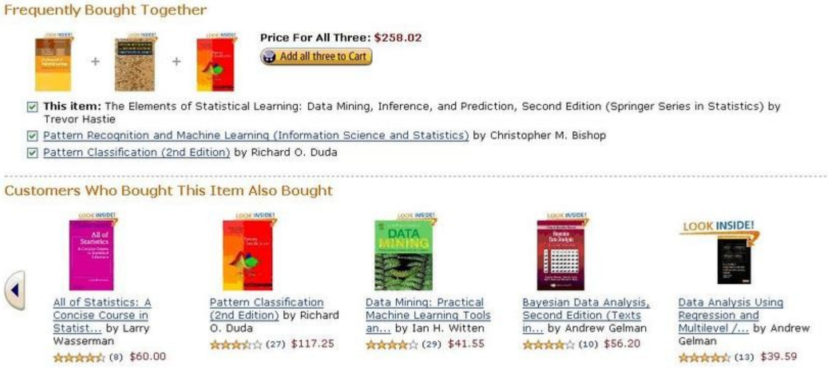


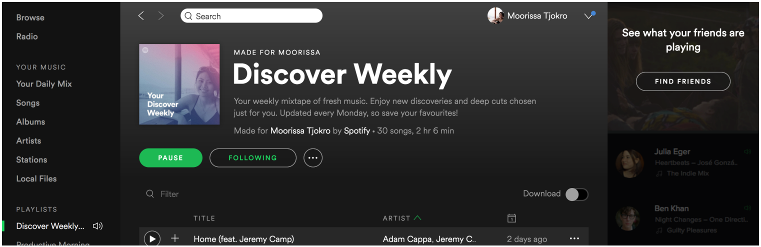
Customer Service Automation



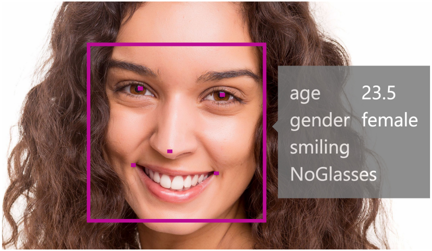
Spam Detection

Product Recommendations

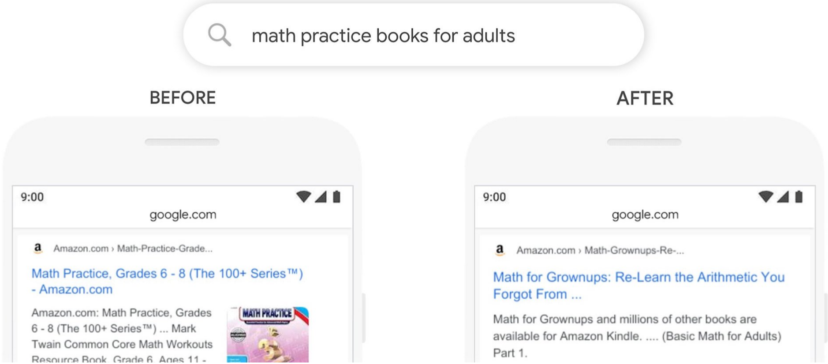




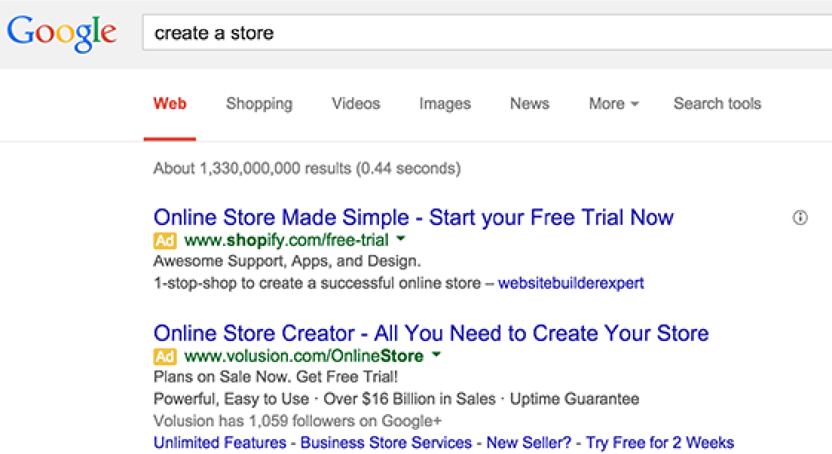
Customer Profiling



Intention Identification



Marketing & Advertisement



- And much more...

How to do it?

How to do it?

Machine Learning

or Data Mining, Deep Learning, NLP, CV, etc.

Outline

Prior vs. Posteriori Knowledge

- To solve a problem, we need an algorithm
  - E.g., sorting

Prior vs. Posteriori Knowledge

- To solve a problem, we need an algorithm
  - E.g., sorting
  - ***A priori knowledge*** is enough

Prior vs. Posteriori Knowledge

- To solve a problem, we need an algorithm
  - E.g., sorting
  - ***A priori knowledge*** is enough
- For some problem, however, we do not have the a priori knowledge
  - E.g., to tell if a post is toxic or not
  - The correct answer varies in time and from site to site

Prior vs. Posteriori Knowledge

- To solve a problem, we need an algorithm
  - E.g., sorting
  - ***A priori knowledge*** is enough
- For some problem, however, we do not have the a priori knowledge
  - E.g., to tell if a post is toxic or not
  - The correct answer varies in time and from site to site
- Machine learning algorithms use the ***a posteriori knowledge*** to solve problems

Prior vs. Posteriori Knowledge

- To solve a problem, we need an algorithm
  - E.g., sorting
  - ***A priori knowledge*** is enough
- For some problem, however, we do not have the a priori knowledge
  - E.g., to tell if a post is toxic or not
  - The correct answer varies in time and from site to site
- Machine learning algorithms use the ***a posteriori knowledge*** to solve problems
  - Takes ***examples*** as extra input

Example Data $\mathbb{X}$ as Extra Input

- Unsupervised:

$$\mathbb{X} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}, \text{ where } \boldsymbol{x}^{(i)} \in \mathbb{R}^{D}$$

  - E.g., $\boldsymbol{x}^{(i)}$ a post

Example Data $\mathbb{X}$ as Extra Input

- Unsupervised:
$$\mathbb{X} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}, \text{ where } \boldsymbol{x}^{(i)} \in \mathbb{R}^{D}$$
    - E.g., $\boldsymbol{x}^{(i)}$ a post
- Supervised:
$$\mathbb{X} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^{N}, \text{ where } \boldsymbol{x}^{(i)} \in \mathbb{R}^{D} \text{ and } \boldsymbol{y}^{(i)} \in \mathbb{R}^{K},$$
    - E.g., label $y^{(i)} \in \{0, 1\}$ indicates if the post $\boldsymbol{x}^{(i)}$ is toxic

3 General Types of Learning (1/2)

- ***Supervised learning***: learn to predict the labels of future data points

$X \in \mathbb{R}^{N \times D}$: 

$\boldsymbol{y} \in \mathbb{R}^{N \times K}$: $[\boldsymbol{e}^{(6)}, \boldsymbol{e}^{(1)}, \boldsymbol{e}^{(9)}, \boldsymbol{e}^{(4)}, \boldsymbol{e}^{(2)}]$

$\boldsymbol{x}' \in \mathbb{R}^{D}$: 

$\boldsymbol{y}' \in \mathbb{R}^{K}$: ?

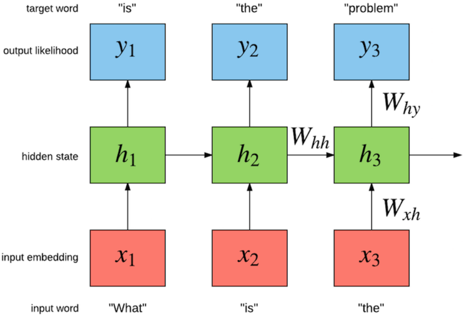
3 General Types of Learning (1/2)

- ***Supervised learning***: learn to predict the labels of future data points

$X \in \mathbb{R}^{N \times D}$:  $\boldsymbol{x}' \in \mathbb{R}^{D}$: 

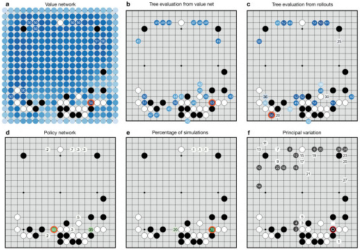
$\boldsymbol{y} \in \mathbb{R}^{N \times K}$: $[\boldsymbol{e}^{(6)}, \boldsymbol{e}^{(1)}, \boldsymbol{e}^{(9)}, \boldsymbol{e}^{(4)}, \boldsymbol{e}^{(2)}]$ $\boldsymbol{y}' \in \mathbb{R}^{K}$: ?

- ***Unsupervised learning***: learn patterns or latent factors in $\boldsymbol{X}$



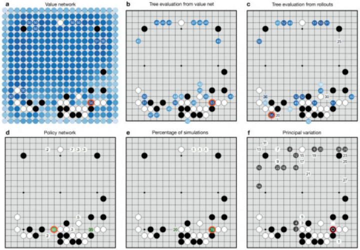
3 General Types of Learning (2/2)

- ***Reinforcement learning***: learn from “good”/“bad” feedback of actions (instead of correct labels) to maximize the goal



3 General Types of Learning (2/2)

- ***Reinforcement learning***: learn from "good"/"bad" feedback of actions (instead of correct labels) to maximize the goal



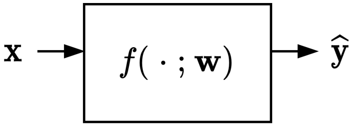
- AlphaGo is a hybrid of reinforcement learning and supervised learning
  - Supervised learning from the game records
  - Then, reinforcement learning from self-play

Supervised ML Step 1: Data Pre-processing

1. Data collection and exploration
2. Data preprocessing (e.g., integration, cleaning, etc.)

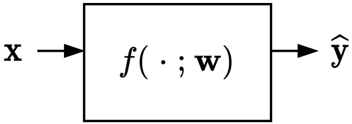
$$\mathbb{X} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^{N}$$

Supervised ML Step 2: Model Development



1. Assume a ***model*** $\{f(\cdot;\boldsymbol{w})\}_{\boldsymbol{w}}$ that is a collection of candidate functions f's
   - Each f predicts label $\hat{\boldsymbol{y}}$ given an input $\boldsymbol{x}$
   - f is assumed to be parametrized by $\boldsymbol{w}$

Supervised ML Step 2: Model Development

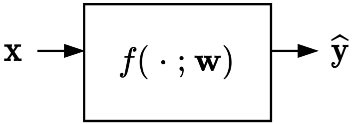


1. Assume a ***model*** $\{f(\cdot\,;\boldsymbol{w})\}_{\boldsymbol{w}}$ that is a collection of candidate functions f's
   - Each f predicts label $\hat{\boldsymbol{y}}$ given an input $\boldsymbol{x}$
   - f is assumed to be parametrized by $\boldsymbol{w}$
2. Define a ***cost function***

$$C(\boldsymbol{w};\mathbb{X})$$

   that measures "how good a particular $f(\cdot\,;\boldsymbol{w})$ can explain the training data $\mathbb{X}$" (posteriori knowledge)

Supervised ML Step 2: Model Development



1. Assume a ***model*** $\{f(\cdot;\boldsymbol{w})\}_{\boldsymbol{w}}$ that is a collection of candidate functions f's
   - Each f predicts label $\hat{y}$ given an input $\boldsymbol{x}$
   - f is assumed to be parametrized by $\boldsymbol{w}$
2. Define a ***cost function***

$$C(\boldsymbol{w};\mathbb{X})$$

   that measures "how good a particular $f(\cdot;\boldsymbol{w})$ can explain the training data $\mathbb{X}$" (posteriori knowledge)

3. ***Training***: employ an algorithm that solves

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} C(\boldsymbol{w};\mathbb{X})$$

Supervised ML Step 3: Testing & Deployment

1. ***Testing***: evaluate the performance of the learned $f(\cdot;\boldsymbol{w}^*)$ using another, ***unseen*** test dataset $\mathbb{X}'$
   - Examples in $\mathbb{X}'$ should have the same distribution with those in $\mathbb{X}$

Supervised ML Step 3: Testing & Deployment

1. ***Testing***: evaluate the performance of the learned $f(\cdot;\boldsymbol{w}^*)$ using another, ***unseen*** test dataset $\mathbb{X}'$
   - Examples in $\mathbb{X}'$ should have the same distribution with those in $\mathbb{X}$
2. If $f(\cdot;\boldsymbol{w}^*)$ has a good test performance, deploy it in a real world system

What is Deep Learning?

- ML where an $f(\cdot;\boldsymbol{w})$ has many (deep) layers

$$\hat{\boldsymbol{y}} = f^{(L)}(\cdots f^{(2)}(f^{(1)}(\boldsymbol{x};\boldsymbol{w}^{(1)});\boldsymbol{w}^{(2)})\cdots;\boldsymbol{w}^{(L)})$$

$$\mathbf{x} \rightarrow \boxed{f^{(1)}\left(\,\cdot\,;\mathbf{w}^{(1)}\right)} \rightarrow \boxed{f^{(2)}\left(\,\cdot\,;\mathbf{w}^{(2)}\right)} \rightarrow \cdots \boxed{f^{(L)}\left(\,\cdot\,;\mathbf{w}^{(L)}\right)} \rightarrow \widehat{\mathbf{y}}$$

What is Deep Learning?

- ML where an $f(\cdot;\boldsymbol{w})$ has many (deep) layers

$$\hat{\boldsymbol{y}} = f^{(L)}(\cdots f^{(2)}(f^{(1)}(\boldsymbol{x};\boldsymbol{w}^{(1)});\boldsymbol{w}^{(2)})\cdots;\boldsymbol{w}^{(L)})$$

$$\mathbf{x} \to \boxed{f^{(1)}\left(\,\cdot\,;\mathbf{w}^{(1)}\right)} \to \boxed{f^{(2)}\left(\,\cdot\,;\mathbf{w}^{(2)}\right)} \to \cdots \boxed{f^{(L)}\left(\,\cdot\,;\mathbf{w}^{(L)}\right)} \to \hat{\mathbf{y}}$$

- Pros:
  - Learns to pre-process data automatically
  - Learns a complex function (e.g., visual objects to labels)

What is Deep Learning?

- ML where an $f(\cdot;\boldsymbol{w})$ has many (deep) layers

$$\hat{\boldsymbol{y}} = f^{(L)}(\cdots f^{(2)}(f^{(1)}(\boldsymbol{x};\boldsymbol{w}^{(1)});\boldsymbol{w}^{(2)})\cdots;\boldsymbol{w}^{(L)})$$

$$\mathbf{x} \to \boxed{f^{(1)}\left(\cdot\,;\mathbf{w}^{(1)}\right)} \to \boxed{f^{(2)}\left(\cdot\,;\mathbf{w}^{(2)}\right)} \to \cdots \boxed{f^{(L)}\left(\cdot\,;\mathbf{w}^{(L)}\right)} \to \hat{\mathbf{y}}$$

- Pros:
  - Learns to pre-process data automatically
  - Learns a complex function (e.g., visual objects to labels)
- Cons:
  - Usually needs large data to train a model well
  - High computation costs (at both training and test time)

Outline

Supervised Learning

text	identity attack	insult	obscene	severe toxicity	sexual explicit	threat	toxicity
We're dudes on computers, moron. You are quite astonishingly stupid.	false	true	false	false	false	false	true

1. Get and preprocess a dataset, e.g., civil comments
    1. Training dataset: $\mathbb{X} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_i$
    2. Testing dataset: $\mathbb{X}' = \{(\boldsymbol{x}'^{(i)}, \boldsymbol{y}'^{(i)})\}_i$

[1] $1(\text{condition}) = 1$ if condition is true; otherwise 0.

Supervised Learning

text	identity attack	insult	obscene	severe toxicity	sexual explicit	threat	toxicity
We're dudes on computers, moron. You are quite astonishingly stupid.	false	true	false	false	false	false	true

1. Get and preprocess a dataset, e.g., civil comments
   1. Training dataset: $\mathbb{X} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_i$
   2. Testing dataset: $\mathbb{X}' = \{(\boldsymbol{x}'^{(i)}, \boldsymbol{y}'^{(i)})\}_i$
2. Model development
   1. ***Model***: $\{f : f(\boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w}^\top \boldsymbol{x}\}_{\boldsymbol{w}}$

[1] $1(\text{condition}) = 1$ if condition is true; otherwise 0.

Supervised Learning

text	identity attack	insult	obscene	severe toxicity	sexual explicit	threat	toxicity
We're dudes on computers, moron. You are quite astonishingly stupid.	false	true	false	false	false	false	true

1. Get and preprocess a dataset, e.g., civil comments
   1. Training dataset: $\mathbb{X} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_i$
   2. Testing dataset: $\mathbb{X}' = \{(\boldsymbol{x}'^{(i)}, \boldsymbol{y}'^{(i)})\}_i$
2. Model development
   1. ***Model***: $\{f : f(\boldsymbol{x};\boldsymbol{w}) = \boldsymbol{w}^\top \boldsymbol{x}\}_{\boldsymbol{w}}$
   2. ***Cost function***: $C(\boldsymbol{w};\mathbb{X}) = \Sigma_{(\boldsymbol{x}^{(i)},\boldsymbol{y}^{(i)})\in\mathbb{X}} 1(f(\boldsymbol{x}^{(i)};\boldsymbol{w}) \neq \boldsymbol{y}^{(i)})$[1]

[1] $1(\text{condition}) = 1$ if condition is true; otherwise 0.

Supervised Learning

text	identity attack	insult	obscene	severe toxicity	sexual explicit	threat	toxicity
We're dudes on computers, moron. You are quite astonishingly stupid.	false	true	false	false	false	false	true

1. Get and preprocess a dataset, e.g., civil comments
   1. Training dataset: $\mathbb{X} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_i$
   2. Testing dataset: $\mathbb{X}' = \{(\boldsymbol{x}'^{(i)}, \boldsymbol{y}'^{(i)})\}_i$
2. Model development
   1. ***Model***: $\{f : f(\boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w}^\top \boldsymbol{x}\}_{\boldsymbol{w}}$
   2. ***Cost function***: $C(\boldsymbol{w}; \mathbb{X}) = \Sigma_{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}) \in \mathbb{X}} 1(f(\boldsymbol{x}^{(i)}; \boldsymbol{w}) \neq \boldsymbol{y}^{(i)})$[1]
   3. ***Training***: to solve $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} C(\boldsymbol{w}; \mathbb{X})$

[1] $1(\text{condition}) = 1$ if condition is true; otherwise 0.

Supervised Learning

text	identity attack	insult	obscene	severe toxicity	sexual explicit	threat	toxicity
We're dudes on computers, moron. You are quite astonishingly stupid.	false	true	false	false	false	false	true

1. Get and preprocess a dataset, e.g., civil comments
   1. Training dataset: $\mathbb{X} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_i$
   2. Testing dataset: $\mathbb{X}' = \{(\boldsymbol{x}'^{(i)}, \boldsymbol{y}'^{(i)})\}_i$
2. Model development
   1. ***Model***: $\{f : f(\boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w}^\top \boldsymbol{x}\}_{\boldsymbol{w}}$
   2. ***Cost function***: $C(\boldsymbol{w}; \mathbb{X}) = \Sigma_{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}) \in \mathbb{X}} 1(f(\boldsymbol{x}^{(i)}; \boldsymbol{w}) \neq \boldsymbol{y}^{(i)})$[1]
   3. ***Training***: to solve $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} C(\boldsymbol{w}; \mathbb{X})$
3. ***Testing***: accuracy $\frac{1}{|\mathbb{X}'|} \Sigma_{(\boldsymbol{x}'^{(i)}, \boldsymbol{y}'^{(i)}) \in \mathbb{X}'} 1(f(\boldsymbol{x}'^{(i)}; \boldsymbol{w}^*) = \boldsymbol{y}'^{(i)})$

[1] $1(\text{condition}) = 1$ if condition is true; otherwise 0.

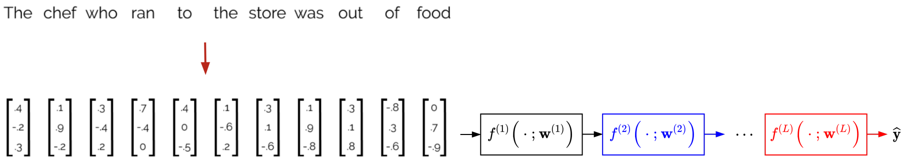
Supervised Learning

text	identity attack	insult	obscene	severe toxicity	sexual explicit	threat	toxicity
We're dudes on computers, moron. You are quite astonishingly stupid.	false	true	false	false	false	false	true

1. Get and preprocess a dataset, e.g., civil comments
   1. Training dataset: $\mathbb{X} = \{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})\}_i$
   2. Testing dataset: $\mathbb{X}' = \{(\boldsymbol{x}'^{(i)}, \boldsymbol{y}'^{(i)})\}_i$
2. Model development
   1. ***Model***: $\{f : f(\boldsymbol{x};\boldsymbol{w}) = \boldsymbol{w}^\top \boldsymbol{x}\}_{\boldsymbol{w}}$
   2. ***Cost function***: $C(\boldsymbol{w};\mathbb{X}) = \Sigma_{(\boldsymbol{x}^{(i)},\boldsymbol{y}^{(i)})\in\mathbb{X}} 1(f(\boldsymbol{x}^{(i)};\boldsymbol{w}) \neq \boldsymbol{y}^{(i)})$[1]
   3. ***Training***: to solve $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} C(\boldsymbol{w};\mathbb{X})$
3. ***Testing***: accuracy $\frac{1}{|\mathbb{X}'|}\Sigma_{(\boldsymbol{x}'^{(i)},\boldsymbol{y}'^{(i)})\in\mathbb{X}'} 1(f(\boldsymbol{x}'^{(i)};\boldsymbol{w}^*) = \boldsymbol{y}'^{(i)})$
   1. Then, integrate $f(\cdot;\boldsymbol{w}^*)$ into WeatherMood

[1] $1(\text{condition}) = 1$ if condition is true; otherwise 0.

Pre-trained Toxicity Classifier



- Google's Pre-trained Toxicity Classifier on GitHub
  - It's free
- Deep model:
  1. Transforms each word into a fixed-length vector
  2. Sums then normalizes the word vectors
  3. Feeds the sum into a deep classification model

Using the Pre-trained Toxicity Classifier

```
// installation
$ npm install @tensorflow/tfjs @tensorflow-models/toxicity

// usage in code
const toxicity = require('@tensorflow-models/toxicity');
const model = await toxicity.load(0.9); // threshold
const inputs = ['We're dudes on comupters, moron...'];
const classes = await model.classify(inputs);
inputs.forEach((text, i) => {
  console.log(text);
  classes.forEach(cls => {
    console.log(cls.label, cls.results[i].match);
  });
});
```

text	identity attack	insult	obscene	severe toxicity	sexual explicit	threat	toxicity
We're dudes on computers, moron. You are quite astonishingly stupid.	false	true	false	false	false	false	true

Demo 3

- Implement the prototype you shown in Demo 2
- Final project demo:
  - 6/20 1pm-6pm
  - 4 min for team (strict)
  - 10 min for QA

Demo 3

- Implement the prototype you shown in Demo 2
- Final project demo:
  - 6/20 1pm-6pm
  - 4 min for team (strict)
  - 10 min for QA
- Evaluation:
  - Completeness (60%)
  - Complexity (40%)

Completeness (60%)

- How many main features have you completed?
- List each of them
- How well does your final implementation match your Demo 2 design? (***40%***)
    - Key features
    - Key flows
    - UI & transitions

Complexity (40%)

- Explain ***one or two*** most
  - challenging aspects you implemented, or
- Discuss issues encountered and you solutions

Bonus

- Best Minimal Viable Products (MVPs)
  - *+15%*, *+10%*, and *+5%* for #1, #2, and #3, respectively
- Cross-team peer review
  - Each team has three non-self votes
  - Judged by completeness, complexity, and design
- Intra-team peer review
  - Scaled based on team score