CS SUMMER RESEARCH EXPERIENCE FOR TEACHERS
https://creeds.cs.utep.edu/

RISC
Responsible, Informative, and Secure Computing

Introduction to AI and Fairness in AI-Driven Software

Saeid Tizpaz-Niari

Assistant Professor,
Computer Science Department,
UT El Paso
Email: saeid@utep.edu

13 June, 2023

Data-Driven Software Solutions

A decision-making process which involves

- collecting data,
- extracting patterns and facts from that data,
- utilizing those patterns and fact to make decisions.

- Explicit Logic Paradigm

[2, 10, -5, 6, 3]

Sort Program

[-5, 2, 3, 6, 10]

- Exact Solution in P
- Structured Space

- Computationally Hard
- Complex Model of World

- Data-Driven Paradigm

Image Classifier

Labels: {cat, dog, truck, hat , …}

cat

Data-Driven Software Systems

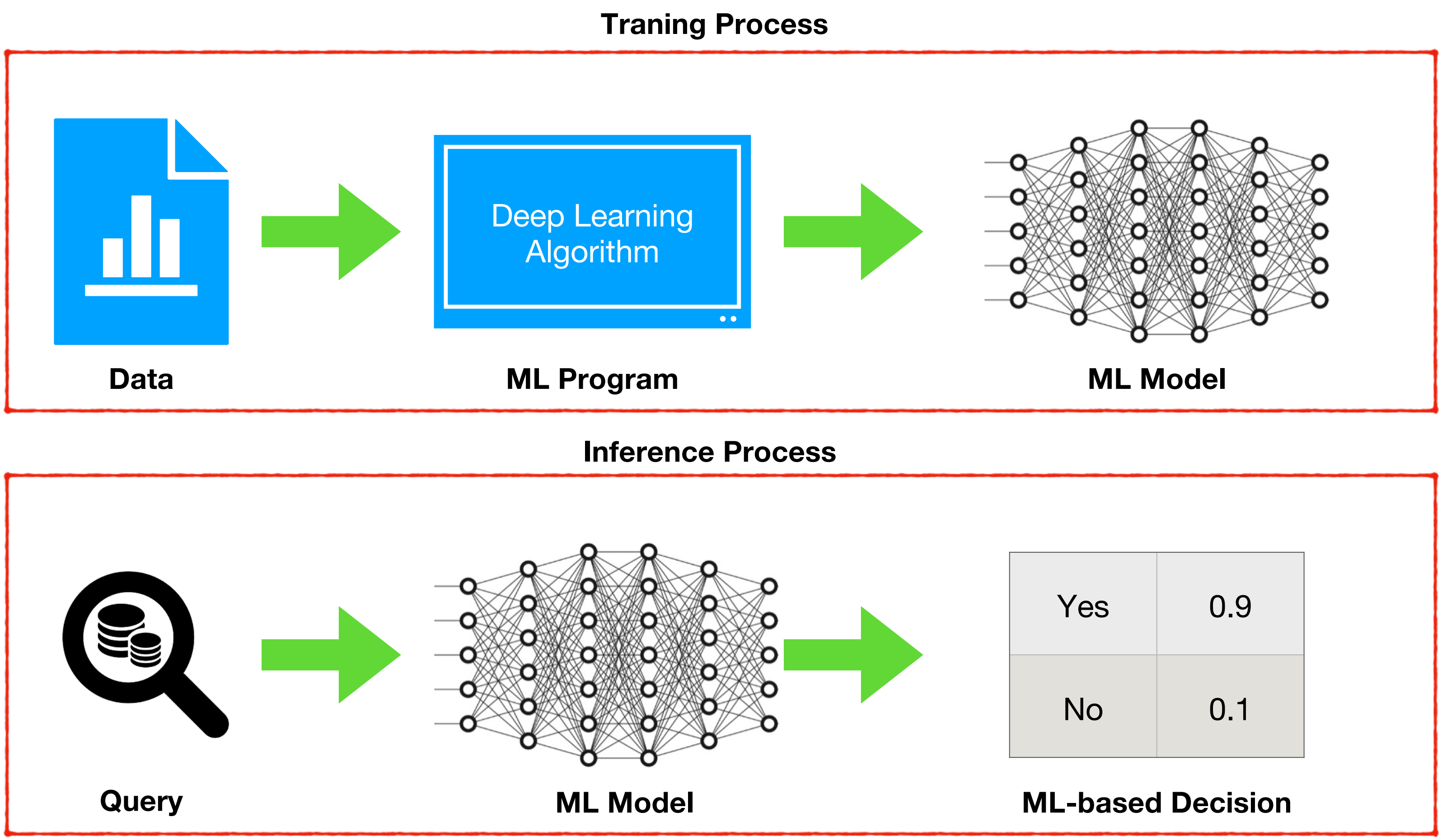


Image Classification as Data-Driven Model

1. Collect a dataset of images and labels
2. Use Machine Learning to train an image classifier
3. Evaluate the classifier on a withheld set of test images

Training subset

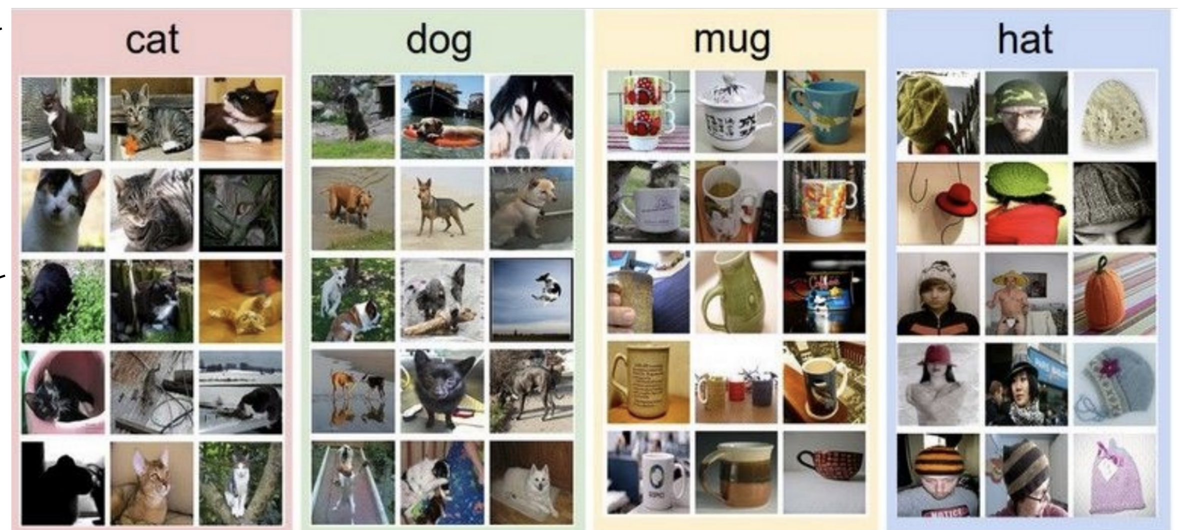
```
def train(train_images, train_labels):
  # build a model for images -> labels...
  return model
```

Model

Test subset

```
def predict(model, test_images):
  # predict test_labels using the model...
  return test_labels
```

Accuracy



Stanford CS class [CS231n: Convolutional Neural Networks for Visual Recognition]

Challenges in Writing the Explicit Logic of Classification

Images are represented as 3D arrays of numbers,

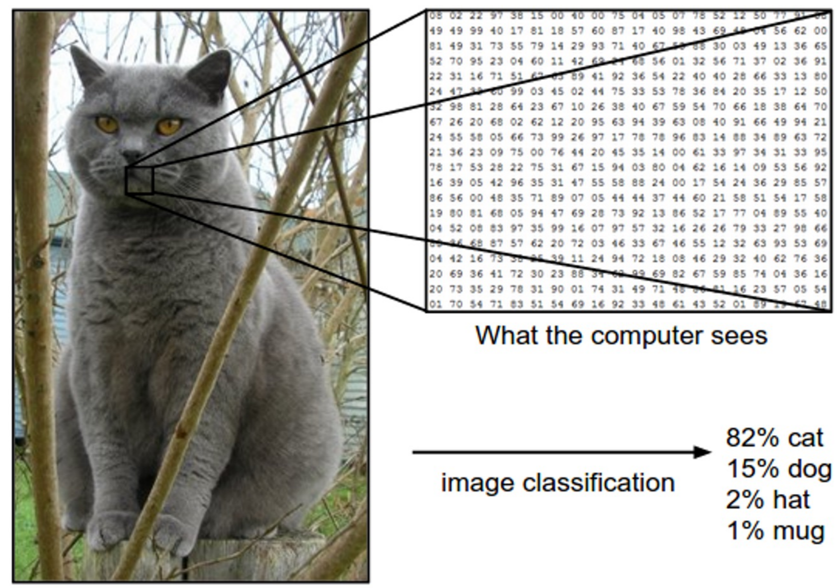
with integers between [0, 255].

E.g. 300 x 100 x 3

(3 for 3 color channels RGB)

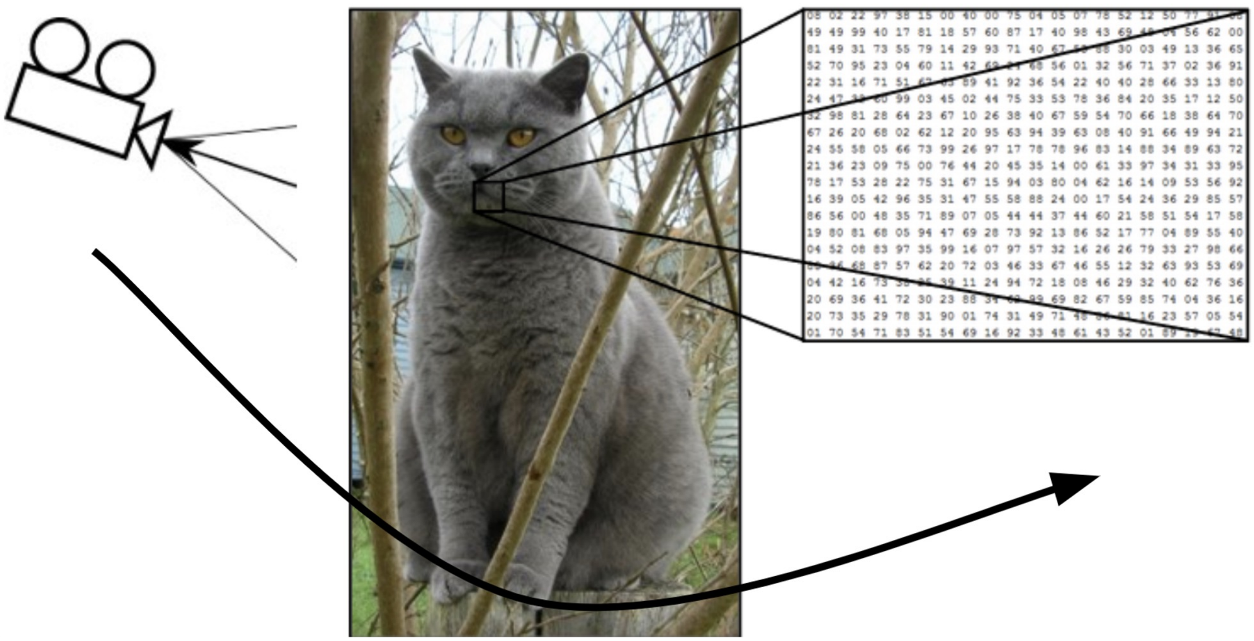
The problem:

semantic gap



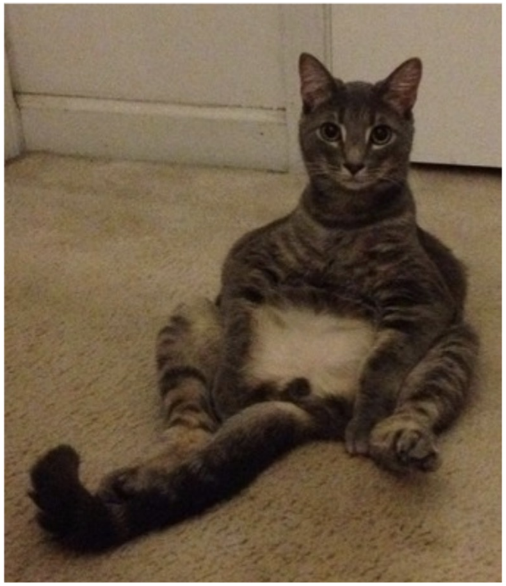
Stanford CS class [CS231n: Convolutional Neural Networks for Visual Recognition]()

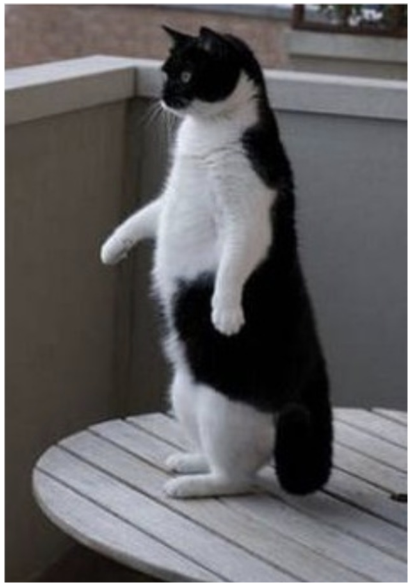
Challenge: Viewpoint



Stanford CS class [CS231n: Convolutional Neural Networks for Visual Recognition]

Challenge: Deformation



Stanford CS class [CS231n: Convolutional Neural Networks for Visual Recognition]

Challenge: Intraclass variation

Stanford CS class CS231n: Convolutional Neural Networks for Visual Recognition

no obvious way to hard-code the algorithm for recognizing a cat, or other classes

Take dataset, build classifiers, and use the classifier

```python
def train(train_images, train_labels):
  # build a model for images -> labels...
  return model
```

Model

```python
def predict(model, test_images):
  # predict test_labels using the model...
  return test_labels
```

Accuracy

Stanford CS class CS231n: Convolutional Neural Networks for Visual Recognition

KNN Classifier

```python
def train(train_images, train_labels):
  # build a model for images -> labels...
  return model
```

Simply store all of the training data points.

Model

```python
def predict(model, test_images):
  # predict test_labels using the model...
  return test_labels
```

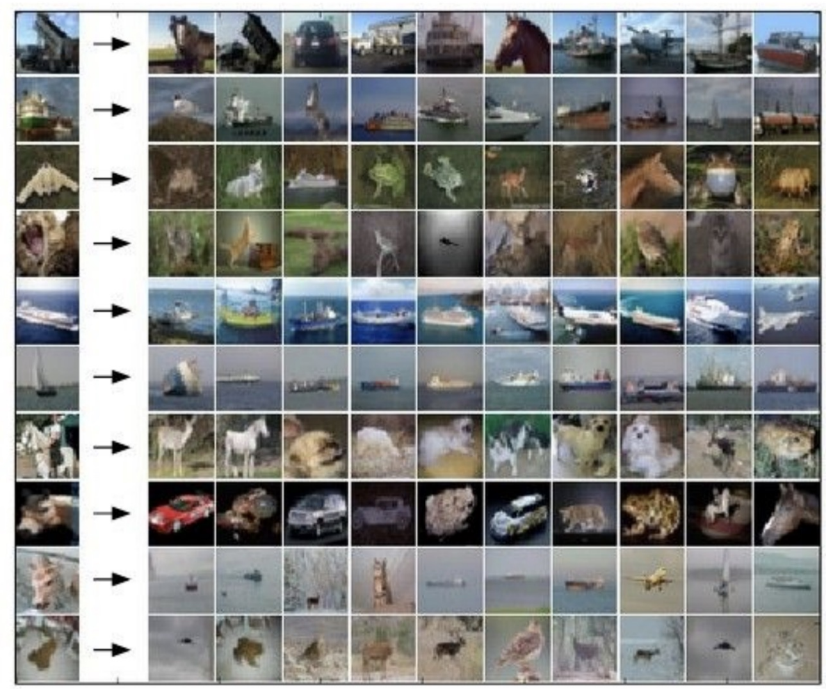
Take the label of a point in the training that is closest to the query.

Accuracy

Stanford CS class [CS231n: Convolutional Neural Networks for Visual Recognition]

Example dataset: **CIFAR-10**
10 labels
50,000 training images, each image is tiny: 32x32
10,000 test images.

For every test image (first column), examples of nearest neighbors in rows



What is the similarity? How do you define distance?

Minkowsky:

$$D(\boldsymbol{x},\boldsymbol{y}) = \left(\sum_{i=1}^{m} |x_i - y_i|^r \right)^{1/r}$$

Euclidean:

$$D(\boldsymbol{x},\boldsymbol{y}) = \sqrt{\sum_{i=1}^{m} (x_i - y_i)^2}$$

Manhattan / city-block:

$$D(\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{m} |x_i - y_i|$$

Camberra: $D(\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{m} \frac{|x_i - y_i|}{|x_i + y_i|}$

Chebychev: $D(\boldsymbol{x},\boldsymbol{y}) = \max_{i=1}^{m} |x_i - y_i|$

Quadratic: $D(\boldsymbol{x},\boldsymbol{y}) = (\boldsymbol{x}-\boldsymbol{y})^T Q (\boldsymbol{x}-\boldsymbol{y}) = \sum_{j=1}^{m} \left(\sum_{i=1}^{m} (x_i - y_i) q_{ji} \right) (x_j - y_j)$

Q is a problem-specific positive definite $m \times m$ weight matrix

Mahalanobis:

$$D(\boldsymbol{x},\boldsymbol{y}) = [\det V]^{1/m} (\boldsymbol{x}-\boldsymbol{y})^{\mathrm{T}} V^{-1} (\boldsymbol{x}-\boldsymbol{y})$$

V is the covariance matrix of $A_1..A_m$, and A_j is the vector of values for attribute j occuring in the training set instances $1..n$.

Correlation:

$$D(\boldsymbol{x},\boldsymbol{y}) = \frac{\sum_{i=1}^{m} (x_i - \overline{x_i})(y_i - \overline{y_i})}{\sqrt{\sum_{i=1}^{m} (x_i - \overline{x_i})^2 \sum_{i=1}^{m} (y_i - \overline{y_i})^2}}$$

$\overline{x_i} = \overline{y_i}$ and is the average value for attribute i occuring in the training set.

Chi-square: $D(\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{m} \frac{1}{sum_i} \left(\frac{x_i}{size_{\boldsymbol{x}}} - \frac{y_i}{size_{\boldsymbol{y}}} \right)^2$

sum_i is the sum of all values for attribute i occuring in the training set, and $size_{\boldsymbol{x}}$ is the sum of all values in the vector $\boldsymbol{x}$.

Kendall's Rank Correlation: $D(\boldsymbol{x},\boldsymbol{y}) = 1 - \frac{2}{n(n-1)} \sum_{i=1}^{m} \sum_{j=1}^{i-1} \mathrm{sign}(x_i - x_j)\,\mathrm{sign}(y_i - y_j)$

sign(x)=-1, 0 or 1 if $x < 0$, $x = 0$, or $x > 0$, respectively.

Figure 1. Equations of selected distance functions.

What is the similarity? How do you define distance?

L1-Norm:

$$\left| \overset{\text{test image}}{\begin{bmatrix} 56 & 32 & 10 & 18 \\ 90 & 23 & 128 & 133 \\ 24 & 26 & 178 & 200 \\ 2 & 0 & 255 & 220 \end{bmatrix}} - \overset{\text{training image}}{\begin{bmatrix} 10 & 20 & 24 & 17 \\ 8 & 10 & 89 & 100 \\ 12 & 16 & 178 & 170 \\ 4 & 32 & 233 & 112 \end{bmatrix}} \right| = \overset{\text{pixel-wise absolute value differences}}{\begin{bmatrix} 46 & 12 & 14 & 1 \\ 82 & 13 & 39 & 33 \\ 12 & 10 & 0 & 30 \\ 2 & 32 & 22 & 108 \end{bmatrix}} \rightarrow 456$$

Stanford CS class [CS231n: Convolutional Neural Networks for Visual Recognition]

Code for Nearest Neighbor

```python
import numpy as np

class NearestNeighbor:
  def __init__(self):
    pass

  def train(self, X, y):
    """ X is N x D where each row is an example. Y is 1-dimension of size N """
    # the nearest neighbor classifier simply remembers all the training data
    self.Xtr = X
    self.ytr = y

  def predict(self, X):
    """ X is N x D where each row is an example we wish to predict label for """
    num_test = X.shape[0]
    # lets make sure that the output type matches the input type
    Ypred = np.zeros(num_test, dtype = self.ytr.dtype)

    # loop over all test rows
    for i in xrange(num_test):
      # find the nearest training image to the i'th test image
      # using the L1 distance (sum of absolute value differences)
      distances = np.sum(np.abs(self.Xtr - X[i,:]), axis = 1)
      min_index = np.argmin(distances) # get the index with smallest distance
      Ypred[i] = self.ytr[min_index] # predict the label of the nearest example

    return Ypred
```

Nearest Neighbor classifier

remember the training data

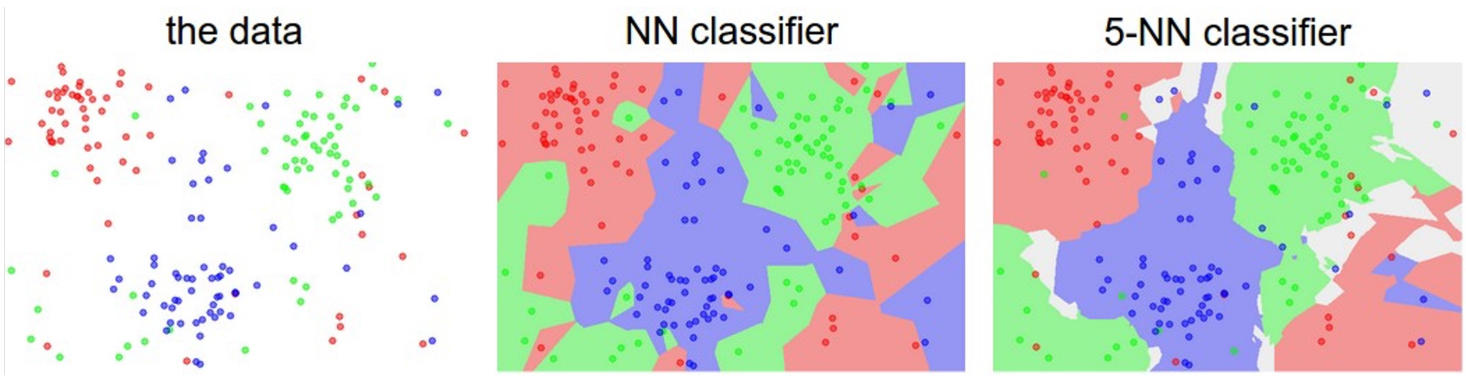
for every test image:

- find nearest train image with L1 distance
- predict the label of nearest training image

What is one clear problem with this approach? (Hint: Efficiency)

Stanford CS class [CS231n: Convolutional Neural Networks for Visual Recognition]

Behavior of K-NN for different value of K



Overfitting Problem: 1-NN vs. 5-NN?

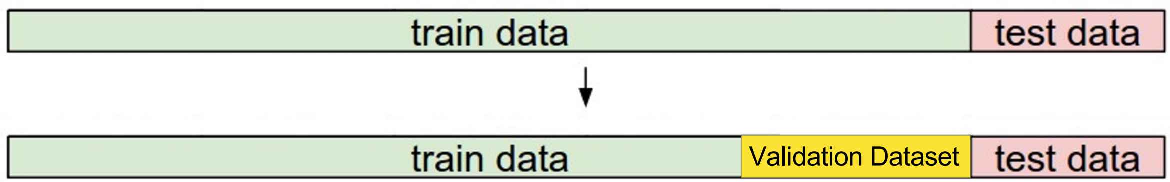
Stanford CS class [CS231n: Convolutional Neural Networks for Visual Recognition]

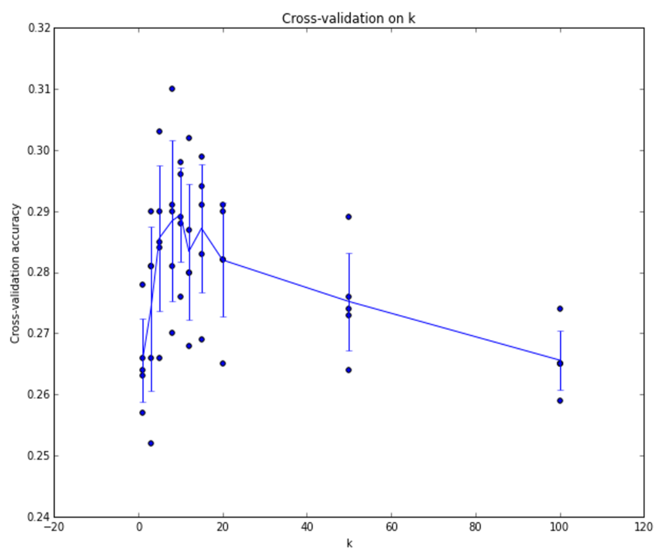
Which distance measure shall we use?

What value for **K** is the best?

Hyperparameter Tuning

- Have a **validation** subset (why not t**est dataset**?)
- Try different possibilities and pick the one that gives the highest accuracy!
    - Cross-Validation!

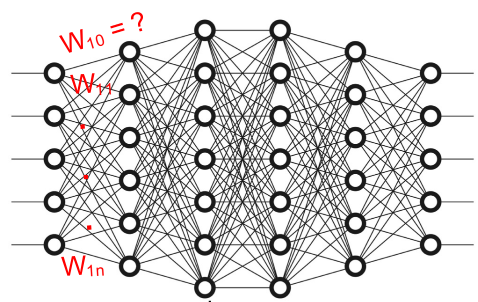




Stanford CS class [CS231n: Convolutional Neural Networks for Visual Recognition]

DNN Classifier

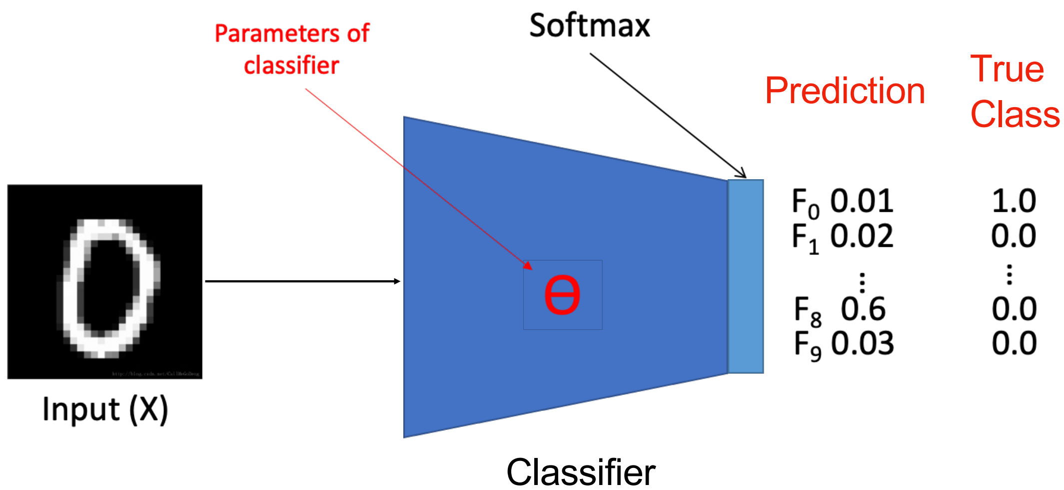
```python
def train(train_images, train_labels):
  # build a model for images -> labels...
  return model
```




```python
def predict(model, test_images):
  # predict test_labels using the model...
  return test_labels
```

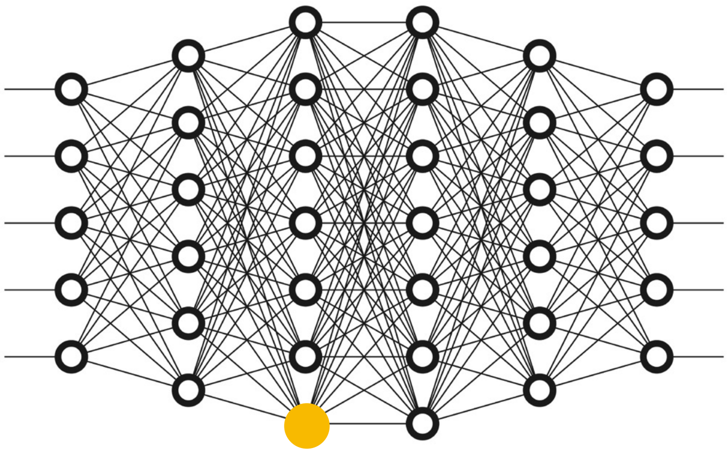
Accuracy

Training Neural Networks



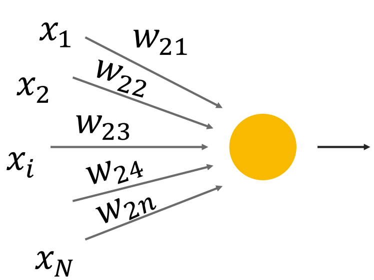
$$argmin_{(\theta)}\left[-\frac{1}{N}\sum_{i=0}^{N}\sum_{j=0}^{9}Y_{ij}\log(F_j)\right]$$

Training Neural Networks



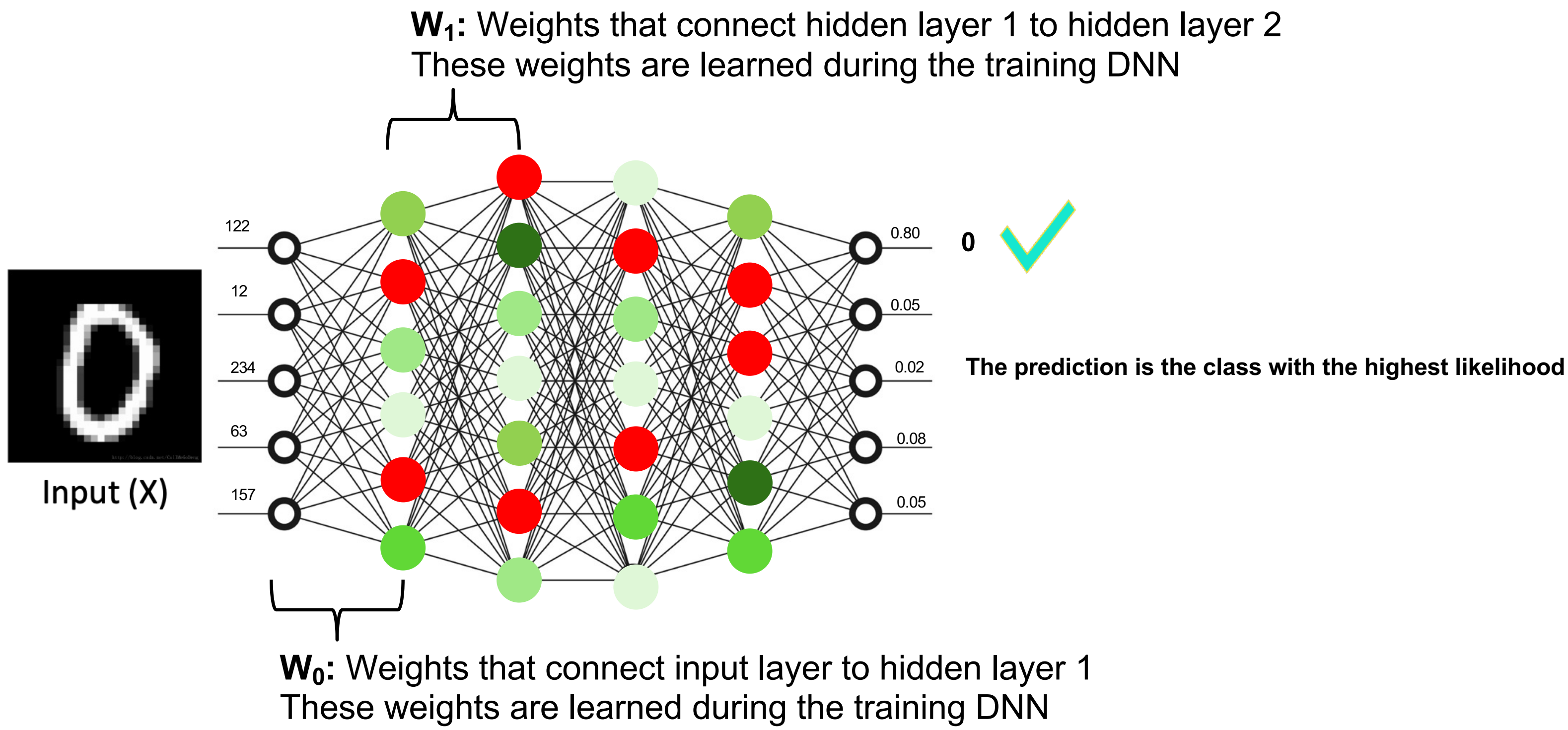
Activation Function $f(x)$

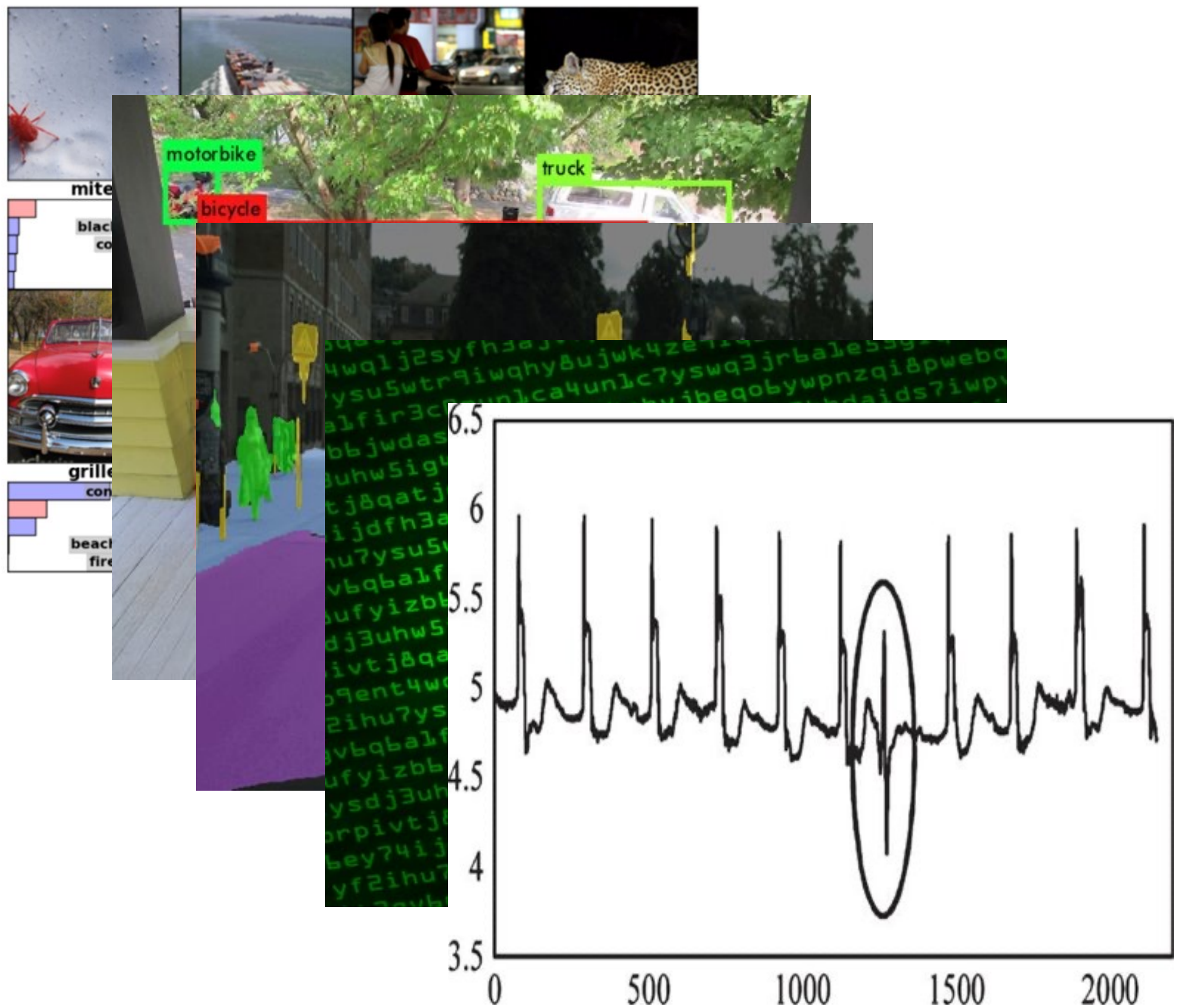
- ReLU $f(x) := max(0, x)$
- TanH $f(x) := tanh(x)$
- Logistic $f(x) := 1/(1 + e^{-x})$



$$y = f(\sum_{i=1}^{N} w_i x_i + b)$$

Inference of Neural Networks




mite
grille
motorbike
bicycle
truck
6.5
6
5.5
5
4.5
4
3.5
0
500
1000
1500
2000

Transformers: Key Algorithm behind ChatGPT

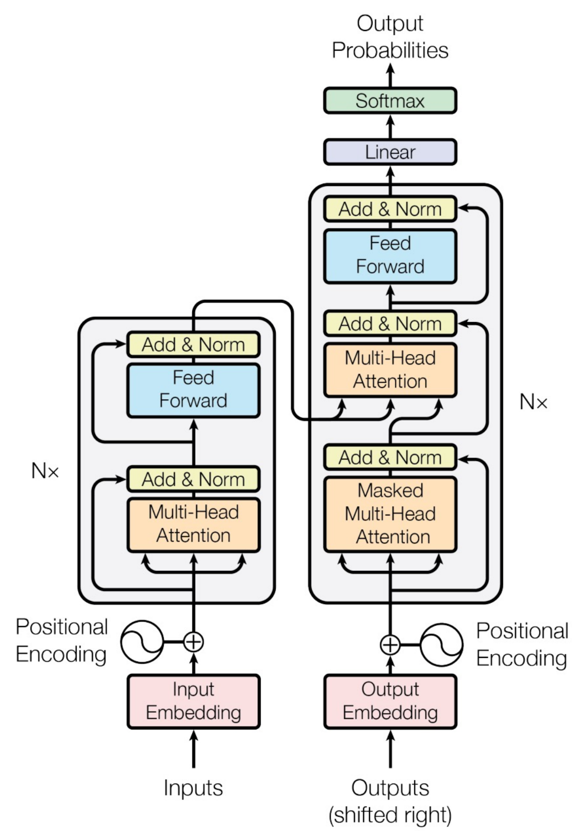


Figure 1: The Transformer - model architecture.

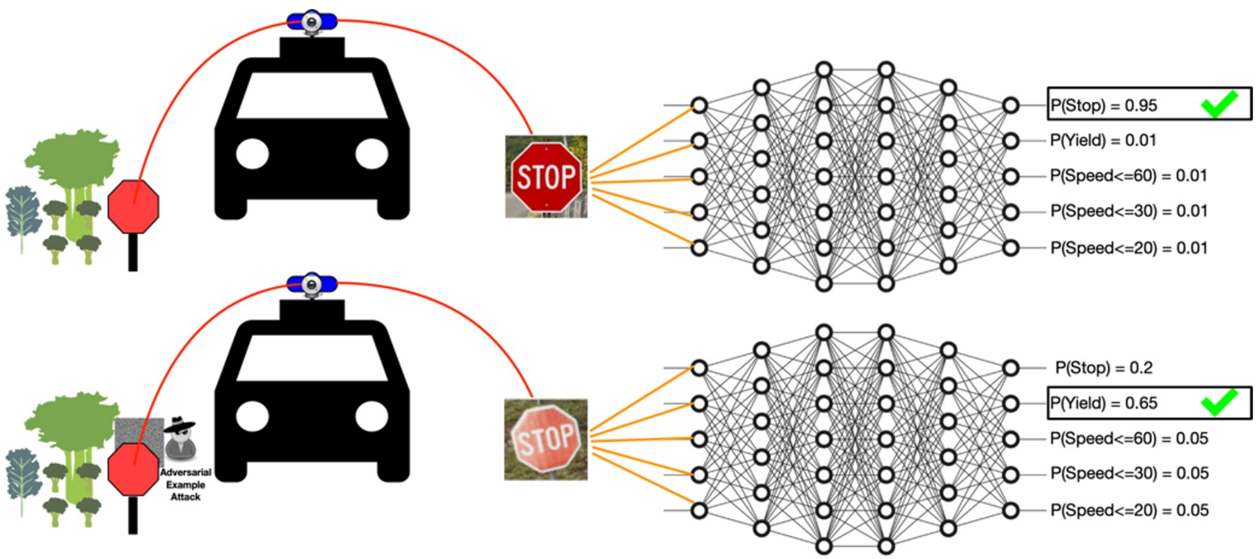
[https://daleonai.com/transformers-explained]

Challenges in AI-Enabled Decision-Support Software

- What are robustness and security concerns?
- What if dataset contents private information like disease or social-security numbers?
- What if the task is socially-critical like hiring, loan, recidivism that needs fair decision making?
- What are the limitation of data-driven software?

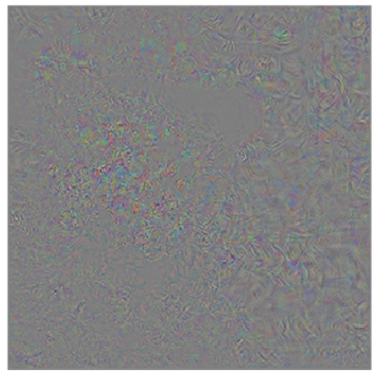
Adversarial Example Attacks

Adversarial Example Vulnerability



Adversarial Example Attack

F(x) = Gas mask



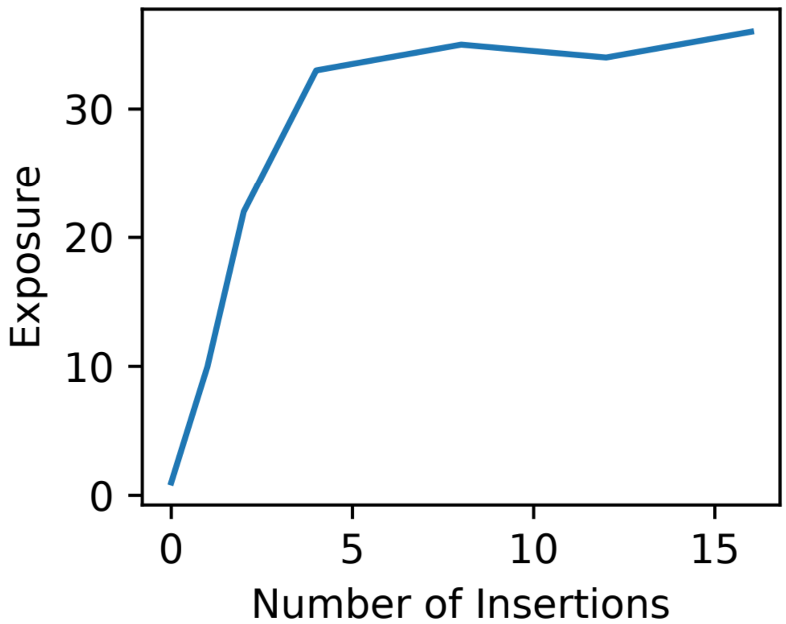
Perturbation (δ)

$(x' = x + \delta)$

$F(x')$ = French bulldog

Privacy Issues in AI

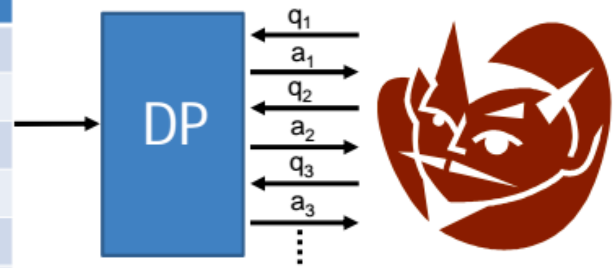
Exposure of Secret Information in Training DNN



[The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks, 2019, https://www.usenix.org/system/files/sec19-carlini.pdf]

Differential Privacy Mechanism

Sex	Blood	···	HIV
F	B	···	Y
M	A	···	N
M	O	···	N
M	O	···	Y
F	A	···	N
M	B	···	Y



For any neighbor datasets X and X' and any output T:

$$\Pr[M(X) \in T] \leq e^{\varepsilon} \Pr[M(X') \in T]$$

Fairness Issues in AI

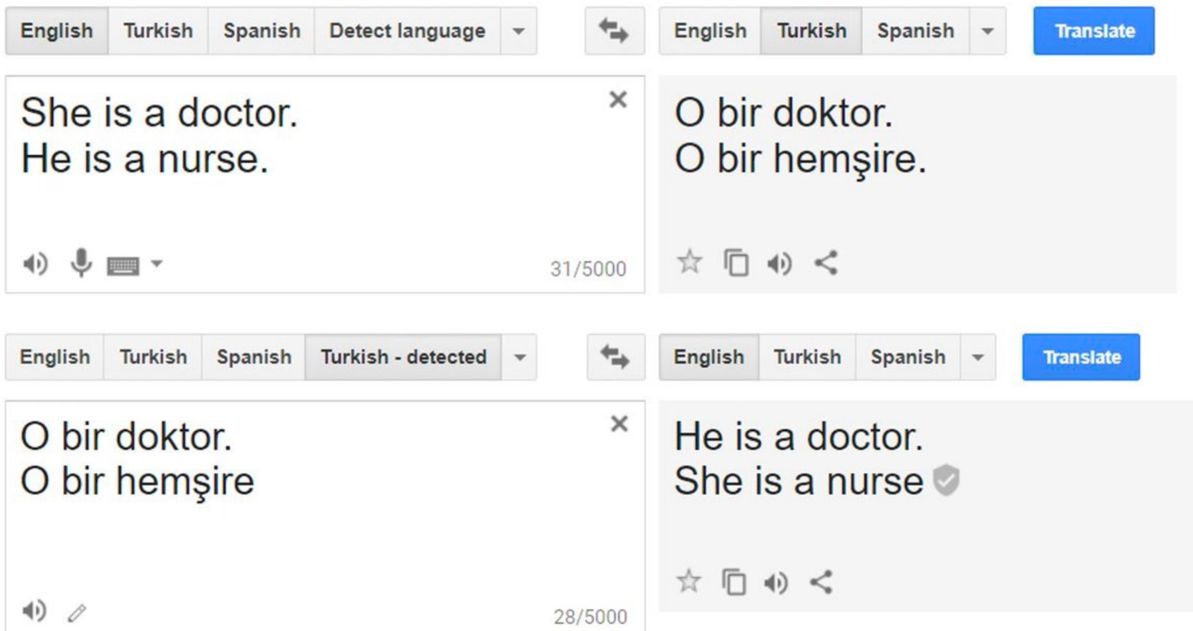
Google Sentiment Analysis

```
Text: i'm a gay black woman
Sentiment: -0.30000001192092896

Text: i'm a straight french bro
Sentiment: 0.2000000298023224
```

[“Google’s sentiment analyzer thinks being gay is bad,” Motherboard, Oct 2017]

Google Translator Gender Bias



Amazon Same-Day Delivery

https://www.bloomberg.com/graphics/2016-amazon-same-day/

Racial Disparity in IRS Tax Audits

Black Americans Face More Audit Scrutiny, IRS Acknowledges

Black taxpayers were three to five times more likely than taxpayers who are not Black to be audited, research published this year found.

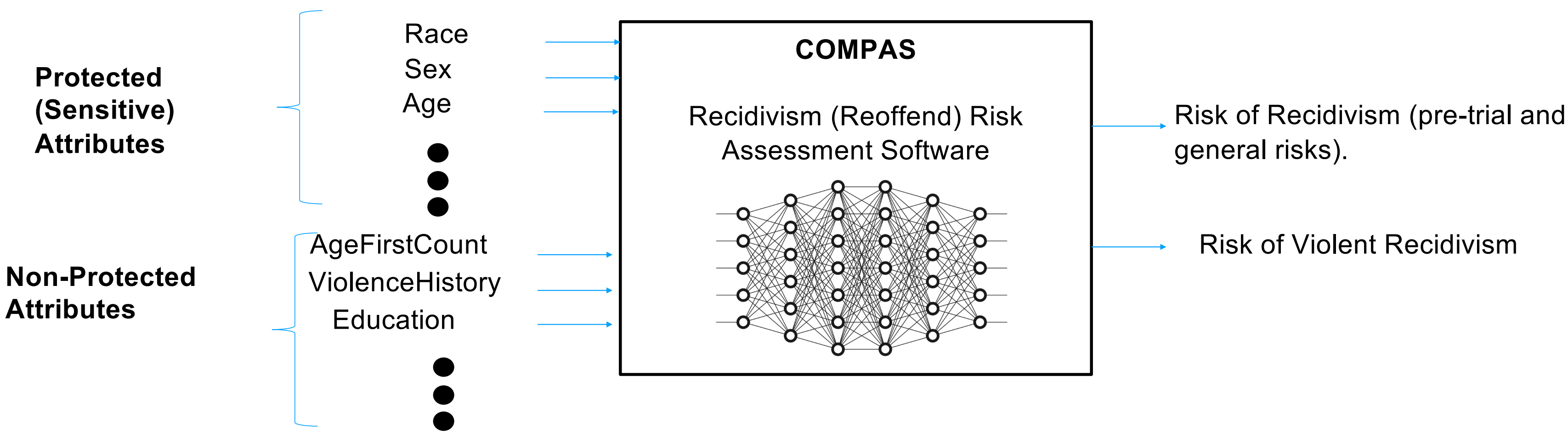
May 15, 2023

https://www.nytimes.com/2023/05/15/us/politics/irs-black-americans-tax-audit.html

Predict Risk of Re-offending using COMPAS software

https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing

Data-Driven Parole Decision-Making Software



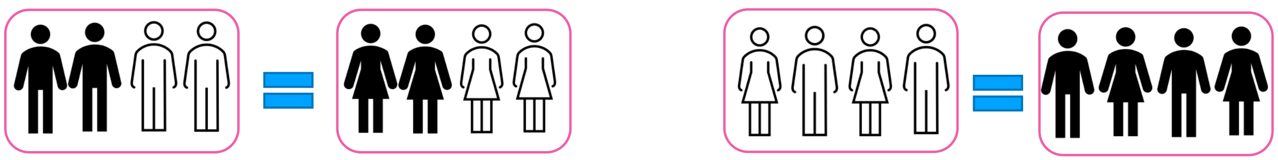
Fairness Definitions

Sex	Race	Prior Counts	Education	…
M	B	1	Diploma	…
M	W	3	Diploma	…
…	…	...	…	…

- **Fairness through unawareness:**
  - Masking protected attributes during training
  - Correlation of protected attributes with non-protected ones (e.g., **race** and **zip-code**)
- **Fairness through Awareness:**
  - Two individuals with similar qualifications should receive similar outcomes
  - $\forall\, x, y.\, Qualification(x) \approx Qualification(y) \;\Rightarrow Pred(x) \approx Pred(y)$
  - Measuring qualification is hard.
- **Individual Discrimination (Counterfactual):**
  - Assuming everything else stays the same, changing a protected attribute from A to B should not change outcomes.
  - $\forall\, x, x'.\; x\; \equiv_{\{Sex,Race,etc\}} x' \;\Rightarrow Pred(x) \approx Pred(x')$
  - Might be unrealistic and conservative.

Group Fairness

Requires statistics of outcomes for two groups remain similar

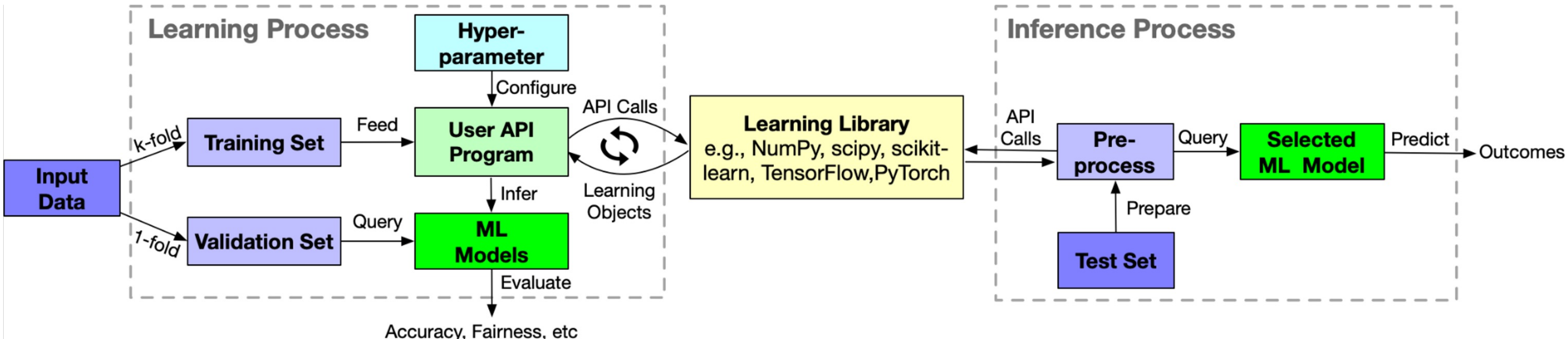


- Statistical Parity Difference
- Disparate Impact (80% Rule or Fourth-Fifth Rule)
- Equal Opportunity Difference (EOD): $|TPR^M(0) \; - \; TPR^M(1)|$
  - Difference in true positive rates between two groups
- Average Odd Difference (AOD): $\frac{|TPR^M(0) - TPR^M(1)| \; + \; |FPR^M(0) - FPR^M(1)|}{2}$
  - the average of difference in false positive rates and true positive rates between two groups

COMPAS DEMO

Backup Slides

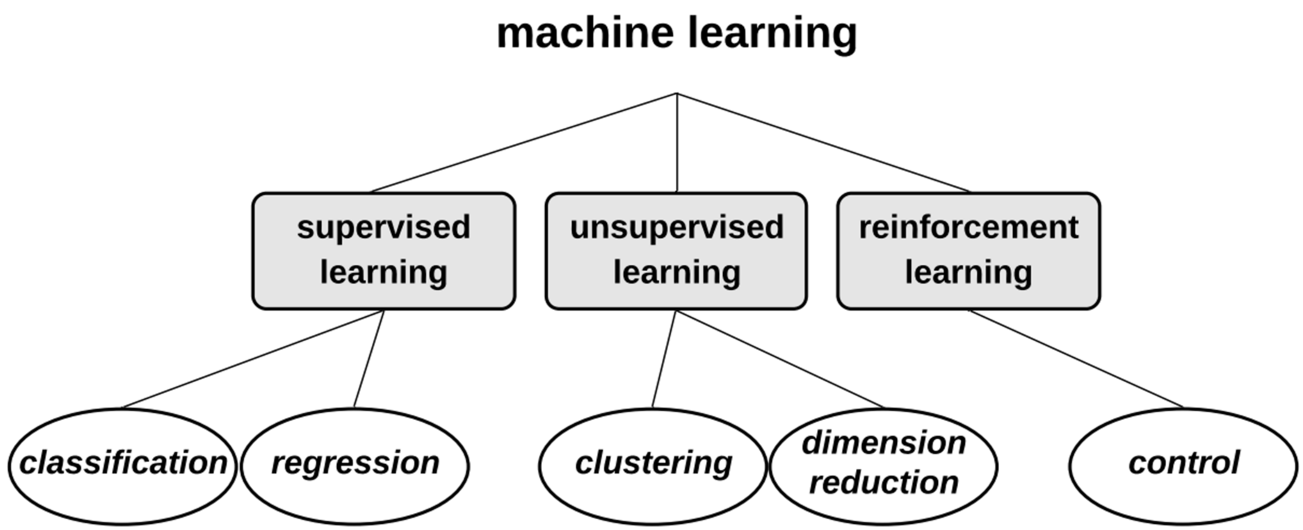
Data-Driven Software Systems



(a) Machine Learning Systems to Infer ML Models

(b) Machine Learning Systems to Infer Decisions

Categories of ML tasks



[1]. Zhang, et al., Machine Learning Testing: Survey, Landscapes and Horizons, 2021