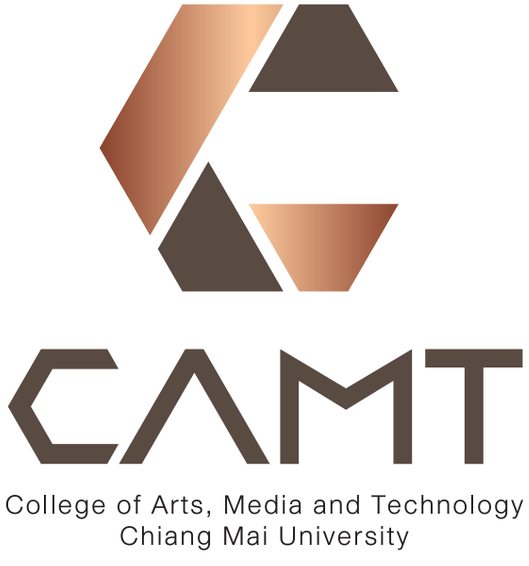
SE 481 Introduction to Information Retrieval

Module #7 — Machine Learning and IR



Passakorn Phannachitta, D.Eng.

passakorn.p@cmu.ac.th

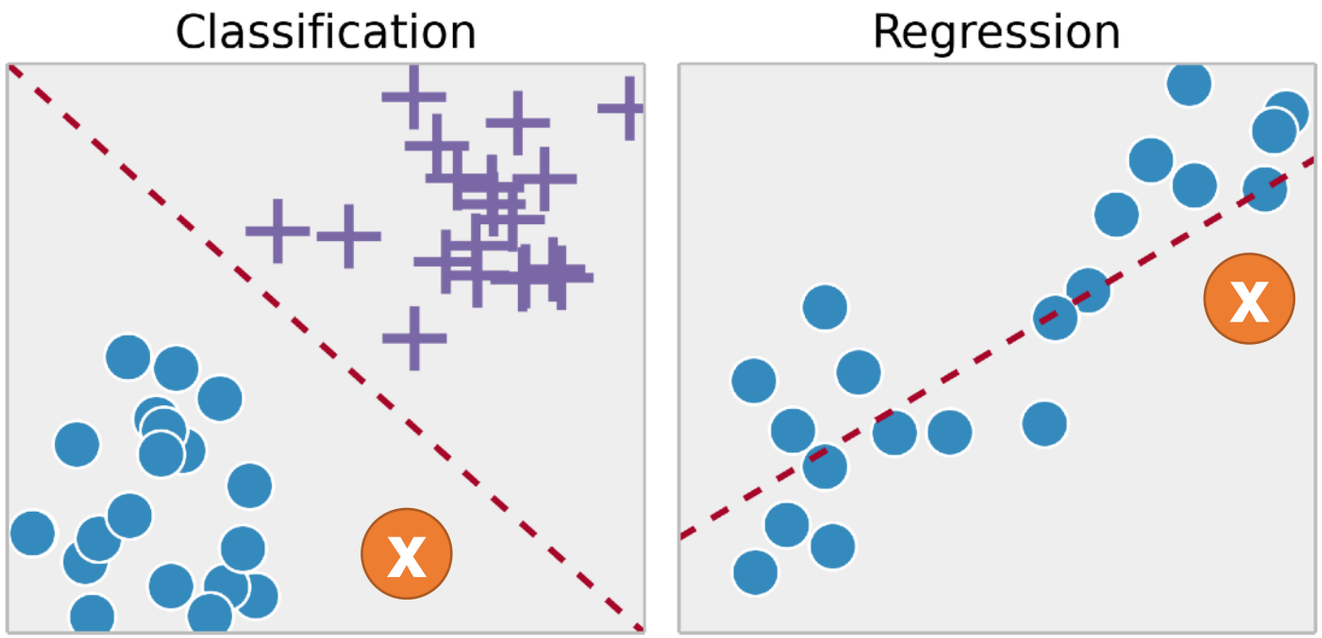
College of Arts, Media and Technology

Chiang Mai University, Chiangmai, Thailand

Agenda

- Basic machine learning algorithms

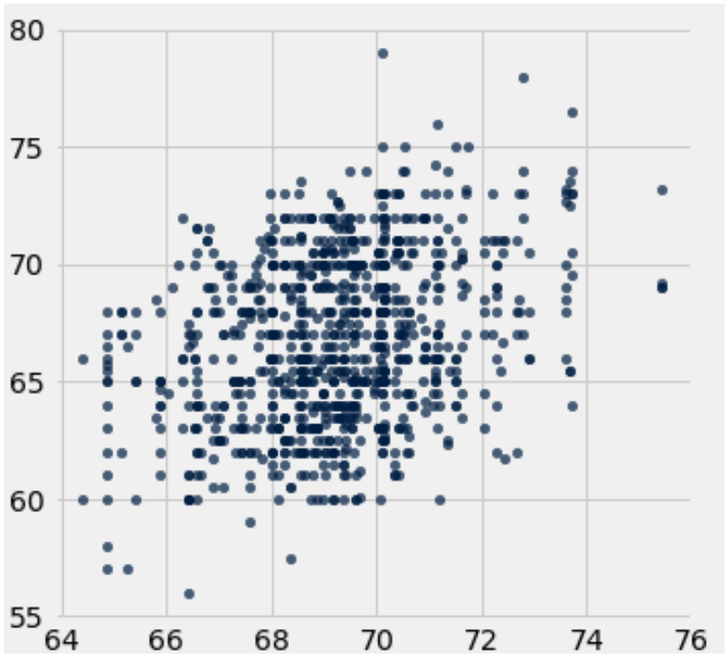
Regression vs Classification

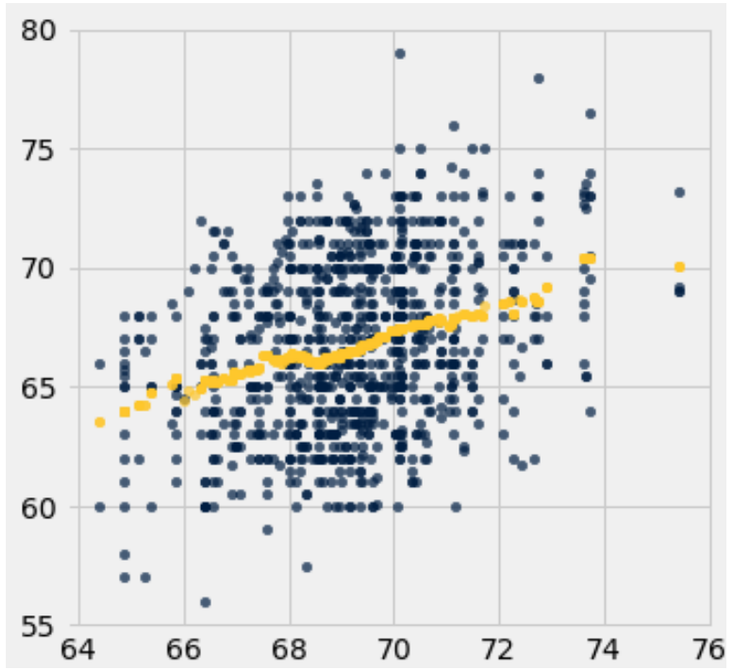


$x \in \{0,+\}$ $x \in (-\infty,\infty)$

In terms of statistics

- Regression is the model of the mean





In terms of statistics

$$\frac{\text{estimate of } y \;-\; \text{average of } y}{\text{SD of } y} \;=\; r \times \frac{\text{the given } x \;-\; \text{average of } x}{\text{SD of } x}$$

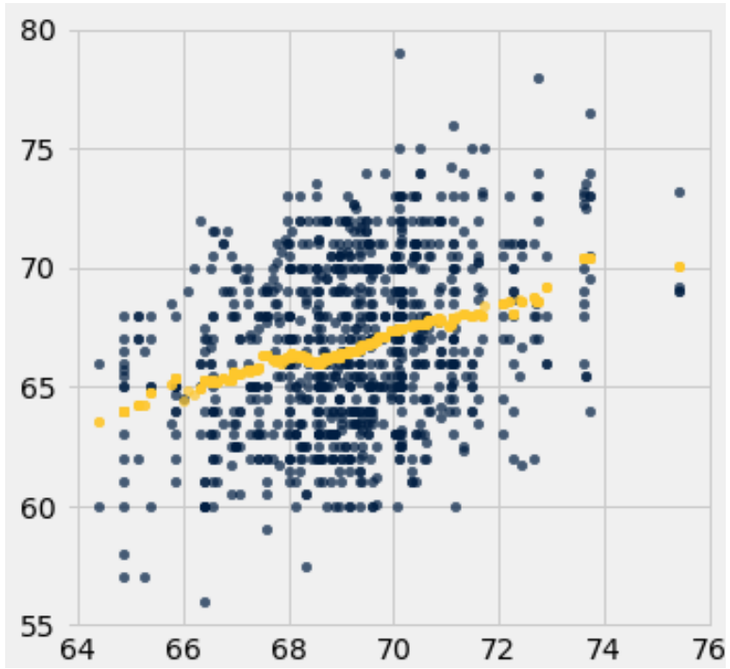
$$\textbf{slope of the regression line} \;=\; r \cdot \frac{\text{SD of } y}{\text{SD of } x}$$

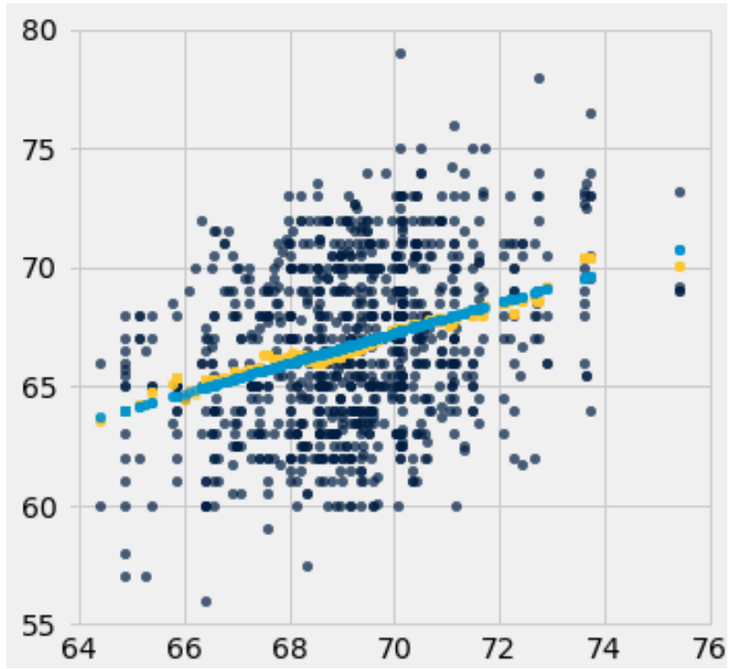
$$\textbf{intercept of the regression line} \;=\; \text{average of } y \;-\; \text{slope} \cdot \text{average of } x$$

Estimate of **y** = slope • (the given **x**) + intercept

In terms of statistics

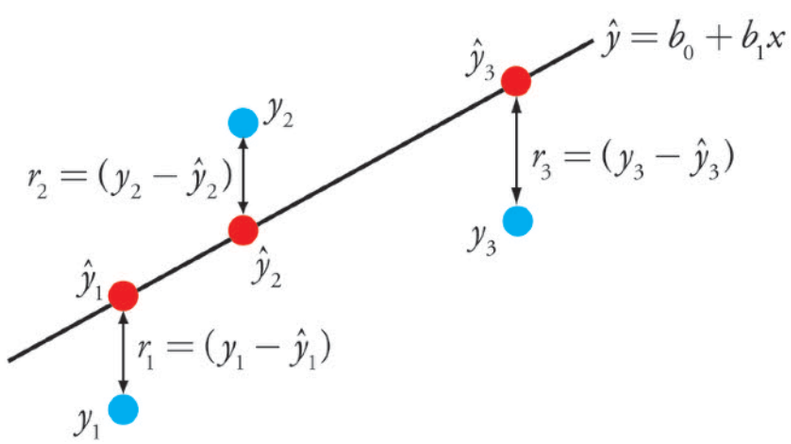
- Regression is the model of the mean





View as an optimization problem

- Ordinary least squares (OLS)
  - Map a straight line, i.e., y = b0 + b1x, to the data
  - **Goal**: to determine the best estimates for b_0 and b_1
  - Having the minimum sum of the difference between each data point and the line



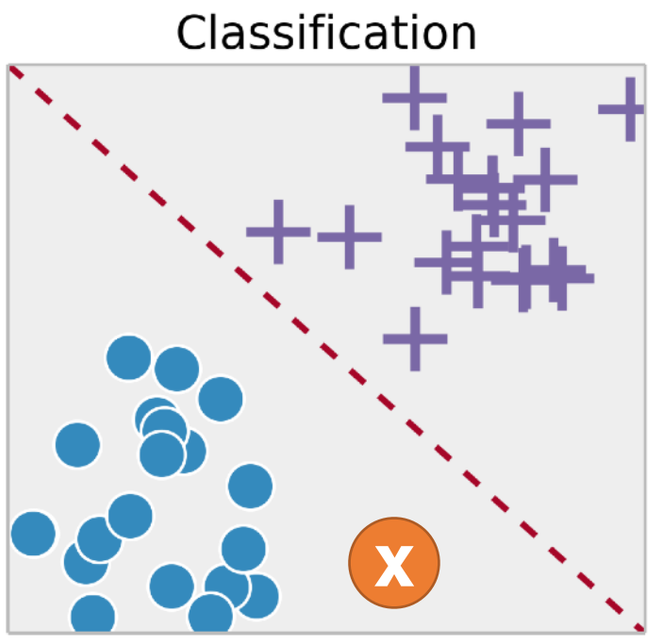
Find b_0 and b_1 that minimize

Mean square error ***(MSE):***

$$MSE = \frac{1}{n}\sum_{I=1}^{n}(y_i - \hat{y}_i)^2$$

Note: Lower MSE = better performance

Classification



$x \in \{o,+\}$

Classification

- Machine learning is a class of techniques for automatically finding patterns in data and using it to draw inferences or make predictions.

- **Classification** is about learning how to make predictions, in particular for a yes/no question, based on what is already happened in past examples.

Example

- For each order an e-commerce website receives, it may would like to predict: **is this order fraudulent?**
- What data/information/insights are available:
  - some about each order (e.g., its total value, whether the order is being shipped to an address this customer has used before, whether the shipping address is the same as the credit card holder's billing address).
  - lots of data on past orders, and they know which of those past orders were fraudulent and which weren't.
- They want to learn patterns that will help them predict, as new orders arrive, whether those new orders are fraudulent.

Example

- For each order an e-commerce website receives, it may would like to predict: **is this order fraudulent?**
- Online dating sites may would like to predict: **are these two people compatible? Will they hit it off?**
- Doctors may would like to know: **does the particular patient has cancer?**
- Politicians would like to predict: **are you going to vote for them?**

Example

- Online dating sites may would like to predict: **are these two people compatible?**
- What data/information/insights are available:
  - These site have lots of data on which matches they've suggested to their customers in the past
  - some idea which ones were successful.
- As new customers sign up, they'd like to make predictions about who might be a good match for them.

Example

- Doctors may would like to know: **does the particular patient has cancer?**
- What data/information/insights are available:
  - They have lots of data on past patients, such as their lab measurements;
  - Whether the particular patient ultimately developed cancer, and from that, the doctors would like to try to infer what measurements tend to be characteristic of cancer (or non-cancer), so they can diagnose future patients accurately.

Example

- Politicians would like to predict: **are you going to vote for them?**
- What data/information/insights are available:
  - Public databases and commercial databases have a lot of information about most people: e.g., whether they own a home or rent; their interests and hobbies; their shopping habits; and so on.
  - Political campaigns have surveyed some voters and found out who they plan to vote for, so they have some examples where the correct answer is known.
- From this data, the campaigns would like to find patterns that will help them make predictions about all other potential voters.

In classification

- Each individual or situation where we'd like to make a prediction is called an observation
  - we ordinarily have many observations.
  - each observation has multiple attributes, which are known (for example, the total value of the order on Amazon, or the voter's annual salary).
- Also, each observation has a **class**, which is the answer to the question we care about (for example, fraudulent or not, or voting for you or not).
- We try to predict the class using the available attributes.

Ref: https://archive.ics.uci.edu/ml/datasets/banknote+authentication

Starting from a simple approach

- Suppose that we want to predicting whether a banknote is counterfeit or legitimate.
  - Based on photographs of many individual banknotes: some counterfeit, some legitimate, researchers found that using only wavelets is sufficiently enough to determine whether the banknote is counterfeit or legitimate.
  - The wavelet features are obtained from applying wavelet transformation on grayscale banknote images
  - Three types of wavelets are used along with the image entropy.

Ref: https://archive.ics.uci.edu/ml/datasets/banknote+authentication

Banknote authentication dataset

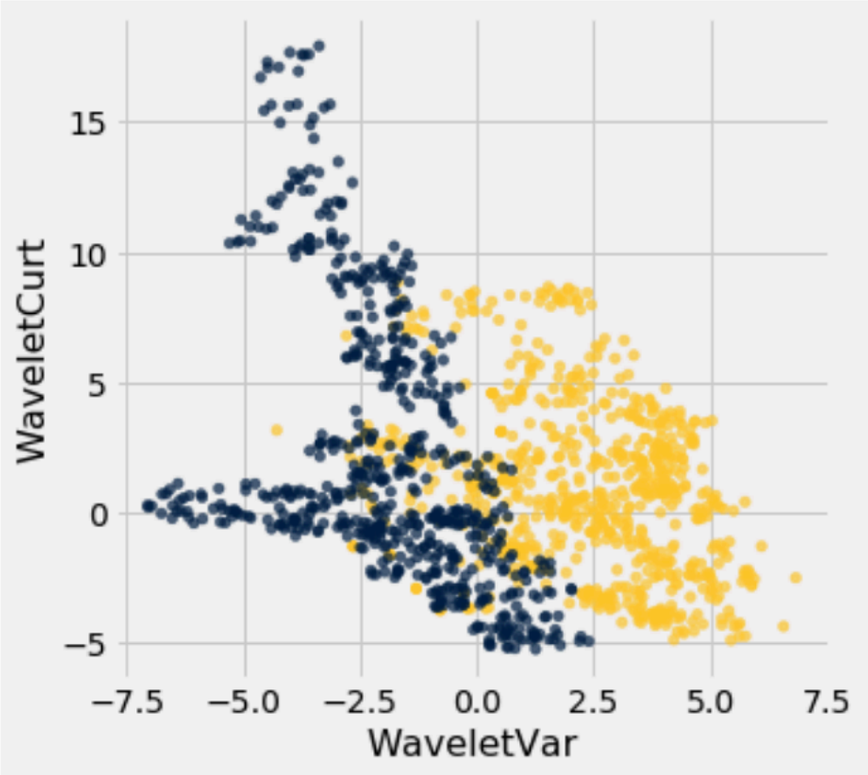
- A UCI dataset
  - Owner: Volker Lohweg (University of Applied Sciences, Ostwestfalen-Lippe)
  - Available since April 2013

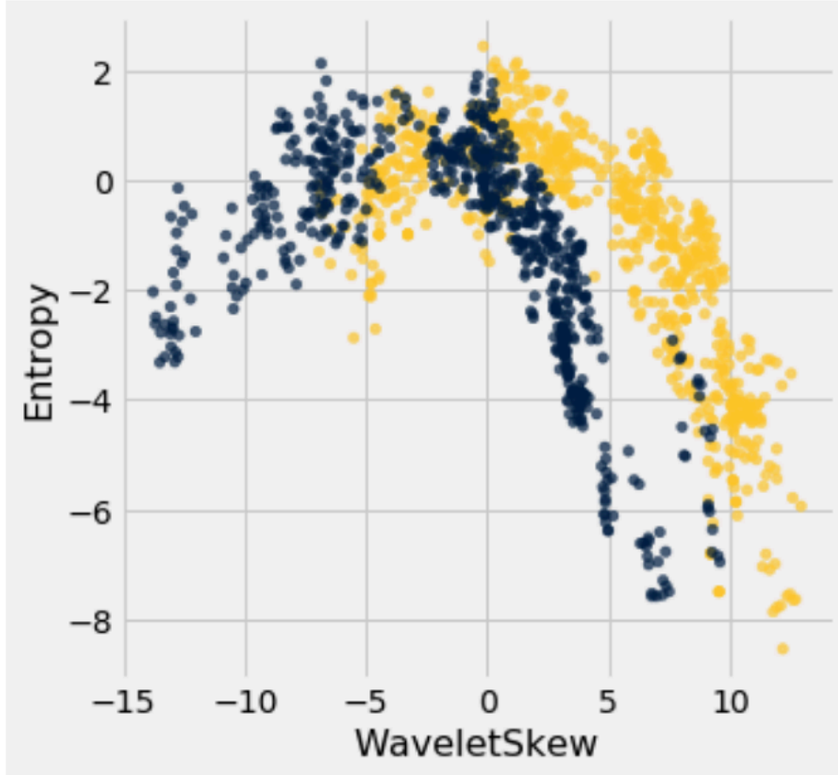
WaveletVar	WaveletSkew	WaveletCurt	Entropy	Class
3.6216	8.6661	-2.8073	-0.44699	0
4.5459	8.1674	-2.4586	-1.4621	0
3.866	-2.6383	1.9242	0.10645	0
3.4566	9.5228	-4.0112	-3.5944	0
0.32924	-4.4552	4.5718	-0.9888	0
4.3684	9.6718	-3.9606	-3.1625	0
3.5912	3.0129	0.72888	0.56421	0
2.0922	-6.81	8.4636	-0.60216	0
3.2032	5.7588	-0.75345	-0.61251	0
1.5356	9.1772	-2.2718	-0.73535	0

... (1362 rows omitted)

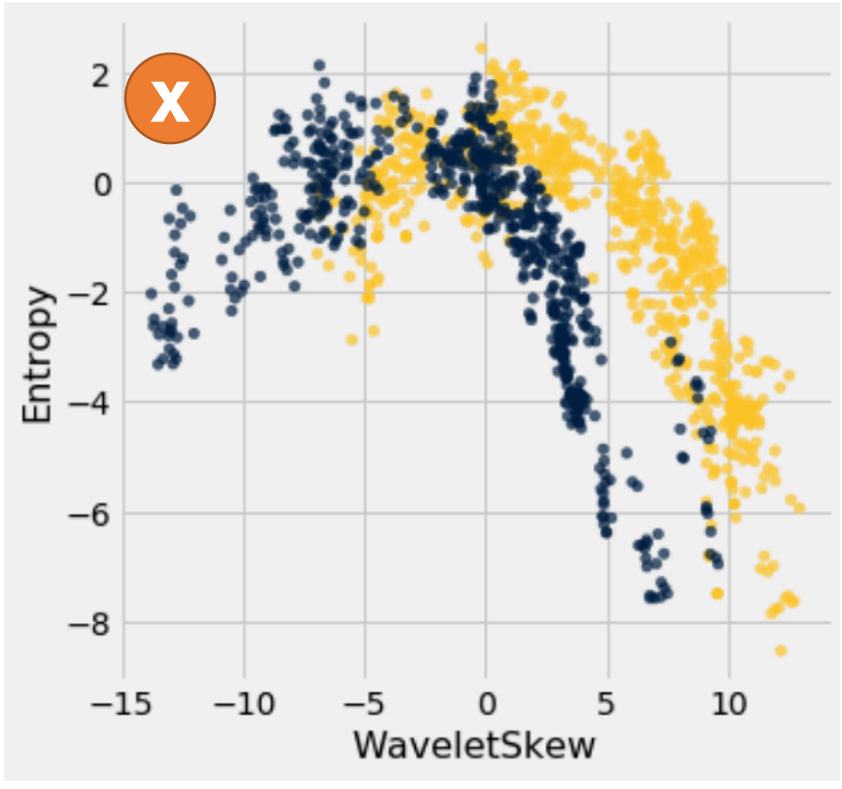
- Var is variance
- Skew is skewness
- Curt is curtosis

Some pairs of features

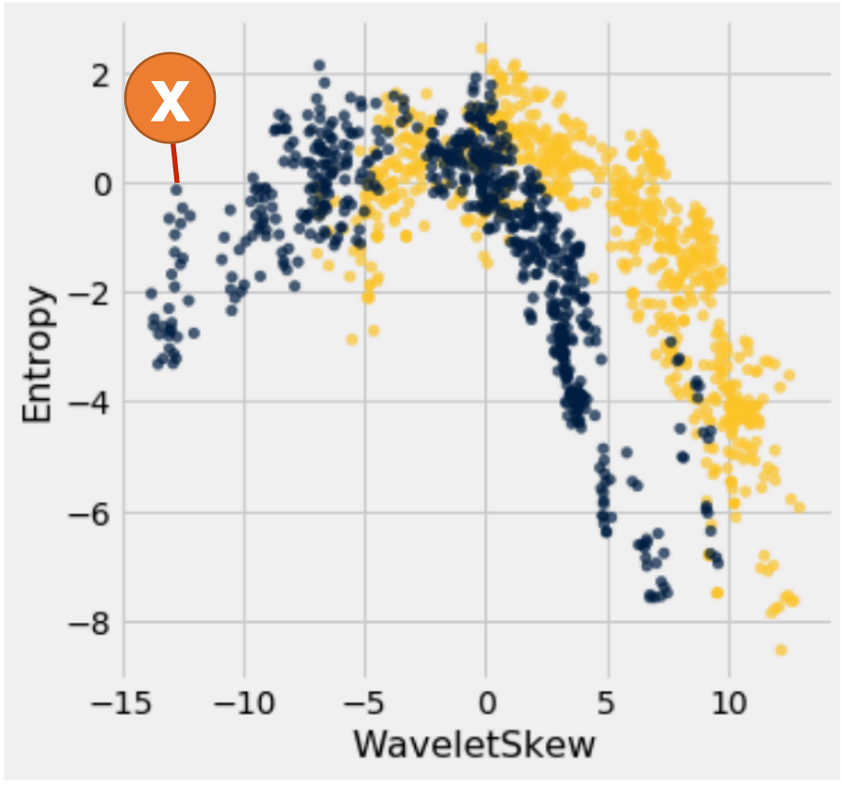




K Nearest Neighbour

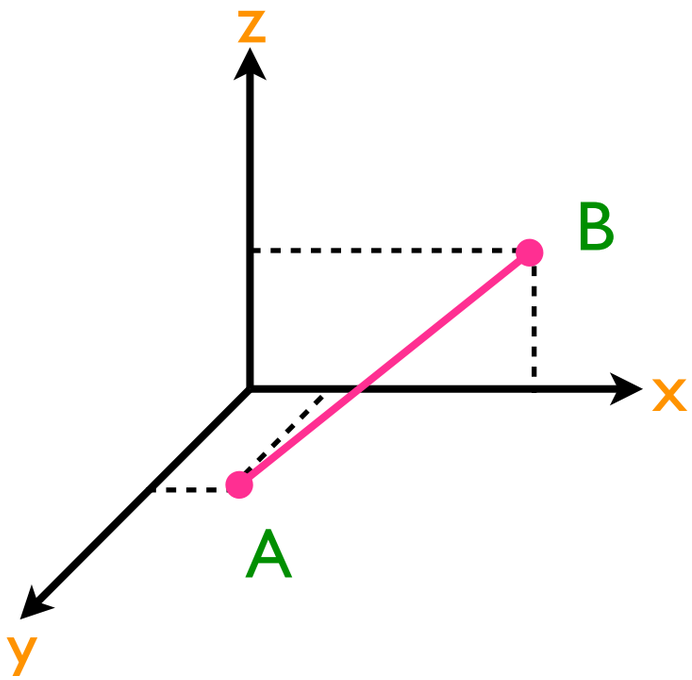


K Nearest Neighbour



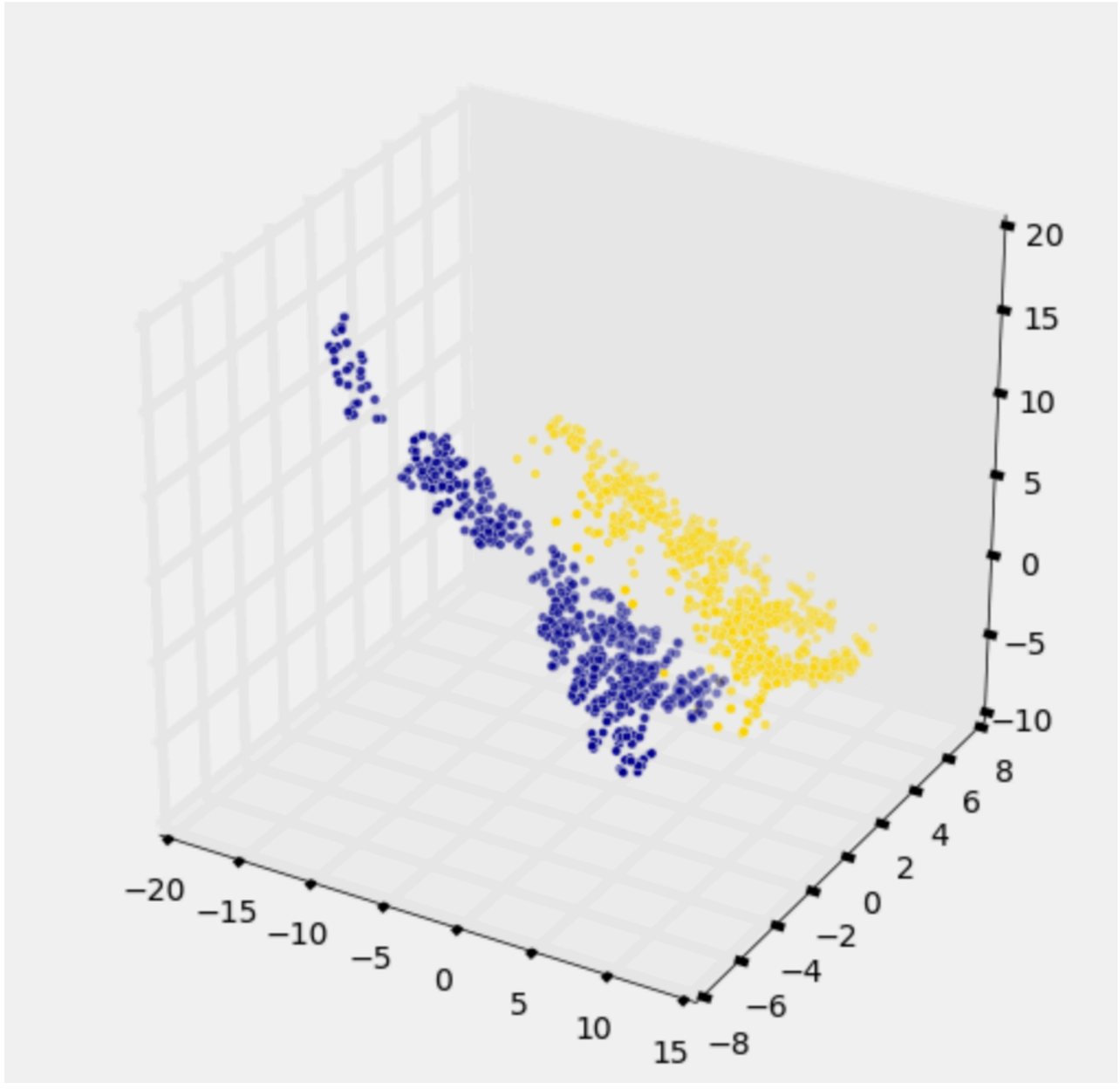
$$D = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}.$$

K Nearest Neighbour



$$D(A,B) = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}$$

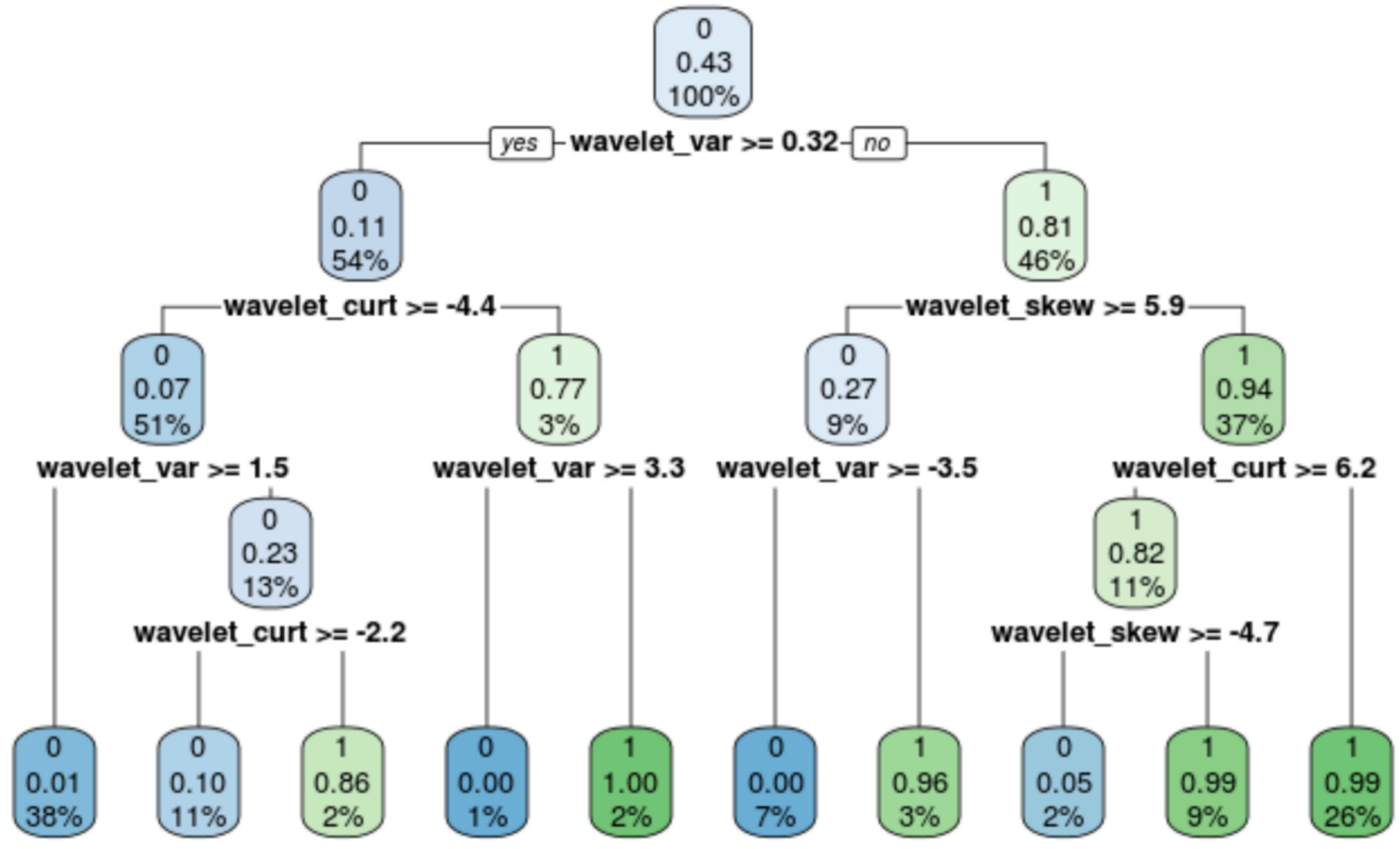
K Nearest Neighbour



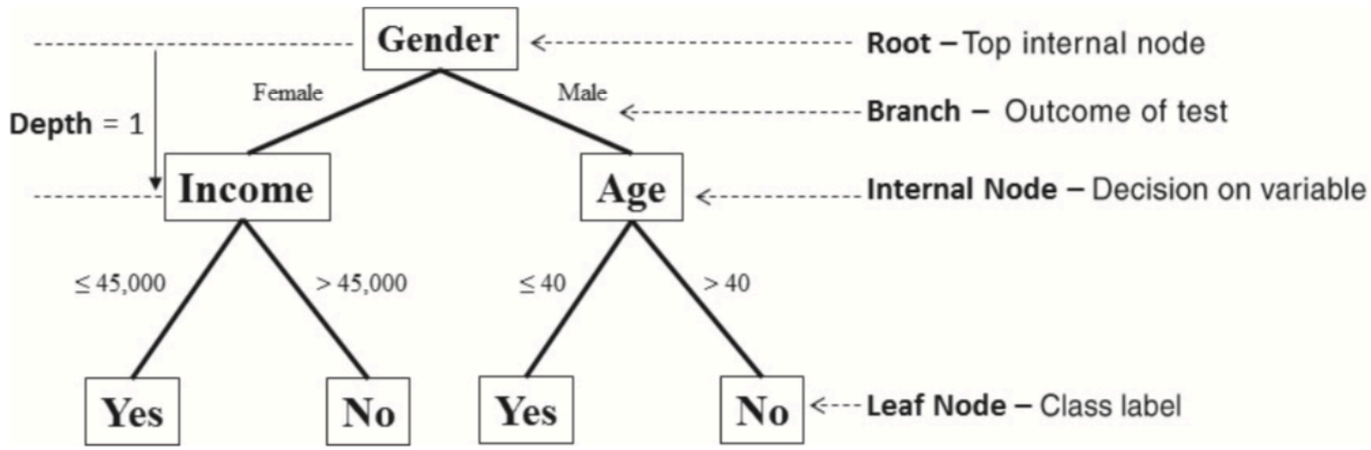
- 3 variables
  - WaveletVar
  - WaveletSkew
  - WaveletCurt

Using other classifier

- Decision tree



Decision tree



- In general, the objective of a decision tree algorithm is to construct a tree T from a training set S. If all the records in S belong to some class C (legitimate = yes, for example), or if S is sufficiently pure (greater than a preset threshold), then that node is considered a leaf node and assigned the label C.

Decision tree — more example

- A UCI dataset
  - Available since February 2013

	job	marital	education	default	housing	loan	contact	poutcome	subscribed
1	management	single	tertiary	no	yes	no	cellular	unknown	no
2	entrepreneur	married	tertiary	no	yes	yes	cellular	unknown	no
3	services	divorced	secondary	no	no	no	cellular	unknown	yes
4	management	married	tertiary	no	yes	no	cellular	unknown	no
5	management	married	secondary	no	yes	no	unknown	unknown	no
6	management	single	tertiary	no	yes	no	unknown	unknown	no
7	entrepreneur	married	tertiary	no	yes	no	cellular	failure	yes
8	admin.	married	secondary	no	no	no	cellular	unknown	no
9	blue-collar	married	secondary	no	yes	no	cellular	other	no
10	management	married	tertiary	yes	no	no	cellular	unknown	no
11	blue-collar	married	secondary	no	yes	no	cellular	unknown	no
12	management	divorced	secondary	no	no	no	unknown	unknown	no
13	blue-collar	married	secondary	no	yes	no	cellular	unknown	no
14	retired	married	secondary	no	no	no	cellular	unknown	no
15	management	single	tertiary	no	yes	no	cellular	unknown	no
16	retired	married	secondary	yes	yes	no	cellular	unknown	no
17	unemployed	married	secondary	no	yes	no	telephone	unknown	no
18	management	divorced	tertiary	no	yes	no	cellular	unknown	no
19	management	married	tertiary	no	yes	no	cellular	unknown	no
20	blue-collar	married	secondary	no	yes	no	unknown	unknown	no
21	management	divorced	tertiary	no	yes	yes	cellular	failure	yes
22	blue-collar	divorced	secondary	no	yes	no	cellular	failure	no
23	blue-collar	single	secondary	no	yes	no	cellular	failure	no
24	admin.	single	secondary	no	no	no	unknown	unknown	no
25	blue-collar	married	secondary	no	yes	no	cellular	failure	no
26	blue-collar	single	secondary	no	yes	no	unknown	unknown	no
27	housemaid	married	secondary	no	no	no	cellular	unknown	no
28	technician	married	tertiary	no	no	no	cellular	unknown	no

Ref: https://archive.ics.uci.edu/ml/datasets/Bank+Marketing

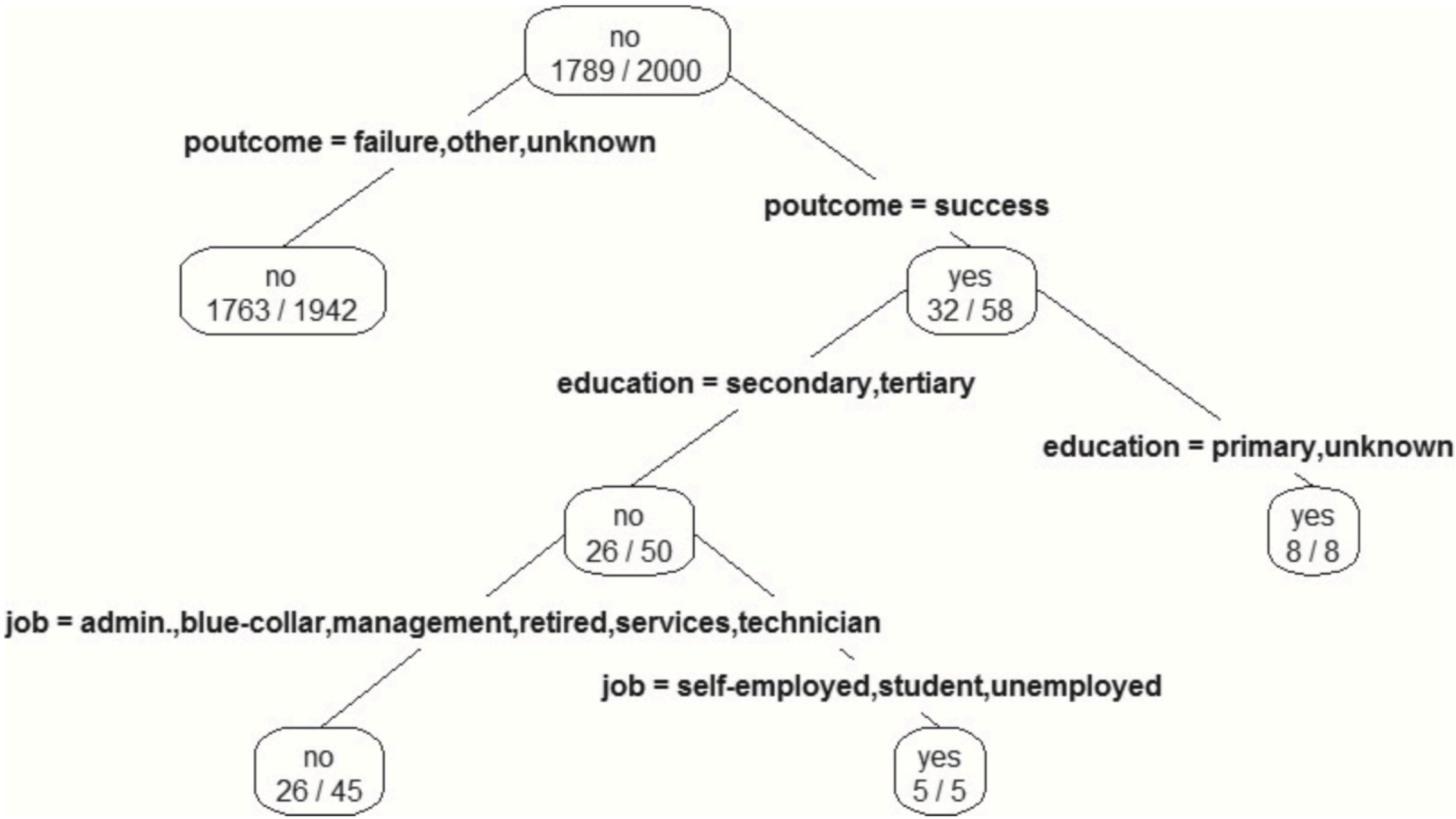
Dataset summarization

```
          job            marital               education    default
blue-collar:435   divorced: 228   primary  : 335   no :1961
management :423   married :1201   secondary:1010   yes:  39
technician :339   single  : 571   tertiary : 564
admin.     :235                   unknown  :  91
services   :168
retired    : 92
(Other)    :308

 housing      loan              contact           month          poutcome
no : 916   no :1717   cellular :1287   may     :581   failure: 210
yes:1084   yes: 283   telephone: 136   jul     :340   other  :  79
                      unknown  : 577   aug     :278   success:  58
                                       jun     :232   unknown:1653
                                       nov     :183
                                       apr     :118
                                       (Other):268

 subscribed
no :1789
yes: 211
```

Decision tree

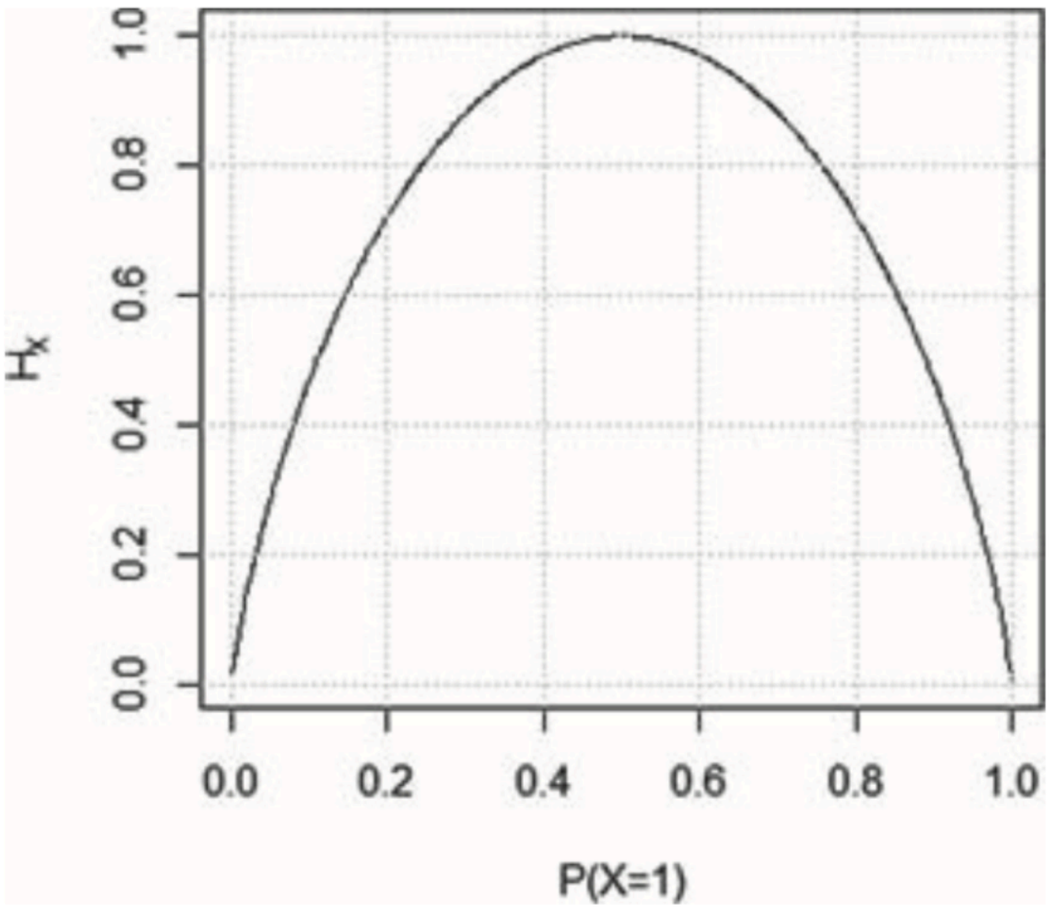


Decision tree

- At each split, the decision tree algorithm picks the most informative attribute out of the remaining attributes. The extent to which an attribute is informative is determined by measures such as entropy and information gain.
  - Entropy measures the impurity of an attribute
  - Information gain measures the purity of an attribute
- Given a classX and its label x∈X, letP(x) be the probability of x. H_X, the entropy of X, is defined as

$$H_X = -\sum_{\forall x \in X} P(x) \log_2 P(x)$$

Decision tree



Decision tree

- The information gain of an attribute A is defined as the difference between the base entropy and the conditional entropy of the attribute:

$$InfoGain_A = H_S - H_{S|A}$$

	Cellular	Telephone	Unknown
P(contact)	0.6435	0.0680	0.2885
P(subscribed=yes | contact)	0.1399	0.0809	0.0347
P(subscribed=no | contact)	0.8601	0.9192	0.9653

$$H_{Y|X} = \sum_x P(x) H(Y|X=x)$$

$$= -\sum_{\forall x \in x} P(x) \sum_{\forall y \in Y} P(y|x) \log_2 P(y|x)$$

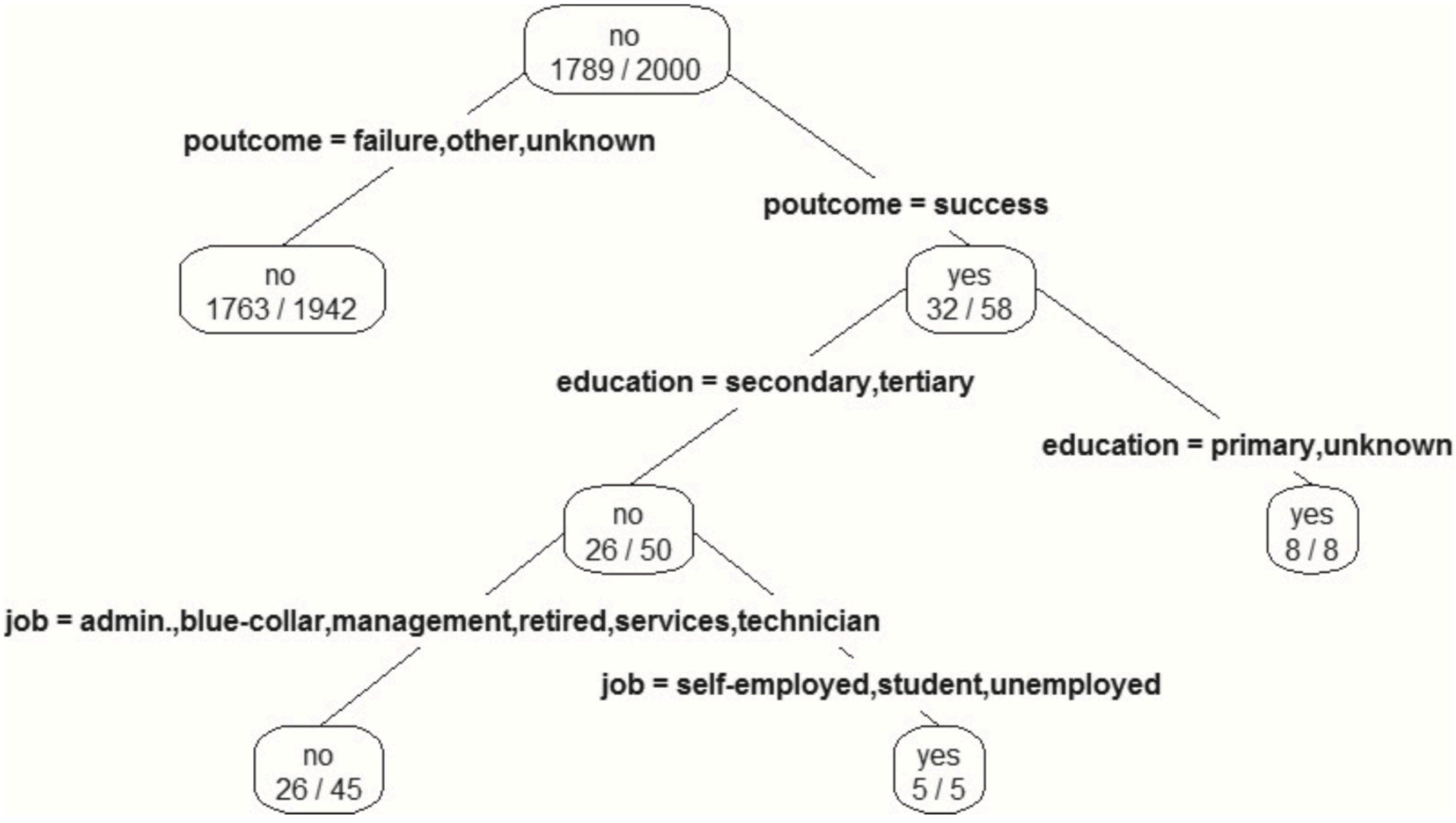
$$\begin{aligned} H_{subscribed|contact} = -\big[&0.6435 \cdot \left(0.1399 \cdot \log_2 0.1399 + 0.8601 \cdot \log_2 0.8601\right) \\ &+ 0.0680 \cdot \left(0.0809 \cdot \log_2 0.0809 + 0.9192 \cdot \log_2 0.9192\right) \\ &+ 0.2885 \cdot \left(0.0347 \cdot \log_2 0.0347 + 0.9653 \cdot \log_2 0.9653\right)\big] \\ &= 0.4661 \end{aligned}$$

Decision tree

- Information gain compares the degree of purity of the parent node before a split with the degree of purity of the child node after a split.
- At each split, an attribute with the greatest information gain is considered the most informative attribute.

Attribute	Information Gain
poutcome	0.0289
contact	0.0201
housing	0.0133
job	0.0101
education	0.0034
marital	0.0018
loan	0.0010
default	0.0005

Decision tree



Decision tree

- In practice we usually limits the number of splits to create a short tree.
- Creating many short trees are often used as components (also called weak learners or base learners) in ensemble methods.

Naïve Bayes

- Naïve Bayes is a probabilistic classification method based on Bayes' theorem
- A naïve Bayes classifier assumes that the presence or absence of a particular feature of a class is unrelated to the presence or absence of other features.
- The conditional probability of event C occurring, given that event A has already occurred, is denoted as P(C|A)

$$P(C|A) = \frac{P(A|C) \cdot P(C)}{P(A)}$$

where C is the class label C ∈ $\{c_1, c_2, \ldots c_n\}$ and A is the observed attributes A=$\{a_1, a_2, \ldots a_m\}$

$$P(A) = P(A \cap C) + P(A \cap \neg C)$$

$$P(c_i|A) = \frac{P(a_1, a_2, \ldots, a_m|c_i) \cdot P(c_i)}{P(a_1, a_2, \ldots, a_m)}, i = 1,2,\ldots n$$

Naïve Bayes

- A Naïve Bayes classifier was created by simplifying Bayes' theorem,
- $$P(a_1, a_2, \ldots, a_m | c_i) = P(a_1|c_i)P(a_2|c_i)\cdots P(a_m|c_i) = \prod_{j=1}^{m} P(a_j|c_i)$$
- E.g., c related to $p_{marital}$ are

$$P(single \mid subscribed = yes) \approx 0.35$$

$$P(married \mid subscribed = yes) \approx 0.53$$

$$P(divorced \mid subscribed = yes) \approx 0.12$$

$$P(single \mid subscribed = no) \approx 0.28$$

$$P(married \mid subscribed = no) \approx 0.61$$

$$P(divorced \mid subscribed = no) \approx 0.11$$

Naïve Bayes

- Naïve Bayes is a probabilistic classification method based on Bayes' theorem
- A naïve Bayes classifier assumes that the presence or absence of a particular feature of a class is unrelated to the presence or absence of other features.
- The conditional probability of event C occurring, given that event A has already occurred, is denoted as P(C|A)

$$P(C|A) = \frac{P(A|C) \cdot P(C)}{P(A)}$$

where C is the class label C ∈ $\{c_1, c_2, \ldots c_n\}$ and A is the observed attributes A=$\{a_1, a_2, \ldots a_m\}$

$$P(A) = P(A \cap C) + P(A \cap \neg C)$$

$$P(c_i|A) = \frac{P(a_1, a_2, \ldots, a_m|c_i) \cdot P(c_i)}{P(a_1, a_2, \ldots, a_m)}, i = 1,2,\ldots n$$

Naïve Bayes

- To estimate a new case, e.g.,

Job	Marital	Education	Default	Housing	Loan	Contact	Poutcome
management	married	secondary	no	yes	no	cellular	Success

Naïve Bayes

- We first compute the conditional probabilities for the new case.

j	a_j	$P(a_j \mid \text{subscribed} = \text{yes})$	$P(a_j \mid \text{subscribed} = \text{no})$
1	job = management	0.22	0.21
2	marital = married	0.53	0.61
3	education = secondary	0.46	0.51
4	default = no	0.99	0.98
5	housing = yes	0.35	0.57
6	loan = no	0.90	0.85
7	contact = cellular	0.85	0.62
8	poutcome = success	0.15	0.01

Naïve Bayes

Job	Marital	Education	Default	Housing	Loan	Contact	Poutcome
management	married	secondary	no	yes	no	cellular	Success

- For A = {management, married, secondary, no, yes, no, cellular, success},

$$P(yes|A) \propto 0.11 \cdot (0.22 \cdot 0.53 \cdot 0.46 \cdot 0.99 \cdot 0.35 \cdot 0.90 \cdot 0.85 \cdot 0.15) \approx 0.00023$$

$$P(no|A) \propto 0.89 \cdot (0.21 \cdot 0.61 \cdot 0.51 \cdot 0.98 \cdot 0.57 \cdot 0.85 \cdot 0.62 \cdot 0.01) \approx 0.00017$$

- Then, this Naïve Bayes will answer as yes

Coding example

```
01  dataset = pd.read_csv('src/resource/bank/bank-full.csv',delimiter=';')
02  dataset = dataset.drop(['age', 'balance', 'day', 'month', 'pdays', 'duration', 'campaign',
'previous'], axis=1)
03  le = {}
04
05  for col in dataset:
06      if dataset[col].dtype == 'object':
07          le[col] = preprocessing.LabelEncoder()
08          dataset[col] = le[col].fit_transform(dataset[col])
09
10  unknown_case = pd.DataFrame.from_dict({'job':'management', 'marital':'married',
'education':'secondary', 'default': 'no', 'housing': 'yes', 'loan': 'no', 'contact': 'cellular',
'poutcome':'success'}, orient='index').T
11
12  for col in unknown_case:
13      if unknown_case[col].dtype == 'object':
14          unknown_case[col] = le[col].transform(unknown_case[col])
15
16  X = dataset.drop(['y'],axis=1)
17  y = dataset['y']
```

Read the dataset

Select only some features

Encode categorical variables

Create a new case to predict

Encode the new case

Identify X and y

Coding example — KNN

```
1   knn_model = neighbors.KNeighborsClassifier(n_neighbors=5)
2   knn_model.fit(X,y)
3   knn_predict = knn_model.predict(unknown_case)
4   print(knn_predict)
```

Construct a Knn model

Build the model

Build the model

Coding example — Decision tree

Construct a decision tree model

```
1   dt_model = tree.DecisionTreeClassifier()
2   dt_model.fit(X,y)
3   dt_predict = dt_model.predict(unknown_case)
4   print(dt_predict)
```

Coding example — Naive bayes

Construct a naive bayes model

```
1   nb_model = naive_bayes.CategoricalNB()
2   nb_model.fit(X,y)
3   nb_predict = nb_model.predict(unknown_case)
4   print(nb_predict)
```

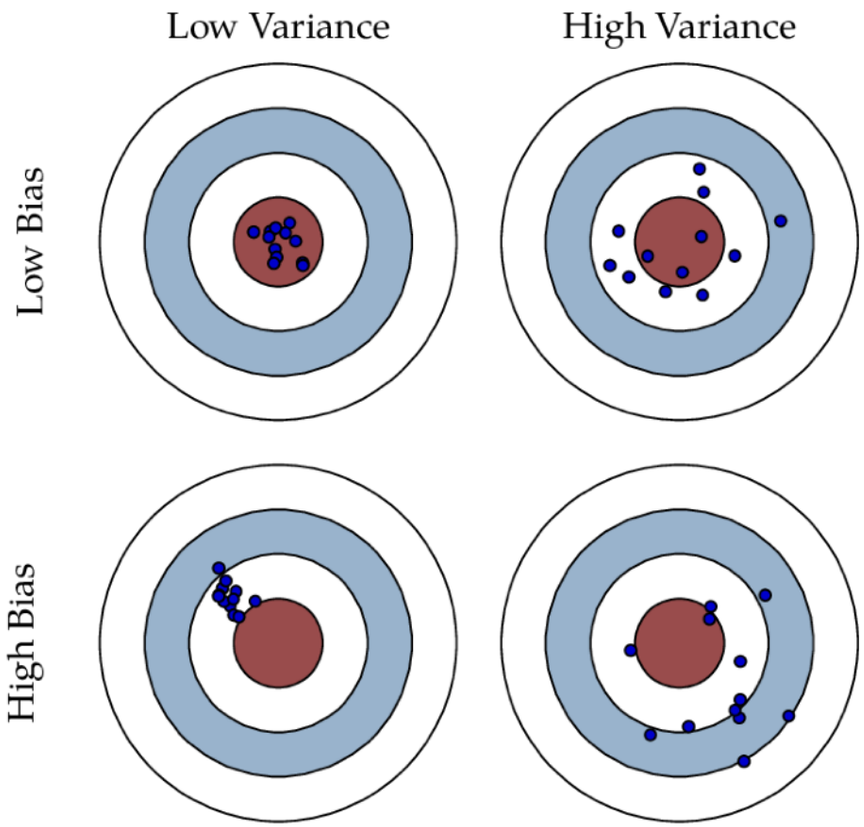
Error(X) = noise(X) + bias(X) + variance(X)

- **Bias** is the algorithm's tendency to consistently learn the wrong thing by not taking into account all the information in the data **(underfitting)**.
- **Variance** is the algorithm's tendency to learn random things irrespective of the real signal by fitting highly flexible models that follow the error/noise in the data too closely **(overfitting)**.

Simplified explanation, e.g.,

- A person with high bias is someone who starts to answer before you can even finish asking.
- A person with high variance is someone who can think of all sorts of crazy answers.
- Combining these gives you different personalities

Variance and accuracy



Ref: https://i1.wp.com/www.d4t4v1z.com/wp-content/uploads/2017/09/bias.png?resize=835%2C770

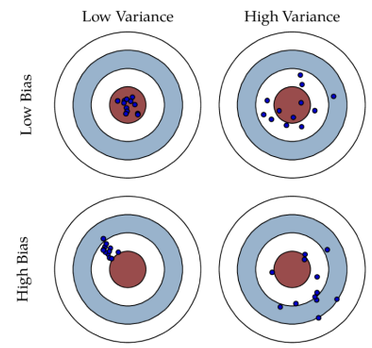
Simplified explanation, e.g.,

- **High bias/low variance:** this is someone who usually gives you the same answer, no matter what you ask, and is usually wrong about it;
- **High bias/high variance:** someone who takes wild guesses, all of which are sort of wrong;
- **Low bias/high variance:** a person who listens to you and tries to answer the best they can, but that daydreams a lot and may say something totally crazy;
- **Low bias/low variance:** a person who listens to you very carefully and gives you good answers pretty much all the time.

Ensemble

- Combine multiple predictors
- Reduce risks of inaccurate prediction due to one technique may be most suitable with only one particular type of data
- Generally classified into
  - Combining prediction results of one type of model over difference data splits — Random forest
  - Combining prediction results of many based regressors — Stacking, a.k.a., stack generalization

In theory



- MSE of one single model is due to bias and variance
- It can be shown that if we combine each predicted values by using the mean, over all MSE will be reduced to

$$MSE = \overline{Bias^2} + \frac{1}{m}\overline{Var} + \left(1 - \frac{1}{m}\right)\overline{Covar}$$

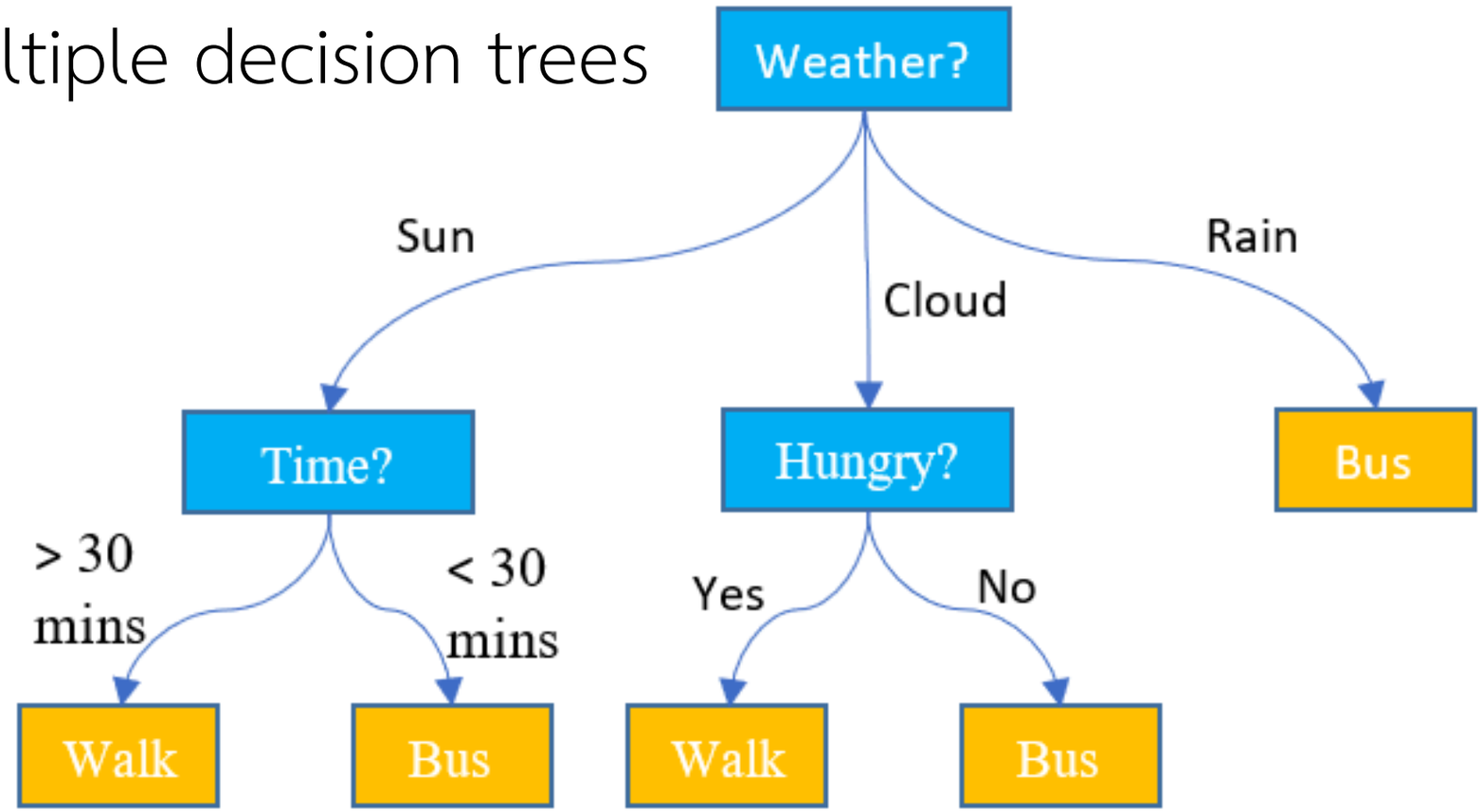
- We will no longer need to care much about variance of one single model if we can ensure that each of the selected solo models are independent and unbiased.

Bootstrap aggregation (Bagging)

- Randomly generating data subsets from the original training datasets, then applying a model, e.g., decision trees to all the instance subsets
- The final estimated value is the mean value aggregated from all the estimated values produced
- Potentially reduce the error due to **variance**

Random Forest (RF)

- Ensemble of multiple decision trees



- Both data entries and features are subsetted

Adaptive boosting (AdaBoost)

- First builds a model, e.g., regression tree over the entire training dataset.
- Then, iteratively build a weaker regressor focusing on more difficult sub tasks and adds them to the stronger estimator built in the earlier iterations.
- The method used in building weak learners is based on assigning weights on the less accurately estimated instances higher than those of accurately estimated.
- In the succeeding iteration, the model will focus more on the instances with high weight values.
- Generally offer better accuracy as compared with Bagging
- May amplify noise in noisy dataset

Gradient boosting regressor (GBR)

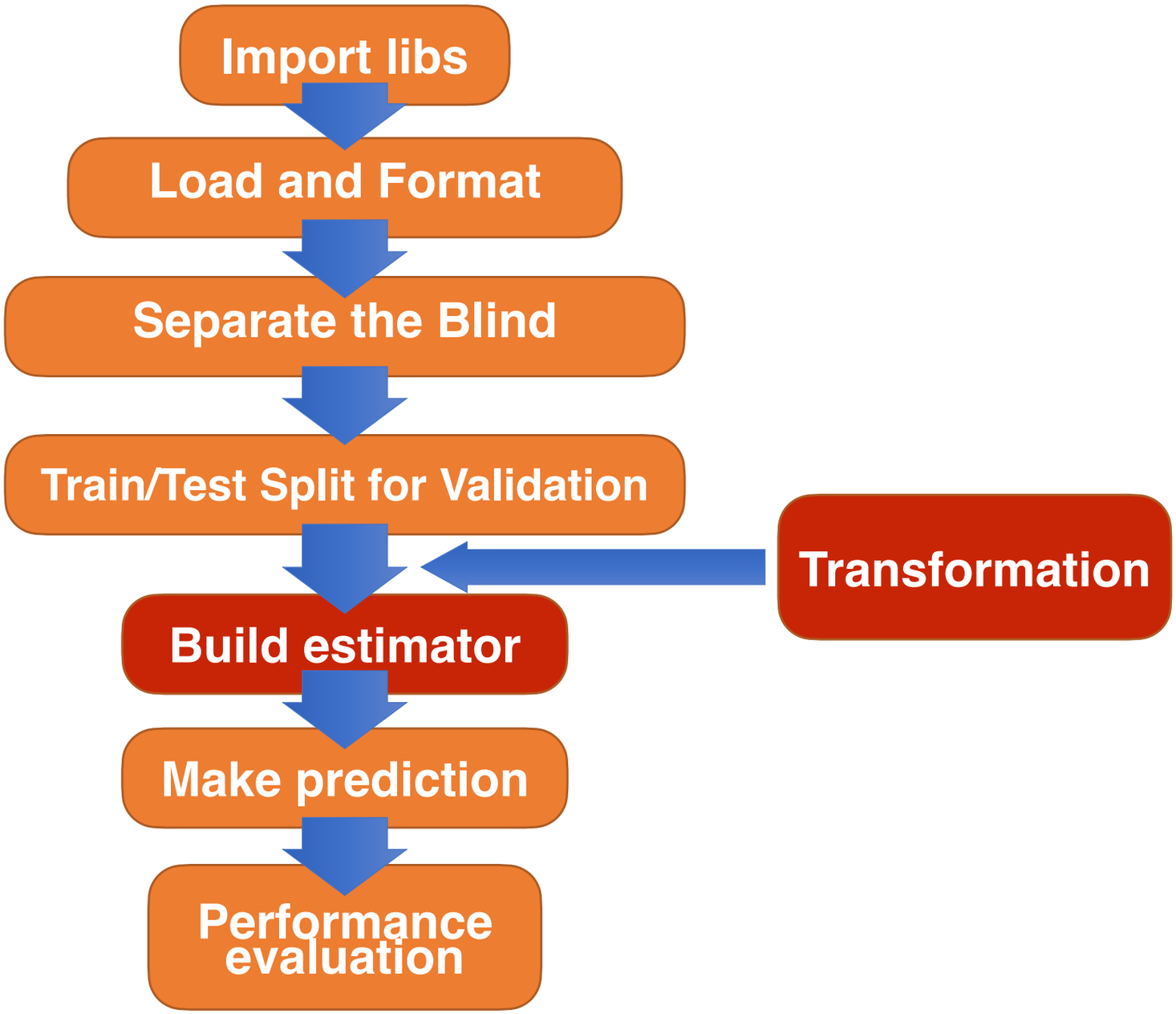
- “If linear regression was a Toyota Camry, then gradient boosting would be a UH-60 Blackhawk Helicopter.”

http://blog.kaggle.com/2017/01/23/a-kaggle-master-explains-gradient-boosting/

Gradient boosting regressor (GBR)

- Combining numerous performance factors of many techniques into one single technique
- Utilize regression tree as the based model
- Fine tune the model by adding “error-correcting” mechanism to the based tree model
- Avoid overfitting using regularization — utilizing both shrinkage and gradient descent
- Being an ensemble technique

Common model building steps

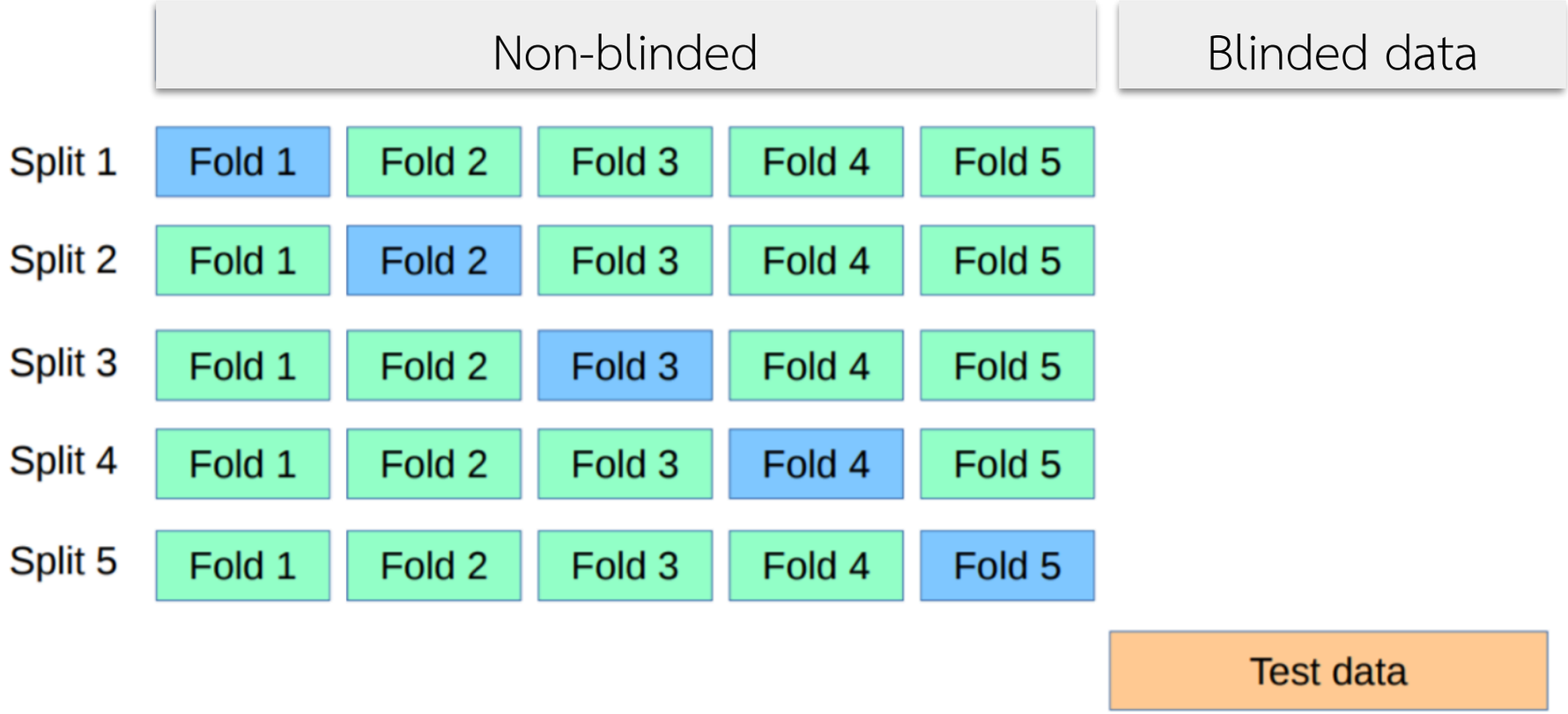


Separate the blind samples

Non-blinded data	Blinded data
eg. 90% of the entries	eg. 10% of the entries

Common model building steps

- Cross validation, e.g., 5-fold cross validation



Ref: https://scikit-learn.org/stable/modules/cross_validation.html

Common model building steps

- Pipeline

Ref: https://www.slideshare.net/PyData/julie-michelman-pandas-pipelines-and-custom-transformers

Evaluation — Some common terms

- **True Positives (TP)** — number of predicted positive in the actual positive group
- **True Negatives (TN)** — number of predicted negative in the actual negative group
- **False Positives (FP)** — number of predicted positive in the actual negative group
- **False Negatives (FN)** — number of predicted negative in the actual positive group

Some common terms -- Accuracy

- Accuracy = $$\frac{TP + TN}{TP + TN + FP + FN}$$

- Example:

	Predicted Positive	Predicted Negative
Actual Positive	10 (TP)	15 (FN)
Actual Negative	25 (FP)	100 (TN)

- Accuracy = (10 + 100) / (10 + 100 + 25 + 15) = 0.73

Some common terms -- Precision

- ratio between the correct prediction and total case of predicting true.
- **Precision =** $\frac{TP}{TP + FP}$
- **Example:**

	Predicted Positive	Predicted Negative
Actual Positive	10 (TP)	15 (FN)
Actual Negative	25 (FP)	100 (TN)

- Precision = (10) / (10 +25) = 0.29 .. very low

Some common terms -- Recall

- ratio between the correct prediction and total correct case.
- Recall = $\frac{TP}{TP + FN}$
- Example:

	Predicted Positive	Predicted Negative
Actual Positive	10 (TP)	15 (FN)
Actual Negative	25 (FP)	100 (TN)

- Precision = (10) / (10 +15) = 0.4 .. still low

CV example

Split blind test samples

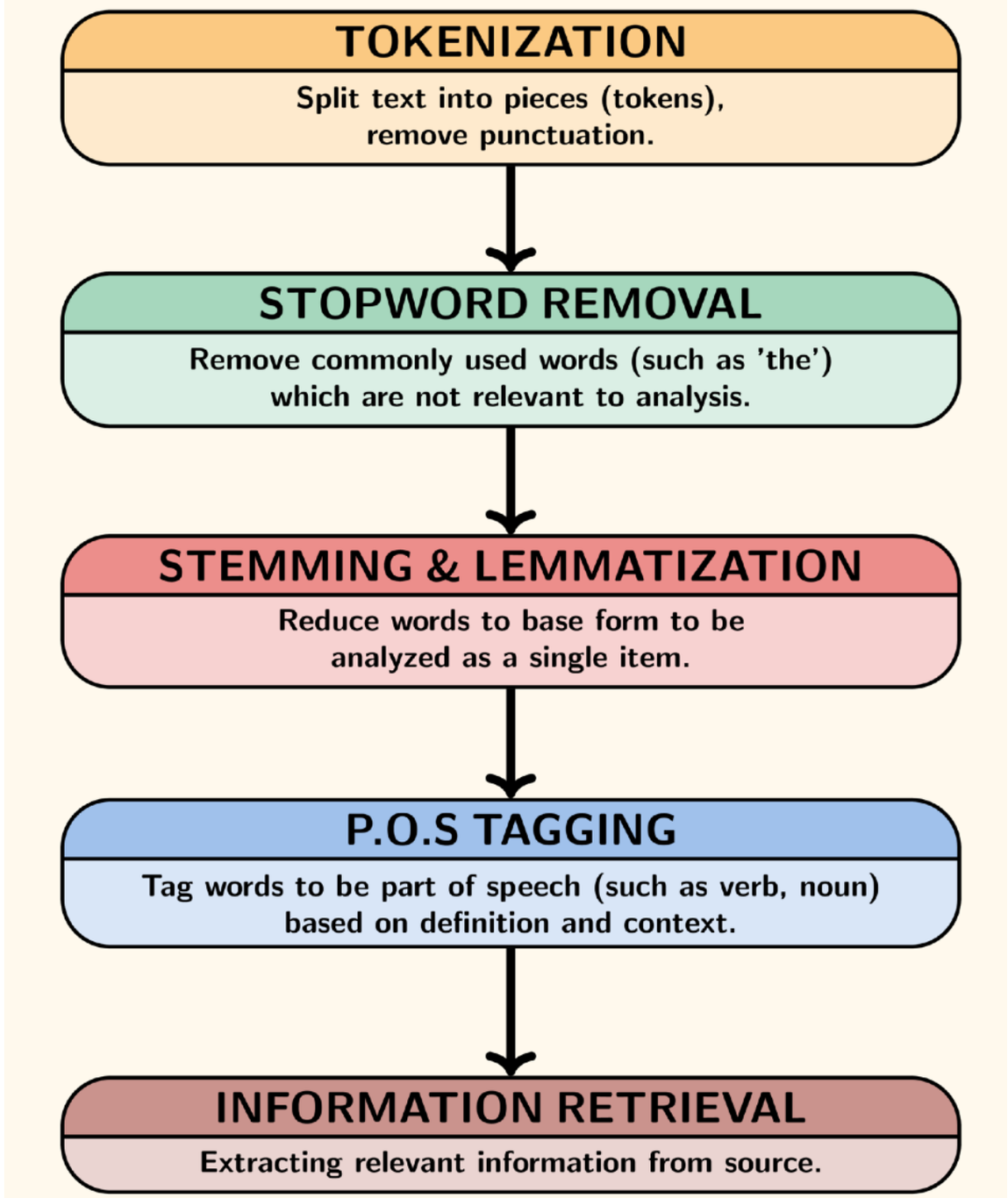
CV

```
01  X_fit, X_blindtest, y_fit, y_blindtest = model_selection.train_test_split(X, y, test_size = 0.1)
02
03  precision_cv_score = model_selection.cross_val_score(dt_model, X_fit, y_fit, cv=5,
scoring='precision_macro').mean()
04  recall_cv_score = model_selection.cross_val_score(dt_model, X_fit, y_fit, cv=5,
scoring='recall_macro').mean()
05  f1_cv_score = model_selection.cross_val_score(dt_model, X_fit, y_fit, cv=5,
scoring='f1_macro').mean()
06
07  print('CV: p:{0:.2f} r:{1:.2f} f:{2:.2f}'.format(precision_cv_score, recall_cv_score,
f1_cv_score))
```

Test example

```
1   dt_model.fit(X_fit,y_fit)
2
3   precision_test_score = metrics.precision_score(dt_model.predict(X_blindtest), y_blindtest,
average='macro')
4   recall_test_score = metrics.recall_score(dt_model.predict(X_blindtest), y_blindtest,
average='macro')
5   f1_test_score = metrics.f1_score(dt_model.predict(X_blindtest), y_blindtest, average='macro')
6
7   print('test: p:{0:.2f} r:{1:.2f} f:{2:.2f}'.format(precision_test_score, recall_test_score,
f1_test_score))
```

Application in IR



A big example — Bug report notation

- https://www.kaggle.com/anmolkumar/github-bugs-prediction
- A Kaggle dataset.
- A hackathon challenging the community to devise an algorithm that can predict the bugs, features, and questions based on **GitHub titles** and the **text body**.

Dataset Description

- **Train.json** - 150000 rows x 3 columns (Includes label Column as Target variable)
- Test.json - 30000 rows x 2 columns
- **Train_extra.json** - 300000 rows x 3 columns (Includes label Column as Target variable)
- Sample Submission.csv - Please check the Evaluation section for more details on how to generate a valid submission

Attribute Description

- **Title -** the title of the GitHub bug, feature, question
- **Body -** the body of the GitHub bug, feature, question
- **Label -** represents various classes of Labels
  - Bug - 0
  - Feature - 1
  - Question - 2

To do

- Build an estimation model
  - to predict whether a particular message saying about bug or non bug.

To do

- Build an estimation model
  - to predict whether a particular message saying about bug or non bug.

Some process walkthroughs

```
1   dataset = pd.read_json('src/resource/embold_train.json')
2   dataset.loc[dataset['label'] > 0, 'label'] = 1
3
4   with mp.Pool(processes=7) as pool:
5         cleaned_title = pool.map(preprocess, dataset.title)
6
7   with mp.Pool(processes=7) as pool:
8         cleaned_body = pool.map(preprocess, dataset.body)
```

Parallel IR preprocess

Some process walkthroughs

```
01  def preprocess(text):
02      cleaned_text = text.translate(str.maketrans('', '', '!"#$%&\'()*+,.<=>?@[]^`{|}~' + u'\xa0'))
03      cleaned_text = cleaned_text.lower()
04      cleaned_text = cleaned_text.translate(str.maketrans(string.whitespace, ' ' * len(string.whitespace), ''))
05
06      cleaned_text = ' '.join(['#weburl' if t.startswith('http') and '/' in t else t for t in
cleaned_text.split()])
07      cleaned_text = ' '.join(['#number' if re.sub('[\\/;:_-]','',t).isdigit() else t for t in
cleaned_text.split()])
08      cleaned_text = ' '.join(['#version' if any(i.isdigit() for i in t) and t.startswith('v') else t for t in
cleaned_text.split()])
09      cleaned_text = ' '.join(['#localpath' if ('\\' in t or '/' in t) and ':' not in t else t for t in
cleaned_text.split()])
10      cleaned_text = ' '.join(['#image_size' if t.endswith('px') else t for t in cleaned_text.split()])
11      cleaned_text = ' '.join(['#variable_with_underscore' if '_' in t else t for t in cleaned_text.split()])
12      cleaned_text = ' '.join(['#variable_with_dash' if '-' in t else t for t in cleaned_text.split()])
13      cleaned_text = ' '.join(['#long_variable_name' if len(t) > 20 else t for t in cleaned_text.split()])
14
15      tokenized_text = word_tokenize(cleaned_text)
16      stop_dict = {s: 1 for s in stopwords.words()}
17      sw_removed_text = [word for word in tokenized_text if word not in stop_dict]
18      sw_removed_text =[word for word in sw_removed_text if len(word) > 2]
19
20      ps = PorterStemmer()
21      stemmed_text = ' '.join([ps.stem(w) for w in sw_removed_text])
22
23      return stemmed_text
```

Same as those of in the first and the second modules

Some process walkthroughs

```
1   vectorizer = TfidfVectorizer(ngram_range=(1,1))
2   X_title = vectorizer.fit_transform(cleaned_title)
3   X_body = vectorizer.fit_transform(cleaned_body)
4
5   X = hstack([X_title, X_body])
6   y = dataset['label']
```

Try bigram if you have an access to high computing power.

Transform the preprocess texts into ngram tf-idf

Some process walkthroughs

```
1   dt_model = tree.DecisionTreeClassifier()
2
3   X_fit, X_blindtest, y_fit, y_blindtest = model_selection.train_test_split(X, y,
test_size=0.1)
4
5   precision_cv_score = model_selection.cross_val_score(dt_model, X_fit, y_fit, cv=3,
n_jobs=-2, scoring='precision_macro').mean()
6   recall_cv_score = model_selection.cross_val_score(dt_model, X_fit, y_fit, cv=3,
n_jobs=-2, scoring='recall_macro').mean()
7   f1_cv_score = model_selection.cross_val_score(dt_model, X_fit, y_fit, cv=3,
n_jobs=-2, scoring='f1_macro').mean()
8
9   print('CV: p:{0:.2f} r:{1:.2f} f:{2:.2f}'.format(precision_cv_score,
recall_cv_score, f1_cv_score))
```

Use #number of total CPU cores minus 1

Try CV>3 if you have an access to high computing power.

Cross validation

Some process walkthroughs

```
1   dt_model.fit(X_fit, y_fit)
2
3   precision_test_score = metrics.precision_score(dt_model.predict(X_blindtest),
y_blindtest, average='macro')
4   recall_test_score = metrics.recall_score(dt_model.predict(X_blindtest),
y_blindtest, average='macro')
5   f1_test_score = metrics.f1_score(dt_model.predict(X_blindtest), y_blindtest,
average='macro')
6
7   print('test: p:{0:.2f} r:{1:.2f} f:{2:.2f}'.format(precision_test_score,
recall_test_score, f1_test_score))
```

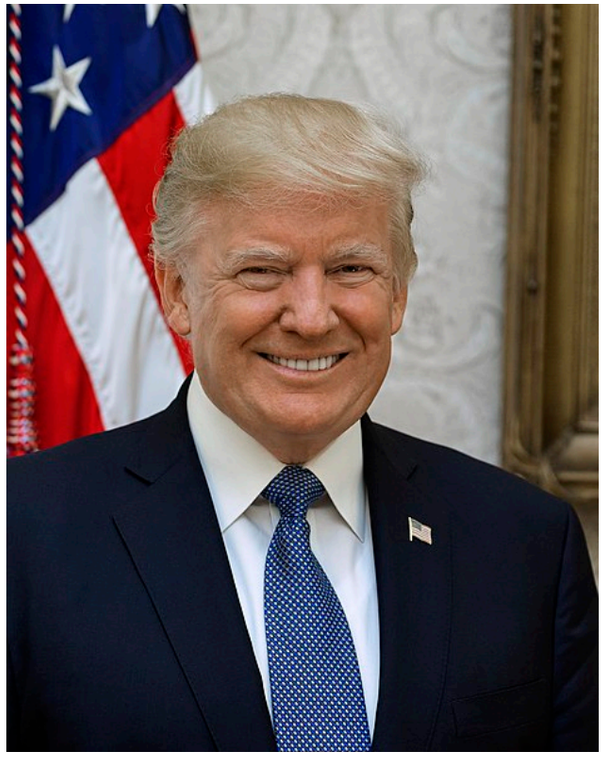
Predict the unseen samples

Further reads

- There are so many advanced techniques used in IR and NPL nowadays, e.g.,
  - Deep learning — Transformer, Dynamic embedding, Attention, Language model, etc.
  - Topic modeling
  - Etc.

Storytelling #1

Motivational example



They have something in common

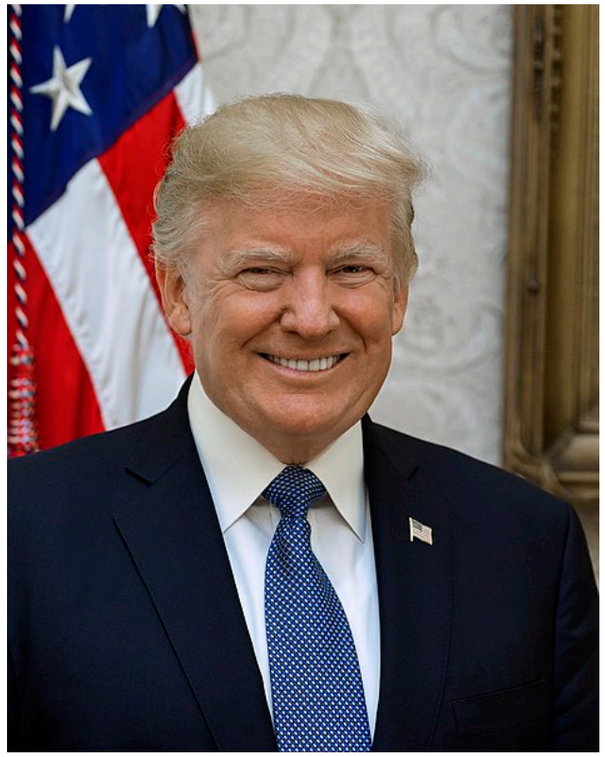
Data science in politics

- Traditional approach
  - Talk Talk Talk
  - Speech Speech Speech
    - Build the policy based on the experts' opinion
- A modern approach utilizes the data to help with the campaign.
- Social network is the new source of information.

Motivational example

How Obama’s Team Used Big Data to Rally Voters
by Sasha Issenberg

https://www.technologyreview.com/s/509026/how-obamas-team-used-big-data-to-rally-voters/



Leaked: Cambridge Analytica's blueprint for Trump victory

https://www.theguardian.com/uk-news/2018/mar/23/leaked-cambridge-analyticas-blueprint-for-trump-victory

Ref: https://www.inc.com/alyssa-satara/if-you-dont-fully-understand-cambridge-analytica-scandal-read-this-simplified-version.html

Simplified explanation

- It started with an academic researcher made a facebook app to let people take a survey to learn about **their personality**, and invited people to take the survey and help academic research.
- Close to 300,000 people took the survey, they were glad to allow their data to be used for research.
- They dont know that the app also collect the information about their profile and information about the profiles of all of their friends

Simplified explanation

- Cambridge Analytica bought this data from the academic researcher and maybe started using it for some of their political campaigns.

- But how come, these data may not seem to be sensitive but they did matter dramatically.

Simplified explanation

- Cambridge Analytica ended up getting access to information about the private profiles of 87 million Facebook users without the permission of those Facebook users.
- Mark Zuckerberg went to testify in Congress.

There were two papers which seem relavant

PNAS

Private traits and attributes are predictable from digital records of human behavior

Michal Kosinski[a,1], David Stillwell[a], and Thore Graepel[b]

[a]Free School Lane, The Psychometrics Centre, University of Cambridge, Cambridge CB2 3RQ United Kingdom; and [b]Microsoft Research, Cambridge CB1 2FB, United Kingdom

Edited by Kenneth Wachter, University of California, Berkeley, CA, and approved February 12, 2013 (received for review October 29, 2012)

We show that easily accessible digital records of behavior, Facebook Likes, can be used to automatically and accurately predict a range of highly sensitive personal attributes including: sexual orientation, ethnicity, religious and political views, personality traits, intelligence, happiness, use of addictive substances, parental separation, age, and gender. The analysis presented is based on a dataset

browsing logs (11–15). Similarly, it has been shown that personality can be predicted based on the contents of personal Web sites (16), music collections (17), properties of Facebook or Twitter profiles such as the number of friends or the density of friendship networks (18–21), or language used by their users (22). Furthermore, location within a friendship network at Facebook was shown to be

PNAS

Computer-based personality judgments are more accurate than those made by humans

Wu Youyou[a,1,2], Michal Kosinski[b,1], and David Stillwell[a]

[a]Department of Psychology, University of Cambridge, Cambridge CB2 3EB, United Kingdom; and [b]Department of Computer Science, Stanford University, Stanford, CA 94305

Edited by David Funder, University of California, Riverside, CA, and accepted by the Editorial Board December 2, 2014 (received for review September 28, 2014)

Judging others' personalities is an essential skill in successful social living, as personality is a key driver behind people's interactions, behaviors, and emotions. Although accurate personality judg-

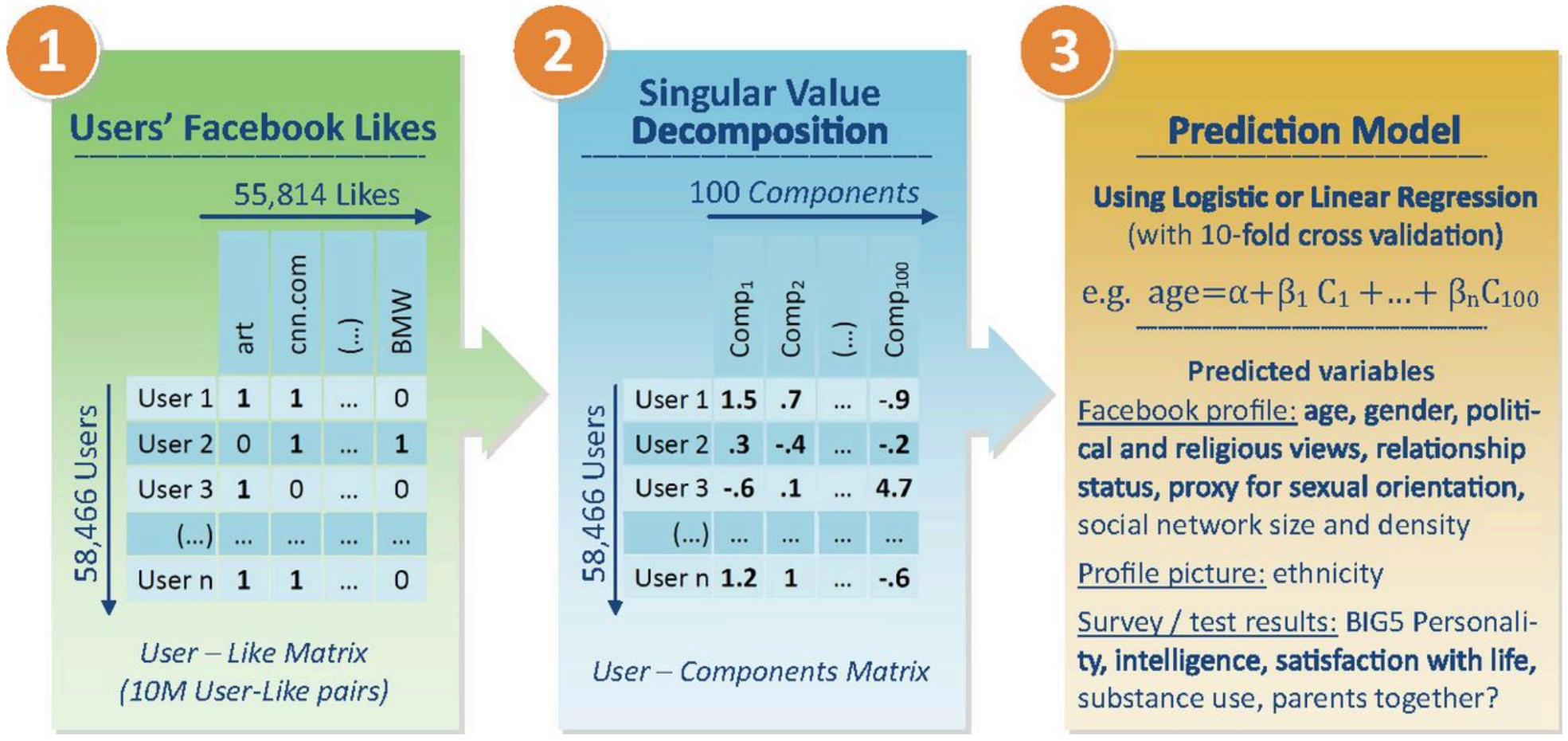
psychological traits (11). We used LASSO (Least Absolute Shrinkage and Selection Operator) linear regressions (16) with 10-fold cross-validations, so that judgments for each participant

The two papers which seem relevant

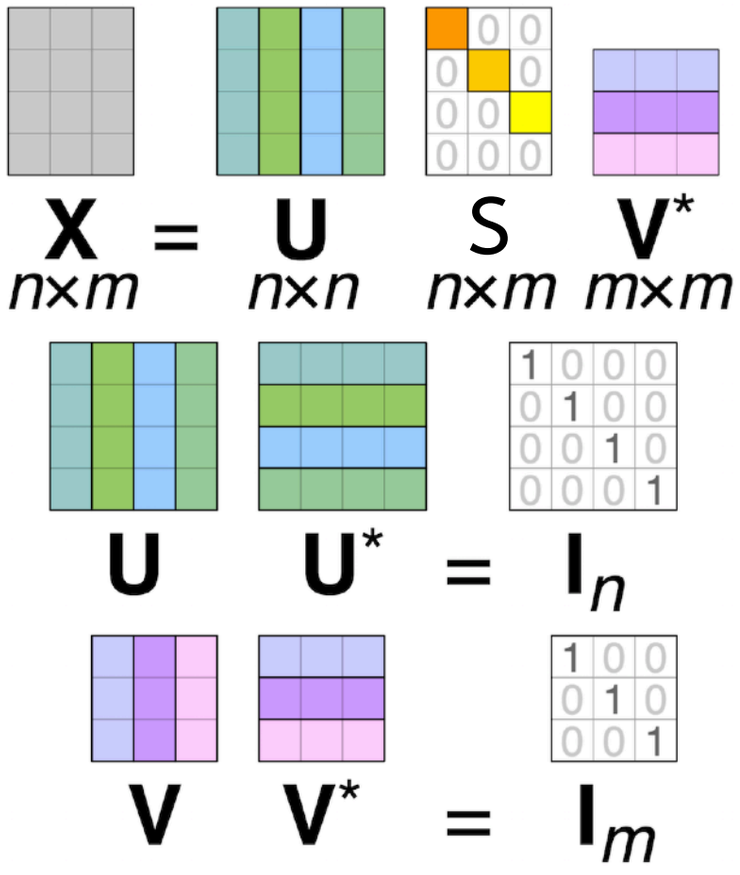
- Kosinski, M., Stillwell, D., & Graepel, T. (2013). **Private traits and attributes are predictable from digital records of human behavior**. Proceedings of the National Academy of Sciences, 110(15), 5802-5805.

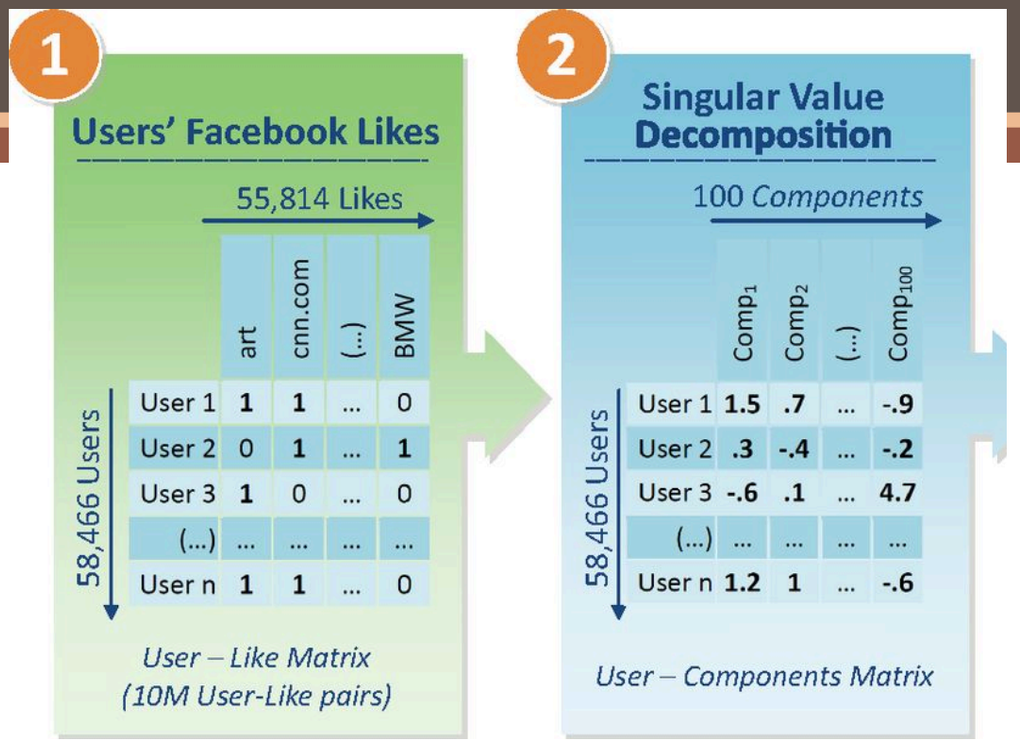
- Youyou, W., Kosinski, M., & Stillwell, D. (2015). **Computer-based personality judgments are more accurate than those made by humans**. Proceedings of the National Academy of Sciences, 112(4), 1036-1040.

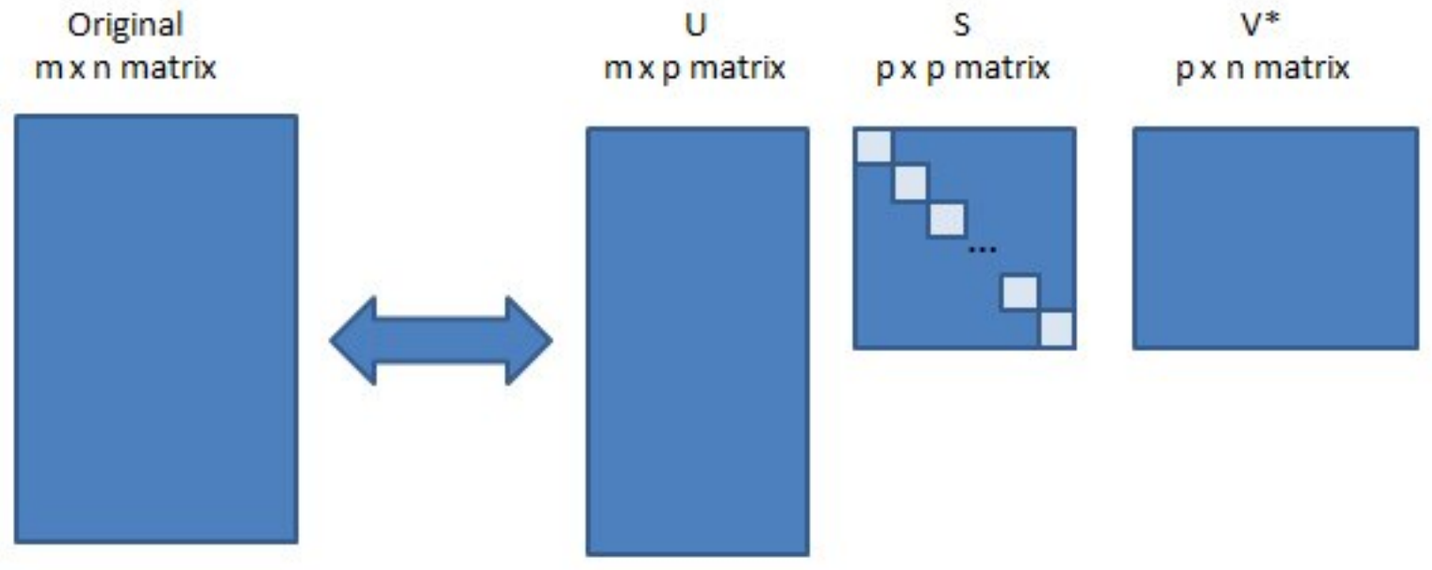
The 2013 paper



Singular value decomposition







Ref: https://blogs.oracle.com/r/using-svd-for-dimensionality-reduction

Singular value decomposition

Consider the 4 × 5 matrix

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \end{bmatrix}$$

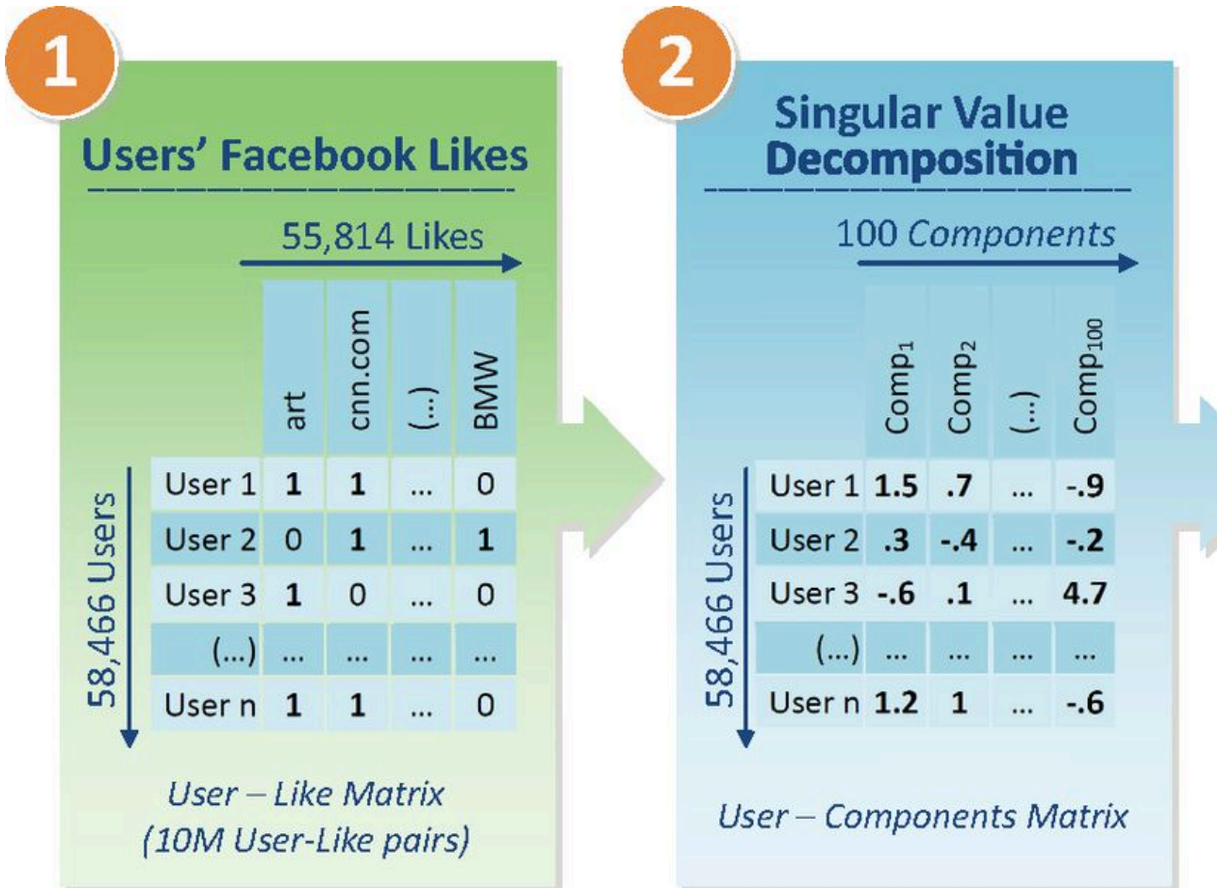
A singular value decomposition of this matrix is given by $\mathbf{U}$

$$\mathbf{U} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{\Sigma} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & \sqrt{5} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{0} & 0 \end{bmatrix}$$

$$\mathbf{V}^* = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sqrt{0.2} & 0 & 0 & 0 & \sqrt{0.8} \\ 0 & 0 & 0 & 1 & 0 \\ -\sqrt{0.8} & 0 & 0 & 0 & \sqrt{0.2} \end{bmatrix}$$

Singular value decomposition



Ref: https://blogs.oracle.com/r/using-svd-for-dimensionality-reduction

Singular value decomposition

3

Prediction Model

Using Logistic or Linear Regression
(with 10-fold cross validation)

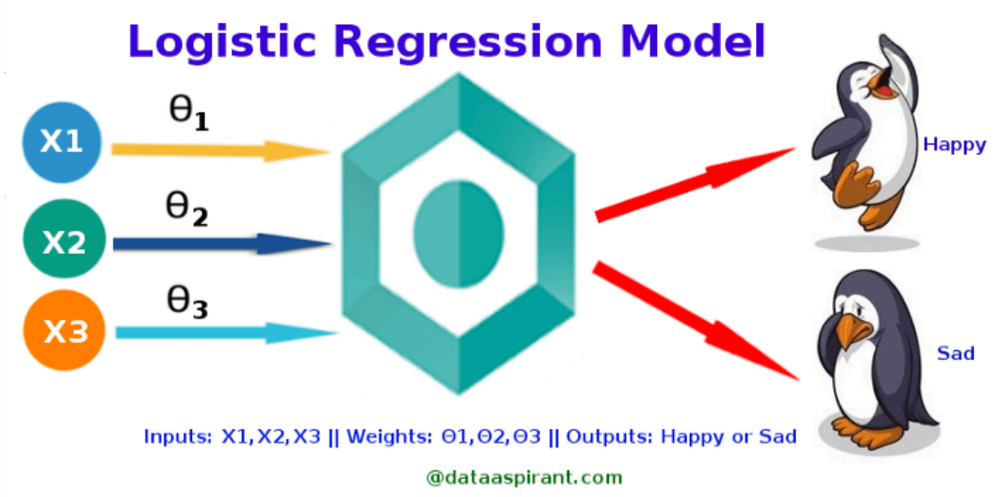
e.g. $age = \alpha + \beta_1 C_1 + \ldots + \beta_n C_{100}$

Predicted variables

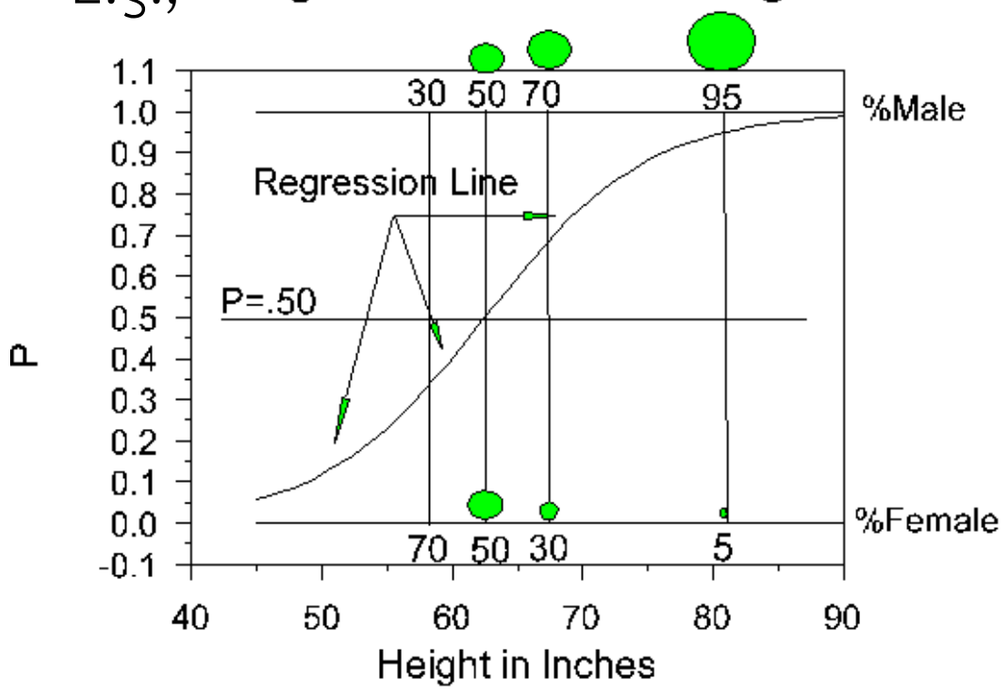
Facebook profile: **age, gender, political and religious views, relationship status, proxy for sexual orientation,** social network size and density

Profile picture: ethnicity

Survey / test results: BIG5 Personali-**ty, intelligence, satisfaction with life,** substance use, parents together?



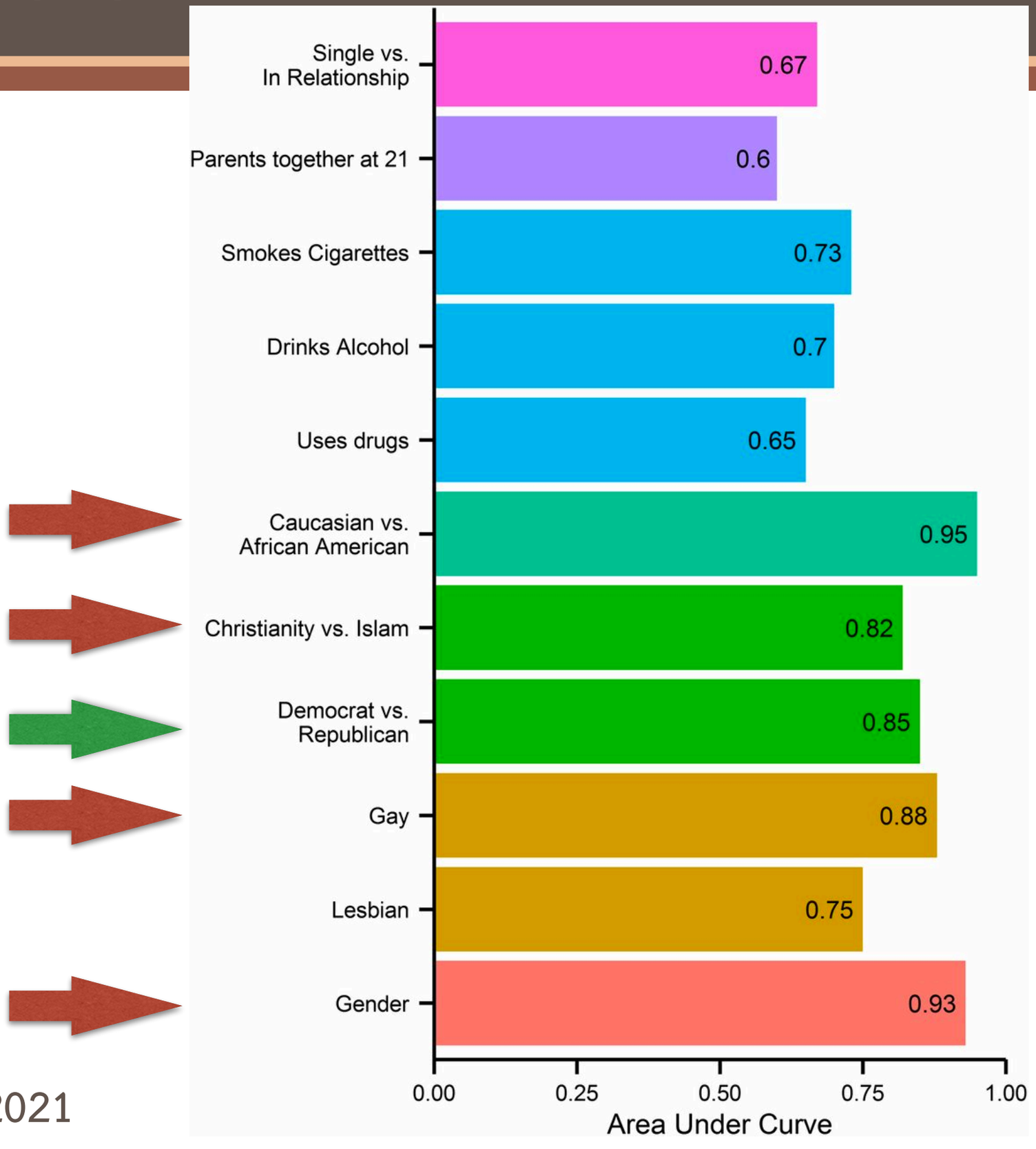
- E.g., Regression of Sex on Height



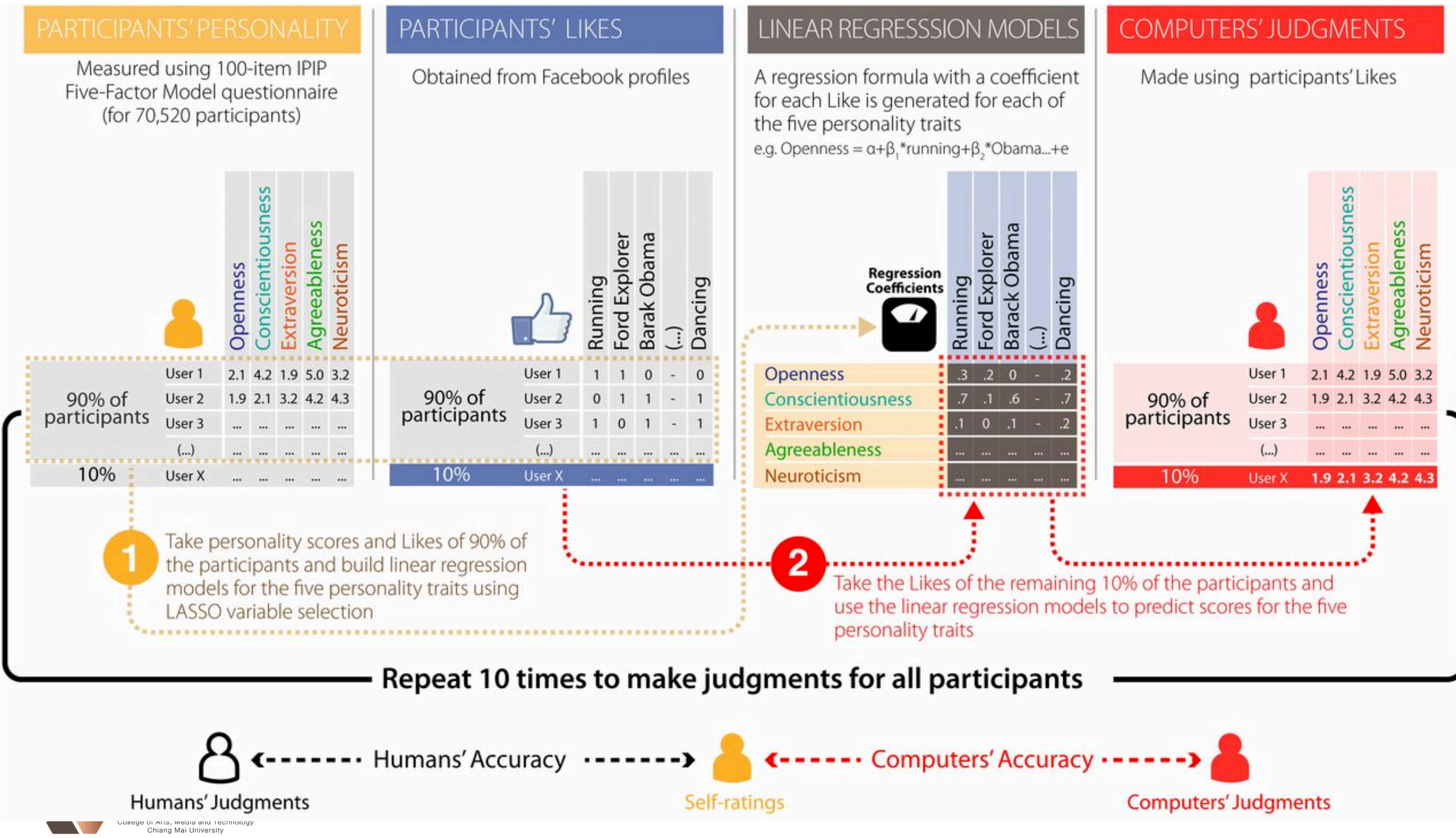
Ref: https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc

Ref: https://blogs.oracle.com/r/using-svd-for-dimensionality-reduction

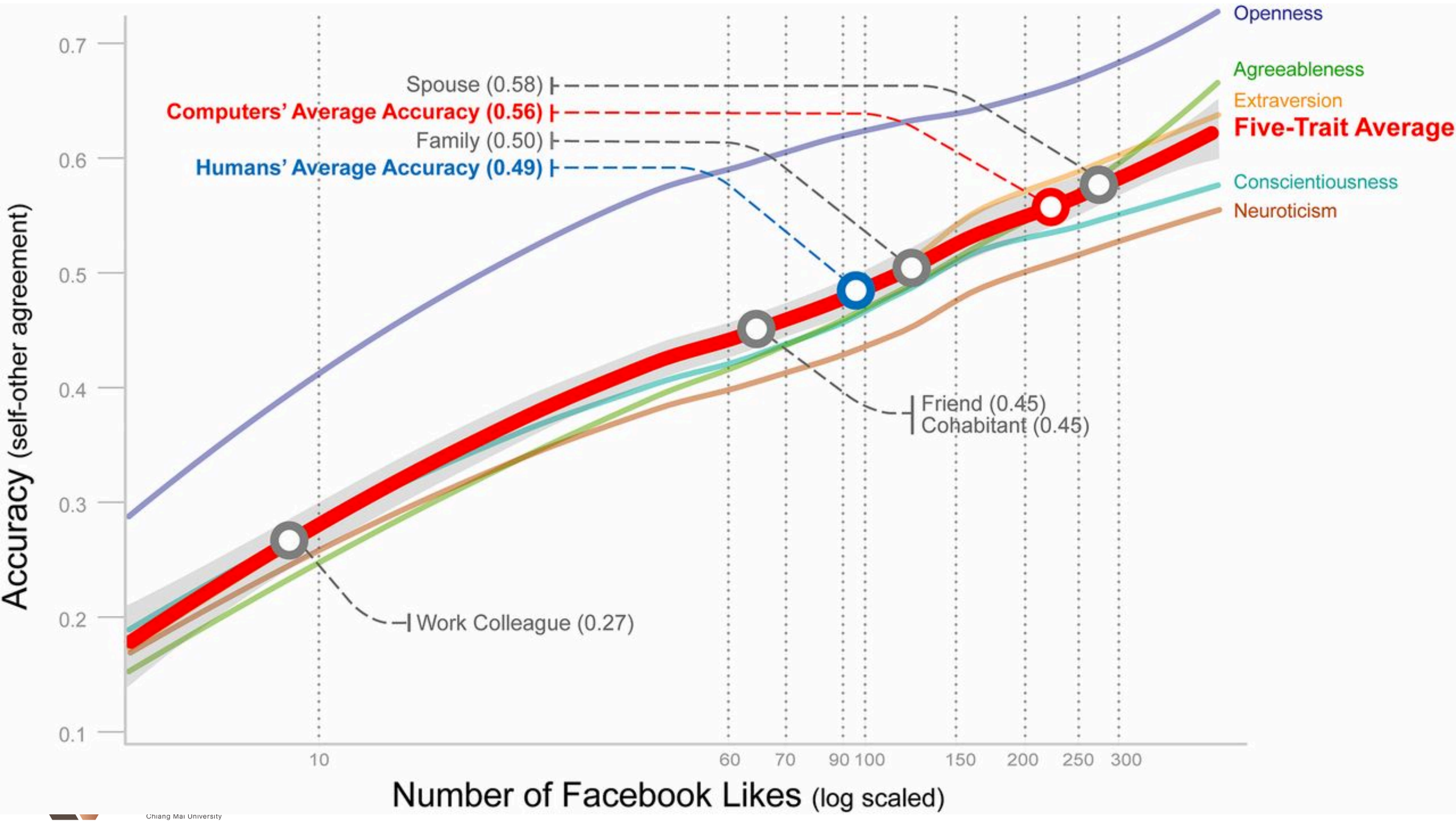
The 2013 paper



The 2015 paper



The 2015 paper



Earlier experiments by Facebook

- Facebook did an experiment a couple of years back in a US election where they sent a get out the vote message to a bunch of people on Facebook.
- Facebook showed people a message that said, hey, it's election day. You should go vote.
- Actually, they had two different messages.
  - Half the people saw the original message.
  - For the other half, Facebook also showed these people thumbnail
  - images of some of their friends who'd already voted.

There were two papers which seem relevant

- Facebook claimed that it can hundred thousand of people to go to vote.

What if, for some reasons, the influencing message version is just appeared to the users who lean to one party over the other?

Some notes

- None of the data science techniques (as far as I know) used in the papers seems to be an ultimately advance technique.
- The procedure seems to be straightforward, but what it can deliver is extremely lucrative.

Time for questions