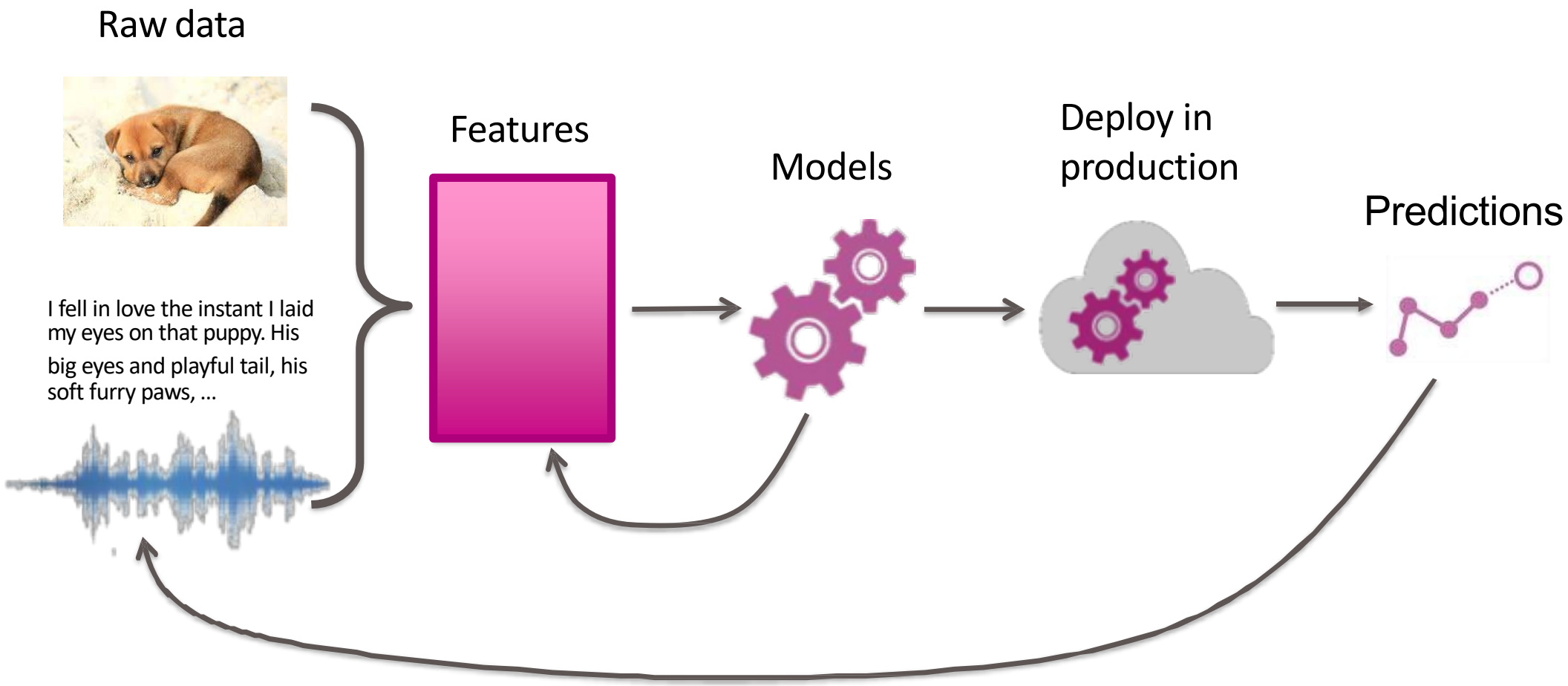
Creating, Evaluating and Improving Machine Learning Models

Gaurav Goswami

AI/ML Expert

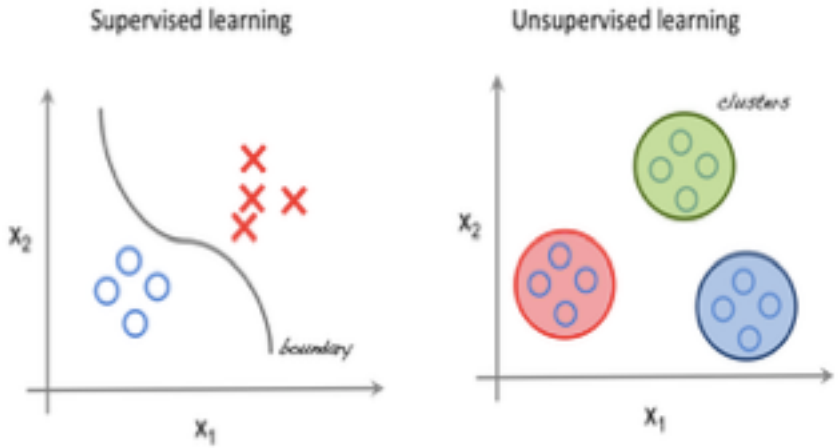
Slides prepared and compiled using various online sources.

The machine learning pipeline



Different Modes of Machine Learning

- **Supervised:**
  - Ground truth is available for each data point
  - Map input (features) to output (labels)
  - Examples: find the sentiment of a new tweet given a database of labeled tweets
- **Unsupervised:**
  - No ground truth is available
  - Map data points to groups (or clusters) in the data
  - Examples: categorize people into different spending capability groups based on financial data
- **Reinforcement:**
  - Behavior inspired: Action and consequence (reward)
  - No “correct” input-output pairs
  - Focus is on the overall ‘reward’
  - Examples: robot control



Features

member_id	loan_amnt	funded_amnt	funded_amnt_inv	term	batch_enrolled	int_rate	grade	sub_grade	emp_title
11937648	14000	14000	14000	60 months	BAT4711174	16.24	C	C5	Data Analyst
38983318	16000	16000	16000	60 months	BAT4318899	9.49	B	B2	Senior Database Administrator
27999917	11050	11050	11050	60 months	BAT446479	15.61	D	D1	Customer service representative
61514932	35000	35000	34700	60 months	BAT4664105	12.69	C	C2	ACCT OFFICER
59622821	6500	6500	6500	36 months		6.89	A	A3	Paralegal
28822038	13475	13475	13475	60 months		18.99	E	E1	Human Resource
10718089	5000	5000	5000	36 months		7.62	A	A3	Software Engineer
58114582	10000	10000	10000	60 months	BAT5662637	22.99	F	F2	Deli Manager
35023176	30000	30000	30000	36 months	BAT6248271	9.17	B	B1	Registered Nurse
1268247	7000	7000	7000	60 months	BAT4467682	15.96	C	C5	HSBC
15559751	6000	6000	6000	36 months	BAT3274746	11.99	B	B3	3rd Mate
41234300	6000	6000	6000	36 months		14.31	C	C4	
53958774	21600	21600	21600	60 months	BAT422167	8.18	B	B1	Maintenance
8074949	17000	17000	16950	36 months		11.55	B	B3	postal service
67601509	10000	10000	10000	60 months	BAT4106493	13.33	C	C3	Mail carrier
65560239	30000	30000	30000	60 months		17.57	D	D4	Documetn Control Manager
14767770	6350	6350	6350	36 months	BAT3016651	23.43	F	F1	trucking assistant
1726792	1200	1200	1200	36 months		7.9	A	A4	Twin Peaks
9899439	20000	20000	20000	36 months		17.1	C	C5	sales associate
15120294	27600	27600	27600	60 months	BAT3016651	15.31	C	C4	CLAIMS SUPERVISOR
12019444	35000	35000	35000	36 months	BAT4711174	14.98	C	C3	Operator
35242810	5750	5750	5750	36 months		13.98	C	C3	Branch Assistant Manager

Label

member_id	loan_status
1296599	1
1314167	0
1313524	1
1277178	1
1288686	1
1306721	0
1303503	0
1269083	0
1304821	1
1304810	1
1304634	0
1304679	0
1304764	0
1304678	1
1278806	1
1261745	1
1299514	0
1304255	1
1301955	1
1304202	1
1282787	1
1304166	1
1304171	1

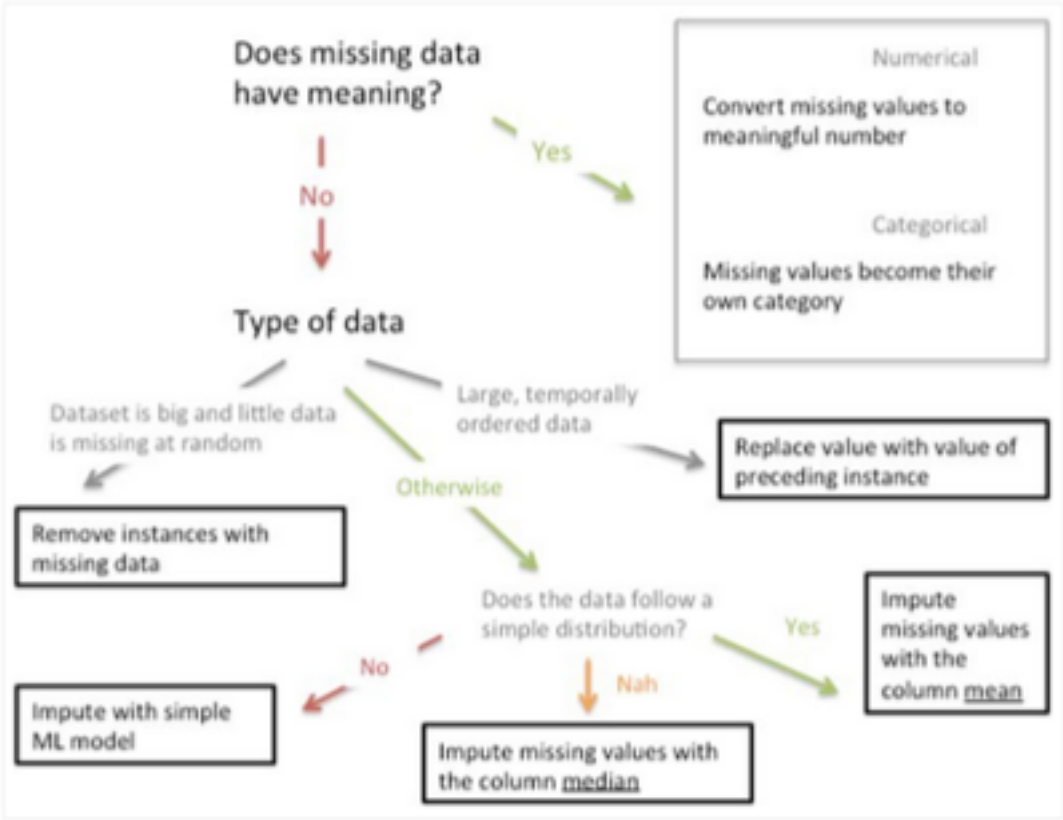
Golden Rule of ML: Data is King

- Everything in ML revolves around the data
- First step of creating a model: Explore the data (EDA)
- Various plots and visualizations
- Determine preprocessing steps required for the data
- Custom build the initial pipeline based on the data

Goals of Preprocessing

- Adapt and coalesce the data to a compatible format
- Cleaning the data
  - Handle missing values
  - Handle outliers
  - Encode categorical variables
  - Drop irrelevant features
- Transformation of the data
- Splitting the data into training/validation/testing sets

Preprocessing: Handling Missing Data



Preprocessing: Data Transformation

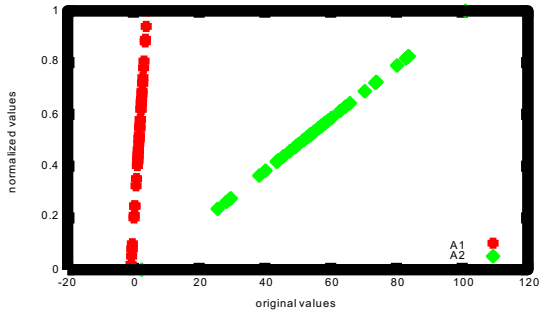
Some data mining tools tends to give variables with a large range a higher significance than variables with a smaller range. For example,

- Age versus income.

The typical approach is to standardize the scales:

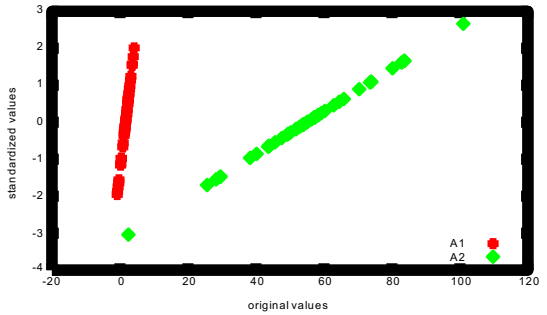
Min-Max Normalization:

$$X^* = \frac{X - min(X)}{max(X) - min(X)}.$$



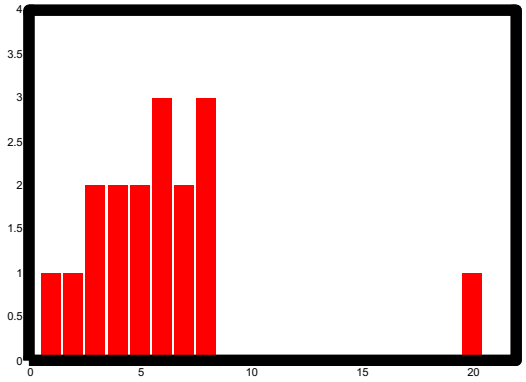
Z-score standardization:

$$X^* = \frac{X - mean(X)}{SD(X)}.$$



Preprocessing: Outliers

Data: $1, 2, 3, 3, 4, 4, 5, 5, 6, 6, 6, 6, 7, 7, 8, 8, 8, 20.$



Summary statistics:

- First quartile ($1Q$): 25% of the data $= 4$.
- Second quartile ($2Q$): 50% of the data $= 6$.
- Third quartile ($3Q$): 75% of the data $= 7$.

Interquartile range $IQR = 3Q - 1Q = 3$.

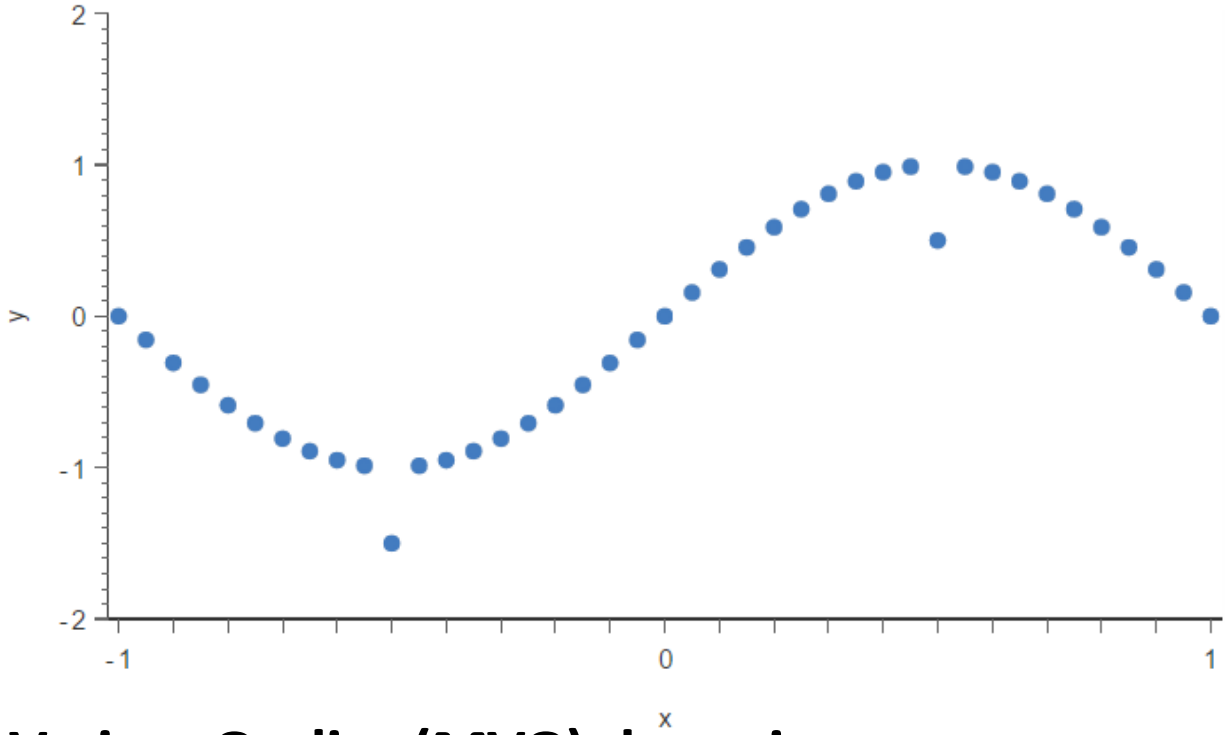
A data point may be an outlier if:

- It is lower than $1Q - 1.5 \cdot IQR = 4 - 1.5 \cdot 3 = -0.5$.
- It is higher than $3Q + 1.5 \cdot IQR = 7 + 1.5 \cdot 3 = 11.5$.

This approach is extreme value analysis

Preprocessing:Outliers

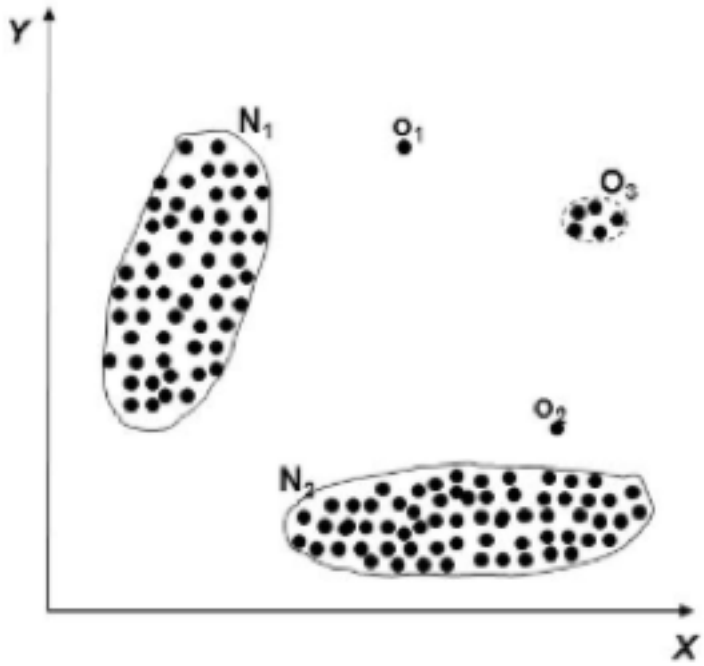
- What if ?



Multi Variate Outlier (MVO) detection approach

Preprocessing:Outliers

- Univariate Example: Age and height , but either only age or height is Analysed. (bar chart , histogram)
- Bivariate Example : Analyzing caloric intake vs weight , temp vs ice cream sale (Scatterplot)
- Multi variate Example (cluster /Discrimant Analysis
- / PCA/..)
  - Proximity method
    - Identify the natural cluster in the data (k-means)
    - Identify cluster centroids
    - Identify data instances that are fixed distance from centroid
    - Filter out outliers candidate from dataset



Outlier - Example

nvoiceNo	StockCode	Description	Quantity	I
536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	
536365	71053	WHITE METAL LANTERN	6	
536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	
536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	
536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	
536365	22752	SET 7 BABUSHKA NESTING BOXES	2	
536365	21730	GLASS STAR FROSTED T-LIGHT HOLDER	6	
536366	22633	HAND WARMER UNION JACK	6	
536366	22632	HAND WARMER RED POLKA DOT	6	
536367	84879	ASSORTED COLOUR BIRD ORNAMENT	32	
536367	22745	POPPY'S PLAYHOUSE BEDROOM	6	
536367	22748	POPPY'S PLAYHOUSE KITCHEN	6	

Preprocessing: Categorical Data

- Label encoding:
  - Assign an ID to each unique value
  - Days = [Mon, Tues, Wed, Thurs, Fri, Sat, Sun]
  - Days_transformed = [0,1,2,3,4,5,6]
  - Consideration: Possible implications for some ML models
- One-hot encoding:
  - Create as many new columns as there are distinct values in the feature
  - From one Days column to: IsMon, IsTues, IsWed, IsThurs, IsFri, IsSat, IsSun
  - 1 for the feature value, 0 for other columns
  - Consideration: dimensionality of the feature matrix

Label Vs One-Hot Encoding

Country	Age	Salary	Purchased
France	44	72000	No
Spain	27	48000	Yes
Germany	30	54000	No
Spain	38	61000	No
Germany	40	nan	Yes
France	35	58000	Yes
Spain	nan	52000	No
France	48	79000	Yes
Germany	50	83000	No
France	37	67000	Yes

```
array([[0, 44.0, 72000.0],
       [2, 27.0, 48000.0],
       [1, 30.0, 54000.0],
       [2, 38.0, 61000.0],
       [1, 40.0, nan],
       [0, 35.0, 58000.0],
       [2, nan, 52000.0],
       [0, 48.0, 79000.0],
       [1, 50.0, 83000.0],
       [0, 37.0, 67000.0]], dtype=object)
```

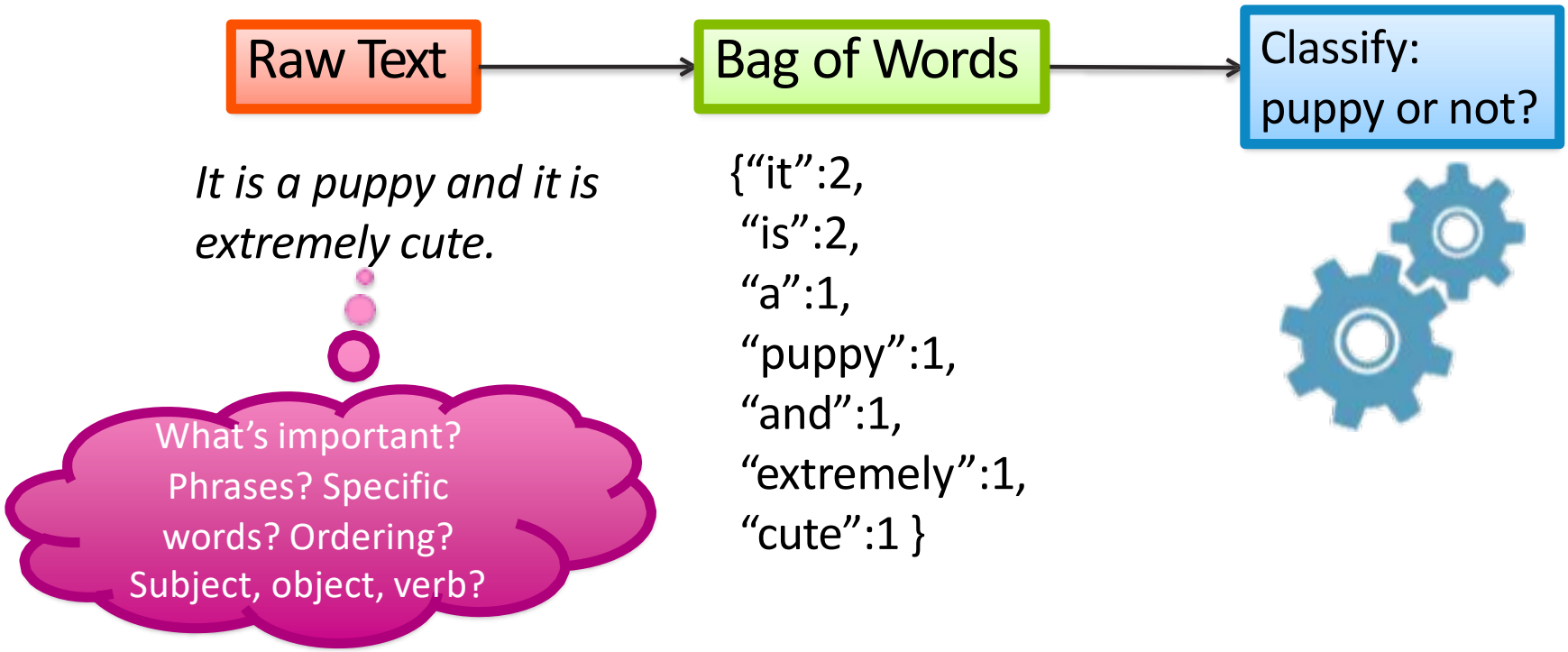
	0	1	2	3	4
0	1	0	0	44	72000
1	0	0	1	27	48000
2	0	1	0	30	54000
3	0	0	1	38	61000
4	0	1	0	40	63777.8
5	1	0	0	35	58000
6	0	0	1	38.7778	52000
7	1	0	0	48	79000
8	0	1	0	50	83000
9	1	0	0	37	67000

Features

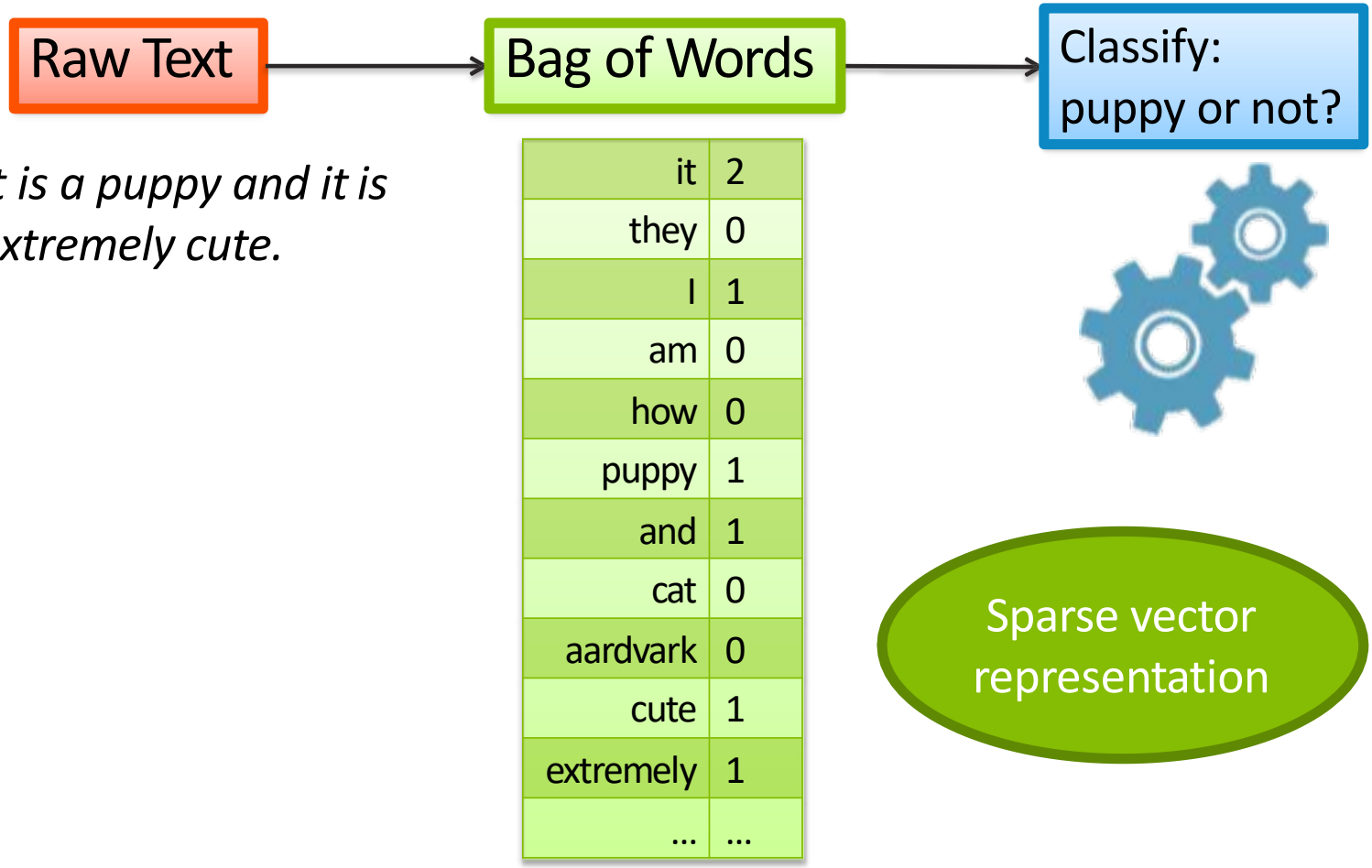
- Extract representations that capture the “essence” of data
- For classification: Maximize the intra-class similarity and inter-class distance
- For regression: Maximize the explanation power
- Multiple methods
- No clear “right” or “wrong” answers

- **Question:** What are the potential “features” for:
  - Assessing the credit risk of an individual
  - Predicting the type of loan a customer is most likely to accept

Features: Representing natural text



Features: Representing natural text



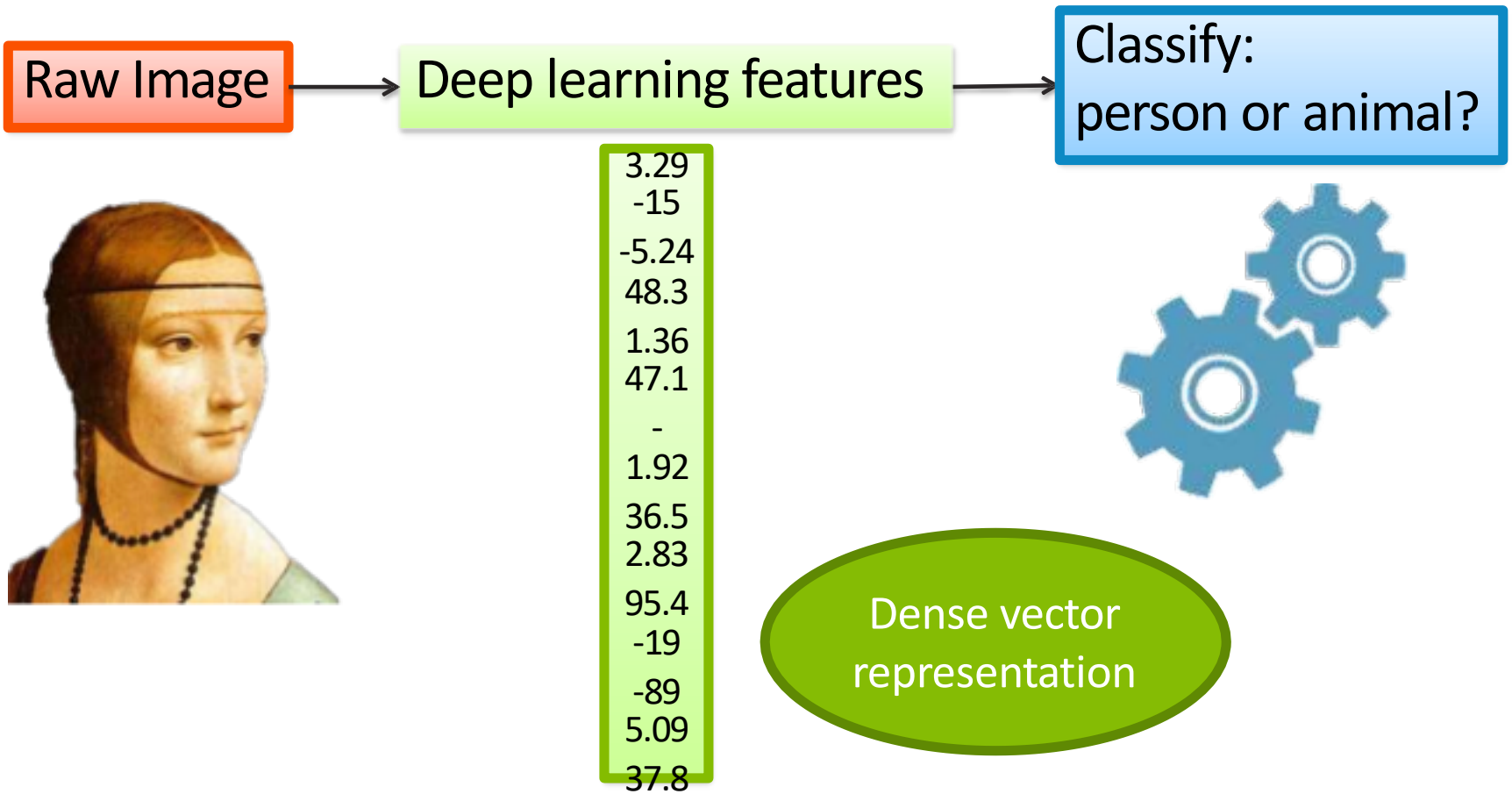
Features: Representing images

Raw Image → Bag of Visual Words → Classify: person or animal?

Raw image:
millions of RGB triplets,
one for each pixel

Image source: “Recognizing and learning object categories,”
Li Fei-Fei, Rob Fergus, Anthony Torralba, ICCV 2005—2009.

Features: Representing images



	loan_amnt	funded_amnt	funded_amnt_inv	term	int_rate	emp_length	annual_inc	dti
loan_amnt	1.0	1.0	1.0	0.43	0.15	0.11	0.49	0.076
funded_amnt	1.0	1.0	1.0	0.43	0.15	0.11	0.49	0.077
funded_amnt_inv	1.0	1.0	1.0	0.43	0.15	0.11	0.49	0.079
term	0.43	0.43	0.43	1.0	0.42	0.071	0.13	0.11
int_rate	0.15	0.15	0.15	0.42	1.0	0.0091	-0.12	0.17
emp_length	0.11	0.11	0.11	0.071	0.0091	1.0	0.12	0.043
annual_inc	0.49	0.49	0.49	0.13	-0.12	0.12	1.0	-0.18
dti	0.076	0.077	0.079	0.11	0.17	0.043	-0.18	1.0

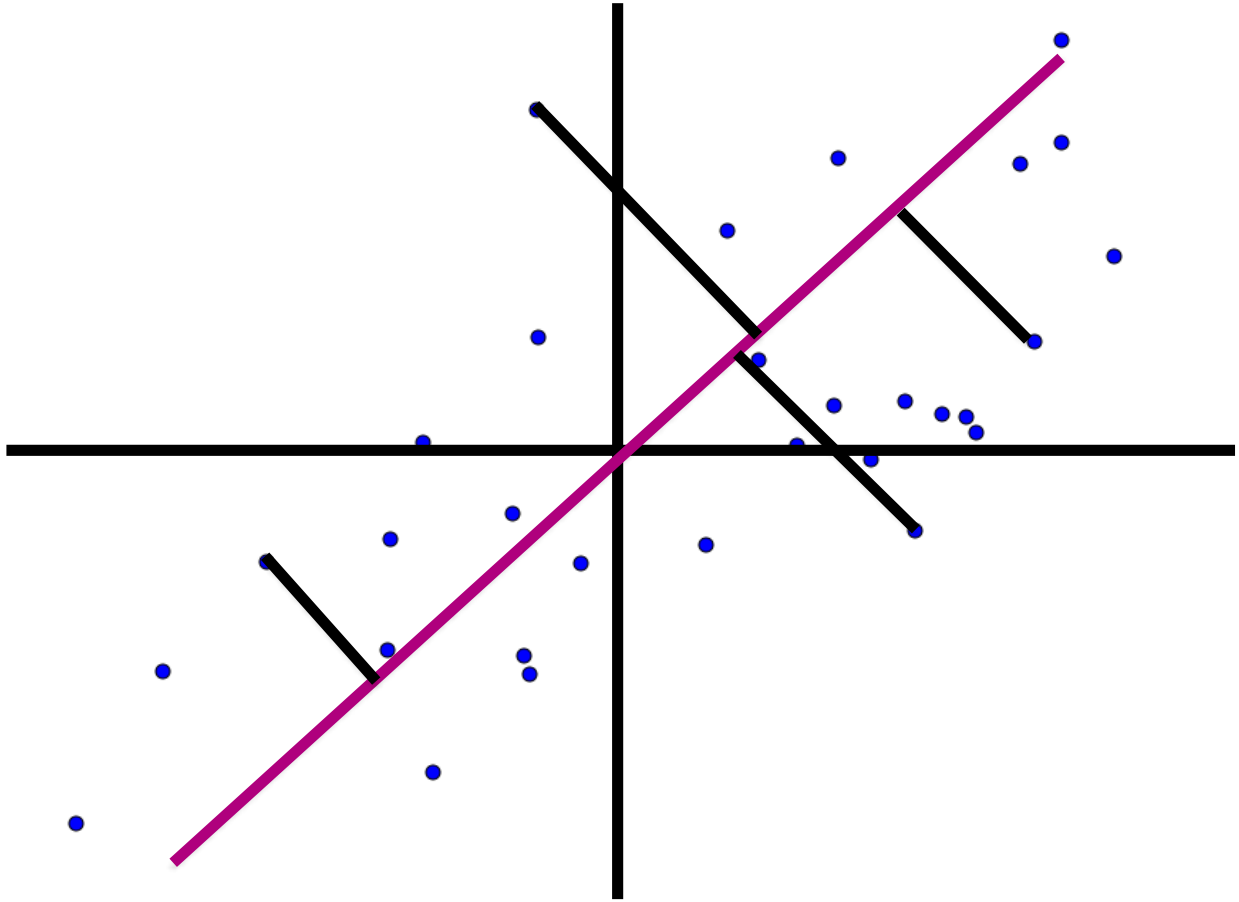
Features: Dimensionality Reduction

- Preserve the most relevant features
- Converge to a good model in lesser number of iterations
- Reduce computational complexity of training and testing
- Various techniques are available: PCA, Mutual Information, Correlation analysis, Feature Importance, Quality, Visual Saliency, etc.

PCA : Principal Component Analysis

Find a line, such that the average distance of every data point to the line is minimized.

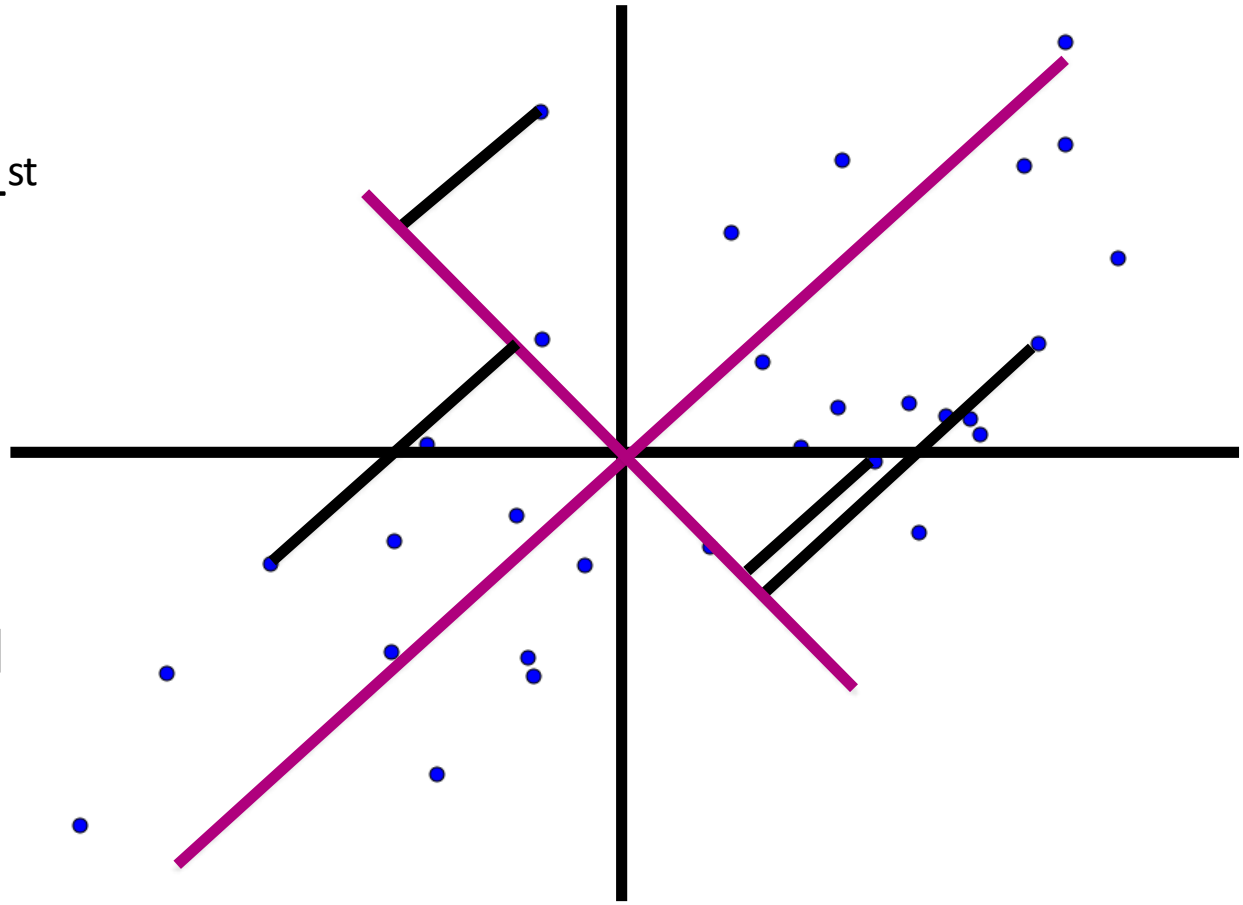
This is the 1st Principal Component



PCA : Principal Component Analysis

Find a 2nd line,
-at right angles to the 1st
-such that the average distance of every data point to the line is minimized.

This is the 2nd Principal Component

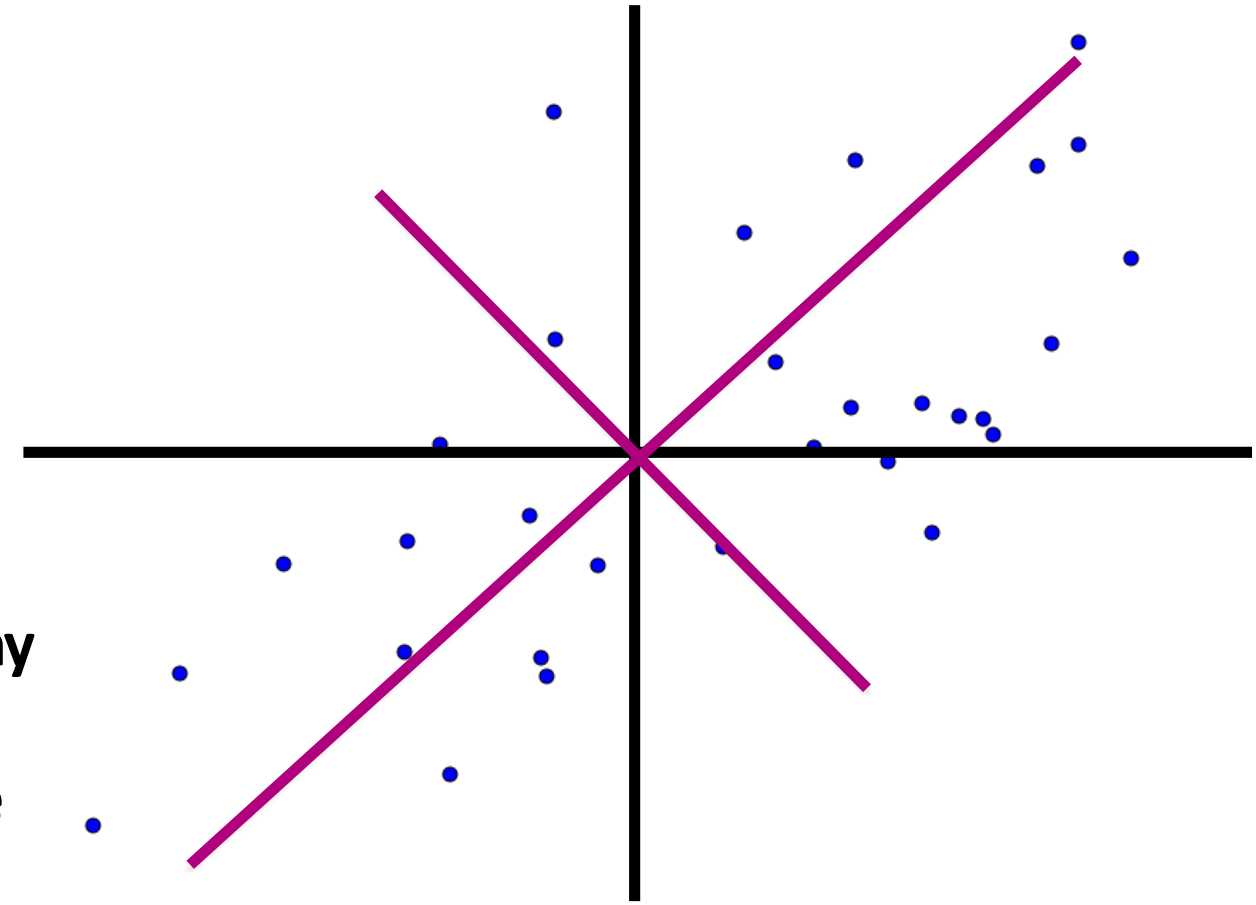


PCA : Principal Component Analysis

Find a 3rd line
-at right angles to the previous lines
-such that the average distance of every data point to the line is minimized.

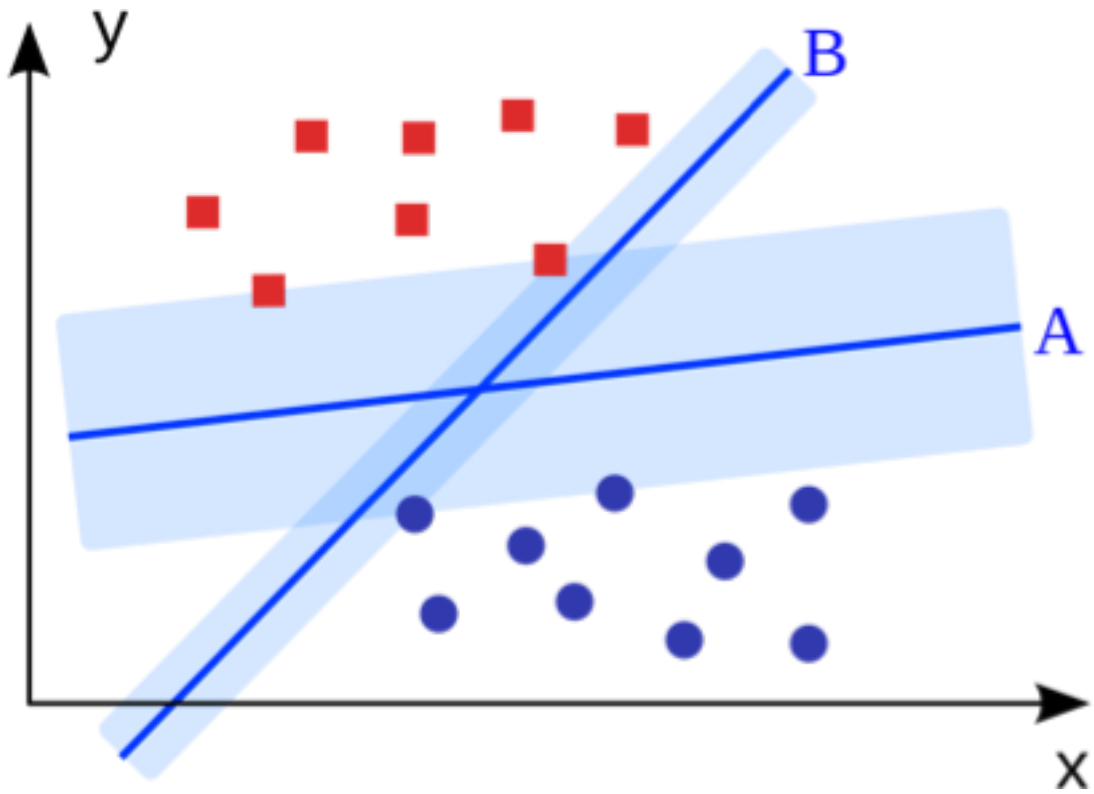
…

There can only be as many principle components as the dimensionality of the data.



Model Training

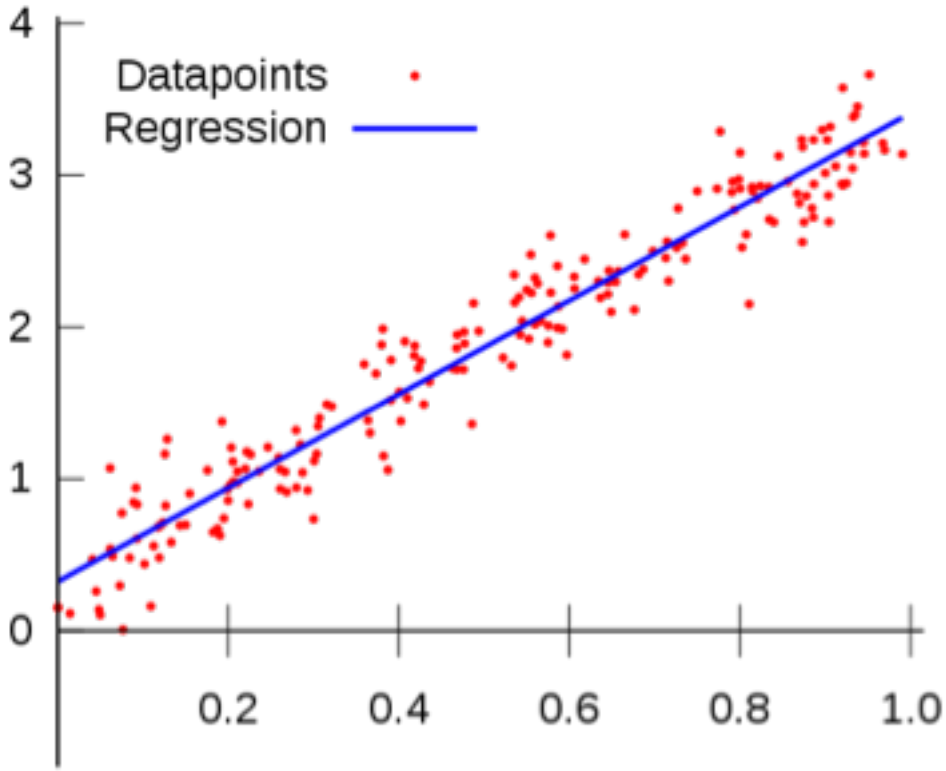
- Learn the mapping from features to output
- Ideal: 100% accuracy in differentiating features of different classes
- Practical: Maximize accuracy while maintaining “generalizability”
- Quality dependent on training data quality/quantity
- Choosing an ML algorithm for the task
- Choosing the labels



Linear Regression

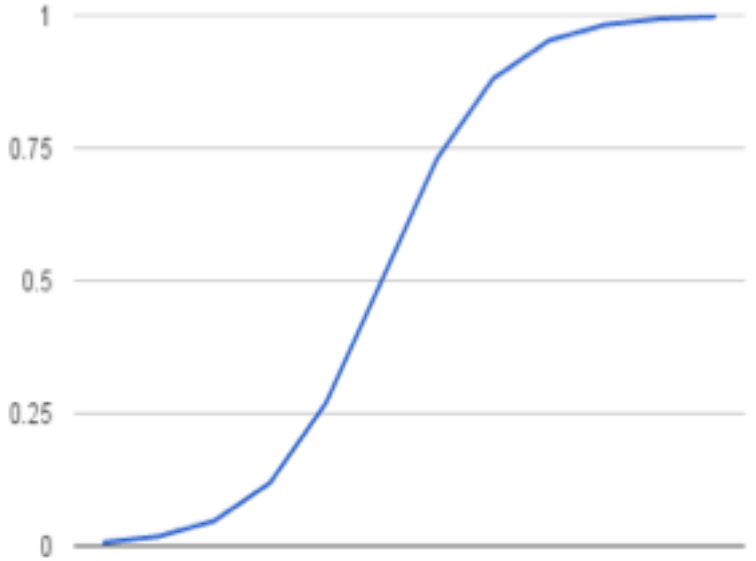
- Fit a model to compute the value of a dependent scalar variable based on one or more predictors
- Single predictor: Simple linear regression
- y = B0 + B1 * x1
  - B0 – bias / error
  - B1- coefficient/weights

- Weight = B0 +B1*Height
  - For example, lets use B0 = 0.1 and B1 = 0.5
  - Weight=91 = 0.1 +0.5*182
- Multiple predictors: Multiple linear regression
- Vector dependent variable: Multivariate linear regression



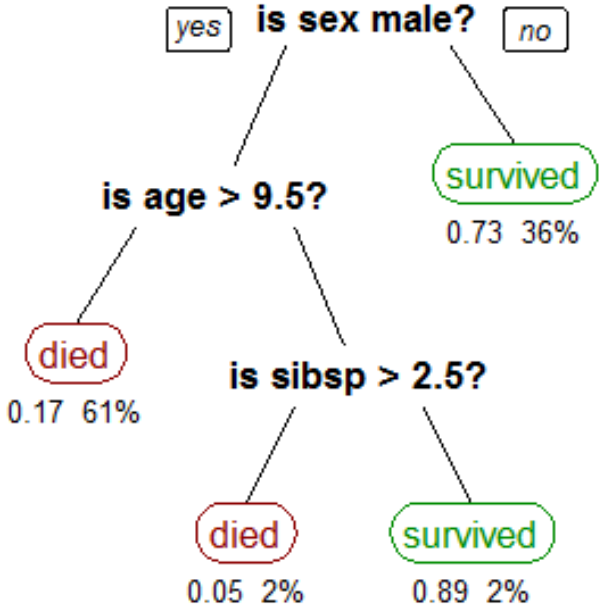
Logistic Regression

- sigmoid function to describe properties of population growth in ecology
- It's an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1,
- y = e^(b0 + b1*x) / (1 + e^(b0 + b1*x))
  - Y -> predicted output
  - b – bias (error) / co-efficient (weights)
- P(X) = P(Y=1|X)
  - Given Y=1 (diabetic)/given age)
- Model – Learn the co-efficients (b0/b1)
- Remove highly co-related features



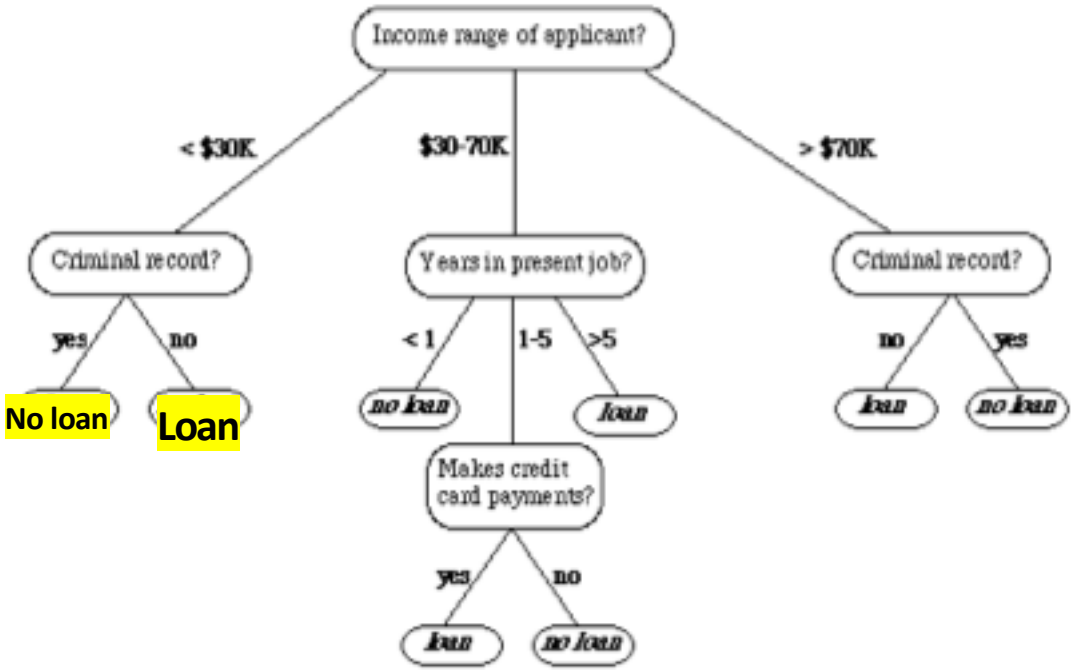
Decision Tree

- Performs intrinsic feature ranking
- Branches out at each level based on the feature that can provide the best separation
- Decides based on intervals of the features
- Different types of algorithms



Decision Tree

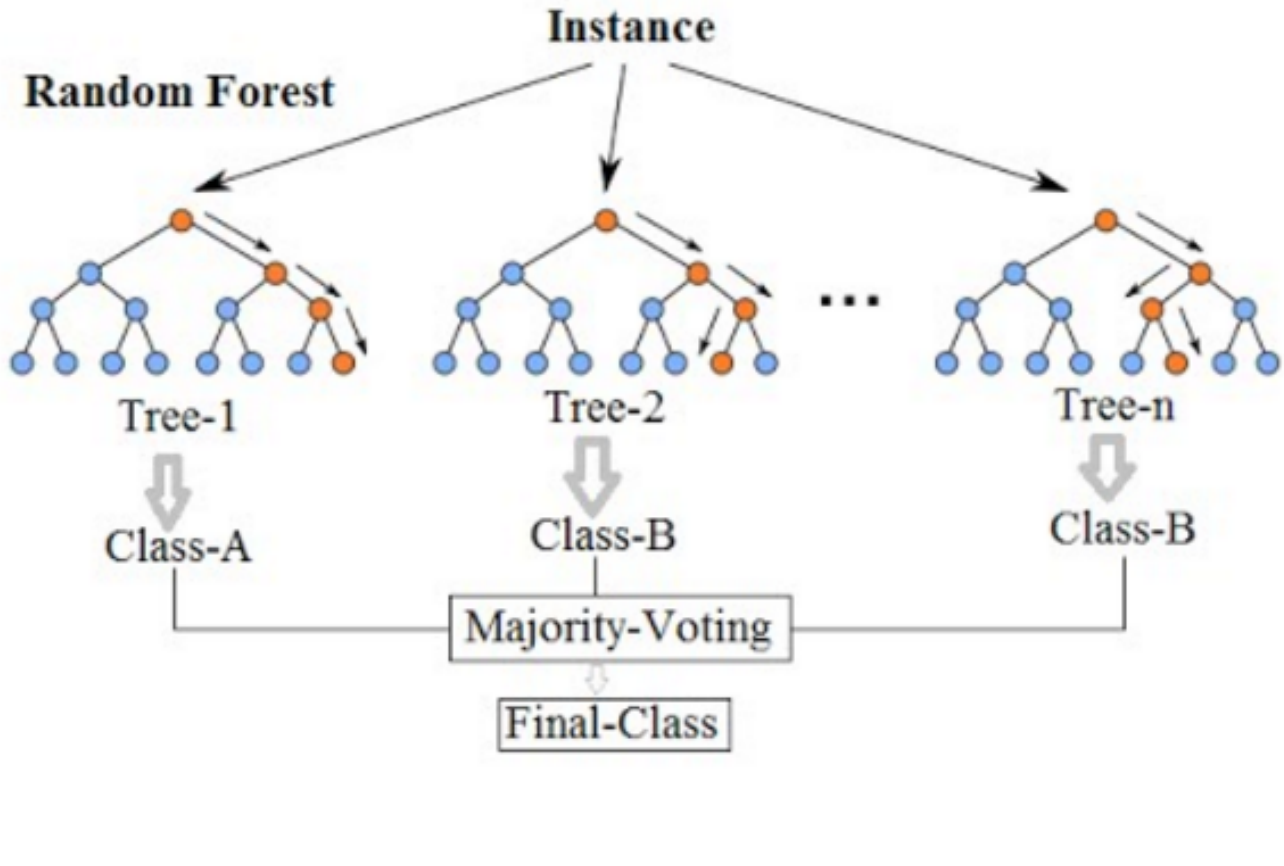
- Not a black box Algo – can visualize the logic
- Performs intrinsic feature ranking
  - Each node represents a feature(attribute), each link(branch) represents a decision(rule) and each leaf represents an outcome(categorical or continues value)



- Different types of algorithms
  - CART

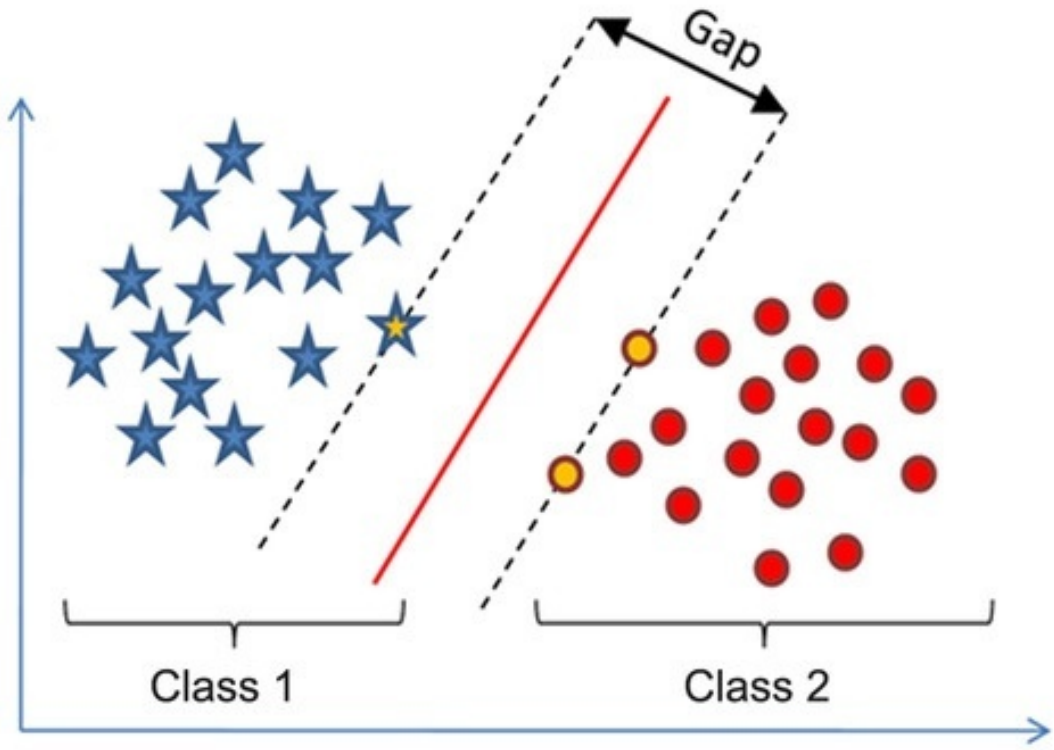
Random Decision Forest

- “Ensemble” of decision trees
- Train many classifiers on part of the data
- “Vote” for the right answer



Support Vector Machine

- Perform classification based on the features closest to the decision boundary
- Maximize the distance between the boundary and the “support vectors”
- Many variants



HyperParameters – Knobs to tune..

- Define higher level concepts about the model such as complexity, or capacity to learn.
- Cannot be learned directly from the data in the standard model training process and need to be predefined.
- Can be decided by setting different values, training different models, and choosing the values that test better
- Some examples of hyperparameters:
  - Number of leaves or depth of a tree
  - Number of latent factors in a matrix factorization
  - Learning rate (in many models)
  - Number of hidden layers in a deep neural network
  - Number of clusters in a k-means clustering

Random Forest – Hyper parameter

- n_estimators = number of trees in the foreset
- max_features = max number of features considered for splitting a node
- max_depth = max number of levels in each decision tree
- min_samples_split = min number of data points placed in a node before the node is split
- min_samples_leaf = min number of data points allowed in a leaf node
- bootstrap = method for sampling data points (with or without replacement)

```
{'bootstrap': [True, False],
 'max_depth': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, None],
 'max_features': ['auto', 'sqrt'],
 'min_samples_leaf': [1, 2, 4],
 'min_samples_split': [2, 5, 10],
 'n_estimators': [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800,
2000]}
```

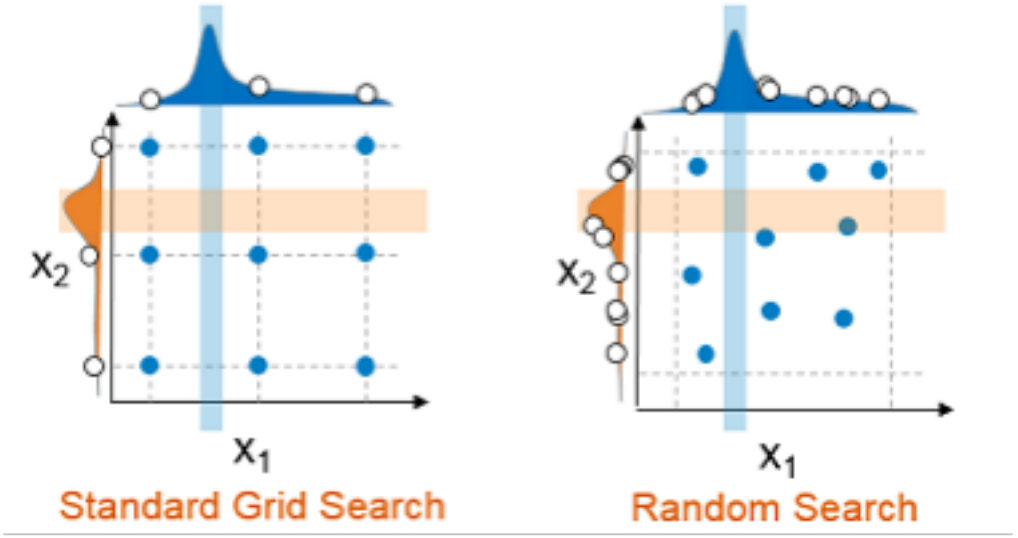
How to achieve optimized hyper parameter ?


```
rf_random.best_params_

{'bootstrap': True,
 'max_depth': 70,
 'max_features': 'auto',
 'min_samples_leaf': 4,
 'min_samples_split': 10,
 'n_estimators': 400}
```

HyperParameter Optimization

Objective: Find the ideal value(s) of the parameters of your algorithm to maximize the evaluation criterion



Grid Search

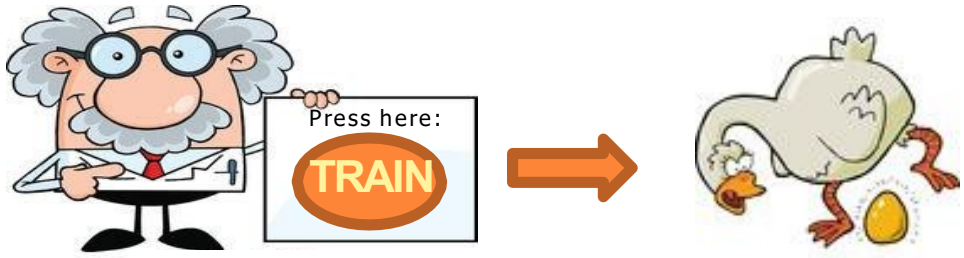
- Model – X
- Hyperparameters – a1,a2,a3
  - a1={0,1,2,3,4,5}
  - a2=[10,20,30,40,50,60]
  - a3 =[105,105,110,115,120,125]
- start with the combination of [0,10,105], and it will end with [5,60,125].
- It will also go through all the intermediate combinations - *computationally very expensive*.

Feedback and Model Deployment

- Machine Learning is an iterative process
- The model is re-trained many times till validation performance is satisfactory
- What changes in future iterations?
  - Preprocessing
  - Features
  - Choice of algorithm
  - Choice of parameters
  - Training data
- Testing performance indicates estimated real-world performance of the model
- The testing performance is the final reported value

ML EVALUATION BASICS: THE GOLDEN RULE

- Creating ML models is easy.



- Creating **good** ML models is not that easy.
  - Especially if we are not crystal clear about the criteria to tell how **good** our models are!

- So, good for what?

TIP

ML models should perform well during **deployment**.

ML EVALUATION BASICS: THE GOLDEN RULE

- We need performance metrics and evaluation procedures that best match the deployment conditions.

- Classification, regression, clustering, association rules, … use different metrics and procedures.

- Estimating how good a model is crucial:

TIP

Golden rule: never overstate the performance that a ML model is expected to have during deployment because of good performance in optimal "laboratory conditions"

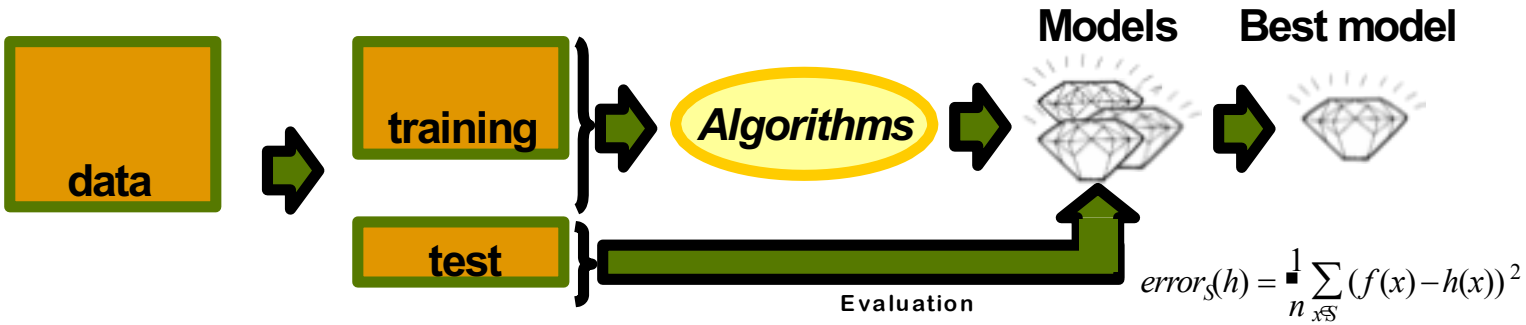
ML EVALUATION BASICS: THE GOLDEN RULE

- Caveat: Overfitting and underfitting

  - In predictive tasks, the golden rule is simplified to:

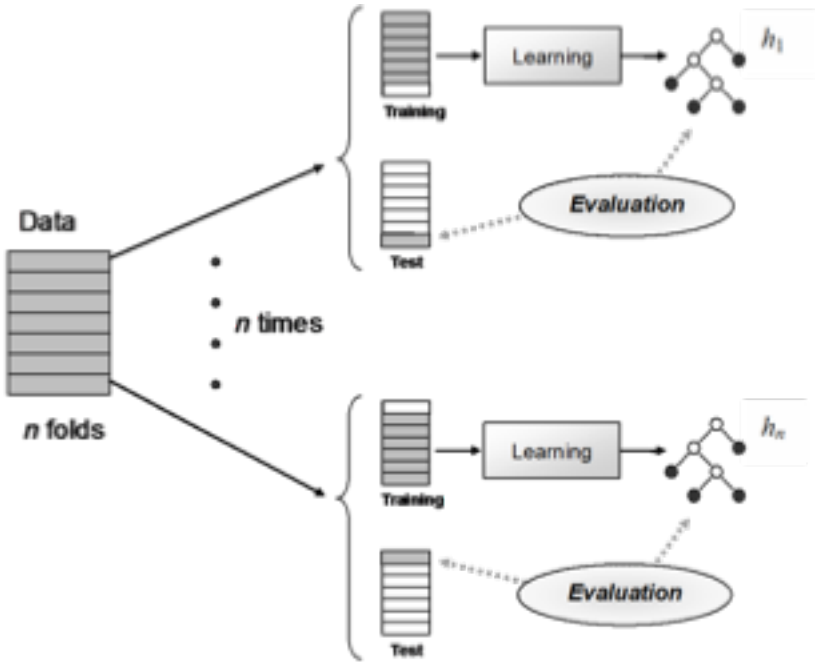
Golden rule for predictive tasks:
Never use the same examples for training the model and evaluating it

TIP



ML EVALUATION BASICS: THE GOLDEN RULE

- Caveat: What if there is not much data available?
  - Bootstrap or cross-validation



- We take all possible combinations with *n*–1 for training and the remaining fold for test.
- The error (or any other metric) is calculated *n* times and then averaged.
- A final model is trained with all the data.

TIP

No need to use cross-validation for large datasets

TEST VS. DEPLOYMENT: CONTEXT CHANGE

- Is this enough?

Testing conditions (lab)

Deployment conditions (production)

Context is everything

- Caveat: the simplified golden rule assumes that the context is the same for testing conditions as for deployment conditions.

ROC ANALYSIS

- The context or skew (the class distribution and the costs of each error) determines classifier goodness.

  - Caveat:
    - In many circumstances, until deployment time, we do not know the class distribution and/or it is difficult to estimate the cost matrix.
      - E.g. a spam filter.
    - But models are usually learned before.

  - SOLUTION:
    - ROC (Receiver Operating Characteristic) Analysis.

ROC ANALYSIS

- TO understand ROC curves, it is helpful to get a grasp of sensitivity, specificity, positive preditive value and negative predictive value:

	Disease				
Test	**Present**	**n**	**Absent**	**n**	**Total**
Positive	True Positive (TP)	*a*	False Positive (FP)	*c*	*a + c*
Negative	False Negative (FN)	*b*	True Negative (TN)	*d*	*b + d*
Total		*a + b*		*c + d*	

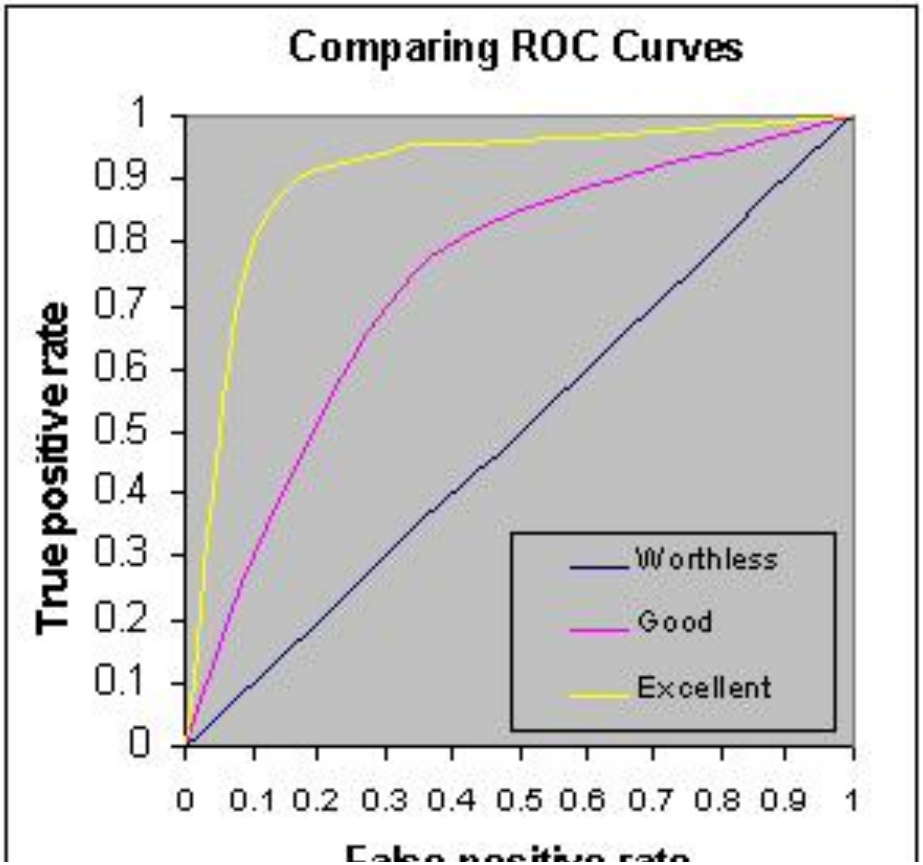
- TP=True Positive: cases with the disease correctly classified as positive
- FN= False Negative: cases with the disease incorrectly classified as negative
- TN= True Negative: cases without the disease correctly classified as negative
- FP= False Positive: cases without the disease incorrectly classified as positive

Sensitivity	$\frac{a}{a+b}$	**Specificity**	$\frac{d}{c+d}$

ROC ANALYSIS

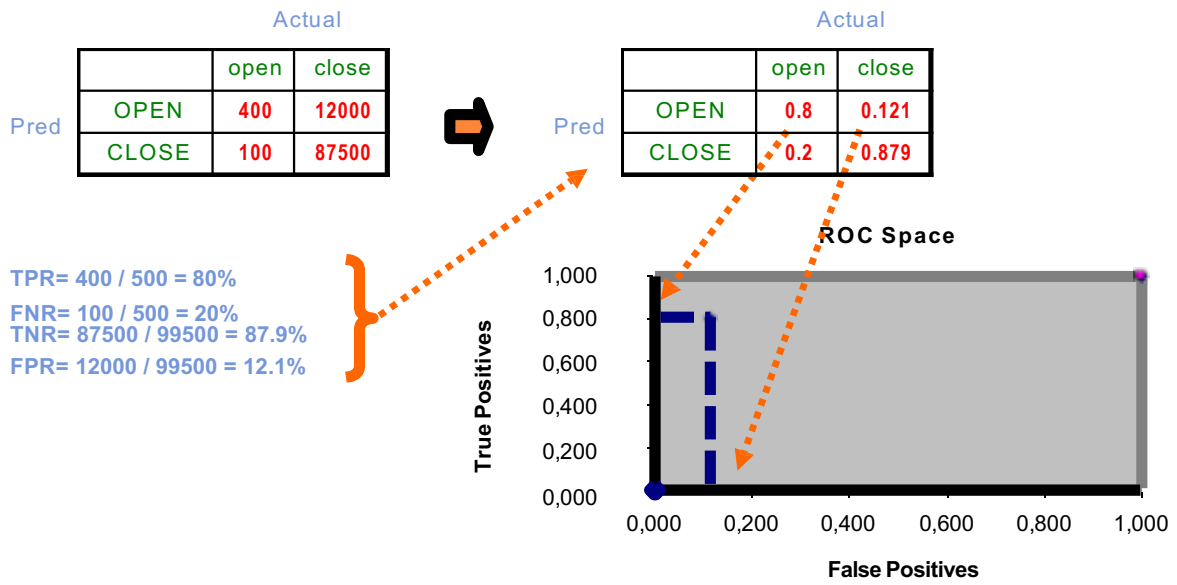
The closer the curve follows the left side border and the top border, the more accurate the test.

The closer the curve is to the 45-degree diagonal, the less accurate the test.



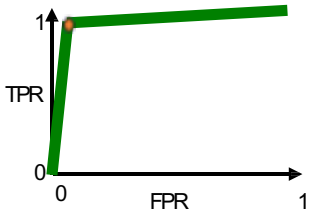
ROC ANALYSIS

- The ROC Space
  - Using the normalised terms of the confusion matrix:
    - TPR, FNR, TNR, FPR:

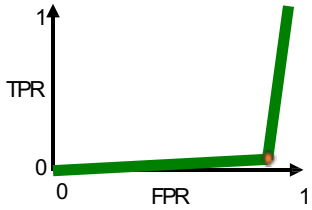


ROC ANALYSIS

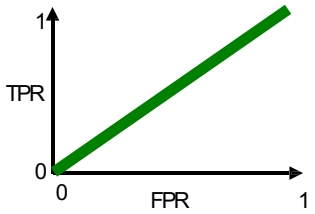
- Good and bad classifiers



- Good classifier.
  - High TPR.
  - Low FPR.



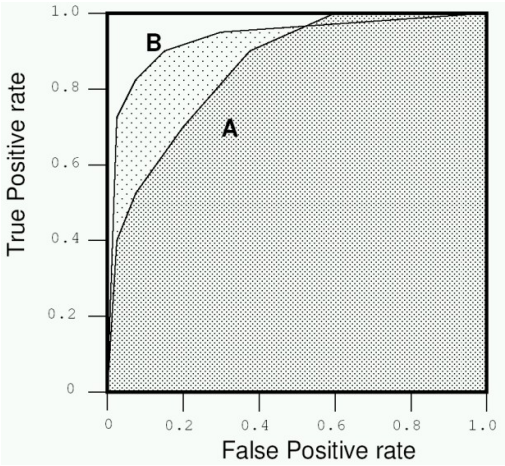
- Bad classifier.
  - Low TPR.
  - High FPR.



- Bad classifier (more realistic).

METRICS FOR A RANGE OF CONTEXTS

- What if we want to select just one soft classifier?
  - The classifier with greatest Area Under the ROC Curve (AUC) is chosen.



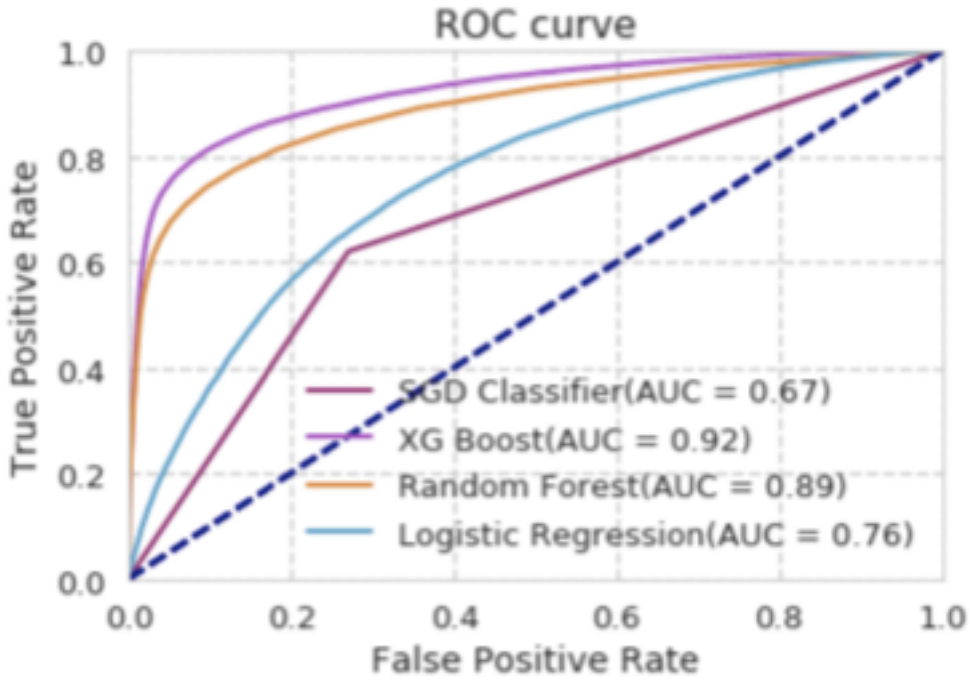
AUC is useful but it is always better to draw the curves and choose depending on the operating condition.

TIP

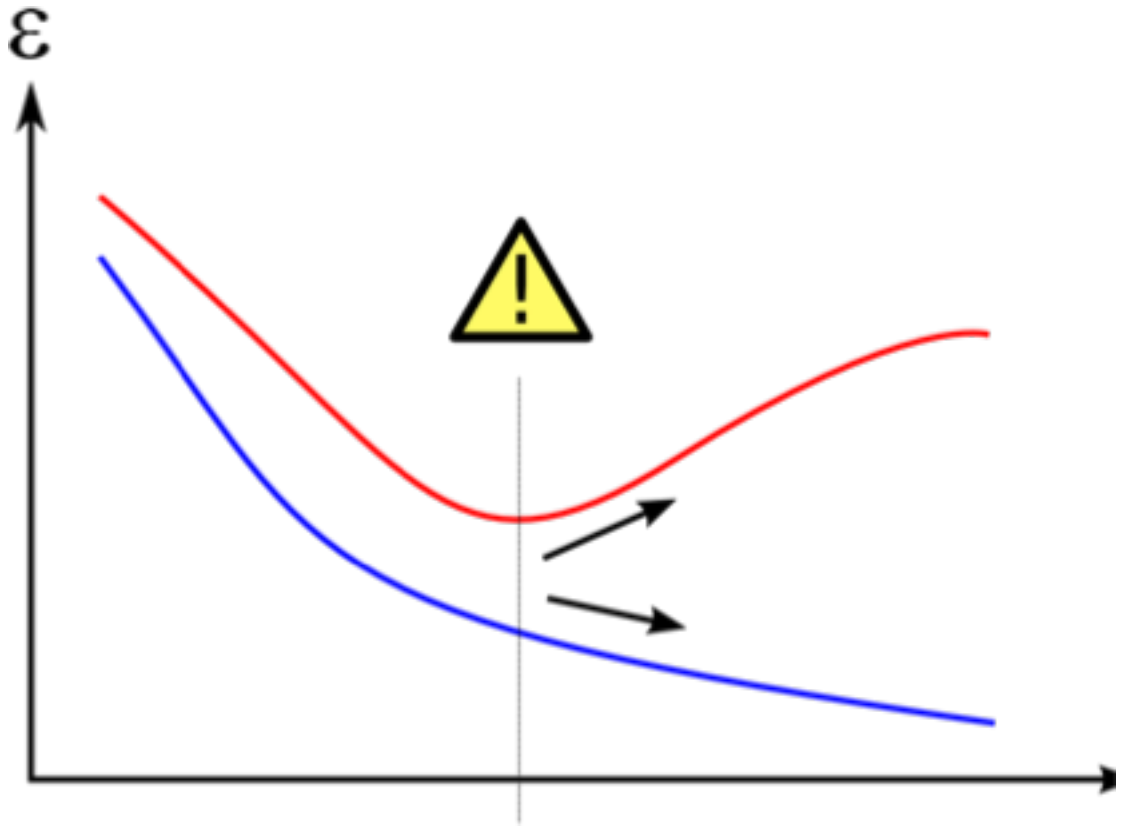
AUC does not consider calibration. If calibration is important, use other metrics, such as the Brier score.

TIP

AUC ROC Score

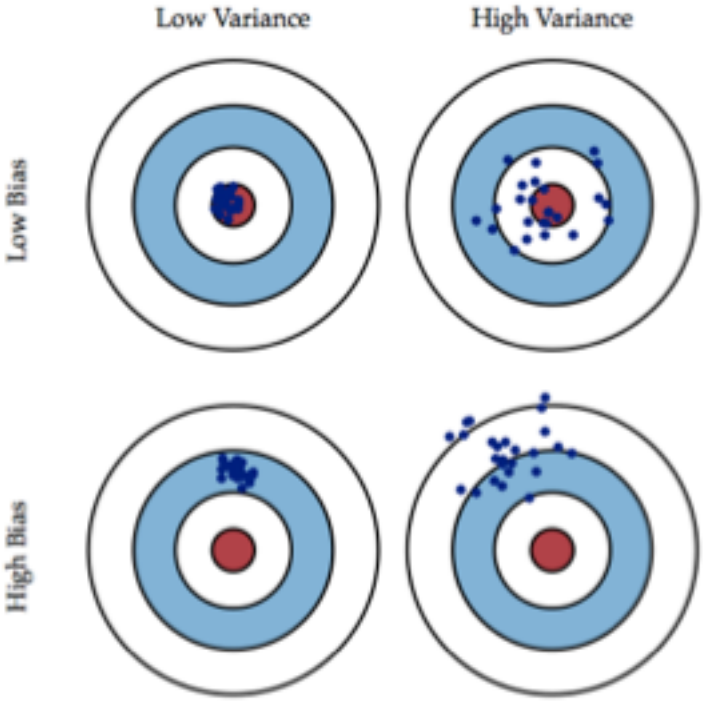


Underfitting and Overfitting

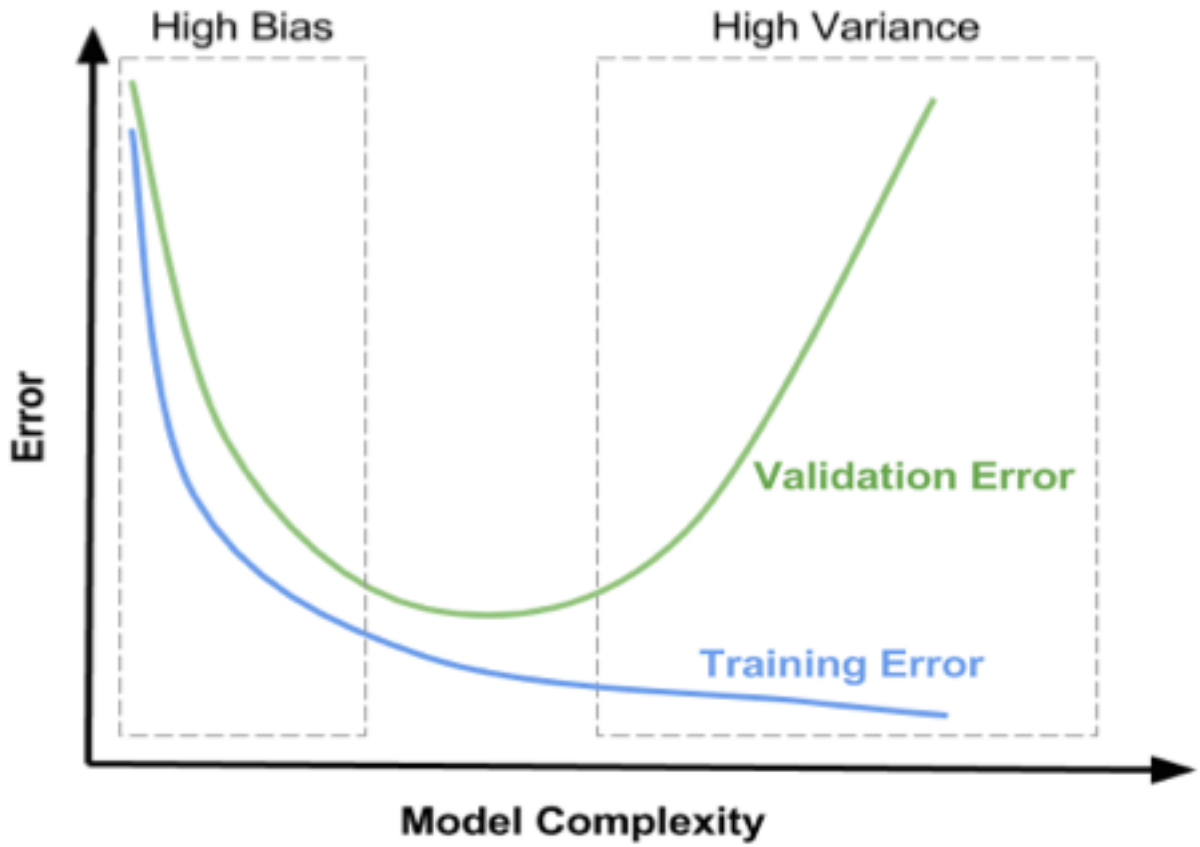


Underfitting and Overfitting

- **Underfitting:**
  - High bias
  - High training/validation/test error
  - Low model complexity
  - Increase features
  - Choose a more expressive model
- **Overfitting:**
  - High variance
  - Low training error, high validation/test error
  - High model complexity, low generalizability
  - Increase training data, lower feature dimension
  - Regularization (making the model simpler)

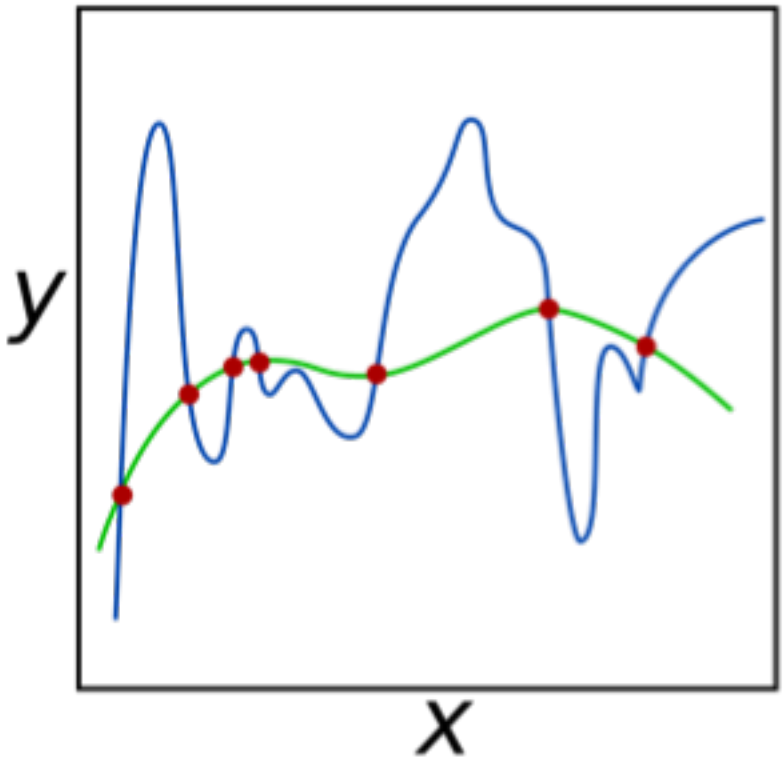


Error – Bias / Variance tradeoff



Regularization

- Reduce the decision boundary complexity
- Reduce overfitting on training data
- Increase real world performance
- Existing regularization methods: L-1, L-2, Elastic Net, dropout, dropconnect (deep learning)



Regularization: Example

- Overfit decision rule:
  - If X <= 3.7, class = 1
- Regularized decision rule:
  - If X <=3 class = 1
- Sacrifice training performance to obtain a more generalizable decision rule that can potentially scale better to unseen data
- Trade-off

Class Label	Feature (X)
1	2.7
2	3.8
1	2.8
2	5.5
1	3.7
1	3.0
1	2.9
2	3.9

Feature Engineering

- "A model is only as good as the data"
- How to improve the data?
  - Higher dimension of features (polynomial, interactive terms, kernel trick)
  - Specialized features (domain knowledge)
  - More feature sources (external data)
- Some tools are available to help in feature engineering (https://docs.featuretools.com/getting_started/install.html)
- The "art" aspect of machine learning

Logistic regression - Hyperparameter

- C = 1/λ, where λ is the regularisation parameter
- Smaller values of C specify stronger regularisation

```
LogisticRegression(C=0.0001, class_weight=None, dual=False,
          fit_intercept=True, intercept_scaling=1, max_iter=100,
          multi_class='ovr', n_jobs=-1, penalty='l2', random_state=None,
          solver='liblinear', tol=0.0001, verbose=0, warm_start=False) 0.712918040912
```

```
SGDClassifier(alpha=0.1, average=False, class_weight=None, epsilon=0.1,
       eta0=0.0, fit_intercept=True, l1_ratio=0.15,
       learning_rate='optimal', loss='log', max_iter=None, n_iter=1000,
       n_jobs=-1, penalty='l2', power_t=0.5, random_state=None,
       shuffle=True, tol=None, verbose=0, warm_start=False) 0.712416071628
```

Random Forest – Hyper parameter

- **n_estimators : -** number of trees you want to build
- **min_sample_leaf -** No. of end nodes in a tree
- Max_depth - how deep you want to make your trees.
- Max_feature - maximum number of features Random Forest is allowed to try in individual tree
- Bootstrap - bootstrap method can be used to estimate a quantity of a population – statistical sampling
- Criteria – Gini – minimize misclassification , Entropy – exploratory

```
{'bootstrap': [True, False],
 'max_depth': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, None],
 'max_features': ['auto', 'sqrt'],
 'min_samples_leaf': [1, 2, 4]
```

Backup

Spearman Rank Correlation: Worked Example (No Tied Ranks)

The formula for the Spearman rank correlation coefficient when there are no tied ranks is:

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)}$$

Sample Question:
The scores for nine students in physics and math are as follows:
Physics: 35, 23, 47, 17, 10, 43, 9, 6, 28
Mathematics: 30, 33, 45, 23, 8, 49, 12, 4, 31
Compute the student's ranks in the two subjects and compute the Spearman rank correlation.

Step 1: Find the ranks for each individual subject. I used the Excel rank function to find the ranks. If you want to rank by hand, order the scores from greatest to smallest; assign the rank 1 to the highest score, 2 to the next highest and so on:

Physics	Rank	Math	Rank
35	3	30	5
23	5	33	3
47	1	45	2
17	6	23	6
10	7	8	8
43	2	49	1
9	8	12	7
6	9	4	9
28	4	31	4

Step 2: Add a third column, d, to your data. The d is the difference between ranks. For example, the first student's physics rank is 3 and math rank is 5, so the difference is 3 points. In a fourth column, square your d values.

Physics	Rank	Math	Rank	d	d squared
35	3	30	5	2	4
23	5	33	3	2	4
47	1	45	2	1	1
17	6	23	6	0	0
10	7	8	8	1	1
43	2	49	1	1	1
9	8	12	7	1	1
6	9	4	9	0	0
28	4	31	4	0	0

Step 4: Sum (add up) all of your d-squared values.
4 + 4 + 1 + 0 + 1 + 1 + 1 + 0 + 0 = 12. You'll need this for the formula (the $\Sigma\ d^2$ is just "the sum of d-squared values").

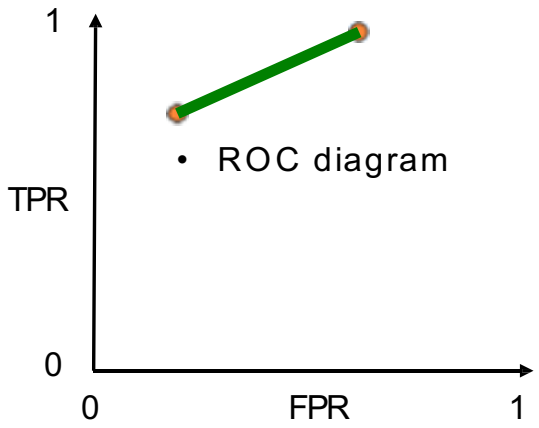
Step 5: Insert the values into the formula. These ranks are not tied, so use the first formula:

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)}$$

= 1 – (6*12)/(9(81-1))
= 1 – 72/720
= 1-0.1
= 0.9
The Spearman Rank Correlation for this set of data is 0.9.

ROC ANALYSIS - Bkup

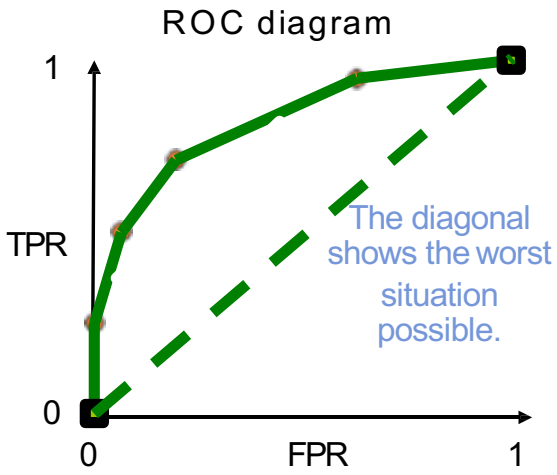
- The ROC "Curve": "Continuity".



- Given two classifiers:
  - ✓ We can construct any "intermediate" classifier just randomly weighting both classifiers (giving more or less weight to one or the other).
  - ✓ This creates a "continuum" of classifiers between any two classifiers.

ROC ANALYSIS - Bkup

- The ROC "Curve": Construction



- Given several classifiers:
  - ✓ We construct the convex hull of their points (FPR,TPR) as well as the two trivial classifiers (0,0) and (1,1).
  - ✓ The classifiers below the ROC curve are discarded.
  - ✓ The best classifier (from those remaining) will be selected in application time…

TIP

We can discard those which are below because there is no context (combination of class distribution / cost matrix) for which they could be optimal.

ROC ANALYSIS - Bkup

- Crisp and Soft Classifiers:
  - A "hard" or "crisp" classifier predicts a class between a set of possible classes.
    - Caveat: crisp classifiers are not versatile to changing contexts.

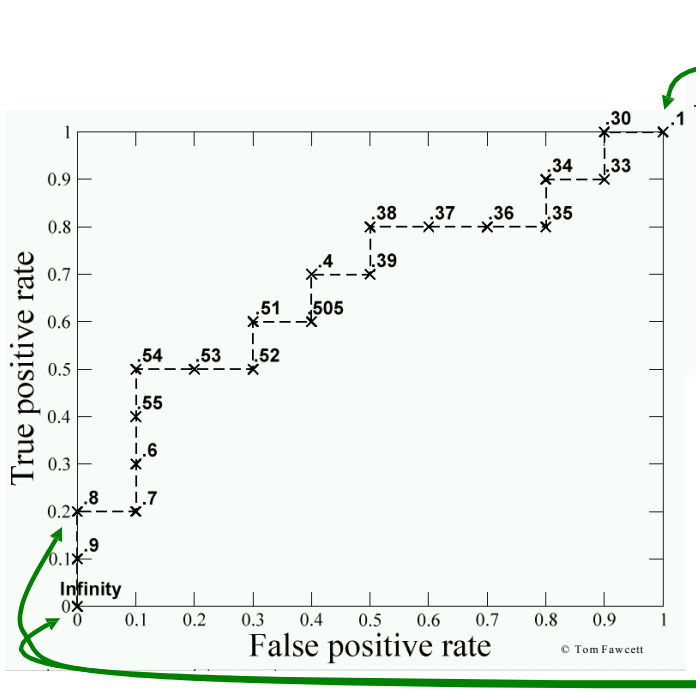
TIP

Soft or scoring classifiers can be reframed to each context.

  - A "soft" or "scoring" (probabilistic) classifier predicts a class, but accompanies each prediction with an estimation of the reliability (confidence) of each prediction.
    - Most learning methods can be adapted to generate soft classifiers.
  - A soft classifier can be converted into a crisp classifier using a threshold.
    - Example: "if score > 0.7 then class A, otherwise class B".
    - With different thresholds, we have different classifiers, giving more or less relevance to each of the classes

ROC ANALYSIS - Bkup

- ROC Curve of a Soft Classifier:
  - We can consider each threshold as a different classifier and draw them in the ROC space. This generates a curve…



Actual Class

Predicted Class

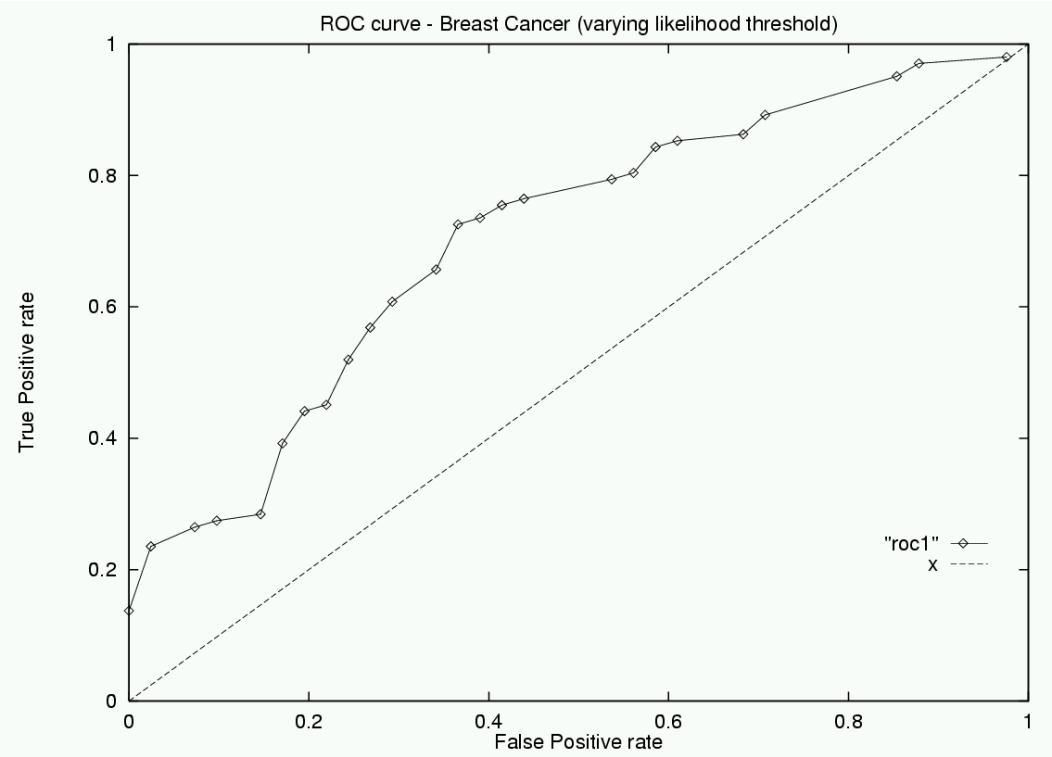
Inst#	Class	Score
1	p	.9
2	p	.8
3	n	.7
4	p	.6
5	p	.55
6	p	.54
7	n	.53
8	n	.52
9	p	.51
10	n	.505
11	p	.4
12	n	.39
13	p	.38
14	n	.37
15	n	.36
16	n	.35
17	p	.34
18	n	.33
19	p	.30
20	n	.1

21

We have a “curve” for just one soft classifier

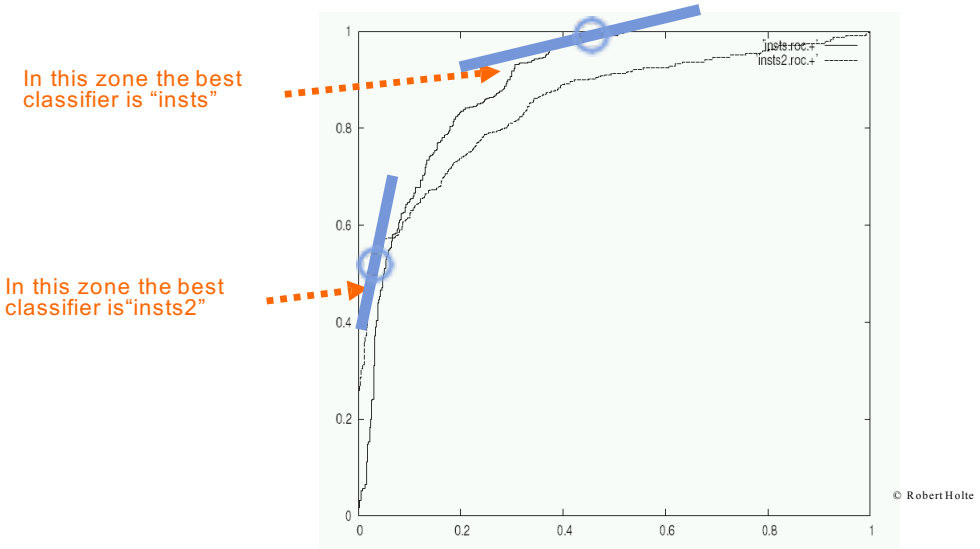
ROC ANALYSIS – Bkup

- ROC Curve of a soft classifier.



ROC ANALYSIS - Bkup

- ROC Curve of a soft classifier.



TIP

We must preserve the classifiers that have at least one “best zone” (dominance) and then behave in the same way as we did for crisp classifiers.