COS80027

Machine Learning

Feature Engineering

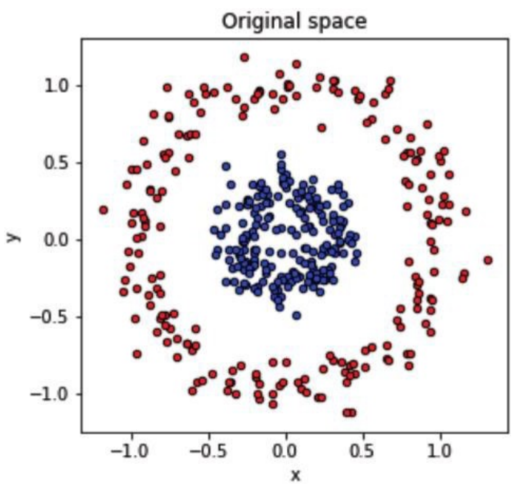
Lecturer: Dr. Dana Rezazadegan

SWINBURNE UNIVERSITY OF TECHNOLOGY

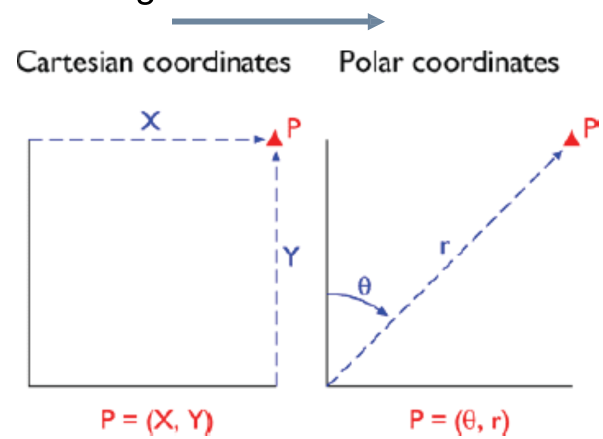
Features are the key to the success of ML

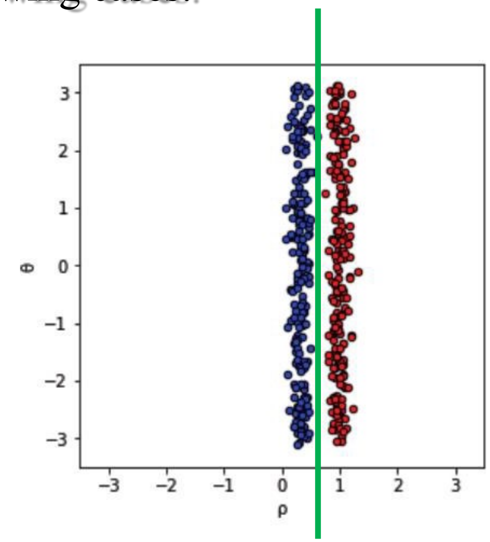
It's often said that *data is the fuel of ML*. However, data is just like the crude oil of ML which means it has to be refined into **features** to be useful for training a ML model. Without appropriate features, a ML system may not work!

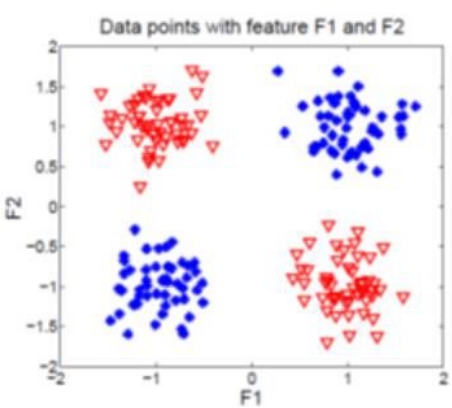
A linear classification model (a.k.a. classifier) does not work in the following cases:



Crafting new features **θ** and **r**

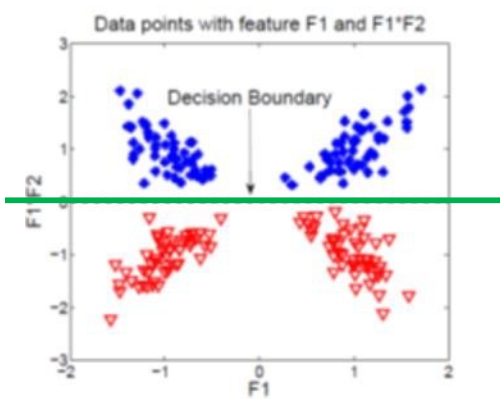




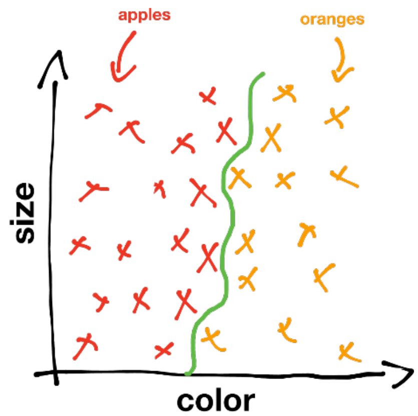


Crafting a new feature F3
F3 = F1*F2

Using F1 and F3



Features are the key to the success of ML (cont'd)



- Using one single feature "size" cannot distinguish between apples and oranges.
- Adding the second feature "sweetness" may increase distinguishability but still cannot separate them well.
- Adding the third feature "shape" does not help at all.
- Adding the fourth feature "colour" can lead to very good separation.

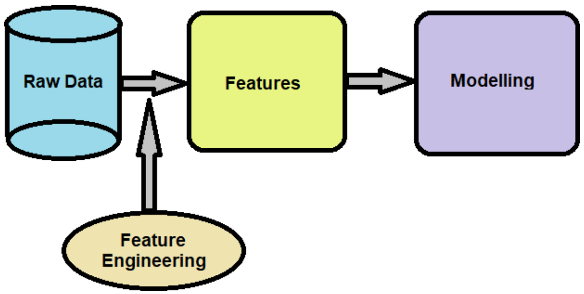
Maybe the "colour" feature by itself is enough good?

Feature engineering

Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work. (https://en.wikipedia.org/wiki/Feature_engineering)

Coming up with features is difficult, time-consuming, requires expert knowledge. "Applied machine learning" is basically feature engineering.

— Andrew Ng, *Machine Learning and AI via Brain simulations*[1]



It typically aims to transform raw data/features into relevant (good) features which are:

- **Informative**, i.e., it provides useful information for the ML model.
- **Discriminative**, i.e., it helps the ML model to distinguish training examples.
- **Non-redundant**, i.e., it does not say the same thing as another feature.

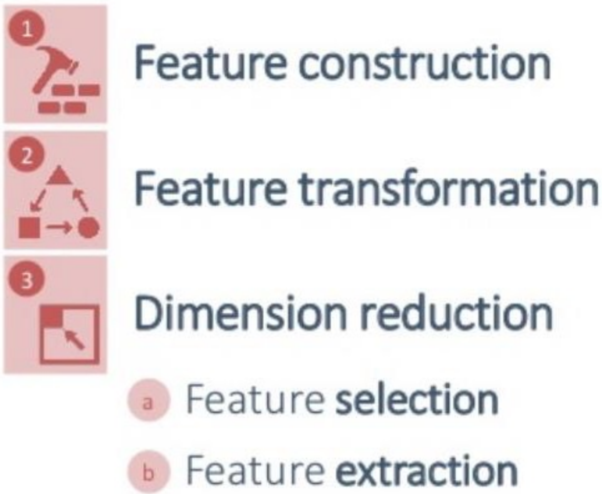
resulting in the improved model performance.

Trends: manual feature engineering to automated feature learning (deep learning).

SWINBURNE UNIVERSITY OF TECHNOLOGY

Feature engineering techniques

Major categories of feature engineering techniques:



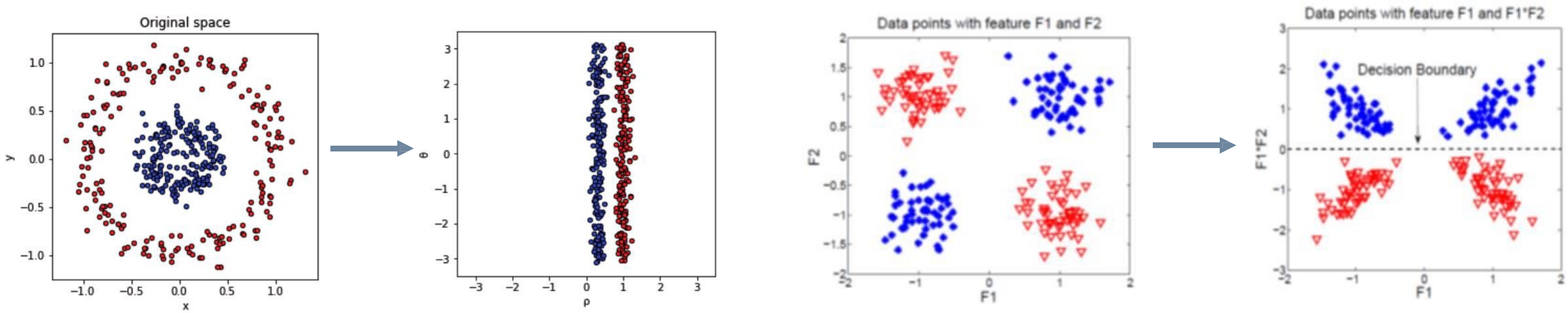
Sometimes, more data (in terms of features) needs to be acquired due to the intrinsic limitation of available data.

For example, if “size”, “sweetness” and “shape” are the only available features in data when classifying apples and oranges.

Feature construction

Feature construction is the process of *crafting* informative features that are useful for the task to be solved from raw data/features.

- Need domain expertise
- Key to the performance of the ML model



Example: Decompose a Date-Time

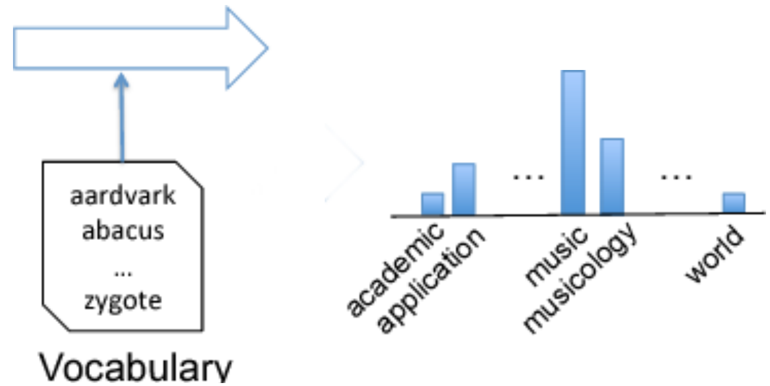
Same raw data	Different problems	Different features
2017-01-03 15:00:00	Predict how much hungry someone is 	"Hours elapsed since last meal": 2
2017-01-03 15:00:00	Predict the likelihood of a burglary 	"Night": 0 (numerical value for "False")

SWINBURNE UNIVERSITY OF TECHNOLOGY

Feature construction (cont'd)

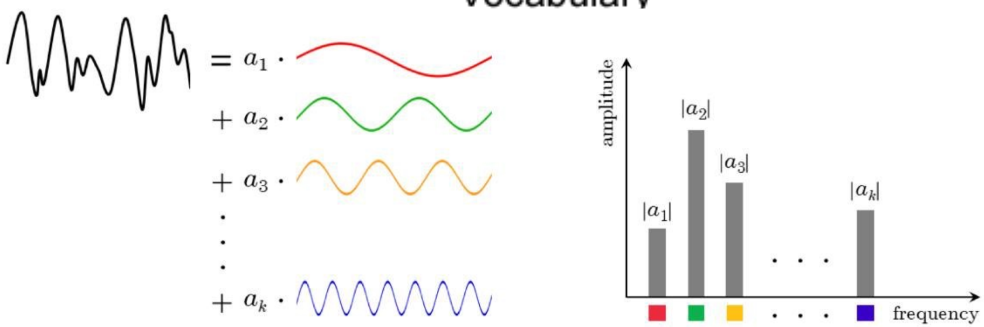
Text

Music information retrieval (MIR) is the interdisciplinary science of retrieving information from music. MIR is a small but growing field of research with many real-world applications...



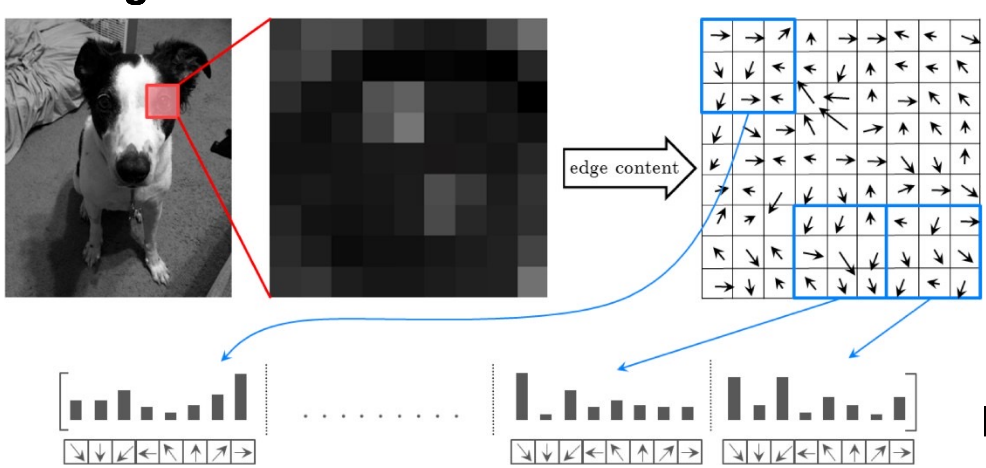
Vocabulary histogram

Audio



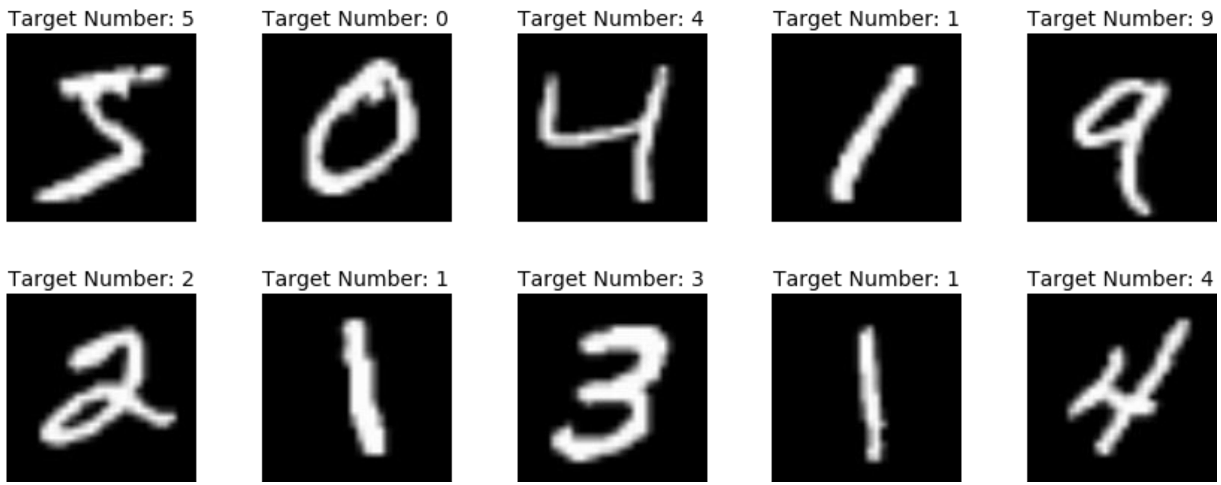
Frequency histogram (spectrum)

Image

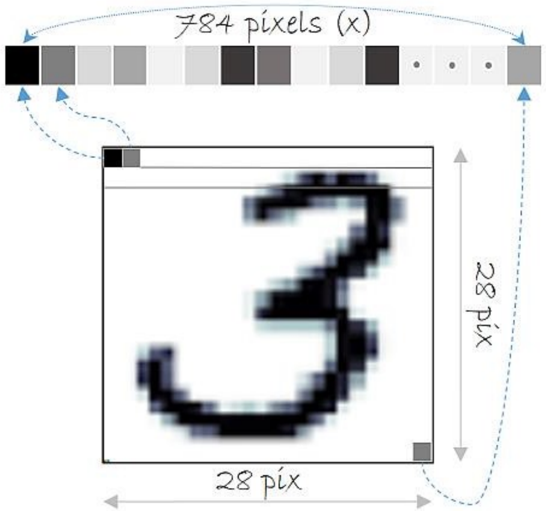


Edge histogram

Feature construction (cont'd)



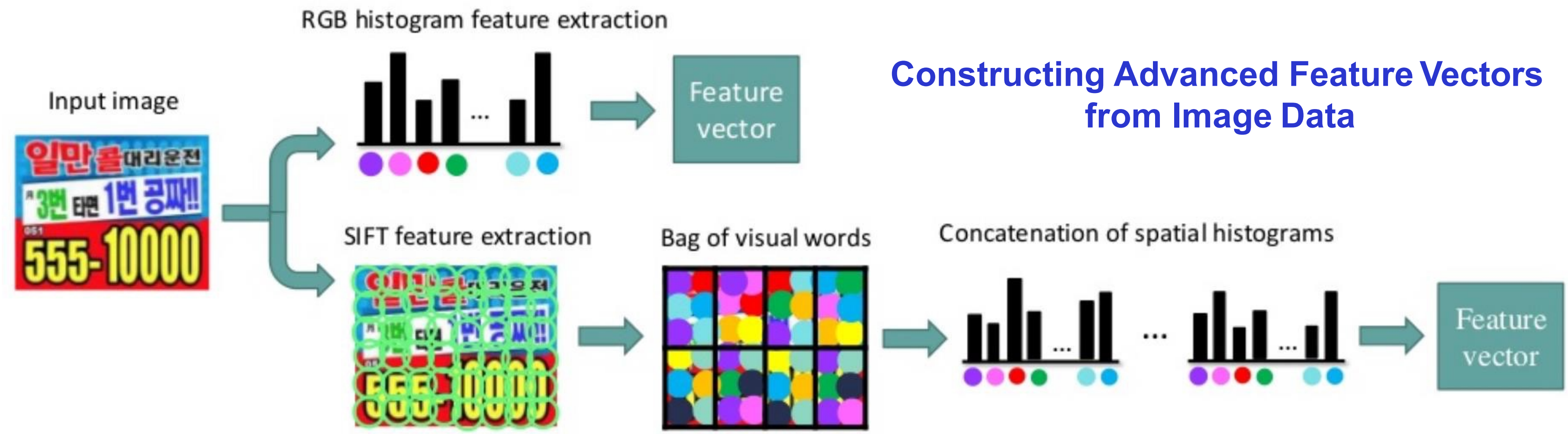
A sample of the images in the MNIST dataset



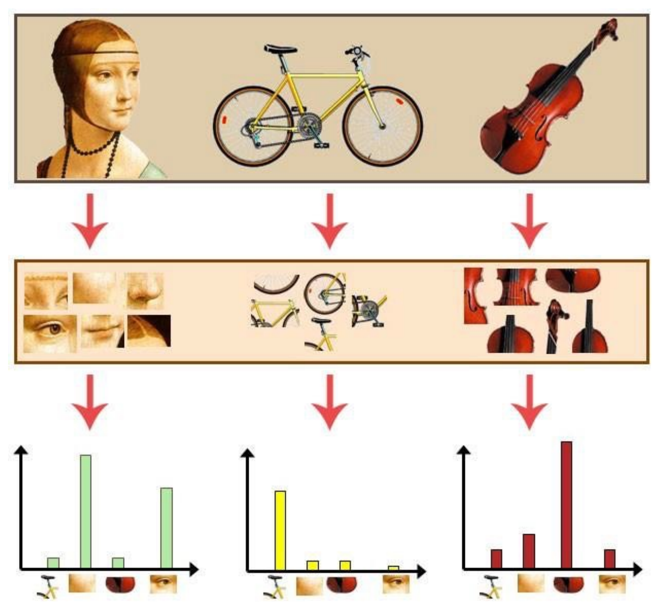
Constructing Simple Feature Vectors from Image Data

Feature construction (cont’d)

Constructing Advanced Feature Vectors from Image Data



Bag of Visual Words



Feature transformation

Feature transformation is the process of transforming a feature into a new one via a specific function.

What we've already learned:

- **Binning (discretisation):** transforming a numerical feature into a categorical (ordinal) one.

df1

	Name	Score
0	George	63
1	Andrea	48
2	micheal	56
3	maggie	75
4	Ravi	32
5	Xien	77
6	Jalpa	85
7	Tyieren	22

```
''' binning or bucketing with range'''

bins = [0, 25, 50, 75, 100]
df1['binned'] = pd.cut(df1['Score'], bins)
print (df1)
```

	Name	Score	binned
0	George	63	(50, 75]
1	Andrea	48	(25, 50]
2	micheal	56	(50, 75]
3	maggie	75	(50, 75]
4	Ravi	32	(25, 50]
5	Xien	77	(75, 100]
6	Jalpa	85	(75, 100]
7	Tyieren	22	(0, 25]

```
''' binning or bucketing with labels'''

bins = [0, 25, 50, 75, 100]
labels =[1,2,3,4]
df1['binned'] = pd.cut(df1['Score'], bins,labels=labels)
print (df1)
```

	Name	Score	binned
0	George	63	3
1	Andrea	48	2
2	micheal	56	3
3	maggie	75	3
4	Ravi	32	2
5	Xien	77	4
6	Jalpa	85	4
7	Tyieren	22	1

- **Encoding (i.e. creating dummy variables)**

Transforming categorical features into numerical vectors so that you can do vector operations (such as calculating the distance) on them.

df1

	country
0	russia
1	germany
2	australia
3	korea
4	germany

```
pd.get_dummies(df1, prefix = 'country', drop_first = True)
```

	country_germany	country_korea	country_russia
0	0	0	1
1	1	0	0
2	0	0	0
3	0	1	0
4	1	0	0

Feature transformation (cont'd)

- **Feature scaling** changes the range of a feature into the one with the specified property, aiming to handle highly varying magnitudes/values/units.

Why bother feature scaling?

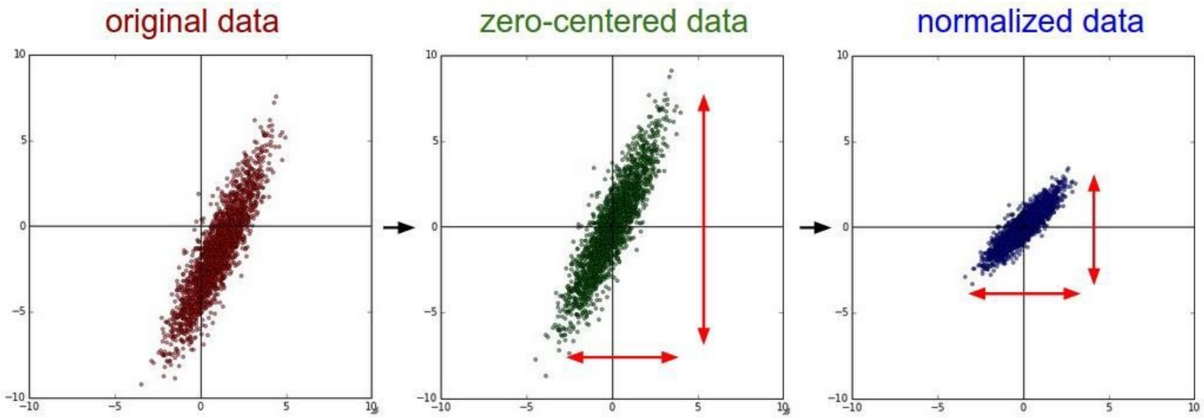
- Most often, different features in the data might be have different magnitudes.
- Many ML algorithms use the Euclidean distance between data point in their computation. Accordingly, having two features with different ranges will let the feature with the bigger range dominate the algorithm.

Weight:
165g

Price:
AU$2.25

Two common techniques:

1. **Standardization (a.k.a Z-score Normalization)** re-scales a feature to have the mean value of 0 and the standard deviation of 1.



$$z = \frac{x - \mu}{\sigma}$$

$\mu =$ Mean
$\sigma =$ Standard Deviation

Feature transformation (cont'd)

x

	Country	Age	Salary	Purchased
0	France	44	72000	0
1	Spain	27	48000	1
2	Germany	30	54000	0
3	Spain	38	61000	0
4	Germany	40	1000	1

```python
from sklearn import preprocessing

Standardisation = preprocessing.StandardScaler()

# Scaled feature
x_after_Standardisation = Standardisation.fit_transform(x)
```

```
After Standardisation :
 [[ 0.09536935  0.66527061]
 [-1.15176827 -0.43586695]
 [-0.93168516 -0.16058256]
 [-0.34479687  0.16058256]
 [-0.1980748  -2.59226136]
 [-0.56487998  0.02294037]
 [ 2.58964459 -0.25234403]
 [ 0.38881349  0.98643574]
 [ 0.53553557  1.16995867]
 [-0.41815791  0.43586695]]
```

2. **Min-Max Normalization** re-scales a feature to have the range between 0 and 1.

$$\mathbf{x}' = \frac{\mathbf{x} - \min(\mathbf{x})}{\max(\mathbf{x}) - \min(\mathbf{x})}$$

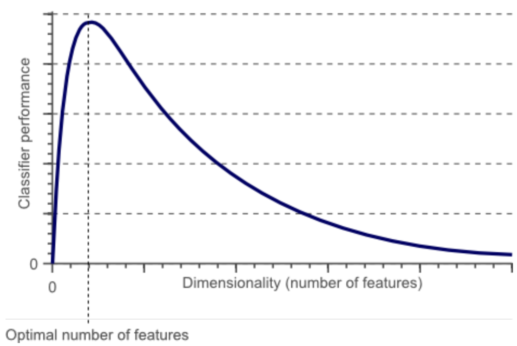
```python
min_max_scaler = preprocessing.MinMaxScaler(feature_range =(0, 1))

# Scaled feature
x_after_min_max_scaler = min_max_scaler.fit_transform(x)
```

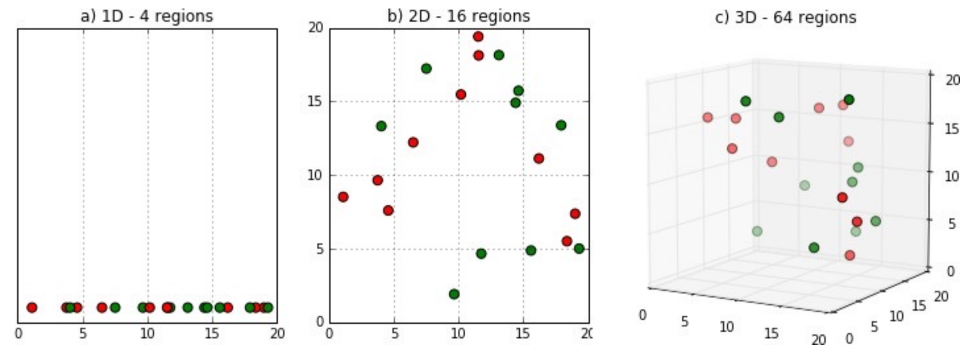
```
After min max Scaling :
 [[ 0.33333333  0.86585366]
 [ 0.          0.57317073]
 [ 0.05882353  0.64634146]
 [ 0.21568627  0.73170732]
 [ 0.25490196  0.        ]
 [ 0.15686275  0.69512195]
 [ 1.          0.62195122]
 [ 0.41176471  0.95121951]
 [ 0.45098039  1.        ]
 [ 0.19607843  0.80487805]]
```

Dimension reduction

Dimension reduction is the process of reducing the number of features used to build a ML model, with the purpose of keeping only relevant features.

It is typically used to overcome the curse of dimensionality issue which refers to various phenomena that arise when analysing and organizing data in high-dimensional spaces (often with hundreds or thousands of dimensions) that do not occur in low-dimensional settings.





The main benefits are:

- Faster computations
- Less storage space required
- Increased model performance
- Data visualization *(when reduced to 2D or 3D)*

- If we have more features than observations than we run the risk of massively overfitting our model.

- When we have too many features, observations become harder to cluster because large dimension causes every observation in the data set to appear equidistant from each other.

SWINBURNE UNIVERSITY OF TECHNOLOGY

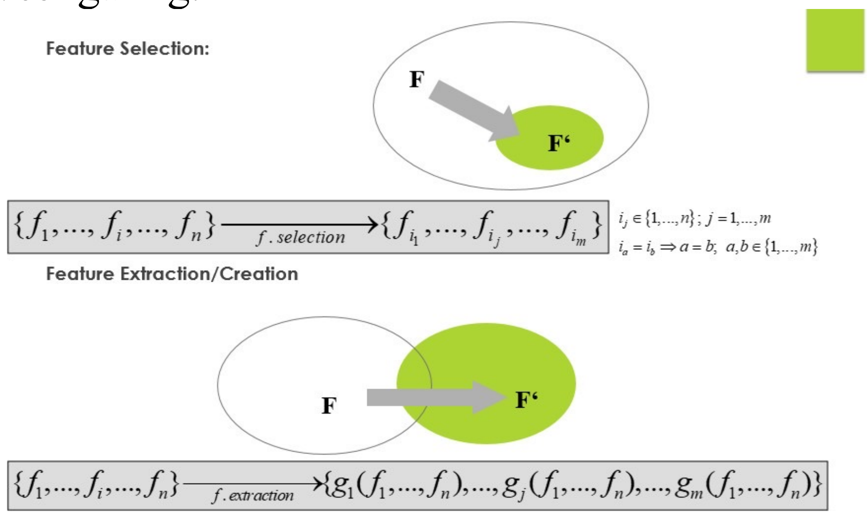
Dimension reduction (cont'd)

Two major categories of dimension reduction techniques:

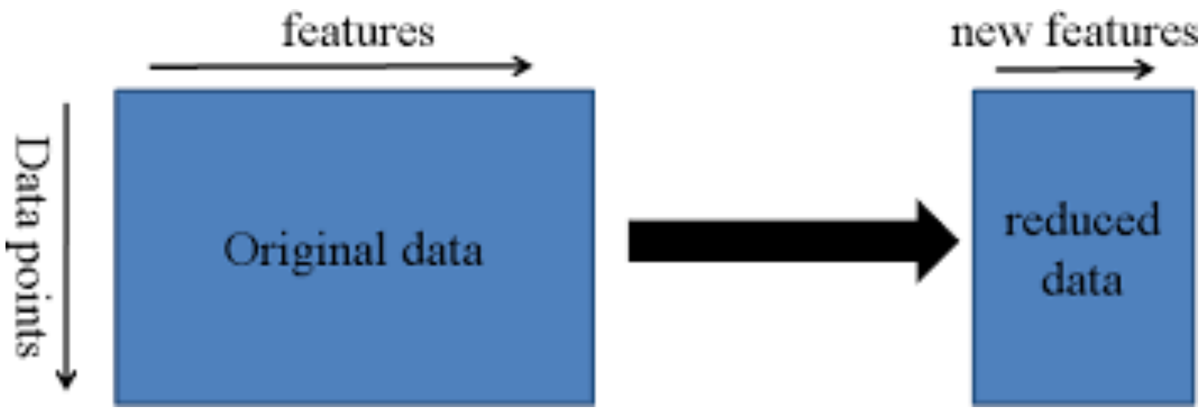
- **Feature extraction:** starting from an initial set of features and automatically building derived features that are more relevant.
- **Feature selection:** selecting a subset of more relevant features from all existing features.

Feature Extraction vs. Feature Selection

- Feature extraction may result in more relevant features than feature selection.
- Feature selection does not modify the original features but feature extract often does so. As a result, feature selection is more suitable for identifying the key factors which drive the phenomenon we are investigating.



Feature extraction

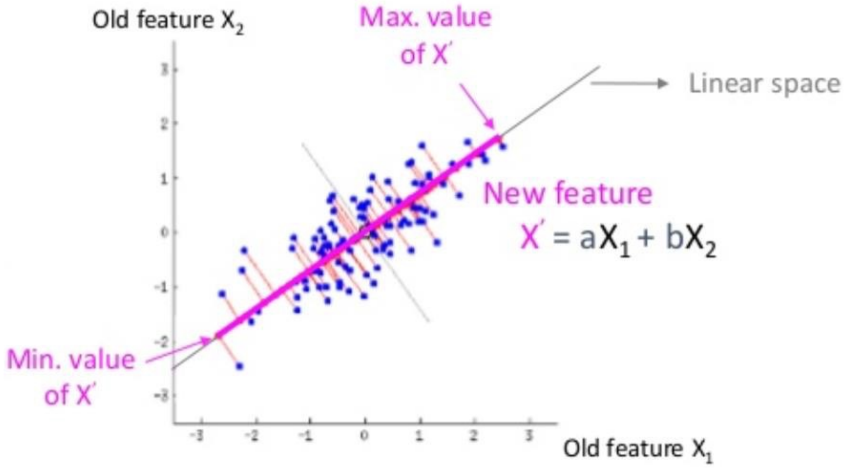


- **Principal component analysis**
- … (many other techniques)

Principal component analysis (PCA)

PCA is the most commonly used feature extraction technique.

- It makes an orthogonal projection on a *linear space* to determine new features, called principal components, that are linear combinations of the old features.

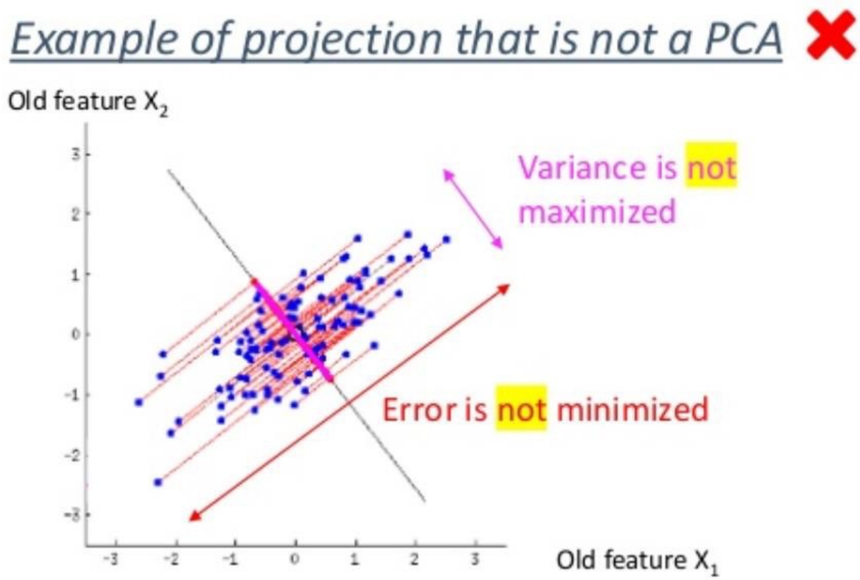


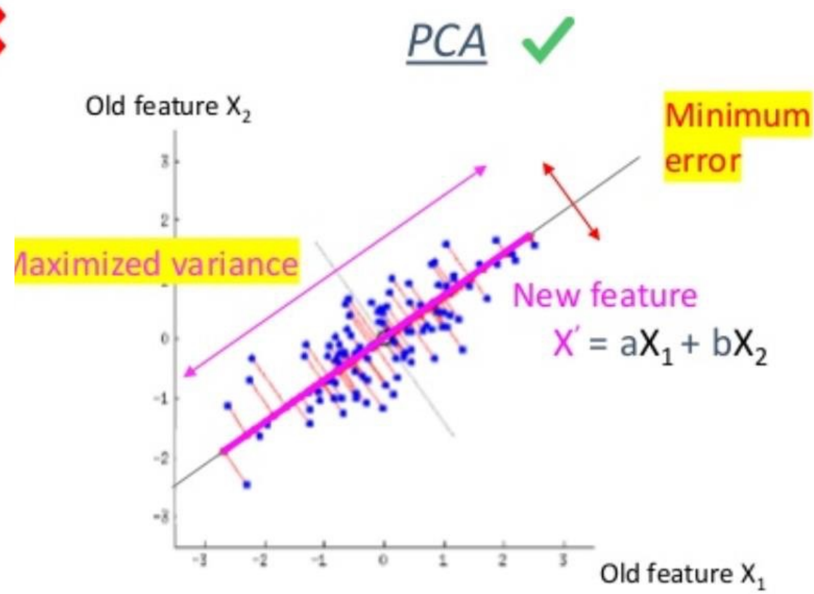
Example of reduction of two features into a single one

Principle of PCA

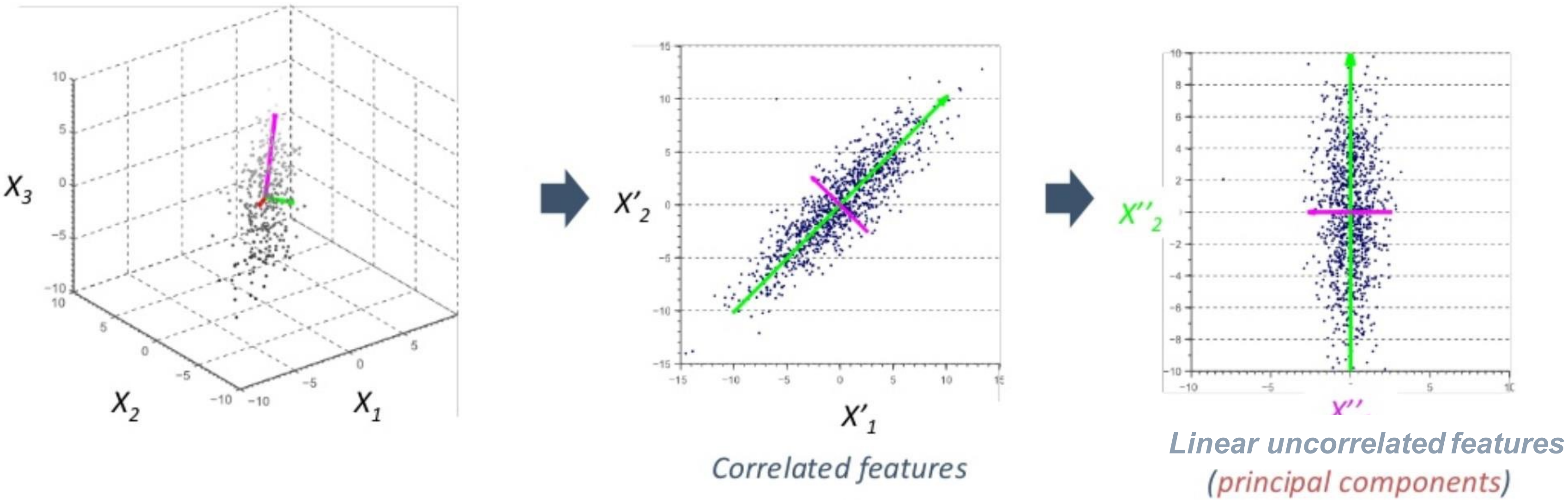
The principal component (PC) is built along an axis so that it is, as much as possible:

- Discriminative (its variance is maximized)
- Informative (the error to original values is minimized)
- Linearly uncorrelated with other PCs (i.e., non-redundant)





Principle of PCA (cont'd)

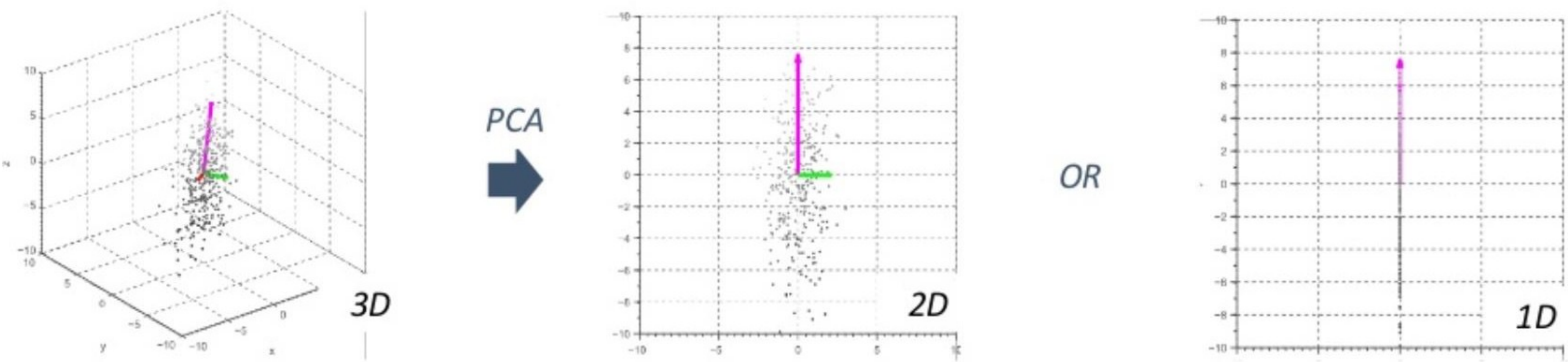


Example of reduction of three featuresinto 2 PCs

- By identifying which "directions" are most "important", PCA can compress the data into a smaller space by dropping the "directions" that are "least important".
- By projecting the data into a smaller space, we're reducing the dimension of the feature space.

Principle of PCA (cont'd)

Given a desired number of final features, PCA will create principal components by *minimizing the loss of information* from the initial data and thus *maximizing their relevance* (i.e., informative, discriminative, and non-redundant).




```python
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
Principal_components = pca.fit(X).transform(X)
```

Desired number of final features

Principle of PCA (cont'd)

A famous implementation of PCA is in face recognition.

These are called eigen-faces, which correspond to the axes (directions) to make projections.

Steps of PCA (optional)

1. Compute the mean feature vector

   $\mu = \frac{1}{p}\sum_{k=1}^{p} x_k$, where, x_k is a pattern ($k = 1$ to p), p = number of patterns, x is the feature matrix

2. Find the covariance matrix

   $C = \frac{1}{p}\sum_{k=1}^{p}\{x_k - \mu\}\{x_k - \mu\}^T$ where, T represents matrix transposition

3. Compute Eigen values λ_i and Eigen vectors v_i of covariance matrix

   $Cv_i = \lambda_i v_i$ $\quad$ $(i = 1, 2, 3, \ldots. q)$, q = number of features

4. Estimating high-valued Eigen vectors

   (i) Arrange all the Eigen values (λ_i) in descending order

   (ii) Choose a threshold value, θ

   (iii) Number of high-valued λ_i can be chosen so as to satisfy the relationship

   $$\left(\sum_{i=1}^{s}\lambda_i\right)\left(\sum_{i=1}^{q}\lambda_i\right)^{-1} \geq \theta$$, where, s = number of high valued λ_i chosen

   (iv) Select Eigen vectors corresponding to selected high valued λ_i

5. Extract low dimensional feature vectors (principal components) from raw feature matrix.

   $P = V^T x$, where, V is the matrix of selected Eigen vectors and x is the feature matrix

This is an advanced content and optional to learn depending on your knowledge level and interest. You may refer to https://www.youtube.com/watch?v=rng04VJxUt4 for more detailed explanations.

SWINBURNE UNIVERSITY OF TECHNOLOGY

Feature selection

Keeping “relevant” features only by removing features which are:

- Non-informative
- Non-discriminative
- Redundant

Removing non-informative features

Method: Recursive Feature Elimination (RFE)

Principle: Eliminating a single feature *in turn*, running the model each time and noting impact on the performance of the model.

<u>Example: Predicting the price of a house</u>

~~Size~~: 300 sqm

Location: Paris 6th

Room: 5

Year: 1970

Test model

What is the impact on the performance of the ML model?

The lower the impact, the less informative the feature is, and vice versa.

```
from sklearn.feature_selection import RFE
rfe = RFE(estimator=estimator, n_features_to_select=1, step=1)
rfe.fit(X, y)
```

Removing non-discriminative features

Method: Variance Threshold Filter

Principle: Removing features whose values are close across all different training examples (i.e., having *low variance*).

<u>Example: Predict the price of houses that are all white</u>

A feature that always says the same thing won't help the model.

```
from sklearn.feature_selection import VarianceThreshold
sel = VarianceThreshold(threshold=threshold_value)
sel.fit_transform(X)
```

Removing redundant features

Method: High Correlation Filter

Principle: Removing features that are similar or highly correlated with other features.

<u>Example: Same size in square meters and square inches</u>

The model does not need the same information twice.

You can identify highly correlated features via the Pearson correlation coefficients between all pairs of features.

```
import numpy as np
matrix = np.corrcoef(X)
```

Further reading and references

- Müller, Andreas C., author.; Guido, Sarah, ***Introduction to machine learning with Python : a guide for data scientists***, 1st edition Sebastopol, CA : O'Reilly Media 2017.
- Cutler, Josh ; Dickenson, Matt, ***Introduction to Machine Learning with Python***, Springer International Publishing.
- Mueller, J & Massaron, L 2015, ***Python for data science for dummies***, **1st edition.**
- Cielen, Davy, author.; Meysman, Arno, author.; Ali, Mohamed, ***Introducing data science : big data, machine learning, and more, using Python tools***, 1st edition.
- Amr, Tarek, ***Hands-On Machine Learning with Scikit-learn and Scientific Python Toolkits: A Practical Guide to Implementing Supervised and Unsupervised Machine Learning Algorithms in Python***, 2020.
- Haroon, Danish, **Python Machine Learning Case Studies: Five Case Studies for the Data Scientist**, 2017.

Further reading and references (cont'd)

References:

https://www.whatcounts.com/wp-content/uploads/2012/03/7002754749_d11d3e3931-compressor.jpg
https://i.ytimg.com/vi/QcCjmWwEUgg/maxresdefault.jpg
https://www.researchgate.net/profile/Mohsen_Attaran/publication/325934828/figure/fig8/AS:654140678094848@1532970699006/Key-Benefits-of-Predictive-Analytics-in-Manufacturing.png
https://www.cerait.com/sites/default/files/pictures/3-3infographic-pa-12345.jpg
https://www.marketreportgazette.com/wp-content/uploads/2019/09/Healthcare-Predictive-Analytics-1-1.jpg
https://engmrk.com/wp-content/uploads/2018/01/Rec_System.png
https://www.ncdc.noaa.gov/sites/default/files/NAM_20120710_0000_refcclm-small.gif
https://securityboulevard.com/wp-content/uploads/2018/12/Facial-Recognition-Surveillance.jpg
https://www.ft.com/_origami/service/image/v2/images/raw/https%3A%2F%2Fs3-ap-northeast-1.amazonaws.com%2Fpsh-ex-ftnikkei-3937bb4%2Fimages%2F2%2F3%2F6%2F7%2F13707632-3-eng-GB%2F20180420-Fingerprint.jpg
https://images.deepai.org/ai-pages/sections/d87f553815fd45e384b93815e1887e21/brain-tumor-detection-deep-learning.jpg
https://miro.medium.com/max/896/1*MnbUSSXG1IDS9mXC0H8cKQ.png
https://1.bp.blogspot.com/-K65Ed68KGXk/WOa9jaRWC6I/AAAAAAAABsM/gglycD_anuQSp-i67fxER1FOlVTulvV2gCLcB/s1600/FederatedLearning_FinalFiles_Flow%2BChart1.png
http://kerckhoffs.schaathun.net/FPIA/Slides/09OF.pdf
https://miro.medium.com/max/3960/1*QNy78Yre3WttiHuuaTvdYA.jpeg
https://miro.medium.com/max/928/1*OJVhBtg5YgeW7rKXoxKQxg.png
https://www.kdnuggets.com/wp-content/uploads/dataiku-holdout-strategy.jpg
https://sebastianraschka.com/images/blog/2016/model-evaluation-selection-part3/holdout-vs-2fold.png
https://miro.medium.com/max/915/1*obKmc_bTKbUFgcgryhaAnA.png
http://ethen8181.github.io/machine-learning/model_selection/img/kfolds.png
https://miro.medium.com/max/960/1*D9OJZ-n0xh9d87i58WfMLQ.png
https://miro.medium.com/max/1051/1*Oz6Hopj7ipJh0vtA2FE-FQ.png
https://sebastianraschka.com/images/blog/2016/model-evaluation-selection-part3/kfold.png
https://www.kdnuggets.com/wp-content/uploads/dataiku-kfold-strategy.jpg
http://www.cs.nthu.edu.tw/~shwu/courses/ml/labs/08_CV_Ensembling/fig-nestedcv.png
https://miro.medium.com/max/385/1*7EYylA6XlXSGBCF77j_rOA.png
https://miro.medium.com/max/964/1*BH9sKc-Mx8kIH4oSuFJp-A.png
https://miro.medium.com/max/964/1*Rv6rLSX-_b4rSnrhAcUMgw.png
https://miro.medium.com/max/964/1*eLYGnIwLNc25IsiNqu2WIg.png
https://miro.medium.com/max/1078/1*0exdQRxrXQgIBZdPFIxbTw.png
https://miro.medium.com/max/665/1*5XuZ_86Rfce3qyLt7XMlhw.png
https://miro.medium.com/max/893/1*KhlD7Js9leo0B0zfsIfAIA.png
https://miro.medium.com/max/964/1*a8hkMGVHg3fl4kDmSIDY_A.png
https://miro.medium.com/max/812/1*JqzG5_K4fWLa-_VFswQbxQ.png
https://scikit-learn.org/stable/_static/scikit-learn-logo-small.png
http://victoriamarchese.com/wp-content/uploads/2015/06/house-price-up-1.jpg
https://github.com/codebasics/py/raw/fd3fa3081681aa3f299a38d403ae9ecddeb26f25/ML/1_linear_reg/homepricetable.JPG
https://www.katesmathlessons.com/uploads/1/6/1/0/1610286/1163738_orig.png
https://www.youtube.com/watch?v=8jazNUpO3lQ
https://github.com/codebasics/py/raw/fd3fa3081681aa3f299a38d403ae9ecddeb26f25/ML/1_linear_reg/scatterplot.JPG
https://github.com/codebasics/py/raw/fd3fa3081681aa3f299a38d403ae9ecddeb26f25/ML/1_linear_reg/equation.PNG
https://media.geeksforgeeks.org/wp-content/uploads/boston3.png
http://res.cloudinary.com/dyd911kmh/image/upload/f_auto,q_auto:best/v1531424125/KNN_final1_ibdm8a.png
https://upload.wikimedia.org/wikipedia/commons/thumb/e/e7/KnnClassification.svg/330px-KnnClassification.svg.png
https://miro.medium.com/max/784/0*2pplo_uzo7nL8mca.jpg
https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/
https://kevinzakka.github.io/assets/knn/1nn.png
https://kevinzakka.github.io/assets/knn/20nn.png
https://yculz33w9skgdkhey8rajqm6-wpengine.netdna-ssl.com/wp-content/uploads/2018/11/versicolor.jpg

SWINBURNE UNIVERSITY OF TECHNOLOGY