

# Feature Selection and Data Reduction (DRAFT)

Patrick Boily<sup>1,2,3,4</sup>, Olivier Leduc<sup>1</sup>, Andrew Macfie<sup>3</sup>, Aditya Maheshwari<sup>3</sup>, Maia Pelletier<sup>1</sup>

## Abstract

Data mining is the collection of processes by which we can extract useful insights from data. Inherent in this definition is the idea of **data reduction**: useful insights (whether in the form of summaries, sentiment analyses, etc.) ought to be “smaller” and “more organized” than the original raw data.

The challenges presented by high data dimensionality (the so-called **curse of dimensionality**) must be addressed in order to achieve insightful and interpretable analytical results.

In this report, we introduce the basic principles of dimensionality reduction and a number of feature selection methods (filter, wrapper, regularization), discuss some current advanced topics (SVD, spectral feature selection, UMAP) and provide examples (with code).

## Keywords

feature selection, dimension reduction, curse of dimensionality, principal component analysis, manifold hypothesis, manifold learning, regularization, subset selection, spectral feature selection, uniform manifold approximation and projection

## Funding Acknowledgement

Parts of this report were funded by a University of Ottawa grant to develop teaching material in French (2019-2020). These were subsequently translated into English before being incorporated into this document.

<sup>1</sup>Department of Mathematics and Statistics, University of Ottawa, Ottawa

<sup>2</sup>Sprott School of Business, Carleton University, Ottawa

<sup>3</sup>Idlewyld Analytics and Consulting Services, Wakefield, Canada

<sup>4</sup>Data Action Lab, Ottawa, Canada

Email: pboily@uottawa.ca



## Contents

<b>1</b>	<b>Data Reduction for Insight</b>	<b>1</b>
1.1	Reduction of an NHL Game	1
1.2	Meaning in Macbeth	5
<b>2</b>	<b>Dimension Reduction</b>	<b>6</b>
2.1	Sampling Observations	6
2.2	The Curse of Dimensionality	7
2.3	Principal Component Analysis	7
2.4	The Manifold Hypothesis	8
<b>3</b>	<b>Feature Selection</b>	<b>12</b>
3.1	Filter Methods	14
3.2	Wrapper Methods	17
3.3	Subset Selection Methods	17
3.4	Regularization (Embedded) Methods	19
3.5	Supervised and Unsupervised Feature Selection	20
<b>4</b>	<b>Advanced Topics</b>	<b>20</b>
4.1	Singular Value Decomposition	20
4.2	Spectral Feature Selection	22
4.3	Uniform Manifold Approximation and Projection	27

## 1. Data Reduction for Insight

For small datasets, the benefits of data mining may not always be evident. Consider, for instance, the following excerpt from a lawn mowing instruction manual:

Before starting your mower inspect it carefully to ensure that there are no loose parts and that it is in good working order.

It is a fairly short and organized way to convey a message. It could be further shortened and organized, perhaps, but it's not clear that one would gain much from the process.

### 1.1 Reduction of an NHL Game

For a meatier example, consider the NHL game that took place between the Ottawa Senators and the Toronto Maple Leafs on February 18, 2017 [7].

As a first approximation, we shall think of a hockey game as a series of sequential and non-overlapping “events” involving two teams of skaters. What does it mean to have extracted useful insights from such a series of events?

At some level, the most complete raw understanding of that night's game belongs to the game's active and passive

participants (players, referees, coaches, general managers, official scorer and time-keeper, etc.).<sup>1</sup>

The larger group of individuals who attended the game in person, watched it on TV/Internet, or listened to it on the radio presumably also have a lot of the facts at their disposal, with some contamination, as it were, by commentators (in the two latter cases).

Presumably, the participants and the witnesses also possess insights into the specific game: how could that information best be relayed to members of the public who did not catch the game? There are many ways to do so, depending on the intended level of abstraction and on the target audience (see Figure 1).

**Play-by-Play Text File** If a hockey game is a series of events, why not simply list the events, in the order in which they occurred? Of course, not everything that happens in the “raw” game requires reporting – it might be impressive to see Auston Matthews skate by Dion Phaneuf on his way to the Senators’ net at the 8:45 mark of the 2nd period, but reporting this “event” would only serve to highlight the fact that Matthews is a better skater than Phaneuf. It is true, to be sure, but some level of filtering will be applied in order to retain only relevant (or “high-level”) information, such as:

blocked shots, face-off wins, giveaways, goals, hits, missed shots, penalties, power play events, saves, shorthanded events, shots on goal, stoppage (goalie stopped, icing, offside, puck in benches), takeaways, etc.

In a typical game, between 300 and 400 events are recorded (see Table 4 for an extract of the play-by-play file for the game under consideration and pp.30-38 for a full list).

A certain amount of knowledge about the sport is required to make sense of some of the entries (colouring, use of bold text, etc.), but if one has the patience, one can pretty much re-constitute the flow of the game. This approach is, of course, fully descriptive.

**Boxscore** The play-by-play does convey the game’s events, but the relevance of its entries is sometimes questionable. In the general context of the game, how useful is to know that Nikita Zaitsev blocked a shot by Erik Karlsson at the 2:38 mark of the 1st period (see Table 4)? Had this blocked shot saved a certain Ottawa goal or directly lead to a Toronto goal, one could have argued for its inclusion in the list of crucial events to report, but only the most fastidious observer (or a statistical analyst) would bemoan its removal from the game’s report.

The game’s boxscore provides relevant information, at the cost of completeness: it distils the play-by-play file into

<sup>1</sup>This simple assumption is rather old-fashioned and would be disputed by many in the age of hockey analytics, but let it stand for now.

a series of meaningful statistics and summaries, providing insights into the game that even a fan in attendance might have missed while the game was going on (see Tables 5, 6, and 7).

Once again, a certain amount of knowledge about the sport is required to make sense of the statistics, and to place them in the right context: is it meaningful that the Senators won 36 faceoffs to the Maple Leafs’ 31 (see Table 5, top right)? That Mark Stone was a +4 on the night (see Table 6)? That both teams went 1-for-4 on the powerplay (see Table 7)?

One cannot re-constitute the full flow of the game from the boxscore alone, but the approach is not solely descriptive – questions can be asked, and answers provided... the analytical game is afoot!

**Recap/Highlights** One of the boxscore’s shortcomings is that it does not provide much in the way of narrative, which has become a staple of sports reporting – what really happened during that game? How does it impact the current season for either team?

Associated Press, 19 February 2017

TORONTO – The Ottawa Senators have the Atlantic Division lead in their sights.

Mark Stone had a goal and four assists, Derick Brassard scored twice in the third period and the Senators recovered after blowing a two-goal lead to beat the Toronto Maple Leafs 6-3 on Saturday night.

The Senators pulled within two points of Montreal for first place in the Atlantic Division with three games in hand.

“We like where we’re at. We’re in a good spot,” Stone said. “But there’s a little bit more that we want. Obviously, there’s teams coming and we want to try and create separation, so the only way to do that is keep winning hockey games.”

Ottawa led 2-0 after one period but trailed 3-2 in the third before getting a tying goal from Mike Hoffman and a power-play goal from Brassard. Stone and Brassard added empty-netters, and Chris Wideman and Ryan Dzingel also scored for the Senators.

Ottawa has won four of five overall and three of four against the Leafs this season. Craig Anderson stopped 34 shots.

Morgan Rielly, Nazem Kadri and William Nylander scored and Auston Matthews had two assists for the Maple Leafs. Frederik Andersen allowed four goals on 40 shots.

Toronto has lost eight of 11 and entered the night with a tenuous grip on the final wild-card spot in the Eastern Conference.

“The reality is we’re all big boys, we can read the standings. You’ve got to win hockey games,” Babcock said. After Nylander made it 3-2 with a power-play goal 2:04 into the third, Hoffman tied it by rifling a shot from the right faceoff circle off the post and in. On a power play 54 seconds later, Andersen stopped Erik Karlsson’s point shot, but Brassard jumped on the rebound and put it in for a 4-3 lead.

Wideman started the scoring in the first, firing a point shot through traffic moments after Stone beat Nikita Zaitsev for a puck behind the Leafs goal. Dzingel added to the lead when he deflected Marc Methot’s point shot 20 seconds later.

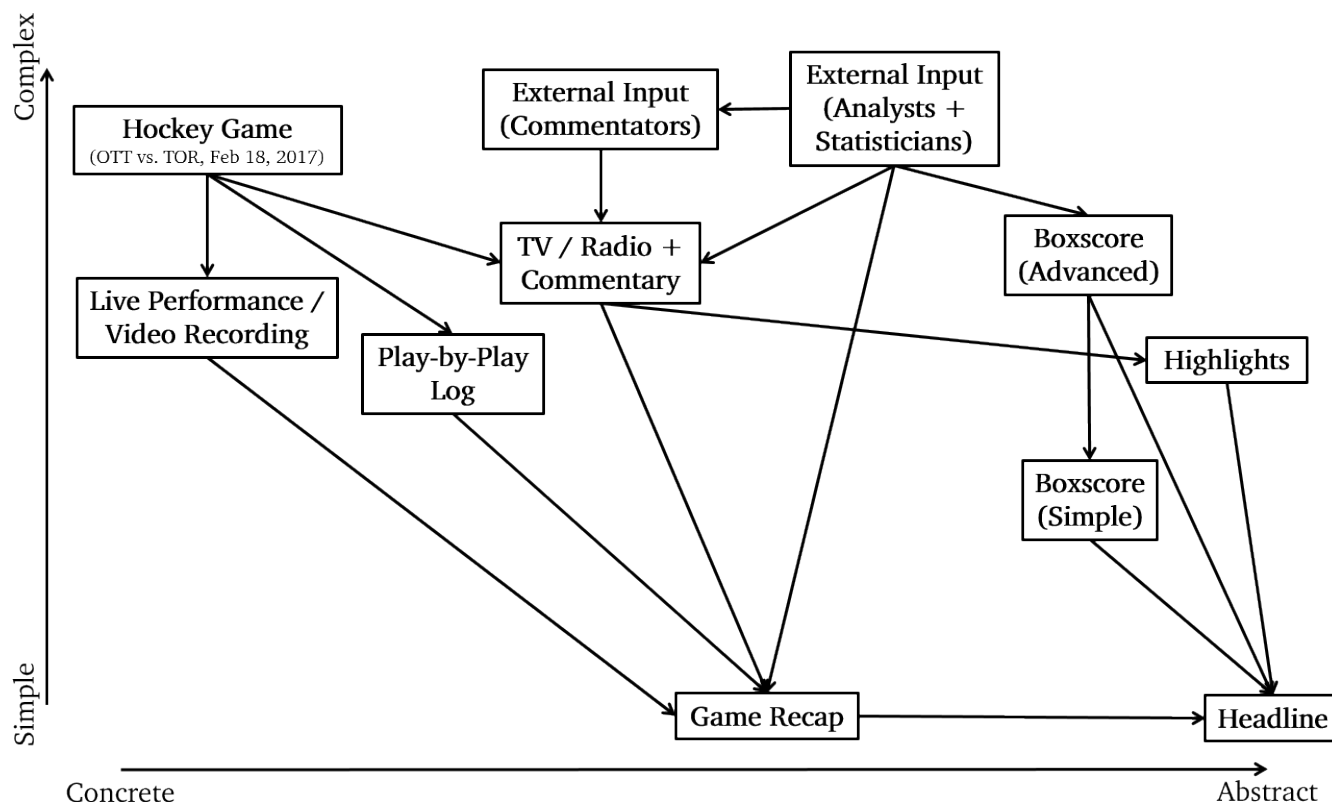


Figure 1. A schematic diagram of data reduction as it applies to a professional hockey game.

Andersen stopped three shots during a lengthy 5-on-3 during the second period, and the Leafs got on the board about three minutes later. Rielly scored with 5:22 left in the second by chasing down a wide shot from Matthews, carrying it to the point and shooting through a crowd in front.

About three minutes later, Zaitsev fired a shot from the right point that sneaked through Anderson’s pads and slid behind the net. Kadri chased it down and banked it off Dzingel’s helmet and in for his 24th goal of the season. Dzingel had fallen in the crease trying to prevent Kadri from stuffing the rebound in.

“Our game plan didn’t change for the third period, and that’s just the maturity we’re gaining over time,” Senators coach Guy Boucher said. “Our leaders have been doing a great job, but collectively, the team has grown dramatically in terms of having poise, executing under pressure.”

**Game notes:** Mitch Marner sat out for Toronto with an upper-body injury. Marner leads Toronto with 48 points and is also expected to sit Sunday night against Carolina.

UP NEXT

Senators: Host Winnipeg on Sunday night.

Maple Leafs: Travel to Carolina for a game Sunday night

**Simple Boxscore** A hockey pool participant might be interested in the fact that Auston Matthews spent nearly 4 minutes on the powerplay (see Table 6), but a casual observer is likely to find the full boxscore monstrous overkill. How much crucial information is lost/provided by Table 1?

Ottawa Senators	31-19-6		6	Toronto Maple Leafs	26-20-11
OTT	2	0	4	6	★ Stone (Senators - RW): Goals: 1, Assists: 4
TOR	0	2	1	3	★★ Brassard (Senators - C): Goals: 2, Assists: 1
					★★★ Nylander (Maple Leafs - RW): Goals: 1, Assists: 1

Table 1. Simple Boxscore, Ottawa Senators @ Toronto Maple Leafs, February 18, 2017 [7].

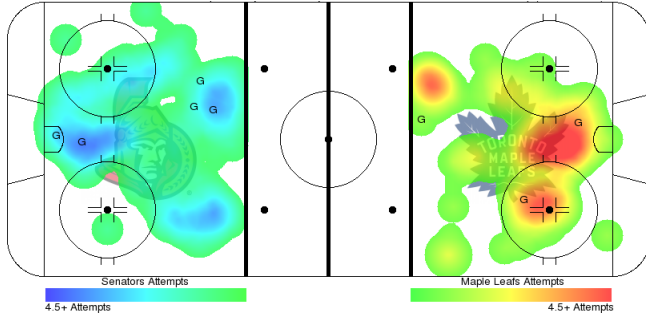
**Headline** If one takes the view that human beings impose a narrative on sporting events (rather than unearth it), it could be argued that the only “true” informational content is found in the following headline (courtesy of AP):

**Sens rally after blowing lead, beat Leafs, gain on Habs.**

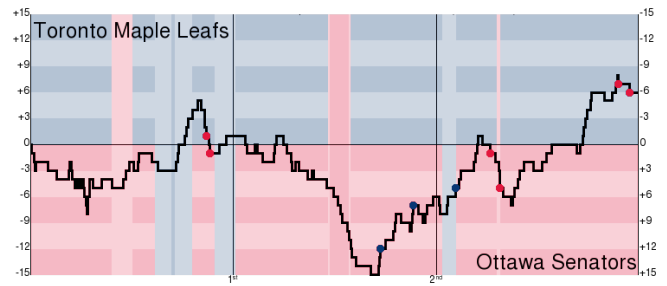
**Visualization** It is easy to get lost in row after row of statistics and events description, or in large bodies of text – doubly so for a machine in the latter case. Visualizations can help complement our understanding of any data analytic situation.

While visualizations can be appealing on their own, a certain amount of external context is required to make sense of most of them (see Figure 2).

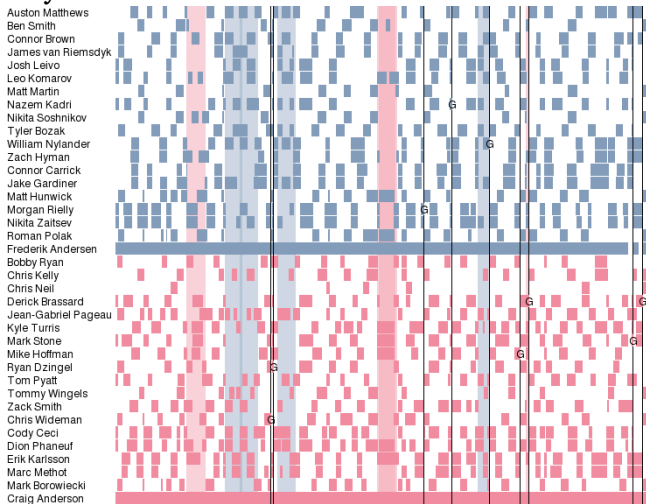
Offensive Zone Unblocked Shots Heat Map



Gameflow Chart, Corsi +/- (All Situations)



Player Shift Chart



Shots and Goals

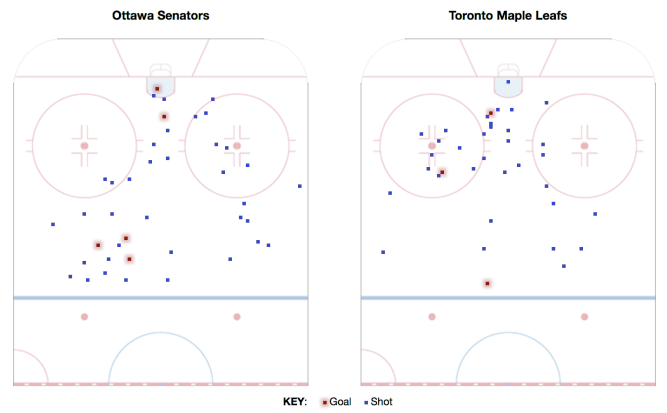


Figure 2. Visualizations, Ottawa Senators @ Toronto Maple Leafs, February 18, 2017 [8].

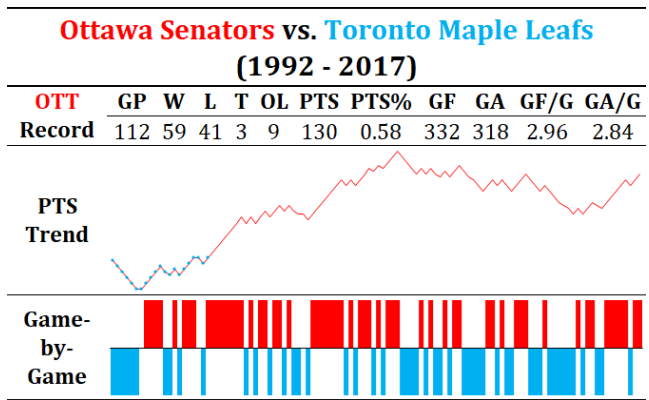
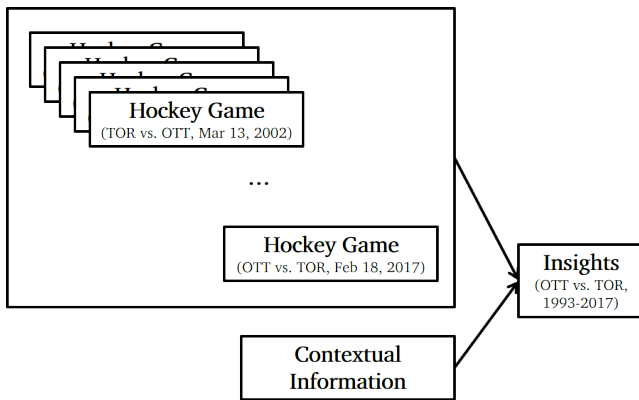


Figure 3. A schematic diagram of data reduction as it applies to a “corpus” of professional hockey games, with visualization and summarizing of regular season games between the Ottawa Senators and Toronto Maple Leafs (1993-2017).



**General Context** A document which is prepared for analysis is often part of a more general context or collection. Can the analysis of all the games between the Senators and the Maple Leafs shed some light on their rivalry on the ice? Obviously, the more arcane the representation method, the more in-depth knowledge of the game and its statistics is required, but to those in the know, summaries and visualizations can provide valuable insight (see Figure 3).

There are thus various ways to understand a single hockey game – and a series of games – depending on the desired (or required) levels of abstraction and complexity.

Clearly, the specific details of data reduction as applied to a hockey game are not always **portable**, but the main concept is. It would be easy to create similar schematic diagrams for *Macbeth*, for instance.

## 1.2 Meaning in Macbeth

### A Metaphor for Meaning?

It is a tale told by an idiot, full of sound and fury, signifying nothing.

– *Macbeth*, Act V, Scene 5, Line 30

In a sense, in order to extract the full **meaning** out of a document, said document needs to be read and understood in its entirety. But even if we have the luxury of doing so, some issues appear:

- do all readers extract the same meaning?
- does meaning stay constant over time?
- is meaning retained by the language of the document?
- do the author's intentions constitute the true (baseline) meaning?
- does re-reading the document change its meaning?

Given the uncertain nature of what a document's meaning actually is, it is counter-productive to talk about insight or meaning (in the singular); rather we look for insights and meanings (in the plural).

Consider the following passage from *Macbeth* (Act I, Scene 5, Lines 45-52):

[Enter **MACBETH**]

**LADY MACBETH:** Great Glamis, worthy Cawdor,  
Greater than both, by the all-hail hereafter,  
Thy letters have transported me beyond  
This ignorant present, and I feel now  
The future in the instant

**MACBETH:** My dearest love, Duncan comes here tonight.

**LADY MACBETH:** And when goes hence?

**MACBETH:** Tomorrow, as he purposes.

What is the “meaning” of this scene? What is the “meaning” of *Macbeth* as a whole?<sup>2</sup>

For non-native English speakers (and for a number of native speakers as well...), the preceding passage might prove difficult to parse and understand.

A modern translation (which is a form of data reduction) is available at *No Fear Shakespeare*, shedding some light on the semantic role of the scene:

**MACBETH** enters.

**LADY MACBETH:** Great thane of Glamis! Worthy thane of Cawdor! You'll soon be greater than both those titles, once you become king! Your letter has transported me from the present moment, when who knows what will happen, and has made me feel like the future is already here.

**MACBETH:** My dearest love, Duncan is coming here tonight.

**LADY MACBETH:** And when is he leaving?

**MACBETH:** He plans to leave tomorrow.

Consider, also, the French translation by F. Victor Hugo:

Entre **MACBETH**.

**LADY MACBETH, *continuant*:** Grand Glamis! Digne Cawdor! plus grand que tout cela par le salut futur! Ta lettre m'a transportée au delà de ce présent ignorant, et je ne ne sens plus dans l'instant que l'avenir.

**MACBETH:** Mon cher amour, Duncan arrive ici ce soir.

**LADY MACBETH:** Et quand repart-il?

**MACBETH:** Demain... C'est son intention.

Do these all carry a *Macbeth* essence? Are they all *Macbeth*? How much, if anything, of *Macbeth* do they preserve? The French translation, for instance, seems to add a very ominous tone to *Macbeth*'s last reply.

One way or another, similar questions must be addressed when investigating aspects of the universe through data analysis (see Figure 4.)

<sup>2</sup>As a starting point, it's crucial to note that the “meaning” of the scene is not independent of the play's context up to this scene (a description of the plot in modern prose is provided on pp. 38–38). Does the plot description carry the same “meaning” as the play itself? What about TVTropes.org's laconic description of *Macbeth* [10]:

Hen-pecked Scottish nobleman murders his king and spends the rest of the play regretting it.

Or Mister Apple's haiku description:

Macbeth and his wife  
Want to become the royals  
So they kill 'em all.

Or this literary description, from an unknown author:

Macbeth dramatizes the battle between good and evil, exploring the psychological effects of King Duncan's murder on Macbeth and Lady Macbeth. His conflicting feelings of guilt and ambition embody this timeless battle of good vs evil.

Or the 2001 W. Morrisette movie *Scotland, PA*, featuring James LeGros, Maura Tierney, and Christopher Walken [11]?

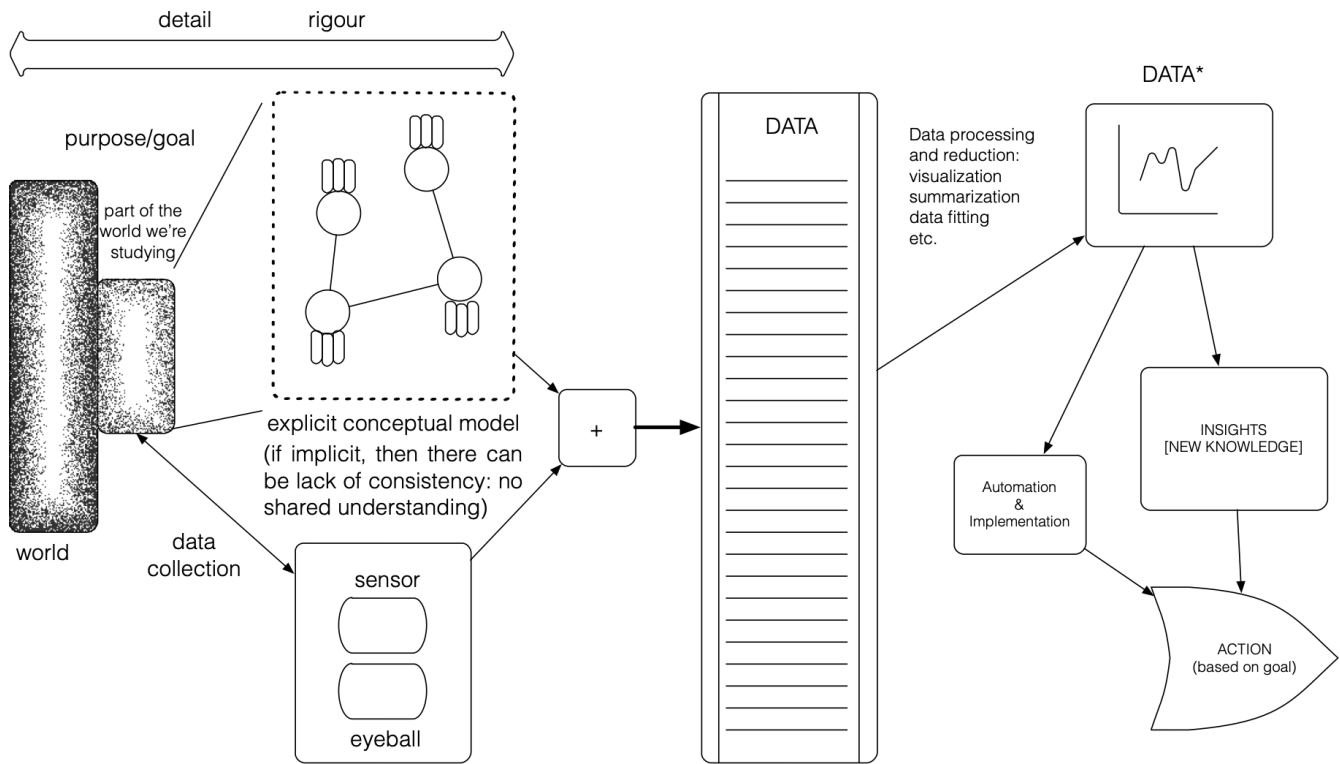


Figure 4. A schematic diagram of data reduction as it applies to a general problem (with J.Schellinck).

## 2. Dimension Reduction

There are many advantages to working with reduced, low-dimensional data:

- **visualisation methods** of all kinds are available and (more) readily applicable to such data in order to extract and present insights;
- high-dimensional datasets are subject to the so-called **curse of dimensionality** (CoD), which asserts (among other things) that multi-dimensional spaces are well, vast, and when the number of features in a model increases, the number of observations required to maintain predictive power also increases, but at a **substantially larger rate** (see Figure 5 for an illustration of CoD and Section 2.2);
- another consequence of the curse is that in high-dimension sets, all observations are roughly **dissimilar** to one another – observations tend to be nearer the dataset’s boundaries than they are to one another.

Dimension reduction techniques such as the ubiquitous **principal component analysis**, **independent component analysis**, **factor analysis** (for numerical data), or **multiple correspondence analysis** (for categorical data) project multi-dimensional datasets onto low-dimensional but high-information spaces (the so-called **Manifold Hypothesis**, see Section 2.4).

Some information is necessarily lost in the process, but in many instances the drain can be kept under control and

the gains made by working with smaller datasets can offset the losses of completeness.

### 2.1 Sampling Observations

Datasets can be “big” in a variety of ways:

- they can be too large for the **hardware** to handle (that is to say, they cannot be stored or accessed properly due to the number of observations, the number of features, or the overall size of the dataset), or
- the dimensions can go against specific modeling assumptions (such as the number of features being much larger than the number of observations, say).

For instance, multiple sensors which record 100+ observations per second in a large geographical area over a long time period can lead to excessively big datasets, say. A natural question then, regarding such a dataset, is whether every one of its rows needs to be used: if rows are selected randomly (with or without replacement), the resulting sample might be **representative**<sup>3</sup> of the entire dataset, and the smaller set might be easier to handle.

There are some drawbacks to the sampling approach, however:

- if the signal of interest is rare, sampling might lose it altogether;

<sup>3</sup>An entire field of statistical endeavour – **statistical survey sampling** – has been developed to quantify the extent to which the sample is representative of the population, but that’s outside the scope of this report.

- if aggregation happens down the road, sampling will necessarily affect the totals<sup>4</sup>, and
- even simple operations on large files (finding the number of lines, say) can be taxing on the memory or in term of computation time – some knowledge or **prior information** about the dataset structure can help.

Sampled datasets can also be used to work the kinks out of the data analysis workflows, but the key take-away is that if data is too big to store, access, and manipulate in a reasonable amount of time, the issue is mostly a **Big Data problem** – this is the time to start considering the use of distributed computing<sup>5</sup>.

## 2.2 The Curse of Dimensionality

A model is said to be **local** if it depends solely on the observations near the input vector ( $k$  nearest neighbours classification is local, whereas linear regression is global). With a large training set, increasing  $k$  in a  $k$ NN model, say, will yield enough data points to provide a solid approximation to the theoretical classification boundary.

The **curse of dimensionality** is the breakdown of this approach in high-dimensional spaces: when the number of features increases, the number of observations required to maintain predictive power also increases, **but at a substantially higher rate**.

### Manifestations of CoD

Let  $x_i \sim U^1(0, 1)$  be i.i.d. for  $i = 1, \dots, N$ . For any  $z \in [0, 1]$  and  $\varepsilon > 0$  such that

$$I_1(z; \varepsilon) = \{y \in \mathbb{R} : |z - y|_\infty < \varepsilon\} \subseteq [0, 1],$$

the expected number of observations  $x_i$  in  $I_1(z; \varepsilon)$  is

$$|I_1(z; \varepsilon) \cap \{x_i\}_{i=1}^N| \approx \varepsilon \cdot N.$$

In other words, a  $\varepsilon_\infty$ -ball subset of  $[0, 1]^1$  contains approximately  $\varepsilon$  of the observations in  $\{x_i\}_{i=1}^N \subseteq \mathbb{R}$ , on average.

Let  $\mathbf{x}_i \sim U^2(0, 1)$  be i.i.d. for  $i = 1, \dots, N$ . For any  $\mathbf{z} \in [0, 1]^2$  and  $\varepsilon > 0$  such that

$$I_2(\mathbf{z}; \varepsilon) = \{y \in \mathbb{R}^2 : \|\mathbf{z} - \mathbf{y}\|_\infty < \varepsilon\} \subseteq [0, 1]^2,$$

the expected number of observations  $\mathbf{x}_i$  in  $I_2(\mathbf{z}; \varepsilon)$  is

$$|I_2(\mathbf{z}; \varepsilon) \cap \{\mathbf{x}_i\}_{i=1}^N| \approx \varepsilon^2 \cdot N.$$

In other words, a  $\varepsilon_\infty$ -ball subset of  $[0, 1]^2$  contains approximately  $\varepsilon^2$  of the observations in  $\{\mathbf{x}_i\}_{i=1}^N \subseteq \mathbb{R}^2$ , on average.

<sup>4</sup>For instance, if we're interested in predicting the number of passengers per flight leaving YOW and the total population of passengers is sampled, then the sampled number of passengers per flight is necessarily below the actual number of passengers per flight. Estimation methods exist to overcome these issues.

<sup>5</sup>Also out-of-scope for this report

In general, the same reasoning shows that a  $\varepsilon_\infty$ -ball subset of  $[0, 1]^p \subseteq \mathbb{R}^p$  contains approximately  $\varepsilon^p$  of the observations in  $\{\mathbf{x}_i\}_{i=1}^N \subseteq \mathbb{R}^p$ , on average. Thus, to capture  $r$  percent of uniformly distributed observations in a unit  $p$ -hypercube, a hyper-subset with edge

$$\varepsilon_p(r) = r^{1/p}$$

is needed, on average. For instance, to capture  $r = 1/3$  of the observations in a unit  $p$ -hypercube in  $\mathbb{R}$ ,  $\mathbb{R}^2$ , and  $\mathbb{R}^{10}$ , a hyper-subset with edge  $\varepsilon_1(1/3) \approx 0.33$ ,  $\varepsilon_2(1/3) \approx 0.58$ , and  $\varepsilon_{10}(1/3) \approx 0.90$ , respectively.

The inference is simple: in general, as  $p$  increases, the nearest observations to a given point  $\mathbf{x}_j \in \mathbb{R}^p$  are in fact quite distant from  $\mathbf{x}_j$ , in the Euclidean sense, on average – **locality is lost!**<sup>6</sup> This can wreak havoc on models and algorithms that rely on the (Euclidean) nearness of observations ( $k$  nearest neighbours,  $k$ -means clustering, etc.).

The CoD manifests itself in various ways. In datasets with a large number of features:

- most observations are **nearer the edge of the sample than to other observations**, and
- realistic training sets are necessarily **sparse**.

Imposing restrictions on models can help mitigate the effects of the CoD, but if the assumptions are not warranted the end result may be even worse.

## 2.3 Principal Component Analysis

**Principal component analysis** (PCA) can be used to find the combinations of variables along which the data points are **most spread out**; it attempts to fit a  $p$ -**ellipsoid** to a centered representation of the data.

The ellipsoid axes are the principal components of the data. Small axes are components along which the variance is “small”; removing these component leads, in an ideal setting, to a “small” loss of information<sup>7</sup> (see Figure 6)

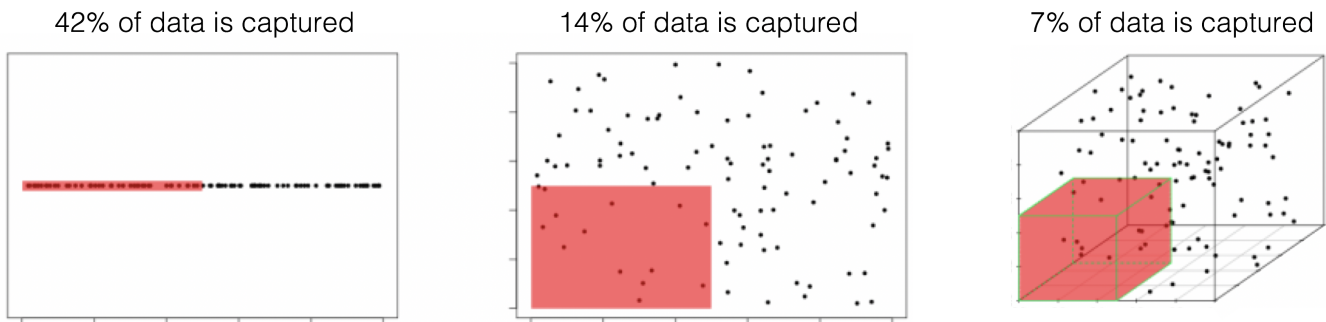
The procedure is simple:

1. centre and “scale” the data to obtain a matrix  $\mathbf{X}$ ;
2. compute the data's covariance matrix  $\mathbf{K} = \mathbf{X}^T \mathbf{X}$ ;
3. compute  $\mathbf{K}$ 's eigenvalues  $\Lambda$  and its orthonormal eigenvectors matrix  $\mathbf{W}$ ;
4. each eigenvector  $\mathbf{w}$  (also known as **loading**) represents an axis, whose variance is given by the associated eigenvalue  $\lambda$ .

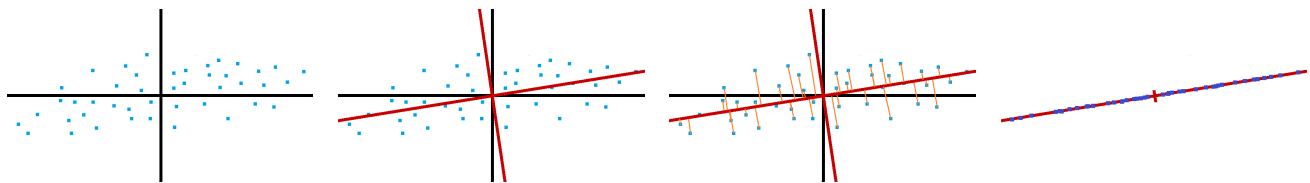
The loading that explains the most variance along a single axis (the **first principal component**) is the eigenvector of the empirical covariance matrix corresponding to the largest eigenvalue, and that variance is proportional to the eigenvalue; the second largest eigenvalue and its corresponding

<sup>6</sup>The situation can be different when the observations are not i.i.d.

<sup>7</sup>Although there are scenarios where it could be those “small” axes that are more interesting – such as the “pancake stack” problem.



**Figure 5.** Illustration of the curse of dimensionality;  $N = 100$  observations are uniformly distributed on the unit hypercube  $[0, 1]^d$ ,  $d = 1, 2, 3$ . The red regions represent the smaller hypercubes  $[0, 0.5]^d$ ,  $d = 1, 2, 3$ . The percentage of captured datapoints is seen to decrease with an increase in  $d$  [12].



**Figure 6.** Illustration of PCA on an artificial 2D dataset. The red axes (second image from left) represent the axes of the best elliptic fit. Removing the minor axis by projecting the points on the major axis leads to a dimension reduction and a (small) loss of information (last image on the right).

eigenvector form the **second principal component** and loading pair, and so on, yielding **orthonormal** principal components  $PC_1, \dots, PC_r$ , where  $r = \text{rank}(\mathbf{X})$ .<sup>8</sup>

Principal component analysis can provide an avenue for dimension reduction, by “removing” components with small eigenvalues (see Figure 6): place the **proportion of the spread in the data** which can be explained by each principal component (PC) can be placed in a **scree plot** (a plot of eigenvalues against ordered component indices) and retain the ordered PCs for which:

- the eigenvalue is above some threshold (say, 25%);
- the cumulative proportion of the spread falls below some threshold (say 95%), or
- all PCs leading to a **kink** in the scree plot.

For instance, consider an 8–dimensional dataset for which the ordered PCA eigenvalues are provided below:

PC	1	2	3	4	5	6	7	8
<b>Var</b>	17	8	3	2	1	0.5	0.25	0
<b>Prop</b>	54	25	9	6	3	2	1	0
<b>Cumul</b>	54	79	88	94	98	99	100	100

If only the PCs that explain up to 95% of the cumulative variance are retained, the original data reduces to a 4-dimensional subset; if only the PCs that individually explain more than 25% of the variance are retained, to a 2-dimensional subset; if only the PCs that lead into the first kink in the scree plot are retained, to a 3-dimensional subset (see Figure 7). Consult [44, 45] for PCA tutorials in R.

<sup>8</sup>If some of the eigenvalues are 0,  $r < p$ , and *vice-versa*, implying that the data was embedded in a  $r$ –dimensional manifold to begin with.

PCA is commonly-used, but often without regard to its inherent **limitations**:

- it is dependent on scaling, and so is not uniquely determined;
- with little domain expertise, it may be difficult to interpret the PCs;
- it is quite sensitive to outliers;
- the analysis goals are not always aligned with the principal components, and
- the data assumptions are not always met – in particular, does it always make sense that important data structures and data spread be correlated (the so-called **counting pancakes** problem), or that the components be **orthogonal**?

There are other methods to find the **principal manifolds** of a dataset, including UMAP, self-organizing maps, auto-encoders, curvilinear component analysis, manifold sculpting and kernel PCA [6].

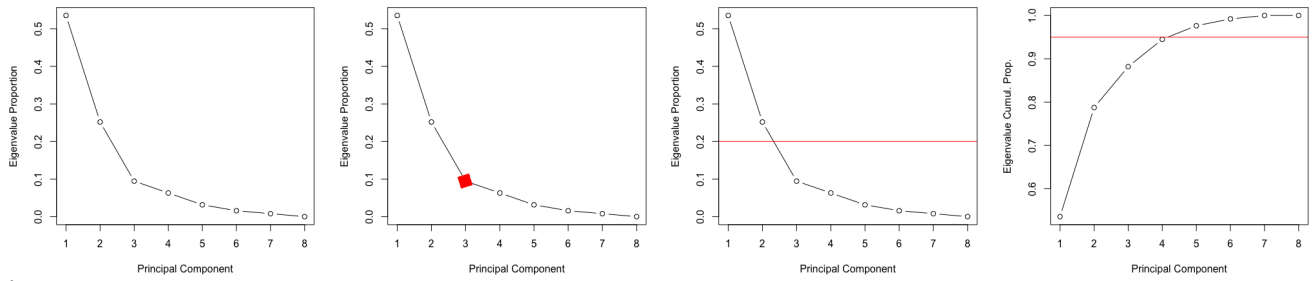
### 2.4 The Manifold Hypothesis

**Manifold learning** involves mapping high-dimensional data to a lower dimensional manifold, such as mapping a set of points in  $\mathbb{R}^3$  to a torus shape, which can then be unfolded (or embedded) into a 2D object.

Techniques for manifold learning are commonly-used because data is often (usually?) sampled from unknown and underlying sources which cannot be measured directly.

Learning a suitable “low-dimension” manifold from a higher-dimensional space is approximately as complicated as learning the sources (in a machine learning sense).





**Figure 7.** Selecting the number of PCs. The proportion of the variance explained by each (ordered) component is shown in the first 3 charts; the cumulative proportion is shown in the last chart. The kink method is shown in the second image, the individual threshold component in the third, and the cumulative proportion in the fourth.

This problem can also be re-cast as finding a set of **degrees of freedom** which can reproduce most of the variability in a dataset. For instance, a set of multiple photographs of a 3D object taken from different positions but all at the same distance from the object can be represented by two degrees of freedom: the horizontal angle and the vertical angle from which the picture was taken.

As another example, consider a set of hand-written drawings of the digit “2” [26]. Each of these drawings can also be represented using a small number of degrees of freedom:

- the ratio of the length of the lowest horizontal line to the height of the hand-written drawing;
- the ratio of the length of the arch in the curve at the top to the smallest horizontal distance from the end point of the arch to the main vertical curve;
- the rotation of the digit as a whole with respect to some baseline orientation, etc.).

These two scenarios are illustrated in Figure 9 [26].

Dimensionality reduction and manifold learning are often used for one of three purposes:

- to reduce the overall dimensionality of the data while trying to preserve the variance;
- to display high-dimensional datasets, or
- to reduce the processing time of supervised learning algorithms by lowering the dimensionality of the processed data.

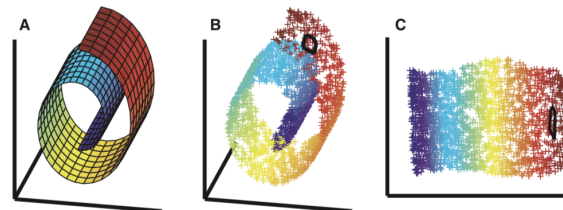
PCA, for instance, provides a sequence of best linear approximations to high-dimensional observations (see previous section); the process has fantastic theoretical properties for computation and applications, but data is not always well-approximated by a fully linear process.

In this section, the focus is on non-linear dimensionality reduction methods, most of which are a variant of **kernel PCA**: LLE, Laplacian eigenmap, isomap, semidefinite embedding, and t-SNE.

By way of illustration, the different methods are applied to an “S”-shaped coloured 2D object living in 3D space (see Figure 10).

**Kernel Principal Component Analysis**

High-dimensional datasets often have a non-linear nature, in the sense that a linear PCA may only weakly capture/-explain the variance across the entire dataset. This is, in part, due to PCA relying on Euclidean distance as opposed to **geodesic distance** – the distance between two points *along* the manifold, that is to say, the distance that would be recorded if the high-dimensional object was first unrolled (see Figure 8, from [26]). Residents of the Earth are familiar with this concept: the Euclidean distance (“as the mole burrows”) between Ottawa and Reykjavik is the length of the shortest tunnel joining the two cities, whereas the geodesic distance (“as the crow flies”) is the arclength of the **great circle** through the two locations.



**Figure 8.** Unfolding of a high-dimensional manifold [26].

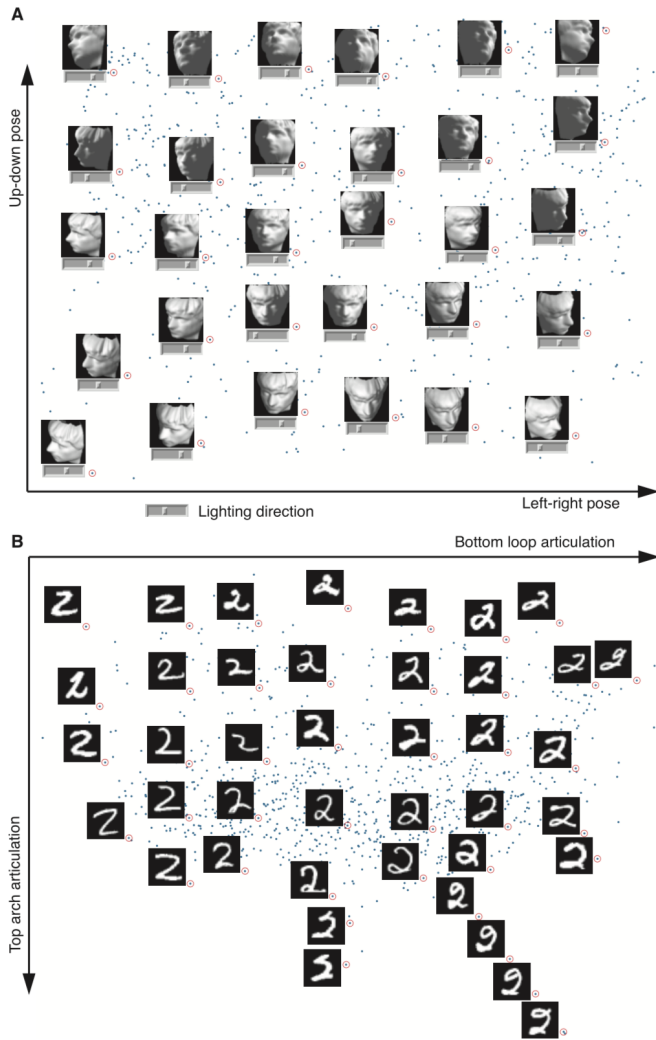
High-dimensional manifolds can be unfolded with the use of **transformations**  $\Phi$  which map the input set of points in  $\mathbb{R}^n$  onto a new set of points in  $\mathbb{R}^m$ , with  $m \geq N$ . If  $\Phi$  is chosen so that  $\sum_{i=1}^N \Phi(\mathbf{x}_i) = 0$  (i.e., the transformed data is also centered in  $\mathbb{R}^m$ ), we can formulate the kernel PCA objective in  $\mathbb{R}^n$  as a linear PCA objective in  $\mathbb{R}^m$ :

$$\min \left\{ \sum_{i=1}^N \|\Phi(\mathbf{x}_i) - V_q V_q^T \Phi(\mathbf{x}_i)\| \right\},$$

over the set of  $m \times q$  matrices  $V_q$  with orthonormal columns, where  $q$  is the desired dimension of the manifold.<sup>9</sup>

In practice, it can be quite difficult to determine  $\Phi$  explicitly; in many instances, it is inner-product-like quantities that are of interest to the analyst.

<sup>9</sup>This approach to PCA is called the **error reconstruction** approach and it gives the same results as the **covariance** approach of the previous section [34].



**Figure 9.** Plots showing degrees of freedom manifolds for images of faces (3D object) and handwritten digits [26].

The problem can be resolved by working with **positive-definite kernel functions**  $K : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}_+$  which satisfy  $K(\mathbf{x}, \mathbf{y}) = K(\mathbf{y}, \mathbf{x})$  for all  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$  and

$$\sum_{i=1}^k \sum_{j=1}^k c_i c_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0$$

for any integer  $k$ , coefficients  $c_1, \dots, c_k \in \mathbb{R}$  and vectors  $\mathbf{x}_1, \dots, \mathbf{x}_k \in \mathbb{R}^n$ , with equality if and only if  $c_1, \dots, c_k = 0$ . Popular data analysis kernels include the:

- **linear kernel**  $K(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\top \mathbf{y}$ ;
- **polynomial kernel**  $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^\top \mathbf{y} + r)^k$ ,  $n \in \mathbb{N}$ ,  $r \geq 0$ , and
- **Gaussian kernel**  $K(\mathbf{x}, \mathbf{y}) = \exp\left\{-\frac{\|\mathbf{x}-\mathbf{y}\|}{2\sigma^2}\right\}$ ,  $\sigma > 0$ .

Most dimension reduction algorithms can be re-expressed as some form of kernel PCA.

### Locally Linear Embedding

**Locally linear embedding** (LLE) is another manifold learning approach which addresses the problem of nonlinear dimension reduction by computing low-dimensional, neighbourhood-preserving embedding of high-dimensional data.

The main assumption is that for any subset  $\{\mathbf{x}_i\} \subseteq \mathbb{R}^n$  lying on some  $d$ -dimensional, sufficiently well-behaved underlying manifold  $\mathcal{M}$ , each data point and its neighbours lie on a **locally linear patch** of  $\mathcal{M}$ . Using translations, rotations, and rescaling, the (high-dimensional) coordinates of each locally linear neighbourhood can be mapped to a set of  $d$ -dimensional global coordinates of  $\mathcal{M}$ . This needs to be done in such a way that the relationships between neighbouring points are preserved. This can be done in 3 steps:

1. identify the punctured neighbourhood  $N_i = \{i_1, \dots, i_k\}$  of each data point  $\mathbf{x}_i$  via  $k$  nearest neighbours (this could also be done by selecting all points within some fixed radius  $\epsilon$ , but  $k$  is not a constant anymore, and that complicates matters);
2. find the weights  $w_{i,j}$  that provide the best linear reconstruction of each  $\mathbf{x}_i \in \mathbb{R}^n$  from their respective punctured neighbourhoods<sup>10</sup>, i.e., solve

$$\min_{\mathbf{W}} \left\{ \sum_{i=1}^N \left\| \mathbf{x}_i - \sum_{j \in N_i} w_{i,j} \mathbf{x}_{N_i(j)} \right\|^2 \right\},$$

where  $\mathbf{W} = (w_{i,j})$  is an  $N \times N$  matrix ( $w_{i,j} = 0$  if  $j \notin N_i$ ), and

3. find the low-dimensional embedding (or code) vectors  $\mathbf{y}_i \in \mathcal{M} (\subseteq \mathbb{R}^d)$  and neighbours  $\mathbf{y}_{N_i(j)} \in \mathcal{M}$  for each  $i$  which are best reconstructed by the weights determined in the previous step, i.e., solve

$$\min_{\mathbf{Y}} \left\{ \sum_{i=1}^N \left\| \mathbf{y}_i - \sum_{j \in N_i} w_{i,j} \mathbf{y}_{N_i(j)} \right\|^2 \right\} = \min_{\mathbf{Y}} \{ \text{Tr}(\mathbf{Y}^\top \mathbf{Y} \mathbf{L}) \},$$

where  $\mathbf{L} = (\mathbf{I} - \mathbf{W})^\top (\mathbf{I} - \mathbf{W})$  and  $\mathbf{Y}$  is an  $N \times d$  matrix.

If the global coordinates of the sampled points are centered at the origin and have unit variance (which can always be done by adding an appropriate set of restrictions), it can be shown that  $\mathbf{L}$  has a 0 eigenvalue with associated eigenvector. The  $j^{\text{th}}$  column of  $\mathbf{Y}$  is then simply the eigenvector associated with the  $j^{\text{th}}$  smallest non-zero eigenvalue of  $\mathbf{L}$  [27].

### Laplacian Eigenmaps

**Laplacian eigenmaps** are similar to LLE, except that the first step consists in constructing a **weighted graph**  $\mathcal{G}$  with  $N$  nodes (number of  $n$ -dimensional observations) and a set of edges connecting the neighbouring points.<sup>11</sup>

<sup>10</sup>Excluding  $\mathbf{x}_i$  itself.

<sup>11</sup>As with LLE, the edges of  $\mathcal{G}$  can be obtained by finding the  $k$  nearest neighbours of each node, or by selecting all points within some fixed radius  $\epsilon$ .



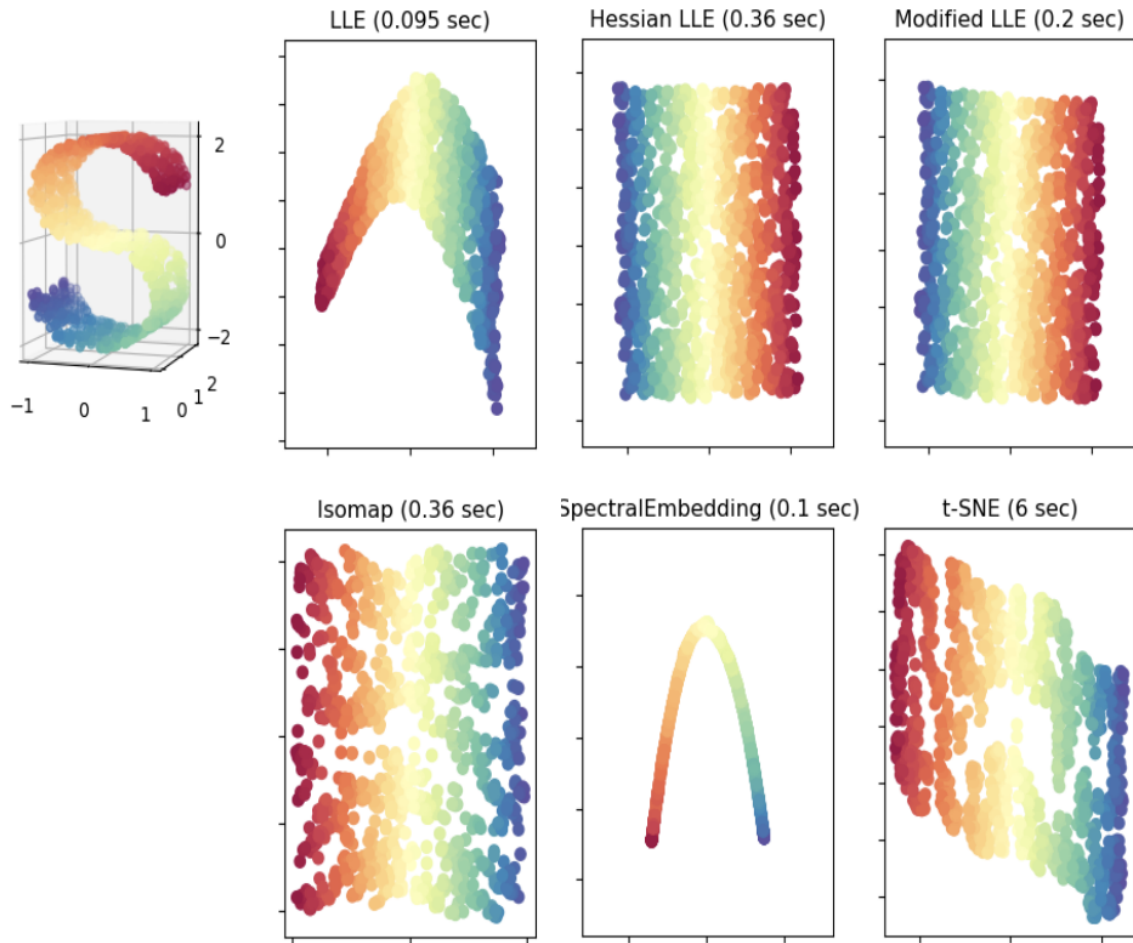


Figure 10. Comparison of manifold learning methods on an artificial dataset [28].

In practice, the edges’ weights are determined either:

- by using the inverse exponential with respect to the Euclidean distance  $w_{i,j} = \exp\left(-\frac{\|x_i - x_j\|^2}{s}\right)$ , for all  $i, j$ , for some parameter  $s > 0$ , or
- by setting  $w_{i,j} = 1$ , for all  $i, j$ .

The embedding map is then provided by the following objective

$$\min_Y \sum_{i=1}^N \sum_{j=1}^N w_{i,j} (\mathbf{y}_i - \mathbf{y}_j)^2 = \min_Y \{ \text{Tr}(YLY^T) \},$$

subject to appropriate constraints, with the **Laplacian**  $L$  given by  $L = D - W$ , where  $D$  is the (diagonal) **degree matrix** of  $\mathcal{G}$  (the sum of weights emanating from each node), and  $W$  its **weight matrix**.

The Laplacian eigenmap construction is identical to the LLE construction, save for their definition of  $L$ .

**Isomap**

**Isomap** follows the same steps as LLE except that it uses **geodesic distance** instead of Euclidean distance when looking for each point’s neighbours (as always, neighbourhoods can be selected with  $k$ NN or with a fixed  $\epsilon$ ). These neighbourhood relations are represented by a graph  $\mathcal{G}$  in which each observation is connected to its neighbours *via* edges with weight  $d_x(i, j)$  between neighbours. The geodesic distances  $d_{\mathcal{M}}(i, j)$  between all pairs of points on the manifold  $\mathcal{M}$  are then estimated in the second step.

**Semidefinite Embedding**

**Semidefinite embeddings** (SDE) involve learning the kernel  $K(\mathbf{x}, \mathbf{z}) = \Phi(\mathbf{x})^T \Phi(\mathbf{z})$  from the data before applying the kernel PCA transformation  $\Phi$ , which is achieved by formulating the problem of learning  $K$  as an instance of **semidefinite programming**. The distances and angles between observations and their neighbours are preserved under transformations by  $\Phi$ :  $\|\Phi(\mathbf{x}_i) - \Phi(\mathbf{x}_j)\|^2 = \|\mathbf{x}_i - \mathbf{x}_j\|^2$ , for all  $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^p$ .

In terms of the kernel matrix, this constraint can be written as

$$K(\mathbf{x}_i, \mathbf{x}_i) - 2(\mathbf{x}_i, \mathbf{x}_j) + K(\mathbf{x}_j, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|^2,$$

for all  $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^n$ . By adding an objective function to maximize  $\text{Tr}(K)$ , that is, the variance of the observations in the learned feature space, SDE constructs a semidefinite program for learning the **kernel matrix**

$$K = (K_{i,j})_{i,j=1}^N = (K(\mathbf{x}_i, \mathbf{x}_j))_{i,j=1}^N,$$

from which kernel PCA can proceed.

### Unified Framework

All of the above algorithms (LLE, Laplacian Eigenmaps, Isomap, SDE) can all be rewritten in the kernel PCA framework:

- in the case of LLE, if  $\lambda_{max}$  is the largest eigenvalue of  $L = (I - W)^T(I - W)$ , then  $K_{LLE} = \lambda_{max}I - L$ ;
- With  $L = D - W$ ,  $D$  a (diagonal) degree matrix with  $D_{i,i} = \sum_{j=1}^N W_{i,j}$ , then then the corresponding  $K_{LE}$  is related to commute times of diffusion on the underlying graph, and
- with the Isomap element-wise squared geodesic distance matrix  $\mathcal{D}^2$ ,

$$K_{\text{Isomap}} = -\frac{1}{2} \left( I - \frac{1}{n} \mathbf{e} \mathbf{e}^T \right) \mathcal{D}^2 \left( I - \frac{1}{n} \mathbf{e} \mathbf{e}^T \right),$$

where  $\mathbf{e}$  is an  $n$ -dimensional vector consisting solely of 1's (note that this kernel is not always p.s.d.).

### t-SNE

There are a few relatively new manifold learning techniques that do not fit neatly in the kernel PCA framework: Uniform Manifold Approximation and Projection (UMAP; see Section 4.3) and  $T$ -distributed Stochastic Neighbour Embedding ( $t$ -SNE). For a dataset  $\{\mathbf{x}_i\}_{i=1}^N \subseteq \mathbb{R}^n$ , the latter involves calculating probabilities

$$P_{i,j} = \frac{1}{2N} \left\{ \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2 / 2\sigma_i^2)} + \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma_j^2)}{\sum_{k \neq j} \exp(-\|\mathbf{x}_j - \mathbf{x}_k\|^2 / 2\sigma_j^2)} \right\},$$

which are proportional to the similarity of points in high-dimensional space  $\mathbb{R}^n$  for all  $i, j$ , and  $p_{ii}$  is set to 0 for all  $i$ .<sup>12</sup> The bandwidths  $\sigma_i$  are selected in such a way that they are smaller in denser data areas.

The lower-dimensional manifold  $\{\mathbf{y}_i\}_{i=1}^N \subseteq \mathcal{M} \subseteq \mathbb{R}^d$  is selected in such a way as to preserve the similarities  $p_{i,j}$

<sup>12</sup>The first component in the similarity metric measures how likely it is that  $\mathbf{x}_i$  would choose  $\mathbf{x}_j$  as its neighbour if neighbours were sampled from a Gaussian centered at  $\mathbf{x}_i$ , for all  $i, j$ .

as much as possible; this can be achieved by building the (reduced) probabilities

$$q_{i,j} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k \neq i} (1 + \|\mathbf{y}_i - \mathbf{y}_k\|^2)^{-1}}$$

for all  $i, j$  (note the asymmetry) and minimizing the **Kullback-Leibler divergence** of  $Q$  from  $P$ :

$$\text{KL}(P||Q) = \sum_{i \neq j} p_{i,j} \log \frac{p_{i,j}}{q_{i,j}}$$

over possible coordinates  $\{\mathbf{y}_i\}_{i=1}^N$  [35].

### MNIST Example

In [28], the methods above are used to learn manifolds for the MNIST dataset [36], a database of handwritten digits (see Figure 11). The results for 4 of those are shown in Figure 12. The analysis of optimal manifold learning methods remains fairly subjective, as it depends not only on the outcome, but also on how much computing power is used and how long it takes to obtain the mapping.

Naïvely, one would expect to see the coordinates in the reduced manifold congregate in 10 (or more) distinct groups; in that regard,  $t$ -SNE seems to perform admirably on MNIST.

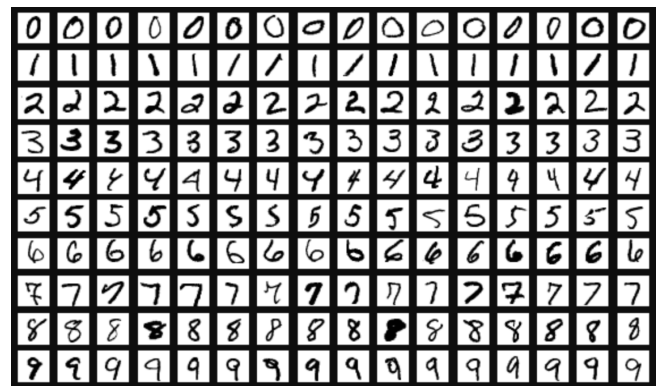


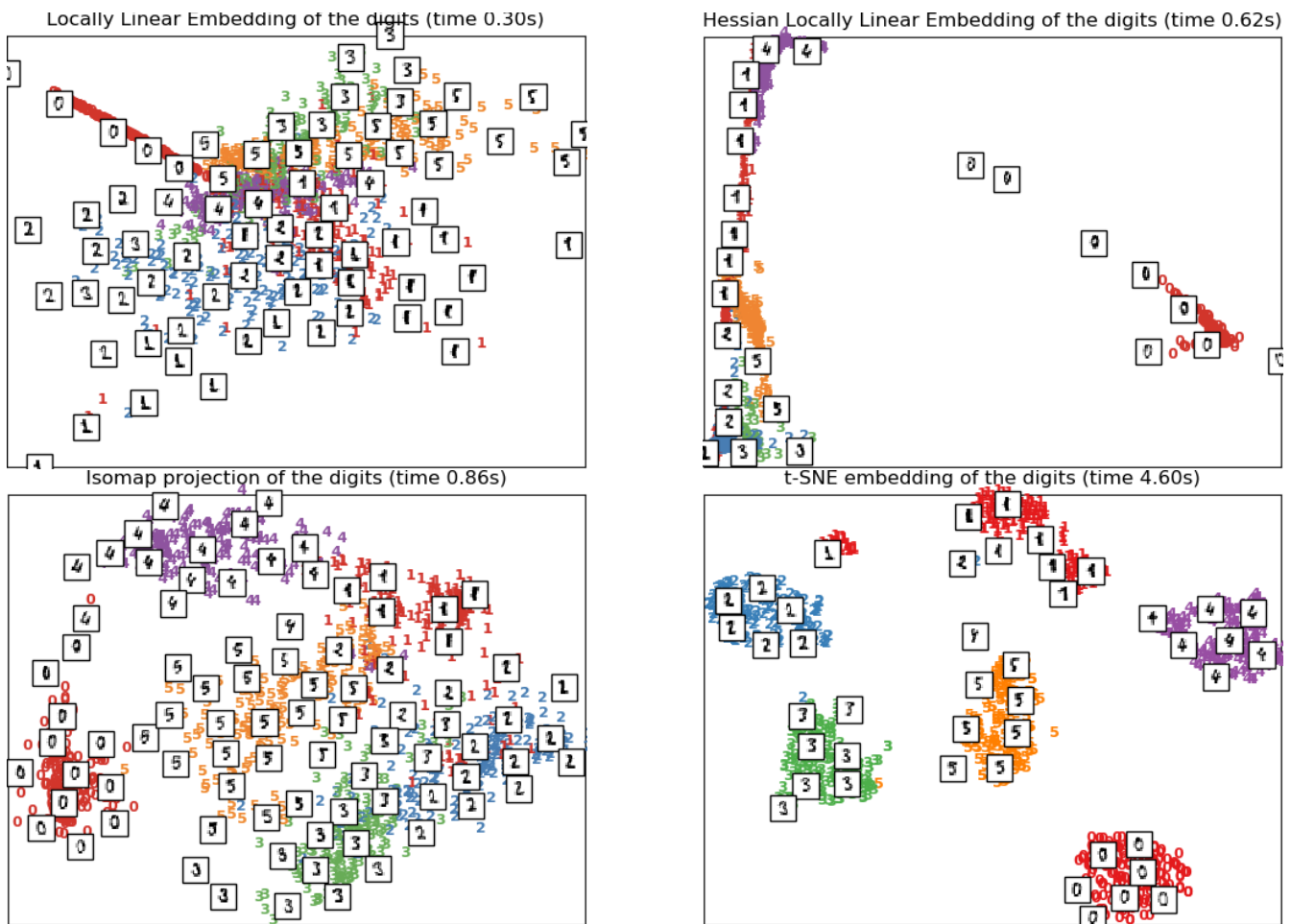
Figure 11. Sample of the MNIST dataset [28, 36].

### 3. Feature Selection

As seen in the previous section, dimension reduction methods can be used to learn low-dimensional manifolds for high-dimensional data, with the hope that the resulting loss in information content can be kept small. Unfortunately, this is not always feasible.

There is another non-technical, yet perhaps more problematic, issue with manifold learning techniques: the reduction often fail to provide an easily **interpretable** set of coordinates in the context of the original dataset.<sup>13</sup>

<sup>13</sup>In a dataset with 4 features ( $X_1 = \text{Age}$ ,  $X_2 = \text{Height}$ ,  $X_3 = \text{Weight}$ , and  $X_4 = \text{Gender}$  (0, 1), say) it is straightforward to **justify** a data-driven decision based on the rule  $X_1 = \text{Age} > 25$ , for example, but substantially



**Figure 12.** Manifold Learning on subset of MNIST (digits 0-5): LLE, Hessian LLE, Isomap,  $t$ -SNE [28, 29].

Datasets often contain **irrelevant** and/or **redundant** features; identifying and removing these variables is a common data processing task. The motivations for doing so are varied, but usually fall into one of two categories:

- the modeling tools do not handle redundant variables well, due to **variance inflation** or similar issues, and
- as an attempt by analysts to overcome the **curse of dimensionality** or to avoid situations where the number of variables is larger than the number of observations.

In light of the footnote on pp. 12-13, the goal of **feature selection** is to remove (not transform or project) any attribute that adds noise and reduces the performance of a model, that is to say, to retain a subset of the most **relevant features**<sup>14</sup>, which can help create simpler models, decrease a statistical learner's training time, and reduce overfitting.

harder to do so for a reduced rule such as

$$Y_2 = 3(\text{Age} - \overline{\text{Age}}) - (\text{Height} - \overline{\text{Height}}) + 4(\text{Weight} - \overline{\text{Weight}}) + \text{Gender} > 7,$$

even if there is nothing wrong with the rule from a technical perspective.

<sup>14</sup>This usually requires there to be a value to predict, against which the features can be evaluated for reference.

There are various feature selection methods, typically falling in one of three families: **filter** methods, **wrapper** methods, and **embedded** methods (most of the next two sections is inspired by [20]):

- **filter methods** focus on the relevance of the features, applying a specific **ranking metric** to each feature, individually. The variables that do not meet a preset benchmark<sup>15</sup> are then removed from consideration, yielding a subset of the most relevant features according to the selected ranking metric; different metrics, and different thresholds, might retain different relevant features;
- **wrapper methods** focus on the usefulness of each feature to the task of interest (usually classification or regression), but do not consider features individually; rather, they evaluate and compare the performance of different combinations of features in order to select the best-performing subset of features, and
- **embedded methods** are a combination of both, using implicit metrics to evaluate the performance of various subsets.

<sup>15</sup>Either a threshold on the ranking or on the ranking metric value itself.

Feature selection methods can also be categorized as **unsupervised** or **supervised**:

- **unsupervised methods** determine the importance of features using only their values (with potential feature interactions), while
- **supervised methods** evaluate each feature's importance in relationship with the **target feature**.

Wrapper methods are typically supervised. Unsupervised filter methods search for noisy features and include the removal of constant variables, of ID-like variables (i.e. different on all observations), or features with low variability.

### 3.1 Filter Methods

Filter methods evaluate features without resorting to the use of any classification/regression algorithm. These methods can either be

- **univariate**, where each feature is ranked independently, or
- **multivariate**, where features are considered jointly.

A filter criterion is chosen based on which metric suits the data or problem best.<sup>16</sup> The selected criteria is used to assign a score to, and rank, the features which are then retained or removed in order to yield a **relevant** subset of features.

Features whose score lies above (or below, as the case may be) some pre-selected threshold  $\tau$  are retained (or removed); alternatively, features whose rank lies below (or above as the case may be) some pre-selected threshold  $\nu$  are retained (or removed).

Such methods are advantageous in that they are computationally efficient. They also tend to be robust against overfitting as they do not incorporate the effects of the feature subset selection on classification/regression performance.

There are a number of commonly-used filter criteria, including the **Pearson correlation coefficient**, **information gain** (or mutual information), and **relief** [20]. Throughout, let  $Y$  be the target variable, and  $X_1, \dots, X_p$  be the predictors.

#### Pearson Correlation Coefficient

The **Pearson correlation coefficient** quantifies the linear relationship between two continuous variables [16]. For a predictor  $X_i$ , the Pearson correlation coefficient between  $X_i$  and  $Y$  is

$$\rho_i = \frac{\text{Cov}(X_i, Y)}{\sigma_{X_i} \sigma_Y}.$$

Features for which  $|\rho_i|$  is large (near 1) are linearly correlated with  $Y$ , those for which  $|\rho_i| \approx 0$  are not linearly (or anti-linearly) correlated with  $Y$  (which could mean that they are uncorrelated with  $Y$ , or that the correlation is not linear or anti-linear). Only those features with (relatively) strong linear (or anti-linear) correlation are retained.

<sup>16</sup>This can be difficult to determine.

This correlation is only defined if both the predictor  $X_p$  and the outcome  $Y$  are numerical; there are alternatives for categorical  $X_p$  and  $Y$ , or mixed categorical-numerical  $X_p$  and  $Y$  [17–19].

#### Mutual Information

**Information gain** is a popular **entropy-based** method of feature selection that measures the amount of dependence between features by quantifying the amount of mutual information between them. In general, this quantifies the **amount of information** obtained about a predictor  $X_i$  by observing the target feature  $Y$ . Mutual information can be expressed as

$$\text{IG}(X_i; Y) = H(X_i) - H(X_i|Y),$$

where  $H(X_i)$  is the **marginal entropy** of  $X_i$  and  $H(X_i|Y)$  is the **conditional entropy** of  $X_i$  given  $Y$  [15], and

$$H(X_i) = E_{X_i}[-\log p(X_i)], \quad H(X_i|Y) = E_{(X_i, Y)}[-\log p(X_i|Y)]$$

where  $p(X_i)$  and  $p(X_i|Y)$  are the probability density functions of the random variables  $X_i$  and  $X_i|Y$ , respectively.

How is IG interpreted? Consider the following example: let  $Y$  represent the salary of an individual (continuous),  $X_1$  their hair colour (categorical),  $X_2$  their age (continuous),  $X_3$  their height (continuous), and  $X_4$  their self-reported gender (categorical). Some summary statistics for a sample of 2144 individuals are shown in Table 2. In a general population, one would expect that the distribution of salaries, say, is likely to be fairly haphazard, and it might be hard to explain why, specifically, it has the shape that it does (see Figure 13), but it could be perhaps be explained by knowing the relationship between the salary and the other variables. It is this idea that lies at the basis of mutual information feature selection. By definition, one sees that

$$H(X_1) = - \sum_{\text{colour}} p(\text{colour}) \log p(\text{colour})$$

$$H(X_2) = - \int p(\text{age}) \log p(\text{age}) d\text{age}$$

$$H(X_3) = - \int p(\text{height}) \log p(\text{height}) d\text{height}$$

$$H(X_4) = - \sum_{\text{gender}} p(\text{gender}) \log p(\text{gender})$$

$$H(X_1|Y) = - \int p(Y) \left\{ \sum_{\text{colour}} p(\text{colour}|Y) \log p(\text{colour}|Y) \right\} dY$$

$$H(X_2|Y) = - \iint p(Y) p(\text{age}|Y) \log p(\text{age}|Y) d\text{age} dY$$

$$H(X_3|Y) = - \iint p(Y) \int p(\text{ht}|Y) \log p(\text{ht}|Y) d\text{ht} dY$$

$$H(X_4|Y) = - \int p(Y) \left\{ \sum_{\text{gender}} p(\text{gender}|Y) \log p(\text{gender}|Y) \right\} dY$$



Hair		Gender		Summary			Age	Height	Salary	Salary					Hair				Total
Deciles	Black	Blonde	Brown	Red	MIN	Q1	MED	Q3	MAX	MEAN	STDEV	SKEW	Deciles	Black	Blonde	Brown	Red	Total	
1	1813	Female	1083	233	16	29	40	53	65	171.6	11.2	-0.04	1	193	7	25	8	233	
2	58	Male	1061	222	136	164	172	179	208	47674.4	19710.3		2	180	9	25	8	222	
3	221		<b>2144</b>	224									3	180	4	30	6	220	
4	52			208									4	187	4	14	3	208	
	<b>2144</b>			224									5	182	8	26	8	224	
				209									6	185	5	18	1	209	
				211									7	183	5	18	5	211	
				217									8	184	6	19	8	217	
				203									9	168	8	22	5	203	
				197									10	171	2	24		197	
				<b>2144</b>									<b>Total</b>	<b>1813</b>	<b>58</b>	<b>221</b>	<b>52</b>	<b>2144</b>	

Salary Deciles	Age Deciles										Total	Salary Deciles	Height Deciles										Total	Salary Deciles	Gender		Total
Deciles	1	2	3	4	5	6	7	8	9	10	Deciles	1	2	3	4	5	6	7	8	9	10	Deciles	Female	Male	Total		
1	203	30									1	28	22	21	24	25	17	27	21	21	27	233	1	144	89	233	
2	40	133	39	1	1	5	1			2	2	19	26	22	18	30	19	23	22	22	21	2	117	105	222		
3		26	73	21	24	15	11	4	14	32	3	34	38	34	26	39	12	15	9	6	7	3	198	22	220		
4		6	26	50	27	17	13	14	24	31	4	33	32	31	28	18	16	15	20	8	7	4	169	39	208		
5		5	31	44	31	18	16	25	26	28	5	34	29	19	25	32	18	21	20	14	12	5	158	66	224		
6		3	21	39	34	27	16	21	33	15	6	33	25	26	14	33	13	22	15	13	15	6	128	81	209		
7			7	40	33	36	23	31	19	22	7	20	24	24	17	24	16	16	19	30	21	7	95	116	211		
8			2	30	38	36	28	33	20	30	8	12	15	15	22	32	16	18	36	20	31	8	54	163	217		
9				2	19	54	30	39	36	23	9	6	13	12	17	27	9	24	28	34	33	9	18	185	203		
10					2	12	41	32	57	47	10	1	4	10	15	18	13	30	35	31	40	10	2	195	197		
<b>Total</b>	<b>243</b>	<b>203</b>	<b>199</b>	<b>229</b>	<b>219</b>	<b>249</b>	<b>170</b>	<b>224</b>	<b>219</b>	<b>189</b>	<b>Total</b>	<b>220</b>	<b>228</b>	<b>214</b>	<b>206</b>	<b>278</b>	<b>149</b>	<b>211</b>	<b>225</b>	<b>199</b>	<b>214</b>	<b>Total</b>	<b>1083</b>	<b>1061</b>	<b>2144</b>		

Table 2. Summary statistics for the salary dataset; two-way tables use decile data.

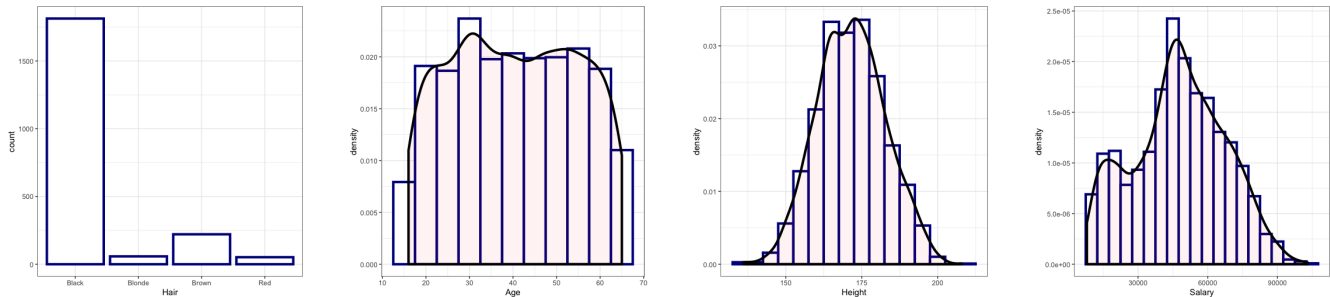


Figure 13. Univariate distributions (hair colour, age, height, salary). The gender distribution is omitted.

If the theoretical distributions are known, the entropy integrals can be computed directly (or approximated using standard numerical integration methods). Gender and hair colour can be modeled using multinomial distributions, but there is more uncertainty related to the numerical variables.

A potential approach is to recode the continuous variables age, height, and salary as **decile variables**  $a_d$ ,  $h_d$ , and  $Y_d$  taking values  $\{1, \dots, 10\}$  according to which decile of the original variable the observation falls (see Table 2 for the decile breakdown). The integrals can then be replaced by sums:

$$H(X_1) = - \sum_{\text{colour}} p(\text{colour}) \log p(\text{colour})$$

$$H(X_2) \approx - \sum_{k=1}^{10} p(\text{age}_d = k) \log p(\text{age}_d = k)$$

$$H(X_3) \approx - \sum_{k=1}^{10} p(\text{ht}_d = k) \log p(\text{ht}_d = k)$$

$$H(X_4) = - \sum_{\text{gender}} p(\text{gender}) \log p(\text{gender})$$

$$H(X_1|Y) \approx - \sum_{j=1}^{10} p(Y_d = j) \sum_{c \in \text{colour}} p(c|Y_d = j) \log p(c|Y_d = j)$$

$$H(X_2|Y) \approx - \sum_{j=1}^{10} p(Y_d = j) \sum_{k=1}^{10} p(a_d = k|Y_d = j) \log p(a_d = k|Y_d = j)$$

$$H(X_3|Y) \approx - \sum_{j=1}^{10} p(Y_d = j) \sum_{k=1}^{10} p(h_d = k|Y_d = j) \log p(h_d = k|Y_d = j)$$

$$H(X_4|Y) \approx - \sum_{j=1}^{10} p(Y_d = j) \sum_{g \in \text{gender}} p(g|Y_d = j) \log p(g|Y_d = j)$$

The results are shown in Table 3.

X	H(X)	H(X Y)	IG(X;Y)	Ratio
Hair	0.24	0.24	0.00	0.00
Age	1.00	0.74	0.26	0.26
Height	1.00	0.96	0.03	0.03
Gender	0.30	0.22	0.08	0.26

**Table 3.** Mutual information obtained about each predictor after observing the target response  $Y$  (salary). The percentage decrease in entropy after having observed  $Y$  is shown in the column “Ratio.” Raw IG numbers would seem to suggest that Gender has a small link to Salary; the Ratio numbers suggest that this could be due to the way the Age and Height levels have been categorized (deciles).

### Relief

**Relief** scores (numerical) features based on the identification of feature value differences between nearest-neighbour instance pairs. If there is a feature value difference in a neighbouring instance pair **of the same class**, the score of the feature decreases; on the other hand, if there exists a feature value difference in a neighbouring instance pair with **different class values**, the feature score increases.

More specifically let  $D = \{(x_n, y_n)\}_{n=1}^N \subset \mathbb{R}^p \times \{\pm 1\}$  be a dataset where  $x_n$  is the  $n$ -th data sample and  $y_n$  is its corresponding class label. For each feature  $i$  and observation  $n$ , two values are selected: the **near hit**  $H(x_{n,i})$  is the value of  $X_i$  which is nearest to  $x_{n,i}$  among all instances in the same class as  $x_n$ , while the **near miss**  $M(x_{n,i})$  is the value of  $X_i$  which is nearest to  $x_{n,i}$  among all instances in the opposite class as  $x_n$ . The **relief score** of the  $i^{\text{th}}$  feature is

$$S_i^d = \frac{1}{N} \sum_{n=1}^N \{d(x_{n,i}, M(x_{n,i})) - d(x_{n,i}, H(x_{n,i}))\},$$

for some pre-selected distance  $d : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}_0^+$ . A feature for which near-hits tend to be nearer to their instances than near-misses are (i.e., for which

$$d(x_{n,i}, M(x_{n,i})) > d(x_{n,i}, H(x_{n,i})),$$

on average) will yield larger relief scores than those for which the opposite is true. Features are deemed **relevant** when their relief score is greater than some threshold  $\tau$ .

There are a variety of Relief-type measurements to accommodate potential feature interactions<sup>17</sup> or multi-class problems (ReliefF), but in general Relief is noise-tolerant and robust to interactions of attributes; its effectiveness decreases for small training sets, however. [23]

<sup>17</sup>For instance, for a  $p$ -distance  $\delta$ , set

$$H^\delta(x_{n,i}) = \operatorname{argmin}_{\pi_i(\mathbf{z})} \{\delta(x_n, \mathbf{z}) : \operatorname{class}(x_n) = \operatorname{class}(\mathbf{z})\}$$

and

$$M^\delta(x_{n,i}) = \operatorname{argmin}_{\pi_i(\mathbf{z})} \{\delta(x_n, \mathbf{z}) : \operatorname{class}(x_n) = -\operatorname{class}(\mathbf{z})\}.$$

Other metrics include:

- other correlation metrics (Kendall, Spearman, point-biserial correlation, etc.)
- other entropy-based metrics (gain ratio, symmetric uncertainty, etc.)
- other relief-type algorithms (ReliefF, Relieved-F, etc.)
- $\chi^2$ -test
- ANOVA
- Fisher Score
- Gini Index
- etc.

The list is by no means exhaustive, but it provides a fair idea of the various types of filter-based feature selection metrics. The following example illustrates how some of these metrics are used to extract relevant attributes.

### Examples

In order to get a better handle on what filter feature selection looks like in practice, consider the *Global Cities Index* dataset [37], which ranks prominent cities around the globe on a general scale of “Alpha”, “Beta”, “Gamma”, and “Sufficiency” (Rating = 1, 2, 3, 4, respectively). This dataset contains geographical, population, and economics data for 68 ranked cities (see Listing 1).<sup>18</sup>

The R package `FSelector` contains feature selection tools, including various filter methods (such as chi-squared score, consistency, various entropy-based filters, etc.). Using its filtering functions, the most relevant features to the ranking of a global city can be extracted. For instance, **linear correlation** (Pearson’s correlation coefficient) as the filtering method for feature selection in this case.

```
library(FSelector)

# Get the corr. coeff. of each feature
lincor <- linear.correlation(formula =
  'Rating ~ .', data = globalcities)

# Retain the top 5 correlations
subset_lincor <- cutoff.k(lincor, k = 5)

# Formula for the selected features
print(as.simple.formula(subset_lincor,
  'Rating'))
Rating ~ Major.Airports +
  Unemployment.Rate +
  GDP.Per.Capita..thousands. +
  Metro.Population..millions. +
  Annual.Population.Growth
```

According to the top-5 linear correlation feature selection method, the “best” features that relate to a city’s global ranking are the number of major airports it has, its unemployment rate, its GDP per capita, its metropolitan popu-

<sup>18</sup>Strictly speaking, the Rating variable should not be treated as a numerical variable (dbl) but as an ordered categorical variable.



lation, and its annual population growth. As filtering is a **pre-processing step**, proper analysis would also require building a model using this subset of features.

`FSelector` also has a built-in function for **information gain** (see above). When applied to the dataset of hockey players (excluding goalies) drafted into the NHL in 2015 [38], it finds 5 relevant features (see Listing 2) linking the players' professional statistics between 2015 and 2019 to the Round in which they were drafted.

```
# Calculate information gain for each
# feature
infogain <- information.gain('Round' ~
  ., draft2015, unit = 'log2')

# Subset features for the top 5
# information gain scores
subset_infogain <- cutoff.k(infogain, 5)

# formula for the selected features
print(as.simple.formula(subset_infogain,
  'Round'))
Round ~ Amateur.Team + Amateur.League
  + Team + Points + Assists
```

According to the top-5 information gain filter, the most relevant features that back-classify a player into their 2015 draft class round are their amateur team, amateur league, professional points scored, number of assists, and penalty minutes, between 2015-2019.

### 3.2 Wrapper Methods

**Wrapper methods** offer a powerful way to address problem of variable selection. Wrapper methods evaluate the quality of subsets of features for predicting the target output under the selected predictive algorithm and select the optimal combination (for the given training set and algorithm).

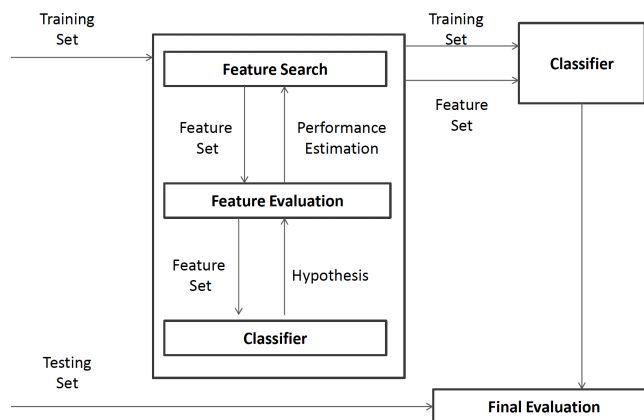
In contrast to filter methods, wrapping methods are integrated directly into the classification or clustering process (see Figure 14 for an illustration of this process).

Wrapper methods treats feature selection as a **search problem** in which different subsets of features are explored. This process can be computationally expensive as the size of the search space increases exponentially with the number of predictors; even for modest problems an exhaustive search can quickly become impractical.

In general, wrapper methods iterate over the following steps, until an “optimal” set of features is identified:

- select a feature subset, and
- evaluate the performance of the selected feature subset.

The search ends once some desired quality is reached (such as adjusted  $R^2$ , accuracy, etc.) Various search methods are



**Figure 14.** Feature selection process for wrapper methods in classification problems [20].

used to traverse the feature phase space and provide an approximate solution to the **optimal feature subset problem**, including: hill-climbing, best-first, and genetic algorithms.

Wrapper methods are not as efficient as filter methods and are not as robust against over-fitting. However, they are very effective at improving the model's performance due to their attempt to minimize the error rate (which unfortunately can also lead to the introduction of implicit bias in the problem [20]).

### 3.3 Subset Selection Methods

**Stepwise selection** is a form of *Occam's Razor*: at each step, a new feature is considered for inclusion or removal from the current features set based on some criterion ( $F$ -test,  $t$ -test, etc.).

Greedy search methods such as backward elimination and forward selection have proven to be both quite robust against over-fitting and among the least computationally expensive wrapper methods.

**Backward elimination** begins with the full set of features and sequentially eliminates the least relevant ones until further removals increase the error rate of the predictive model above some utility threshold. **Forward selection** works in reverse, beginning the search with an empty set of features and progressively adding relevant features to the ever growing set of predictors. In an ideal setting, model performance should be tested using **cross-validation**.

Stepwise selection methods are extremely common, but they have severe limitations (which are not usually addressed) [39, 40]:

- the tests are biased, since they are all based on the same data;
- the adjusted  $R^2$  only takes into account the number of features in the final fit, and not the degrees of freedom that have been used in the entire model;
- if cross-validation is used, stepwise selection has to

**Listing 1. Summary of Global Cities Dataset**


---

```

Observations: 68
Variables: 18
$ Rating
<dbl> 1, 3, 2, 1, 1, 1, 2, 1, 2, 1, 1,
      2, 1, 3, 2, 1, 2, 1...
$ City.Area..km2.
<dbl> 165.00, 30.72, 38.91, 1569.00,
      102.60, 92.54, 892.00...
$ Metro.Area..km2.
<dbl> 807.00, 25437.00, 380.69, 7762.00,
      3236.00, 16410.54...
$ City.Population..millions.
<dbl> 0.76, 3.54, 0.66, 5.72, 1.62,
      14.39, 3.44, 7.44, 0.6...
$ Metro.Population..millions.
<dbl> 1.40, 4.77, 4.01, 6.50, 3.23,
      19.62, 4.97, 9.10, 3.5...
$ Annual.Population.Growth
<dbl> 0.01, 0.26, 0.00, 0.03, 0.01,
      0.04, 0.00, 0.01, 0.01...
$ GDP.Per.Capita..thousands.
<dbl> 46.0, 21.2, 30.5, 23.4, 36.3,
      20.3, 33.3, 15.9, 69.3...
$ Unemployment.Rate
<dbl> 0.05, 0.12, 0.16, 0.02, 0.15,
      0.01, 0.16, 0.10, 0.07...
$ Poverty.Rate
<dbl> 0.18, 0.20, 0.20, 0.00, 0.20,
      0.01, 0.22, 0.22, 0.17...
$ Major.Airports
<dbl> 1, 1, 1, 2, 1, 1, 2, 1, 2, 1,
      1, 2, 1, 1, 1, 1...
$ Major.Ports
<dbl> 1, 0, 1, 1, 1, 0, 2, 0, 1, 1, 1,
      0, 1, 4, 1, 0, 1, 2...
$ Higher.Education.Institutions
<dbl> 23, 10, 21, 37, 8, 89, 30, 19, 35,
      25, 36, 13, 60, 8...
$ Life.Expectancy.in.Years..Male.
<dbl> 76.30, 75.30, 78.00, 69.00, 79.00,
      79.00, 82.00, 74...
$ Life.Expectancy.in.Years..Female.
<dbl> 80.80, 80.80, 83.70, 74.00, 85.20,
      83.00, 88.00, 79...
$ Number.of.Hospitals
<dbl> 7, 7, 23, 173, 45, 551, 79, 22,
      12, 43, 33, 18, 50, ...
$ Number.of.Museums
<dbl> 68, 36, 47, 27, 69, 156, 170, 76,
      30, 25, 131, 15, 3...
$ Air.Quality.
<dbl> 24, 46, 41, 54, 35, 121, 26, 77,
      17, 28, 38, 138, 22...
$ Life.Expectancy
<dbl> 78.550, 78.050, 80.850, 71.500,
      82.100, 81.000, 85.0...

```

---

**Listing 2. Summary of NHL draft dataset**


---

```

Observations: 187
Variables: 15
$ Team
<chr> "Edmonton Oilers", "Buffalo
      Sabres", "Arizona Coyotes", ...
$ Age
<dbl> 18, 18, 18, 18, 18, 18, 18, 18,
      18, 18, 18, 18, 18, 18, 18, 18,
      ...
$ To
<dbl> 2019, 2019, 2019, 2019, 2019,
      2019, 2019, 2019, 2019, 2019,
      20...
$ `Amateur Team`
<chr> "Erie", "Boston University",
      "Erie", "London", "Boston College",
      "Sa..."
$ `Amateur League`
<chr> "OHL", "H-East", "OHL", "OHL",
      "H-East", "OHL", "WHL", "Big Ten",
      "Q..."
$ `Games Played`
<dbl> 287, 286, 106, 241, 319, 201, 246,
      237, 193, 239, 164, 22, 2, 138, 2...
$ Goals
<dbl> 128, 101, 24, 67, 23, 29, 30, 38,
      54, 80, 17, 1, 0, 43, 1, 40, 67, 2...
$ Assists
<dbl> 244, 158, 43, 157, 93, 47, 67, 90,
      54, 129, 21, 3, 0, 42, 0, 107, 61...
$ Points
<dbl> 372, 259, 67, 224, 116, 76, 97,
      128, 108, 209, 38, 4, 0, 85, 1,
      147,...
$ `+/-`
<dbl> 49, -65, -9, 21, -35, -22, -6, 13,
      12, -19, -21, -4, 0, 15, 0, -6, -...
$ `Penalty Minutes`
<dbl> 90, 102, 28, 86, 81, 64, 86, 48,
      116, 112, 122, 0, 0, 37, 0, 82,
      38,...
$ Round
<dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
      1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
      1,...
$ `Draft Year`
<dbl> 2015, 2015, 2015, 2015, 2015,
      2015, 2015, 2015, 2015, 2015,
      20...
$ Nationality
<chr> "Canadian", "American",
      "Canadian", "Canadian", "American",
      "Czech",...
$ Position
<chr> "Centre", "Centre", "Centre",
      "Centre", "Defense", "Centre", ...

```

---

be repeated for each sub-model but that is not usually done, and

- it represents a classic example of  $p$ -hacking.

Consequently, the use of stepwise methods is contraindicated in the machine learning context.

### 3.4 Regularization (Embedded) Methods

An interesting hybrid is provided by the **least absolute shrinkage and selection operator** (LASSO) and its variants. In what follows, assume that the training set consists of  $N$  **centered** and **scaled** observations  $\mathbf{x}_i = (x_{1,i}, \dots, x_{p,i})^\top$ , together with target observations  $y_i$ . Let

$$\hat{\beta}_{LS,j} = [(\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}]_j$$

be the  $j^{\text{th}}$  ordinary least square (OLS) coefficient, and set a threshold  $\lambda > 0$ , whose value depends on the training dataset (there is a lot more here than meets the eye [5,41]).

Recall that  $\hat{\beta}_{LS}$  is the exact solution to the OLS problem

$$\hat{\beta}_{LS} = \arg_{\beta} \min \{ \|\mathbf{y} - \mathbf{X}\beta\|_2^2 \}.$$

In general, there are **no restrictions** on the values taken by the coefficients  $\hat{\beta}_{LS,j}$  – large magnitudes imply that corresponding features **play an important role** in predicting the target. This observation forms the basis of a series of useful OLS variants.

**Ridge regression** is a method to **regularize** the OLS regression coefficients. Effectively, it shrinks the OLS coefficients by penalizing solutions with large magnitudes – if, in spite of this, the magnitude of a specific coefficient is large, then it must have great relevance in predicting the target variable. The problem consists in solving a modified version of the OLS scenario:

$$\hat{\beta}_{RR} = \arg_{\beta} \min \{ \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + N\lambda \|\beta\|_2^2 \}.$$

Solving the RR problem typically requires the use of numerical methods (and of cross-validation to used to determine the optimal  $\lambda$ ); for **orthonormal covariates** ( $\mathbf{X}^\top \mathbf{X} = \mathbf{I}_p$ ), however, the **ridge coefficients** can be expressed in terms of the OLS coefficients:

$$\hat{\beta}_{RR,j} = \frac{\hat{\beta}_{LS,j}}{1 + N\lambda}.$$

**Regression with best subset selection** runs on the same principle but uses a different penalty term, which effectively sets some of the coefficients to 0 (this could be used to select the features with non-zero coefficients, potentially). The problem consists in solving another modified version of the OLS scenario, namely

$$\hat{\beta}_{BS} = \arg_{\beta} \min \{ \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + N\lambda \|\beta\|_0 \}, \|\beta\|_0 = \sum_j \text{sgn}(|\beta_j|).$$

Solving the BS problem typically (also) requires numerical methods and cross-validation; for orthonormal covariates, the **best subset** coefficients can (also) be expressed in terms of the OLS coefficients:

$$\hat{\beta}_{BS,j} = \begin{cases} 0 & \text{if } |\hat{\beta}_{LS,j}| < \sqrt{N\lambda} \\ \hat{\beta}_{LS,j} & \text{if } |\hat{\beta}_{LS,j}| \geq \sqrt{N\lambda} \end{cases}$$

For the **LASSO** problem

$$\hat{\beta}_{BS} = \arg_{\beta} \min \{ \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + N\lambda \|\beta\|_1 \},$$

the penalty effectively yields coefficients which combine the properties of RR and BS, selecting at most  $\max\{p, N\}$  features, and usually no more than one per group of highly correlated variables. For orthonormal covariates, the LASSO coefficients can (yet again) be expressed in term of the OLS coefficients:

$$\hat{\beta}_{L,j} = \hat{\beta}_{LS,j} \cdot \max \left( 0, 1 - \frac{N\lambda}{|\hat{\beta}_{LS,j}|} \right).$$

The use of other penalty functions (or combinations thereof) provides various extensions: elastic nets; group, fused and adaptive lassos; bridge regression, etc. [5]

The modifications described above were defined assuming an underlying linear regression model, but they generalize to arbitrary classification/regression models as well. For a **loss** (cost) function  $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}(\mathbf{W}))$  between the actual target and the values predicted by the model parameterized by  $\mathcal{W}$ , and a **penalty** vector  $\mathbf{R}(\mathbf{W}) = (R_1(\mathbf{W}), \dots, R_k(\mathbf{W}))^\top$ , the **regularized parametrization**  $\mathbf{W}^*$  solves the **general regularization** problem

$$\mathbf{W}^* = \arg_{\mathbf{W}} \min \{ \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}(\mathbf{W})) + \lambda^\top \mathbf{R}(\mathbf{W}) \},$$

which can be solved numerically, assuming some nice properties on  $\mathcal{L}$  and  $\mathbf{R}$  [41]; as before, cross-validation can be used to determine the optimal vector  $\lambda$  [34].

### Examples

In R, regularization is implemented in the package `glmnet` (among others) [42]. An elastic net can be used to select features that are related with the global city rating in the *Global Cities Index* dataset [37], for instance.

```
library(glmnet)
globalcities <-
  read.csv("globalcities.csv")

# centering and scaling the predictors
# assigning predictors and outcome
library(dplyr)
y <- globalcities %>% select(Rating) %>%
  as.matrix()
x <- globalcities %>% select(-Rating)
%>% scale(center = TRUE, scale =
  TRUE) %>% as.matrix()
```

```
# running a 5-fold cv LASSO regression
# alpha controls the elastic net mixture
# alpha = 1: LASSO || alpha = 0: ridge
glmnet1<-cv.glmnet(x=x,y=y,
  type.measure='mse',nfolds=5,alpha=.5)

c<-coef(glmnet1,s='lambda.min',exact=TRUE)
inds<-which(c!=0)
variables<-row.names(c)[inds]
`%ni%`<-Negate(`%in%`)
(variables<-variables[variables %ni%
  '(Intercept)'])
[1] "Metro.Population..millions."
[2] "Annual.Population.Growth"
[3] "GDP.Per.Capita..thousands."
[4] "Unemployment.Rate"
[5] "Major.Airports"
```

The fact that the features that are selected by the elastic net ( $\alpha = 0.5$ ) are the same as those selected by the linear correlation method is a coincidence; selecting  $\alpha = 1$  (LASSO) leads to none of the non-intercept parameters being relevant, while  $\alpha = 0$  (ridge regression) leads to all of them being relevant. That being said, when multiple feature selection methods agree on a core set of features, that provides some model-independent support for the relevance of that set of features to the prediction task at hand.

### 3.5 Supervised and Unsupervised Feature Selection

While feature selection methods are usually categorised as filter, wrapper, or embedded methods, they can also be categorised as **supervised** and **unsupervised** methods. Whether a feature selection method is supervised or not boils down to whether the labels of objects/instances are incorporated into the feature reduction process (or not). The methods that have been described in this section were all supervised.

In unsupervised methods, feature selection is carried out based on the characteristics of the attributes, without any reference to labels or a target variable. In particular, for **clustering problems**, only unsupervised feature selection methods can be used [22].

## 4. Advanced Topics

When used appropriately, the approaches to feature selection and dimension reduction methods presented in the last two sections provide a solid toolkit to help mitigate the effects of the curse of dimensionality. They are, however, for the most part rather straightforward.

The methods discussed in this section are decidedly more sophisticated, from a mathematical perspective; an increase in conceptual complexity can lead to insights that are out of reach by more direct approaches.

### 4.1 Singular Value Decomposition

From a database management perspective, it pays not to view datasets simply as flat file arrays; from an analytical perspective, however, viewing datasets as **matrices** allows analysts to use the full machinery of linear algebra and **matrix factorization** techniques, of which **singular value decomposition** (SVD) is a well-known component.<sup>19</sup>

As before, let  $\{\mathbf{x}_i\}_{i=1}^n \subseteq \mathbb{R}^p$  and denote the **matrix of observations** by

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{bmatrix} \in \mathbb{M}_{n,p}(\mathbb{R}) = \mathbb{R}^{n \times p}.$$

Let  $d \geq \min p, N$ . From a dimension reduction perspective, the goal of matrix factorization is to find two narrow matrices  $\mathbf{W}_d \in \mathbb{R}^{n \times d}$  (the **casas**) and  $\mathbf{C}_d \in \mathbb{R}^{p \times d}$  (the **concepts**) such that the product  $\mathbf{W}_d \mathbf{C}_d^T = \tilde{\mathbf{X}}_d$  is the best rank  $d$  approximation of  $\mathbf{X}$ , i.e.

$$\tilde{\mathbf{X}}_d = \arg_{\mathbf{X}'} \min \{ \|\mathbf{X} - \mathbf{X}'\|_F \text{ with } \text{rank}(\mathbf{X}') = d \},$$

where the **Frobenius norm**  $F$  of a matrix is

$$\|\mathbf{A}\|_F^2 = \sum_{i,j} |a_{i,j}|^2.$$

In a sense,  $\tilde{\mathbf{X}}_d$  is a “**smooth**” **representation** of  $\mathbf{X}$ ; the dimension reduction takes place when  $\mathbf{W}_d$  is used as a dense  $d$ -representation of  $\mathbf{X}$ .

The link with the singular value decomposition of  $\mathbf{X}$  can be made explicit: there exist orthonormal matrices  $\mathbf{U} \in \mathbb{R}^{n \times n}$ ,  $\mathbf{V} \in \mathbb{R}^{p \times p}$ , and a diagonal matrix  $\Sigma \in \mathbb{R}^{n \times p}$  with  $\sigma_{i,j} = 0$  if  $i \neq j$  and  $\sigma_{i,i} \geq \sigma_{i+1,i+1} \geq 0$  for all  $i$ ,<sup>20</sup> such that

$$\mathbf{X} = \mathbf{U} \Sigma \mathbf{V}^T;$$

the decomposition is unique if and only if  $n = p$ .

Let  $\Sigma_d \in \mathbb{R}^{d \times d}$  be the matrix obtained by retaining the first  $d$  rows and the first  $d$  columns of  $\Sigma$ ;  $\mathbf{U}_d \in \mathbb{R}^{n \times d}$  be the matrix obtained by retaining the first  $d$  columns of  $\mathbf{U}$ , and  $\mathbf{V}_d^T \in \mathbb{R}^{d \times p}$  be the matrix obtained by retaining the first  $d$  rows of  $\mathbf{V}^T$ . Then

$$\tilde{\mathbf{X}}_d = \underbrace{\mathbf{U}_d \Sigma_d}_{\mathbf{W}_d} \mathbf{V}_d^T.$$

The  $d$ -dimensional rows of  $\mathbf{W}_d$  are approximations of the  $p$ -dimensional rows of  $\mathbf{X}$  in the sense that

$$\langle \mathbf{W}_d[i], \mathbf{W}_d[j] \rangle = \langle \tilde{\mathbf{X}}_d[i], \tilde{\mathbf{X}}_d[j] \rangle \approx \langle \mathbf{X}_d[i], \mathbf{X}_d[j] \rangle \text{ for all } i, j.$$

<sup>19</sup>Matrix factorization techniques have applications to other data analytic tasks; notably, they can be used to impute missing values and to build recommender systems.

<sup>20</sup>Each **singular value** is the principal square root of the corresponding eigenvalue of the covariance matrix  $\mathbf{X}^T \mathbf{X}$  (see Section 2.3).

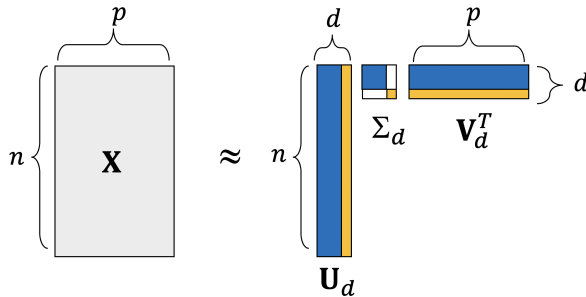




Figure 15. Image reconstruction:  $d = 1400$  (left),  $d = 10$  (middle),  $d = 50$  (right); Llewellyn and Gwynneth Rayfield.

**Applications**

1. One of the advantages of SVD is that it allows for substantial savings in data storage (modified from [46]):



Storing  $X$  requires  $np$  saved entries, but an approximate version of the original requires only  $d(n + p + d)$  saved entries; if  $X$  represents a  $2000 \times 2000$  image (with 4 million entries) to be transmitted, say, a decent approximation can be sent *via*  $d = 10$  using only 40100 entries, roughly 1% of the original number of entries (see Figure 15 for an illustration).

2. SVD can also be used to learn **word embedding vectors**. In the traditional approach to text mining and natural language processing (NLP), words and associated concepts are represented using **one-hot encoding**.<sup>21</sup> For instance, if the task is to predict the part-of-speech (POS) tag of a word given its context in a sentence (current and previous word identities  $w$  and  $pw$ , as well as the latter's part-of-speech (POS) tag  $pt$ ), the input vector could be obtained by concatenation of the one-hot encoding of  $w$ , the one-hot encoding of  $pw$ , and the one-hot encoding of  $pt$ .

The input vector that would be fed into a classifier to predict the POS of the word "house" in the sentence fragment "my house", say, given that "my" has been tagged as a 'determiner' could be

<sup>21</sup>Sparse vectors whose entries are 0 or 1, based on the identity of the words and POS tags under consideration.

current word $w$ is					previous word $pw$ is				previous POS tag $pt$ is											
aardvark	...	hour	house	hover	...	zygote	aardvark	...	mutual	my	nab	...	zygote	noun	...	adj	det	adv	...	prop
x = (	0	...	0	1	0	...	0	...	0	1	0	...	0	0	...	0	1	0	...	0

The sparsity of this vector is a major CoD liability: a reasonably restrictive vocabulary subset of English might contain  $|V_w| \approx 20,000$  words, while the Penn Treebank project recognizes  $\approx 40$  POS tags, which means that  $x \in \mathbb{R}^{40,040}$  (at least).

Another issue is that the one-hot encoding of words does not allow for meaningful similarity comparisons between words: in NLP, words are considered to be similar if they appear in similar sentence **contexts**.<sup>22</sup> The terms "black" and "white" are similar in this framework as they both often occur immediately preceding the same noun (such as "car", "dress", etc.); human beings recognize that the similarity goes further than both of the terms being adjectives – they are both colours, and are often used as opposite poles on a variety of spectra. This similarity is impossible to detect from the one-hot encoding vectors, however, as all its word representations are exactly as far from one another.

SVD proposes a single resolution to both of these issues. Let  $M^f \in \mathbb{R}^{|V_w| \times |V_c|}$  be the **word-context** matrix of the association measure  $f$ , derived from some appropriate corpus, that is, if  $V_w = \{w_1, \dots, w_{|V_w|}\}$  and  $V_c = \{c_1, \dots, c_{|V_c|}\}$  are the vocabulary and contexts of the corpus, respectively, then  $M_{i,j}^f = f(w_i, c_j)$  for

<sup>22</sup>"Ye shall know a word by the company it keeps", as the **distributional semantics** saying goes. The term "kumipwam" is not found in any English dictionary, but its probable meaning as "a small beach/sand lizard" could be inferred from its presence in sentences such as "Elwyn saw a tiny scaly kumipwam digging a hole on the beach". It is easy to come up with examples where the context is ambiguous, but on the whole the contextual approach has proven itself to be mostly reliable.

all  $i, j$ . For instance, one could have

$$V_W = \{\text{aardvark}, \dots, \text{zygote}\},$$

$$V_C = \{\dots, \text{word appearing before "cat"}, \dots\},$$

and  $f$  given by **positive pointwise mutual information** for words and contexts in the corpus (the specifics of which are not germane to the discussion at hand; see [47] for details). The SVD

$$M^f \approx M_d^f = U_d \Sigma_d V_d^T$$

yields  $d$ -dimensional word embedding vectors  $U_d \Sigma_d$  which preserve the context-similarity property (see the last line of p. 20). The decomposition of the POS-context matrix, where words are replaced by POS tags, produces POS embedding vectors.

For instance, a pre-calculated 4-dimensional word embedding of  $V_W$  could be

Word Embeddings	
aardvark	( -0.37 -0.23 0.33 -0.02 )
⋮	⋮
hour	( 0.38 -0.37 -0.21 -0.11 )
house	( 0.31 0.12 -0.03 0.31 )
hover	( -0.10 0.07 0.37 0.15 )
⋮	⋮
mutual	( 0.26 0.25 -0.39 -0.07 )
my	( -0.41 0.37 -0.12 0.02 )
nab	( -0.43 -0.37 -0.12 0.13 )
⋮	⋮
zygote	( 0.06 -0.21 -0.38 -0.28 )

while a 3-dimensional POS embeddings could be

POS Embeddings	
noun	( 0.11 -0.21 0.01 )
⋮	⋮
adj	( 0.21 0.88 0.71 )
det	( 0.17 0.51 0.64 )
adv	( -0.35 0.37 0.37 )
⋮	⋮
prop	( -0.01 0.08 0.74 )

leading to a 11-dimensional representation  $x'$  of  $x$

$$x' = ( \begin{array}{|c|c|c|c|c|c|c|c|c|c|c|} \hline \text{current word } w \text{ is} & & & & \text{previous word } pw \text{ is} & & & & \text{previous POS tag } pr \text{ is} & & & \\ \hline 0.31 & 0.12 & -0.03 & 0.81 & -0.41 & 0.32 & -0.12 & 0.02 & 0.17 & 0.51 & 0.64 & \\ \hline \end{array} )$$

which provides a substantial reduction in the dimension of the input space.

### 4.2 Spectral Feature Selection

Text mining tasks often give rise to datasets which are likely to be affected by the CoD; the problem also occurs when dealing with-high resolution images, with each of the millions of pixels it contains as a feature viewed as a feature; as images have each image containing millions of pixels.

**Spectral feature selection** attempts to identify "good" or "useful" training features in such datasets by measuring their relevance to a given task *via* spectral analysis of the data.

### General Formulation for Feature Selection

Let  $X \in \mathbb{R}^{n \times m}$  be a data set with  $m$  features and  $n$  observations. The problem of  $\ell$ -feature selection, with  $1 \leq \ell \leq m$ , can be formulated as an optimisation problem [31]:

$$\max_W r(\hat{X})$$

$$s.t. \hat{X} = XW, W \in \{0, 1\}^{m \times \ell}$$

$$W^T \mathbf{1}_{m \times 1} = \mathbf{1}_{\ell \times 1}, \|W \mathbf{1}_{\ell \times 1}\|_0 = \ell$$

The **score function**  $r(\cdot)$  is the objective which evaluates the relevance of the features in  $\hat{X}$ , the data set with only certain feature selected by the **selection matrix**  $W$  with entries either 0 or 1. To ensure that only the original feature are selected (and not a linear combination of features), the problem stipulates that each column of  $W$  contains only one 1 ( $W^T \mathbf{1}_{m \times 1} = \mathbf{1}_{\ell \times 1}$ ); to ensure that exactly  $\ell$  rows contain one 1, the constraint  $\|W \mathbf{1}_{\ell \times 1}\|_0 = \ell$  is added. The selected features are often represented by

$$\hat{X} = XW = (f_{i_1}, \dots, f_{i_\ell}) \quad \text{with } \{i_1, \dots, i_\ell\} \subset \{1, \dots, m\}.$$

If  $r(\cdot)$  does not evaluate features independently, this optimisation problem is NP-hard. To make to problem easier to solve, the feature are assumed to be **independent** of one another.<sup>23</sup> In that case, the objective function reduces to

$$\max_W r(\hat{X}) = \max_W (r(f_{i_1}) + \dots + r(f_{i_\ell}));$$

the optimisation problem can then be solved by selecting the  $\ell$  features with the largest individual scores.

The link with **spectral analysis** will be explored in the following pages.

### Similarity Matrix

Let  $s_{i,j}$  denote the **pair-wise similarity** between observations  $x_i$  and  $x_j$ . If class labels  $y(x) \in \{1, \dots, K\}$  are known for all instances  $x$ , the following function can be used

$$s_{i,j} = \begin{cases} \frac{1}{n_k}, & y(x_i) = y(x_j) = k \\ 0, & \text{otherwise} \end{cases}$$

where  $n_k$  is the number of observations with class label  $k$ .

If class labels are not available, a popular similarity function is the **Gaussian radial basis function** (RBF) kernel, given by

$$s_{i,j} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right),$$

where  $\sigma$  is a parameter that is used to control the Gaussian's width.<sup>24</sup> For a given  $s_{i,j}$  and  $n$  observations, the **similarity matrix**  $S$  is an  $n \times n$  matrix containing the observations' pair-wise similarities,  $S(i, j) = s_{i,j}$ ,  $i, j = 1, \dots, n$ . By convention,  $\text{diag}(S) = \mathbf{0}$ .

<sup>23</sup>Or that their interactions are negligible.

<sup>24</sup>One can think of this as the "reach" of each point.



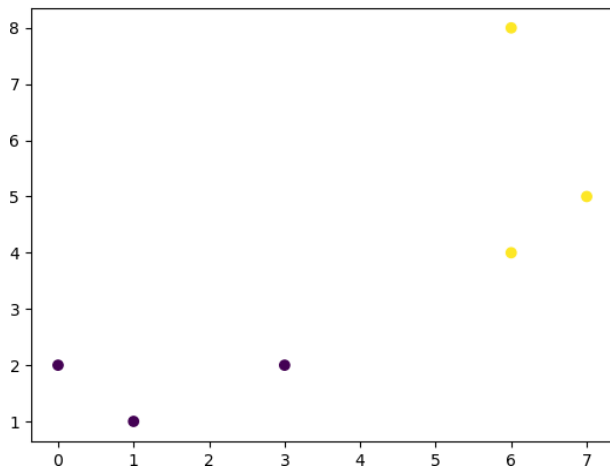


Figure 16. Scatter plot of the example data  $\mathbf{X}$ .

Other similarity functions include the following kernels [32]:

1. **linear** –  $s_{i,j} = \mathbf{x}_i^\top \mathbf{x}_j + c, c \in \mathbb{R}$ ;
2. **polynomial** –  $s_{i,j} = (\alpha \mathbf{x}_i^\top \mathbf{x}_j + c)^d, \alpha, c \in \mathbb{R},^{25} d \in \mathbb{N}$ , and
3. **cosine** –  $s_{i,j} = \frac{\mathbf{x}_i^\top \mathbf{x}_j}{\|\mathbf{x}_i\| \|\mathbf{x}_j\|}$ , which measures the similarity of 2 vectors by determining the angle between them.<sup>26</sup>

**Similarity Graph**

For each similarity matrix  $S$ , one can construct a **weighted graph**  $G(V, E)$  in which each observation corresponds to a node and the associated pairwise similarities correspond to the respective edge weights;  $V$  is the set of all **vertices** (nodes) and  $E$  the set of all graph **edges**.

As an example, consider the simple dataset:

$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 0 & 2 \\ 3 & 2 \\ 6 & 4 \\ 7 & 4 \\ 6 & 8 \end{bmatrix},$$

whose scatter plot is shown in Figure 16. Using an RBF kernel with  $\sigma = 0.5$ , the corresponding similarity matrix is

$$S = \begin{bmatrix} 0 & 0.972 & 0.428 & 0.012 & 0.003 & 0.001 \\ 0.972 & 0 & 0.199 & 0.007 & 0.002 & 0.001 \\ 0.428 & 0.199 & 0 & 0.109 & 0.027 & 0.005 \\ 0.012 & 0.007 & 0.109 & 0 & 0.972 & 0.073 \\ 0.003 & 0.002 & 0.027 & 0.972 & 0 & 0.169 \\ 0.001 & 0.001 & 0.005 & 0.073 & 0.169 & 0 \end{bmatrix},$$

and the resulting graph  $G$  is shown in Figure 17.

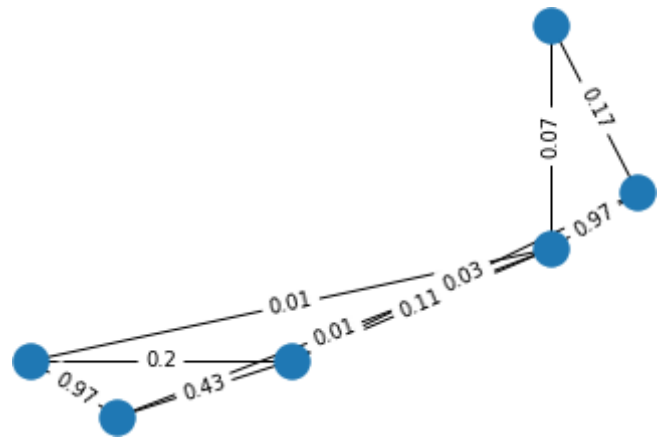


Figure 17. Graph representation of  $\mathbf{X}$ , using a RBF similarity matrix with  $\sigma = 0.5$ . The graph has 6 nodes, one for each observation. The edges with weights below a certain threshold ( $\tau = 0.01$ ) are not displayed.

**Laplacian Matrix of a Graph**

The similarity matrix  $S$  is also known as the **adjacency matrix**  $A$  of the graph  $G$ , from which the **degree matrix**  $D$  can be constructed:

$$D(i, j) = \begin{cases} d_{i,i} = \sum_{k=1}^n a_{ik}, & i = j \\ 0, & \text{otherwise} \end{cases}$$

By definition,  $D$  is diagonal; the element  $d_{i,i}$  can be viewed as an estimate of the **density** around  $\mathbf{x}_i$ ; as  $a_{i,k} (= s_{i,k})$  is a measure of similarity between  $\mathbf{x}_i$  and  $\mathbf{x}_k$ , the larger  $a_{i,k}$  is, the more similar the two observations are. A large value of  $d_{i,i}$  indicates the presence of one or more observations "near"  $\mathbf{x}_i$ ; conversely, a small value of  $d_{i,i}$  suggests that  $\mathbf{x}_i$  is isolated.

The **Laplacian** and **normalized Laplacian** matrices can now be defined as

$$L = D - A \quad \text{and} \quad \mathcal{L} = D^{-1/2} L D^{-1/2},$$

respectively. Since  $D$  is diagonal,  $D^{-1/2} = \text{diag}(d_{i,i}^{-1/2})$ .

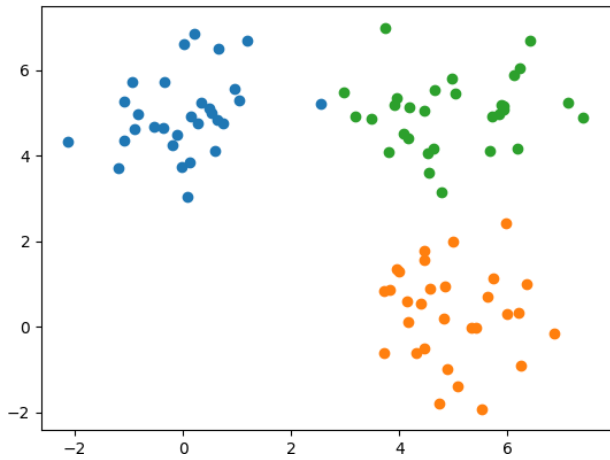
It can be shown that  $L$  and  $\mathcal{L}$  are both positive semi-definite matrices. By construction, the smallest eigenvalue of  $L$  is 0, with associated eigenvector  $\mathbf{1}$ . For  $\mathcal{L}$ , the corresponding eigenpair is 0 and  $\text{diag}(D^{1/2})$ .

**Feature Ranking**

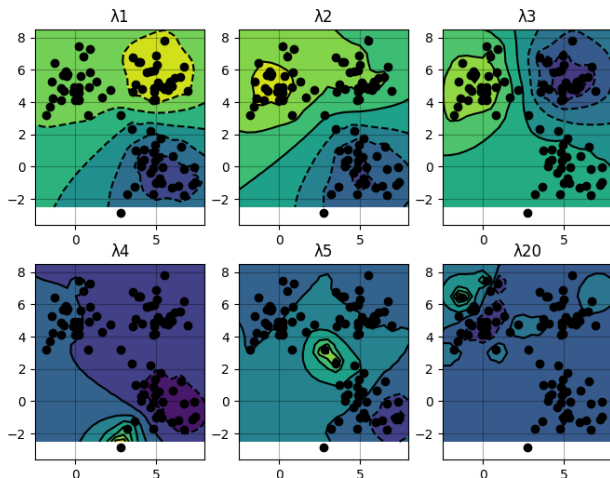
The eigenvectors of the Laplacian matrices have some very useful properties relating to features selection. If  $\xi \in \mathbb{R}^n$  is an eigenvector of  $L$  or  $\mathcal{L}$ , then  $\xi$  can be viewed as a function that assigns a value to each observation in  $\mathbf{X}$ . This point-of-view can prove quite useful, as the following simple example from [31] shows.

<sup>25</sup>For image processing, this kernel is often used with  $\alpha = c = 1$ .

<sup>26</sup>It is often used in high-dimensional applications such as text mining.



**Figure 18.** Scatter plot of the generated data; each colour corresponds to a different generative mechanism.



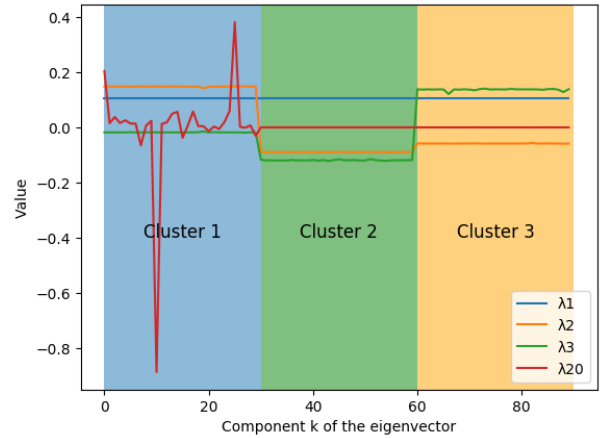
**Figure 19.** Contour plot of the eigenvectors corresponding to the eigenvalues  $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5$  and  $\lambda_{20}$ .

Let  $\mathbf{X}$  be constructed of three two-dimensional Gaussians, each with unit variance but with different means. A scatter plot of the generated data can be found at Figure 18.

The Laplacian matrix of  $\mathcal{L}$  of  $\mathbf{X}$  is built using an RBF kernel with  $\sigma = 1$ . The contour plot of the ranked eigenvectors  $\xi_1, \xi_2, \xi_3, \xi_4, \xi_5$  and  $\xi_{20}$  (corresponding to the eigenvalues  $\lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \lambda_4 \leq \lambda_5 \leq \lambda_{20}$ ) are shown in Figure 19.<sup>27</sup>

From those plots, it seems as though the first eigenvector does a better job at capturing the cluster structure in the data, while larger values tends to capture more of the sub-cluster structure. One thing to note is that it might appear that  $\lambda_1$  is as good (or better) as  $\lambda_2$  and  $\lambda_3$ , but a closer look at the scale of the contour plot of  $\lambda_1$  shows that its values range from  $0.1054 + 88.4 \times 10^{-13}$  to  $0.1054 + 91.2 \times 10^{-13}$ , an interval which is quite small. The fact that there is any

<sup>27</sup>For a given eigenvector  $\lambda_j$ , the contour value at each point  $\mathbf{x}_i$  is the value of the associated eigenvector  $\xi_j$  in the  $i^{\text{th}}$  position, namely  $\xi_{j,i}$ . For any point  $\mathbf{x}$  not in the dataset, the contour value is given by averaging the  $\xi_{j,k}$  of the observations  $\mathbf{x}_k$  near  $\mathbf{x}$ , inversely weighted by the distances  $\|\mathbf{x}_k - \mathbf{x}\|$ .



**Figure 20.** Value of each eigenvector component (one for each observation) associated to  $\lambda_1, \lambda_2, \lambda_3$ , and  $\lambda_{20}$ .

variation at all is due to floating point errors in the practical computation of the eigenvalue  $\lambda_1$  and the eigenvector  $\xi_1$ ; as seen previously, these should be exactly 0 and 1, respectively.

This process shows how the eigenpairs of the Laplacian matrix contains information about the structure of  $\mathbf{X}$ . In spectral graph theory, the eigenvalues of the Laplacian measure the "smoothness" of the eigenvectors.

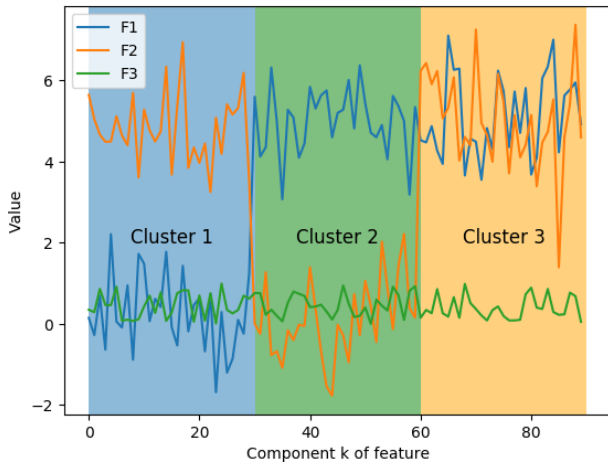
An eigenvector is said to be **smooth** if it assigns similar values to points that are near one another. In the previous example, assume that the data is **unshuffled**, that is, the first  $k_1$  points are in the same cluster, the next  $k_2$  are also in the same, albeit different, cluster, and so on. Figure 20 shows a plot of the value of each eigenvector component (one per observation) associated to different eigenvalues.

Both  $\lambda_2$  and  $\lambda_3$  are fairly smooth, as they seem to be piece-wise constant on each cluster, whereas  $\lambda_{20}$  is all over the place on cluster 1 and constant on the rest of the data. As discussed previously  $\lambda_1$  is constant over the entirety of the dataset, marking it as maximally smooth but not very useful from the perspective of differentiating data structure.<sup>28</sup> Indeed, let  $\mathbf{x} \in \mathbb{R}^n$ . then

$$\begin{aligned} \mathbf{x}^T \mathbf{L} \mathbf{x} &= \mathbf{x}^T \mathbf{D} \mathbf{x} - \mathbf{x}^T \mathbf{A} \mathbf{x} = \sum_{i=1}^n d_i x_i^2 - \sum_{i,j=1}^n a_{i,j} x_i x_j \\ &= \frac{1}{2} \left( \sum_{i=1}^n d_i x_i^2 - 2 \sum_{i,j=1}^n a_{i,j} x_i x_j + \sum_{j=1}^n d_j x_j^2 \right) \\ &= \frac{1}{2} \sum_{i,j=1}^n a_{i,j} (x_i - x_j)^2 \end{aligned}$$

<sup>28</sup>As a reminder, the eigenvalues themselves are ordered in increasing sequence: for the current example,

$$\lambda_1 = 0 \leq \lambda_2 = 1.30 \times 10^{-2} \leq \lambda_3 = 3.94 \times 10^{-2} \leq \dots \leq \lambda_{20} = 2.95 \leq \dots$$



**Figure 21.** Value of  $f_i$  component (one for each observation), for  $i = 1, 2, 3$ .

If  $\mathbf{x} = \xi$  is a normalized eigenvector of  $L$ , then  $\xi^T L \xi = \lambda \xi^T \xi = \lambda$ , thus

$$\lambda = \xi^T L \xi = \frac{1}{2} \sum_{i,j=1}^n a_{ij} (\xi_i - \xi_j)^2.$$

Instinctively, if the eigenvector component does not vary a lot for observations that are near one another, one would expect the corresponding eigenvalue to be small; this result illustrates why the small magnitude of the eigenvalue is a good measure of the "smoothness" of its associated eigenvector.

If  $\mathbf{x}$  is not an eigenvector of  $L$ , the value  $\mathbf{x}^T L \mathbf{x}$  can also be seen as a measure of how much  $\mathbf{x}$  varies locally. This can be used to measure how meaningful a feature  $\mathbf{f} \in \mathbb{R}^n$  is.

In the current example, the only two features are the Euclidean coordinates of the observations  $\mathbf{f}_1$  and  $\mathbf{f}_2$ . Add a useless feature to the dataset  $\mathbf{f}_3$ , say a uniformly distributed random variable across the clusters. Figure 21 shows that the third feature is not able to distinguish between the clusters. However, it can be shown that

$$\mathbf{f}_1^T L \mathbf{f}_1 = 112.8, \quad \mathbf{f}_2^T L \mathbf{f}_2 = 113.3, \quad \mathbf{f}_3^T L \mathbf{f}_3 = 49.6;$$

by the previous assumption relating the magnitude of  $\xi^T L \xi$  to the smoothness of  $\xi$ , this would seem to indicate that  $\mathbf{f}_3$  is a "better" feature than the other two. The issue is that the value of  $\mathbf{f}_i^T L \mathbf{f}_i$  is affected by the respective norms of  $\mathbf{f}_i$  and  $L$ . This needs to be addressed. The relation between  $L$  and  $\mathcal{L}$  yields

$$\mathbf{f}_i^T L \mathbf{f}_i = \mathbf{f}_i^T D^{1/2} \mathcal{L} D^{1/2} \mathbf{f}_i = (D^{1/2} \mathbf{f}_i)^T \mathcal{L} (D^{1/2} \mathbf{f}_i).$$

Set  $\tilde{\mathbf{f}}_i = (D^{1/2} \mathbf{f}_i)$  and  $\hat{\mathbf{f}}_i = \tilde{\mathbf{f}}_i / \|\tilde{\mathbf{f}}_i\|$ . The **feature score metric**  $\varphi_1$  is a normalized version of the smoothness measure:

$$\varphi_1(\mathbf{f}_i) = \hat{\mathbf{f}}_i^T \mathcal{L} \hat{\mathbf{f}}_i, \quad i = 1, \dots, p.$$

In the current example, the feature scores are

$$\varphi_1(\mathbf{f}_1) = 0.01, \quad \varphi_1(\mathbf{f}_2) = 0.01, \quad \varphi_1(\mathbf{f}_3) = 0.28.$$

This makes more sense, as the pattern is similar to the pattern obtained for the eigenvalues:  $\mathbf{f}_1, \mathbf{f}_2$ , being able to differentiate the clusters, have smaller  $\varphi_1$  scores than  $\mathbf{f}_3$ .

The scoring function can also be defined using the spectral decomposition of  $\mathcal{L}$ . Suppose that  $(\lambda_k, \xi_k)$ ,  $1 \leq k \leq n$  are **eigenpairs** of  $\mathcal{L}$  and let  $\alpha_k = \hat{\mathbf{f}}_i^T \xi_k$ , for a given  $i$ . Then

$$\varphi_1(\mathbf{f}_i) = \sum_{k=1}^n \alpha_k^2 \lambda_k, \quad \text{where} \quad \sum_{k=1}^n \alpha_k^2 = 1.$$

Indeed, let  $\mathcal{L} = \mathbf{U} \mathbf{\Sigma} \mathbf{U}^T$  be the eigendecomposition of  $\mathcal{L}$ . By construction,  $\mathbf{U} = [\xi_1 | \xi_2 | \dots | \xi_n]$  and  $\mathbf{\Sigma} = \text{diag}(\lambda_k)$ , so that

$$\begin{aligned} \varphi_1(\mathbf{f}_i) &= \hat{\mathbf{f}}_i^T \mathcal{L} \hat{\mathbf{f}}_i = \hat{\mathbf{f}}_i^T \mathbf{U} \mathbf{\Sigma} \mathbf{U}^T \hat{\mathbf{f}}_i \\ &= (\alpha_1, \dots, \alpha_n) \mathbf{\Sigma} (\alpha_1, \dots, \alpha_n)^T = \sum_{k=1}^n \alpha_k^2 \lambda_k. \end{aligned}$$

This representation allows for a better comprehension of the  $\varphi_1$  score;  $\alpha_k$  is the cosine of the angle between the normalized feature  $\hat{\mathbf{f}}_i$  and eigenvector  $\xi_k$ . If a feature aligns with "good" eigenvectors (i.e., those with small eigenvalues), its  $\varphi_1$  score will also be small.

Returning to the current example, while the score of the useless feature 3,  $\varphi_1(\mathbf{f}_3)$  is larger than the other scores, it is still small when compared to the eigenvalues of  $L$ . This is due to the fact the  $\mathbf{f}_3$  and  $\xi_1$  are nearly co-linear. The larger  $\alpha_1^2$  is, the smaller  $\sum_{k=2}^n \alpha_k^2$  is; this is problematic because, in such cases, a small value of  $\varphi_1$  indicates smoothness but not separability. To overcome this issue,  $\varphi_1$  can be normalized by  $\sum_{k=2}^n \alpha_k^2$ , which yields a new scoring function:

$$\varphi_2(\mathbf{f}_i) = \frac{\sum_{k=1}^n \alpha_k^2 \lambda_k}{\sum_{k=2}^n \alpha_k^2} = \frac{\hat{\mathbf{f}}_i^T \mathcal{L} \hat{\mathbf{f}}_i}{1 - (\hat{\mathbf{f}}_i^T \xi_1)}$$

A small value for  $\varphi_2$  indicates that a feature closely aligns with "good" eigenvectors.

Computing  $\varphi_2$  for our three features yields:

$$\varphi_2(\mathbf{f}_1) = 0.04, \quad \varphi_2(\mathbf{f}_2) = 0.03, \quad \varphi_2(\mathbf{f}_3) = 0.94.$$

The distinction between the real features and the random, useless one is far greater with  $\varphi_2$ .

Another ranking feature is closely related to the other two. According to spectral clustering, the first  $k$  non-trivial eigenvectors form an optimal set of soft cluster indicators that separate the graph  $G$  into  $k$  connected parts. Therefore, define  $\varphi_3$  as :

$$\varphi_3(\mathbf{f}_i, k) = \sum_{j=2}^k (2 - \lambda_j) \alpha_j^2.$$

Contrary to the other scoring functions,  $\varphi_3$  assigns larger value to feature that are more relevant. It also prioritizes the leading eigenvectors, which helps to reduce noise. Using this ranking function requires a number of categories or clusters  $k$  to be selected (depending on the nature of the ultimate task at hand); if this value is unknown,  $k$  becomes a hyper-parameter to be tuned. In the current example, there are 3 clusters by design; setting  $k = 3$  yields

$$\varphi_3(\mathbf{f}_1) = 0.58, \quad \varphi_3(\mathbf{f}_2) = 0.58, \quad \varphi_3(\mathbf{f}_3) = 0.00.$$

**Regularization**

There is one glaring problem with the ranking functions that have been defined in the previous subsection: they all assume the existence of a gap between subsets of "large" and "small" eigenvalues. For clearly separated data, that is to be expected; but in noisy data, this gap may be reduced, which leads to an increase in the score value of poor features [33]. This issue can be tackled by applying a **spectral matrix function**  $\gamma(\cdot)$  to the Laplacian  $\mathcal{L}$ , replacing the original eigenvalues by **regularized eigenvalues** as follows:

$$\gamma(\mathcal{L}) = \sum_{j=1}^n \gamma(\lambda_j) \xi_j \xi_j^T.$$

In order For this to work properly,  $\gamma$  needs to be (strictly) increasing. Examples of such regularization functions include:

$\gamma(\lambda)$	(name)
$1 + \sigma^2 \lambda$	(regularized Laplacian)
$\exp(\sigma^2/2\lambda)$	(diffusion Process)
$\lambda^\nu, \nu \geq 2$	(high-order polynomial)
$(a - \lambda)^{-p}, a \geq 2$	( $p$ -step random walk)
$(\cos \lambda\pi/4)^{-1}$	(inverse cosine)

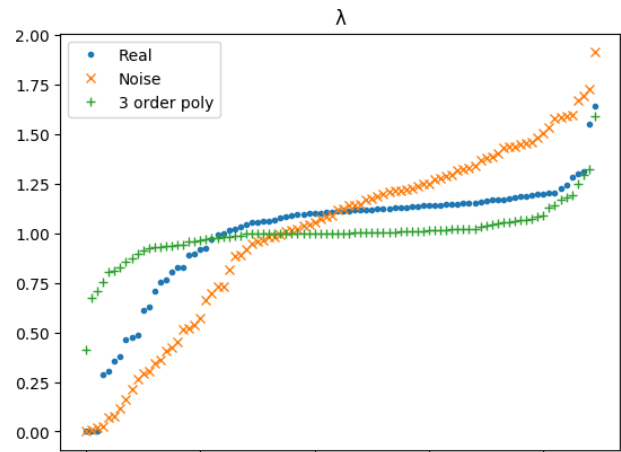
The ranking function  $\varphi_1, \varphi_2, \varphi_3$  can be regularized via

$$\hat{\varphi}_1(\mathbf{f}_i) = \sum_{k=1}^n \alpha_k^2 \gamma(\lambda_k)$$

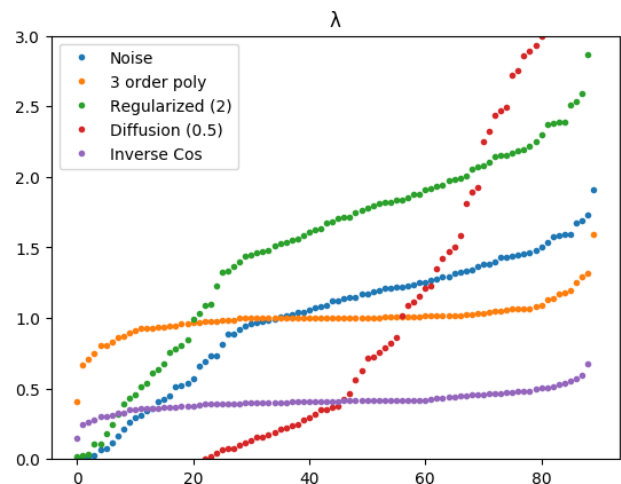
$$\hat{\varphi}_2(\mathbf{f}_i) = \frac{\hat{\mathbf{f}}_i^T \gamma(\mathcal{L}) \hat{\mathbf{f}}_i}{1 - (\hat{\mathbf{f}}_i^T \xi_1)}$$

$$\hat{\varphi}_3(\mathbf{f}_i) = \sum_{j=2}^k (\gamma(2) - \gamma(\lambda_j)) \alpha_j^2$$

To illustrate how this regularization process can help reduce noise (still using the framework from the previous example),  $\mathbf{X}$  was contaminated with random values from a normal distribution with a variance of 1.1. The normalized Laplacians of the original and of the contaminated data were then computed. Figure 22 shows the effect of noise on  $\mathcal{L}$ 's: it tends to linearize the eigenvalues, and this provides much support to the poorer eigenvectors. In the same figure, the eigenvalues of the noisy Laplacian have



**Figure 22.** Effect of noise on the eigenvalues of the normalized Laplacian.



**Figure 23.** Effect of different regularization function on a the eigenvalues of the Laplacian of a noisy dataset.

been regularized using the standard cubic  $\gamma(\lambda) = \lambda^3$ ; the distinction between the first eigenvalues and the rest is clear.

To effect of other regularisation functions is shown in Figure 23. The choice of a specific regularization function depends on the context and the goals of the data analysis task; for large datasets, considerations of ease of computation may also form part of the selection strategy.

**Spectral Feature Selection with SPEC**

The remarks from the previous subsections can be combined to create a feature selection framework called SPEC [30]:

1. using a specified similarity function  $s$ , construct a similarity matrix  $S$  of the data  $\mathbf{X}$  (optionally with labels  $\mathbf{Y}$ );
2. construct the graph  $G$  of the data;
3. extract the normalized Laplacian  $\mathcal{L}$  from this graph;
4. compute the eigenpairs (eigenvalues and eigenvectors) of  $\mathcal{L}$ ;

5. select a regularization function  $\gamma(\cdot)$ ;
6. for each feature  $\mathbf{f}_i$ ,  $i = 1, \dots, p$ , compute the relevance  $\hat{\varphi}(\mathbf{f}_i)$ , where  $\hat{\varphi} \in \{\hat{\varphi}_1, \hat{\varphi}_2, \hat{\varphi}_3\}$ , and
7. return the features in descending order of relevance.

In order for SPEC to provide "good" results, proper choices for the similarity, ranking, and regularization functions are needed. Among other considerations, the similarity matrix should reflect the true relationships between the observations. Furthermore, if the data is noisy, it might be helpful to opt for  $\hat{\varphi} = \hat{\varphi}_3$  and/or  $\gamma(\lambda) = \lambda^\nu$ ,  $\nu \geq 2$ . Finally, when the gap between the small and the large eigenvalues is wide,  $\hat{\varphi} = \hat{\varphi}_2$  or  $\hat{\varphi} = \hat{\varphi}_3$  provide good choices, although  $\hat{\varphi}_2$  has been shown to be more robust.<sup>29</sup>

### 4.3 Uniform Manifold Approximation and Projection

Quite a lot of attention has been paid in recent years to feature selection and dimension reduction methods, and there is no doubt that the landscape will change a fair amount in the short-to-medium term future. Case in point, consider **Uniform Manifold Approximation and Projection** (UMAP) methods, a recent development which is generating a lot of interest at the moment.

#### Dimensionality Reduction and UMAP

A mountain is certainly a 3-dimensional object. And the surface of a mountain range is 2-dimensional – it can be represented with a flat map – even though the surface, and the map for that matter, still actually exist in 3-dimensional space.

What does it mean to say that a shape is  $q$ -dimensional for some  $q$ ? An answer to that question first requires an understanding of what a **shape** is.

Shapes could be lines, cubes, spheres, polyhedrons, or more complicated things. In geometry, the customary way to represent a shape is *via* a set of points  $S \subseteq \mathbb{R}^p$ . For instance, a circle is the set of points whose distance to a fixed point (the centre) is exactly equal to the radius  $r$ , whereas a disk is the set of points whose distance to the centre is at most the radius.

In the mountain example,  $p = 3$  for both the mountains  $S_m$  and the mountain surface  $S_s$ . So the question is, when is the (effective) dimension of a set  $S$  less than  $p$ , and how is that dimension calculated?

It turns out that there are many definitions of the **dimension**  $q$  of a set  $S \subseteq \mathbb{R}^p$ , such as [13, §III.17]:

- the smallest  $q$  such that  $S$  is  $q$ -dimensional manifold;
- how many nontrivial  $n^{\text{th}}$  homology groups of  $S$  there are;
- how the “size” of the set scales as the coordinates scale.

<sup>29</sup>The Python code for the examples in this section is available in the appendix, pp.39-48 and in the accompanying Python notebook.

A  **$q$ -dimensional manifold** is a set where each small area is approximately the same as a small area of  $\mathbb{R}^q$ . For instance, if a small piece of a (stretchable) sphere is cut out with a cookie cutter, it could theoretically be bent so that it looks like it came from a flat plane, without changing its “essential” shape.

Dimensionality reduction is more than just a matter of selecting a definition and computing  $q$ , however.

Any dataset  $\mathbf{X}$  is necessarily finite and is thus, by definition, actually 0-dimensional; the object of interest is the shape that the data **would** form if there were infinitely many available data points, or, in other words, the **support of the distribution** generating the data.

Furthermore, any dataset is probably noisy and may only **approximately lie** in a lower-dimensional shape.

Lastly, it is not clear how to build an algorithm that would, for example, determine what all the **homology groups** of some set  $S$  are. The problem is quite thorny.

Let  $X \subseteq \mathbb{R}^p$  be a finite set of points. A **dimensionality reducer** for  $\mathbf{X}$  is a function  $f_X : \mathbf{X} \rightarrow \mathbb{R}^q$  for some  $q$  which satisfy certain properties that imply that  $f_X(\mathbf{X})$  has similar structure to  $\mathbf{X}$ .<sup>30</sup> Various dimensionality reducers were discussed in Section 2; they each differ based on the relationship between  $\mathbf{X}$  to  $f_X(\mathbf{X})$ .

For instance, in PCA, the dataset  $\mathbf{X}$  is first translated, so that its points (or at least its “principal components”) lie in a linear subspace. Then  $q$  unit-length linear basis elements are chosen to span a subspace, projection onto which yields an affine map  $f$  from  $\mathbf{X}$  to  $\mathbb{R}^q$  that preserves Euclidean distances between points (a rigid transformation), assuming that the non-principal dimensions are ignored.

PCA seems reasonable but what if a rigid transformation down to  $\mathbb{R}^q$  is not possible? As an example, consider the **swiss roll** of Figure 8, which is a loosely rolled up rectangle in 3-dimensional space. If all structure cannot be preserved, what can be preserved? Only local structure? Global structure?

UMAP is a dimension reduction method that attempts to approximately preserve both local and global structure. It can be especially useful for visualization purposes, i.e. reducing to  $q = 3$  or fewer dimensions. While the semantics of UMAP can be stated in terms of graph layouts, the method was derived from abstract topological assumptions. For its full mathematical properties, see [14].

Note that UMAP works best when the data  $\mathbf{X}$  is evenly distributed on its support  $S$ . In this way, the points of  $X$  “cover”  $S$  and UMAP can determine where the true gaps or holes in  $S$  are.

<sup>30</sup>In the remainder of this section, the subscript is dropped. Note that  $q$  is assumed, not found by the process.



**UMAP Semantics**

Let the (scaled) data be denoted by  $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$  where  $\mathbf{x}_i \in \mathbb{R}^p$  for all  $i$ , let  $d : \mathbf{X} \times \mathbf{X} \rightarrow \mathbb{R}_{\geq 0}$  be a distance function, and let  $k \geq 1$  be an integer.

Consider a **directed graph**  $D = (V, E, A)$ , with

- vertices  $V(D) = \mathbf{X}$ ;
- edges  $E(D)$  consisting of the ordered pairs  $(\mathbf{x}_i, \mathbf{x}_j)$  such that  $\mathbf{x}_j$  is one of the  $k$  nearest neighbours of  $\mathbf{x}_i$  according to  $d$ ;
- weight function  $W : E(D) \rightarrow \mathbb{R}$  such that

$$w(\mathbf{x}_i, \mathbf{x}_{i,j}) = \exp\left(\frac{-\max(0, d(\mathbf{x}_i, \mathbf{x}_{i,j}) - \rho_i)}{\sigma_i}\right),$$

where  $\mathbf{x}_{i,1}, \dots, \mathbf{x}_{i,k}$  are the  $k$  nearest neighbours of  $\mathbf{x}_i$  according to  $d$ ,  $\rho_i$  is the minimum nonzero distance from  $\mathbf{x}_i$  to any of its neighbours, and  $\sigma_i$  is the unique real solution of

$$\sum_{j=1}^k \exp\left(\frac{-\max(0, d(\mathbf{x}_i, \mathbf{x}_{i,j}) - \rho_i)}{\sigma_i}\right) = \log_2(k),$$

and

- $A$  is the weighted adjacency matrix of  $D$  with vertex ordering  $\mathbf{x}_1, \dots, \mathbf{x}_n$ .

Define a symmetric matrix

$$B = A + A^T - A \circ A^T,$$

where  $\circ$  is Hadamard's component-wise product. The **weighted graph**  $G = (V, W, B)$  has the same vertex set, the same vertex ordering, and the same edge set as  $D$ , but its edge weights are given by  $B$ . Since  $B$  is symmetric,  $G$  can be considered to be undirected.

UMAP returns the (reduced) points  $f(\mathbf{x}_1), \dots, f(\mathbf{x}_n) \in \mathbb{R}^q$  by finding the position of each vertex in a **force directed graph layout**, which is defined via a graph, an attractive force function defined on edges, and a repulsive force function defined on all pairs of vertices. Both force functions produce **force values** with a direction and magnitude based on the pair of vertices and their respective positions in  $\mathbb{R}^q$ .

To compute the **layout**, initial positions in  $\mathbb{R}^q$  are chosen for each vertex, and an iterative process of translating points based on their attractive and repulsive forces is carried out until a convergence criterion is met.

In UMAP, the attractive force between vertices  $\mathbf{x}_i, \mathbf{x}_j$  at positions  $y_i, y_j \in \mathbb{R}^q$ , respectively, is

$$\frac{-2ab\|y_i - y_j\|_2^{2(b-1)}}{1 + \|y_i - y_j\|_2^2} w(\mathbf{x}_i, \mathbf{x}_j)(y_i - y_j),$$

where  $a$  and  $b$  are parameters, and the repulsive force is

$$\frac{b(1 - w(\mathbf{x}_i, \mathbf{x}_j))(y_i - y_j)}{(0.001 + \|y_i - y_j\|_2^2)(1 + \|y_i - y_j\|_2^2)}.$$

There are a number of important free parameters to select, namely

- $k$ : nearest neighbor neighborhood count;
- $q$ : target dimension;
- $d$ : distance function, e.g. Euclidean metric.

The UMAP documentation states,

low values of  $k$  will force UMAP to concentrate on very local structure (potentially to the detriment of the big picture), while large values will push UMAP to look at larger neighborhoods of each point ... losing fine detail structure for the sake of getting the broader structure of the data. [14]

The user may set these parameters to appropriate values for the dataset. The choice of a distance metric plays the same role as in clustering, where closer pairs of points are considered to be more similar than farther pairs. There is also a minimum distance value used within the force directed layout algorithm which says how close together the positions may be.

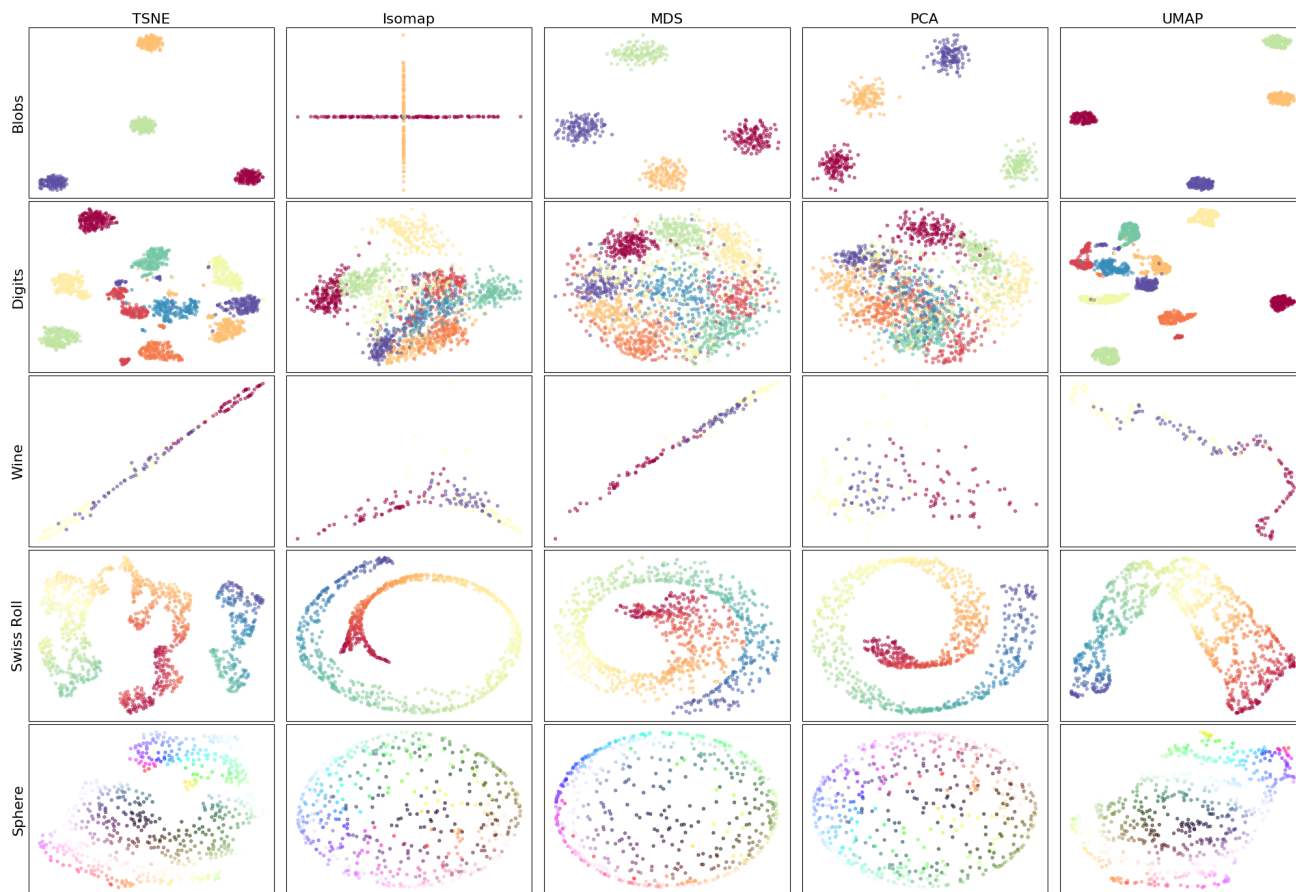
**Example**

A comparison of various dimensionality reducers for a number of datasets is displayed in Figure 24. The Python code for the example is available in the Appendix, on pp. 48-50.

**References**

- [1] Guyon, I., Elisseeff, A., An Introduction to Variable and Feature Selection, Journal of Machine Learning Research, 3(Mar):1157-1182, 2003.
- [2] Cawley, G.C., Talbot, N.L.C., Gene selection in cancer classification using sparse logistic regression with Bayesian regularization, Bioinformatics, (2006) 22 (19): 2348-2355.
- [3] Ambroise, C., McLachlan, G.J., Selection bias in gene extraction on the basis of microarray gene-expression data, PNAS, vol.99, no.10, pp.6562-6566, 2002. Liu, H., Motoda, H. (eds), Computational Methods of Feature Selection, Chapman Hall/ CRC Press.
- [4] Kononenko, I., Kukar, M. [2007], Machine Learning and Data Mining: Introduction to Principles and Algorithms, ch.6, Horwood Publishing.
- [5] **LASSO** on Wikipedia.
- [6] **Nonlinear dimension reduction** on Wikipedia.
- [7] Associated Press [2017], **Sens rally after blowing lead, beat Leafs to gain on Habs**, retrieved from **ESPN.com** on September 20, 2017.
- [8] Natural Stat Trick [2017], **Ottawa Senators @ Toronto Maple Leafs Game Log**, retrieved from **Natural Stat Trick** on September 29, 2017.
- [9] **Macbeth** on Wikipedia.





**Figure 24.** Scatterplots showing datasets reduced to 2 dimensions.

- [10] **Laconic Macbeth** on tvtropes.org.
- [11] W. Morrissette [2001], **Scotland, PA**.
- [12] **An interactive visualization to teach about the curse of dimensionality** from simplystatistics.org.
- [13] Timothy Gowers, June Barrow-Green, and Imre Leader, editors. *The Princeton Companion to Mathematics*. Princeton Univ. Press, 2008.
- [14] Leland McInnes, John Healy, and James Melville. **UMAP: Uniform manifold approximation and projection for dimension reduction**. arXiv preprint 1802.03426, 2018.
- [15] **Mutual information** on Wikipedia.
- [16] **Pearson correlation coefficient** on Wikipedia.
- [17] **Point biserial correlation coefficient** on Wikipedia.
- [18] **Eta-squared** on Wikipedia.
- [19] **Chi-squared test for nominal (categorical) data** on learntech, UWE Bristol.
- [20] Aggarwal, C.C., [2015], *Data Classification: Algorithms and Applications*, ch.2, Chapman and Hall.
- [21] Aggarwal, C.C., Reddy, C.K., [2015]. *Data Clustering: Algorithms and Applications*, ch.2, Chapman and Hall.
- [22] Aggarwal, C.C., [2015], *Data Mining: The Textbook*, Springer.
- [23] Sun, Yijun, Wu, Dapeng. [2008]. A RELIEF Based Feature Extraction Algorithm. 188-195. 10.1137/1.9781611972788.17.
- [24] **Dimensionality Reduction A Short Tutorial**, by Ali Ghodsi
- [25] **Algorithms for manifold learning**, L. Cayton
- [26] **A Global Geometric Framework for Nonlinear Dimensionality Reduction**, by J. Tenenbaum, V. Silva and J. Langford
- [27] **Nonlinear Dimensionality Reduction by Locally Linear Embedding**, S.Roweis, L. Saul
- [28] **Manifold Learning** on scikit-learn.org.
- [29] **Manifold learning on handwritten digits** on scikit-learn.org.
- [30] Smola, A. J., Kondor, R. (2003). *Kernels and regularization on graphs*. In *Learning theory and kernel machines* (pp. 144-158). Springer, Berlin, Heidelberg.
- [31] Zhao, Z.A., Liu, H. [2011], *Spectral Feature Selection for Data Mining*, CRC Press.

- [32] Duvenaud, D.K. [2014], *Automatic Model Construction with Gaussian Processes*, Ph.D. Thesis, University of Cambridge.
- [33] Zhang, Tong, and Rie Kubota Ando. "Analysis of spectral kernel design based semi-supervised learning." *Advances in neural information processing systems*. 2006.
- [34] Hastie, T., Tibshirani, R., Friedman, J. [2008], *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed., Springer.
- [35] **t-distributed stochastic neighbor embedding** on Wikipedia.
- [36] LeCun, Y., Cortes, C., Burges, C.J.C., **The MNIST database of handwritten digits**.
- [37] **Globalization and World Cities Research Network**
- [38] **2015 NHL Entry Draft**
- [39] Harrell, F.E. [2015], *Regression Modeling Strategies: With Applications to Linear Models, Logistic and Ordinal Regression, and Survival Analysis*, Springer.
- [40] Flom, P. [2018], **Stopping stepwise: Why stepwise selection is bad and what you should use instead**, on [towardsdatascience.com](http://towardsdatascience.com).
- [41] Hastie, T., Tibshirani, R., Wainwright, M. [2015], *Statistical learning with sparsity: the lasso and generalizations*, Chapman and Hall/CRC.
- [42] Hastie, T., Qian, J. [2014], **glmnet Vignette**.
- [43] **Using LASSO from lars (or glmnet) package in R for variable selection** on Stack Overflow.
- [44] **Practical Guide: PCA - Principal Component Analysis Essentials**, Articles - Principal Component Methods in R, by user kassambara [2017] on [sthda.com](http://sthda.com).
- [45] **Principal Component Analysis in R: prcomp vs princomp**, Articles - Principal Component Methods in R, by user kassambara [2017] on [sthda.com](http://sthda.com).
- [46] Leskovec, J., Rajaraman, A., Ullman, J. [2014], *Mining of Massive Datasets*, Cambridge University Press.
- [47] Goldberg, Y. [2017], *Neural Network Methods for Natural Language Processing*, Morgan and Claypool.

## Play-by-play and Boxscores, Senators @ Maple Leafs, February 18, 2017 [7]

### 1st Period Summary

Time | Team | Detail

0:00 | Start of 1st period

0:00 | TOR | Nazem Kadri won faceoff in neutral zone

0:08 | OTT | Shot on goal by Dion Phaneuf saved by Frederik Andersen

0:17 | Stoppage - Icing

0:17 | OTT | Jean-Gabriel Pageau won faceoff in offensive zone

0:21 | OTT | Shot on goal by Chris Wideman saved by Frederik Andersen

0:21 | Stoppage - Goalie Stopped

0:21 | TOR | Tyler Bozak won faceoff in defensive zone

0:28 | OTT | Chris Wideman shot blocked by James van Riemsdyk

0:35 | TOR | Shot on goal by James van Riemsdyk saved by Craig Anderson

1:11 | OTT | Giveaway by Craig Anderson in defensive zone

1:28 | TOR | Nikita Zaitsev credited with hit on Erik Karlsson in offensive zone

1:45 | OTT | Zack Smith shot blocked by Josh Leivo

2:38 | OTT | Erik Karlsson shot blocked by Nikita Zaitsev

2:40 | TOR | Nikita Zaitsev credited with hit on Erik Karlsson in defensive zone

3:12 | Stoppage - Puck in Benches

3:12 | TOR | Penalty to Roman Polak 2 minutes for Roughing Mark Borowiecki

3:12 | OTT | Penalty to Mark Borowiecki 2 minutes for Roughing Roman Polak

3:12 | TOR | Nazem Kadri won faceoff in neutral zone

3:37 | Stoppage - Offside

3:37 | TOR | Auston Matthews won faceoff in neutral zone

3:47 | TOR | James van Riemsdyk shot blocked by Dion Phaneuf

3:54 | TOR | Shot on goal by James van Riemsdyk saved by Craig Anderson

3:59 | TOR | James van Riemsdyk credited with hit on Cody Ceci in offensive zone

4:03 | OTT | Shot missed by Tom Pyatt

4:06 | Stoppage - Goalie Stopped

4:06 | OTT | Kyle Turris won faceoff in offensive zone

4:11 | OTT | Erik Karlsson shot blocked by Connor Brown

4:25 | OTT | Shot missed by Zack Smith

4:36 | TOR | Shot on goal by Nikita Zaitsev saved by Craig Anderson

4:48 | OTT | Shot on goal by Kyle Turris saved by Frederik Andersen

5:04 | TOR | William Nylander shot blocked by Cody Ceci

5:16 | OTT | Jean-Gabriel Pageau shot blocked by Matt Hunwick

5:23 | OTT | Shot on goal by Bobby Ryan saved by Frederik Andersen

5:31 | OTT | Jean-Gabriel Pageau shot blocked by Matt Hunwick

5:34 | OTT | Shot on goal by Mike Hoffman saved by Frederik Andersen

5:41 | TOR | Shot missed by William Nylander

5:44 | TOR | Shot on goal by Roman Polak saved by Craig Anderson

5:44 | Stoppage - Goalie Stopped

5:44 | OTT | Derick Brassard won faceoff in defensive zone

5:52 | TOR | Shot on goal by Nikita Zaitsev saved by Craig Anderson

5:52 | Stoppage - Goalie Stopped

5:52 | TOR | Shot missed by Nikita Zaitsev

5:52 | TOR | Nazem Kadri won faceoff in offensive zone

6:14 | OTT | Shot on goal by Cody Ceci saved by Frederik Andersen

6:14 | Stoppage - Goalie Stopped - TV timeout

6:14 | OTT | Zack Smith won faceoff in offensive zone

6:24 | TOR | Takeaway by Tyler Bozak in defensive zone

6:32 | TOR | Connor Brown credited with hit on Marc Methot in offensive zone

6:45 | TOR | Takeaway by Tyler Bozak in defensive zone

6:52 | OTT | Chris Neil credited with hit on James van Riemsdyk in offensive zone  
 7:02 | TOR | Shot on goal by Nikita Soshnikov saved by Craig Anderson  
 7:03 | Stoppage - Goalie Stopped  
 7:03 | OTT | Tommy Wingels won faceoff in defensive zone  
 7:20 | OTT | Takeaway by Tom Pyatt in offensive zone  
 7:38 | TOR | Takeaway by Matt Martin in offensive zone  
 8:03 | TOR | Penalty to Jake Gardiner 2 minutes for Slashing Mike Hoffman  
 8:03 | OTT | Power play - Zack Smith won faceoff in offensive zone  
 8:06 | OTT | Power play - Shot on goal by Dion Phaneuf saved by Frederik Andersen  
 8:07 | Stoppage - Goalie Stopped  
 8:07 | TOR | Shorthanded - Ben Smith won faceoff in defensive zone  
 9:08 | TOR | Shorthanded - Takeaway by Nikita Soshnikov in neutral zone  
 9:12 | TOR | Shorthanded - Shot missed by Connor Brown  
 9:26 | TOR | Shorthanded - Shot on goal by Connor Brown saved by Craig Anderson  
 9:49 | TOR | Shorthanded - Shot on goal by Frederik Andersen saved by Craig Anderson  
 10:08 | Stoppage - Icing  
 10:08 | OTT | Jean-Gabriel Pageau won faceoff in offensive zone  
 10:13 | OTT | Shot on goal by Cody Ceci saved by Frederik Andersen  
 10:26 | OTT | Mark Borowiecki credited with hit on Auston Matthews in defensive zone  
 10:39 | TOR | Shot missed by Morgan Rielly  
 10:44 | OTT | Giveaway by Tom Pyatt in defensive zone  
 10:46 | TOR | Shot on goal by Auston Matthews saved by Craig Anderson  
 12:04 | OTT | Shot on goal by Jean-Gabriel Pageau saved by Frederik Andersen  
 12:06 | Stoppage - Goalie Stopped - TV timeout  
 12:06 | TOR | Nazem Kadri won faceoff in defensive zone  
 12:21 | OTT | Erik Karlsson credited with hit on Nazem Kadri in defensive zone  
 12:21 | OTT | Penalty to Tom Pyatt 2 minutes for Hooking Leo Komarov  
 12:21 | OTT | Shorthanded - Jean-Gabriel Pageau won faceoff in defensive zone  
 12:36 | OTT | Shorthanded - Shot missed by Zack Smith  
 12:45 | OTT | Shorthanded - Jean-Gabriel Pageau credited with hit on William Nylander in offensive zone  
 13:49 | OTT | Shorthanded - Takeaway by Marc Methot in neutral zone  
 13:59 | OTT | Penalty to Kyle Turris 2 minutes for Holding Nikita Zaitsev  
 13:59 | TOR | Power play - Tyler Bozak won faceoff in offensive zone  
 14:25 | TOR | Power play - Shot on goal by Morgan Rielly saved by Craig Anderson  
 14:35 | TOR | Power play - Shot on goal by Morgan Rielly saved by Craig Anderson  
 14:40 | TOR | Power play - Shot on goal by Auston Matthews saved by Craig Anderson  
 15:03 | TOR | Power play - Giveaway by Nazem Kadri in offensive zone  
 15:18 | TOR | Power play - Shot on goal by Nazem Kadri saved by Craig Anderson  
 15:27 | TOR | Power play - Shot on goal by Nazem Kadri saved by Craig Anderson  
 15:36 | TOR | Power play - Shot missed by Nikita Zaitsev  
 15:54 | OTT | Shorthanded - Mark Borowiecki credited with hit on William Nylander in defensive zone  
 16:01 | TOR | Auston Matthews shot blocked by Jean-Gabriel Pageau  
 16:09 | TOR | Takeaway by Auston Matthews in offensive zone  
 16:30 | OTT | Dion Phaneuf credited with hit on Ben Smith in defensive zone  
 16:36 | TOR | Shot on goal by Zach Hyman saved by Craig Anderson  
 16:51 | OTT | Takeaway by Mike Hoffman in offensive zone  
 16:53 | OTT | Shot missed by Derick Brassard  
 17:13 | OTT | Shot on goal by Mark Stone saved by Frederik Andersen  
 17:14 | OTT | Shot on goal by Ryan Dzingel saved by Frederik Andersen  
 17:19 | OTT | Giveaway by Ryan Dzingel in offensive zone

17:20 | TOR | Nikita Zaitsev credited with hit on Ryan Dzingel in defensive zone  
 17:22 | OTT | Takeaway by Mark Stone in offensive zone  
 17:26 | OTT | Goal scored by Chris Wideman assisted by Mark Stone and Derick Brassard  
 17:26 | OTT | Kyle Turris won faceoff in neutral zone  
 17:36 | OTT | Bobby Ryan credited with hit on Matt Hunwick in offensive zone  
 17:39 | OTT | Shot on goal by Mike Hoffman saved by Frederik Andersen  
 17:41 | Stoppage - Goalie Stopped - TV timeout  
 17:41 | TOR | Ben Smith won faceoff in defensive zone  
 17:46 | OTT | Goal scored by Ryan Dzingel assisted by Marc Methot and Mark Stone  
 17:46 | OTT | Kyle Turris won faceoff in neutral zone  
 18:14 | Stoppage - Puck in Netting  
 18:14 | OTT | Penalty to Bobby Ryan 2 minutes for Delaying the game  
 18:14 | OTT | Shorthanded - Zack Smith won faceoff in neutral zone  
 18:42 | TOR | Power play - James van Riemsdyk shot blocked by Cody Ceci  
 18:42 | Stoppage - Puck in Netting  
 18:42 | TOR | Power play - Auston Matthews won faceoff in offensive zone  
 18:48 | TOR | Power play - Giveaway by Jake Gardiner in offensive zone  
 19:15 | TOR | Power play - Connor Brown credited with hit on Jean-Gabriel Pageau in offensive zone  
 19:24 | TOR | Power play - Shot missed by William Nylander  
 20:00 | End of 1st period

## 2nd Period Summary

### Time | Team | Detail

0:00 | Start of 2nd period  
 0:00 | TOR | Power play - Auston Matthews won faceoff in neutral zone  
 0:35 | OTT | Takeaway by Marc Methot in defensive zone  
 1:10 | TOR | Josh Leivo credited with hit on Cody Ceci in offensive zone  
 1:16 | OTT | Giveaway by Dion Phaneuf in defensive zone  
 1:18 | OTT | Derick Brassard shot blocked by Nikita Zaitsev  
 1:25 | OTT | Cody Ceci shot blocked by Morgan Rielly  
 2:13 | TOR | Shot on goal by James van Riemsdyk saved by Craig Anderson  
 2:28 | OTT | Shot on goal by Zack Smith saved by Frederik Andersen  
 3:00 | OTT | Shot on goal by Chris Kelly saved by Frederik Andersen  
 3:15 | TOR | Ben Smith credited with hit on Tommy Wingels in offensive zone  
 3:29 | OTT | Cody Ceci credited with hit on Nikita Soshnikov in neutral zone  
 3:38 | OTT | Dion Phaneuf credited with hit on Matt Martin in defensive zone  
 4:05 | TOR | Shot on goal by Auston Matthews saved by Craig Anderson  
 4:05 | Stoppage - Goalie Stopped  
 4:05 | TOR | Nazem Kadri won faceoff in offensive zone  
 4:12 | TOR | Shot missed by Leo Komarov  
 4:20 | OTT | Cody Ceci credited with hit on Josh Leivo in defensive zone  
 4:22 | TOR | Shot on goal by Leo Komarov saved by Craig Anderson  
 4:39 | OTT | Takeaway by Ryan Dzingel in offensive zone  
 4:58 | OTT | Zack Smith credited with hit on Connor Carrick in offensive zone  
 5:01 | OTT | Shot on goal by Marc Methot saved by Frederik Andersen  
 5:19 | Stoppage - Icing  
 5:19 | OTT | Jean-Gabriel Pageau won faceoff in offensive zone  
 5:24 | OTT | Bobby Ryan shot blocked by Connor Brown  
 5:35 | OTT | Tom Pyatt credited with hit on Connor Carrick in offensive zone  
 5:43 | Stoppage - Offside  
 5:43 | OTT | Kyle Turris won faceoff in neutral zone  
 5:47 | OTT | Shot on goal by Dion Phaneuf saved by Frederik Andersen

5:49   Stoppage - Goalie Stopped	14:17   TOR   Takeaway by Connor Brown in offensive zone
5:49   OTT   Kyle Turriss won faceoff in offensive zone	14:20   Stoppage - Icing
5:53   OTT   Shot missed by Mark Borowiecki	14:20   OTT   Kyle Turriss won faceoff in defensive zone
6:16   TOR   Shot on goal by Roman Polak saved by Craig Anderson	14:26   TOR   Shot on goal by Morgan Rielly saved by Craig Anderson
6:37   OTT   Shot on goal by Kyle Turriss saved by Frederik Andersen	14:27   OTT   Giveaway by Erik Karlsson in defensive zone
6:57   TOR   Nikita Zaitsev credited with hit on Derick Brassard in neutral zone	14:33   TOR   Shot missed by Auston Matthews
7:05   TOR   Morgan Rielly credited with hit on Ryan Dzingel in defensive zone	<b>14:38   TOR   Goal scored by Morgan Rielly assisted by William Nylander and Auston Matthews</b>
7:10   Stoppage - Puck in Benches - TV timeout	14:38   OTT   Jean-Gabriel Pageau won faceoff in neutral zone
7:10   OTT   Kyle Turriss won faceoff in defensive zone	14:46   TOR   Matt Martin credited with hit on Chris Kelly in neutral zone
7:19   TOR   Giveaway by Auston Matthews in offensive zone	15:01   TOR   Nikita Soshnikov credited with hit on Dion Phaneuf in offensive zone
7:26   OTT   Shot on goal by Kyle Turriss saved by Frederik Andersen	15:06   TOR   Matt Hunwick shot blocked by Jean-Gabriel Pageau
7:36   OTT   Marc Methot credited with hit on Zach Hyman in defensive zone	15:45   OTT   Mark Borowiecki credited with hit on Morgan Rielly in defensive zone
7:52   TOR   Shot on goal by William Nylander saved by Craig Anderson	15:51   TOR   Takeaway by Josh Leivo in offensive zone
8:04   OTT   Shot on goal by Mark Borowiecki saved by Frederik Andersen	15:52   TOR   Morgan Rielly shot blocked by Mark Borowiecki
8:21   TOR   Connor Brown credited with hit on Cody Ceci in offensive zone	16:00   TOR   Takeaway by Nikita Zaitsev in defensive zone
8:41   Stoppage - Offside - Player Equipment	16:06   TOR   Shot missed by Leo Komarov
8:41   TOR   Ben Smith won faceoff in neutral zone	16:12   TOR   Shot on goal by Nazem Kadri saved by Craig Anderson
9:00   TOR   Roman Polak credited with hit on Tommy Wingels in defensive zone	16:39   TOR   James van Riemsdyk credited with hit on Zack Smith in neutral zone
9:10   TOR   Giveaway by Roman Polak in defensive zone	16:43   OTT   Shot missed by Mike Hoffman
9:19   TOR   Roman Polak credited with hit on Tommy Wingels in defensive zone	17:06   OTT   Zack Smith credited with hit on Connor Carrick in offensive zone
9:21   OTT   Shot on goal by Chris Wideman saved by Frederik Andersen	17:34   OTT   Shot on goal by Cody Ceci saved by Frederik Andersen
<b>9:29   TOR   Penalty to Matt Martin 2 minutes for Interference Bobby Ryan</b>	17:35   Stoppage - Goalie Stopped - TV timeout
<b>9:29   OTT   Power play - Kyle Turriss won faceoff in offensive zone</b>	17:35   TOR   Nazem Kadri won faceoff in defensive zone
<b>9:40   TOR   Penalty to Zach Hyman 2 minutes for Holding Erik Karlsson</b>	17:43   TOR   Shot missed by Josh Leivo
<b>9:40   OTT   Power play - Kyle Turriss won faceoff in offensive zone</b>	17:47   TOR   Shot on goal by Josh Leivo saved by Craig Anderson
<b>10:00   OTT   Power play - Shot missed by Erik Karlsson</b>	<b>17:52   TOR   Goal scored by Nazem Kadri assisted by Josh Leivo</b>
<b>10:24   OTT   Power play - Shot on goal by Mark Stone saved by Frederik Andersen</b>	17:52   TOR   Auston Matthews won faceoff in neutral zone
10:25   Stoppage - Goalie Stopped	17:59   OTT   Marc Methot credited with hit on Connor Carrick in neutral zone
<b>10:25   OTT   Power play - Kyle Turriss won faceoff in offensive zone</b>	18:11   OTT   Marc Methot credited with hit on Zach Hyman in defensive zone
<b>10:43   OTT   Power play - Shot missed by Mike Hoffman</b>	18:22   OTT   Shot on goal by Erik Karlsson saved by Frederik Andersen
<b>10:49   OTT   Power play - Shot on goal by Mike Hoffman saved by Frederik Andersen</b>	18:48   OTT   Shot on goal by Mike Hoffman saved by Frederik Andersen
10:51   Stoppage - Puck in Netting	18:48   Stoppage - Goalie Stopped
<b>10:51   TOR   Shorthanded - Leo Komarov won faceoff in defensive zone</b>	18:48   TOR   Tyler Bozak won faceoff in defensive zone
<b>11:20   OTT   Power play - Shot on goal by Mike Hoffman saved by Frederik Andersen</b>	19:06   TOR   Nikita Zaitsev shot blocked by Bobby Ryan
11:20   Stoppage - Goalie Stopped	19:07   OTT   Mark Borowiecki credited with hit on Connor Brown in defensive zone
<b>11:20   OTT   Power play - Derick Brassard won faceoff in offensive zone</b>	19:15   TOR   Shot missed by Morgan Rielly
<b>11:32   OTT   Power play - Shot missed by Chris Wideman</b>	19:42   TOR   Leo Komarov credited with hit on Erik Karlsson in offensive zone
11:43   OTT   Shot missed by Ryan Dzingel	19:49   TOR   Nikita Zaitsev shot blocked by Erik Karlsson
11:50   OTT   Shot missed by Derick Brassard	19:52   TOR   Leo Komarov credited with hit on Mark Stone in offensive zone
11:50   Stoppage - Puck in Netting - TV timeout	19:58   TOR   Nikita Zaitsev credited with hit on Zack Smith in neutral zone
11:50   OTT   Jean-Gabriel Pageau won faceoff in neutral zone	20:00   End of 2nd period
11:59   OTT   Mark Borowiecki credited with hit on Connor Carrick in neutral zone	
12:03   OTT   Jean-Gabriel Pageau won faceoff in neutral zone	
12:03   Stoppage - Offside	
12:13   OTT   Shot on goal by Cody Ceci saved by Frederik Andersen	
12:13   Stoppage - Goalie Stopped	
12:13   TOR   Auston Matthews won faceoff in defensive zone	
12:40   Stoppage - Offside	
12:40   OTT   Derick Brassard won faceoff in neutral zone	
13:41   OTT   Shot on goal by Dion Phaneuf saved by Frederik Andersen	
13:42   Stoppage - Goalie Stopped	
13:42   TOR   Tyler Bozak won faceoff in defensive zone	
13:54   OTT   Marc Methot credited with hit on Tyler Bozak in defensive zone	

### 3rd Period Summary

#### Time | Team | Detail

0:00   Start of 3rd period
0:00   OTT   Derick Brassard won faceoff in neutral zone
0:24   OTT   Shot on goal by Erik Karlsson saved by Frederik Andersen
0:31   OTT   Shot on goal by Ryan Dzingel saved by Frederik Andersen
0:31   Stoppage - Goalie Stopped
0:31   TOR   Auston Matthews won faceoff in defensive zone
<b>0:45   OTT   Penalty to Zack Smith 2 minutes for Hooking Zach Hyman</b>
<b>0:45   OTT   Shorthanded - Jean-Gabriel Pageau won faceoff in defensive zone</b>
<b>1:13   TOR   Power play - Nikita Zaitsev shot blocked by Jean-Gabriel Pageau</b>
<b>1:20   TOR   Power play - Shot on goal by Tyler Bozak saved by Craig Anderson</b>



1:21 | Stoppage - Goalie Stopped

1:21 | OTT | **Shorthanded - Kyle Turris won faceoff in defensive zone**

1:45 | TOR | **Power play - Giveaway by Jake Gardiner in offensive zone**

**2:04 | TOR | Power Play Goal Scored by William Nylander assisted by Leo Komarov and Auston Matthews**

2:04 | OTT | Kyle Turris won faceoff in neutral zone

2:26 | TOR | Shot on goal by Tyler Bozak saved by Craig Anderson

2:37 | TOR | James van Riemsdyk shot blocked by Kyle Turris

3:21 | TOR | Takeaway by Nazem Kadri in offensive zone

3:32 | TOR | Roman Polak credited with hit on Ryan Dzingel in defensive zone

3:52 | OTT | Mark Borowiecki credited with hit on Ben Smith in defensive zone

3:52 | TOR | Ben Smith shot blocked by Mark Borowiecki

3:58 | OTT | Mark Borowiecki credited with hit on Ben Smith in defensive zone

4:01 | TOR | Connor Carrick shot blocked by Kyle Turris

4:12 | TOR | Shot on goal by Jake Gardiner saved by Craig Anderson

4:13 | Stoppage - Goalie Stopped - Ice problem

4:13 | TOR | Auston Matthews won faceoff in offensive zone

4:17 | TOR | Shot missed by Jake Gardiner

4:17 | Stoppage - Goalie Stopped

4:17 | OTT | Jean-Gabriel Pageau won faceoff in defensive zone

4:24 | Stoppage - Icing

4:24 | OTT | Jean-Gabriel Pageau won faceoff in defensive zone

4:38 | OTT | Jean-Gabriel Pageau credited with hit on Connor Carrick in offensive zone

4:41 | OTT | Shot on goal by Cody Ceci saved by Frederik Andersen

5:28 | TOR | Morgan Rielly credited with hit on Kyle Turris in defensive zone

**5:32 | OTT | Goal scored by Mike Hoffman assisted by Erik Karlsson and Kyle Turris**

5:32 | TOR | Nazem Kadri won faceoff in neutral zone

5:59 | OTT | Shot on goal by Mark Stone saved by Frederik Andersen

6:12 | Stoppage - Referee or Linesman - TV timeout

**6:12 | TOR | Penalty to Nazem Kadri 2 minutes for Holding Chris Wideman**

6:12 | OTT | **Power play - Kyle Turris won faceoff in offensive zone**

6:23 | OTT | **Power play - Shot on goal by Erik Karlsson saved by Frederik Andersen**

6:25 | OTT | **Power play - Shot on goal by Mark Stone saved by Frederik Andersen**

**6:26 | OTT | Power Play Goal Scored by Derick Brassard assisted by Mark Stone and Erik Karlsson**

6:26 | OTT | Jean-Gabriel Pageau won faceoff in neutral zone

6:49 | OTT | Shot on goal by Chris Kelly saved by Frederik Andersen

7:35 | OTT | Kyle Turris shot blocked by Jake Gardiner

7:45 | TOR | Shot on goal by Tyler Bozak saved by Craig Anderson

7:46 | Stoppage - Goalie Stopped

7:46 | TOR | Nazem Kadri won faceoff in offensive zone

7:51 | TOR | Shot on goal by Leo Komarov saved by Craig Anderson

8:08 | TOR | Takeaway by Josh Leivo in offensive zone

8:13 | TOR | Nikita Zaitsev shot blocked by Mark Stone

8:14 | Stoppage - Puck in Netting

8:14 | OTT | Kyle Turris won faceoff in defensive zone

8:24 | TOR | Shot on goal by Auston Matthews saved by Craig Anderson

8:25 | Stoppage - Goalie Stopped

8:25 | TOR | Auston Matthews won faceoff in offensive zone

8:48 | TOR | William Nylander shot blocked by Cody Ceci

9:08 | OTT | Takeaway by Zack Smith in defensive zone

9:21 | Stoppage - Offside

9:21 | TOR | Nazem Kadri won faceoff in neutral zone

9:47 | OTT | Takeaway by Mark Stone in neutral zone

9:48 | OTT | Mark Stone shot blocked by Morgan Rielly

9:59 | OTT | Mark Stone credited with hit on Morgan Rielly in neutral zone

10:40 | TOR | James van Riemsdyk shot blocked by Bobby Ryan

10:59 | OTT | Chris Kelly credited with hit on Connor Carrick in offensive zone

11:05 | TOR | Shot on goal by Nikita Soshnikov saved by Craig Anderson

11:18 | OTT | Chris Kelly credited with hit on Connor Carrick in defensive zone

11:32 | TOR | Jake Gardiner shot blocked by Tom Pyatt

11:41 | TOR | Matt Martin credited with hit on Cody Ceci in offensive zone

11:50 | OTT | Jean-Gabriel Pageau credited with hit on Jake Gardiner in defensive zone

12:13 | OTT | Giveaway by Erik Karlsson in defensive zone

12:18 | TOR | Giveaway by Auston Matthews in offensive zone

12:47 | OTT | Mark Borowiecki credited with hit on Leo Komarov in defensive zone

13:05 | OTT | Mark Stone credited with hit on Nazem Kadri in neutral zone

13:33 | Stoppage - Icing

13:33 | TOR | Tyler Bozak won faceoff in defensive zone

14:03 | OTT | Chris Kelly credited with hit on Connor Carrick in offensive zone

14:04 | OTT | Takeaway by Chris Kelly in offensive zone

14:05 | OTT | Shot on goal by Chris Kelly saved by Frederik Andersen

14:06 | Stoppage - Goalie Stopped - TV timeout

14:06 | TOR | Auston Matthews won faceoff in defensive zone

14:23 | TOR | Shot missed by Zach Hyman

14:40 | TOR | Zach Hyman shot blocked by Marc Methot

14:44 | TOR | Shot missed by William Nylander

14:55 | TOR | Shot missed by Zach Hyman

14:58 | TOR | Takeaway by Zach Hyman in offensive zone

15:03 | TOR | Jake Gardiner shot blocked by Bobby Ryan

15:27 | TOR | Shot on goal by Leo Komarov saved by Craig Anderson

15:27 | Stoppage - Goalie Stopped - TV timeout

15:27 | TOR | Auston Matthews won faceoff in offensive zone

15:31 | TOR | Shot on goal by Morgan Rielly saved by Craig Anderson

15:32 | Stoppage - Goalie Stopped

15:32 | OTT | Jean-Gabriel Pageau won faceoff in defensive zone

16:49 | OTT | Shot missed by Mike Hoffman

16:53 | OTT | Zack Smith credited with hit on Tyler Bozak in offensive zone

17:08 | OTT | Derick Brassard credited with hit on Nazem Kadri in neutral zone

17:14 | TOR | Leo Komarov credited with hit on Mark Borowiecki in offensive zone

17:29 | TOR | Takeaway by Josh Leivo in offensive zone

17:34 | TOR | Shot on goal by Nazem Kadri saved by Craig Anderson

17:35 | Stoppage - Goalie Stopped

17:35 | TOR | Auston Matthews won faceoff in offensive zone

17:52 | TOR | Shot on goal by Zach Hyman saved by Craig Anderson

17:57 | OTT | Giveaway by Zack Smith in defensive zone

18:03 | TOR | Shot missed by Auston Matthews

18:03 | Stoppage - Puck in Netting

18:03 | OTT | Kyle Turris won faceoff in defensive zone

18:06 | Stoppage - Puck in Netting

18:06 | TOR | Tyler Bozak won faceoff in offensive zone

**18:10 | OTT | Goal scored by Mark Stone assisted by Kyle Turris**

18:10 | OTT | Jean-Gabriel Pageau won faceoff in neutral zone

**19:15 | OTT | Goal scored by Derick Brassard assisted by Kyle Turris and Mark Stone**

19:15 | TOR | Ben Smith won faceoff in neutral zone

20:00 | End of 3rd period; End of Game



1st Period Summary		
Time	Team	Detail
0:00		Start of 1st period
0:00	Toronto	Nazem Kadri won faceoff in neutral zone
0:08	Ottawa	Shot on goal by Dion Phaneuf saved by Frederik Andersen
0:17		Stoppage - Icing
0:17	Ottawa	Jean-Gabriel Pageau won faceoff in offensive zone
0:21	Ottawa	Shot on goal by Chris Wideman saved by Frederik Andersen
0:21		Stoppage - Goalie Stopped
0:21	Toronto	Tyler Bozak won faceoff in defensive zone
0:28	Ottawa	Chris Wideman shot blocked by James van Riemsdyk
0:35	Toronto	Shot on goal by James van Riemsdyk saved by Craig Anderson
1:11	Ottawa	Giveaway by Craig Anderson in defensive zone
1:28	Toronto	Nikita Zaitsev credited with hit on Erik Karlsson in offensive zone
1:45	Ottawa	Zack Smith shot blocked by Josh Leivo
2:38	Ottawa	Erik Karlsson shot blocked by Nikita Zaitsev
...		
17:41	Toronto	Ben Smith won faceoff in defensive zone
<b>17:46</b>	<b>Ottawa</b>	<b>Goal scored by Ryan Dzingel assisted by Marc Methot and Mark Stone</b>
17:46	Ottawa	Kyle Turris won faceoff in neutral zone
18:14		Stoppage - Puck in Netting
18:14	Ottawa	Penalty to Bobby Ryan 2 minutes for Delaying the game
18:14	Ottawa	Shorthanded - Zack Smith won faceoff in neutral zone
18:42	Toronto	Power play - James van Riemsdyk shot blocked by Cody Ceci
18:42		Stoppage - Puck in Netting
18:42	Toronto	Power play - Auston Matthews won faceoff in offensive zone
18:48	Toronto	Power play - Giveaway by Jake Gardiner in offensive zone
19:15	Toronto	Power play - Connor Brown credited with hit on Jean-Gabriel Pageau in offensive zone
19:24	Toronto	Power play - Shot missed by William Nylander
20:00		End of 1st period
2nd Period Summary		
Time	Team	Detail
0:00		Start of 2nd period
0:00	Toronto	Power play - Auston Matthews won faceoff in neutral zone
0:35	Ottawa	Takeaway by Marc Methot in defensive zone
1:10	Toronto	Josh Leivo credited with hit on Cody Ceci in offensive zone
1:16	Ottawa	Giveaway by Dion Phaneuf in defensive zone
...		
17:35		Stoppage - Goalie Stopped
17:35	Toronto	Auston Matthews won faceoff in offensive zone
17:52	Toronto	Shot on goal by Zach Hyman saved by Craig Anderson
17:57	Ottawa	Giveaway by Zack Smith in defensive zone
18:03	Toronto	Shot missed by Auston Matthews
18:03		Stoppage - Puck in Netting
18:03	Ottawa	Kyle Turris won faceoff in defensive zone
18:06		Stoppage - Puck in Netting
18:06	Toronto	Tyler Bozak won faceoff in offensive zone
<b>18:10</b>	<b>Ottawa</b>	<b>Goal scored by Mark Stone assisted by Kyle Turris</b>
18:10	Ottawa	Jean-Gabriel Pageau won faceoff in neutral zone
<b>19:15</b>	<b>Ottawa</b>	<b>Goal scored by Derick Brassard assisted by Kyle Turris and Mark Stone</b>
19:15	Toronto	Ben Smith won faceoff in neutral zone
20:00		End of 3rd period
20:00		End of Game

Table 4. Play-by-play extract, Ottawa Senators @ Toronto Maple Leafs, February 18, 2017 [7].

Game Information	
<b>Arena:</b> Air Canada Centre	<b>Referees:</b> Brian Pochmara, Brad Watson
<b>Location:</b> Toronto, Ontario	<b>Linesmen:</b> Bevan Mills, Scott Driscoll
<b>Attendance:</b> 19,527 (103.9% full)	

Team Statistical Comparison					
Key:  Ottawa  Toronto					
<b>Total Shots</b>	<b>PIM</b>	<b>Hits</b>	<b>Giveaways</b>	<b>Takeaways</b>	<b>Faceoffs Won</b>
 42	 10	 32	 7	 9	 36
 37	 10	 23	 6	 12	 31




















1st Period Summary				
Time	Team	Scoring Detail	OTT	TOR
17:26		Chris Wideman (4) Assists: Derick Brassard, Mark Stone	1	0
17:46		Ryan Dzingel (12) Assists: Marc Methot, Mark Stone	2	0
Time	Team	Penalty Detail		
3:12		Roman Polak: 2 Minutes for Roughing		
3:12		Mark Borowiecki: 2 Minutes for Roughing		
8:03		Jake Gardiner: 2 Minutes for Slashing		
12:21		Tom Pyatt: 2 Minutes for Hooking		
13:59		Kyle Turris: 2 Minutes for Holding		
18:14		Bobby Ryan: 2 Minutes for Delaying Game - Puck over Glass		
2nd Period Summary				
Time	Team	Scoring Detail	OTT	TOR
14:38		Morgan Rielly (3) Assists: Auston Matthews, William Nylander	2	1
17:52		Nazem Kadri (24) Assist: Josh Leivo	2	2
Time	Team	Penalty Detail		
9:29		Matt Martin: 2 Minutes for Interference		
9:40		Zach Hyman: 2 Minutes for Holding		
3rd Period Summary				
Time	Team	Scoring Detail	OTT	TOR
2:04		William Nylander (16) (Power Play) Assists: Auston Matthews, Leo Komarov	2	3
5:32		Mike Hoffman (19) Assists: Kyle Turris, Erik Karlsson	3	3
6:26		Derick Brassard (10) (Power Play) Assists: Mark Stone, Erik Karlsson	4	3
18:10		Mark Stone (21) Assist: Kyle Turris	5	3
19:15		Derick Brassard (11) Assists: Kyle Turris, Mark Stone	6	3
Time	Team	Penalty Detail		
0:45		Zack Smith: 2 Minutes for Hooking		
6:12		Nazem Kadri: 2 Minutes for Holding		

Table 5. Advanced Boxscore (part 1), Ottawa Senators @ Toronto Maple Leafs, February 18, 2017 [7].

Player Summary																								
Ottawa Senators															Time On Ice					Faceoffs				
Player	G	A	+/-	SOG	MS	BS	PN	PIM	HT	TK	GV	SHF	TOT	PP	SH	EV	FW	FL	%					
M. Borowiecki D	0	0	1	1	1	2	1	2	8	0	0	23	12:30	0:00	0:36	11:54	0	0	00.0					
D. Brassard C	2	1	3	2	2	0	0	0	1	0	0	28	15:43	1:52	0:00	13:51	4	11	26.7					
C. Ceci D	0	0	0	5	0	3	0	0	2	0	0	32	23:06	0:00	4:13	18:53	0	0	00.0					
R. Dzingel C	1	0	1	3	1	0	0	0	0	1	1	19	11:56	1:10	0:00	10:46	0	0	00.0					
M. Hoffman LW	1	0	1	6	3	0	0	0	0	1	0	23	15:34	2:55	0:00	12:39	0	0	00.0					
E. Karlsson D	0	2	2	3	1	1	0	0	1	0	2	33	24:57	3:49	2:44	18:24	0	0	00.0					
C. Kelly C	0	0	0	3	0	0	0	0	3	1	0	18	12:37	0:00	2:13	10:24	0	3	00.0					
M. Methot D	0	1	2	1	0	1	0	0	4	2	0	32	21:12	0:00	2:52	18:20	0	0	00.0					
C. Neil RW	0	0	0	0	0	0	0	0	1	0	0	6	3:55	0:00	0:00	3:55	0	0	00.0					
J. Pageau C	0	0	0	1	0	3	0	0	3	0	0	30	18:17	0:36	3:57	13:44	13	6	68.4					
D. Phaneuf D	0	0	0	4	0	1	0	0	2	0	1	35	24:13	2:33	3:29	18:11	0	0	00.0					
T. Pyatt C	0	0	0	0	1	1	1	2	1	1	1	27	15:31	0:16	2:42	12:33	0	1	00.0					
B. Ryan RW	0	0	-1	1	0	3	1	2	1	0	0	21	12:11	0:57	0:00	11:14	0	0	00.0					
Z. Smith C	0	0	-1	1	2	0	1	2	3	1	1	23	16:26	0:31	2:05	13:50	3	5	37.5					
M. Stone RW	1	4	4	5	0	1	0	0	2	2	0	26	18:38	3:12	0:00	15:26	0	0	00.0					
K. Turris C	0	3	2	3	0	2	1	2	0	0	0	31	21:56	3:18	0:31	18:07	15	5	75.0					
C. Wideman D	1	0	1	3	1	0	0	0	0	0	0	19	12:35	0:56	0:00	11:39	0	0	00.0					
T. Wingels C	0	0	0	0	0	0	0	0	0	0	0	16	9:24	0:00	2:04	7:20	1	0	100.0					
Ottawa Senators Scratches																								
Player	Detail																							
C. Lazar	Scratched																							
F. Claesson	Scratched																							
Toronto Maple Leafs															Time On Ice					Faceoffs				
Player	G	A	+/-	SOG	MS	BS	PN	PIM	HT	TK	GV	SHF	TOT	PP	SH	EV	FW	FL	%					
T. Bozak C	0	0	-3	3	0	0	0	0	0	2	0	23	16:41	3:04	0:00	13:37	6	9	40.0					
C. Brown RW	0	0	-3	1	1	2	0	0	3	1	0	24	18:00	3:23	0:52	13:45	0	0	00.0					
C. Carrick D	0	0	-1	0	0	0	0	0	0	0	0	24	18:01	0:30	0:00	17:31	0	0	00.0					
J. Gardiner D	0	0	-1	1	1	1	1	2	0	0	2	28	21:49	3:31	0:00	18:18	0	0	00.0					
M. Hunwick D	0	0	0	0	0	2	0	0	0	0	0	22	15:11	0:00	2:34	12:37	0	0	00.0					
Z. Hyman C	0	0	0	2	2	0	1	2	0	1	0	24	16:11	0:14	1:33	14:24	0	1	00.0					
N. Kadri C	1	0	-1	5	0	0	1	2	0	1	1	27	16:13	3:20	0:00	12:53	9	5	64.3					
L. Komarov C	0	1	0	3	2	0	0	0	3	0	0	27	18:24	3:41	2:38	12:05	1	3	25.0					
J. Leivo LW	0	1	0	1	1	1	0	0	1	3	0	24	14:38	3:04	0:00	11:34	0	0	00.0					
M. Martin LW	0	0	-1	0	0	0	1	2	2	1	0	12	7:56	0:00	0:11	7:45	0	1	00.0					
A. Matthews C	0	2	0	4	2	0	0	0	0	1	2	27	18:54	3:57	0:00	14:57	11	11	50.0					
W. Nylander RW	1	1	0	2	3	0	0	0	0	0	0	26	18:54	3:53	0:00	15:01	0	0	00.0					
R. Polak D	0	0	0	2	0	0	1	2	3	0	1	23	14:42	0:00	2:34	12:08	0	0	00.0					
M. Rielly D	1	0	-2	5	2	2	0	0	2	0	0	35	21:29	1:12	1:51	18:26	0	0	00.0					
B. Smith RW	0	0	-1	0	0	0	0	0	1	0	0	15	9:01	0:00	0:51	8:10	4	6	40.0					
N. Soshnikov RW	0	0	0	2	0	0	0	0	1	1	0	14	9:14	0:00	0:56	8:18	0	0	00.0					
J. van Riemsdyk LW	0	0	-2	3	0	1	0	0	2	0	0	23	14:52	2:46	0:00	12:06	0	0	00.0					
N. Zaitsev D	0	0	-2	2	2	2	0	0	5	1	0	37	22:45	2:14	1:51	18:40	0	0	00.0					
Toronto Maple Leafs Scratches																								
Player	Detail																							
A. Marchenko	Scratched																							
M. Marner	Upper Body																							
M. Marincin	Scratched																							

Table 6. Advanced Boxscore (part 2), Ottawa Senators @ Toronto Maple Leafs, February 18, 2017 [7].



Goaltending Summary						
 <b>Ottawa Senators Goaltending</b>				 <b>Toronto Maple Leafs Goaltending</b>		
Player	SA	GA	Saves	SV%	TOI	PIM
C. Anderson	37	3	34	.919	60:00	0
Player	SA	GA	Saves	SV%	TOI	PIM
F. Andersen	40	4	36	.900	58:51	0
Shots On Goal						
Team	1st	2nd	3rd	T		
Ottawa	14	16	12	42		
Toronto	15	10	12	37		
Power Play Summary						
Team	PPG / PPO					
Ottawa	1 of 4					
Toronto	1 of 4					

Table 7. Advanced Boxscore (part 3), Ottawa Senators @ Toronto Maple Leafs, February 18, 2017 [7].

## Macbeth's Plot [9]

### Act I

The play opens amid thunder and lightning, and the Three Witches decide that their next meeting will be with Macbeth. In the following scene, a wounded sergeant reports to King Duncan of Scotland that his generals Macbeth, who is the Thane of Glamis, and Banquo have just defeated the allied forces of Norway and Ireland, who were led by the traitorous Macdonwald, and the Thane of Cawdor. Macbeth, the King's kinsman, is praised for his bravery and fighting prowess.

In the following scene, Macbeth and Banquo discuss the weather and their victory. As they wander onto a heath, the Three Witches enter and greet them with prophecies. Though Banquo challenges them first, they address Macbeth, hailing him as "Thane of Glamis," "Thane of Cawdor," and that he will "be King hereafter." Macbeth appears to be stunned to silence. When Banquo asks of his own fortunes, the witches respond paradoxically, saying that he will be less than Macbeth, yet happier, less successful, yet more. He will father a line of kings, though he himself will not be one. While the two men wonder at these pronouncements, the witches vanish, and another thane, Ross, arrives and informs Macbeth of his newly bestowed title: Thane of Cawdor. The first prophecy is thus fulfilled, and Macbeth, previously sceptical, immediately begins to harbour ambitions of becoming king.

King Duncan welcomes and praises Macbeth and Banquo, and declares that he will spend the night at Macbeth's castle at Inverness; he also names his son Malcolm as his heir. Macbeth sends a message ahead to his wife, Lady Macbeth, telling her about the witches' prophecies. Lady Macbeth suffers none of her husband's uncertainty and wishes him to murder Duncan in order to obtain kingship. When Macbeth arrives at Inverness, she overrides all of her husband's objections by challenging his manhood and successfully persuades him to kill the king that very night. He and Lady Macbeth plan to get Duncan's two chamberlains drunk so that they will black out; the next morning they will blame the chamberlains for the murder. They will be defenceless as they will remember nothing.

### Act II

While Duncan is asleep, Macbeth stabs him, despite his doubts and a number of supernatural portents, including a hallucination of a bloody dagger. He is so shaken that Lady Macbeth has to take charge. In accordance with her plan, she frames Duncan's sleeping servants for the murder by placing bloody daggers on them. Early the next morning, Lennox, a Scottish nobleman, and Macduff, the loyal Thane of Fife, arrive. A porter opens the gate and Macbeth leads them to the king's chamber, where Macduff discovers Duncan's body. Macbeth murders the guards to prevent them from professing their innocence, but claims he did so in a fit of anger over their misdeeds. Duncan's sons Malcolm and Donalbain flee to England and Ireland, respectively, fearing that whoever killed Duncan desires their demise as well. The rightful heirs' flight makes them suspects and Macbeth assumes the throne as the new King of Scotland as a kinsman of the dead king. Banquo reveals this to the audience, and while sceptical of the new King Macbeth, he remembers the witches' prophecy about how his own descendants would inherit the throne; this makes him suspicious of Macbeth.

### Act III

Despite his success, Macbeth, also aware of this part of the prophecy, remains uneasy. Macbeth invites Banquo to a royal banquet, where he discovers that Banquo and his young son, Fleance, will be riding out that night. Fearing Banquo's suspicions, Macbeth arranges to have him murdered, by hiring two men to kill them, later sending a Third Murderer. The assassins succeed in killing Banquo, but Fleance escapes. Macbeth becomes furious: he fears that his power remains insecure as long as an heir of Banquo remains alive.

At a banquet, Macbeth invites his lords and Lady Macbeth to a night of drinking and merriment. Banquo's ghost enters and sits in Macbeth's place. Macbeth raves fearfully, startling his guests, as the ghost is only visible to him. The others panic at the sight of Macbeth raging at an empty chair, until a desperate Lady Macbeth tells them that her husband is merely afflicted with a familiar and harmless malady. The ghost departs and returns once more, causing the same riotous anger and fear in Macbeth. This time, Lady Macbeth tells the lords to leave, and they do so.

### Act IV

Macbeth, disturbed, visits the three witches once more and asks them to reveal the truth of their prophecies to him. To answer his questions, they summon horrible apparitions, each of which offers predictions and further prophecies to put Macbeth's fears at rest. First, they conjure an armoured head, which tells him to beware of Macduff (IV.i.72). Second, a bloody child tells him that no one born of a woman will be able to harm him. Thirdly, a crowned child holding a tree states that Macbeth will be safe until Great Birnam Wood comes to Dunsinane Hill. Macbeth is relieved and feels secure because he knows that all men are born of women and forests cannot move. Macbeth also asks whether Banquo's sons will ever reign in Scotland: the witches conjure a procession of eight crowned kings, all similar in appearance to Banquo, and the last carrying a mirror that reflects even more kings. Macbeth realises that these are all Banquo's descendants having acquired kingship in numerous countries. After the witches perform a mad dance and leave, Lennox enters and tells Macbeth that Macduff has fled to England. Macbeth orders Macduff's castle be seized, and, most cruelly, sends murderers to slaughter Macduff, as well as Macduff's wife and children. Although Macduff is no longer in the castle, everyone in Macduff's castle is put to death, including Lady Macduff and their young son.

### Act V

Meanwhile, Lady Macbeth becomes racked with guilt from the crimes she and her husband have committed. At night, in the king's palace at Dunsinane, a doctor and a gentlewoman discuss Lady Macbeth's strange habit of sleepwalking. Suddenly, Lady Macbeth enters in a trance with a candle in her hand. Bemoaning the murders of Duncan, Lady Macduff, and Banquo, she tries to wash off imaginary bloodstains from her hands, all the while speaking of the terrible things she knows she pressed her husband to do. She leaves, and the doctor and gentlewoman marvel at her descent into madness. Her belief that nothing can wash away the blood on her hands is an ironic reversal of her earlier claim to Macbeth that "[a] little water clears us of this deed" (II.ii.66).

In England, Macduff is informed by Ross that his "castle is surprised; wife and babes / Savagely slaughter'd" (IV.iii.204–05). When this news of his family's execution reaches him, Macduff is stricken with grief and vows revenge. Prince Malcolm, Duncan's son, has succeeded in raising an army in England, and Macduff joins him as he rides to Scotland to challenge Macbeth's forces. The invasion has the support of the Scottish nobles, who are appalled and frightened by Macbeth's tyrannical and murderous behaviour. Malcolm leads an army, along with Macduff and Englishmen Siward (the Elder), the Earl of Northumberland, against Dunsinane Castle. While encamped in Birnam Wood, the soldiers are ordered to cut down and carry tree limbs to camouflage their numbers.

Before Macbeth's opponents arrive, he receives news that Lady Macbeth has killed herself, causing him to sink into a deep and pessimistic despair and deliver his "Tomorrow, and tomorrow, and tomorrow" soliloquy (V.v.17–28). Though he reflects on the brevity and meaninglessness of life, he nevertheless awaits the English and fortifies Dunsinane. He is certain that the witches' prophecies guarantee his invincibility, but is struck with fear when he learns that the English army is advancing on Dunsinane shielded with boughs cut from Birnam Wood, in apparent fulfillment of one of the prophecies.

A battle culminates in Macduff's confrontation with Macbeth, who kills Young Siward in combat. The English forces overwhelm his army and castle. Macbeth boasts that he has no reason to fear Macduff, for he cannot be killed by any man born of woman. Macduff declares that he was "from his mother's womb / Untimely ripp'd" (V.8.15–16), (i.e., born by Caesarean section) and is not "of woman born" (an example of a literary quibble), fulfilling the second prophecy. Macbeth realises too late that he has misinterpreted the witches' words. Though he realises that he is doomed, he continues to fight. Macduff kills and beheads him, thus fulfilling the remaining prophecy.

Macduff carries Macbeth's head onstage and Malcolm discusses how order has been restored. His last reference to Lady Macbeth, however, reveals "'tis thought, by self and violent hands / Took off her life" (V.ix.71–72), but the method of her suicide is undisclosed. Malcolm, now the King of Scotland, declares his benevolent intentions for the country and invites all to see him crowned at Scone.



## Spectral Analysis and Feature Selection (Python)

### Librairies

This notebook use the following librairies:

- `scipy`: ecosystem of open-source software for mathematics, science, and engineering
- `matplotlib`: plotting library (in scipy by default)
- `numpy`: general package for scientific computing (in scipy by default)
- `networkx`: package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

Other than `networkx`, they should all be installed already if you are using Anaconda. `Networkx` is only used to produce a few graphs and is not at all needed for the main parts of spectral feature selection.

```
[ ] from matplotlib import ticker, cm
import numpy as np
import matplotlib.pyplot as plt
from scipy.spatial.distance import pdist, squareform, cdist
import networkx as nx
from numpy.linalg import eigh, norm
import random # for replicability

plt.style.use('default')
```

### Simple graph example

The first part of this notebook is about producing a scatter plot, the similarity matrix, and the correspond graph for a toy example with labels:

$$X = \begin{bmatrix} 1 & 0 & 3 & 6 & 7 & 6 \\ 1 & 2 & 2 & 4 & 4 & 8 \end{bmatrix}^T \quad Y = [0 \ 0 \ 0 \ 1 \ 1 \ 1]^T$$

```
[ ] # Data
X = np.array([[1,1,1],[0,2,1],[3,2,1],[6,4,1],[7,5,1],[6,8,1]])
Y = np.array([0,0,0,1,1,1])
# Plot
plt.scatter(X[:,0],X[:,1], c=Y)
# Save plot to file
plt.savefig('plot_exemple1.png')
```

Next, we compute the similarity (adjacency matrix). We use the Gaussian RBF kernel with  $\sigma = 0.5$ , and `pdist()` from `scipy.spatial.distance` which computes the pairwise distance of each row from an input data matrix. `pdist()` return a vector, which allows a speed boost for the following computation, but it needs to be converted back to a matrix for the next steps. This is done via `squareform()`, also from `scipy.spatial.distance`.

```
[ ] # Adjacency matrix
A = squareform(np.exp(-pdist(X))/((0.5)**2))
plt.matshow(A)
plt.show()
```

Next, we can use the `networkx` library to display the resulting graph.

```
[ ] # Create a graph object from a adjacency matrix
G = nx.from_numpy_matrix(A)
# Add the weight (similarity) attribute to the graph
pos = {i : [X[i,0],X[i,1]] for i in range(6)}
labels = nx.get_edge_attributes(G, 'weight')
# Draw it
nx.draw(G)
nx.draw_networkx_edge_labels(G,pos)
plt.show()
```

This graph is problematic: we have no information about the similarity between the various observations, and the point are drawn at random position. The following code overcomes these issues.

```
[ ] # Create a graph object from a adjacency matrix
G = nx.from_numpy_matrix(A)
# Add the weight (similarity) attribute to the graph
pos = {i : [X[i,0],X[i,1]] for i in range(6)}
labels = nx.get_edge_attributes(G, 'weight')

# round the similarity (for display)
labels = {k: round(v,2) for k, v in labels.items()}
# Create the edge list and labels. Remove null edges
edge_list = [k for k, v in labels.items() if v != 0]
labels = {k: v for k, v in labels.items() if v != 0}
# Draw it using the edge and labels list, at the right position
nx.draw_networkx_nodes(G,pos)
nx.draw_networkx_edges(G,pos, edgelist=edge_list)
nx.draw_networkx_edge_labels(G,pos, edge_labels=labels)
plt.axis('off')
# Save to file
plt.savefig('graph_exemple.png')
```

Next, we can compute the degree and Laplacian matrices. For the degree matrix we can use the method `sum()` from `numpy` with the argument `axis=1` to sum over the columns and `diag()` to convert the result into a diagonal matrix.

```
[ ] rowsum = A.sum(axis=1)
D = np.diag(rowsum)
```

The laplacian matrix is simply  $L = D - A$ .

```
[ ] L = D - A
```

```
[ ] plt.matshow(L)
plt.show()
```

## 2D Gaussian Mixture Example

The next example consist of 3 clusters all sample from a 2D Gaussian distribution with means  $\mu_1 = (0, 5)$ ,  $\mu_2 = (5, 0)$ ,  $\mu_3 = (5, 5)$  and variance  $\Sigma_{1,2,3} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ . This example is taken from Zhao and Liu's *Spectral Feature Selection for Data Mining*.

```
[ ] # Sample the clusters

random.seed(0) # for replicability

mean1 = [0,5]
cov1 = [[1,0],[0,1]]
X1 = np.random.multivariate_normal(mean1,cov1,30)

mean2 = [5,0]
cov2 = [[1,0],[0,1]]
X2 = np.random.multivariate_normal(mean2,cov2,30)

mean3 = [5,5]
cov3 = [[1,0],[0,1]]
X3 = np.random.multivariate_normal(mean3,cov3,30)

# Plot and save fig
plt.scatter(X1[:,0],X1[:,1])
plt.scatter(X2[:,0],X2[:,1])
plt.scatter(X3[:,0],X3[:,1])
plt.savefig('example1.png')

# Grouping and Labeling
X = np.concatenate((X1,X2,X3), axis=0)
Y = np.array([0] * 30 + [1] * 30 + [2] * 30).reshape((90,1))

# Shuffling the data
df = np.concatenate((X,Y), axis = 1)
np.random.shuffle(df)
X,Y = df[:, :2],df[:,2]
```

We can then compute the similarity (using Gaussian RBF with  $\sigma = 1$ ), adjacency, degree and Laplacian matrix. Using `eigh()` from `numpy` we can find the eigenvalue and eigenvectors of  $L$ . This function returns a vector of all the eigenvalues of  $L$  and a matrix of all its eigenvectors as columns. According to the convention, the eigenspace is sorted so that the eigenvalues satisfy  $0 = \lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \dots$

```
[ ] # Compute the matrices
A = squareform(np.exp(-1 * pdist(X, 'sqeuclidean')))
rowsum = A.sum(axis=1)
D = np.diag(rowsum)
L = D - A

# Find eigenspace of L (vs: eigenvalue vector, es: eigenvector matrix)
vs,es = eigh(L)

# Sort the eigenvalue
arg_sort = vs.argsort()
vs = vs[arg_sort]
es = es[:,arg_sort]
```

```
[ ] def gen_z(lam = 1):
    """ Compute the meshgrid for z
    """
    for i in range(len(x)):
        for j in range(len(y)):
            dist = cdist([[x_[i,j],y_[i,j]],X)[0]
            z[i,j] = np.average(es[:,lam], weights= 1/dist)

    # Setup the meshgrid (you can change the dimension)
    x = np.arange(-3,8.5,0.05)
    y = np.arange(-3,8.5,0.05)
    x_, y_ = np.meshgrid(x,y, indexing='ij')
    z = 0*x_
    l = 1 # Index of the eigenvalue to be plotted (0 is the first, etc)

    # Compute the meshgrid with the gen_z function
    gen_z(l)

    # Setup and plot
    fig,ax = plt.subplots()
    cs = ax.contourf(x_, y_, z)
    ax.contour(cs,colors='k')
    ax.grid(c='k', ls='-', alpha=0.3)
    fig.colorbar(cs)
    ax.scatter(X[:,0],X[:,1], c=Y)
    ax.set_title('\lambda{0}'.format(l+1))
    plt.show()
```

The next code block does pretty much the same as above, but for different eigenvectors/eigenvalues, and arranges the plots in a grid.

```
[ ] x = np.arange(-2.5,8.5,0.5)
    y = np.arange(-3.5,9,0.5)
    x_, y_ = np.meshgrid(x,y, indexing='ij')
    z = 0*x_

    # List of index of the eigvalue to be plotted
    ls = [0,1,2,3]
    # Plot grid length and width. You can change these number be they need to match
    # up with the lenght of ls
    grid_lenght = 2
    grid_width = 2

    # You can change the figsize if the aspect ratio is not good
    plt.figure(figsize=(12,8))
    for i in range(len(ls)):
        # You can change
        ax = plt.subplot(grid_lenght,grid_width,i+1)
        z = 0*x_
        gen_z(ls[i])
        cs = ax.contourf(x_, y_, z)
        ax.contour(cs,colors='k')
        ax.grid(c='k', ls='-', alpha=0.3)
        # Add or remove the next line to toggle the color bar
        #fig.colorbar(cs,ax = ax, orientation='horizontal')
        ax.scatter(X[:,0],X[:,1], c=Y)
        ax.set_title('\lambda{0}'.format(ls[i]+1))
    plt.savefig('spectrum_L.png')
```

To create our next plot, we need to unshuffle our data.

```
[ ] # Recompute everything without the shuffling and sorting part
X = np.concatenate((X1,X2,X3), axis=0)
Y = np.array([0] * 30 + [1] * 30 + [2] * 30).reshape((90,1))

A = squareform(np.exp(-1 * pdist(X, 'sqeuclidean')))
rowsum = A.sum(axis=1)
D = np.diag(rowsum)
L = D - A
vs,es = eigh(L)
```

The next plot shows the "smoothness" of the eigenvector over the cluster. For that, we just do a normal plot of each eigenvector component for different eigenvalues. Color is also added to emphasise the cluster limits.

```
[ ] # List of index of selected eigenvalue. You can change this.
ls = [0,1,2,19]

# Plot a line for each index in ls
for l in ls:
    plt.plot(es[:,l], label = '\lambda{}'.format(l+1))

# Color the cluster limits
plt.axvspan(0,30, alpha = 0.5)
plt.text(15,-0.4, 'Cluster 1', horizontalalignment='center', fontsize=12)
plt.axvspan(30,60, alpha = 0.5, facecolor='g')
plt.text(45,-0.4, 'Cluster 2', horizontalalignment='center', fontsize=12)
plt.axvspan(60,90, alpha = 0.5, facecolor='orange')
plt.text(75,-0.4, 'Cluster 3', horizontalalignment='center', fontsize=12)

plt.legend()
plt.ylabel('Value')
plt.xlabel('Component k of the eigenvector')
plt.show()
```

The next code block produces the same graph as above, but using the feature (columns of the data matrix  $X$ ) rather than  $L$ 's eigenvalues.

```
[ ] # Features index list
f = [0,1]

for i in f:
    plt.plot(X[:,i], label = 'F{}'.format(i+1))

plt.axvspan(0,30, alpha = 0.5)
plt.text(15,2, 'Cluster 1', horizontalalignment='center', fontsize=12)
plt.axvspan(30,60, alpha = 0.5, facecolor='g')
plt.text(45,2, 'Cluster 2', horizontalalignment='center', fontsize=12)
plt.axvspan(60,90, alpha = 0.5, facecolor='orange')
plt.text(75,2, 'Cluster 3', horizontalalignment='center', fontsize=12)

plt.legend()
plt.ylabel('Value')
plt.xlabel('Component k of feature')
plt.show()
```



In the article, we add a useless feature, one that is sample from a 1D uniform distribution. This is done as follows. Next, we add this feature to our previous plot to show its cluster invariance.

```
[ ] # Add a random feature
    U1 = np.random.uniform(size=(90,1))
    X = np.concatenate((X,U1), axis=1)

[ ] f = [0,1,2]

    for i in f:
        plt.plot(X[:,i], label = 'F{0}'.format(i+1))

    plt.axvspan(0,30, alpha = 0.5)
    plt.text(15,2, 'Cluster 1', horizontalalignment='center', fontsize=12)
    plt.axvspan(30,60, alpha = 0.5, facecolor='g')
    plt.text(45,2, 'Cluster 2', horizontalalignment='center', fontsize=12)
    plt.axvspan(60,90, alpha = 0.5, facecolor='orange')
    plt.text(75,2, 'Cluster 3', horizontalalignment='center', fontsize=12)

    plt.legend()
    plt.ylabel('Value')
    plt.xlabel('Component k of feature')
    plt.show()
```

## Feature Selection

We then proceed to compute our first metric. For matrix multiplication, we can use the `dot()` method from `numpy`.

```
[ ] for i in [0,1,2]:
    f = X[:,i]
    print("F{0}".format(i+1), "->", f.T.dot(f.dot(L)))
```

For the next metric, we need to compute the normalized Laplacian. To do so, we also need to obtain  $D^{-1/2}$ , which can be done with a simple function, vectorized using `numpy`'s `vectorize()` to improve its scalability. The normalized Laplacian is given by  $\mathcal{L} = D^{-1/2}LD^{-1/2}$  and its eigenspace is found in just the same way as  $L$ 's.

```
[ ] def isqrt(x):
    if x == 0:
        return 0
    else:
        return x**(-0.5)

    v_isqrt = np.vectorize(isqrt)

[ ] D_is = v_isqrt(D)
    L_norm = D_is.dot(L).dot(D_is)
    vs,es = eigh(L_norm)

    # Sort the eigenvalue
    arg_sort = vs.argsort()
    vs = vs[arg_sort]
    es = es[:,arg_sort]
```

Since  $\varphi_1(f_i) = \hat{f}_i^T \mathcal{L} \hat{f}_i$  with  $\tilde{f}_i = (D^{1/2} f_i)$  and  $\hat{f}_i = \tilde{f}_i / \|\tilde{f}_i\|$ , we define a new function `score_1` as the implementation of  $\varphi_1(f_i)$ .

```
[ ] # D^(1/2)
D_sq = np.sqrt(D)

def score_1(i):
    f_tilde = D_sq.dot(X[:,i])
    f_hat = f_tilde / norm(f_tilde)
    return f_hat.dot(L_norm).dot(f_hat)
```

Similarly, we have :

$$\varphi_2(f_i) = \frac{\hat{f}_i^T \mathcal{L} \hat{f}_i}{1 - \left( \hat{f}_i^T \xi_1 \right)} \quad \text{and} \quad \varphi_3(f_i, k) = \sum_{j=2}^k (2 - \lambda_j) \alpha_j^2$$

Their corresponding implementations are `score_2` and `score_3`.

```
[ ] def score_2(i):
    f_tilde = D_sq.dot(X[:,i])
    f_hat = f_tilde / norm(f_tilde)
    phi_1 = f_hat.dot(L_norm).dot(f_hat)
    return phi_1 / (1 - (f_hat.dot(es[:,0])**2))

def score_3(i,k):
    f_tilde = D_sq.dot(X[:,i])
    f_hat = f_tilde / norm(f_tilde)
    alpha = f_hat.dot(es)
    temp = (2 - vs[1:k]) * (alpha[1:k])**2
    return np.sum(temp)
```

Next is a code block that displays the scores for each feature in the dataset  $X$ . The library `tabulate` is used to prettify the output.

```
[1] from tabulate import tabulate
from IPython.display import HTML, display

n_feature = X.shape[1]
results = {'phi_1':[], 'phi_2':[], 'phi_3': []}
k = 10

for i in range(n_feature):
    results['phi_1'].append(score_1(i))
    results['phi_2'].append(score_2(i))
    results['phi_3'].append(score_3(i,k))

# tablefmt can be changed to 'latex' for easy importing into a latex doc
print(tabulate(results,
               headers="keys",
               showindex=True,
               tablefmt="simple",
               numalign="left"))
```

## Effect of noise and Kernel function

The next section shows the effect of noise over on the eigenvalue of the normalized Laplacian. We first add random noise sample from a Gaussian to the data matrix  $X$ .

```
[ ] noise = np.random.normal(0, 1.1, 90*3).reshape(90,3)
    X_noise = X + noise
```

If we look at the scatter plot of  $X\_noise$ , we can see that it become harder to identify the clusters boundaries.

```
[ ] plt.scatter(X_noise[:,0], X_noise[:,1])
    plt.show()
```

Next, we will plot the components of the eigenvalues of three normalized Laplacian: one from the original data, one from the noise effected data and one from the noise effected data with a 3rd order polynomial regularization function applied.

```
[ ] def find_eig(X, k = lambda x:x):
    A = squareform(np.exp(-1 * pdist(X, 'sqeuclidean')))
    rowsum = A.sum(axis=1)
    D = np.diag(rowsum)
    L = D - A
    D_is = v_isqrt(D)
    L_norm = k(D_is.dot(L).dot(D_is))
    vs,es = eigh(L_norm)
    arg_sort = vs.argsort()
    vs = vs[arg_sort]
    es = es[arg_sort]
    return vs,es, L_norm

vs,es,L = find_eig(X)
vs_n,es_n,L_noise = find_eig(X_noise)
vs_k,es_k, L_k = find_eig(X_noise, k = lambda x:x**3)

plt.plot(vs, '.', label = 'Real')
plt.plot(vs_n, 'x', label = 'Noise')
plt.plot(vs_k, "+", label = '3 order poly')
plt.title('\lambda')
plt.legend()
plt.show()
```

We can do the same to compare different kernel. Here we have used :

- Regularized Laplacian:  $\gamma(\lambda) = 1 + (0.9)^2 \lambda$
- High-order Polynomial:  $\gamma(\lambda) = \lambda^3$
- Diffusion Process:  $\gamma(\lambda) = \exp((0.3)^2/2\lambda)$
- p-Step Random Walk:  $\gamma(\lambda) = (2 - \lambda)^{-1}$
- Inverse cosine:  $\gamma(\lambda) = \cos(\lambda * \pi/4)^{-1}$

```
[ ] vs_n,es_n,_ = find_eig(X_noise)
vs_reg, es_reg,_ = find_eig(X_noise, k = lambda x:1 + (0.9)**2 * x)
vs_k3,es_k3,_ = find_eig(X_noise, k = lambda x:x**3)
vs_diff, es_diff,_ = find_eig(X_noise, k = lambda x: np.exp((0.3)**2/(2*x)))
vs_lstep, es_lstep,_ = find_eig(X_noise, k = lambda x: (2-x)**(-1))
vs_cos, es_cos, _ = find_eig(X_noise, k = lambda x: np.cos(x * (3.1416/4))**(-1))

plt.plot(vs_n, '.', label = 'Noise')
plt.plot(vs_k, '.', label = '3 order poly')
plt.plot(vs_reg, '.', label = 'Regularized (0.9)')
plt.plot(vs_diff, '.', label = 'Diffusion (0.5)')
plt.plot(vs_lstep, '.', label = '1-Step (2)')
plt.plot(vs_cos, '.', label = 'Inverse Cos')

plt.ylim(0,3)
plt.title('\lambda')
plt.legend()
plt.show()
```

Graph representation of the data (was not actually used in the final document)

```
[ ] G = nx.from_numpy_matrix(A)
pos = {i : [X[i,0],X[i,1]] for i in range(90)}
labels = nx.get_edge_attributes(G,'weight')

labels = {k: round(v,2) for k, v in labels.items()} # round
edge_list = [k for k, v in labels.items() if v > 0.1]
labels = {k: v for k, v in labels.items() if v > 0.1} # remove null edges
nx.draw_networkx_nodes(G,pos, node_size=50)
nx.draw_networkx_edges(G,pos, edgelist=edge_list)
#nx.draw_networkx_edge_labels(G,pos, edge_labels=labels)
plt.axis('off')
plt.show()
```

## Uniform Manifold Approximation and Projection (Python)

```
[1] import warnings
     warnings.filterwarnings('ignore')
```

adapted from [https://umap-learn.readthedocs.io/en/latest/auto\\_examples/plot\\_algorithm\\_comparison.html](https://umap-learn.readthedocs.io/en/latest/auto_examples/plot_algorithm_comparison.html)

```
[2] %matplotlib
     import numpy as np
     import matplotlib.pyplot as plt

     from sklearn import datasets, decomposition, manifold, preprocessing
     from colorsys import hsv_to_rgb

     import umap
```

```
↳ Using matplotlib backend: agg
```

We'll plot five different datasets reduced to the two dimensional plane. The points will be coloured to get a sense of which points got sent where (otherwise we would just know what the shape of the entire reduced dataset looks like).

- 4 different 10-dimensional Gaussian distributions

```
[3] blobs, blob_labels = datasets.make_blobs(
     n_samples=500, n_features=10, centers=4
     )
```

- Digit classification dataset

```
[4] digits = datasets.load_digits(n_class=10)
```

- Wine characteristics, essentially a one dimensional set

```
[5] wine = datasets.load_wine()
```

- A 2-d rectangle rolled up in 3-space. Colours indicate position along the unrolled rectangle.

```
[6] swissroll, swissroll_labels = datasets.make_swiss_roll(
     n_samples=1000, noise=0.1
     )
```

- Points on the 2D surface of a 3D sphere, which is not homeomorphic to  $\mathbb{R}^2$ . Even with the north pole removed, the stereographic projection would map points close to the north pole arbitrarily far from the origin. Colour hue indicates angle around equator and darkness indicates distance from south pole.



```
[7] sphere = np.random.normal(size=(600, 3))
# scale points to have same distance from origin
sphere = preprocessing.normalize(sphere)
# compute colours in hue-saturation-value format
sphere_hsv = np.array([
    (
        (np.arctan2(c[1], c[0]) + np.pi) / (2 * np.pi),
        np.abs(c[2]),
        min((c[2] + 1.1), 1.0),
    )
    for c in sphere
])
# convert colours to red-green-blue format
sphere_colors = np.array([hsv_to_rgb(*c) for c in sphere_hsv])
```

Next we set parameters for the reducer algorithms. For UMAP, we set `min_dist` to 0.3 which will spread out the points to a noticeable degree. We also set the number  $k$  of nearest neighbours to 30. Typically in practice we would set these parameters on a dataset-by-dataset basis.

```
[8] reducers = [
    (manifold.TSNE, {"perplexity": 50}),
    (manifold.Isomap, {"n_neighbors": 30}),
    (manifold.MDS, {}),
    (decomposition.PCA, {}),
    (umap.UMAP, {"n_neighbors": 30, "min_dist": 0.3}),
]

test_data = [
    (blobs, blob_labels),
    (digits.data, digits.target),
    (wine.data, wine.target),
    (swissroll, swissroll_labels),
    (sphere, sphere_colors),
]

dataset_names = ["Blobs", "Digits", "Wine", "Swiss Roll", "Sphere"]
```

We compute 2D reductions for every reducer-dataset pair.

```
[9] %%time
reductions_and_labels = [
    (
        reducer(n_components=2, **args).fit_transform(data),
        labels
    )
    for data, labels in test_data
    for reducer, args in reducers
]
```

```
↳ CPU times: user 3min 28s, sys: 1min 39s, total: 5min 8s
Wall time: 2min 59s
```

Initialize matplotlib figure...

```
[10] n_rows = len(test_data)
      n_cols = len(reducers)

      fig = plt.figure(figsize=(16, 16))
      fig.subplots_adjust(
          left=.02, right=.98, bottom=.001, top=.96, wspace=.05, hspace=.02
      )
```

Show scatterplot grid...

```
[11] ax_index = 1
      ax_list = []

      for reduction, labels in reductions_and_labels:
          ax = fig.add_subplot(n_rows, n_cols, ax_index)
          if isinstance(labels[0], tuple):
              # if labels are colours, use them
              ax.scatter(*reduction.T, s=10, c=labels, alpha=0.5)
          else:
              # otherwise, use "spectral" map from labels to colours
              ax.scatter(
                  *reduction.T, s=10, c=labels, cmap="Spectral", alpha=0.5
              )
          ax_list.append(ax)
          ax_index += 1

      plt.setp(ax_list, xticks=[], yticks=[])

      for i in np.arange(n_rows) * n_cols:
          ax_list[i].set_ylabel(dataset_names[i // n_cols], size=16)
      for i in range(n_cols):
          ax_list[i].set_xlabel(repr(reducers[i][0]()).split("(")[0], size=16)
          ax_list[i].xaxis.set_label_position("top")

[12] fig.show()
```

Notice that isomap removes exactly the wrong dimension in the swiss roll. t-SNE (and MDS to some extent) reduces the wine data down to one dimension (its true dimensionality) even though it was only asked for a reduction to two dimensions. UMAP manages to give the most sensible output for the swiss roll.