A MACHINE LEARNING HANDBOOK

PUBLISHER OF THIS BOOK

JEAN-PAUL EBEJER, DYLAN SEYCHELL, NEIL MICALLEF

First printing, December 2018

Contents

Introduction

This book explains popular Machine Learning terms. We focus to explain each term comprehensively, through the use of examples and diagrams. The description of each term is written by a student sitting in for ICS5110 APPLIED MACHINE LEARNING[1] at the University of Malta (class 2018/2019). This study-unit is part of the MSc. in AI offered by the Department of Artificial Intelligence, Faculty of ICT.

[1] https://www.um.edu.mt/courses/studyunit/ICS5110

On the Use of the `tufte-book` Document Class

The Tufte-LaTeX document classes define a style similar to the style Edward Tufte uses in his books and handouts. Tufte's style is known for its extensive use of sidenotes, tight integration of graphics with text, and well-set typography. This document aims to be at once a demonstration of the features of the Tufte-LaTeX document classes and a style guide to their use.

Page Layout

Headings

This style provides A- and B-heads (that is, `\section` and `\subsection`), demonstrated above.

If you need more than two levels of section headings, you'll have to define them yourself at the moment; there are no pre-defined styles for anything below a `\subsection`. As Bringhurst points out in *The Elements of Typographic Style*,[2] you should "use as many levels of headings as you need: no more, and no fewer."

The Tufte-LaTeX classes will emit an error if you try to use `\subsubsection` and smaller headings.

[2]

In his later books,[3] Tufte starts each section with a bit of vertical space, a non-indented paragraph, and sets the first few words of the sentence in small caps. To accomplish this using this style, use the `\newthought` command:

[3]

```
\newthought{In his later books}, Tufte starts...
```

Feature	*VDQI*	*EI*	*VE*	*BE*
Author				
Typeface	serif	serif	serif	sans serif
Style	italics	italics	italics	upright, caps
Size	24 pt	20 pt	20 pt	20 pt
Title				
Typeface	serif	serif	serif	sans serif
Style	upright	italics	upright	upright, caps
Size	36 pt	48 pt	48 pt	36 pt
Subtitle				
Typeface	–	–	serif	–
Style	–	–	upright	–
Size	–	–	20 pt	–
Edition				
Typeface	sans serif	–	–	–
Style	upright, caps	–	–	–
Size	14 pt	–	–	–
Publisher				
Typeface	serif	serif	serif	sans serif
Style	italics	italics	italics	upright, caps
Size	14 pt	14 pt	14 pt	14 pt

Sidenotes

One of the most prominent and distinctive features of this style is the extensive use of sidenotes. There is a wide margin to provide ample room for sidenotes and small figures. Any `\footnotes` will automatically be converted to sidenotes.[4] If you'd like to place ancillary information in the margin without the sidenote mark (the superscript number), you can use the `\marginnote` command.

The specification of the `\sidenote` command is:

`\sidenote[`⟨*number*⟩`][`⟨*offset*⟩`]{`*Sidenote text.*`}`

Both the ⟨*number*⟩ and ⟨*offset*⟩ arguments are optional. If you provide a ⟨*number*⟩ argument, then that number will be used as the sidenote number. It will change of the number of the current sidenote only and will not affect the numbering sequence of subsequent sidenotes.

Sometimes a sidenote may run over the top of other text or graphics in the margin space. If this happens, you can adjust the vertical position of the sidenote by providing a dimension in the ⟨*offset*⟩ argument. Some examples of valid dimensions are:

```
1.0in    2.54cm    254mm    6\baselineskip
```

If the dimension is positive it will push the sidenote down the page; if the dimension is negative, it will move the sidenote up the page.

While both the ⟨*number*⟩ and ⟨*offset*⟩ arguments are optional, they must be provided in order. To adjust the vertical position of the sidenote while leaving the sidenote number alone, use the following syntax:

`\sidenote[][`⟨*offset*⟩`]{`*Sidenote text.*`}`

The empty brackets tell the `\sidenote` command to use the default sidenote number.

[4] This is a sidenote that was entered using the `\footnote` command.

This is a margin note. Notice that there isn't a number preceding the note, and there is no number in the main text where this note was written.

If you *only* want to change the sidenote number, however, you may completely omit the ⟨*offset*⟩ argument:

`\sidenote[`⟨*number*⟩`]{`*Sidenote text.*`}`

The `\marginnote` command has a similar *offset* argument:

`\marginnote[`⟨*offset*⟩`]{`*Margin note text.*`}`

References

References are placed alongside their citations as sidenotes, as well. This can be accomplished using the normal `\cite` command.[5]

The complete list of references may also be printed automatically by using the `\bibliography` command. (See the end of this document for an example.) If you do not want to print a bibliography at the end of your document, use the `\nobibliography` command in its place.

To enter multiple citations at one location,[6] you can provide a list of keys separated by commas and the same optional vertical offset argument: `\cite{Tufte2006,Tufte1990}`.

`\cite[`⟨*offset*⟩`]{`*bibkey1,bibkey2,...*`}`

[5] The first paragraph of this document includes a citation.

[6] ; and

Figures and Tables

Images and graphics play an integral role in Tufte's work. In addition to the standard `figure` and `tabular` environments, this style provides special figure and table environments for full-width floats.

Full page–width figures and tables may be placed in `figure*` or `table*` environments. To place figures or tables in the margin, use the `marginfigure` or `margintable` environments as follows (see figure 1):

```
\begin{marginfigure}
\includegraphics{helix}
\caption{This is a margin figure.}
\label{fig:marginfig}
\end{marginfigure}
```

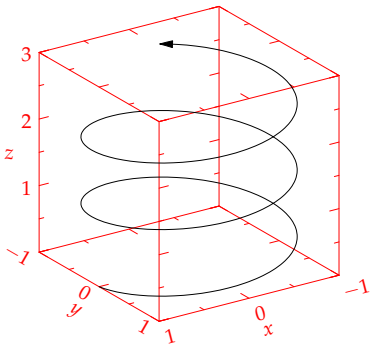

Figure 1: This is a margin figure. The helix is defined by $x = \cos(2\pi z)$, $y = \sin(2\pi z)$, and $z = [0, 2.7]$. The figure was drawn using Asymptote (`http://asymptote.sf.net/`).

The `marginfigure` and `margintable` environments accept an optional parameter ⟨*offset*⟩ that adjusts the vertical position of the figure or table. See the "Sidenotes" section above for examples. The specifications are:

```
\begin{marginfigure}[⟨offset⟩]
...
\end{marginfigure}

\begin{margintable}[⟨offset⟩]
...
\end{margintable}
```

Figure 2 is an example of the `figure*` environment and figure 3 is an example of the normal `figure` environment.

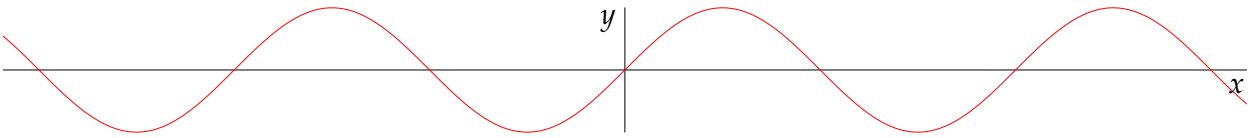

Figure 2: This graph shows $y = \sin x$ from about $x = [-10, 10]$. *Notice that this figure takes up the full page width.*

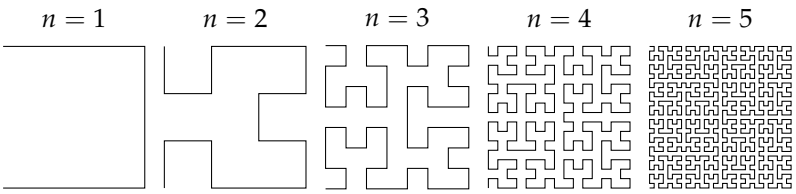

Figure 3: Hilbert curves of various degrees n. *Notice that this figure only takes up the main textblock width.*

As with sidenotes and marginnotes, a caption may sometimes require vertical adjustment. The `\caption` command now takes a second optional argument that enables you to do this by providing a dimension ⟨*offset*⟩. You may specify the caption in any one of the following forms:

```
\caption{long caption}
\caption[short caption]{long caption}
\caption[][⟨offset⟩]{long caption}
\caption[short caption][⟨offset⟩]{long caption}
```

A positive ⟨*offset*⟩ will push the caption down the page. The short caption, if provided, is what appears in the list of figures/tables, otherwise the "long" caption appears there. Note that although the arguments ⟨*short caption*⟩ and ⟨*offset*⟩ are both optional, they must be provided in order. Thus, to specify an ⟨*offset*⟩ without specifying a ⟨*short caption*⟩, you must include the first set of empty brackets `[]`, which tell `\caption` to use the default "long" caption. As an example, the caption to figure 3 above was given in the form

```
\caption[Hilbert curves...][6pt]{Hilbert curves...}
```

Table 1 shows table created with the `booktabs` package. Notice the lack of vertical rules—they serve only to clutter the table's data.

Margin	Length
Paper width	8 1/2 inches
Paper height	11 inches
Textblock width	6 1/2 inches
Textblock/sidenote gutter	3/8 inches
Sidenote width	2 inches

Table 1: Here are the dimensions of the various margins used in the Tufte-handout class.

Occasionally LaTeX will generate an error message:

```
Error:  Too many unprocessed floats
```

LaTeX tries to place floats in the best position on the page. Until it's finished composing the page, however, it won't know where those positions are. If you have a lot of floats on a page (including sidenotes,

margin notes, figures, tables, etc.), LaTeX may run out of "slots" to keep track of them and will generate the above error.

LaTeX initially allocates 18 slots for storing floats. To work around this limitation, the Tufte-LaTeX document classes provide a `\morefloats` command that will reserve more slots.

The first time `\morefloats` is called, it allocates an additional 34 slots. The second time `\morefloats` is called, it allocates another 26 slots.

The `\morefloats` command may only be used two times. Calling it a third time will generate an error message. (This is because we can't safely allocate many more floats or LaTeX will run out of memory.)

If, after using the `\morefloats` command twice, you continue to get the `Too many unprocessed floats` error, there are a couple things you can do.

The `\FloatBarrier` command will immediately process all the floats before typesetting more material. Since `\FloatBarrier` will start a new paragraph, you should place this command at the beginning or end of a paragraph.

The `\clearpage` command will also process the floats before continuing, but instead of starting a new paragraph, it will start a new page.

You can also try moving your floats around a bit: move a figure or table to the next page or reduce the number of sidenotes. (Each sidenote actually uses *two* slots.)

After the floats have placed, LaTeX will mark those slots as unused so they are available for the next page to be composed.

Captions

You may notice that the captions are sometimes misaligned. Due to the way LaTeX's float mechanism works, we can't know for sure where it decided to put a float. Therefore, the Tufte-LaTeX document classes provide commands to override the caption position.

Vertical alignment To override the vertical alignment, use the `\setfloatalignment` command inside the float environment. For example:

```
\begin{figure}[btp]
\includegraphics{sinewave}
\caption{This is an example of a sine wave.}
\label{fig:sinewave}
\setfloatalignment{b}% forces caption to be bottom-aligned
\end{figure}
```

The syntax of the `\setfloatalignment` command is:

```
\setfloatalignment{⟨pos⟩}
```

where ⟨*pos*⟩ can be either `b` for bottom-aligned captions, or `t` for top-aligned captions.

Horizontal alignment To override the horizontal alignment, use either the `\forceversofloat` or the `\forcerectofloat` command inside of the float environment. For example:

```
\begin{figure}[btp]
\includegraphics{sinewave}
\caption{This is an example of a sine wave.}
\label{fig:sinewave}
\forceversofloat% forces caption to be set to the left of the float
\end{figure}
```

The `\forceversofloat` command causes the algorithm to assume the float has been placed on a verso page—that is, a page on the left side of a two-page spread. Conversely, the `\forcerectofloat` command causes the algorithm to assume the float has been placed on a recto page—that is, a page on the right side of a two-page spread.

Full-width text blocks

In addition to the new float types, there is a `fullwidth` environment that stretches across the main text block and the sidenotes area.

```
\begin{fullwidth}
Lorem ipsum dolor sit amet...
\end{fullwidth}
```

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Typography

Typefaces

If the Palatino, Helvetica, and `Bera Mono` typefaces are installed, this style will use them automatically. Otherwise, we'll fall back on the Computer Modern typefaces.

Letterspacing

This document class includes two new commands and some improvements on existing commands for letterspacing.

When setting strings of ALL CAPS or SMALL CAPS, the letterspacing—that is, the spacing between the letters—should be increased slightly.[7] The `\allcaps` command has proper letterspacing for strings of FULL CAPITAL LETTERS, and the `\smallcaps` command has letterspacing for SMALL CAPITAL LETTERS. These commands will also automatically convert the case of the text to upper- or lowercase, respectively.

[7]

The `\textsc` command has also been redefined to include letterspacing. The case of the `\textsc` argument is left as is, however. This allows one to use both uppercase and lowercase letters: The Initial Letters Of The Words In This Sentence Are Capitalized.

Document Class Options

The `tufte-book` class is based on the LaTeX `book` document class. Therefore, you can pass any of the typical book options. There are a few options that are specific to the `tufte-book` document class, however.

The `a4paper` option will set the paper size to A4 instead of the default US letter size.

The `sfsidenotes` option will set the sidenotes and title block in a sans serif typeface instead of the default roman.

The `twoside` option will modify the running heads so that the page number is printed on the outside edge (as opposed to always printing the page number on the right-side edge in `oneside` mode).

The `symmetric` option typesets the sidenotes on the outside edge of the page. This is how books are traditionally printed, but is contrary to Tufte's book design which sets the sidenotes on the right side of the page. This option implicitly sets the `twoside` option.

The `justified` option sets all the text fully justified (flush left and right). The default is to set the text ragged right. The body text of Tufte's books are set ragged right. This prevents needless hyphenation and makes it easier to read the text in the slightly narrower column.

The `bidi` option loads the `bidi` package which is used with XeLaTeX to typeset bi-directional text. Since the `bidi` package needs to be loaded before the sidenotes and cite commands are defined, it can't be loaded in the document preamble.

The `debug` option causes the Tufte-LaTeX classes to output debug information to the log file which is useful in troubleshooting bugs. It will also cause the graphics to be replaced by outlines.

The `nofonts` option prevents the Tufte-LaTeX classes from automatically loading the Palatino and Helvetica typefaces. You should use this option if you wish to load your own fonts. If you're using XeLaTeX, this option is implied (*i.e.*, the Palatino and Helvetica fonts aren't loaded if you use XeLaTeX).

The `nols` option inhibits the letterspacing code. The Tufte-LaTeX classes try to load the appropriate letterspacing package (either pdfTeX's `letterspace` package or the `soul` package). If you're using XeLaTeX with `fontenc`, however, you should configure your own letterspacing.

The `notitlepage` option causes `\maketitle` to generate a title block instead of a title page. The `book` class defaults to a title page and the `handout` class defaults to the title block. There is an analogous `titlepage` option that forces `\maketitle` to generate a full title page instead of the title block.

The `notoc` option suppresses Tufte-LaTeX's custom table of contents (TOC) design. The current TOC design only shows unnumbered chapter titles; it doesn't show sections or subsections. The `notoc` option will revert to LaTeX's TOC design.

The `nohyper` option prevents the `hyperref` package from being loaded. The default is to load the `hyperref` package and use the `\title` and `\author` contents as metadata for the generated PDF.

Hyperparameters

As stated by Li *et al.* in [8], "performance of machine learning algorithms depends critically on identifying a good set of hyperparameters". Hyperparameter tuning is a topic often discussed in literature describing any combination of machine learning algorithms. Adequate values set for hyperparameters will massively influence training of a model, be it in terms of processing time, as well as the actual robustness of the model in question.

[8] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, 18(1):6765–6816, 2017

Introduction

Defining Hyperparameters

Hyperparameters are variables which are initialised before a machine learning model undergoes training. The general representation is as a vector, with common examples of hyperparameters including the learning rate of a model, regularization coefficients, and kernel parameters. Considering the learning rate, an improperly set value will greatly influence the training potential of a regression model or neural network. Extremely high learning rate values may render a system volatile, causing weight/parameter values to fluctuate very easily, whilst lower values may inhibit learning capabilities. Similarly, altering the regularization coefficient for a Support Vector Machine (SVM) may greatly affect the shape of its decision boundary.

Approaches for Hyperparameter Optimization

Owing to the necessity of optimal hyperparameter values in machine learning problems, various works of literature are dedicated solely to their optimization. During the past decade, methods have been refined and applied in a large number of model methods. The sections below will go through a number of these approaches found in published works.

Gradient-Based Hyperparameter Optimization

A particular project by Sathiya *et al.*[9] investigates hyperparameter values in SVMs and Bayesian models. In this system, the variables C and γ are considered. C is the regularization coefficient, influencing how leniently the SVM hyperplane classifies observations. γ defines

[9] S Sathiya Keerthi, Vikas Sindhwani, and Olivier Chapelle. An efficient method for gradient-based adaptation of hyperparameters in svm models., 01 2006

the influence of points on the variance of a model. Experiments in this study showed that performance of the SVM varied considerably as a function of hyperparameter values. Gradient-based optimization techniques were used, computing the slope of a performance validation function with respect to the vector h of hyperparameters. The gradient-based system described allowed for a fast method to determine optimal sets of hyperparameters, which outperformed a *grid search* method even for models with solely 2 hyperparameters.

Hyperparameter Optimization Through Adaptive Resource Allocation

As stated by Li *et al.* in a more recently published paper[10], the interaction between hyperparameters themselves is often not well understood. Generally, the main point of interest in a system is how the hyperparameters influence its performance, rather than how they affect each other as one configuration. This leads to brute force methods being used, such as the grid search method mentioned above, and random searching. Whilst these methods may be used to find good values of hyperparameters, they will more than likely result in suboptimal settings, especially for larger scale models. Bayesian techniques aim to optimize hyperparameter configurations by selecting them adaptively, beating out brute-force methods. Adaptive resource allocation is one such method which is used for selection. Strong hyperparameter configurations bearing positive results are provided more resources, such as an increased amount of training attributes. Other reward mechanisms include a larger number of features and epochs being provided. Other settings with less promising results may be discarded or allotted an inferior amount of resources.

[10] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, 18(1):6765–6816, 2017

Bandit-Based Approach in Conjunction with Adaptive Allocation

The HYPERBAND system was constructed by the same authors as above. It attempts to evaluate randomly-sampled hyperparameter configurations by adaptively assigning the corresponding amount of resources according to validation losses. This is termed a bandit-based approach since the project makes use of 'infinite armed bandits' in its implementation. Bandit arms represent mean-level reward distributions from particular experiments. The reward budget of each arm is essential for selection of the best distributions. As detailed by Wang *et al.* in [11], multi-armed bandit problems focus on *exploration* vs. *exploitation*. Arms can be 'exploited', meaning that they performed well and may be selected for their respective task. Exploration may entail pulling entirely new arms, or pulling already discovered ones of interest. Scoping back to the HYPERBAND paper, the focus is on exploration infinite-armed bandit problems, selecting arms within the top reward percentiles for hyperparameter optimization.

[11] Yizao Wang, Jean-Yves Audibert, and Rémi Munos. Algorithms for infinitely many-armed bandits. In *Advances in Neural Information Processing Systems*, pages 1729–1736, 2009

Scenario: Hyperparameter Tuning for Gradient Descent

This scenario will look into a practical example of how hyperparameter values affect the performance of a gradient descent algorithm. Consider the following setup with cost function:

$$y = 4x^2 - 2x - 1 \tag{1}$$
$$dy/dx = 8x - 2 \tag{2}$$

The objective of gradient descent algorithms is to minimize a function. Mathematically, this implies that the algorithm will cycle through different values of x until a minimum value of y is obtained, generally to a predefined precision. This process may be represented by the decrease in magnitude of the gradient of the function, hence the inclusion of the first derivative. The primary hyperparameter to be optimized in this case will be α, representing the learning rate. As previously stated, the learning rate will affect the capability of the system to evolve. Values are calculated as follows for each iteration:

$$x_{new} = x_i - \alpha(dy/dx_i) \tag{3}$$
$$y_{new} = (4(x_{new}^2) - 2x_{new} - 1 \tag{4}$$

Another important parameter in this setup is x_0, which is the initial x-value to be used as an estimate for the minimum point. The following plot shows the algorithm with an α of 0.0001:

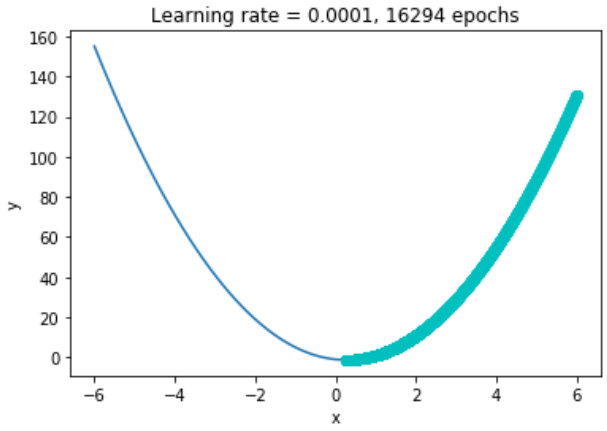

Figure 4: Plot of gradient descent with $\alpha = 0.0001$

In the figure above, x_0 has been set to 6 to simulate a poor estimate for the minimum x value. It can be observed from this plot, that the algorithm took 16294 iterations to converge on the local minima. The below plots will show the same setup with α values of 0.005 and 0.1:

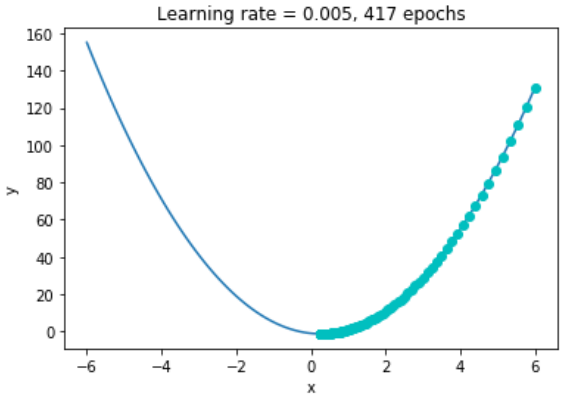

Figure 5: Plot of gradient descent with $\alpha = 0.005$

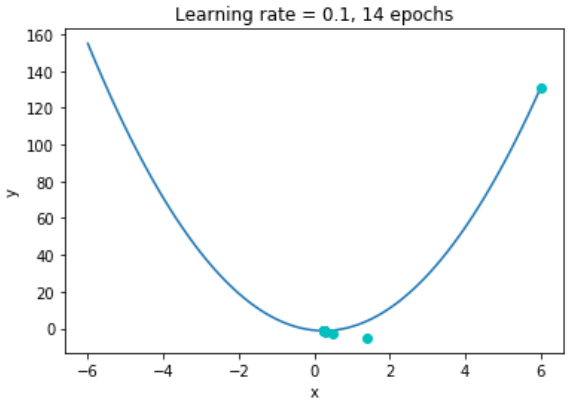

Figure 6: Plot of gradient descent with $\alpha = 0.1$

Increasing the learning rate will allow for the algorithm to converge much faster in this case. It is noteworthy that this was a straightforward example for visualization purposes. Learning models will generally be composed of various hyperparameters, and even variables such as the learning rate have certain intricacies, as discussed briefly above.

Bibliography

James Bergstra, Daniel Yamins, and David Daniel Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. 2013.

D. Cox and N. Pinto. Beyond simple features a large-scale feature search approach to unconstrained face recognition. In *Face and Gesture 2011*, pages 8–15, March 2011. DOI: 10.1109FG.2011.5771385.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, 18(1):6765–6816, 2017.

S Sathiya Keerthi, Vikas Sindhwani, and Olivier Chapelle. An efficient method for gradient-based adaptation of hyperparameters in svm models., 01 2006.

Yizao Wang, Jean-Yves Audibert, and Rémi Munos. Algorithms for infinitely many-armed bandits. In *Advances in Neural Information Processing Systems*, pages 1729–1736, 2009.