⟨Dissertation Title⟩

Louie Middle

Bachelor of Science in Computer Science
The University of Bath
2023

⟨Dissertation Title⟩

Submitted by: Louie Middle

Copyright

Declaration

This dissertation is submitted to the University of Bath in accordance with the requirements of the degree of Bachelor of Science in the Department of Computer Science. No portion of the work in this dissertation has been submitted in support of an application for any other degree or qualification of this or any other university or institution of learning. Except where specifically acknowledged, it is the work of the author.

Abstract

⟨ The abstract should appear here. An abstract is a short paragraph describing the aims of the project, what was achieved and what contributions it has made. ⟩

Contents

Acknowledgements **vi**

1 Introduction **1**
1.1 Moneyball . . . 1
1.2 Indian Premier League . . . 2
1.3 Project Plan . . . 2
1.4 Resources Required . . . 3

2 Literature and Technology Survey **4**
2.1 Existing Machine Learning in Sport . . . 4
2.2 Existing Machine Learning in Cricket . . . 5
2.2.1 Existing Machine Learning in the Indian Premier League . . . 5
2.3 Research with similar goals . . . 5
2.4 Drawbacks of Existing Methods . . . 6
2.5 Gaussian Processes in Sport . . . 6

3 Background **7**
3.1 Gaussian Processes . . . 7
3.1.1 Gaussian Processes for Regression . . . 7
3.1.2 Kernels . . . 9
3.1.3 Predictions and Posterior . . . 10
3.1.4 Gaussian Processes for Classification . . . 10
3.1.5 Sparse Gaussian Processes . . . 11
3.2 Data Association with Gaussian Processes . . . 11
3.2.1 Data Association . . . 11
3.2.2 Summary . . . 11
3.2.3 Covariance Function . . . 12
3.2.4 GP Layers . . . 12
3.2.5 Inducing Points . . . 13
3.2.6 Gaussian Likelihood . . . 13
3.2.7 Prediction and Assignment Layers . . . 14

4 Results **15**

5 Discussion **16**

6 Conclusion **17**

7 Future Work **18**

8 Personal Experiences **19**
8.1 Short Section Title 19
8.2 Example Lists 19
8.2.1 Enumerated 19
8.2.2 Itemised 19
8.2.3 Description 20

Bibliography **21**

A Mathematical Background **23**

B Design Diagrams **24**

C Raw Results Output **25**

D Code **26**
D.1 File: yourCodeFile.java 27

List of Figures

1.1 Gantt Chart For Project 3

2.1 Number of papers using a particular ML algorithm group (Horvat and Job, 2020) 4

3.1 SE kernel with $\sigma = 1$ and $l = 1$ (Kaiser, 2017) 10
3.2 SE kernel with $\sigma = \sqrt{2}$ and $l = 1$ (Kaiser, 2017) 10
3.3 SE kernel with $\sigma = 1$ and $l = 0.25$ (Kaiser, 2017) 10
3.4 A data association problem consisting of two generating processes, one of which is a signl we wish to recover and one is an uncorrelated noise process (Kaiser et al., 2018) 11

List of Tables

8.1 An example table 19

Acknowledgements

I would like to acknowledge my supervisor Adam Hartshorne. I would like to acknowledge my partner Hoi Ching Leung. I would like to acknowledge my parents.

Chapter 1

Introduction

Over the last few decades the amount of data driven techniques to improve the outcomes of sports games has increased greatly. The multi-billion pound market of cricket is no exception. There is a strong incentive to improve the techniques used to better the results of teams. This study aims to investigate the possibility of predicting the outcome of a bowler batter match-up in cricket using modern machine learning techniques and how this could then aid cricket bowling choices and team selection. The main training and testing data will be the 2022 Indian Premier League (IPL) season (TODO: Change) The hope is that using knowledge of bowler pitch trajectories and batter shot trajectories gathered from modern ball tracking will add another layer of granularity in addition to simply considering the resultant runs scored of each delivery. Furthermore, building a model which can incorporate pitch, atmospheric and ground conditions could further improve any models.

1.1 Moneyball

The release of Moneyball (Lewis, 2004), was a big driver in the increased use of statistical driven techniques in Baseball player selection and scouting techniques. Pioneered by the likes of Bill James and Sabermetrics, Moneyball has entered baseball's lexicon; teams that value Sabermetrics are often said to be playing "Moneyball". One of the notable benefits of a Moneyball approach is in reducing player salaries, whilst maintaining high performance. Notable recent Moneyball successes include the Tampa Bays, whose entire 2019 roster was around 63% of the total budget of the $40 million the Houston Astros were spending on Gerrit Cole's contract. It was reported that the Rays spending totalled $648,000 per victory, compared to the Astros $1.54 million per win. Despite this large difference pay, the Rays still had a successful season and finished second in the American League East (Sports, 2019).

Similar successes can be found in other sports, such as association football (or soccer). In 2010 Liverpool F.C. were purchased by Boston Red Sox owner John W. Henry. With the Red Sox, Henry hired Sabermetrics pioneer Bill James and their Moneyball approach saw the team win the World Series in 2004, 2007, 2013 and 2018. With Liverpool, Henry hired University of Cambridge PhD Ian Graham in 2012 as head of analysis and Jürgen Klopp as Manager in 2015 (News, 2022). Graham influenced the signings of key players, such as Mohammed Salah, Philippe Coutinho and Naby Keïta. Grahams data suggested Salah would pair especially well with Roberto Firmino, who creates more expected goals than nearly anyone else in his position (Echo, 2019). Expected goals turned to real goals in the 2017-2018 season, with Salah scoring

32 goals and Firmino scoring 15. The combination of Klopp and his intuitive knowledge, mixed with the likes of Grahams data-driven knowledge, has led to Liverpool having fantastic recent success winning the 2018-2019 UEFA Champions League and the 2019-2020 English Premier League.

1.2 Indian Premier League

The Indian Premier League (IPL) is a professional cricket league based on the Twenty20 format. As reported by ESPNcricinfo (2018), Star Sports invested $2.55 billion for exclusive broadcasting rights for the 2018 IPL season. This season saw a 29% increment in the number of viewers, through both digital streaming and television. The interest in the IPL is clear to see, thus increasing the desire to use modern techniques to improve results.

1.3 Project Plan

TODO: This can be removed

There are 26 weeks from Friday the 4th November until the final deadline of Friday the 5th May. The individual project is 24 credits out of a total 60 credits for the year, meaning 40% of my time can be used for the project. This is 10.4 weeks. To allow for buffer and holidays I will assume I have 8 working weeks to complete the project. I have split my project into 3 main sections:

1. Literature review and pre-processing (2 weeks)
2. Developing and improving models (4 weeks)
3. Analysis of models and write up (2 weeks)

The buffer time can be used for any road bumps, or sections that need it. See Gantt chart for overview of plan in figure 1.1.

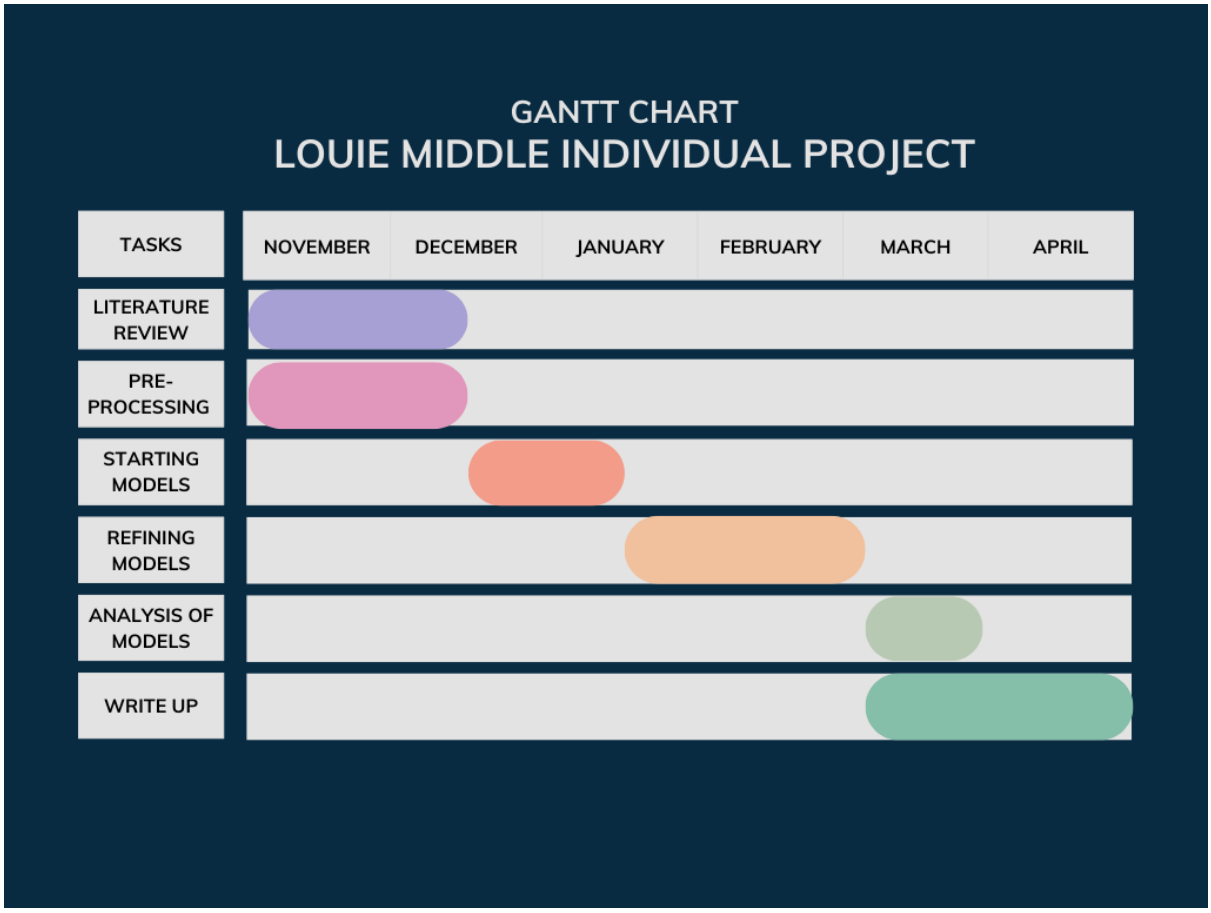

Figure 1.1: Gantt Chart For Project

1.4 Resources Required

TODO: This section should be removed or amended. I think get some data from just one year, or all years and compare to test Horvat and Job.

In order to train any machine learning model an appropriate amount of data for training and testing. Because it is not beneficial to use data that is too old (Horvat and Job, 2020), a recent season's worth of data would be good, saving some data for testing.

Training machine learning models may also require appropriate computing power depending on the models used and size of the data set. This could potentially be achieved with the University of Bath's Hex cluster.

Chapter 2

Literature and Technology Survey

2.1 Existing Machine Learning in Sport

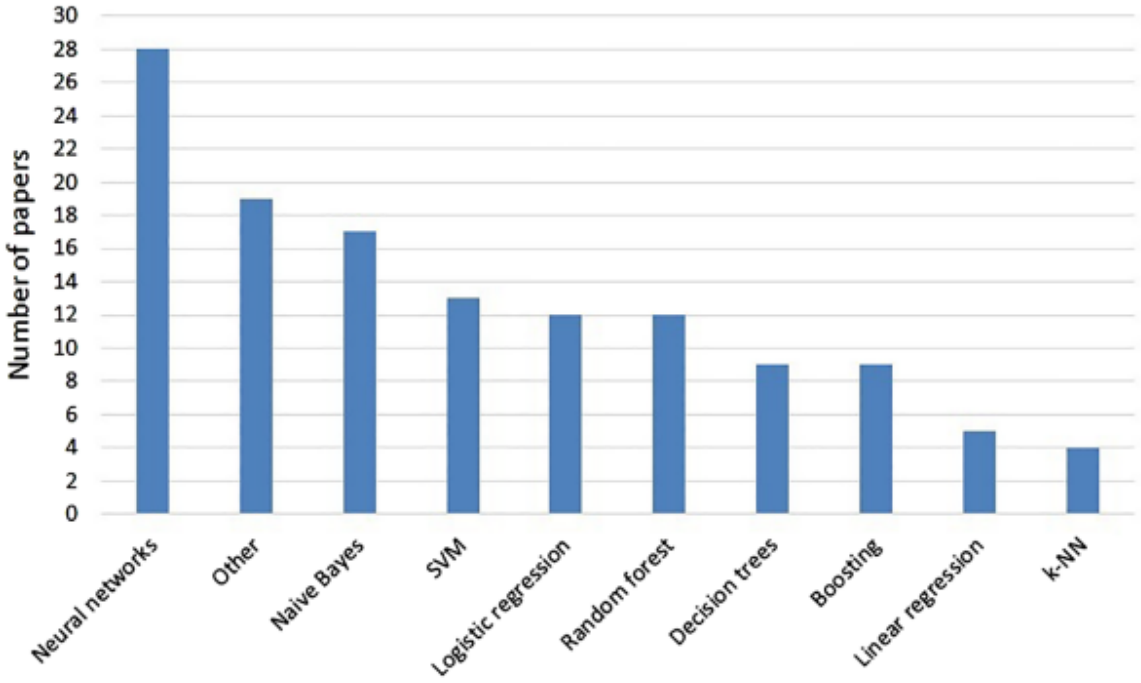

Figure 2.1: Number of papers using a particular ML algorithm group (Horvat and Job, 2020)

TODO: Could be worth putting the one with publication year in to show uptick in research? But not necessary right now.

Figure 2.1 from Horvat and Job (2020)'s literature review of machine learning in sport for score prediction shows what current methods are currently used in the field as of May 2020. As you can see, neural networks have a large research output, followed by other popular methods such as regression, SVMs, regression, decision trees, and gradient boosting with k-NN being the least used. It is important to note, that just because a large body of research has been carried out with certain methods, it does not necessarily mean they are the best.

Horvat and Job (2020) also showed that including too many seasons worth of data for training models reduces the quality of results. To those with a basic knowledge of cricket and sport this is not surprising given that in just a few years, many things related to team composition and tactics can change. The best prediction results were achieved by researchers who used data from a single season and a data segmentation evaluation method. When using data from a single season, most of the data is used for training and a small portion for testing. Some researchers used the same data set for training and testing yielding unrealistically accurate results.

2.2 Existing Machine Learning in Cricket

Kampakis and Thomas (2015) used Naïve Bayes, logistic regression, random forests and gradient boosted decision trees to predict the outcome of English County 20 over Cricket Matches from 2009 - 2014. The performance of each algorithm was assessed using one year's data as the training data set and the following year's data for testing. Each model was tested over these seasons and achieved an accuracy of 62.4% for Naïve Bayes, 60.1% for logistic regression, 55.6% for random forests and 57.2% for decision trees.

2.2.1 Existing Machine Learning in the Indian Premier League

Saikia, Bhattacharjee and Lemmer (2012) have used Artificial Neural Network models to predict the performance of bowlers based on their performance in the first three seasons of the IPL. When the predicted results were validated with the players performances in the fourth season, the model had an accuracy of 71.43%.

2.3 Research with similar goals

The Singlearity-PA model (Silver and Huffman, 2021) wanted to attempt to solve one of the most fundamental questions in baseball: How can we predict the outcome of a batter vs. pitcher plate appearance (PA)? This is similar to my goal with cricket: To predict the outcome of batter vs bowler for an over.

The Singlearity-PA model (Silver and Huffman, 2021) was able to accurately predict the results of a batter versus pitcher plate appearance using a neural-network based AI model. The details of the model used are vague, however the network was able to take in 87 inputs and then output probabilities for each of the 21 possible outcomes of a plate appearance (PA) in baseball. Comparisons can be made between this and cricket. A plate appearance can be compared to an over, comprising 6 balls (or more including no balls and wides) between a single bowler and 1 or more batsmen. The outcome of the over could be considered to be the runs scored.

Silver and Huffman (2021) also split their player base up by how many PAs they had for each player. The best players had greater than 500 PAs worth of data each, but the vast majority had less than 100. SinglearityPA was able to accurately predict the result of match-ups for these players with fewer PAs better than existing solutions. Parallels can be drawn between this and cricket, as there are often players who have little data, yet team selectors would want to know who the best player is to match-up against them.

Extending Singlearity-PA with Markov chains improved more complicated strategies, such as

optimal player lineups or to decide on pinch hitters and relief pitchers. Similarly, in cricket the batting lineup and choice of bowler at different points in a game have a large impact on the score. In the example provided, Singlearity-PA's predicted runs scored for an optimal lineup was 6.7% better than the actual lineup in the 2019 National League All-Star game. It is important not to compare baseball and cricket too closely, but the techniques used by Silver and Huffman could potentially work well in Cricket.

2.4 Drawbacks of Existing Methods

Sport outcome predictors are most commonly used by supervised ML methods, typically classification methods or regression methods (Horvat and Job, 2020). Whilst existing research can achieve impressive results, surprises in sport still happen. For example, the odds of Leicester City winning the Premier League in the 2015/2016 season were 1-5000. However, analysis of their performances show their title was absolutely deserved.

This could be explained in that a problem with existing models is that they only output a single value. There is no uncertainty in the output as to how confident the model is in its prediction. This has a problem in that making decisions based on this prediction becomes much more difficult, as decision makers can't be sure how much to trust the prediction. Furthermore, certain inputs exist where very little similar training data might exist. In such cases, the uncertainty in a models output should be much greater. Yet, existing models will still provide a prediction the same as it would for inputs where there was a large amount of training data.

TODO: talk about (Blumberg, 2020).

2.5 Gaussian Processes in Sport

TODO: Talk about Gaussian processes in sport

Chapter 3

Background

3.1 Gaussian Processes

TODO: Put cholesky decomposition and KL divergence in the appendix

This following section introduces Gaussian processes and the following section then explains how to extend their application to solve data association problems. This project is mostly concerned with using Gaussian processes for classification, but a background on using Gaussian processes for regression is provided first as a basis to their extension to classification tasks.

Gaussian processes are a supervised learning technique, where we start with a training data set $\mathcal{D}$ of n observations, $\mathcal{D} = (\mathbf{x}_i, y_i | i = 1, ..., n)$ where $\mathbf{x}$ denotes and input vector (covariates) of dimension D and y denotes a scalar output or target. The column vector inputs for all n cases are aggregated in the Dxn design matrix X and the targets collected in the vector $\mathbf{y}$, such that $\mathcal{D} = (X, \mathbf{y})$.

3.1.1 Gaussian Processes for Regression

Weight Space View

TODO: This section might not be necessary, but might be good to show understanding....

One can think of a Gaussian process as defining a distribution over functions and inference taking place directly in the space of functions. Although this view is appealing, it is difficult to grasp on first attempt, and so we will start with reviewing the *weight-space view*.

First, lets review the standard linear regression model with Gaussian noise

$$f(\mathbf{x}) = \mathbf{x}^T \mathbf{w}, \quad y = f(\mathbf{x}) + \epsilon \tag{3.1}$$

where $\mathbf{x}$ is the input vector, $\mathbf{w}$ is a vector of weights of the linear model, f is the function value, and y is the observed target value. It is often assumed that the observed values differ from the function values by some noise, which we will treat as an independent, identically distributed Gaussian distribution, with zero mean and variance σ_n^2.

$$\epsilon \sim \mathcal{N}(0, \sigma_n^2) \tag{3.2}$$

This noise assumption together with the model gives rise to the likelihood, the probability density of the observations given the parameters, which is factored over cases in the training set to give

$$p(\mathbf{y}|X,\mathbf{w}) = \prod_{i=1}^{n} = p(y_i|\mathbf{x}_i,\mathbf{w}) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_n} exp(-\frac{(y_i - \mathbf{x}_i^T\mathbf{w})^2}{2\sigma_n^2}) \, TODO : Candotherestlater \tag{3.3}$$

We put a zero mean Gaussian prior with covariance matrix Σ_p on the weights

$$\mathbf{w} \sim \mathcal{N}(0, \Sigma_p) \tag{3.4}$$

Inference is based on the posterior distribution over the weights computed by Baye's rule, given by

$$P(\mathbf{w}|\mathbf{y}, X) = \frac{p(\mathbf{y}|X,\mathbf{w})p(\mathbf{w})}{p(\mathbf{y}|X)} \tag{3.5}$$

Note that prior $p(\mathbf{w})$ neglects the conditioning on X, as it is independent of the inputs. The normalising constant $p(\mathbf{y}|X)$, also know as the marginal likelihood is independent of the weights and is given by

$$P(\mathbf{y}|X) = \int p(\mathbf{y}|X,\mathbf{w})p(\mathbf{w})d\mathbf{w} \tag{3.6}$$

TODO: Finish this later when you can ask Adam about it....

Since the number of obervations is finite and the function f lives in an infinite dimensional function space, the estimation of f is uncertain and base on prior assumptions about its structure. Gaussian processes are a non-parametric regression model. *Non-parametric models* are not based on insights about the concrete structure of the function to be modelled, but instead make assumptions about the function itself, such as its smoothness or differentiability. Instead of modeling a distribution of parameter values, a non-parametric model tries to find a distribution $p(f*)$ of probable functions that represents the function f to be estimated.

(Yi, 2019) TODO: This has good explanations of stuff

To make predictions we average over all possible parameter values, weighted by their posterior probability. Thus the predictive distribution for $f_* \triangleq f(\mathbf{x}_*)$ at $\mathbf{x}_*$ is given by averaging the output of all possible linear models w.r.t the Gaussian posterior

$$\begin{aligned} p(f_*|\mathbf{x}_*, X, \mathbf{y}) &= \int p(f_*|\mathbf{x}_*,\mathbf{w})p(\mathbf{w}|X,\mathbf{y})d\mathbf{w} \\ &= \mathcal{N}(\frac{1}{\sigma_n^2}\mathbf{x}_*^T A^{-1} X\mathbf{y}, \;\; \mathbf{x}_*^T A^{-1}\mathbf{x}_*). \end{aligned} \tag{3.7}$$

The predictive distribution is again Gaussian.

Function-space View

Definition 3.1.1 (Gaussian Process)**.** A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.

A Gaussian process is completely specified by its mean function, $m(\mathbf{x})$, and its co-variance function $k(\mathbf{x}, \mathbf{x}')$. Usually the mean function will be taken to be zero, but this need not be the case. A mean function and covariance function can be defined for a real process $f(x)$ as

$$\begin{aligned} m(\mathbf{x}) &= \mathbb{E}[f(\mathbf{x})], \\ k(\mathbf{x}, \mathbf{x}') &= \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))], \end{aligned} \tag{3.8}$$

where we will write the Gaussian process as

$$f(\mathbf{x}) \sim GP(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')). \tag{3.9}$$

The covariance function, also called a kernel, specifies the covariance between pairs of random variables. It is the covariance functions which encode the assumptions about the underlying function.

3.1.2 Kernels

Kernels are crucial in encoding the assumptions about the function a Gaussian process should estimate. It is a measure of similarity of different points in the observed data and of new points to be predicted. For example, a natural assumption is to assume the closer two points lie, the more similar their function values should be. Furthermore, when prediciting test points, training points close to it are probably more informative than those further away. However, it should be noted, that this need not be the case. For example, consider a sinusoidal wave where two points which are multiple wavelengths apart should have similar function values. A kernel that only depends on the distance between two points is called *stationary*. Conversely, kernels that do depend on two points position in the input space are called *non-stationary*.

A common kernel and one used throughout this project is the squared exponential (SE) covariance function (also known as the radial basis function, RBF). This is defined by

$$k(\mathbf{x}, \mathbf{x}') = exp(-\frac{1}{2}|\mathbf{x}_p - \mathbf{x}_q|^2). \tag{3.10}$$

Kernels have characteristic length scales which informally can be thought of as roughly the distance you have to move in input space before the function value will change significantly. For eq 3.10 the length-scale is one. By replacing $\mathbf{x}_p - \mathbf{x}_q$ with $\mathbf{x}_p - \mathbf{x}_q/l$ for some positive constant l we can change the characteristic length-scale of the process. Choice of such parameters will be discussed more later.

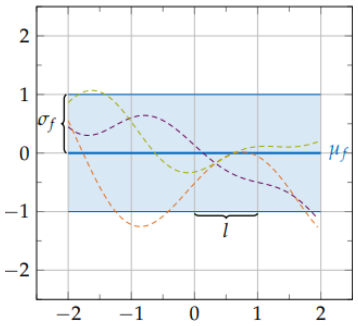

Figure 3.1: SE kernel with $\sigma = 1$ and $l = 1$ (Kaiser, 2017)

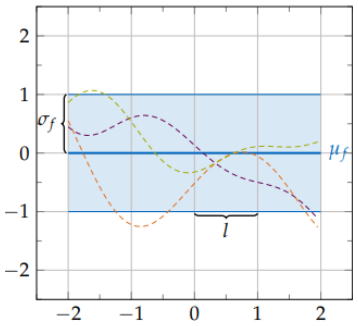

Figure 3.2: SE kernel with $\sigma = \sqrt{2}$ and $l = 1$ (Kaiser, 2017)

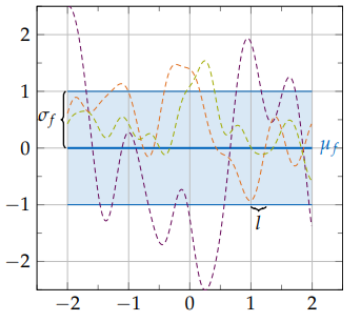

Figure 3.3: SE kernel with $\sigma = 1$ and $l = 0.25$ (Kaiser, 2017)

Figure 3.1, figure 3.2 and figure 3.3 compare sample functions drawn from Gaussian processes with SE kernels with different hyperparameters.

3.1.3 Predictions and Posterior

The joint distribution of the training outputs, $\mathbf{f}$, and the test outputs $\mathbf{f}_*$ according to the prior is

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K(X,X) + \sigma_n^2 I & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right). \tag{3.11}$$

If there are n training points and n_* test points then $K(X, X_*)$ denotes the nxn_* matrix of covariances evaluated at all pairs of training and test points, and similarly for $K(X,X)$, $K(X_*,X)$ and $K(X_*,X_*)$.

In order to use Gaussian processes for regression, it is necessary to combine observations with a Gaussian process prior $f \sim GP(0,K)$. The distribution is obtained by integrating over all possible latent function values f and therefore taking all possible functions into account. This is called the *marginilisation of f*.

The joint prior in equation 3.11 may be conditioned on the observations through $p(\mathbf{f}_*|\mathbf{y}) = \frac{p(\mathbf{f}_*,\mathbf{y})}{p(\mathbf{y})}$ which enforces that the joint prior agrees with the observations $\mathbf{y}$. The posterior ... TODO: How does he make the following jumps?

(Griffiths, 2023)

3.1.4 Gaussian Processes for Classification

A Gaussian process is a generalisation of the gaussian probability distribution. Both classification and regression can be seen as function approximation problems. Unfortunately, the solution of classification problems using Gaussian processes is tougher than regression problems. For regression problems, the likelihood is often assumed to be Gaussian. A Gaussian process prior combined with a Gaussian likelihood gives a posterior Gaussian process over functions, where

everything remains analytically tractable. For classification models, the Gaussian likelihood is inappropriate; a different likelihood such as a Bernoulli likelihood must be used.

(Rasmussen and Williams, 2006)

3.1.5 Sparse Gaussian Processes

TODO: Write about sparse GPs

3.2 Data Association with Gaussian Processes

3.2.1 Data Association

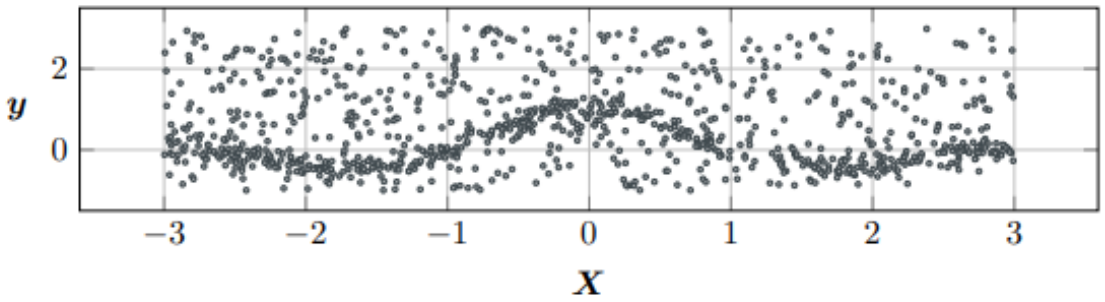

Figure 3.4: A data association problem consisting of two generating processes, one of which is a signl we wish to recover and one is an uncorrelated noise process (Kaiser et al., 2018)

A *data association problem* is one where we consider the data to have been generated by a mixture of processes and we are interested in factorising the data into these components. For example, as described by Kaiser et al. (2018), figure 3.4 could represent faulty sensor data, where sensor readings are disturbed by uncorrelated and asymettric noise. Standard machine learning approaches can pollute any models, where the model starts to explain the noise instead of the underlying signal.

TODO: explain why the cricket problem is a data association problem.

(Kaiser et al., 2018) (Liu et al., 2020)

3.2.2 Summary

A GP is a probabilistic model used for regression and classification tasks in machine learning. It models the relationship between inputs and outputs as a random process and makes predictions by modeling the distribution of outputs given the inputs.

First, two kernel objects "pred_kernel" and "assign_kernel" are created using the "KERNEL.kern" function. These kernels define the covariance function used in the GP model. The input dimension, lengthscales, variance, and ARD (Automatic Relevance Determination) flag are set using the "input_dim", "lengthscales", "variance", and "ARD" arguments, respectively.

The centroids "Z" and "Z_assign" of the input data "Xtrain" are computed using the k-means clustering algorithm "kmeans", where "num_ind" is the number of inducing points.

Two GP layers "pred_layer" and "assign_layer" are created using the "SVGP_Layer" class. The "pred_layer" uses the "pred_kernel" and the "Z" centroids, while the "assign_layer" uses the "assign_kernel" and the "Z_assign" centroids. The number of inducing points and outputs is set using the "num_inducing" and "num_outputs" arguments.

A Gaussian likelihood object "lik" is created using the "Gaussian" class. This likelihood object defines the probability distribution over the outputs.

Finally, a sparse Multi-output Gaussian Process (SMGP) model is created using the "SMGP" class. The "likelihood" argument is set to "lik", and the "pred_layer" and "assign_layer" arguments are set to "pred_layer" and "assign_layer" respectively. The number of output dimensions "K", the number of Monte Carlo samples "num_samples", and the number of data points "num_data" are also specified.

3.2.3 Covariance Function

The covariance function, also known as the kernel function, is a key component in Gaussian Process (GP) models. It defines the covariance or similarity between the input points. The covariance function models the assumptions about the underlying function that generates the data.

The covariance function takes two inputs, x and x', and returns the covariance between the two inputs. This can be interpreted as the degree of similarity between the two inputs. The choice of the covariance function determines the type of GP model and its flexibility in fitting the data. Common choices for covariance functions include radial basis functions, polynomial functions, and the squared exponential function.

The covariance function can be parameterized by several hyperparameters, such as the lengthscales, which determine the smoothness of the function, and the variance, which determines the magnitude of the output. These hyperparameters can be learned during the training process.

In summary, the covariance function in a GP model determines the similarity between the inputs, and is a critical component in defining the assumptions about the underlying function and its flexibility in fitting the data.

3.2.4 GP Layers

TODO: This section can potentially never be mentioned, as now using GPFlow SVGP which isn't called 'layer'.

In Gaussian Process (GP) models, layers can be used to define the structure of the model and simplify the modeling process. A layer in a GP model typically consists of a set of inducing points, which are used to make predictions about the output. The choice of inducing points and the number of inducing points can greatly affect the accuracy and computational efficiency of the GP model.

Layers can be used to model different aspects of the data, such as the mean function and the covariance function. The mean function can be modeled as a linear combination of the

inputs, while the covariance function can be defined by a radial basis function or a polynomial function.

Layers can also be used to define hierarchical GP models, where different levels of the hierarchy are modeled by different layers. This allows the model to capture complex dependencies between the inputs and outputs.

In summary, layers in GP models provide a flexible and powerful way to define the structure of the model and make predictions about the output. They can be used to model different aspects of the data, and can help simplify the modeling process while improving the accuracy and computational efficiency of the model.

3.2.5 Inducing Points

In the code you provided, the inducing points are calculated using the k-means clustering method. This is because k-means clustering is a commonly used method for selecting a fixed set of representative points that summarize the structure of the data.

The purpose of the inducing points in a Gaussian Process (GP) model is to provide a compact representation of the model that can be used to make predictions about the output. The inducing points are typically chosen to be a small subset of the inputs, and the model is defined in terms of the inducing points rather than the inputs themselves. This results in a more computationally efficient model that can be trained and used to make predictions faster.

By using the k-means clustering method to calculate the inducing points, the model can take advantage of the structure of the data. The k-means algorithm clusters the data into a set of k clusters, where each cluster is represented by its centroid. These centroids can be used as the inducing points in the GP model.

In summary, the use of k-means clustering to calculate the inducing points in a GP model is an efficient way to summarize the structure of the data and reduce the dimensionality of the model, while still maintaining its ability to make accurate predictions about the output.

3.2.6 Gaussian Likelihood

TODO: Will need to add a Bernoulli / Multinomial likelihood section?

In a Gaussian Process (GP) model, the likelihood of the data is the probability of observing the data given the model parameters. The Gaussian likelihood is a commonly used likelihood function for GP models, and it assumes that the output data is normally distributed.

The Gaussian likelihood is calculated as follows:

Define the mean function, $\mu(x)$, and the covariance function, $K(x, x')$, for the GP model. These functions define the structure of the model and determine how the inputs x and outputs y are related.

Calculate the predicted mean, μ, and the predicted covariance, Σ, of the output data given the inputs and the model parameters. This can be done using the mean function and the covariance function.

Calculate the likelihood of the data, $p(y|x, \mu, \Sigma)$, using the predicted mean and covariance. The likelihood is given by:

$$p(y|x,\mu,\Sigma) = (2\pi)^(-n/2) * det(\Sigma)^(-1/2) * exp(-1/2 * (y-\mu)^T * \Sigma^(-1) * (y-\mu)) \quad (3.12)$$

where n is the number of data points, y is the vector of observed outputs, and $det(\Sigma)$ and $\Sigma^(-1)$ are the determinant and inverse of the covariance matrix Σ.

The Gaussian likelihood for the entire data-set is then given by the product of the likelihoods of individual data points:

$$p(y|x) = \Pi_i p(y_i|x_i,\mu,\Sigma) \quad (3.13)$$

In summary, the Gaussian likelihood is calculated by defining the mean function and the covariance function for the GP model, calculating the predicted mean and covariance of the output data, and evaluating the likelihood of the data given the predicted mean and covariance. The Gaussian likelihood provides a way to assess the fit of the model to the data, and it is used to determine the optimal values of the model parameters that maximize the likelihood.

3.2.7 Prediction and Assignment Layers

The pred and assign layers in the code you posted are used in a Sparse Variational Gaussian Process (SVGP) model. SVGP is a type of GP model that is used to handle large datasets, where computing the covariance matrix of all data points is computationally infeasible.

The main idea behind SVGP is to use a smaller number of inducing points to represent the covariance structure of the entire dataset. The pred and assign layers in the code represent two separate models for predicting the outputs of the GP.

The pred layer is used for prediction, i.e., to make predictions for new inputs. This layer models the covariance between the inducing points and the input points. The inducing points are obtained using a clustering method, such as k-means, and they are used to represent the entire dataset.

The assign layer is used to assign each data point to the closest inducing point, based on the similarity between the data points and the inducing points. This layer models the covariance between the data points and the inducing points, and it allows the GP to capture the relationship between the data points and the inducing points.

In summary, the pred and assign layers are used to make predictions and assign data points to the closest inducing points in a SVGP model, respectively. These layers allow the GP to handle large datasets efficiently and effectively.

Chapter 4

Results

If you are doing a primarily software development project, this is the chapter in which you review the requirements decisions and critique the requirements process.

Chapter 5

Discussion

This is the chapter in which you review your design decisions at various levels and critique the design process.

Chapter 6

Conclusion

This is the chapter in which you review the implementation and testing decisions and issues, and critique these processes.

Code can be output inline using `\lstinline|some code|`. For example, this code is inline: `public static int example = 0;` (we have used the character | as a delimiter, but any non-reserved character not in the code text can be used.)

Code snippets can be output using the `\begin{lstlisting} ... \end{lstlisting}` environment with the code given in the environment. For example, consider listing 6.1, below.

Listing 6.1: Example code

```
public static void main() {

  System.out.println("Hello World");

}
```

Code listings are produced using the package 'listings'. This has many useful options, so have a look at the package documentation for further ideas.

Chapter 7

Future Work

This is the chapter in which you review the outcomes, and critique the outcomes process. You may include user evaluation here too.

Chapter 8

Personal Experiences

This is the chapter in which you review the major achievements in the light of your original objectives, critique the process, critique your own learning and identify possible future work.

8.1 Another Section With a Long Title and Whose Title Is Abbreviated in the Table of Contents

Table 8.1: An example table

Items	Values
Item 1	Value 1
Item 2	Value 2

Another section, just for good measure. You can reference a table, figure or equation using `\ref`, just like this reference to Table 8.1.

8.2 Example Lists

8.2.1 Enumerated

1. Example enumerated list:
    - a nested enumerated list item;
    - and another one.
2. Second item in the list.

8.2.2 Itemised

- Example itemised list.
    - A nested itemised list item.
- Second item in the list.

8.2.3 Description

Item 1 First item in the list.

Item 2 Second item in the list.

Bibliography

Blumberg, R., 2020. A fully transparent expected-goals model [Online]. Available from: https://robblumberg.github.io/exp-goals/.

Echo, L., 2019. The unknown liverpool backroom analyst who convinced jurgen klopp to commission transfers of salah and keita [Online]. Available from: https://www.liverpoolecho.co.uk/sport/football/transfer-news/unknown-liverpool-backroom-analyst-who-16320524.

ESPNcricinfo, 2018. How can the ipl become a global sports giant? [Online]. Available from: https://www.espncricinfo.com/story/tim-wigmore-how-can-the-ipl-become-a-global-sports-giant-1150459.

Griffiths, R.R., 2023. *Applications of gaussian processes at extreme lengthscales: From molecules to black holes* [Online]. Ph.D. thesis. Available from: https://arxiv.org/pdf/2303.14291.pdf.

Horvat, T. and Job, J., 2020. The use of machine learning in sport outcome prediction: A review. *Wiley interdisciplinary reviews: Data mining and knowledge discovery* [Online], 10(5), p.e1380. Available from: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1380?casa_token=aXIDtR-gOHQAAAAA%3A-YGsS9bhdiOkYkurvCmBCdZpbAx-MV4VF806rtp_h1CXJlCC0pg4hFFknpmc9dAb3wBAVu7NZ_FOBQ.

Kaiser, M., 2017. *Incorporating uncertainty into reinforcement learning through gaussian processes* [Online]. Master's thesis. Available from: https://github.com/mrksr/masters-thesis/blob/master/thesis/thesis.pdf.

Kaiser, M., Otte, C., Runkler, T. and Ek, C.H., 2018. Data association with gaussian processes [Online]. Available from: https://doi.org/10.48550/ARXIV.1810.07158.

Kampakis, S. and Thomas, W., 2015. Using machine learning to predict the outcome of english county twenty over cricket matches [Online]. Available from: https://arxiv.org/ftp/arxiv/papers/1511/1511.05837.pdf.

Lewis, M., 2004. *Moneyball: The art of winning an unfair game*. WW Norton & Company.

Liu, H., Ong, Y.S., Jiang, X. and Wang, X., 2020. Modulating scalable gaussian processes for expressive statistical learning [Online]. Available from: https://doi.org/10.48550/ARXIV.2008.12922.

News, L.F., 2022. Who is ian graham? liverpool suffer second transfer blow after julian ward departure [Online]. Available from: https://www.liverpool.com/liverpool-fc-news/features/ian-graham-liverpool-transfer-ward-25596815.

Rasmussen, C.E. and Williams, C.K., 2006. *Gaussian process for machine learning* [Online], vol. 2. MIT Press Cambridge, MA. Available from: `http://gaussianprocess.org/gpml/`.

Saikia, H., Bhattacharjee, D. and Lemmer, H.H., 2012. Predicting the performance of bowlers in ipl: An application of artificial neural network. *International journal of performance analysis in sport*, 12(1), pp.75–90.

Silver, J. and Huffman, T., 2021. Baseball predictions and strategies using explainable ai [Online]. *The 15th annual mit sloan sports analytics conference*. Available from: `https://singlearity.com/static/assets/sloan-singlearity.pdf`.

Sports, F., 2019. The tampa bays are the undisputed kings of moneyball [Online]. Available from: `https://www.foxsports.com/stories/mlb/the-tampa-bay-rays-are-the-undisputed-kings-of-moneyball`.

Yi, W., 2019. Understanding gaussian process, the socratic way [Online]. Available from: `https://towardsdatascience.com/understanding-gaussian-process-the-socratic-way-ba02369d804`.

Appendix A

Mathematical Background

Appendix B

Design Diagrams

Appendix C

Raw Results Output

Appendix D

Code

D.1 File: yourCodeFile.java

```
// This is an example java code file, just for illustration
    purposes
public static void main() {
    System.out.print ("Hello World");
}
```