M5MS10 Machine Learning - Coursework 2

15th March 2019

All code for this report is available at its end.

Question 1

Question 1 (a)

Write down and describe the equations involved in the Bayesian approach to linear regression, which may be used to describe an underlying hidden function of time given noisy data points. You may assume an independent Gaussian prior over each of the weights and independent noise on each data point.

Let $\mathbf{y} = (y_1, y_2, \ldots, y_n)^T$ be a random vector representing the continuous-valued response and $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ be a matrix containing known fixed values $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,d})^T$, each of which represent the d-dimensional feature vector associated with the ith observation. Assume that the response is linearly related to the features and is subject to additive noise ϵ,

$$\mathbf{y} = \mathbf{X}^T\mathbf{w} + \epsilon \tag{1}$$

where $\mathbf{w}, \epsilon$ are random vectors taking on values in $\mathbb{R}^d$ and $\mathbb{R}^n$ respectively. Assume that σ_n and σ_p are known fixed values and that $\epsilon \sim \mathcal{N}_n(0, \sigma_n^2 I_n)$, $\mathbf{w} \sim \mathcal{N}_d(0, \Sigma_p)$.

We perform inference by applying Bayes' theorem:

$$p(\mathbf{w}|\mathbf{y}, \mathbf{X}) \propto p(\mathbf{y}|\mathbf{w}, \mathbf{X})p(\mathbf{w}) \tag{2}$$

where the constant of proportionality is with respect to $\mathbf{w}$. For convenience, let $A = \sigma_n^{-2}\mathbf{X}\mathbf{X}^T + \Sigma_p^{-1}$. Then the posterior distribution of $\mathbf{w}$ can be derived as:

$$\mathcal{N}\big(\sigma_n^{-2}A^{-1}\mathbf{X}\mathbf{y}, A^{-1}\big) \tag{3}$$

Let $\mathbf{x}_*$ be a known, fixed query feature vector for which we wish to predict the expected value of the random variable $f_* = \mathbf{w}^T\mathbf{x}_*$. A well-known property of the multivariate Gaussian distribution is that if $\mathbf{a}$ is a random vector with distribution $\mathcal{N}\big(\mu_a, \Sigma_a\big)$ and $\mathbf{b}$ is a fixed vector of the same length as $\mathbf{a}$, then $\mathbf{b}^T\mathbf{a} \sim \mathcal{N}\big(\mathbf{b}^T\mathbf{a},\ \mathbf{b}^T\Sigma_a\mathbf{b}\big)$. Consequently, the distribution of $f_* = \mathbf{w}^T\mathbf{x}_*$ conditional on $\mathbf{X}$ and $\mathbf{y}$ is

$$f_* \sim \mathcal{N}(\sigma_n^{-2}\mathbf{x}_*^T A^{-1}\mathbf{X}\mathbf{y},\ \mathbf{x}_*^T A^{-1}\mathbf{x}_*) \tag{4}$$

Question 1 (b)

By considering the covariance of the function defined in part (a) at two time points, show how we can derive an equivalent kernel function using an inner product of basis functions. Hence show that the Bayesian approach to linear regression is equivalent to the Gaussian process approach with a suitable choice of kernel function.

An equivalent kernel function can be derived as an inner product of basis functions as follows. Let input $\mathbf{x} \in \mathbb{R}^d$ be mapped to feature space $\mathbb{R}^k$ by the function $\boldsymbol{\phi} : \mathbb{R}^d \to \mathbb{R}^k$. Let $\mathbf{w}$ be a k-dimensional Gaussian random vector with mean zero and covariance matrix Σ_p. The stochastic process $\{f(\mathbf{x})\}_{\mathbf{x}\in\mathbb{R}^d}$ defined by the linear function in Question 1 (a):

$$f(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) = \sum_j w_j \phi_j(\mathbf{x}) \tag{5}$$

is a Gaussian process with covariance function:

$$\begin{aligned}
\mathrm{E}[f(\mathbf{x})f(\mathbf{x}')] &= \mathrm{E}\left[\sum_j w_j\phi_j(\mathbf{x}) \sum_i w_i\phi_i(\mathbf{x}')\right] && (6)\\
&= \mathrm{E}\left[\sum_j \sum_i w_j w_i \phi_j(\mathbf{x})\phi_i(\mathbf{x}')\right] && (7)\\
&= \sum_j \sum_i \phi_j(\mathbf{x})\, \mathrm{E}[w_j w_i] \phi_i(\mathbf{x}') && (8)\\
&= \sum_j \phi_j(\mathbf{x}) \sum_i \Sigma_p^{ji} \phi_i(\mathbf{x}') && (9)\\
&= \sum_j \phi_j(\mathbf{x}) \Sigma_p^{j\cdot} \boldsymbol{\phi}(\mathbf{x}') && (10)\\
&= \boldsymbol{\phi}(\mathbf{x})^T \Sigma_p \boldsymbol{\phi}(\mathbf{x}') && (11)
\end{aligned}$$

where Σ_p^{ji} is row j and column i of Σ_p and $\Sigma_p^{j\cdot}$ is row j of Σ_p. Define $\Sigma_p^{1/2}$ such that $(\Sigma_p^{1/2})^T \Sigma_p^{1/2} = \Sigma_p$[1]. We then have $\mathrm{E}[f(\mathbf{x})f(\mathbf{x}')] = (\Sigma_p^{1/2}\boldsymbol{\phi}(\mathbf{x}))^T \Sigma_p^{1/2}\boldsymbol{\phi}(\mathbf{x}')$, a dot product and therefore an inner product. The kernel function for a Gaussian process defined according to Equation 5 is therefore equivalent to an inner product of basis functions.

We now show that the Bayesian approach to linear regression is equivalent to the Gaussian process approach with a suitable choice of kernel function. Using the notation of Question 1 (a), the Bayesian predictive distribution of f_* conditional on $(y_1, \ldots, y_n)$ is a Gaussian with mean and variance:

$$\begin{aligned}
\mathrm{E}[f_*|\mathbf{y}, \mathbf{X}] &= \sigma_n^{-2} \mathbf{x}_*^T A^{-1} \mathbf{X}\mathbf{y} && (12)\\
\mathrm{Var}(f_*|\mathbf{y}, \mathbf{X}) &= \mathbf{x}_*^T A^{-1} \mathbf{x}_* && (13)
\end{aligned}$$

where $A = \sigma_n^{-2}\mathbf{X}\mathbf{X}^T + \Sigma_p^{-1}$.

Now consider the Gaussian process $\{f(\mathbf{x})\}_{\mathbf{x}\in\mathbb{R}^d}$ with mean $\mathbf{0}$ and covariance function $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\Sigma_p\mathbf{x}'$. The joint distribution of $(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n), f(\mathbf{x}_*))$ is $\mathcal{N}(0, \mathbf{X}_+^T\Sigma_p\mathbf{X}_+)$, where $\mathbf{X}_+ = (\mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{x}_*)$. This implies that the joint distribution of $(y_1 = f(\mathbf{x}_1) + \epsilon_1, \ldots, y_n = f(\mathbf{x}_n) + \epsilon_n, f(\mathbf{x}_*))$, where $\boldsymbol{\epsilon} \sim \mathcal{N}_n(0, \sigma_n^2 I_n)$, is $\mathcal{N}_{n+1}(0,\ \mathbf{X}_+^T\Sigma_p\mathbf{X}_+ + E_+)$, where E_+ is $\sigma_n^2 I_n$ bordered on the bottom and right by zeros. The mean and variance of f_* conditional on $y_1, \ldots, y_n$ follows as:

$$\begin{aligned}
\mathrm{E}[f_*|\mathbf{y}, \mathbf{X}] &= \mathbf{x}_*^T \Sigma_p \mathbf{X} P^{-1} \mathbf{y} && (14)\\
\mathrm{Var}(f_*|\mathbf{y}, \mathbf{X}) &= \mathbf{x}_*^T \Sigma_p \mathbf{x}_* - \mathbf{x}_*^T \Sigma_p \mathbf{X} P^{-1} \mathbf{X}^T \Sigma_p \mathbf{x}_* && (15)
\end{aligned}$$

where $P = (\mathbf{X}^T\Sigma_p\mathbf{X} + \sigma_n^2 I_n)$.

[1] $\Sigma_p^{1/2}$ exists because Σ_p is positive definite by assumption.

To see that the Bayesian posterior predictive mean (Equation 12) and the expected value of f_* in the Gaussian process (Equation 14) are equivalent, observe that:

$$A\Sigma_p\mathbf{X} = \mathbf{X} + \sigma_n^{-2}\mathbf{X}\mathbf{X}^T\Sigma_p\mathbf{X} \tag{16}$$
$$= \sigma_n^{-2}\mathbf{X}(\sigma_n^2 I_n + \mathbf{X}^T\Sigma_p\mathbf{X}) \tag{17}$$
$$= \sigma_n^{-2}\mathbf{X}P \tag{18}$$

which implies that

$$\Sigma_p\mathbf{X}P^{-1} = \sigma_n^{-2}A^{-1}\mathbf{X} \tag{19}$$

Substitute for $\Sigma_p\mathbf{X}P^{-1}$ in Equation 14 to see that:

$$\mathrm{E}[f_*|\mathbf{y},\mathbf{X}] = \mathbf{x}_*^T\Sigma_p\mathbf{X}P^{-1}\mathbf{y} = \sigma_n^{-2}\mathbf{x}_*^T A^{-1}\mathbf{X}\mathbf{y} \tag{20}$$

Applying the Woodbury identity[2] to A^{-1} in Equation 13 reveals that Equations 13 and 15 are equal:

$$A^{-1} = (\Sigma_p^{-1} + \sigma_n^{-2}\mathbf{X}\mathbf{X}^T)^{-1} \tag{21}$$
$$= \Sigma_p - \sigma_n^{-2}\Sigma_p\mathbf{X}(I_n + \sigma_n^{-2}\mathbf{X}^T\Sigma_p\mathbf{X})^{-1}\mathbf{X}^T\Sigma_p \tag{22}$$
$$= \Sigma_p - \Sigma_p\mathbf{X}(\sigma_n^2 I_n + \mathbf{X}^T\Sigma_p\mathbf{X})^{-1}\mathbf{X}^T\Sigma_p \tag{23}$$

Note that the final line exploits the fact that $(a\mathbf{Z})^{-1} = a^{-1}\mathbf{Z}^{-1}$ where $\mathbf{Z}$ is an invertible matrix and a is a non-zero scalar.

The Bayesian approach to linear regression is therefore equivalent to the Gaussian process approach when the kernel function is chosen to be $\mathbf{k}(\mathbf{x},\mathbf{x}') = \mathbf{x}^T\Sigma_p\mathbf{x}'$. This conclusion applies equally well in the feature space associated with mapping ϕ: simply replace the entries of $\mathbf{X}, \mathbf{x}_*$ with their mapped values.

[2] $(\mathbf{X}+\mathbf{Y}\mathbf{Z})^{-1} = \mathbf{X}^{-1} - \mathbf{X}^{-1}\mathbf{Y}(I + \mathbf{Z}\mathbf{X}^{-1}\mathbf{Y})^{-1}\mathbf{Z}\mathbf{X}^{-1}$

Question 1 (c)

Using a Gaussian process, construct three plausible predictive models for describing CO_2 emissions. Compare the models using a quantitative approach and hence rank your models in order of how well they describe the data. Give full details of any assumptions made and equations used when fitting your models to the data and comparing them. In 2013 the concentration of CO_2 topped 400ppm for the first time in human history. Comment on how this observation compares with the predictions made by your models.

We consider a Gaussian process $\{f_x\}_{x\in\mathbb{R}}$. Assume that the mean of a finite subset from the process has zero mean and a covariance matrix defined by the covariance function $k(x_i, x_j)$. Let $\mathbf{Y} = (Y_{x_1}, \ldots, Y_{x_n})$ be a random vector, with $Y_{x_i} = f_{x_i} + \epsilon_i$, where $\epsilon_i \sim \mathcal{N}(0, \sigma_n^2)$. Let f_{x_*} be the element of the process that coresponds to $x = x_*$. The joint distribution of $Y_{x_1}, \ldots, Y_{x_n}, f_{x_*}$ is

$$\mathcal{N}_{n+1}(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_n^2 I_n & \mathbf{k}^T \\ \mathbf{k} & k_* \end{bmatrix}) \tag{24}$$

where $\mathbf{K}$ is a matrix consisting of covariance entries $k(x_i, x_j)$ for $i = 1, \ldots, n$, $j = 1, \ldots, n$, the vector $\mathbf{k}$ is defined by $\mathbf{k}_i = k(x_*, x_i)$, and $k_* = k(x_*, x_*)$.

For a multivariate normal distribution defined by

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_a \\ \mathbf{x}_b \end{bmatrix} \quad \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{bmatrix} \quad \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_a & \boldsymbol{\Sigma}_c \\ \boldsymbol{\Sigma}_c^T & \boldsymbol{\Sigma}_b \end{bmatrix} \tag{25}$$

$\mathbf{x}_b|\mathbf{x}_a$ is Gaussian and has mean and covariance matrix

$$\hat{\boldsymbol{\mu}}_b = \boldsymbol{\mu}_b + \boldsymbol{\Sigma}_c^T \boldsymbol{\Sigma}_a^{-1} (\mathbf{x}_a - \boldsymbol{\mu}_a) \tag{26}$$

$$\hat{\boldsymbol{\Sigma}}_b = \boldsymbol{\Sigma}_b - \boldsymbol{\Sigma}_c^T \boldsymbol{\Sigma}_a^{-1} \boldsymbol{\Sigma}_c \tag{27}$$

Applying this result to condition on $\mathbf{Y} = \mathbf{y}$, we find that:

$$f_{x_*} \sim \mathcal{N}_1(\mathbf{k}(\mathbf{K} + \sigma_n^2 I_n)^{-1}\mathbf{y},\ k_* - \mathbf{k}(\mathbf{K} + \sigma_n^2 I_n)^{-1}\mathbf{k}^T) \tag{28}$$

is the distribution of the response. The prediction that minimizes mean-squared error is the expected value of f_*, $\mathrm{E}[f_{x_*}|\mathbf{Y} = \mathbf{y}] = \mathbf{k}(\mathbf{K} + \sigma_n^2 I_n)^{-1}\mathbf{y}$.

Parameterize our covariance kernel by the vector θ. We select the value of θ by maximizing the probability that $\mathbf{Y} = y$. The observations $\mathbf{Y}$ have distribution $\mathcal{N}_n(\mathbf{0}, \mathbf{K}_\theta + \sigma_n^2 I_n)$, where $\mathbf{K}_\theta$ is the Gram matrix generated by covariance function $k(x, x'; \theta)$. This implies that the observations' log-likelihood is[3]:

$$\log p(\mathbf{y}|\mathbf{X}, \theta) = -\frac{1}{2}\big(n \log 2\pi + \log|\mathbf{K}_\theta + \sigma_n^2 I_n| + \mathbf{y}^T(\mathbf{K}_\theta + \sigma_n^2 I_n)^{-1}\mathbf{y}\big) \tag{29}$$

Where $|\mathbf{K}_\theta + \sigma_n^2 I_n|$ denotes the determinant of $\mathbf{K}_\theta + \sigma_n^2 I_n$. This function is in general non-convex and we will optimize it numerically.

[3]Referred to as the 'marginal-likelihood' in Rasmussen and Williams.

Model 1: Basis function regression

The first model we consider is equivalent to Bayesian linear regression with a finite number of basis functions and a zero-mean spherical Gaussian prior on their weights:

$$f(x) = w_1 + w_2 x + w_3 x^2 + w_4 x^3 + w_5 \sin\left(\frac{2\pi x}{T_0}\right) + w_6 \cos\left(\frac{2\pi x}{T_0}\right) \tag{30}$$

where T_0 corresponds to the period of the oscillations, assumed to be equal to 12 (the data is monthly). The angle sum trigonometric identity can be applied to show that the sinusoidal terms in this expression permit the time-invariant amplitude and phase of a single sinusoid to be learnt.

Using the notation from Question 1 (a), we take $\Sigma_p = \sigma_p^2 I_p$ with $\sigma_p^2 = 1$ and define the covariance function to be

$$k_1(x_i, x_j) = \boldsymbol{\phi}(x_i)^T \Sigma_p \boldsymbol{\phi}(x_j) \tag{31}$$

Since Σ_p is fixed, the only hyperparameter that must be learnt by optimizing the marginal log-likelihood is σ_n^2 using the Nelder-Mead simplex method. The trajectory of the marginal log-likelihood over the course of the optimization is shown in Figure 2. The procedure converges to a local maximum likelihood noise standard deviation of $\hat{\sigma}_n = 0.752$.

Diagnostic plots for the model are shown in Figure 3. We can see that the distribution of the residuals is approximately normal and stationary for most of time period that the data spans. There is no evidence that the model has been mis-specified, apart perhaps from the residual disribution nearby to $x = 1$.

The model's performance was evaluated by fitting it to the data available for 1958 - 1977 (389 datapoints) and computing the empirical mean squared error of its predictions for 1978 - 1983. The predictions are shown in Figure 1, along with a credible interval for f_* (note that the observations are values of y_*). The model had a test-set mean-squared error of approximtely 4.95.

The expected value and 95% highest density interval of the process at each x-value when conditioning on the entire dataset is shown in Figure 4. These values will be discussed at the end of this section of the report.

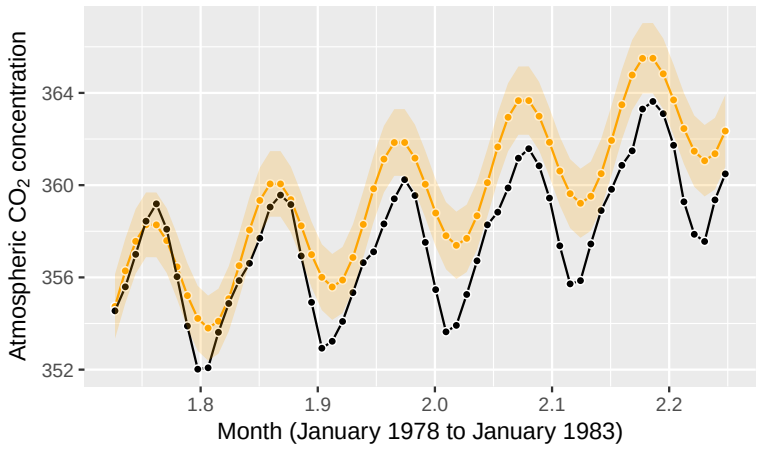

Figure 1: Test-set values (black points) versus model predictions (yellow points) for Model 1. The shaded interval corresponds to the 95% highest posterior density interval for f_x.

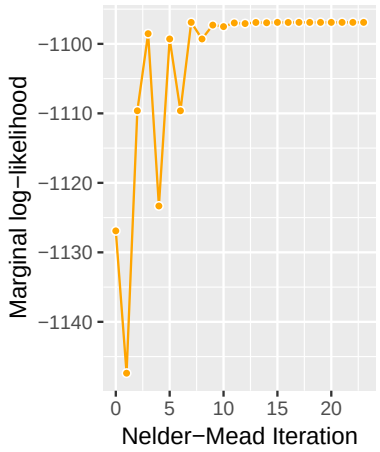

Figure 2: Trajectory of the marginal log-likelihood for Nelder-Mead simplex optimization procedure for Model 1.

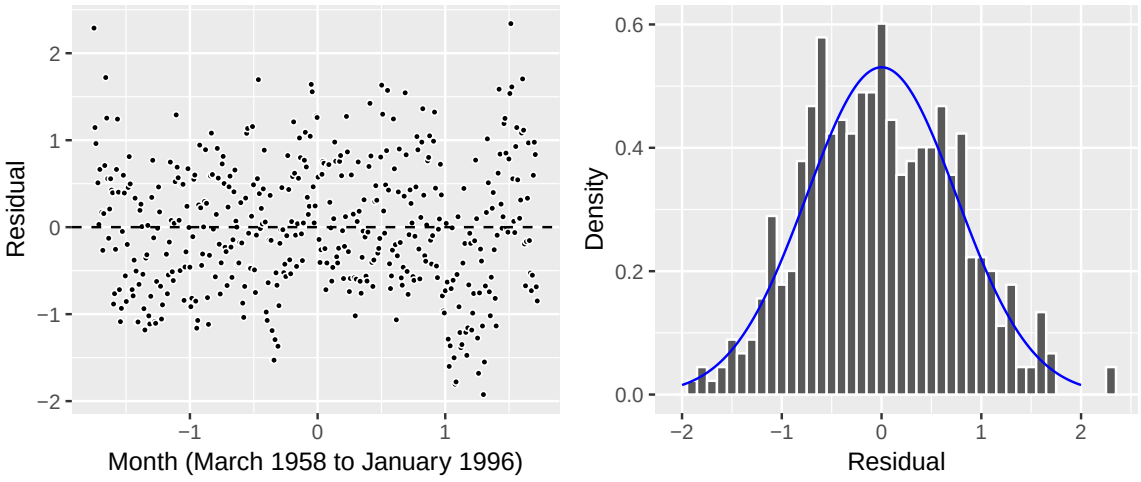

Figure 3: (Left) Plot of residuals $\mathrm{E}[f_{x_i}|\mathbf{y}] - y_{x_i}$ against x_i. (Right) Distribution of residuals relative to the density of $\mathcal{N}(0, \hat{\sigma}_n^2)$.

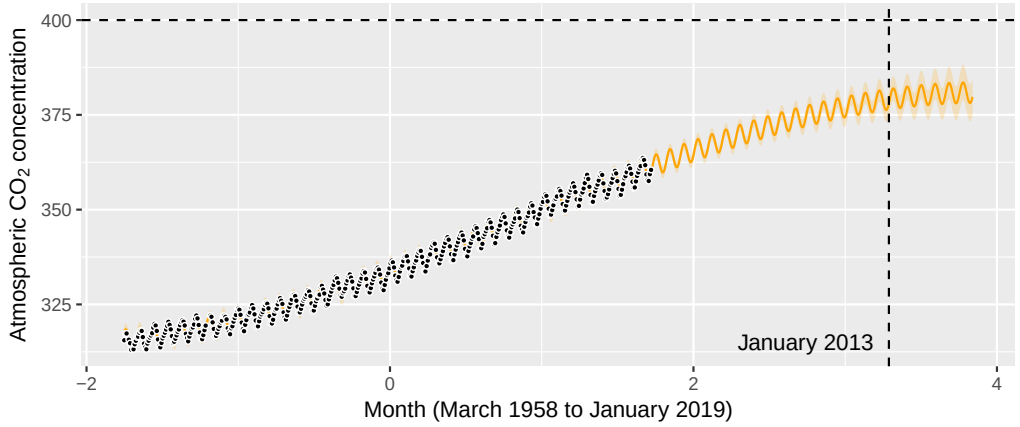

Figure 4: Predictions of Model 1 up until 2019. Shaded interval is a 95% highest density interval for f_x. Solid yellow line is $\mathrm{E}[f_*|\mathbf{y}]$. Black points correspond to observations $\mathbf{y} = (y_{x_1}, \ldots, y_{x_n})$.

Model 2: Periodic Gaussian process

The second model we consider captures the periodic component of the response without using explicit basis functions and expresses the fact that there is likely to be a smoothly varying long-timescale trend. This model has covariance function:

$$k_2(x_i, x_j) = \theta_1^2 \exp\left(\frac{-(x_i - x_j)^2}{\theta_2^2}\right) + \theta_3^2 \exp\left(-\frac{\sin^2(\pi|x_i - x_j|/T_0)}{\theta_4^2}\right) \tag{32}$$

Where θ_1, θ_2, θ_3, θ_4 are kernel parameters. The squared exponential term expresses that the response should be smooth, whereas the sinusoidal term captures that the signal has a periodic component with period T_0. The constituent kernels are added together so that local and periodic covariation superimpose to yield the signal, as opposed to amplifying or attuning one another. Unlike Model 1, the feature spaces corresponding to each term in this covariance function are not finite-dimensional, so it is not possible to write down the response as a function of the inputs in closed-form, as is done in Equation 30.

A model using the SE kernel alone was fit to obtain initial values for θ_1, θ_2 when fitting the full model. The initial values for θ_3, θ_4 were arbitrarily set to 10. The trajectory of the marginal log-likelihood over the course of optimizing the kernel parameters is shown in Figure 7. We can see that the procedure is unstable initially, before settling in a local optimum. The close fit of the model to the data

suggested that the optimization procedure's result was adequate, if not necessarily globally optimal. The parameter values at convergence were $\theta_1 = 267.2, \theta_2 = 6.40, \theta_3 = 32.13, \theta_4 = 3.34, \sigma = 0.47$.

Fitting the model to the data available for 1958 - 1977 (389 datapoints) and computing predictions for 1978 - 1983 yielded a test-set empirical MSE of 3.51, a marginal improvement over the 4.95 that Model 1 achieved. Inspecting the process's predictions, shown in Figure 5, and comparing them with the predictions of Model 1 in Figure 1, we can see that the periodic kernel captures periodic variation that is non-sinusoidal. This suggests that the difference in empirical MSE is representative of the fact that Model 2 is an improvement over Model 1. Aside from this, the predictions for the five-year period seem to be sufficiently good for the model to be practically useful.

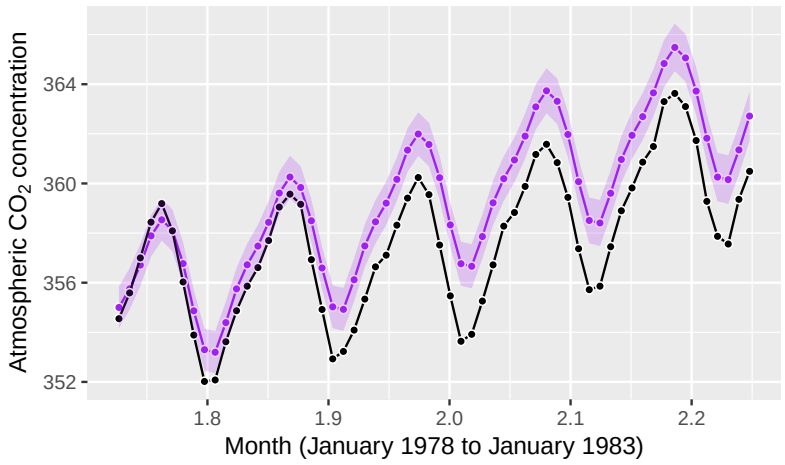

Figure 5: Test-set values (black points) versus model predictions (purple points). The shaded interval corresponds to the 95% highest posterior density interval for f_x.

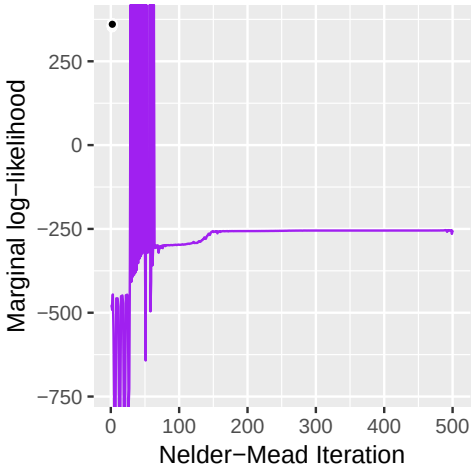

Figure 6: Model 2 log-likelihood trajectory for Nelder-Mead simplex optimization procedure.

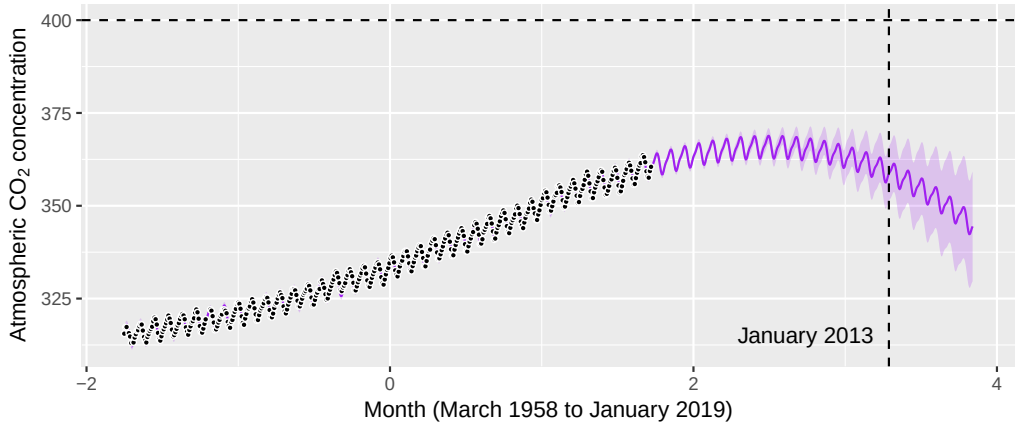

Figure 7: Predictions of Model 2 up until 2019. Shaded interval is a 95% highest density interval for f_x. Solid purple line is $\mathrm{E}[f_*|\mathbf{y}]$. Black points correspond to observations $\mathbf{y} = (y_{x_1}, \ldots, y_{x_n})$. The increasing width of the highest density interval is attributable to the squared exponential kernel.

Model 3: Linear periodic Gaussian process

A problem with Model 2 is that it fails to capture a linear trend in the CO_2 concentration as we move further into the future. If the physical processes driving the upward trend continue to exist (i.e. we have a 'business as usual' scenario), then this trend will persist. We introduce a linear kernel to capture this linearity in our Gaussian process. The overall kernel function is then:

$$k_3(x_i, x_j) = (\theta_1^2 + \theta_2^2 x_i x_j) + \theta_3^2 \exp\left(- \frac{\sin^2(\pi|x_i - x_j|/T_0)}{\theta_4^2} \right) + \theta_5^2 \exp\left(\frac{-(x_i - x_j)^2}{\theta_6^2} \right) \tag{33}$$

The linear component of the kernel corresponds to simple linear regression in input space. The periodic component is included to capture seasonal variation in atmospheric CO_2 concentration. The squared exponential term permits single-timescale local variation that is not consistent with the overall linear trend. An important drawback of this model is that it has more parameters than Models 1 and 2, which makes locating the maximum likelihood values more computationally expensive and technically challenging. This cost is justified by the model's increased expressiveness.

Figures 8, 9, and 10 present the test-set predictions, marginal log-likelihood trajectory, and 1983 - 2019 predictions respectively. The predictions shown are for a model with parameter values $\theta_1 = 115.56$, $\theta_2 = 5.91$, $\theta_3 = -18.05$, $\theta_4 = 1.79$, $\theta_5 = 2.18$, $\theta_6 = -0.17$, $\sigma_n = 0.33$. The test-set empirical MSE of the model is 3.40, slightly below the 3.51 of Model 2. This difference is sufficiently small that it is not possible to claim one model is preferable to the other for five-year forecasting. Over longer timescales, however, Model 3 is superior on account of the fact that it preserves the historical linear trend.

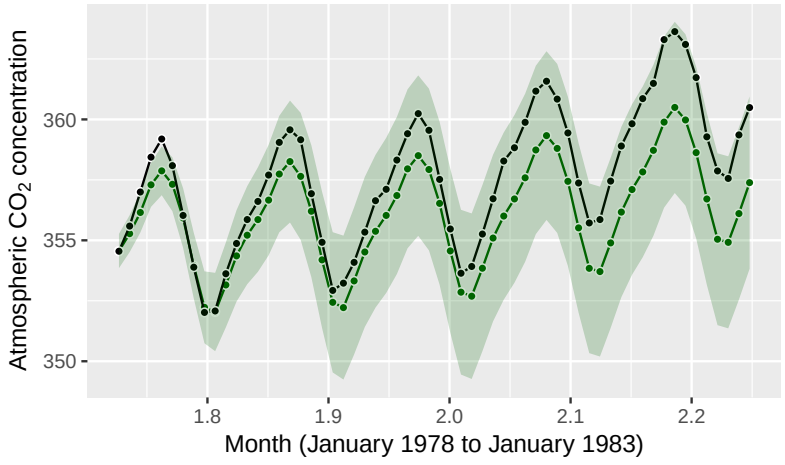

Figure 8: Test-set values (black points) versus model predictions (green points). The shaded interval corresponds to the 95% highest posterior density interval for f_x (not $y_x = f_x + \epsilon$).

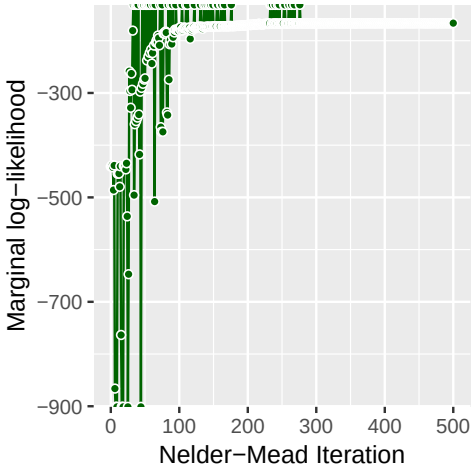

Figure 9: Trajectory of the marginal log-likelihood for Nelder-Mead simplex optimization procedure for Model 3.

In 2013 the concentration of CO2 topped 400 ppm for the first time in human history. Comment on how this observation compares with the predictions made by your models?

Models 1 and 3 are most consistent with this observation, with Model 3 marginally more so. Both models substantially under-predict however, returning predictions of $\approx$ 380ppm. The 95% highest density intervals for the predictions do not encompass 400ppm either, indicating that we would view this outcome as highly improbable if we were talking in 1983. Given the costs associated with under-predicting an increase in CO_2 concentration, this situation is highly undesirable. Referring to Figures 1, 5, and 8, we can see that the test-set performance of each model degrades as we moved further from the training data. This trend could have been evaluated over a longer timescale by using a more balanced train/test split. The information gained from this analysis may have allowed us to anticipate that the CO_2 concentration would surpass 400ppm in 2013.

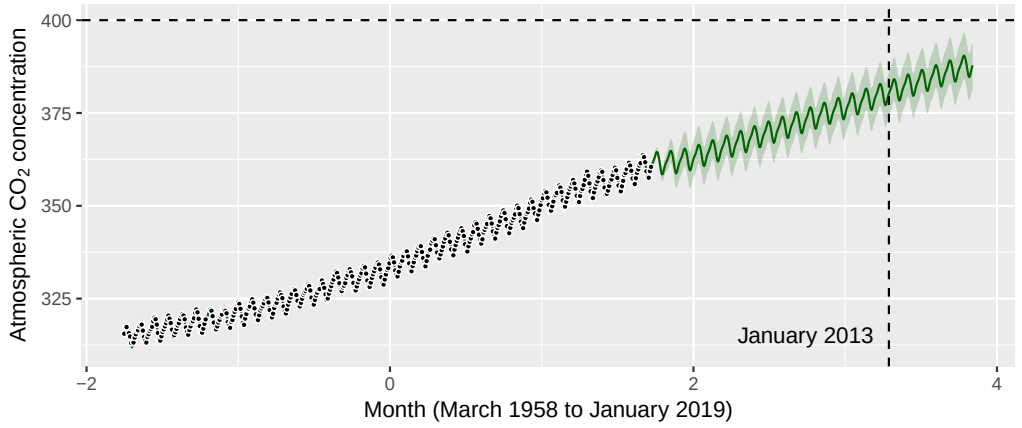

Figure 10: Predictions of Model 3 up until 2019. Shaded interval is a 95% highest density interval for f_x. Solid green line is $\mathrm{E}[f_*|\mathbf{y}]$. Black points correspond to observations $\mathbf{y} = (y_{x_1}, \ldots, y_{x_n})$.

Question 2

Question 2 (a)

On Blackboard you will find a paper in which two researchers have developed an automated inference method for discriminating between criminals and non-criminals based solely on still face images. Do you think this is a good research paper? Back up your opinion with 5 bullet points, briefly describing either positive or negative aspects of how the research was conducted. (8 marks)

I think this is not a good research paper

1. The purpose of the research is unclear and the paper's mode of publication is not appropriate to its claims and subject
    - The social consequences of successfully predicting criminality would be tremendous. The authors of the paper do not seem to appreciate this. They do not discuss the ethics of their work and suggest that their curiosity is sufficient reason to broach the topic of inferring criminality from appearance. From the paper's introduction,

      > *"... we want to satisfy our curiosity in the accuracy of fully automated inference on criminality."*

    - This paper could have been extremely controversial. The authors should have chosen to try to make their work available by way of a credible, peer-reviewed scientific journal, as opposed to arXiv[4].
    - The technology presented in the paper is useless for practical purposes (e.g. risk scoring of people in real-time CCTV). This is because criminals make up a very small proportion of the population and the model is insufficiently specific as a diagnostic tool. This reality is addressed by the authors in their response to peer criticism.
2. The experimental methodology is flawed and the analysis is not replicable.
    - The data-set's characteristics and provenance are not described and scrutinized sufficiently to justify the conclusions presented. The sampling mechanism used to obtain the face images is different for criminals and non-criminals. The web scraper retrieves images from an unspecified location according to an algorithm that is not described. These images could have been self-uploaded. The face images provided by the 'city police department in China',

[4]arXiv is a research repository that is popular among machine learning researchers and practitioners. Submitting a paper to arXiv for moderation only requires endorsement by an arXiv-approved academic: moderation itself may be partially automated[5]. This means that low-quality papers can be accepted.

a department that had full knowledge of the research's purpose, may have been selected on the basis that their subjects looked especially criminal. Ethnic variation within China may have caused differences in facial structure between the classes.

- The data-set is confidential and cannot be accessed by other researchers.
- The authors cite research indicating that people can achieve minor success in discriminating criminals from non-criminals, but do not attempt to replicate this result on their own data-set. A preliminary study in which participants were trained on a set of labelled images and then tested on their ability to classify images in an unseen test set would have provided a compelling reference point for evaluating the success of the machine learning models in the paper.

3. The authors reason in terms of ill-defined concepts and do not explore the statistical properties of their experiments formally.
   - The authors' conjecture is that facial structure is predictive of criminality, but they do not define what facial structure nor how it is retrieved from an image (for example, how does facial structure differ from facial expression?).
   - Differences between the values and empiricial distributions of statistics computed on either class are assumed to be linked to criminality as opposed to sampling variation.
     - The authors do not verify that the ρ, θ, d statistics they identify as discriminating between the classes are what permit the classifiers' to accurately classify their queries.
     - The Hellinger distance's sampling distribution under the null hypothesis of class-independence is not mentioned when evaluating the significance of its observed value.
     - The values of the Isomap distances D_x, D_c, D_n are unmentioned and are not subjected to e.g. a bootstrap estimate of their variability.
4. The authors make several faulty generalizations and conclusions.
   - They conclude that the classifier's excellent performance is attributable to differences in face structure rather than other possible differences between the images, of which there are many that are not eliminated by the experiments and analyses contained in this paper[6].
   - The authors consistently conflate criminality with having a criminal conviction.
5. The authors present experimental findings that are irrelevant.
   - Demonstrating that a single fit of a classifier on several sets of face images with randomly assigned labels performs approximately in line with the base rate of the majority class. Only the Chinese men data-set should have been repeatedly scrambled and classified, as opposed to introducing other datasets. The resulting distribution over classification accuracies could then be used to evaluate how plausible it is that the exceptional classification performance was attributable to sampling variability.
   - It seemed unnecessary to discuss the performance of the various different classifiers and their respective features if the properties of these were not going to be used later in the analysis[7].

I felt the research had positive aspects, but only its shortcomings are highlighted here.

[6] The fact that so many differences must be considered is linked to the fact the input feature space has many dimensions ($80 \times 80 = 6400$)

[7] On a related note, it is surprising that the authors did not consider whether the simpler classifiers they evaluated can approximate statistics ρ, θ, d from the features made available to them.

Question 2 (b)

Write up to 500 words discussing some of the possible consequences on society of wide-spread application of AI and machine learning, and how these technologies might impact our lives, both positively and negatively. Feel free to refer to any statistical approaches, methodologies, and ideas presented in the lectures (12 marks).

The current impact of machine learning (ML) provides information about how it may affect society in the future[8]. One way to perceive the effects of existing ML technologies is to consider how your life would change if the British government ruled that ML were prohibited in the UK[9].

Under prohibition, computational prediction would become illegal. This would make the labour of Tarot card readers more valuable: your visits to them more expensive. It would also mean that the recommendations you receive for articles to read, music to listen to, films to watch, events to attend, and people to meet would become less aligned to your personal preferences. You would be less willing to seek out recommendations[10] and would therefore receive them less frequently. The volume of media you consume would drop and the media you did consume would be less specific to you. The former outcome would apply downward pressure on the number of experiences you have in common with other people, whereas the latter outcome would apply an upward pressure, since generic recommendations would make it more likely you had experienced the same media as someone you meet. Browsing and searching, which you would have to do more of in the absence of prediction, would become more laborious because unsupervised learning could no longer be used to group similar media automatically. The broader impact of all of this would be to reduce rate of demand for entertainment, meaning that new music, films, articles, events, and so on would be released less frequently, causing the rate of creative innovation to decrease[11].

Data is valuable both because of what it contains explicitly and what can be inferred from it. Banning prediction would make computational inference illegal, making data less valuable. If data were to become less valuable, you would probably find that fewer companies attempted to obtain it[12]. This would benefit your privacy and personal security, albeit at the expense of your convenience, pleasure, and the relevance of your ad recommendations. The web and mobile phones also derive their value, in part, from machine learning: ML makes these platforms useful, usable, and pleasurable to use, but also enables them to optimize your behaviours on behalf of parties that you may not be aware of, such as corporations and political groups. Outlawing machine learning would make it less likely that you are unknowingly exploited.

Widespread adoption of machine learning technologies would result in the effects described above becoming more extreme and new, similar ones being observed in other domains of society. Data will become more valuable as better techniques for using it become available. This will make companies decide to capture more of it, motivating advances in machine learning, making data more valuable still, perpetuating the cycle. Every organization's success is contingent on some form of optimization, which suggests that ML is relevant to every organization. The societal impact of machine learning is already far-reaching and will almost certainly reach even further in the future.

[8] In statistical terms, we could say that we use historical data to predict future events.

[9] This discussion focuses on the societal impact of machine learning at the level of the individual.

[10] e.g. by using Spotify, Netflix, or Amazon.

[11] Banning machine learning would mean that the promise that generative models hold for computer-generated media would go unfulfilled.

[12] 'Personal data' is data generated by your activities, including your personal information, browsing habits, the items you own, who your friends are, what your preferences are, and whether you own pets, among other things.

Question 3

Choose one machine learning approach to create a predictive model for binary classification of future Bitcoin prices i.e. whether the price will rise or fall. You may for example use a model we have covered in class, or one of their extensions. Think very carefully about how you should define, train, and test your model.

The classifier was derived based on the following trading strategy. Assume that the price trajectory within an hour is a continuous path and that real-time data is available to trigger a buy order. Let sequences $(o_t)_{t\in T}$, $(l_t)_{t\in T}$, $(c_t)_{t\in T}$, contain hour opening, low, and closing prices respectively, where T is the set of hours that we trade within. Assume that for each hour we buy Bitcoin ($b_t = 1$) if we see the price drop below a fraction z of its opening value, for a price of zo_t (i.e. we buy if $l_t \leq zo_t$). If the price remains above zo_t over the hour, then no purchase is made ($b_t = 0$). If a purchase is made, we sell our unit of Bitcoin at the hour's closing price c_t. The payoff of this trading strategy is:

$$\mathcal{P}(z) = \sum_{t\in T} b_t(c_t - zo_t) \tag{34}$$

A preliminary study, the results of which are shown in Figure 11, revealed that it was possible to achieve a positive payoff using the sequences contained in the data-set and an appropriate z-value. It suggested that a classifier might be successful in predicting whether c_t was greater or less than zo_t, given that price drops to at least zo_t within the hour. The parameter z could then be tuned so that the classifier maximized some objective function related to payoff.

Predicting price changes directly based on historical data seemed unlikely to fruitful. Exploratory data analysis suggested that the direction of future price changes tended to be independent of past feature values. Nonlinear time series models are a common tool used by people attempting to predict and trade on Bitcoin's price[13]. Some of these people have access to more information than is available in this data-set and more computing power than is available to a humble MSc Statistics student. Assuming that a profitable model could be found, this model would be identified by many traders and Bitcoin's price would adjust accordingly. Based on this reasoning, other problem structures were explored.

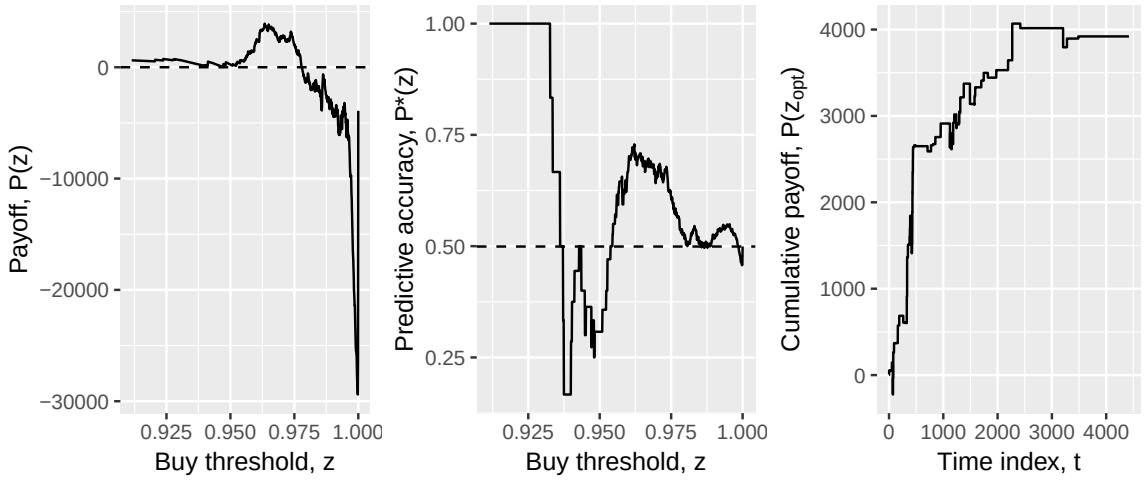

Figure 11: (Left) Payoff $\mathcal{P}(z)$ of the threshold-based trading strategy when applied to the Bitcoin data-set. $z = z_{opt} \approx 0.963$ maximized $\mathcal{P}(z)$ and resulted in 70 trades ($\mathcal{P}(z_{opt}) = 3920$. (Centre) Predictive accuracy of the buy signal as a function of z. Under z_{opt}, 67.4% of price movements for which the buy signal $l_t \leq zo_t$ was triggered were positive. (Right) Cumulative payoff $\mathcal{P}(z)$ of the threshold-based trading strategy over the dataset as a function of time index.

[13]In fact, there is an entire section of the Stock Market Prediction Wikipedia page that discusses how neural networks can be used to predict prices - `https://en.wikipedia.org/wiki/Stock_market_prediction`

Describe in detail how to generate labels for the data and set up the data for your model (6 marks).

For a given $z \in (0,1)$, we seek to predict the value of $y_t = I(c_t \geq zo_t)$ given binary feature $x_t = I(l_t \leq zo_t)$. The problem is treated as a sequential classification problem, meaning that learning and prediction both take place on-line.

We generate labels $y_t = I(c_t - zo_t \geq 0)$ and features $x_t = I(l_t \leq zo_t)$ for our model.

```
z <- 0.99 # for example
y <- as.numeric(Data$Close >= z*Data$Open) # labels
x <- as.numeric(Data$Low <= z*Data$Open) # features
```

Describe how your model is defined and applied (6 marks). Describe what inference method you are using, including any necessary derivations (6 marks).

Distinguish between two components of the classifier: its probability model and its classification rule.

Start with the probability model. Assume that $(O_t, L_t, C_t)_{t\in T}$ is a sequence of random vectors taking on values in the positive orthant. This sequence of random vectors need not be independent nor identically distributed[14]. Set $Y_t = I(C_t \geq zO_t)$ and $X_t = I(L_t \leq zO_t)$. By plotting c_t/o_t against l_t/o_t for t in T_i where T_i form a chronological partition of $\mathcal{T}$, the set of times in the data-set, it is possible to verify that $(L_t/O_t, C_t/O_t)_{t\in T}$ is plausibly modeled as a sequence of independent and identically distributed random vectors[15]. Under this assumption $(X_t, Y_t)_{t\in T}$ is also a sequence of i.i.d. random vectors.

The probability model is defined in terms of the distribution of (X_t, Y_t), the support of which is $\{(1,0),(1,1),(0,1)\}$. $(0,0)$ is not part of the support because $L_t > zO_t$ implies that $C_t > zO_t$ by virtue of the fact that $C_t \geq L_t$ with probability 1. Define a probability mass function over this support in terms of parameters:

$$\theta_0 = P(Y_t = 1 | X_t = 1) \tag{35}$$
$$\theta_1 = P(X_t = 0) \implies P(Y_t = 1, X_t = 0) = \theta_1 \tag{36}$$

Assume that these parameters are random variables with prior distributions:

$$\theta_0 \sim \text{Beta}(\alpha_0, \beta_0) \tag{37}$$
$$\theta_1 \sim \text{Beta}(\alpha_1, \beta_1) \tag{38}$$

θ_0 corresponds to the probability that $C_t > zO_t$, given that the threshold is breached. θ_1 is the probability that the threshold is not breached.

We apply Bayes' theorem upon receiving the observation (x_t, y_t) to obtain the posterior distribution for θ_0 and θ_1:

$$P(\theta_0, \theta_1 | X_t, Y_t) \propto P(X_t, Y_t | \theta_0, \theta_1) P(\theta_0, \theta_1) \tag{39}$$

where:

$$P(X_t, Y_t | \theta_0, \theta_1) = P(Y_t | X_t, \theta_0, \theta_1) P(X_t | \theta_0, \theta_1) \tag{40}$$
$$= \theta_0^{Y_t X_t} (1-\theta_0)^{(1-Y_t)X_t} \theta_1^{1-X_t} (1-\theta_1)^{X_t} \tag{41}$$

for values of (X_t, Y_t) in the support. The posterior distribution can be derived as a product of Beta distributions with parameter values:

$$\alpha_0' = \alpha_0 + y_t x_t \tag{42}$$
$$\beta_0' = \beta_0 + (1-y_t) x_t \tag{43}$$
$$\alpha_1' = \alpha_1 + (1-x_t) \tag{44}$$
$$\beta_1' = \beta_1 + x_t \tag{45}$$

[14] Their observed values in this data-set would certainly not support an assertion that they were.
[15] In fact the L_t/O_t are autocorrelated, but only weakly.

We update these parameters on-line as each observation arrives. This process constitutes our learning procedure.

The classification rule used by the classifier was chosen by reasoning about the probability of misclassification in the context of perfect knowledge about the data-generating distribution.

The classification rule is a function $d : \{0,1\} \mapsto \{0,1\}$. Only four rules are possible[16]: they are shown in Table 1. The probability a classifier misclassifies a query sampled from the joint distribution of (X_t, Y_t) is displayed in Table 2. We can see that d_2 is necessarily better than d_1 and that d_4 is necessarily better than d_3. d_2 is better than d_4 if:

$$\theta_0(1-\theta_1) < 1 - \big(\theta_0(1-\theta_1) + \theta_1\big) \tag{46}$$

$$: \theta_0 < \frac{1}{2} \tag{47}$$

This ordering of classification rules is only relevant if we seek to minimize the misclassification rate across all queries. If we minimized the misclassification rate given $X_t = 1$, referred to as the 'conditional misclassification rate', $P(d(X_t) \neq Y_t | X_t = 1)$, then we would obtain a classifier that minimized the probability of making an incorrect prediction when the threshold was breached. This is what we will do. The optimal classifier d_* with respect to conditional misclassification rate in this situation is a function of $\theta_0 = P(Y_t = 1 | X_t = 1)$ only:

$$d_*(\theta_0) = \begin{cases} 1 \text{ if } \theta_0 > 0.5 \\ 0 \text{ otherwise} \end{cases} \tag{48}$$

In our probability model θ_0 this quantity is a random variable, not a fixed value. We therefore evaluate the probability that $\theta_0 > 0.5$ conditional on all observations available until the present time. Letting $\alpha_{0,t}, \beta_{0,t}$ denote the posterior parameters for θ_0 after receiving all observations up to time $t-1$, we can compute this probability by integrating $\text{Beta}(\alpha_{0,t}, \beta_{0,t})$ between 0.5 and 1:

$$P(\theta_0 > 0.5 | \alpha_{0,t}, \beta_{0,t}) = 1 - F_{\text{Beta}(\alpha_{0,t}, \beta_{0,t})}(0.5) \tag{49}$$

The classification rule that minimizes the probability of misclassifying Y_t conditional on $X_t = 1$ is then the function:

$$d_*(\alpha_{0,t}, \beta_{0,t}) = \begin{cases} 1 \text{ if } P(\theta_0 > 0.5 | \alpha_{0,t}, \beta_{0,t}) > 0.5 \\ 0 \text{ otherwise} \end{cases} \tag{50}$$

Notice that this classifier is a function of all previous observations (x_t, y_t) and θ_0's prior parameter values $\alpha_{0,0}, \beta_{0,0}$. The probability that a prediction is correct is simply $P(Y_t = 1 | X_t = 1, \alpha_{0,t}, \beta_{0,t}) = \text{E}[\theta_0 | \alpha_{0,t}, \beta_{0,t}] = \frac{\alpha_{0,t}}{\alpha_{0,t}, \beta_{0,t}}$ when 1 is predicted, and $1 - \frac{\alpha_{0,t}}{\alpha_{0,t}, \beta_{0,t}}$ when 0 is predicted.

To summarize, the classifier works as follows. Initialize a threshold z and prior parameters $\alpha_{0,0},\ \beta_{1,0}$. Receive (o_t, l_t). Make a prediction using feature $x_t = I(l_t \leq z o_t)$, evaluate $d_*(\alpha_{0,t}, \beta_{0,t})$. Receive c_t and compute $y_t = I(c_t > z o_t)$. Calculate posterior parameter values $\alpha_{0,t+1}, \beta_{0,t+1}$ according to Equations 42 and 43.

[16] We ignore non-deterministic classification rules.

Table 1: The four $d : \{0,1\} \mapsto \{0,1\}$ decision rules.

	d_1	d_2	d_3	d_4
$x_t = 0$	1	1	0	0
$x_t = 1$	1	0	1	0

Table 2: Bayes error for each d_i.

i	$P(d_i(X_t) \neq Y_t)$
1	$\theta_0(1-\theta_1) + \theta_1$
2	$\theta_0(1-\theta_1)$
3	$1 - \theta_0(1-\theta_1)$
4	$1 - \big(\theta_0(1-\theta_1) + \theta_1\big)$

Discuss the problems you face and highlight the model's potential advantages and disadvantages (6 marks).

The key problem is that our assumption of independent random vectors (X_t, Y_t) may not be consistent with the data (which could exhibit serial dependence), and it is possible that these random vectors are not identically distributed (i.e. the process is non-stationary). These are common obstacles encountered in time series. If our i.i.d. assumption is violated, then the classifiers described above are not optimal. Furthermore, our choice of classifier rests on the assumption that our probability model is well-specified: if it is not, then we could perform very poorly indeed. A non-Bayesian approach would escape the difficulties of reasoning about an appropriate probabilistic description of the process, but might pass up an optimal model in the process.

The classifier's key advantages are its simplicity, which make it both amenable to analysis and straightfoward to implement, its low computation and storage requirements, and guarantees of optimality (provided modeling assumptions are satisfied). Separating our probability model from the classification rule means that we can simulate from and explore the properties of the data-generating process. One neat aspect of the model is that many thresholds z can be maintained, allowing us to search over the space of classification problems (i.e. distributions for (X_t, Y_t)) for the one with an associated optimal classifier that is most likely to classify a query correctly.

Disadvantages of the classifier are its vulnerability to probability model mis-specification, discussed above, and the fact that it does nothing to explore the possibility that there is dependency structure with respect to information at previous time-steps and it does not search over a very large space of candidate features. We also do do not model the distribution of $C_t - zO_t$, meaning that it is not possible to optimize the classifier with respect to trade payoff, which would be desirable in practice.

Discuss the accuracy of the results you obtain (4 marks).

The classifier's performance was evaluated across a range of z-values. Figure 12 (left) presents the $x_t =$ 1-conditional misclassification rate of d_* for each z-value when the underlying probability distribution is updated sequentially, as evaluated onthe original ordering of the data-set. Very few queries are included in this rate for low values of z, whereas for $z = 1$ the entire data-set is included. The rate for low values of z would be likely to shift towards 0.5 if we were to receive more data. For $z \approx 0.963$, 70 queries are classified to yield a $x_t = 1$-conditional misclassifcation rate of ≈ 0.3.

The performance of the $z = 0.963$ classifier was evaluated further by bootstrapping the dataset's entries (4000 bootstrap samples were used) to obtain a distribution over $x_t = 1$-conditional misclassification rates. This bootstrap procedure is valid provided that the original data-set's entries were not sequentially dependent. The resulting bootstrap distribution is shown in Figure 12 (right). We can see that the classifier's performance is highly dependent on the sequence in which observations are presented to it and that its performance is symmetrically distributed about the data-set's base rate.

The distribution of θ_0 conditional upon all entries in the data-set is shown for each value of z in Figure 13 (left). Taking a Bayesian approach to selecting the value of z that maximizes prospective $x_t = 1$-conditional classification accuracy is possible from the plot in this figure. For small z fewer queries are informative about θ_0's value and there is greater uncertainty regarding it. For larger values of z, more queries are classified and there is less uncertainty. This plot suggests that a value of $z = 0.963$ is the buy threshold for which the classifier would perform best.

The final plot, shown in Figure 13, is included to demonstrate that the value of the chosen modelling approach lies in the conditional distribution of Y_t given $X_t = 1$. The marginal distribution of Y_t is irrelevant, since we are only interested in the hours for which the buy signal ($l_t \leq zo_t$) is triggered.

Comment on whether you would use this to actually trade? If not, why not? (2 marks)

I would not use this classifier to trade, since I am not convinced that the distribution of (X_t, Y_t) is stationary and have not evaluated the model on a sufficiently large data set to feel confident that its performance is robust. By adjusting the underlying probability model to account for possible non-

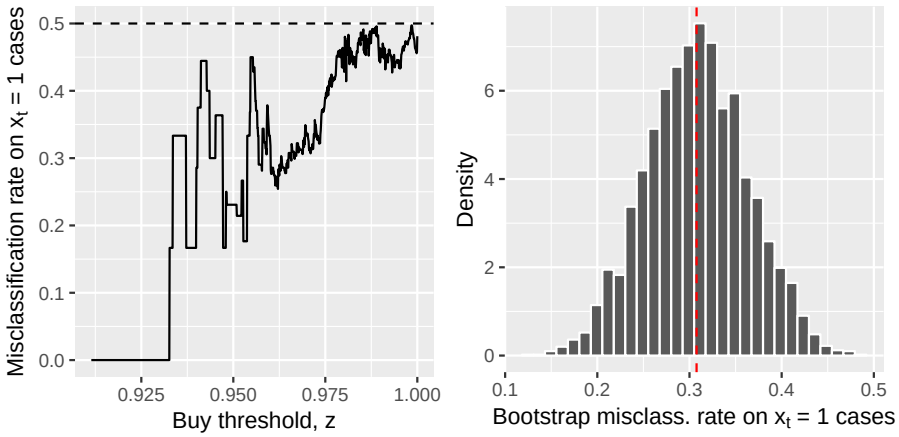

Figure 12: (Left) Sequential misclassification rate (on cases with $x_t = 1$) for d_* with specified z-value when evaluated on original dataset's order. (Right) Distribution of misclassification rate on cases with $x_t = 1$ when the $z = 0.963$ classifier is evaluated against bootstrapped datatsets. The red dashed line corresponds to this classifier's misclassification rate on the original data-set's order.

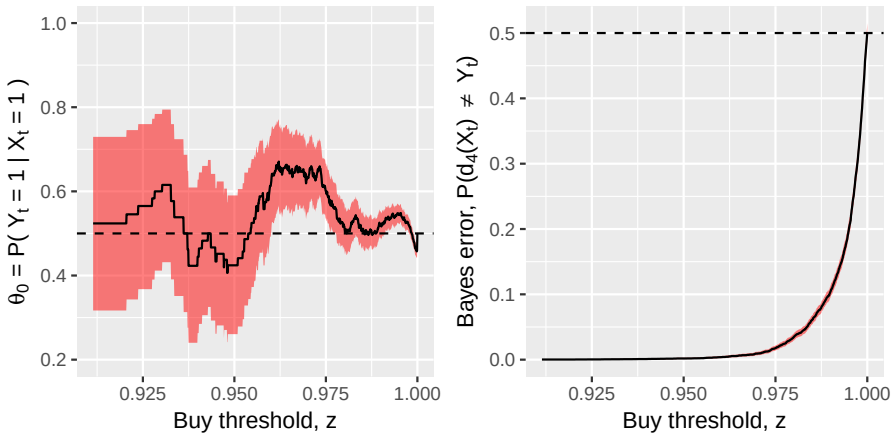

Figure 13: (Left) Posterior mean (black line), 95% highest density interval (red), prior parameters $\alpha_{0,0} = 10, \beta_{0,0} = 10$ for the beta distribution when updated using all observations in the data-set. (Right) Expected value and 95% highest density interval of Bayes error for classification rule $P(d_3(X_t) \neq Y_t)$ after updating on all observations in the data-set.

stationarity and evaluating the payoff of several trading strategies based on the classifier, the classifier might become more usable. Obviously if the classifier were to be used for trading the classification rule should be optimized in a way that takes account of transaction costs and the payoff $c_t - zo_t$ when a buy order is made.

Code

(Question 1) The following code implements Model 1.

```
require(MASS)
rm(list=ls())

Data <- read.csv('CO2_Mauna_Loa_Data.csv')
colnames(Data) <- c('X', 'Y') # MONTHS, CO2_ppm

# Specify input data
x <- (Data$X - mean(Data$X))/sd(Data$X) # x_1, ..., x_n
# 1958 to 1983
x2019 <- ((2019 - 1960)*12 - mean(Data$X))/sd(Data$X)
x2013 <- ((2013 - 1960)*12 - mean(Data$X))/sd(Data$X)
y <- Data$Y # Y_{x_1}, ..., Y_{x_n}
n <- length(x) # No. of observations

# Define covariance function and noise level
sigma2_p <- 100
Sigma_p <- diag(rep(sigma2_p, length(x)))
T0 <- 12/sd(Data$X)

cov_k <- function(x, x_){
  phi_x <- c(1, x, x^2, x^3, sin(2*pi*x/T0), cos(2*pi*x/T0))
  phi_x_ <- c(1, x_, x_^2, x_^3, sin(2*pi*x_/T0), cos(2*pi*x_/T0))
  return( (sigma2_p) * phi_x %*% phi_x_ )
}
K <- sapply(x, function(x1) sapply(x, function(x2) cov_k(x1, x2)))

# Optimize for the noise variance
log_likelihood_path <- c()
log_likelihood <- function(z) {
  sigma2_nI <- diag(rep(z^2, n))
  L <- -0.5*( n*log(2*pi) + log(det(K + sigma2_nI)) +
                t(y) %*% solve(K + sigma2_nI) %*% y )
  log_likelihood_path <<- c(log_likelihood_path, L)
  return(L)
}

sigma_n0 <- 1 # Noise initial value
optim_result <- optim(par=sigma_n0, fn=function(sigma_n0) -2*log_likelihood(sigma_n0))
sigma_nhat <- optim_result$par

GP_predict <- function(xstar, K_inv) {
  # K_inv = (Ktheta + sigma2_n In)^{-1}
  # x is observation covariate vector
  # xstar is query point
  k     <- sapply(X=x, function(x1) cov_k(xstar, x1))
  EY_xstar   <- k %*% K_inv %*% y
  VarY_xstar <- cov_k(xstar, xstar) - k %*% K_inv %*% k
  return(list(EY_xstar=EY_xstar, VarY_xstar=VarY_xstar))
}
```

```
# Get predictions and variances
xstar <- seq(min(x), x2019, length.out=1000)
K_inv <- solve(K + diag(rep(sigma_nhat^2, n)))
tmp <- sapply(xstar, function(x1) GP_predict(x1, K_inv))
EY_xstar <- unlist(tmp['EY_xstar', ])
VarY_xstar <- unlist(tmp['VarY_xstar', ])
```

(Question 1) The following code implements Model 3.

```
require(MASS)
rm(list=ls())

Data <- read.csv('CO2_Mauna_Loa_Data.csv')
colnames(Data) <- c('X', 'Y') # MONTHS, CO2_ppm

# Specify input data
x <- (Data$X - mean(Data$X))/sd(Data$X) # x_1, ..., x_n
# 1958 to 1983
x2019 <- ((2019 - 1960)*12 - mean(Data$X))/sd(Data$X)
x2013 <- ((2013 - 1960)*12 - mean(Data$X))/sd(Data$X)
T0 <- 12/sd(Data$X)
y <- Data$Y # Y_{x_1}, ..., Y_{x_n}
n <- length(x) # No. of observations

# Define covariance function and noise level
cov_k <- function(x, x_, theta){
  return(((theta[1]^2) + (theta[2]^2)*(x*x_)) +
           (theta[3]^2)*exp(-sin(pi*(x - x_)/T0)^2/(theta[4]^2)) +
           (theta[5]^2)*exp(-(x - x_)^2/(theta[6]^2)))
}

# Optimize for the parameter values
log_likelihood <- function(theta, sigma_n) {
  sigma2_nI <- diag(rep(sigma_n^2, n))
  K <- sapply(x, function(x1) sapply(x, function(x2) cov_k(x1, x2, theta)))
  L <- -0.5*( n*log(2*pi) + log(det(K + sigma2_nI)) +
                t(y) %*% solve(K + sigma2_nI) %*% y )
  return(L)
}

sigma_n0 <- 1 # Noise initial value
theta    <- c(114.525412, -3.880875, -19.249946, 2.929246, 1, 1) # Kernel theta initial values
optim_result <- optim(par=c(theta, sigma_n0), fn=function(p) -2*log_likelihood(p[1:6], p[7]))
theta_hat    <- optim_result$par[1:6]
sigma_nhat   <- optim_result$par[7]

GP_predict <- function(xstar, K_inv) {
  # K_inv = (Ktheta + sigma2_n In)^{-1}
  # x is observation covariate vector
  # xstar is query point
  k     <- sapply(X=x, function(x1) cov_k(xstar, x1, theta_hat))
  EY_xstar   <- k %*% K_inv %*% y
  VarY_xstar <- cov_k(xstar, xstar, theta_hat) - k %*% K_inv %*% k
  return(list(EY_xstar=EY_xstar, VarY_xstar=VarY_xstar))
```

```
}

# Get predictions and variances
xstar <- seq(min(x), x2019, length.out=1000)
K_hat <- sapply(x, function(x1) sapply(x, function(x2) cov_k(x1, x2, theta_hat)))
K_inv <- solve(K_hat + diag(rep(sigma_nhat^2, n)))
tmp <- sapply(xstar, function(x1) GP_predict(x1, K_inv))
EY_xstar <- unlist(tmp['EY_xstar', ])
VarY_xstar <- unlist(tmp['VarY_xstar', ])
```

(Question 1) The following code implements Model 2.

```
# Periodic covariance function with SE term
require(MASS)
rm(list=ls())

Data <- read.csv('CO2_Mauna_Loa_Data.csv')
colnames(Data) <- c('X', 'Y') # MONTHS, CO2_ppm

# Specify input data
x <- (Data$X - mean(Data$X))/sd(Data$X) # x_1, ..., x_n
# 1958 to 1983
x2019 <- ((2019 - 1960)*12 - mean(Data$X))/sd(Data$X)
x2013 <- ((2013 - 1960)*12 - mean(Data$X))/sd(Data$X)
T0 <- 12/sd(Data$X)
y <- Data$Y # Y_{x_1}, ..., Y_{x_n}
n <- length(x) # No. of observations

# Define covariance function and noise level
cov_k <- function(x, x_, theta){
  return((theta[1]^2)*exp(-(x - x_)^2/(theta[2]^2)) + (theta[3]^2)*exp(-sin(pi*(x - x_)/T0)^2/
}

# Optimize for the parameter values
log_likelihood <- function(theta, sigma_n) {
  sigma2_nI <- diag(rep(sigma_n^2, n))
  K <- sapply(x, function(x1) sapply(x, function(x2) cov_k(x1, x2, theta)))
  L <- -0.5*( n*log(2*pi) + log(det(K + sigma2_nI)) +
                t(y) %*% solve(K + sigma2_nI) %*% y )
  return(L)
}

sigma_n0 <- 1 # Noise initial value
theta    <- c(266.023859, 8.114217, 10, 10) # Kernel theta initial values
optim_result <- optim(par=c(theta, sigma_n0), fn=function(p) -2*log_likelihood(p[1:4], p[5]))
theta_hat    <- optim_result$par[1:4]
sigma_nhat   <- optim_result$par[5]

GP_predict <- function(xstar, K_inv) {
  # K_inv = (Ktheta + sigma2_n In)^{-1}
  # x is observation covariate vector
  # xstar is query point
  k     <- sapply(X=x, function(x1) cov_k(xstar, x1, theta_hat))
  EY_xstar   <- k %*% K_inv %*% y
```

```
  VarY_xstar <- cov_k(xstar, xstar, theta_hat) - k %*% K_inv %*% k
  return(list(EY_xstar=EY_xstar, VarY_xstar=VarY_xstar))
}

# Get predictions and variances
xstar <- seq(min(x), x2019, length.out=1000)
K_hat <- sapply(x, function(x1) sapply(x, function(x2) cov_k(x1, x2, theta_hat)))
K_inv <- solve(K_hat + diag(rep(sigma_nhat^2, n)))
tmp <- sapply(xstar, function(x1) GP_predict(x1, K_inv))
EY_xstar <- unlist(tmp['EY_xstar', ])
VarY_xstar <- unlist(tmp['VarY_xstar', ])
```

(Question 3) The following script pre-processes the Bitcoin data-set.

```
Data <- read.csv('Bitcoin_Data_Hourly.csv', na.strings="NaN", header=TRUE)
# nrow(Data)
Data <- Data[complete.cases(Data), ] # Remove NaN entries
# nrow(Data) # 37 entries removed

Data$Timestamp <- strptime(Data$Timestamp, format="%d/%m/%Y %H:%M")
# apply(Data, MARGIN = 2, FUN = function(col) class(col))

# Omit days when there is no price change (mostly an artefact)
Data <- Data[(Data[ , 'Open']!=Data[ , 'Close']), ]

# Fix column datatypes
Data <- transform(Data,
                  Open=as.numeric(Open),
                  High=as.numeric(High),
                  Low=as.numeric(Low),
                  Close=as.numeric(Close),
                  Volume=as.numeric(Volume))
```

(Question 3) The following script implements the described experiments.

```
# Sequential classification

# Prior that favours theta0 = 0
alpha00 <- 10
beta00 <- 10
alpha10 <- 1
beta10 <- 1

zEtheta0LU <- c()
zEPd4neqYLU <- c() # assume P(theta0 > 0.5|observations) > 0.5

for (z in seq(min(Data$Low/Data$Open), 1, length.out=1000)){
  o <- Data$Open
  c <- Data$Close
  l <- Data$Low
  x <- (l <= z*o) # Threshold is observed to be breached during the hour
  y <- (c >= z*o) # Hour close is geq to buy price

  # Compute posterior parameters after all observations
  alpha0N <- alpha00 + sum(y*x)
```

```
  beta0N  <- beta00 + sum((1 - y)*x)
  alpha1N <- alpha10 + sum(1 - x)
  beta1N  <- beta10 + sum(x)

  # Estimate 95% hdi via sampling
  rtheta0 <- rbeta(n = 1000, shape1=alpha0N, shape2=beta0N)
  rtheta1 <- rbeta(n = 1000, shape1=alpha1N, shape2=beta1N)
  Pd4neqY <- 1 - (rtheta0*(1 - rtheta1) + rtheta1)
  hpd1 <- hdi(Pd4neqY, credMass=0.95)

  Etheta0N <- alpha0N/(alpha0N + beta0N)
  Etheta1N <- alpha1N/(alpha1N + beta1N)

  # Compute highest posterior density interval
  hpd2 <- hdi(qbeta, 0.95, shape1=alpha0N, shape2=beta0N)

  # Store d4's Bayes error distribution's EV, 95% CI
  zEPd4neqYLU <- rbind(zEPd4neqYLU,
                       c(z, 1 - (Etheta0N*(1 - Etheta1N) + Etheta1N),
                         hpd1[['lower']], hpd1[['upper']]))
  # Store d4's conditional error distribution's EV, 95% CI
  zEtheta0LU <- rbind(zEtheta0LU, c(z, alpha0N/(alpha0N + beta0N),
                                    hpd2[['lower']], hpd2[['upper']]))
}

plot(zEtheta0LU[ , 1], zEtheta0LU[ , 2], type='l', ylim=c(0.2, 0.8))
lines(zEtheta0LU[ , 1], zEtheta0LU[ , 3], col='red', lty=2)
lines(zEtheta0LU[ , 1], zEtheta0LU[ , 4], col='red', lty=2)

# Evaluate the classifier's sequential conditional misclassification rate,
# i.e. the misclassifcation rate across queries for which x_t = 1.
o <- Data$Open
c <- Data$Close
l <- Data$Low
smr <- c()
for (z in seq(min(Data$Low/Data$Open), 1, length.out=1000)) {
  x <- (l <= z*o) # Threshold is observed to be breached during the hour
  y <- (c >= z*o) # Hour close is geq to buy price

  # Compute posterior parameters after each step
  alpha0t <- alpha00 + Reduce(function(a, b) a + b, y*x, accumulate = T)
  beta0t  <- beta00 + Reduce(function(a, b) a + b, (1-y)*x, accumulate = T)

  # Compute decisions
  dt <- as.numeric(pbeta(0.5, shape1=alpha0t, shape2=beta0t) < 0.5)

  # Evaluate sequential misclassification rate on examples for which xt = 1
  ix <- (x == 1)
  smr <- rbind(smr, c(z, mean(y[ix] != dt[ix])))
}

# Bootstrap resample dataset, re-compute sequential misclassification rate
B <- 4000
```

```r
z <- 0.963
Bsmr <- c()
for (s in 1:B) {
  set.seed(s)
  ix <- sample(1:length(l), size=length(l), replace=T)
  x  <- (l <= z*o) # Threshold is observed to be breached during the hour
  y  <- (c >= z*o) # Hour close is geq to buy price
  x  <- x[ix]
  y  <- y[ix]

  # Compute posterior parameters after each step
  alpha0t <- alpha00 + Reduce(function(a, b) a + b, y*x, accumulate = T)
  beta0t  <- beta00 + Reduce(function(a, b) a + b, (1-y)*x, accumulate = T)

  # Compute decisions
  dt <- as.numeric(pbeta(0.5, shape1=alpha0t, shape2=beta0t) < 0.5)

  # Evaluate sequential misclassification rate on examples for which xt = 1
  ix <- (x == 1)
  Bsmr <- rbind(Bsmr, c(s, mean(y[ix] != dt[ix])))
}
```