# Homework #2

Due: February 28, 2025 at 11:59 PM

## Classification and Bias-Variance Trade-offs

### Introduction

This homework is about classification, bias-variance trade-offs, and uncertainty quantification.

The datasets that we will be working with relate to astronomical observations and loan applicants The first dataset, found at `data/planet-obs.csv`, contains information on whether a planet was observed (as a binary variable) at given points in time. This will be used in Problem 1. The second dataset, available at `data/hr.csv`, details different loan applicants and their measured debt to income ratio and credit score. You will work with this data in Problem 3.

As a general note, for classification problems we imagine that we have the input matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ (or perhaps they have been mapped to some basis $\mathbf{\Phi}$, without loss of generality) with outputs now "one-hot encoded." This means that if there are $K$ output classes, rather than representing the output label $y$ as an integer $1, 2, \ldots, K$, we represent $\mathbf{y}$ as a "one-hot" vector of length $K$. A "one-hot" vector is defined as having every component equal to 0 except for a single component which has value equal to 1. For example, if there are $K = 7$ classes and a particular data point belongs to class 3, then the target vector for this data point would be $\mathbf{y} = [0, 0, 1, 0, 0, 0, 0]$. We will define $C_1$ to be the one-hot vector for the 1st class, $C_2$ for the 2nd class, etc. Thus, in the previous example $\mathbf{y} = C_3$. If there are $K$ total classes, then the set of possible labels is $\{C_1 \ldots C_K\} = \{C_k\}_{k=1}^K$. Throughout the assignment we will assume that each label $\mathbf{y} \in \{C_k\}_{k=1}^K$ unless otherwise specified. The most common exception is the case of binary classification ($K = 2$), in which case labels are the typical integers $y \in \{0, 1\}$.

### Resources and Submission Instructions

We encourage you to read CS181 Textbook's Chapter 3 for more information on linear classification, gradient descent, and classification in the discriminative setting. Read Chapter 2.8 for more information on the trade-offs between bias and variance.

In problems 1 and 3, you may use `numpy` or `scipy`, but not `scipy.optimize` or `sklearn`. Example code is given in the provided notebook. **We highly recommend that you use Google Colab for problems 1 and 3 to avoid numerical stability issues.**

Please type your solutions after the corresponding problems using this LaTeX template, and start each problem on a new page.

Please submit the **writeup PDF to the Gradescope assignment 'HW2'**. Remember to assign pages for each question. Please submit your **LaTeX file and code files to the Gradescope assignment 'HW2 - Supplemental'**. **You must include your plots in your writeup PDF.** The supplemental files will only be checked in special cases, e.g. honor code issues, etc.

**Problem 1** (Exploring Bias-Variance and Uncertainty)

In this problem, we will explore the bias and variance of a few different model classes when it comes to logistic regression and investigate two sources of predictive uncertainty in a synthetic (made-up) scenario.

We are using a powerful telescope in the northern hemisphere to gather measurements of some planet of interest. At certain times however, our telescope is unable to detect the planet due to its positioning around its star. The data in `data/planet-obs.csv` records the observation time in the "Time" column and whether the planet was detected in the "Observed" column (with the value 1 representing that it was observed). These observations were taken over a dark, clear week, which is representative of the region. Since telescope time is expensive, we would like to build a model to help us schedule and find times when we are likely to detect the planet.

1. Split the data into 10 mini-datasets of size $N = 30$ (i.e. dataset 1 consists of the first 30 observations, dataset 2 consists of the next 30, etc. This has already been done for you). Consider the three bases $\boldsymbol{\phi}_1(t) = [1, t]$, $\boldsymbol{\phi}_2(t) = [1, t, t^2]$, and $\boldsymbol{\phi}_3(t) = [1, t, t^2, t^3, t^4, t^5]$. For each of these bases, fit a logistic regression model using $\text{sigmoid}(\mathbf{w}^\top \boldsymbol{\phi}(t))$ to each dataset by using gradient descent to minimize the negative log-likelihood. This means you will be running gradient descent 10 times for each basis, once for each dataset.

   Use the given starting values of $\mathbf{w}$ and a learning rate of $\eta = 0.001$, take 1,000 update steps for each gradient descent run, and make sure to average the gradient over the data points at each step. These parameters, while not perfect, will ensure your code runs reasonably quickly.

2. After consulting with a domain expert, we find that the probability of observing the planet is periodic as the planet revolves around its star—we are more likely to observe the planet when it is in front of its star than when it is behind it. In fact, the expert determines that observation follows the generating process $y \sim \text{Bern}(f(t))$, where $f(t) = 0.4 \times \cos(1.1t + 1) + 0.5$ for $t \in [0, 6]$ and $y \in \{0, 1\}$. Note that we, the modelers, do not usually see the true data distribution. Knowledge of the true $f(t)$ is only exposed in this problem to allow for verification of the true bias.

   Use the given code to plot the true process versus your learned models. Include your plots in your solution PDF.

   **In no more than 5 sentences**, explain how bias and variance reflected in the 3 types of curves on the graphs. How do the fits of the individual and mean prediction functions change? Keeping in mind that none of the model classes match the true generating process exactly, discuss the extent to which each of the bases approximates the true process.

**Problem 1** (cont.)

3. If we were to increase the size of each dataset drawn from $N = 30$ to a larger number, how would the bias and variance change for each basis? Why might this be the case? You may experiment with generating your own data that follows the true process and plotting the results, but this is **not** necessary. **Your response should not be longer than 5 sentences**.

4. Consider the test point $t = 0.1$. Using your models trained on basis $\boldsymbol{\phi}_3$, report the predicted probability of observation of the *first* model (the model trained on the first 30 data points). How can we interpret this probability as a measure of uncertainty? Then, compute the variance of the classification probability over your 10 models at the same point $t = 0.1$. How does this measurement capture another source of uncertainty, and how does this differ from the uncertainty represented by the classification probability? Repeat this process (reporting the first model's classification probability and the variance over the 10 models) for the point $t = 3.2$.

   Compare the uncertainties and their sources at times $t = 0.1$ and $t = 3.2$.

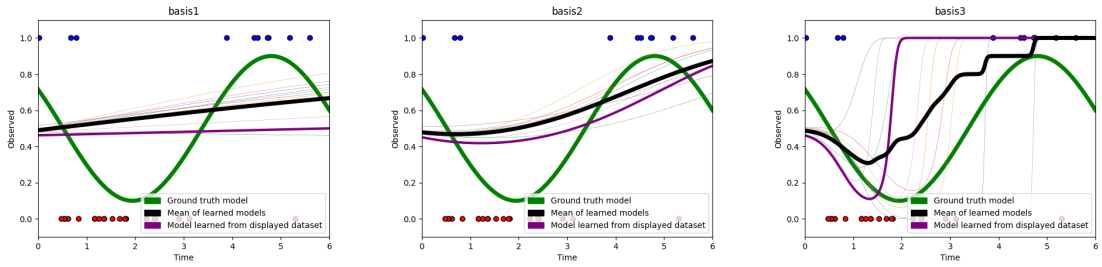
5. We now need to make some decisions about when to request time on the telescope. The justifications of your decisions will be sent to your funding agency, which will determine whether you will be allocated funds to use the telescope for your project. **In no more than 10 lines**, answer the following questions.

   - To identify the ideal time, which model(s) would you use and why?
   - What time would you request, and why?
   - Your funding agency suggests using a different telescope in a humid area near the equator. Can you still use your model to determine when the planet is likely to be visible? Why? Are there adaptations that may be necessary?
   - You seek out a team that has used the alternative telescope for observing this planet, and they provide you their observation file `data/planet-obs-alternate.csv`. Compare the observations from your telescope to theirs. What seems to be happening? What might be an appropriate model for this? Your funding agency asks you to refit your models on these new data. Do you think this is a reasonable ask, and if so, how will it help you make better decisions about when to request viewing time? If not, why do you think the additional modeling will not help? You do *not* need to do any modeling for this question!

   In these questions, we are looking for your reasoning; there may be more than one valid answer.

**Solution**:

1. Code in hw2.ipynb file. Nothing is necessary here.
2. Here are the relevant visualizations of the bases.




The bias-variance trade-off is visible in the differences in the black and purple lines. The purple line is an individually fit model, and thus is fit on a smaller subset of the data. This model will have higher variance because in the smaller dataset, it is more influenced by an individual data point and thus is more overfit to it, however, when we compare that to the mean of the models (i.e. the black line) we can see the bias being increased because as it tries to generalize more, it loses a little specificity to the data. Since the model classes are not able to match the generating function, we are able to get closer by increasing dataset dimensionality (i.e. model dimensionality) via the basis changes. On the domain that we are focused on we are seeing high-low-high-low, with the first high not being the peak, and only a glimpse of the decline when $x > 6$, as such, as expected the linear model does very poorly (can't recognize the changes in derivative of any point), the model with quadratic basis does better (can capture high-low-high) but has high bias (flattens out) when generalized due to the continued periodic nature, and finally the cubic basis does the best as it is able to capture more of the peaks and valleys of the ground truth model (high variance for purple lines, though, as it is able to react more strongly to individual data points).

3. I am not sure whether this question refers to increasing the size of the total dataset so that each of the 10 mini-datasets still have no overlap, but are larger, or whether the total size stays the same and increasing $N$ simply causes overlap in the mini-datasets that individual models are trained on. In as much, I will answer both separately:

   (a) (Same size) Since the original dataset contains 300 points, and we split it into 10 subsets, $N = 30$. However, if we were to increase $N$ then there would be overlap in the data that trains each individual model. As such, variance would decline in the individual models because they are trained on more data that more comprehensively represents the true nature of the system. Furthermore, the variance between each individual models would decline and bias increase, because they would be trained on similar data so they would be more overlap in which datapoints an individual model would fit towards. This trend would be true for every basis, yet for the more complex basis the variance would decrease more: for linear basis each individual model would be more similar, though its variance was never that high due to its inability to classify system, the quadratic basis tells similar story as the individual models converge towards the mean, and the cubic which had the highest varianced models would benefit the most.

   (b) (Increase size of dataset) In this case, if there is still no overlap in the mini-datasets that each dataset is trained on, you get a similar story to above, but for slightly different reason. Before, the variance of individual models decreased because each mini-dataset had increasing similarity with the others due to overlap, here, the increasing similarity stems from each mini-dataset being more representative of overall trends. By increasing the size of dataset, a mini-dataset will be closer to

the actual trends of the data, and thus each model will have more accurate information to best fit towards what it should be. In as much, for each basis, bias remains similar, and variance can decline as the models will train on more representative datasets. However, if we suspect that the original dataset we were provided was not actually representative, then bias can actually increase as when we increase the size because, the periodic nature will be more visible and thus all of these basis which cannot fit a periodic function will become more flattened (i.e. more like just the mean of the data), to compensate, thus increasing bias.

4. $t = 0.1$ first model probability: 0.49897.
   First off these probabilities are the probabilities of classification to class 0. Here we can see that it is extremely close to 0.5 which would simply be equal probability for either class, suggesting that there is a lot of uncertainty as to which class to predict for points of value $t = 0.1$.
   Variance of models at $t = 0.1$: 0.00028
   Despite the uncertainty of the classification, given there is such a low variance, all the models would predict very similar predictions. Even though the models may agree, it is not necessarily accurate, because we have shown that these basis functions will struggle to model periodicity (especially on an end range point like $t = 0.1$).

   $t = 3.2$ first model probability: $9.358 * 10^{-6}$
   Here we can see a very low probability of being class 0 (i.e. this model would likely predict class 1). Thus there seems to be quite high certainty for this specific model in this prediction.
   Variance of models at $t = 3.2$: 0.15753
   However, despite that individual model's prediction probability, the variance across the models suggest there is more uncertainty in predicting points at this value. This is in line with how we saw that the cubic basis models had the highest variance across individual models, and thus a point like $t = 3.2$ can have very different predicted probabilities across different individual models.

5. (a) To identify the ideal time I would use the basis-3 models. Modeling periodicity with polynomial basis is difficult, but the increased complexity in basis3 is the best of the 3 options. Furthermore, even though it has the highest variance at the individual model level, training on the whole dataset (unlike what we did) allows it to reduce that variance by not being so focused on each individual points.

   (b) I would request time at time 3 (this is me attempting to be unbiased by my knowledge of the true data distribution). The reason I chose this is via inspecting the basis3 graphs. I suspect that the individual models that made that jump, towards high probability, earliest have a skewed mini-dataset, and then also on true for those who have not made jump yet. By choosing somewhat of a aggregated value I try to counteract the variance in individual models, while also knowing that I should not wait until after all the models predict high probability of class 1 because planetary orbits will be periodic, so the predictions on the ends of domain (where they flatten) are likely inaccurate.

   (c) I would not be comfortable using the model to know when to use a different telescope. The positioning on the earth means a different time is best to view the planet, as well as the humidity in the air will make the images less clear. If it is absolutely necessary, you could do some transformations to the data, to discover the shifted period based on being at the equator, but I do not know the relevant calculations.

   (d) I would also have reservations with using our model trained on different telescope data to predict when we should use our telescope. Further investigation as to its location is necessary, to learn wether we should expect similar visibility trends as there are for our telescope. Exploring this alternate telescope's dataset, it seems that for the same number of observations (300), there are many more times in the alternate dataset that the planet is observed (203 versus 155). This suggests that the alternate telescope is located in a different place, or at least for some reason

is capable of observing the planet much more. From a location perspective this telescope could be on a different latitude which would affect the time of year and weather conditions that make for best viewing of the planet. If on a different longitude, the planet will be visible at different times of day, so unless a transformation was performed on the data it would be difficult to think that the additional modeling could help at all. However, even then, I would not be sure how to perform the transformation. It is not simply adjusting the hours, because both Earth and the planet are spinning and moving through space. Furthermore, this is an oversimplification of the factors that affect this, so more than that simple transformation would be necessary. As such, unless they are in the same location, and a reason for why the planet was observed more in the alternate telescope is found, I would not be comfortable using our model trained on that data to predict about our telescope.

**Problem 2** (Maximum likelihood in classification)

Consider now a generative $K$-class model. We adopt class prior $p(\mathbf{y} = C_k; \boldsymbol{\pi}) = \pi_k$ for all $k \in \{1, \ldots, K\}$ (where $\pi_k$ is a parameter of the prior). Let $p(\mathbf{x}|\mathbf{y} = C_k)$ denote the class-conditional density of features $\mathbf{x}$ (in this case for class $C_k$). Consider the data set $D = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$ where as above $\mathbf{y}_i \in \{C_k\}_{k=1}^K$ is encoded as a one-hot target vector and the data are independent.

1. Write out the log-likelihood of the data set, $\ln p(D; \boldsymbol{\pi})$.
2. Since the prior forms a distribution, it has the constraint that $\sum_k \pi_k - 1 = 0$. Using the hint on Lagrange multipliers below, give the expression for the maximum-likelihood estimator for the prior class-membership probabilities, i.e. $\hat{\pi}_k$. Make sure to write out the intermediary equation you need to solve to obtain this estimator. Briefly state why your final answer is intuitive.

For the remaining questions, let the class-conditional probabilities be Gaussian distributions with the same covariance matrix

$$p(\mathbf{x}|\mathbf{y} = C_k) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}), \text{ for } k \in \{1, \ldots, K\}$$

and different means $\boldsymbol{\mu}_k$ for each class.

3. Derive the gradient of the log-likelihood with respect to vector $\boldsymbol{\mu}_k$. Write the expression in matrix form as a function of the variables defined throughout this exercise. Simplify as much as possible for full credit.
4. Derive the maximum-likelihood estimator $\hat{\mu}_k$ for vector $\boldsymbol{\mu}_k$. Briefly state why your final answer is intuitive.
5. Derive the gradient for the log-likelihood with respect to the covariance matrix $\boldsymbol{\Sigma}$ (i.e., looking to find an MLE for the covariance). Since you are differentiating with respect to a *matrix*, the resulting expression should be a matrix!
6. Derive the maximum likelihood estimator $\hat{\Sigma}$ of the covariance matrix.

**Hint: Lagrange Multipliers.** Lagrange Multipliers are a method for optimizing a function $f$ with respect to an equality constraint, i.e.

$$\min_{\mathbf{x}} f(\mathbf{x}) \text{ s.t. } g(\mathbf{x}) = 0.$$

This can be turned into an unconstrained problem by introducing a Lagrange multiplier $\lambda$ and constructing the Lagrangian function,

$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x}).$$

It can be shown that it is a necessary condition that the optimum is a critical point of this new function. We can find this point by solving two equations:

$$\frac{\partial L(\mathbf{x}, \lambda)}{\partial \mathbf{x}} = 0 \text{ and } \frac{\partial L(\mathbf{x}, \lambda)}{\partial \lambda} = 0$$

**Cookbook formulas.** Here are some formulas you might want to consider using to compute difficult gradients. You can use them in the homework without proof. If you are looking to hone your matrix calculus skills, try to find different ways to prove these formulas yourself (will not be part of the evaluation of this homework). In general, you can use any formula from the matrix cookbook, as long as you cite it. We opt for the following common notation: $\mathbf{X}^{-\top} := (\mathbf{X}^\top)^{-1}$

$$\frac{\partial \mathbf{a}^\top \mathbf{X}^{-1} \mathbf{b}}{\partial \mathbf{X}} = -\mathbf{X}^{-\top} \mathbf{a}\mathbf{b}^\top \mathbf{X}^{-\top}$$
$$\frac{\partial \ln |\det(\mathbf{X})|}{\partial \mathbf{X}} = \mathbf{X}^{-\top}$$

**Solution**:

1. We are tasked with finding $\ln p(D;\pi)$ which is also just the multiplied probabilities of each point:

$$\Pi p(y_i, x_i; \pi)$$

And based on Bayes we know it is equivalent to

$$\Pi p(y_i|\pi)p(x_i|y_i,\pi) = \Pi p(y_i|\pi)p(x_i|y_i)$$

Because $x_i$ is not dependent on $\pi$ thus $\pi$ can be dropped. From there we were given $p(y = C_k;\pi) = \pi_k$, and we take the log of it, which changes it from products to sum. Giving our final answer as:

$$\sum_{i=0}^{n} \ln \pi_{y_i} p(x_i|y_i)$$

Note that $\pi_{y_i}$ is the prior distribution $\pi_k$ for the class of $y_i$. This final answer makes intuitive sense as it recognizes the data points ($p(x_i|y_i)$) with respect to how the priors would have expected for that class ($\pi_{y_i}$).

2. In order to find MLE for prior class membership probabilities we want to take this log-likelihood, and maximize it in order to find the priors, $\pi_k$ that would lead to the greatest probability of $D$ occurring. As such, we will find $dL/d\pi_k$ where $L$ is the log-likelihood of the dataset.
From above we have (split the sum into its two pieces):

$$\sum_{i=1}^{N} \ln \pi_{y_i} + \sum_{i=1}^{N} \ln p(x_i|y_i)$$

The second term is not important as it does not in terms of $\pi_k$. We will add a Lagrange multiplier using the fact that we know that $\sum_{k=1}^{K} \pi_k = 1$ in order to solve.

$$L' = \sum_{i=1}^{N} \ln \pi_{y_i} + \lambda \sum_{k=1}^{K} (\pi_k - 1)$$

$$dL'/d\pi_k = \frac{1}{\pi_k} \sum_{i=1}^{N} I(y_i = C_k) + \lambda = 0$$

Here I am using $I(y_i = C_k)$ as the indicator function that is 1 if $y_i = C_k$, and 0 if it is not. As such that sum will be equivalent to $n_k$, that is, the number of data points in class $k$.

$$\frac{1}{\pi_k} n_k + \lambda = 0$$

$$\frac{n_k}{\pi_k} = -\lambda \Rightarrow \frac{-n_k}{\lambda} = \pi_k$$

Using Langrange multipliers also relies on our other function $\sum_{k=1}^{K} \pi_k = 1$ so lets substitute in to find optimal lambda.

$$\sum_{k=1}^{k} \frac{-n_k}{\lambda} = 1 \Rightarrow \sum_{k=1}^{K} n_k = -\lambda$$

And since the sum of the number of points in a class $k$ for all $K$ classes is simply the number of data points, then $\lambda = -N$. Returning to what we had above $\pi_k = \frac{-n_k}{\lambda}$, we get...

$$\pi_k = \frac{n_k}{N}$$

This makes intuitive sense as the optimal prior distribution for a class $k$ in order to maximize the likelihood of the dataset is the proportion of data points in the dataset in that class.

3. Similar to above, we can ignore the part of the sum that does not rely on $\boldsymbol{\mu}_k$ (i.e. $\pi_k$ term). As such we are focused on $L = \sum_{i=1}^{N} \ln p(x_i|y_i)$ where $p(x_i|y_i) = \mathcal{N}(x_i|\mu_{y_i}, \Sigma)$. However we want this in matrix form, not just over each individual data point. As such, for $\boldsymbol{\mu}_k$ we need to only focus on $y_i = C_k$. Thus we are really looking at the $n_k$ terms in class $C_k$. So the MLE with respect to the classes is $L = \sum_{k=1}^{K} n_k \ln p(\mathbf{x}_k|\mathbf{y}_k = C_k)$. However, importantly, when you pull out the sum with respect to $i$ and replace it with $n_k$ and a sum across $k$, $\mathbf{x}_k$ is actually the average $x_i$ for class $k$. This is because $\sum_{i=1}^{N} x_i = \sum_{k=1}^{K} n_k \mathbf{x}_k$. So from there, lets write the Gaussian normal PDF for this situation, and investigate $\boldsymbol{\mu}_k$.

$$L = \sum_{k=1}^{K} n_k \ln(\frac{1}{\sqrt{2\pi\boldsymbol{\Sigma}}} \exp(-0.5(\mathbf{x}_k - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu}_k))$$

Fortunately we are able to separate some of these terms and reduce a bit, so lets do that.

$$L = \sum_{k=1}^{K} n_k\{-0.5\ln(2\pi) - 0.5\ln(\Sigma)\} - 0.5(n_k)\sum_{k=1}^{K}(\mathbf{x}_k - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu}_k)$$

The terms in brackets can be dropped, and we can now take the derivative with respect to $\boldsymbol{\mu}_k$

$$L = -0.5(n_k)\sum_{k=1}^{K} \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k^T\mathbf{x}_k - 2\boldsymbol{\mu}_k^T\mathbf{x}_k + \boldsymbol{\mu}_k^T\boldsymbol{\mu}_k)$$

When we take the gradient with respect to $\boldsymbol{\mu}_k$ the sum is removed because only one instance in the sum is based on $\boldsymbol{\mu}_k$ (i.e. the specific $k$ for $\boldsymbol{\mu}_k$)

$$\nabla_{\boldsymbol{\mu}_k} = -0.5(n_k)\boldsymbol{\Sigma}^{-1}(-2\mathbf{x}_k + 2\boldsymbol{\mu}_k) \Rightarrow n_k\boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu}_k)$$

4. To solve for the optimal $\boldsymbol{\mu}_k$ we set the gradient to 0, and solve for $\boldsymbol{\mu}_k$.

$$n_k\boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu}_k) = 0$$

$$n_k\boldsymbol{\Sigma}^{-1}\mathbf{x}_k - n_k\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k = 0 \Rightarrow n_k\boldsymbol{\Sigma}^{-1}\mathbf{x}_k = n_k\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k$$

$$\mathbf{x}_k = \boldsymbol{\mu}_k$$

This makes intuitive sense, because, remember from above that $\mathbf{x}_k$ is the average $x_i$ in class $k$ the optimal $\boldsymbol{\mu}_k$ (i.e. optimal mean), is simply the mean of points in class $C_k$.

5. In order to find the gradient log-likelihood with respect to $\boldsymbol{\Sigma}$, we will start with the same expression.

$$L = -0.5\sum_{k=1}^{K}(n_k)(\{\ln(2\pi)\} + \ln(\boldsymbol{\Sigma})) - 0.5\sum_{k=1}^{K}(n_k)(\mathbf{x}_k - \boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu}_k)$$

Now lets take the gradient with respect to $\boldsymbol{\Sigma}$. (Second term is from provided formula from matrix cookbook)

$$\nabla_{\boldsymbol{\Sigma}} = -0.5\sum_{k=1}^{K}(n_k)\boldsymbol{\Sigma}^{-1} + 0.5\sum_{k=1}^{K}(n_k)\boldsymbol{\Sigma}^{-\top}((\mathbf{x}_k - \boldsymbol{\mu}_k)(\mathbf{x}_k - \boldsymbol{\mu}_k))^T\boldsymbol{\Sigma}^{-\top}$$

Lets say that $(\mathbf{x}_k - \boldsymbol{\mu}_k)(\mathbf{x}_k - \boldsymbol{\mu}_k)^\top = S_k$ to make things cleaner. Also the sum term on the left can be reduced to a product with $N$ since $\boldsymbol{\Sigma}$ does not depend on $k$, and all the data points in each class added together is simply the number of datapoints.

$$\nabla_{\boldsymbol{\Sigma}} = -0.5(N)\boldsymbol{\Sigma}^{-1} + 0.5\sum_{k=1}^{K}(n_k)\boldsymbol{\Sigma}^{-\top}S_k\boldsymbol{\Sigma}^{-\top}$$

We can go even further to say that $\sum_{k=1}^{K}(n_k S_k) = S$ and pull that out to get rid of the sum, because, again, $\boldsymbol{\Sigma}$ does not depend on $k$.

$$\nabla_{\boldsymbol{\Sigma}} = -0.5(N)\boldsymbol{\Sigma}^{-1} + 0.5\boldsymbol{\Sigma}^{-\top} S \boldsymbol{\Sigma}^{-\top}$$

Importantly, covariance matrices are always symmetric, so $\boldsymbol{\Sigma}^{-\top} = \boldsymbol{\Sigma}^{-1}$, thus we can reduce further

$$\nabla_{\boldsymbol{\Sigma}} = 0.5\boldsymbol{\Sigma}^{-1}(S\boldsymbol{\Sigma}^{-1} - N)$$

6. To find the MLE for $\boldsymbol{\Sigma}$ we set the above gradient to 0.

$$0 = 0.5\boldsymbol{\Sigma}^{-1}(S\boldsymbol{\Sigma}^{-1} - N)$$

$$0 = \boldsymbol{\Sigma}^{-1}(S\boldsymbol{\Sigma}^{-1} - N)$$

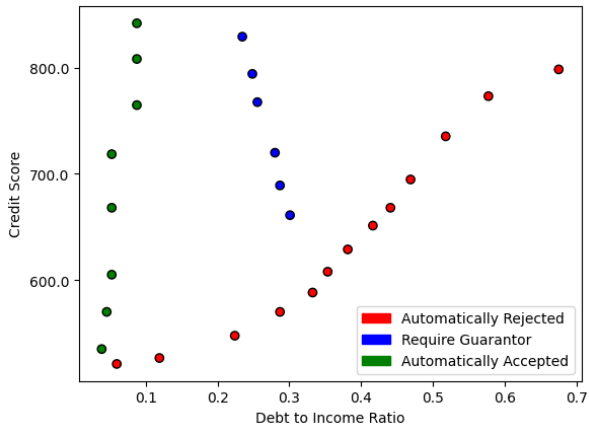
Since we know that $\boldsymbol{\Sigma}$ can't be 0 since it is an invertible matrix then the term $(S\boldsymbol{\Sigma}^{-1} - N)$ must be attributed to that side of the equation equalling 0. Thus...

$$0 = S\boldsymbol{\Sigma}^{-1} - N \Rightarrow \boldsymbol{\Sigma}^{-1} = N/S$$

$$\boldsymbol{\Sigma} = \frac{\sum_{k=1}^{K} n_k(\mathbf{x}_k - \boldsymbol{\mu}_k)(\mathbf{x}_k - \boldsymbol{\mu}_k)^\top}{N}$$

**Problem 3** (Classifying Loan Applicants)

In this problem, you will code up three different classifiers to classify different types of loan applicants. The file `data/hr.csv` contains data on debt to income ratio measured in tenths of a percent and credit score. The data can be plotted on these two axes:




Please implement the following classifiers in the `SoftmaxRegression` and `KNNClassifier` classes. **For this problem, apply the following transformation to all data:**

$$\phi(\mathbf{x}) = \left[x_1 \cdot \frac{200}{7} - 7.5, \frac{x_2 - 500}{140} + 0.5\right]^\top$$

where $x_1$ and $x_2$ represent the values for debt to income ratio and credit score, respectively. This transformation has been applied to the data for you in the notebook.

a) **A generative classifier with Gaussian class-conditional densities with a *shared covariance* matrix** across all classes. Feel free to re-use your Problem 2 results.

b) **Another generative classifier with Gaussian class-conditional densities , but now with a *separate covariance* matrix** learned for each class. (Note: The staff implementation can switch between the two Gaussian generative classifiers with just a few lines of code.)

c) **A multi-class logistic regression classifier** using the softmax activation function. In your implementation of gradient descent, **make sure to include a bias term and use L2 regularization** with regularization parameter $\lambda = 0.001$. Limit the number of iterations of gradient descent to 200,000, and set the learning rate to be $\eta = 0.001$.

d) **Another multi-class logistic regression classifier** with the additional feature map:

$$\phi(\mathbf{x}) = [\ln(x_1 + 10), x_2^2]^\top$$

where $x_1$ and $x_2$ represent the values for debt to income ratio and credit score, respectively.

e) **A kNN classifier** in which you classify based on the $k = 1$ and $k = 5$ nearest neighbors and the following distance function:

$$dist(loan\_app_1, loan\_app_2) = (debt_1 - debt_2)^2/9 + (credit_1 - credit_2)^2$$

where nearest neighbors are those with the smallest distances from a given point.

Note 1: When there are more than two labels, no label may have the majority of neighbors. Use the label that has the most votes among the neighbors as the choice of label.

Note 2: The grid of points for which you are making predictions should be interpreted as our test space. Thus, it is not necessary to make a test point that happens to be on top of a training point ignore itself when selecting neighbors.

**Problem 3** (cont.)

After implementing the above classifiers, complete the following exercises:

1. Plot the decision boundaries generated by each classifier for the dataset. Include them in your PDF. Identify the similarities and differences among the classifiers. What explains the differences—in particular, which aspects or properties of each model dictate the shape of its decision boundary?

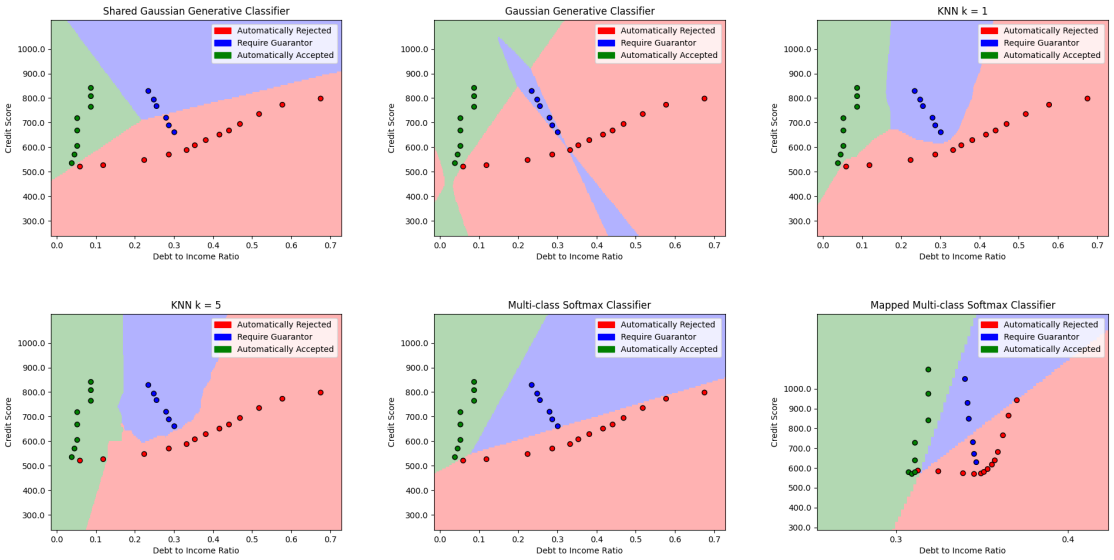
2. Consider a loan applicant with Debt to Income Ratio 0.32 and Credit Score 350. To which class does each classifier assign this applicant? Report the classification probabilities of this applicant for models (c) and (d).

   Interpret how each model makes its classification decision. What else should we, the modelers, be aware of when making predictions on a point "far" from our training data? **Your response should no be longer than 5 sentences.**

3. Can you think of any ethical problem that might arise from using this classifier to make loan decisions? You may approach this from any angle you like. For instance, can you think of someone who might have a low credit score and high debt-to-income ratio that you believe should nonetheless be offered a loan? Are there other variables that should be accounted for to ensure fair decisions? Are credit scores and debt-to-income ratio good bases for loan decisions? More generally, is using a classifier trained on past decisions to determine loan eligibility problematic in any way?

**Solution**:

1. Here are the following plotted decision boundaries. We can see how the shared Gaussian, and Logistic classifiers (untransformed) are forced to have linear boundary lines. This is because they are based on linear classifiers, and the dataset is not distorted polynomial-ly. This forces their decision boundaries to be similar, because they are all still trying to optimize the best classifying decision boundary. On the other hand, when each class has its own covariance matrix, it creates the opportunity for a quadratic decision boundary, which makes the weird plot below. In KNN, you also have a non-linear decision boundary, because the points are not linear, and thus if the model relies on the distances to a non-linear set of points, the boundary will be non-linear. Furthermore, as k increases, a more varied line can occur, as your neighbors can change in many different ways (3 neighbors change in one step), as opposed to only max one neighbor change with K=1.




2. With the point (0.32, 350), the classifiers predict the following:
   Shared Gaussian at (0.32, 350) predicts class: 0
   Gaussian at (0.32, 350) predicts class: 0
   Multi-class Logistic Classifier at (0.32, 350) predicts class: 0
   Mapped Multi-class Logistic Classifier at (0.32, 350) predicts class: 1
   KNN k=1 at (0.32, 350) predicts class: 0
   KNN k=5 at (0.32, 350) predicts class: 0

   The models are generally consistent about predicting class 0 (except for the basis change). For the Gaussian classifiers, it does so by using probability of a certain class with respect to priors, the means and covariances of a class. The logistic classifiers make a decision based on an optimal linear combination of weights, however we can see from the above, that the mapped classifier, predicted a different class, which shows that it was more unsure about which class to predict (also seen below). For the KNN models, the prediction is based on nearest points, which means it relies on the assumption that each point is correctly categorized, especially with low k-values. When we are making predictions on a point far from the training data, we have to be careful, because our model is not trained on data that is representative of that point, and as such, even if it may be difficult for the model to generalize that far from the domain that it is comfortable in.

   Then when we are exploring the classification probabilities for models (c) and (d) (i.e. the mapped, and unmapped multi-class logistic classifiers), we learn the following.
   **Unmapped**:

Class $0 \rightarrow 1.00000000e+00$
Class $1 \rightarrow 3.37978041e-23$
Class $3 \rightarrow 2.33032653e-35$
**Mapped:**
Class $0 \rightarrow 2.19047413e-01$
Class $1 \rightarrow 7.80928775e-01$
Class $3 \rightarrow 2.38124931e-05$

3. Yes, there is problems with using this classifier to make loan decisions, because every situation is unique. This classifier only relies on 2 points of information, which is not nearly sufficient to explain ever persons situation. For example, a person who has a low credit score and high-debt-to-income ration might still deserve a loan in the situation where maybe they are going to get promoted soon, and just had large medical expenses. Both these scenarios would improve the person's ability to get a loan, but under this classifier it would suggest that no loan should be extended. As such, it may be necessary to add more variables to the model, but also still recognize it may not be perfect, because more variables in the model might still not be sufficient. We have seen that even models that generalize well, make mistakes. This goes to show that a classifier trained on past decisions may be problematic, nonetheless banks still use it because it is an effective way to streamline the process (saving them money).

**Name**: Ethan Veghte
**Collaborators and Resources**: