Homework 2: 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. 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 kinds of stars and their measured magnitude and temperature. 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.

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 10,000 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 4 (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 $t = 3.2$.

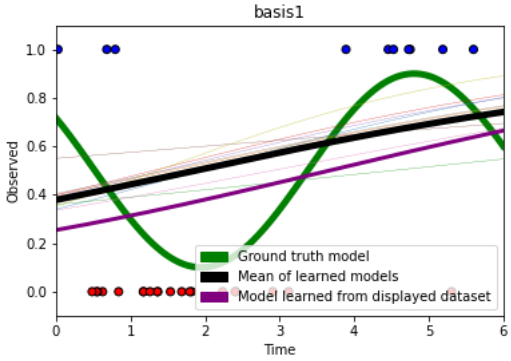
5. We now need to make some decisions about when to request time on the telescope. The justifications of your decisions will be sent to your funding agency, which will determine whether you will be allocated funds to use the telescope for your project.

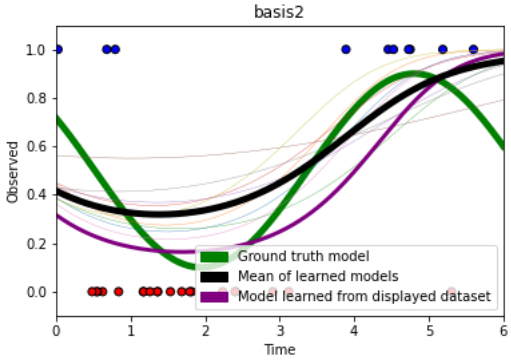
   - To identify the ideal time, which model(s) would you use and why?
   - What time would you request, and why?
   - Your funding agency suggests using a different telescope in a humid area near the equator. Can you still use your model to determine when the planet is likely to be visible? Why? Are there 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!
   - The team who shared the data tells you that the effect you're seeing is due to the weather around the telescope. Weather forecasts for the area are usually accurate a few days in advance. Does that affect your strategy for requesting time? If so, how? If not, why not?

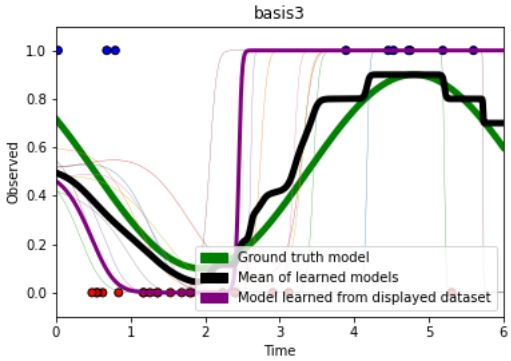
   In these questions, we are looking for your reasoning; there may be more than one valid answer.

Solution

1. See `main_hw2_soln.ipynb` for reference.
2. Students should have identical-looking plots, but if a student's plots do not look identical to the plots below, please give them the benefit of the doubt.







For basis 1, we can see that for most t, the difference between the green and black curves is quite large; hence bias is relatively high. The vertical pread of the individual models is relatively small; hence variance is relatively low.

As we continue to basis 2 and then basis 3 (increasing complexity), the difference between the green and black curves progressively shrinks for most x, and the vertical spread of the individual models progressively grows. That is, the bias decreases while the variance increases.

In expectation (over datasets), the more complex basis is better able to approximate the true process. However, we can see that it also has greater potential to overfit to specific datasets, especially when

the number points is small.

3. In general, as the size of the dataset increases, models no longer overfit as much, thus reducing the variance. We may also expect that the variance decreases for the more complex bases because the variance for the first basis is already small. However, the bias generally levels off, as our model families (polynomial bases) are simply limited in how well they can approximate $\cos t$.

4. The classification probabilities and model variances are given below*.

   Classification probabilities for $t = 0.1$: 0.5197126318279172, $t = 3.2$: 1.3930894701224269e-08

   Model variances for $t = 0.1$: 0.0033857352352883617, $t = 3.2$: 0.18473313961274132

   ***Numpy instability issues mean that some students might not get the same answer—we've seen (1, 0.2328) for the values in the second row. The first point should have close to 0.5 classification probability with low variance, while the second should be either 0 or 1 with high variance.**

   The classification probability represents the uncertainty associated with the inherent randomness of the data and the true process: this is known as *stochastic* or *aleatoric* uncertainty. This uncertainty captures the fact that, according to the true process, if we could run several "experiments" looking through our telescope at time $t = 0.1$, we would sometimes detect the planet and sometimes not.

   Model variance captures how well our models can actually represent the true process. This is related to the concept of *epistemic* uncertainty, which is uncertainty that can, in principle, be reduced with more information (data). In essence, the first type of uncertainty is inherently tied to the true nature of the problem being modelled and cannot be reduced, while the second type of uncertainty is tied to our limited understanding of the true nature or process.

   Most student answers should be fine as long as they are reasonable and take into account the above discussion. It turns out that according to the true process, the probability of observation at time $t = 0.1$ is 0.678 while that at time $t = 3.2$ is 0.424, which is much more similar than one would expect just by looking at the first model.

5. For all responses, there is more than one correct answer. The purpose of this question is to encourage students to think beyond the data collection and model training process, to consider environmental and other factors involved in model development.

   - **Ideal time:** Students should discuss the bias-variance trade-off when comparing the 3 models. If they want a model with more consistent prediction of planet observation probabilities (low variance), they might choose the basis 2 model, acknowledging that it does not include models with the lowest bias.
   - **Requested time:** Students should describe requesting a time that both has a high confidence and high probability of observing the planet.
   - **Humid area:** Many answers are acceptable, describing how re-purposing models in different environments might be in/effective. For example, the rate of planet orbiting its start doesn't change based on the location of the telescope, but the weather and viewing angles may be considered.
   - **New data:** A fraction of 1's in the alternate data file are randomly changed to 0's. Since this is unpredictable, fitting a model to such data will not help, so it is an unreasonable ask. A model will need to do something in addition to fitting the curve (students may suggest alternative solutions), or accept aleatoric uncertainty with more inaccurate predictions.
   - **Weather:** The additional weather information could be incorporated into interpretation of model predictions. *Q: Was there a more in-depth answer in mind here?*

Problem 2 (Maximum likelihood in classification, 15pts)

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 π_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 λ 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

Let $\mathbb{1}_{\mathbf{y}_i=C_k}$ be an indicator variable denoting whether some $\mathbf{y}_i$ is in the class C_k.

1. The log-likelihood is given by:

$$\boxed{\ln p(\{\mathbf{x}_i, \mathbf{y}_i\}|\{\pi_k\}) = \sum_{i=1}^{N}\sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i=C_k}\big(\ln p(\mathbf{x}_i|\mathbf{y}_i = C_k) + \ln \pi_k\big)}$$

   Note that the class prior π_k is part of the likelihood, as opposed to a *model prior*

2. Using the note at the end of the exercise, we know we need to maximize:

$$\ln p(\{\mathbf{x}_i, \mathbf{y}_i\}|\{\pi_k\}) = \sum_{i=1}^{N}\sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i=C_k}\big(\ln p(\mathbf{x}_i|\mathbf{y}_i = C_k) + \ln \pi_k\big) + \lambda\left(\left(\sum_{k=1}^{K}\pi_k\right) - 1\right) \qquad (\square)$$

   We take the derivative of $(\square)$ with respect to π_k and set it to 0 to solve for the π_k's maximizing the constraint:

$$\sum_{i=1}^{N}\frac{\mathbb{1}_{\mathbf{y}_i=C_k}}{\pi_k} + \lambda = 0$$

   Rearrange to solve for π_k. We can pull the π_k out of the sum because it does not depend on i:

$$\pi_k = \frac{-1}{\lambda}\sum_{i=1}^{N}\mathbb{1}_{\mathbf{y}_i=C_k}$$

   We take the derivative of $(\square)$ with respect to λ and set it to 0 to solve for the λ maximizing the constraint:

$$\sum_{k=1}^{K}\pi_k - 1 = 0$$

$$\implies \sum_{k=1}^{K}\pi_k = 1$$

   Plug our expression for the optimal π_k into this original constraint:

$$\sum_{k=1}^{K}\left(\frac{-1}{\lambda}\sum_{i=1}^{N}\mathbb{1}_{\mathbf{y}_i=C_k}\right) = 1$$

   Solve for λ:

$$\lambda = -\sum_{k=1}^{K}\sum_{i=1}^{N}\mathbb{1}_{\mathbf{y}_i=C_k} = -N$$

   Now plug this back into the expression for π_k :

$$\boxed{\hat{\pi_k} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{1}_{\mathbf{y}_i=C_k}}$$

   Note that the MLE for the class-prior simply reflects the proportion of classes in our dataset.

3. The log-likelihood we got in part 1 was:

$$\ln p(\{\mathbf{x}_i, \mathbf{y}_i\}|\{\pi_k\}) = \sum_{i=1}^{N}\sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i=C_k}\big(\ln p(\mathbf{x}_i|\mathbf{y}_i = C_k) + \ln \pi_k\big)$$

We plug in the Gaussian PDF for the $p(\mathbf{x_i}|\mathbf{y_i} = C_k)$ term and simplify out the logs, rewriting the log-likelihood as:

$$\ln p(\{\mathbf{x}_i, \mathbf{y}_i\}|\{\pi_k\}) = \sum_{i=1}^{N}\sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i=C_k}\left(-\frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu}_k)\right) + \text{constants w.r.t } \boldsymbol{\mu}_k$$

By Equation 86 from the Matrix Cookbook, we know for a symmetric matrix W and vectors x, s that:

$$\frac{\partial}{\partial s}(x-s)^T W(x-s) = -2W(x-s)$$

Then the gradient of the log likelihood w.r.t $\boldsymbol{\mu}_k$ will be:

$$\boxed{\sum_{i=1}^{N} \mathbb{1}_{\mathbf{y}_i=C_k}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu}_k)}$$

Common Errors: Often, people would write something like the below to indicate that the log likelihood and the exponent of the gaussian are separated by an additive constant (wrt μ_k)

$$\ln p(\{\mathbf{x}_i, \mathbf{y}_i\}|\{\pi_k\}) \propto \sum_{i=1}^{N}\sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i=C_k}\left(-\frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu}_k)\right)$$

This is technically incorrect since the lefthand side and righthand side differ by an additive constant (wrt μ_k) not a multiplicative constant. The confusion may stem from the fact that we can do the same type of analysis with multiplicative constants, in which case the $\propto$ captures the proportional relationship. In other words, we could do all of this analysis before the log, which would look something like the below.

$$\begin{aligned}
p(\{\mathbf{x}_i, \mathbf{y}_i\}|\{\pi_k\}) &= \prod_{i=1}^{N}\prod_{k=1}^{K}\left(p(\mathbf{x}_i|\mathbf{y}_i = C_k)\pi_k\right)^{\mathbb{1}_{\mathbf{y}_i=C_k}} \\
&= \prod_{i=1}^{N}\prod_{k=1}^{K}\left((2\pi)^{\frac{d}{2}}|\Sigma|^{-\frac{1}{2}}e^{-\frac{1}{2}(\mathbf{x}_i-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i-\boldsymbol{\mu}_k)}\pi_k\right)^{\mathbb{1}_{\mathbf{y}_i=C_k}} \\
&= \prod_{i=1}^{N}\prod_{k=1}^{K}\left(e^{-\frac{1}{2}(\mathbf{x}_i-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i-\boldsymbol{\mu}_k)}\right)^{\mathbb{1}_{\mathbf{y}_i=C_k}}\prod_{i=1}^{N}\prod_{k=1}^{K}\left((2\pi)^{\frac{d}{2}}|\Sigma|^{-\frac{1}{2}}\pi_k\right)^{\mathbb{1}_{\mathbf{y}_i=C_k}} \\
&\propto \prod_{i=1}^{N}\prod_{k=1}^{K}\left(e^{-\frac{1}{2}(\mathbf{x}_i-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i-\boldsymbol{\mu}_k)}\right)^{\mathbb{1}_{\mathbf{y}_i=C_k}}
\end{aligned} \tag{1}$$

The last line is equivalent to

$$\ln p(\{\mathbf{x}_i, \mathbf{y}_i\}|\{\pi_k\}) = \sum_{i=1}^{N}\sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i=C_k}\left(-\frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu}_k)\right) + \text{constants w.r.t } \boldsymbol{\mu}_k$$

Either methods for approaching this are fine, but just don't get confused between them!

4. We derive the MLE estimator for $\boldsymbol{\mu_k}$ by setting the derivative we found in Part 3 equal to 0. We have:

$$\sum_{i=1}^{N} \mathbb{1}_{\mathbf{y}_i = C_k} \cdot \boldsymbol{\Sigma}^{-1}(\mathbf{x_i} - \boldsymbol{\mu_k}) = 0$$

Breaking up the summation, separating $(\mathbf{x_i} - \boldsymbol{\mu_k})$, and pulling out the $\boldsymbol{\Sigma}^{-1}$, we have:

$$\boldsymbol{\Sigma}^{-1} \sum_{i=1}^{N} \mathbb{1}_{\mathbf{y}_i = C_k} \cdot \mathbf{x_i} = \boldsymbol{\Sigma}^{-1} \sum_{i=1}^{N} \mathbb{1}_{\mathbf{y}_i = C_k} \cdot \boldsymbol{\mu_k}$$

Because it is very clear that the covariance matrix is invertible, as indicated by the -1 exponent indicating its inverse, we can multiply both sides of the equation by the covariance matrix. We can also remove our mean vector from the RHS summation, as the mean vector should be constant and not depend on the iterations of the summation. Thus, we have:

$$\sum_{i=1}^{N} \mathbb{1}_{\mathbf{y}_i = C_k} \cdot \mathbf{x_i} = \boldsymbol{\mu_k} \sum_{i=1}^{N} \mathbb{1}_{\mathbf{y}_i = C_k}$$

Finally, isolating our mean vector, we have found our MLE estimator for $\boldsymbol{\mu_k}$:

$$\boxed{\hat{\boldsymbol{\mu_k}} = \frac{\sum_{i=1}^{N} \mathbb{1}_{\mathbf{y}_i = C_k} \cdot \mathbf{x_i}}{\sum_{i=1}^{N} \mathbb{1}_{\mathbf{y}_i = C_k}}}$$

This expression for the MLE of $\boldsymbol{\mu_k}$ is intuitive because it is the average of the $\mathbf{x}$ vectors in our dataset who are classified as Class k. This makes sense because $\boldsymbol{\mu_k}$ is supposed to be the mean vector of points that are generated to fall into Class k. We are effectively taking a sample mean here.

5. Once again, we proceed by copying over the expression for the log-likelihood:

$$\ell(\boldsymbol{\Sigma}; D) = \sum_{i=1}^{N} \sum_{k=1}^{K} [(\mathbb{1}_{\mathbf{y}_i = C_k} \cdot ln(\pi_k)) + (\mathbb{1}_{\mathbf{y}_i = C_k} \cdot ln(p(\mathbf{x_i} \mid \mathbf{y_i} = C_k)))]$$

Substituting the Multivariate Normal PDF into this expression (and taking the natural log in the process), and discarding any constants and terms that do not depend on $\boldsymbol{\Sigma}$, we have:

$$\ell(\boldsymbol{\Sigma}; D) = \sum_{i=1}^{N} \sum_{k=1}^{K} (\mathbb{1}_{\mathbf{y}_i = C_k} \cdot ln(p(\mathbf{x_i} \mid \mathbf{y_i} = C_k)))$$

$$\ell(\boldsymbol{\Sigma}; D) = \sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i = C_k} \cdot \Big[ln\Big(\frac{1}{(2\pi)^{n/2} \cdot \sqrt{det(\boldsymbol{\Sigma})}} \cdot exp\Big[-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu_k})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu_k}) \Big]$$

$$\ell(\boldsymbol{\Sigma}; D) = \sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i = C_k} \cdot \Big[-\frac{1}{2} \cdot ln(det(\boldsymbol{\Sigma})) + (-\frac{1}{2}) (\mathbf{x_i} - \boldsymbol{\mu_k})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x_i} - \boldsymbol{\mu_k}) \Big]$$

Using the two cookbook formulas provided to us in the problem statement and taking the derivative with respect to the covariance matrix, we have:

$$\frac{\partial}{\partial \boldsymbol{\Sigma}} \ell(\boldsymbol{\Sigma}; D) = \sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i = C_k} \big[-\frac{1}{2} \boldsymbol{\Sigma}^{-T} + (-\frac{1}{2})(-\boldsymbol{\Sigma})^{-T} (\mathbf{x_i} - \boldsymbol{\mu_k})(\mathbf{x_i} - \boldsymbol{\mu_k})^T \boldsymbol{\Sigma}^{-T} \big]$$

Simplifying and multiplying out the negative signs, we have:

$$\boxed{\frac{\partial}{\partial \boldsymbol{\Sigma}} \ell(\boldsymbol{\Sigma}; D) = \sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i = C_k} \big[-\frac{1}{2} \boldsymbol{\Sigma}^{-T} + \frac{1}{2} \boldsymbol{\Sigma}^{-T} (\mathbf{x_i} - \boldsymbol{\mu_k})(\mathbf{x_i} - \boldsymbol{\mu_k})^T \boldsymbol{\Sigma}^{-T} \big]}$$

6. We proceed by taking the derivative we calculated in part 5 and setting it equal to 0.

$$\sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i = C_k} \big[-\frac{1}{2} (\boldsymbol{\Sigma})^{-T} + \frac{1}{2} (\boldsymbol{\Sigma})^{-T} (\mathbf{x_i} - \boldsymbol{\mu_k})(\mathbf{x_i} - \boldsymbol{\mu_k})^T (\boldsymbol{\Sigma})^{-T} \big] = 0$$

Breaking up the summation, multiplying both sides of the resultant equation by $\frac{1}{2}$, and moving the covariance matrix (which we are treating as constant) out of the summation on the LHS, we have:

$$\boldsymbol{\Sigma}^{-T} \sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i = C_k} = \sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i = C_k} \cdot \boldsymbol{\Sigma}^{-T} (\mathbf{x_i} - \boldsymbol{\mu_k})(\mathbf{x_i} - \boldsymbol{\mu_k})^T \boldsymbol{\Sigma}^{-T}$$

We proceed to left-multiply, and then right-multiply both sides of the equation by $\boldsymbol{\Sigma}^T$. Observe that $\sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i = C_k} = N$, as every y_i must be a member of one and only one class. Thus, we have the following:

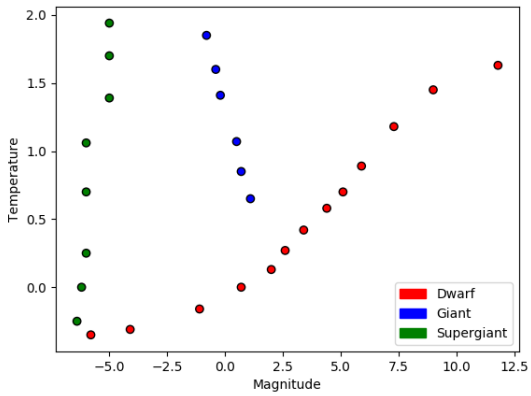
$$N \cdot \boldsymbol{\Sigma}^T = \sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i = C_k} \cdot (\mathbf{x_i} - \boldsymbol{\mu_k})(\mathbf{x_i} - \boldsymbol{\mu_k})^T$$

Finally, we can divide by N to isolate $\boldsymbol{\Sigma}^T$. Note that because the covariance matrix is symmetrical by virtue of the commutative property of covariance, the MLE for $\boldsymbol{\Sigma}^T$ is precisely the MLE for $\boldsymbol{\Sigma}$ itself. Thus, our MLE for the covariance matrix is as follows:

$$\boxed{\hat{\boldsymbol{\Sigma}} = \frac{\sum_{i=1}^{N} \sum_{k=1}^{K} \mathbb{1}_{\mathbf{y}_i = C_k} \cdot (\mathbf{x_i} - \boldsymbol{\mu_k})(\mathbf{x_i} - \boldsymbol{\mu_k})^T}{N}}$$

Problem 3 (Classifying Stars)

In this problem, you will code up three different classifiers to classify different types of stars. The file `data/hr.csv` contains data on magnitude and temperature. The data can be plotted on these two axes:



Please implement the following classifiers in the `SoftmaxRegression` and `KNNClassifier` classes:

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 feature map $\phi(\mathbf{x}) = [\ln(x_1 + 10), x_2^2]^\top$, where x_1 and x_2 represent the values for magnitude and temperature, respectively.

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

$$dist(star_1, star_2) = (mag_1 - mag_2)^2/9 + (temp_1 - temp_2)^2$$

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.

After implementing the above classifiers, complete the following exercises:

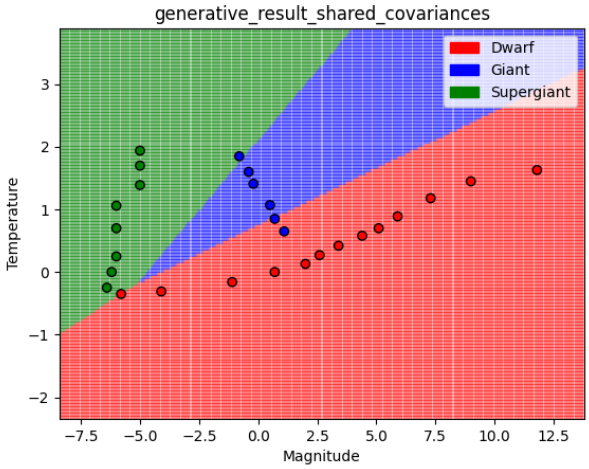
1. Plot the decision boundaries generated by each classifier for the dataset. Include them in your PDF. Identify the similarities and differences among the classifiers. What explains the differences—in particular, which aspects or properties of each model dictate the shape of its decision boundary?

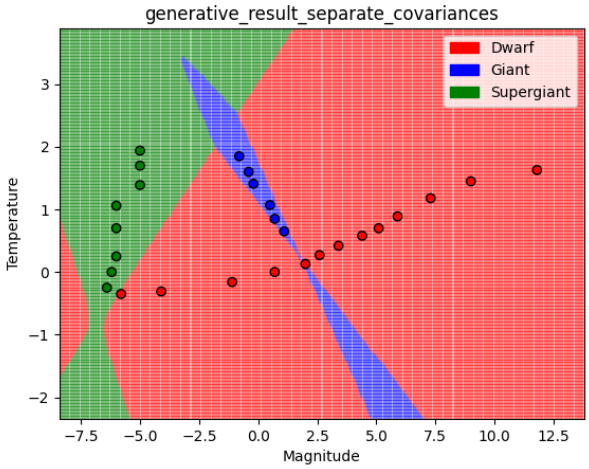
2. Consider a star with Magnitude 3 and Temperature -2. To which class does each classifier assign this star? Report the classification probabilities of this star for each model.

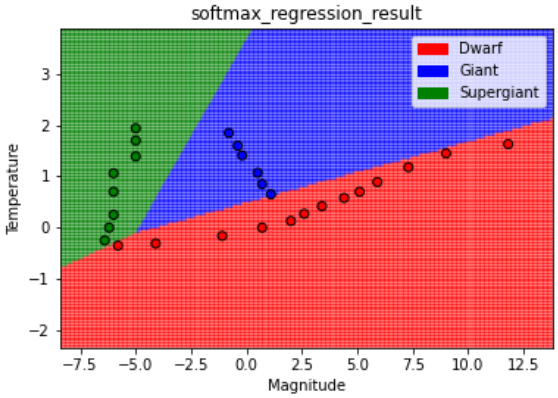
   Interpret how each model makes its classification decision. What are the pros and cons of each interpretation? What else should we, the modelers, be aware of when making predictions on a test point "far" from our training data? **Your response should no be longer than 5 sentences.**

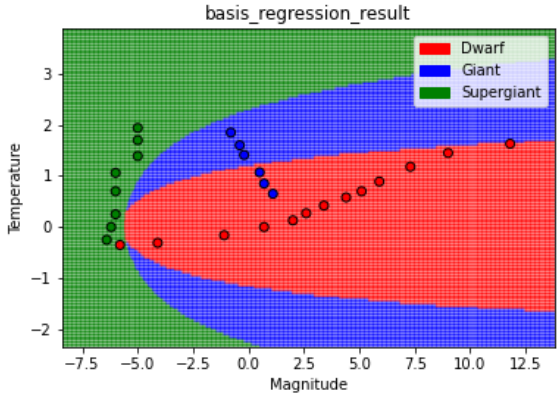
Solution

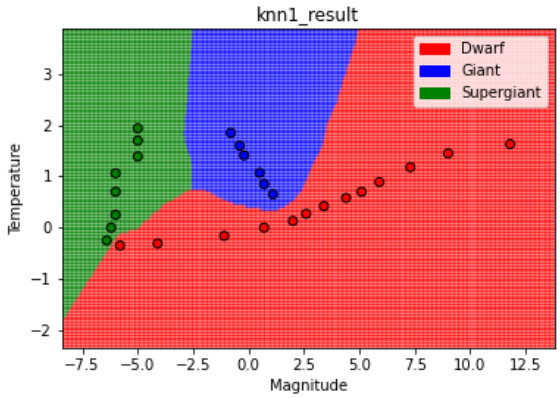
1. See `main_hw2_soln.ipynb` for reference on implementing the different classifiers. The plots are shown below.

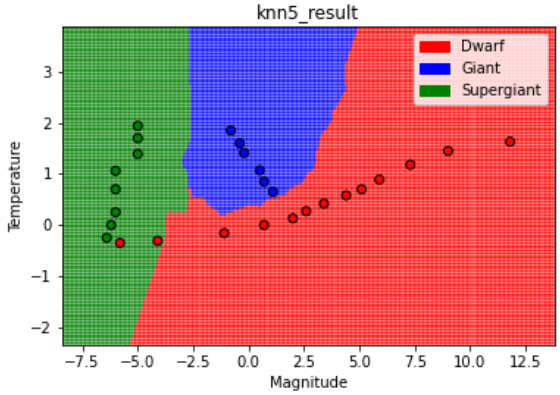












Softmax regression has linear boundaries for the same reasons as logistic regression—the solution for $\mathbf{w}^\top \mathbf{x} = 0$ is linear, and the classifications for multi-class regression are equivalent to doing 3 pairwise classifications and then finding the areas one class has greater probability than both of the other classes.

Basis regression is non-linear due to the non-linear transformation of the input data; if we were to transform the axes as well, we would see a linear decision boundary. In particular, because the feature map has $x_2 \to x_2^2$, we have symmetry about the temperature axis.

Both kNN classifiers have non-linear boundaries due to the way they classify points in a non-parametric manner. We see that the boundaries for $k = 1$ are smoother than $k = 5$, which is expected. Overall,

they are quite similar, and they predict somewhat similarly to the softmax classifier, while the basis regression is very different.

2. The classification probabilities for each model are given below

	Dwarf	Giant	Supergiant
Softmax*	1	0	0
Basis	0.034	0.964	0.001
kNN, (both $k = 1, 5$)	1	0	0

***Hopefully no numerical instability issues arise for this question, but TFs should keep an eye out**

Because this test point is very far from the boundary, the softmax classifier predicts with probability almost 1 that it is a dwarf star. The basis regression predicts that it is a giant with high probability for the same reason. Since all (1 or 5) of its closest neighbors are dwarfs, kNN also predicts that it is a dwarf star.

There is an intuitive geometric reason for the softmax and basis regression, which is an advantage, and kNN is also highly interpretable and makes no probabilistic assumptions. The main disadvantage of the basis regression is the arbitrary choice of choosing a feature map—this often reguires some prior knowledge to justify a transformation. However, the biggest detail to notice is that **our training data may not represent this test point, and therefore our model assumptions** (i.e. being further from the boundary in softmax regression means higher probability, being similar to neighbors in kNN) **may be incorrect.**

Name

Collaborators and Resources

Whom did you work with, and did you use any resources beyond cs181-textbook and your notes?