1. [8 pts] Assume you are trying to build a classifier for two leukemia subtypes, ALL and AML, based on the expression levels of p genes. You are given $n/2$ samples for each of the two subtypes (n samples in total) as training data.

   (a) [5 pts] Provide an algorithm that, given training data as described above, a single test data point and user-defined variable k, finds the label of a test data point. K should be utilized in classification.

Algorithm:

Step 1: Select value for K number of neighbors.

Step 2: Calculate the Euclidean distance of K number of neighbors.

Step 3: Based on the Euclidean distance calculated above, take K numbers of nearest neighbors.

Step 4: Count the number of data points in each of K-nearest neighbors.

Step 5: Assign the new data point to that category for which we obtain the maximum number of data points in the previous step.

Pseudocode:

Input (x, K) where x is the test data point, K is the number of neighbors.

Output: Label for the category the test data point belongs to based on the KNN algorithm.

```
def KNN-classifier(x, K):
    distance = { }      # a dictionary with key = data point and value = euclidean dist b/w that
                          data point and the test data point.
    Set distance[x_i] = d(x, x_i)
    Sort dictionary distance by values  # sorting in the increasing order of the euclidean dist.
    kNearestNeighbors = slice the dictionary (distance) by taking the first K keys.
    AML = 0
    ALL = 0
```

```
for i in range(len(kNearestNeighbors)):
    if kNearestNeighbor[i].label == AML:
        AML += 1
    else:
        ALL += 1
if AML > ALL:
    X.label = AML    # Set the label of the test data point x = AML
else:
    X.label = ALL    # Set the label of the test data point x = ALL
```

(b) [3 pts] Provide runtime of your algorithm.

O(n) : ranging (0,n) to find the euclidean distance.

O(n log(n)) : sorting the dictionary (distance) by values

O(K) : ranging (len(kNearestNeighbors))

Total time taken : $O(n) + O(n\log(n)) + O(K)$

2. [8 pts] Maximum Likelihood estimation(MLE) and Maximum A-posteriori Probability(MAP) estimation are fundamental of the machine learning. In this problem, we will compute them. The distribution we are interested in is Poisson distribution, given as

$$P(X = x; \lambda) = \frac{\lambda^x e^{-\lambda}}{x!}$$

(a) [4 pts] Given independently drawn observations $X_1 = x_1, X_2 = x_2, \cdots, X_n = x_n$, compute $\hat{\theta}^{MLE}$, which is $\text{argmax}_{\hat{\theta}} P(X_1, \cdots, X_n | \theta = \hat{\theta})$ $P(D|\hat{\theta})$

$$L_n(\lambda) = P(D|\lambda) = \prod_{i=1}^{n} P(X_i|\lambda) = \prod_{i=1}^{n} \frac{e^{-\lambda}\lambda^{x_i}}{x_i!} = \frac{e^{-n\lambda}\lambda^{x_1}\lambda^{x_2}\cdots\lambda^{x_n}}{\prod x_i!} = \frac{e^{-n\lambda}\lambda^{\sum_{i=1}^{n} x_i}}{\prod x_i!}$$

$$l_n(\lambda) = -n\lambda + \sum_{i=1}^{n} x_i \log\lambda - \log\prod x_i!$$

$$\text{argmax } \ln(\lambda) = \frac{\partial \ln(\lambda)}{\partial \lambda} = -n + \sum_{i=1}^{n} x_i \cdot \frac{1}{\lambda} = 0 \Rightarrow \sum_{i=1}^{n} x_i \cdot \frac{1}{\lambda} = n$$

$$\boxed{\lambda = \frac{\sum_{i=1}^{n} x_i}{n}}$$

It is known that the Gamma distribution can be used as a conjugate prior for Poisson distribution. The Gamma distribution is denoted as

$$G(\lambda; \alpha, \beta) = \frac{\beta^{\alpha} \lambda^{\alpha-1} e^{-\beta\lambda}}{\Gamma(\alpha)}$$

where $\Gamma(\alpha)$ is a Gamma function, $\Gamma(\alpha) = (\alpha - 1)!$ for positive integer α

(b) [4 pts] Now, given a Gamma distribution as above, calculate MAP.

$$\ln(\lambda) = P(D|\lambda)\,P(\lambda) = \prod_{i=1}^{n} P(x_i|\lambda)\, \frac{\beta^{\alpha} \lambda^{\alpha-1} e^{-\beta\lambda}}{\Gamma(\alpha)} = \prod_{i=1}^{n} \left[e^{-\lambda} \frac{\lambda^{x_i}}{x_i!} \frac{\beta^{\alpha} \lambda^{\alpha-1} e^{-\beta\lambda}}{\Gamma(\alpha)} \right]$$

$$\ln(\lambda) = \sum \left[x_i \ln\lambda - \lambda \ln e + \alpha \ln\beta + (\alpha-1)\ln\lambda - \beta\lambda \ln e - \ln(x_i!) - \ln(\Gamma(\alpha)) \right]$$

$$\sum x_i [\ln\lambda] - n\lambda + n\alpha\ln\beta + n(\alpha-1)\ln\lambda - n\beta\lambda - \sum \ln(x_i!) - n\ln[\Gamma(\alpha)]$$

$$\therefore \text{argmax } \hat{\lambda}_{MAP} : \frac{\partial \ln(\lambda)}{\partial \lambda} = 0$$

$$\Rightarrow \frac{\sum x_i}{\lambda} - n + \frac{n(\alpha-1)}{\lambda} - n\beta = 0$$

$$\boxed{\hat{\lambda}_{MAP} = \frac{\sum_{i=1}^{n} x_i + n(\alpha-1)}{n(\beta+1)}}$$

3. [14 pts] Assume that we have n data points $(X_i, Y_i)_{i=1,2,..,n}$, where the labels Y_i are some (random) function of the features X_i. Suppose we are in a classification setting so $Y_i \in \{1, 2, ..., C\}$ where C is the total number of classes. The kNN predictor for class c for a point x is an estimate of the probability of belonging to the class, i.e.

$$\hat{m}_c(x) = P[Y = c|X = x].$$

(a) [4 pts] Let $N_k(x)$ be a function that gives the set of indices of the k nearest neighbours of x in the dataset. Write down a mathematical expression for the kNN predictor for class c for an arbitrary point x using $N_k(x)$. What is the overall kNN prediction (i.e. the predicted class) for a point x in terms of the expression you found?

$$\hat{m}_c(x) = P[Y=c \mid X=x] = \frac{1}{K} \sum_{i \in N_K(x,D)} \mathbb{I}(y_i = c)$$

where $N_K(x,D)$ are the indices of the K nearest points to x in D and $\mathbb{I}(e)$ is the indicator function defined as follows:

$$\mathbb{I}(e) = \begin{cases} 1 & \text{if } e \text{ is true} \\ 0 & \text{if } e \text{ is false} \end{cases}$$

KNN prediction is c such that $\hat{m}_c(x) = \max_i (\hat{m}_i(x))$

(b) [4 pts] Now assume we are in a regression setting so $Y_i \in R$. The kNN prediction for a point x is now defined as the average label of its k nearest neighbours. Write down a mathematical expression for the kNN predictor $\hat{m}_r(x)$ for an arbitrary point x again using $N_k(x)$.

$$\frac{1}{K} \sum_{i \in N_K(x)} Y_i = \hat{m}_r(x)$$

(c) [3 pts] Assume now that $Y_i = f(X_i) + \epsilon_i$, where $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$ independently for each i. Write down an expression for

$$E[\hat{m}_r(x)]$$

$$E[\hat{m}_r(x)] = E\left[\frac{1}{k}\sum f(x_i) + \epsilon_i\right]$$

$$= \frac{1}{k} E\left[\sum f(x_i)\right] + \cancel{E[\epsilon_i]}^{0}$$

$$= \frac{1}{k}\sum\left[f(x_i)\right]$$

(d) [3 pts] Write down an expression for

$$\mathrm{Var}[\hat{m}_r(x)]$$

$$\mathrm{Var}[\hat{m}_r(x)] = \mathrm{Var}\left[\frac{1}{k}\sum\left[f(x_i) + \epsilon_i\right]\right]$$

$$= \frac{1}{k^2}\left[\cancel{\mathrm{Var}\left[\sum f(x_i)\right]}^{0} + \mathrm{Var}\left[\sum(\epsilon_i)\right]\right]$$

$$= \frac{1}{k^2}\sum_{i=1}^{k}\sigma_i^2$$

4. [20 pts] You learned the following naive Bayes classifier for diagnosing a patient ($Y = 1$ for case and $Y = 0$ for healthy) based on genotypes at two loci X_1 and X_2. The naive Bayes classifier is given as follows:

$$\begin{aligned}
P(Y) &\sim \text{Bernoulli}(0.4)\\
P(X_1|Y=0) &\sim \text{Multinoulli}(0.2, 0.5, 0.3)\\
P(X_2|Y=0) &\sim \text{Multinoulli}(0.1, 0.3, 0.6)\\
P(X_1|Y=1) &\sim \text{Multinoulli}(0.4, 0.3, 0.3)\\
P(X_2|Y=1) &\sim \text{Multinoulli}(0.2, 0.6, 0.2),
\end{aligned}$$

where $Z \sim \text{Multinoulli}(a, b, c)$ means $P(Z = 0) = a$, $P(Z = 1) = b$, $P(Z = 2) = c$.

(a) [5 pts] What is the joint probability distribution $P(Y, X_1, X_2)$ (Hint: think of the numerator of the Bayes theorem)?

$$P(Y, X_1, X_2) = P(X_1, X_2 | Y) P(Y)$$
$$= P(X_1|Y) P(X_2|Y) P(Y)$$

(b) [5 pts] What is the probability distribution $P(X_1, X_2)$ (Hint: think of the denominator in the Bayes theorem.)?

$$P(X_1, X_2) = P(X_1, X_2 | Y=0) + P(X_1, X_2 | Y=1)$$

(c) [5 pts] Classify a new patient with genotype $X_1 = 1$ and $X_2 = 2$. Provide $P(Y = 1|X_1 = 1, X_2 = 2)$ and $P(Y = 0|X_1 = 1, X_2 = 2)$.

$$P(Y=1 | X_1=1, X_2=2) = \frac{P(X_1=1|Y=1) P(X_2=2|Y=1) P(Y=1)}{P(X_1=1|Y=1) P(X_2=2|Y=1) P(Y=1) + P(X_1=1|Y=0) P(X_2=2|Y=0) P(Y=0)}$$

$$= \frac{(0.3)(0.6)(0.4)}{(0.3)(0.6)(0.4) + (0.5)(0.6)(0.6)}$$

$$= 0.118$$

$$P(Y=0 | X_1=1, X_2) = 1 - P(Y=1 | X_1=1, X_2=2)$$
$$= 1 - 0.118$$
$$= 0.882$$

(d) [5 pts] Compared to k nearest neighbor classifier, what is the advantage of using a naive Bayes classifer for the problem stated above?

It is advantageous to use Naive Bayes classifier because it is a generative model and it learns in real time to produce more accurate predictions. On the other hand, because KNN does not learn in real time (no training) it presents risk of overfitting the data. Therefore, naive bayes classifier is computationally more efficient compared to KNN classifier.

5. [50 pts] You will use linear regression to analyze the genotype and gene expression data collected from two yeast strains, called BY and RM, and their 112 progenies. The first two files below contain data to be analyzed and the last two files contain the meta-data.

   - SNPs.txt: Genotype data for 114 yeast strains and 1,260 SNPs. Each row corresponds to the SNP genotypes for each of the 114 strains. The top two rows are the genotypes of BY and RM strains, coded with all 1's for BY and all 0's for RM. The genotypes of the other progeny strains have a mixture of 1's from BY and 0's from RM. SNPs are ordered along the genome.
   - expression.txt: Gene expression levels for 114 yeast strains and 3,684 genes.
   - gene_list.txt: The names of the 3,684 genes, for your reference.
   - strain_list.txt: The names of the 114 yeast strains, for your reference. Note that the top two strains are BY and RM.

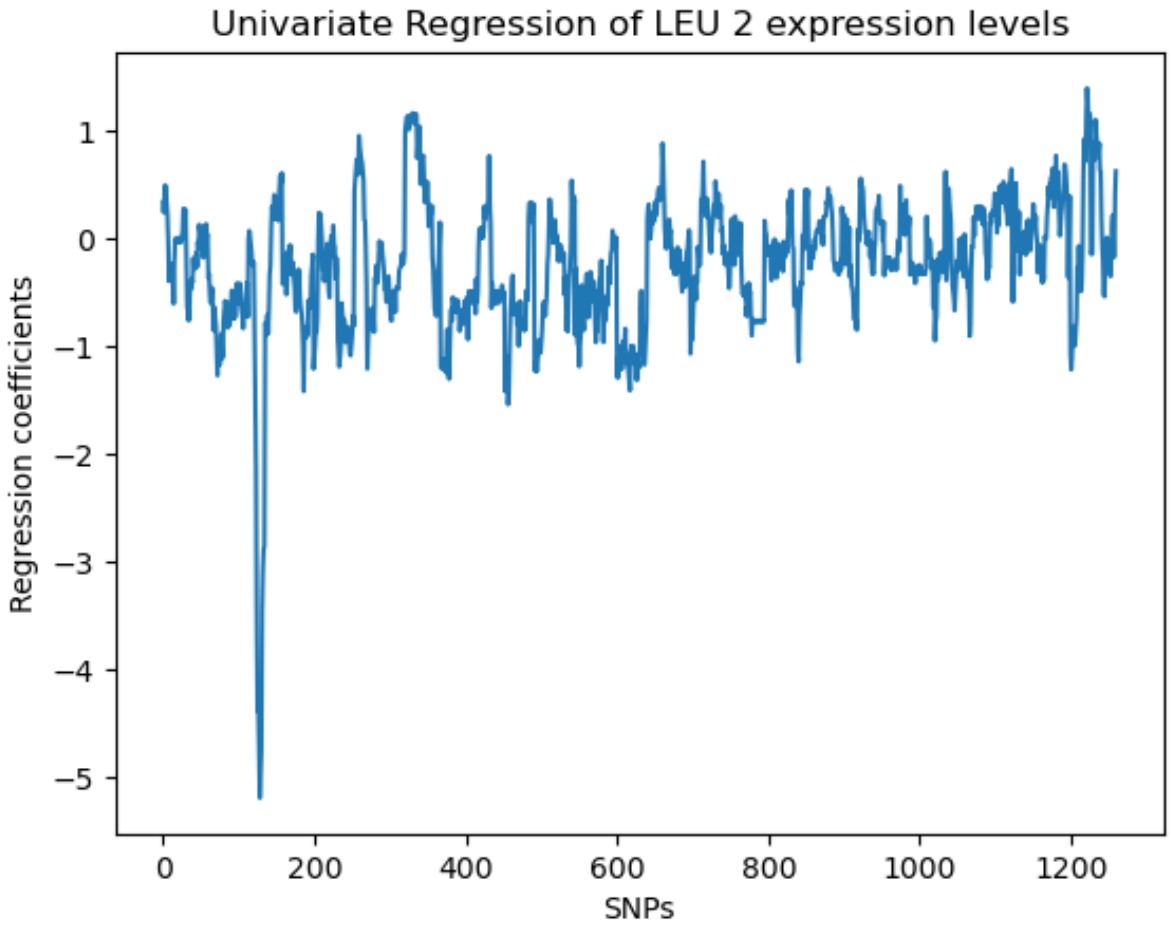
   BY and RM strains are known to have differences in the sequence of gene $LEU2$, which corresponds to YCL018W. In this homework, you will perform regression analysis to identify SNPs that affect $LEU2$ expression levels. (From the data above, you can verify that the expression levels of $LEU2$ for all strains are given in column 395 of expression.txt.)

   Note: In all of your analysis below, work with the expression and SNP data after mean-centering so that you can work with the regression model without intercept.

   (a) Univariate regression

   - [10 pts] Perform a univariate regression, using the $LEU2$ expression levels as output and each SNP as input. Obtain $\beta = [\beta_1, \ldots, \beta_J]$, where $J = 1,260$ and β_i is the regression parameter estimated by regressing $LEU2$ expression levels on SNP i. Plot β_i's across the genome.

→ on python file



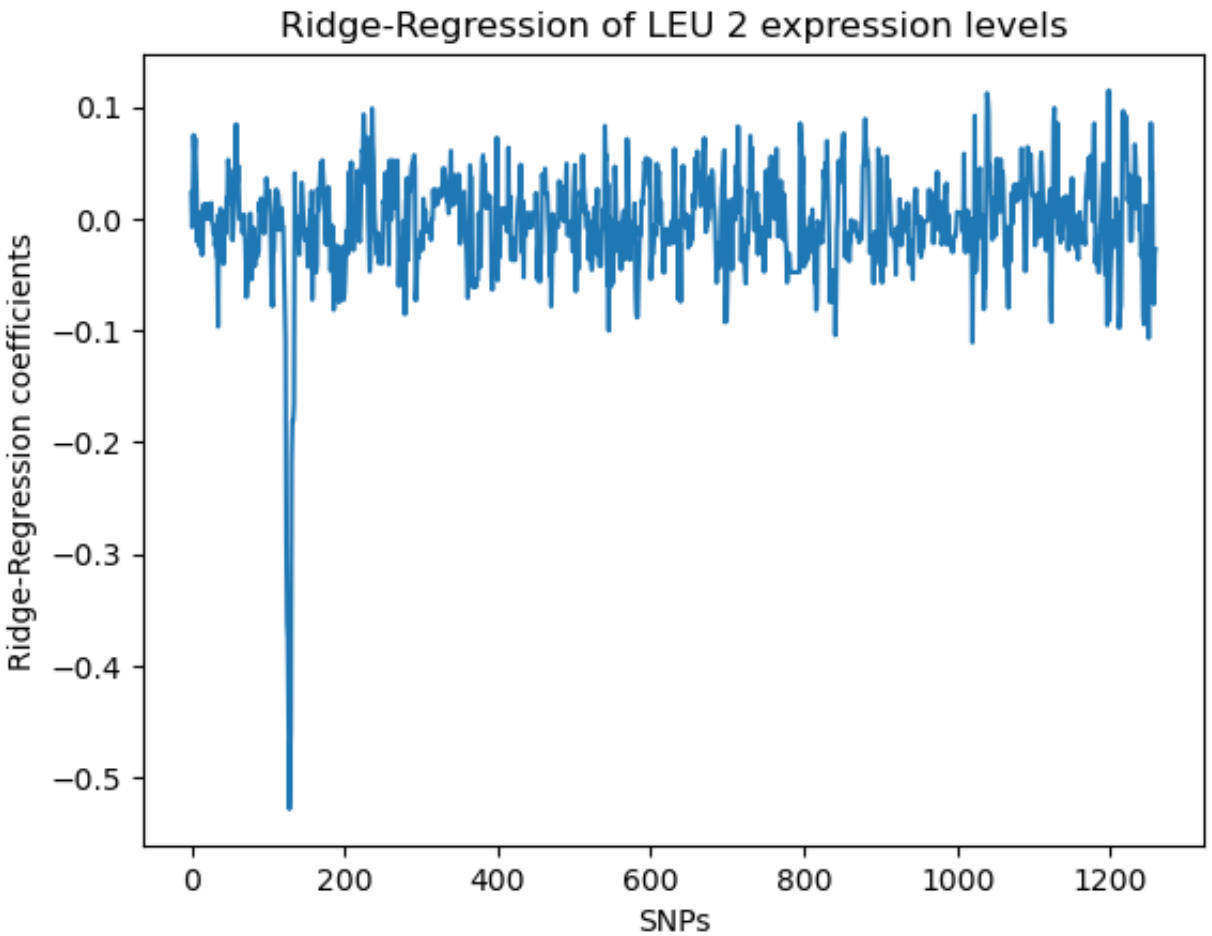
(b) Multivariate regression

- [10 pts] Can you obtain an estimate of the regression coefficients for a multivariate regression model using all SNPs as inputs and $LEU2$ expression as output? Explain why this is not possible. [Hint: you do not need to write a code to answer this question.]

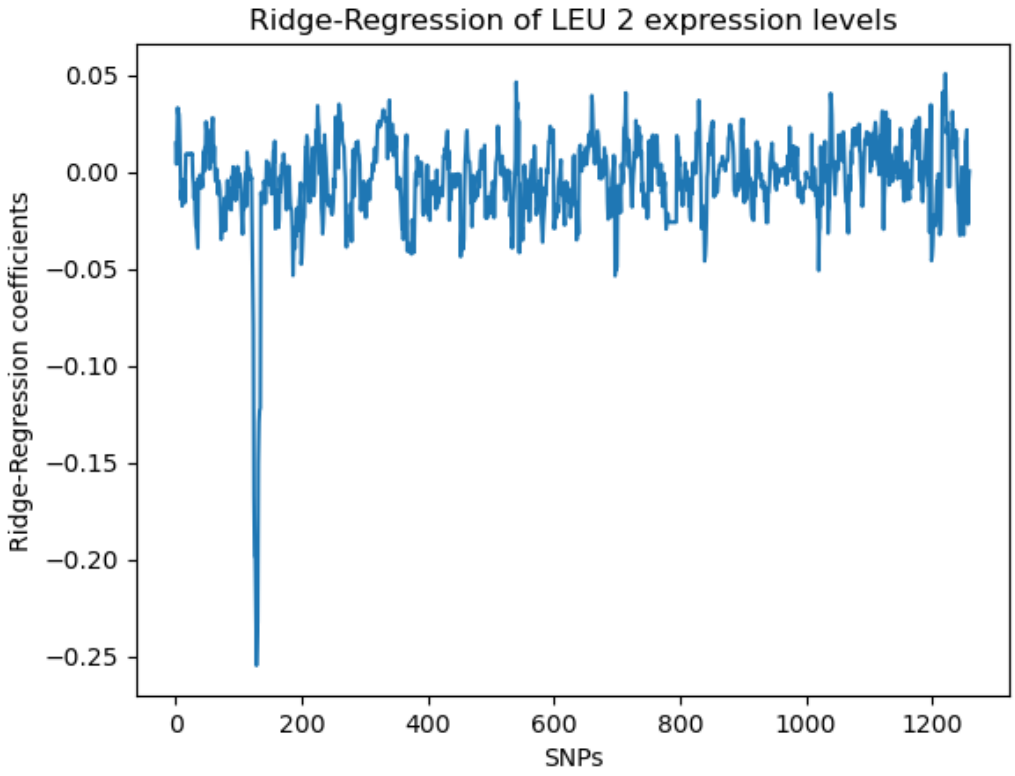
We cannot obtain an estimate of the regression coefficients for a multivariate regression model using all SNPs as inputs and LEU2 expression as output. This is because we have far more number of features (SNPs) than number of samples. In this case, we have 1260 SNPs and only 114 individuals. When samples < features, X^TX is not invertible and therefore $(X^TX)^{-1}$ cannot be computed.

(c) Ridge regression.

- [10 pts] Perform a ridge regression, assuming a prior distribution $\beta_i \sim N(0, \sigma_0^2)$ for $i = 1, \ldots, J$, $J = 1,260$, with $\sigma_0^2 = 5.0$. Your multivariate regression model should predict $LEU2$ expression levels given all SNPs. Plot your estimated regression coefficients $\beta = [\beta_1, \ldots, \beta_J]$.



- [10 pts] Repeat the analysis above with a prior distribution $\beta_i \sim N(0, \sigma_0^2)$ for $i = 1, \ldots, J$, $J = 1,260$, with $\sigma_0^2 = 0.005$. Plot your estimated regression coefficients $\beta = [\beta_1, \ldots, \beta_J]$.



- [10 pts] Explain the different effects of the two prior distributions above on the regression parameter estimates. Which SNP has the strongest influence on the $LEU2$ expression?

Smaller σ_0^2 means a stronger prior belief. Therefore $\sigma_0^2 = 0.005$ represents a stronger prior belief compared to $\sigma_0^2 = 5.0$. A smaller σ_0^2 leads to a larger λ which causes a stronger pull of the regression parameter estimates toward 0.

Based on 0th index, SNP 128 has the strongest influence on the LEU2 expression.