**Post Genomics – Fall 2025**

**Homework 6**

**Due: Thursday, October 16, 2025**

**Part 1. Manipulating Data. (Programming - Python)** *(50 points)*

1.1 Using the Normal and Tumor CSV files from Homework 3. Subset the two CSV files with only the columns, ["chrom", "left", "ref\_seq", "alt\_seq", “Patient\_ID”, ‘VCF\_ID”]

* + 1. How many unique normal patients do we have?

4 normal patients

* + 1. How many unique tumor patients do we have?

5 tumor patients

* + 1. Group by variant info, chrom, left, ref\_seq, and alt\_seq, let the other columns turn into list.

Done. Check output of the code. (Also Available at the end of this document)

* + 1. Create a new column with the number of patients per variant on both the normal and tumor (name the column, N# and T#, respectively).

Done. Check output of the code. (Also Available at the end of this document)

* + 1. Rename the columns, Patient\_ID and VCF\_ID, to have, \_Normal or \_Tumor, added depending which file you are working with.

Done. Check output of the code. (Also Available at the end of this document)

1.2 Using the output from part A, merge (how = outer) the Normal and Tumor together based on the columns [chrom, left, ref\_seq, alt\_seq] into a single CSV file named AML.

* + 1. How many unique normal variants?

Zero (0).

* + 1. How many unique tumor variants?

1408

* + 1. How many variants are shared between normal and tumor (common)?

165

* 1. Using the Normal and Tumor files from Homework 3, concatenate these files along the axis = 0, with this Expand/Explode the rows based on the CSQ columns and save this file as AML\_Expand.csv. Remove duplicate rows.

Saved and uploaded.

* + 1. How many rows are in this file?

10,234

* + 1. Create two new CSVs:

1. Subset of expianded with only the columns, ["SYMBOL", "Gene", "Feature"], name this AML\_gene.csv.

(Completed)

1. Subset of expanded CSV with only the columns, ["chrom", "left", "right", "ref\_seq", "alt\_seq", "Feature", "cDNA\_position", "BIOTYPE"], name this AML\_tx.csv.

(Completed)

**Part 2 (Random Forest)** *(25 points)*

The Iris Dataset is a useful example set for machine learning classification problems. Work through the tutorial (<https://www.geeksforgeeks.org/random-forest-classifier-using-scikit-learn/>), and answer the questions below:

2.1 What was the accuracy of the model you built?

100% accuracy

2.2 What order were the important features ranked?

1. Petal length

2. Petal width

3. sepal length

4. sepal width

2.3 Change 2 of the parameters and repeat the model generation. What 2 parameters did you choose, what effects did they have on the model, and why do you think that was the case?

I’ve chosen test\_size = 0.45 and n\_stimators = 10. Accuracy decreased to 98.53%, the top of most relevant characteristics also changed, now being width the most relevant feature, also confusion matrix, the numbers in the diagonal squares are larger compared to previous and the value [3,2] of the matrix changed to one. All of these happened because of the reduction of the training set, since there was lesser data to practice, the model was not so optimized, that’s why there was a misidentification.

Images of both outputs at the end.

**Part 3 (K-Means Clustering)** *(25 points)*

Work through the tutorial (<https://scikitlearn.org/stable/auto_examples/cluster/plot_cluster_iris.html#sphx-glr-auto-examples-cluster-plot-cluster-iris-py>), and answer the questions below:

* 1. What are some conclusions you can draw about the clustering analysis?

The original clustering results really nail home how crucial choosing the right k is. Setting k=3, which matches the Iris flowers' actual species count, gives you a result that's nearly identical to the Ground Truth. The distinct Setosa group is always perfectly isolated, but the remaining clusters show K-Means' flaws: k=8 massively shreds the data into too many pieces, and trying k=3 with a bad random start messes up the separation of the two overlapping species, proving that good initialization isn't just nice to have it's essential to avoid getting stuck in a bad solution.

* 1. Repeat the process, except change out the cluster number from 8 to 4, and 3 to 2. How does the clustering change?

When we changed the numbers, the results changed predictably: k=4 is a bit of an **overkill**, creating a fourth cluster by splitting an existing group, but it's much better than k=8. On the flip side, k=2 is a classic case of **under-segmentation**, where the model accurately finds the isolated Setosa, but is forced to **lump the other two species together** into a single, big group. Just like before, using a bad random start with k=2 makes things worse, proving the initialization problem is a persistent weak spot, regardless of whether you're over- or under-clustering the data.

Outputs:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A diagram of blue squares

AI-generated content may be incorrect.A blue and white bar graph

AI-generated content may be incorrect.A diagram of a diagram

AI-generated content may be incorrect.A blue and white bar graph

AI-generated content may be incorrect.

A diagram of different types of clustering

AI-generated content may be incorrect.

A diagram of different types of clustering

AI-generated content may be incorrect.

**What to Submit:**

1. **A single PDF with responses and screen shots from Part 1,2, and 3. (Your Last Name) \_HW5.pdf \*\**Submit on Blackboard\*\****
2. **A single Python file with code for Part 1, 2, and 3. (Your Last Name) \_HW5.py \*\* *Submit on GitHub \*\****
3. **The two CSV files from Part 1. AML\_gene.csv and AML\_tx.csv \*\* *Submit on Blackboard \*\****