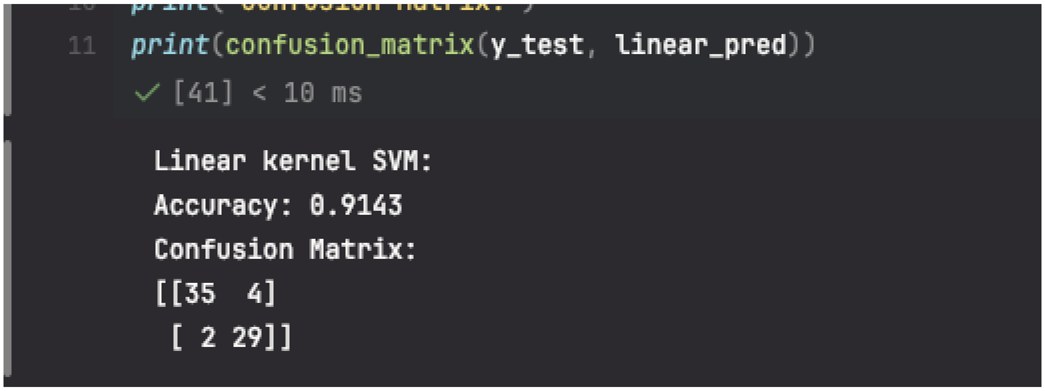
Ananya Singh
Assignment #6

My BUID ends with 8 therefore my remainder is 2.
R= 2: class L= 1 (negative) and L= 3 (positive)

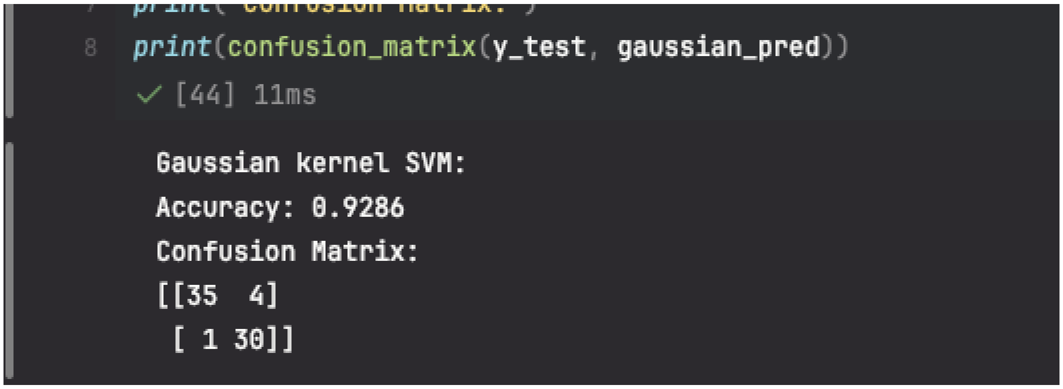
Question #1 Part 1



```
print("Confusion Matrix:")
print(confusion_matrix(y_test, linear_pred))
✓ [41] < 10 ms

Linear kernel SVM:
Accuracy: 0.9143
Confusion Matrix:
[[35  4]
 [ 2 29]]
```

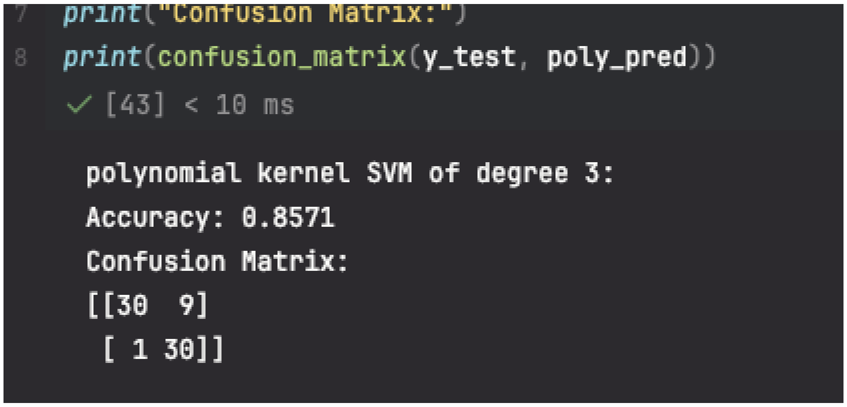
Question #1 Part 2



```
print("Confusion Matrix:")
print(confusion_matrix(y_test, gaussian_pred))
✓ [44] 11ms

Gaussian kernel SVM:
Accuracy: 0.9286
Confusion Matrix:
[[35  4]
 [ 1 30]]
```

Question #1 Part 3

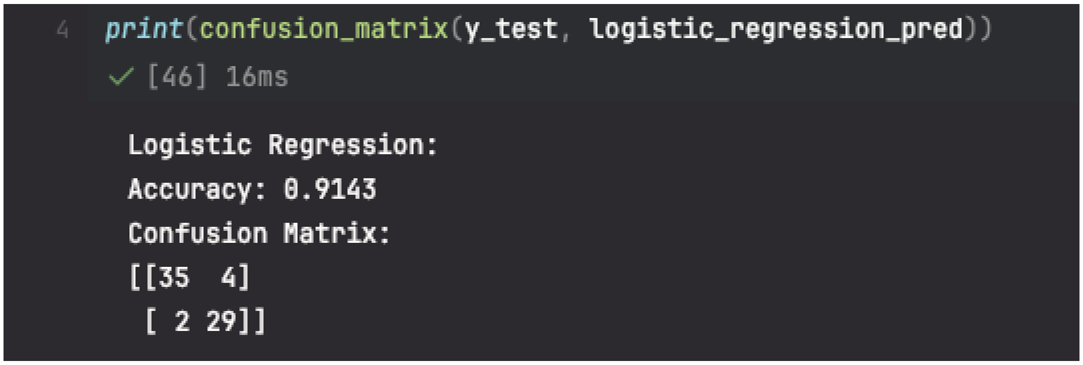


```
print("Confusion Matrix:")
print(confusion_matrix(y_test, poly_pred))
✓ [43] < 10 ms

polynomial kernel SVM of degree 3:
Accuracy: 0.8571
Confusion Matrix:
[[30  9]
 [ 1 30]]
```

Question #2 Part 1

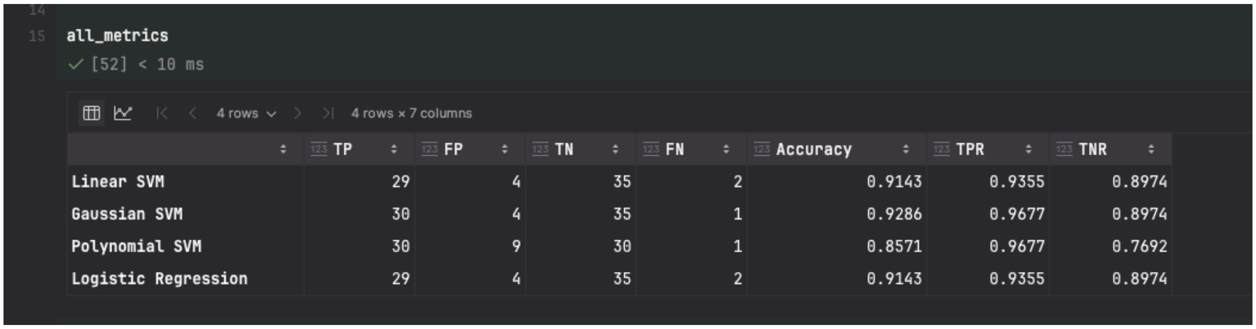
I used Linear Regression:



```
print(confusion_matrix(y_test, logistic_regression_pred))
✓ [46] 16ms

Logistic Regression:
Accuracy: 0.9143
Confusion Matrix:
[[35  4]
 [ 2 29]]
```

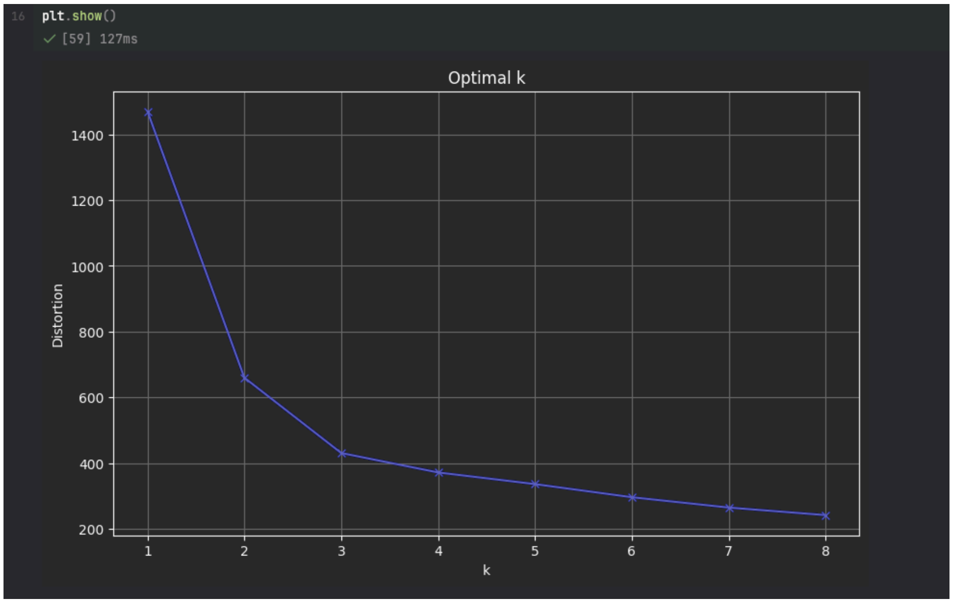
Question #2 Part 2



```
all_metrics
✓ [52] < 10 ms
```

	TP	FP	TN	FN	Accuracy	TPR	TNR
Linear SVM	29	4	35	2	0.9143	0.9355	0.8974
Gaussian SVM	30	4	35	1	0.9286	0.9677	0.8974
Polynomial SVM	30	9	30	1	0.8571	0.9677	0.7692
Logistic Regression	29	4	35	2	0.9143	0.9355	0.8974

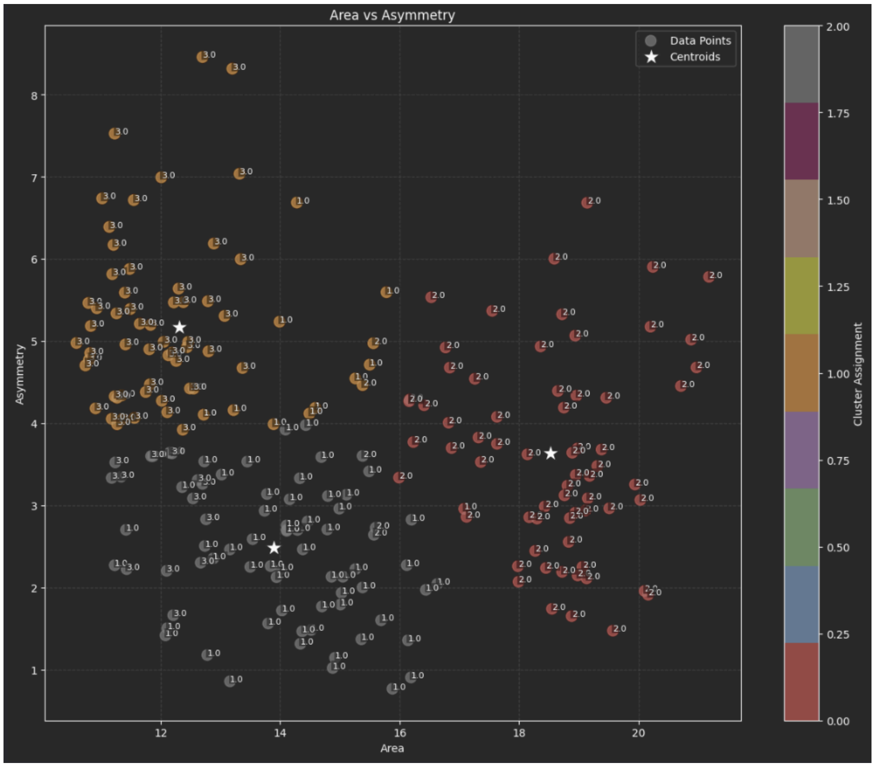
Question #3 Part 1



```
plt.show()
✓ [59] 127ms
```

Based on lecture notes:
k is typically chosen "visually" by the "knee" method, no significant decrease in "loss" function beyond some k
Therefore, visually we can see the **best k is 3** because that’s where the curve transitions from steep to gradual in terms of the slope.

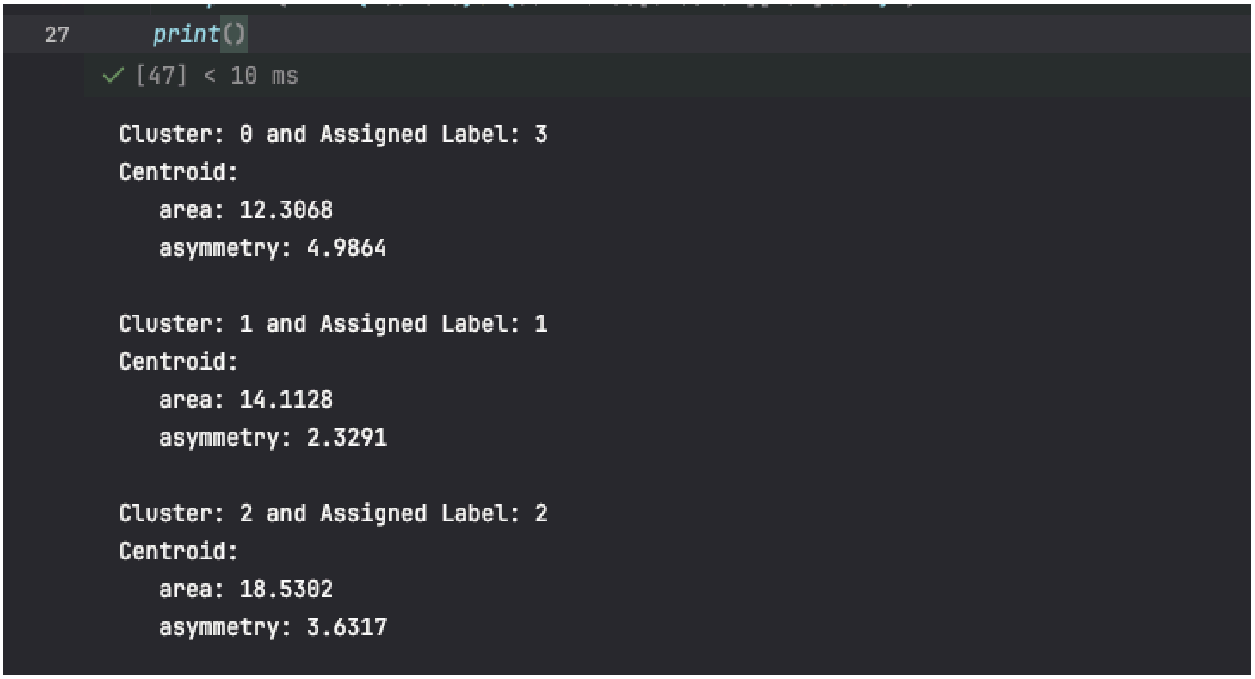
Question #3 Part 2



Patterns noticed:

Class 1 has smaller areas while class 2 has larger areas. Class 3 shows higher asymmetric values. There are some overlaps between clusters with larger areas. All the centroids seem to show the central tendencies accurately.

Question #3 Part 3



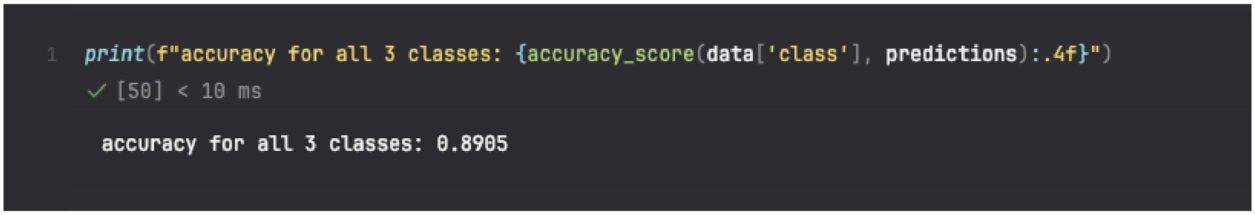
```
    print()
✓ [47] < 10 ms

  Cluster: 0 and Assigned Label: 3
  Centroid:
     area: 12.3068
     asymmetry: 4.9864

  Cluster: 1 and Assigned Label: 1
  Centroid:
     area: 14.1128
     asymmetry: 2.3291

  Cluster: 2 and Assigned Label: 2
  Centroid:
     area: 18.5302
     asymmetry: 3.6317
```

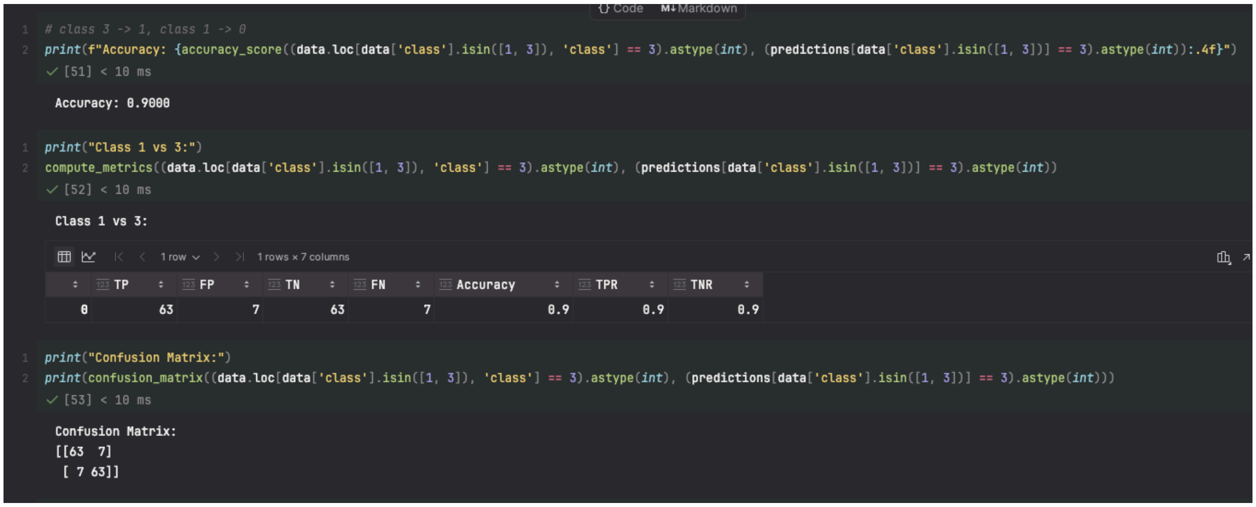
Question #3 Part 4



```
print(f"accuracy for all 3 classes: {accuracy_score(data['class'], predictions):.4f}")
✓ [50] < 10 ms

 accuracy for all 3 classes: 0.8905
```

Question #3 Part 5



```
# class 3 -> 1, class 1 -> 0
print(f"Accuracy: {accuracy_score((data.loc[data['class'].isin([1, 3]), 'class'] == 3).astype(int), (predictions[data['class'].isin([1, 3])] == 3).astype(int)):.4f}")
✓ [51] < 10 ms
```

```
Accuracy: 0.9000
```

```
print("Class 1 vs 3:")
compute_metrics((data.loc[data['class'].isin([1, 3]), 'class'] == 3).astype(int), (predictions[data['class'].isin([1, 3])] == 3).astype(int))
✓ [52] < 10 ms
```

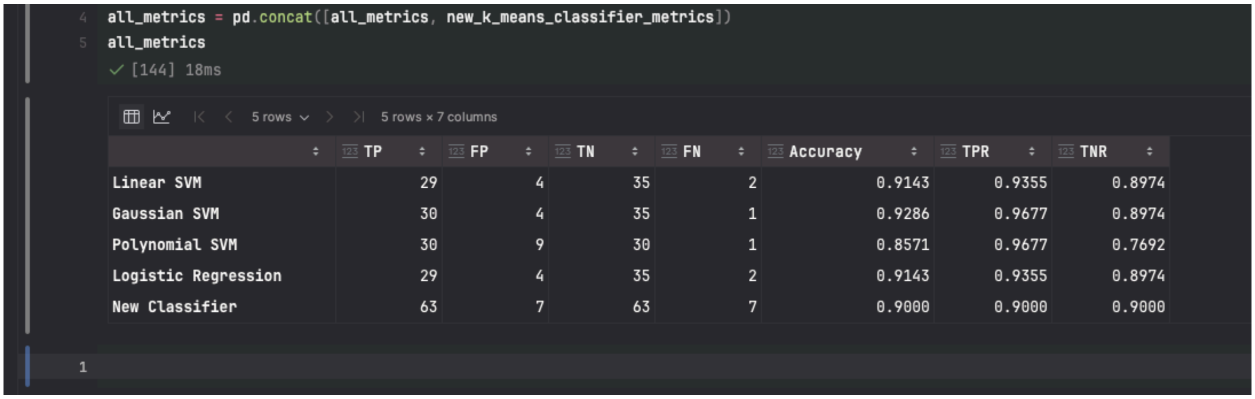
```
Class 1 vs 3:
```

1 row
1 rows × 7 columns

	TP	FP	TN	FN	Accuracy	TPR	TNR
0	63	7	63	7	0.9	0.9	0.9

```
print("Confusion Matrix:")
print(confusion_matrix((data.loc[data['class'].isin([1, 3]), 'class'] == 3).astype(int), (predictions[data['class'].isin([1, 3])] == 3).astype(int)))
✓ [53] < 10 ms
```

```
Confusion Matrix:
[[63  7]
 [ 7 63]]
```



```
all_metrics = pd.concat([all_metrics, new_k_means_classifier_metrics])
all_metrics
✓ [144] 18ms
```

5 rows
5 rows × 7 columns

	TP	FP	TN	FN	Accuracy	TPR	TNR
Linear SVM	29	4	35	2	0.9143	0.9355	0.8974
Gaussian SVM	30	4	35	1	0.9286	0.9677	0.8974
Polynomial SVM	30	9	30	1	0.8571	0.9677	0.7692
Logistic Regression	29	4	35	2	0.9143	0.9355	0.8974
New Classifier	63	7	63	7	0.9000	0.9000	0.9000

Our new classifier did not outperform Gaussian, Logistic Regression in terms of accuracy and TPR. It is also extremely balanced, hitting 0.90 for both TPR and TNR.