Capstone Project: Heatmap Anomaly Detection

Week 6 Progress Report

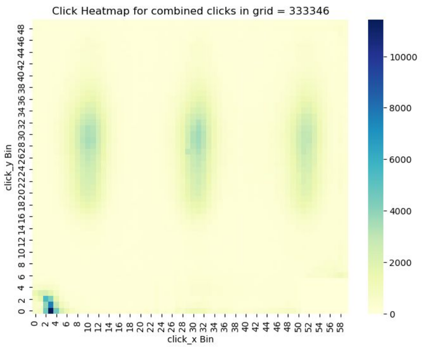
This week:

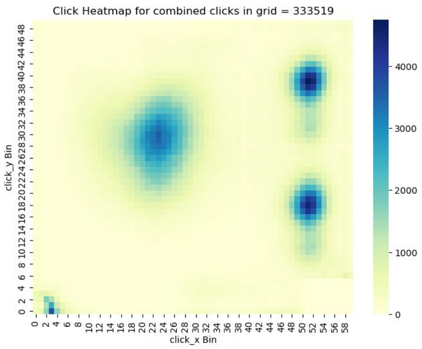
- Train/test split
- Basic methods → evaluations
- Further exploration of Metrics dataset
- PCA/Dimensional reduction

Recap:

Definition of “clearly broken grid:”

Triangle/line structure not visible even with noisy bootstrap enhancement. We do not care about rest of heatmap as long as this pattern is clearly defined.

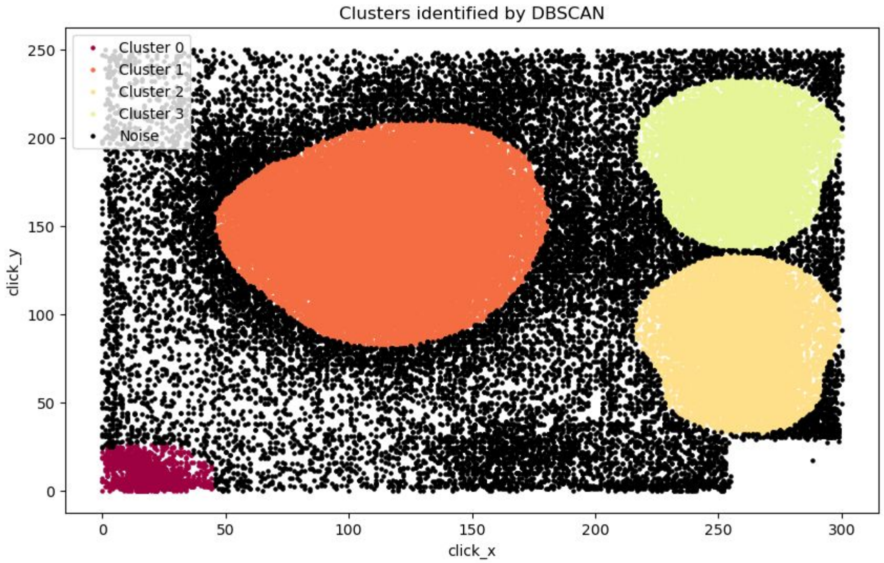




1. Created baseline of “clearly broken banners” for two grids:
   - “Triangle grid”: 57 clearly broken out of 872 → **~6.5%**
   - “Line grid”: 113 clearly broken out of 861 → **~13.5%**
   - Is it reasonable that there is such a discrepancy?

2. Added heatmap images for banners classified as “broken” to GitHub.

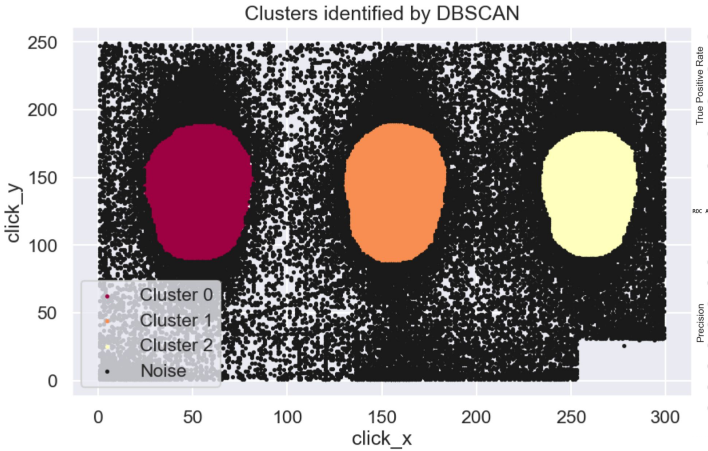
Click clustering method (recap):

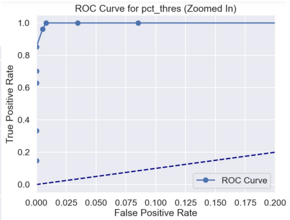


1. Bootstrap 100’000 clicks from fully aggregated dataset (filtered by grid_id).
2. Normalize (Standardize)
3. Run DBSCAN cluster with eps = .2 and min_samples = 1000
   → 4 clusters + noise.
4. For given (noisy bootstrap enhanced) domain, get 1-nn for each click in training data and select that label {0,1,2,3}
   a. If pct of points labelled as noise above a certain threshold → anomalous.
   b. Hypothesis testing: p_0 = pct of noise points in training data. H_0: p_0 < noise/total, H_A: p_0 >= noise/total → p-value larger than threshold (cannot reject null) → anomalous.

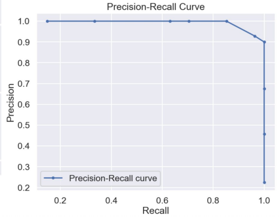
Click clustering method:

```
Epsilon: 0.13
Estimated number of clusters: 3
Estimated number of noise points: 37041
Pct of noise points: 0.370410
```





ROC AUC: 0.25

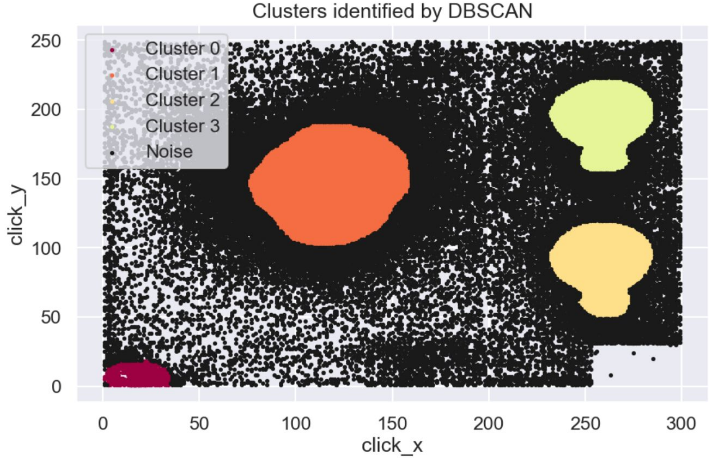


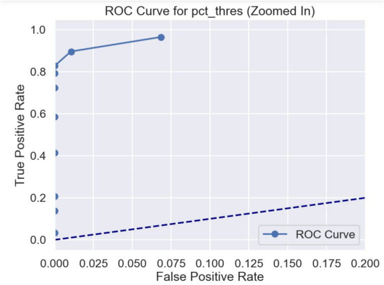

```
------------------------
Epsilon: 0.13 delta pct thres: 0.00 Pct_thres: 0.63

 Broken pct: 0.14, Total in CB: 27, Missed in CB: 1, Pct missed: 0.04, Not in CB: 2
confusion matrix (rate) ((TPR,FNR),(FPR,TNR)):
 ( 0.96 , 0.04 )
 ( 0.01 , 0.99 )
```

- **100k** bootstrapped samples to generate clusters.
- Use enhanced **5k** bootstrapped samples per domain.
  - The two grid's perform very well upon hyperparameter tuning:
    - Grid_id = 333346 performs best with eps = 0.13 and 1k min_sample

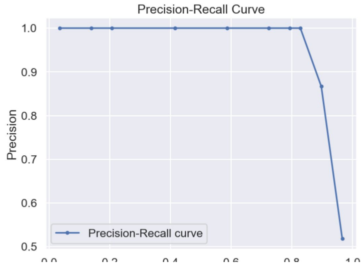
Click clustering method:

```
Epsilon: 0.13
Estimated number of clusters: 4
Estimated number of noise points: 38634
Pct of noise points: 0.521510
```





ROC AUC: 0.06




```
Epsilon: 0.15 delta pct thres: 0.00 Pct_thres: 0.57

 Broken pct: 0.06, Total in CB: 29, Missed in CB: 5, Pct missed: 0.17, Not in CB: 0
confusion matrix (rate) ((TPR,FNR),(FPR,TNR)):
 ( 0.83 , 0.17 )
 ( 0.00 , 1.00 )
```

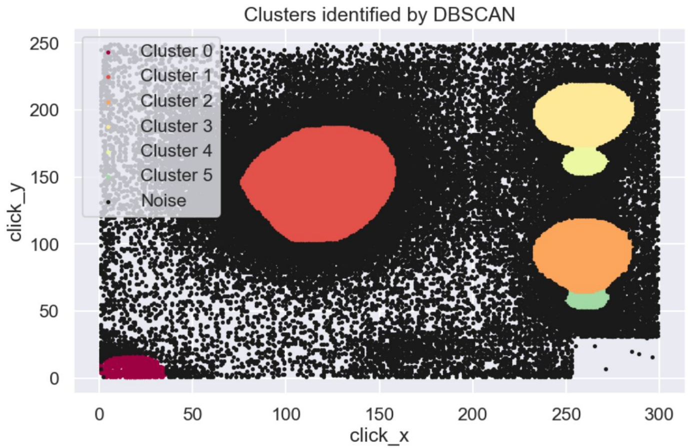
100k bootstrapped samples to generate clusters.

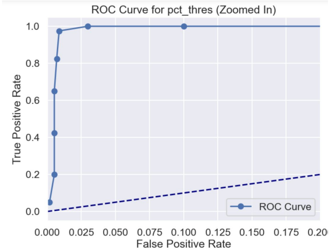
Use enhanced **5k** bootstrapped samples per domain.

- The two grid's perform very well upon hyperparameter tuning:
  - Grid_id = 333519 performs best with eps = 0.13 and 1k min_sample
  - We remove "corner clusters".

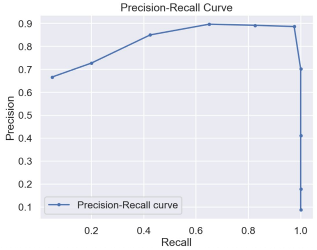
Performance across grid:

```
Epsilon: 0.13
Estimated number of clusters: 6
Estimated number of noise points: 39161
Pct of noise points: 0.391610
```






```
ROC AUC: 0.72
```



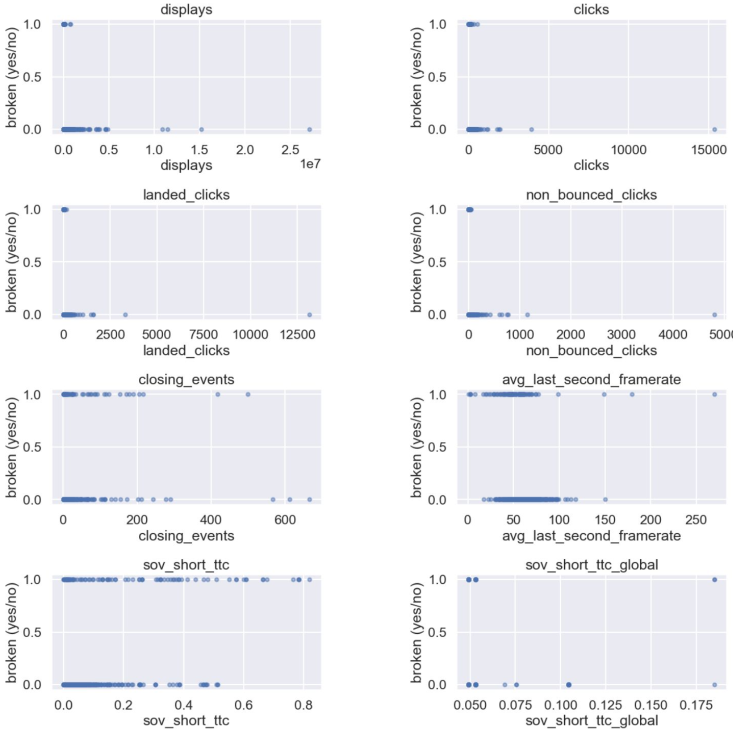

```
Epsilon: 0.13 Pct_thres: 0.64

 Broken pct: 0.07, Total in CB: 40, Missed in CB: 1, Pct missed: 0.03, Not in CB: 5
confusion matrix (rate) ((TPR,FNR),(FPR,TNR)):
 ( 0.97 , 0.03 )
 ( 0.01 , 0.99 )
```

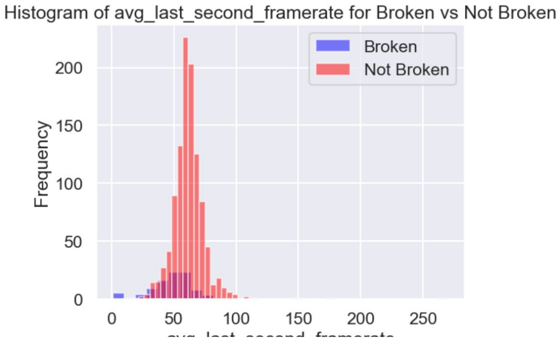
- Correspondingly, translating the same parameters from one to the does perform well:
  - 333346 → 333519:
    - **Remove corner cluster**:
      - **TPR**: 0.97 (1/40 missed)
      - **FNR**: 0.03 (5/600-ish)
    - Include corner cluster:
      - TPR: 0.8 (8/40 missed)
      - FNR: 0.0 (0/600-ish)
  - 333519 → 333346:
    -

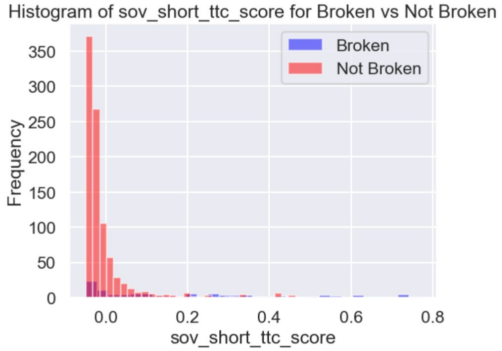
Generalization:

- **Zero-shot learning:** How many different banners are there overall and could we run baselines there, or is the aim to do “zero-shot learning”?
- Expect that the results will be consistent for 3-banner types, but thresholds will change depending on the number of products per banner.
- Crude but powerful baseline model → expect this to be hard to top.

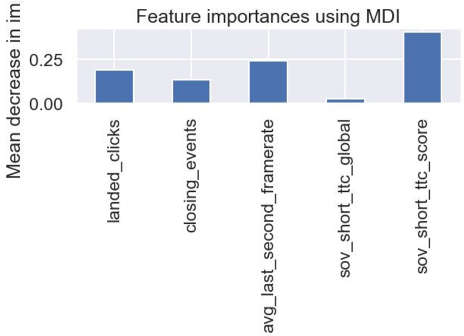
Metrics Dataset: Visualizations



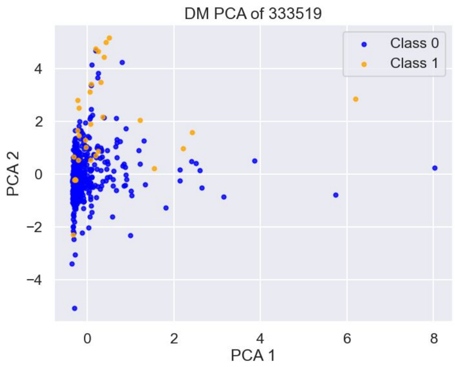


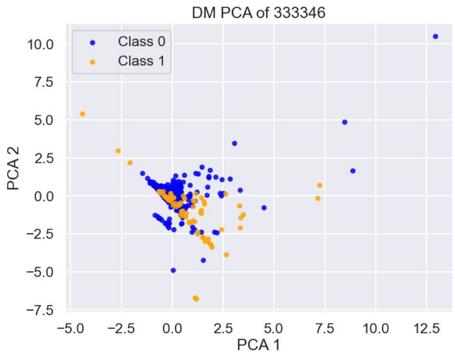


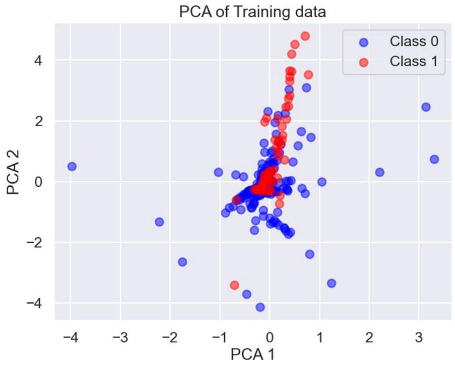
- Drop many collinear features. (e.g. Sov_short_ttc highly correlated with sov_short_ttc_score, etc)
- Compare distributions of broken vs non-broken
- Importance of features from Decision Tree methods:



Metrics Dataset: PCA

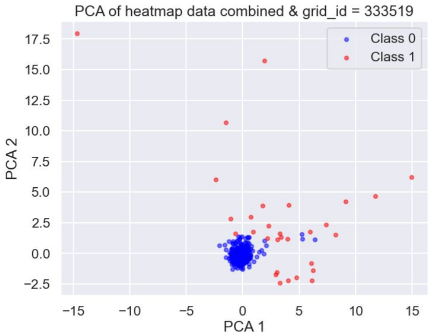


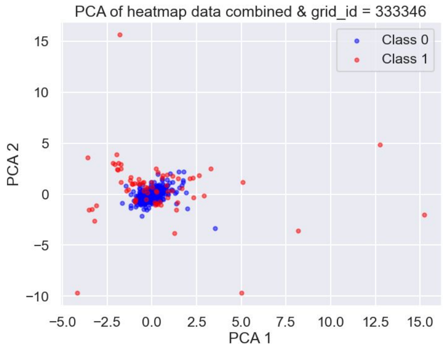


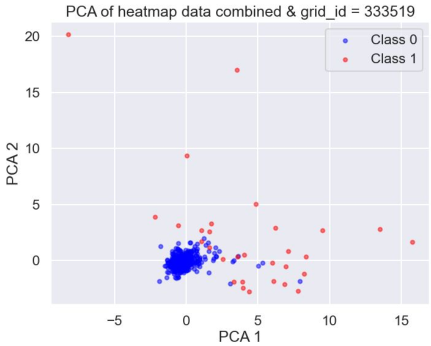


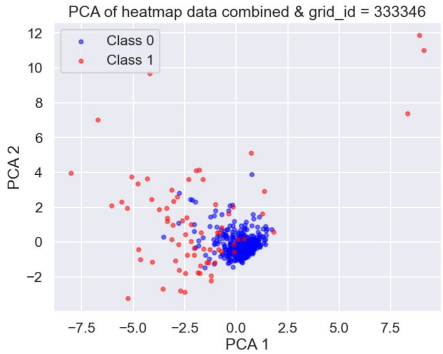
- Basic supervised classifiers (Decision Trees, Random Forest, Logistic Regression) don't perform well
  - Problem with unbalanced dataset and overfitting.
  - Grid search over model complexity (regularization).
  - Upsample/downsample → slight improvements but not competitive
  - Add interactions/higher-order features
- Consistently ~40 misclassifications (FP+FN) in test set from 31 positives and 320 negative instances.
- PCA less powerful but shows some structure.

Combined Datasets







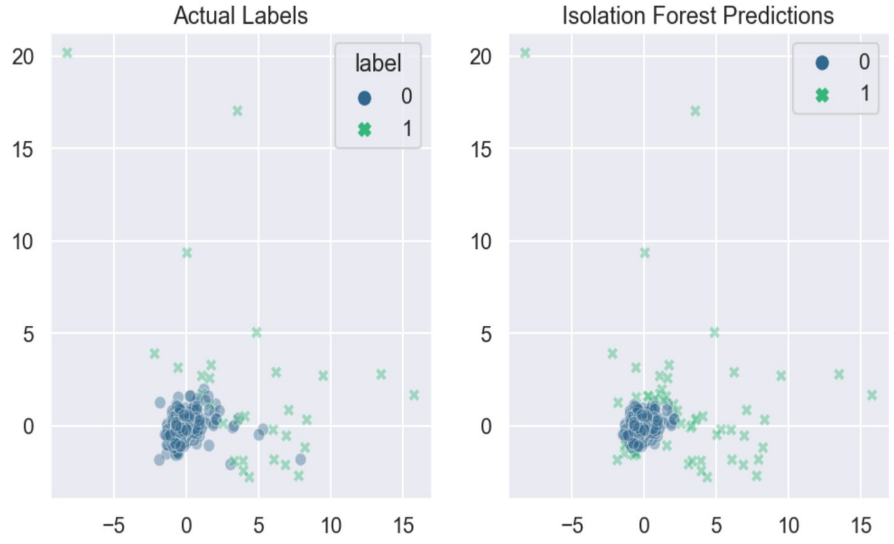


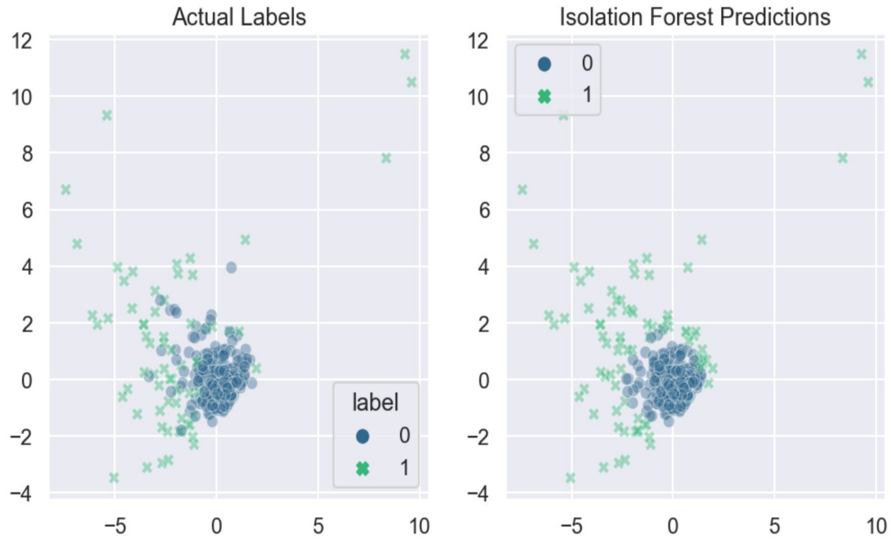
- Different methods of aggregating PCA's:
  - PCA on both individually → combine → scale → combined PCA (top)
  - PCA on HM only → combine → scale → combined PCA (bottom)
- Adding more components changes scores a little bit.

Combined Datasets: Clustering results

```
              precision    recall  f1-score   support

           0       1.00      0.95      0.98       559
           1       0.53      0.97      0.68        30

    accuracy                           0.95       589
   macro avg       0.76      0.96      0.83       589
weighted avg       0.97      0.95      0.96       589

Accuracy Score: 0.9541595925297114
```

```
              precision    recall  f1-score   support

           0       0.97      0.95      0.96       508
           1       0.70      0.76      0.73        72

    accuracy                           0.93       580
   macro avg       0.83      0.86      0.84       580
weighted avg       0.93      0.93      0.93       580

Accuracy Score: 0.9293103448275862
```





KNN - Grid 333519

- PCA: n=2 components
- Train/Test data split: 80% vs. 20%
- CV on KNN: best k = 5
- Accuracy results test on
  - test dataset
  - whole dataset

```
Accuracy: 0.9714
Confusion Matrix:
[[157   2]
 [  3  13]]
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.99      0.98       159
           1       0.87      0.81      0.84        16

    accuracy                           0.97       175
   macro avg       0.92      0.90      0.91       175
weighted avg       0.97      0.97      0.97       175
```

```
Accuracy: 0.9862
Confusion Matrix:
[[810   5]
 [  7  50]]
Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       815
           1       0.91      0.88      0.89        57

    accuracy                           0.99       872
   macro avg       0.95      0.94      0.94       872
weighted avg       0.99      0.99      0.99       872
```

KNN - Grid 333346

- PCA: n=2 components
- Train/Test data split: 80% vs. 20%
- CV on KNN: best k = 5
- Accuracy results test on
  - test dataset
  - whole dataset

```
Accuracy: 0.9827
Confusion Matrix:
[[145   3]
 [  0  25]]
Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.98      0.99       148
           1       0.89      1.00      0.94        25

    accuracy                           0.98       173
   macro avg       0.95      0.99      0.97       173
weighted avg       0.98      0.98      0.98       173
```

```
Accuracy: 0.9756
Confusion Matrix:
[[734  14]
 [  7 106]]
Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.98      0.99       748
           1       0.88      0.94      0.91       113

    accuracy                           0.98       861
   macro avg       0.94      0.96      0.95       861
weighted avg       0.98      0.98      0.98       861
```

Isolation Forest with PCA

- 80% Train, 20% Test
- Perform separately onto different grid
- With PCA, n= 2 components

Hyperparameters:

```
-   n_estimators=500,
-   max_features = 2000,
-   max_samples='auto',
-   contamination=0.1,
-   bootstrap = True
```

Grid 333519

```
Confusion Matrix:
 [[153  10]
 [  5   6]]
Accuracy: 0.9137931034482759
Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.94      0.95       163
           1       0.38      0.55      0.44        11

    accuracy                           0.91       174
   macro avg       0.67      0.74      0.70       174
weighted avg       0.93      0.91      0.92       174

True Positive Rate (TPR): 0.5454545454545454
True Negative Rate (TNR): 0.9386503067484663
```

Grid 333346

```
Confusion Matrix:
 [[145   4]
 [ 15   8]]
Accuracy: 0.8895348837209303
Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.97      0.94       149
           1       0.67      0.35      0.46        23

    accuracy                           0.89       172
   macro avg       0.79      0.66      0.70       172
weighted avg       0.87      0.89      0.87       172

True Positive Rate (TPR): 0.34782608695652173
True Negative Rate (TNR): 0.9731543624161074
```

One Class SVM with PCA

- 80% Train, 20% Test
- Perform separately onto different grid
- With PCA, n= 2 components

Hyperparameters:

- `nu=0.06,` sets an upper limit, expecting up to 5% of data points to be outliers
- `kernel='rbf',` employs the Radial Basis Function to handle non-linear data
- `gamma='auto',` adjusts gamma based on the number of features to prevent overfitting

Grid 333519

```
Confusion Matrix:
 [[163   0]
 [  5   6]]
Accuracy: 0.9712643678160919
Classification Report:
              precision    recall  f1-score   support

           0       0.97      1.00      0.98       163
           1       1.00      0.55      0.71        11

    accuracy                           0.97       174
   macro avg       0.99      0.77      0.85       174
weighted avg       0.97      0.97      0.97       174

True Positive Rate (TPR): 0.5454545454545454
True Negative Rate (TNR): 1.0
```

Grid 333346

```
Confusion Matrix:
 [[140   9]
 [  6  17]]
Accuracy: 0.9127906976744186
Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.94      0.95       149
           1       0.65      0.74      0.69        23

    accuracy                           0.91       172
   macro avg       0.81      0.84      0.82       172
weighted avg       0.92      0.91      0.92       172

True Positive Rate (TPR): 0.7391304347826086
True Negative Rate (TNR): 0.9395973154362416
```

K-means - Grid 333519

train_test_split

Apply pca with 2 components (w/o pca the result was terrible)

k=4 gains the best result (merge the 3 minority clusters to be considered as anomaly)

```
[[156   3]
 [ 11   5]]
              precision    recall  f1-score   support

           0       0.93      0.98      0.96       159
           1       0.62      0.31      0.42        16

    accuracy                           0.92       175
   macro avg       0.78      0.65      0.69       175
weighted avg       0.91      0.92      0.91       175

True Positive Rate: 31.25%  True Negative Rate: 98.11%
```

K-means - Grid 333346

k=6

```
[[144   4]
 [  0  25]]
              precision    recall  f1-score   support

           0       1.00      0.97      0.99       148
           1       0.86      1.00      0.93        25

    accuracy                           0.98       173
   macro avg       0.93      0.99      0.96       173
weighted avg       0.98      0.98      0.98       173

True Positive Rate: 100.00%  True Negative Rate: 97.30%
```

k=5

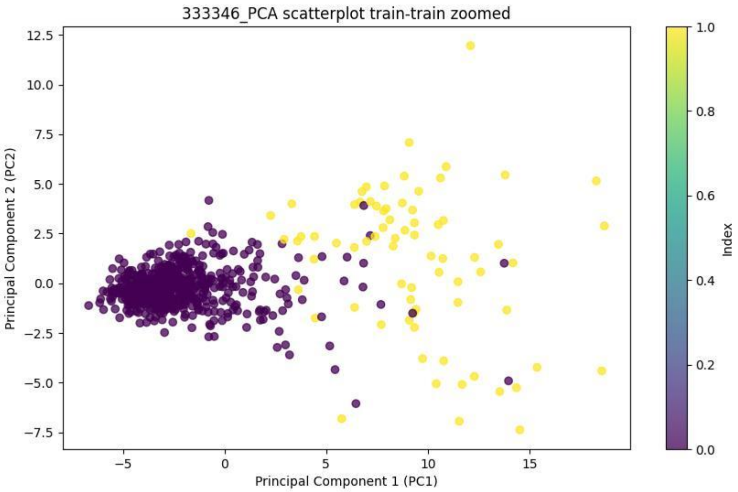
```
[[147   1]
 [  4  21]]
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       148
           1       0.95      0.84      0.89        25

    accuracy                           0.97       173
   macro avg       0.96      0.92      0.94       173
weighted avg       0.97      0.97      0.97       173

True Positive Rate: 84.00%  True Negative Rate: 99.32%
```

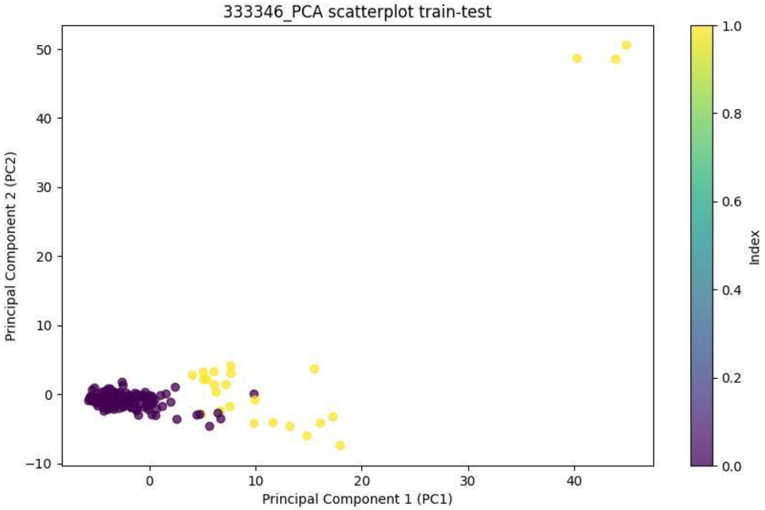
PCA clustering results

Tests:

- Splitting the broken banners dataset into train/test
- Training on 333519 and testing on 333346

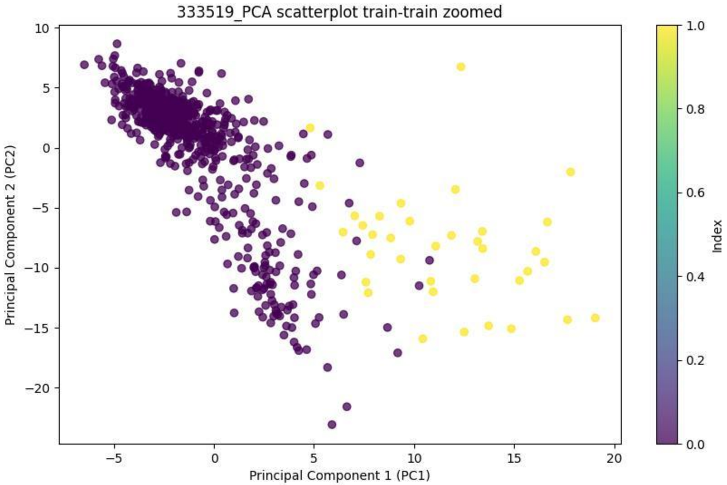
Grid_id: 333346

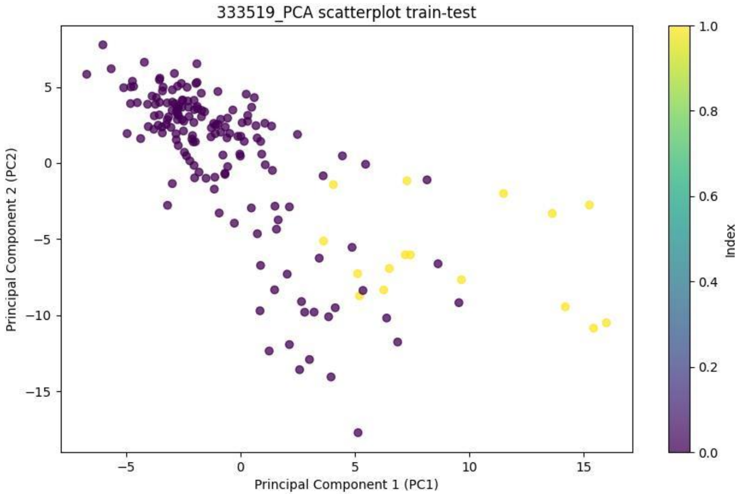




333346	Accuracy	Precision	Recall
Train-train	0.88	1.0	0.07
Train-test	0.86	0.0	0.0

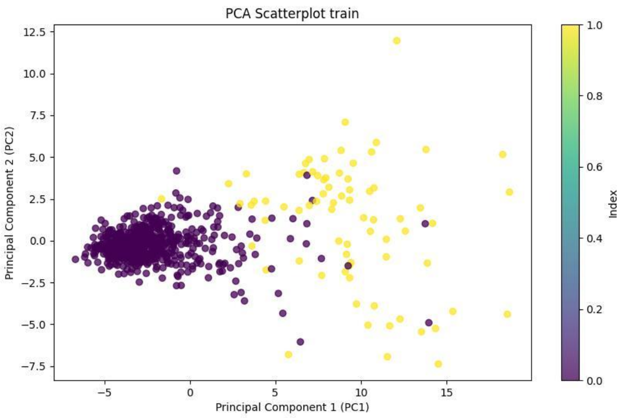
Grid_id: 333519



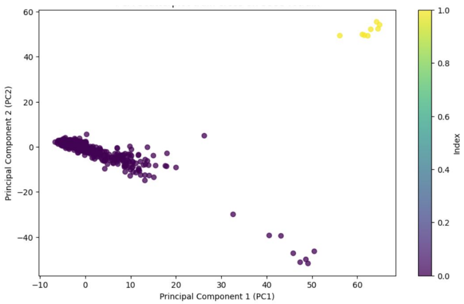


333519	Accuracy	Precision	Recall
Train-train	0.937	1.0	0.04
Train-test	0.908	0	0

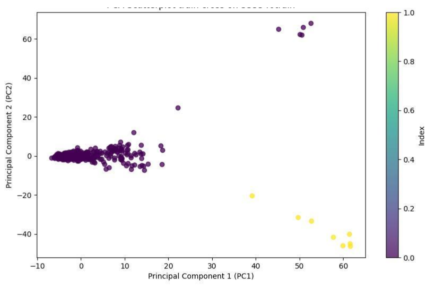
Train 333519 to classify 333346



333346/333346



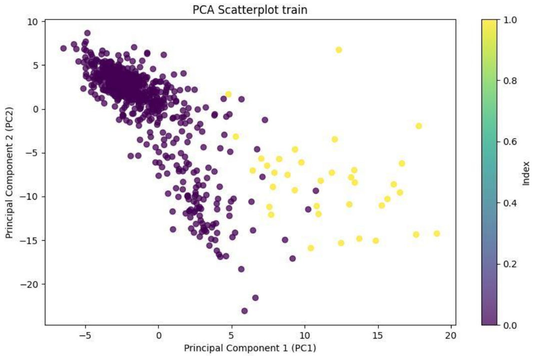
333519/333346



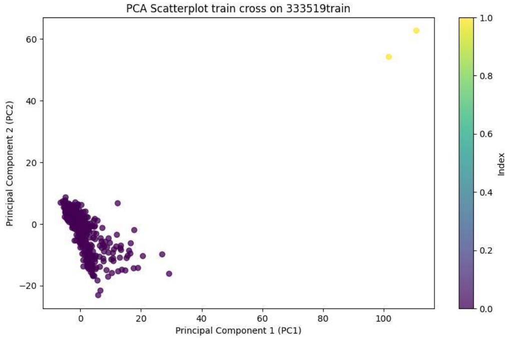
	Accuracy	Precision	Recall
333519/333346	0.89	1.0	0.10
333346/333346	0.88	1.0	0.07

Train 333346 to classify 333519

Ground truth



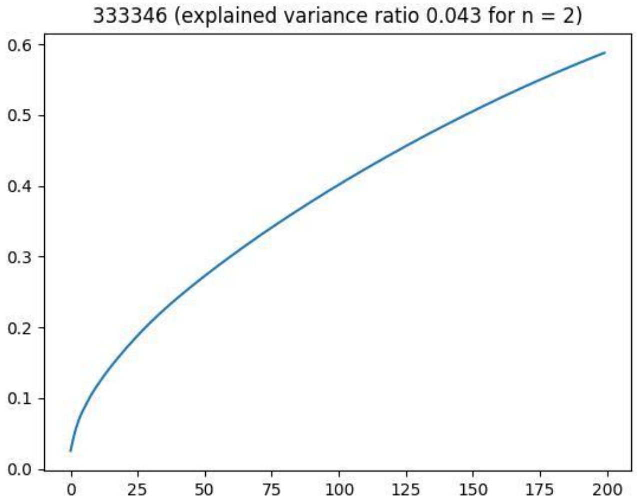
KNN predictions



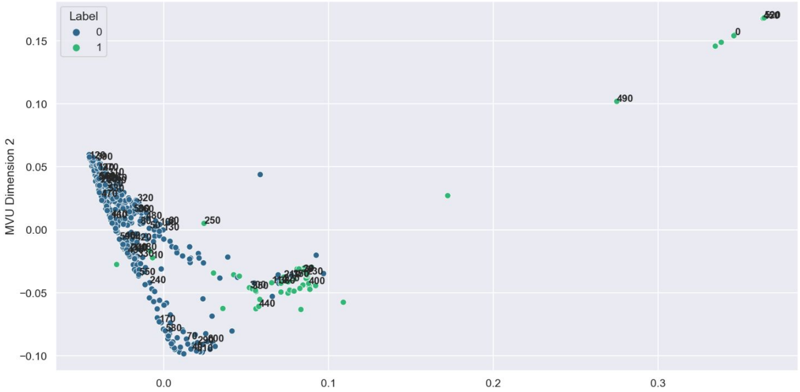
	Accuracy	Precision	Recall
333346/333519	0.94	1.0	0.05
333519/333519	0.937	1.0	0.04

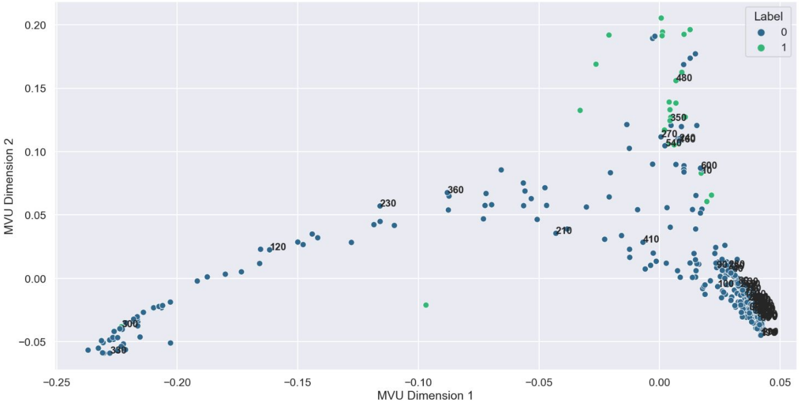
Conclusions and next steps

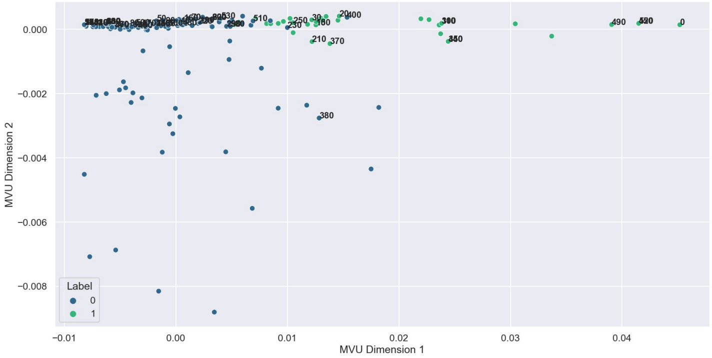
- Explained variance for all PCA plots for n = 2 is 4%
- Refine clustering algorithm



MVU – study “data manifold”:







- Study data manifold using Maximum Variance Unfolding/Semidefinite Embedding (MVU)
  - Intuition: create graph of close points in high dimensional space with distances
  - Use convex optimization to maximize distance between disconnected points s.t. connected points being nearby.

Next steps:

- Further exploration of “basic models”:
  - How does data enhancement affect them (different enhancements?)?
  - Ensemble techniques: can we boost basic methods (majority voting)?
  - Other grids?
  - Understand edge cases.
- Understand PCA:
  - What “boxes” are important in the first few components → can we understand these features?
- Leverage metrics/combined dataset more.
- “Productionize current models”
- Implement fancier method → instead of features being explicit bins/clicks, create more meaningful feature vectors:
  - Use pretrained ResNet/ViT/… feature extractors and run clustering on feature vectors of heatmaps
  - Train Auto-Encoder on arbitrary synthetic clusterings (ask AE to recreate original image with discriminative loss).
  - Train/Fine-tune ResNet/ViT/… on synthetic data to count number of clusters → might lead to interesting feature vectors.