Applied Analytics Project - Week 7 Assignment
Develop Second Modeling Approach

Brady Levenson
ADAN 8888 - Applied Analytics Project
Professor Nurtekin Savas
October 20, 2024

Revisitation of Problem Statement for Motivation to Perform Modeling

It is known that fraudulent financial transactions occur on a regular basis, but it is difficult to distinguish between them and legitimate financial transactions simply by viewing the transactions. The inability to correctly classify fraudulent financial transactions can promote uncontrolled and untraceable levels of financial losses for banks, investment firms, corporations, and numerous other entities. We are given a dataset of financial transactions and their unique properties. We will analyze this dataset using various machine learning techniques to determine effective measures to identify historical fraudulent transactions and prevent their occurrence in the future.

Dataset Structure Update

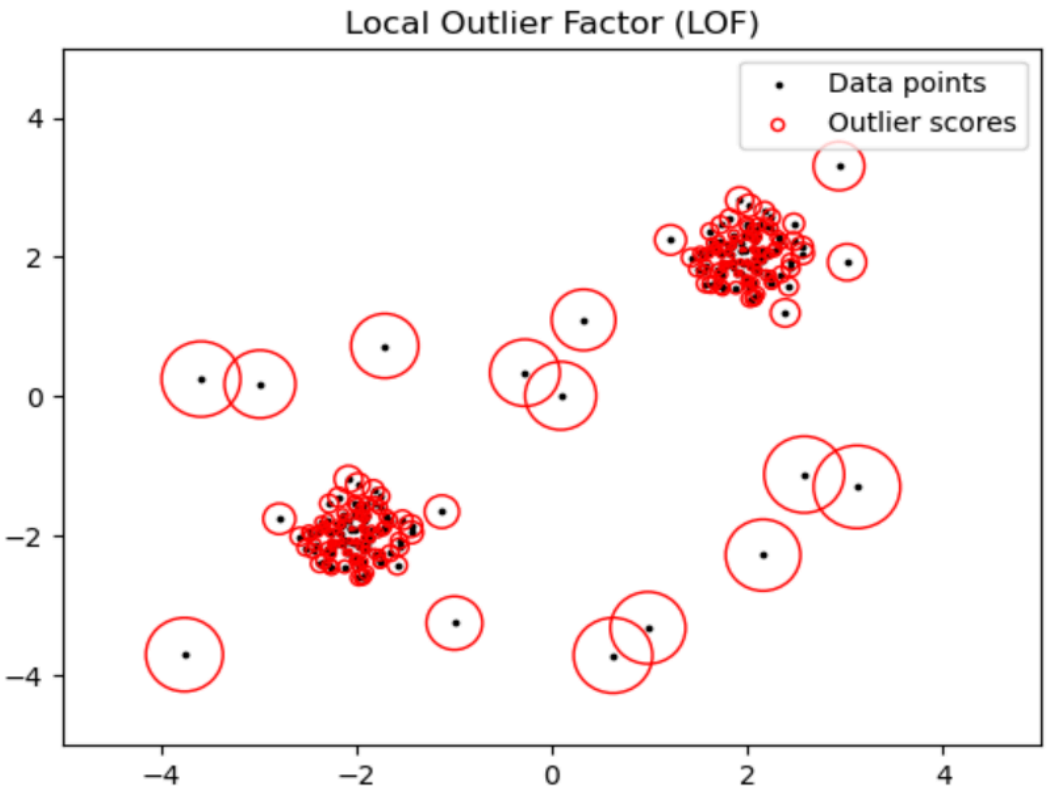
My dataset's dimensionality has been reduced all the way down to 16 unique columns, namely .Timestamp, Amount (log normalized), Date, Day of the Week (7 columns, one-hot encoded), Deviation From Mean (normalized), Account (embeddings), Merchant (embeddings), Transaction Type (embeddings), Location (embeddings), and Hour Group (embeddings). I am happy with the current state of my dataset and believe all of the transformations I have performed on it up to this point will result in each of my models producing viable results that could ultimately be applied in industry.

Revisitation of First Model Approach

My first model approach for this project was an unsupervised K-Means Clustering Model. There are a couple reasons why I decided this modeling approach may be appropriate for this task. Firstly, the structure of my chosen dataset indicates that this problem should be solved in an unsupervised manner. While the target variable of this project is the binary classification of yes, a financial transaction is an anomaly, or no, a financial transaction is not an anomaly, we do not have a clearly defined labeled training set that our models can learn from directly. Thus, we needed to exercise a more elaborate approach. The K-Means Clustering technique is great for anomaly detection tasks as it allows for comparisons to be made between financial transactions within the same dataset, specifically aiding in the determination of which groups of transactions reside farthest away from the rest. This can help us to identify patterns of data that could potentially represent fraudulent transactions, which can then be isolated for further evaluation.

Second Model Approach (this week)

My second model for this project is an unsupervised Local Outlier Factor (LOF) Model. The primary benefit I see with the LOF model over my K-means model is its ability to capture the local densities of points within the context of the financial transaction dataset. This is a factor that my K-means model would have more trouble considering as this technique mainly knows that a point is located within a certain cluster, but does not capture the relative volume of points within specifically compacted spaces. By considering the local density of data points in our dataset, we can determine which points are more isolated relative to others, which may result in those most isolated points being classified as outliers in the dataset. I believe the complexity introduced by this model will present a more trustworthy view of which financial transactions in my dataset may represent outliers, whereas my K-means model may suggest transactions with a relatively lower level of confidence.



Sample Diagram of Local Outlier Factor Classification

Complexity of Modeling Approach

Following my projected upward progression of modeling approach complexity, this model is the second most complex of the three I have planned, and I would say it has a moderate level of complexity. A portion of the complexity of this model comes from the feature engineering that I conducted in the past few weeks, though the model standing alone still possesses some complexity that is important to be addressed. This model requires calculating the distance between every pair of data points present in the dataset, so given the raw size of the data we are considering for this project and the high level transformations we have performed on it, the Local Outlier Factor Modeling Approach possesses noteworthy complexity. Additionally, adjusting hyperparameter values can also increase or decrease the base complexity of the model, as we can perform hyperparameter tuning to discover setups that work the best for the specific feature set at hand.

Hyperparameters Evaluated

n_neighbors- number of neighbors

- The number of nearest neighboring data points considered in the model can determine the local densities of certain data points e.g. too few neighbors = poor generalization and over classification of outliers; too many neighbors = smoothing of local variations that may result in false negative classifications..

leaf_size- number of nodes per decision tree

- Results in varying convergence speeds and levels of memory usage. Adjusting this hyperparameter did not impact my models' results, so I will not include it in the results analysis.

metric- method for distance calculation

- Two metrics were used for calculating the distance between two points in the dataset.
    - **Euclidean Distance**- straight line distance between two points in euclidean space.
    - **Manhattan Distance**- distance between two points calculated by only moving along grid lines.

algorithm- method for nearest neighbor search

- I am using **auto** for this hyperparameter, which automatically selects the best algorithm for nearest neighbor search based on dimensionality and input data structure, but here are the options that algorithm has to choose from:
    - **ball_tree-** best for high dimensional data.
    - **kd_tree-** best for low dimensional data.

- **brute-** computes distances directly regardless of dimensionality.

Model Performance Metrics

Silhouette Score: Measures a transaction's similarity to its own cluster versus others. This metric can be leveraged indirectly to determine if certain points are poorly clustered and may be more likely to be anomalies. Lower scores indicate poorer clustering, which indicates greater anomaly potential. Higher scores indicate better clustering, which may make it easier to distinguish anomaly transactions from the well-clustered normal transactions.

Calinski-Harabasz Index- Ratio between the ratio of the sum of between-cluster dispersion to within cluster dispersion. This index can be used to determine how well-separated our clusters of data are given the modeling approach and hyperparameter specifications in effect. Higher scores indicate well-separated clusters that make it easier to distinguish anomaly transactions from normal transactions given how well normal transactions are clustered together in this scenario.

Davies-Bouldin Index- Average similarity ratio between each cluster and its most similar cluster. Higher scores indicate clusters that are very similar to one another and difficult to differentiate, while lower scores point to clusters that are more well-separated and obviously-defined. We are aiming for lower scores for this index to identify good clustering that will give us more confidence to classify data points that stray away from the characteristics of well-clustered data as anomalies.

Metric Calculations Results Table

	Variation 1.1 Silhouette	Variation 1.1 Calinski-Harabasz Index	Variation 1.1 Davies-Bouldin Index
Training	0.0996	88.1241	4.7576
Validation	0.2972	64.4077	3.668

	Variation 2.1 Silhouette	Variation 2.1 Calinski-Harabasz Index	Variation 2.1 Davies-Bouldin Index
Training	-0.0267	198.7778	5.5308
Validation	0.4804	143.9189	2.0772

	Variation 3.1 Silhouette	Variation 3.1 Calinski-Harabasz Index	Variation 3.1 Davies-Bouldin Index
Training	0.0713	162.3979	4.5217
Validation	0.5046	194.8525	1.6319

	Variation 1.2 Silhouette	Variation 1.2 Calinski-Harabasz Index	Variation 1.2 Davies-Bouldin Index
Training	-0.014	30.7389	9.3945
Validation	0.0949	5.5474	10.6232

	Variation 2.2 Silhouette	Variation 2.2 Calinski-Harabasz Index	Variation 2.2 Davies-Bouldin Index
Training	-0.0595	182.6127	5.6467
Validation	0.3006	21.2976	5.0632

	Variation 3.2 Silhouette	Variation 3.2 Calinski-Harabasz Index	Variation 3.2 Davies-Bouldin Index
Training	0.0234	113.4173	5.4388
Validation	0.3937	51.1523	3.4516

Variation Specifications

Variation 1.1: n_neighbors=5, metric=euclidean

Variation 2.1: n_neighbors=10, metric=euclidean

Variation 3.1: n_neighbors=15, metric=euclidean

Variation 1.2: n_neighbors=5, metric=manhattan

Variation 2.2: n_neighbors=10, metric=manhattan

Variation 3.2: n_neighbors=15, metric=manhattan

Considering Factors for Hyperparameter Tuning

I conducted my hyperparameter tuning to gain perspectives of varying levels of nearest neighbor consideration to determine what local density specifications this dataset would benefit best from. I

created models to consider 5, 10, and 15 neighbors respectively to capture this factor. For each of those three neighbor levels, I used euclidean and manhattan metrics to calculate the distance between neighboring transactions to see which metric is most effective at clustering the data well.

Top 3 Train Set Performance Metric Rankings (best to worst):
Silhouette Score: Variation 1.1, Variation 3.1, Variation 3.2
Calinski-Harabasz Index: Variation 2.1, Variation 2.2, Variation 3.1
Davies-Bouldin Index: Variation 3.1, Variation 1.1, Variation 1.2

Validation Dataset Performance Rankings:
Silhouette Score: Variation 3.1, Variation 2.1, Variation 3.2
Calinski-Harabasz Index: Variation 3.1, Variation 2.1, Variation 1.1
Davies-Bouldin Index: Variation 3.1, Variation 2.1, Variation 3.2

My Variation 3.1 Model exhibited high Silhouette and Calinski Harabasz Scores as well as low Davies-Bouldin scores when faced with validation set data, indicating that it may be the best equipped of the variations produced here today at identifying potentially fraudulent financial transaction due to their anomaly status within the dataset.

Week 7 Winning Model
When considering all variations across both training and validation data, I select Variation 3.1 as my winning model of the week. In 4 out of 6 performance categories, this model placed first amongst the six unique variations with a narrow second place finish in train set Silhouette Score and a third place finish in Calinski-Harabasz Index. Based on the statistics calculated in this portion of the project, I would expect Variation 3.1's hyperparameter combination of 15 neighbors and euclidean distance calculation to perform the best on an unseen test set. Since 5, 10, and 15 neighbors were the only n_neighbors values incorporated in this week's experimentation, it is still possible that a higher value for this hyperparameter could work more effectively. If it turns out to be a difficult process still to classify anomaly transactions using Model 3.1, it may be beneficial to return to this week's modeling and test out larger values for n_neighbors.

Next Steps

Next week, I will be developing my next modeling approach, which will be the third tier of complexity of my three models that will be produced by the end of this project. By the conclusion of next week, it should be fairly apparent what level of model complexity works most effectively on a dataset of the caliber of the one that we have crafted for the purposes of this financial anomaly detection task.