Here is the README file for the **K-means Based Anomaly Detection** project in the requested format:

K-means Based Anomaly Detection

Overview

This project implements **Anomaly Detection using K-means**, a clustering-based unsupervised learning algorithm. The objective is to detect anomalies by clustering the data points and using the distance to the nearest centroid to identify outliers. Anomalous points are defined as those farthest from their assigned centroids.

Key Features

1. **Data Loading**:
    - The dataset is loaded using **pandas**, which allows for efficient handling and manipulation of the data.
2. **Feature Selection**:
    - Relevant features for anomaly detection are selected from the dataset.
3. **K-Means Clustering**:
    - **K-means clustering** is applied to partition the dataset into clusters.
    - The algorithm assigns each point to the nearest centroid.
4. **Anomaly Detection**:
    - Anomalies are detected by calculating the distance of each point to its assigned centroid. Points with distances above a defined threshold are labeled as anomalies.
5. **Visualization**:
    - The results are visualized using **Matplotlib**, where normal points are plotted in blue and anomalies in red.

How It Works

1. **Data Loading**:
    - The dataset is loaded using **pandas** `read_csv` function, and relevant features are selected for anomaly detection.
2. **K-Means Clustering**:
    - **K-means** is applied to the selected features. The number of clusters is predefined (in this case, 3).
    - Each point is assigned to one of the clusters.
3. **Distance Calculation**:
    - The distance of each data point to the nearest centroid is computed using **pairwise distances**.
4. **Anomaly Definition**:
    - A threshold is defined as the 95th percentile of the calculated distances.
    - Points with distances greater than this threshold are considered anomalies.

Code Walkthrough

1. **Data Loading and Feature Selection**:

```
data = pd.read_csv('your_dataset.csv')
X = data[['feature1', 'feature2']]  # Replace with relevant feature columns
```

2. **K-Means Clustering**:

```
kmeans = KMeans(n_clusters=3, random_state=42)
data['cluster'] = kmeans.fit_predict(X)
```

3. **Distance Calculation**:

```
distances = pairwise_distances_argmin_min(X, kmeans.cluster_centers_)[1]
```

4. **Anomaly Detection**:
   - Anomalies are identified based on a distance threshold:

```
threshold = np.percentile(distances, 95)
data['anomaly'] = (distances > threshold).astype(int)
```

5. **Visualization**:

```
plt.scatter(X['feature1'], X['feature2'], label='Normal', c='blue', s=20)
plt.scatter(anomalies['feature1'], anomalies['feature2'], label='Anomaly', c='red',
plt.title('k-Means-Based Anomaly Detection')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.legend()
plt.show()
```

Advantages

- **Simple and Intuitive**: K-means is easy to implement and understand.
- **Scalable**: Works well with large datasets and can be adapted for high-dimensional data.
- **No Assumptions About Data**: Unlike other anomaly detection methods, K-means does not make strong assumptions about data distribution.

Future Work

- **Cluster Number Optimization**: Experiment with different values for the number of clusters (`n_clusters`).
- **Advanced Distance Metrics**: Use other distance metrics (e.g., Mahalanobis distance) to improve anomaly detection.

- **Comparison with Other Methods**: Compare the performance of K-means with other anomaly detection methods, such as Isolation Forest or DBSCAN.

References

- K-Means Clustering - scikit-learn documentation
- Anomaly Detection using K-means