ANOMALY MODELS

Dr. Aric LaBarr
Institute for Advanced Analytics

Course Layout

Anomaly Models

- Univariate Analysis
- Clustering
- Isolation Forests
- CADE

Fraud Supervised Models

- SMOTE
- Classification Models
- Labeled vs. Unlabeled Bias
- Not Fraud Model
- Model Evaluation

Clusters of Not Goods

- Cluster Analysis
- Social Network Analysis

Implementation

- Investigators
- Traffic Light Indicators
- Backtesting

Fraud Maturity

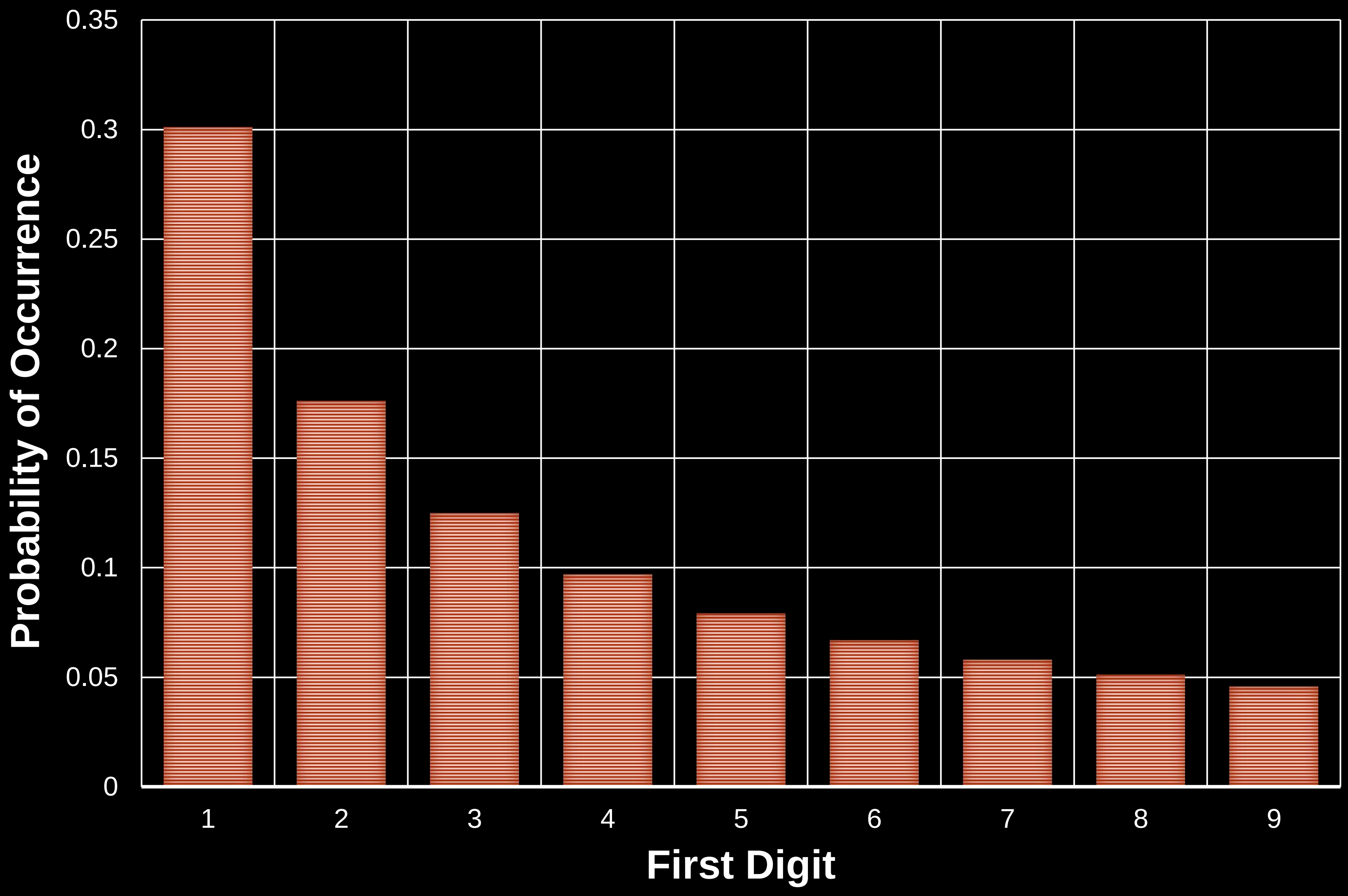
Components	New / Young	Emerging SIU	Fraud Scoring	Holistic Solution
Simple Rules	Yes	Yes	Yes	Yes
Unlabeled Data	Yes / No	Yes / No	Yes	Yes
Labeled Fraud Cases	No	Yes	Yes	Yes
Anomaly Models	No	Yes / No	Yes	Yes
Supervised Models	No	No	Yes	Yes
Non-Fraud Models	No	No	No	Yes
Clusters of not Good	No	No	No	Yes

NON-STATISTICAL TECHNIQUES

Benford's Law

- Certain numbers do not occur uniformly despite what we might think.
- Digits of certain numbers follow Benford's Law.
- Example:
  - First digit of house/building numbers in addresses.
  - First digit of transaction amounts.

Benford's Law



Benford's Law

- This wasn't mathematically proven until the mid-90's.
- http://testingbenfordslaw.com/

- Benford's Law – First Digit

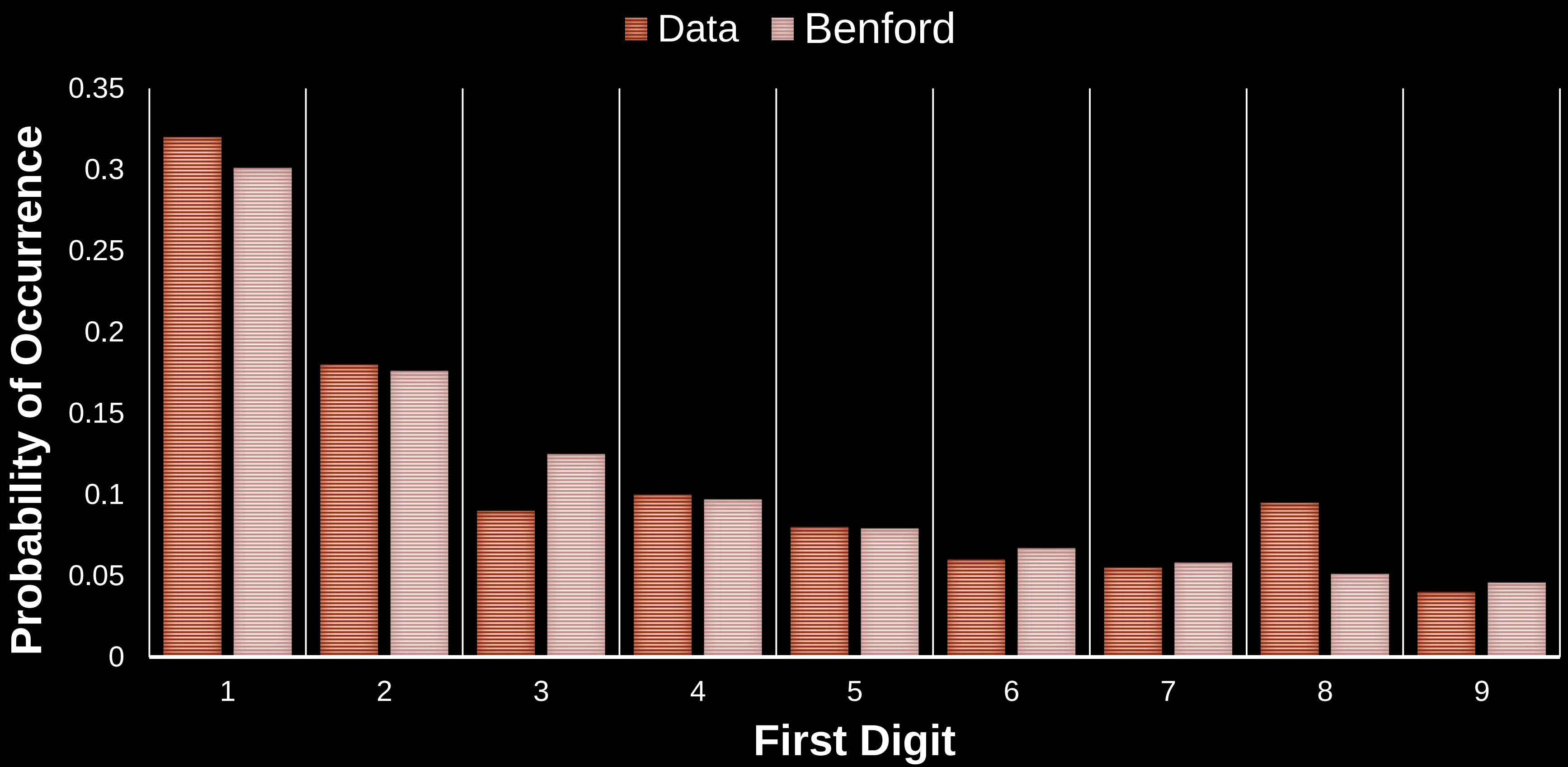
$$P(d_1) = \log_{10}\left(1 + \frac{1}{d_1}\right)$$

Benford’s Law – Fraud Detection

- Fraud transactions typically involve inventing new numbers or changing real transactions into fraudulent ones.
- Legally admissible in Federal, State, and Local courts in United States as evidence.

Benford's Law – Fraud Detection

- Example transaction amounts submitted for reimbursement from scanned receipts



Benford's Law

- Fraud detection typically uses the first two digits in Benford's Law.
- Benford's Law – First Two Digits

$$P(d_1 d_2) = \log_{10}\left(1 + \frac{1}{d_1 d_2}\right)$$

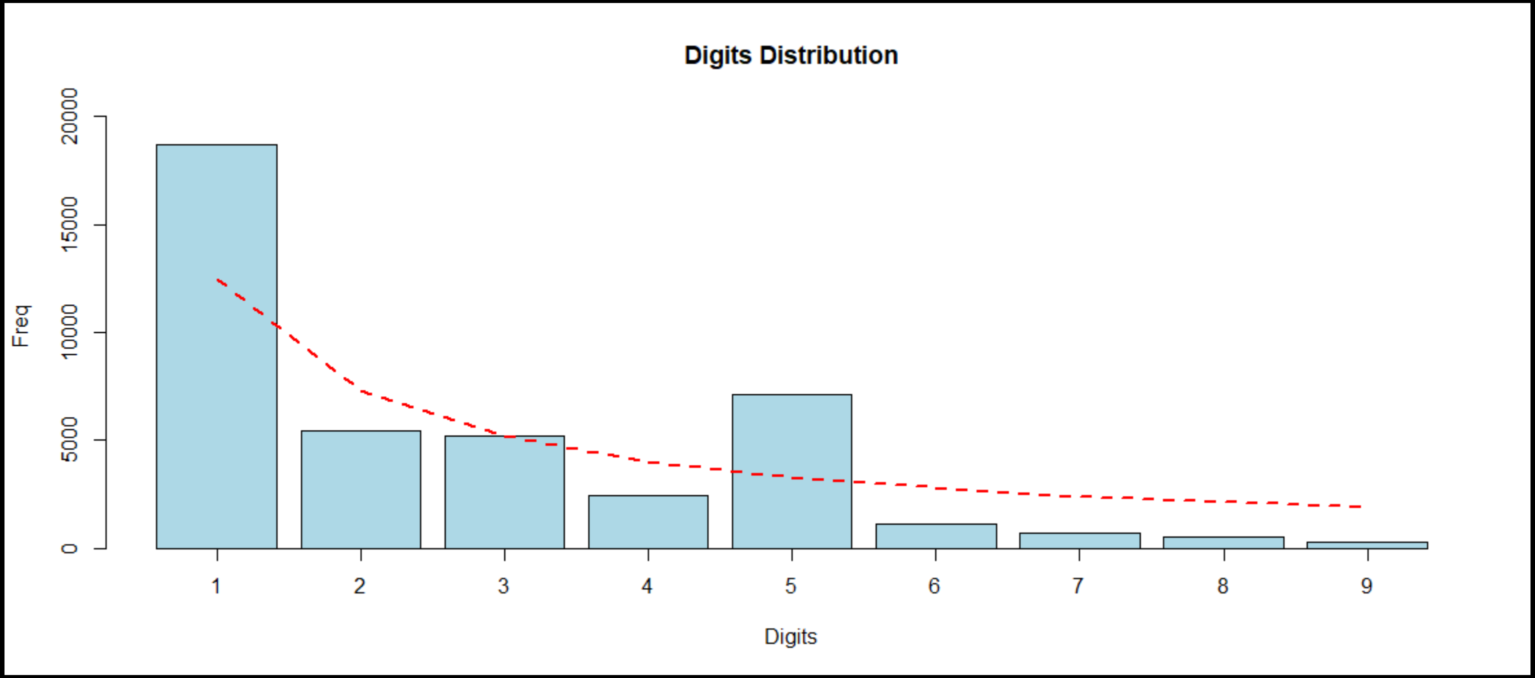
$$d_1 d_2 \in [10,11,12,13, \dots, 99]$$

Benford's Law – R

```
# Benford's Law #
rewarddigit1 <- as.numeric(substring(ins$Reward_Amount, first = 1, last = 1))
ben <- benford(rewarddigit1, number.of.digits = 1)

plot(ben, multiple = FALSE)
chisq(ben)
```

Benford's Law – R

```
# Benford's Law #
rewarddigit1 <- as.numeric(substring(ins$Reward_Amount, first = 1, last = 1))
ben <- benford(rewarddigit1, number.of.digits = 1)

plot(ben, multiple = FALSE)
chisq(ben)
```



Benford's Law – R

```
# Benford's Law #
rewarddigit1 <- as.numeric(substring(ins$Reward_Amount, first = 1, last = 1))
ben <- benford(rewarddigit1, number.of.digits = 1)

plot(ben, multiple = FALSE)
chisq(ben)
```

```
        Pearson's Chi-squared test

data:  rewarddigit1
X-squared = 13511, df = 8, p-value < 2.2e-16

>
```

?

UNIVARIATE ANALYSIS

Outliers

Statistical Methods

- Basic fraudulent systems look for abnormal observations from a statistical standpoint.
- Univariate analysis can help identify fraudulent **transactions** or **people** (aggregated transactions).

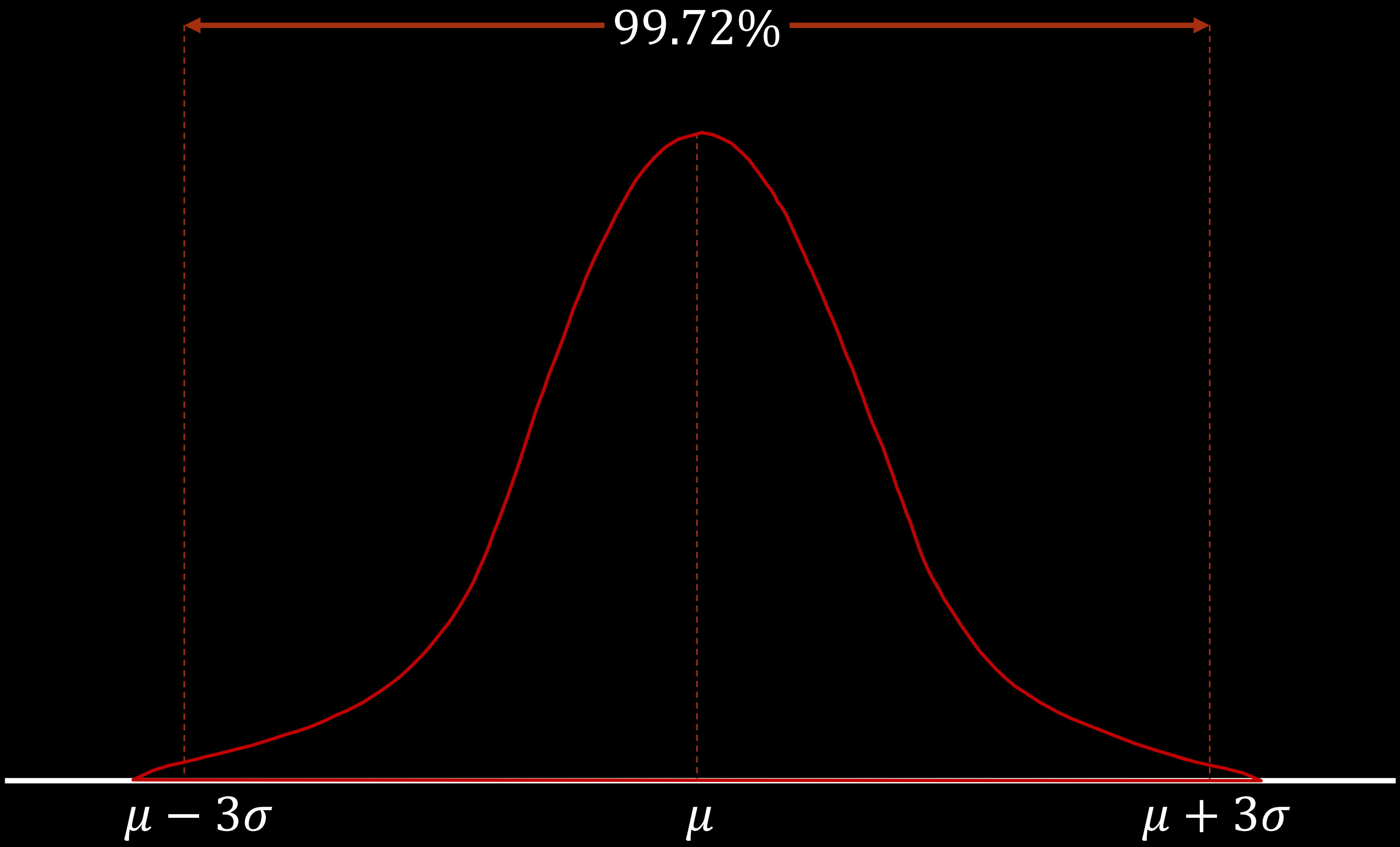
Z-Scores

- Typical with Normal distributions.

$$z_i = \frac{x_i - \bar{x}}{s}$$

- Measures how many standard deviations away from mean each point is.
- Works best with **symmetric** distributions.

Empirical Rule



Z-Scores

- Typical with Normal distributions.

$$z_i = \frac{x_i - \bar{x}}{s}$$

Bothered by outliers

- Measures how many standard deviations away from mean each point is.
- Works best with **symmetric** distributions.

Robust Statistics

- Outliers can greatly influence results.
- Robust techniques
  1. Reliable when outliers present
  2. Reliable when outliers **not** present (ideally)

Robust Z-Scores

- Robust adjustments to mean and standard deviation.

$$z_{R,i} = \frac{x_i - \mathrm{median(x)}}{\mathrm{MAD}(x)}$$

- Median Absolute Deviation (MAD):

$$\mathrm{MAD}(x) = k \times \mathrm{median}(|x_i - \mathrm{median}(x)|)$$

Robust Z-Scores

- Robust adjustments to mean and standard deviation.

$$z_{R,i} = \frac{x_i - \mathrm{median(x)}}{\mathrm{MAD}(x)}$$

- Median Absolute Deviation (MAD):

$$\mathrm{MAD}(x) = k \times \mathrm{median}(|x_i - \mathrm{median}(x)|)$$

Adjustment factor per distribution

Robust Z-Scores

- Robust adjustments to mean and standard deviation.

$$z_{R,i} = \frac{x_i - \mathrm{median(x)}}{\mathrm{MAD}(x)}$$

- Median Absolute Deviation (MAD):

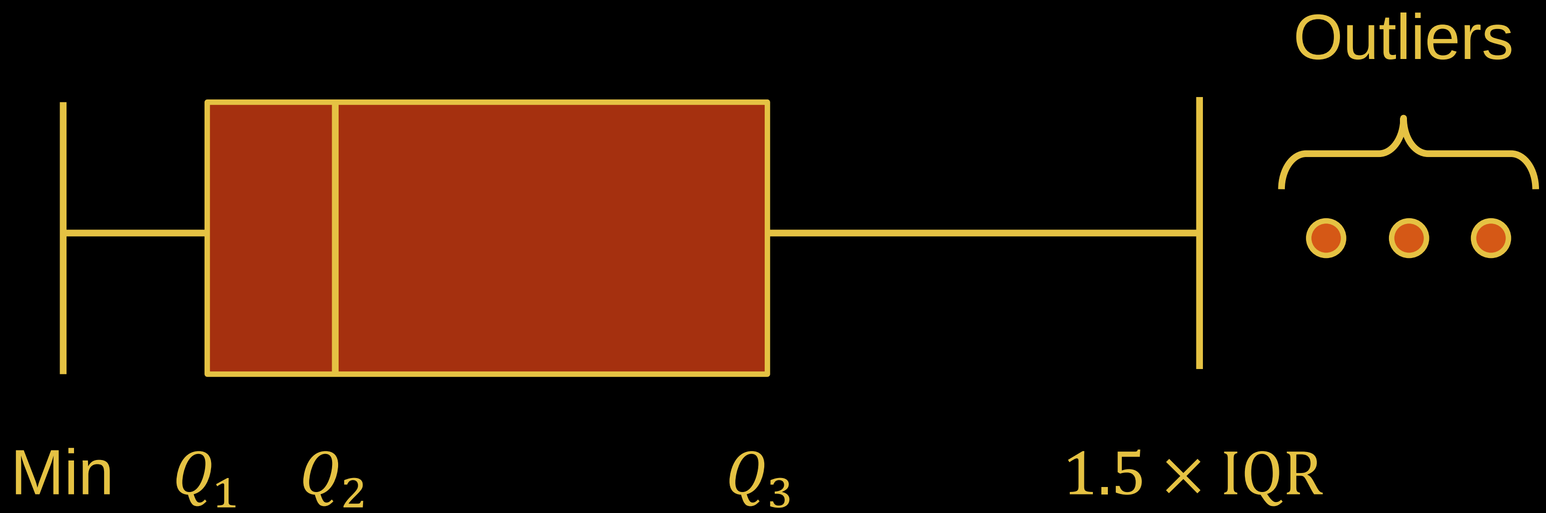
$$\mathrm{MAD}(x) = k \times \mathrm{median}(|x_i - \mathrm{median}(x)|)$$

1.4826 for Normal distribution

Z-scores & Robust Z-scores – R

```
# Z-scores #
hist(ins$Coverage_Income_Ratio_Claim)
x <- ins$Coverage_Income_Ratio_Claim

ins$Z_Coverage_Income_Ratio_Claim <- abs((x - mean(x))/sd(x))
hist(ins$Z_Coverage_Income_Ratio_Claim)

length(which(ins$Z_Coverage_Income_Ratio_Claim > 3))

# Robust Z-scores #
ins$RZ_Coverage_Income_Ratio_Claim <- abs((x - median(x))/mad(x))
hist(ins$RZ_Coverage_Income_Ratio_Claim)

length(which(ins$RZ_Coverage_Income_Ratio_Claim > 3))
```

1.5 IQR Rule



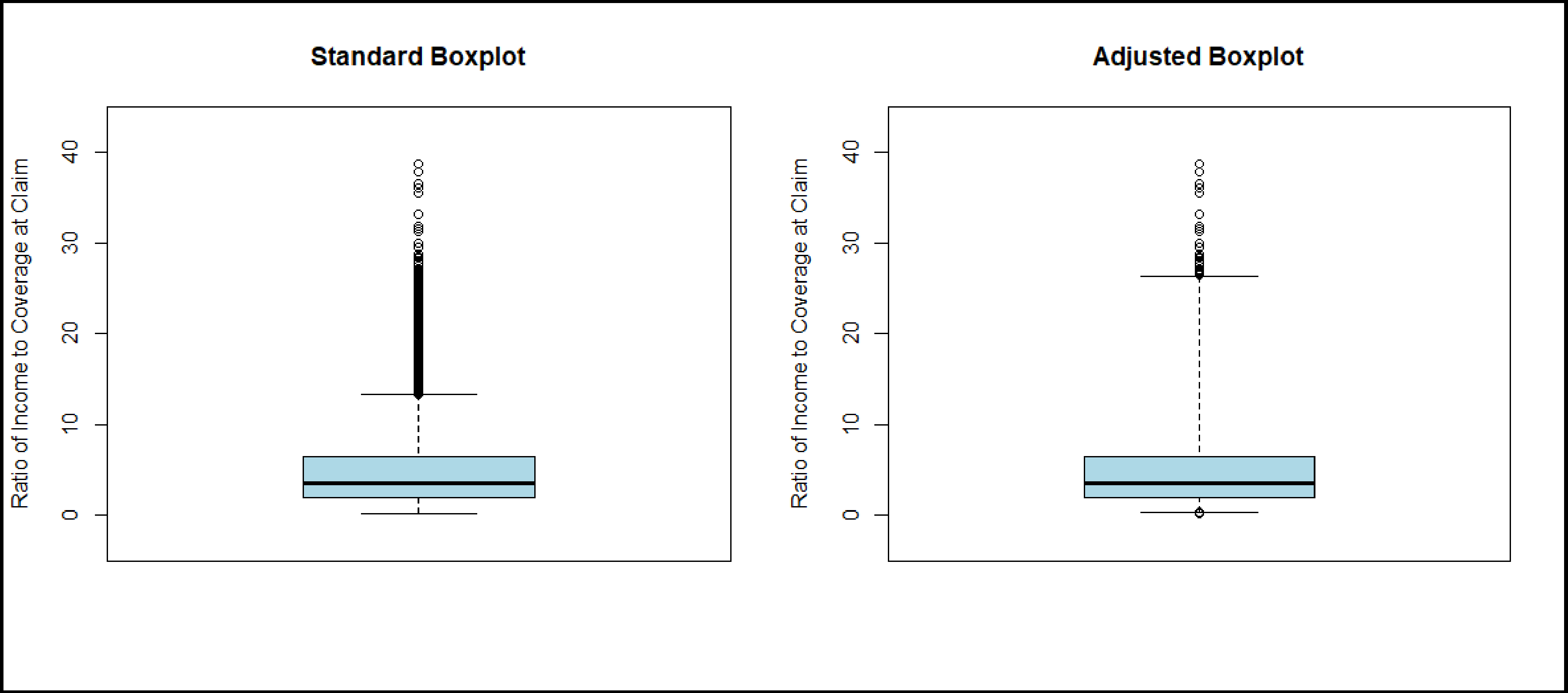
1.5 IQR Rule

- Works best for **symmetric** distributions.
- Severely skewed distributions tend to report large number of outliers.
- Use **adjusted boxplot** instead – more robust to skewed distributions.

1.5 IQR Rule & Adjusted IQR Rule – R

```
bp <- boxplot(ins$Coverage_Income_Ratio_Claim,
              col = "lightblue",
              main = "Standard Boxplot",
              ylab = "Ratio of Income to Coverage at Claim",
              ylim = c(-5, 45))

adjbp <- adjbox(ins$Coverage_Income_Ratio_Claim,
                col = "lightblue",
                main = "Adjusted Boxplot",
                ylab = "Ratio of Income to Coverage at Claim",
                ylim = c(-5, 45))

length(bp$out)
length(adjbp$out)
```

Comparison


Standard Boxplot
Adjusted Boxplot
Ratio of Income to Coverage at Claim
Ratio of Income to Coverage at Claim
0
10
20
30
40
0
10
20
30
40

?

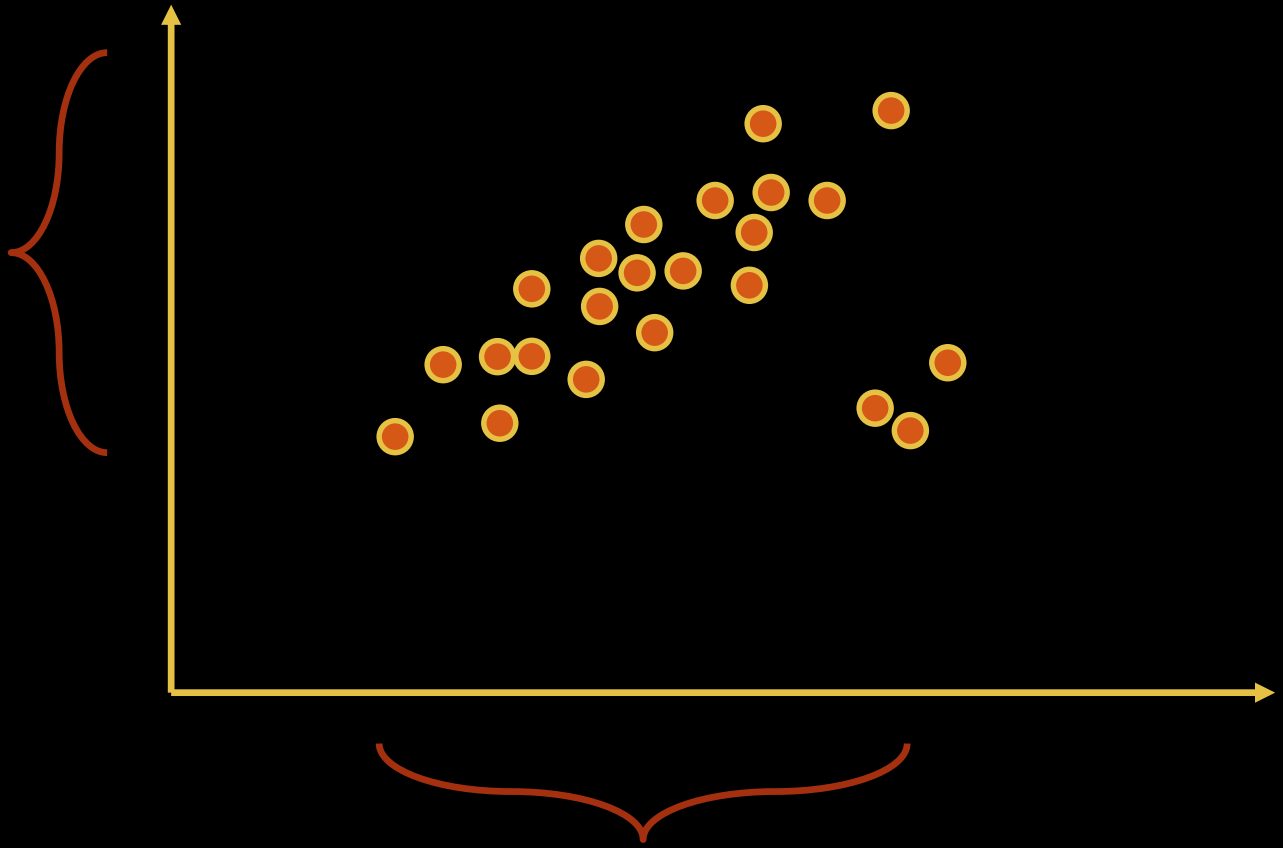
MULTIVARIATE ANALYSIS

Multiple Dimensions

- Outliers in one dimension are possibly restrictive.

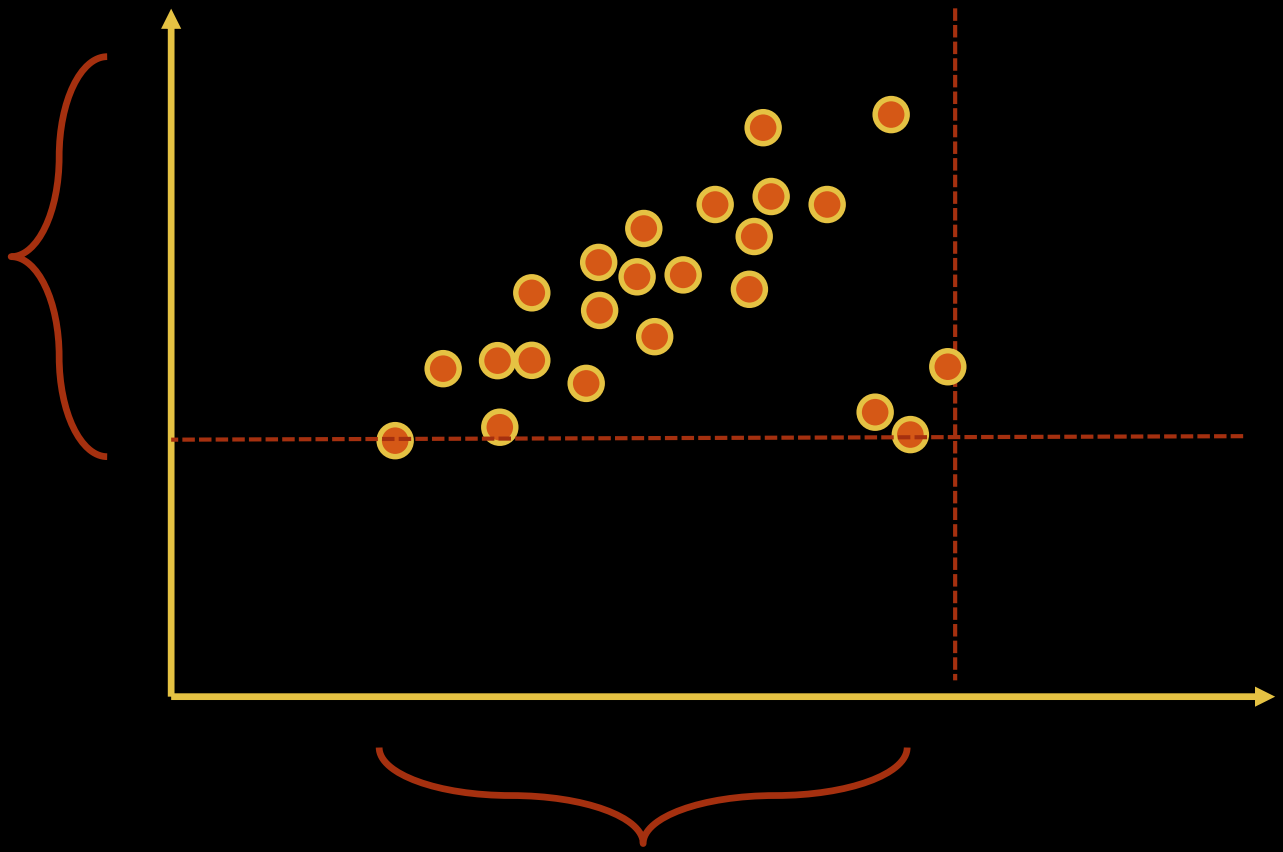
Multiple Dimensions

- Outliers in one dimension are possibly restrictive.



Multiple Dimensions

- Outliers in one dimension are possibly restrictive.



Multiple Dimensions

- Outliers in one dimension are possibly restrictive.
- Multivariate outlier detection:
  1. Mahalanobis distances
  2. Robust Mahalanobis distances
  3. k-Nearest Neighbors (kNN)
  4. Local Outlier Factor (LOF)
  5. Isolation Forests
  6. Classifier-Adjusted Density Estimation

Multiple Dimensions

- Outliers in one dimension are possibly restrictive.
- Multivariate outlier detection:

1. Mahalanobis distances
2. Robust Mahalanobis distances
3. k-Nearest Neighbors (kNN)
4. Local Outlier Factor (LOF)
5. Isolation Forests
6. Classifier-Adjusted Density Estimation

Mahalanobis Distances

- Generalization of z-scores to multi-dimensional space.
  - Replace univariate mean with **multivariate mean**
  - Replace standard deviation with **covariance matrix**

Mahalanobis Distances

- Generalization of z-scores to multi-dimensional space.
  - Replace univariate mean with **multivariate mean**
  - Replace standard deviation with **covariance matrix**

- Euclidean Distance (L2):

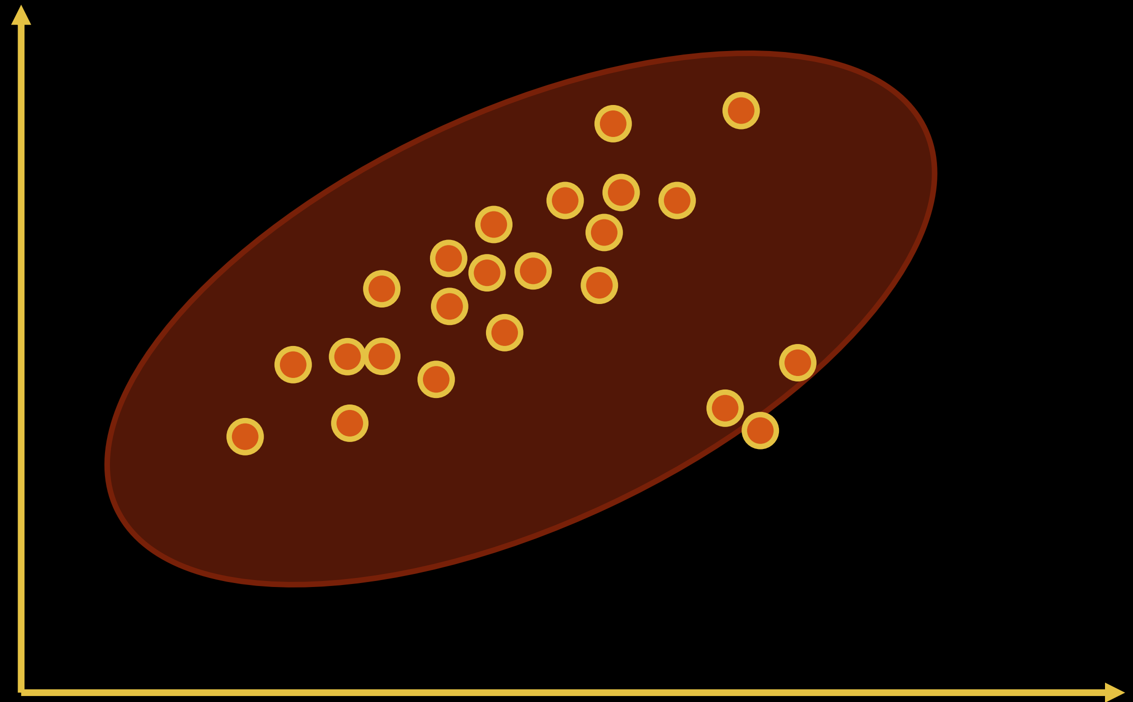
$$D_{L2} = \sqrt{(x-\mu)^T(x-\mu)}$$

- Mahalanobis Distance:

$$D_M = \sqrt{(x-\mu)^T\Sigma^{-1}(x-\mu)}$$

Confidence Ellipsoids

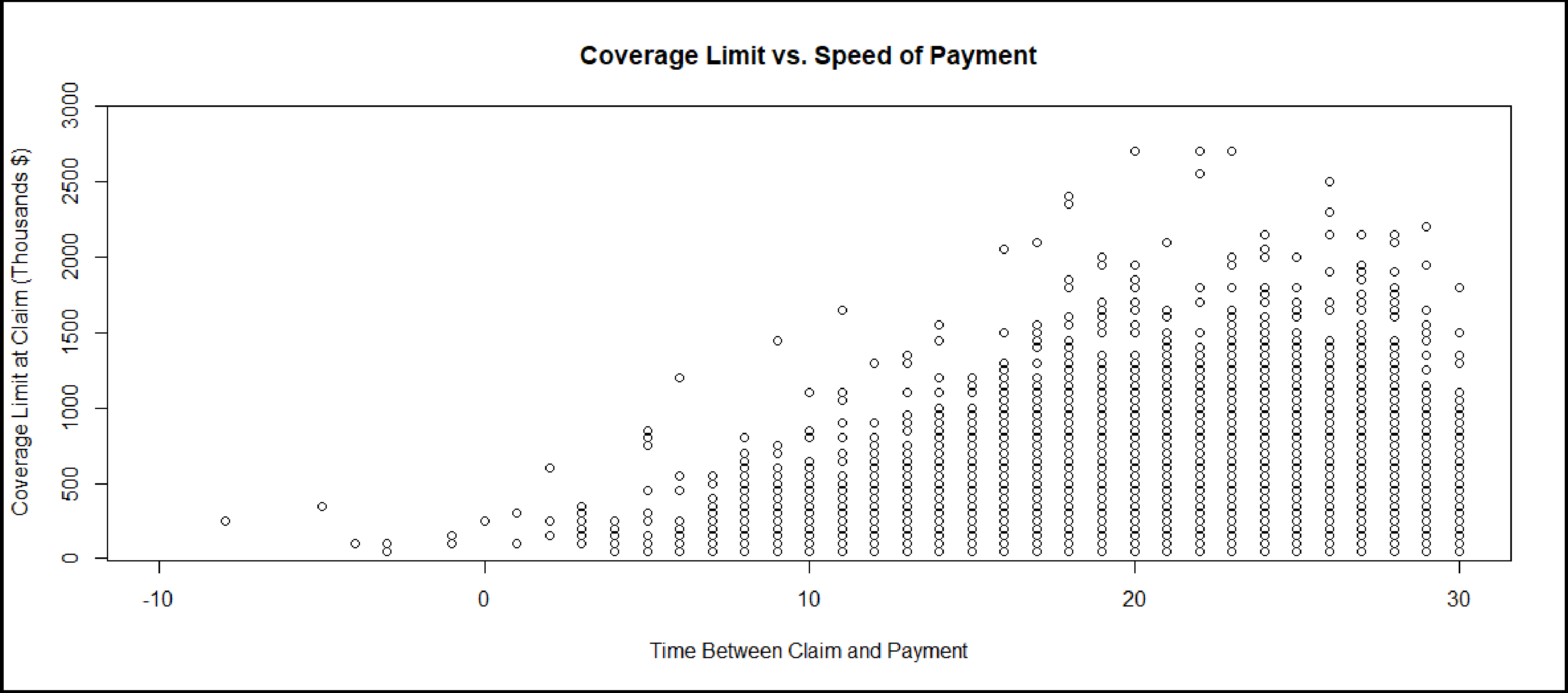
- Still bothered by outliers since standard mean and covariance matrix used.



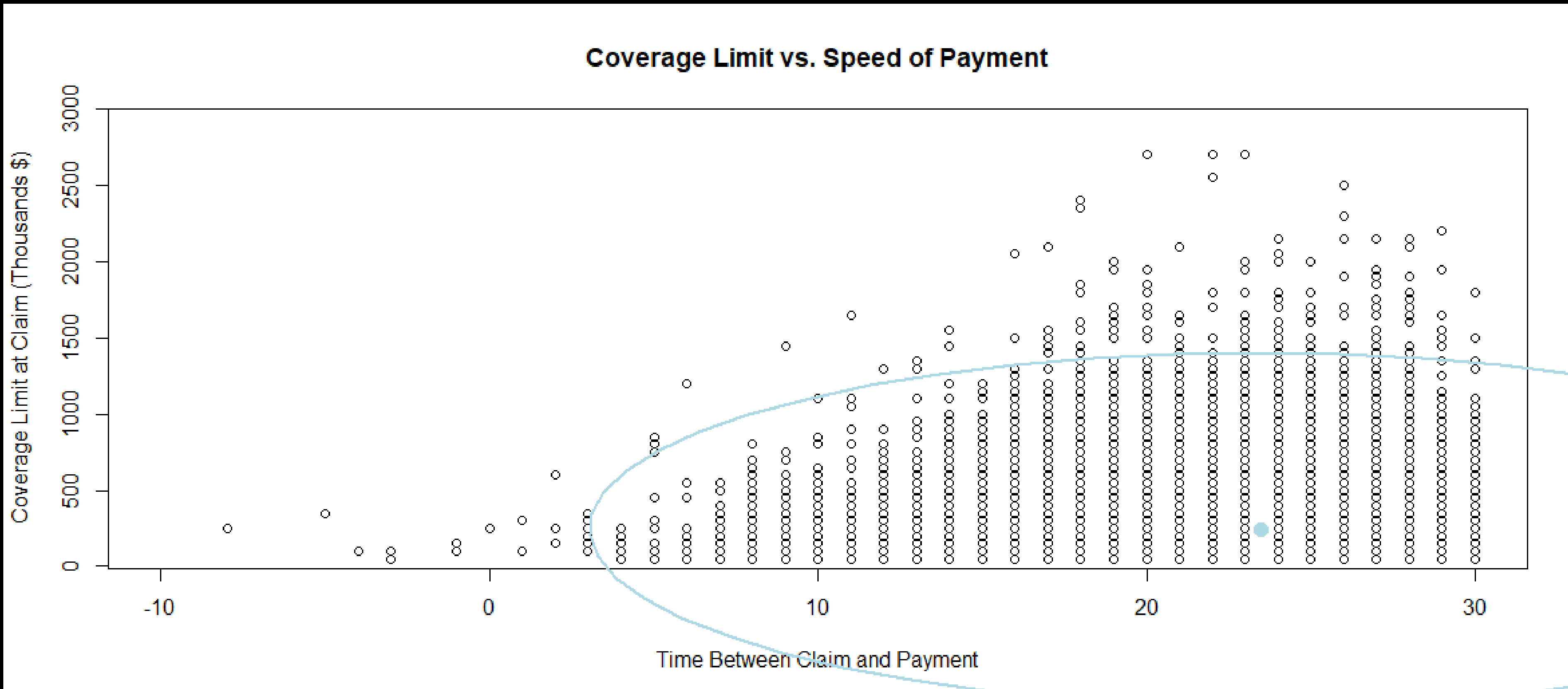
MD – R

```
# Mahalanobis Distances #
plot(x = ins$Time_Between_CL_R, y = ins$Cov_Limit_Claim/1000,
     xlim = c(-10, 30),
     ylim = c(-10, 3000),
     main = "Coverage Limit vs. Speed of Payment",
     xlab = "Time Between Claim and Payment",
     ylab = "Coverage Limit at Claim (Thousands $)")

df <- data.frame(Time = ins$Time_Between_CL_R,
                 CovLimit = ins$Cov_Limit_Claim/1000)

rad <- sqrt(qchisq(0.9999975, ncol(df)))
ellipse(center = colMeans(df, na.rm = TRUE),
        shape = cov(df), radius = rad, col = "lightblue")
```

MD – R


Coverage Limit vs. Speed of Payment
Coverage Limit at Claim (Thousands $)
0
500
1000
1500
2000
2500
3000
-10
0
10
20
30
Time Between Claim and Payment

MD – R


Coverage Limit vs. Speed of Payment
Coverage Limit at Claim (Thousands $)
0
500
1000
1500
2000
2500
3000
-10
0
10
20
30
Time Between Claim and Payment

Multiple Dimensions

- Outliers in one dimension are possibly restrictive.
- Multivariate outlier detection:
  1. Mahalanobis distances
  2. Robust Mahalanobis distances
  3. k-Nearest Neighbors (kNN)
  4. Local Outlier Factor (LOF)
  5. Isolation Forests
  6. Classifier-Adjusted Density Estimation

Robust Mahalanobis Distances

- Mahalanobis distances use mean and covariance matrix influenced by outliers.
- Use **robust** calculations of mean vector and covariance matrix instead:

$$D_M = \sqrt{(x - \mu_{MCD})^T \Sigma_{MCD}^{-1} (x - \mu_{MCD})}$$

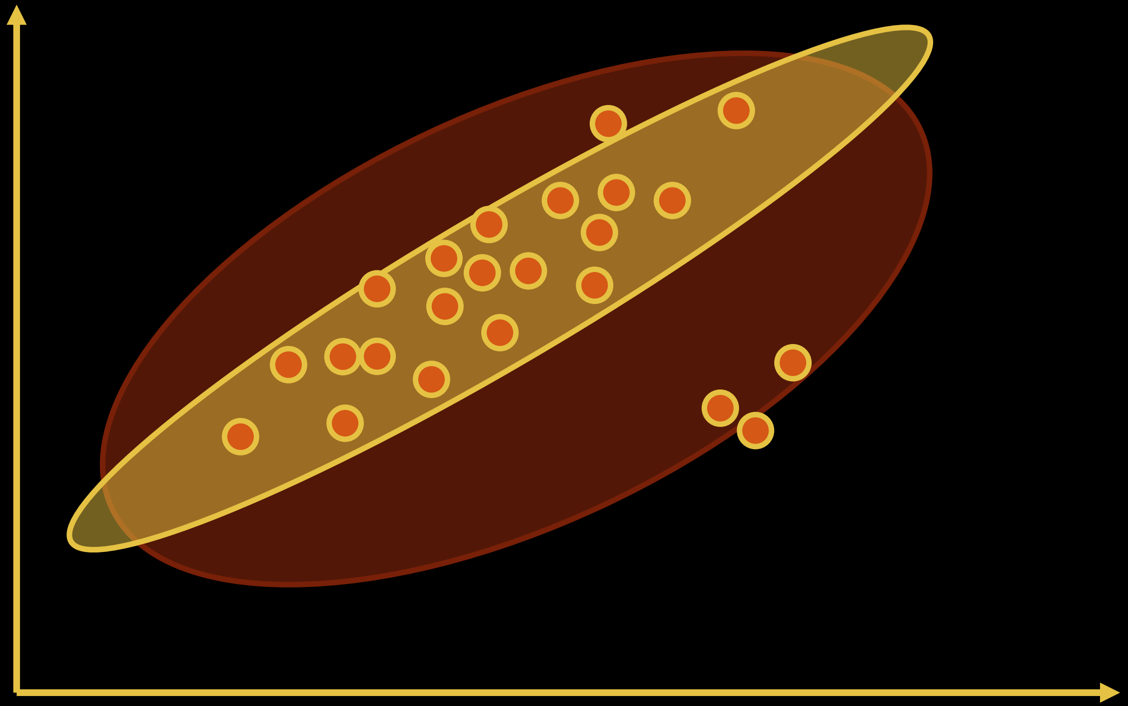
Minimum Covariance Determinant

$$D_M = \sqrt{(x - \mu_{MCD})^T \Sigma_{MCD}^{-1} (x - \mu_{MCD})}$$

- MCD: Minimum Covariance Determinant
  - Find h $(< n)$ observations that have MCD (essentially the tightest cloud)
  - Typically $h = 0.75 \times n$
  - Problem: How to find the right *h* observations?
  - Fast algorithms exist

Confidence Ellipsoids

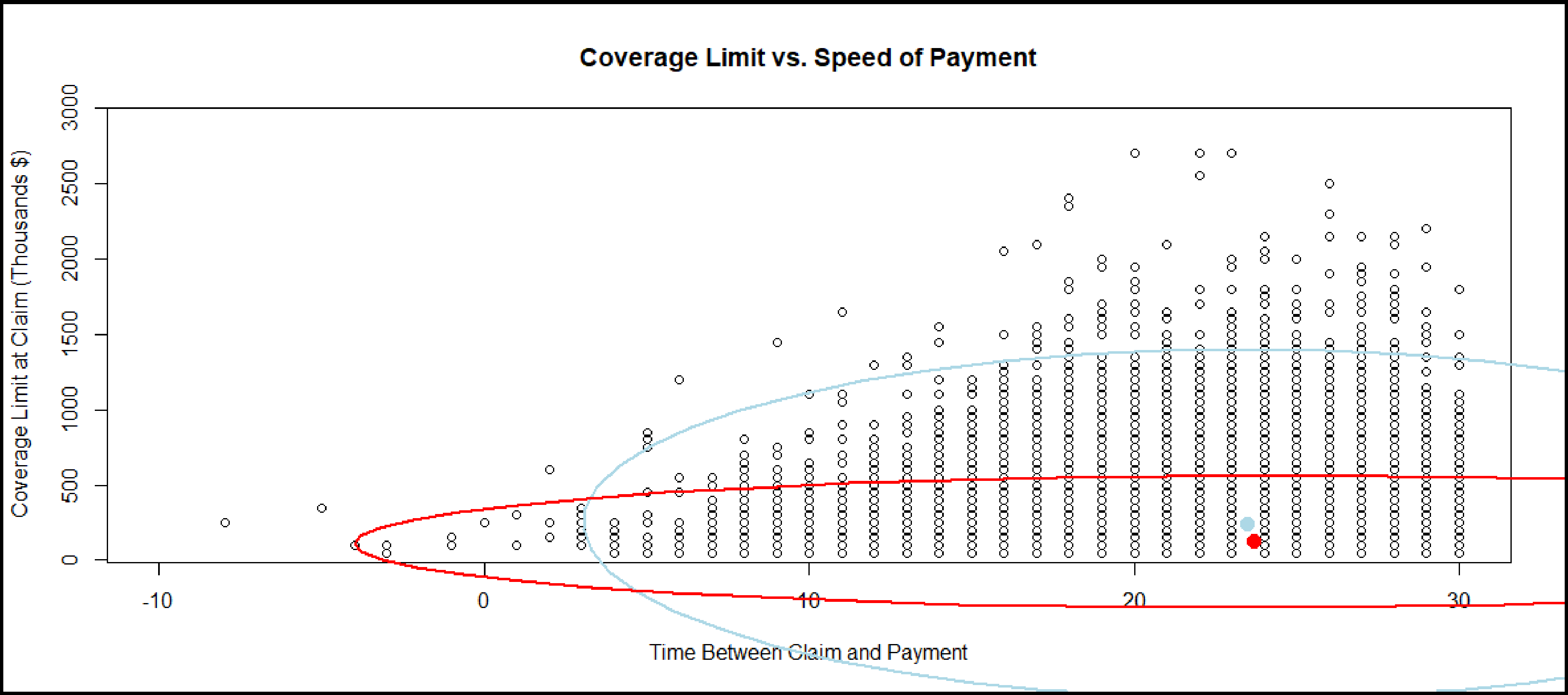
- Still bothered by outliers since standard mean and covariance matrix used.



Robust MD – R

```
# MCD Adjustment to MD #
mcdresult <- covMcd(df)
robustcenter <- mcdresult$center
robustcov <- mcdresult$cov

rad <- sqrt(qchisq(0.9999975, ncol(df)))
ellipse(center = robustcenter, shape = robustcov,
        radius = rad, col = "red")
```

Robust MD – R


Coverage Limit vs. Speed of Payment
Coverage Limit at Claim (Thousands $)
0
500
1000
1500
2000
2500
3000
-10
0
10
20
30
Time Between Claim and Payment

Multiple Dimensions

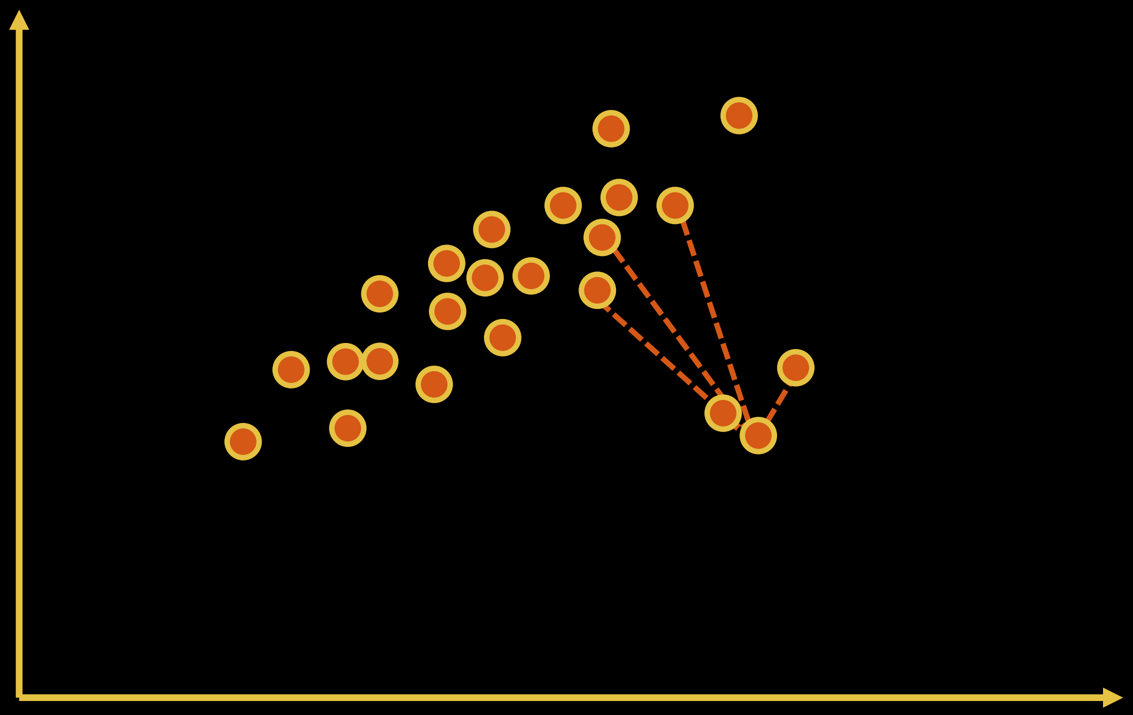
- Outliers in one dimension are possibly restrictive.
- Multivariate outlier detection:
  1. Mahalanobis distances
  2. Robust Mahalanobis distances
  3. k-Nearest Neighbors (kNN)
  4. Local Outlier Factor (LOF)
  5. Isolation Forests
  6. Classifier-Adjusted Density Estimation

k-Nearest Neighbors

- Want to discover points that are “not close” to the rest.
- Instead of distance from center of cloud, k-NN looks at distance from close points.
- Measure **average** distance from a point to each of the k-closest points.
  - Default: Euclidean distance

k-Nearest Neighbors

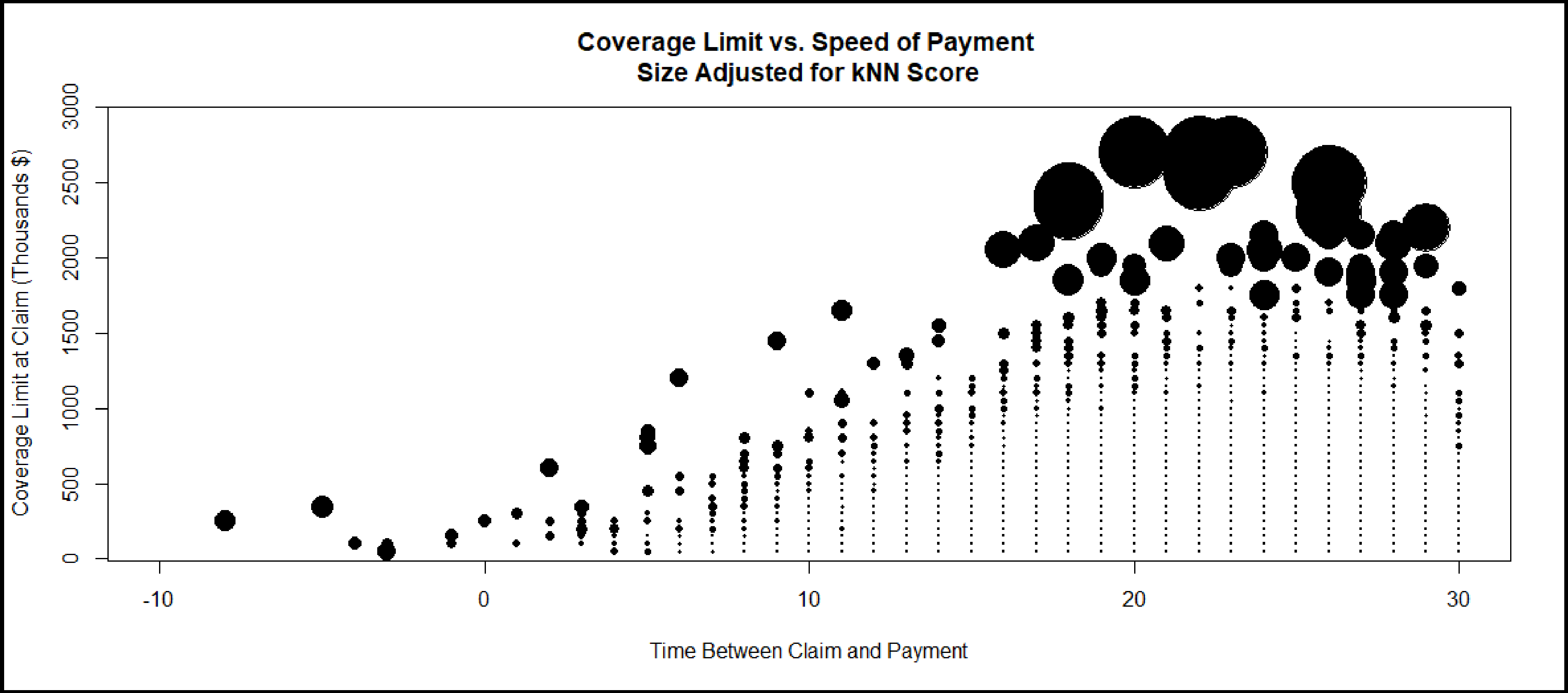
- Still bothered by outliers since standard mean and covariance matrix used.



k-Nearest Neighbors – R

```
# k-Nearest Neighbors #
df <- data.frame(Time = ins$Time_Between_CL_R,
                 CovLimit = ins$Cov_Limit_Claim/1000)

ins_knn <- get.knn(data = df, k = 5)

df$knn_score <- rowMeans(ins_knn$nn.dist)

plot(CovLimit ~ Time, data = df, cex = sqrt(knn_score)+0.01, pch = 20,
     xlim = c(-10, 30),
     ylim = c(-10, 3000),
     main = "Coverage Limit vs. Speed of Payment \nSize Adjusted for kNN Score",
     xlab = "Time Between Claim and Payment",
     ylab = "Coverage Limit at Claim (Thousands $)")
```

k-Nearest Neighbors – R



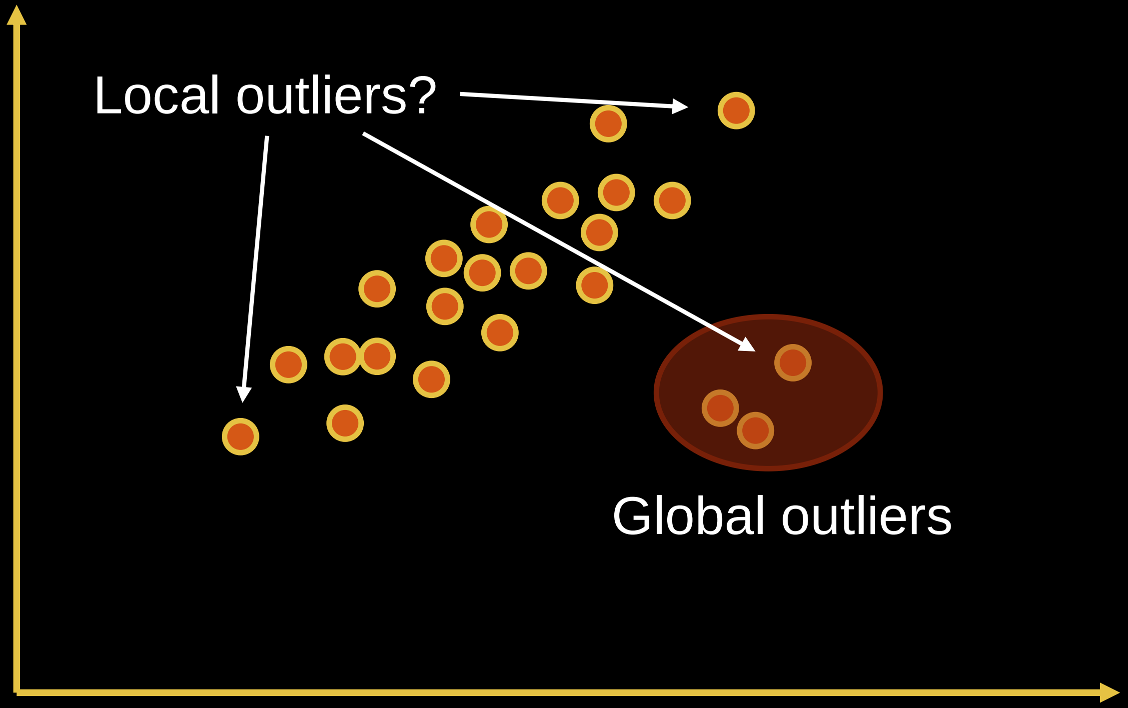
Multiple Dimensions

- Outliers in one dimension are possibly restrictive.
- Multivariate outlier detection:

1. Mahalanobis distances
2. Robust Mahalanobis distances
3. k-Nearest Neighbors (kNN)
4. Local Outlier Factor (LOF)
5. Isolation Forests
6. Classifier-Adjusted Density Estimation

Global vs. Local Outliers

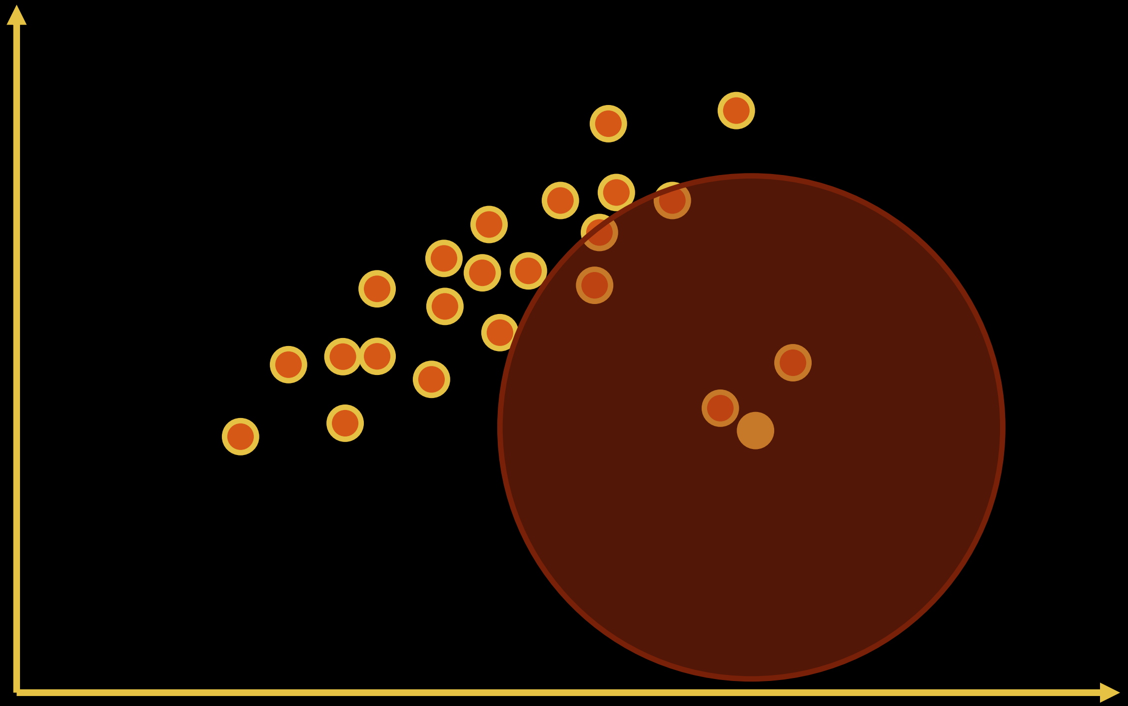
- k-NN great at detecting **global** outliers, but not **local** outliers.



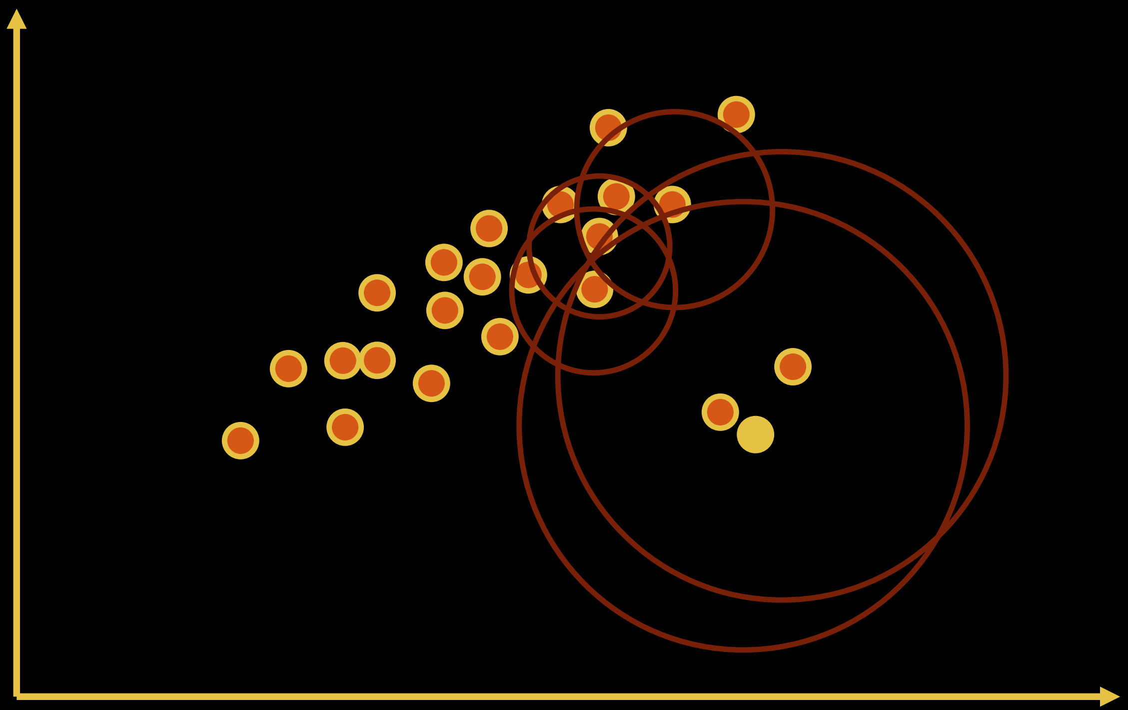
Local Outlier Factor (LOF)

- LOF:
  - Ratio (comparison) of the average **density** of the k-NN of an observation to the **density** of the observation itself.
  - > 1 means more likely to be anomaly
  - < 1 means less likely to be anomaly
- Density:
  - Inverse of the average **reachability** (distances) from observation to all of its k-NN.
  - Essentially, how far do we have to travel to nearest point, so less dense means farther travel.

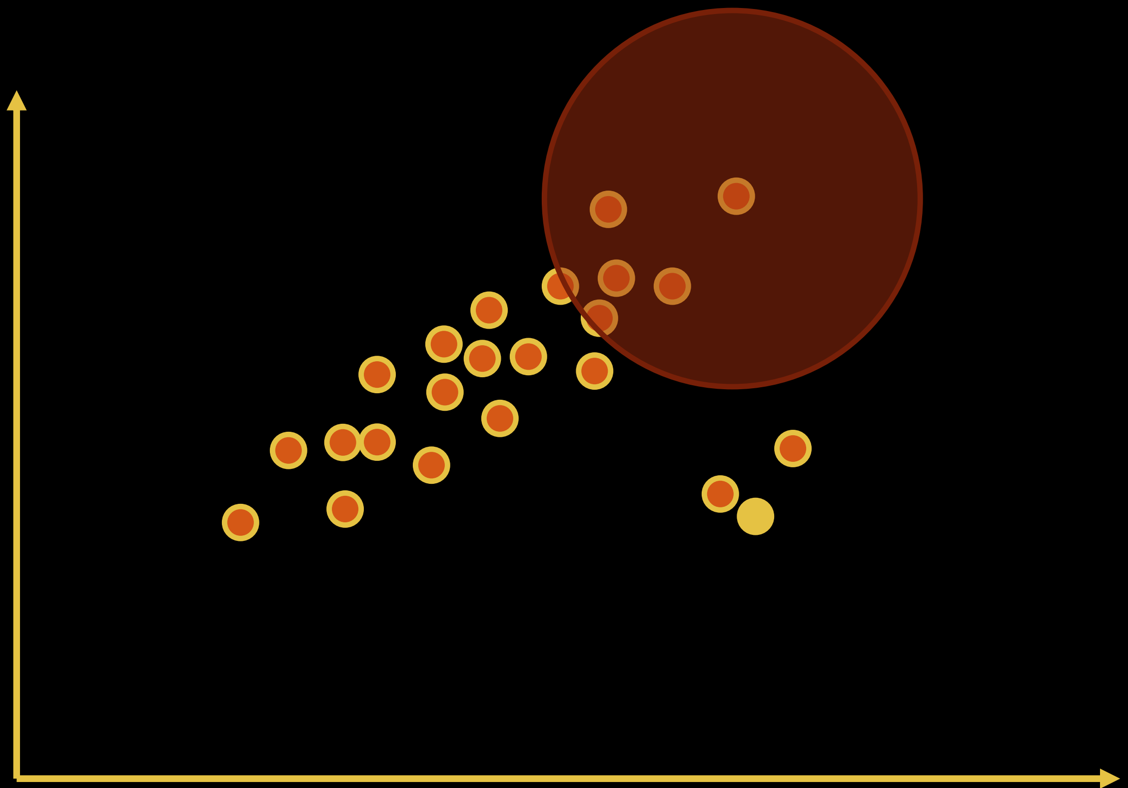
Local Outlier Factor (LOF)



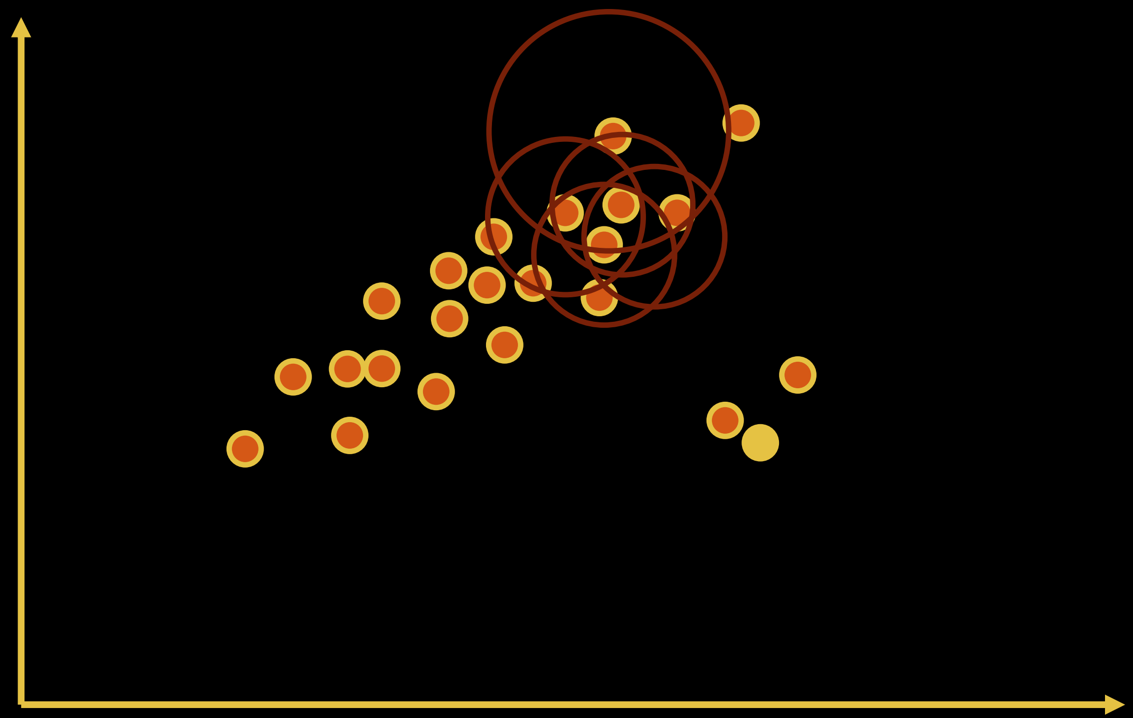
Local Outlier Factor (LOF)



Local Outlier Factor (LOF)



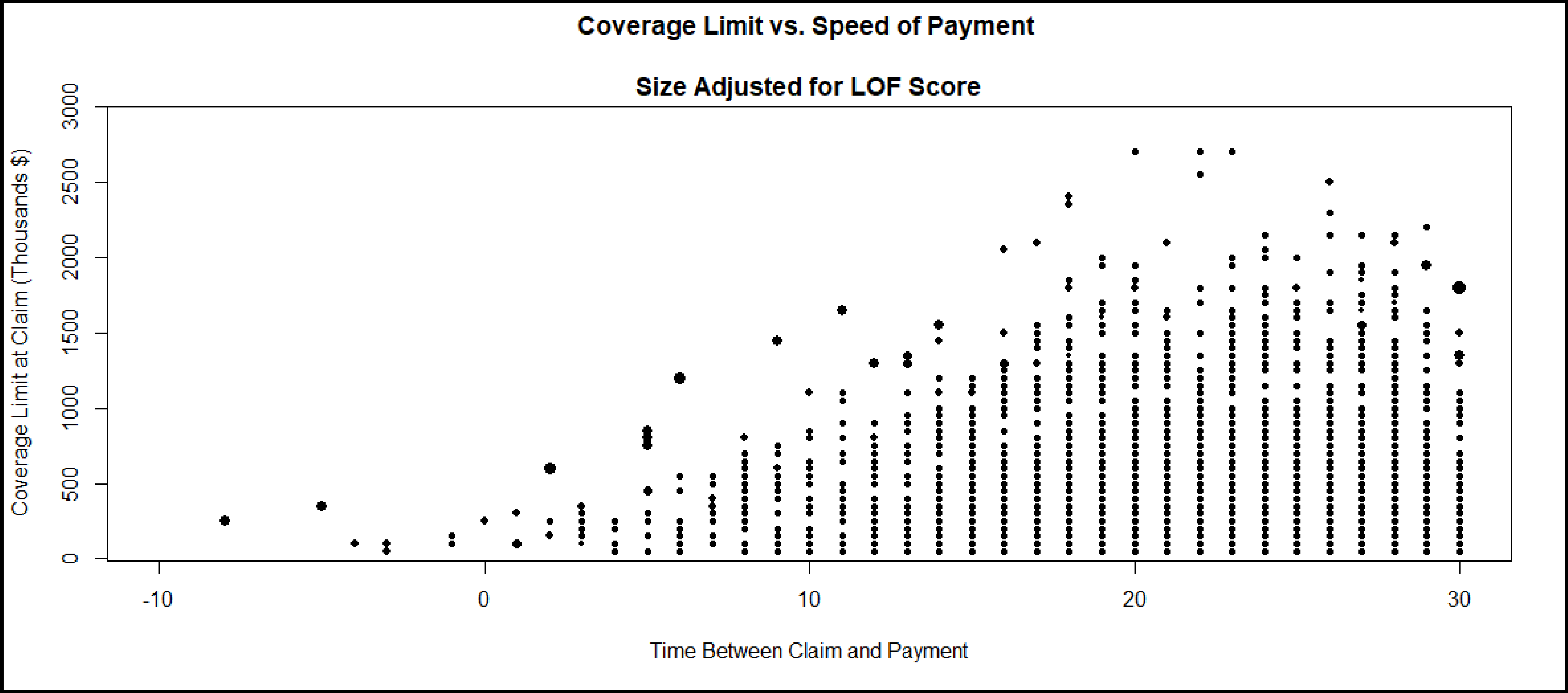
Local Outlier Factor (LOF)



Local Outlier Factor (LOF) – R

```
# Local Outlier Factor #
df <- data.frame(Time = ins$Time_Between_CL_R,
                 CovLimit = ins$Cov_Limit_Claim/1000)

ins_lof <- lof(scale(df), k = 5)

df$lof_score <- ins_lof

plot(CovLimit ~ Time, data = df, cex = lof_score, pch = 20,
     xlim = c(-10, 30),
     ylim = c(-10, 3000),
     main = "Coverage Limit vs. Speed of Payment
            \nSize Adjusted for LOF Score",
     xlab = "Time Between Claim and Payment",
     ylab = "Coverage Limit at Claim (Thousands $)")
```

Local Outlier Factor (LOF) – R

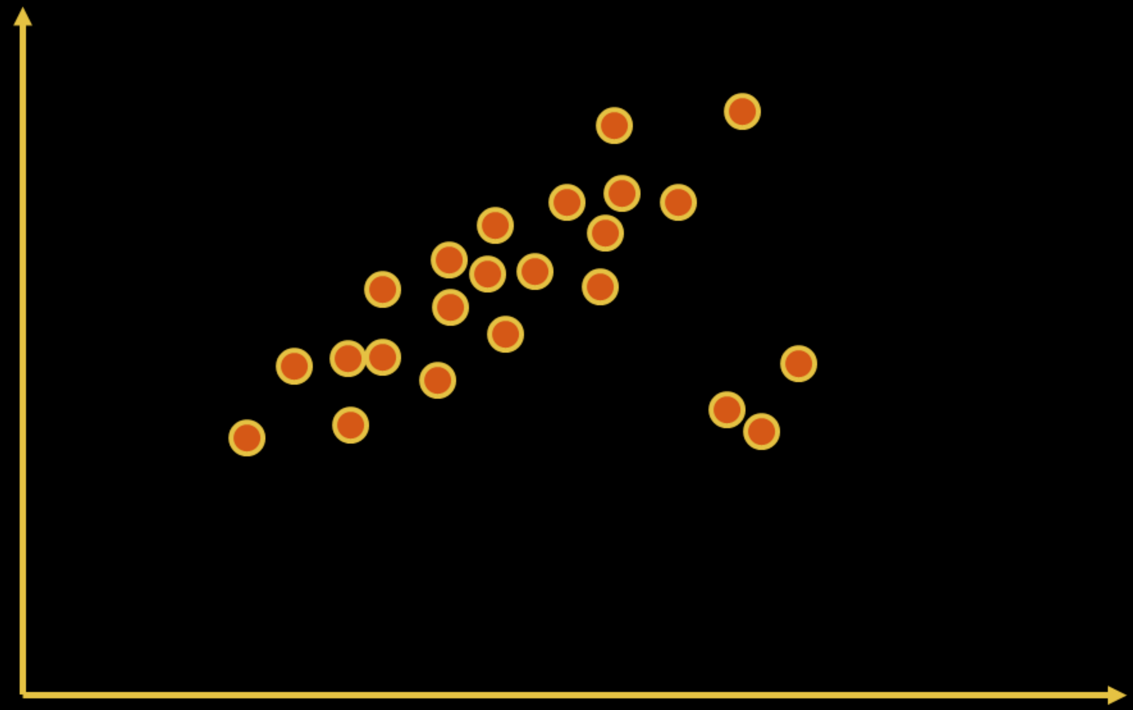


Multiple Dimensions

- Outliers in one dimension are possibly restrictive.
- Multivariate outlier detection:
  1. Mahalanobis distances
  2. Robust Mahalanobis distances
  3. k-Nearest Neighbors (kNN)
  4. Local Outlier Factor (LOF)
  5. Isolation Forests
  6. Classifier-Adjusted Density Estimation

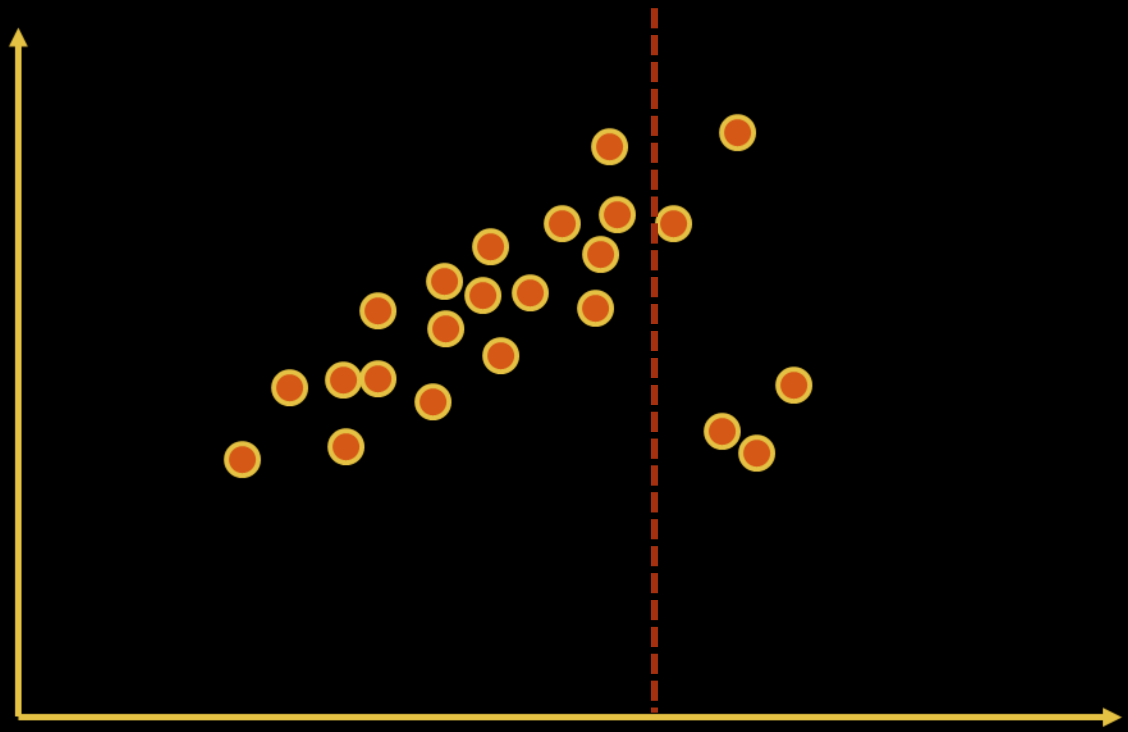
Isolation Tree

- Tree-based algorithm to isolate observations.
- Easier the isolation → More likely an anomaly!



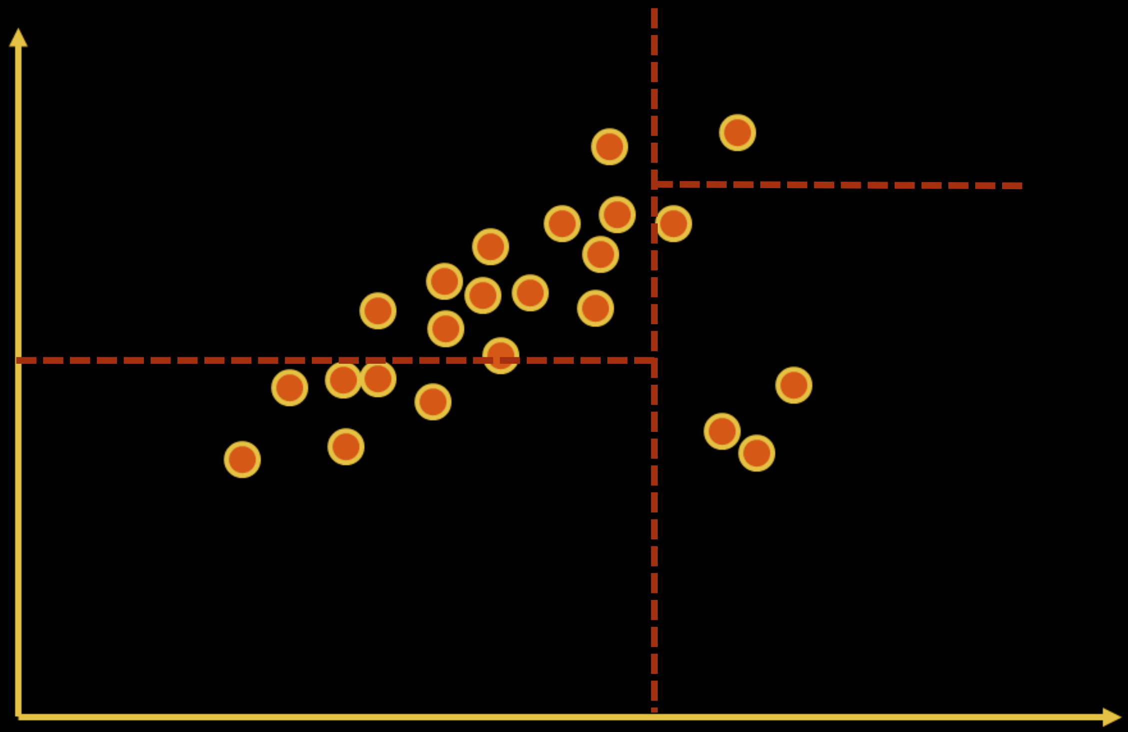
Isolation Tree

- Tree-based algorithm to isolate observations.
- Easier the isolation → More likely an anomaly!



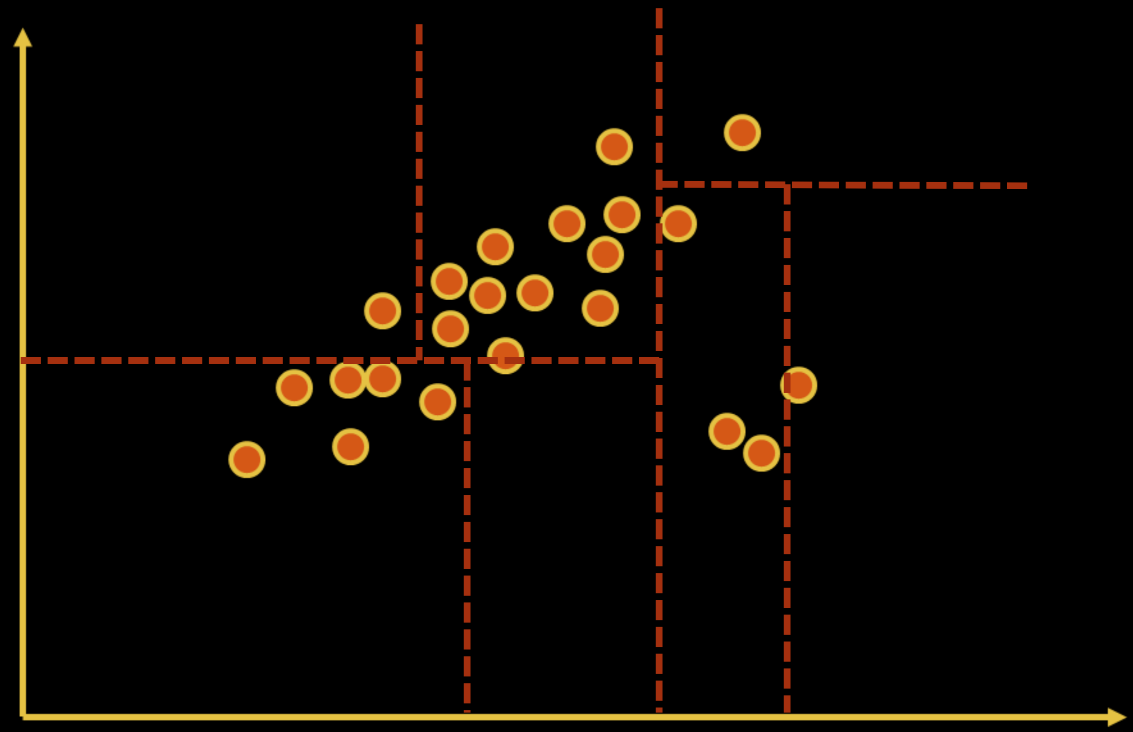
Isolation Tree

- Tree-based algorithm to isolate observations.
- Easier the isolation → More likely an anomaly!



Isolation Tree

- Tree-based algorithm to isolate observations.
- Easier the isolation → More likely an anomaly!



Isolation Tree

- Tree-based algorithm to isolate observations.
- Easier the isolation → More likely an anomaly!
- Isolation score is inversely related to number of needed splits to isolate observation.
  - Bounded between 0 and 1.
  - Closer to 1 → more likely an anomaly
  - Closer to 0 → less likely an anomaly
  - All observations ~ 0.5, no real anomalies

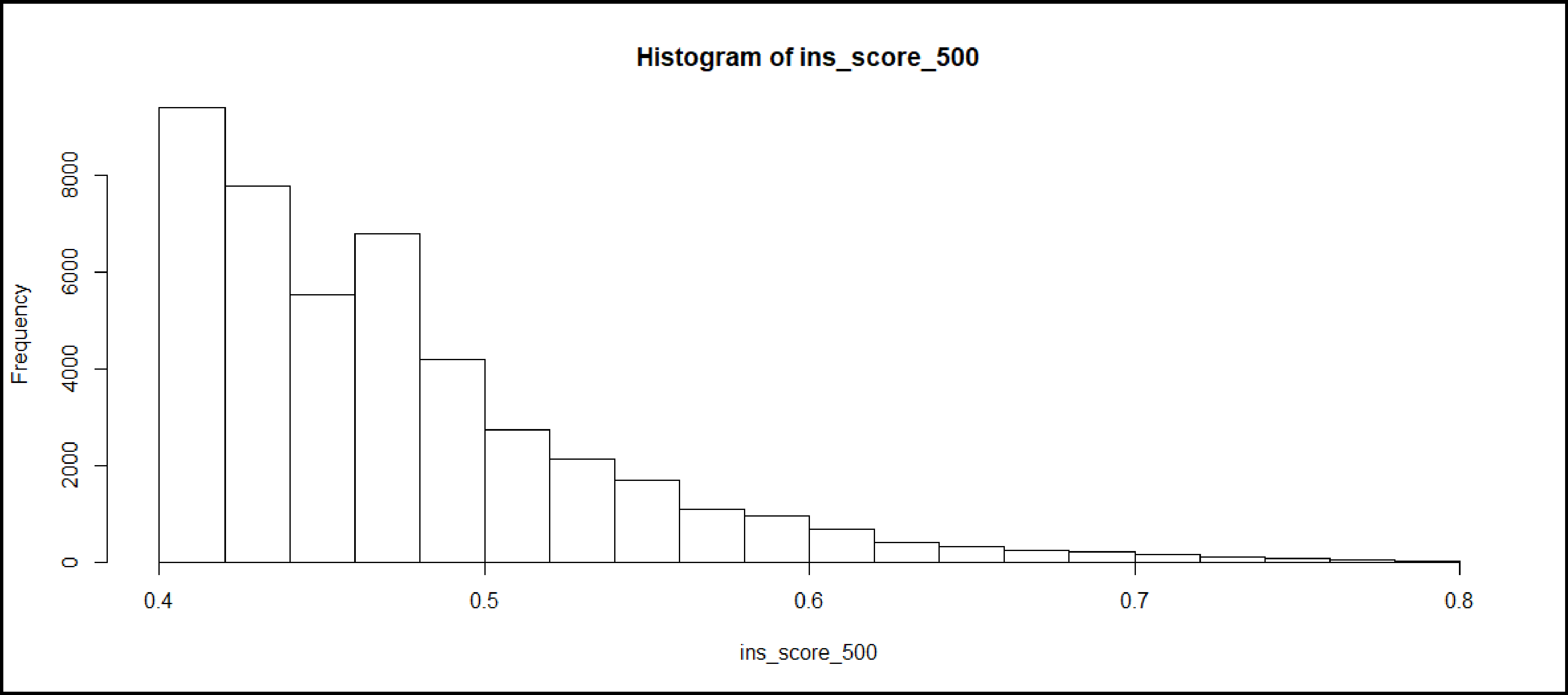
Isolation Forest

- Since the isolation trees are based on random splits on random dimensions, outlier might get lucky and survive longer than it really should.
- Isolation forest – combination of MANY isolation trees with averaged scores.
- Look for convergence of scores for optimal number of trees.

Isolation Forest – R

```r
# Isolation Forests #
df <- data.frame(Time = ins$Time_Between_CL_R,
                 CovLimit = ins$Cov_Limit_Claim/1000)

ins_for_100 <- iForest(X = df, nt = 100, phi = 5000, seed = 12345)
ins_score_100 <- predict(ins_for, newdata = df)

ins_for_500 <- iForest(X = df, nt = 500, phi = 1000, seed = 12345)
ins_score_500 <- predict(ins_for, newdata = df)

hist(ins_score_500)
df$iso_score <- ins_score_500
```

Isolation Forest – R



Multiple Dimensions

- Outliers in one dimension are possibly restrictive.
- Multivariate outlier detection:
  1. Mahalanobis distances
  2. Robust Mahalanobis distances
  3. k-Nearest Neighbors (kNN)
  4. Local Outlier Factor (LOF)
  5. Isolation Forests
  6. Classifier-Adjusted Density Estimation

CADE

- Newer technique for density estimation.
- Value been found in anomaly detection and fraud applications.

CADE

- Newer technique for density estimation.
- Value been found in anomaly detection and fraud applications.
- Process:
  1. Label all original data as **not outliers**
  2. Create new observations (same *n* as data) but variables are all uniformly distributed
  3. Label all new data as **outliers**, merge old and new data
  4. Use classification model to predict “outliers” (1’s).
  5. Score original data

CADE

- Newer technique for density estimation.
- Value been found in anomaly detection and fraud applications.

- High predicted probabilities → More likely an anomaly!
- Observation looks more like fake uniform data than actual distribution from which it came in multivariate space.

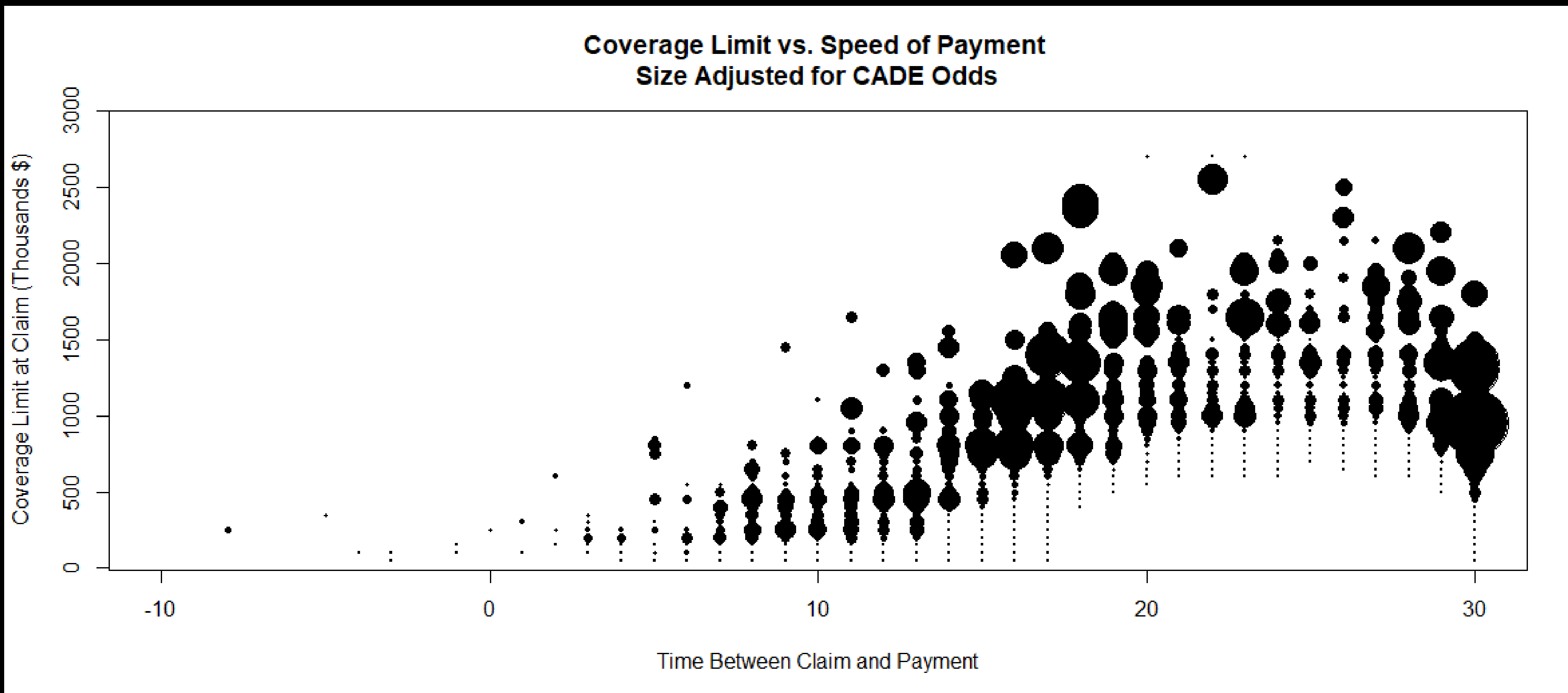
CADE – R

```
# Classifier-Adjusted Density Estimation #
df <- data.frame(Time = ins$Time_Between_CL_R,
                 CovLimit = ins$Cov_Limit_Claim/1000)

trans_uni <- function(x, len = length(x)) {
  if ( is.integer(x) ) {
    sample(min(x):max(x), len, replace = TRUE)
  } else if ( is.numeric(x) ) {
    runif(len, min(x), max(x))
  } else if ( is.factor(x) ) {
    factor(sample(levels(x), len, replace = TRUE))
  } else {
    sample(unique(x), len, replace = TRUE)
  }
}
```

CADE – R

```
cade <- function(df, n_tree) {

  actual <- df

  rand <- as.data.frame(lapply(actual, trans_uni))

  actual$y <- 0
  rand$y <- 1

  data <- rbind(actual, rand)

  tree <- randomForest(as.factor(y) ~ ., data = data, ntree = n_tree)

  # The classifier probabilities
  df$prob <- predict(tree, newdata = df, type = 'prob')[, 2]
  df$odds <- df$prob / (1 - df$prob)

  df
}

ins_cade <- cade(df = df, n_tree = 500)
```

CADE – R


Coverage Limit vs. Speed of Payment
Size Adjusted for CADE Odds
Coverage Limit at Claim (Thousands $)
0
500
1000
1500
2000
2500
3000
-10
0
10
20
30
Time Between Claim and Payment