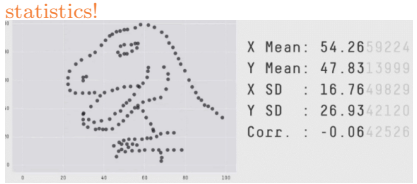
Remember the T-Rex: Go beyond the summary statistics!



Fitting Distributions

1. Collect data whose patterns are believed similar to the distribution of the random variable of interest.
2. Postulate one (or more) candidate probability distributions (e.g., normal (R: `"norm"`), lognormal (R: `"lnorm"`), uniform (R: `"unif"`)...)
3. Fit the postulated distribution to the data: finding values of the parameters that best fit the data. Two methods: Moment Matching (R: `"mme"`) and Maximum Likelihood Estimation Method (R: `"mle"`)

```
model <- fitdist(df, "norm",
    method="mme")
```

4. Evaluate candidate distributions using goodness-of-fit. E.g., **Akaike Information Criterion (AIC) -** smaller the better.
5. Tweak the distributions if necessary to account for any data limitations rendering the past not quite representative of the future.

Linear Regression

Linear regression means the dependent variable is linear in the model coefficients.

- Linear: $y = b_0 + b_1 x_1 + b_2 x_2$
- Not linear: $y = b_0 x_1^{b_1} + b_3 x_2^{b_2}$

`lm` in R

Rentals as a function of temperature and humidity:

```
mod <- lm(data = df, rentals ~ temp
    + rel_humidity)
summary(mod)
```

Consider a "Best Fitting" Line

$$\hat{y} = b_0 + b_1 x$$

Slope b_1 - sign is same as the sign of $\text{CORR}(X,Y)$. $\text{CORR}(X,Y)$ is between $[-1,1]$ and unit-less. But b_1 has units.

$$b_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2} = \text{CORR}(X,Y) \cdot \frac{\text{SD}(Y)}{\text{SD}(X)}$$

Intercept - $b_0 = \bar{y} - b_1 \bar{x}$

Mean-center data - if both y and x variables are mean-centered, rerun linear regression to get:

$$\hat{y}_i - \bar{y} = b_1 (x_i - \bar{x})$$

- new intercept will be 0
- new slope remains b_1

Regression to the mean - If $x = \bar{x}$ (the average of all x values), the predicted $\hat{y} = \bar{y}$ (average of all y values), independent from x.

Independent Variables

Categorical variables ("factor variable" in R) - if encoded with one-hot, one category should be dropped to avoid perfect multicollinearity (a situation where one predictor variable can be perfectly predicted from the others). This omitted category serves as a reference category against which the other categories are compared. This approach is known as creating "dummy variables."

Evaluation

Residuals (error) - $e_i = y_i - \hat{y}_i$

Sum of Squared Residuals -

$$SSR = \sum_{i=1}^n e_i^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Total Sum of Squares - a measure of the total variability in the observed data.

$$SST = \sum_{i=1}^n (y_i - \bar{y}_i)^2$$

R^2 **-** the proportion reduction in sum of squared residuals by the regression model compared to the baseline model (which always predicts the average value of all ys in the data: $\hat{y} = \bar{y}$).

$$\boxed{R^2 = 1 - \frac{SSR}{SST} = \frac{SST - SSR}{SST}}$$

- R^2 of a regression model lies between 0 and 1.
- Adding a new independent variable to a regression model can only increase R^2.
- R^2 on its own cannot judge how good a model is.

R^2 in multiple linear regression - equals the square of the correlation between the actual values y and the predicted values $\hat{y}$.

$$R^2 = (\text{CORR}(y, \hat{y}))^2$$

In the case of a perfect fit where every prediction of y_i is going to be correct, then $R^2 = 1$

R^2 in simple linear regression - equals the square of correlation between dependent variable y and independent variable x.

$$R^2 = (\text{CORR}(y, x))^2$$

Degrees of Freedom - the number of observations minus the number of parameters estimated (including the intercept): $df = n - p - 1$

Residual Variance - $\frac{SSR}{df}$

Residual Standard Error - provides a measure of the average distance that the observed values fall from the regression line.

$$RSE = \sqrt{\frac{SSR}{df}}$$

- Smaller RSE: the model's predictions are closer to the actual data points, suggesting a good fit.
- Larger RSE: the model's predictions are further from the actual data points, suggesting a poor fit.

Troubleshooting

The model's ability to predict the future and its interpretability can be impaired by

- The presence of irrelevant independent variables
- The presence of highly correlated independent variables
- The presence of "too many" variables relative to the size of the dataset

Irrelevant variables - have large p-values. Smaller p-value (R: `Pr(>|t|)`), the better. If lower than 0.05, consider it significant at the 5% significance level; otherwise, consider it non-significant at the 5% significance level. In R, a variable with one or more * is considered significant at 5% level. A variable not significant at the 5% level but at the 10% level can be stated as "less clear".

p-value of a variable x_i - the probability of observing a coefficient estimate as extreme, or more extreme, than the one actually obtained by the regression run in a similar sample assuming $\beta_i = 0$.

Highly correlated variables - Detected by inspecting the correlation matrix. (R: `cor(df)`).

Multicollinearity - When two variables are highly correlated (typically correlations higher than 0.75 in magnitude), resolve this by removing either of the independent variables and running the linear regression **again**.

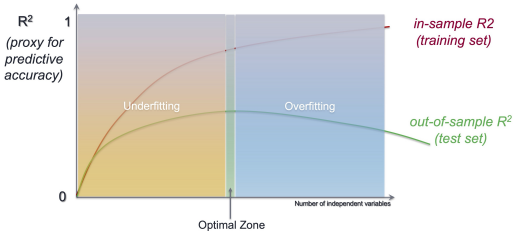
Overfitting ("too many" variables) - leads to poorly predicting future data. Remedy by (1) avoiding "non-significant" variables uising the starts * as a guide and (2) out-of-sample testing.

Out-of-Sample Testing -

1. Partition data set into 70% trainning set and 30% test set before creating the regression model.
2. Generate a set of models (e.g. with different sets of variables) using only the training data.
3. Evaluate models on the test set using out-of-sample R^2.

Out-of-Sample R^2 -

$$\text{OSR}^2 = 1 - \frac{\text{SSR of regression model applied to the test set}}{\text{SSR of baseline model applied to the test set}}$$



Key takeaways -

- Choose final model based on out-of-sample predictive quality metrics.
- Use metrics like R^2 and the standard error of regression in combination because they have different strength and weaknesses.
- Only include significant variables that are not highly correlated.
- Coefficients should "make sense".
- Use for interpolation rather than extrapolation.

Inferring model's coefficient - from the practice: "Correlation is not the same as causation. It could be that individuals with other contributing factors happen to sleep longer. It could be that these other factors, not sleep per se, are causally linked to the condition under study."

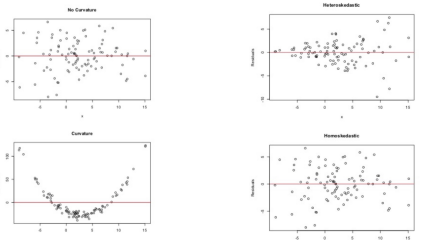
Technical caution - for the p-value to be correct, the "unaccounted for" differences in the regression model ϵ (think of the residuals) needs to have zero mean, constant standard deviation, be independent and follow a normal distribution.

Linearity - there is no curvature

Homoskedasticity - the dispersion of $e_i = y_i - \hat{y}_i$ is not systematically smaller or larger for large x_i values compared to small x_i values.

Nonlinearity or Curvature in Residuals?

Heteroskedasticity in error variances?



Top-right: hetero. Bottom-right:homo.

Normality - the residuals are approximately normal.

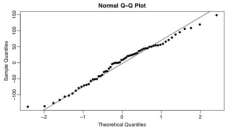
Checking Normality of Residuals.

QQ-Plot (quantile vs quantile plot) is commonly used here.

Make a normal quantile plot of the residuals interpretation:

Distribution of Residuals Normal?


```
qqnorm(resid.both)
qqline(resid.both)
```



Clustering

k-means - with initial centroids $\mu_1, \ldots, \mu_K \in \mathbb{R}^n$ (usually randomly selected), repeat until convergence: assign each point $i = 1, \ldots, n$ $\boxed{z_i \leftarrow \arg\min_{k=1,\ldots,K} \|\phi(s_i) - \mu_k\|^2}$ and then recenter each cluster $k = 1, \ldots, K$ $\boxed{\mu_k \leftarrow \frac{\text{all points in this cluster sumed}}{\text{\# points in this cluster}}}$. k-means is guaranteed to converge to a local (not global) optima even with random initialization of centroids. Solution: run multiple times with different random init or init with heuristic.

k-means objective func -

$$\boxed{L_{k\text{-means}}(z, \mu) = \sum_{i=1}^{n} \|\phi(x_i) - \mu_{z_i}\|^2}$$

Works the best - when clusters have "round shape".

```
set.seed(123)
# k = 2
# nstart = 100 is usually a good
    choice
my_clustering <- kmeans(df, centers
    =2, nstart=100)
my_clustering$centers # Show centers
df <- mutate(df, cluster=my_
    clustering$cluster) # Label the
    cluster assignments
```

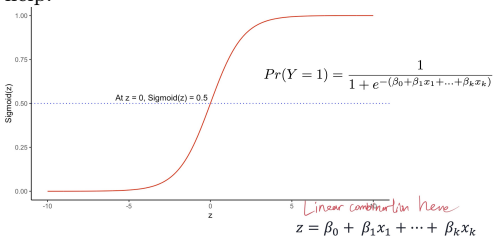
Scree Plot - to help choosing k value by displaying the tradeoff between number of clusters k and the within-cluster sum of squared distances. As k increases, this value should decrease.

If SSD decreases very slowly as we add more clusters, this dataset isn't cluster-able.

Classification

Why not Linear Regression? - (1) Predicted y can be negative or more than 1.0; (2) hard to interpret coefficients and hard to assess variable significance using *s.

Logistic Regression - a sigmoid function to help.



$$Pr(Y = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k)}}$$

$$z = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k$$

`glm` (generalized model) in R

```
mod <- glm(data = train, default ~
    ., family="binomial")
```

Binary classification, not multi-class.

Confusion Matrix

		In Reality: Defaulted Y = 1	In Reality: Repaid Y = 0
Model Prediction	Will default Pr(Y = 1) >= cutoff	A	B
	Will repay Pr(Y = 1) < cutoff	C	D

We can calculate metrics like:

Accuracy = $\frac{A + D}{A + B + C + D}$ — the proportion of cases that are correctly classified

Error Rate = 1 – Accuracy = $\frac{B + C}{A + B + C + D}$ — the proportion of cases that are incorrectly classified

Usually a trained classification model can be compared with a baseline model that always predicts the most common outcome.

Accuracy/error rate isn't enough: as long as TP and TN adds to the same number, accuracy won't change assuming that all errors are equally important. To mitigate, add in financial considerations.

True Positive Rate - i.e., sensitivity or **recall**.

$\boxed{TPR = \frac{TP}{TP+FN}}$ TPR + FNR = 1

True Negative Rate - i.e., specificity.

$\boxed{TNR = \frac{TN}{TN+FP}}$ TNR + FPR = 1

- TPR and FPR cannot be chosen independently.
- If a cutoff chosen improves FPR, TPR will be adversely affected and vice-versa.

ROC Curve - receiver operating characteristic, captures the trade-off between FPR and TPR as the cutoff is varied.

1. Identify the more expensive type of error (FN or FP)
2. Decide on the max you are willing to tolerate that type of error
3. Use the ROC curve to pick the cutoff

Area Under the ROC Curve - in general, models with higher AUC are preferable.

CART

1. For each independent variable and each possible way to split the dataset based on that variable, calculate the impurity (reduction in overall SSR)
2. Choose the split (independent variable to split on + the cutoff value to split at) that has the max reduction in SSR and split the dataset into two accordingly
3. Repeat

CART models can be used for both a categorical dependent variable and continuous dependent variable.

```
mdl <- rpart(data=train, default ~
    installment + fico_score,
    minbucket=value, cp=value)
prp(mdl) # Visualize
predict(mdl, newdata=test)
```

`minsplit` **-** the minimum number of observations that must exist in a node in order for a split to be attempted.

`minbucket` **-** the minimum number of observations in any terminal node. If only one of `minbucket` or `minsplit` is specified, the code either sets `minsplit` to `minbucket*3` or `minbucket` to `minsplit/3`, as appropriate. Smaller `minbucket`, more splits in the final tree.

`cp` **complexity parameter -** Any split that does not decrease the overall lack of fit by a factor of cp is not attempted. For instance, with anova splitting, this means that the overall R-squared must increase by cp at each step. The main role of this parameter is to save computing time by pruning off splits that are obviously not worthwhile. Essentially, the user informs the program that any split which does not improve the fit by cp is not of interest, and that hence the program need not pursue it Smaller cp, more layers (more underlying splits in the final tree).

Advantages of CART -

- Can discover interactions between variables without explicitly incorporating interaction terms
- Can learn nonlinear patterns
- May mirror human decision making processes
- Decision Rule is highly transparent: A series of if/else statements.Even simpler than linear regression. Can be explained to non-experts

Limiations of CART -

- CART predictions lack smoothness
- CART models may exhibit instability
- May not lead to the strongest out-of-sample predictive performance

Exam Cautions: Watch out what the leaf nodes represent and what the question asks for: could be two opposite.

Cross-Validation

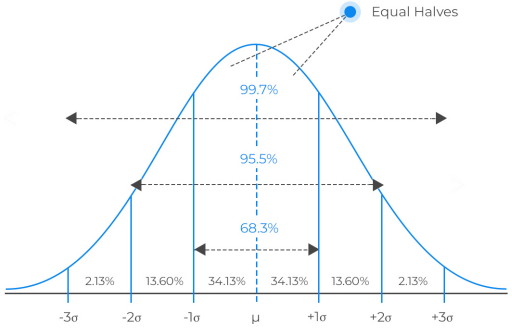
Goal - finding the values of `minbucket` and `cp`.

k-fold Cross-validation:

Divide	• Divide the **training set** into k pieces ("folds") — Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5
Train & Evaluate	• For each candidate parameter value... ○ Use $k-1$ folds to construct a model ○ Test that model on the remaining fold (e.g "loss" = mean squared residuals) ○ Repeat for each of the k folds
Average	• For each candidate parameter value... ○ compute the "average loss" over the k folds
Select	• Select parameter that minimizes the average loss ○ Use of entire training set but nothing from test set

`minbucket` is usually fixed throughout the cross-validation.

Side bar



- about 68% of the total values lie within 1 standard deviation of the mean.
- about 95% lie within 2 standard deviations of the mean
- about 99.7% lie within 3 standard deviations of the mean.