MESSAGE FROM AMBROSE

This review sheet (a.k.a. a "cheat sheet") follows the order of topics in the *ACTEX Study Manual for Exam PA* and provides a "helicopter" view of the entire PA exam syllabus. As you probably know, the focus of Exam PA is on *conceptual understanding* and *written communication*. To write well and score high, there are quite a lot of things you have to memorize in advance (e.g., describe best subset selection, explain how cost-complexity pruning works, the pros and cons of GLMs vs. decision trees), and this cheat sheet collects, I believe, the most important items in one place for your convenience. Although no substitute for a thorough review of the study manual, this cheat sheet should be a valuable aid to enhance retention and memorization. When I myself took Exam PA in December 2019, I used a preliminary version of this cheat sheet and found it quite useful. (It was only 7 pages long then!) Please feel free to refine it and put in additional facts and tips that you think are valuable.

1 General Model Building Steps

1.1 Problem Definition

- **Three main categories of predictive modeling problems:**

  (More than one category can apply in a given business problem.)

Category	Focus	Aim
Descriptive	What happened in the past	To "describe" or interpret observed trends by identifying relationships between variables
Predictive	What will happen in the future	To make accurate "predictions"
Prescriptive	The impacts of different "prescribed" decisions	To answer the "what if?" and "what is the best course of action" questions

- **Characteristics of predictive modeling problems:**
  - ▷ *(Issue)* There is a clearly identified and defined business issue to be addressed.
  - ▷ *(Questions)* The issue can be addressed with a few well-defined questions.
  - ▷ *(Data)* Good and useful data is available for answering the questions above.
  - ▷ *(Impact)* The predictions will likely drive actions or increase understanding.
  - ▷ *(Better solution)* Predictive analytics likely produces a solution better than any existing approach.
  - ▷ *(Update)* We can continue to monitor and update the models when new data becomes available.

- **How to produce a meaningful problem definition?**
  - ▷ *General strategy:* Get to the root cause of the business issue and make it specific enough to be solvable.
  - ▷ *Specific strategies:*
    - ☐ *(Hypotheses)* Use prior knowledge of the business problem to ask questions and develop testable hypotheses.
    - ☐ *(KPIs)* Select appropriate key performance indicators to provide a quantitative basis for measuring success.

- **Constraints to face:**
  - ▷ The availability of easily accessible and high quality data
  - ▷ Implementation issues, e.g., the presence of necessary IT infrastructure and technology to fit complex models efficiently, the cost and effort required to maintain the selected model

1.2 Data Collection and Validation

Data design

- **Relevance**: Need to ensure that the data is unbiased, i.e., representative of the environment where the model will operate.
  - ▷ *Population:* Important for the data source to be a good proxy of the true population of interest.
  - ▷ *Time frame:* Choose the time period which best reflects the business environment of interest.

    In general, recent history is better than distant history.
- **Sampling**: The process of taking a subset of observations from the data source to generate the dataset
  - ▷ ***Random sampling***: "Randomly" draw observations from the underlying population without replacement. Each record is equally likely to be sampled.
  - ▷ ***Stratified sampling***: Divide the underlying population into a no. of non-overlapping "strata" (often w.r.t. target) non-randomly, then randomly sample a set no. of observations from each stratum ⇒ get a more representative sample.

    A special case—systematic sampling: Draw observations according to a set pattern; no random mechanism controlling which observations are sampled.
- **Granularity**: Refers to how precisely a variable is measured, i.e., level of detail for the information contained by the variable.

Data quality issues

- **Reasonableness**: Data values should be reasonable (make sense) in the context of the business problem, e.g., variables such as age, time, and income $ should be non-negative.
- **Consistency:** Records in the data should be inputted consistently on the same basis and rules, e.g.:
  - ▷ Same measurement unit for numeric variables
  - ▷ Same coding scheme for categorical variables
- **Sufficient documentation**: Examples of useful elements:
  - ▷ A description of the dataset overall, including the data source
  - ▷ A clear description of each variable (definition and format)
  - ▷ Notes about any past updates or other irregularities of the dataset
  - ▷ A statement of accountability for the correctness of the dataset
  - ▷ A description of the governance processes used to manage the dataset

Other data issues

- **Personally identifiable information (PII)**: Information that can be used to trace an individual's identity, e.g., name, SSN, address, photographs, and biometric records

How to handle PII?

- ▷ *Anonymization:* Anonymize or de-identify the data to remove the PII.
- ▷ *Data security:* Ensure that the data receives sufficient protection.
- ▷ *Terms of use:* Be well aware of the terms and conditions, and the privacy policy related to the collection and use of data.

- **Variables with legal/ethical concerns**:
  - ▷ *Sensitive variables:* Differential treatment based on sensitive variables may lead to unfair discrimination and raise equity concerns.

    Examples: Race, ethnicity, gender, age, income, disability status, or other prohibited classes
  - ▷ *Proxy variables:* Variables that are closely related to (hence serve as a "proxy" of) prohibited variables.

    Examples:
    - ☐ Occupation (possibly a proxy of gender)
    - ☐ Geographical location (possibly a proxy of age and income)

- **Target leakage**: **(Important to watch out for!)**
  - ▷ *Definition:* When predictors in a model "leak" information about the target variable that would not be available when the model is deployed in practice
  - ▷ *Key to detecting target leakage—Timing:* These variables are observed at the same time as or after the target variable.
  - ▷ *Problem with this issue:* These variables cannot serve as predictors in practice and would lead to artificially good model performance if mistakenly included.

1.3 Exploratory Data Analysis (EDA)

- **Aim:** Use summary statistics + graphical displays to gain insights into the distribution of variables on their own and in relation to one another (esp. the target variable).

  Some typical uses:
  - ▷ Clean and validate the data to make it ready for analysis
  - ▷ Identify potentially useful predictors
  - ▷ Generate useful features (e.g., variable transformations)
  - ▷ *(Important!)* Decide which type of model (GLMs or trees) is more suitable, e.g., for a complex, non-monotonic relation, trees may do better

- **Univariate exploration tools:**

Variable Type	Summary Statistics	Visual Displays	Observations
Numeric	**Mean**, **median**, **variance**, minimum, maximum	**Histograms**, **boxplots**	▷ Any (right) skew? ▷ Any unusual values?
Categorical	Class frequencies	**Bar charts**	▷ Which levels are most common? ▷ Any sparse levels? ▷ (For binary targets) Presence of imbalance

- **Bivariate exploration tools:**

Variable Pair	Summary Statistics	Visual Displays	Observations
Numeric × Numeric	**Correlations** (only for linear relations)	**Scatterplots**	Any noticeable relationships, e.g., monotonic, non-linear?
Numeric × Categorical	Mean/median of numeric variable split by categorical variable	**Split boxplots**, **histograms** (stacked or dodged)	Any sizable differences in the means/medians among the factor levels?
Categorical × Categorical	2-way frequency table	**Bar charts** (stacked, dodged, or filled)	Any sizable differences in the class proportions among different factor levels?

- **Common data issues for numeric variables:**

Issue 1: Highly correlated predictors	
Problems	▷ Difficult to separate out the individual effects of different predictors on the target variable ▷ For GLMs, coefficients become widely varying in sign and magnitude, and difficult to interpret.
Possible Solutions	▷ Drop one of the strongly correlated predictors. ▷ Use **PCA** to compress the correlated predictors into a few **PCs**.

Issue 2: Skewness (esp. right skewness due to outliers)	
Problems	Extreme values: ▷ Exert a disproportionate effect on model fit ▷ Distort visualizations (e.g., axes expanded inordinately to take care of outliers)
Possible Solutions	Apply **transformations** to reduce right skewness: ▷ **Log transformation** (works only for strictly positive variables; remedy: add a small positive number to each value of the variable if there are zeros) ▷ **Square root transformation** (works for non-negative variables) Options to handle **outliers**: ▷ *(Remove)* If an outlier is unlikely to have a material effect on the model, then OK to remove it. ▷ *(Keep)* If the outliers make up only an insignificant proportion of the data, then OK to leave them in the data. ▷ *(Modify)* Modify the outliers to make them more reasonable, e.g., change negative values to zero. ▷ *(Using robust model forms)* Fit models by minimizing the absolute error (instead of squared error) between predicted values and the observed values. *Reason:* Absolute error places much less relative weight on the large errors and reduces the impact of outliers on the fitted model.

Issue 3: Should they be converted to a factor?	
Considerations	"Yes" if... ▷ Variable has a small no. of distinct values, e.g., quarter of the year (1 to 4). ▷ Variable values are merely numeric labels (no sense of numeric order, e.g., group no.). ▷ Variable has a complex relationship with target variable ⇒ factor conversion gives models (esp. GLMs) more flexibility to capture relationship "No" if... ▷ Variable has a large no. of distinct values, e.g., hour of the day (would cause a high dimension and overfitting if converted into a factor). ▷ Variable values have a sense of numeric order that may be useful for predicting the target variable. ▷ Variable has a simple monotonic relationship with target ⇒ its effect can be effectively captured by treating it as a numeric variable. ▷ Future observations will have new variable values (e.g., calendar year)

- **Common issue for categorical predictors: Sparse levels**

  ▷ *Problem with high dimensionality/granularity:* Sparse factor levels reduce robustness of models and may cause overfitting.

  ▷ *A solution:* Combine sparse levels with more populous levels where the target variable behaves similarly to form more representative and interpretable groups.

  ▷ *Trade-off:* To strike a balance ⚖ between:

  ☐ Ensuring each level has a sufficient no. of observations

  ☐ Preserving the differences in the behavior of the target variable among different factor levels for prediction

  ▷ *Tip:* Knowledge of the meaning of the variables is often useful when making combinations, e.g., regrouping hour of day as "morning," "afternoon," and "evening."
  (Use common sense and check the data dictionary!)

- **Interaction:**

  ▷ *Definition:* Relationship between a predictor and the target variable depends on the value/level of another predictor.
  (*Tip:* Good to include the definition in your response whenever an exam subtask tests interaction!)

  ▷ *Graphical displays to detect interactions:*

Predictor Combination	Numeric Target	Categorical Target
Numeric × Categorical	Scatterplot colored by categorical predictor	Boxplot for numeric predictor split by target and faceted by categorical predictor
Categorical × Categorical	Boxplot for target split by one predictor and faceted by the other predictor	Bar chart for one predictor filled by target and faceted by the other predictor
Numeric × Numeric	Bin one of the predictors (i.e., cut it into several ranges), or try a decision tree.	

- ▷ *Interaction vs. correlation:* Literally similar, but different
  - ☐ *Interaction:* Concerns a 3-way relationship (1 target variable and 2 predictors)
  - ☐ *Correlation:* Concerns the relationship between two numeric predictors

1.4 Model Construction and Evaluation

Training/test set split

- **How?**

  Before fitting models, split the data into the training set (70-80%) and the test set (20-30%) by stratified sampling.

  ↓

  Models are fitted to → Training set

  Prediction performance is evaluated on → Test set

  Test set observations must be truly unseen to the trained model.

- **Why do the split?**
  - ▷ Model performance on the training set tends to be overly optimistic and favor complex models.
  - ▷ Test set provides a more objective ground for assessing the performance of models on new, unseen data.
  - ▷ Split replicates the way the models will be used in practice.

- **Why use stratified sampling:** To produce representative training and test sets w.r.t. target variable (not predictors).

- **Trade-off about the sizes of the two sets:**

  Larger training set ⇒ { Training is more robust; Evaluation on test set is less reliable }

- **Alternative:** Do the split based on a time variable, e.g., year, to evaluate how well a model extrapolates past time trends to future, unseen years.

Common performance metrics

- **General**
  - ▷ *Regression vs. classification problems:*
    - ☐ ***Regression:*** When target is numeric (quantitative)
    - ☐ ***Classification:*** When target is categorical (qualitative)

    (**Note:** The predictors can be numeric or categorical. ⚠)
  - ▷ What do metrics computed on training and test sets measure:
    - ☐ ***Training:*** Goodness of fit to training data
    - ☐ ***Test:*** Prediction performance on new, unseen data
  - ▷ *Loss function:* Most performance metrics use a loss function to capture the discrepancy between the actual and predicted values for each observation of the target variable.

    Examples:
    - ☐ Square loss (most common for numeric targets)
    - ☐ Absolute loss
    - ☐ Zero-one loss (mostly for categorical targets)

- **Metrics for regression problems:**
  - ▷ ***RMSE:*** $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}$
  - ▷ ***Pearson χ^2 statistic:*** $\frac{1}{n}\sum_{i=1}^{n}\frac{(y_i-\hat{y}_i)^2}{\hat{y}_i}$ (often for count data)

- **Metrics for (binary) classification problems:**
  - ▷ *Classification rule:*

    Predicted probability for "+" > **cutoff** ⇔ Predicted class = "+"

▷ ***Confusion matrices:***

Prediction	Reference (= Actual) −	Reference (= Actual) +
−	TN	FN
+	FP	TP

☐ **Accuracy** $= \dfrac{\text{TN}+\text{TP}}{n} =$ proportion of correctly classified obs.

☐ **Classification error rate** $= \dfrac{\text{FN}+\text{FP}}{n} =$ proportion of misclassified obs.

☐ **Sensitivity** $= \dfrac{\text{TP}}{\text{TP}+\text{FN}} =$ proportion of +ve obs. correctly classified as +ve

☐ **Specificity** $= \dfrac{\text{TN}}{\text{TN}+\text{FP}} =$ proportion of -ve obs. correctly classified as -ve

☐ **Precision** $= \dfrac{\text{TP}}{\text{FP}+\text{TP}} =$ proportion of +ve predictions truly belonging to +ve class

Weighted average relation:

$$\text{accuracy} = \frac{n_-}{n} \times \text{specificity} + \frac{n_+}{n} \times \text{sensitivity}.$$

▷ *How confusion matrix metrics vary with cutoff:*

$$\text{Cutoff} \uparrow \quad \Rightarrow \begin{cases} \text{Sensitivity} \downarrow \\ \text{Specificity} \uparrow \end{cases}$$

May use a cost-benefit analysis to optimize cutoff.

▷ ***Area under the ROC curve (AUC)***

☐ Plot sensitivity against specificity for all cutoffs from 0 to 1 and compute the area under the curve.

☐ Two special points on an **ROC curve**:

$$(\text{sensitivity}, \text{specificity}) = \begin{cases} (1,0), & \text{if cutoff} = 0, \\ (0,1), & \text{if cutoff} = 1. \end{cases}$$

☐ Typically ranges between 0.5 (random classifier) and 1 (perfect classifier).

- **Summary of performance metrics:**

Target Type	Model Metrics	Criterion
Numeric	(R)MSE, Pearson chi-square	Lower, better
Categorical	Accuracy, sensitivity, specificity, AUC	Higher, better

Cross-validation (CV)

- **How it works:**

For a fixed +ve integer k (e.g., 10), randomly split the training data into k folds of approximately equal size

⇓

Train the model on all but one folds and measure performance on left-out fold

⇓

Repeat with each fold left out in turn to get k performance values

⇓

Average to get overall CV metric

- **Common uses of CV:**

▷ *Model assessment:* To evaluate a model's test set performance without using any test set.

▷ *Hyperparameter tuning:* To tune **hyperparameters** (= parameters with values supplied in advance; not optimized by the model fitting algorithm) by picking the values that produce the best CV performance (lowest MSE or highest accuracy).

- **Considerations when selecting the best model:**
  - ▷ *(Prediction performance)* The model should perform well on test data w.r.t. certain performance metrics.
  - ▷ *(Interpretability)* The model should be reasonably interpretable, i.e., the predictions should be easily explained in terms of the predictors and lead to specific insights.
  - ▷ *(Ease of implementation)* The easier for a model to be implemented (computationally, financially, or logistically), the better the model.

Sidebar: Unbalanced data (for binary targets)

- **Meaning:** One class is much more dominant than the other.
- **Problems with unbalanced data:**
  - ▷ A classifier implicitly places more weight on the majority class and tries to fit those observations well, but the minority class may be the +ve class.
  - ▷ A high accuracy can be deceptive.
- **Solution 1—Undersampling:** Keep all observations from the minority class, but draw fewer observations ("undersample") from the majority class.
  - ▷ *Drawback:* Less data ⇒ training becomes less robust and the classifier becomes more prone to overfitting.
- **Solution 2—Oversampling:** Keep all observations from the majority class, but draw more observations ("oversample") from the minority class.
  - ▷ *Drawback:* More data ⇒ heavier computational burden
  - ▷ *Caution:* Should be done after training/test set split
- **Effects of undersampling and oversampling on model results:**

+ve class becomes more prevalent in the balanced data

⇓

Predicted probabilities for +ve class will increase

⇓

For a fixed cutoff, sensitivity ↑ but specificity ↓

Controlling model complexity

- **Overfitting:**
  - ▷ *Definition:* Model is trying too hard to capture not only the signal, but also the noise specific to the training data.
  - ▷ *Indications:* Small training error, but large test error
  - ▷ *Problem:* An overfitted model fits training data well, but does not generalize well to new, unseen data (poor predictions). Not a useful model!
- **Quantitative framework—Bias-variance trade-off:**

← Feature selection	Feature generation →	
Bias ↑	Bias ↓	→ Complexity
Variance ↓	Variance ↑	
Training error ↑	Training error ↓	

Test error has a U-shape ↘ ↗

  - ▷ *Bias-variance decomposition of expected test MSE:*

$$\mathbb{E}_{\text{Tr},Y_0}\left[\left(Y_0-\hat{f}(\mathbf{X}_0)\right)^2\right] = \overbrace{[\text{Bias}_{\text{Tr}}(\hat{f}(\mathbf{X}_0))]^2+\text{Var}_{\text{Tr}}[\hat{f}(\mathbf{X}_0)]}^{\text{reducible error}} + \overbrace{\text{Var}(\varepsilon_0)}^{\text{irreducible error}}$$

Quantity	Bias	Variance
Mathematical definition	Difference between the expected value of prediction and the true value of signal function	Amount of variability of prediction
Significance in PA	Part of the test error caused by the model *not being flexible enough* to capture the signal (underfitting)	Part of the test error caused by the model being *too complex* (overfitting)

- **Practical implications of bias-variance trade-off:**

  Need to set model complexity to a reasonable level

  $\Downarrow$

  $\begin{cases} \text{optimize bias-variance trade-off} \\ \text{avoid underfitting \& overfitting} \end{cases} \Rightarrow$ improve prediction performance

- **Sidebar: Dimensionality vs. granularity**
  - ▷ Granularity $\uparrow \Rightarrow$ model complexity tends to $\uparrow$
  - ▷ Two main differences between the two concepts:

Concept	Applicability	Comparability
Dimensionality	Specific to categorical variables	Two categorical variables can always be ordered by dimension.
Granularity	Applies to both numeric and categorical variables	Not always possible to order two variables by granularity

1.5 Model Validation

- **Aim:** To check that the selected model has no obvious deficiencies and the model assumptions are largely satisfied.
- **Validation method based on the training set:** For a "nice" GLM, the deviance residuals should:

  ① *(Purely random)* Have no systematic patterns.

  ② *(**Homoscedasticity**)* Have approximately constant variance upon standardization.

  ③ *(Normality)* Be approximately normal (for most target distributions).

  Check "**Residuals vs Fitted**" plot for ① & ②; **Q-Q plot** for ③.

- **Validation methods based on the test set:**
  - ▷ *Predicted vs. actual values of target:* The two sets of values should be close (can check this quantitatively or graphically).
  - ▷ *Benchmark model:* Show that the recommended model outperforms a benchmark model, if one exists (e.g., intercept-only GLM, purely random classifier), on the test set.

1.6 Recommendations for Next Steps

- *(Adjust the business problem)* Changes in external factors, e.g., market conditions, regulations, may cause initial assumptions to shift $\Rightarrow$ need to modify the business problem to incorporate the new conditions.
- *(Consult with subject matter experts)* Seek validation of model results from external subject matter experts.
- *(Gather additional data)* Enlarge training data with new obs. and/or variables, and retrain the model to improve robustness.

- *(Apply new types of models)* Try new types of models when new technology or implementation possibilities are available.
- *(Refine existing models)* Try new combinations or transformations of predictors, alternative hyperparameter values, alternative accuracy measures, etc.
- *(Field test proposed model)* Implement the recommended model in the exact way it will be used to gain users' confidence.

2 Specific Types of Model

2.1 GLMs

- **Assumptions: LMs vs. GLMs**

	LMs	GLMs
Independence	Given the predictor values, the observations of the target variable are independent. (Same for both LMs and GLMs.)	
Target distribution	Given the predictor values, the target variable follows a normal distribution.	Given the predictor values, the target distribution is a member of the linear exponential family.
Mean	The target mean directly equals the linear predictor: $\mu = \eta = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$.	A function ("link") of the target mean equals the linear predictor: $\underbrace{g}_{\text{link}}(\mu) = \underbrace{\eta}_{\text{linear predictor}}$.
Variance	Constant, regardless of the predictor values	Varies with μ and the predictor values

(**Note:** The link function in a GLM is applied to the target mean μ; the target variable itself is not transformed. ⚠)

- **Two key components:**

  ① ***Target distribution***: Choose one (in the linear exponential family) that aligns with the characteristics of the target.

  ② ***Link functions***: Some important considerations:

  ▷ Ensure the predictions match the range of values of the target mean.

  ▷ Ensure ease of interpretation, e.g., log link.

  ▷ (Minor) Canonical links make convergence more likely.

  (**Note:** The log link may or may not work when the target variable has zero values ⚠; see Exercise 4.1.4 (c) in the manual.)

- **Common e.g. of target distributions and link functions:**

Variable Type	Common Dist.	Common Link
Real-valued with a bell-shaped dist.	Normal (Gaussian)	Identity
Binary (0/1)	Binomial	Logit
Count (≥ 0, integers)	Poisson	Log
+ve, continuous with right skew	Gamma, inverse Gaussian	Log
≥ 0, continuous with a large mass at zero	Tweedie	Log

(**Note:** For gamma and inverse Gaussian, the target variable has to be strictly positive. Values of zero are not allowed. ⚠)

Feature generation

- **Methods for handling non-monotonic relations:** GLMs, in their basic form, assume that numeric predictors have a monotonic relationship with the target variable.

① ***Polynomial regression**:* Add polynomial terms to the model equation:

$$g(\mu) = \beta_0 + \beta_1 X \underbrace{+ \beta_2 X^2 + \cdots + \beta_m X^m}_{\text{polynomial terms}} + \cdots .$$

- ▷ *Pros:* Can take care of more complex relationships between the target variable and predictors. The more polynomial terms included, the more flexible the fit.
- ▷ *Cons:*
  - ☐ Coefficients become harder to interpret (all polynomial terms move together).
  - ☐ Usually no clear choice of m; can be tuned by CV (EDA can also help)

② ***Binning**:* "Bin" the numeric variable and convert it into a categorical variable with levels defined as non-overlapping intervals over the range of the original variable.

- ▷ *Pros:* No definite order among the coefficients of the dummy variables corresponding to different bins ⇒ target mean can vary highly irregularly over the bins.
- ▷ *Cons:*
  - ☐ Usually no clear choice of the no. of bins and the associated boundaries
  - ☐ Results in a loss of information (exact values of the numeric predictor gone)

③ *Adding **piecewise linear functions**:* Add features of the form $(X - c)_+$.

- ▷ *Pros:* A simple way to allow the relationship between a numeric variable and the target mean to vary over different intervals
- ▷ *Cons:* Usually no clear choice of the break points

- **Handling categorical predictors—Binarization::**
  - ▷ *How it works:* (Done in R behind the scenes.)

Categorical predictor

⇓

A collection of dummy (binary) variables
indicating one and only one level
(= 1 for that level, = 0 otherwise)

⇓

Dummy variables serve as predictors in model equation

  - ▷ ***Baseline level**:* The level at which all **dummy variables** equal 0.
    - ☐ *R's default:* The alpha-numerically first level
    - ☐ *Good practice:* Reset it to the most common level.

- **Interactions:**

Need to "manually" include interaction terms
of the product form $X_j X_k$

⇓

Coefficient of X_j will vary with the value of X_k

Interpretation of coefficients

- **General statements:**
  - ▷ Coefficient estimates capture the effects (magnitude + direction) of features on the target mean.
  - ▷ p-values express statistical significance of features; the smaller, the more significant.

- **Specific statements based on log link:** Assume all else equal.
  - ▷ *Numeric case:* For a unit change in a numeric predictor with estimated coefficient $\hat{\beta}_j$,

$$\begin{matrix}\text{multiplicative change}\\ \text{in target mean}\end{matrix} = e^{\hat{\beta}_j}, \qquad \begin{matrix}\text{\% change}\\ \text{in target mean}\end{matrix} = e^{\hat{\beta}_j} - 1.$$

  - ▷ *Categorical case:* For a non-baseline level of a categorical predictor with estimated coefficient $\hat{\beta}_j$,

$$\underset{\text{@non-baseline level}}{\hat{\mu}} = e^{\hat{\beta}_j} \times \underset{\text{@baseline level}}{\hat{\mu}}.$$

Other modeling techniques: Offsets vs. weights

	Offsets	**Weights**
Form of the target variable	Aggregate (e.g., **total** # claims in a group of similar policyholders)	Average (e.g., **average** # claims in a group of similar policyholders)
Do they affect the target mean or variance?	Target **mean** is directly proportional to **exposure**, e.g., with log link, $\mu_i = E_i \exp(\cdots)$.	**Variance** is inversely related to exposure: $\text{Var}(Y_i) = \dfrac{\text{(some terms)}}{E_i}$. Observations with a larger exposure will play a more important role in model fitting.

Stepwise selection

- **Selection process:** Sequentially add/drop features, one at a time, until there is no improvement in the selection criterion.

	Area	**Backward**	**Forward**
1.	Which model to start with?	Full model	Intercept-only model
2.	Add or drop variables?	Drop	Add
3.	Which method tends to produce a simpler model?	Forward selection	

- **Selection criteria based on penalized likelihood:**
  - ▷ *Idea:* Prevent overfitting by requiring an included/retained feature to improve model fit by at least a specified amount.
  - ▷ *Two common choices:*

Criterion	Definition	Penalty per Parameter
AIC	$-2l + 2(p+1)$	2
BIC	$-2l + [\ln(n_{\text{tr}})](p+1)$	$\ln(n_{\text{tr}})$

  (In R, $-2l$ is treated as the deviance.)

  - ▷ *AIC vs. BIC:*
    - ☐ For both, the lower the value, the better.
    - ☐ BIC is more conservative and results in simpler models.

- **Manual binarization:** Convert factor variables to dummy variables manually *before* running stepwise selection.
  - ▷ *Pros:* To be able to add (resp. drop) individual factor levels that are statistically significant (resp. insignificant) w.r.t. baseline level
  - ▷ *Cons:*
    - ☐ More steps in the `stepAIC()` procedure

- ☐ Possibly non-intuitive results (e.g., only few levels of a factor are retained)

Regularization

- **Idea:** Reduce overfitting by shrinking the size of the coefficient estimates, especially those of non-predictive features.
- **How it works:** To optimize training loglikelihood (equivalently, training deviance) adjusted by a penalty term that reflects the size of the coefficients, i.e., to minimize

$$\text{deviance} + \text{regularization penalty}.$$

The formulation serves to strike a balance ⚖ between goodness of fit and model complexity.

- **Common forms of penalty term:**

Method	Penalty	Characteristic
Lasso	$Ⓛ = \lambda \sum_{j=1}^{p} \lvert\beta_j\rvert$	Some coef. may be zero
Ridge regression	$Ⓡ = \lambda \sum_{j=1}^{p} \beta_j^2$	None reduced to zero
Elastic net	$\alpha Ⓛ + (1-\alpha) Ⓡ$	Some coef. may be zero

- **Two hyperparameters:**

① λ: Regularization (a.k.a. shrinkage) parameter

- ▷ Controls the amount of regularization:

$$\lambda \uparrow \overset{\text{more shrinkage}}{\Rightarrow} \text{complexity} \downarrow \Rightarrow \begin{cases} \text{bias}^2 \uparrow \\ \text{variance} \downarrow \end{cases}.$$

- ▷ ***Feature selection property:*** For elastic nets with $\alpha > 0$ (lasso, in particular), some coefficient estimates become exactly zero when λ is large enough.
- ▷ *Typically tuned by CV:* Choose λ with the smallest CV error.

② α: Mixing parameter

- ▷ Controls the mix between ridge ($\alpha = 0$) and lasso ($\alpha = 1$) (**Note:** Need to remember $\alpha = 0$ is ridge regression and $\alpha = 1$ is lasso. ⚠)
- ▷ Provided that λ is large enough, increasing α from 0 to 1 makes more coefficient estimates zero.
- ▷ Cannot be tuned by `cv.glmnet()`; need to tune manually.

2.2 Single Decision Trees

- **Basics:**
  - ▷ *Idea:* Divide ✂ the feature space into a set of non-overlapping regions containing relatively homogeneous observations (w.r.t. target).
  - ▷ *Deliverable:* A set of classification rules based on the values/levels of predictors and represented in the form of a "tree" 🌲
  - ▷ *Predictions:* Observations in the same terminal node share the same predicted mean (for numeric targets) or same predicted class (for categorical targets).
- **Recursive binary splitting:**
  - ▷ *Two terms:* The algorithm is...
    - ☐ ***Greedy:*** At each step, adopt the split that leads to the greatest reduction in impurity at that point, instead of looking ahead and selecting a split that results in a better tree in a future step. (Repeat until a stopping criterion is reached.)
    - ☐ ***Top-down:*** Start from the "top" of the tree, go "down," and sequentially partition the feature space in a series of splits.

▷ *Node impurity measures:*

Tree Type	Name of Measure	Formula
Regression	Residual sum of squares	$\sum_{i \in R_m}(y_i - \hat{y}_{R_m})^2$
Classification	**Classification error rate**	$1 - \max_{1 \le k \le K} \hat{p}_{mk}$
	Gini index	$\sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$
	Entropy	$-\sum_{k=1}^{K} \hat{p}_{mk} \log_2(\hat{p}_{mk})$

Properties:

- ☐ The smaller, the purer the observations in the node.
- ☐ Gini index and entropy are similar numerically.
- ☐ Gini index and entropy are more sensitive to node impurity than classification error rate
  Reason: They depend on all $\hat{p}_{mk}$, not just the max. class proportion.

- **Tree parameters:**

Parameter	Name in R	Meaning	Effect
Minimum bucket size	`minbucket`	Min. # obs. in a terminal node	Higher, tree less complex
Complexity parameter	`cp`	Min. improvement required for a split to be made (not 100% right...)	Higher, tree less complex
Maximum depth	`maxdepth`	# edges from root node to furthest node	Higher, tree more complex

▷ Be sure to know how these parameters limit tree complexity!

▷ `cp` can be tuned by CV within `rpart()`; `minbucket` and `maxdepth` have to be tuned by trial and error.

- **Interpretation of trees:** Things you can comment on:

▷ No. of tree splits

▷ Split sequence, e.g., start with X_1, further split the larger bucket by X_2, ...

▷ Which are the most important predictors (usually those in early splits)?

▷ Which terminal nodes have the most observations? Any sparse nodes?

▷ Any prominent interactions?

▷ *(Classification trees)* Combinations leading to the +ve event

- **Cost-complexity pruning:**

▷ *Rationale:* To reduce tree complexity by pruning branches from bottom that do not improve goodness of fit by a sufficient amount ⇒ prevent overfitting and ease interpretation.

▷ *How it works:*

Step 1. Grow a large tree T_0. (**Note:** Don't miss this step. ⚠)

Step 2. Minimize the penalized objective function

$$\underbrace{\text{relative training error}}_{\text{(model fit to training data)}} + \underbrace{c_p \times |T|}_{\text{(tree complexity)}},$$

over all subtrees of T_0, where

$$\text{training error} = \begin{cases} \text{RSS}, & \text{for regression,} \\ \#\text{ misclassifications}, & \text{for classification.} \end{cases}$$

▷ *About the hyperparameter* `cp`*:*

- ☐ `cp` ↑ ⇒ tree less complex (smaller)
- ☐ *Typically tuned by CV:* Set `cp` to the value that minimizes CV error (`xerror` in `cptable`).

▷ *Alternative:* ***One-standard-error (1-SE) rule***

- ☐ *How:* Select the smallest tree whose CV error is within 1 SE of the minimum CV error.
- ☐ *Rationale:* Select a simpler and more interpretable tree with comparable prediction performance. (Occam's razor)

- **Do variable transformations affect GLMs and trees?**

	GLMs	Trees
Transformations on target variable	Yes (The transformations alter the values of the predictors and target variable that go into the likelihood function.)	Yes (The transformations can alter the calculations of node impurity measures, e.g., RSS, that define the tree splits.)
Transformations on predictors	Yes (Same reasoning as above)	Yes, unless the transformations are monotonic, e.g., log (Monotonic transformations will not change the way tree splits are made.)

2.3 Ensemble Trees

Random forests

- **Idea:**

▷ *(Variance reduction)* Combine the results of multiple trees fitted to different **bootstrapped training samples** in parallel ⇒ reduce variance of overall predictions.

▷ *(Randomization)* Take a random sample of predictors as candidates for each split ⇒ reduce correlation between base trees ⇒ further reduce variance of overall predictions.

- **Combining base predictions to form overall prediction:**

▷ *Case 1 (Regression trees):* By averaging:

$$\hat{f}_{\text{rf}}(\mathbf{x}) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(\mathbf{x}).$$

▷ *Case 2 (Classification trees):* Two methods:

Probability		Class
base probabilities	$\xrightarrow{\text{(converted based on cutoff)}}$	base classes
$\downarrow$ (averaged)		$\downarrow$ (take "majority vote")
average probability	$\xrightarrow{\text{(converted based on cutoff)}}$	overall class

(The default is to take the majority vote.)

- **Key parameters:**

▷ `mtry`: # features sampled as candidates at each split

- ☐ Lower `mtry` ⇒ greater variance reduction
- ☐ **Common choice**: $\sqrt{p}$ (classification) or $p/3$ (regression)
- ☐ Typically tuned by CV

▷ `ntree`: # trees to be grown

- ☐ Higher `ntree`, more variance reduction
- ☐ Often overfitting does not arise even if set to a large no.
- ☐ Set to a relatively small value to save run time

Boosting

- **Idea:**

▷ In each iteration, fit a tree to the residuals of the preceding tree and subtract a scaled-down version of the current tree's predictions from the residuals to form the new residuals.

▷ Each tree focuses on observations the previous tree predicted poorly.

▷ Overall prediction: $\hat{f}(\mathbf{x}) = \sum_{b=1}^{B}\lambda\hat{f}^{b}(\mathbf{x})$.

- **Key parameters**:
  - ▷ `eta`: Learning rate (or shrinkage) parameter
    - ☐ *Effects of* `eta`*:* Higher `eta` ⇒ algorithm converges faster but is more prone to overfitting.
    - ☐ *Rule of thumb:* Set to a relatively small value
  - ▷ `nrounds`: Max. # rounds in the tree construction process
    - ☐ *Effects of* `nrounds`*:* Higher `nrounds` ⇒ algorithm learns better but is more prone to overfitting.
    - ☐ *Rule of thumb:* Set to a relatively large value
- **Random forests vs. boosted trees:**

Item	Random Forest	Boosting
Fitting process	In parallel	In series (sequential)
Focus	Variance	Bias
Overfitting	Less vulnerable	More vulnerable
Hyperparameter tuning	Less sensitive	More sensitive

Two interpretational tools for ensemble trees

- **Variable importance plots**:
  - ▷ *Definition of importance scores:* The total drop in node impurity (RSS for regression trees and Gini index for classification trees) due to splits over a given predictor, averaged over all base trees:

$$\begin{array}{c}\text{importance}\\\text{score}\end{array} = \frac{1}{B} \times \sum_{\substack{\text{all splits over}\\\text{that predictor}}} \begin{array}{c}\text{impurity}\\\text{reduction}\end{array}.$$

  - ▷ *Use:* To identify important variables (those with a large score)
  - ▷ *Limitation:* Unclear how the important variables affect the target.
- **Partial dependence plots**:
  - ▷ *Definition of partial dependence:* Model prediction obtained after averaging the values/levels of variables not of interest:

$$\mathrm{PD}(x_1) := \frac{1}{n_{\mathrm{tr}}} \sum_{i=1}^{n_{\mathrm{tr}}} \hat{f}(\underbrace{x_1}_{\text{fixed}}, \underbrace{x_{i2}, \ldots, x_{ip}}_{\text{averaged}}).$$

  - ▷ *Use:* Plot $\mathrm{PD}(x_1)$ against various x_1 to show the marginal effect of X_1 on the target variable.
  - ▷ *Limitations:*
    - ☐ Assume predictor of interest is independent of other predictors.
    - ☐ Some predictions may be based on practically unreasonable combinations of predictor values.

2.4 Pros and Cons of Different Models

- **Tips for recommending a model:** Refer to the business problem (prediction vs. interpretation) and characteristics of data (e.g., any complex, non-monotonic relations?)
- **GLMs:**
  - ▷ *Pros:*
    1. *(Target distribution)* GLMs excel in accommodating a wide variety of distributions for the target variable.
    2. *(Interpretability)* The model equation clearly shows how the target mean depends on the features; coefficients = interpretable measure of directional effect of features.
    3. *(Implementation)* Simple to implement
  - ▷ *Cons:*
    1. *(Complex relationships)* Unable to capture non-monotonic (e.g., polynomial) or non-additive relationships (e.g., interaction), unless additional features are manually incorporated.

② *(Interpretability)* For some link functions (e.g., inverse link), the coefficients may be difficult to interpret.

- **Regularized GLMs:**
  - ▷ *Pros:*
    ① *(Categorical predictors)* Via the use of model matrices, binarization of categorical variables is done automatically and each factor level treated as a separate feature to be removed.
    ② *(Tuning)* An elastic net can be tuned by CV using the same criterion (e.g., MSE, accuracy) ultimately used to judge the model against unseen test data.
    ③ *(Variable selection)* For elastic nets with $\alpha > 0$, variable selection can be done by making λ large enough.
  - ▷ *Cons:*
    ① *(Categorical predictors)* Possible to see some non-intuitive or nonsensical results when only a handful of the levels of a categorical predictor are selected.
    ② *(Target distribution)* Limited/restricted model forms allowed by `glmnet()` (Weak point!)
    ③ *(Interpretability)* Coefficient estimates are more difficult to interpret $\because$ variables are standardized. (Weak point!)

- **Single trees:**
  - ▷ *Pros:*
    ① *(Interpretability)* If there are not too many buckets, trees are easy to interpret because of the if/then nature of the classification rules and their graphical representation.
    ② *(Complex relationships)* Trees excel in handling non-monotonic and non-additive relationships without the need to insert extra features manually.
    ③ *(Categorical variables)* Categorical predictors are automatically handled by separating their levels into two groups without the need for binarization.
    ④ *(Variable selection)* Variables are automatically selected as part of the model building process. Variables that do not appear in the tree are filtered out and the most important variables show up at the top of the tree.
  - ▷ *Cons:*
    ① *(Overfitting)* Strongly dependent on training data (prone to overfitting) $\Rightarrow$ predictions unstable with a high variance $\Rightarrow$ lower user confidence
    ② *(Numeric variables)* Usually need to split based on a numeric predictor repeatedly to capture its effect effectively $\Rightarrow$ tree becomes large, difficult to interpret.
    ③ *(Categorical variables)* Tend to favor categorical predictors with a large no. of levels
    (*Reason:* Too many ways to split $\Rightarrow$ easy to find a spurious split that looks good on training data, but doesn't really exist in the signal.)

- **Ensemble trees:**
  - ▷ *Pros:* Much more robust and predictive than base trees by combining the results of multiple trees
  - ▷ *Cons:*
    ① Opaque ("black box"), difficult to interpret
    (*Reason:* Many base trees are used, but variable importance or partial dependence plots can help.)
    ② Computationally prohibitive to implement
    (*Reason:* Huge computational burden with fitting multiple base trees.)

3 Unsupervised Learning

- **Supervised vs. unsupervised learning:**

	Supervised	Unsupervised
Target	Present	Absent (or ignored if present)
Goal	To make inference or predictions for the target	To extract relationships between variables

- **Two reasons why unsupervised learning is often more challenging than supervised learning:**

① *(Objectives)* Objectives in unsupervised learning are more fuzzy and subjective (no simple goal like prediction).

② *(Hard to assess results)* Methods for assessing model quality based on the target variable (e.g., CV) are generally not applicable.

3.1 Principal Components Analysis (PCA)

- **Idea**:
  - ▷ To transform a set of numeric variables into a smaller set of representative variables (**PCs**) ⇒ reduce dimension of data
  - ▷ Especially useful for highly correlated data ⇒ a few PCs are enough to capture most information.

- **Properties of PCs:**
  - ▷ Linear combinations of the original features:

$$z_{im} = \phi_{1m}x_{i1} + \phi_{2m}x_{i2} + \cdots + \phi_{pm}x_{ip}.$$

with $\phi_{1m}^2 + \phi_{2m}^2 + \cdots + \phi_{pm}^2 = 1$ (normalization).

  - ▷ Generated to capture as much information in the data (w.r.t. variance) as possible
  - ▷ Mutually uncorrelated (different PCs capture different aspects of data)
  - ▷ Relationship between **PC scores** and **PC loadings**:

$$\underset{\text{(scores)}}{\mathbf{z}_m} = \mathbf{X} \underset{\text{(loadings)}}{\boldsymbol{\phi}_m}.$$

  - ▷ **Amount of variance explained** decreases with PC order, i.e., PC1 explains the **most variance** and subsequent PCs explain less and less.

- **Two applications of PCA:**

① *EDA:* Plot the scores of the 1st PC vs. the scores of the 2nd PC to gain a 2D view of the data in a scatterplot.

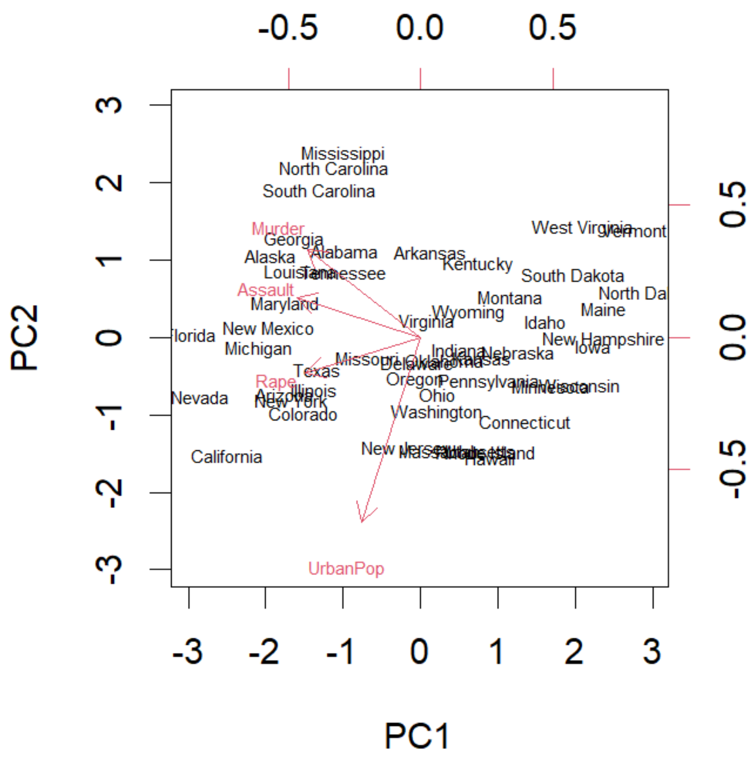
② *Feature generation:* Replace the original variables by PCs to reduce overfitting and improve prediction performance.

- **Interpretation of PCs:**
  - ▷ *Signs and magnitudes of PC loadings:* What do the PCs represent, e.g., proxy, average, or contrast of which variables? Which variables are more correlated with one another?
  - ▷ *Sizes of* ***proportions of variance explained*** *(PVEs):*

$$\text{PVE}_m = \frac{\text{Variance explained by } m\text{th PC}}{\text{Total variance}}.$$

Are the first few PVEs large enough (related to the strong correlations between variables)? If so, the PCs are useful.

- **Biplots**: Visualization of PCA output by displaying both the scores and loading vectors of the first two PCs. Example:



▷ PC loadings on top and right axes ⇒ deduce meaning of PCs

▷ PC scores on bottom and left axes ⇒ deduce characteristics of observations (based on meaning of PCs)

- **Number of PCs (M) to use:**

  ▷ *Trade-off:* $M \uparrow \Rightarrow \begin{cases} \text{cumulative PVE} \uparrow \\ \text{dimension} \uparrow \\ \text{(if } y \text{ exists) model complexity} \uparrow \end{cases}$

  ▷ *How to choose M:*

  ☐ ***Scree plot:*** Eyeball the plot and locate the "elbow" (point at which the PVEs of subsequent PCs have dropped off to a sufficiently low level).

  ☐ *CV:* Treat M as a hyperparameter to be tuned if y exists.

- **Drawbacks of PCA:**

  ▷ Loss of interpretability
  (*Reason:* PCs as composite variables can be hard to interpret.)

  ▷ Not good for non-linearly related variables
  (*Reason:* PCs rely on linear transformations of variables.)

  ▷ PCA does dimension reduction, but not feature selection.
  (*Reason:* PCs are constructed from all original features.)

  ▷ Target variable is ignored. (*Remember:* PC is unsupervised.)

3.2 Cluster Analysis

- **Idea:**

  ▷ To partition observations into a set of non-overlapping subgroups ("clusters") and uncover hidden patterns.

  ▷ Observations within each cluster should be rather similar to one another.

  ▷ Observations in different clusters should be rather different (well separated).

- **Two feature generation methods based on clustering:**

  ▷ *Cluster groups:* As a new factor variable

  ▷ *Cluster means:* As a new numeric variable

K-means clustering

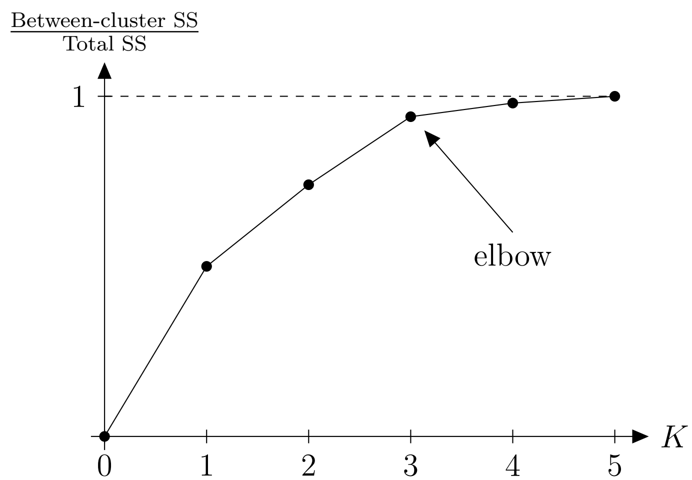
- **Idea:** For a fixed K (a +ve integer), choose K clusters $C_1, \ldots, C_K$ to minimize the total within-cluster SS, $\sum_{k=1}^{K} W(C_k)$.

- **How the algorithm works:**
  - ▷ *Step 1 (Initialization):* Given K, randomly select K points in the feature space as initial cluster centers.
  - ▷ *Step 2 (Iteration):* Repeat the following steps until the cluster assignments no longer change:
    - (a) Assign each obs. to the cluster with the closest center.
    - (b) Recalculate the K cluster centers (hence "K-means").
- **Good practice:** Set `nstart` to a large integer, e.g., ≥ 20. Reason:

  The algorithm produces a local optimum, which depends on the randomly selected initial cluster centers.

  ⇓

  Run the algorithm multiple times to improve the chance of finding a better local optimum.
- **Selecting the value of K by elbow method:**
  - ▷ Make a plot of the proportion of variation explained ($= \dfrac{\text{between-cluster SS}}{\text{total SS}}$) against K.



  - ▷ Choose the "elbow," beyond which the proportion of variation explained is marginal.

Hierarchical clustering

- **Idea:**
  - ▷ *Algorithm:*
    - ☐ Start with the individual observations, each treated as a separate cluster.
    - ☐ Successively fuse the closest pair of clusters, one at a time.
    - ☐ Stop when all clusters are fused into a single cluster containing all observations.
  - ▷ *Output:* A "hierarchy" of clusters which can be visualized by a dendrogram
- **Linkage:** To measure the dissimilarity between two clusters, at least one of which has ≥ 2 observations

Linkage	The **Inter-cluster Dissimilarity** Is...
Complete (default)	Maximal pairwise distance
Single	Minimal pairwise distance
Average	Average of all pairwise distances
Centroid	Distance between the two cluster centroids

  - ▷ Complete and average linkage are commonly used. (*Reason*: They tend to result in more balanced clusters.)
  - ▷ Single linkage tends to produce extended, trailing clusters with single observations fused one-at-a-time.
  - ▷ Centroid linkage may lead to inversion (some later fusions occur at a lower height than an earlier fusion).
- **Dendrogram:** An upside-down tree showing the sequence of fusions and the inter-cluster dissimilarity ("Height") when each fusion occurs on the vertical axis.

Some insights from a dendrogram:

▷ *(Similarities between clusters)* Clusters joined towards the bottom of a dendrogram are rather similar to one another, while those fused towards the top are rather far apart.

▷ *(Considerations when choosing the no. of clusters)* Try to cut the dendrogram at a height such that:

- ☐ The resulting clusters have similar no. of obs. (balanced)
- ☐ The difference between the height and the next threshold should be large enough ⇒ obs. in different clusters have materially different characteristics.

- **K-means vs. hierarchical clustering:**

Item	K-means	Hierarchical
Is randomization needed?	Yes (for initial cluster centers)	No
Is the no. of clusters pre-specified?	Yes (K needs to be specified)	No (Specify the height of the dendrogram later)
Are the clusters nested?	No	Yes (a hierarchy of clusters)

Other issues

- **Scaling of variables matters for both PCA and clustering**

▷ *Without scaling:*

Variables with a large order of magnitude will dominate variance and distance calculations

⇓

have a disproportionate effect on PC loadings & cluster groups

▷ *With* ***scaling*** *(generally recommended):* All variables are on the same scale and share the same degree of importance.

- **Alternative distance measures:**
Correlation-based distance

▷ *Motivation:* Focuses on shapes of feature values rather than their exact magnitudes.

▷ *Limitation:* Only makes sense when $p \geq 3$, for otherwise the correlation between two observations always equals ± 1.

- **Clustering and curse of dimensionality:**

▷ Visualization of the results of cluster analysis becomes problematic in high dimensions ($p \geq 3$).

▷ As the number of dimensions increases, our intuition breaks down and it becomes harder to differentiate between observations that are close and those that are far apart.

♢ ♣ ♡ ♠ ☻ **THE END** ☺ ♠ ♡ ♣ ♢