PA/SRM 2022 Study Notes

Actuarial Predictive Models

Terence Lim

December 2022

PRELIMINARY AND INCOMPLETE

THIS IS ONLY A ROUGH OUTLINE FOR PERSONAL USE:
IT DOES CONTAIN ERRORS AND TYPOS

https://terence-lim.github.io

Contents

Terminology

Bias-Variance trade-off – the challenge of finding a model for which both the variance and squared variance of test set performance is low.

Training data used to develop the model – run through algorithms, adjust hyperparameters

Validation data used to measure fitness of algorithms and selected parameters

Testing data should only be used one at the end of model development process – final measure of model performance

Cross-Validation estimates the test error rate by holding a subset of training observations from the fitting process

Train-Validate-Test:

1. Split the data into training and test sets.
2. Split the training data into k folds (for k-fold cross validation) – find model and hyperparameters with lowest CV error
3. Train a model on all the training data using the optimal hyperparameters
4. Test the model on the test data

Data

Types

- Numeric (continuous, discrete)
- Nonnumeric
  - Character (string)
  - Factor (categorical): Ordered or Unordered (levels)
- Boolean/binary/indicator
  - Binarized
- Date/time, geospatial (latitude and longitude)
- Unstructured: Information that doesn't naturally fit into a tabular structure, e.g. text and video

Dimensionality

- Number of variables (columns) in data set
- Categorical variable: high dimensional (number of levels), granularity
- Low exposure (or occurrence)
- Unstable and unintuitive results when large number of variable levels
- Difficult to comprehend

Unsupervised Feature Construction

PCA

- Summarizes data set (i.e. dimension reduction) by maximizing variance
- Creates numerical features called principal components
- Original variables are typically scaled
- Principal components are uncorrelated to each other
- Consecutive principal components explain less variance
- Use scree plot to visualize variance explained and determine smallest number required
- Can be used to address collinearity

Cluster

- Group observations that are similar in the same cluster
- Creates a factor based on the groupings
- Original variables are typically scaled

Principal Components Analysis

- Variance is a measure of the spread of a data distribution
- Projection onto a direction of maximal variance minimizes distance from an old higher dimensional data point to its new transformed value
  - Minimizes information loss, retaining maximum amount of information from the original data
  - These directions are just linear combinations of the input variables and can be used as summary variables themselves
- PCA finds directions of maximum variance that are mutually orthogonal (perpendicular)
- General properties:
  - Maximum number of PC's is min(number of variables, number of data points)
  - The more variance of data explained, the higher that PC is ranked
- Assume each raw variable has been centered and scaled. If range and scale of the variables are similar or in same units of measure, covariance is appropriate need not scaling

Principal Components Analysis

- For the i'th row, the p'th PC is:
  - $PC_i^{(p)} = w_1^{(p)} x_{i1} + \ldots + w_n^{(p)} x_{in}$
  - Coefficients $w_j^{(p)}$ are called loadings or weights or rotations for the pth PC
  - Sum of squares of loadings constrained equal to one
- Because the average is zero due to centering, the goal is max $\sum_{i \in m} PC_i^{(p)2}$
  - Subject to $\sum_{j \in n} w_j^{(1)2} = 1$
  - Subsequence PC's determined same way with additional constraint the latest loadings must be orthogonal to the previous loadings: $\sum_{j \in n} w_j^{(r)} w_j^{(p)} = 0 for r = 1, 2..p-1$
- First component is normalized linear combination $z_1 = \sum_{i \in p} \phi_{i1} x_i$
  - normalized $\sum \phi^{(1)} = 1$
  - loadings ϕ_1 of first PC
  - maximum $\sum z_{i1}^2 \sim$ sample variance of z_{i1}
- Loading vector ϕ is direction in feature space where data varies most
- Project X onto this direction $\Rightarrow$ projected values are the scores z
- Low dimensional space closest to the n observations
- Z_2 is uncorrelated to $Z_1 \iff \phi_2$ is orthogonal to direction ϕ_1.

Principal Components Analysis

When to use:

- Feature transformation – PC's are linear combinations of original
- Feature extraction – principal components
- Feature selection – fewer variables needed to capture most information
- Visualize high-dimensional data
- Important preprocessing step for other algorithms that work better with fewer inputs
- Finding latent as opposed to measurable variables

K-Means Clustering

- Assign each observation to a similarity grouping: one of k groups
- K-means defines a group by its center, each observation is then assigned to the group with the closest center
- Find centers that minimize objective function, using iterative algorithm:
  - Pick some random centers
  - Assign each observation to the group that has the closest center
  - Calculate new centers for the groups formed in Step 2
  - If the difference in objective function is below some threshold, stop. Else repeat from reassign
- Objective is $\sum_{i \in k} \sum_{j \in p} (x_i j - \bar{x}_{kj})^2$ always improves, finds local minimum
- Algorithm needs to be repeated for different initial cluster assignments
- Should always standardize, so that algorithm places equal weights on all features when determining where clusters should be
- Algorithm needs to be repeated for each number of clusters k.
- Curse of dimensionality: as number of dimensions increases, the data points on average are the same distance away from each other, clustering techniques become almost meaningless
- Elbow method for selecting k – as soon as amount of additional variance being explained by a new cluster drops off

Hiearchical Clustering

- Agglomerative approach: starts by considering each observation as its cluster, then gradually grouping them with nearby clusters – bottom-up
- Linkage function – as clusters contain multiple points, closest clusters is ambiguous statement
  - Single: minimum distance between cluster points
  - Complete: distance between two further elements of two clusters
  - Average: Average pairwise dissimilarity between cluster points
  - Centroid: Dissimilarity between cluster centroids 00 but inversion can occur where fuse at height below individual clusters
- Creates clusters by joining the two closest observations, then the next two closest, until all observations are in a single cluster
- Algorithm does not need to be repeated for each number of clusters k
- Produces a dendrogram: similarity of observations from location on vertical axis where branches containing observations fuse. Height of cut controls number of clusters obtained
- Typically avoided for large datasets
- Not robust to perturbations in data

Hiearchical Clustering

Algorithm

1. Select the dissimilarity measure and linkage.
2. Treat each observation as its own cluster
3. For $k = n, n-1, ..., 2$:
   - Compute the inter-cluster dissimilarity between all k clusters
   - The two clusters with the lowest inter-cluster dissimilarity are fused; the dissimilarity indicates the height in the dendrogram at which the two clusters join

Choices

1. Should features be standardized?
2. What dissimilarity measure?
3. What linkage?
4. Where to cut dendogram to obtain clusters?

Review of OLS

- Linear relationship target and predictors
- Additive error term
- Independent observations
- Minimize SSE = SSR = $\sum_i (\hat{y} - y_i)^2$ (if gaussian noise, then is also MLE)
- X is called "design matrix"
- Check Assumptions:
- Mean of residuals is zero
  - Variance is constant (homoscedasticity) – residual plots
  - Residuals have no autocorrelation
  - Residuals and Predictor variables are uncorrelated
  - Residuals have a normal distribution – qq-plot
- Limitations
  - Target maybe positive, binary or variance depends on the mean
  - Linear models are sensitive to outliers
  - Nonlinear relationships (but may be transformations)

Estimation

The simple linear regression (SLR) model relates a continuous **response** variable y_i with one **predictor** variable x_i and an **error term** ϵ_i:
$y_i = f(x_i) + \epsilon_i = a + bx_i + \epsilon_i$
Coefficent estimates are chosen to minimize the residual sum of squares:

- Slope $\hat{b} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \hat{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$
- Intercept: $\hat{a} = \bar{y} - \hat{b}\bar{x}$

Residual is the difference between the observed response value and the response value predicted by the model, $e_i = y_i - \hat{y}_i$.
Residual sum of squares (RSS) over all observations is
$RSS = e_1^2 + e_2^2 + ... + e_n^2$ or equivalently $RSS = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$.
Mean square error (MSE) is an estimate of the variance of the residuals
$s^2 = \hat{\sigma}^2 = \frac{1}{n-2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$.

Confidence Intervals

Residual standard error (RSE) or residual standard deviation is the estimate of the (square root of the) variance of the residuals.
Standard errors associated with linear regression coefficient and mean response estimates are:

$$\text{Slope: } se(\hat{b}) = \sqrt{\frac{s^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}}$$

$$\text{Intercept: } se(\hat{a}) = \sqrt{s^2[\frac{1}{n} + \frac{\bar{x}^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}]}$$

$$\text{Mean response: } se(\hat{y}) = \sqrt{s^2[\frac{1}{n} + \frac{(x - \bar{x})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}]}$$

$$\text{Prediction of a new response: } se(\hat{y}_{n+1}) = \sqrt{s^2(1 + \frac{1}{n} + \frac{(x - \bar{x})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2})}$$

Inferences

t-statistic or t-ratio $t(b_j) = \dfrac{b_j}{se(b_j)}$ can be interpreted to be the number of standard errors that b_j is away from zero. In a t-test, the null hypothesis ($H_0 : \beta_j = 0$) is rejected in favor of the alternative if the absolute value of the t-ratio $|t(b_j)|$ exceeds a t-value, denoted $t_{n-(k+1),1-\frac{\alpha}{2}}$, equal to the $(1-\frac{\alpha}{2})$th percentile from the t-distribution using $df = n-(k+1)$ degrees of freedom.

F-test whether all regression slope coefficients are zero $H_0 : b_1 = ... = b_p = 0$, versus the alternative H_a : at least one b_j is non-zero. This hypothesis test is performed by computing the F-statistic.

$$F = \frac{(TSS - RSS)/p}{RSS/(n-p-1)}$$

Partial F-test to test that a particular subset of q of the coefficients are zero. We fit a second model that uses all the variables except those last q, then compute residual sum of squares for that model and the appropriate F-statistic

$$F = \frac{(RSS_q - RSS)/(p-q)}{RSS/(n-p-1)}$$

Linear Regression Assumptions

With multiple regressors, the linear model $y = b_0 + b_1 x_1 + ... + b_k x_k + \epsilon$ has coefficient estimates: $\hat{b} = (X^T X)^{-1} X^T y$

- The coefficient b_j quantifies the association between the jth predictor and the response. It is the average effect on Y of a one unit increase in X_j, holding all other predictors fixed.

Observables Representation	Error Representation
F1. $E[y_i] = b_0 + b_1 x_{i1} + ... + b_k x_{ik}$.	E1. $y_i = b_0 + b_1 x_{i1} + ... + b_k x_{ik} + \epsilon_i$.
F2. $\{x_{i1}, ..., x_{ik}\}$ are non-stochastic variables.	E2. $\{x_{i1}, ..., x_{ik}\}$ are nonstochastic variables.
F3. $Var(y_i) = \sigma^2$.	E3. $E[\epsilon_i] = 0$ and $Var(\epsilon_i) = \sigma^2$.
F4. $\{y_i\}$ are independent random variables.	E4. $\{\epsilon_i\}$ are independent random variables.
F5. $\{y_i\}$ are normally distributed.	E5. $\{\epsilon_i\}$ are normally distributed.

1. Under assumptions F1-F4, the least squares regression estimator $b = (X'X)^{-1}X'y$ is an unbiased estimator of the parameter vector b;
2. has variance-covariance matrix $Var(b) = \sigma^2 (X'X)^{-1}$,
3. and standard error b_j $se(b_j) = \sigma \sqrt{(X'X)^{-1}_{[j+1,j+1]}}$.
4. Under assumption F1-F5, the least squares stimator is normally distributed.

Collinearity

When one explanatory variable is, or nearly is, a linear combination of the other explanatory variable.

1. Does not preclude us from getting good fits nor from making predictions.
2. Estimates of error variances and tests of model adequacy are still reliable.
3. Standard errors of regression coefficients are greater, reducing power of the hypothesis test (probability of correctly detecting a non-zero coefficient)

The variance inflation factor (VIF) of each predictor variable to assess multicollinearity:

$$VIF_j = \frac{1}{1 - R_j^2}, j = 1, 2, ..., k$$

where R_j is the multiple correlation coefficient between x_j as the response and linear combinations of the other x's

- Equivalent to ratio of the variance of a coefficient estimate when fitting the full model divided by the variance of of the coefficient estimate if fit on its own.
- A larger VIF_j associated with a larger standard error of the jth slope, b_j.
- A VIF value that exceeds 5 or 10 indicates a problematic amount.

Residual Analysis

With the ith residual $e_i = y_i - \hat{y}_i$ and s the residual standard deviation, commonly-used definitions for standardizing residuals are:

standardized residual $\frac{e_i}{s\sqrt{1-h_{ii}}}$, where $h_{ii} = x_i^T (X^T X)^{-1} x_i$ is the ith **leverage** of the explanatory variables. The denominator is the estimated standard error for e_i.

studentized residual $\frac{e_i}{s_{(i)}\sqrt{1-h_{ii}}}$, where $s_{(i)}$ is the residual standard deviation when running a regression after deleting the ith observation. Omits effect on the denominator from a large residual.

Potential problems for a linear regression model

1. Non-linearity of the data
2. Correlation of error terms
3. Non-constant variance of error terms (heteroscedasticity)
4. Outliers
5. High leverage points

Heteroskedasticity

How to test for heteroscedasticity:

1. Fit a regression model and calculate the model residuals, e_i.
2. Calculate squared standardized residuals, $e_i^{*2} = e_i^2/s^2$.
3. Fit a regression model of e_i^{*2} on a known vector of p variables z_i.
4. The test statistic is $LM = (SSR_z)/2$, where SSR_z is the regression sum of squares from the model fit in the previous step.
5. Reject the null hypothesis if LM exceeds a percentile from a chi-square distribution with p degrees of freedom. The percentile is one minus the significance level of the test.

Heteroscedasticity may be handled by:

1. Define the empirical, or robust, estimate of the variance covariance matrix as $\hat{Var}(b) = (X'X)^{-1}(\sum_{i=1}^{n} e_i^2 x_i x_i')(X'X)^{-1}$ to adjust the corresponding "heteroscedasticity-consistent" standard errors.
2. Use a variation of least squares estimation by weighting observations.
3. Transform the dependent variable, typically with a logarithmic transformation.

Outliers

An outlier is a point for which its response value is far from the value predicted by the model. Outliers can arise for a variety of reasons, such as incorrect recording of an observation during data collection. Removing the outlier may (or may not) have little effect on the least squares line, but can affect the RSE and hence interpretation of the fit.
Options for handling outliers:

1. Include the observation in the usual summary statistics but comment on its effects.
2. Delete the observation from the dataset.
3. Create a binary variable to indicate the presence of an outlier.

Leverage

Removing the high leverage observation may have substantial impact on the estimated least squares line.

Hat matrix $H = X(X^TX)^{-1}X^T$, is the projection matrix that expresses the values of the response variable as linear combinations of the predictor variables: $\hat{y} = Hy$.

Leverage for$i$th observation is $h_{ii} = x_i'(X'X)^{-1}x_i$.A widely adopted convention is to declare an observation to be a high leverage point if the leverage exceeds three times the average, that is, if $h_{ii} > 3(k+1)/n$.

Cook's distance considers both the response and the explanatory variables: $D_i = \frac{\sum_{j=1}^{n}(\hat{y}_j - \hat{y}_{j(i)})^2}{(k+1)s^2} = (\frac{e_i}{se(e_i)})^2 \frac{h_{ii}}{(k+1)(1-h_{ii})}$, where $\hat{y}_{j(i)}$ is the prediction of the jth observation.

Handling high leverage points:

1. Include observation in the summary statistics but comment on its effect.
2. Delete the observation from the datase if deemed not representative of some larger population.
3. Choose another variable to represent the information.
4. Use a nonlinear transformation of an explanatory variable

Model Evaluation and Validation

Model Selection and Validation

- Hypothesis test, significance of predictor, t-test, p-value
- R-square, adjusted-R-square, F-test
- Deviance is measure of goodness of fit for GLM (similar to SSE for OLS)
- Penalized likelihood by number of parameters
  - AIC: based on deviance, penalizes more complicated model $= 2k - 2\ln L$
  - BIC is more conservative and results in simpler models $= \ln(n)k - 2\ln L$

Null and Residual Deviance, and AIC

- Null deviance $= 2(LL_{sat} - LL_{null})$
  - Saturated model has one parameter per observation
  - Null model is intercept-only model
- Residual deviance $= 2(LL_{sat} - LL)$
- $AIC = 2p - 2LL$ (low AIC is good $\Rightarrow$ high LL and small p)

Residuals

Types of residuals

- Response = observed – predicted
- Working = (observed – predicted) / predicted
- Pearson = (observed – predicted) / sqrt(predicted)
- Deviance = sign(observed – predicted) deviance

Fixing Residuals

- Heteroscedasticity: transform a variable
- Nonlinear: transform variable, or need a non-linear model
- Outlier: data entry error, assess impact (do model coefficients change)
- Residuals vs Fitted graph: check homogeneity of variance and linearity of relationship
- Normal Q-Q: normality of distribution of residuals
- Scale-Location: Similar to Residuals vs Fitted, but with Y-axis standardized
- Residuals vs Leverage: detect observations with large impact on model and considered for removal

Feature Selection

- Forward and backward stepwise selection
- Regularization: glmnet() to penalize size of coefficients

  Ridge Regression $(X'X + \lambda I)^{-1}X'Y$: penalize sum of squared parameters

$$\min SSE + \lambda \sum_{j=1}^{p} b_j^2$$

  Lasso penalize absolute value of parameters – allows for deletion of feature

$$\min SSE + \lambda \sum_{j=1}^{p} |b_j|$$

  Elastic net mixing coefficient alpha (0 is ridge, 1 is lasso)

Regularization

Model Complexity, Bias and Variance

- Loss: Average squared difference between the predicted values and the target values
- Bias: expected loss from model not being flexible enough to capture the underlying signal
- Variance: Expected loss from the model overfitting to a specific instance of the data
- Process noise: When making predictions, there is always some error that cannot be avoided, even if we know the true model and parameters
- Cp, AIC, BIC: performs an adjustment to the training error that accounts for potential overfitting: hence variable selection is an indirect method for reducing variance
- Regularization: optimize variance versus bias – reduce complexity
  - Cross-validation to estimate true model variance, and...
  - ...hyperparameter tuning (i.e. control the amount of regularization)

Reducing model complexity:

- Potentially improve prediction accuracy
- More interpretable models

Generalized Linear Models

Generalized assumptions: Given predictor variables,

- target variables are independent (unchanged)
- target variable's distribution is member of exponential family (e.g. normal)
- expected value of target variable is $E[Y] = g^{-1}(X\beta)$ (e.g. identity)
  - where g is called the (canonical) link function
  - and g^{-1} is its inverse (aka mean function)

Definition: Linear separable if straight line, or hyperplane in more than two dimensions, completely separates data

Examples

Log link function and normal distribution

- $Y \sim N(exp(X\beta), \sigma^2)$ is linear but may be negative (need gamma distribution for non-negative)
- Note: OLS with $\log Y \Rightarrow Y = (\Pi \exp(X\beta)) \exp(\epsilon)$ is not linear model

Logit link (logistic sigmoid mean)

- Binomial distribution for binary variable

Reciprocal link (reciprocal mean)
Log link (exponential mean): ensures positive prediction

- Poisson distribution for count variables
- Gamma or inverse Gaussian for positive numeric with right skew

Counts and Aggregate Loss

Claim counts:

- poisson distribution (positive target) and log link (positive prediction): variance = mean
- Gamma distribution with log link (other distributions for positive-valued data: lognormal, inverse gaussian)
- Quasipoisson family – account for overdispersion = variance of response ¿ mean

Tweedie Distribution – unique application for insurance

- Compound Poisson Linear Models: R package cplm not in exam
- N claims poisson(lambda), each claim has gamma (alpha, beta) *Rightarrow* mu = lambda X alpha/beta
- Discrete probability at zero, then continuous probability on positive total claims values

Interpretation of coefficients

- Not "is this variable valuable", but "in the presence of the other variables, does this variable provide additional value?""
- With log link, expected change in response is multiplicative by a factor of exp(beta)
- When predictor is binary, effect is a separation of the response variable by group represented
- With factor variable, each beta represents difference with the base group (the one left out as predictor)
- An interaction occurs when the response depends on a combination of features, not just in isolation
- With logit link: exponentiated coefficients approximate (ignoring denominator term) multiplicate effect of a unit change in the variable

Target Distributions and Link Function

Mean Functions and Canonical Links for Selected Distributions:

Distribution	Mean function $b(\theta)$	Canonical link $g(\mu)$	Model
Normal	θ	μ	linear regression
Bernoulli	$e^{\theta}/(1+e^{\theta})$	$\text{logit}(\mu)$	logistic regression
Poisson	e^{θ}	$\ln \mu$	poisson regression
Gamma	$-1/\theta$	$-1/\mu$	
Inverse Gaussian	$(-2\theta)^{\frac{1}{2}}$	$-1/(2\mu^2)$	

Weights and Offsets

Offset variables

- incorporates a measure of exposures (larger exposure $\Rightarrow$ larger mean)
- at times a coefficient's value is known and does not need to be estimated
- e.g. Poisson GLM with log link: $\ln \mu = \ln E + \eta$ (assumes μ proportional to E)

Prior weights

- incorporate a measure of precision. more precise observations higher weight
- give information about credibility of each observation, model will be more influenced by these observations

Model Assessment in Binary Classification

- TN: true negatives
- FN: false negatives, i.e. type II errors
- FP: false positives, i.e. type I errors
- TP: true positives
- Accuracy = (TP + TP)/N
- Error rate = (FP + FN)/N
- Precision = TP / (TP + FP)
- Recall = Sensitivity = TPR = TP / (TP + FN)
- Specificity = TNR = TN / (TN + FP)
- ROC curve: plot of TPR and FPR over a range of cutoff values. (0,0) and (1,1) always a point on curve
- AUC = area under the ROC curve: estimate of model fit

Time Series

Time series can be decomposed a series into three types of patterns:

1. The trend is that part of a series that corresponds to a long-term, slow evolution of the series. This is the most important part for long-term forecasts.
2. The seasonal part of the series corresponds to aspects that repeat itself periodically, say, over a year.
3. The irregular patterns of a series are short-term movements that are typically harder to anticipate.

Forecast future values of the series by extrapolating each of the three patterns.

Autocorrelation Function:

- Summarizes the linear relationship between observations that are k time units apart: $r_k = \frac{\sum_{t=k+1}^{T}(y_{t-k}-\bar{y})(y_y=\bar{y})}{\sum_{t=k+1}^{T}(y_{t-k}-\bar{y})^2}$.
- Plotting the autocorrelation function as a function of k determines if the autocorrelation decreases as the lag gets larger, or if there is any particular lag for which the autocorrelation is large.

AR(1) Model

- For a (stationary) AR(1), the correlation between points k time units apart

$$\rho^k = \frac{Cov(y_t, y_{t-k})}{\sqrt{Var(y_t)Var(y_{t-k})}} = \frac{Cov(y_t, y_{t-k})}{\sigma_y^2} = \beta_1^k$$

- β_1 this is restricted to be between -1 and 1. By making this restriction, the AR(1) series $\{y_t\}$ is stationary.
  - If $\beta_1 = 1$, model is (non-stationary) random walk
  - If $\beta_1 = 0$, model reduces to a white noise process
- Under the hypothesis of no autocorrelation, a good approximation to the standard error of the lag k autocorrelation statistics is $se(r_k) = \frac{1}{\sqrt{T}}$.
- To estimate the variance of the error terms: $s^2 = \frac{1}{T-3}\sum_{t=2}^{T}(e_t - \overline{e})^2$
- The smoothed series (i.e. the values fitted under the model) for the AR(1) model is: $\hat{y}_t = b_0 + b_1 y_{t-1}$.
- The chain rule of forecasting: For an AR(1) model, the k-step ahead forecast is recursively determined by $\hat{y}_{T+k} = b_0 = b_1 \hat{y}_{T+k-1}$.

Model Comparison

- Mean error $ME = \frac{1}{T}\sum_{t=1}^{T}\hat{\epsilon}_t$
- Mean percentage error $MPE = \frac{100}{T}\sum_{t=1}^{T}\frac{\hat{\epsilon}_t}{y_t}$
- Mean square error $MSE = \frac{1}{T}\sum_{t=1}^{T}\hat{\epsilon}_t^2$
- Mean absolute error $MSE = \frac{1}{T}\sum_{t=1}^{T}|\hat{\epsilon}_t|$
- Mean absolute percentage error $MAPE = \frac{100}{T}\sum_{t=1}^{T}|\frac{\hat{\epsilon}_t}{y_t}|$

Random Walk

A random walk is the partial sum of a white noise process. Let $\epsilon_1, ..., \epsilon_T$ be T observations from a white noise process with mean μ and variance σ_ϵ^2. A random walk can be expressed recursively as

$$y_t = y_{t-1} + \epsilon_t = y_0 + \mu_c t + u_t$$

where $u_t = \sum_{j=1}^{t} \epsilon_j$.

The random walk is not a stationary process because the variability, and possibly the mean, depends on the time point t at which the series is observed.

- Variance $Var[y_t] = t\sigma_\epsilon^2$. Hence, as long as there is some variability in the white noise process, the random walk is nonstationary in the variance.
- Mean $E[y_t] = y_0 + t\mu_t$. If $\mu_\epsilon \neq 0$, then the random walk is nonstationary in the mean.

Unit Root

A unit root test is used to evaluate the fit of a random walk model. The Dickey-Fuller test is the t-statistic associated with the y_{t-1} variable using ordinary least squares on the following equation

$$y_t - y_{t-1} = \beta_0 + (\phi - 1)y_{t-1} + \beta_1 t + \sum_{j=1}^{P} \phi_j (y_{t-j} - y_{t-j-1} + \epsilon_t$$

- The (possibly serially correlated) disturbance term has been augmented by autoregressive terms in the differences $\{y_{t-j} - y_{t-j-1}\}$. These terms serve to capture serial correlation in the disturbance term, where results for a number of choices of lags P should be checked to reach qualitatively similar conclusions.
- The t-statistic, which does not follow the usual t-distribution (because $\{y_{t-1}\}$ is a random-walk process under the null hypothesis) but rather follows a special Dickey-Fuller distribution, tests $H_0 : \phi = 1$ versus the one-sided alternative that $H_a : \phi < 1$.

Smoothing

- Moving, or running, average estimates are defined by

$$\hat{s}_t = \frac{y_t + y_{t-1} + \ldots + y_{t-k+1}}{k}$$

, where k is the running average length.

- Exponential smoothing estimates are weighted averages of past values of a series, where the weights are given by a series that becomes exponentially small. Let w be a weight number that is between zero and one, and consider the weighted average:

$$\frac{y_t + wy_{t-1} + w^2 y_{t-2} + w^3 y_{t-3} + \ldots}{1/(1-w)}$$

Smoothing is only appropropriate for time series data without a linear trend

- A double smoothing procedure is used for time series with a linear trend.

Seasonality

Seasonal adjustment is the removal of seasonal patterns. Seasonal effects can be represented using binary or categorical variables, or trigonometric functions.

- A fixed seasonal effects model represents the seasonal component S_t as a function of time t. The two most important examples are the seasonal binary and trigonometric functions.
- A seasonal autoregressive model of order P represents the correlation between y_t and y_{t-k} using autoregressive models, with P lagged response explanatory variables:

$$y_t = \beta_0 + \beta_1 y_{t-k} + ... + \beta_P y_{t-Pk} + \epsilon_t$$

- The Holt-Winter additive seasonal model is a seasonal exponential smoothing method – with three smoothing parameters for the level, trend and seasonality respectively – that appears to work well in practice.

$$y_t = \beta_0 + \beta_1 t + S_t + \epsilon_t$$

where $S_t = S_{t-g}$ and $\sum_{t=1}^{g} S_t = 0$

Volatility Models

ARCH/GARCH models allow for changing variances in a time series by conditioning on the past.

- The autoregressive changing heteroscedasticity model of order p, or *ARCH (p)*, specifies that the conditional variance is determined recursively by

$$\sigma_t^2 = w + \gamma_1 \epsilon_{t-1}^2 + ... + \gamma_p \epsilon_{t-p}^2$$

where $w > 0$ is the "long-run" volatility parameter and $\gamma_1, ..., \gamma_p$ are coefficients such that $\gamma_j \geq 0$ and $\sum_{j=1}^{p} \gamma_j = 1$. The unconditional variance remains constant over time despite having a changing conditional variance,

- The generalized ARCH model of order p, or *GARCH(p,q)*, adds a moving average component, where the conditional variance is determined recursively by:

$$\sigma_t^2 - \delta_1 \sigma_{t-1}^2 - ... - \delta_q \sigma_{t-q}^2 = w + \gamma_1 \epsilon_{t-1}^2 + ... + \gamma_p \epsilon_{t-p}^2$$

where additionally $\delta_j \geq 0$ and $\sum_{j=1}^{p} \gamma_j + \sum_{j=1}^{q} \delta_j < 1$. The *GARCH(p,q)* is also a weakly stationary model, with mean zero and (unconditional) variance $Var(\epsilon_t) = \omega/(1 - \sum_{j=1}^{p} \gamma_j - \sum_{j=1}^{q} \delta_j)$
The marginal distribution (ϵ_t) has fatter tails than the Gaussian distribution, even though the conditional distribution (ϵ_t/σ_t) is Gaussian.

Base Decision Tree

- Split feature space into exhaustive and mutually exclusive segments
- that minimize variability of target with each segment (node purity)
- by splitting parent node by a variable that causes segments to be most different from each other
- A series of splitting rules that divides the dataset into nodes
- Splitting rules are determined using recursive binary splitting
- Number of terminal nodes is the flexibility measure
- No underlying assumptions about target and predictors

Impurity Function

- Entropy: $-\sum_{i \in c} p_i \log_2 p_i$
  - If all data from same class: entropy $= 0$;
  - if data are evenly split: entropy $= 1$
- Information Gain $=$ Entropy(parent) $-\sum_k \frac{N_k}{N_p}$ Entropy(Child$_k$)
  - N_p is number of observations in parent, N_k is number in kth child node
  - Natural bias that favors categorical features with many levels
- Gini: $1 - \sum_{i \in c} p_i^2$
  - Goal is to make Gini as small as possible
- Classification Error: $1 - max_{i \in c} p_i$
- Splitting and cost-complexity pruning

Pruning

Reduced error pruning:

- Consider making each node a leaf and assigning it the most common classification of training examples in its subtree
- Remove node only if resulting pruned tree performans no worse over the validation set
- Choose best level of complexity to prune back that minimizes the cross validation error
- "smallest error plus one standard deviation" rule

Advantages and Disadvantages

Decision Trees

- Intuitive and quick to run
- Good for recognizing natural breakpoint in continuous variable
- Good for nonlinear interactions between variables
- Unstable and prone to overfitting

Advantages:

- Easy to interpret and explain
- Easy to construct
- Mimic human decision-making
- Insignificant predictors are removed automatically
- Interactions are captured automatically
- Factors are handled automatically; binarization is not needed
- Handles missing values

Disadvantages

- Requires an abundance of observations to perform well
- High risk of overfitting
- Not robust: sensitive to training set
- Tends towards selecting categorical features with many levels
- Greedy algorithm
- Low prediction accuracy, is a "weak learner"

Interpretation

Feature Importance: method that analyzes structure of model and ranks contribution of each feature

Total RSS of Gini Index decreased when split on a predictor

Partial Dependence plots: visualize structure of the model's dependence on a feature or pair of features

- Average predicted value of target value by varying the value of one (or more) input features
- Can only visualize one feature at a time (and a maximum of two or three manually)
- Cannot explore high-level interaction
- Is just average model behavior and only a "qualitative" view

Ensemble Methods

Ensemble Methods

- Limited ability of GLM's to extract complex relationships
- Sensitivity to noise and tendency to overfit (decision trees)
- Bagging (bootstrap aggregation): train multiple models independently in parallel on random subsets of the data. When trees used as base model (and random subset of predictors) $\rightarrow$ random forest
  - In comparison to boosting, bagged models are less flexible in ability to reduce bias, and more tied to the capacity of their underlying models. Less prone to overfitting than gradient boosting machines
  - One can show that on average, each bagged tree makes use of around two-thirds of the observations. The remaining one-third of the observations not used to fit a given bagged tree are referred to as the out-of-bag (OOB) observations. We can predict the response for the ith observation using each of the trees in which that observation was OOB. The resulting OOB error is a valid estimate of the test error for the bagged model, since the response for each observation is predicted using only the trees that were not fit using that observation. With sufficiently large B, virtually equivalent to LOOCV
  - majority vote for prediction
- Boosting: train sequence of models on residuals obtained from predicting previously. Each model is obtained by adding a scaled down version of new model until sum of all component models predicts entire data set well.

Random Forests

Random Forests

- Random forests provide an improvement over bagged trees by way of a random forest small tweak that decorrelates the trees. As in bagging, we build a number of decision trees on bootstrapped training samples. But when building these decision trees, each time a split in a tree is considered, a random sample of m predictors is chosen as split candidates from the full set of p predictors.
- Natural extension of decision trees, using many weak learners
- "Bagging" to use subset of features to build each decision tree – reduce model variance
- Retain many properties of decision trees while removing large variance
- Higher potential for predictive accuracy by simulating many different starting points
- But much less interpretable than an individual decision tree
- The main difference between bagging and random forests is the choice of predictor subset size m.
- Force each split to consider only a subset of the predictors. Therefore, on average $(p - m)/p$ of the splits will not even consider the strong predictor, and so other predictors will have more of a chance. We can think of this process as decorrelating the trees.

Training and Prediction

Training:

- Bootstrap random sample of observations with replacement
- At each split, search from random sample of features
- Train a decision tree with above
- Repeat to train many decision trees

Predicting

- Predict target using each tree previously trained
- Average the predictions (may be accuracy or optimally weighted)

Parameters

- Number of trees (as large as possible)
- Proportion of observations in each random sample to build each tree (typically 60%)
- Proportion of features used at each split (mtry is only parameter supported by caret)
- Decision tree parameters: depth, min bucket, min size, method, improvement, complexity (as high as possible because variance is reduced through bagging)

Boosting

- Like bagging, boosting is a general approach that can be applied to many statistical learning methods for regression or classification
- Sequential learning: build multiple models one at a time, and at each step adjust the training data to place more emphasis on the data points that previous models predicted poorly: models are not independently trained (unlike bagging)

$$F_N = \sum_i^N alpha_i f_i() = F_{N-1} + alpha_N f_N()$$

- often uses regularization, such as shrinkage, in the processing of adding functions to the current models, to prevent overfitting
- Loss function:
  - Least squares, logistic, poisson, gamma, tweedie for regression
  - Logistic for binary, softmax for multiclass classification
  - Pairwise loss for ranking tasks
- boosting to continuously refit residuals – reduce model bias
- Boosting does not involve bootstrap sampling, but grows trees sequentially on residuals. Each tree is fit on a modified version of the original data set

Boosting

- More prone to overfitting than random forests, and sensitive to hyperparameter inputs
- Learns slowly: shrinkage parameters lambda typical 0.01 or 0.002
- Small trees: interaction depth d, often decision stump
- Number of trees B: overfitting slow to occur if at all
- Aggregate predictions from multiple trees that are fitted consecutively using updated datasets
- Individual trees have high bias, low variance
- Because the growth of a particular tree takes into account the other trees that have already been grown, smaller trees are typically sufficient. Using smaller trees can aid in interpretability as well; for instance, using stumps leads to an additive model.
- Number of trees is the flexibility measure
- Use variable importance and partial dependence pilots to visualize the effect of predictors
- Gradient Boosting Machines (GBM): approximate solution by fitting each new model in the direction of the negative gradient of the loss function (in squared loss for regression, derivative is given by residuals, so this is not new), adding new model multiplied by a learning or shrinkage parameter – takes smaller steps and longer to converge but can reduce overfitting.

Pros and Cons

Decision Tree: Pros

- Easy to interpret, if not too many splits.
- Handles nonlinear and interaction relationships.
- Categorical predictors automatically handled without binarization.
- Variables are automatically selected: variables that do not appear are filtered out; most important appear at top of tree.

Cons

- More prone to overfitting, even with pruning.
- Strongly dependent on training data: predictions unstable with high variance.

Ensembles: Pros

- More robust and predictive than base tree by combining the results of multiple trees.

Cons

- Opaque/difficult to interpret: many tree are used, requiring variable importance measures or partial dependency plots.
- Difficult to implement: large computational burden to fit multiple base trees.

Pros and Cons

GLM: Pros

- Accommodates wide variety of distributions for target variable
- Model shows how target mean depends on features.
- Coefficients provide interpretable measure of directional effect.
- Simple to implement.

Cons

- Does not capture non-linear or interactive relationships (unless engineered in advance)
- For some link functions, coefficients may be difficult to interpret.

Regularized GLM: Pros

- Categorical variables are binarized automatically and each factor level is treated as a separate feature to be removed
- For lasso, variable selection is automatically performed

Cons

- Coefficient estimates are more difficult to interpret because variables are standardized.
- Limited/restricted model forms allowed by glmnet

Terminology

1. Predictive analytics focuses on the future: What might happen next?
   - Identify customers who have highest probability of purchasing additional products
2. Prescriptive analytics suggests decision options: What would happen if I do this, what is the best course of action?
   - Identify customers for a marketing campaign.
3. Descriptive analytics focuses on insights from the past: What happened?
   - Understand current customers

Common characteristics:

- Clearly identify and define the business issues that needs to be addressed
- Can address the issue with a few well-defined questions
- Have lots of good and useful data that can be used
- Reasonably certain the predictions will drive actions or increase understanding
- Reasonably determine it is better than any existing approach
- Can continue to monitor and update the models as new data comes in

Problem Definition Framework

1. Define: articulate the problem definition.
   - Clarity of business issues, hypotheses to be tested,
2. Impact on the business issue, define objectively without placing blame.
   - Key to achieving clarity is to ask questions
3. Evaluate: examine feasibility of proposed solutions.
   - Have everything you need – data, resources, management buy-in.
   - If have solution, would it work – implementation constraints, regulatory, legal and consumer acceptance potential for gaming, effort and cost requirements reasonable
   - Do you have data required, is data of sufficient quality, will you have continuing access, are there any other risks to consider (systems, permissions)
   - Trade off ease of implementation and use for predictive accuracy – if you cannot implement and use the model, then it is useless (e.g. neural networks are universal approximators but take a long time and need lots of data).
4. Prioritize: what is business impact, what are risks of project
5. Prepare: get additional data, internal communications, what information stakeholders need, any external subject matter expertise (SME)

Sample project statement:
"Determine factors that relate to consumer decision to buy term life insurance and for those who do, the amount purchased. Selected factors should be able to be used by Marketing to better target their sales efforts."

Data Preparation

Data Cleaning

- Dealing with missing values
- Dealing with incorrect data types on import (e.g. dates)
- Transforming variables to allow models to pick up signal more effectively e.g. PCA, reduce number of categories/levels, additional variables/flags to highlight rows or fields
- Create multiple datasets for use in model fitting and testing

Intended use:

- target vs predictor variables

Feature Generation and Selection

- Derivations from the original variables/raw data, i.e. final inputs into the model, e.g.
  - Sentiment
  - Presence of certain words or phrases
  - Polynomial, log, exp, sin
- Feature generation and selection to
  - Transform data to build simpler model (requires human input):
  - Occam's razor
  - Help create more interpretable model
  - Improve accuracy without cost of increasing complexity of the model
- Binarization – turn categorical variable into several binary variables
- Combining variables
- PCA or cluster center

Data Issues

Data Quality

- Are the data reasonable (key statistics, reconcile with independent source)?
- Are quantities measured consistently?
- Is the format of the data consistent across data sets
- Are the data sufficiently documented (data dictionary?)
- Are the data representative of the modeled population?
- Understand the source of the data

Professional issues

- ASOP 23 Data Quality
- PII Personally Identifiable Information
- Regulations: HIPAA, FTC Act, GLB Act
- Unfair Discrimination – disparate impact on protected class

Missing Data

- Remove columns or rows with missing values – throws away valuable data points
- Replace with mean, median or mode (assumes missing at random)
- Assign category/value (if systematically missing for some reason)
- Predict using other variables (imputation)

Data Design

- Historical data should be reflective of what will happen in the future
- Time Frame: black swan events, changing cultures/social behaviors, new developments
- Sample: random, stratified (oversampling, undersampling)
  - Size
  - Model testing
  - Imbalanced Data
- Granularity

Unbalanced Data

- Undersampling: undersample the dominant class. But using less data, and could lead to curse of dimensionality issues
- Oversampling: oversample with replacement, but may run into memory issues
- Combination: hybrid techniques such as SMOTE and ROSE
- Performing sampling after the data have been split into trading and holdout to prevent leaking

Visualization

- `ggplot2` "grammar of graphics"
- *tidy* data "long format"
- cluttered, confusing, inaccurate, misleading, deceptive, dishonest, ugly?

Data Exploration

- Natural vs erroneous outliers
  - Remove: if not important
  - Ignore: if small or insignificant proportion
  - Modify: censoring
  - Use robust model forms: L1 norm
- Univariate Outliers
  - IQ range
  - Histogram
  - Boxplot
  - logscale
- Bivariate Outliers - Scatter plots: Numeric vs Numeric
  - box plots: Categorical vs Numeric
  - frequency table: Categorical vs Categorical
- Relationships - Pairs
  - Nonlinear relationships: Log transform
- Transformations
  - Binarize categorical variables
  - Grouping and combining into factors; New features: ifelse, ratios

Univariate Exploration

- Understand basic Relationships
- Check relationships against common knowledge and intuition
- Identify outliers and potential effects
- Variable Distributions
  - Numeric summaries (mean, variance, counts, summary)
  - Visualize values (histogram, bar plot)
  - Boxplot: visual summary of statistics and outliers
  - Frequencies: bar chart, table
- Inform variable transformation and modeling choices
- Log transform to remedy right skewness of positive numeric variables
- Combine levels of factor variables
  - Low counts increase variance of some levels.
  - High dimensions reduce predictive power
  - Ensure each level has sufficient observations...
  - ...while preserving mean differences across different levels for prediction

Bivariate Exploration

- Split Box Plot: numeric-categorical
- Split Histograms
- Stacked Bar Charts: categorical-categorical
- Scatter Plots: numeric-numeric
- Identify potentially useful predictors
- Identify factor levels that could be combined
- Interactions: effect of one variable on target variables depends on the value or level of another variable

Predictors	Numeric target	Categorical target
numeric-categorical	scatterplot, colored by categorical predictor	box plot, split by target, faceted by categorical predictor
categorical-categorical	box plot for target, split and faceted by predictors	bar chart for one predictor, filled by target, faceted by the other predictor
numeric-numeric	decision tree	decision tree

R and RStudio

RStudio shortcuts:

Alt -	$<-$
Ctrl Shift C	(un)comment multiple lines
Ctrl Shift M	%>%
Ctrl + Alt + I	insert chunk
Ctrl Shift Enter	run chunk
Ctrl Enter	run line

R

- operations: getwd(), setwd(), do.call(), lapply(), sapply(), rbind(), cbind(), function(), %in%, !, :, rm(list = ls())
- values: length(), summary(), class(), str(), NA, NULL, is.na(), levels(), table(), complete.cases()
- math: sample(size, replace, prob), all(), solve(), rowMeans(), optim()
- stats: sd(), cor.test(), runif(), rnorm(), quantile(), var(), diag(), which.min(), which.max()
- transforms: log(), jitter(), factor(), ifelse(), cut(label), paste(sep, collapse), as.factor(), as.character(), scale()
- subsets: setdiff(), split(), seq(from, to, by, length.out), sort(), subset(subset, select), rep()
- data.frame: colnames(), nrow(), read.csv(stringsAsFactors), expand.grid(), model.frame(), write.csv()
- plotting: plot(), points(), abline(), qplot(), segments(), grid.arrange() gridExtra: grid.arrange()

Making ggplots

- ggplot(data, mapping),
- geom_point(alpha, size),
- geom_count(stat='sum')
- geom_col()
- geom_bar(position=['dodge', 'stack', 'identity'], stat),
- geom_histogram(bins, binwidth, position='dodge'),
- geom_freqpoly(),
- geom_density(),
- geom_smooth(method, se),
- geom_line(), geom_hline(), geom_vline(), geom_abline()
- geom_segment(), geom_curve()
- aes(x, y, color, weight, fill),
- labs(title, subtitle, caption, x, y, color),
- annotate(x, xend, y, yend, size)
- coord_cartesian(xlim, ylim),
- theme()
- xlim(), ylim(), scale_x_log10(limits), scale_y_log10(),
- facet_wrap(), facet_grid()

Models

- linear model: lm(), coef(), residuals(), as.formula(offset), model.matrix(), anova(), confint()
- generalized linear model: glm(family, weights,offset), aic(), predict(newdata, type='response'), offset()
- penalized: as.matrix, model.matrix, glmnet(family, alpha, lambda), predict(newx), stepAIC(direction), cv.glmnet(nfolds), drop1()
- model assessment: pROC::auc(), pdp:partial()
- trees: rpart(method, control, parms), rpart.plot(), rpart.control(), printcp(), plotcp(), prune(cp)
- random forests: randomForest(ntree, mtry, sampsize, nodesize, importance)
- xgboost: xgb.DMatrix(), xgb.cv(), xgb.train(), xgb.importance()
- PCA: prcomp(center, scale), biplot(), screeplot()
- Clusterings: kmeans(centers, iter.max, nstart, algorithm="Lloyd"), dist(), hclust(), cuttree(),

Caret Cross-validation

- createDataPartition(p, list),
- dummyVars(), as.character()
- trainControl(method, number, repeats),
- expand.grid()
- train(data, method, trControl, metric, tuneGrid, na.action, parms),
- confusionMatrix(data, reference)
- varImp()

Data Preparation

Bivariate Analysis:

```
data.all.task4$pure = ifelse(data.all.task4$mix, "mix", "pure")
p <- ggplot(data=data.all.task4, aes(x=in.month, fill=pure))
p + geom_bar() + facet_grid(cols=vars(in.year), rows=vars(animal)) +
  theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())
```

Binarize:

```
library(caret)
# List the variables we want to binarize
vars.bin <- c("sex", "smoker", "prodcat", "region", "distchan", "uwkey", "uwtype", "resind_ind")
# dummyVars is not compatible with factors
for (var in vars.bin) {
  data.mortality[, var] <- as.character(data.mortality[, var])
}
# Binarize variables
# The paste function makes it easy to create the formula, it could have been typed out.
# fullRank = F implies that all values get coded. For example, for sex, both M and F will be in the binari
binarizer <- caret::dummyVars(paste("~", paste(vars.bin, collapse = "+")), data = data.mortality, fullRank
data.mortality <- cbind(
  data.mortality,
  data.frame(predict(binarizer, newdata = data.mortality))
)
```

Decision Tree with Cross-Validation

```
ibrary(rpart)
library(caret)
library(rpart.plot)
set.seed(10)
fitControl <- trainControl(method = "cv", number = 6)

Grid <- expand.grid(cp = seq(0, 0.1, 0.001))

credit.f <- as.formula(Credit ~ CreditAmount + Age + CreditHistory + Employment)


credit.m <- train(credit.f,
  data = credit,
  method = "rpart",
  trControl = fitControl,
  metric = "Accuracy",
  tuneGrid = Grid,
  na.action = na.omit,
  parms = list(split = "information")
)

credit.m$finalModel # Best model can be accessed with the caret_model_object$finalModel call.

plot(credit.m)
rpart.plot(credit.m$finalModel, extra = 4)

pred_caret <- predict(credit.m, type = "raw")

confusionMatrix(pred_caret, factor(credit$Credit)) # Arrive at confusion matrix with caret.
```

Random Forest

```
library(randomForest)
credit$Credit <- as.factor(credit$Credit)
credit.rf <- randomForest(
  formula = Credit ~ CreditHistory + Purpose + Employment + Duration + Employment + Age,
  data = credit,
  ntree = 500,
  mtry = 4, # The number of features to use in each split
  importance = TRUE
)
credit.rf
```

With cross-validation and imbalanced data:

```
rfGrid <- expand.grid(mtry = c(1, 3, 5, 7)) # The number of features to select at each split.
ctrl <- trainControl(
  method = "repeatedcv",
  number = 5,
  repeats = 3, # We want to do 5-fold cross validation (repeated 3 times for robustness)
  sampling = "down"
) # This is undersampling - other methods include "up" (oversampling), "SMOTE" and "ROSE" (hybrid methods)
model.rf.tuned <- train(target ~ .,
  data = data.training,
  method = "rf", # This is so we use the randomForest algorithm.
  trControl = ctrl,
  tuneGrid = rfGrid,
  # We can specify the other parameters for the randomForest model here if we wish to.
  # If we don't they will take on their default values.
  ntree = 50, # The default is 500. save us a lot of computation time but may not produce the best results
  importance = TRUE
)
model.rf.tuned
ggplot(model.rf.tuned)
```

Boosting with Imbalanced Data

```
xgbGrid <- expand.grid(
  max_depth = c(1:7),
  nrounds = 500,
  eta = c(.01, .05, .01),
  colsample_bytree = c(.5, .8),
  gamma = 0,
  min_child_weight = 1,
  subsample = .6
)

ctrl <- trainControl(
  method = "cv", number = 4,
  classProbs = TRUE,
  sampling = c("down", "up")
) # Sample unbalanced sample up and down

xgb.tuned <- train(Purchase ~ .,
  data = data.Caravan.train,
  method = "xgbTree",
  metric = "Accuracy",
  trControl = ctrl,
  tuneGrid = xgbGrid,
  na.action = na.pass
)

xgb.tuned
ggplot(xgb.tuned)

varImp(xgb.tuned) # Get variable importance.
```

Extra Slides

An extra slide

Famous Datasets

- Gapminder
- AnscombesQuartet
- SOA Mortality Data Set

Quotes:

- The combination of some data and an aching desire for answer does not sure that a reasonable answer can be extracted from a given body of data – John Tukey
- If I had one hour to save the world, I would spend 55 minutes defining the problem and 5 minutes designing the solution – Albert Einstein

Back