The Business School for the World®

DS(ML)B: Data Science (& Machine Learning) for Business

Profs. Anton Ovchinnikov, Theos Evgeniou, Spyros Zoumpoulis

Sessions 05-06

- Introduction to Classification

Plan for the day – Learning objectives

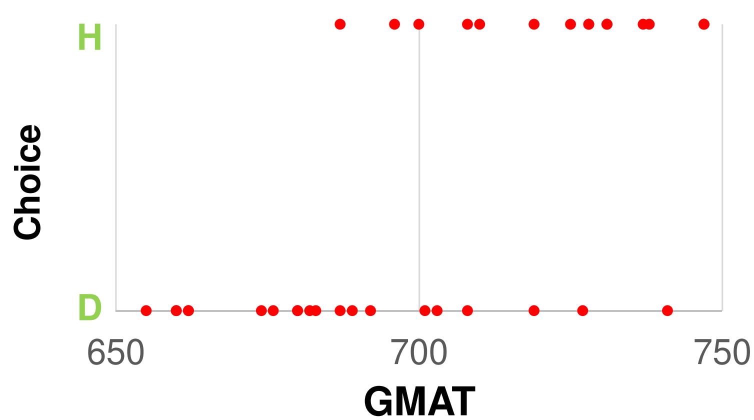
- Conceptual introduction to classification: what/why/metrics
- Data science methodologies for classification:
  - Stats: logistic regression (generalized linear model, `glm`) + `stepAIC` variable selection
  - Machine Learning:
    - « mother of all methods », CART (classification and regression tree)
    - « derivartives » of CART: `randomforest`, gradient boosting machines (`xgboost`)
    - in Session 0708: additional methods: regularizations (`LASSO, Ridge`), support vector machines (`SVM`)
- Application:
  - Customer Churn case [perhaps the most common business application of DS&ML]
  - Scholastic Travel Company (STC) case: part (A) today, continue with (B) in Tutorial 2

What is classification, and why do we need it?

- In sessions 1-4 we considered a task of predicting a quantity (price of a diamond, electricity rate, number of website users)
- But an equally* common task is to predict an outcome of an event + understand actionable drivers
- Binary outcomes ("events"):
  - Will a customer churn? Will a customer default on a loan?
  - Will an employee/student accept a job/school offer
- Multi-nomial outcomes:
  - Will a person walk/drive/bike/take a public transit?
  - Will a customer buy iPhone X/8/8+/nothing?

Predicting events: what if “Y” is categorical?

- Examples of categorical dependent variable? Pre-class reading?
- Customer choice:
  - Business school "D" versus "H" as a function of GMAT score



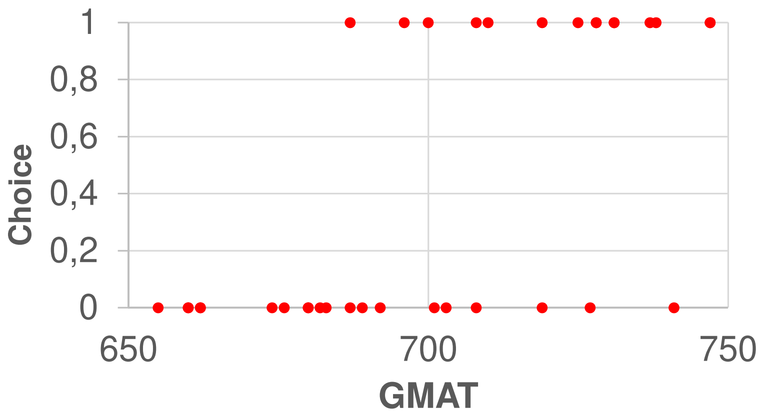
If we know GMAT, can we predict choice?

	A	B	C
1	ID	GMAT	Choice
2	1	655	D
3	2	660	D
4	3	660	D
5	4	662	D
6	5	662	D
7	6	674	D
8	7	676	D
9	8	680	D
10	9	680	D
11	10	682	D
12	11	683	D
13	12	687	H
14	13	687	D
15	14	689	D
16	15	692	D
17	16	696	H
18	17	700	H
19	18	701	D
20	19	703	D
21	20	708	H
22	21	708	D

Source: 0506 Data and model from the Modeling Discrete Choice reading.xls

Predicting choice: Regression?

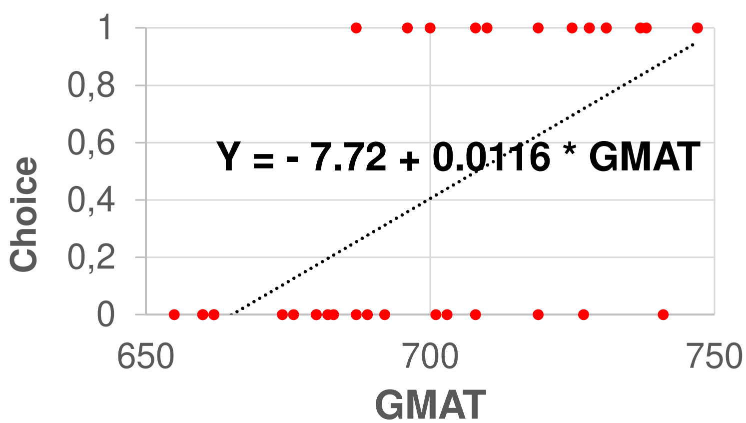
- Step one: Transform D/H into a dummy variable (0,1)



- Step two: Run a (linear) regression

Predicting choice: Regression?

- Step one: Transform D/H into a dummy variable (0,1)



Multiple Regı Summary	Multiple R	R-Square	Adjusted R-Square	StErr of Estimate		
	0.6385	0.4076	0.3897	0.392249		
ANOVA Table	Degrees of Freedom	Sum of Squares	Mean of Squares	F-Ratio	p-Value	
Explained	1	3.494068	3.494068	22.7095	< 0.0001	
Unexplained	33	5.07736	0.153859			
Regression Ta	Coefficient	Standard Error	t-Value	p-Value	Confidence Interval 95% Lower	Upper
Constant	-7.72912	1.713126	-4.5117	< 0.0001	-11.2145	-4.24374
GMAT	0.011619	0.002438	4.7654	< 0.0001	0.006659	0.01658

- Step two: Run a (linear) regression
  - How should we interpret the Y variable?
    - E.g. for GMAT=700, Y = 0.4... [of what?]
    - What about GMAT = 650?

Predicting **Probability** of Choice: Logistic Regression

- It is natural to interpret the "Y" variable in the preceding example as a probability of choice
  - Hence we are predicting the probability of choice, not the choice itself

- But linear model is not suitable to predict probabilities (e.g., because it cannot guarantee probability >0 or <1)

- Logit model (hence "logistic" regression) is one such model
  - The term "logit" refers to the Log of odds prob/(1-prob).
  - Logit is not the only model used to model choice; another popular model is called "probit"

$$Prob(A\ is\ chosen\ from\ set\ of\ S\ alternatives) = \frac{\exp(utility\ of\ A)}{\sum \exp(utilties\ of\ all\ alternatives)}$$

$"utility\ of\ A"$ = linear function of various variables

Logistic/Logit Model

- General form:

$$Prob(A\ is\ chosen\ from\ set\ of\ S\ alternatives) = \frac{\exp(utility\ of\ A)}{\sum \exp(utilties\ of\ all\ alternatives)}$$

- Here "all" might also include an alternative to do/buy nothing
- With only two alternatives:

$$Prob(A\ is\ chosen\ over\ B) = \frac{\exp(utility\ of\ A)}{\exp(utility\ of\ A) + \exp(utility\ of\ B)}$$

- Further, since only relative utility matters, we can [arbitrarily] normalize $utility\ of$ B=0, and then noting that $exp(0) = 1$

$$Prob(A\ is\ chosen\ over\ B) = \frac{\exp(utility\ of\ A)}{1 + \exp(utility\ of\ A)}$$

Back to Our Example: School H Versus D

Let $utility\ of\ D = 0$ [arbitrarily]

Let $utility\ of\ H\ = a + b * GMAT$

We can then express the probabilities of choices as a function of a and b

…and estimate utility coefficients a and b – “fit” the logistic regression model

`=$G$2+$G$1*B`

`=F4/(1+F4)`

	A	B	C	D	E	F	G	J
1		uH=a + b*GMAT				a=	-48	
2						b=	0.07	
3	ID	GMAT	Choice	Dummy	uH	EXP(uH)	Prob(H is chosen)	
4	1	655	D	0	-2.1500	0.1165	0.1043	
5	2	660	D	0	-1.8000	0.1653	0.1419	
6	3	660	D	0	-1.8000	0.1653	0.1419	
7	4	662	D	0	-1.6600	0.1901	0.1598	
8	5	662	D	0	-1.6600	0.1901	0.1598	
9	6	674	D	0	-0.8200	0.4404	0.3058	
10	7	676	D	0	-0.6800	0.5066	0.3363	
11	8	680	D	0	-0.4000	0.6703	0.4013	
12	9	680	D	0	-0.4000	0.6703	0.4013	
13	10	682	D	0	-0.2600	0.7711	0.4354	
14	11	683	D	0	-0.1900	0.8270	0.4526	
15	12	687	H	1	0.0900	1.0942	0.5225	
16	13	687	D	0	0.0900	1.0942	0.5225	
17	14	689	D	0	0.2300	1.2586	0.5572	
18	15	692	D	0	0.4400	1.5527	0.6083	
19	16	696	H	1	0.7200	2.0544	0.6726	
20	17	700	H	1	1.0000	2.7183	0.7311	
21	18	701	D	0	1.0700	2.9154	0.7446	
22	19	703	D	0	1.2100	3.3535	0.7703	
23	20	708	H	1	1.5600	4.7588	0.8264	

Estimating Utility Coefficients: Maximum Likelihood Estimation (MLE)

- For ID1:
  - Actual choice: D
  - Predicted probability of choosing H is 0.1043
  - Hence the likelihood that ID1 indeed chooses D in our model is 1-0.1043=0.8957
- For ID2: Choice is D, predicted prob(H)=0.1419, hence the likelihood is 0.8581
- The likelihood of ID1 choosing D and ID2 choosing D is 0.8957*0.8581, etc…
- We would like to select model coefficients **a** and **b** to maximize likelihood (Maximum Likelihood Estimation, MLE)
- Note:
  - With many data points such product will be very small - inconvenient for optimization
  - However, Log (X*Y*Z)=Log(X)+Log(Y)+Log(Z)
- Hence instead of maximizing likelihood, we maximize log-likelihood (LL)

	A	B	C	D	G	H	I	J
1		uH=a + b*GMAT			-48			
2					0.07			
3	ID	GMAT	Choice	Dummy	Prob(H is chosen)	Likelihood	Log(Likelihood)	
4	1	655	D	0	0.1043	0.8957	-0.1102	
5	2	660	D	0	0.1419	0.8581	-0.1530	
6	3	660	D	0	0.1419	0.8581	-0.1530	
7	4	662	D	0	0.1598	0.8402	-0.1741	
8	5	662	D	0	0.1598	0.8402	-0.1741	
9	6	674	D	0	0.3058	0.6942	-0.3649	
10	7	676	D	0	0.3363	0.6637	-0.4099	
11	8	680	D	0	0.4013	0.5987	-0.5130	
12	9	680	D	0	0.4013	0.5987	-0.5130	
13	10	682	D	0	0.4354	0.5646	-0.5716	
14	11	683	D	0	0.4526	0.5474	-0.6027	
15	12	687	H	1	0.5225	0.5225	-0.6492	
16	13	687	D	0	0.5225	0.4775	-0.7392	
17	14	689	D	0	0.5572	0.4428	-0.8147	
18	15	692	D	0	0.6083	0.3917	-0.9372	
19	16	696	H	1	0.6726	0.6726	-0.3966	
20	17	700	H	1	0.7311	0.7311	-0.3133	
21	18	701	D	0	0.7446	0.2554	-1.3649	
22	19	703	D	0	0.7703	0.2297	-1.4710	
23	20	708	H	1	0.8264	0.8264	-0.1907	

=F4/(1+F4)

=IF(D4=1,G4,1-G4)

=LN(H4)

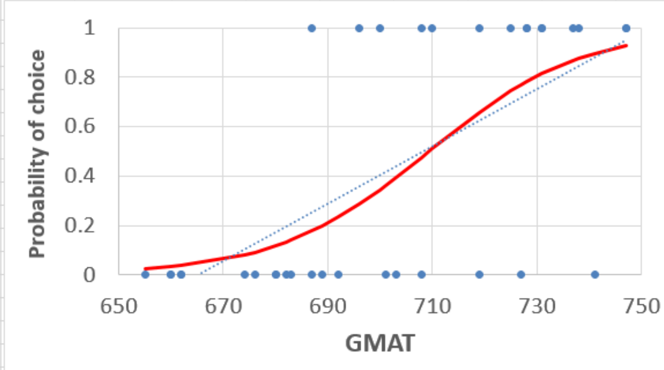
Results:
School H Versus D example

uH=a + b*GMAT

a=	-48.47108126
b=	0.068326197

=\$G\$2+\$G\$1*B   =IF(D4=1,G4,1-G4)   =LN(H4)

ID	GMAT	Choice	Dummy	uH	EXP(uH)	Prob(H is chosen)	Likelihood	Log(Likelihood)
1	655	D	0	-3.7174	0.0243	0.0237	0.9763	-0.0240
2	660	D	0	-3.3758	0.0342	0.0331	0.9669	-0.0336
3	660	D	0	-3.3758	0.0342	0.0331	0.9669	-0.0336
4	662	D	0	-3.2391	0.0392	0.0377	0.9623	-0.0384
5	662	D	0	-3.2391	0.0392	0.0377	0.9623	-0.0384
6	674	D	0	-2.4192	0.0890	0.0817	0.9183	-0.0853
7	676	D	0	-2.2826	0.1020	0.0926	0.9074	-0.0971
8	680	D	0	-2.0093	0.1341	0.1182	0.8818	-0.1258
9	680	D	0	-2.0093	0.1341	0.1182	0.8818	-0.1258
10	682	D	0	-1.8726	0.1537	0.1332	0.8668	-0.1430
11	683	D	0	-1.8043	0.1646	0.1413	0.8587	-0.1524
12	687	H	1	-1.5310	0.2163	0.1778	0.1778	-1.7268
13	687	D	0	-1.5310	0.2163	0.1778	0.8222	-0.1958
14	689	D	0	-1.3943	0.2480	0.1987	0.8013	-0.2215



Logistic Regression for Dummy

Summary Measures	
Null Deviance	47.80356733
Model Deviance	30.96154543
Improvement	16.8420219
p-Value	< 0.0001

Regression Coefficients	Coefficient	Standard Error	Wald Value	p-Value
Constant	-48.47037424	15.38526195	-3.150441923	0.0016
GMAT	0.068325198	0.021740921	3.142700311	0.0017

Classification Matrix	1	0	Percent Correct
1	11	4	73.33%
0	3	17	85.00%

$$Prob(H\ is\ chosen|GMAT) = \frac{\exp(utility\ of\ H)}{1+\exp(utility\ of\ H)} = \frac{\exp(-48{,}47 + 0{,}0683 * GMAT)}{1+\exp(-48{,}47 + 0{,}0683 * GMAT)}$$

With GMAT=700:

- Utility of H = -48,47+0,0683*700 = -0,66 [why is it negative?]
- Prob of H = exp(-0.66)/(1+exp(-0.66)) = 0,5168/1,5168 =0,34

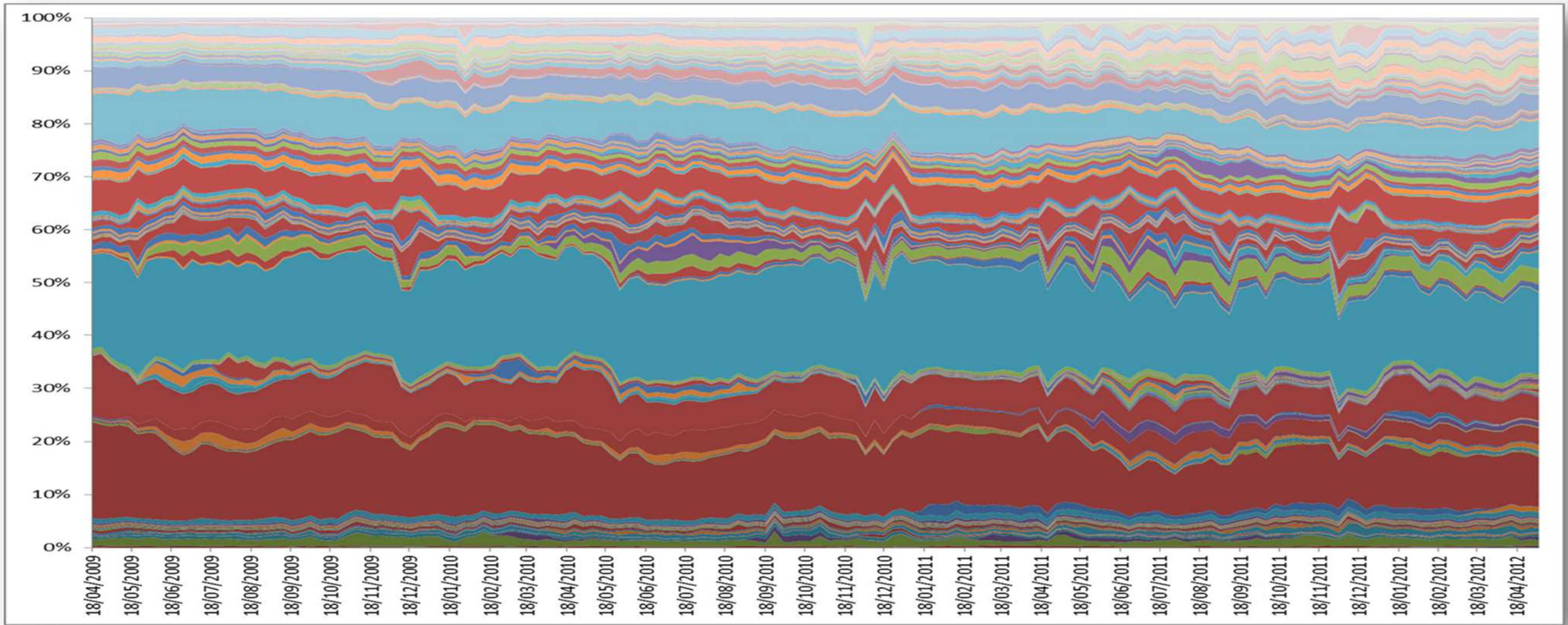
Logistic Regression in R: School H vs D example

```
ChoiceData<-read.csv(file.choose()) #load data
str(ChoiceData) #make sure that the field types are interpreted correctly (as
numbers/integers, factors, etc.)
Logistic_Model<-glm(Choice ~ GMAT, data = ChoiceData, family="binomial"(link="logit"))
#logistic regression is part of the "generalized linear models" family, hence glm
summary(Logistic_Model) #summary of the model
par(mfrow=c(1,4)) # This command sets the plot window to show 1 row of 4 plots
plot(Logistic_Model) # check the model using diagnostic plots
predict(Logistic_Model, newdata=data.frame("GMAT"=700),type="response") #predict the
probability of choice as a function of GMAT; type="link" will predict the utility
```

```
Call:
glm(formula = Choice ~ GMAT, family = binomial(link = "logit"),
    data = ChoiceData)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1298  -0.5889  -0.2593   0.6726   1.8584

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -48.47108   15.38544  -3.150  0.00163 **
GMAT          0.06833    0.02174   3.143  0.00167 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> predict(Logistic_Model, newdata=data.frame("GMAT"=700),type="response")
#predict the probability of choice as a function of GMAT
        1
0.3446266
```

Logit models are rather accurate

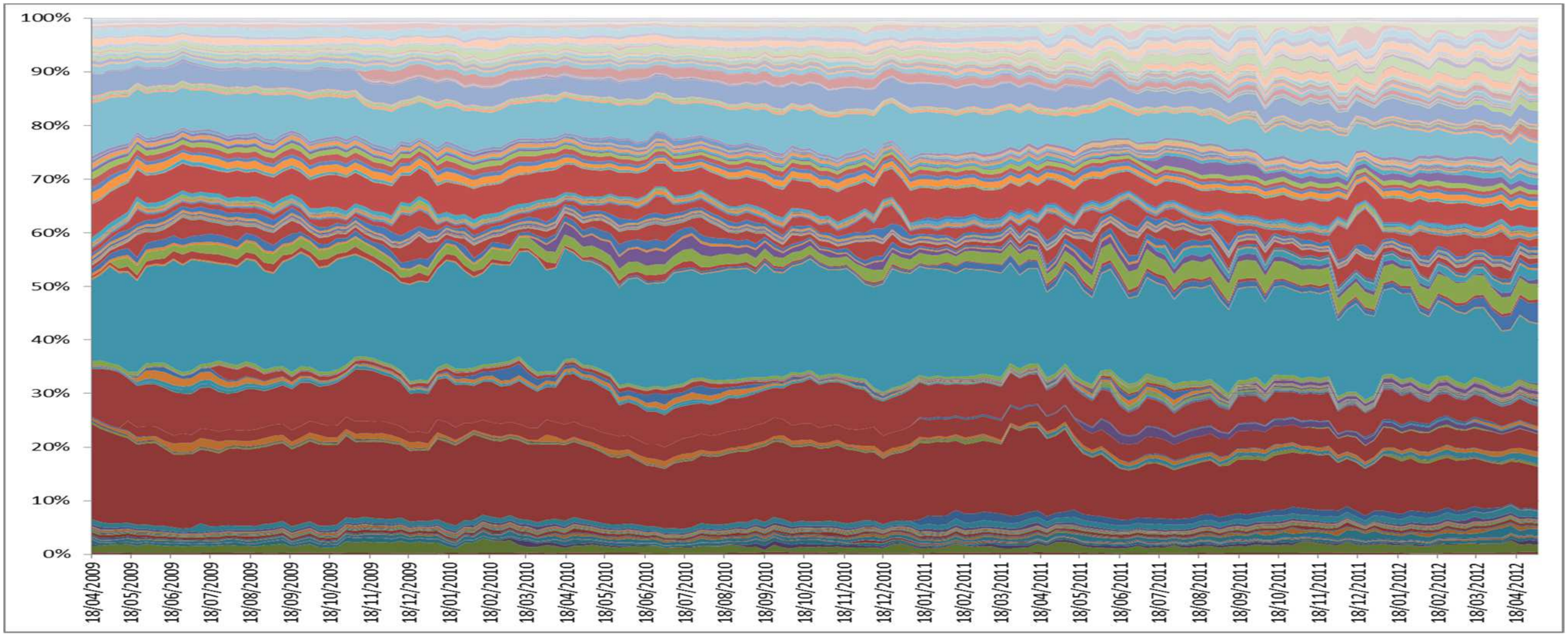
Nested MNL, beer sales, 35m observations, <500 variables [prices, promos…]



actual

Logit models are rather accurate

Nested MNL, beer sales, 35m observations, <500 variables [prices, promos…]



predicted

Back to classification: threshold and metrics

- For models with continuous quantities we discussed multiple metrics: r^2, MAPE, (R)MSE

- For classification models we need other metrics, that account for the fact that the predicted object is an event:
  1. Confusion matrix and its measures
  2. ROC ("receiver operating characteristic") curve
  3. AUC ("area under curve) and Gini coefficient
  4. Lift chart / Gains chart

- All metrics rely on Threshold rule: **IF(Prob > T, then "Yes", otherwise "No")** that converts from probability to events/classification

Confusion Matrix
[customer retention example]

	Actual Retained	Actual Not Retained
Predicted Retained	a (TP)	b (FP)
Predicted Not Retained	c (FN)	d (TN)

TP stands for True Positive

FN stands for False Negative, etc.

https://en.wikipedia.org/wiki/Sensitivity_and_specificity

Confusion Matrix
[customer retention example]

	Actual Retained	Actual Not Retained		
Predicted Retained	a (TP)	b (FP)	Positive Predictive Value	a/(a+b)
Predicted Not Retained	c (FN)	d (TN)	Negative Predictive Value	d/(c+d)
	Sensitivity [TPR] a/(a+c)	Specificity [TNR] d/(b+d)		

https://en.wikipedia.org/wiki/Sensitivity_and_specificity

Confusion Matrix
[customer retention example]

	Actual Retained	Actual Not Retained		
Predicted Retained	a (TP)	b (FP)	Positive Predictive Value	a/(a+b)
Predicted Not Retained	c (FN)	d (TN)	Negative Predictive Value	d/(c+d)
	Sensitivity [TPR]	Specificity [TNR]		

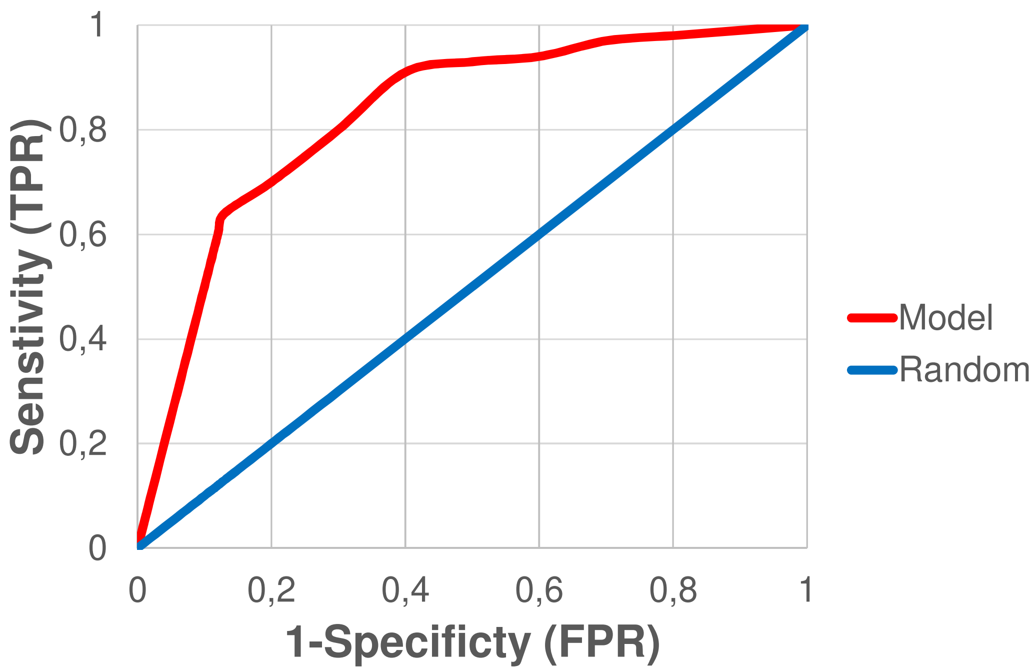
Overall measure:
Accuracy=(a+d)/(a+b+c+d)

Misclassification error = 1- accuracy

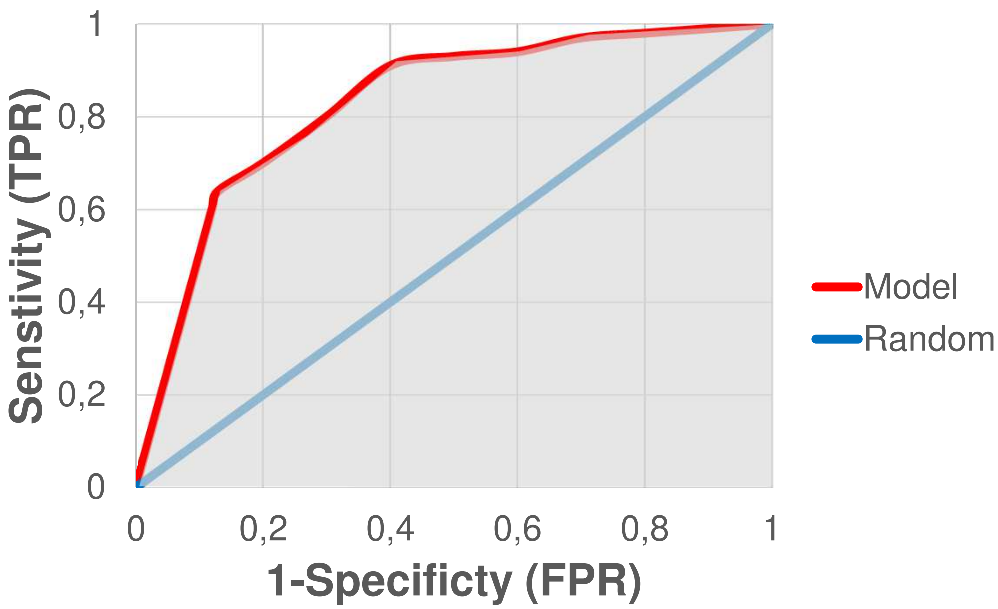
https://en.wikipedia.org/wiki/Sensitivity_and_specificity

ROC Curve:
Where are errors made?

- Varying threshold T and plotting the [sensitivity, 1-specificity] points leads to a curve called “ROC” ["receiver operating characteristic“, from analyzing radar signals during WWII]



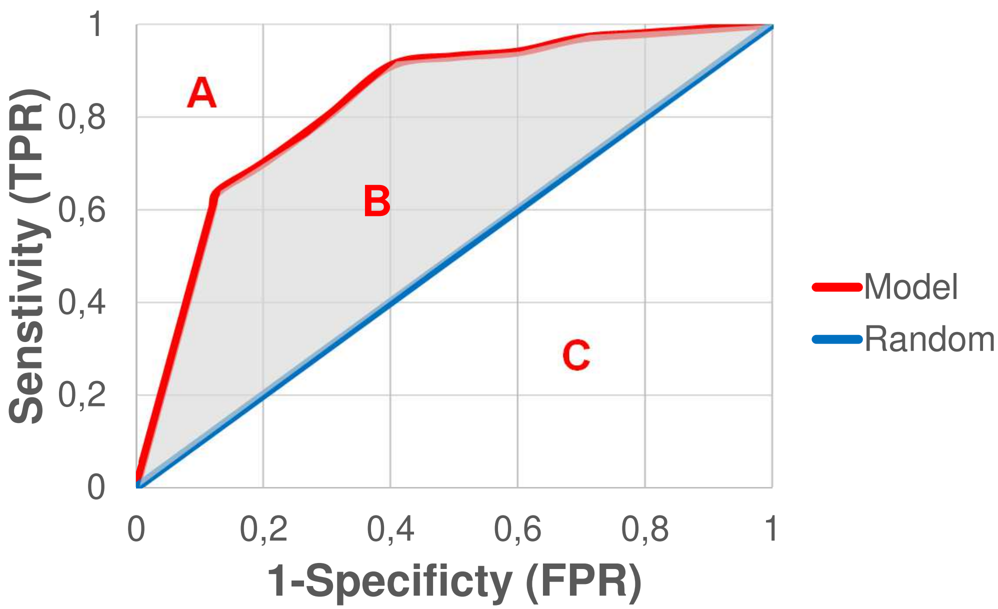
AUC (Area Under Curve)



AUC	Quality of Prediction
0.50	Random
0.50-0.60	Fail
0.60-0.70	Poor
0.70-0.80	Fair
0.80-0.90	Good
0.90-1	Excellent

*context specific (driverless car vs fin. instrument)

Gini Coefficient



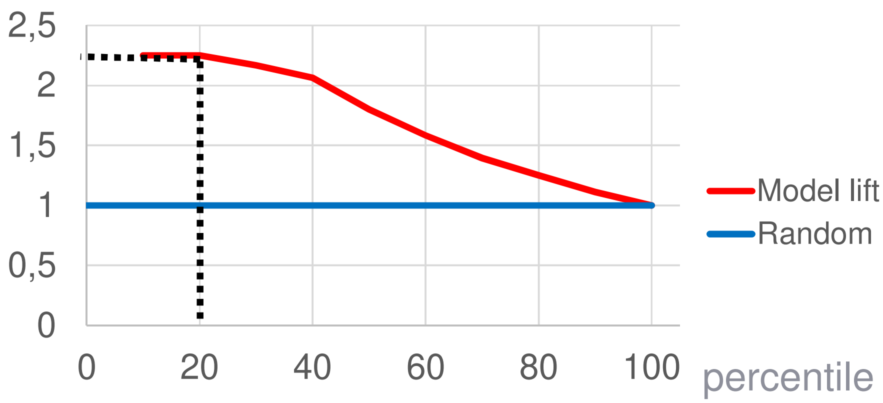
Gini coefficient (index, ratio):

Common measure of income distribution, named after the Italian statistician's 1912 paper

- Gini = B/(A+B)
- Note that AUC = B+C
- Because A+B+C=1, A+B=C=1/2:
- Gini = 2*AUC-1

Lift Chart / Gains Chart

- Lift is a metric of how much better a model does compared to just guessing.
- It evaluates **cumulative** model performance (especially relevant in marketing analytics)
- Example:
  - 20% of random customers correspond to 20% of those who are retained: random lift at 20^{th} percentile = 20/20=1
  - if 20% of the "best" customers per the model correspond to 45% of all retained customers, then model lift at 20^{th} percentile = 45/20 = 2.25



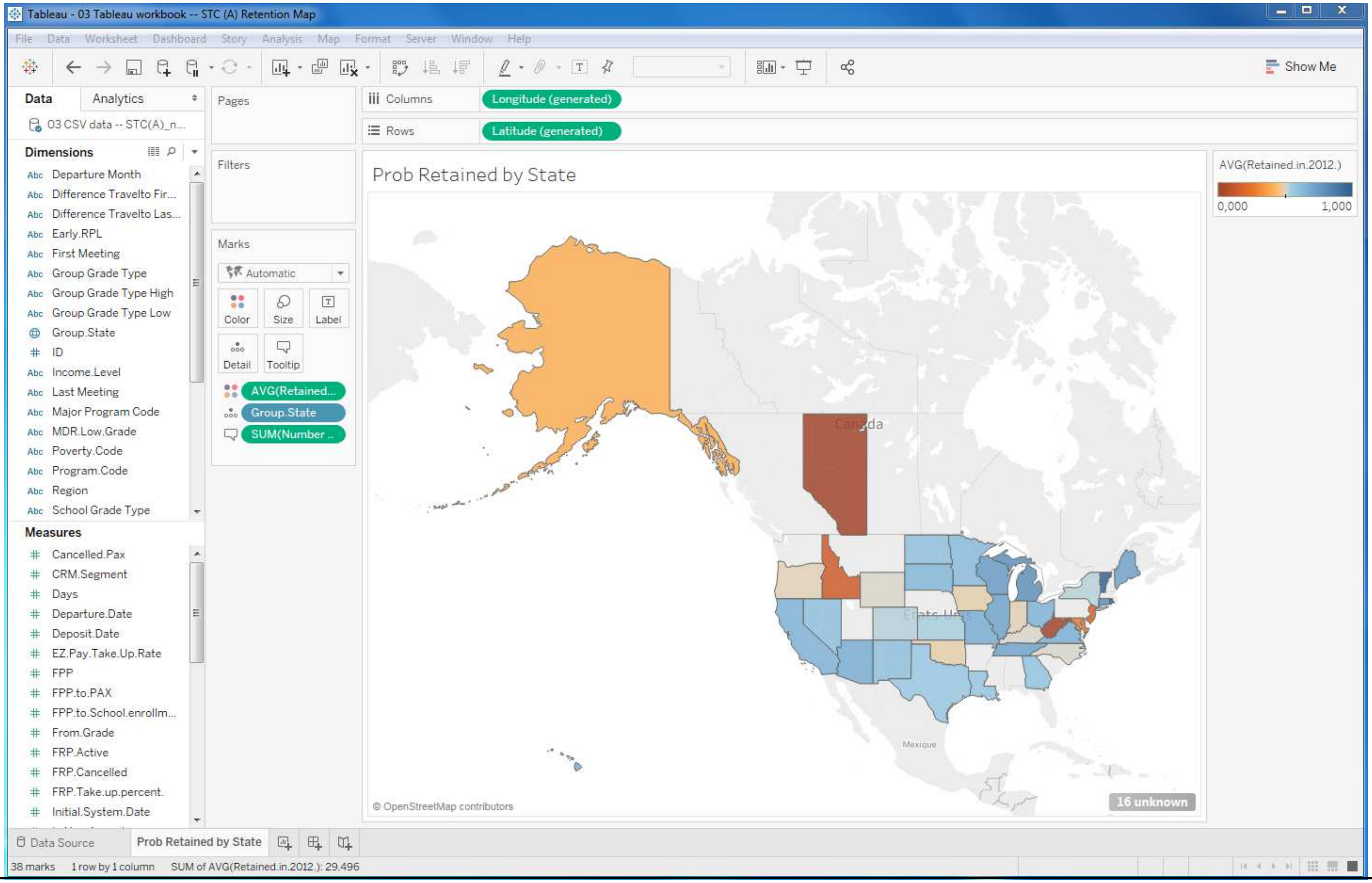
Now lets practice
STC(A) case

- What is the case about?
- What do we need to do?

- What do we have with the case?

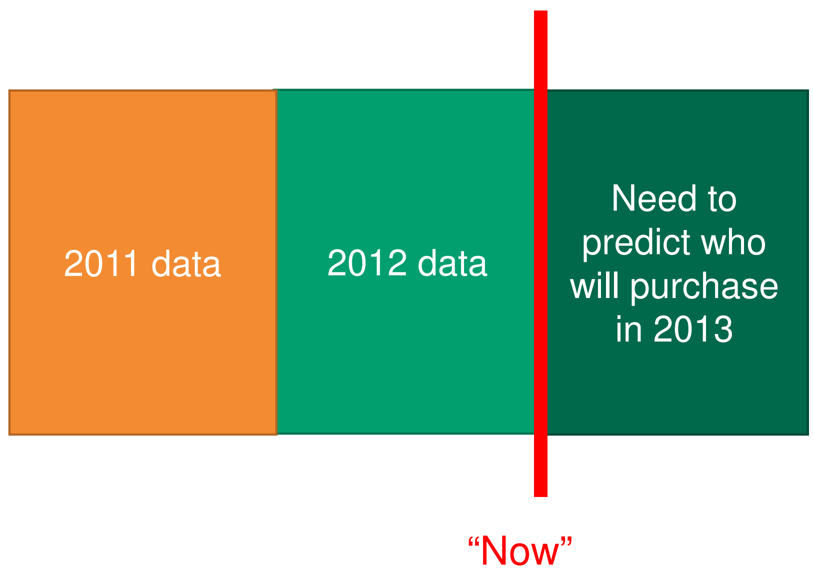
1. Data
   - "Dirty data" – data with some inaccuracies and omissions. Need to "clean" first. [This data is already pre-cleaned, BTW]
   - High-dimensional and "rare" categories [why is this a problem?]
2. Data dictionary – detailed description of the data fields – SUPER important document in any data analytics project
   - What does "Single.Grade.Trip.Flag" mean?
   - What does "Is.Non.Annual" mean?
   - What does "SPR.New.Existing" mean?
   - What does "FPP" "FRP", PAX" mean?

Mapping in Tableau (practice on your own)



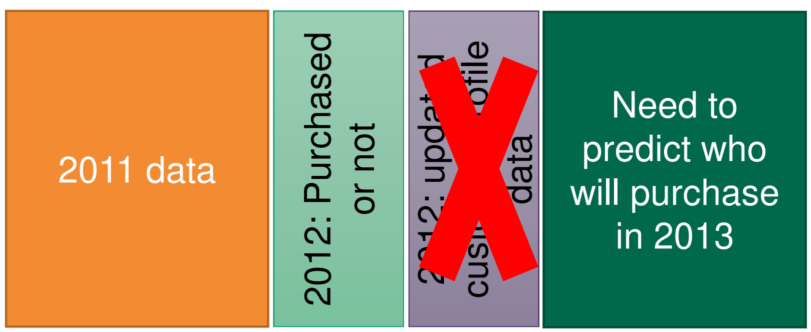
STC (A) Case: Data *versus* Prediction Task

- It is now spring 2013 and we want to predict which of our current (i.e. year 2012) customers will be retained in 2013
- What kind of data we have/need to best mimic that task?



STC (A) Case: Data *versus* Prediction Task (continued…)

- **Key idea:** The database will contain mode data that you know when you are making your prediction task
- Need to be very careful not to use data you learn concurrently with the outcome of your prediction
  - If not done, this is called "Target Leakage" – a major problem in analytics



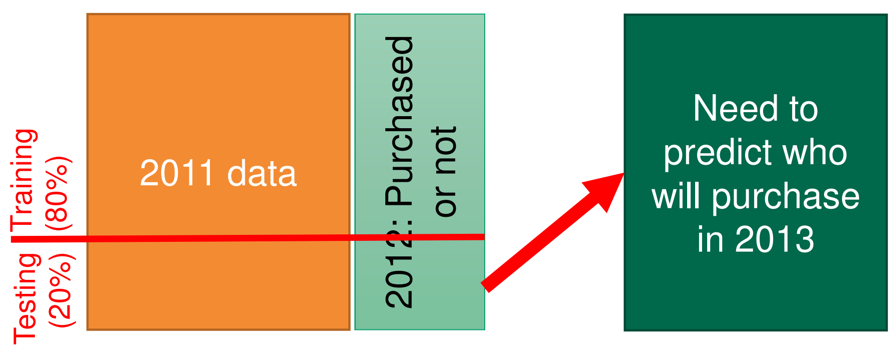
STC (A) Case: Training *versus* Testing Data

INSEAD

- How can we mimic the 2013 prediction task with 2011/12 data?

Key idea:

- Leave out some 2011/12 data and "pretend" as if it is a 2013 data [holdout]
- Use the rest of 2011/12 data [training] to build models, and assess quality of their predictions on the holdout



Now lets practice
STC(A) case

- Files on portal:
  - R-code 0506 R code -- STC (A) Logistic.R
  - CSV data 0506 CSV data -- STC(A) data_numerical dates.csv
    - BTW, how to generate the CSV datafile from an Excel case exhibit?
- The general structure of the code has the following steps:

1. Packages & libraries: package for managing packages, `pacman`
2. Load data
3. "Clean" data: formats, missing values (custom function `fixNAs`) + combine rare categories (`table` function to explore + custom function)
4. Split the dataset into testing vs training randomly
5. Run ("train") a model on the <u>training</u> data: `stepAIC` variable selection
6. Obtain model prediction for the <u>testing</u> data
7. Obtain metrics (confusion matrix + metrics, ROC curve, AUC, lift chart) for the <u>testing</u> data

Missing values

- VERY often, some of the data entries will be missing
- What should we do about it?
- Ignore? Bad idea: missing is often not random
- Fix missing values:
  - Categorical variables [easy] add a missing category
  - Continuous variables [harder]:
    - replace (with 0/mean/ median)
    - impute (create a separate mini-model to predict the missing values based on what's not missing)
- + add a "surrogate" dummy for each missing value

AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK
Poverty C	Region	CRM Segm	School Ty	Parent Me	Parent Me	Parent Me	MDR Low	MDR High	Total Scho	Income Le
B	Southern	4	PUBLIC	1	#########		K	5	927	Q
C	Other	10	PUBLIC	1	#########	#########	7	8	850	A
C	Other	10	PUBLIC	1	#########		6	8	955	O
	Other	7	CHD	0					0	
D	Other	10	PUBLIC	1	#########		6	8	720	C
C	Other	8	PUBLIC	1	#########		10	12	939	I
	Other	8	Catholic	1	#########		9	12	225	G
	Other	7	CHD	1	9/8/2010				0	
	Other	5	CHD	1	9/8/2010		6	12	500	K
	Houston	5	Private nc	1	#########		PK	8	635	K
	Other	10	CHD	1	9/9/2010		K	12	746	O
	Other	10	CHD	1	#########		PK	8	650	L
A	Northern	5	PUBLIC	1	#########		6	8	670	Q
B	Northern	5	PUBLIC	1		9/1/2010	6	8	750	L
	Northern	7	PUBLIC	1		9/9/2010			0	P5
B	Other	6	PUBLIC	1	#########	#########	6	8	753	I

Handling missing values: custom function “fixNAs”

INSEAD

```
fixNAs<-function(data_frame){      # Crete a function to fix NAs
integer_reac<-0  # Define reactions to Nas for different classes of variables as shown in your data str
factor_reac<-"FIXED_NA"
character_reac<-"FIXED_NA"
date_reac<-as.Date("1900-01-01")
for (i in 1 : ncol(data_frame)){   # Loop through columns, apply the defined reaction + create surrogate
    if (class(data_frame[,i]) %in% c("numeric","integer")) {
      if (any(is.na(data_frame[,i]))){
        data_frame[,paste0(colnames(data_frame)[i],"_surrogate")]<-
          as.factor(ifelse(is.na(data_frame[,i]),"1","0"))
        data_frame[is.na(data_frame[,i]),i]<-integer_reac      }
    } else
      if (class(data_frame[,i]) %in% c("factor")) {
        if (any(is.na(data_frame[,i]))){
          data_frame[,i]<-as.character(data_frame[,i])
          data_frame[,paste0(colnames(data_frame)[i],"_surrogate")]<-
            as.factor(ifelse(is.na(data_frame[,i]),"1","0"))
          data_frame[is.na(data_frame[,i]),i]<-factor_reac
          data_frame[,i]<-as.factor(data_frame[,i])          }
      } else {
        if (class(data_frame[,i]) %in% c("character")) {
          if (any(is.na(data_frame[,i]))){
            data_frame[,paste0(colnames(data_frame)[i],"_surrogate")]<-
              as.factor(ifelse(is.na(data_frame[,i]),"1","0"))
            data_frame[is.na(data_frame[,i]),i]<-character_reac          }
        } else {
          if (class(data_frame[,i]) %in% c("Date")) {
            if (any(is.na(data_frame[,i]))){
              data_frame[,paste0(colnames(data_frame)[i],"_surrogate")]<-
                as.factor(ifelse(is.na(data_frame[,i]),"1","0"))
              data_frame[is.na(data_frame[,i]),i]<-date_reac            }}}}    return(data_frame)       }
```

Do you need to know how to write such custom functions?
NO!
But you certainly can copy-paste this function and use it any time you need to deal with missing values

Further technical R remarks: all vars and combining categories

- Running a model with all variables included (use “dot” . for independent variables:

```
glm(Retained.in.2012.~., data=training, family="binomial"(link="logit")) # for logistic
ctree_tree<-ctree(Retained.in.2012.~.,data=training) # for CART
```

- Combining categories (this example, with less than **10** datapoints):

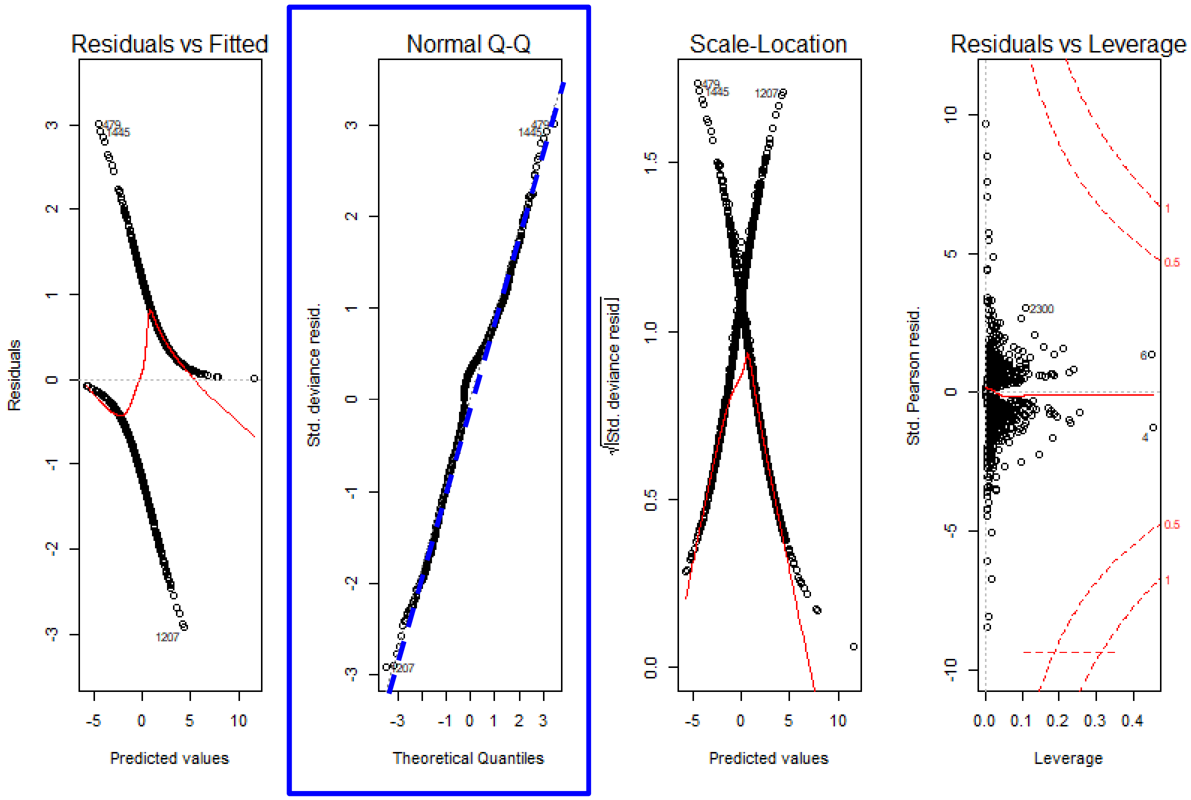
```
combinerarecategories<-function(data_frame,mincount){ #custom function to combine rare categories
  for (i in 1 : ncol(data_frame)){
    a<-data_frame[,i]
    replace <- names(which(table(a) < mincount))
    levels(a)[levels(a) %in% replace] <-
paste("Other",colnames(data_frame)[i],sep=".")
    data_frame[,i]<-a  }
return(data_frame) }
STCdata<-combinerarecategories(STCdata,10) #combine categories with <10 values in STCdata into “Other”
```

Variable selection

- In the world of "Big Data" 000s of variables can be easily engineered/created (e.g., through interactions)
- How to select which variables should be in the model?
- Forward/Backward/Stepwise regressions:
- Main idea (forward example):
  - Find which single X is most correlated with Y, add X1 to the model.
  - Given that X1 is already in the model, find which other variable adds most explanatory power. Add X2 and re-estimate the model.
  - Repeat until no variable can be added
- Today: `stepAIC` ["Akaike information criterion"]
  - `stepAIC(model_logistic,direction = c("both"),trace = 1)`
- Session 07-08: `LASSO/Ridge` – penalties for many variables ("regularizations")

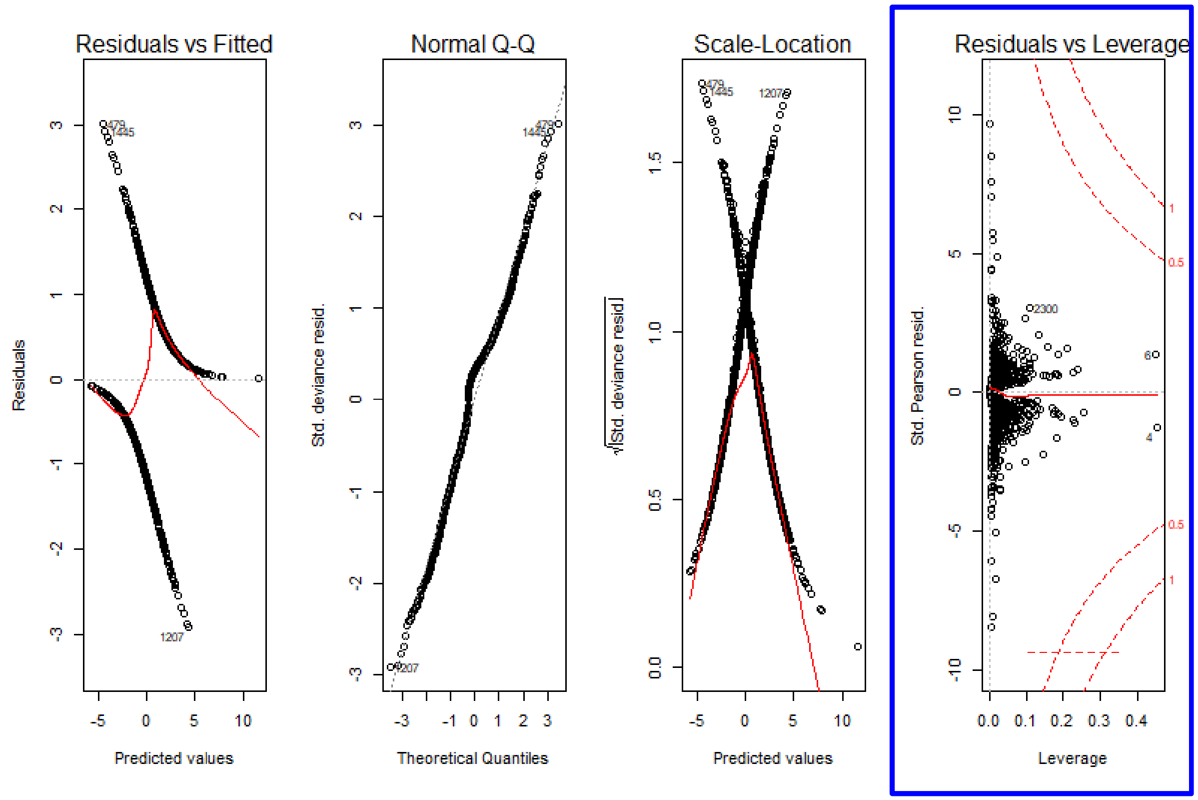
How “good” is our model?

- Model diagnositcs `plot(model_logistic_stepwiseAIC)`



How “good” is our model?

- Model diagnositcs `plot(model_logistic_stepwiseAIC)`



STC(A) results confusion matrix

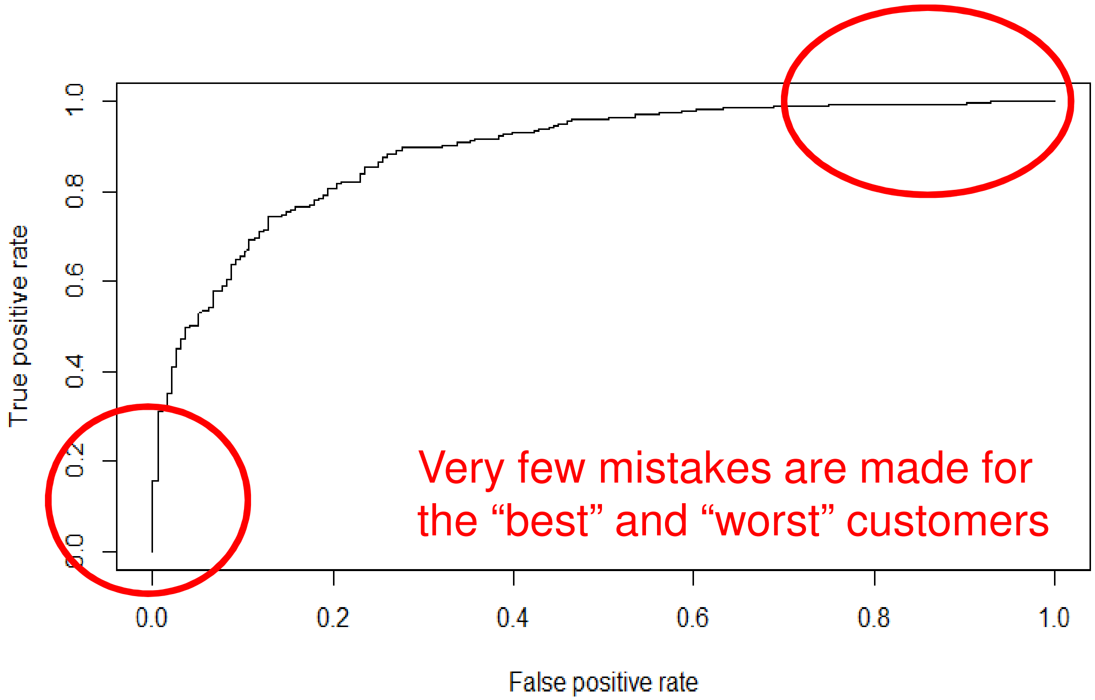
```
> confusionMatrix(logistic_classification,testing$Retained.in.2012.,positive = "1") #Display confusion matrix
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 151  71
         1  45 233

               Accuracy : 0.768
                 95% CI : (0.7285, 0.8043)
    No Information Rate : 0.608
    P-Value [Acc > NIR] : 2.293e-14

                  Kappa : 0.5245
 Mcnemar's Test P-Value : 0.02028

            Sensitivity : 0.7664
            Specificity : 0.7704
         Pos Pred Value : 0.8381
         Neg Pred Value : 0.6802
             Prevalence : 0.6080
         Detection Rate : 0.4660
   Detection Prevalence : 0.5560
      Balanced Accuracy : 0.7684

       'Positive' Class : 1
```

Our model is ~80% accurate

Accuracy is balanced

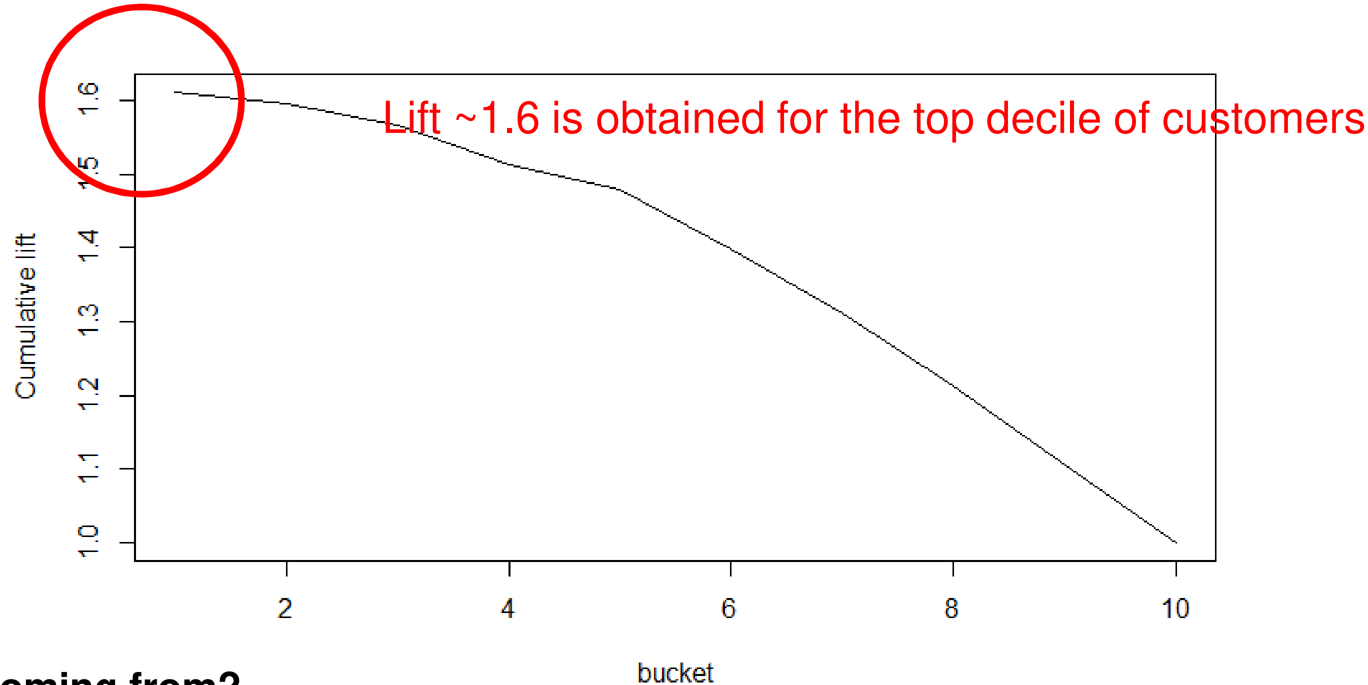
STC(A) results
ROC curve and AUC

*full data + advanced pre-processing



AUC = 89% ~ "Excellent"

STC(A) results
ROC curve and AUC



Where is 1.6 coming from?
On average 60.73% of customers are retained. So from the a decile of the testing data (50 customers), ~30 are expected to be retained.
But in the top decile, all 50 were retained, we see this from the ROC curve [how?], which is 50/30~1.6 times more than average

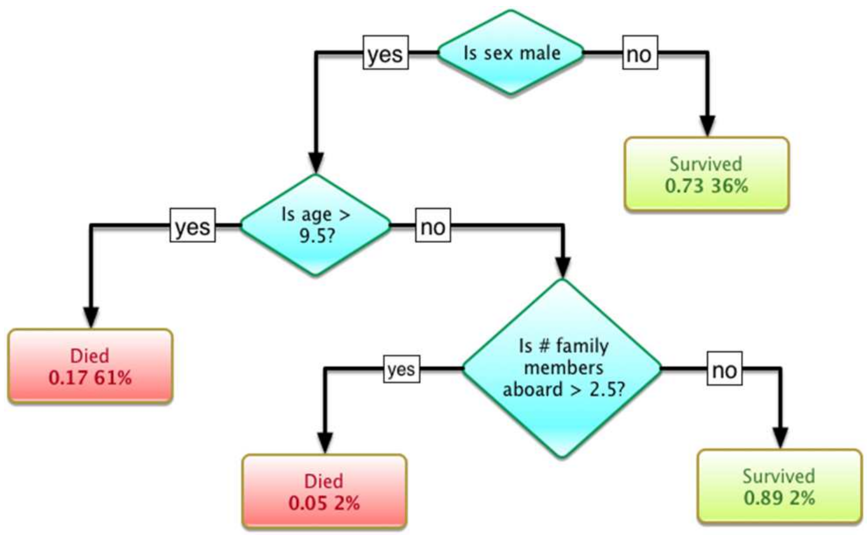
Intermediate summary: classification metrics, logistic regression and STC(A) case

- Classification ~ predicting events
- STC(A) case: need to predict which customers will purchase next year
- Logistic regression: predicting the probability of a purchase
- R "tricks":
  - "Cleaning" dirty data: fixing types, missing values and combining rare categories
  - Stepwise variable selection
  - Cross-validation: training the model on one subset of data, testing on another
- STC(A) case results so far with logistic regression:
  - Pretty good: overall accuracy ~80%, very few errors on top and bottom 30% of customers; clear guidance to marketing/operations
  - Structure of the model (significant variables) give insight into why some customers may not purchase

Next: CART
classification and regression tree

- **Main idea**: set of questions (business/decision rules) which partition data into pockets ("clusters") with similar characteristics

- These rules/questions form a tree-like graphic:
- Example: surviving the Titanic crash
  - #s in parenthesis: (prob. survive, % of data)



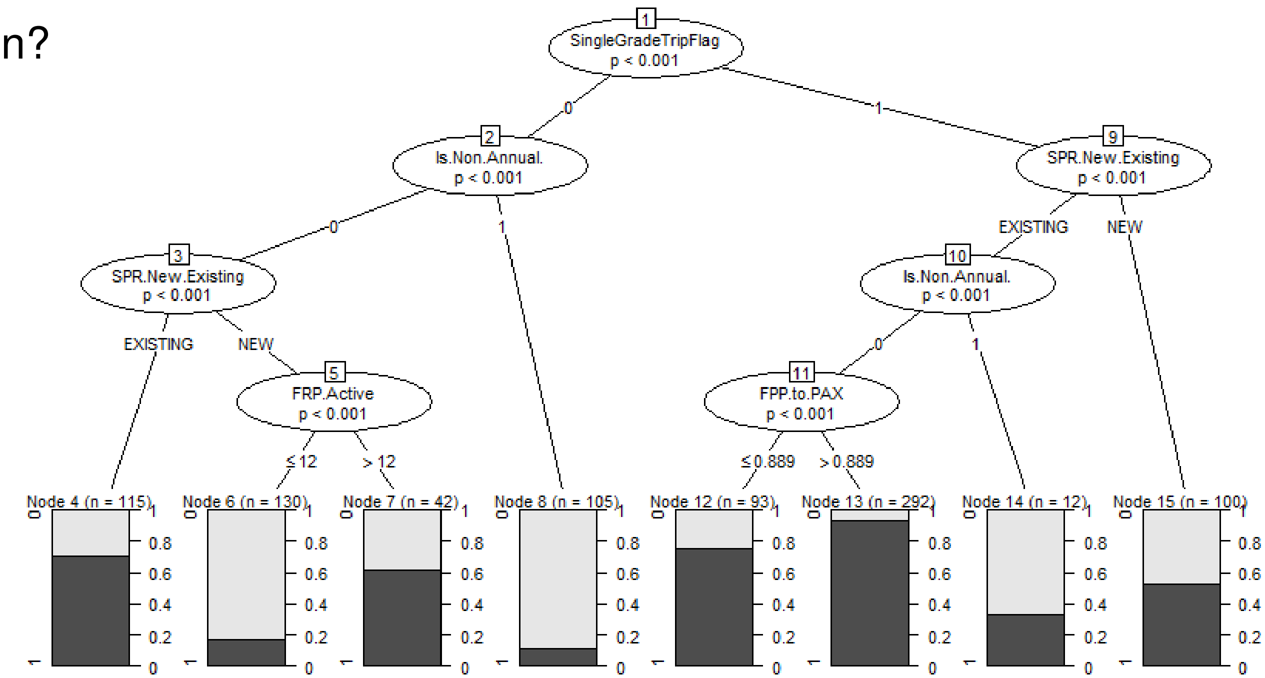
- Several way to "build" trees
  - We will look at two:
    - Conditional inference, `ctree`
    - Recursive partitioning, `rpart`
- CART is a "mother" (father?;) of many machine learning methods, e.g., random forest, gradient boosting machines (`xgboost`) [time-permitting or Session 0708]

Source: https://en.wikipedia.org/wiki/Decision_tree_learning

STC(A), `ctree` CART

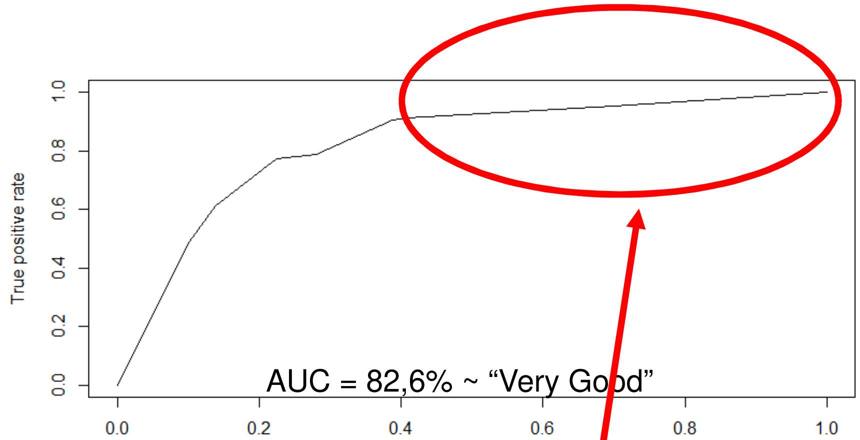
Remarks:

- `ctree` is slow and takes lots of memory when dealing with high-dimensional categorical data: combine categories [next slide] or shrink training set
- Resultant tree:
- Interpretation?

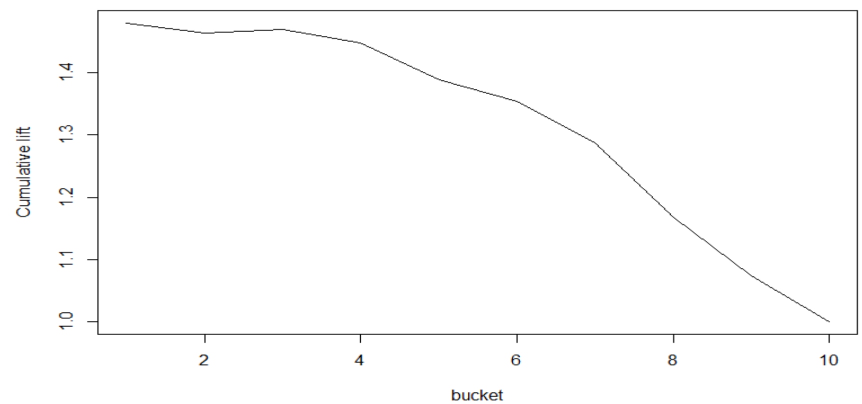


STC(A) results:
ctree CART

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 129  35
         1  67 269

               Accuracy : 0.796
                 95% CI : (0.758, 0.8305)
    No Information Rate : 0.608
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5593
 Mcnemar's Test P-Value : 0.002144

            Sensitivity : 0.8849
            Specificity : 0.6582
         Pos Pred Value : 0.8006
         Neg Pred Value : 0.7866
             Prevalence : 0.6080
         Detection Rate : 0.5380
   Detection Prevalence : 0.6720
      Balanced Accuracy : 0.7715

       'Positive' Class : 1
```



Disbalanced: much better at predicting who will purchase: under 10% mistakes in top 60%

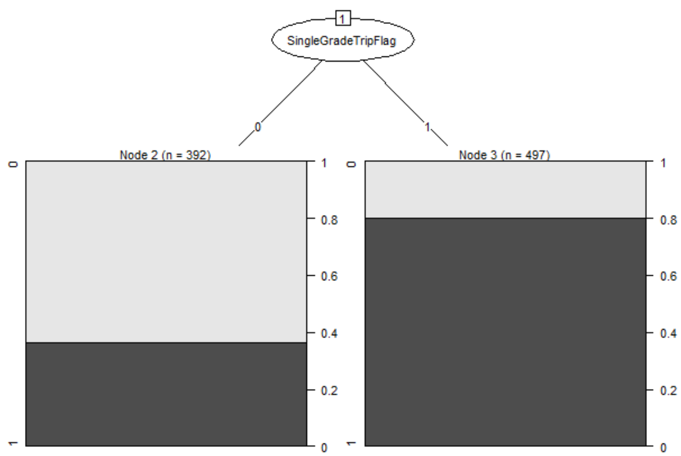


File: 0506 R code -- STC (A) CART.R

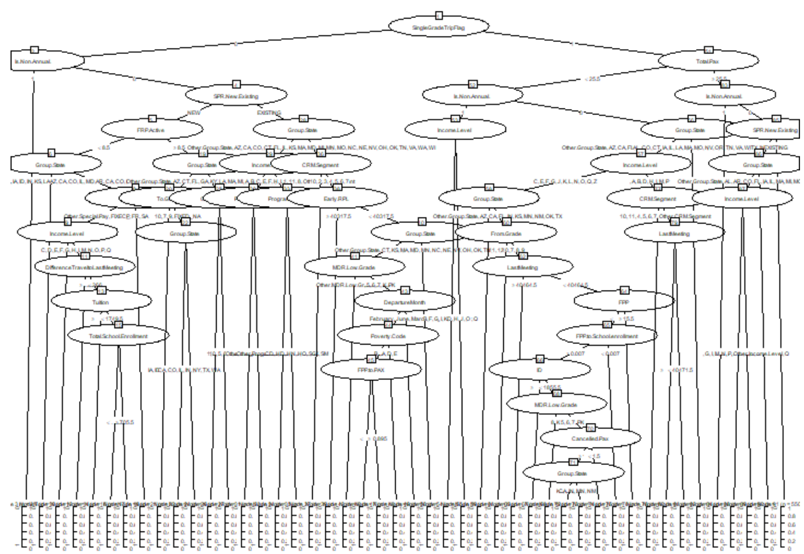
STC(A), `rpart` CART

Remarks:

- Unlike ctree, `rpart` methodology relies on a user-specified “cost paramter” (`cp`) to decide how to prune the tree
  - High `cp`: small tree, possible loss of precision on training and testing
  - Low `cp`: large tree, better fit on testing, but overfitting on training
- Interpretation?



cp=0.2
(too little complexity, “underfit”)

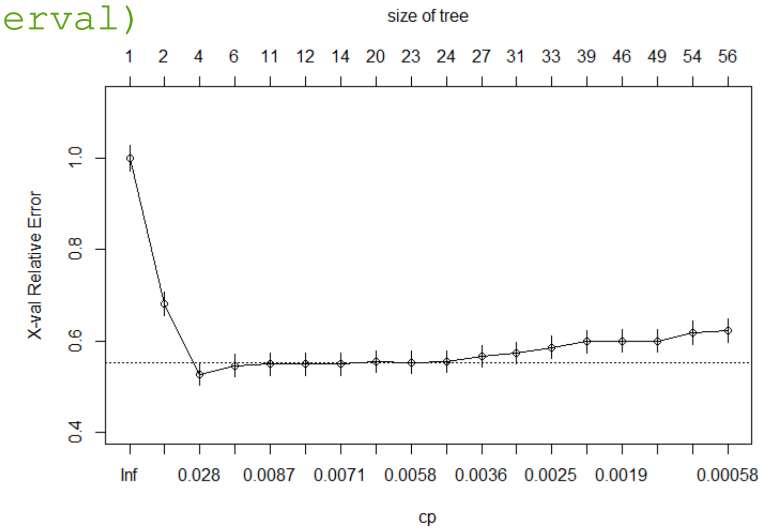


cp=0.002
(too much complexity, “overfit”)

STC(A), `rpart` CART

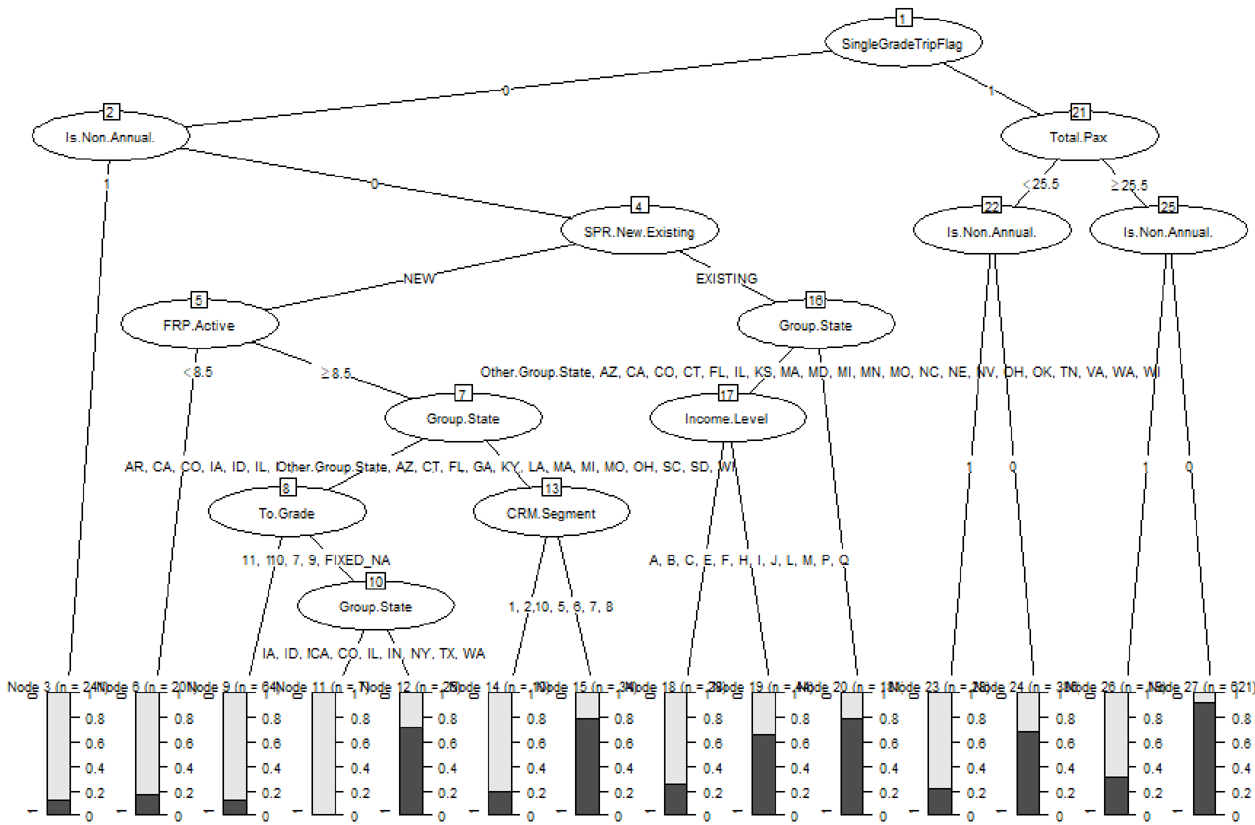
Remarks:

- Unlike ctree, rpart methodology relies on a user-specified "cost paramter" (`cp`) to decide how to prune the tree
  - High `cp`: small tree, possible loss of precision on training and testing
  - Low `cp`: large tree, better fit on training, but overfitting on testing
- Which `cp` to use?
- `plotcp(rpart_tree) # rule of thumb: pick the largest cp at which error crosses dotted line ("confidence interval)`
- In our case, 0.028~0.0036



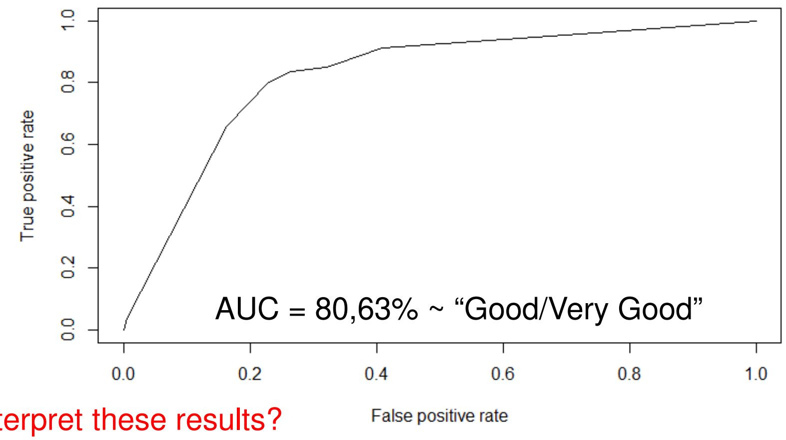
STC(A) results:
`rpart` CART with cp=0.007

- Interpretation? Does the tree “make sense”?

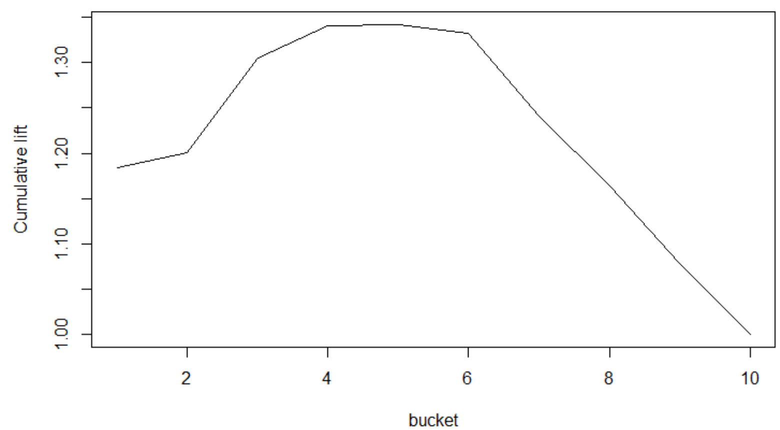


STC(A) results:
`rpart` CART with cp=0.007

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 123  36
         1  73 268

               Accuracy : 0.782
                 95% CI : (0.7432, 0.8174)
    No Information Rate : 0.608
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5268
 Mcnemar's Test P-Value : 0.0005644

            Sensitivity : 0.8816
            Specificity : 0.6276
         Pos Pred Value : 0.7859
         Neg Pred Value : 0.7736
             Prevalence : 0.6080
         Detection Rate : 0.5360
   Detection Prevalence : 0.6820
      Balanced Accuracy : 0.7546

       'Positive' Class : 1
```



Do we know how to interpret these results?



File: 0506 R code -- STC (A) CART.R

Exercise TODO at home: first-hand glance at overfitting

- Create a table of AUCs for the `rpart` method using various cps on both training and testing data

- Do you observe that while on training the AUC improves the lower `cp` you use?
  - Why? A: the tree becomes more elaborate.

- But what happens on testing data?
  - Do you observe that those elaborate trees preform worse - exactly because they too elaborately capture the nuances of the training data, which may not be present in testing.

- That's overfitting!

cp	$AUC_{testing}$	$AUC_{training}$
0.1	0.735768	0.730846
0.05	0.783994	0.7806043
0.01	0.783994	0.8281842
0.005	0.8070875	0.8901623
0.001	0.8070875	0.9386828
0.0005	0.7989729	0.9418476
0.0001	0.7989729	0.9418476

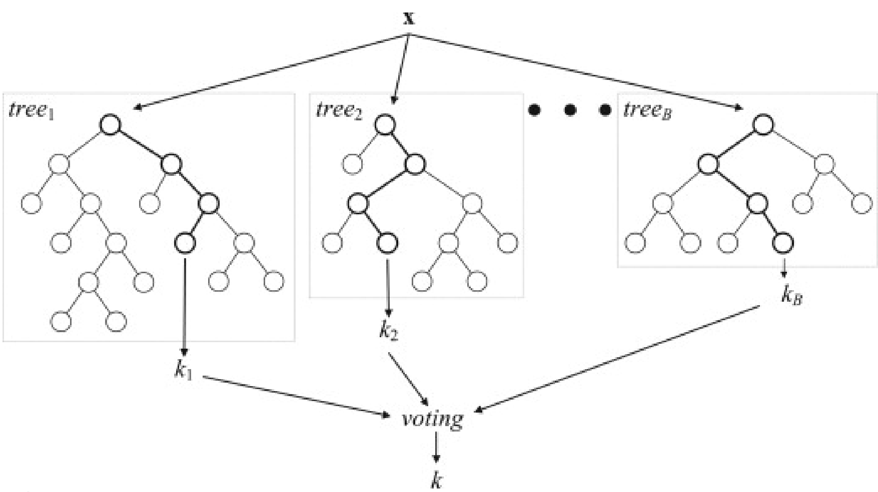
[Optional/Time-permitting] additional methods

- CART-based
  - `randomforest`
  - gradient boosting machines (`xgboost`)

- Session 07-08
  - regularizations (LASSO/Ridge)
  - support vector machines (SVM)
  - artificial neural networks (ANN)

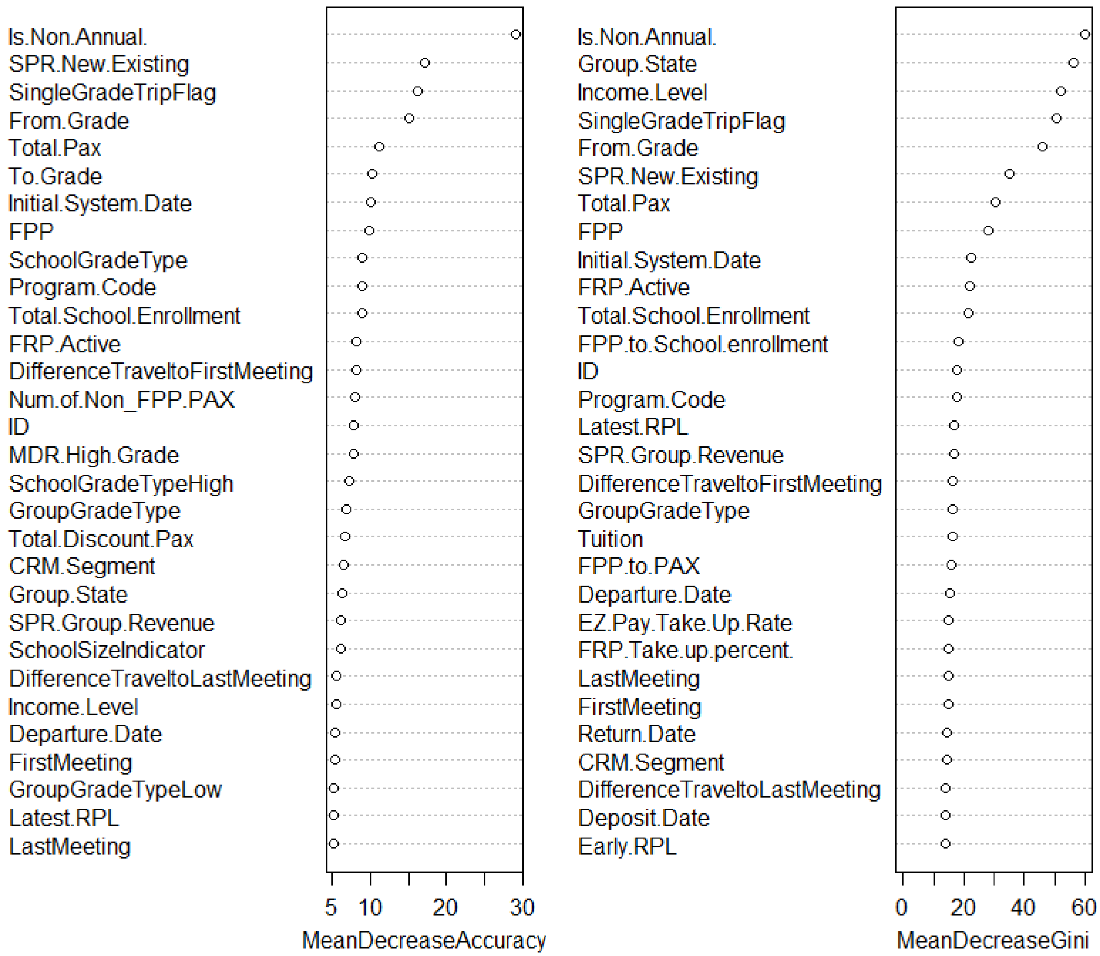
Random Forest

- **Key idea**: Fit many trees to different samples of data, then ensemble them
- Note: No need for a separate testing data; as it "rotates" with each new tree
- In addition to simply making a prediction, random forest provides an important insight into which variables show up in many trees - "important variables"



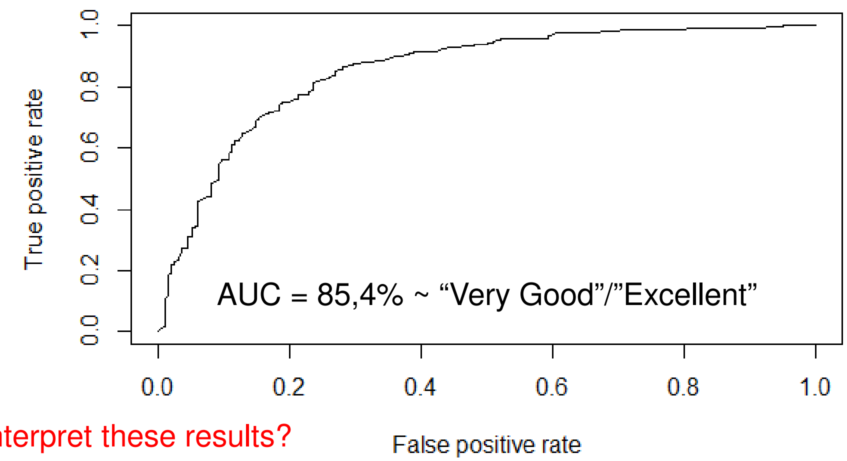
- Why does this work?
  - “Wisdom of Crowds”
  - Decision-making by a committee: Parliament vs Dictator, Board vs CEO

Variable Importance

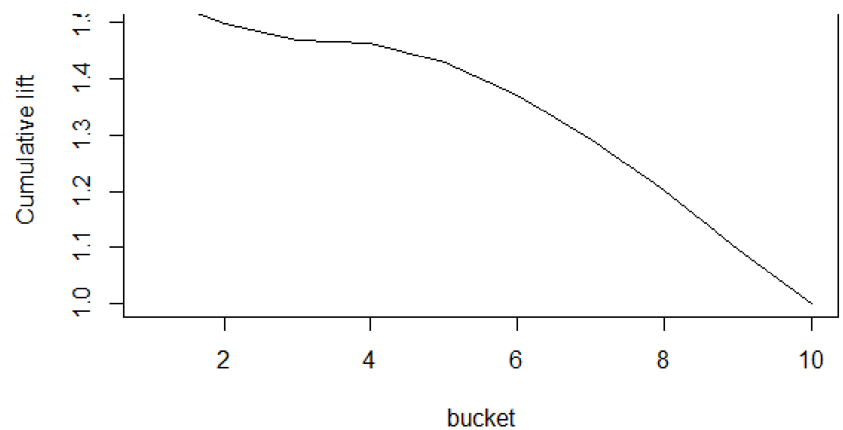


STC(A) results: randomforest

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 163  82
         1  33 222

               Accuracy : 0.77
                 95% CI : (0.7306, 0.8062)
    No Information Rate : 0.608
    P-Value [Acc > NIR] : 1.062e-14

                  Kappa : 0.538
 Mcnemar's Test P-Value : 7.605e-06

            Sensitivity : 0.7303
            Specificity : 0.8316
         Pos Pred Value : 0.8706
         Neg Pred Value : 0.6653
             Prevalence : 0.6080
         Detection Rate : 0.4440
   Detection Prevalence : 0.5100
      Balanced Accuracy : 0.7809

       'Positive' Class : 1
```



Do we know how to interpret these results?



File: 0506 R code -- STC (AB) random forest.R

Parameters vs **Hyper-**Parameters `randomForest`

- Recall regression. There were two kinds of “things” to figure out:
  - What are the coefficients of the variables?
  - How many (which?) variables to include in regression?

Parameters, learned from data

Hyper-parameters, determine how the learning will take place

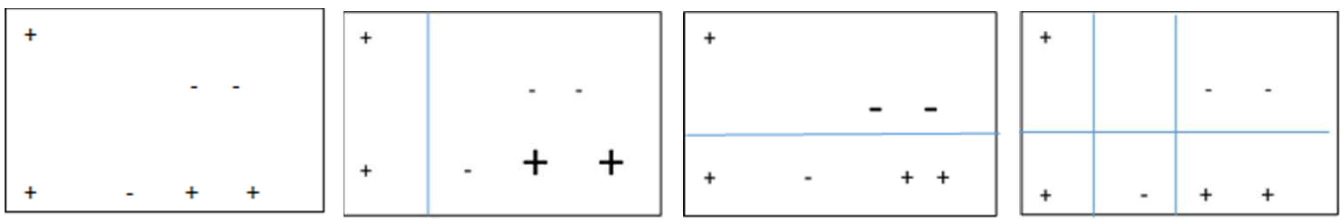
- In `rpart`:
  - Complexity parameter `cp` is a hyperparameter
- In `randomforest`:
  - Number of trees in the forest, size of each tree, number of columns to sample to grow a tree, how sampling works (replacement / no), how voting works, etc.

```
model_forest <- randomForest(Retained.in.2012.~ ., data=training,
                             type="classification",
                             importance=TRUE,
                             ntree = 500,           # hyperparameter: number of trees in the forest
                             mtry = 10,             # hyperparameter: number of random columns to grow each tree
                             nodesize = 10,         # hyperparameter: min number of datapoints on the leaf of each tree
                             maxnodes = 10,         # hyperparameter: maximum number of leafs of a tree
                             cutoff = c(0.5, 0.5)   # hyperparameter: how the voting works; (0.5, 0.5) means majority vote
                             )
```

- How to find more info/code? help: `?randomForest`
- How to determine (“tune”) the values of hyper-parameters: grid-search with cross-validation

Gradient Boosted Trees:
`xgboost`

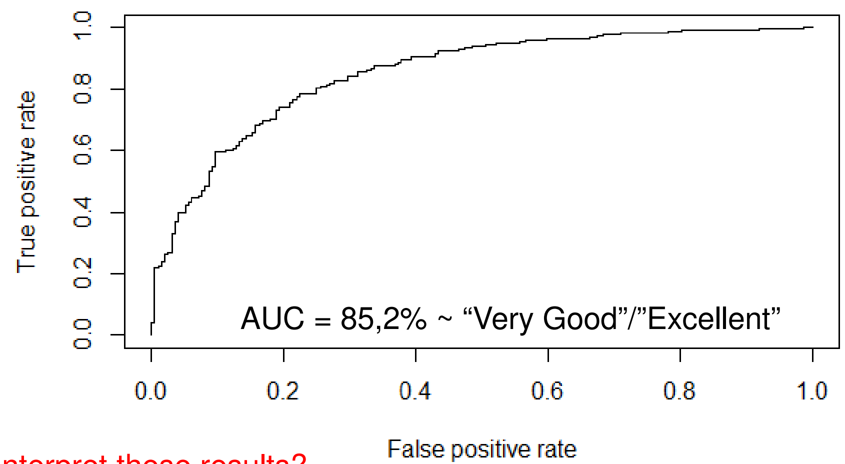
- **Key idea:** Notice which data points are not explained well by the existing tree, make those data points more important ("higher weight") and re-fit to describe them better
- Combine variables and add new splits to explain those higher weight data points



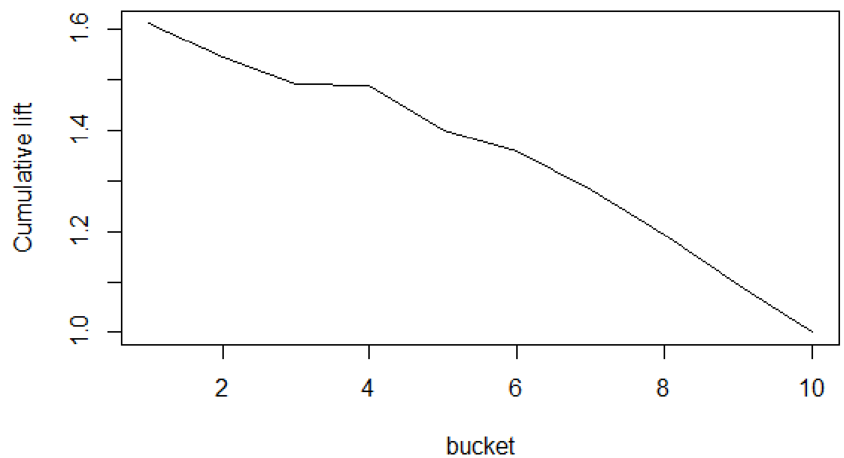
- Requires specifying additional hyper-parameters: size of tree ("depth"), learning/decay rate, number of boosting iterations, etc.
- Determined vis grid search with cross-validation;
  - Example: 10 values for size of tree * 10 values for learning rate * 10 values for # of iterations *5-fold cross-validation = 5000 models to run (each time learning model parameters from training data and testing on testing data) → parallel computing(!)

STC(A) results:
xgboost

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 143  56
         1  53 248

               Accuracy : 0.782
                 95% CI : (0.7432, 0.8174)
    No Information Rate : 0.608
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.5439
 Mcnemar's Test P-Value : 0.8481

            Sensitivity : 0.8158
            Specificity : 0.7296
         Pos Pred Value : 0.8239
         Neg Pred Value : 0.7186
             Prevalence : 0.6080
         Detection Rate : 0.4960
   Detection Prevalence : 0.6020
      Balanced Accuracy : 0.7727

       'Positive' Class : 1
```



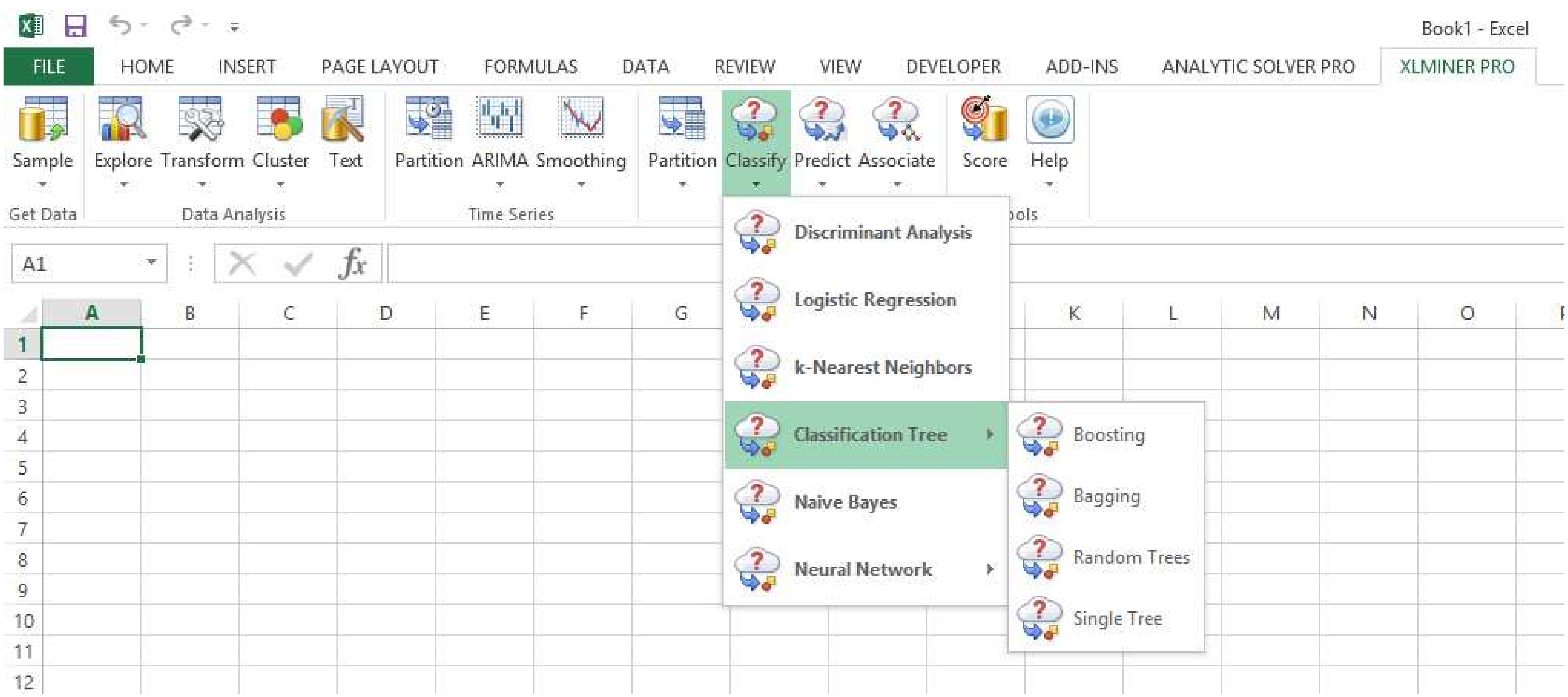
Do we know how to interpret these results?



File: 0506 R code -- STC (AB) xgboost.R

CART-like Methods in Excel

Part of XLMiner Pro by Frontline Systems (same company that makes Solver)



Summary of Sessions 5-6

- Large volumes of data about people/behavior increased the importance of an analytical task to predict an outcome of an event:
  - Will a customer churn? Default? Open email? [binary outcome]
  - Which item from a set will the customer choose? (iPhone model, bottle size, transit mode, job offer) [multinomial outcome]
- Predicting events ~ prob $\rightarrow$ T $\rightarrow$ classification
- We studied "two-plus" Data Science methods for classification:
  - Logistic regression: build a linear model for utility and an exp transformation to predict the probability of an event
  - CART: build a decision-tree-like structure for describing pockets of data with similar properties wrp the occurrence of the event
  - CART generalizations: random forest and gradient boosting – most-commonly used ML models today. Hyper-parameters
- R code templates + some further "tricks"
  - data cleaning, custom functions, random split into train vs test, stepAIC

Next…

- Tutorial 2:
  - Mid-term R help, specifically on predicting events, STC(B) case
  - Data manipulations in R: `dplyr` package
- Assignment 2:
  - “Predicting credit defaults”, due by Session 07-08; we will discuss A2 in class 0708
  - Note:
    - A2 is harder than A1, budget the time accordingly
    - Reach out to TA if “in trouble”

INSEAD

The Business School
for the World®

Europe | Asia | Middle East