BAM - AS&P

Aleksander Odziemkowski

October 13, 2022

Individual Assignment 3

1 Panel data modeling: time to export

This section examines a quality of trade conditions across countries in the period 2014-2019 using World Bank data. This is being done by means of regression analysis on panel data. Given the longitudinal structure of the the data various regression techniques will be applied to end up at unbiased estimators.

1.1 Population model specification

Throughout this part of the document attention will be drawn to the following specification of the population regression model:

$Time2Export = \beta_0 + \beta_1 Cost2Export + \beta_2 TaxOnExport + \beta_3 ExportGrowthRate + \varepsilon, \quad \varepsilon \quad n(0, \sigma)$

The descriptive nature of such association surface being defined is as follows. Both costs of exporting goods as well as tax rate applied to those goods are viewed by the exporting firm as a monetary cost. That being said, the time it takes to complete the process of exporting sold goods can be intentionally delayed by the company so the costs are not accrued when it is not desired. However, the nature of the relationship can also be flipped around as the process of exporting might be lengthier due to higher cost-intensity of the process. It is interesting to see whether the deferrals or the costliness are affecting time to export more.

Additionally, annual growth rate of export is expected to be associated with longer time to export. With growing economy, firms cannot adjust/increase the their output of exporting goods as quickly as quickly the market improves. Therefore, in the time of bull market and growing export, time it takes to complete the process should be longer.

1.2 Summary statistics

After downloading and balancing the data to reflect only complete observations of features of interest in the period 2014-2019, the set of variables is ready for the analysis. Table 1 compiles said variables and presents their descriptive statistics.

Table 1: Summary statistics of the variables in the panel data model.

Statistic	Mean	St. Dev.	Min	Pctl(25)	Median	Pctl(75)	Max
Time2Export	48.496	54.642	0.000	6.000	36.000	71.625	312.000
TaxOnExport	1.104	4.582	0.000	0.000	0.000	0.005	43.730
Cost2Export	331.792	328.182	0.000	100.000	274.000	456.005	1,975.000
ExportGrowthRate	6.088	48.293	-39.051	0.826	4.150	7.376	1,247.420

The variable of interest, *Time2Export*, reflects hours to be spent on adhering to the border compliance and completing the process of exporting sold goods. On average it took 48.50 hours to export goods in countries included in the balanced data set over the period in question. It is worth noting how positively skewed this measure is - mean to the left of the median indicates more extreme observations in the right tail. *TaxOnExport* is defined as amount of taxes to be paid on exported goods as percent of revenues. Percents are presented as rational number instead of proportion for meaningful interpretation of regression results. One can notice that for most of the observations taxes on exports are zero, though they can go as high as 43.73%.

Costs of exporting goods and fulfilling border compliance expressed in USD are captured by *Cost2Export*. Mean costs are equal to \$331.79 with standard deviation at \$328.18. Given that SD is almost as high as the mean, one can say the data for *Cost2Export* is quite dispersed. *ExportGrowthRate* indicates an annual growth rate of exported goods and services and has also been transformed from proportion to rational number as was

TaxOnExport. Given the fact that data set looks at countries worldwide, meaningful differences among the economies exist. It results in a significant spread of data as the range extends from -39.05% to 1,247.42% while standard deviation equals 8x the mean.

1.3 Model estimation in various specifications

Previous formulation of population regression model will be now estimated based on pooled regression, between regression, fixed-effect regression, and random-effect regression. In total 4 models are presented. Data set this document is working with is a panel data across countries and years, which is the reason for going beyond OLS estimation. Various techniques of regression result in different estimates of coefficients, standard errors, goodness-of-fit and ultimately - interpretation of the estimates. Table 2 summarizes the results of each of the model.

Table 2: Model estimation using various techniques.

	Dependent variable:			
	Time2Export			
	Pooled	*Between*	*Fixed*	*Random*
	(1)	(2)	(3)	(4)
Constant	8.054***	6.463		17.939***
	(2.248)	(5.561)		(4.894)
Cost2Export	0.113***	0.113***	0.047***	0.089***
	(0.005)	(0.012)	(0.015)	(0.009)
TaxOnExport	1.754***	1.900**	-0.025	0.193
	(0.332)	(0.824)	(0.336)	(0.315)
ExportGrowthRate	0.034	0.239	-0.001	-0.001
	(0.031)	(0.196)	(0.008)	(0.008)
Observations	606	101	606	606
R^2	0.492	0.513	0.019	0.130
Adjusted R^2	0.489	0.498	-0.182	0.126
Residual Std. Error		38.353 (df = 97)		
F Statistic	194.368*** (df = 3; 602)	34.044*** (df = 3; 97)	3.293** (df = 3; 502)	89.926***
Note:				*p<0.1; **p<0.05; ***p<0.01

Pooled regression is the closest to the OLS technique in terms of the interpretation among all the models above. In fact, the panel data is being "pooled" together and estimated using ordinary least squares so the insights are straightforward. Increase of costs of exporting per USD as associated with an increase of time to export by 0.113 hours. Increase of growth rate of export by 1 percentage point corresponds with an increase of time to export by just 0.034 hours. Among the variables in the model, the highest change of time to export, 1.754 hours, is associated with an increase of taxes paid on exported goods by 1 percentage point.

Model of between-groups reflects the variation of specific countries' means around the overall means. In other words, it is the portion of a total variation in the explained variable that is not a variation enclosed in each of the group/country. The direction of the effects remained the same as in pooled model, though some variables' coefficients were magnified. *TaxOnExport* increase by 1 percentage point is associated with an increase of *Time2Export* by 1.9 hours and for growth of export the effect increased to 0.239 hour increase per a unit of percent. Worth noting is that the time-dimension is collapsed, which can be observed in the number of observations as it is 6 times smaller than in other models. Sampling variation is therefore bigger in this model as standard errors are higher.

In fixed-effect model the aim is to capture individual-specific attributes that do not vary across time by allowing differences across groups to be assigned to the intercept. In fact, each country has its own intercept, that is why it is not visible in Table 2. Overall, the estimates of fixed-effect model are further away from pooled model than the between-model. It implies more weight being applied to the results of the latter model. Interestingly enough, when looking at each country separately in FE, the associations between *Time2Export* and *TaxOnExport* and *ExportGrowthRate* turn negative. Each increase of percentage point of taxes on export leads to shorter time to export by 0.025 hours. The rate at which export growth seems to not affect the exporting time much, with an effect of only -0.001.

In contrast to fixed-effects, random-effects model assumes that the effects vary across cross-sectional units. Basically, countries are being partially pooled together again. The effects are once again positive with respect to dependent variable for costs of export and taxes applied to exported goods and remain almost non-existent for annual growth rate of export. However, the magnitude of coefficients is smaller than for pooled model.

1.4 Specification of choice

Having estimated many models for the panel data, one might apply format statistical tests to decide between the specification of choice. At the end of the day, it is useful to have a formal proof of using one method over the other. The order in which choice of model is tested is derived from the complexity of assumptions - one would use a simpler model instead of a more complicated one.

Firstly, partial F test is applied to decided between a pooled regression model and within-variation model (fixed-effects). By applying partial F test joint contribution of estimated coefficients is tested for significance. Insignificant test result would indicate that both models are consistent, which nudges the user in favor of pooled model in terms of simplicity. Results of that test are available in Table 3.

Table 3: Decision rule for choosing between pooled-regression and fixed-effects by the means of partial F test.

df1	df2	Statistic	p-value	Method	Alternative
176	502	64.906	2.99499088898141e-265	F test for individual effects	significant effects

With test statistic F = 64.906 at p-value < 0.0001 the result of test is statistically significant and null hypothesis is rejected, which deems user to prefer specification with fixed-effects as there is unobserved heterogeneity in the data.

Secondly, once the claim of unobserved heterogeneity has been made, one might test whether fixed-effects or random-effects are a preferable specification to account for said heterogeneity. For that purpose Hausman test is conducted. For this specification test, null hypothesis states that there is no correlation between disturbances and explanatory variables in the model, whilst alternative hypothesis informs about existence of said correlation in the data. Results are displayed in Table 4. Given the p-value < 0.0001 the result is significant and informs about preference of fixed-effects over random-effects model due to consistency.

Table 4: Decision rule for choosing between fixed- and random-effects by the means of Hausman test.

Statistic	p-value	df	Method	Alternative
19.249	0.000242851919144067	3	Hausman Test	One model is inconsistent

2 Counts data modeling: the Mashable case

This part of the document pertains to modelling count data of the number of news articles shares on a particular website. For this purpose use is being made of Online News Popularity data set. It describes 39,797 instances with 61 attributes of which 58 are predictive attributes, 2 are non-predictive and 1 is goal field (that is the number of shares in social network which describes a popularity of a link).

2.1 Variables of choice and summary statistics

In this section nine most important variables that explain the sharing of online news articles are determined using author's comprehension of the business case. Variables in the model are as follows:

- *shares* - number of shares (target),
- *is_weekend* - was the article published on the weekend (yes = 1, no = 0)?,
- *n_tokens_content* - number of words in the content,
- *num_hrefs* - number of links,
- *num_imgs* - number of images,
- *num_videos* - number of videos,
- *num_keywords* - number of keywords in the metadata,
- *n_tokens_title* - number of words in the title,
- *global_sentiment_polarity* - text sentiment polarity (included as an index v*100 for the sake of interpretation),
- *global_subjectivity* - text subjectivity (included as an index v*100 for the sake of interpretation).

Summary statistics for the variables outlined above are presented in Table 5. The dependent variable has a mean of 3,395.38 shares per article with a relatively high standard deviation of 11,626.95. It means the data is highly skewed to the right (as median is 1,400 vs higher mean). It appears that might be caused by extremely popular articles as the maximum value is 843,300 shares.

Number of words in the title are distributed quite symmetrically as one might assume there is a certain standard of short and concise web titles. Number of words in the published content suggests that on average texts are relatively short (mean 546.51 words) but also large pieces are available in the dataset (max 8,474 words). Variables describing number of links, images, videos and words in the metadata follow similar pattern with distributions just slightly positively skewed.

Articles were mostly published on the weekdays as the mean of variable *is_weekend* shows proportion to be roughly 90:10. Two variables have been transformed in relation to how they appear in the original dataset. Indicator of text sentiment polarity (text presenting opposite or contradictory opinions) and indicator of text subjectivity (text being influenced by personal feelings or opinions) are multiplied by 100 to properly reflect the unit change in regression results later on. For instance, the polarity index can be both positive and negative as it measures two side of the spectrum. Given the mean of 11.93 for index sentiment polarity one can say that on average content in the link tends towards a certain direction (side of the argument) that was assumed to be positive. A value of 100 would mean that content was pushing one side of the spectrum to the extreme, the same with -100 - just the other side of the argument. Subjectivity index assumes values from 0 to 100 as the content can be objective (*index_subjectivity* = 0) or subjective to the extreme (*index_subjectivity* = 100). On average the content was almost as much objective as it was subjective (mean 44.34).

Table 5: Summary statistics of the variables in the count model.

Statistic	Mean	St. Dev.	Min	Pctl(25)	Median	Pctl(75)	Max
shares	3,395.38	11,626.95	1	946	1,400	2,800	843,300
n_tokens_title	10.40	2.11	2	9	10	12	23
n_tokens_content	546.51	471.11	0	246	409	716	8,474
num_hrefs	10.88	11.33	0	4	8	14	304
num_imgs	4.54	8.31	0	1	1	4	128
num_videos	1.25	4.11	0	0	0	1	91
num_keywords	7.22	1.91	1	6	7	9	10
is_weekend	0.13	0.34	0	0	0	0	1
index_sentiment_polarity	11.93	9.69	-39.38	5.78	11.91	17.78	72.78
index_subjectivity	44.34	11.67	0.00	39.62	45.35	50.83	100.00

2.2 Model estimation in various regimes

This subsection focuses on model formalization and estimation. Using nine variables introduced in the preceding subsection the formal specification of relation between the number shares (*shares*) and the explanatories is established:

$$shares = \beta_0 + \beta_1 n\_tokens\_title + \beta_2 n\_tokens\_content + \beta_3 num\_hrefs + \beta_4 num\_imgs + \beta_5 num\_videos + \beta_6 num\_hrefs + \beta_7 num\_keywords + \beta_8 is\_weekend + \beta_9 index\_sentiment\_polarity + \beta_{10} index\_subjectivity + \varepsilon, \quad \varepsilon \quad n(0, \sigma)$$

In line with approaches to estimating count data models presented in the lecture, four models are being estimated. Estimation using OLS is established as a baseline. Further, Poisson distribution is applied to account for specific characteristics of count data (discrete-valued, skewed to the right, only positive responses). Poisson with robust standard errors and quasi-Poisson are introduced to mitigate the over-dispersion of the data. Additionally, negative binomial regression model is estimated to provide an indicator of dispersion using theta estimator. Basically, testing significance of theta provides information on whether dispersion is a problem in the model.

For the sake of use of space in the document, model's estimations are split into two tables. Part 1 in Table 6 looks at the estimation of the define specification with use of OLS, Poisson distribution and Poisson with robust standard errors. Part 2 Table 7 compares results of estimation with use of OLS, quasi-Poisson and negative binominal.

Overall, interpretation of the coefficients is not straightforward for multiple reasons. First of all, OLS cannot be applied to the explanation of count response variable. Additionally, coefficients for all other models are not representing the association between independent and response variable the same way as OLS coefficient do due to the nonlinear nature of maximum likelihood method. To mitigate the issue with interpretation partial effects are computed.

Table 6: Estimation results for OLS, Poisson and Poisson with robust SE.

	Dependent variable:		
	shares		
	OLS	*Poisson*	
	(1)	(2)	(3)
Constant	673.838	7.307***	7.307***
	(433.909)	(0.001)	(0.119)
n_tokens_title	63.818**	0.019***	0.019**
	(27.742)	(0.00004)	(0.007)
n_tokens_content	-0.821***	-0.0002***	-0.0002***
	(0.141)	(0.00000)	(0.0001)
num_hrefs	38.791***	0.008***	0.008***
	(5.983)	(0.00001)	(0.001)
num_imgs	49.657***	0.012***	0.012***
	(7.749)	(0.00001)	(0.002)
num_videos	65.231***	0.014***	0.014***
	(14.513)	(0.00002)	(0.003)
num_keywords	95.161***	0.030***	0.030***
	(30.960)	(0.00005)	(0.008)
is_weekend	447.860***	0.124***	0.124***
	(173.675)	(0.0002)	(0.043)
index_sentiment_polarity	-9.644	-0.002***	-0.002
	(6.432)	(0.00001)	(0.002)
index_subjectivity	25.848***	0.008***	0.008***
	(5.429)	(0.00001)	(0.002)
Observations	39,644	39,644	39,644
R^2	0.005		
Adjusted R^2	0.005		
Log Likelihood		-127,157,995.000	-127,157,995.000
Akaike Inf. Crit.		254,316,010.000	254,316,010.000
Residual Std. Error	11,598.970 (df = 39634)		
F Statistic	22.280*** (df = 9; 39634)		
Note:			*p<0.1; **p<0.05; ***p<0.01

However, other insights can be derived from those tables. In terms of goodness-of-fit, all 4 models exhibit consistently rather a bad fit of the model to the data set. For OLS estimation R-squared is only 0.5%. For negative binominal and Poisson models both log likelihood and Akaike Information Criterion manifest rather bad fit - the lower the log likelihood the worse the fit and the higher the AIC the worse the fit.

Additionally, question of which model to finally choose to estimate arises. Basis for the answer is given by theta parameter in Table 7. Estimation yields results of theta = 0.926*** (0.006). Significant result supports applying the negative binominal model to combat over-dispersion in the data.

2.3 Partial effects' comparison

To accommodate for a use of a nonlinear method of maximum likelihood partial effects need to be computed for a model of choice in order to meaningfully interpret the results of estimation. In Table 8 both OLS estimates of coefficients and partial effects computed for negative binominal estimation of model are presented.

The effects of independent variables on the count response variables of shares of articles online are quite similar between OLS and negative binominal, though different - and that is the point. Partial effects can be interpreted as change in the mean counts of response due to a change in explanatory variable. Looking at the right-hand side column in Table 8 variables with the strongest effect on number of shares can be distinguished: *is_weekend, num_keywords, num_videos and n_tokens_title*. These are the parameters that stakeholders and managers in media industry should be particularly interested in.

For *n_tokens_title* an increase of one word in the title is associated with the shares of online news article increasing by 74.968. Very close to that effect is also *num_videos* where additional video in the news article increases its number of shares by 79.359. Strong marginal effect is also recognized for *num_keywords* as adding additional key phrase in the metadata results in 108.777 more shares of the article. Even five times larger is the effect of sharing the news article in the weekend. Table 8 contains two row in-between dashed-lines where

Table 7: Estimation results for OLS, quasi-Poisson and negative binominal.

	Dependent variable:		
	shares		
	OLS	*quasi-Poisson*	*negative binominal*
	(1)	(2)	(3)
Constant	673.838	7.307***	7.302***
	(433.909)	(0.130)	(0.039)
n_tokens_title	63.818**	0.019**	0.022***
	(27.742)	(0.008)	(0.002)
n_tokens_content	-0.821***	-0.0002***	-0.0002***
	(0.141)	(0.00004)	(0.00001)
num_hrefs	38.791***	0.008***	0.010***
	(5.983)	(0.001)	(0.001)
num_imgs	49.657***	0.012***	0.014***
	(7.749)	(0.002)	(0.001)
num_videos	65.231***	0.014***	0.023***
	(14.513)	(0.003)	(0.001)
num_keywords	95.161***	0.030***	0.032***
	(30.960)	(0.009)	(0.003)
is_weekend	447.860***	0.124***	0.142***
	(173.675)	(0.047)	(0.016)
index_sentiment_polarity	-9.644	-0.002	-0.002***
	(6.432)	(0.002)	(0.001)
index_subjectivity	25.848***	0.008***	0.005***
	(5.429)	(0.002)	(0.0005)
Observations	39,644	39,644	39,644
R^2	0.005		
Adjusted R^2	0.005		
Log Likelihood			-360,858.000
θ			0.926*** (0.006)
Akaike Inf. Crit.			721,735.900
Residual Std. Error	11,598.970 (df = 39634)		
F Statistic	22.280*** (df = 9; 39634)		
Note:			*p<0.1; **p<0.05; ***p<0.01

marginal effect computation for *is_weekend* is set up. Given that it is a dummy variable, mean count for both conditions of the dummy is calculated and the difference between those two expected values is the partial effect for the indicator. When news article is published in the weekend, its number of shares increases by 508.955. Interestingly enough, increasing the divergence of article's attitude away from the centre and towards a specified extreme leads to decrease in sharing of the article by -5.899 shares.

Table 8: Partial effects for the count model.

	OLS coefficients	Partial effects NegBin
(Intercept)	673.838	24,930.890
n_tokens_title	63.818	74.968
n_tokens_content	-0.821	-0.549
num_hrefs	38.791	32.445
num_imgs	49.657	46.763
num_videos	65.231	79.359
num_keywords	95.161	108.777
index_sentiment_polarity	-9.644	-5.899
index_subjectivity	25.848	18.372
when_weekend		*3,853.791*
when_not_weekend		*3,344.836*
is_weekend	447.860	508.955

3 Ordinal logistic data modeling: the Yelp case

The last section of this document deals with an analysis of the influence of certain characteristics of the review posted on the platform on the online rating given to the restaurant. There will be two regressands of interest: *review_stars* which indicates a number of stars awarded to the restaurant by the reviewer on the scale from 1 to 5; dummy *dFiveStars* which is an indicator of whether the reviewer awarded the restaurant with 5-star grade or not. Models estimated in this section aim to capture relevant factors contributing to restaurants' ratings in general and to very high, positive ratings of 5 stars.

3.1 Variables of choice and summary statistics

In order to introduce variables chosen as explanatories in this section, summary statistics of those are presented below in Table 9. Differently from preceding sections, this part will focus on a ordinal-choice and binary-choice models. Ordinal response variable is *review_stars* - ratings from 1 to 5, where 1 indicates the worse rating and 5 indicates the best rating. Overall, reviews in the dataset are grading the restaurants rather positively, with an average rating of 3.71 (above half 2.50) and first quartile as high as 3. The latter indicates that 75% of the reviews award rating higher than 3. Binary response variable is *dFiveStars*. The proportion of restaurant receiving 5-star rating in the reviews from data set is roughly 70:30 in favor of ratings lower than 5.

Table 9: Summary statistics for the ordinal model.

Statistic	Mean	St. Dev.	Min	Pctl(25)	Median	Pctl(75)	Max
review_stars	3.71	1.18	1	3	4	5	5
dFiveStars	0.29	0.46	0	0	0	1	1
travel	0.10	0.30	0	0	0	0	1
reviews_in_city_so_far	41.76	103.77	0	2	9	37	1,601
numb_friends	280.98	634.74	0	14	77	243	6,093
length	752.27	652.33	1	304	559	988	5,000
yelping_for_months	138.57	24.96	69	122	140	157	216

Variable *travel* is dichotomous and indicates whether the review was published whilst travelling or not. Most of the reviews are published in hometown given the mean 0.10. *reviews_in_city_so_far* counts the number of reviews already published in a given city of the review. Variable *numb_friends* counts the number of friends on the platform. Distribution of this variable is skewed to the right, most likely due to a few extreme observations given max[6,093]. The length of the review in characters is measured by *length* - on average reviews have 752.27 characters, whereas 50% of users wrote reviews with roughly 560 characters and less which indicates skewness of the distribution prompted by a maximum length of the review at 5,000 characters. Variable *yelping_for_months* is author's construct which yields the number of months the user has been active on the platform. This is a symmetrically distributed variable with users ranging from 69 to 216 months old at average of roughly 139 months.

3.2 Formal specification of the model

This section will formalize specifications of the relation between two independent variables (*review_stars*, *dFiveStars*) and five explanatory variables outlined in the preceding section. The subsequent section will make use of those specifications. Both models are presented below:

$$review\_stars = \beta_0 + \beta_1 travel + \beta_2 reviews\_in\_the\_city\_so\_far + (\beta_{31} + \beta_{32} travel) length + \beta_4 numb\_friends + \beta_5 yelping\_for\_months + \varepsilon, \quad \varepsilon \quad n(0, \sigma)$$

$$dFiveStars = \beta_0 + \beta_1 travel + \beta_2 reviews\_in\_the\_city\_so\_far + (\beta_{31} + \beta_{32} travel) length + \beta_4 numb\_friends + \beta_5 yelping\_for\_months + \varepsilon, \quad \varepsilon \quad n(0, \sigma)$$

3.3 Models estimation and key insights

3.3.1 Uninterpretable linear estimates

Estimation of binary-choice and ordered-response regression models with OLS is, once again, not suitable given the nature of dependent variables. Nonlinear logistic and probit analyses are applied which render the interpretation of results to be not a straightforward linear interpretation of marginal effects. That is why the following subsections estimate partial effects at the average and average partial effects to analyze the influence of predictors on regressands in pair with goodness-of-fit measures.

3.3.2 Partial effects for binary-choice model

For a binary-choice model with *dFiveStars* as a response variable partial effects at the average (PEA) and average partial effects (APE) are computed. This document utilizes maximum-likelihood estimation for this model using logistic and probit regression analyses. Estimation of marginal effects yields results that can be interpreted as a degree of change in probability of awarding restaurant with 5-star rating by a reviewer due to a small increase in the explanatory variables.

Table 10: Partial effects (PEA, APE) for a binary-choice model.

	PEA Logit	PEA Probit	APE Logit	APE Probit
(Intercept)	0.091	0.153	0.091	0.152
travel	-0.012	0.002	-0.012	0.002
numb_friends	0.00001	0.00001	0.00001	0.00001
length	-0.0001	-0.0001	-0.0001	-0.0001
yelping_for_months	-0.002	-0.003	-0.002	-0.003
reviews_in_city_so_far	-0.0004	-0.001	-0.0004	-0.001
travel:reviews_in_city_so_far	-0.002	-0.003	-0.002	-0.003

What can be noted in Table 10 is that PEA and APE are very similar to each other for logit and probit regressions. Among the results it is interesting to highlight that reviews submitted while travelling increase chances of giving a 5-star rating by 0.2% in probit whilst in logit reviewing restaurant when travelling results in decreased probability of giving the highest rating by 1.2%. Including results of the interaction between travel indicator and number of reviews in the given city thus far result in strengthening the negative relationship with chances of giving maximum rating in logit analysis, whereas in probit regression analysis the interaction term in fact inverses aforementioned positive relationship and overall causes decrease of chances of giving a 5-star in a review by 0.1%.

Number of friends on the platform and length of the review appear to have very limited influence on changes of giving a 5-star rating and are consistent in the direction of that probability change across models. Partial effects also indicate that the longer a user is on the platform, the lower the probability of giving a very positive review by 0.2% for logit and 0.3% for probit estimations. Additionally, albeit with a limited impact, having already submitted more reviews in a certain city yields slightly lower probability of an enthusiastic 5-star review.

3.3.3 Goodness-of-fit for binary response model

Below in Table 11 pseudo R-squared measures are presented yielding inconsistent results for binary-choice model. This is a testament to a peculiar nature of those calculations as unequivocal measures of goodness-of-fit cannot be obtained. McFadden resembles R-squared by looking at the improvement of fitted model over null model. Adjusted measure penalizes for using additional independent variables. Cox and Snell build on top of the McFadden measure by taking N-th root to extract the contribution of each observation. Nagelkerke divides Cox and Snell's measure by its maximum to examine a relative explanatory power. Efron defines the pseudo R-squared as percentage of explained variation modified with squared correlation between predicted and actual dependent values. Count R-squared is interested in accuracy of the model, so number of correct classifications is compared in a ratio with the frequency of the dominant outcome.

Table 11: Pseudo R-squared measures computed for binary-choice model.

	pseudo R-squared
McFadden R-squared	0.025
McFadden R-squared adjusted	0.025
Cox & Snell R-squared	*-Inf*
Nagelkerke R-squared	*-Inf*
Efron R-squared	0.032
Count R-squared	0.708
Count R-squared adjusted	0.002

As for the results, it is worth noting that due to exponentiation of a very low number in the denominator, both Cox & Snell and Nagelkerke R-squared ends up with a 0 / 0 division in the formula in software. To underline this occurrences, both measures have been displayed as *-Inf* to be clear on reasons for not presenting the actual result.

It is interesting to note that even though the pseudo-measures do not agree fully on the degree of goodness-of-fit, McFadden's and Efron's measures are similar enough to agree on a message they convey - the fit of the

model is rather low, around 2-3%. Count R-squared is to be taken with a grain of salt due to the low score on an adjusted version of the measure - it means that even though data has some common data points that could be predicted with null model quite well, adjusting for those common outcomes in comparison to baseline model yields a rather bad fit of 0.2%.

3.3.4 Partial effects for ordinal-choice model

For an ordinal-choice model estimated by means of the maximum likelihood method with *review_stars* as a count response variable average partial effects (APE) are computed. Once again calculations are done for both logistic and probit regression analyses. There are as many partial effects per a regressor as there are many outcomes of the ordinal dependent variable. Partial effects give insight on changes in probability of being in a certain ordinal response interval given a small change in explanatory variable.

In order to provide a concise summary, both logit- and probit-based marginal effects are presented in Table 12. Overall, for the ordinal response model all variables but *travel* yield small and uninteresting results. Moreover, changes in the probability of having a given realization of dependent variable P(Y = y) are very consistent between logit and probit effects across all regressors.

Table 12: Partial effects (APE) for an ordinal-choice model.

	travel	numb.friends	length	yelping...months	reviews_in_city...	travel:reviews_in_city...
P(y=1)L	-0.008	-0.00001	0.00002	0.0003	0.00003	0.00003
P(y=2)L	-0.010	-0.00001	0.00002	0.0003	0.00004	0.00003
P(y=3)L	-0.012	-0.00001	0.00003	0.0004	0.00005	0.00004
P(y=4)L	0.003	0.00000	-0.00001	-0.0001	-0.00001	-0.00001
P(y=5)L	0.027	0.00002	-0.0001	-0.001	-0.0001	-0.0001
P(y=1)P	-0.007	-0.00001	0.00002	0.0003	0.00004	-0.00004
P(y=2)P	-0.006	-0.00001	0.00002	0.0003	0.00003	-0.00004
P(y=3)P	-0.006	-0.00001	0.00002	0.0003	0.00004	-0.00004
P(y=4)P	0.001	0.00000	-0.00000	-0.0001	-0.00001	0.00001
P(y=5)P	0.018	0.00003	-0.0001	-0.001	-0.0001	0.0001

It is worth looking closely at the result for the indicator of travel. Reviews posted when travelling decrease the probability of being either 1-, 2-, or 3-stars reviews by 0.8%-1.2%. At the same time, reviewing a restaurant whilst travelling increases the probability of a rating being 4 by 0.3% for logit and 0.1% for probit, but more importantly - while travelling the probability of giving a 5-star review increases by as much as 2.7% for logit analysis and 1.8% for probit. This underlines the importance of travel indicator for stakeholders willing to use this model.

A Appendix - code

```
#-------------------------------------------------------------------
# BM01BAM session 05: examples maximum likelihood estimation
#-------------------------------------------------------------------

# Clean the global environment
remove(list=ls())

#-----------------------------------------------------------
# Load libraries
#-----------------------------------------------------------

# install.packages("ggplot2", dependencies = TRUE)
library(ggplot2)

# install.packages("RColorBrewer", dependencies = TRUE)
library(RColorBrewer)

# install.packages("psych", dependencies = TRUE)
library(psych)

# install.packages("stargazer", dependencies = TRUE)
library(stargazer)

# Install and load the ROCR package
# install.packages("ROCR", dependencies = TRUE)
library(ROCR)


#-----------------------------------------------------------
# Make a function that calculates accuracy measures based
# on observed binary target and estimated probabilities of
# a succesful outcome; the threshold can be set as prefered
# --- not needed for class
#-----------------------------------------------------------

# Function 'myAccuracy' calculates the four accuracy measures
# based on the observed target value and the predicted class
# probability; a threshold tau can be specified to manipulate
# the classification (default 0.5)
myAccuracy <- function(y, f, tau = 0.5) {

  # y : observed value (binary), either a factor or numeric. If
  #     numeric, then (0,1) is assumed and converted to factor
  # f : predicted classification probability, converted to factor
  #     with the same levels as y

  # Convert y to factor, if necessary
  if (class(y) == "numeric" | class(y) == "integer") {
    y <- factor(y, levels = c(0,1))
  }

  # Convert f to factor with the same levels as y, and
  # irrespective of whether f is a class probability or
  # a predicted class
  if (class(f) == "numeric") {
    f <- factor(as.numeric(f > tau), levels=c(0,1), labels=levels(y))
  } else f <- factor(f, labels=labels(y))

  # Make a classification table
  tblConfusion <- table(Predicted = f, Observed = y)

  # Identify classifications
  TP <- tblConfusion[2,2]
  FN <- tblConfusion[1,2]
  TN <- tblConfusion[1,1]
  FP <- tblConfusion[2,1]

  # Measure performance
  prfMeasures <- c(
    Accuracy    = (TP+TN)/sum(tblConfusion),
    Sensitivity = TP/(TP + FN),
    Specificity = TN/(FP + TN),
    Precision   = TP/(FP + TP)
  )

  # Return the results
  return(list(tblConfusion, prfMeasures))
}

# myAccuracy(dfMash.TestEveryPop, predLogit)


#-----------------------------------------------------------
# Define paths
#-----------------------------------------------------------

setwd("C:/STUDIA - materia␣ly/RSM-MASTERS/COURSES/BLOCK 1/Advanced Statistics and Programming/Session 6/Tutorial 5")

dir <- "C:/STUDIA - materia␣ly/RSM-MASTERS/COURSES/BLOCK 1/Advanced Statistics and Programming/Session 6/Tutorial 5/"

dirData <- paste0(dir, "Data/")
dirProg <- paste0(dir, "Programs/")
dirRslt <- paste0(dir, "Results/")


#-----------------------------------------------------------
# Read the flight taxation data from csv file
#-----------------------------------------------------------
dsFlightTax <- read.csv2(file=paste0(dirData,"FlightTaxation.csv"),
                         stringsAsFactors=FALSE)
head(dsFlightTax)


#-----------------------------------------------------------
# Select observations that meet manipulation checks for
# destination and flight tax (the third destination, flight
# information, was hardly picked up)
#-----------------------------------------------------------

# Define the criterion
meetCriterion <-
  (dsFlightTax$ManipDest == dsFlightTax$CheckDest) &
  (dsFlightTax$ManipTax == dsFlightTax$CheckTax)

# Implement the criterion
dsFlightTax <- dsFlightTax[meetCriterion,]


#-----------------------------------------------------------
# Make new variables (for the example; not for class)
#-----------------------------------------------------------
```

```r
# Define ecological sense of guilt
myGuilt <- c("Guilt01", "Guilt02", "Guilt03", "Guilt04", "Guilt05")
dsFlightTax$Guilt <-
  alpha(dsFlightTax[myGuilt],
        keys=c("Guilt03", "Guilt04", "Guilt05"),
        cumulative = FALSE)$scores

# Define dependent variable dAirplane, which indicates the
# likelihood to accept the travel by train offer, and
# independent variable dNearAir, which indicates train
# distance from Schiphol of less than 45 minutes
dsFlightTax$dAirplane <- as.numeric(dsFlightTax$rateAirplane > 80)
dsFlightTax$dNearAir  <- as.numeric(dsFlightTax$SchipholTrain <= 2)


#-------------------------------------------------------------
# Maximum likelihood estimation of probability to accept the
# travel offer by airplane -- straightformard example of
# using ML to estimate the parameter of a distribution
#-------------------------------------------------------------

# View of the data
dsFlightTax$dAirplane

# Maxmum likelihood estimate of probability to
# accept the offer
round(mean(dsFlightTax$dAirplane), 3)


#-------------------------------------------------------------
# Illustrate the probability densities of the logistic and
# normal distributions  -- used as an example to clarify the
# notions of probability density and cumulative distribution,
# and demonstrating the difference between the logistic and
# normal densities
#-------------------------------------------------------------

# Probability density function
ggplot(data = data.frame(x = 0), aes(x = x)) +
  stat_function(fun = dlogis, colour="Blue4", lwd=1) + # logit
  stat_function(fun = dnorm,  colour="Red3", lwd=1) +  # probit
  xlim(-3,3) + ylab("probability density") +
  # Add labels for the two functions
  annotate("text", label = "logistic", fontface="italic",
           x = 2.25, y = 0.15, size = 6, colour = "Blue4") +
  annotate("text", label = "normal", fontface="italic",
           x = 1.25, y = 0.35, size = 6, colour = "Red3") +
  theme(axis.text  = element_text(size=rel(1.2)),
        axis.title = element_text(size=rel(1.2)))
ggsave(paste0(dirRslt,"Session05LogisticNormalDensities01.pdf"))

# Cumulative density function
ggplot(data = data.frame(x = 0), aes(x = x)) +
  stat_function(fun = plogis, colour="Blue4", lwd=1) + # logit
  stat_function(fun = pnorm,  colour="Red3", lwd=1) +  # probit
  xlim(-3,3) + ylab("cumulative distribution") +
  # Add labels for the two functions
  annotate("text", label = "logistic", fontface="italic",
           x = 2.0, y = 0.75, size = 6, colour = "Blue4") +
  annotate("text", label = "normal", fontface="italic",
           x = 0.2, y = 0.85, size = 6, colour = "Red3") +
  theme(axis.text  = element_text(size=rel(1.2)),
        axis.title = element_text(size=rel(1.2)))
ggsave(paste0(dirRslt,"Session05LogisticNormalDensities02.pdf"))


1250*5
#-------------------------------------------------------------
# Perform analyses of the acceptance of a travel offer
# involving a city trip in Europe by airplane
#-------------------------------------------------------------

# Subsequent analysis is based on a numeric dependent
# variable, even though a binary categorical variable
# (factor) might be expected. However, the outcomes
# are the same regardless of whether the target is
# defined as numeric or factor.

# Define the model
mdlA <- dAirplane ~ Guilt + ImportPrice + dNearAir

# Estimate the model
rsltOLS    <-  lm(mdlA, data = dsFlightTax)
rsltLogit  <- glm(mdlA, data = dsFlightTax,
                  family=binomial(link = "logit"))
rsltProbit <- glm(mdlA, data = dsFlightTax,
                  family=binomial(link = "probit"))

# Summarise the results
summary(rsltLogit)

# Make table of the results
stargazer(rsltOLS, rsltLogit, rsltProbit,
          align = TRUE, no.space = TRUE,
          intercept.bottom = FALSE, type="text")

# odds ratio

stargazer(exp(rsltLogit$coefficients), type="text")


#-------------------------------------------------------------
# Estimated parameters cannot be directly compared between
# the three models and do not have an immediate interpre-
# tation; the concept of partial (marginal) effects is
# needed to explore the implications of the estimated
# effects for the acceptance probability -- note that an
# adjustment is needed and implemented for the effects of
# dummy variables, as the idea of 'partial' does not apply
# in this case
#-------------------------------------------------------------


#------------------
# Preparation
#------------------

# Collect estimated coefficients, and make subsets of the
# data frame (for convenience); data frame dsFlightSub
# contains observations of the exlanatory, and data frame
# dsFlightAvg the averages of these variables
betaLogit  <- coefficients(rsltLogit)
betaProbit <- coefficients(rsltProbit)

myVar       <- all.vars(mdlA) # picking up all variables
```

```r
dsFlightSub <- dsFlightTax[myVar[-1]] # Exclude explained variables as we do not its average
dsFlightAvg <- as.data.frame(t(colMeans(dsFlightSub))) # for partial effect at average we need averages of explanatories

# dsFlightSub <- cbind(`(Intercept)` = 1, dsFlightSub)

stargazer(dsFlightTax, type="text")
#------------------
# PEA: partial effect at the average
#------------------

# Determine partial effects at the average for all
# explanatory variables, discarding their measurement
# levels

# For non-dummy variables --- Predict (type="link") gives avg-x'beta, function dlogis/dnorm
# calculates the density f(avg-x'beta)
PEAlogit.1 <- dlogis(predict(rsltLogit, newdata=dsFlightAvg,
                             type="link"))*betaLogit
PEAprobit.1<- dnorm(predict(rsltProbit, newdata=dsFlightAvg,
                            type="link"))*betaProbit

# The partial effect for dummy variables is a simple difference
# between success probabilities if the dummy event does occur
# and does not occur; in the example this only needs to be done
# for variable dNearby
tmp <- dsFlightAvg

# Predict (type="response") gives P(Y=1|avg-x'beta)
tmp$dNearAir <- 0

tmpPEAlogit.0   <-
  predict(rsltLogit, newdata=tmp, type="response")
tmpPEAprobit.0  <-
  predict(rsltProbit, newdata=tmp,type="response")

tmp$dNearAir <- 1

tmpPEAlogit.1   <-
  predict(rsltLogit, newdata=tmp, type="response")
tmpPEAprobit.1  <-
  predict(rsltProbit, newdata=tmp,type="response")

# Difference between success of response variable when dummy is 1 and success of Y when dummy is 0
# For both models; this gives a partial effect at the average for dummy variable
PEAlogit.2 <- PEAlogit.1
PEAlogit.2["dNearAir"] <- tmpPEAlogit.1 - tmpPEAlogit.0

PEAprobit.2 <- PEAprobit.1
PEAprobit.2["dNearAir"] <- tmpPEAprobit.1 - tmpPEAprobit.0


#------------------
# APE: average partial effects
#------------------

# Determine average partial effects for all explanatory
# variables regardless of the measurement levels

# Predict (type="link") gives avg-x'beta, function dlogis/dnorm
# calculates the density f(avg-x'beta); different from PEA, these
# predictions are made for individual observations, which are
# averaged with function mean  - note that the estimates beta's
# are the same for each observation and are therefore left outside
# the averaging
APElogit.1   <- mean(dlogis(predict(rsltLogit, type="link")))*betaLogit
APEprobit.1  <- mean(dnorm(predict(rsltProbit, type="link")))*betaProbit

# The average partial effect for dummy variables is the average
# of the differences between success probabilities if the dummy
# event does occur and does not occur for each observations; in
# the example this only needs to be done for variable dNearby
tmp             <- dsFlightSub

# Predict (type="response") gives P(Y=1|avg-x'beta)
tmp$dNearAir  <- 0

tmpAPElogit.0 <-
  mean(predict(rsltLogit, newdata=tmp, type="response"))
tmpAPEprobit.0<-
  mean(predict(rsltProbit,newdata=tmp, type="response"))

tmp$dNearAir  <- 1

tmpAPElogit.1 <-
  mean(predict(rsltLogit, newdata=tmp, type="response"))
tmpAPEprobit.1<-
  mean(predict(rsltProbit,newdata=tmp, type="response"))

APElogit.2 <- APElogit.1
APElogit.2["dNearAir"] <- tmpAPElogit.1 - tmpAPElogit.0

APEprobit.2 <- APEprobit.1
APEprobit.2["dNearAir"]<- tmpAPEprobit.1 -tmpAPEprobit.0


# Present the results
stargazer(round(cbind(PEAlogit.1, PEAlogit.2,PEAprobit.1,PEAprobit.2), 4), type="text")
stargazer(round(cbind(APElogit.1, APElogit.2,APEprobit.1,APEprobit.2), 4), type="text")


#------------------
# Compare the effect on the offer acceptance probability of
# dNearby for different values of ImportPrice
#------------------

# An artificial data frame is collected based on the existing
# dsFlightSub, for convenience; ImportPrice has values evenly
# spread over its empirical range, Guilt is set to its sample
# average, dNearby alternately has values 0 and 1
tmp             <- dsFlightSub[1:100,]
tmp$ImportPrice <- seq(1, 100, length.out = nrow(tmp))
tmp$Guilt       <- dsFlightAvg$Guilt

tmp$dNearAir     <- 0
tmp$probdNearby0 <- predict.glm(rsltLogit, newdata = tmp, type="response")

tmp$dNearAir     <- 1
tmp$probdNearby1 <- predict.glm(rsltLogit, newdata = tmp, type="response")

# Offer acceptance probability against ImportPrice, for dNearby
# equal to 0 and 1 respectively
ggplot(data = tmp, aes(x = ImportPrice)) +
  geom_line(aes(y=probdNearby0), colour="Blue4", lwd=1) +
  geom_line(aes(y=probdNearby1), colour="Red3", lwd=1) +
  ylab("Prob(Y=1)") +
```

```
  # Add labels for the two functions
  annotate("text", label = "dNearby = 1", fontface="italic",
           x = 50.0, y = 0.7, size = 6, colour = "Red3") +
  annotate("text", label = "dNearby = 0", fontface="italic",
           x = 75.0, y = 0.5, size = 6, colour = "Blue4") +
  theme(axis.text  = element_text(size=rel(1.2)),
        axis.title = element_text(size=rel(1.2)))
ggsave(paste0(dirRslt,"Session05SimulatedEffectNearby01.pdf"))


#------------------
# Compare the effect on the offer acceptance probability of
# dNearby for different values of Guilt
#------------------

# An artificial data frame is collected based on the existing
# dsFlightSub, for convenience; Guilt has values evenly spread
# over its empirical range, ImportPrice is set to its sample
# average, dNearby alternately has values 0 and 1
tmp             <- dsFlightSub[1:100,]
tmp$Guilt       <- seq(1, 5, length.out = nrow(tmp))
tmp$ImportPrice <- dsFlightAvg$ImportPrice

tmp$dNearAir     <- 0
tmp$probdNearby0 <- predict.glm(rsltLogit, newdata = tmp, type="response")

tmp$dNearAir     <- 1
tmp$probdNearby1 <- predict.glm(rsltLogit, newdata = tmp, type="response")

# Offer acceptance probability against Guilt, for dNearby
# equal to 0 and 1 respectively
ggplot(data = tmp, aes(x = Guilt)) +
  geom_line(aes(y=probdNearby0), colour="Blue4", lwd=1) +
  geom_line(aes(y=probdNearby1), colour="Red3", lwd=1) +
  ylab("Prob(Y=1)") +
  # Add labels for the two functions
  annotate("text", label = "dNearby = 1", fontface="italic",
           x = 3.5, y = 0.6, size = 6, colour = "Red3") +
  annotate("text", label = "dNearby = 0", fontface="italic",
           x = 2.0, y = 0.5, size = 6, colour = "Blue4") +
  theme(axis.text  = element_text(size=rel(1.2)),
        axis.title = element_text(size=rel(1.2)))
ggsave(paste0(dirRslt,"Session05SimulatedEffectNearby02.pdf"))



#------------------------------------------------------------
# Goodness of fit and hypothesis tests
#------------------------------------------------------------

# Maximum likelihood estimation does not come with straight-
# forward goodness of fit measures, like R2 in OLS. Also,
# there is no agreement about a single fit measure. Several
# have been developed over time. Here, two popular ones are
# presented

# Interpretation R2 in OLS: (i) percentage explained variation
# (or 1 minus percentage unexplained variation); (ii) improvement
# of fitted model with respect to null model (say, the sample
# average); (iii) squared correlation of predicted and actual
# values

#------------------
# Collect relevant info
#------------------

# Check contents of the glm object
str(rsltLogit)

# Log-likelihood values
# it looks at how the model performed by looking at the predicted probabilities vs. actual outcome
# it sums the probabilities of predicted values and actual outcomes times the log of probability
# large negative values would indicate that the model is doing poor job

# calculation below is actually the inverse of log-likelihood because we only have deviance (2LL) in the model
lnL.fitted <- -0.5*rsltLogit$deviance
lnL.null   <- -0.5*rsltLogit$null.deviance

lnL.fitted
lnL.null

# Degrees of freedom of both models
df.fitted  <- rsltLogit$df.residual
df.null    <- rsltLogit$df.null

df.fitted
df.null

# Number of predictors and sample size
K <- df.null - df.fitted
N <- df.null + 1

# Predicted and observed value of the target
probLogit <- predict.glm(rsltLogit, type = "response")
yvalue    <- dsFlightTax$dAirplane


#------------------
# Define pseudo R2 measures
#------------------
# 1. Pseudo R2 and adjusted pseudo R2, McFadden. Resembles
# R2 defined as percentage explained variation and as
# improvement of fitted model over null model. The adjusted
# version penalizes extra predictor variables. The closer
# to 1, the larger the improvement of the model against the naive null model
McFadden.R2    <- 1 - (lnL.fitted/lnL.null)
McFadden.R2adj <- 1 - ((lnL.fitted - K)/lnL.null)

# 2. Cox & Snell. Resembles R2 as improvement of fitted
# model over the null model. Taking N-th root yields the
# contribution of each observation (it is based on the
# LRT statistic, which explains the '2'). Note that the
# maximum is smaller than 1.
CoxSnell.R2 <- 1 - (exp(lnL.null)/exp(lnL.fitted))^(2/N)

# 3. Nagelkerke/Crag and Uhle. Resembles R2 as improvement
# of fitted model over the null model. SImilar to Cox and
# Snell's pseudo R2, but divided by the latter's maximum.
Nagelkerke.R2 <- CoxSnell.R2/(1 - exp(lnL.null)^(2/N))

# 4. Efron. Resembles R2 defined as percentage explained
# variation and as squared correlation between predicted
# and actual dependent values
Efron.R2 <- 1 -
  sum((yvalue-probLogit)^2)/sum((yvalue-mean(yvalue))^2)
```

```r
# 5. Count R2 and adjusted count R2. Percentage of correct
# classification (Accuracy). The adjusted version compares
# the number of correct classifications to the frequency of
# the dominant outcome.
# It counts how often the model predicts the right outcome
Count.R2 <- sum(yvalue == (probLogit>0.5))/length(yvalue)

maxFreq     <- max(table(yvalue))
Count.R2adj <- (sum(yvalue == (probLogit>0.5)) - maxFreq)/
  (length(yvalue) - maxFreq)

t(round(
  cbind(McFadden.R2, McFadden.R2adj, CoxSnell.R2,
        Nagelkerke.R2, Efron.R2,
        Count.R2, Count.R2adj), 3))


#------------------
# Individual effects
#------------------
summary(rsltLogit)

#------------------
# Perform likelihood ratio test
#------------------

# Test of joint contribution of Guilt and dNearby in the
# logit and OLS regression models. For the logit model, the
# LRT statistic is equal to -2 times the difference between
# the log-likelihoods of the restricted and unrestricted
# models. Under H0: no effects of Guilt and dNearby, the LRT
# statistic is chi-square distributed
anova(update(rsltLogit, . ~ . - Guilt - dNearAir),
      rsltLogit, test="Chisq")
anova(update(rsltOLS, . ~ . - Guilt - dNearAir),
      rsltOLS)



#-------------------------------------------------------------
# Assess predictive performance of the estimated models
#-------------------------------------------------------------

# The instructions below prepare for the examples in the
# slides

# Find predicted success probabilities for the logit model;
# type = "response" gives the probabilities
probLogit <- predict.glm(rsltLogit, type="response")

#predLogit <- predict.glm(rsltLogit, type="response")


# Convert probabilities to class predictions using different
# values of the threshold
predLogit.25 <- as.numeric(probLogit > 0.25)
predLogit.50 <- as.numeric(probLogit > 0.50)
predLogit.75 <- as.numeric(probLogit > 0.75)

# Show first five lines with observed and predicted values
# of the dependent variable
head(cbind(Observed  = dsFlightTax$dAirplane,
           Predicted = probLogit,
           predLogit.25, predLogit.50, predLogit.75))

# Make confusion matrix for the observed and predicted
# (tau=0.5) values of the dependent variables; note that
# outcome `1` is considered a success, and outcome `0`
# a failure
table(Predicted = predLogit.50,
      Observed = dsFlightTax$dAirplane)

# Calculate the performance measures
tbl <- table(Predicted = predLogit.50,
             Observed = dsFlightTax$dAirplane)
tbl

TP <- tbl[2, 2] # True positive
TN <- tbl[1, 1] # True negative
FP <- tbl[2, 1] # False positive
FN <- tbl[1, 2] # False negative

Accuracy    <- (TP+TN)/sum(tbl)
Sensitivity <- TP/(TP + FN)
Specificity <- TN/(FP + TN)
Precision   <- TP/(FP+TP)

round(
  cbind(Accuracy, Sensitivity, Specificity, Precision), 3)

# Use the self-made function
myAccuracy(dsFlightTax$dAirplane, probLogit)


#-------------------------------------------------------------
# ROC curves: basic approach based on logit regression
# results
#-------------------------------------------------------------

# The construction of ROC curves involves a number of steps
# which are conveniently numbered 1 to 5

# Step 1: identify the observed value of the dependent
yvalue <-dsFlightTax$dAirplane

# Step 2: collect model predictions  with predict function
probLogit  <- predict(rsltLogit, type = c("response"))

# Step 3: determine predictive performance measures using
# ROCR's prediction function
prd.Logit  <- prediction(probLogit, yvalue)

# Step 4: prepare for the ROC plots with ROCR's
# performance function
prf.Logit <-
  performance(prd.Logit, measure = "tpr", x.measure = "fpr")

# Step 5: make the plot (combined in one graph)
pdf(paste0(dirRslt,"Session05ROCcurves01.pdf"))

plot(prf.Logit, lty = 1, lwd = 2.0, col = "Red3")
abline(a = 0, b = 1, lty = 3, lwd = 1.5)

dev.off()
```

```
#-------------------------------------------------------------
# ROC curves: same, but comparing logit, probit and
# linear regression model performance
#-------------------------------------------------------------

# Step 1: identify the observed value of the dependent
yvalue <-dsFlightTax$dAirplane

# Step 2: collect model predictions  with predict function
probLogit  <- predict(rsltLogit,  type = c("response"))
probProbit <- predict(rsltProbit, type = c("response"))
probOLS    <- predict(rsltOLS)

# Step 3: determine predictive performance measures using
# ROCR's prediction function
prd.Logit  <- prediction(probLogit,  yvalue)
prd.Probit <- prediction(probProbit, yvalue)
prd.OLS    <- prediction(probOLS,    yvalue)

# Step 4: prepare for the ROC plots with ROCR's
# performance function
prf.Logit <-
  performance(prd.Logit, measure = "tpr", x.measure = "fpr")
prf.Probit <-
  performance(prd.Probit,measure = "tpr", x.measure = "fpr")
prf.OLS <-
  performance(prd.OLS,   measure = "tpr", x.measure = "fpr")

# Step 5: make the plot (combined in single graph)
pdf(paste0(dirRslt,"Session05ROCcurves02.pdf"))

plot(prf.Logit, lty=1, lwd=2.0, col="Red3")
plot(prf.Probit,lty=1, lwd=2.0, col="Blue4",  add=TRUE)
plot(prf.OLS,   lty=1, lwd=2.0, col="Orange2",add=TRUE)

abline(a = 0, b = 1, lty = 3, lwd = 1.5)

legend(0.6,0.5, c("Logistic regression",
                  "Probit regression",
                  "OLS regression"),
       col = c("Red3", "Blue4", "Orange2"), lwd=3)

#dev.off()



#-------------------------------------------------------------
# ROC curves: Area under the curve
#-------------------------------------------------------------

# Check contents of the performance object
str(performance(prd.Logit, measure = "auc"))

performance(prd.Logit, measure = "auc")@y.name
performance(prd.Logit, measure = "auc")@y.values

# Combine the AUC values of the three models in a single
# data frame
tmpAuc <- rbind(
  data.frame(Model="Logistic regression",
             Auc  = performance(prd.Logit,
                        measure="auc")@y.values[[1]]),
  data.frame(Model="Probit regression",
             Auc  = performance(prd.Probit,
                        measure="auc")@y.values[[1]]),
  data.frame(Model="OLS regression",
             Auc  = performance(prd.OLS,
                        measure="auc")@y.values[[1]])
  )

# Summarize the results in a nice looking table
stargazer(tmpAuc, summary = FALSE,
          align = TRUE, rownames = FALSE, type="text")
```