Qualitative Variablen
Variablenselektion

Peter Büchel

HSLU I

Stoc: Block 13

Qualitative erklärende Variablen

- Bisher angenommen: Alle Variablen *quantitativ* in linearem Regressionssystem
- Aber: Oft sind einige erklärenden Variablen *qualitativ*

Beispiel

- Datensatz `Credit` wurde in den USA erhoben
- Enthält für eine grössere Anzahl Individuen:
  - `balance` (monatliche Kreditkartenrechnung): Zielgrösse, quantitativ
  - `age` (Alter): erklärend, quantitativ
  - `cards` (Anzahl Kreditkarten): erklärend, quantitativ
  - `education` (Anzahl Jahre Ausbildung): erklärend, quantitativ
  - `income` (Einkommen in Tausenden Dollars): erklärend, quantitativ
  - `limit` (Kreditkartenlimite): erklärend, quantitativ
  - `rating` (Kreditwürdigkeit): erklärend, quantitativ

- Datensatz:

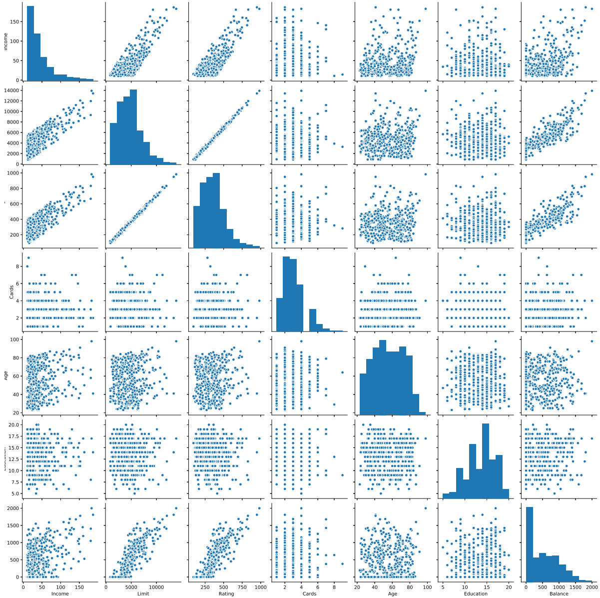
```
import pandas as pd
import statsmodels.api as sm
from statsmodels.formula.api import ols
import numpy as np

df = pd.read_csv("../Data/Credit.csv").drop("Unnamed: 0", axis=1)

df.head()
```

```
##      Income  Limit  Rating  Cards  ...  Student  Married  Ethnicity Balance
## 0    14.891   3606     283      2  ...       No      Yes  Caucasian     333
## 1   106.025   6645     483      3  ...      Yes      Yes      Asian     903
## 2   104.593   7075     514      4  ...       No       No      Asian     580
## 3   148.924   9504     681      3  ...       No       No      Asian     964
## 4    55.882   4897     357      2  ...       No      Yes  Caucasian     331
##
## [5 rows x 11 columns]
```

- Abbildung:



- Code:

```
import seaborn as sb

sb.pairplot(df)
```

- Streudiagramme von Paaren von Variablen: Identität gegeben durch entsprechenden Spalten- und Zeilenkennzeichnungen
- Plot direkt rechts des Wortes „Balance": Streudiagramm der Variablen `age` und `balance`
- Streudiagramme:
  - `age` - `balance`: Kein Zusammenhang
  - `Education` - `balance`: Kein Zusammenhang
  - `Income` - `balance`: Schwacher Zusammenhang
  - `Limit` - `balance`: Starker Zusammenhang

- Neben quantitativen noch vier erklärende qualitative Variablen:
  - `gender` (Geschlecht)
  - `student` (Studentenstatus)
  - `ethnicity` (Kaukasier, Afroamerikaner, Asiat)
- Qualitativ erklärende Variablen heissen auch *Faktoren*
- Faktoren nehmen *Stufen* oder *Levels* an:
  - `gender`: male, female
  - `student`: ja, nein
  - `ethnicity`: Kaukasier, Afroamerikaner, Asiat

Qualitative erklärende Variable mit nur zwei Levels

- Beispiel `balance`: Unterschied zwischen Männern und Frauen
- Andere Variablen werden für den Moment ignoriert
- Qualitative erklärende Variable mit zwei *Levels* (mögliche Werte): Hinzunahme dieser Variable in Regressionsmodell sehr einfach
- Führen Indikatorvariable (oder *Dummy-Variable*) ein, die nur zwei mögliche numerische Werte annehmen kann

Beispiel

- Für `gender`:

$$x_i = \begin{cases} 1 & \text{falls } i\text{-te Person weiblich} \\ 0 & \text{falls } i\text{-te Person männlich} \end{cases}$$

- Verwenden diese Variable als erklärende Variable im Regressionsmodell
- Modell:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i = \begin{cases} \beta_0 + \beta_1 + \varepsilon_i & \text{falls } i\text{-te Person weiblich} \\ \beta_0 + \varepsilon_i & \text{falls } i\text{-te Person männlich} \end{cases}$$

- β_0: durchschn. Kreditkartenrechnungen der Männern
- $\beta_0 + \beta_1$: durchschn. Kreditkartenrechnungen der Frauen
- β_1: durchschn. *Unterschied* der Rechnungen Männern/Frauen

- Tabelle: Koeffizientenschätzungen für unser Modell:

	Koeffizient	Std.fehler	t-Statistik	p-Wert
Intercept	509.80	33.13	15.389	< 0.0001
gender[female]	19.73	46.05	0.429	0.6690

```
from statsmodels.formula.api import ols

fit = ols("Balance ~ Gender", data=df).fit()

fit.params
## Intercept             509.803109
## Gender[T.Female]       19.733123
## dtype: float64
fit.pvalues
## Intercept             2.908941e-42
## Gender[T.Female]      6.685161e-01
## dtype: float64
```

- Geschätzte durchschnittliche Rechnungen für Männer: \$ 509.80
- Geschätzter Unterschied zu Frauen: \$ 19.73
- Frauen: $\$\ 509.80 + \$\ 19.73 = \$\ 529.53$
- *p*-Wert für Indikatorvariable β_1 mit 0.6690 sehr hoch
- Kein statistisch signifikanter Unterschied der `balance` von Frauen und Männern

- Beispiel vorher: Frauen mit 1 und Männer mit 0 kodiert
- Völlig willkürlich
- Kodierung: *Kein* Einfluss auf Grad der Anpassung des Modells an Daten
- Unterschiedliche Kodierung: Unterschiedliche Interpretation der Koeffizienten
- Kodierung Männer mit 1 und Frauen mit 0
- Schätzung für die Parameter β_0 und β_1 \$ 529.53, resp. \$ -19.73
- Entspricht wiederum Rechnungen von:
  - Frauen: \$ 529.53
  - Männer: \$ 529.73 – \$ 19.73 = \$ 509.80
- Dasselbe Resultat wie vorher

Beispiel

- Anstatt der 0/1-Kodierung:

$$x_i = \begin{cases} 1 & \text{falls } i\text{-te Person weiblich} \\ -1 & \text{falls } i\text{-te Person männlich} \end{cases}$$

- Regressionsmodell:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i = \begin{cases} \beta_0 + \beta_1 + \varepsilon_i & \text{falls } i\text{-te Person weiblich} \\ \beta_0 - \beta_1 + \varepsilon_i & \text{falls } i\text{-te Person männlich} \end{cases}$$

- β_0: Durchschn. Rechnungen ohne Berücksichtigung des Geschlechts

- β_1: Wert, mit welchem Frauen über dem Durchschnitt liegen und mit welchem Männer unter dem Durchschnitt liegen

- β_0 durch \$ 519.665 geschätzt: Durchschn. Rechnungen von \$ 509.80 für Männer und von \$ 529.53 für Frauen

- Schätzung \$ 9.865 für β_1: Hälfte vom Unterschied \$ 19.73 zwischen Männern und Frauen

- Wichtig: Vorhersagen für d Zielgrösse hängen *nicht* von Kodierung ab

- Einziger Unterschied: Interpretation der Koeffizienten

Qualitative erklärende Variablen mit mehr als zwei Levels

- Qualitative erklärende Variable kann mehr als zwei Levels haben
- *Eine* Indikatorvariable für alle möglichen Werte reicht nicht
- In dieser Situation: Zusätzliche Indikatorvariable hinzufügen

Beispiel

- Variable `ethnicity`: *Drei* mögliche Levels
- Wählen *zwei* verschiedene Indikatorvariablen
- *Wahl* der 1. Indikatorvariablen:

$$x_{i1} = \begin{cases} 1 & \text{falls } i\text{-te Person asiatisch} \\ 0 & \text{falls } i\text{-te Person nicht asiatisch} \end{cases}$$

- 2. Indikatorvariable:

$$x_{i2} = \begin{cases} 1 & \text{falls } i\text{-te Person kaukasisch} \\ 0 & \text{falls } i\text{-te Person nicht kaukasisch} \end{cases}$$

- Beide Variablen in Regressionsgleichung aufnehmen:

$$y_i = \beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\varepsilon_i = \begin{cases} \beta_0+\beta_1+\varepsilon_i & \text{falls } i\text{-te Person asiatisch} \\ \beta_0+\beta_2+\varepsilon_i & \text{falls } i\text{-te Person kaukasisch} \\ \beta_0+\varepsilon_i & \text{falls } i\text{-te Person afroamerikanisch} \end{cases}$$

- β_0: Durchschn. Kreditkartenrechnungen von Afroamerikanern

- β_1: Differenz der durchschn. Rechnungen von Afroamerikanern und Asiaten

- β_2: Differenz der durchschn. Rechnungen von Afroamerikanern und Kaukasiern

Bemerkungen

- Es gibt immer eine Indikatorvariable weniger, als es Levels hat
- Level ohne Indikatorvariable (hier Afroamerikaner): *Baseline*
- Folgende Gleichung macht *keinen* Sinn:

$$y_i = \beta_0 + \beta_1 + \beta_2 + \varepsilon_i$$

  - Person müsste asiatisch *und* kaukasisch sein

- Output: Geschätzte `balance` \$ 531.00 für Baseline (Afroamerikaner):

```
fit = ols("Balance~Ethnicity", data=df).fit()

fit.params

## Intercept                     531.000000
## Ethnicity[T.Asian]            -18.686275
## Ethnicity[T.Caucasian]        -12.502513
## dtype: float64

fit.pvalues

## Intercept                     1.774117e-26
## Ethnicity[T.Asian]            7.739652e-01
## Ethnicity[T.Caucasian]        8.255355e-01
## dtype: float64
```

- Schätzung für Kategorie Asiaten: \$ -18.69
- Durchschn. Rechnungen um diesen Betrag kleiner als die von Afroamerikanern
- Kaukasier haben um durchschn. \$ 12.50 kleinere Rechnungen als die Afroamerikaner
- p-Werte gross $\quad\rightarrow\quad$ Zufällige Abweichungen
- Kein signifikanter Unterschied bei den Kreditkartenrechnungen zwischen den Ethnien
- Level, für Baseline willkürlich
- Vorhersage der Zielvariable hängt nicht von der Kodierung ab

- p-Werte hängen von der Kodierung ab
- F-Statistik betrachten
- F-Test und testen

$$H_0 : \quad \beta_1 = \beta_2 = 0$$

- p-Wert dieser Statistik hängt *nicht* von der Kodierung ab
- p-Wert 0.96 $\quad \rightarrow \quad$ Relativ hoch
- Vermutung bestätigt: Nullhypothese *nicht* verwerfen
- Es gibt keinen Zusammenhang zwischen `balance` und `ethnicity`

- Indikatorvariablen: Qualitative *und* quantitative erklärende Variablen in Regressionsmodell integrieren

- Regression von `balance` mit quantitativer erklärenden Variable `income` und qualitativer erklärenden Variable `student` durchführen

- `student` mit Indikatorvariablen

- Multiple lineare Regression

Beispiel: Datensatz `Credit`

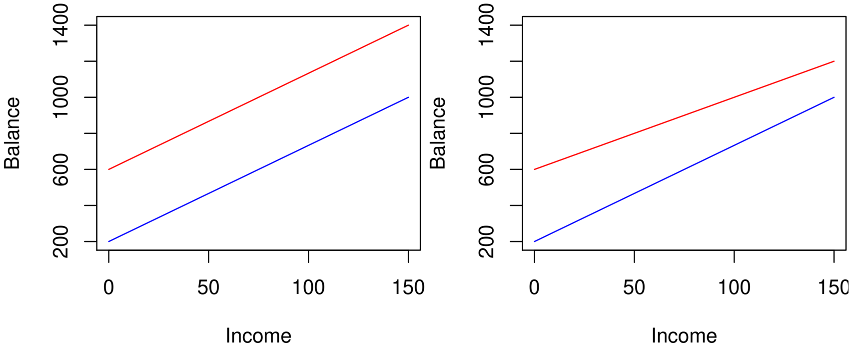
- Zielgrösse `balance` durch die erklärenden Variablen `income` (quantitativ) und `student` (qualitativ) vorhersagen

- Ohne Interaktionsterm:

$$
\begin{aligned}
\texttt{balance}_i &\approx \beta_0 + \beta_1 \cdot \texttt{income}_i + \begin{cases} \beta_2 & \text{falls } i\text{-te Person Student} \\ 0 & \text{falls } i\text{-te Person kein Student} \end{cases} \\
&= \beta_1 \cdot \texttt{income}_i + \begin{cases} \beta_0 + \beta_2 & \text{falls } i\text{-te Person Student} \\ \beta_0 & \text{falls } i\text{-te Person kein Student} \end{cases}
\end{aligned}
$$

- Modell beschreibt zwei parallele Geraden: eine für Studenierende und eine für Nichtstudierende
  - Steigung β_1 ist bei beiden gleich
  - y-Achsenabschnitte sind verschieden ($\beta_0 + \beta_2$ und β_0)

- Abbildung links:



- Durchschn. Zunahme von `balance` für Vergrösserung von `income` um eine Einheit hängt nicht davon ab, ob entsprechendes Individuum studiert oder nicht

- Mögliche Einschränkung des Modells: Änderung in `income` kann eine unterschiedliche Wirkung auf Rechnungen haben kann, ob jemand studiert oder nicht

- Lockerung dieser Einschränkung: Einführung einer Interaktionsvariablen

- `income` wird mit der Indikatorvariablen für `student` „multipliziert"

- Modell:

$$
\begin{aligned}
\texttt{balance}_i &\approx \beta_0 + \beta_1 \cdot \texttt{income}_i + \begin{cases} \beta_2 + \beta_3 \cdot \texttt{income}_i & \text{falls studierend} \\ 0 & \text{falls nicht studierend} \end{cases} \\
&= \begin{cases} (\beta_0 + \beta_2) + (\beta_1 + \beta_3) \cdot \texttt{income}_i & \text{falls studierend} \\ \beta_0 + \beta_1 \cdot \texttt{income}_i & \text{falls nicht studierend} \end{cases}
\end{aligned}
$$

- Zwei unterschiedliche Regressionsgeraden für Studierende und Nichtstudierende (Abbildung oben rechts):
  - Verschiedene Steigungen $\beta_1 + \beta_3$ und β_1
  - Unterschiedliche y-Achsenabschnitte $\beta_0 + \beta_2$ und β_0

- Möglichkeit, Änderung der Zielgrösse (Kreditkartenrechnungen) aufgrund der Änderungen im Einkommen für Studenten und Nichtstudenten getrennt zu betrachten

- Rechte Seite von Abbildung oben: Geschätzter Zusammenhang zwischen `income` und `balance` für Studenierende (rot) und Nichtstudierende (blau)

- Steigung für Studierende ist grösser als für Nichtstudierende

- Deutet an: Zunahme im Einkommen eines Studierenden eine grössere Zunahme der Kreditkartenrechnungen zur Folge hat als für Nichtstudierenden

Variablenselektion

- Lineares Standardregressionsmodell:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots \beta_p X_p + \varepsilon$$

- Beschreibung des Zusammenhanges zwischen der Zielvariable Y und den erklärenden Variablen $X_1, X_2, \ldots, X_p$ verwendet

- Schon gesehen: Nicht alle erklärenden Variablen spielen eine Rolle für die Vorhersage der Zielgrösse

- Frage: Ob Weglassen einer erklärenden Variablen den Grad, wie gut das Modell zu den Daten passt, wesentlich verschlechtert oder nicht

- Beide Regressionsmodelle miteinander vergleichen

- Es gibt viele Möglichkeiten, dies zu machen
- Hier einige *sehr* einfache Möglichkeiten
- Die hier beschriebenen Verfahren sind sehr einfach und sollen das Prinzip hinter der Variablenselektion erklären
- Sie werden Schritt für Schritt erklärt
- Diese Verfahren werden so nicht verwendet (darum in `Python` nicht implementiert)
- Kompliziertere Verfahren funktionieren aber ähnlich
- Erster Schritt in Richtung Machine Learning

Beispiel: Datensatz `Werbung`

- Multiples lineare Regressionsmodell:

$$\texttt{Verkauf} = \beta_0 + \beta_1 \cdot \texttt{TV} + \beta_2 \cdot \texttt{Radio} + \beta_3 \cdot \texttt{Zeitung} + \varepsilon$$

- Schon gesehen: `Zeitung` hat keinen (oder kaum) Einfluss auf `Verkauf`

- Vergleichen „grosses" Modell mit „kleinem" Modell (ohne `Zeitung`)

$$\texttt{Verkauf} = \beta_0 + \beta_1 \cdot \texttt{TV} + \beta_2 \cdot \texttt{Radio} + \varepsilon$$

- Vergleichen R^2 Werte

- Python-Ausgabe:

```
import pandas as pd
from statsmodels.formula.api import ols
df = pd.read_csv("../Data/Werbung.csv").drop("Unnamed: 0",
axis=1)

ols("Verkauf~TV+Radio+Zeitung", data=df).fit().rsquared
## 0.8972106381789522
ols("Verkauf~TV+Radio", data=df).fit().rsquared
## 0.8971942610828957
```

- R^2-Wert ändert sich kaum, wenn Variable `Zeitung` weggelassen wird
- Variable `Zeitung` überflüssig
- Weglassen

- Vergleichen ursprüngliches „grosses" Modell mit dem „kleinen" Modell (ohne `TV`)

$$\texttt{Verkauf} = \beta_0 + \beta_1 \cdot \texttt{Radio} + \beta_2 \cdot \texttt{Zeitung} + \varepsilon$$

- Output:

```
ols("Verkauf~TV+Radio+Zeitung", data=df).fit().rsquared
## 0.8972106381789522
ols("Verkauf~Radio+Zeitung", data=df).fit().rsquared
## 0.3327051839503228
```

- Massive Verschlechterung des R^2-Wertes, durch weglassen der Variable `TV`
- Die beiden Modelle passen folglich unterschiedlich gut zu den Daten
- Weglassen der Variable `TV` bewirkt eindeutige Verschlechterung auf die Güte des Modells, mit welcher das Modell zu den Daten passt

- Dasselbe bei Weglassen der Variable `Radio`:

```
ols("Verkauf~TV+Radio+Zeitung", data=df).fit().rsquared
## 0.8972106381789522
ols("Verkauf~TV+Zeitung", data=df).fit().rsquared
## 0.6458354938293274
```

- Was ist eine „deutliche” Verschlechterung?
- Kann mit Hypothesentest gemacht werden (nicht hier)

Schrittweise Vorwärtsselektion

- *Schrittweise Vorwärtsselektion*: Rechnerisch effiziente Methode, um Variablen zu eliminieren
- Beginnt mit Modell, das gar keine erklärenden Variablen enthält
- Dann wird schrittweise eine Variable um die andere zum Modell hinzugefügt, bis alle Variablen im Modell sind
- In jedem Schritt wird jene Variable ins Modell aufgenommen, die die grösste *zusätzliche* Verbesserung der Anpassung mit sich bringt

Credit

- *Nullmodell* $\mathcal{M}_0$: Enthält keine erklärenden Variablen:

$$\texttt{Balance} = \beta_0 + \varepsilon$$

- Fügen eine erklärende Variable zum Nullmodell hinzu
- Python-Befehl: Jede vorkommende Variable wird getrennt addiert (siehe Jupyter Notebook `forward_py.ipynb`):

```
df = pd.read_csv("../Data/Credit.csv").drop("Unnamed: 0", axis=1)

predictors = set(df.columns)
predictors.remove("Balance")
selected = []
for candidate in predictors:
    formula = "{} ~ {}".format("Balance", ' +
'.join(selected+[candidate]))
    score = ols(formula, data=df).fit().ssr
    print("{:<10}{}".format(candidate,score))
```

- Wählen *beste* Variable aus: Kleinster RSS-Wert

```
## Income    66208744.5107842
## Gender    84301019.9963956
## Cards     83709496.36968993
## Ethnicity 84321457.70952794
## Education 84334430.74220678
## Rating    21435122.032732937
## Limit     21715656.65911369
## Married   84337197.13548377
## Age       84339627.8817488
## Student   78681539.63888894
```

- Damit passt diese Variable am besten zu den Daten
- Hier: Variable `Rating`
- Modell $\mathcal{M}_1$:

$$\texttt{Balance} = \beta_0 + \beta_1 \cdot \texttt{Rating} + \varepsilon$$

- Zu diesem Modell fügen wir nun eine weitere Variable hinzu

- Code:

```
predictors.remove("Rating")
selected.append("Rating")
for candidate in predictors:
    formula = "{} ~ {}".format("Balance", ' +
'.join(selected+[candidate]))
    score = ols(formula, data=df).fit().ssr
    print("{:<10}{}".format(candidate,score))
```

```
## Income    10532541.29016962
## Gender    21419056.62479064
## Cards     21296542.449293762
## Ethnicity 21384022.44676935
## Education 21407879.418936443
## Limit     21427162.19690806
## Married   21316912.951277886
## Age       20786012.220216528
## Student   15699959.061316613
```

- Wählen wieder diejenige Variable aus, aufgrund welcher das ergänzte Regressionsmodell den kleinsten RSS-Wert hat

- Dies ist in diesem Fall `Income`

- Modell $\mathcal{M}_2$:

$$\texttt{Balance} = \beta_0 + \beta_1 \cdot \texttt{Rating} + \beta_2 \cdot \texttt{Income} + \varepsilon$$

- Verfahren wiederholt sich

- Fügen jene Variable zum Modell $\mathcal{M}_2$ hinzu, aufgrund welcher das neue Regressionsmodell den kleinsten RSS-Wert hat

- Dies ist hier `Student`
- Modell $\mathcal{M}_3$:

$$\texttt{Balance} = \beta_0 + \beta_1 \cdot \texttt{Rating} + \beta_2 \cdot \texttt{Income} + \beta_3 \cdot \texttt{Student} + \varepsilon$$

- Erhalten 11 Modelle $\mathcal{M}_0, \mathcal{M}_1, \dots, \mathcal{M}_{10}$
- Welches ist nun aber das beste unter diesen 11 Modellen?
- Als Entscheidungskriterium: AIC-Wert (letzten Spalte)
- Aufgrund von diesem Wert lassen sich verschiedene Modelle miteinander vergleichen

Schrittweise Rückwärtsselektion

- *Schrittweise Rückwärtsselektion* ist rechnerisch ebenfalls effizient und funktioniert ähnlich wie die schrittweise Vorwärtsselektion
- Beginnen allerdings mit dem vollen Modell, das alle erklärenden Variablen enthält
- Dann wird schrittweise eine Variable um die andere vom Modell entfernt, bis keine erklärende Variable mehr im Modell vorhanden ist
- In jedem Schritt wird jene Variable vom Modell entfernt, die am wenigsten nützlich ist
- Lassen die Variable weg, die den kleinsten p-Wert hat

- Code (siehe `backward_py.ipynb`:

```
pd.options.display.float_format = '{:.10f}'.format
predictors = set(df.columns)
predictors.remove("Balance")
selected = list(predictors)
formula = "{} ~ {}".format("Balance", ' + '.join(selected))
ols(formula, data=df).fit().pvalues
```

```
## Intercept                    0.0000000000
## Gender[T.Female]             0.2832368443
## Ethnicity[T.Asian]           0.2347046731
## Ethnicity[T.Caucasian]       0.4083088190
## Married[T.Yes]               0.4107255745
## Student[T.Yes]               0.0000000000
## Income                       0.0000000000
## Cards                        0.0000540120
## Education                    0.4920745729
## Rating                       0.0211221287
## Limit                        0.0000000121
## Age                          0.0374312744
## dtype: float64
```

- Lassen `Education` weg

```
predictors.remove("Education")
selected = list(predictors)
formula = "{} ~ {}".format("Balance", ' + '.join(selected))
ols(formula, data=df).fit().pvalues
```

```
## Intercept                  0.0000000000
## Gender[T.Female]           0.2864840001
## Ethnicity[T.Asian]         0.2333142753
## Ethnicity[T.Caucasian]     0.3965189924
## Married[T.Yes]             0.3845849442
## Student[T.Yes]             0.0000000000
## Income                     0.0000000000
## Cards                      0.0000534242
## Rating                     0.0179340937
## Limit                      0.0000000143
## Age                        0.0360867222
## dtype: float64
```

- Lassen `Ethnicity` weg
- Usw.

- Modell mit drei erklärenden Variablen: `Income`, `Limit` und `Student`
- Dieses Modell unterscheidet sich also vom Modell mit drei Variablen, das durch Vorwärtsselektion gewonnen wurde
- Hier kommt `Rating` anstelle von `Limit` vor

Wieviele Variablen wählen wir?

- Vorwärts- und Rückwärtsselektion: Nur beschrieben, *wie* Variablen ausgewählt werden, aber nicht *wieviele*
- Problem Vorwärtsselektion: R^2 nimmt mit zunehmeder Zahl von Variablen zu
- Sagt nichts aus, wieviele Variablen wir wählen sollen
- Es gibt mehrere Gütekriterien, die abhängig sind von der Anzahl der Variablen
- Beispiel: Adjusted-R^2
- Siehe Jupyter Notebook `r_squared_adj_py.ipynb`

- Gleiche Idee wie bei Vorwärtsselektion
- Addiert Variablen mit grösstem Adjusted-R^2-Wert
- Bricht ab, wenn Adjusted-R^2-Wert abnimmt

- Weitere Möglichkeit mit AIC (Akaike information criterion)
- Kleiner AIC-Wert ist besser
- Variablen werden addiert, solange AIC-Wert abnimmt
- Siehe Juypter Notebook `aic_py.ipynb`