1. Erste Analyse Datensatz

Wir verwenden den Titanic-Datensatz.

1.1. Ist der Datensatz vollständig?

Untersuchen Sie ggf. ein konkretes Beispiel für einen Passagier oder andere Quellen.

1.2. Wo fehlt es konkret noch an Verständnis der Semantik?

Nennen Sie Spaltenname(n) und die Fragen dazu.

2. Datenformat

Wir verwenden den Titanic-Datensatz.

2.1. Verändern Sie „Embarked“, so dass statt dem Kürzel der entsprechende Hafen genannt wird.

2.2. Formatieren Sie „Fare“ als Zahl mit zwei Nachkommastellen.

Was passiert mit dem Datentyp? Ist das gut?

2.3. Was passiert, wenn sie map() auf Null-Werte anwenden?

2.4. Ergänzen Sie 2.1 indem Sie auch ein Default-Mapping für Null-Werte definieren.

2.5. Was passiert, wenn Sie Werte in der Spalte haben, für die Sie kein Mapping definiert haben? Beispiel: In der Sex-Spalte steht neben „male“, „female“ auch „Mann“.

2.6. Sie wollen basierend auf „Age“ jeweils einen Satz ausgeben. Beispiel: „Ich bin 33 Jahre alt.“ ausgeben. Vermeiden Sie den Satz „Ich bin nan Jahre alt.

2.7. Bauen Sie ein (chained) Kommando, welches auf die Passagiere der ersten Klasse filtert und dann die Spalte „Sex“ auf die Werte M / F / X abbildet.

3. String Vektoroperationen

Wir arbeiten wieder mit dem Titanic-Datensatz.

3.1. Wandeln Sie die Spalte „Cabin“ in Kleinbuchstaben um.

a) Verwenden Sie die passende String Vektorfunktion.
b) Verwenden Sie .map() und rufen damit elementweise die Python Funktion str.lower auf.
c) Was geht bei b) schief und warum?

3.2. Ermitteln Sie die Nachnamen der Passagiere (jeweils nur der erste Name).

3.3. Die Kabinennummer besteht aus dem Deck und einer Raumnummer.

a) Ermitteln Sie das Deck aus der Kabinennummer.
b) Eine Kabinennummer sieht z.B. so aus: „C34“. Überprüfen Sie, ob alle Kabinennummer diese Form haben.
c) Wie ist die Beziehung zwischen Deck und Klasse?

3.4. Stimmt die Anrede im Namen mit dem Geschlecht überein?

a) Extrahieren Sie die Anrede („Mr.“, „Mrs.“ „Miss“,…) mit einem regulären Ausdruck aus dem Namen.
b) Mappen Sie die Anrede auf ein vermutetes Geschlecht („M“ oder „F“).
c) Stimmt das vermutete Geschlecht immer mit dem „Sex“ aus dem Datensatz überein? Zum Vergleich zweier Spalten gibt es z.B. die compare-Methode.

3.5. Zerlegen Sie die Namen in alle Bestandteile (Anrede, Nachname, Vornamen, Mädchenname). Nicht so schnell aufgeben, dass erfordert etwas Mühe.

a) Hierzu sollte man mit einem komplexeren Namensbeispiel starten und dessen Aufbau analysieren (Erst kommen die Nachname(n), dann ein Komma, dann…).
b) Daraus kann man dann einen regulären Ausdruck definieren.
c) Ermitteln Sie den Vornamen jeder Person (für ledige und verheiratete).
d) Welche Vornamen waren beliebt?

4. Bins

4.1. Lösen Sie das Beispiel aus der Vorlesung ohne die Methode pd.cut() zu verwenden.

$$\text{Age class} = \begin{cases} \text{'child'} \\ \text{'teenager'} \\ \text{'adult'} \\ \text{'elder'} \\ \text{pd.NA} \end{cases}, \text{ if Age} \in \begin{cases} [0,12] \\ ]12,18] \\ ]18,65] \\ ]65,100] \\ \text{else} \end{cases}$$

Hierzu müssen Sie eine eigene Funktion definieren und diese mit `map()` oder `apply()` anwenden. Probieren Sie beides aus.

4.2. Wir modifizieren die Anforderung leicht, so dass nun zwei Spalten als Input für die Funktion verwendet werden. Anstelle ‚adult' wollen wir ‚male' / ‚female' entsprechend des „Sex" stehen haben.

$$\text{Age class} = \begin{cases} \text{'child'} \\ \text{'teenager'} \\ 'male' \\ 'female' \\ \text{'elder'} \\ \text{pd.NA} \end{cases}, \text{ if Age} \in \begin{cases} [0,12] \\ ]12,18] \\ ]18,65] \; and \; "Sex" = 'male' \\ ]18,65] \; and \; "Sex" = 'female' \\ ]65,100] \\ \text{else} \end{cases}$$

Damit reicht `map()` nicht mehr aus und Sie müssten vermutlich `apply()` verwenden.

4.3. Bins anhand von Quartilen (`qcut()`)

a) Teilen Sie die „Fare" in die vier Quartile ein. Benennen Sie die Quartile sinnvoll („billig", „normal",..). Das Ergebnis sollte eine Spalte im titanic-Datensatz sein, welches das entsprechende Quartil enthält.

b) Wie sind die Quartile definiert (was sind die oberen und unteren Grenzen der „Fare" jedes Quartils?

c) Was ist der Median von „Fare"?

d) Wiederholen Sie die Aufgabe, aber nun getrennt nach „Pclass".

4.4. Alleine oder mit Familie?

a) Berechnen Sie eine Spalte „Alone", die anzeigt, ob ein Passagier allein gereist ist oder mit Familie. Familie würde bedeuten, dass entweder Bruder / Schwester / Ehefrau / Ehemann oder Eltern / Kinder mit an Bord waren.

b) Wie könnte man die Korrektheit dieser Information in den Daten überprüfen?

5. Plotting

5.1. Untersuchen Sie die Alterstruktur

a) Machen Sie ein Histogramm des Alters.

b) Untersuchen Sie dies genauer hinsichtlich Geschlechtes, Überlebende und Klasse.

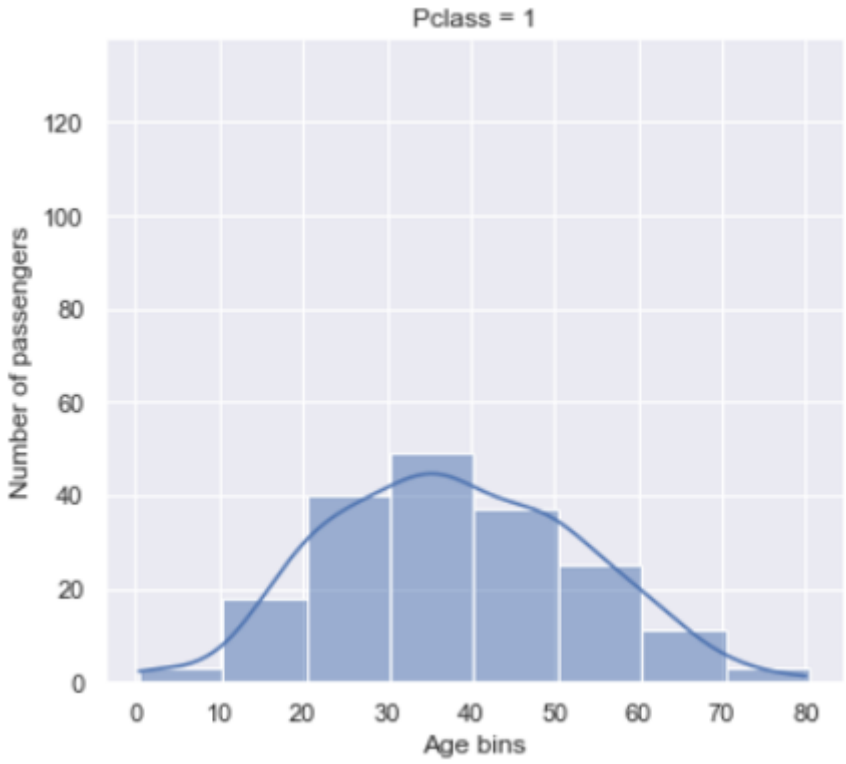
5.2. Untersuchen Sie „Fare“

a) Untersuchen Sie die Werte von „Fare“. Welches Diagramm ist hilfreich?

b) War die Fare gleich für Männer und Frauen? Von welchen anderen Merkmalen hängt dies ab?

6. Histogramm und Ausreißer

6.1. Rechnen Sie die Daten der Grafik aus der Vorlesung nach, d.h. geben Sie für jeden Altersbereich die Anzahl der Passagiere an.



6.2. Ausreißer

a) Finden Sie die Ausreißer mit „Fare“ größer als 300 GBP.
b) Ersetzen Sie deren Werte mit 300.
c) Entfernen Sie die 5% der Passagiere mit den höchsten Fahrpreisen.

7. Nulls

7.1. Detect

a) Geben sie alle Passagiere aus, für die die Altersangabe fehlt.
b) Wie viele sind dies?
c) Wie viele Werte sind jeweils für jede andere Spalte Null?
d) Wie viele Zeilen haben in irgendeiner Spalte eine Null?

7.2. Deal with

a) Ersetzen Sie die fehlenden Alterswerte durch den Median aller Passagiere.
b) Ersetzen Sie die fehlenden Alterswerte durch den Mittelwert in der jeweiligen P-Klasse.
c) Löschen Sie alle Passagiere mit unbekanntem „Embarked“.
d) Wie viele Zeilen wurden gelöscht?

8. Duplikate

Wir wollen prüfen, ob es doppelte Personen im Titanic-Datensatz gibt.

a) Extrahieren Sie die Nachnamen

b) Machen Sie sich mit Soundex vertraut.

Da es keine passende Bibliothek in anaconda gibt, können Sie sich den folgenden Code (von https://code.activestate.com/recipes/52213-soundex-algorithm/) kopieren

```
def soundex(name, len=4):
    """ soundex module conforming to Knuth's algorithm
        implementation 2000-12-24 by Gregory Jorgensen
        public domain
    """

    # digits holds the soundex values for the alphabet
    digits = '01230120022455012623010202'
    sndx = ''
    fc = ''

    # translate alpha chars in name to soundex digits
    for c in name.upper():
        if c.isalpha():
            if not fc: fc = c   # remember first letter
            d = digits[ord(c)-ord('A')]
            # duplicate consecutive soundex digits are skipped
            if not sndx or (d != sndx[-1]):
                sndx += d

    # replace first digit with first alpha character
    sndx = fc + sndx[1:]

    # remove all 0s from the soundex code
    sndx = sndx.replace('0','')

    # return soundex code padded to len characters
    return (sndx + (len * '0'))[:len]
```

Testen Sie die Funktion:

```
soundex('Cardeza')
```

```
'C632'
```

Lesen Sie einen Artikel zu Soundex.

c) Berechnen Sie für die Nachnamen die Soundex-Werte

d) Untersuchen Sie Passagiere mit identischem Soundex, um Duplikate zu finden (das kann man mit einem Join machen). Die Ergebnismenge muss weiter eingeschränkt werden, z.B. indem man das Geschlecht oder die Ticket Nummer berücksichtigt.

Finden Sie ein echtes Duplikat?

9. Tidy data

9.1. Billboard

Die Datei billboard.csv enthält für jeden Song wann er zuerst in den Top 100 Charts auftauchte. Dieses Datum entspricht der ersten Woche (wk1) für diesen Song. Der Platz in den Charts für jede der folgenden Wochen (wk1 bis wk75) wird verzeichnet.

Formen Sie billboard in tidy data um, benennen sie die Spalten um und in der Woche soll nur die Zahl stehen ohne „wk“. Optional: Erzeugen Sie die neue Spalte „date“ berechnet aus „date.entered“ plus die jeweilige Woche (haben wir noch nicht gemacht).

9.2. Pivot

Gegeben sei (kopieren Sie den Code und führen ihn aus):

```
data = {
    "name" : ["Philipp Woods", "Philipp Woods", "Philipp Woods",
"Jessica Cordero", "Jessica Cordero"],
    "property" : ["age", "height", "age", "age", "height"],
    "values" : [45, 186, 50, 37, 156]
}
df = pd.DataFrame(data)
df
```

	name	names	values
0	Philipp Woods	age	45
1	Philipp Woods	height	186
2	Philipp Woods	age	50
3	Jessica Cordero	age	37
4	Jessica Cordero	height	156

a) Formen Sie nach tidy data um. Welches Problem entsteht?
b) Wie kann man das lösen?
c) Wie könnte man eine Spalte ergänzen, so dass die Werte eindeutig identifiziert werden?

9.3. Long or wide?

Wandeln Sie nach tidy data. Was sind die Variablen?

pregnant	male	female
yes	<NA>	10
no	20	12

9.4. Ebola

Untersuchen Sie die Datei country_timeseries.csv und wandeln Sie in Tidy-Form. Berechnen Sie auch die Todesrate (Deaths je Cases).

Date	Day	Cases_Guinea	Cases_Liberia	Cases_SierraLeone	Cases_Nigeria
1/5/2015	289	2776		10030	
1/4/2015	288	2775		9780	
1/3/2015	287	2769	8166	9722	
1/2/2015	286		8157		
12/31/2014	284	2730	8115	9633	

9.5. Weather

Untersuchen Sie weather.csv. Es sind einige Wetterdaten (nur) einer Wetterstation MX17004 in Mexiko.

id	year	month	element	d1	d2	d3	d4	d5	d6	...	d22	d23	d24	d25	d26	d27	d28	d29	d30	d31
MX17004	2010	1	tmax	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	27.8	NaN
MX17004	2010	1	tmin	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	14.5	NaN
MX17004	2010	2	tmax	NaN	27.3	24.1	NaN	NaN	NaN	...	NaN	29.9	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

a) Bringen Sie das Wetter in tidy-Form. Hier tauchen mehrere Probleme gleichzeitig auf.
b) Was ist mit den fehlenden Werten? Was passiert Ende Februar?
c) Erzeugen Sie eine „date“ Spalte mit dem Datum im ISO-Format („2010-01-07“)

9.6. CORDIS

Wir verwenden die Daten von https://data.europa.eu/euodp/de/data/dataset/cordisH2020projects . Es sind EU geförderte Projekte. Die Datei der H2020 Projekte ist https://cordis.europa.eu/data/cordis-h2020projects.csv

Wir konzentrieren uns nur auf den Aspekt der Projektlaufzeit. (startDate , endDate) geben die Projektlaufzeit an. Wir wollen die Projekte nach Jahren auswerten, d.h. für brauchen für jedes Projekt und jedes Jahr in dem es aktiv war einen Eintrag.

Vorgehen: Als erstes brauchen wir eine Spalte, die eine Liste der aktiven Jahre enthält. Dann können wir diese Spalte explodieren.

Hinweis: In Python kann man auf diese Weise eine Liste erstellen:

```
[*range(7,12)]
```

```
[7, 8, 9, 10, 11]
```

Machen Sie testweise eine Auswertung, wie viele Projekte es per Jahr gab.

9.7. Normalisierung (decompose) Billboard

Wir betrachten billboard.csv von Aufgabe 9.1.

a) Wie sollte man billboard zerlegen? Geben Sie die beiden Schemata (Liste von Spalten) an und nennen jeweils den Schlüssel.
b) Zerlegen Sie billboard in diese zwei Tabellen.
c) Prüfen Sie, ob ihr angedachter Schlüssel wirklich einer ist (keine Duplikate).
d) Wieviel Speicherplatz benötigt die Zerlegung im Vergleich zur originalen billboard-Tabelle?
e) Joinen Sie die beiden Tabellen wieder zu einer billboard2-Tabelle. Sind die Daten identisch wie zuvor?

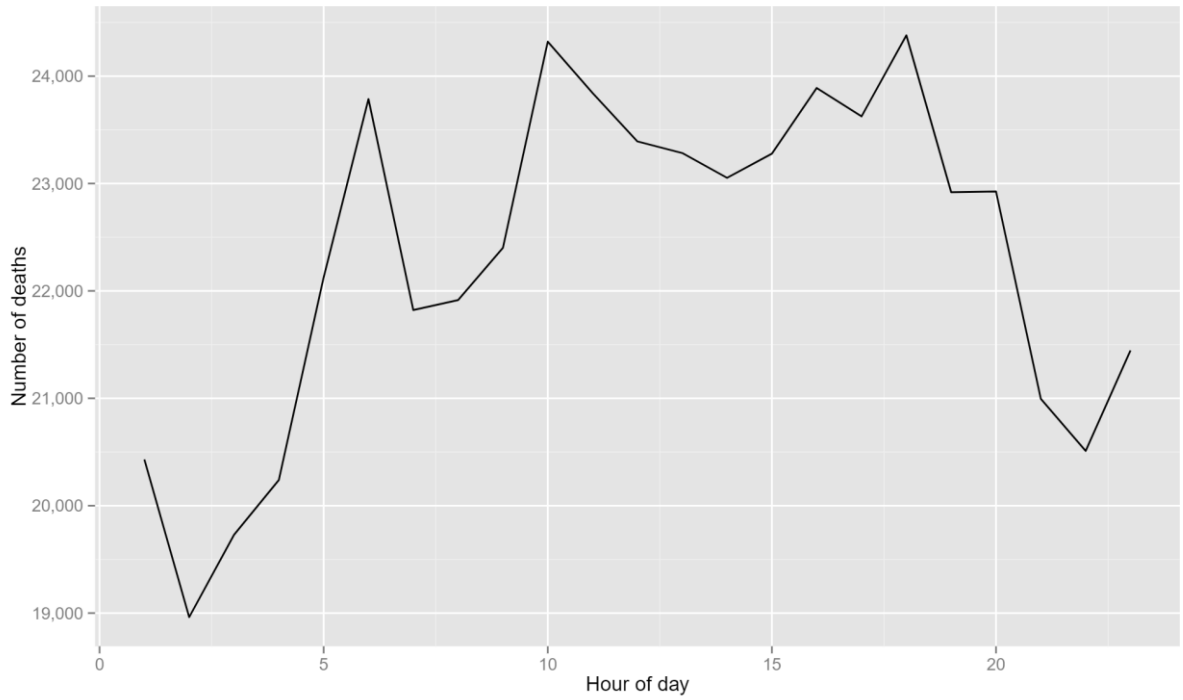
9.8. Fallstudie Todesraten in Mexiko

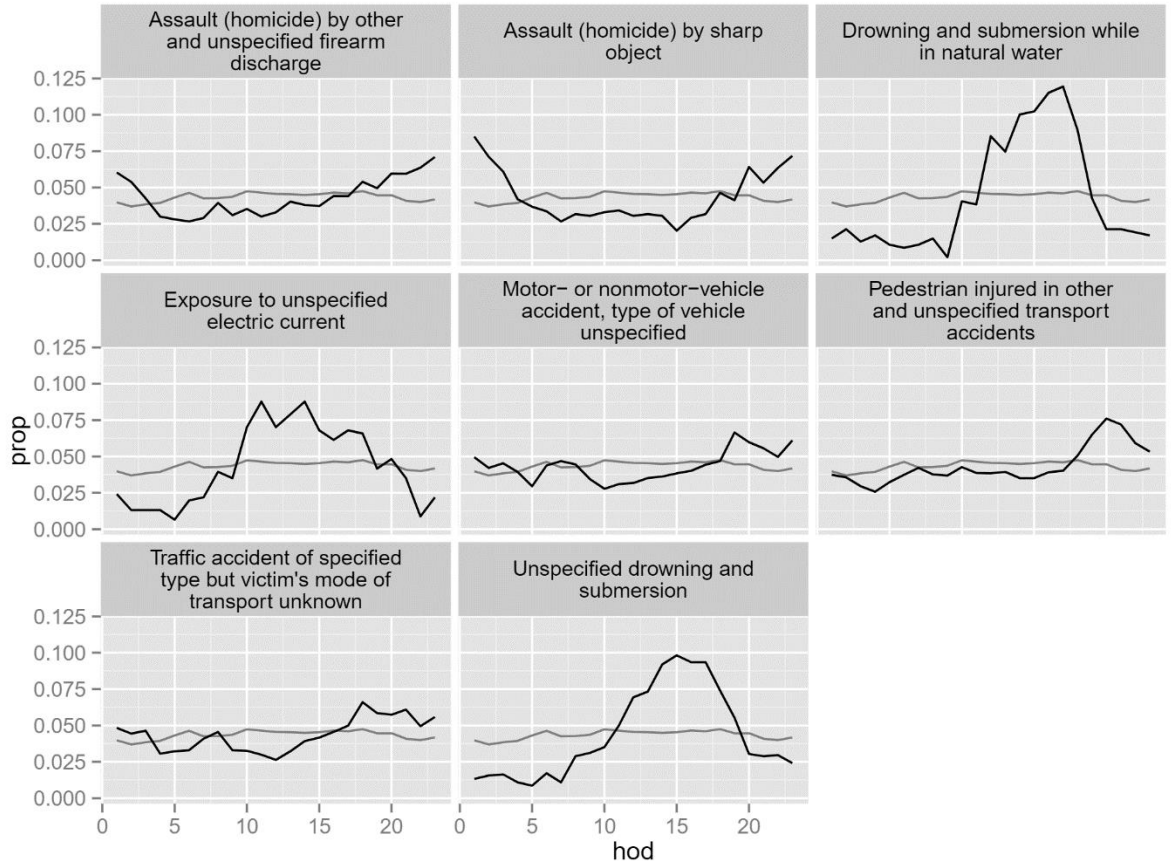
Quelle: Paper von Hadley Wickham http://dx.doi.org/10.18637/jss.v059.i10 . Die Fallstudie ist in Abschnitt 5 beschrieben und R Quellen liegen unter https://github.com/hadley/tidy-data/tree/master/case-study . Wir man am case-study.r Skript sieht, findet man die Originaldaten unter https://github.com/hadley/mexico-mortality/raw/master/deaths/deaths08.csv.bz2

Führen Sie die Fallstudie wie beschrieben durch mit Ausnahme der statistischen Überlegungen (ab kleinste Quadrate als distance; deviation; aber dann wieder im letzten Absatz die finalen Plots)!

Die CSV-Datei mit der Endung BZ2 ist eine komprimierte Datei. Kann man diese mit `pd.read_csv()` einlesen?

Erzeugen Sie die einige der folgenden Plots





9.9. Case Study Tuberkulose

Quelle: In Anlehnung https://r4ds.had.co.nz/tidy-data.html#case-study . Die Daten sind so nicht mehr im Zugriff, daher nehmen wir den aktuellen Bericht von https://www.who.int/teams/global-tuberculosis-programme/data und laden den Datensatz „Data provided by countries and territories > Case notifications [2Mb]“. Das Dictionary findet man weiter oben auf der Seite.

Formen Sie nach tidy und analysieren Sie!

9.10. Titanic Fahrpreis

Wir hatten in Aufgabe 5.2 festgestellt, dass die Lösung nicht korrekt ist, da „Fare“ in der Tabelle so nicht summierbar ist oder anders gesagt, „Fare“ kommt für ein Ticket mehrfach vor.

```
titanic.loc[titanic["Fare"] > 300]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
258	259	1	1	Ward, Miss. Anna	female	35.0	0	0	PC 17755	512.3292	NaN
679	680	1	1	Cardeza, Mr. Thomas Drake Martinez	male	36.0	0	1	PC 17755	512.3292	B51 B53 B55
737	738	1	1	Lesurer, Mr. Gustave J	male	35.0	0	0	PC 17755	512.3292	B101

a) Wenn man nach „Cabin“ auswerten will müsste man zunächst „Cabin“ explodieren. Das machen wir nicht, weil es keine sinnvolle Auswertung gibt. Die Kabinen bilden für diese Reise ohnehin eine Einheit. Simmt das?

b) Hinsichtlich „Fare“:
Man könnte näherungsweise die „Fare“ auf die Personen aufteilen, die mit dem Ticket fahren. Diese neue „fare_share“ wäre dann summierbar. Das ist nicht ganz realistisch, da der zweite Erwachsene und Kinder in der Kalkulation vermutlich günstiger sind.

Berechnen Sie das und wiederholen eine Auswertung von zuvor. Verändert sich das Ergebnis.

c) Alternativ könnte man (Ticket, Fare, Pclass) herausziehen und zumindest die Auswertung über Pclass wiederholen.

10. Method chaining

10.1. Schreiben Sie das Folgende als Kette

```
url = "https://raw.githubusercontent.com/pandas-
dev/pandas/master/doc/data/titanic.csv"
df = pd.read_csv(url)
df["class"] = df.pclass.map({1: "First", 2: "Second", 3: "Third"})
df["who"] = df[["age", "sex"]].apply(woman_child_or_man, axis=1)
df["adult_male"] = df.who == "man"
df["deck"] = df.cabin.str[0].map(lambda s: np.nan if s == "T" else s)
df["embark_town"] = df.embarked.map({"C": "Cherbourg", "Q":
"Queenstown", "S": "Southampton"})
df["alive"] = df.survived.map({0: "no", 1: "yes"})
df["alone"] = ~(df.parch + df.sibsp).astype(bool)
df = df.drop(["name", "ticket", "cabin"], axis=1)

def woman_child_or_man(passenger):
    age, sex = passenger
    if age < 16:
        return "child"
    else:
        return dict(male="man", female="woman")[sex]
```