Capstone Project: Supervised Learning

How much money you should take for your AirBnB listing as a host?

Robert Knop

Outline

1. Dataset and Research Question
2. Feature Engineering and Modeling
3. Practical Use
4. Weak Points and Shortcomings
5. Wrap up

1

Dataset and Research Question

Dataset

- Downloaded from Kaggle
- AirBnB listings in Berlin

Dataset Overview

Dataset info

Number of variables	16
Number of observations	22552
Total Missing (%)	2.2%
Total size in memory	2.8 MiB
Average record size in memory	128.0 B

Variables types

Numeric	10
Categorical	6
Boolean	0
Date	0
Text (Unique)	0
Rejected	0
Unsupported	0

Warnings

`availability_365` has 10788 / 47.8% zeros **Zeros**
`host_name` has a high cardinality: 5998 distinct values **Warning**
`last_review` has 3908 / 17.3% missing values **Missing**
`last_review` has a high cardinality: 1313 distinct values **Warning**
`minimum_nights` is highly skewed (γ1 = 85.888) **Skewed**
`name` has a high cardinality: 21874 distinct values **Warning**
`neighbourhood` has a high cardinality: 136 distinct values **Warning**
`number_of_reviews` has 3890 / 17.2% zeros **Zeros**
`price` is highly skewed (γ1 = 26.733) **Skewed**
`reviews_per_month` has 3914 / 17.4% missing values **Missing**

Research Question

- As a host you might want to know what is a reasonable price for your listing.
- Build a model which can suggest a price, given certain inputs.
- Is the model able to identify the most important inputs (features)

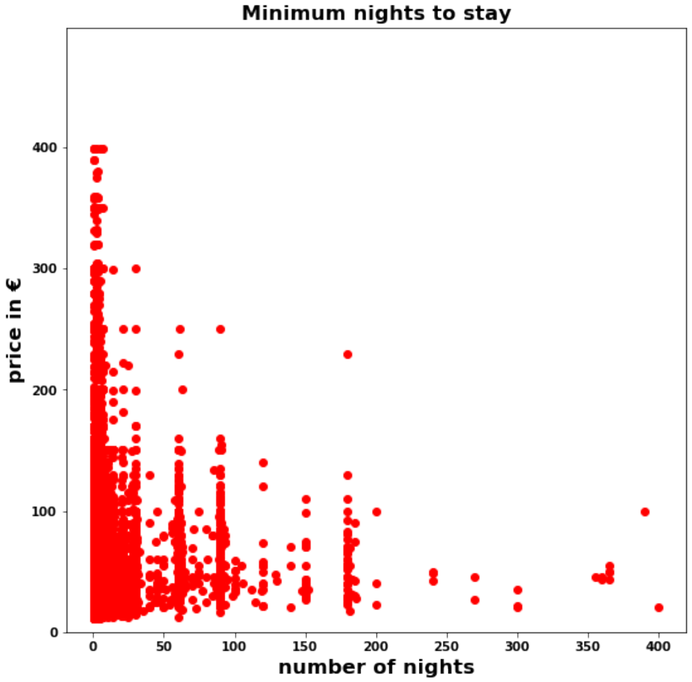
2

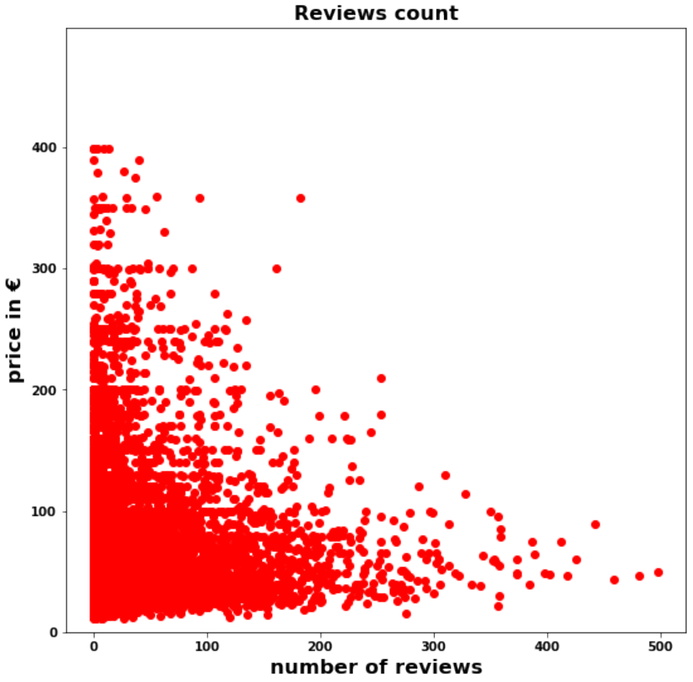
Feature Engineering and Modeling

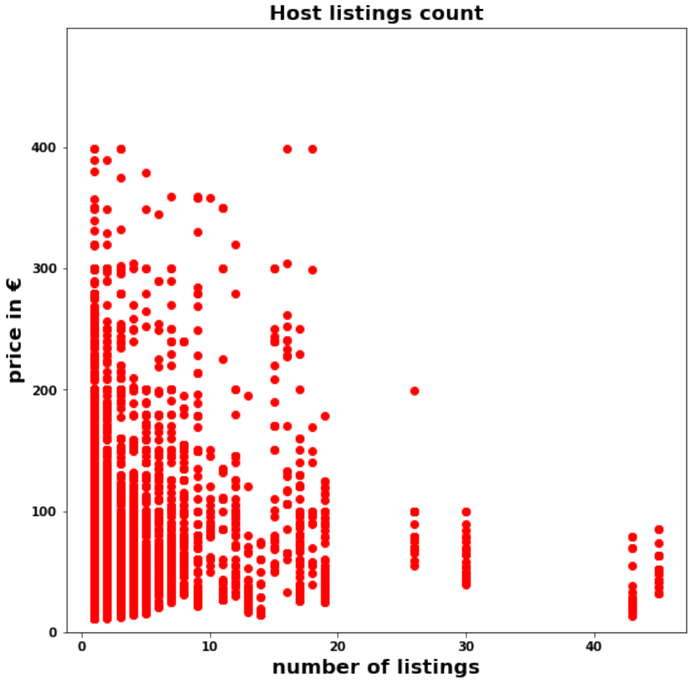
Feature Engineering

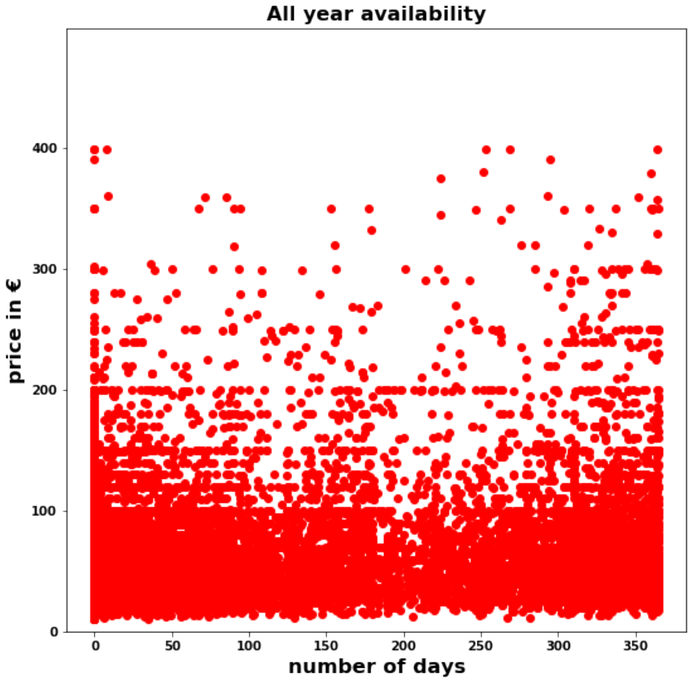
2

```python
# Feature Engineering and Selection

# Transform the date variable into something a ML model can use.
# In this case we calculate the number of days passed since the last review. After that we have an integer.
df['last_review'] = pd.to_datetime(df['last_review'])
df['days_since_last_review'] = (df['last_review'] - dt.datetime(2018, 12, 31)).dt.days

# Rename our target variable (Y)
df['y_price'] = df['price']

# Drop unnecessary columns
df = df.drop(columns=[
    'name', # we won't do any NLP here
    'last_review', # already converted into 'days_since_last_review'
    'price', # was copied into 'y_price'
    'id', # just a increasing number
])

# Cleaning: Get rid of outliers
# Drop examples where
# the price is higher than 400€
# and lower than 10€
df = df[df['y_price'] > 10] # 22522 - 22491 = 31 --> under 0.1% of all data
df = df[df['y_price'] < 400] # 22491 - 22374 = 117 --> 0.5% of all data
# remove all listings which require a minimum stay of more than 500 nights
df = df[df['minimum_nights'] < 500] # 22374 - 22368 = 6 --> under 0.1% of all data
```

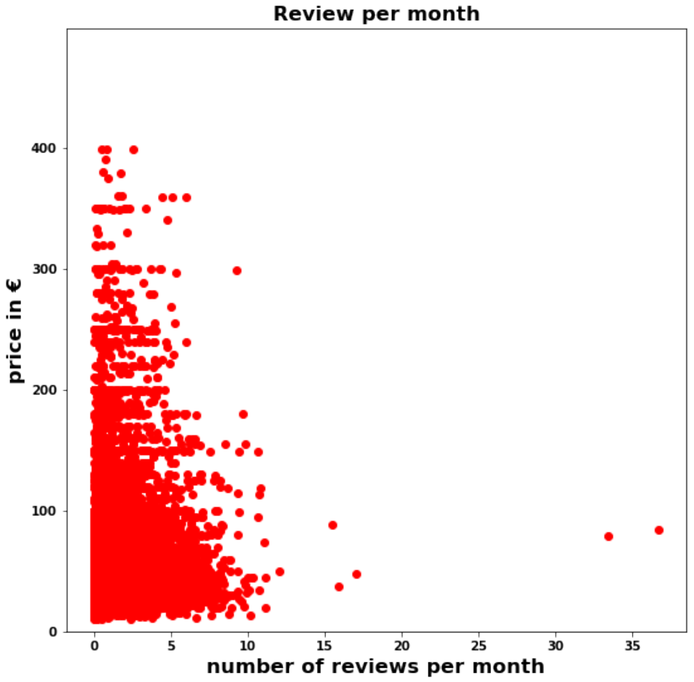

Minimum nights to stay
price in €
number of nights

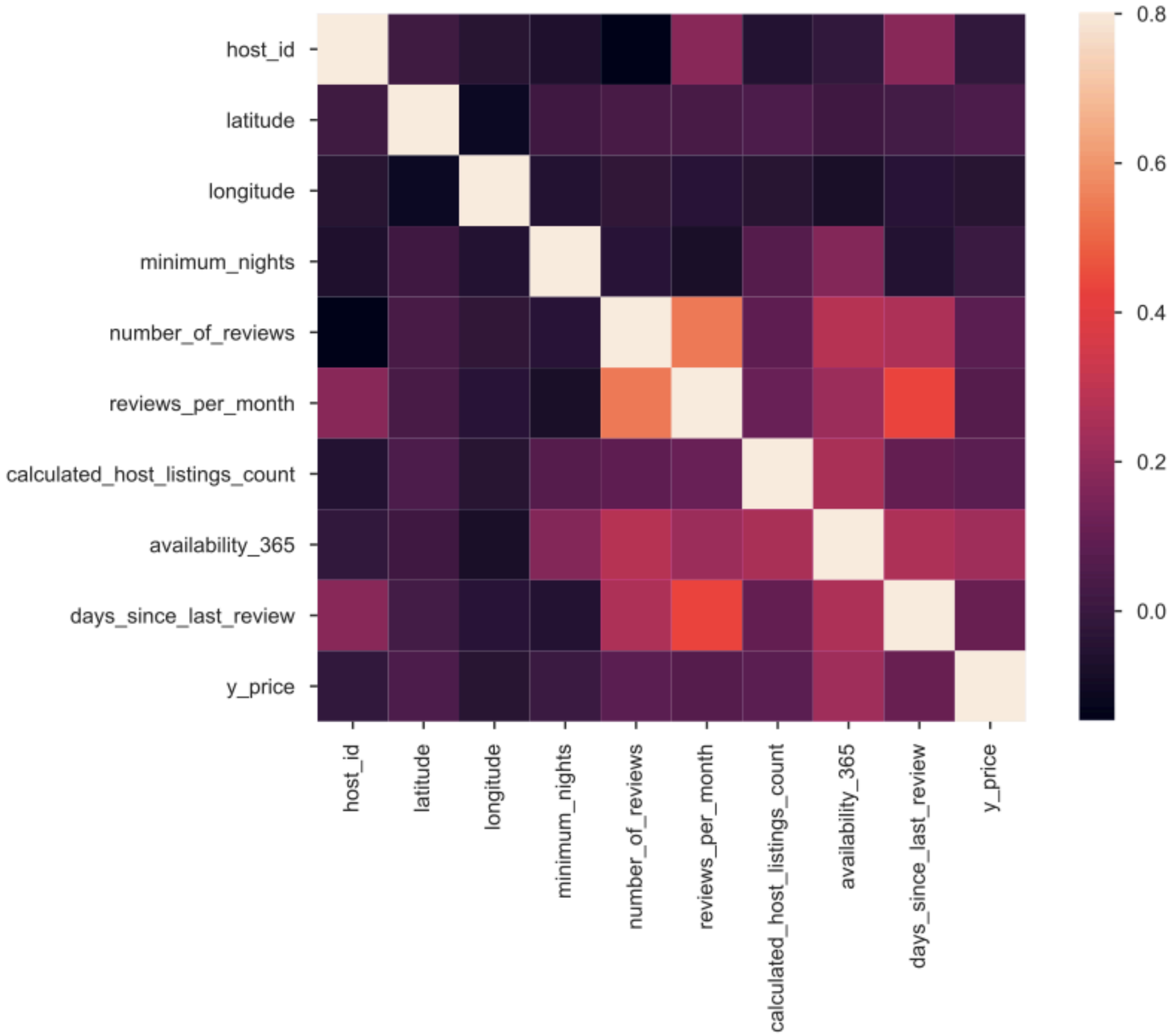

Reviews count
price in €
number of reviews


Host listings count
price in €
number of listings


All year availability
price in €
number of days


Days since last review
price in €
number of days


Review per month
price in €
number of reviews per month


host_id
latitude
longitude
minimum_nights
number_of_reviews
reviews_per_month
calculated_host_listings_count
availability_365
days_since_last_review
y_price
0.8
0.6
0.4
0.2
0.0

Prepossessing and Selection

2

```
# Normalize
mm_scaler = MinMaxScaler()
df[['longitude']] = mm_scaler.fit_transform(df[['longitude']].values)
df[['latitude']] = mm_scaler.fit_transform(df[['latitude']].values)
df[['minimum_nights']] = mm_scaler.fit_transform(df[['minimum_nights']].values)
df[['number_of_reviews']] = mm_scaler.fit_transform(df[['number_of_reviews']].values)
df[['reviews_per_month']] = mm_scaler.fit_transform(df[['reviews_per_month']].values)
df[['availability_365']] = mm_scaler.fit_transform(df[['availability_365']].values)
df[['calculated_host_listings_count']] = mm_scaler.fit_transform(df[['calculated_host_listings_count']].values)
df[['days_since_last_review']] = mm_scaler.fit_transform(df[['days_since_last_review']].values)

# Define X and y
X = df.drop(columns=[
                    'y_price', # is the Y
                    'neighbourhood_group', # is categorical
                    'neighbourhood', # is categorical
                    'room_type', # is categorical
                    'host_id', # it should not be an input, model should suggest a price independent of the host id
                    'host_name' # a name should not be a good predictor (also add bias)
                    ])
X = pd.concat([X, pd.get_dummies(df['neighbourhood_group'])], axis=1)
X = pd.concat([X, pd.get_dummies(df['neighbourhood'])], axis=1)
X = pd.concat([X, pd.get_dummies(df['room_type'])], axis=1)

y = df['y_price']
```

Model Selection

2

- Models:
    1. Linear Regression (also Lasso and Ridge)
    2. K Nearest Neighbour
    3. Random Forest Regressor
    4. SVM
    5. Gradient Boosting Regressor
- Methods:
    - SelectKBest
    - Hyperparameter tuning with GridSearchCV
    - PCA
    - Cross Validation

Model Selection

2

	Ridge Regression	KNN k = 24 weights=distance	Random Forest	SVM	Gradient Boosting
rms error	34.77	34.26	33.00 with GridSearchCV	37.61	33.39 with GridSearchCV
rmse (cross validate cv= 5)	34.66 (+/- 24.83)	35.02 (+/- 25.66)	33.61 (+/- 25.53) with GridSearchCV	—	33.79 (+/- 25.82) with GridSearchCV
SelectKBest k=120	—	—	33.02 33.56 (+/- 24.93) with GridSearchCV	—	33.22 33.93 (+/- 25.39) with GridSearchCV
PCA (100)	34.66 (+/- 24.83)	—	33.43 33.63 (+/- 25.07)	—	33.93 33.95 (+/- 25.25) with GridSearchCV

Model Selection

2

	Ridge Regression	KNN k = 24 weights=distance	Random Forest	SVM	Gradient Boosting
rms error	34.77	34.26	33.00 with GridSearchCV	37.61	33.39 with GridSearchCV
rmse (cross validate cv= 5)	34.66 (+/- 24.83)	35.02 (+/- 25.66)	33.61 (+/- 25.53) with GridSearchCV	—	33.79 (+/- 25.82) with GridSearchCV
SelectKBest k=120	—	—	33.02 33.56 (+/- 24.93) with GridSearchCV	—	33.22 33.93 (+/- 25.39) with GridSearchCV
PCA (100)	34.66 (+/- 24.83)	—	33.43 33.63 (+/- 25.07)	—	33.93 33.95 (+/- 25.25) with GridSearchCV

Explanatory Power of Linear Regression

2

	feature	coef_value		feature	coef_value
20	Entire home/apt	28.10	18	Tempelhof - Schöneberg	-0.97
7	days_since_last_review	24.00	16	Spandau	-2.68
6	availability_365	23.21	10	Lichtenberg	-3.45
12	Mitte	16.29	11	Marzahn - Hellersdorf	-6.23
14	Pankow	14.21	13	Neukölln	-7.58
4	reviews_per_month	11.88	21	Private room	-9.09
1	longitude	8.33	19	Treptow - Köpenick	-12.33
8	Charlottenburg-Wilm.	6.34	3	number_of_reviews	-13.52
9	Friedrichshain-Kreuzberg	5.86	17	Steglitz - Zehlendorf	-15.11
15	Reinickendorf	5.66	22	Shared room	-19.01
5	calculated_host_listings_count	4.35	0	latitude	-59.40
			2	minimum_nights	-94.36

3 Practical Use

☀️Urban Jungle Suite in Sunny 1910s Kreuzberg Oasis

2 guests **1 bedroom** **1 bed** **1 bath**

Get €25 off your booking.

"Start your day with a cup of coffee and watch the amazing sunrise from the balcony."

Hosted by Mae & Mike

ENTIRE GUEST SUITE IN BERLIN
★★★★★97

€89 / night

Request to Book

4

Weak Points and Shortcomings

Weak Points and Shortcomings

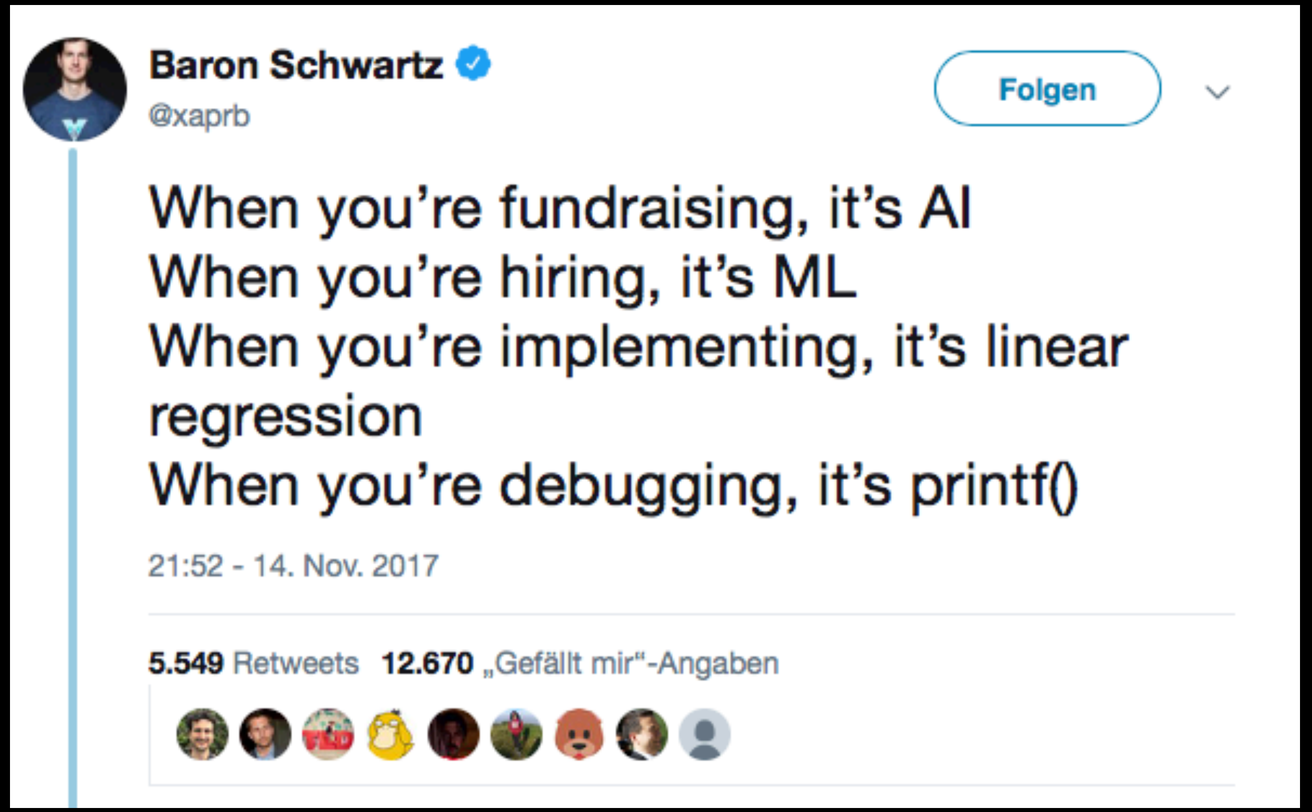
- suggests a price only between 10€ and 400€, minimum nights < 500d
- No NLP interpretation of listing name
- features missing: rooms, floor size, specials like roof top, sauna
- rethink other outliers (e.g. number of reviews month > 31?)
- time dynamics like seasonal events / periods

5 Wrap Up

Wrap Up

- AirBnB is already using ML for suggesting prices (good business case)
- Learning: Remove outliers improves the models by a huge margin
- Domain knowledge is helping a lot

Robert Knop
@robertknop
www.linkedin.com/in/robert-knop94
https://github.com/RobKnop


Baron Schwartz
@xaprb
Folgen
When you’re fundraising, it’s AI
When you’re hiring, it’s ML
When you’re implementing, it’s linear regression
When you’re debugging, it’s printf()
21:52 - 14. Nov. 2017
5.549 Retweets 12.670 „Gefällt mir“-Angaben