# Product Development

# Final Report

Team 6

# AirBnB Custom host guidelines using text mining

Yeo Sungdong Eom Jiwoo Isaí González Olmos

SUNGKYUNKWAN UNIV. Department of Systems Management Engineering

## ABSTRACT

### Background

Airbnb has been raising ability of hosts to preventing downgrade of brand image because of hosts' immature hosting ability. But general guidelines AirBnB's offering haven't been a real help for hosts to get better response.

### Purpose

By creating a program that enables the host to understatnd the needs of customers based on written reviews on AirBnB site, we offer benefits for those stakeholders (hosts, customers, AirBnb itself).

### Methods

-From text mining the host reviews, find hosts' preparation for strong and weak points

-From text mining whole reviews in specific city where hosting place exist, find City's local characteristics.

### Results

-figure1

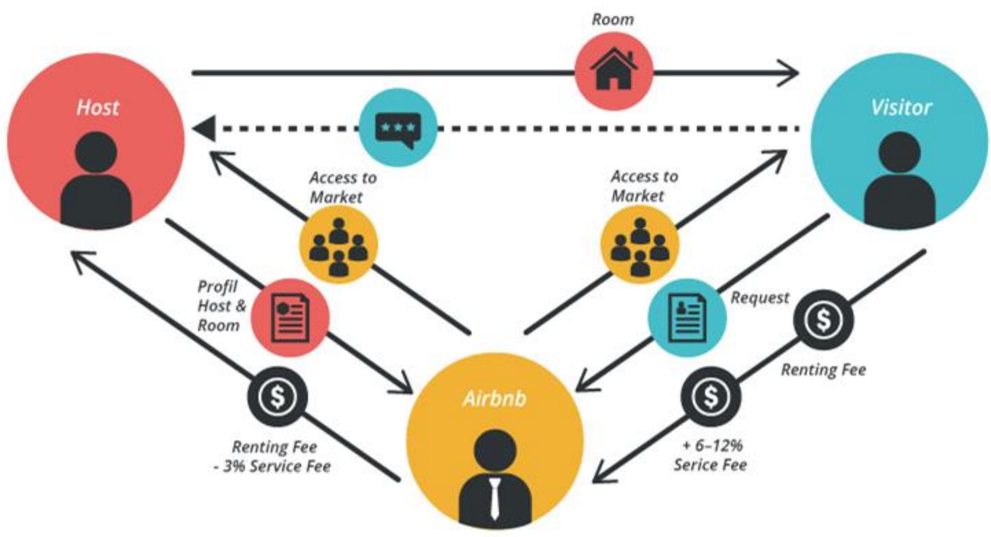
-figure2

### Conclusions

-limitation

-expectation

-Future Work




AirBnB - business model

Seeing the relationship between Airbnb and Host, Airbnb has been trying to raise ability of hosts to preventing downgrade of brand image because of hosts' immature hosting ability. But general guidelines AirBnB's offering haven't been a real help for hosts to get better response.

We could summary their host basic requirements.

**Maintain a high overall rating** (To guarantee guests a level of quality, no matter where they go)

**Provide essential amenities** (soap, towels, pillows,etc)

**Be responsive** (Respond booking inquiries and reservation requests within 24 hrs)

**Accept reservation requests** (Accept guests when available)

**Avoid cancellations** (These are taken seriously as they can affect the guests plans)

From the problems.

- General guidelines
- Guests may change necessities on different cities
- May need to address hosts specific problems

We brought a solution to help all stakeholders(hosts, customers, AirBnB itself).

**Solution**

We create a program that enables the host to understand the needs of customers based on written reviews on AirBnB site. By this solution, the host will know easily and in detail what to provide to the customer, and the customer experience will improve. And Airbnb's reputation and income will also rise as the overall quality of accommodation improves.

# Methods

## <Dataset>

We could get Airbnb review datasets without crawling. and the dataset looks like this.

```
import pandas as pd

df=pd.read_csv("../content/gdrive/My Drive/Colab Notebooks/reviews_tex.csv")

df.head()
```

| | listing_id | id | date | reviewer_id | reviewer_name | comments |
|---|---|---|---|---|---|---|
| **0** | 1078 | 142 | 2008-09-13 | 1344 | Amy | A fabulously cozy place! It was slightly diff... |
| **1** | 1078 | 207074 | 2011-03-22 | 451657 | Angela | perfect!---:) Will stay again! |
| **2** | 1078 | 217779 | 2011-04-04 | 473682 | Jenn | I had a great stay at Brian's garden apartment... |
| **3** | 1078 | 6841593 | 2013-08-26 | 8153398 | Kathryn | The reservation was canceled 19 days before ar... |
| **4** | 1078 | 7071134 | 2013-09-05 | 420381 | Rita | Fantastic Experience! My daughter and I staye... |

'listing_id' column is telling the host house number. So i dropped useless columns and used only ['listing_id', 'comments'], these two columns.

## <Preprocessing>

```
listing_id       312965
id               312965
date             312965
reviewer_id      312965
reviewer_name    312965
comments         312818
dtype: int64
```

```
[ ]  # df_2818=df
     def ingest_train():
         data = pd.read_csv('/content/gdrive/My Drive/Colab Notebooks/reviews_tex.csv')
         data = data[data.comments.isnull() == False]

         data = data[data['comments'].isnull() == False]
         data['comments'] = data['comments'].map(str)
         data.reset_index(inplace=True)
         data.drop('index', axis=1, inplace=True)
         return data
     df_2818=ingest_train()
     df_2818.head()
```

Because commets column has 312818 which has some null values, so I dropped those rows and dataset has been replaced to 312818 rows dataset.

*For topic modeling we used LDA model(Latent Dirichlet Allocation), and it only can be applied when Vectorized by count vectorizer.

And for the countvectorizer, this is sequence of feature vectorization.

- Preprocessing(lowercase, deleting space, punctuation etc)
- Tokenization
- Text Normalization(Stopwords, Lemmatization etc)
- Feature extraction from token words and apply vectorization

```python
# return the wordnet object value corresponding to the POS tag
from nltk.corpus import wordnet

def get_wordnet_pos(pos_tag):
    if pos_tag.startswith('J'):
        return wordnet.ADJ
    elif pos_tag.startswith('V'):
        return wordnet.VERB
    elif pos_tag.startswith('N'):
        return wordnet.NOUN
    elif pos_tag.startswith('R'):
        return wordnet.ADV
    else:
        return wordnet.NOUN

#preprocessing~

import string
from nltk import pos_tag
from nltk.corpus import stopwords
from nltk.tokenize import WhitespaceTokenizer
from nltk.stem import WordNetLemmatizer

def clean_text(text):
    # lower text
    text = text.lower()
    # tokenize text and remove puncutation
    text = [word.strip(string.punctuation) for word in text.split(" ")]
    # remove words that contain numbers
    text = [word for word in text if not any(c.isdigit() for c in word)]
    # remove stop words
    stop = stopwords.words('english')
    text = [x for x in text if x not in stop]
    # remove empty tokens
    text = [t for t in text if len(t) > 0]
    # pos tag text
    pos_tags = pos_tag(text)
    # lemmatize text
    text = [WordNetLemmatizer().lemmatize(t[0], get_wordnet_pos(t[1])) for t in pos_tags]
    # remove words with only one letter
    text = [t for t in text if len(t) > 1]
    # join all
    text = " ".join(text)
    return(text)

# clean text data
```

**<Sentimental Analysis>**

This dataset has no rating column, which means that i need to sentimental analysis by unsupervised learning.

There are some sentimental dictionary using lexicon in python.

And i used VADER package which mainly offer sentimental analysis for text from social media. Because this package offer better sentimental analysis result and fast, i picked it.

```
from nltk.sentiment.vader import SentimentIntensityAnalyzer

#use vader to use lexicon of words
sid = SentimentIntensityAnalyzer()
df_2818["sentiments"] = df_2818["comments"].apply(lambda x: sid.polarity_scores(x))
df_2818 = pd.concat([df_2818.drop(['sentiments'], axis=1), df_2818['sentiments'].apply(pd.Series)], axis=1)

# add number of characters column
df_2818["nb_chars"] = df_2818["comments"].apply(lambda x: len(x))

# add number of words column
df_2818["nb_words"] = df_2818["comments"].apply(lambda x: len(x.split(" ")))
```

And for visualization, we used wordcloud.

```python
# highest positive sentiment reviews (with more than 5 words)
pd.set_option('display.max_colwidth', -1)
df_2818[df_2818["nb_words"] >= 5].sort_values("pos", ascending = False)[["comments", "pos"]].head(10)

# Lowest negative sentiment reviews (with more than 5 words)

pd.set_option('display.max_colwidth', -1)
df_2818[df_2818["nb_words"] >= 5].sort_values("neg", ascending = False)[["comments", "neg"]].head(10)
```

And from the polarity_score function, I could make the positive score columns and negative score columns, finally extracted the top 5 positive comments, and top 5 negative comments.

| | comments | pos |
|---|---|---|
| 193 | Everything is great, thank you, Daniel! | 0.633 |
| 96 | Daniel is nice. It was fun. | 0.604 |
| 188 | Daniel was an amazing host! His apartment was really nice and clean. Love the neighborhood as well. | 0.578 |
| 234 | Daniel is a great host | 0.577 |
| 56 | Daniel's is fantastic! ₩r₩nGreat location. Quiet room. Good beds. Fantastic bathroom. Spotlessly clean. Nice host. Fun bikes!₩r₩nHe has thought of everything to make his guests comfortable. We enjoyed our stay in Amsterdam very much thanks to Daniel!₩r₩nDefinitely recommend and would love to go back. | 0.566 |

| | comments | neg |
|---|---|---|
| 136 | Wir haben uns bei Daniel sehr wohl gefühlt. Er ist ein zuvorkommender Gastgeber. Das Zimmer war modern und komfortabel eingerichtet, super für einen kurzen Städtetripp. Besonders ausgefallen ist die Sauberkeit und die tolle Organisation von Daniel. Die Anbindung zur Innenstadt war angenehm und mit einem Tagesticket auch nicht zu teuer. Die Nachbarschaft ist eher ruhig, es gibt aber alle Geschäfte des täglichen Bedarfs. Wir kommen gerne wieder! | 0.270 |
| 160 | Daniel,₩r₩n₩r₩nGracias y mil gracias por tu estadía, no pudo ser mejor!!! Sin lugar a dudas el mejor bnb en el que estuvimos.₩r₩n | 0.260 |
| 131 | Das Zimmer bzw. die Wohnung ist genauso wie beschrieben! ₩r₩nDaniel ist sehr angenehm, ein sehr netter Gastgeber. Er hat uns viele Tipps gegeben, war immer sehr hilfsbereit. Wenn er nicht persönlich für Fragen da war konnten wir ihn mobil erreichen - er meldete sich immer umgehend.₩r₩nDie Wohnung ist sehr schön eingerichtet, alles picobello sauber.₩r₩nDie Anbindung mit dem Bus bzw. mit der Straßenbahn ist super.₩r₩nFalls wir nochmals nach Amsterdam fahren, dann wieder zu Daniel! | 0.216 |
| 76 | Sehr saubere und gepflegte Wohnung. Daniel war sehr hilfsbereit und zuvorkommend. Die Verkehrsanbindung ist super. Man ist in ca. 10 min. am Hauptbahnhof. Ich würde jederzeit wieder bei Daniels übernachten, wenn ich in Amsterdam bin. | 0.178 |
| 105 | Daniels Unterkunft ist ein perfekter Ort um in Amsterdam zu übernachten. Die Anbindung mit Bus und Bahn ist perfekt. Man kann auch zu Fuß die Innenstadt gut erreichen und auf dem Weg gibt es einiges zu entdecken. Die Betten sind gemütlich und die Wohnung geschmackvoll eingerichtet. Daniel hatte gute Tipps für uns und ist ein sehr angenehmer Zeitgenosse. Mein persönlicher Tipp ist das Thai Restaurant in der Javastraat um die Ecke. Nächstes mal in Amsterdam bin ich gerne wieder Gast dort. Thank's for the time at your place. See you next time. Peace. | 0.177 |

## <Topic Modeling>

I used Latent Dirichlet Allocation(LDA) topic modeling, so the count vectorized the preprocessed column name:' df_2818["df_2818_clean"]'.

```python
from sklearn.feature_extraction.text import CountVectorizer

count_vect=CountVectorizer()
feat_vect = count_vect.fit_transform(df_2818["df_2818_clean"])
feat_vect.shape
```

```
(255, 2321)
```

```python
[19] from sklearn.decomposition import LatentDirichletAllocation

lda=LatentDirichletAllocation(n_components=10, random_state=0)
lda.fit(feat_vect)
```

And picked 10 topics by changing the parameter ‘n_components’=10.

```
Topic # 0
daniel stay great host home get recommend would city apartment need room make amsterdam bike
Topic # 1
de et très est daniel nous bien la un pour séjour en le tout ce
Topic # 2
und bei he centre great wir uns request gute provide apartment exactly daniel helpful love
Topic # 3
el por heart organise gracias mejor al lugar tranquilo barrio distance gran daniel en muy
Topic # 4
daniel und ist die sehr zu er ich war ein amsterdam late time night block
Topic # 5
daniele abbiamo hear window prepare view arrival 总之 there cancel reservation arrivo siamo ottima accoglienza
Topic # 6
daniel stay place amsterdam host clean room great would everything get nice well recommend also
Topic # 7
daniel stay amsterdam place room go get would one make city key home bus truly
Topic # 8
muy la de daniel en casa todo que un para es como el lo una
Topic # 9
daniel molto amsterdam una ci send ha bathroom we il per città da sempre after
```

And could see topics like this picture.

## Results

For example, we choosed a host name 'Uncle B', doing AirBnB host in Texas.

Name: 'Uncle B', City: 'Texas'

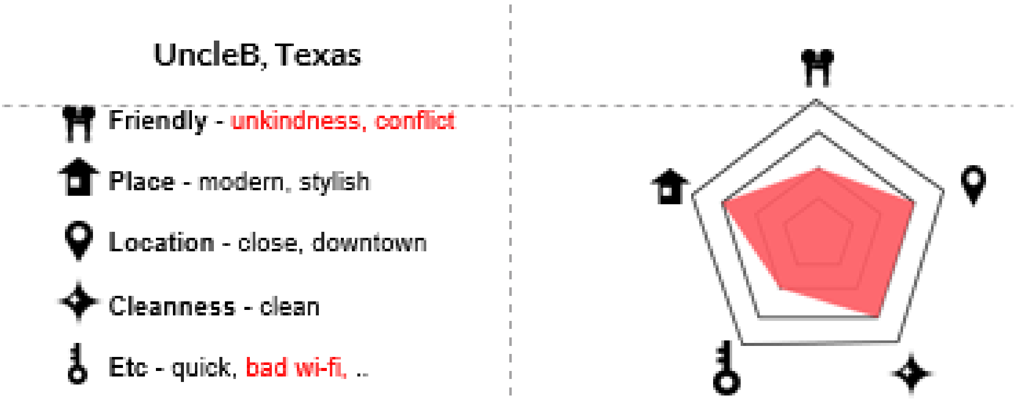



Figure1

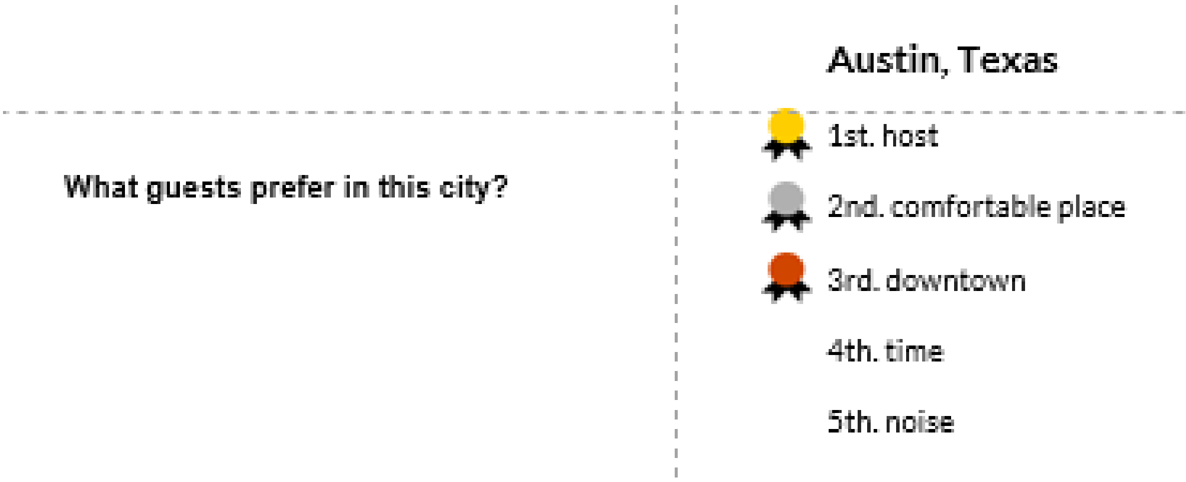



Figure2

From the wordcloud and topic modeling we could make the result which is figure 1 and 2. Figure 1 is Strong and weak points of the hosting and Figure 2 is what hosts should prepare to bit competitors in the city.

## Conclusions

We could do sentimental analysis and topic modeling from the written reviews on AirBnB site. And we find out that the sentimental analysis is not that useful for this solution, because of the specific characteristic of Airbnb. Customers from hotels actually don't much care about writing the negative reviews, but in the Airbnb case, they usually do the business person by person, so the reviews in AirBnB are almost positive reviews. This cause the algorithm don't make proper weak points for hosts.

**Expectancy**

- Host can see the summarized review without having to look at all reviews.
- Host become more aware of the pros and cons in theirs, so they can improve.
- The host can see what needs to be further developed with local characteristics.
- Customers are more likely to choose a place with the accommodations they want.
- As the overall quality of host increases, there can be more people using AirBnB.

**Future Work**

- Make model more accurate
- Summarize the result so simple that hosts can get it intuitively
- Expand the application of the program to other cities
- Automate our program so that it can provide customized guideline in real-time