



From ETL to Insights

A Dream Team Presentation



Project Workflow

01

Brainstorming

- Gathering ideas
- Selecting our topic and data sets

02

Objectives

- What do we wish to discover?
- What do we need to do?

03

Data Preparing

- Extracting variables

04

Transforming/Loading

- Cleaning
- Loading data to SQL server

05

Analysis

- Using analyzing tools to answer our questions

06

Conclusion

- Challenges faced
- What could be done better?

01

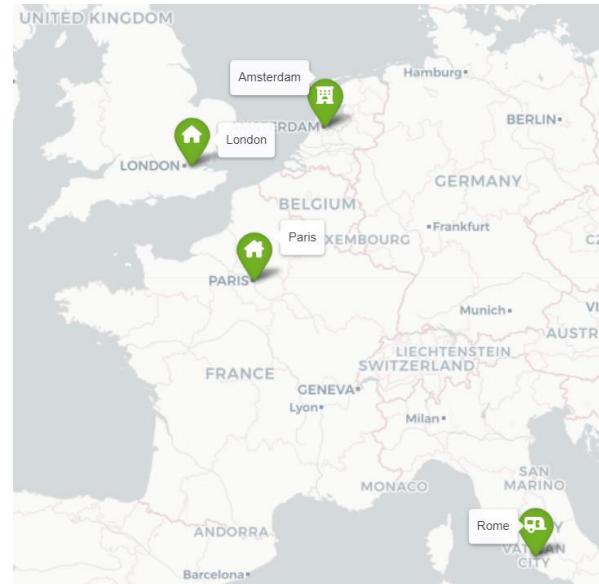
Brainstorming

Why Airbnb?



- Airbnb's data is collected by an advocacy group called Inside Airbnb.
- Data files are organized by city ► can easily select preferred cities for study.
- Data is easy to understand ► data dictionary.
- Gives us a taste of what it's like to handle big data for a consumer brand.
- As Travellers, we're familiar with Airbnb's data set 

Which Cities to Focus On?



Data Observations



In our Airbnb data sets, location data was stored as latitude and longitude coordinates. Possibly mirrors how Airbnb stores location data. If coordinates are derived from a physical address ➤ "geocoding."

Some advantages:

- Pinpoint a specific location within a city.
- Caters for locations with the same name.
- Precise distance calculations between a potential stay and landmarks or attractions ➤ only possible with exact coordinates.
- Regulatory requirements, tax rates, and legal permissions can vary within different areas of a city.

Data Dictionary

Although no description was provided for the "review_scores" fields, as Airbnb users, we knew what these scores referred to.

Data Dictionary for `listings.csv` detailed file

Field	Type	Calculated	Description	Reference
review_scores_rating				
review_scores_accuracy				
review_scores_cleanliness				
review_scores_checkin				
review_scores_communication				
review_scores_location				
review_scores_value				
license	text		The licence/permit/registration number	
instant_bookable	boolean		[t=true; f=false]. Whether the guest can automatically book the listing without the host requiring to accept their booking request. An indicator of a commercial listing.	
calculated_host_listings_count	integer	y	The number of listings the host has in the current scrape, in the city/region geography.	
calculated_host_listings_count	integer	y	The number of Entire home/apt listings the host has in the current scrape, in the city/region geography	
calculated_host_listings_count	integer	y	The number of Private room listings the host has in the current scrape, in the city/region geography	
calculated_host_listings_count	integer	y	The number of Shared room listings the host has in the current scrape, in the city/region geography	
reviews_per_month	numeric	y	The number of reviews the listing has over the lifetime of the listing	

02

Objectives

Our Questions

Primary questions:

1. For each city, are reviews more positive or negative?
2. What are the most frequent words for the reviews? What can this tell us about consumer concerns?
3. Is there a correlation between price and review sentiment?
(E.g. Do reviews tend to become more positive or negative as prices increase?)
4. Which cities have the highest rating scores — overall, as well as for specific qualities?

"Bonus" questions:

1. Do more experienced Airbnb hosts get more positive or negative reviews?
2. For Airbnb Superhosts, are their reviews more positive or negative?
3. For Airbnb hosts who don't verify their identity, does this impact their rating?
4. Which types of homes receive the most positive reviews?

03

Data Preparing

Data Selection: Listings

From 75 fields in the “listings” data, we extracted:

1. id | the unique id for each listing
2. name | name of the listing
3. host_since
4. host_location
5. host_is_superhost
6. host_identity_verified
7. property_type
8. price
9. review_scores_rating
10. review_scores_accuracy | does the actual home match its description?
11. review_scores_cleanliness | how clean is the home?
12. review_scores_checkin | how smooth is the check-in process?
13. review_scores_communication | is it easy to communicate with the host?
14. review_scores_location | is the location good for travellers?
15. review_scores_value | is the home value for money?

Data Selection: Reviews

From 6 fields in the “reviews” data, we extracted:

1. listing_id
2. date
3. comments | review text

Project Goals



Data Cleaning & Transformation

Use real-world data to simulate the ETL work of data engineers.



Data Analysis

Explore Python libraries for text analysis:
Which tools are more effective?



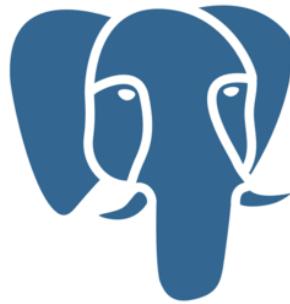
Data Insights

Find different ways to draw useful insights from our data set.

04

Transforming/Loading

Where to
Host the
Data?



Where to Host the Data?

What is Supabase:

- Build on PostgreSQL online server
- Allows user to host database online and share with others

Advantage over PgAdmin?:

- User can do query online and is recorded in real time.
- Provides coding help similar like chatgpt for SQL.

Disadvantage:

- Limit to the file size being uploaded.
- Limited Database size.
- Limit to 2 Database.
- Only 1 way to create table header, manually setup in Supabase.

Data Preparation

2 groups of data:

- Listing Info, Guest Reviews

Listing Group:

- Remove unwanted columns and special characters.

Review Group:

- Preprocess the CSV files before reading into dataframes.
- Encoding set to UTF-8
- Remove HTML tag

- Parallel Processing to optimize the process

Data Loading

Workflow:

- Set up connection to PostgreSQL Server.
- Define a database name.
- Create a connection engine to the database.
- Use a for loop to create all the 10 tables.
- Read all the cleaned data into dataframes and ingest to PostgreSQL



05

Analysis

Sentiment Analysis: Quick Facts



How it Works:

1. Breaking Down the Text:

Text is split into smaller pieces (common phrases, words, or parts of words), for easier analysis.

2. Understanding the Tone:

Sentiment analysis models may look at individual words and phrases, or they may consider the whole context of the sentence or paragraph.

Deciding the Overall Sentiment:

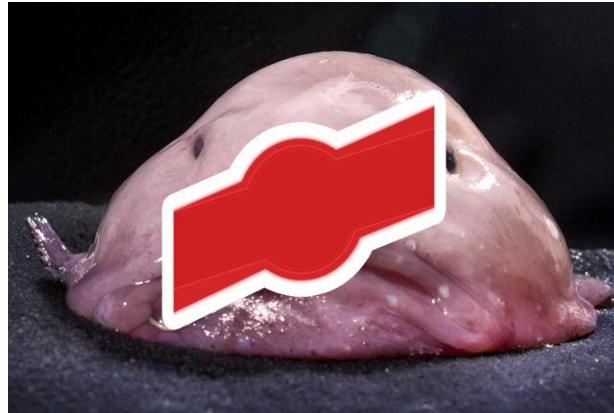
A score is generated, to indicate if the text is more positive or negative. Some models also score for neutrality.

Sentiment Analysis: Different Tools

Comparing 3 Sentiment Analysis Tools

- Text Blob
- Transformer
- VADER

Sentiment Analysis: Text Blob



```
In [81]: from textblob import TextBlob  
  
TextBlob("I don't like this movie.").sentiment  
  
Out[81]: Sentiment(polarity=0.0, subjectivity=0.0)
```

```
In [82]: from textblob.sentiments import PatternAnalyzer  
  
blob = TextBlob("I don't like this movie.", analyzer=PatternAnalyzer())  
blob.sentiment  
  
Out[82]: Sentiment(polarity=0.0, subjectivity=0.0)
```

Sentiment Analysis: Transformers Vs. Vader



```
In [83]: # !pip install transformers
# !pip install torch

from transformers import pipeline

model_name = "distilbert-base-uncased-finetuned-sst-2-english"
sentiment_pipeline = pipeline("sentiment-analysis", model=model_name)
result1 = sentiment_pipeline("I don't like this movie.")
result2 = sentiment_pipeline("This movie is pretty bad.")
result3 = sentiment_pipeline("This movie is ok.")
result4 = sentiment_pipeline("This movie is great.")

print(result1)
print(result2)
print(result3)
print(result4)

[{"label": "NEGATIVE", "score": 0.9938645958900452}]
[{"label": "NEGATIVE", "score": 0.9997928738594055}]
[{"label": "POSITIVE", "score": 0.999839186668396}]
[{"label": "POSITIVE", "score": 0.9998801946640015}]
```



```
In [4]: from nltk.sentiment.vader import SentimentIntensityAnalyzer
sia = SentimentIntensityAnalyzer()

In [5]: result1 = sia.polarity_scores("I don't like this movie.")
result2 = sia.polarity_scores("This movie is pretty bad.")
result3 = sia.polarity_scores("This movie is ok.")
result4 = sia.polarity_scores("This movie is great.")

result = [result1, result2, result3, result4]
result_overall = pd.DataFrame.from_dict(result)
```

	Sentences	compound	type
result 1	I don't like this movie.	-0.2755	negative
result 2	This movie is pretty bad.	-0.0772	negative
result 3	This movie is ok.	0.2960	positive
result 4	This movie is great.	0.6249	positive

Sentiment Analysis: Models

Model 1: "DistilBERT base uncased finetuned SST-2"

VS

Model 2: VADER

Example Sentence: "The plot had some truly unexpected twists that really threw me off, but in a good way!"

VADER: assigns scores to words like "unexpected" and "good." If a common phrase like "threw me off" is in its lexicon, it will score that as a phrase.

BERT: BERT stands for Bidirectional Encoder Representations from Transformers. Looks at the whole sentence, is designed to understand that "unexpected" and "threw me off" are positives in this sentence.

Sentiment Analysis: Process

Putting "BERT" from Transformers to work:

1. Each comment is checked for length, truncated if too long.
2. "BERT" analyses the sentiment of each comment—generates a sentiment score and label, but we only need the label.
3. Add "sentiment_label" as a new column to the DataFrame. Based on the count of Positive sentiment labels, the overall sentiment for the comments in each table is determined and printed—'Mostly Positive,' 'Mostly Negative,' or 'Neutral' (same number of positive and negative comments).

Applying "VADER":

1. VADER generates a compound score for each comment.
2. Depending on the score, they are categorised as follows:
 - **Positive** sentiment, compound ≥ 0.05
 - **Negative** sentiment, compound ≤ -0.05
 - **Neutral** sentiment, compound between -0.05 to 0.05

Sentiment
Analysis:
Answering
Data
Questions

Meeting our
Objectives

Sentiment Analysis:

For each city, are reviews more positive or negative?



```
# Determine overall sentiment
positive_count = comments_df['sentiment'].value_counts().get('POSITIVE', 0)
overall_sentiment = "Neutral"
if positive_count > len(comments_df) / 2:
    overall_sentiment = "Mostly Positive"
elif positive_count < len(comments_df) / 2:
    overall_sentiment = "Mostly Negative"

print(f"Table {table_name} ({'sample' if table_size > THRESHOLD else 'full'}) is {overall_sentiment}")
```

Table amsterdam_review (sample) is Mostly Positive
Table london_review (sample) is Mostly Positive
Table new_york_review (sample) is Mostly Positive
Table paris_review (sample) is Mostly Positive
Table rome_review (sample) is Mostly Positive

```
# Determine overall sentiment
positive_count = comments_df['type'].value_counts().get('positive', 0)
percent_count = round((positive_count/SAMPLE_SIZE)*100, 2)
overall_sentiment = "Neutral"
if positive_count > len(comments_df) / 2:
    overall_sentiment = "Positive"
elif positive_count < len(comments_df) / 2:
    overall_sentiment = "Negative"

print(f"{table_name} ({'sample' if table_size > THRESHOLD else 'full'}) is {percent_count}% {overall_sentiment}")
```

amsterdam_review (sample) is 97.2 % Positive
london_review (sample) is 96.4 % Positive
new_york_review (sample) is 94.4 % Positive
paris_review (sample) is 95.6 % Positive
rome_review (sample) is 98.0 % Positive

Word Frequency: NLTK + Wordcloud

What are the most frequent words for the reviews?

What can this tell us about consumer concerns?



```
In [85]: from collections import Counter
import matplotlib.pyplot as plt
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import pandas as pd
from sqlalchemy import create_engine, text
from wordcloud import WordCloud

table_names = ["amsterdam_review", "london_review", "new_york_review", "paris_review", "rome_review"]

THRESHOLD = 10000 # If table size is more than this, use random sampling
SAMPLE_SIZE = 1000 # Size of the random sample if sampling is used

# Stop words are common words (e.g., 'the', 'is', 'in')
# Often filtered out during text preprocessing and analysis
stop_words = set(stopwords.words('english'))
```

Word Cloud for amsterdam review



Applying sentimental analysis to Word Cloud

Positive Word Cloud for amsterdam review



Negative Word Cloud for amsterdam_review



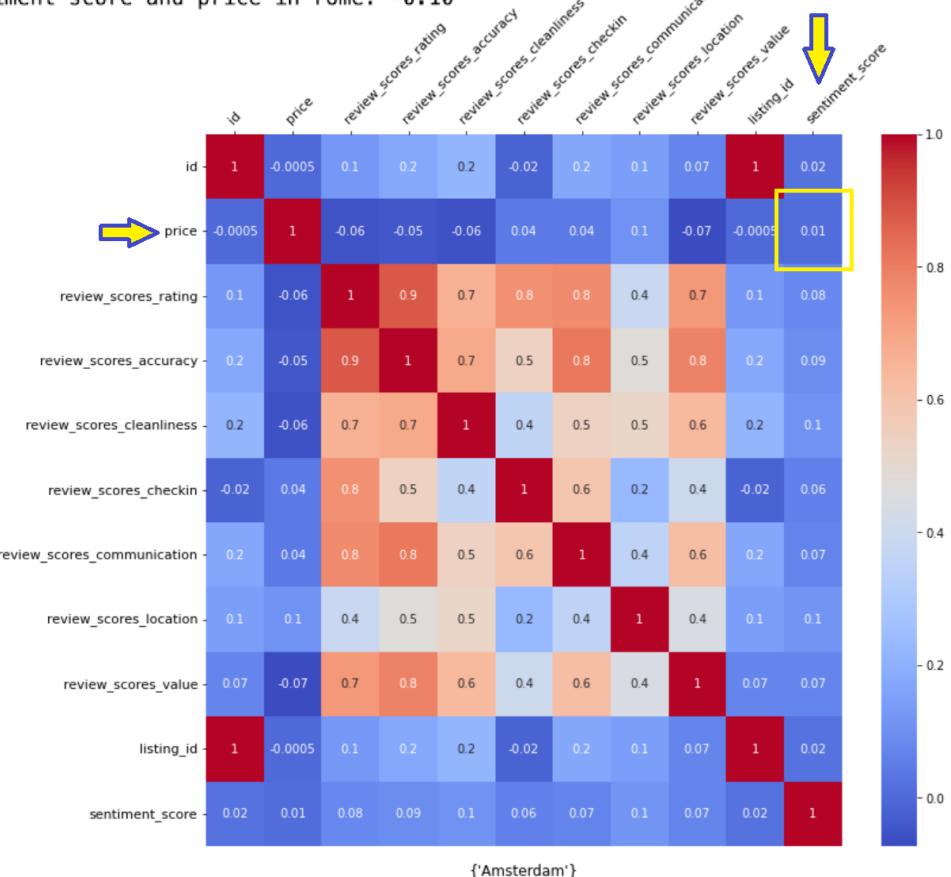
Neutral Word Cloud for amsterdam review



Correlation Heatmap

Is there a correlation between price and review sentiment?

Correlation between sentiment score and price in amsterdam: 0.01
Correlation between sentiment score and price in london: 0.00
Correlation between sentiment score and price in new_york: 0.00
Correlation between sentiment score and price in paris: 0.02
Correlation between sentiment score and price in rome: -0.10



Bar Plot: Seaborn + Matplotlib

Which city has the highest rating scores – overall?

Seaborn makes it easy to use colour palettes for plots, with the palette parameter. It is built on top of matplotlib, and they work hand in hand. As such, when fine-tuning graph features such as title, x-axis labels, y-axis labels, it's common to revert to matplotlib commands.

Visualise the findings

```
In [10]: import pandas as pd
from sqlalchemy import create_engine, text
import seaborn as sns
import matplotlib.pyplot as plt

for field in review_fields:
    with engine.connect() as connection:

        # Define a custom color palette using a list of colors
        custom_palette = ["#FF595E", "#FFD4BF", "#FFBA2", "#484B48", "#767676"]

        sns.barplot(x='city', y=field, data=df, palette=custom_palette)
        plt.title(f"Average {field.replace('_', ' ').title()} by City")
        plt.ylabel("Average Score")
plt.show()
```

```
plt.figure(figsize=(8,6))
sns.barplot(x='city', y=field, data=df, palette=custom_palette)
plt.title(f"Average {field.replace('_', ' ').title()} by City")
plt.ylabel("Average Score")
plt.ylim(4.5, 5) #added this line to make the chart more "obvious"
plt.show()
```

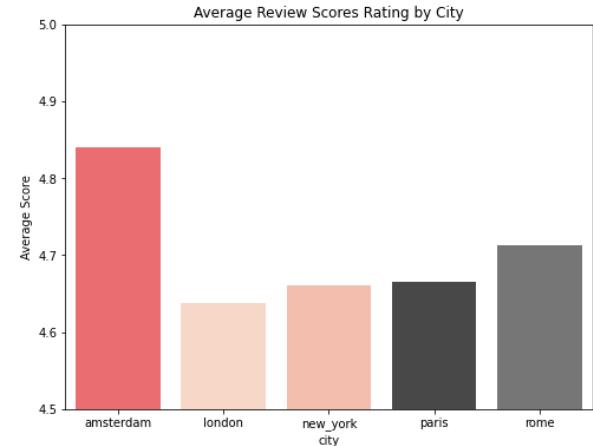
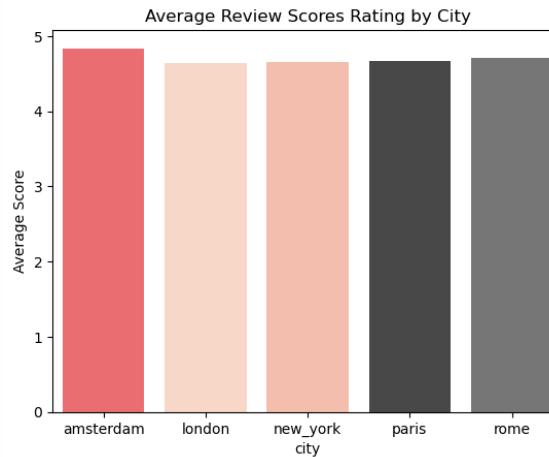


Tableau Public

Which homes are the
best rated?



Best-Rated Airbnb Homes

Property Type	Rating
Island	5.000
Private room in floor	5.000
Private room in island	5.000
Private room in religious building	5.000
Private room in treehouse	5.000
Shared room in guesthouse	5.000
Shared room in religious building	5.000
Yurt	5.000
Private room in camper/rv	4.960
Private room in minsu	4.960
Earthen home	4.935
Entire chalet	4.910
Tipi	4.910
Private room in hut	4.900
Houseboat	4.887
Farm stay	4.867
Shared room in townhouse	4.865
Private room in houseboat	4.860
Windmill	4.860
Private room in cottage	4.851
Private room in boat	4.837
Shared room in casa particular	4.830
Private room in guest suite	4.820
Private room in chalet	4.820

Tableau Public

How do the home
ratings differ by city?



Ratings by Host Location

Host Location	Barn	Boat	Camper/..	Campsite particular	Casa	Cave	Dome	Earthen home
Amsterdam..	4.890	4.829	4.835		5.000			
London, Uni..	4.960	4.563	5.000		4.960			4.870
New York, ..		4.840	4.773		4.800	4.500		
Paris, France	4.035	4.883	4.730		4.500	4.710		5.000
Rome, Italy		4.250	3.780		4.330	4.360	4.150	

Tableau Public

Do verified hosts and Superhosts get better reviews?



Superhost + Verified ID = Better Reviews?

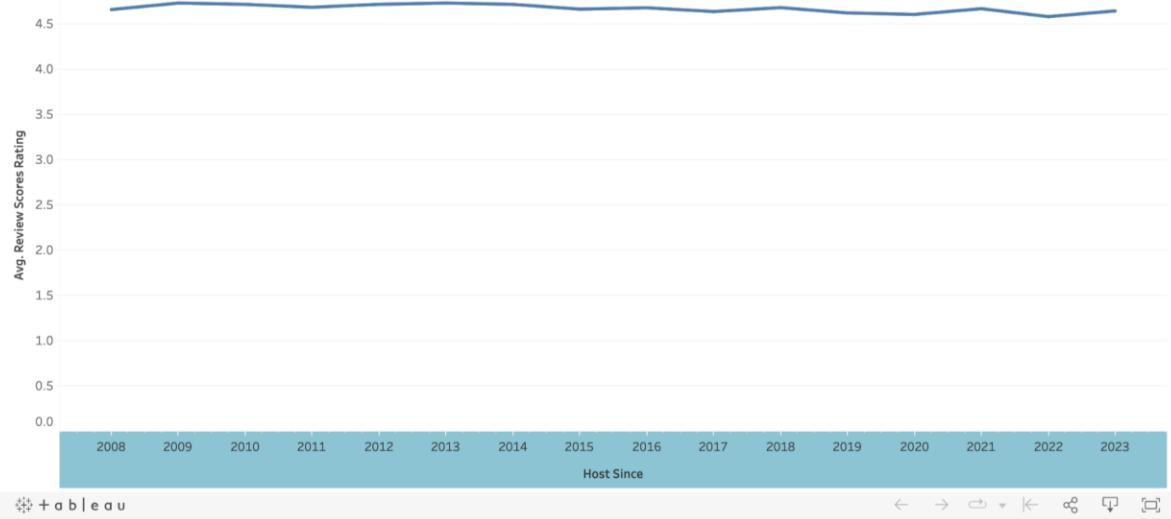
Host Identity	Not Superhost	Is Superhost
Verified	4.6261	4.8651
Unverified	4.4789	4.8546

Tableau Public

Do experienced hosts
get better reviews?



More Hosting Experience = Better Reviews?



Using ChatGPT: Example Queries

Identifying gaps in existing code:

E

```
sentiment_pipeline = pipeline("sentiment-analysis",
model="distilbert-base-uncased-finetuned-sst-2-english",
tokenizer="distilbert-base-uncased-finetuned-sst-2-
english")
tokenizer = DistilBertTokenizer.from_pretrained("distilbert-base-
uncased") #this feels like redundant code
```



You're right. If you're specifying the tokenizer directly in the '`pipeline`', then initializing it again separately is redundant.

Here's how you can simplify this part:

1. Use the initialized tokenizer from the pipeline itself.
2. Only initialize the pipeline once.



E in the code above, would it be possible that a comment is fetched twice?



Yes, in the code provided above, there is a possibility that the same comment can be fetched more than once. This could happen in the section of the code where you are compensating for long comments by fetching random comments:

Troubleshooting:



Token indices sequence length is longer than the specified maximum sequence length for this model (733 > 512). Running this sequence through the model will result in indexing errors # why am i still getting this message when i've specified a token length of 500 max?



The error message you're seeing is due to feeding tokens of length greater than 512 into the DistilBERT model, which has a maximum token length of 512.

There could be several reasons why you're still facing this issue:

1. **Improper Filtering:** Check if the filtering step that removes long comments is working correctly. There may be a problem in the '`is_comment_too_long`' function.

Clarifying unfamiliar code:



so even though it's "zip", it's actually split?



Yes, that's correct. The '`zip`' function is often used for combining multiple iterables into tuples, but when used with the '*' operator (known as "splat" or "unpacking" operator), it does the opposite. It "unzips" the tuples, effectively separating the elements of the tuples into individual sequences.

In this case:

```
python
    reviews_df['sentiment_score'] = zip(*sentiment_data)
```

06

Conclusion

Project Challenges + Lessons





Have fun doing whatever it is
that you desire to accomplish.

Learn it one step at a time,
emphasizing the fundamentals,
and do it because you love it,
not because it's work.



Paul Westphal (NBA Hall of Famer)