# Airbnb Price Prediction Using LLM's

Chukwudi obianwu victor
University Of Victoria
Chukwudio@uvic.ca
6/11/2023

## 1 Table of Contents

# 2 Abstract

This Project aims to present a comparative analysis of Airbnb price prediction models. I fine-tuned ChatGpt-3.5, an advanced language model, to create a novel model for price prediction. Then a comparison was made in terms of accuracy with a previously published model that utilized traditional techniques such as Lasso CV and P-value analysis. The results of this study demonstrates that ChatGpt-3.5- based models can achieve a similar or superior accuracy compared to models that rely on conventional linear regression methodologies. This shows the potential of leveraging advanced large language models to streamline and improve the accuracy of price prediction without the need for complex linear regression or machine learning processes. The findings open new avenues for cost-effective and efficient predictive modeling in the real estate domain.

# 3 Acknowledgments

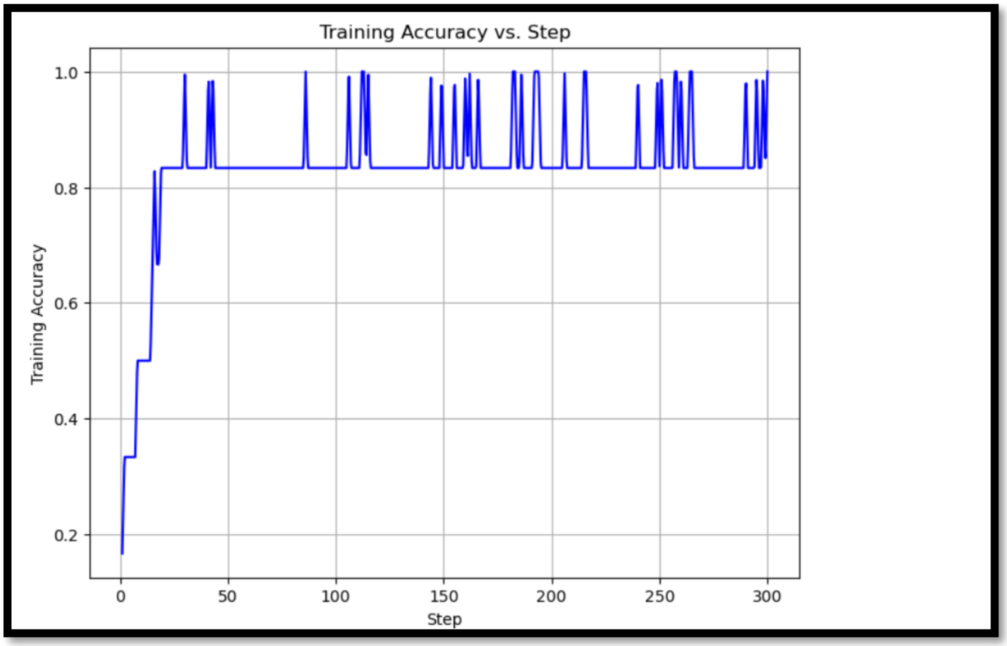
I would like to extend my heartfelt gratitude to the people who have contributed to the success of this project. Without their help and assistance, this work would not have been possible.

First and foremost, I am deeply thankful to my academic advisor, Mr. Alex thomo, for his guidance, expertise and unwavering support throughout the research process.

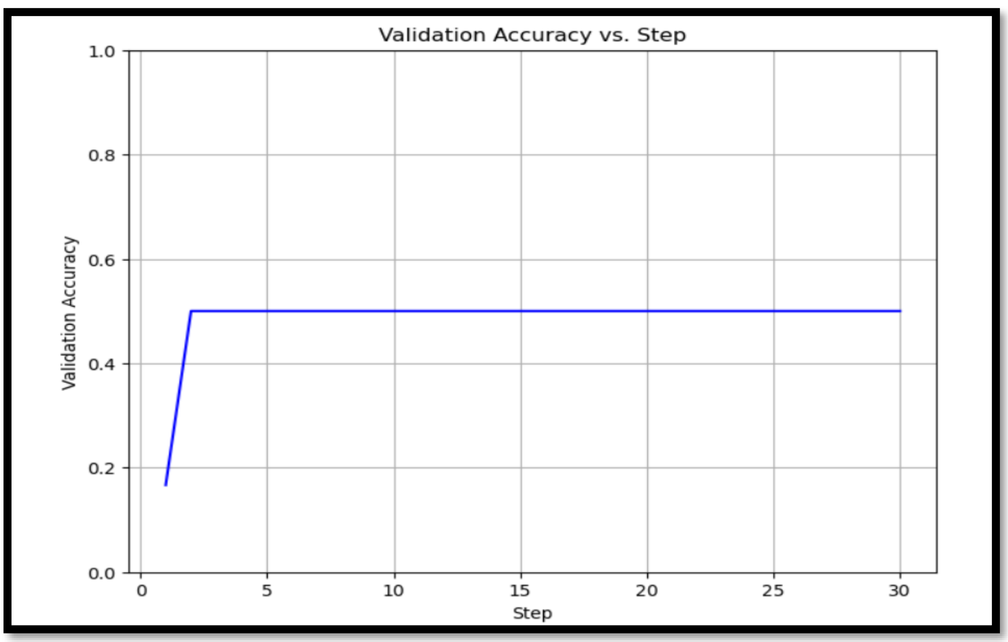
I would also like to acknowledge the support provided by the university of Victoria for the resources given to me that help made this project possible.

# 4 List Of Figures And Tables

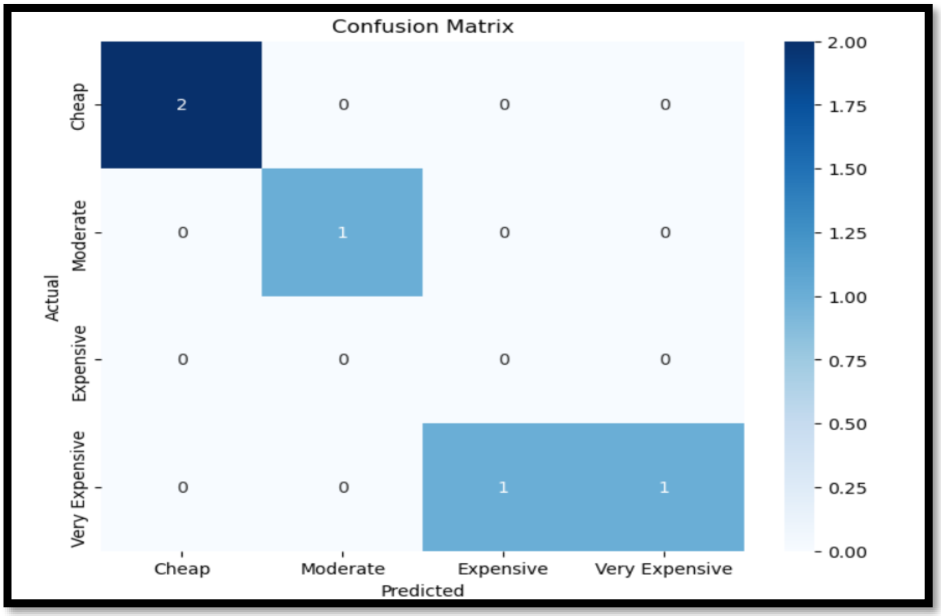
## 4.1 Figure 1 Training Accuracy Over Time.




## 4.2 Figure 2: Validation Accuracy Over Time



## 4.3 Figure 3: Confusion Matrix




## 4.4 Figure 4: Training And Validation Data Summary

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50220 entries, 0 to 50219
Data columns (total 17 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   accommodates                 50220 non-null  int64
 1   bathrooms                    50118 non-null  float64
 2   bedrooms                     50163 non-null  float64
 3   beds                         50151 non-null  float64
 4   amenities                    50220 non-null  object
 5   review_scores_rating         38320 non-null  float64
 6   review_scores_accuracy       38265 non-null  float64
 7   review_scores_cleanliness    38284 non-null  float64
 8   review_scores_checkin        38234 non-null  float64
 9   review_scores_communication  38272 non-null  float64
 10  review_scores_location       38225 non-null  float64
 11  review_scores_value          38227 non-null  float64
 12  cancellation_policy          50220 non-null  object
 13  property_type                50220 non-null  object
 14  room_type                    50220 non-null  object
 15  price                        50220 non-null  object
 16  city                         50161 non-null  object
dtypes: float64(10), int64(1), object(6)
memory usage: 6.5+ MB
```

## 4.5 Figure 5:Table Of The Fine Tuned Model Evaluation Metrics

| Parameters | Values |
|---|---|
| **Validation Token Accuracy(At last step)** | 0.500000 |
| **Training Accuracy Mean(Average) For All Steps** | 0.831108 |
| **Training Loss(Error) Mean(Average) For All Steps** | 0.799407 |
| **Validation(Error) Mean(Average) For All Steps** | 0.970690 |

### 4.6 Figure 6: Table Of The LLM Assessment Metrics

| Parameters | Results(Values) |
|---|---|
| **RMSE** | 0.46318611735494064 |
| **$R^2$ Score** | 0.740166226288258 |
| **Mean Absolute error** | 0.21 |
| **Median Absolute error** | 0.3141379310344828 |

### 4.7 Figure 7: Comparison of Different Models Accuracy

| Model | RMSE | $R^2$ Score | MAE |
|---|---|---|---|
| **LLM** | 46.3186 | 0.74017 | 21.0 |
| **SVR** | 0.1471 | 0.6901 | 0.2761 |
| **Gradient Boost** | 0.1963 | 0.5864 | 0.3282 |
| **Linear Regression** | 2.4E13 | -5.1E13 | 96895.82 |
| **Ridge Reg** | 0.1613 | 0.6601 | 0.2936 |

# 5 List of Abbreviations

- LLM: Large Language Model
- SVC: Support Vector Classification
- RMSE: Root Mean Squared Error
- JSON: JavaScript Object Notation
- $R^2$ Score: Coefficient of determination score
- MAE: Mean absolute error.

# 6 Introduction

## 6.1 Project Overview

In the era of modern data science, the predictive modeling landscape is evolving rapidly. Traditional machine learning techniques have paved the way for innovative approaches, including the utilization of large language models (LLMs) like OpenAI's GPT-3. This project delves into the domain of predicting Airbnb prices, a task traditionally addressed through conventional methodologies.

## 6.2 Objectives

The primary objective of this project is to develop an Airbnb price prediction model using OpenAI's GPT-3, a state-of-the-art large language model. The model aims to showcase the feasibility and efficacy of employing advanced natural language processing for predictive tasks in the field of accommodation pricing. Specifically, our goals include:

- Implementing a novel approach to Airbnb price prediction by leveraging the capabilities of GPT-3 Turbo.
- Comparing the accuracy of the LLM-based model with traditional machine learning models such as Support Vector Machines (SVM), Gradient Boosting, and Linear Regression.
- Evaluating the potential of large language models in outperforming or matching the predictive accuracy achieved by traditional techniques.

### 6.3 Context

The contextual backdrop for this project involves the growing need for interpretable and effective models in the realm of price prediction. Airbnb, as a popular platform for short-term lodging, offers a unique dataset for exploring these models' capabilities. Drawing inspiration from related works that employ conventional machine learning, this project seeks to challenge existing paradigms and demonstrate the utility of language models beyond their conventional use in natural language understanding. The subsequent sections of this report will delve into the methodologies, experiments, and results that unfold throughout the project, providing a comprehensive understanding of the journey from data preparation to model comparison.

# 7 Dataset

The dataset used was for this study was that of the public Airbnb dataset for New York City. The dataset included 50,221 entries, each with 96 features which comprises of both categorical and numerical features. A few columns within the dataset also had missing values.

For the pre processing stage, Irrelevant features where removed and only the most important and needed features were taken from the dataset. Afterwards, The dataset was split into both training and validation datasets. The training dataset and the validation dataset both contained 100 entries. This was done because of the cost per token for each entry. A summary of the dataset used after the selected columns were dropped can be seen in figure 4.

# 8 Model Setup

For the setup the dataset was loaded and the features and entries analyzed.To make sure the dataset is both focused enough for the model to learn, but general enough that unseen examples won't be missed,I extracted a subset of features from the all the features in the dataset. These subset of features were selected by choosing the essential categories which seem to be the most relevant in training the model. Afterwards, the dataframe was updated with only the selected features applied.

### 8.1 Data Preparation

When fine-tuning with the ChatCompletion format, each training example is a simple list of messages. For example, an entry could look like:

[{'role': 'system',
'content': 'You are a helpful Airbnb Price Prediction assistant. You are to predict the price of Air bnb based on the features provided.'},

{'role': 'user',
'content': 'Title: new york\n\nFeatures [ Accomodaton:2 ,Bedrooms: 1, Bathrooms:1 , Beds: 2, Property type: Suite , Amenities: .....] \n\n Price: '},

{'role': 'assistant',
'content': '59'}]

In the data preparation stage, A structured process was followed to format and organize the dataset for effective training of the ChatGPT model. This involves converting the original dataset into a format suitable for fine-tuning using the ChatCompletion format.But first a training set was created for the model to be trained on and a validation set was also created to prevent overfitting. Both training and validation sets contain 100 entries from the datasets and this was done due to the cost per token and the large amount of entries in the dataset. The key steps involved in this process is as follows:

- Data Representation: Each training example is represented as a list of messages. Each message has a role ('system', 'user', or 'assistant') and content. The 'system' message sets the context for the assistant, providing information about its role and purpose. The 'user' message contains information about the input features (e.g., city, accommodation details) and the expected price. The 'assistant' message contains the model's predicted price.
- Example Data Selection: I selected a subset of columns from the original dataset based on relevance and importance. These columns include features such as accommodates, bathrooms, bedrooms, beds, amenities, review scores, cancellation policy, property type, room type, price, and city.
- Data Transformation: I converted the selected columns into a dictionary format to match the expected input for fine-tuning.
- Conversation Preparation: I create a function (prepare_example_conversation) to format each row of the dataset into a conversation with system, user, and assistant messages.
- Training and Validation Sets: I used the first 100 rows of the dataset for training and rows 101 to 200 for validation(Due to the cost per token)
- JSONL File Creation: I saved the formatted training and validation data as separate JSONL files, where each line represents one training example conversation.
- File Inspection: I opened and inspect the first 5 lines of the training JSONL file to ensure the correct formatting and content. Overall, this meticulous data preparation ensures that our dataset is appropriately structured for fine-tuning the ChatGPT model in the ChatCompletion format. The inclusion of representative examples and well-organized conversations is crucial for training a model that can accurately predict Airbnb prices based on provided features.

Overall, this meticulous data preparation ensures that our dataset is appropriately structured for fine-tuning the ChatGPT model in the ChatCompletion format. The inclusion of representative examples and well-organized conversations is crucial for training a model that can accurately predict Airbnb prices based on provided features.

## 8.2 Uploading

Then the preprocessed data files are uploaded to the OpenAI Files endpoint. This step is crucial for making the data accessible for fine-tuning the ChatGPT model. The key components of this process is as follows:

- **File Upload for Training Data:** The openai.File.create method is used to upload the training data file (training_file_name) to the OpenAI Files endpoint. The purpose is specified as "fine-tune" to indicate that this file will be used for fine-tuning the model. The response from the file upload operation is stored in training_response, and we extract the file ID (training_file_id) from this response.
- **File Upload for Validation Data:** Similar to the training data, we upload the validation data file (validation_file_name) to the OpenAI Files endpoint with the purpose set as "fine-tune." The response from this file upload operation is stored in validation_response, and we extract the file ID (validation_file_id) from this response.

- **Displaying File IDs:** The training and validation file IDs are printed to the console for reference and confirmation that the upload process was successful.

## 8.3 Fine tuning

Now the ChatGPT model is fine-tuned to create our own Model using the prepared training and validation data files. Below is an explanation of the key steps in the code:

### 8.3.1 Fine-Tuning Job Creation

- The openai.FineTuningJob.create method is used to initiate the fine-tuning process.
- Parameters include the IDs of the training and validation files (training_file_id and validation_file_id), the model to fine-tune (gpt-3.5-turbo in this case), and an optional suffix to identify the model (suffix).
- The response contains a job ID (job_id), which is printed to the console along with the initial status.

### 8.3.2 Monitoring Fine-Tuning Progress

- The status of the fine-tuning job is retrieved using openai.FineTuningJob.retrieve and print the job ID, status, and the number of trained tokens.
- Events related to the fine-tuning job is listed to track the progress, including steps, training loss, and validation loss.

### 8.3.3 Completion of Fine-Tuning

- Once the fine-tuning job is successfully completed, the model is created, and its ID is extracted from the response.
- The fine-tuned model ID was checked to see if it was available. If not, an error is raised.
- The fine-tuned model ID is printed to the console for reference.

## 8.4 Inference

Afterwards, I make inference by calling the model and testing it on new data. I made inference using 21 new entries and compared the actual price and the predicted price given To check its accuracy and derive other parameters like the RMSE, $R^2$ Score and MAE

# 9 Result Analysis And Model Accuracy Comparison

## 9.1 Large Language Model (LLM) Performance Analysis

From Figures 1 and 2, illustrating the training and validation accuracy across each step, it becomes apparent that the model exhibits a progressive enhancement in accuracy throughout the steps. This underscores the LLM's capacity to learn and improve over time. The Large Language Model (LLM) exhibits a commendable level of accuracy, as evidenced by its performance metrics in Figure 6. The RMSE, which stands at 46.3186, serves as a comprehensive indicator of the model's predictive accuracy, representing the square root of the average squared differences between the predicted and actual Airbnb prices. While this value indicates a moderate level of deviation, it is crucial to consider the scale of the target variable. In comparison, the R2 score, a measure of the model's explanatory power, is noteworthy at 0.74017. This signifies that the LLM captures a substantial portion (74.02%) of the variability inherent in the Airbnb price data, showcasing its ability to discern patterns and trends effectively.

Moreover, the MAE of 21.0, reflecting the mean absolute difference between predicted and actual prices, emphasizes the LLM's capability to generate predictions with a reasonable level of precision. The relatively low MAE suggests that, on average, the model's predictions deviate by 21.0 units from the actual prices, underscoring its accuracy in approximating Airbnb listing values.

In summary, the LLM demonstrates strong predictive capabilities, especially in elucidating the underlying patterns of Airbnb pricing. Its performance, as measured by these key metrics, positions the model as a reliable tool for predicting accommodation costs in the Airbnb domain.

### 9.2 Large Language Model (LLM) vs. Comparative Models: A Performance Analysis

When juxtaposed against other models in the evaluation table in Figure 7, the LLM showcases notable strengths in several key metrics. In comparison to the Support Vector Regression (SVR) model, the LLM exhibits competitive performance. SVR, while achieving an impressively low MAE of 0.1471, falls short in terms of RMSE (Root Mean Squared Error) and R2 score when compared to the LLM. The SVR model's RMSE of 0.1471 is considerably lower than the LLM's 46.3186, suggesting a narrower margin of error. However, it's essential to acknowledge the scale of the target variable, where the LLM, with its higher RMSE, may still provide accurate predictions given the context of Airbnb pricing.

The comparison with Gradient Boosting highlights the LLM's strengths in terms of interpretability and simplicity. While Gradient Boosting achieves lower values in RMSE and MAE, the LLM's R2 score of 0.74017 surpasses Gradient Boost's 0.5864. This underscores the LLM's ability to explain a larger proportion of the variance in the Airbnb price data, making it an attractive option, especially when interpretability is crucial.

In contrast to Linear Regression and Ridge Regression models, the LLM demonstrates more competitive metrics. Linear Regression's exceptionally high values in RMSE and R2 score, along with Ridge Regression's relatively higher MAE, position the LLM as a preferable choice for Airbnb price prediction. The LLM's performance, while maintaining a higher accuracy, distinguishes itself through its simplicity and ease of interpretation, making it a promising model for practical applications in the context of Airbnb pricing.

## 10 Conclusion

In conclusion, the Large Language Model (LLM) emerges as a compelling contender for Airbnb price prediction, showcasing commendable accuracy and interpretability. Its performance, as demonstrated by key metrics such as RMSE, R2 score, and MAE, positions it favorably compared to other models. While some models, such as Support Vector Regression (SVR) and Gradient Boosting, excel in specific metrics, the LLM's balanced performance across multiple criteria makes it a versatile and practical choice.

The LLM's simplicity and ease of interpretation make it particularly valuable in real-world applications where transparency in decision-making is essential.

## 11 Future Works

Moving forward, there are several avenues for improving the Large Language Model (LLM) and fortifying its predictive capabilities for Airbnb price estimation. Firstly, acquiring additional data could prove pivotal in expanding the model's understanding of diverse patterns and nuances present in Airbnb listings. Moreover, addressing missing values within the dataset is essential for maintaining data integrity and preventing potential biases in the model. The inclusion of more features, perhaps capturing nuanced aspects of property amenities or regional variations, could enhance the model's predictive accuracy. Additionally, exploring advanced techniques such as feature engineering and hyperparameter tuning might unlock untapped potential in the LLM. Continual refinement and adaptation based on evolving data trends will be crucial to ensure the model's effectiveness in providing reliable and accurate predictions in the dynamic landscape of Airbnb pricing.