New Jersey Institute of Technology
CS 700B
Effects of Covid-19 on NYC Real Estate Market

Guide: Prof. Dr. Guiling Wang
PhD Student Advisor: Junyi Ye
Student: Ketki Ambekar

Abstract

In this paper, we study effects of covid 19 on New York City real estate prices. Real estate markets normally follow seasonal patterns. However, certain rare events tend to have huge impacts on the prices. The covid-19 pandemic, a rare event not seen for at-least a century, has had some interesting effects on the NYC real-estate market. We study the sources, characteristics and feature engineering of the real estate transactions' data. We identify the right features to work with among numerous available features. We visualize the data to spot patterns. We then apply deep learning and time series forecasting methods to create a model that predicts average market rates.

1 Introduction

New York City is among the top 10 most expensive real estate markets in the world. However, during the covid-19 pandemic some interesting trends were observed. The city observed a massive decline in the market size, losing billions of dollars in tax revenue[2]. Housing prices are driven by numerous factors like market imperfections, credit market frictions, or even irrationality. [5]

2 Exploratory Data Analysis

2.1 The Data Sources

The data used in the project was sourced from the following two sources:

1. **NYC - Property Rolling Sales Data:** This data is published by the Department of Finance of NYC. It lists data for last 12 months, but data from since 2003 is available when they first started collecting it. The data describes the tax classes, the neighborhood type, building type, square footage, among other data. The files available are in MS Excel spreadsheet format. There is a separate file for each of the five NYC boroughs (Manhattan, Bronx, Brooklyn, Queens and Staten Island). Consists of 21 Features describing the location and different features of properties such as tax brackets, building classes, etc. A glossary is also available that describes the unique terms.

2. **Pluto Data set:** This dataset is made available by the Department of City Planning, NYC. Its records geographic data about land usage and tax level information about the same. It consists of approximately a hundred features.

The dataset used for this project was created by a former student by combining the number of Covid Cases per day, NYC Property Rolling Sales Data and the Pluto Dataset The final dataset contained 77 Columns and 124,537 rows. Further adjustments were made to make the data usable as described subsequently.

2.2 Data Distribution

Using visualization techniques, we were able to understand the data better by looking at the data distributions.

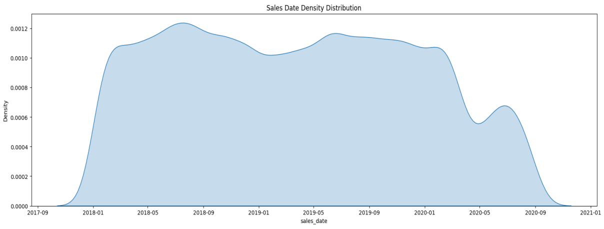


Figure 1: Date Wise Distribution: Data was between the range September 2017 to October 2020

From Figure 2. we see that the approximate mean sales price is one million dollars. Some outlying sales transactions are priced at 16 million dollars. And oddly, some transactions are valued at

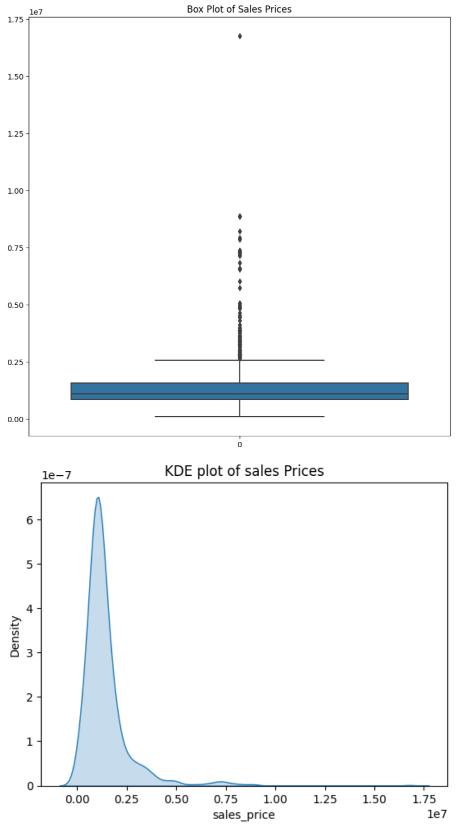


Figure 2: Sales Price Analysis

zero dollars. The data set glossary revealed that $0 transactions indicated transfer of ownership of the property (e.g. from parents to children). On further consultation with a real estate expert (with help from prof. Dr. Wang), it was found that it is quite uncommon in NYC for property sale transactions to be priced below $100,000. These rows might've indicated rent or token money transactions. We therefore dropped such rows (23,849 in number) whose sales price value was under $100,000

2.3 Missing Values

Missing values in the dataset could be classified into two types: 1) Data exists but is not recorded. 2) Data does not exist (e.g. garage area for apartments not having a garage).

Because we had good amount of data, missing values were handled by removing rows with missing values. Out of 124,537 rows in the original dataset, 23,846 rows had missing values.

After removing the anomalous and missing values, our dataset finally had 76,842 rows.

2.4 Visualizing the Data

Figure 3 shows a line plot of covid cases and average sales price in NYC spanning the dataset time frame. The Data (Salers Price and Covid Cases) are scaled between 0 and 1 using MinMaxScaler otherwise it would be difficult to fit them onto a single plot. The scaled data was then used to create a Line Plot using the Matplotlib library. (Figure 3). Noticeable absence of large peaks during peak covid times is indicative of a lull in the real estate sales transactions. The market bounced back up around the months of June 2020 to September 2020, which shows decline in the number of covid cases.

3 Feature Engineering

The aim of feature engineering to determine the most useful and pertinent features to be used in a predictive model. Our final dataset consisted of 77 columns. Some of which are as shown in Figure 4.

We created a correlation matrix to examine effects of columns on the 'sales price' target. From the same, we found that 'area' related columns had a major impact on sales price. We therefore, created a new feature called 'total area' that combined nine types of areas in the dataset namely lotarea, bldgarea, comarea, resarea, officearea, retailarea, garagearea, strgearea and Factryarea.

Further along, the feature 'total area' was converted to 'per sqft rate' by performing:

$$Per\ sqft\ rate = \frac{Sales\ Price}{Total\ Area} \quad (1)$$

This made for a better predictive feature.

4 Predictive Modelling

We created a predictive time series forecast model to predict the per square feet rate in NYC for x number of days in the future (say a week). This is a multivariate model that takes per sq ft rate and the number of covid cases as input and predicts the per sqft rate for couple of weeks (adjustable window) in the future.

4.1 Time series data: Characteristics and Challenges

Our data is a time series as it represents transactions taking place on a daily basis along with number of covid-19 cases on that given date. Time series are special datasets having certain characteristics that distinguish them from normal datasets. One major characteristic is that ordering of the data points is important in time series. Time series may reveal trends and seasonality in data which can help in predictions.

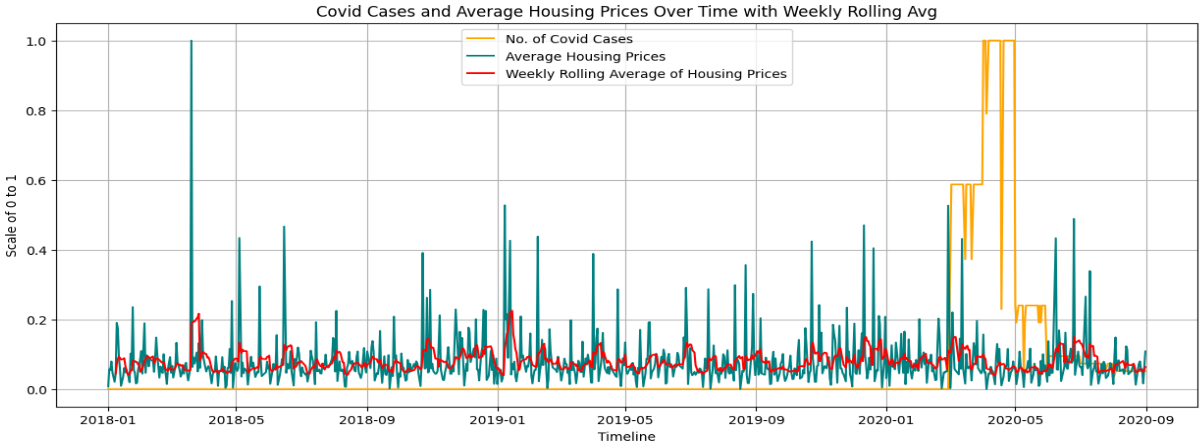
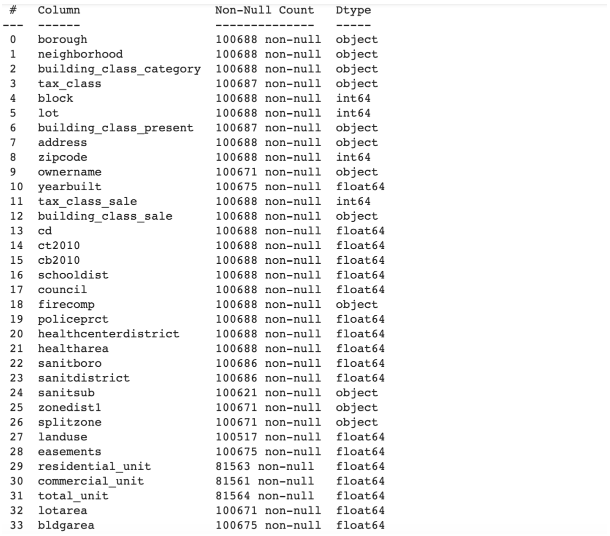


Figure 3: Covid cases and Average Housing Prices over Time with Weekly Rolling Average



```
 #   Column                   Non-Null Count   Dtype
---  ------                   --------------   -----
 0   borough                  100688 non-null  object
 1   neighborhood             100688 non-null  object
 2   building_class_category  100688 non-null  object
 3   tax_class                100687 non-null  object
 4   block                    100688 non-null  int64
 5   lot                      100688 non-null  int64
 6   building_class_present   100687 non-null  object
 7   address                  100688 non-null  object
 8   zipcode                  100688 non-null  int64
 9   ownername                100671 non-null  object
 10  yearbuilt                100675 non-null  float64
 11  tax_class_sale           100688 non-null  int64
 12  building_class_sale      100688 non-null  object
 13  cd                       100688 non-null  float64
 14  ct2010                   100688 non-null  float64
 15  cb2010                   100688 non-null  float64
 16  schooldist               100688 non-null  float64
 17  council                  100688 non-null  float64
 18  firecomp                 100688 non-null  object
 19  policeprct               100688 non-null  float64
 20  healthcenterdistrict     100688 non-null  float64
 21  healtharea               100688 non-null  float64
 22  sanitboro                100686 non-null  float64
 23  sanitdistrict            100686 non-null  float64
 24  sanitsub                 100621 non-null  object
 25  zonedist1                100671 non-null  object
 26  splitzone                100671 non-null  object
 27  landuse                  100517 non-null  float64
 28  easements                100675 non-null  float64
 29  residential_unit         81563 non-null   float64
 30  commercial_unit          81561 non-null   float64
 31  total_unit               81564 non-null   float64
 32  lotarea                  100671 non-null  float64
 33  bldgarea                 100675 non-null  float64
```

Figure 4: Some of the features in our dataset

Generally, time series have limited amount of data points as compared to a standard ML, NLP problems, as daily collected data over a year would result in only about 365 data points. Traditional train/ test split and cross validation wouldn't work because of this. Time series are also characterized by high amount of uncertainty, as it is improbable that the predicted price will be exactly the actual price on a given date. Also, if more and more people rely on the forecasts from a given predictive model, for say real estate or stock market data, it might affect people's buying and selling decisions and influence the market in the process, therefore rendering the model ineffective.

The biggest challenge with time series is that we have to re-train the model, every time we need a new prediction. This is in contrast to continuous learning for some common ML models, where an already trained model is updated as new data comes in.

Some of the common time series forecasting techniques include Auto Regression (AR), Moving Average (MA), Simple Exponential Smoothing (SES), Auto Regressive Integrated Moving Average (ARIMA), and more recently, Neural Networks and LSTMs are used.

4.2 Preparing Time Series Data

Our raw data contained records of all transactions took place in NYC in period from Jan 2018 to Sept 2020. However, it is required that time series data have only one data point per date. We therefore needed to process our data to achieve this.

We used the 'groupby' function available in pandas library and applied it to the date column. Thus, we grouped our data by date, averaged all values of 'per sqft rate' against a particular date and selected the maximum number of covid cases against the same date. Choosing maximum number of covid cases wasn't strictly required, as all values of covid cases on a given date would be same, but this was just a redundant safeguard, in case of future anomalous data.

After this process, our data had one record for each date.

4.3 Converting time series data to supervised learning data

Time series data must be transformed before it can be used in a supervised learning model. Generally, a time series is in a format of, say, a pandas data frame with date column as its index and number of dates being equal to number of samples.

In order for us to be able to use this for a supervised learning model, the data needs to be converted in the following format:

Train_X.shape ⇒ [samples, time_steps, features]
Train_Y.shape ⇒ [samples, output_features]
where,

- **Samples**. One sequence is one sample. A batch is comprised of one or more samples.
- **Time Steps**. One time step is one point of observation in the sample. One sample is comprised of multiple time steps.
- **Features**. One feature is one observation at a time step. One time step is comprised of one or more features.

4.4 Autoregressive Integrated Moving Average (ARIMA)

A stationary time series is one where its statistical properties such as mean, variance, etc. do not change over time. A non-stationary time series in contrast, is one whose aforementioned properties change over time. In ARIMA models a non-stationary time series is made stationary by applying finite differencing of the data points. ARIMA models are generally denoted as ARIMA(p,d,q) where parameters p, d, and q are non-negative integers, p is the order (number of time lags) of the autoregressive model, d is the degree of differencing (the number of times the data have had past values subtracted), and q is the order of the moving-average model[7][4].

The statsmodels library was used to predict per sqft rate using its ARIMA module. The (p,d,g) values were chosen to be (5,1,0). The evaluation metric of the model was Root Mean Square Error a Test RMSE of 50.942 was obtained for our model. The forecast plots can be seen in figure 5.

4.5 LSTM

Traditional neural networks can't persist information. Recurrent neural networks address this issue. They are networks with loops in them, allowing information to persist. Figure 6[6] pictorially represents a unrolled RNN. "LSTMs" – are a special kind of RNN, capable of learning long-term dependencies.

LSTMs achieve this with a memory cell unit designed for the purpose of retaining information.

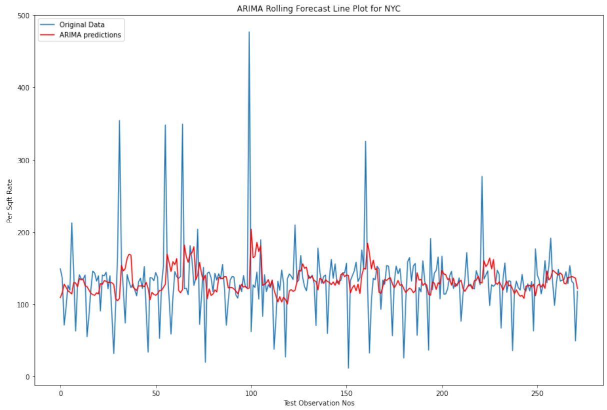


Figure 5: ARIMA forecast with Root Mean Square Error: 50.942

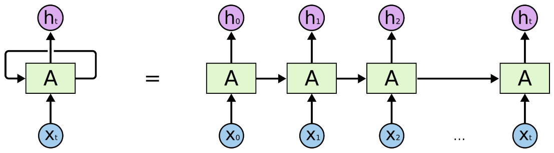


Figure 6: An unrolled recurrent neural network.

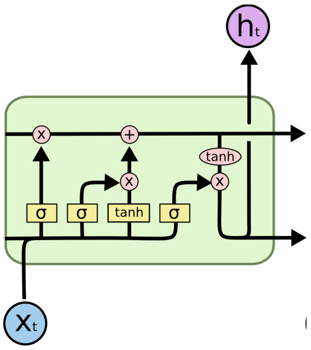


Figure 7: The repeating module in an LSTM containing four interacting layers (yellow blocks).

Gates are a combination of a sigmoid/tanh layer and a point-wise operation gate. Depending on weights chosen, they can optionally let information through.[6]. The memory unit consists of three gates:

1. **Forget Gate**: This gate decides what information we're going to throw away from the cell state. A sigmoid layer acts as a forget gate layer. It looks at h_{t-1} and x_t, and outputs a number between 0 and 1 for each number in the cell state C_{t-1}. A 1 acts for completely retaining the info while a 0 acts for completely deleting the info.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t]) + b_f) \tag{2}$$

2. **Input Gate**: The Input gate helps determine the info to be retained in the cell. This is achieved by using a sigmoid layer to decide the values to be updated and a tanh layer that creates new candidate values C_t to be added to the state. These are then multiplied pointwise.

$$i_t = (W_i \cdot [h_{t-1}, x_t] + b_c) \tag{3}$$

$$C_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{4}$$

Next, the input gate takes the values from the forget gate and the new cell state values C_t and actually updates the cell state with them. To forget the value, it multiplies the old cell state with f_t. It then adds the new candidate values $i_t * C_t$

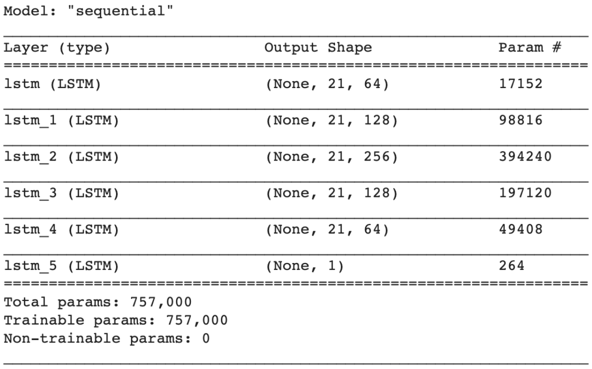
$$C_t = f_t * C_{t-1} + i_t * C_t \tag{5}$$

3. **Output Gate**: The final sigmoid layer helps us decide which of the cell state values are to be output. We apply tanh to the output values so as to compress them between -1 and 1 and sigmoid further helps us select the pertinent parts to output.

$$o_t = (W_o \cdot [h_{t-1}, x_t] + b_o) \tag{6}$$

$$h_t = tanh(C_t) * o_t \tag{7}$$

In other words, time_steps is the window size we want to use to prepare our data for supervised learning.



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 21, 64)            17152
_________________________________________________________________
lstm_1 (LSTM)                (None, 21, 128)           98816
_________________________________________________________________
lstm_2 (LSTM)                (None, 21, 256)           394240
_________________________________________________________________
lstm_3 (LSTM)                (None, 21, 128)           197120
_________________________________________________________________
lstm_4 (LSTM)                (None, 21, 64)            49408
_________________________________________________________________
lstm_5 (LSTM)                (None, 1)                 264
=================================================================
Total params: 757,000
Trainable params: 757,000
Non-trainable params: 0
_________________________________________________________________
```

Figure 8: Keras LSTM model

4.6 Implementation

We use the keras implementation of LSTM models for our implementation.

Some of the hyper parameters are:
Epochs: 1000
batch_size:70
Validation Split:0.3
Loss function: Mean Squared Error (mse)
Optimizer Function: Adam

Because it is important that time series data should be ordered, we added an extra parameter: 'shuffle=False'.

The LSTM model had Mean Squared Error of 0.3725 (training loss) and 3.0848 (validation loss). The results are comparable to those obtained by Peng [1]

4.7 ARIMA versus LSTM

Between the two, ARIMA provided a more stable output and captured the variability better than the LSTM model. LSTM and machine learning models in general have better results for more complex data with more variables [3]. The line plots 5 and 9 reflect this observation well. The model can be used to predict prices over next few weeks in order to make decision to perhaps wait to sell the property if a price drop is predicted, or delay buying property if price climb is predicted.

Evaluation Metrics:
The LSTM model had Mean Squared Error of 0.3725 (training loss) and 3.0848 (validation loss). Same can be visualized in the part 1 of Figure 9. The ARIMA model achieved a Root Mean Squared Error (RMSE) of 50.942 on the test set.

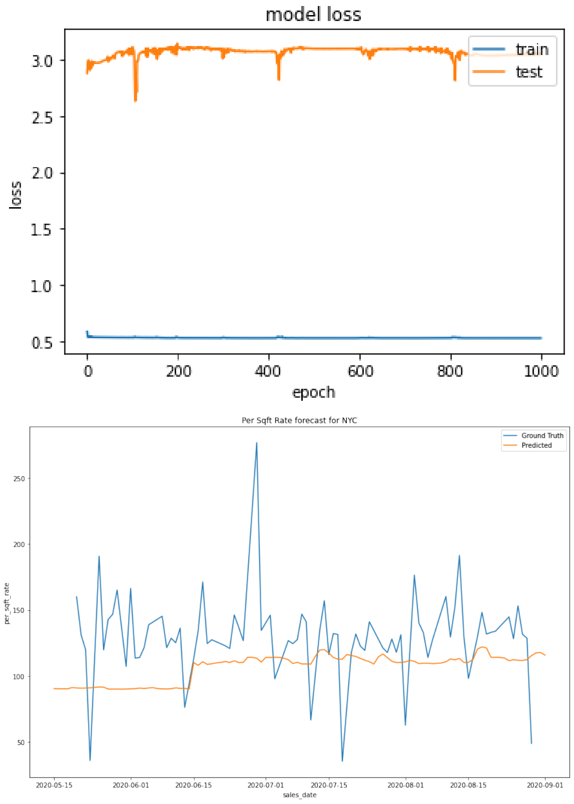


Figure 9: Train and Validation Loss (top). LSTM forecast (bottom)

5 Conclusion

We saw from our exercise in data visualization that there was a distinct dip in the average housing prices during peak season of covid cases. From this project we were able to build a predictive model to predict per sq.ft prices using covid cases as input.

6 Future Work

The multivariate LSTM could be expanded to include more features so as to make a more complex model that can consequently make better predictions. Unseeming variables could be studied for their impact on the real estate price fluctuations. (e.g. Temperature).

References

[1] H. Peng et al. "Lifelong Property Price Prediction: A Case Study for the Toronto Real Estate Market". In: (2020), p. 10. URL: https://arxiv.org/abs/2008.05880.

[2] Bloomberg. "NYC's Plummeting Real Estate Sales Cost City $1.2 Billion." In: *National real estate investor* (2020).

[3] J. BrownLee. *Deep Learning for Time Series Forecasting*. Machine Learning Mastery, 2018, p. 4.

[4] A.I. McLeod K.W. Hipel. "Time Series Modelling of Water Resources and Environmental Systems". In: *Elsevier* (1994).

[5] J. A. Kahn. "What Drives Housing Prices?" In: *FRB of New York Staff Report No. 345* (2008). URL: http://dx.doi.org/10.2139/ssrn.1264048.

[6] C. Olah. "Understanding LSTM Networks". In: *colah.github.io* (2015).

[7] J. Flaherty R. Lombardo. "Modelling Private New Housing Starts In Australia". In: *Pacific-Rim Real Estate Society Conference, University of Technology Sydney (UTS)* (2000).