



HDB RE-SALE PRICING PREDICTION

WHAT DRIVES HDB PRICES?

---

# STRATEGY

- By applying Machine learning to predict HDB prices,
- we can utilize ML visualizations to see which features are used to do the prediction
- and hence ... proxy these features as a driver for HDB pricing.

# SLIDESHOW OVERVIEW

- 1) PROBLEM STATEMENT
- 2) DATA OVERVIEW
- 3) DATA CLEANING, FEATURE ENGINEERING. EDA
- 4) AUTO-ML, HYPER-PARAMETER TUNING, ML Visualizations
- 5) Challenges & Thoughts

# PROBLEM STATEMENT

---

- Understand the drivers of prices of the houses
  - We will attempt to understand this via:
    - Exploratory/general data analysis (EDA)
    - as well as machine learning and the analysis of how the machine read the information.

# DATA OVERVIEW

- Given 5 different sets of data, I've combined them into:

## looks about right lets join them

```
: In [15]: new_df = pd.concat([a,b[a.columns],c[a.columns],d[a.columns],e[a.columns]]).reset_index().drop(columns=["index"])
          new_df
```

[15]:

	month	town	flat_type	block	street_name	storey_range	floor_area_sqm	flat_model	lease_commence_date	resale_price	Month	Year
0	1990-01	ANG MO KIO	1 ROOM	309	ANG MO KIO AVE 1	10 TO 12	31.0	IMPROVED	1977	9000.0	1	1990
1	1990-01	ANG MO KIO	1 ROOM	309	ANG MO KIO AVE 1	04 TO 06	31.0	IMPROVED	1977	6000.0	1	1990
2	1990-01	ANG MO KIO	1 ROOM	309	ANG MO KIO AVE 1	10 TO 12	31.0	IMPROVED	1977	8000.0	1	1990
3	1990-01	ANG MO KIO	1 ROOM	309	ANG MO KIO AVE 1	07 TO 09	31.0	IMPROVED	1977	6000.0	1	1990
4	1990-01	ANG MO KIO	3 ROOM	216	ANG MO KIO AVE 1	04 TO 06	73.0	NEW GENERATION	1976	47200.0	1	1990

851294 rows × 13 columns

# DATA CLEANING

---

- **Blocks** have a **letter appended** at the back EG. BLK 333A, 333B, etc.
  - Removed them without amending the block#
- **flat\_type** -> has 'MULTI GENERATION' & 'MULTI-GENERATION' -> convert them to just one kind
- **Flat\_model** -> changing everything to caps to remove extra categoriues of data
- No null/empty values which is great
- No duplicated values either (Similar values are normal since more than 1 kind of a room will be available eg. 3 rooms in CCK will have a similar data format)

# DATA CLEANING – DROPPING COLUMNS

---


- "Street" since the values too unique even after dropping the town name from the street name I had 434 values
- "Month" since I have successfully splitted it already

# FEATURE ENGINEERING

- Column "Month" consist of month and year

- NEW FEATURES


month	
1990-01	
1990-01	
1990-01	
1990-01	
1990-01	



Month	Year
1	1990
1	1990
1	1990
1	1990
1	1990

- Column town narrowed further into regions

town	
ANG	
MO KIO	
ANG	
MO KIO	
ANG	
MO KIO	
ANG	
MO KIO	



Region
NORTH-EAST
NORTH-EAST
NORTH-EAST
NORTH-EAST
NORTH-EAST



# FEATURE ENGINEERING:

## ENCODING STRATEGIES

- Label encoding for columns like:

- Flat\_type, storey range, region

```
['1 ROOM',  
'2 ROOM',  
'3 ROOM',  
'4 ROOM',  
'5 ROOM',  
'MULTI GENERATION',  
'EXECUTIVE']
```

0  
1  
2  
3  
4  
5  
6

```
'01 TO 03', '01 TO 05', '04 TO 06', '06 TO 10', '07 TO 09',  
'10 TO 12', '11 TO 15', '13 TO 15', '16 TO 18', '16 TO 20',  
'19 TO 21', '21 TO 25', '22 TO 24', '25 TO 27', '26 TO 30',  
'28 TO 30', '31 TO 33', '31 TO 35', '34 TO 36', '36 TO 40',  
'37 TO 39', '40 TO 42', '43 TO 45', '46 TO 48', '49 TO 51']
```

- Binary encoding (Hash + one hot mix):

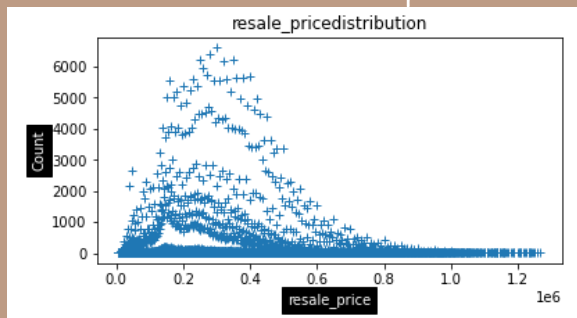
- Town (27 values) & flat\_model (20 values)

town
ANG
MO KIO
ANG
MO KIO
ANG
MO KIO
ANG
MO KIO

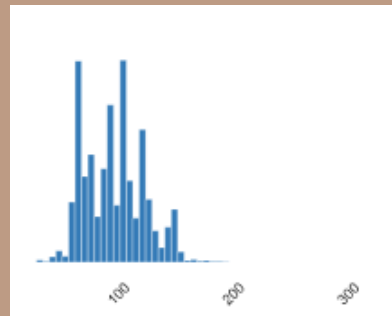
	town_0	town_1	town_2	town_3	town_4	town_5
0	0	0	0	0	0	1
1	0	0	0	0	0	1
2	0	0	0	0	0	1
3	0	0	0	0	0	1
4	0	0	0	0	0	1

# EDA - BRIEF, NOTHING TOO EYE CATCHING...

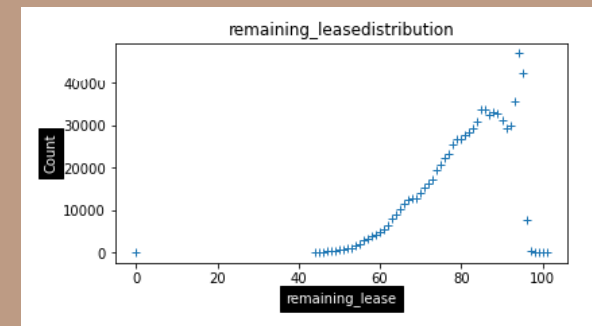
- Left skew on resale price



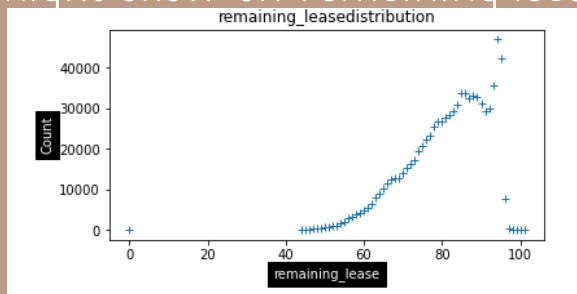
- Left skew on floor area



- Right skew on floor



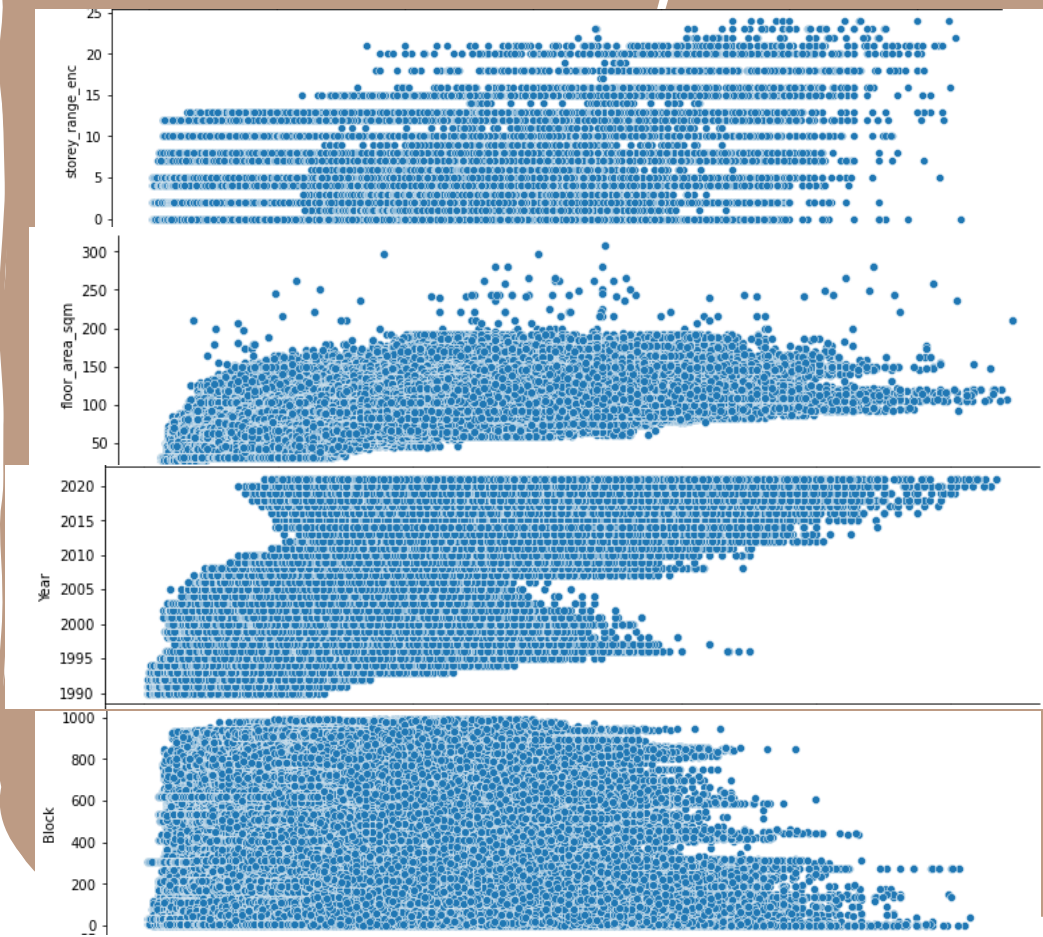
- Right skew on remaining lease



Strategies:

- 1) Normalize data to give a mean of 1 and SD of 0 to "center" the data
- 2) pick bootstrapping models which bypass severe outliers and overtly skewed data (I picked the random forest)

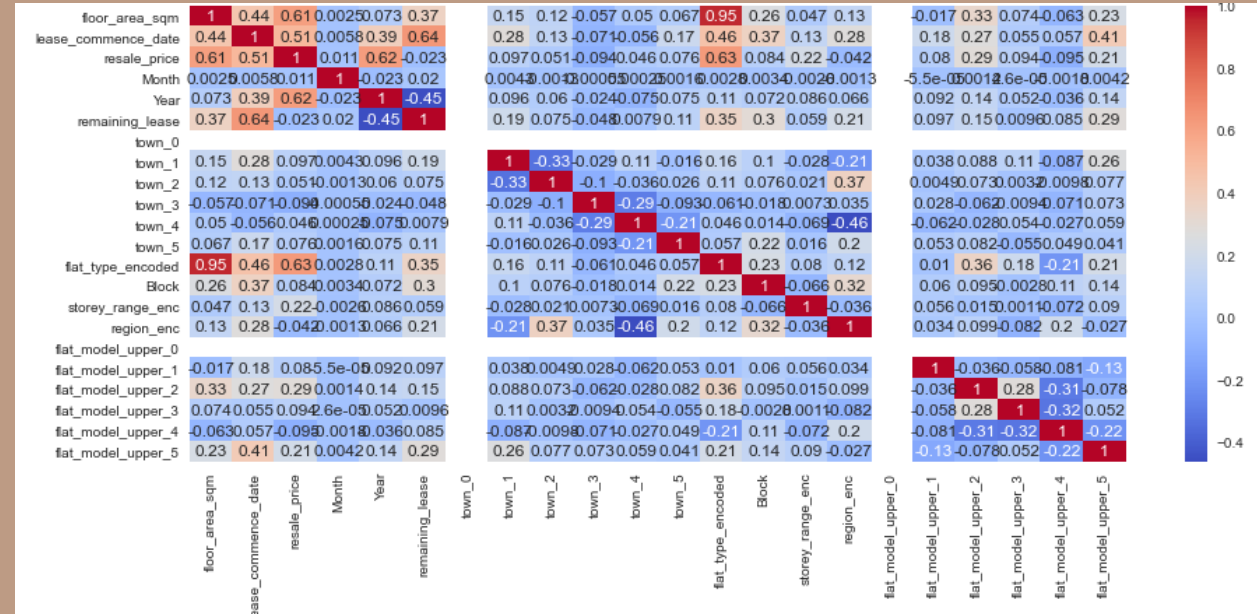
# EDA - BRIEF, NICE TO KNOWS



- Typically higher floors have a higher price floor
- The lesser floor area you have the lower the price ceiling
- The later your building was created the more likely your re-sale price is going to be good.
- And unsurprisingly the price ceiling for low block no.s (Terraces etc.) is rather high

# EDA - BRIEF, NOTHING TOO EYE CATCHING...

- Nothing really correlates too much with resale price either, only moderately at 0.6/1.0 for floor area & 0.5 for year of construction.

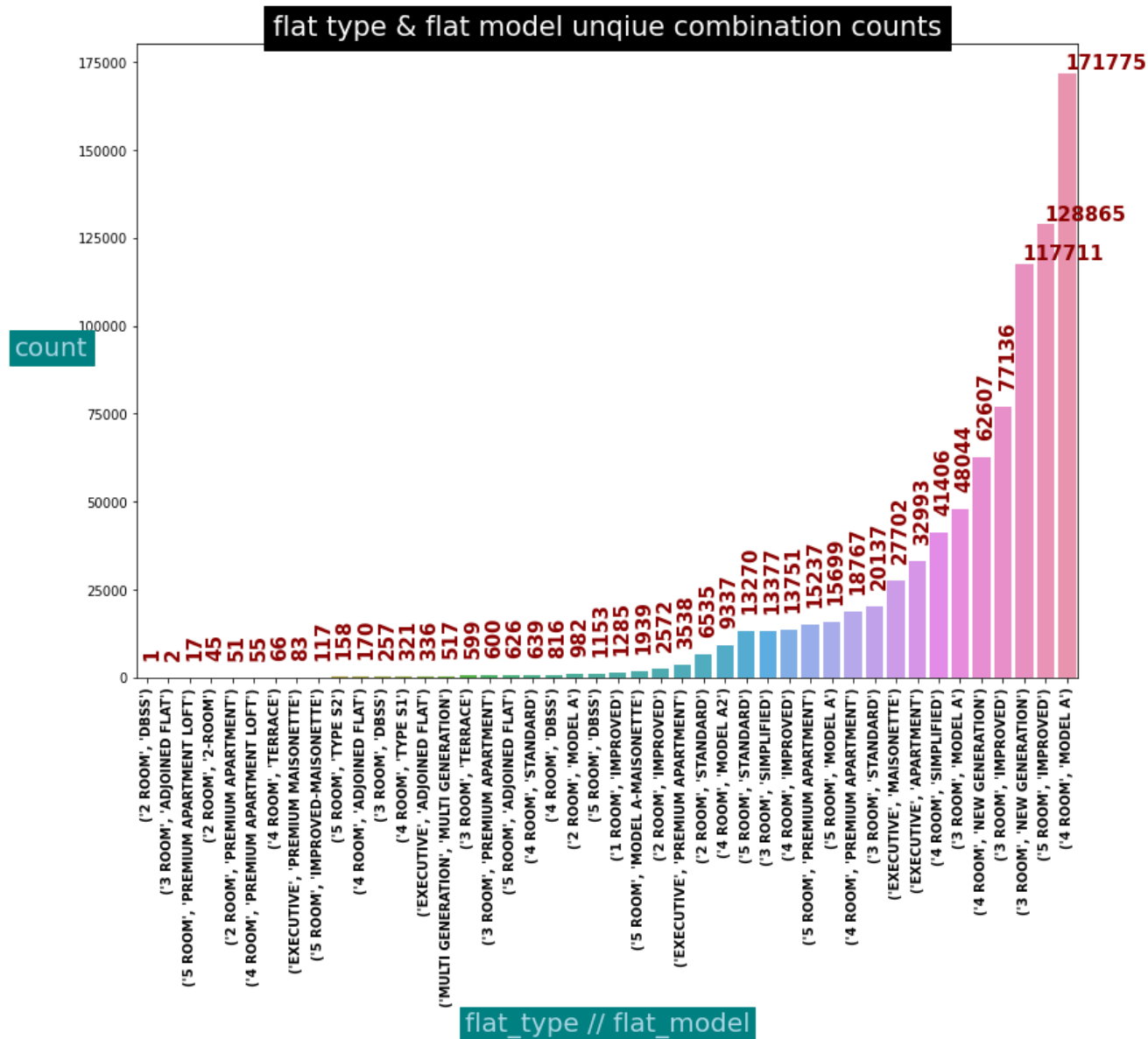


# EDA -

## POSSIBLE DATA EXCLUSIONS

Very sparse data for certain combinations of flat\_type & flat\_model

Possibly dropping combinations with less than 1000 instances each might improve the model



# AUTO-ML - (HYPEROPT)

```
model = HyperoptEstimator(regressor=random_forest_regression('rf'), preprocessing=auto, algo=tpe.suggest, max_evals=30, trial_timeout=30)
model.fit(x_train_scale1.to_numpy(), y_train.to_numpy())
```

```
100%|██████████| 1/1 [00:05<00:00, 5.38s/trial, best loss: 0.539114586848605]
100%|██████████| 2/2 [00:27<00:00, 27.95s/trial, best loss: 0.45965640809331465]
100%|██████████| 3/3 [00:32<00:00, 32.40s/trial, best loss: 0.45965640809331465]
100%|██████████| 4/4 [00:32<00:00, 32.63s/trial, best loss: 0.45965640809331465]
100%|██████████| 5/5 [00:32<00:00, 32.56s/trial, best loss: 0.45965640809331465]
100%|██████████| 6/6 [00:32<00:00, 32.57s/trial, best loss: 0.45965640809331465]
100%|██████████| 7/7 [00:32<00:00, 32.57s/trial, best loss: 0.45965640809331465]
100%|██████████| 8/8 [00:32<00:00, 32.58s/trial, best loss: 0.45965640809331465]
100%|██████████| 9/9 [00:32<00:00, 32.59s/trial, best loss: 0.45965640809331465]
100%|██████████| 10/10 [00:27<00:00, 27.87s/trial, best loss: 0.06246893764854211]
100%|██████████| 11/11 [00:32<00:00, 32.57s/trial, best loss: 0.06246893764854211]
100%|██████████| 12/12 [00:09<00:00, 9.46s/trial, best loss: 0.06246893764854211]
100%|██████████| 13/13 [00:32<00:00, 32.49s/trial, best loss: 0.06246893764854211]
100%|██████████| 14/14 [00:27<00:00, 27.93s/trial, best loss: 0.04859198593343972]
100%|██████████| 15/15 [00:32<00:00, 32.74s/trial, best loss: 0.04859198593343972]
100%|██████████| 16/16 [00:32<00:00, 32.71s/trial, best loss: 0.04859198593343972]
100%|██████████| 17/17 [00:32<00:00, 32.67s/trial, best loss: 0.04859198593343972]
100%|██████████| 18/18 [00:32<00:00, 32.71s/trial, best loss: 0.04859198593343972]
100%|██████████| 19/19 [00:32<00:00, 32.63s/trial, best loss: 0.04859198593343972]
100%|██████████| 20/20 [00:10<00:00, 10.05s/trial, best loss: 0.04859198593343972]
100%|██████████| 21/21 [00:32<00:00, 32.95s/trial, best loss: 0.04859198593343972]
100%|██████████| 22/22 [00:10<00:00, 10.45s/trial, best loss: 0.04859198593343972]
```

	KNN	rf	RF_worse
r_2	0.95	0.98	0.96
mae	24523.91	14263.15	20824.81
rmse	35721.67	20910.33	32405.70

- RMSE = Square root of sum of squared distances between our target variable and predicted values.
- MAE = the average magnitude of errors in a set of predictions, without considering their directions (better for sets with alot of outliers which is this dataset)
- R2 provides a measure of how well future samples are likely to be predicted by the model



# AUTO-ML - (OPTUNA) - HYPERPARAMETER TUNING

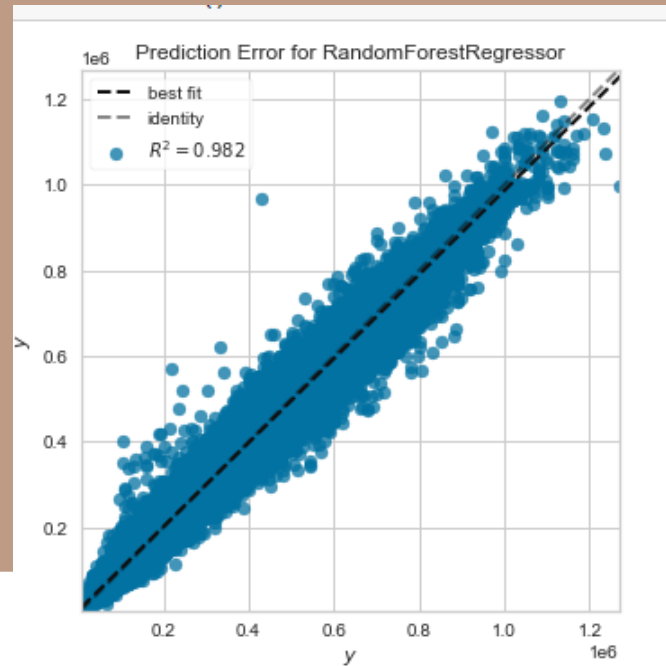
```
[I 2021-08-23 03:01:41,319] Trial 49 finished with value: 14263.14926630103 and parameters: {'n_estimators': 97, 'max_depth': 29.856901218249337}. Best is trial 0 with value: 14263.14926630103.
```

Unfortunately, Gridsearch is crashing consistently and Optuna doesn't improve my parameters despite 50 rounds of tuning IE first round was still better

# MODEL RESULTS - A RANDOM PREDICTION

```
print("prediction:", pd.DataFrame(rf.predict(x_test_scale1), columns=["pred"]).pred.iloc[-1],  
      "\nActual:      ", y_test.iloc[-1])
```

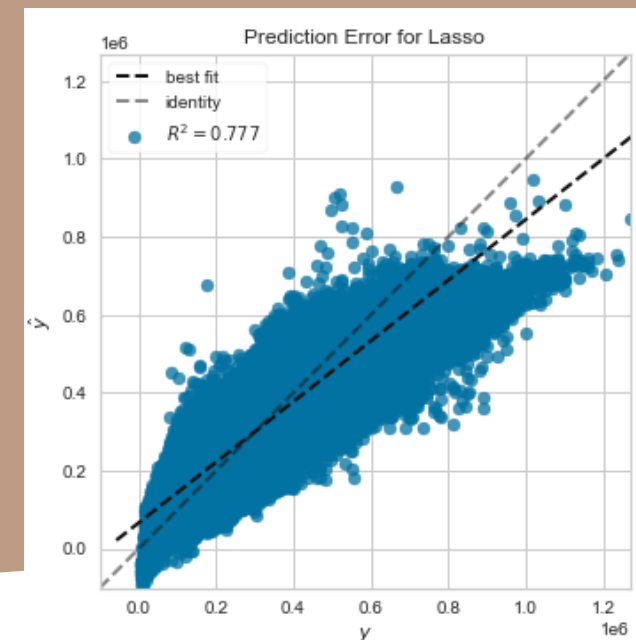
prediction: 336486.3636363637  
Actual: 325000



Roughly every prediction  
will be on average 14k  
away from the actual re-

So This graph plots predictions against actual  
results to show how accurate the model is

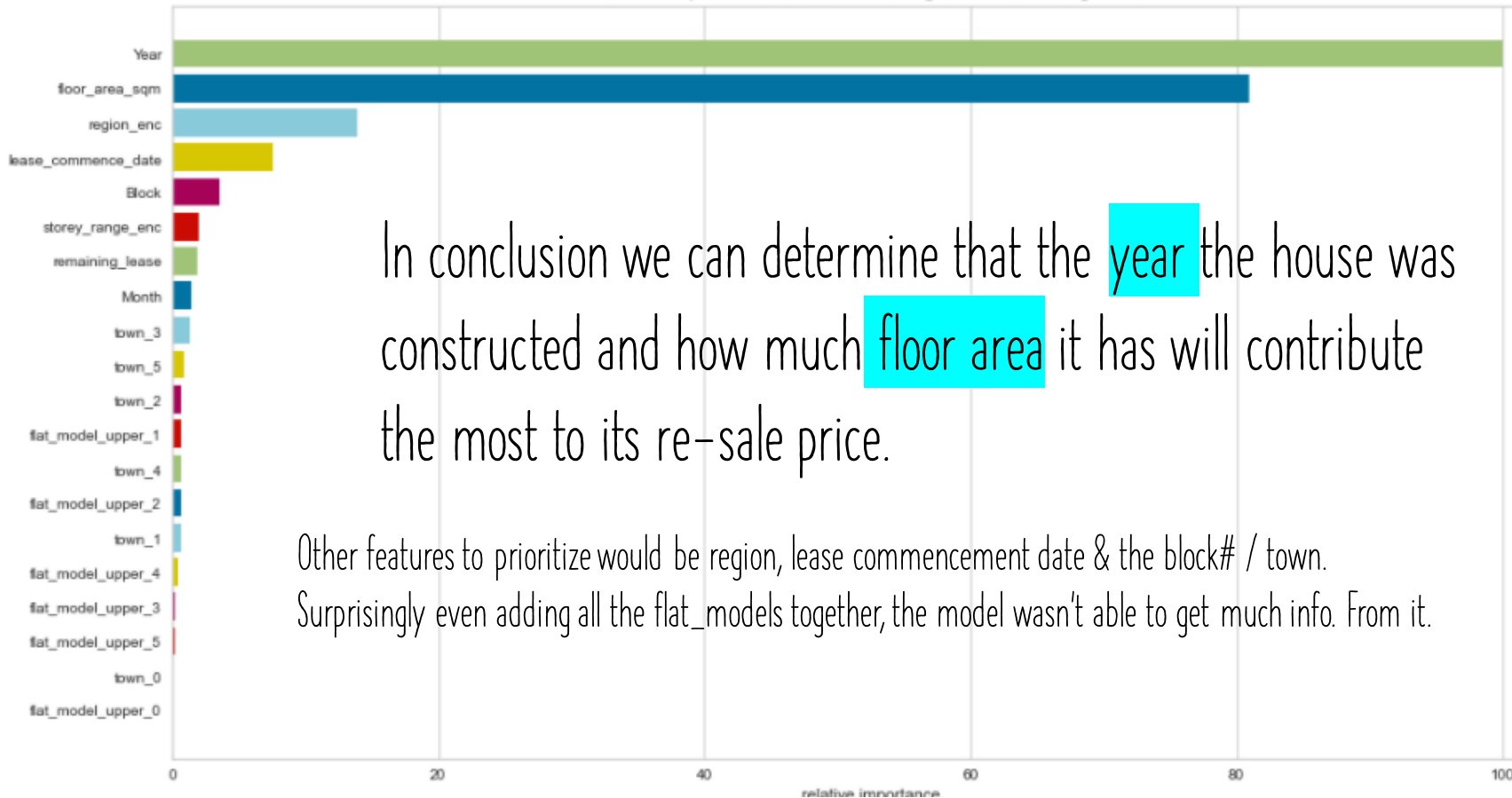
on the right u can tell that a linear lasso model  
isnt doing too well in the regression prediction





# ML VISUALIZATIONS

Feature Importances of 20 Features using RandomForestRegressor



In conclusion we can determine that the **year** the house was constructed and how much **floor area** it has will contribute the most to its re-sale price.

Other features to prioritize would be region, lease commencement date & the block# / town. Surprisingly even adding all the flat\_models together, the model wasn't able to get much info. From it.

Ft. Importance:

- relative importance of each feature when making a prediction

*the sum of splits over the number of splits (across all tress) that include the feature, proportionally to the number of samples it splits.*

# CHALLENGES & THOUGHTS

- Challenges

- Not enough memory, spent hours re-running code only to have it ending in a memory issue. -> lost variables to invert my encoded data.
- Unable to augment data with features such as nearby MRT / Business park since this varies between the blocks itself and requires on the ground knowledge and/or manual updating / scraping of data
- Getting better quality data (more features) would also result in the model requiring to be re-trained

- Thoughts

- Could possibly try minmaxscaler to center the data instead of merely normalizing the data with the standard scaler.
- Could create a pipeline for ease of ML ops in the future

# THE END

---

[https://github.com/lolasery/for\\_holmusk](https://github.com/lolasery/for_holmusk)