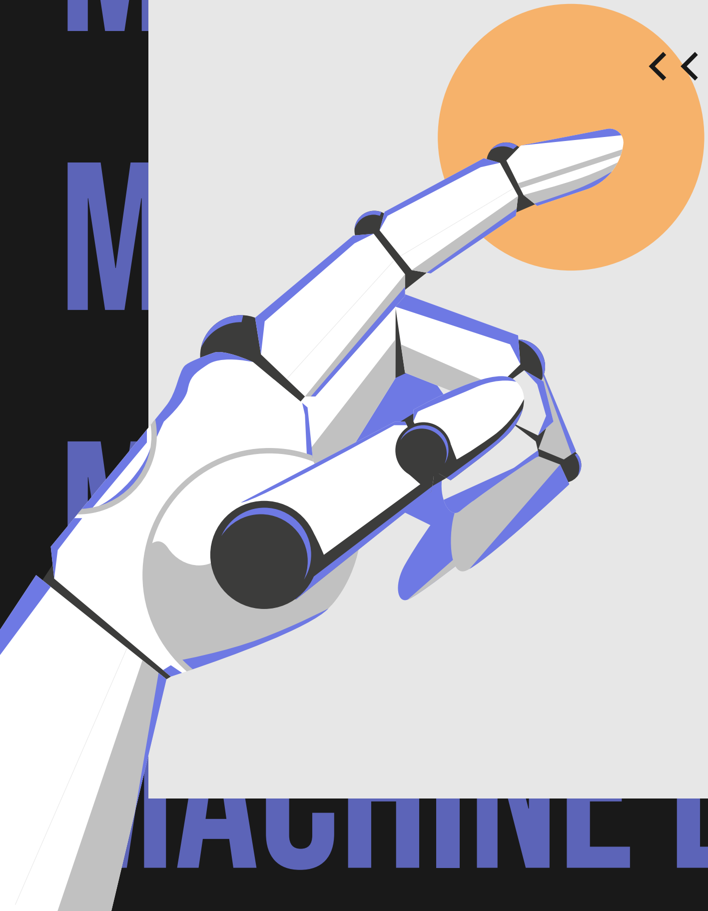


CZ4041 MACHINE LEARNING PROJECT

Zillow Prize: Zillow's Home Value Prediction (Zestimate)

Au Yew Rong Roydon, Chen Kang Ming, Dion Toh Siyong, Lim Ziyi Janesse, Tan Yue Jun

$logerror = log(Zestimate) - log(SalePrice)$

PROBLEM STATEMENT

MACHI
LEARN
PROJEC

Predict the **log-error** between Zillow's Zestimate and the actual sale price, given all the features of a home.

$$\boxed{Logerror = log(Zestimate) - log(SalePrice)}$$

<<<<

Source: https://images.app.goo.gl/tZ194stmWwt7k9n98

TABLE OF CONTENTS

MACHINE LEARNING PROJECT

01.

DATA EXPLORATION & PREPROCESSING

DATASETS PROVIDED

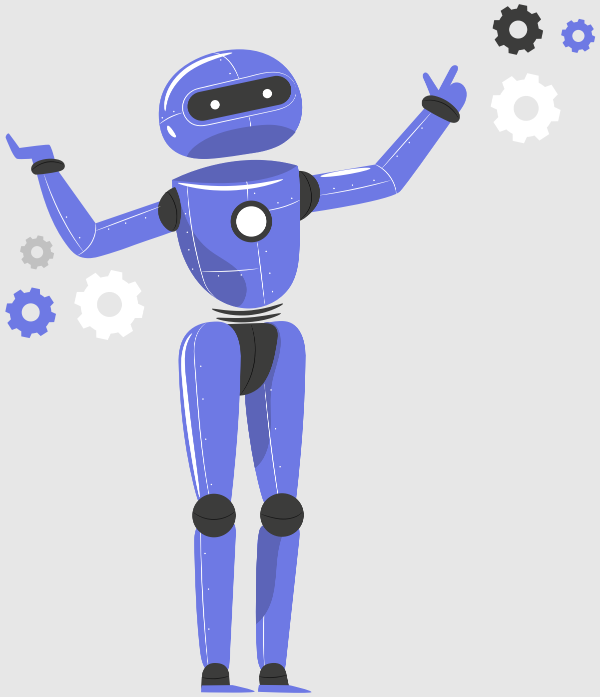
properties_2016
All the properties with home features for the year of 2016

properties_2017
All the properties with home features for the year of 2017

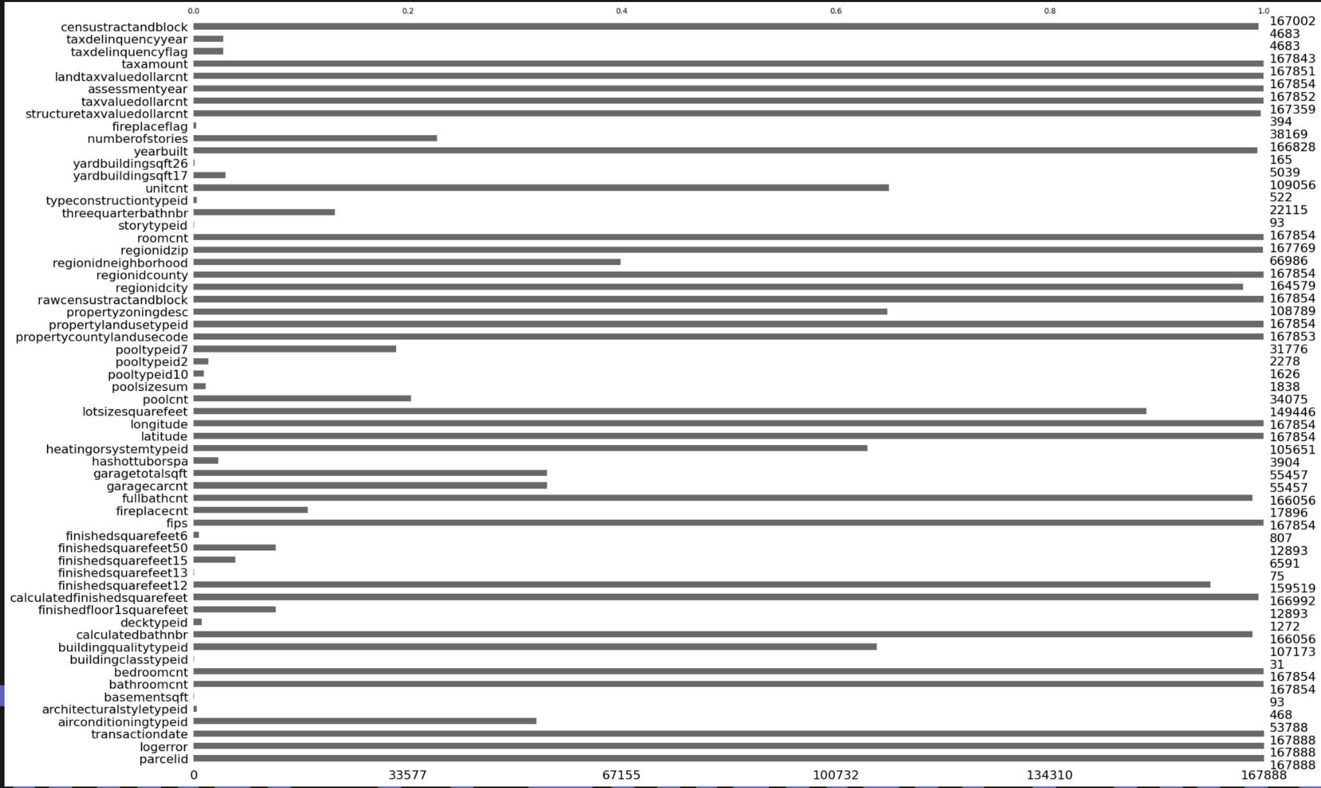
train_2016
Training set with transactions from 1/1/2016 to 31/12/2016

train_2017
Training set with transactions from 1/1/2017 to 15/09/2017

Contains output variable ‘logerror’ to be predicted.



MISSING VALUES



REMOVAL OF MISSING VALUES

- Drop columns with **> 99% empty rows**
  - Since they are mostly empty, we believed it is not useful
- Columns dropped:
  - 'Architecturalstyletypeid'
  - ‘Basementsqft’
  - ‘Buildingclasstypeid’
  - ‘Decktypeid’
  - ‘Finishedsquarefeet13’
  - ‘Storytypeid’
  - ‘Typeconstructiontypeid’
  - ‘Yardbuildingsqft26’
  - ‘fireplaceflag’

Source: ttps://freepngimg.com/save/25938-falling-clipart/3000x3351

REMOVAL OF MISSING VALUES

- Imputation
  - Mean/ Mode/ Median?
- Data exploration of features
  - Characteristics
  - Distribution
  - Plots
- Typically…
  - Significant number of outliers → **Median** [Continuous features]
  - Small number of numerical unique values → **Median/Mode**

Source: https://images.app.goo.gl/dffjRWPoBD1HEenv7

IMPUTATION OF SPECIFIC FEATURES

- ‘Poolcnt’
  - Only has one unique value: ‘1.0’
  - Rest are null values

```
poolcnt
1.0    34075
Name: count, dtype: int64
```

- Contrast poolcnt null values with other pool related columns’ null values
  - Eg: ‘poolsizesum’

```
--------------------Values that should impute with 0----------------------------

Minimum value of poolsizesum: 24.0, Values of poolsizesum, pooltypeid2, pooltypeid7, for poolcnt == null: [nan], [nan], [nan]

Number of missing values of poolcnt out of 167888: 133813, Columns with no missing values for poolcnt == null: ['parcelid', 'logerror', 'transactiondate']
```

  - Null values corresponds

- *Impute missing ‘poolcnt’ values* ***with zeros***

IMPUTATION OF SPECIFIC FEATURES

1. ‘ID’ Columns
   - ‘pooltypeid2’, ‘airconditioningtypeid’, ‘buildingqualitytypeid’, etc

```
-------------------Values that should impute with a new id: 0---------------------------

Minimum value of pooltypeid2: 1.0, Values of poolsizesum for pooltypeid2 == null: [  nan  475.  392.  664.  324.  385.  524.  400.  800.  434.  432.  403.
  416.  360.  380.  480.  608.  560.  377.  600.  450.  277.  500.  589.
  680.  714.  700.  555.  264.  547.  775.  684.  540.  492.  570.  648.
  422.  254.  505.  705.  755.  610.   49.  291.  420.  624.  550.  504.
  512.  440.  415.  288.  948.  612.  558.  510.  630.  666.  477.  576.
  536.  513.  343.  546.  971.  646.  406.  396.  356.  379.  830.  369.
  581.  304.  460.  382.  371.  627.  795.  583.  748.  336.  588.  518.
  580.  378.  465.  299.  448.  472.  412.  525.  216.  426. 1020.  544.
  404.  294.   28. 1052.  444. 1750.  442.  405.  428.  650.  838.  160.
  408.  908.  364.  490. 1125.  649.  702.  740.  750.  880.  585.  727.
  430.  164.  390.  384.  487.  567.  372.  419.  820.  640.  836.  435.
  535.  704.  528.  352.  200.  968.  312.  365.  425.  686.  665.  520.
  345.  489.  495.  250.  447.  720.  591.  556.  394.  386.  397.  350.
  456.  631. 1220.  468.  561.  207.  745.  370.  413.  265.  623.  429.
  595.  395.  300.  647.  625.  920.  418.  564.  900.  340.  575.  634.
  421.  670.  485.  537.  760.  810.  342.  538.   91.  572.  563.  568.
  523.  593.  893.  290.  870.  496.  443.  330.  325.  276.  483.  516.
  462.  653.  615.  476.  531.  501.  592.  722.  514.  527.  539.  759.
  736.  554.  471.  780.  620.  270.  319.  862.  375.  455.  584.  534.
  411.  310.  486.  645.  441.  401.  321.  280.  594.  461.  756.  691.
  398.  295.  451.  308.  855.  690.  333.  960.  338.  668.  672.  521.
  851. 1749.   40.  463.  626.  242.  367.  629.  506.  642.  320. 1000.
  368.  682.  815.  551.  366.  530.  503.  632.  707.  357.  578.  794.
  832.  747.  129.  562.  655.  467.  557.  230.  663.  602.  675.   24.
 1500.  764.  459.  172.  240.  799.  710.  387.   65.  660.  458.  256.
  590.  587.  598.  402.  840.  673.  353.  306.  543.  424.  770.  552.
  362.  885.  112.  427.  478.  688.  969.  361.  347. 1065.  144.   38.
  482.  604.  275. 1070.  453. 1120.  507.  792.  742.  735.  574.  725.
  233.  785.  410.  449.  515.  391. 1364.  238.  431.  931.  635.  850.
  464.  605.  436.  234.  657.  837.  389.  498.  105.  773.  694.  255.
  358.  439.  474.  712.  728. 1109. 1200.  470.  738.  913.  751.  257.
  548.  768.  549.  414.  423.  532.  309.  616.  601.  860.  499.  327.
  582.  661.  762.  990.]

Minimum value of pooltypeid7: 1.0, Values of poolsizesum for pooltypeid7 == null: [nan  1.]

Minimum value of airconditioningtypeid: 1.0,Unique values of airconditioningtypeid: [ 1. nan  5. 13. 11.  9.  3.]

Minimum value of buildingqualitytypeid: 1.0,Unique values of buildingqualitytypeid: [ 4. nan  1.  7. 12. 10.  8.  6. 11.  9.  5.  3.  2.]

Minimum value of heatingorsystemtypeid: 1.0,Unique values of heatingorsystemtypeid: [ 2. nan  7.  6. 24. 13. 20. 18. 11.  1. 14. 12. 10.]

Minimum value of regionidcity: 3491.0, Values of rawcensustractandblock  for regionidcity == null: [60375012.001004 60371032.001016 60590320.432007 ... 60374312.001006
 60377018.02101  60590422.012005]
```

     - *Impute **with zeros***

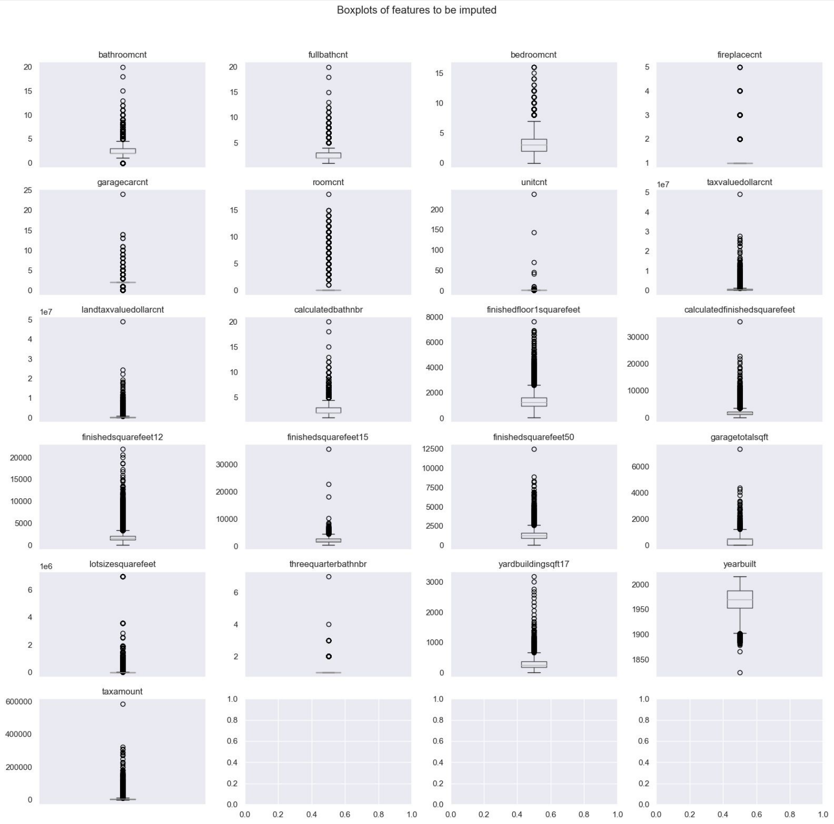
IMPUTATION OF SPECIFIC FEATURES

1. 'cnt' Columns
    - Float type
    - Actually whole numbers

```
bathroomcnt unique values: [ 2.   3.5  3.   2.5  4.   1.   5.   5.5  1.5  8.   0.   4.5  9.   7.
  6.  10.   6.5  7.5 12.  11.  20.   8.5 15.   nan 18.  13. ]

bedroomcnt unique values: [ 3.  4.  2.  5.  1.  6.  7.  0. 12. 11.  8.  9. 10. 16. 14. 13. 15. nan]

fireplacecnt unique values: [nan  1.  2.  3.  4.  5.]

fullbathcnt unique values: [ 2.  3.  4.  1.  5.  8. nan  9.  7.  6. 10. 12. 11. 20. 15. 18. 13.]

garagecarcnt unique values: [nan  2.  1.  3.  0.  4.  6.  8.  5.  7. 11. 10. 24.  9. 13. 14.]

roomcnt unique values: [ 0.  8.  6.  5.  7.  4.  3.  9. 12. 11. 10.  2.  1. 13. 15. 14. 18. nan]

unitcnt unique values: [  1.  nan   2.   4.   3.   6. 143.  11.   9.   5.  70.  45.  42. 237.]

taxvaluedollarcnt unique values: [360170. 585529. 119906. ... 354621.  67205.  49546.]

landtaxvaluedollarcnt unique values: [237416. 239071.  57912. ... 214889. 221068. 283704.]
```

    - *Imputed with* ***mode***

IMPUTATION OF CONTINUOUS COLUMNS



Observe: Many outliers

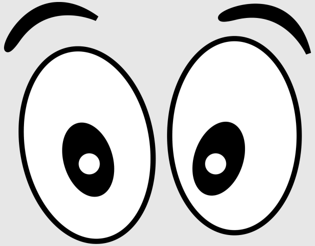
Imputed: ***Median***

CATEGORICAL COLUMNS

1. Some interesting categorical columns
   - **‘hashottuborspa’**
     1. Only value in the dataset was ‘true’
     2. Null values were most likely representing ‘false’
     3. *Imputed with ‘**false’***
   - **‘propertycountylandusecode’**
     1. Only 1 null value
     2. ***This row was dropped***
   - **‘taxdelinquencyflag’**
     1. Only value was ‘Y’
     2. Null values were most likely to be the value of ‘N’
     3. *Imputed with* ***‘N’***

DROPPING OF ROWS

1. Dropping rows with null values for these columns:
   - ‘propertylandusetypeid’
   - ‘regionidcounty’
   - ‘rawcensustractandblock’,
   - ‘censustractandblock’



```
-------------------Na values of object type------------------------
transactiondate                    0
hashottuborspa                163098
propertycountylandusecode          1
propertyzoningdesc             58698
taxdelinquencyflag            162333
dtype: int64
-------------------Na analysis for imputation------------------------
Unique values of hashottuborspa: [nan True]
Unique values of propertyzoningdesc: ['LARS' 'PSR6' 'LAR3' ... 'LCRA 7500*' 'LCRA7000-R' 'BFA15000*']
```

Source:

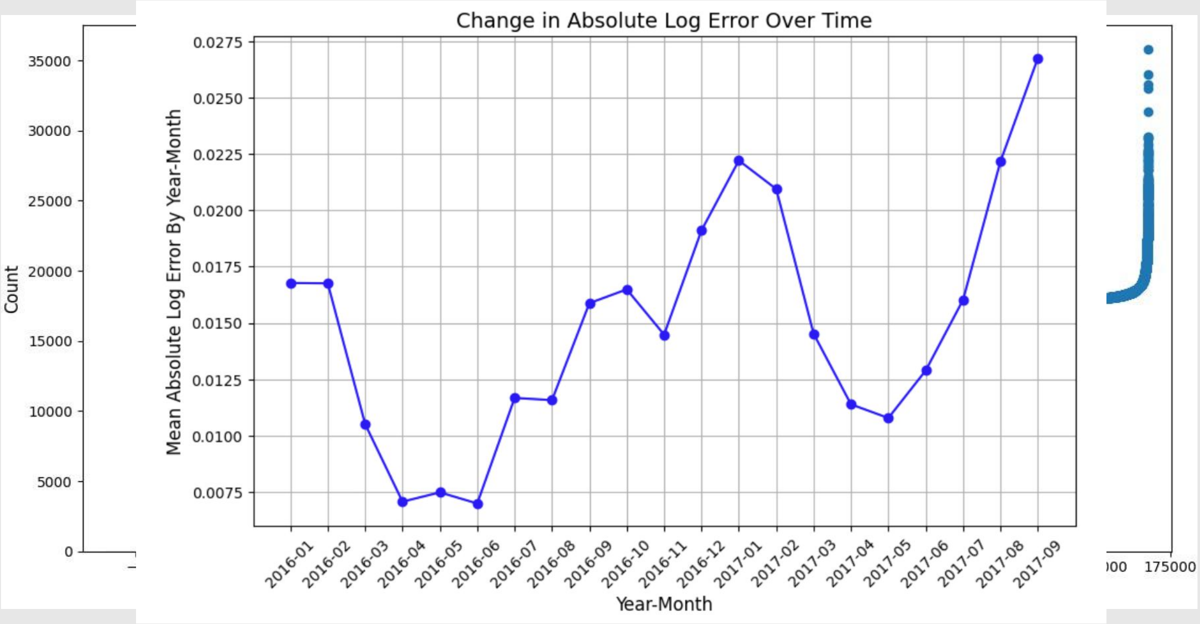
CHANGING OF DATA TYPES

- Dataset was huge (1GB)
- Casting of float type → int type
  - Do this, as long we do not lose a large degree of precision
- Memory usage **significantly decreased (to 34.9MB)**
- Increases **resource efficiency**

Source: https://images.app.goo.gl/QSFLhTfH4AhpbPsg7

EXPLORATION OF LOGERROR

- Logerror is our prediction (Y) variable



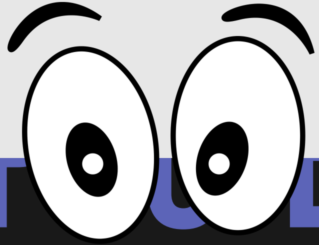
02.

FEATURE ENGINEERING

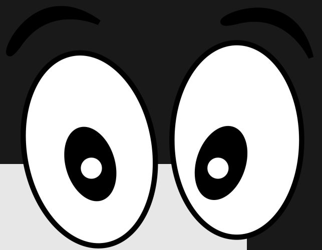
ASSUMPTION

- Features engineered based on *Sales Price* will in turn affect 'logerror'
- 'Logerror' is derived from Sale Price

$$\boxed{Logerror = \log(Zestimate) - \log(SalePrice)}$$



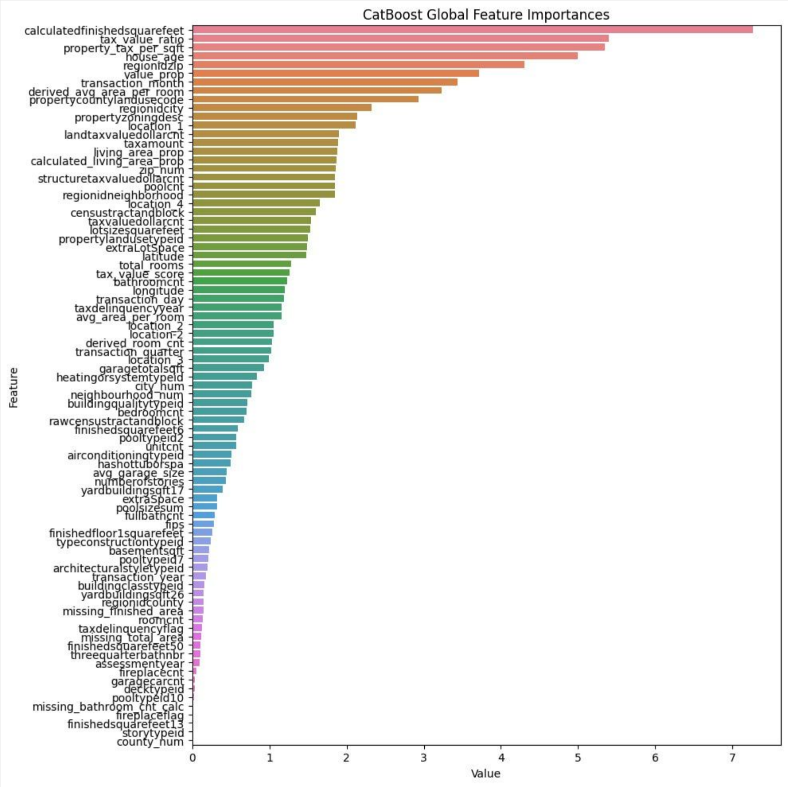
FEATURES ADDED



'percentage_error_of_living_area_12'	'extra_space'	'location' (l)	'location_5'' (l_5)	'city_num'	'property_tax_per_sqft'
'percentage_error_of_living_area_15'	'extra_lot_space'	'location_2' (l_2)	'tax_value_ratio'	'county_num'	'avg_area_per_room'
'calculated_living_area_prop'	'total_rooms'	'location_3' (l_3)	'tax_value_score'	'neighbourhood_num'	'derived_avg_area_per_room'
'living_area_prop'	'value_prop'	'location_4' (l_4)	'zip_num'	'avg_garage_size'	'house_age'

Source:

SIGNIFICANT NEW FEATURES



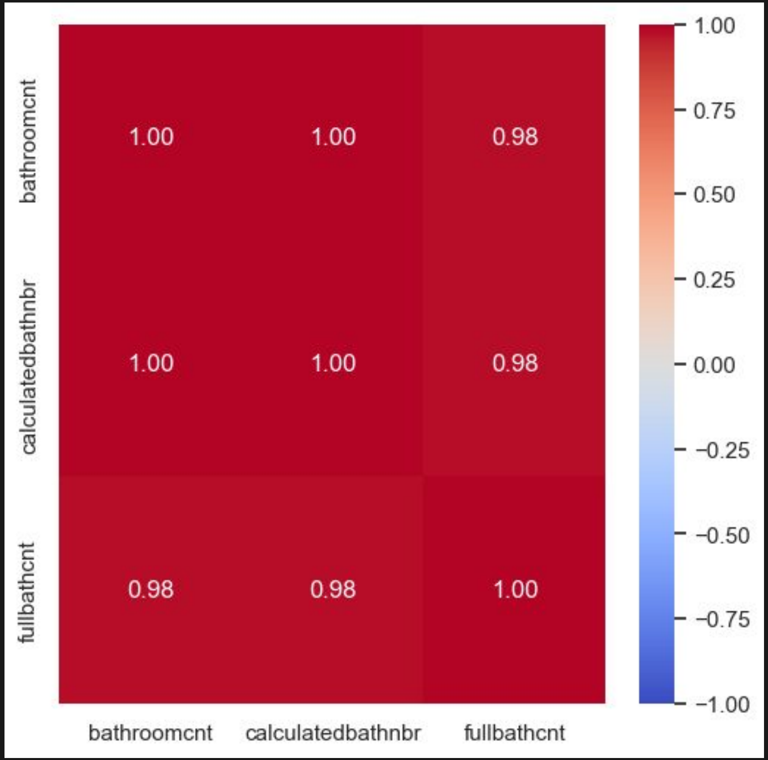
Features importance as ranked by Catboost

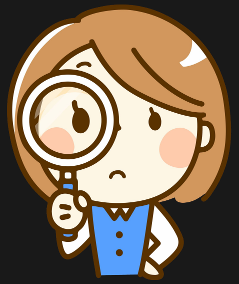
Notice: Some newly created features ranked high

Notably:

1. 'tax_value_ratio'
2. 'property_tax_per_sqft'
3. 'house_age', 'value_prop'
4. 'derived_avg_area_per_room'

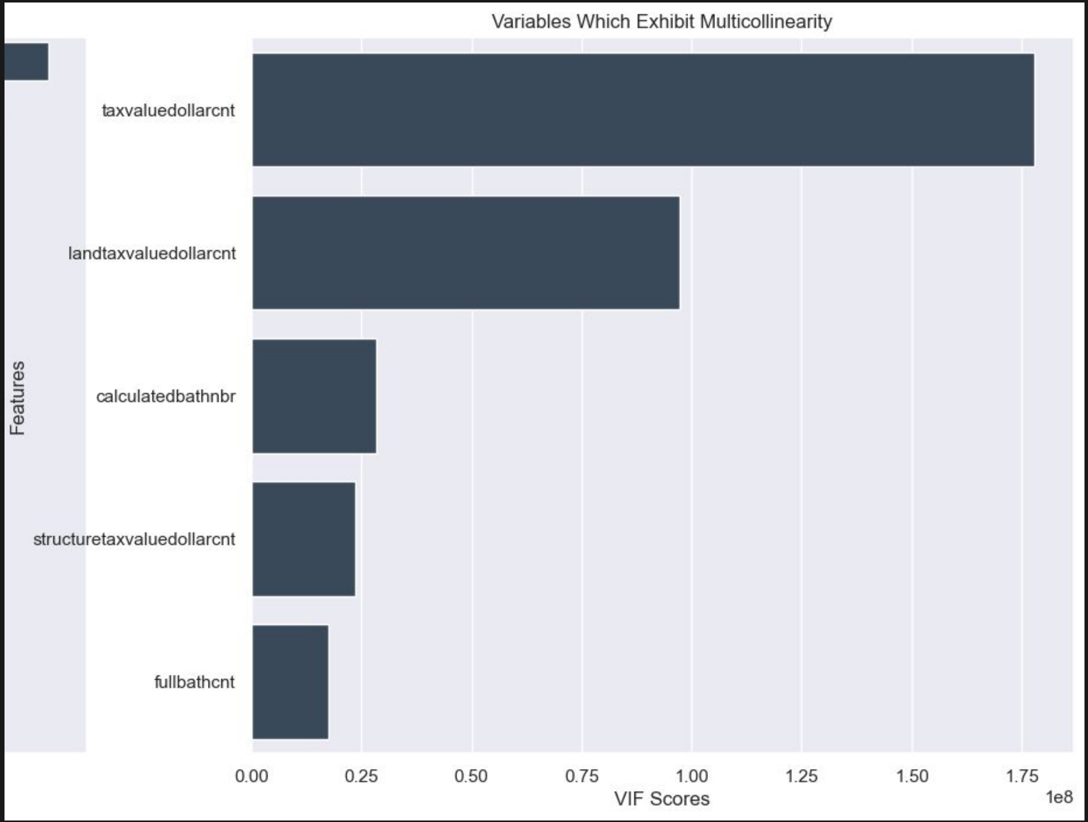
DROPPING OF SOME FEATURES

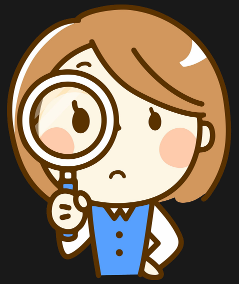




Source: https://images.app.goo.gl/yq4BaAMT39vsYWST9

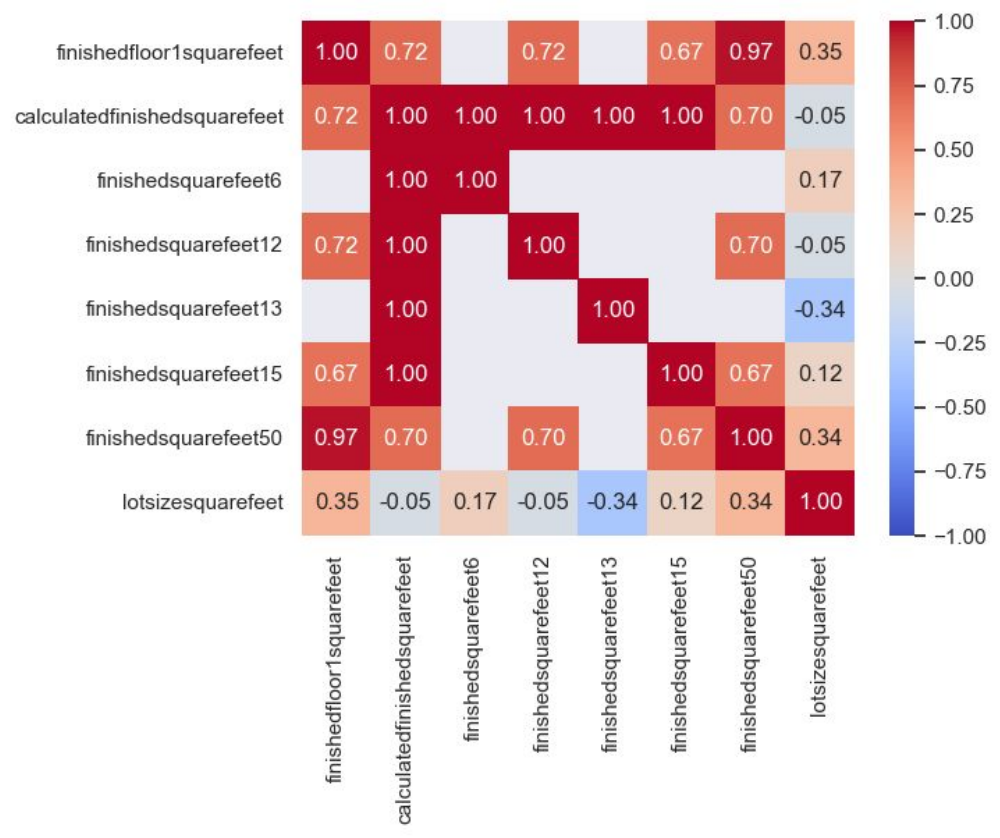
DROPPING OF SOME FEATURES

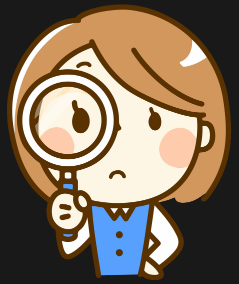




Source: https://images.app.goo.gl/yq4BaAMT39vsYWST9

DROPPING OF SOME FEATURES

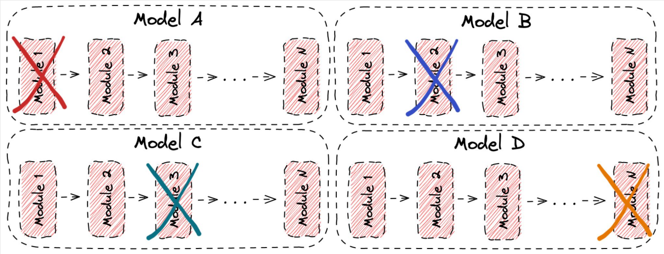




Source: https://images.app.goo.gl/yq4BaAMT39vsYWST9

ABLATION STUDY

- **Investigate** effectiveness of employed strategies
  - Missing value imputation
  - Column/Row removal
  - Addition of engineered features



Source: https://www.baeldung.com/cs/ml-ablation-study

- **Findings**
  - Some models (CatBoost, LGBM, XGBoost) fared better on original dataset
    - Inherent ability to **automatically handle** missing data
  - Imputations **did not help** improve accuracy on these models
  - However, engineered features and dropping of columns did **contribute substantially** to improving the models accuracy :)

FINAL DATASET

Source: https://images.app.goo.gl/rp7YzxoEctmd8KqE8

√ Engineered Features

√ Dropping of non-significant columns

√ No user imputations

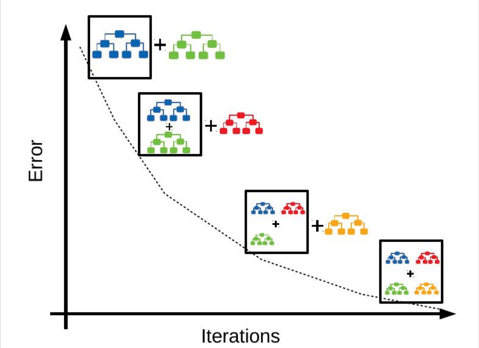
03.

EXPERIMENTATION

GRADIENT BOOSTING MODELS

A type of **Ensemble Learning Method**

1. Sequential decision trees
2. Each tree 'fixes' errors from the previous tree

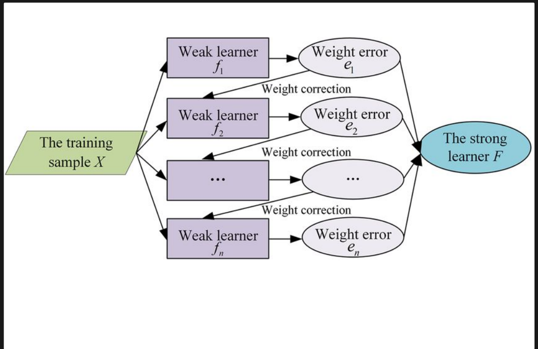


Effective for **Tabular datasets:**

1. Deals with missing values, outliers
2. Handles heterogenous data without extensive processing
3. Robust against overfitting issues

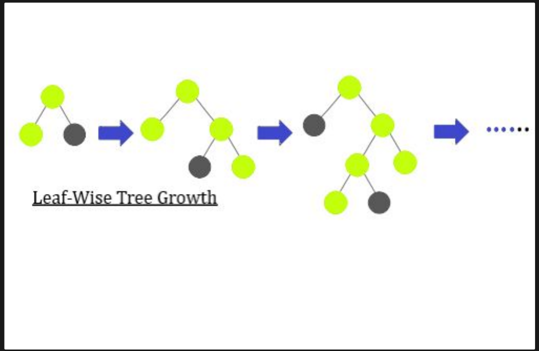
Source: https://medium.com/swlh/gradient-boosting-trees-for-classification-a-beginners-guide-596b594a14ea

MODELS USED



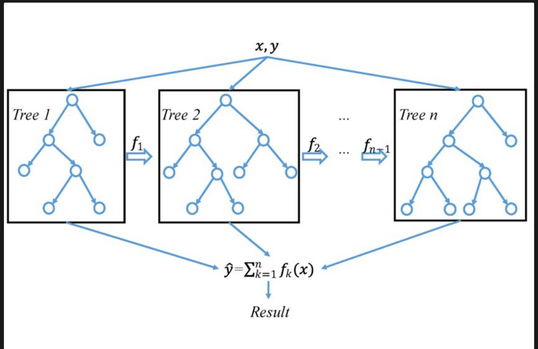
CATBOOST

Handles categorical data w/o extensive preprocessing



LIGHTGBM (LGBM)

Quick and efficient model, works well in model-stacking



XGBOOST

Large amount of tunable hyperparameters for fine-tuning

Source: https://www.researchgate.net/figure/Diagram-of-CatBoost-structure_fig1_367098144
https://www.geeksforgeeks.org/lightgbm-light-gradient-boosting-machine/
https://www.researchgate.net/figure/A-general-architecture-of-XGBoost_fig3_335483097

EVALUATION METRICS FOR MODEL ANALYSIS

KAGGLE PUBLIC SCORE

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \|y(i) - \hat{y}(i)\|^2}{N}},$$

ROOT-MEAN SQUARED ERROR

$$\mathrm{MAE} = \frac{1}{n} \sum_{i=1}^{n} |x_i - x|$$

MEAN ABSOLUTE ERROR

Source:
https://www.iconfinder.com/icons/4373210/kaggle_logo_logos_icon

https://c3.ai/glossary/data-science/root-mean-square-error-rmse/
https://www.statisticshowto.com/absolute-error/

COMPARATIVE ANALYSIS

Condition: **Individual models** with preprocessed data

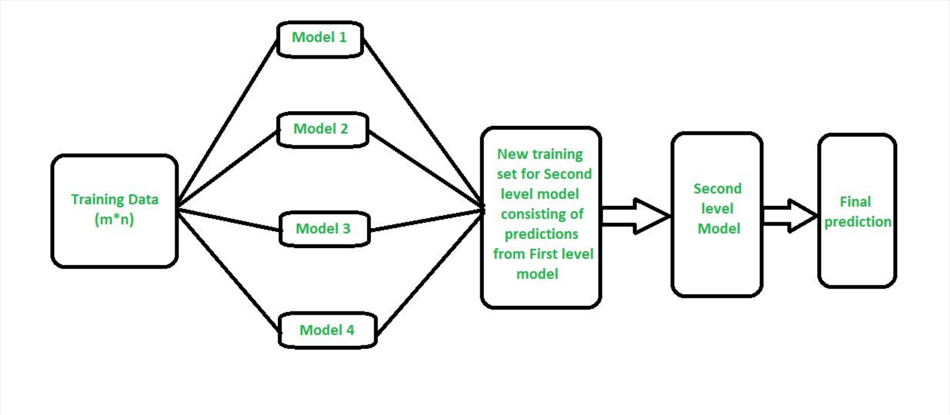
Model	Public Score	RMSE	MAE	*Top Ranking (%)*
CatBoost	**0.06426**	0.082977	0.052141	***12.911***
LGBM	0.06435	0.082652	0.052371	*17.736*
XGBoost	0.06630	0.155033	0.067484	*90.376*

Source: https://images.app.goo.gl/ZBwP1nh3N5VS5eiGA

MODEL STACKING

Model Stacking:

1. Combining the strengths of different models to boost performance
2. Using **CatBoost and LGBM**, with different combinations of each model



Source: https://www.geeksforgeeks.org/stacking-in-machine-learning/

MODEL STACKING

Condition: Stacked CatBoost and LGBM models in **different proportions**

Weightage (%)		Public Score	Top Ranking (%)
CatBoost	LGBM		
10	90	0.06430	14.422
20	80	0.06426	12.911
30	70	0.06420	10.764
40	60	0.06420	10.764
50	50	0.06419	8.192
60	40	0.06418	7.635
70	30	0.06419	8.192
80	20	0.06420	10.764
90	10	0.06423	11.691

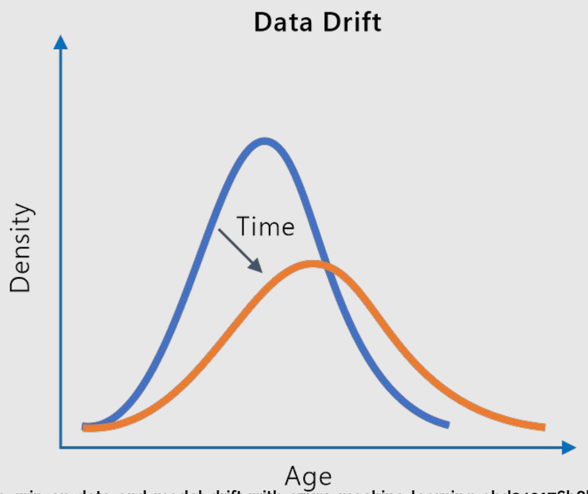
DATA DRIFT

Data drift from model degradation could pose an issue

1. Distribution of new points could differ from old trend
2. e.g. changes in **Govt. Policy, Market Sentiment…**
3. Use **Alibi Detect lib.** → **TabularDrift function** to detect data drift

   transactiondate -- Drift? Yes! -- Chi2 2000.000 -- p-value 0.000
   buildingqualitytypeid -- Drift? Yes! -- Chi2 700.341 -- p-value 0.000
   assessmentyear -- Drift? Yes! -- K-S 1.000 -- p-value 0.000
   year_month -- Drift? Yes! -- K-S 1.000 -- p-value 0.000
   regionidzip -- Drift? Yes! -- K-S 0.064 -- p-value 0.032
   tax_value_ratio -- Drift? Yes! -- K-S 0.094 -- p-value 0.000

4. If **p-value < 0.05, drift has occurred**



Source: https://towardsdatascience.com/getting-a-grip-on-data-and-model-drift-with-azure-machine-learning-ebd240176b8b

DATA DRIFT

How do we use this data to improve?

1. Condition: **Drop columns that are below p-value threshold**

Model	Public Score	Top Ranking (%)
CatBoost + LGBM	0.06420	10.764

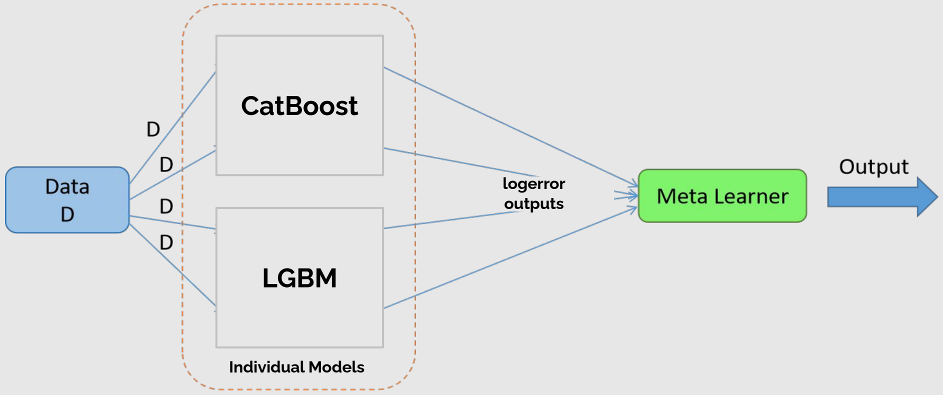
☹️

Conclusion: No improvement after considering model degradation and data drift

META LEARNER

Concept:

1. 'Learning' from 'Learning' algorithms
2. Models are able to **adapt + generalize** quickly to new tasks by **learning** from previous experiences



Source: https://www.analyticsvidhya.com/blog/2021/08/ensemble-stacking-for-machine-learning-and-deep-learning/

META LEARNER

1. Condition: **Individual model outputs w/ Meta Learner**

Public Score	RMSE	MAE	Top Ranking (%)
0.06429	0.082721	0.05222	14.157

☹️

Conclusion: No improvement after using meta-learning strategies

However...

META LEARNER - STACKED?

1. <u>Condition</u>: **3-Stacked Model (CatBoost + LGBM + Meta Learner)**

Public Score	Top Ranking (%)
0.06418	7.635

Conclusion: This model structure can also **be used!**

04.

RESULTS

RESULTS

Category	Model(s)	Public Score	Top Ranking (%)
Individual	CatBoost	0.06426	12.911
Stacked	**CatBoost + LGBM**	**0.06418**	**7.635** ⭐
MetaLearner	LGBM (using CatBoost and LGBM predictions)	0.06429	14.157
Stacked w MetaLearner	**CatBoost + LGBM + MetaLearner**	**0.06418**	**7.635** ⭐

final_lgbm_catboost_40_60_stacked.csv Complete (after deadline) · 3m ago	0.07506	0.06418
final_lgbm_catboost_meta_35_55_10_stacked.csv Complete (after deadline) · 2h ago	0.07506	0.06418

05.

SOLUTION NOVELTY

UNIQUE APPROACHES

Extensive Feature Engineering:

- Different perspectives, like homebuyer's sentiment, general perception were considered on top of other engineered features

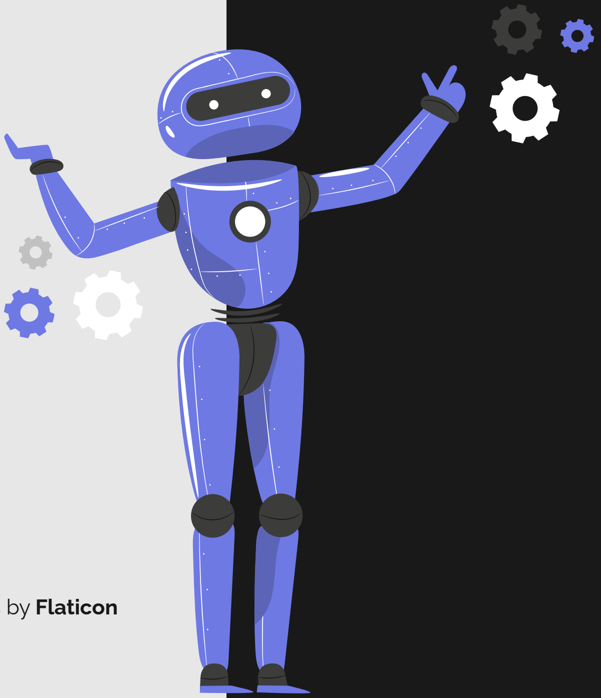
Exploration of Data Drift:

- Although was not as effective as imagined, it was relevant since property prices could change over time due to trends

Use of a 3-Stacked Model:

- CatBoost and LGBM was combined with the MetaLearner, which in itself is a combined model

THANK YOU!



CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon** and infographics & images by **Freepik**