Victoria, Australia: Visualization & Prediction

Kenny Dao
Data Bootcamp, Monash University

Melbourne, Australia Thursday, 18th February, 2021

ABOUT THE PROJECT

The project aims to provide a snapshot of community profiles analysis, stats (demography, employment, income, occupation) and predictions of population growth and house prices across Victoria.

DATA SOURCE

Australian Bureau of Statistics (abs.gov.au)

https://abs.gov.au (csv)

Community Profile

https://content.id.com.au (csv)

Suburb boundaries geoJSON

GeoJson-Data-master:
(https://github.com/tonywr71/GeoJson-Data.git)

vic_suburb_stats.json

Kaggle

Victoria property (cvs)

TOOLBOX

Python: Numpy, Scipy, Sklearn

Pandas library, Flask App

D3, Matplotlib, Seaborn

HTML, Javascript

Leaflet, JQuery, Ajax, ..

Tableau

Deployment: Heroku

ETL - Python / Pandas Library

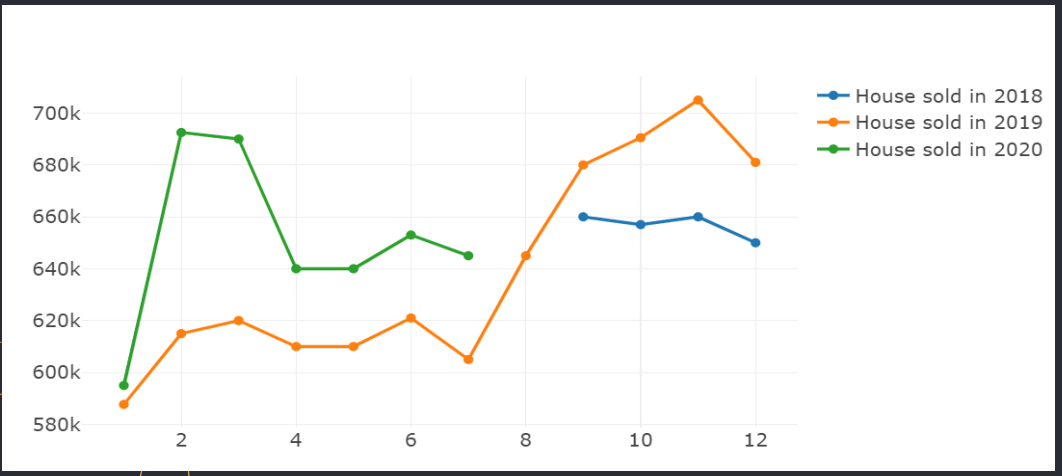
- Data transformation – cleaning, parsing, merging and extracting data from multiple sources & format
- Looking at the data using matplotlib

df.shape (68035, 7)

```
In [21]: # look at the dataframe
         df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 68035 entries, 0 to 68034
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   date_sold      68035 non-null  object
 1   price          68035 non-null  float64
 2   suburb         68035 non-null  object
 3   lat            68035 non-null  float64
 4   lon            68035 non-null  float64
 5   bedrooms       68035 non-null  int64
 6   property_type  68035 non-null  object
dtypes: float64(3), int64(1), object(3)
memory usage: 3.6+ MB

In [22]: # description
         df.describe().T
```

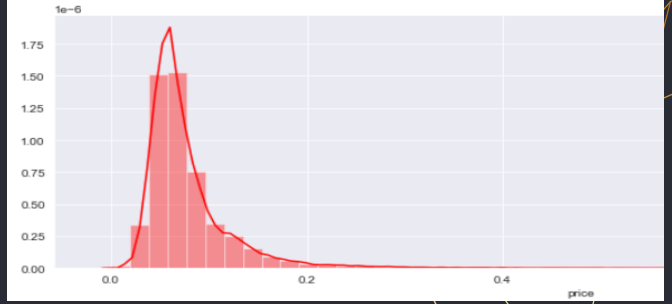
Out[22]:

	count	mean	std	min	25%	50%	75%	max
price	68035.0	789649.636760	513704.212869	1000.000000	518000.000000	650000.000000	885000.000000	9.600000e+06
lat	68035.0	-37.857904	0.173346	-38.405094	-37.936283	-37.823430	-37.749220	-3.742141e+01
lon	68035.0	145.053126	0.191898	144.535403	144.943901	145.041497	145.166624	1.457350e+02
bedrooms	68035.0	2.996399	0.957931	0.000000	2.000000	3.000000	4.000000	5.000000e+00



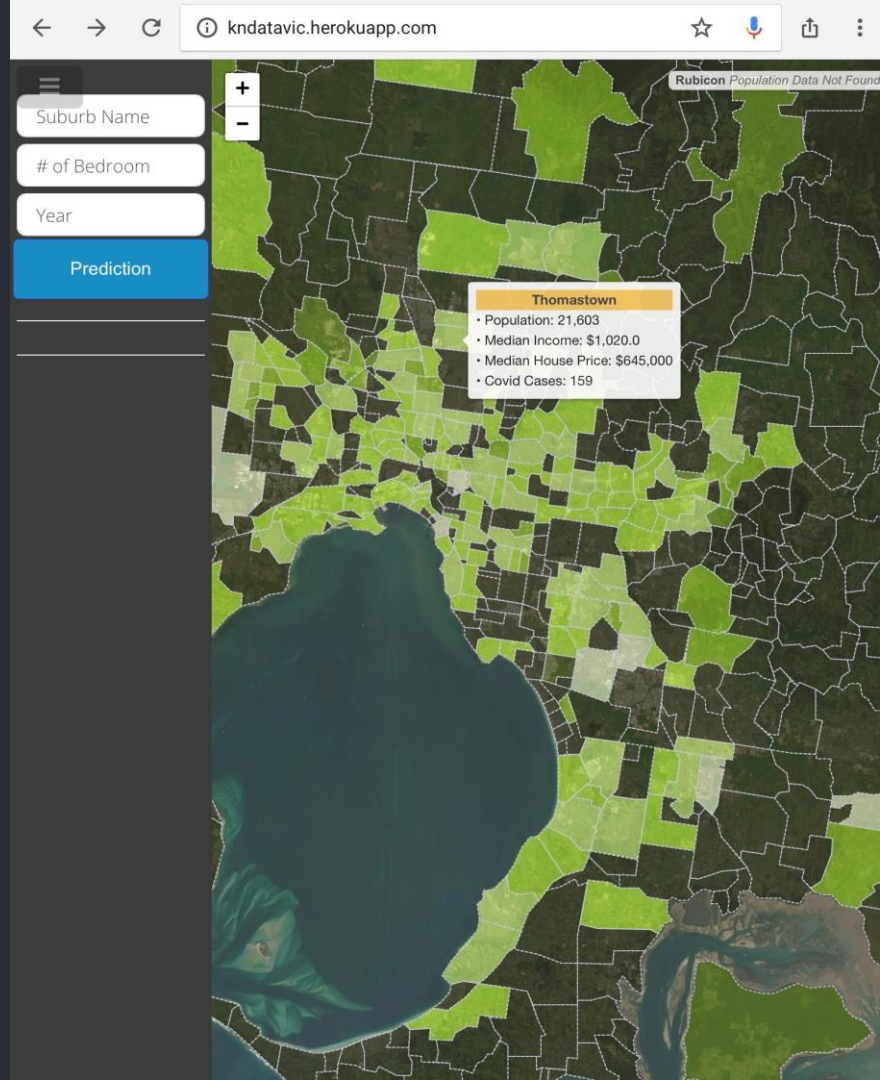
Price distribution

```
plt.subplot(2,1,1)
fig1 = sns.distplot(df['price'],color='red')

plt.subplot(2,1,2)
fig2 = sns.boxplot(data=df,x='price',color='aqua')
```



Front End/Flask App

- Front End:
  - HTML, CSS, Bootstrap
  - JavaScript interactivity powered by D3.js, Leaflet.js, jQuery.js
  - Plugins: $()autocomplete, Datatable (to display data)
- Flask App:
  - Serving html and multiple route
  - Handling <form> tag with [POST] methodology
  - Loading dataset, processing and executing prediction Models.



ML/Prediction

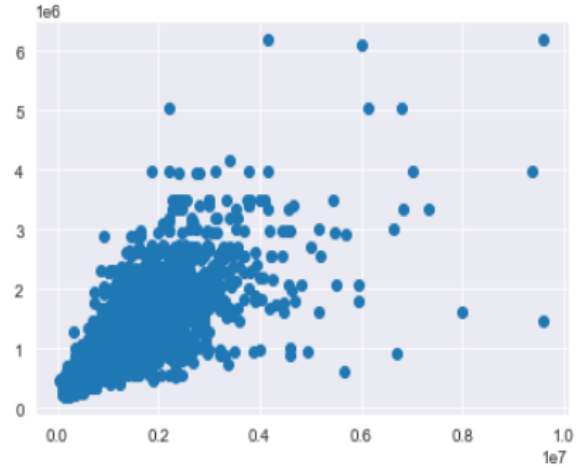
Prediction about population growth and house price based on user input (suburb, # of bedroom, year).

- Population growth prediction: on-the-flight model (linear regression) for over 800 suburbs across Victoria.
- House price prediction: Skcikit, Joblib

```
In [58]: model = ensemble.GradientBoostingRegressor(
                    n_estimators = 150, learning_rate=0.2,
                    max_depth=20, min_samples_split=4, min_samples_leaf=4,
                    max_features = 0.6, loss='huber')

In [59]: model.fit(X_train, y_train)

Out[59]: GradientBoostingRegressor(learning_rate=0.2, loss='huber', max_depth=20,
                                   max_features=0.6, min_samples_leaf=4,
                                   min_samples_split=4, n_estimators=150)
```

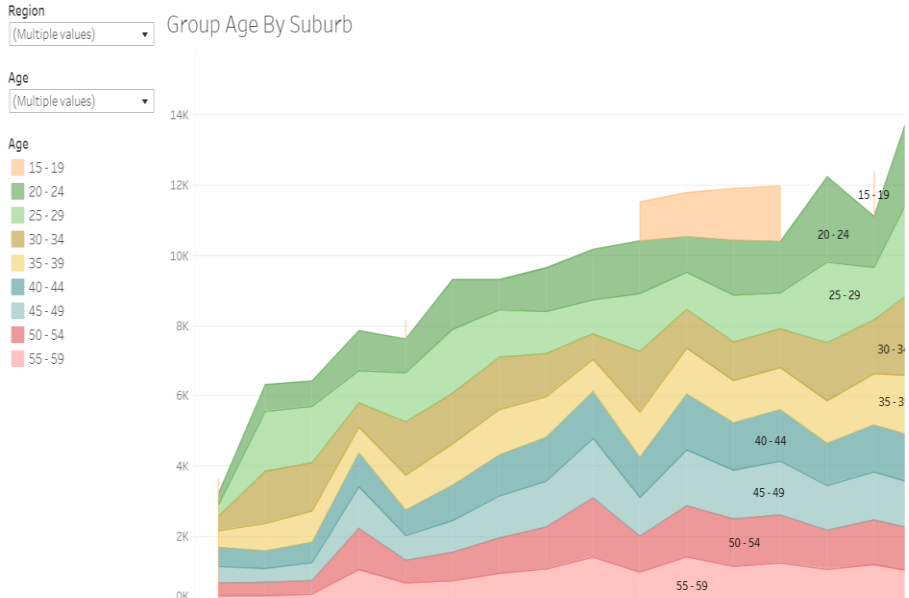
```
In [62]: # loading the model
         filename = 'Vic_house_trained_model_1.pkl'

         loaded_model = joblib.load(filename)

In [63]: result = loaded_model.score(X_test, y_test)
         print(result)

         0.6958833985852149
```

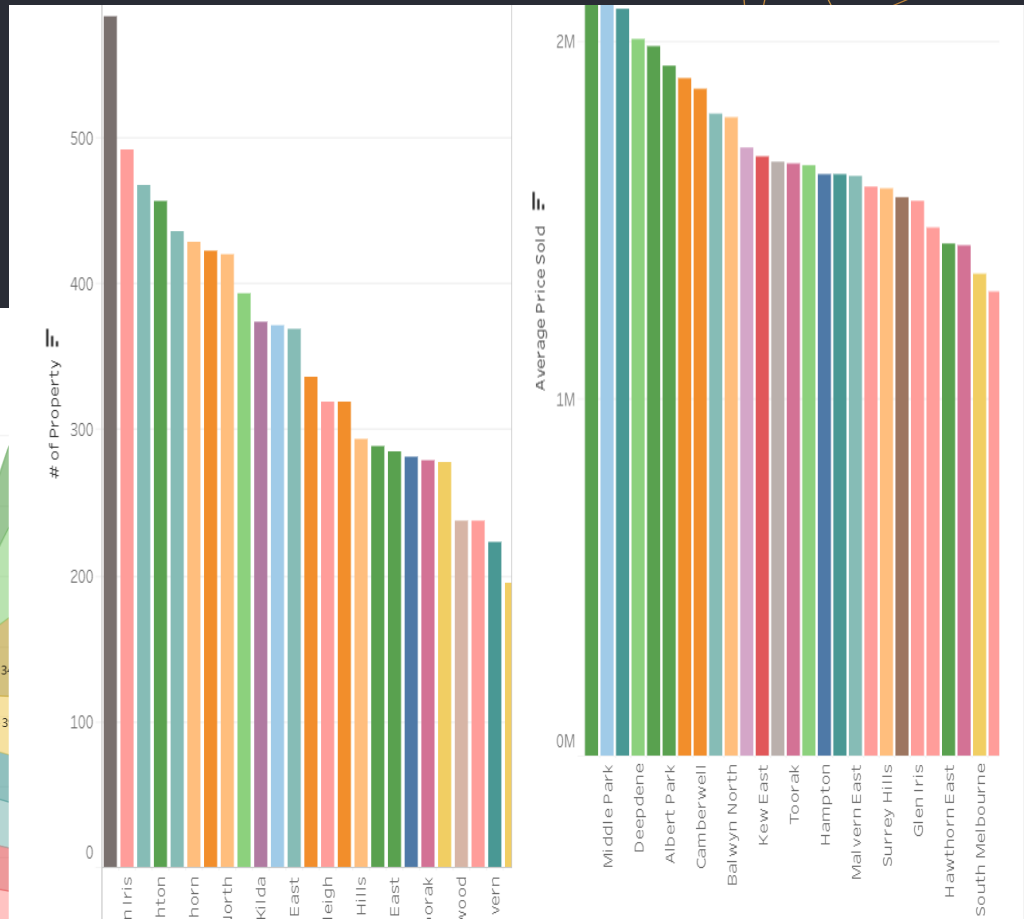
```
In [138]: predictions = model.predict(X_test)

In [139]: plt.scatter(y_test, predictions)

Out[139]: <matplotlib.collections.PathCollection at 0x1b29cffb3c8>
```



Visualization

- Embed visualization from Tableau





Consideration for further enhancement

- Collection & loading additional data into app
- Enhancement of prediction model/algorithm on house prices
- Utilize data available to add on more meaningful charts
- Migrate data onto Cloud platform, enablement of refreshment on a regular basis

Findings & Challenges

- Data collection and cleaning indeed requires lot of time (90% of the project). Data == GOLD .
- Integration of multiple technologies/platform within limited time
- Selection of the prediction model
- Building application is a team effort

HUGE THANKS!

Instruction Team:
David Palamara, Oscar, Hamim
Nathanael Lampe and Ifrin

Bootcamp Team:
Gabriel O'Sullivan, SSM
Nicola King, CSM

THANK YOU
FOR YOUR ATTENTION.