BUDT737 Project Report: House Price in England & Wales

Group 8: Amal Byju, Xin Wan, Yi-Hua Huang

Introduction

UK house prices were increasing at a rapid rate over the last 20 years. The average house price rose from 106,406 in 2002 to 273,000 in 2021 (an increase of 257%). London has the highest average house price followed by other areas in the southeast and southwest of England. Houses get cheaper as you go further north. Affordability has declined precipitously for first-time buyers. Although the residential construction activity rose in 2021, completions continue to be below by quite a margin of the target set by the government, which is 300,000 new homes every year. We were interested in this project because we wanted to find out the past trends in house prices and more importantly, come up with insights and conclusions based on what the data tells us.

About the Dataset

The dataset that we chose was collected and curated by the HM Land Registry. Each row represents a residential property transaction in England & Wales from 2002 to 2021. The total size is 3GB with 19 million rows of records and 16 columns.

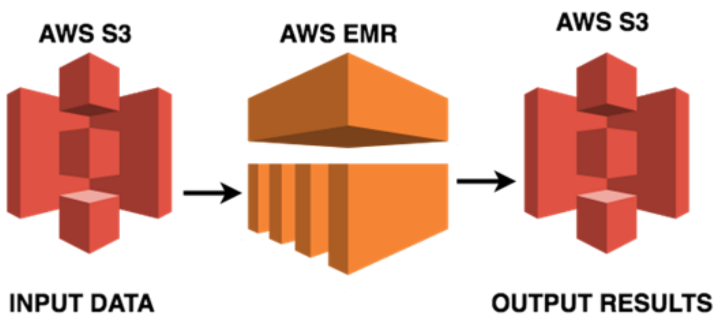
Column	Explanation
Transaction ID	A unique transaction ID for every purchase
Price	Price of the house
Date of Transfer	Date when the transaction was completed
Postcode	Postcode that was used at the time of transaction
Property Type	Categorical variable with 5 levels: D = Detached house, S = Semi-detached house, T = Terraced house, F = Flats/Maisonettes and O = Other.
Old/New	Categorical variable with 2 levels: Y = a recently built property, N = established building.
Duration	Categorical variable with 3 levels: F =

	Freehold, L = Leasehold and U = Unknown.
PAON	House number/ name
SAON	If a property is divided into separate units, the PAON identifies the property while the SAON identifies the unit.
Street	Street address
Locality	Locality
Town/ City	Town/ City
District	District
County	County
PPD Category Type	Type of Price Paid Transaction: A = Standard Price Paid entry. B = Additional Price Paid entry. However, HM Land Registry started collecting information about category B transactions as late as October 2013.
Record Status - Monthly File Only	This column is specific to record-keeping purposes and will not be used in our analysis.

Business Questions

1. What are the different factors that affect house prices?
2. How house prices have changed over the years in different counties or cities of England and Wales.
3. Predict the price of a house given some input features.

Data Infrastructure



We stored the dataset in a S3 bucket and created an EMR cluster with 1 master node (m4 2xlarge) and 5 core nodes (m4 xlarge). Furthermore, we configured the cluster to include RStudio and sparklyr as these tools are essential for our analysis. The S3 bucket has 3 directories: logs for storing the logs generated by the cluster, data for storing the dataset, scripts for the R script and output for the captured results of the R script. After the cluster started running, we accessed RStudio which was at port 8787 of the master node. We ran the script and generated the results in this environment.

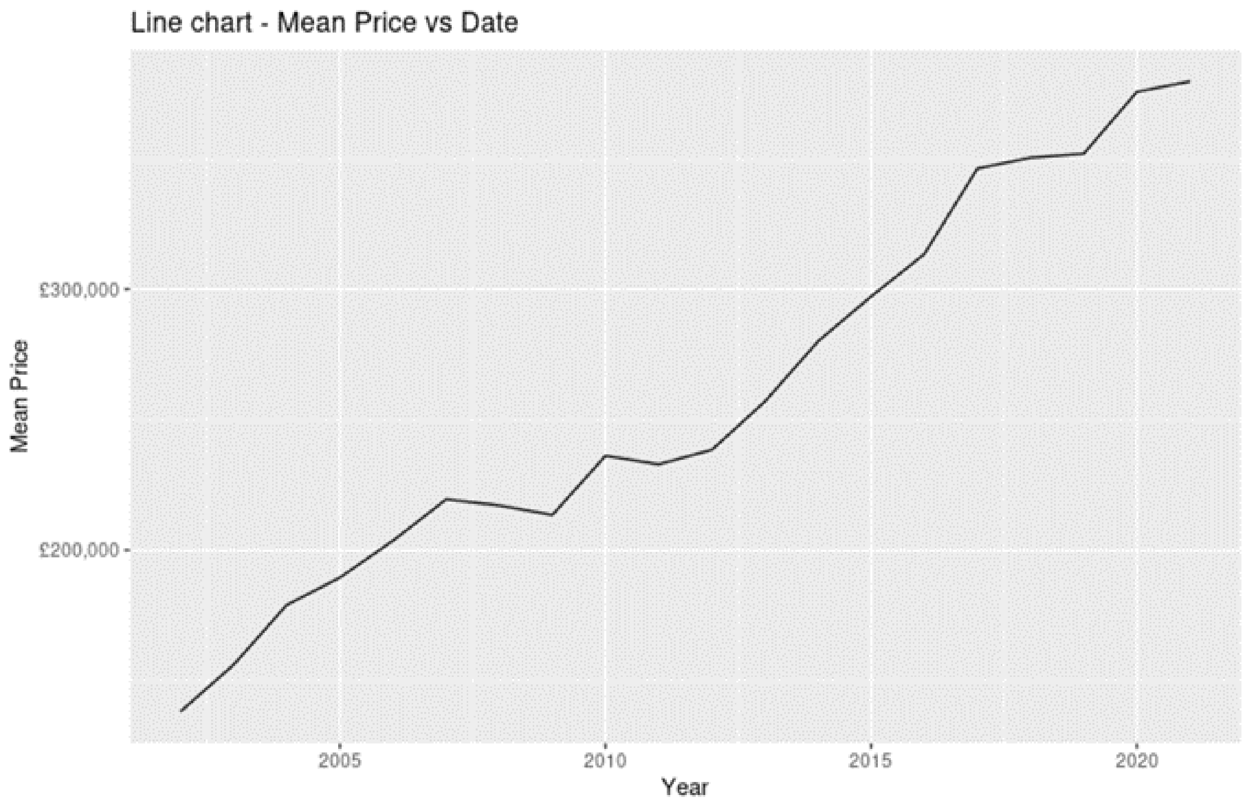
```
# In RStudio environment
# Connect to spark
sc <- spark_connect(master = "yarn-client",
                    spark_home = "/usr/lib/spark/")

# Read the house prices csv file from S3
house_data <- spark_read_csv(sc, name = 'house_data',
                             path = 's3://budt737-group8-s3/data/house_data.csv')
```

Data Cleaning and Transformation in Spark

We performed the following data cleaning tasks:

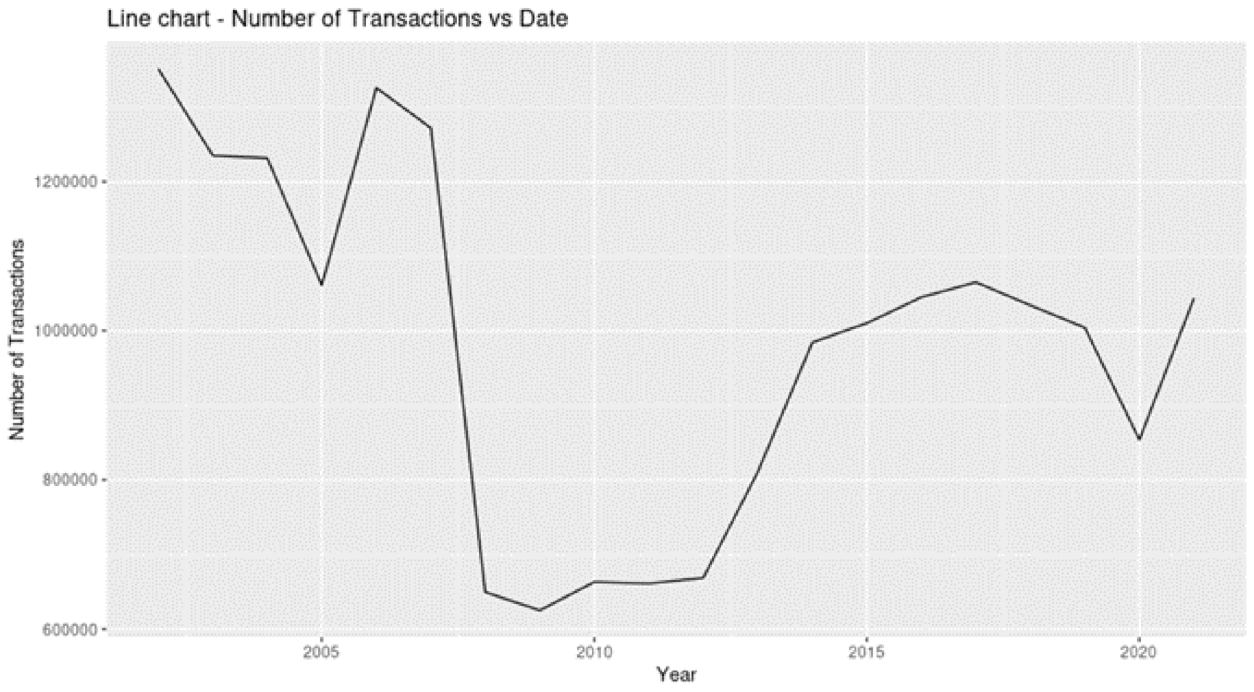
1. Removed columns containing NA (missing) values
2. Few columns contained abbreviations, so those values were replaced with full forms.
3. Removed columns that have discrepancies as the HM Land registrar changed the data collection procedure of these columns.
4. Removed a column that contained information which is specific to the data collection procedure of the HM Land Registrar since this is not required for our analysis.
5. We found that the Price column was right skewed, so we applied log transformation to get a normal distribution.
6. Converted the data type of ‘Date of Transfer’ column from character to date.

Descriptive Analytics in Spark



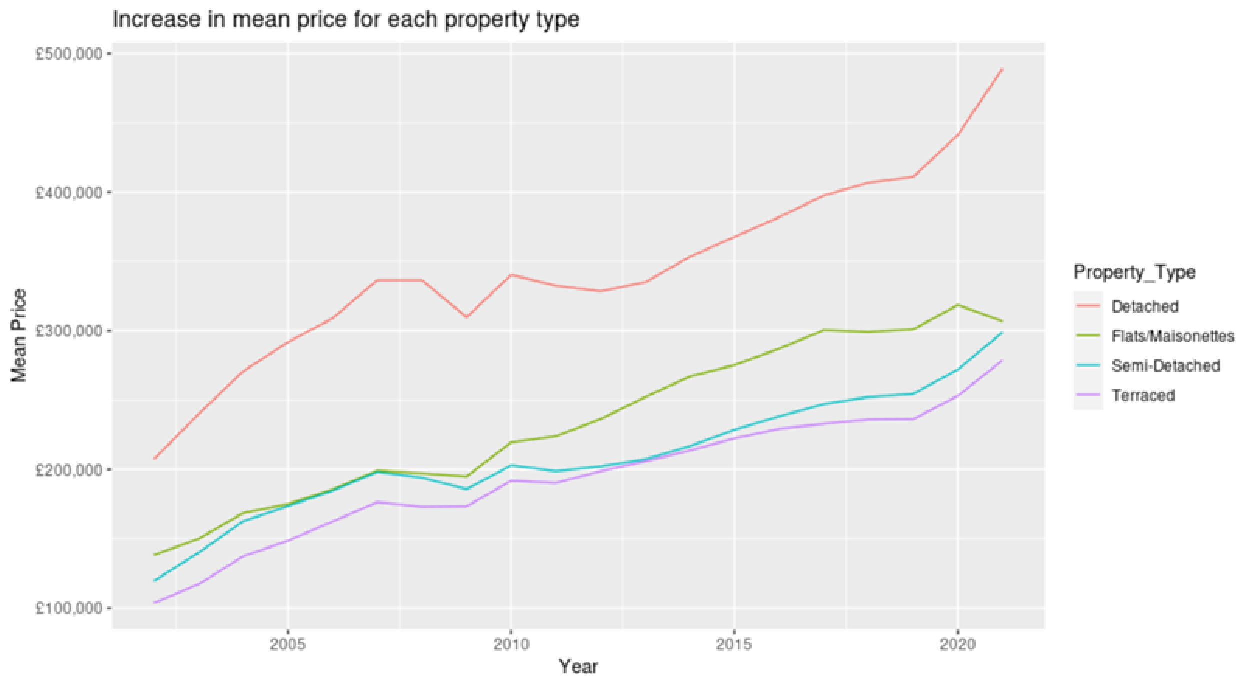
```
# Line chart - mean price vs date
house_data %>%
  mutate(year = year(Date_of_Transfer)) %>%
  group_by(year) %>%
  summarize(Mean_Price = mean(Price)) %>%
  ggplot(aes(x = year, y = Mean_Price)) +
  geom_line() +
  scale_y_continuous(labels=scales::dollar_format(prefix = '£')) +
  labs(x = "Year",
       y = "Mean Price",
       title = "Line chart - Mean Price vs Date")
```

We decided to make a simple line chart of the mean house price (in pounds) over the years. As expected, there is an upward trend.



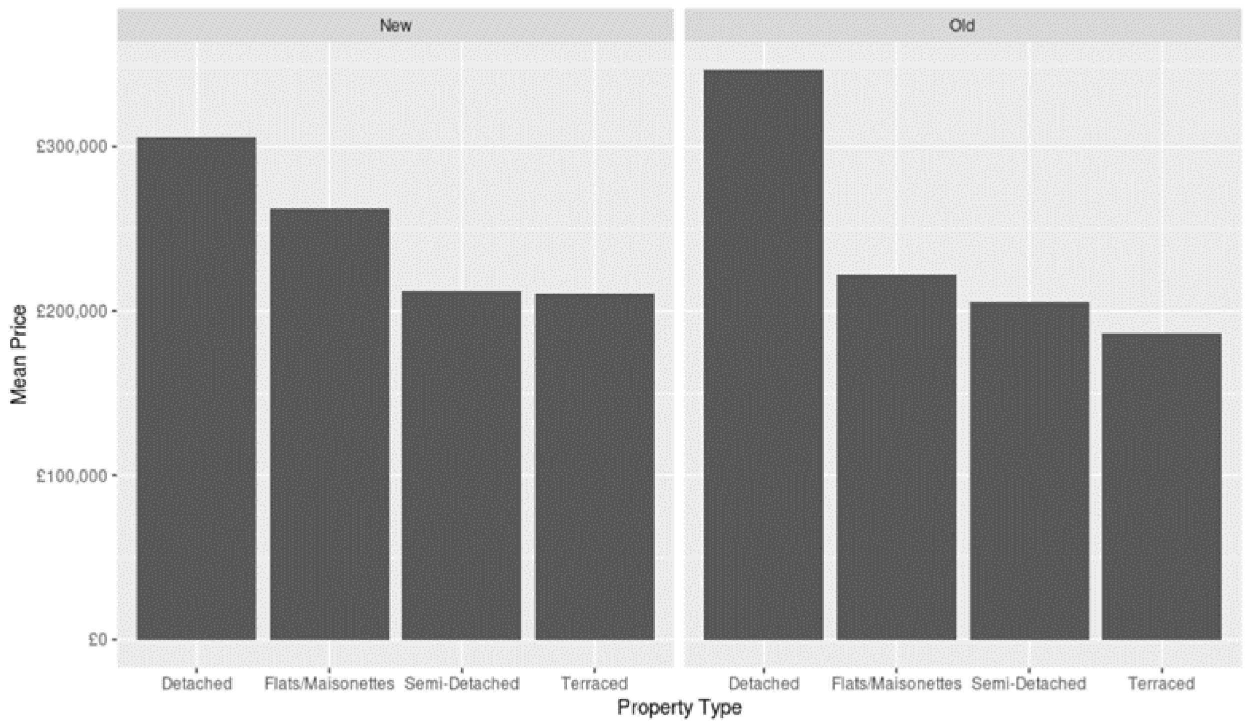
```
# Line chart - Number of Transactions vs Date
house_data %>%
  group_by(year(Date_of_Transfer)) %>%
  summarize(Num_Transactions = n()) %>%
  ggplot(aes(x = `year(Date_of_Transfer)`, y = Num_Transactions)) +
  geom_line() +
  labs(x = 'Number of Transactions',
       y = 'Year',
       title = 'Line chart - Number of Transactions vs Date')
```

Next, we decided to make a line plot of the number of property transactions in each year. The steep decrease in demand is probably because of the Great Recession of 2008.



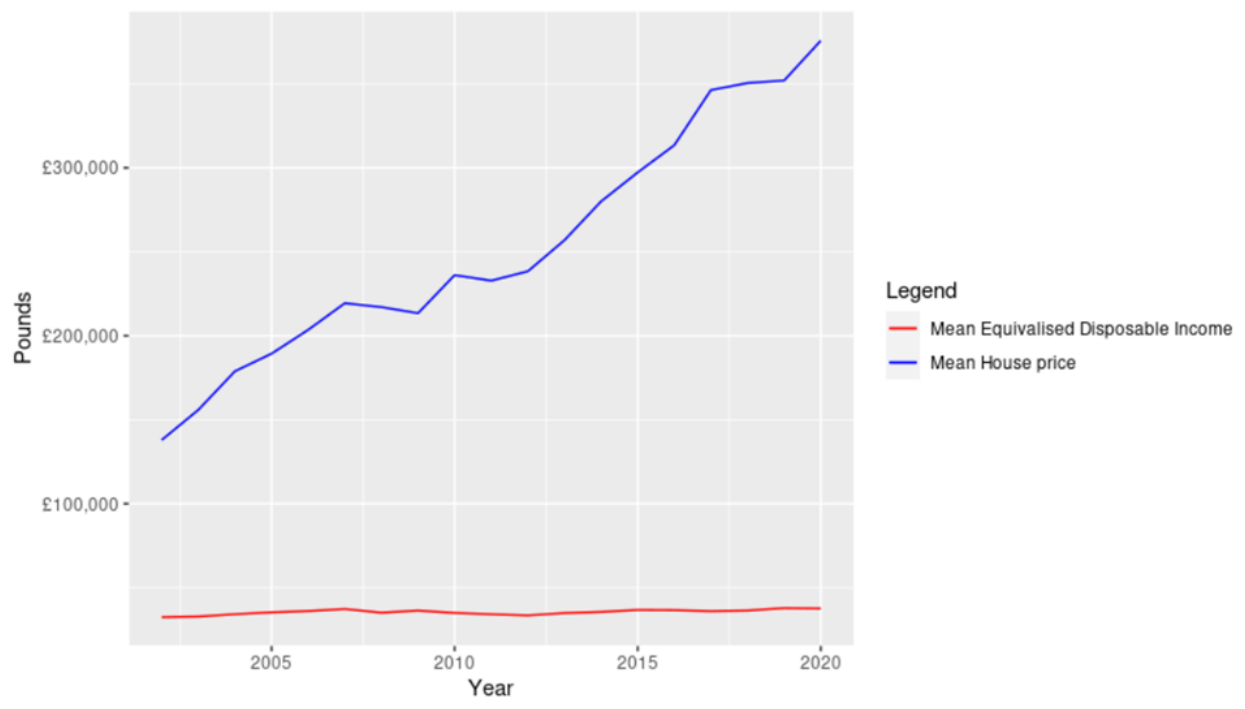
```r
# Line plot of increasing prices for each property type
house_data %>%
  filter(Property_Type != 'Other') %>%
  group_by(Property_Type, year(Date_of_Transfer)) %>%
  summarize(Mean_Price = mean(Price)) %>%
  ggplot(aes(x = `year(Date_of_Transfer)`, y = Mean_Price, color = Property_Type)) +
  geom_line() +
  scale_y_continuous(labels=scales::dollar_format(prefix = '£')) +
  labs(x = 'Year',
       y = 'Mean Price',
       title = 'Increase in mean price for each property type')
```

Here is a line chart of the average house price for each property type. Detached houses are more expensive because of the larger space. Terraced houses are usually more energy efficient and cheap. However, one of the main downsides with terraced houses is noise.



```
# Bar chart
house_data %>%
  filter(Property_Type != 'Other') %>%
  group_by(Old_New, Property_Type) %>%
  summarize(Mean_Price = mean(Price)) %>%
  ggplot(aes(x = Property_Type, y = Mean_Price)) +
  geom_col() +
  scale_y_continuous(labels=scales::dollar_format(prefix = '£')) +
  facet_wrap(~Old_New) +
  labs(x = "Property Type",
       y = "Mean Price")
```

Here is the bar chart that shows the mean house price for each property type. On average, the prices of new properties are higher. However, it should be noted that the average house price of old detached houses is the highest.



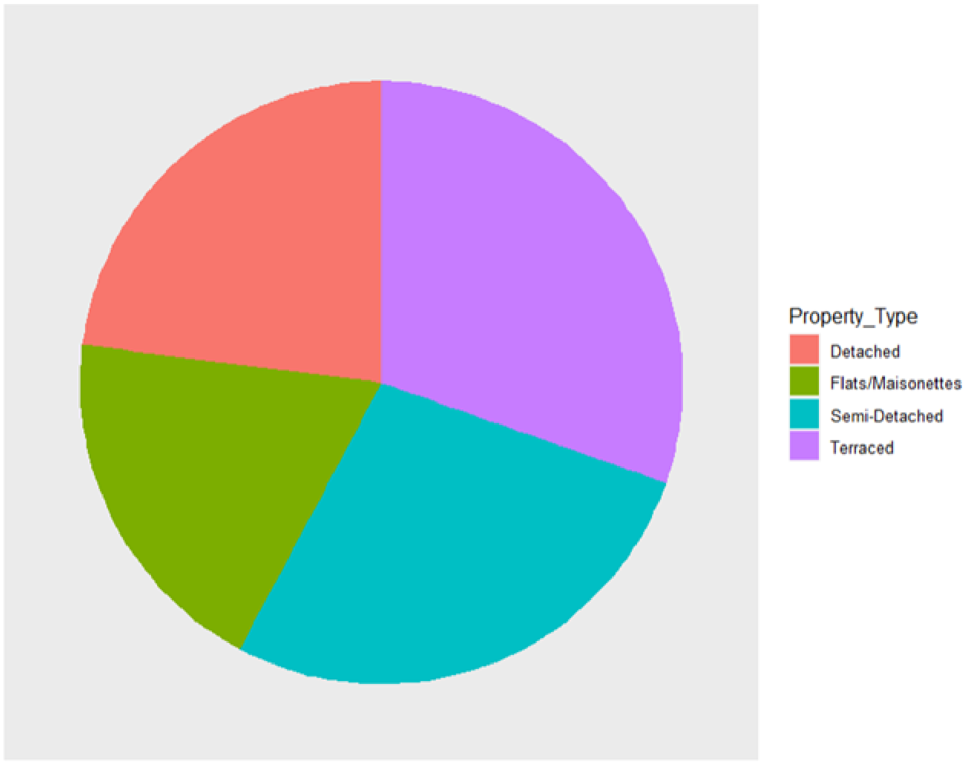
```r
colors <- c("Mean Equivalised Disposable Income" = "red", "Mean House Price" = "blue")
ggplot(time_series_price_income, aes(x = year, y = meanPrice)) +
  geom_line(aes(color = "Mean House Price")) +
  geom_line(data = time_series_price_income, aes(x = year, y =
meanEquivalisedDisposableIncome, color = "Mean Equivalised Disposable Income")) +
  labs(x = "Year",
       y = "Pounds",
       color = "Legend") +
  scale_y_continuous(labels=scales::dollar_format(prefix = '£')) +
  scale_color_manual(values = colors)
```

Here we answer a business question: How house prices have changed over the years and how does it compare to the mean earnings over the years.

The red line (which represents the income) does have an upward trend but it is overshadowed by the steep upward trend of mean house price.

Houses have a low interest rate. So people buy them for investment. That is the reason why the average house price went up.

There are foreign investors too.

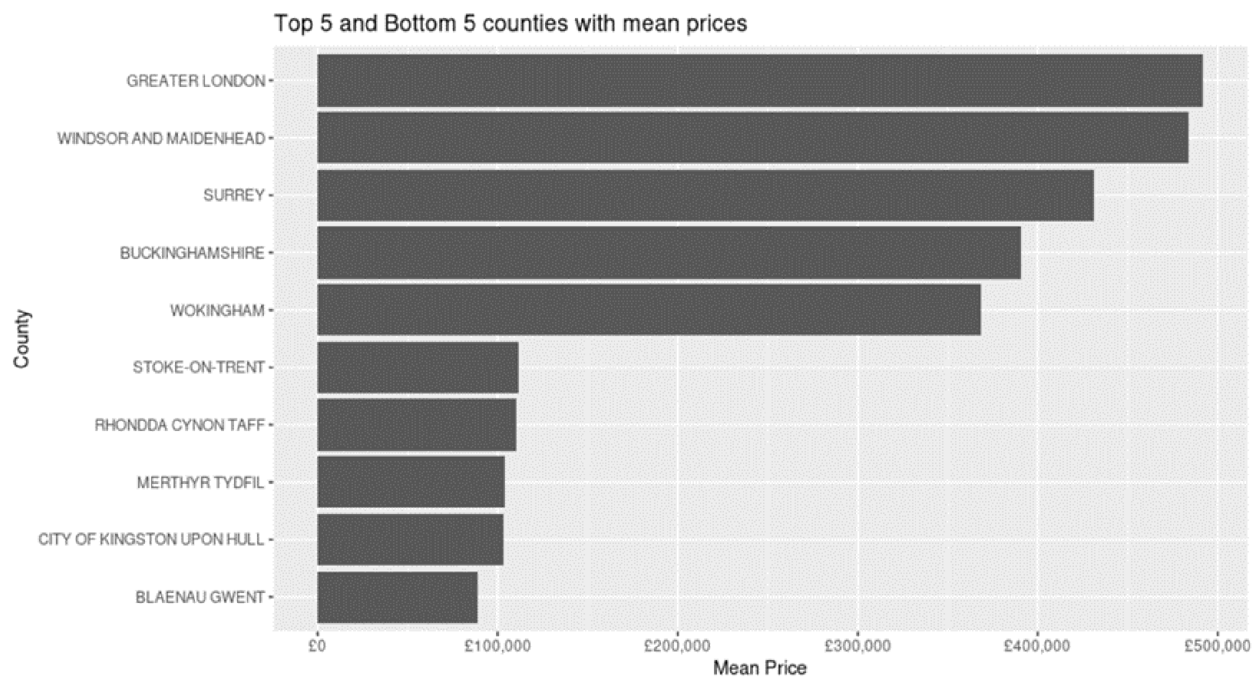


```r
# Pie chart of number of transactions for each property type
house_data %>%
  filter(Property_Type != 'Other') %>%
  ggplot(aes(x = 1, fill = Property_Type)) +
  geom_bar(width = 0.1)+
  coord_polar(theta = "y") +
  theme(line = element_blank(),
        axis.ticks = element_blank(),
        axis.text = element_blank(),
        axis.title = element_blank())
```

This is a pie chart that shows the proportion of transactions for each property type from 2002 to 2021.

As expected, the number of transactions of terraced houses is the highest (because it is cheap).

The number of transactions of detached properties is low because of its price.



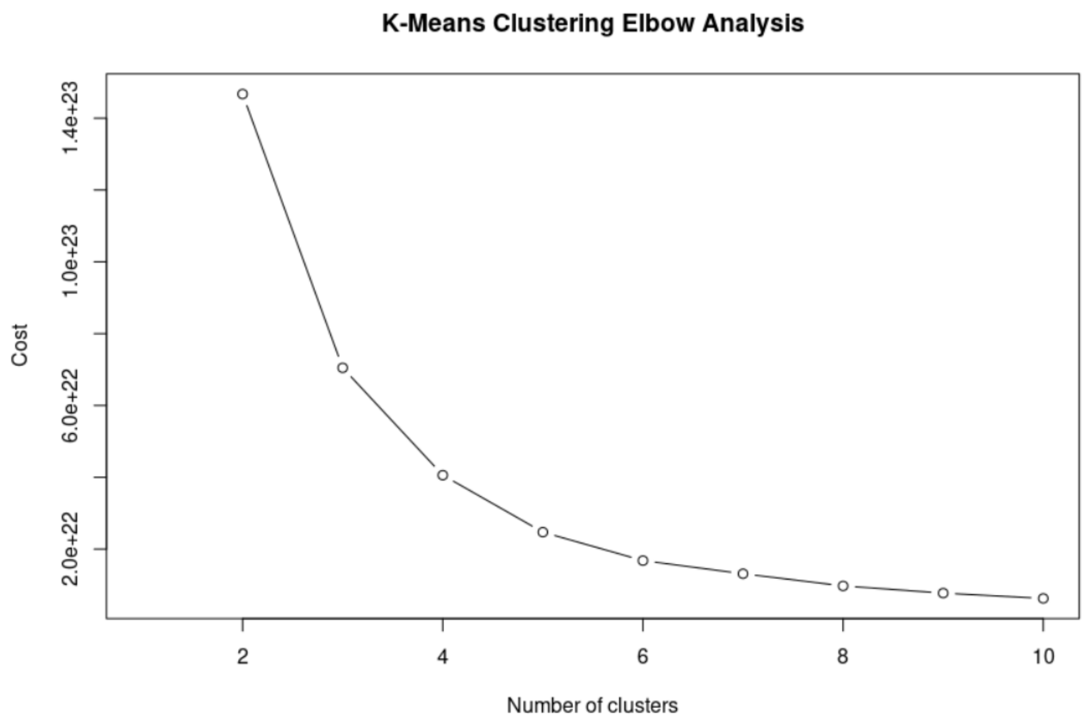
```r
# Top 5 and bottom 5 counties by mean price
county_mean_prices <-
  house_data %>%
  group_by(County) %>%
  summarize(MeanPrice = mean(Price))

rbind(
  county_mean_prices %>%
    slice_max(MeanPrice, n = 5),
  county_mean_prices %>%
    slice_min(MeanPrice, n = 5)
) %>%
  arrange(desc(MeanPrice)) %>%
  ggplot(aes(x = reorder(County, MeanPrice), y = MeanPrice)) +
  geom_col() +
  scale_y_continuous(labels=scales::dollar_format(prefix = '£')) +
  coord_flip() +
  labs(x = 'County',
       y = 'Mean Price',
       title = 'Top 5 and Bottom 5 counties with mean prices')
```

Here is a bar chart that shows the top 5 and bottom 5 counties by average house price.

When we checked out the top 5 counties with the highest average price, we found out that London had the highest average price followed by counties in England that are close to London. This is probably because there might be commuters who live in these regions but work in London, earning London wages (which are higher). Three out of the 5 counties with the lowest prices are in Wales.

Explanatory Model: K-Means Clustering

To find the optimum number of clusters for the K-Means algorithm, we performed elbow analysis.

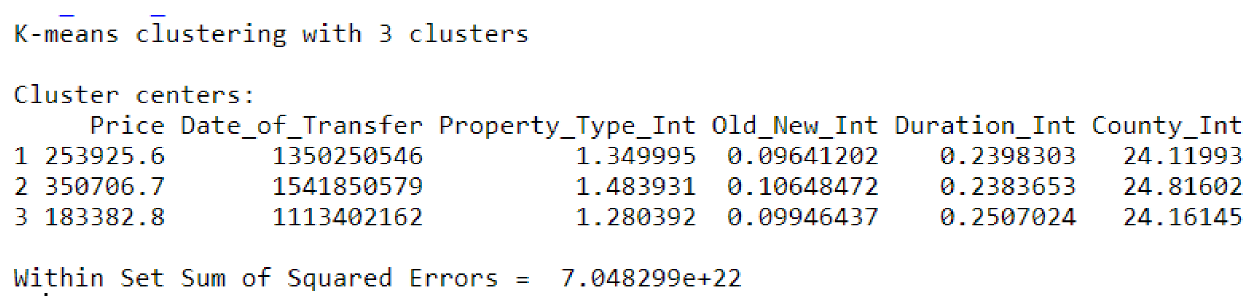


```
# Elbow analysis for k-means
cost_vector = rep(0, 10)
cost_vector[1] = NA

for(i in 2:10) {
  model <- ml_kmeans(house_data_cluster, features = colnames(house_data_cluster), k = i)
  cost_vector[i] <- model$cost
}
plot(1:10, cost_vector, type = 'b', xlab = 'Number of clusters', ylab = 'Cost')
```

The elbow occurs at k = 3 so that is the number of clusters we have selected for the k-means algorithm.

K-Means algorithm with 3 clusters

```
k_means_model <- ml_kmeans(house_data_cluster, features = colnames(house_data_cluster), k =
3)
k_means_model
```



```
K-means clustering with 3 clusters

Cluster centers:
     Price Date_of_Transfer Property_Type_Int Old_New_Int Duration_Int County_Int
1 253925.6       1350250546          1.349995  0.09641202    0.2398303   24.11993
2 350706.7       1541850579          1.483931  0.10648472    0.2383653   24.81602
3 183382.8       1113402162          1.280392  0.09946437    0.2507024   24.16145

Within Set Sum of Squared Errors =  7.048299e+22
```

<u>Results of K-Means</u>

- Features had to be coerced to numbers so some columns don't make sense.
- We can see that the Price (in pounds) is vastly different in the 3 clusters.

Regression Models

Here we address a business question: to predict the price of a house given its input features. We have tried 2 models: Random Forest and Gradient Boosted Tree.

<u>Make 70/30 train-test split</u>

```
# train-test split
partitions <-
house_data %>%
  sdf_random_split(training = 0.7, testing = 0.3)

train <- partitions$training %>% copy_to(sc, ., name = "train")
test <- partitions$testing %>% copy_to(sc, ., name = "test")
```

In this dataset, we made a random split of 70% training and 30% testing. The training dataset is called train and the testing dataset is called test.

Random Forest Model

```
# Random Forest Model
(formula_rf <- logPrice ~ Date_of_Transfer_num+Property_Type+Old_New+Duration+County)
rf_model <- ml_random_forest(train,
                             formula_rf,
                             num_trees = 100,
                             max_depth = 3,
                             type = "regression")
glance(rf_model)
```

```
# A tibble: 1 × 5
  num_trees total_num_nodes max_depth impurity subsampling_rate
      <int>           <int>     <int> <chr>               <dbl>
1       100            1464         3 variance                1
```

Since the distribution of Price is skewed, we have taken log(Price) since it results in a normal distribution. We trained the model on the training data to predict log(Price) based on some features. The number of trees is 100 and the maximum depth of each tree is 3.

Gradient Boosted Tree

```
# Gradient Boosted Tree
(formula_gbt <- logPrice ~ Date_of_Transfer_num+Property_Type+Old_New+Duration+County)
gbt_model <- ml_gbt_regressor(train,
                              formula_gbt,
                              max_iter = 100,
                              max_depth = 3)
glance(gbt_model)
```

```
# A tibble: 1 × 7
  num_trees total_num_nodes max_depth impurity step_size loss_type subsampling_rate
      <int>           <int>     <int> <chr>        <dbl> <chr>                <dbl>
1       100            1500         3 variance       0.1 squared                  1
```

The generated gradient boosted tree has 100 trees and the maximum depth of each tree is 3. The step size is 0.1 (default).

Model Evaluation

Now, we'll compare the performance of both models using the test dataset.

```
# Model Evaluation
test_y <- test['logPrice'] %>% pull() # Pulling the target variable, which is logPrice
sd_test_y <- sd(test_y)               # Standard deviation of the target variable

predicted_y_rf <- predict(rf_model, test) # Predict RF model on test data
resid_rf <- test_y - predicted_y_rf       # RF model residuals
rf_rmse <- sqrt(mean(resid_rf ^ 2, na.rm = TRUE)) # Random Forest RMSE
```

```
predicted_y_gbt <- predict(gbt_model, test)         # Predict GBT on test data
resid_gbt <- test_y - predicted_y_gbt               # GBT residuals
gbt_rmse <- sqrt(mean(resid_gbt ^ 2, na.rm = TRUE)) # Gradient Boosted Tree RMSE

tribble(                                             # Comparison table
  ~model, ~RMSE, ~sd,
  "Random Forest", rf_rmse, sd_test_y,
  "Gradient Boosted Tree", gbt_rmse, sd_test_y       # RMSE vs standard deviation
)
```

```
# A tibble: 2 × 3
  model                  RMSE    sd
  <chr>                 <dbl> <dbl>
1 Random Forest         0.617 0.734
2 Gradient Boosted Tree 0.520 0.734
```

As the RMSE of log(Price) of both models are less than the standard deviation of the test dataset, both models perform better than just predicting the average log(Price) for every observation.

Gradient Boosted Tree is the better model with lower test RMSE.

Feature Importance

```
ml_tree_feature_importance(gbt_model)
```

```
                          feature  importance
1             Date_of_Transfer_num 0.094055724
2           Property_Type_Terraced 0.030484077
3                    County_SURREY 0.027328144
4              County_HERTFORDSHIRE 0.026713840
5      Property_Type_Semi-Detached 0.025286061
```

Here, we answer a business question: what are the different factors that affect house price? Since the Gradient Boosted Tree performed better, we extracted the feature importance of this model. In the above image, we see the most important features, which are Date of Transfer, Property Type and County.

Time Series Forecasting

We thought that it was a good idea to forecast average house prices in the future as this might be of interest to the HM Land Registry and the government. So we used the TBATS model for this purpose.

First, we begin with preparing the data. We'll use average house prices from 2002 to 2019 as the training dataset and then the model will forecast house prices 2 years into the future. Finally, we'll compare the model's predictions with the actual mean house prices in 2020 and 2021.

Data Preparation

```
time_series <-                              # Average house prices 2002 to 2021
  house_data %>%
  mutate(year = year(Date_of_Transfer)) %>%
  group_by(year) %>%
  summarize(meanPrice = mean(Price),
            transaction_count = n()) %>%
  arrange(year) %>%
  copy_to(sc, ., name = "time_series")

train_time_series <-                       # Training data: house prices from 2002 to 2019
  time_series %>%
  filter(year <= 2019) %>%
  arrange(year) %>%
  copy_to(sc, ., name = "train_time_series")

test_time_series <-                        # Testing data: house prices from 2020 to 2021
  time_series %>%
  filter(year > 2019) %>%
  arrange(year) %>%
  copy_to(sc, ., name = "test_time_series")

ts_meanPrice <- time_series['meanPrice'] %>%  # Time series of the entire data
  pull() %>%
  ts(., start = c(2002), end = c(2021), frequency = 1)

train_ts_meanPrice <- train_time_series['meanPrice'] %>% # Time series of training data
  pull() %>%
  ts(., start = c(2002), end = c(2019), frequency = 1)

test_ts_meanPrice <- test_time_series['meanPrice'] %>% # Time series of testing data
  pull() %>%
  ts(., start = c(2020), end = c(2021), frequency = 1)
```

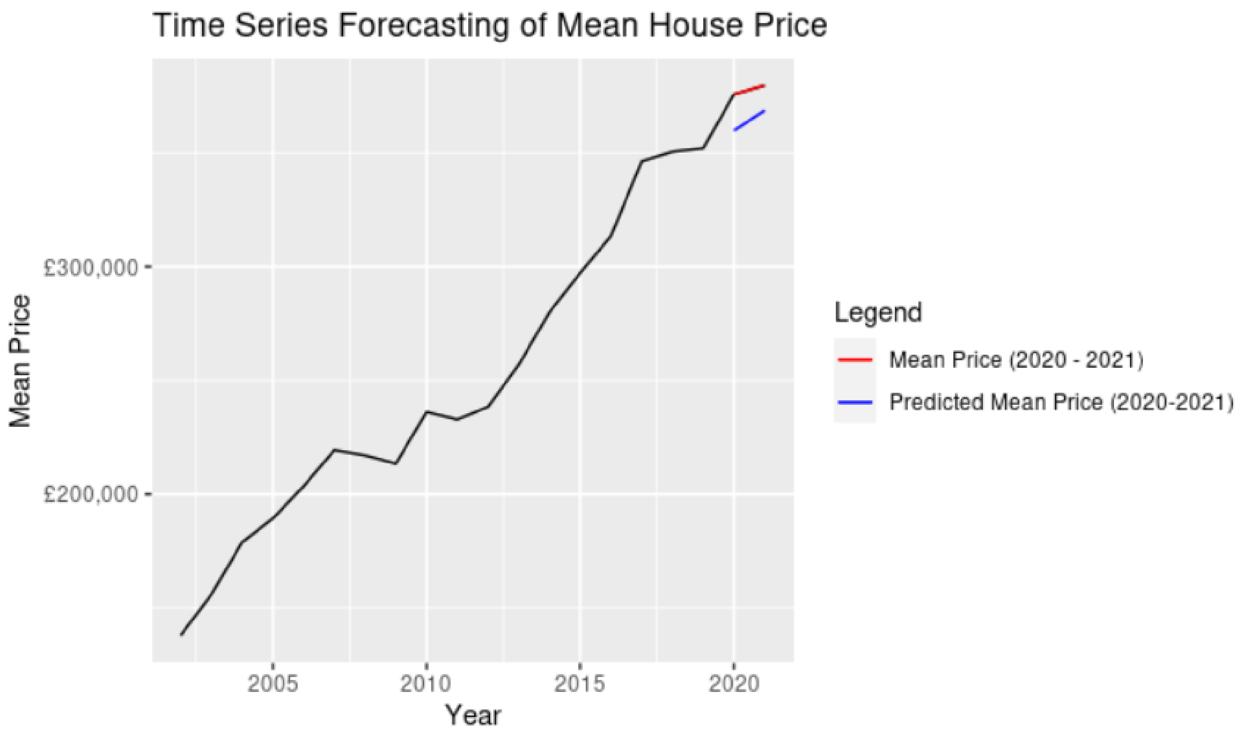
Training the model

```
tbats_model <- tbats(train_ts_meanPrice)
summary(tbats_model)
```

```
                  Length Class  Mode
lambda             0     -none- NULL
alpha              1     -none- numeric
beta               1     -none- numeric
damping.parameter  1     -none- numeric
gamma.values       0     -none- NULL
ar.coefficients    0     -none- NULL
ma.coefficients    0     -none- NULL
likelihood         1     -none- numeric
optim.return.code  1     -none- numeric
variance           1     -none- numeric
AIC                1     -none- numeric
parameters         2     -none- list
seed.states        2     -none- numeric
fitted.values     18     ts     numeric
errors            18     ts     numeric
x                 36     -none- numeric
seasonal.periods   0     -none- NULL
y                 18     ts     numeric
call               2     -none- call
series             1     -none- character
method             1     -none- character
```

Forecasting

```
# Forecast on test data
fore_tbats = forecast::forecast(tbats_model, h=2)
df_tbats = as.data.frame(fore_tbats)

# Plot the results
colors <- c("Mean Price (2020 - 2021)" = "red", "Predicted Mean Price (2020-2021)" = "blue")
ggplot(as.data.frame(ts_meanPrice), aes(time(ts_meanPrice), x)) +
  geom_line() +
  geom_line(data = as.data.frame(test_ts_meanPrice), aes(time(test_ts_meanPrice), x, color =
"Mean Price (2020 - 2021)")) +
  geom_line(data = df_tbats, aes(time(test_ts_meanPrice), `Point Forecast`, color =
"Predicted Mean Price (2020-2021)")) +
  labs(x = "Year",
       y = "Mean Price",
       color = "Legend",
       title = "Time Series Forecasting of Mean House Price"
       ) +
  scale_y_continuous(labels=scales::dollar_format(prefix = '£')) +
  scale_color_manual(values = colors)
```



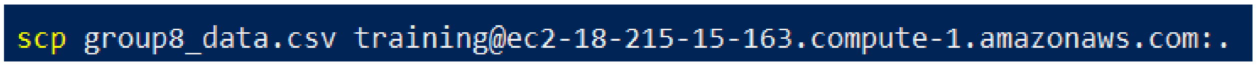
Now, let us calculate the RMSE.

```
residual <- df_tbats$`Point Forecast` - test_ts_meanPrice
sqrt(mean(residual^2))
Output: 13712.74
```

The RMSE is 13712.74.

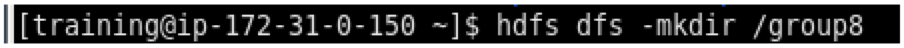
Further Analysis with Impala: Data Infrastructure

We performed a descriptive analysis with our 3GB dataset using Impala. We stored the dataset in our shared google drive and downloaded it into our local computer. Then we used the scp command to copy our dataset (csv file) to our Cloudera virtual machine which we launched from AWS.

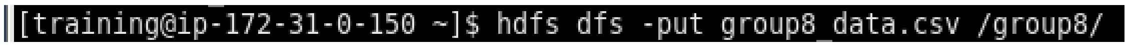


```
scp group8_data.csv training@ec2-18-215-15-163.compute-1.amazonaws.com:.
```

After the dataset was copied into our Cloudera machine, we created a new working directory named “group8” to store our dataset for further analysis. We used the hdfs command to move our dataset into the “group8” working directory.

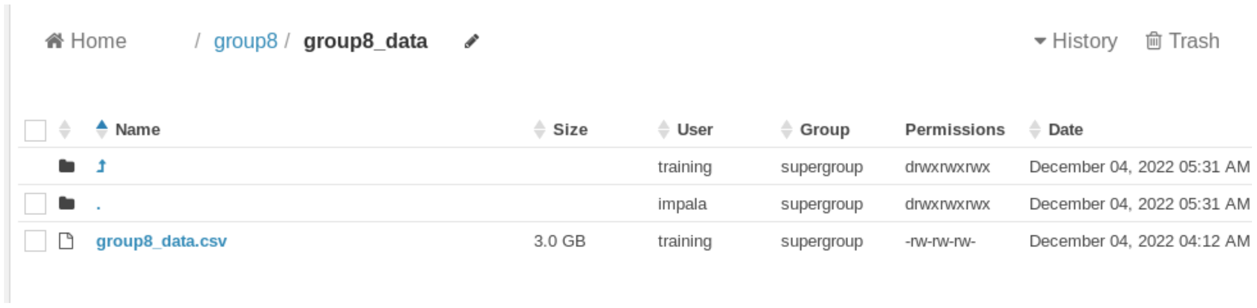


```
[training@ip-172-31-0-150 ~]$ hdfs dfs -mkdir /group8
```



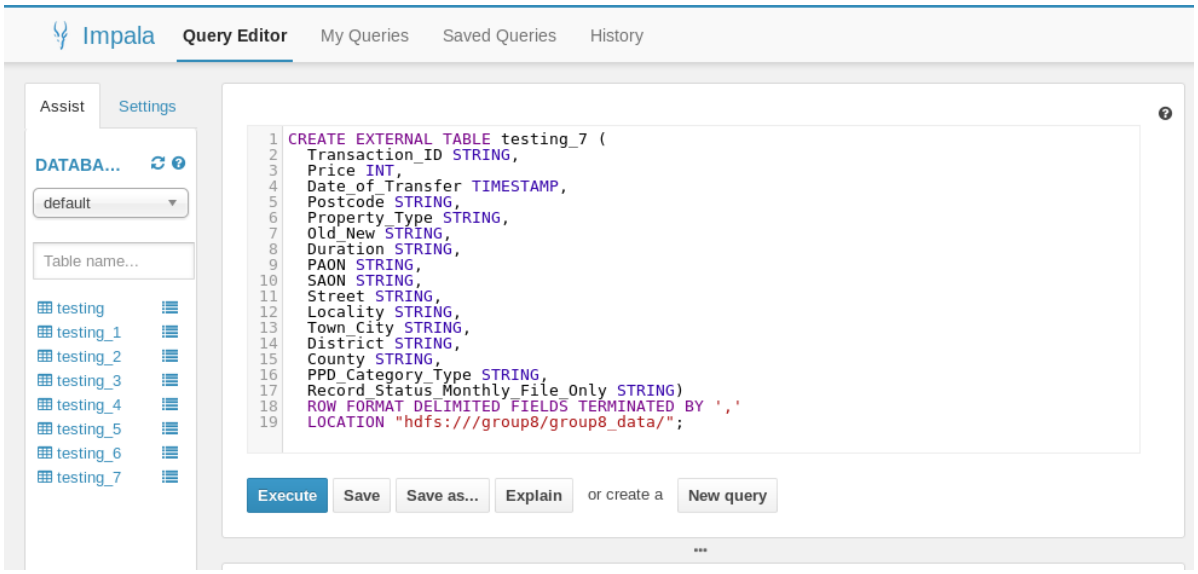
```
[training@ip-172-31-0-150 ~]$ hdfs dfs -put group8_data.csv /group8/
```

We chose to use Impala for our descriptive analysis using SQL because it is fast and easy to access to handle big data.



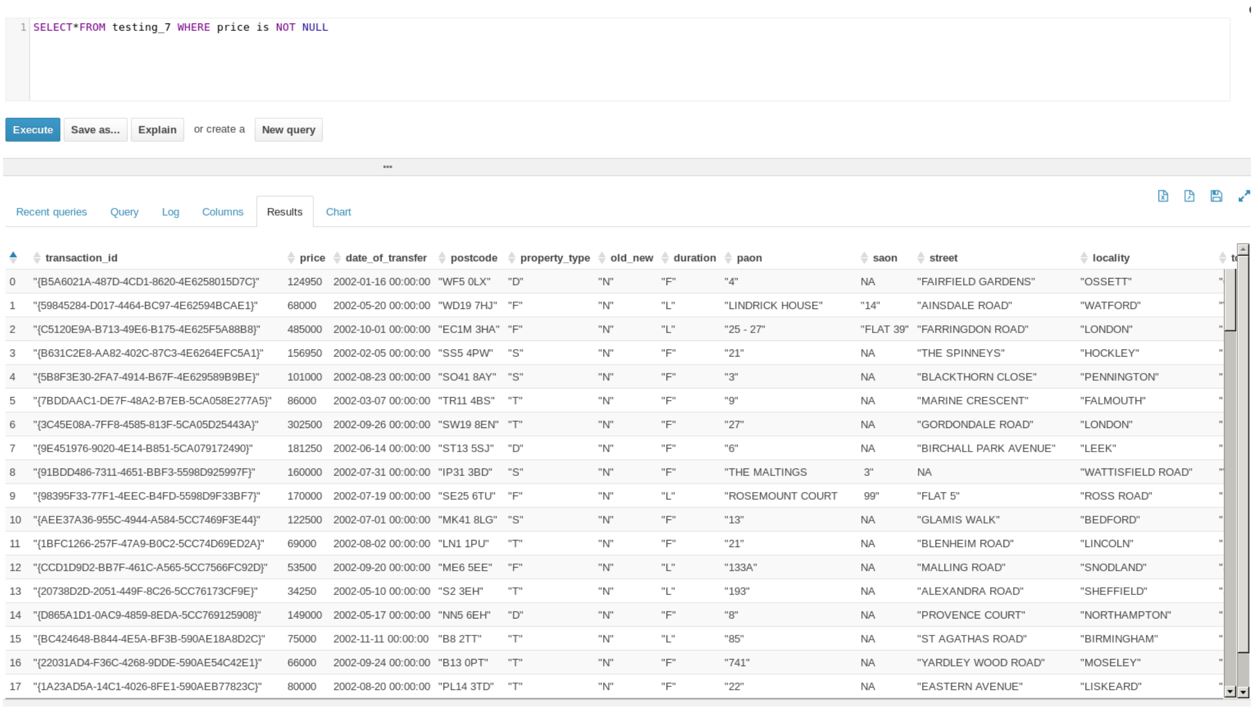
	Name	Size	User	Group	Permissions	Date
	↥		training	supergroup	drwxrwxrwx	December 04, 2022 05:31 AM
	.		impala	supergroup	drwxrwxrwx	December 04, 2022 05:31 AM
	group8_data.csv	3.0 GB	training	supergroup	-rw-rw-rw-	December 04, 2022 04:12 AM

Create SQL Tables



```
CREATE EXTERNAL TABLE testing_7 (
  Transaction_ID STRING,
  Price INT,
  Date_of_Transfer TIMESTAMP,
  Postcode STRING,
  Property_Type STRING,
  Old_New STRING,
  Duration STRING,
  PAON STRING,
  SAON STRING,
  Street STRING,
  Locality STRING,
  Town_City STRING,
  District STRING,
  County STRING,
  PPD_Category_Type STRING,
  Record_Status_Monthly_File_Only STRING)
  ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
  LOCATION "hdfs:///group8/group8_data/";
```

We created a table that covers all 16 variables from our dataset. We took multiple attempts to test and examined errors in certain variables in order to successfully run SQL queries. We noticed that the zip code in the UK is numbers and characters combined, so we would have to convert them into strings. The date of transfer was stored as a TImestamp data type because our date data covers different time zones.



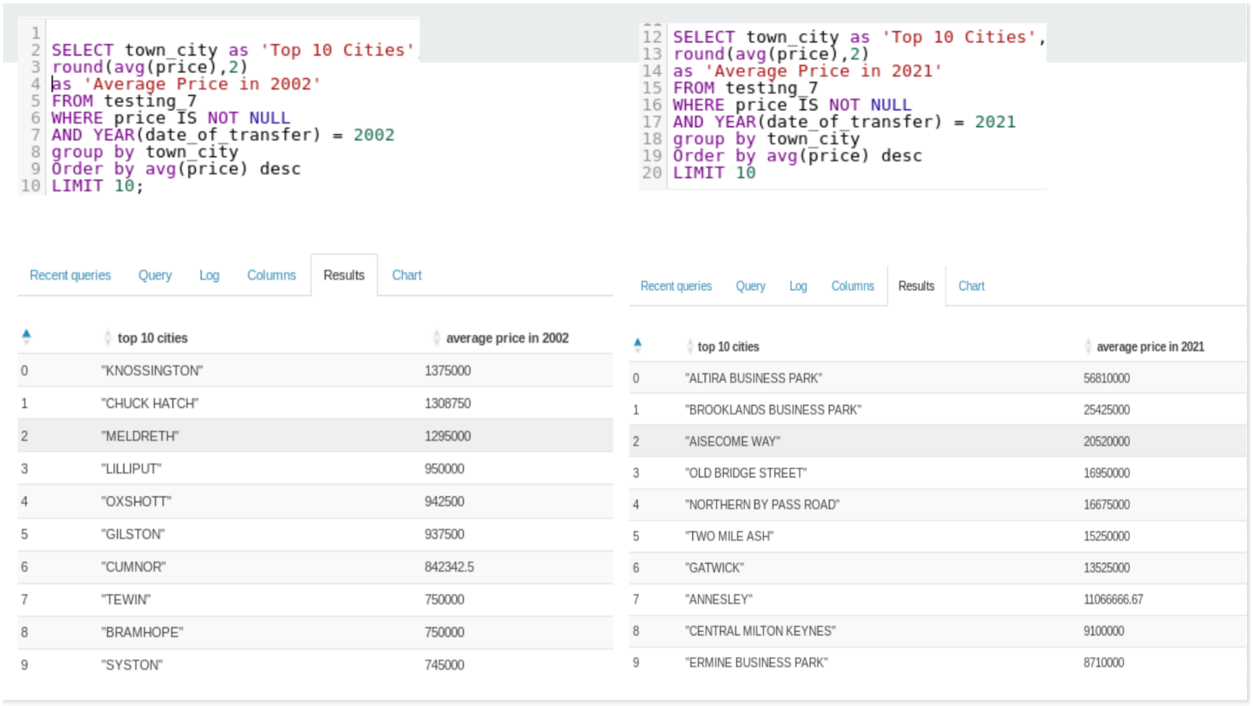
```
SELECT*FROM testing_7 WHERE price is NOT NULL
```

	transaction_id	price	date_of_transfer	postcode	property_type	old_new	duration	paon	saon	street	locality
0	"{B5A6021A-487D-4CD1-8620-4E6258015D7C}"	124950	2002-01-16 00:00:00	"WF5 0LX"	"D"	"N"	"F"	"4"	NA	"FAIRFIELD GARDENS"	"OSSETT"
1	"{59845284-D017-4464-BC97-4E62594BCAE1}"	68000	2002-05-20 00:00:00	"WD19 7HJ"	"F"	"N"	"L"	"LINDRICK HOUSE"	"14"	"AINSDALE ROAD"	"WATFORD"
2	"{C5120E9A-B713-49E6-B175-4E625F5A88B8}"	485000	2002-10-01 00:00:00	"EC1M 3HA"	"F"	"N"	"L"	"25 - 27"	"FLAT 39"	"FARRINGDON ROAD"	"LONDON"
3	"{B631C2E8-AA82-402C-87C3-4E6264EFC5A1}"	156950	2002-02-05 00:00:00	"SS5 4PW"	"S"	"N"	"F"	"21"	NA	"THE SPINNEYS"	"HOCKLEY"
4	"{5B8F3E30-2FA7-4914-B67F-4E629589B9BE}"	101000	2002-08-23 00:00:00	"SO41 8AY"	"S"	"N"	"F"	"3"	NA	"BLACKTHORN CLOSE"	"PENNINGTON"
5	"{7BDDAAC1-DE7F-48A2-B7EB-5CA058E277A5}"	86000	2002-03-07 00:00:00	"TR11 4BS"	"T"	"N"	"F"	"9"	NA	"MARINE CRESCENT"	"FALMOUTH"
6	"{3C45E08A-7FF8-4585-813F-5CA05D25443A}"	302500	2002-09-26 00:00:00	"SW19 8EN"	"T"	"N"	"F"	"27"	NA	"GORDONDALE ROAD"	"LONDON"
7	"{9E451976-9020-4E14-B851-5CA079172490}"	181250	2002-06-14 00:00:00	"ST13 5SJ"	"D"	"N"	"F"	"6"	NA	"BIRCHALL PARK AVENUE"	"LEEK"
8	"{91BDD486-7311-4651-BBF3-5598D925997F}"	160000	2002-07-31 00:00:00	"IP31 3BD"	"S"	"N"	"F"	"THE MALTINGS	3"	NA	"WATTISFIELD ROAD"
9	"{98395F33-77F1-4EEC-B4FD-5598D9F33BF7}"	170000	2002-07-19 00:00:00	"SE25 6TU"	"F"	"N"	"L"	"ROSEMOUNT COURT	99"	"FLAT 5"	"ROSS ROAD"
10	"{AEE37A36-955C-4944-A584-5CC7469F3E44}"	122500	2002-07-01 00:00:00	"MK41 8LG"	"S"	"N"	"F"	"13"	NA	"GLAMIS WALK"	"BEDFORD"
11	"{1BFC1266-257F-47A9-B0C2-5CC74D69ED2A}"	69000	2002-08-02 00:00:00	"LN1 1PU"	"T"	"N"	"F"	"21"	NA	"BLENHEIM ROAD"	"LINCOLN"
12	"{CCD1D9D2-BB7F-461C-A565-5CC7566FC92D}"	53500	2002-09-20 00:00:00	"ME6 5EE"	"F"	"N"	"L"	"133A"	NA	"MALLING ROAD"	"SNODLAND"
13	"{20738D2D-2051-449F-8C26-5CC76173CF9E}"	34250	2002-05-10 00:00:00	"S2 3EH"	"T"	"N"	"L"	"193"	NA	"ALEXANDRA ROAD"	"SHEFFIELD"
14	"{D865A1D1-0AC9-4859-8EDA-5CC769125908}"	149000	2002-05-17 00:00:00	"NN5 6EH"	"D"	"N"	"F"	"8"	NA	"PROVENCE COURT"	"NORTHAMPTON"
15	"{BC424648-B844-4E5A-BF3B-590AE18A8D2C}"	75000	2002-11-11 00:00:00	"B8 2TT"	"T"	"N"	"L"	"85"	NA	"ST AGATHAS ROAD"	"BIRMINGHAM"
16	"{22031AD4-F36C-4268-9DDE-590AE54C42E1}"	66000	2002-09-24 00:00:00	"B13 0PT"	"T"	"N"	"F"	"741"	NA	"YARDLEY WOOD ROAD"	"MOSELEY"
17	"{1A23AD5A-14C1-4026-8FE1-590AEB77823C}"	80000	2002-08-20 00:00:00	"PL14 3TD"	"T"	"N"	"F"	"22"	NA	"EASTERN AVENUE"	"LISKEARD"

After the table was successfully created, we ran a SQL statement to select all columns and rows to display the entire table. We excluded NULL values because there are a small number of rows with NULL values but it will not make any difference to the entire dataset (19 million rows).

Descriptive Analytics in Impala

Our goal is to get some insights from the dataset. First of all, we try to get the Top 10 cities' average price in 2002 and compare it with the Top 10 cities' average price in 2021. In the queries, we use WHERE to filter out NULL and keep either 2002 or 2021 data. Then GROUP BY town_city to get each city's average price. Finally, ORDER BY average price and LIMIT 10 to get the Top 10 cities' average price in 2002 and the Top 10 cities' average price in 2021.



```sql

SELECT town_city as 'Top 10 Cities'
round(avg(price),2)
as 'Average Price in 2002'
FROM testing_7
WHERE price IS NOT NULL
AND YEAR(date_of_transfer) = 2002
group by town_city
Order by avg(price) desc
LIMIT 10;
```

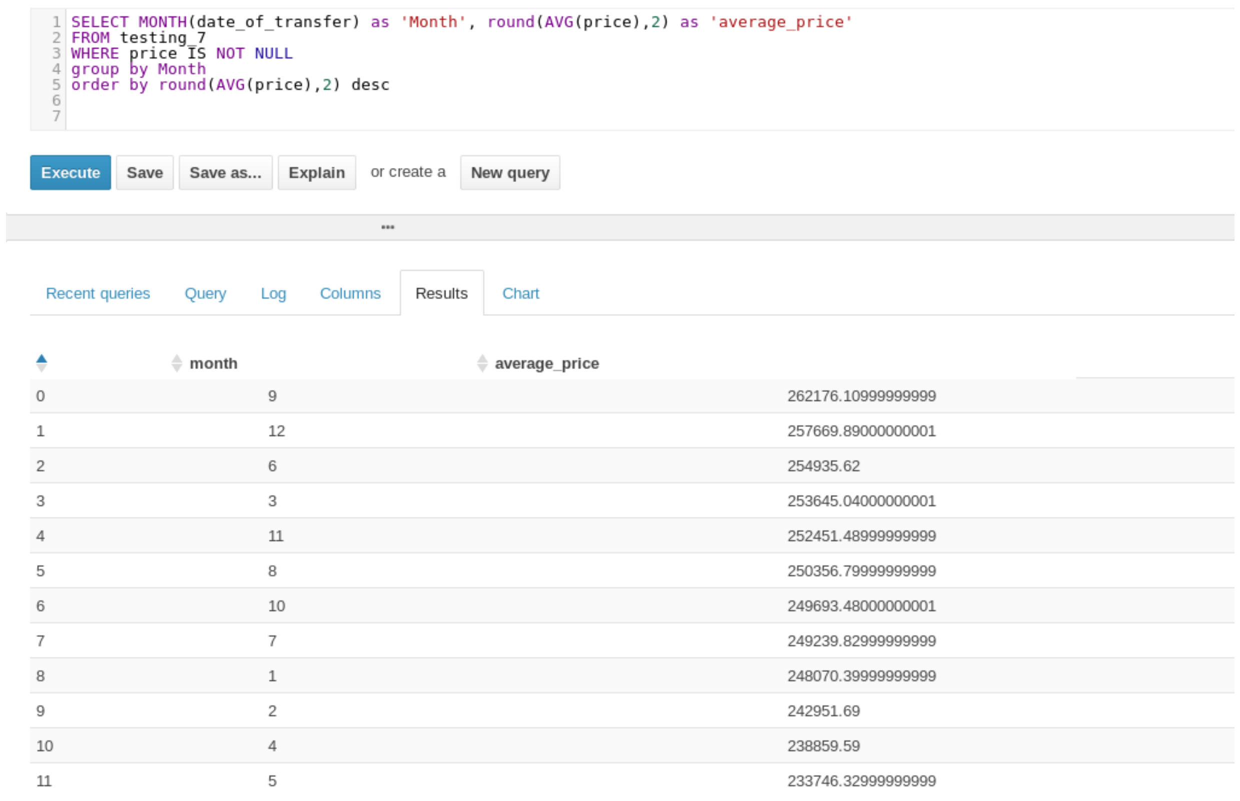
	top 10 cities	average price in 2002
0	"KNOSSINGTON"	1375000
1	"CHUCK HATCH"	1308750
2	"MELDRETH"	1295000
3	"LILLIPUT"	950000
4	"OXSHOTT"	942500
5	"GILSTON"	937500
6	"CUMNOR"	842342.5
7	"TEWIN"	750000
8	"BRAMHOPE"	750000
9	"SYSTON"	745000

```sql
SELECT town_city as 'Top 10 Cities',
round(avg(price),2)
as 'Average Price in 2021'
FROM testing_7
WHERE price IS NOT NULL
AND YEAR(date_of_transfer) = 2021
group by town_city
Order by avg(price) desc
LIMIT 10
```

	top 10 cities	average price in 2021
0	"ALTIRA BUSINESS PARK"	56810000
1	"BROOKLANDS BUSINESS PARK"	25425000
2	"AISECOME WAY"	20520000
3	"OLD BRIDGE STREET"	16950000
4	"NORTHERN BY PASS ROAD"	16675000
5	"TWO MILE ASH"	15250000
6	"GATWICK"	13525000
7	"ANNESLEY"	11066666.67
8	"CENTRAL MILTON KEYNES"	9100000
9	"ERMINE BUSINESS PARK"	8710000

Let's compare the Top 10 cities' average price in 2002 with the Top 10 cities' average price in 2021. We can see the Top 10 cities in 2002. Their distance from London varies. However, Most Top 10 cities in 2021 are a 1-2 hour drive to London. We assumed because of the London Olympics, London's economy grew by about six hundred million during the game. As a result, people rely on London more to get proper jobs, so they live nearby the city.

Furthermore, we try to get the rank of the month's average price. We use WHERE to filter out NULL. Then GROUP BY Month to get the average price for each month. Finally, ORDER BY average price to get the rank of the month's average price.



```
SELECT MONTH(date_of_transfer) as 'Month', round(AVG(price),2) as 'average_price'
FROM testing_7
WHERE price IS NOT NULL
group by Month
order by round(AVG(price),2) desc
```

	month	average_price
0	9	262176.10999999999
1	12	257669.89000000001
2	6	254935.62
3	3	253645.04000000001
4	11	252451.48999999999
5	8	250356.79999999999
6	10	249693.48000000001
7	7	249239.82999999999
8	1	248070.39999999999
9	2	242951.69
10	4	238859.59
11	5	233746.32999999999

We can see that September has the highest average price. We assume this sharp rise is due to the stamp duty holiday which the UK Government announced last year and which ended on 30 September 2021. The stamp duty holiday is when buyers of homes valued at up to £500,000 will no longer pay any stamp duty on the purchase. Stamp duty is a tax that governments place on legal documents, usually involving the transfer of real estate or other assets.

Conclusion

Our descriptive analysis illustrates the trend of UK house prices in the past 20 years. There is a steep increase in house price but the top 10 cities with the highest house price are entirely different in 2002 and in 2021. We conducted a geographic analysis to explore the reason behind this, and we found out these top 10 cities in 2021 are mostly 1 to 2 hours drive from London. This is probably because there might be commuters who live in these regions but work in London, earning London wages (which are higher).
We saw that detached houses are expensive whereas terraced houses are cheap. This is probably the reason why the demand for terraced houses is higher than that of detached houses.
We have also seen that when average income is compared against average house price, the upward trend of the latter is much steeper than the former. This is because houses have a low interest rate. So people buy them for investment. Moreover, there are foreign investors too.

Moreover, we have tried 2 regression models - Random Forest and Gradient Boosted Tree and saw that the latter performed better. According to the latter model, the most important features that determine the price of a house are Date of Transfer, Property Type and County.

Business Impact of the Analysis

The Gradient Boosted Tree might be useful to predict the price of a house, which might be useful to property owners.
Since the HM Land Registrar and the government would be interested in average house price forecasts in the coming years, we trained a time series model and used it to forecast average house price in 2020 and 2021. The RMSE is 13712.74.
Moreover, our explanatory analysis using Spark and Impala might be a useful source of information for the government, HM Land Registrar and the housing market.

References

1. https://www.homes.com/blog/2017/10/millennials-buying-detached-homes-pros-cons/
2. https://www.andersonassociates.co.uk/news/semi-detached-or-terraced-house-which-should-you-choose/
3. https://www.schroders.com/en/insights/economics/what-174-years-of-data-tell-us-about-house-price-affordability-in-the-uk/
4. https://www.globalpropertyguide.com/Europe/United-Kingdom/Price-History
5. https://www.kaggle.com/datasets/dmaso01dsta/house-price-data-england-wales-1995-to-2019?select=README.md