Theo Bacon Gardner, tjb17@ic.ac.uk, MSc NT CID: 01439118

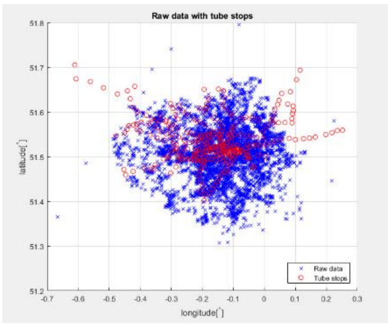
MLNC – Assessed Coursework 1: Supervised learning of 1-bedroom/studio rents from London location

Objective

Supervised learning is a form of machine learning that uses known data, called training data, to build a model that can predict the response of a new dataset. The objective of this coursework is to use supervised learning techniques to build a model that can accurately predict rental prices in London based on geographic location, using an existing dataset of London property locations and their corresponding prices as training data. A portion of the data must be used as test data, in order to compare predicted values with actual values, validating the accuracy of the model.

Cleaning the data

Before starting the process of deciding how to model and predict the given London data, the raw data had to be cleaned by removing any properties that were not located within the desired area of Greater London from the dataset. Initially this was going to be achieved by utilising the coordinates given for the London tube stops, using the furthest tube stops from the centre of London as limits in each cardinal direction, however as can be seen in Fig.(a) this leaves a considerable portion of data in the south of London out of the dataset. Consequently the M25 was instead used as the boundary for Greater London. The furthest point of the M25 in each cardinal direction was used as a limit. This seemed like a reasonable method as opposed to using the entire boundary of the M25 to clean the data, which would have required a considerably larger and more complex set of code only to include/exclude a negligible number of data points.



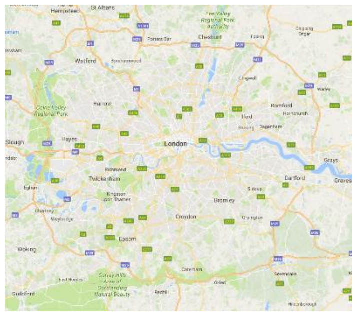


Fig.(a): (left) Raw London location data plotted with overlaid Tube stop locations. Fig.(b): (right) M25 surrounding greater London.

1 Visualising in a 3D plot

Figure 1(a) shows a plot of the rental price pcm against location for the cleaned data of Greater London. There seems to be a general trend that would be expected for London rental prices: The vast majority of rental properties are shown to cost up to roughly £5000pcm, but as we move further and further central we see properties of between £5000-10,000pcm occurring more frequently and the occasional property reaching prices of £10,000-25,000pcm. Consequently the model for estimating London rental prices must follow this general trend and provides a good indicator to whether the model is performing in a satisfactory manner.

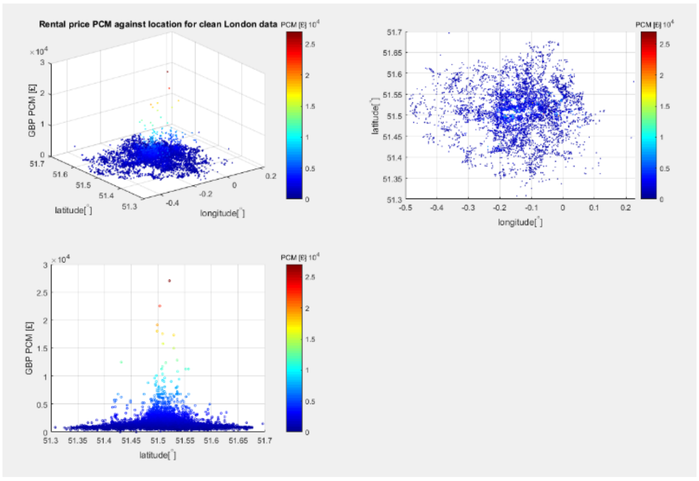


Fig.1(a): Rental price plotted against location for cleaned London data.

2 Design & Training

2.1 Approach

In order to create a model that can accurately predict the rental price in a given area of London, first a supervised learning algorithm must be established. The design of this algorithm determines the effectiveness of the function inferred from the training data, and thus the accuracy of the model to predict the test data as well as mapping the prices of novel locations.

After experimenting with different approaches it was decided that an effective regression technique would be to use a least squares solution with Gaussian basis functions. In this case using local basis functions is more appropriate, hence the use of a Gaussian model. The parameters required for the model are therefore the location and spread of each Gaussian given by the mean and covariance, which together with the training data will define the basis functions (shown in Eq. 2(a)), as well as the weights for each basis function.

$$\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\}$$

Eq.2(a):Gaussian basis function

2.2 Clustering the data

In order to define the number of Gaussian basis functions used and parameters for each basis function, first the data needs to be clustered. To do this a simple grid method was used. The algorithm used allows for the number of grid spaces to be altered, thereby varying the number of Gaussian basis functions. The mean for each Gaussian was determined by the arithmetic mean coordinates within each corresponding grid space. Finally the co-variance was determined as the co-variance for all the data within each grid space. Consequently for *i* Gaussians the regression will provide *i* means and *i* covariance matrices. Fig.2(a) shows a plot of the clustered data, with each grid space indicated by different coloured data points, when the algorithm was set to twelve Gaussian basis functions (4x3 grid).

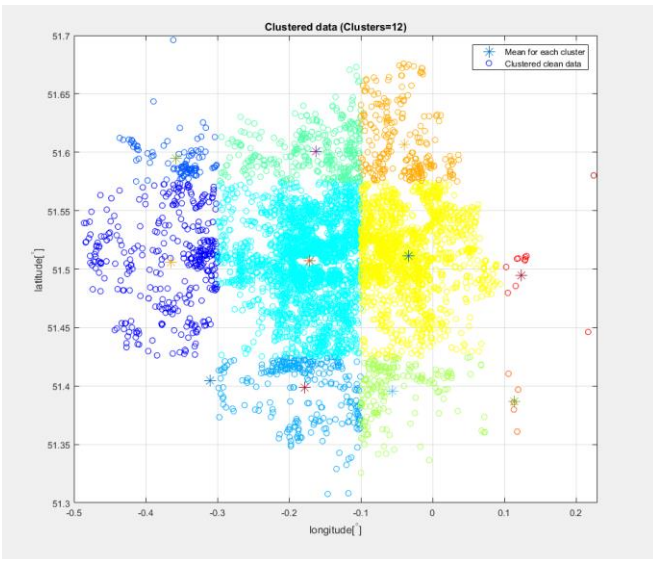


Fig.2(a): Clean London data clustered into a 4x3 grid. Each grid space is indicated by different coloured data points. Mean values for each grid are also shown.

2.3 Train regressor

Given the definitive design of the model, the train regressor was constructed with two inputs being the locations of the training data in latitude and longitude and their corresponding prices. The output from the regerssor was a set of parameters, including the values for the mean, covariances, weights and number of basis functions.

3 Testing & validation

3.1 Test regressor

Following on from the train regressor, the parameters acquired from training the system on the location data are put through a test regressor along with the location data for testing. The output given by this function corresponds to the rental prices of the properties in the test data predicted by the model.

3.2 Root-mean squared error

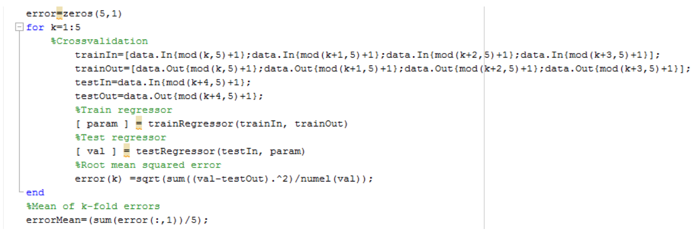
To test the validity of the model the root-mean-squared error (RMSE) was calculated. This essentially compares the prices of the test data predicted by the model with the actual prices. Eq.3(a) shows the equation used to calculate the error.

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}\left(P_i - O_i\right)^2}{n}}$$

Eq.3(a): Equation used to calculate the root-mean-squared error

3.3 Splitting the data/cross-validation

In splitting the data 80% was taken for training and 20% for testing. A five-fold cross-validation algorithm was also used to ensure all the data was both trained and tested on. This algorithm is shown in Fig.3(b). Consequently five errors were computed each time the regressor was run, from which a mean error was calculated.



```
error=zeros(5,1)
for k=1:5
    %Crossvalidation
        trainIn=[data.In{mod(k,5)+1};data.In{mod(k+1,5)+1};data.In{mod(k+2,5)+1};data.In{mod(k+3,5)+1}];
        trainOut=[data.Out{mod(k,5)+1};data.Out{mod(k+1,5)+1};data.Out{mod(k+2,5)+1};data.Out{mod(k+3,5)+1}];
        testIn=data.In{mod(k+4,5)+1};
        testOut=data.Out{mod(k+4,5)+1};
        %Train regressor
        [ param ] = trainRegressor(trainIn, trainOut)
        %Test regressor
        [ val ] = testRegressor(testIn, param)
        %Root mean squared error
        error(k) =sqrt(sum((val-testOut).^2)/numel(val));
end
%Mean of k-fold errors
errorMean=(sum(error(:,1))/5);
```

Fig.3(b): Algorithm used to conduct five-fold cross validation.

Furthermore all the London data was randomised each time the regressor was run, meaning the contents of each fold changed with every run. This allowed for a mean of the mean error to be calculated, adding an extra dimension of cross-validation.

3.4 Number of Gaussians

Using the clustering algorithm mentioned previously the number and location of the Gaussian basis functions can be varied according to the grid size. To investigate the effect this had on the RMS error a plot was made, shown in Fig.3(c), with the number of Gaussians varying between 4-36. It was felt that above this range the model would be in danger of overfitting. The model was run three times for each set number of gaussians, so that a mean of the mean error could be calculated, ensuring thorough cross-validation. As shown the error is seen to fluctuate with the increase in number of Gaussians. Notable lowest errors occur at 12 and 30 Gaussians.

After cross-referencing with information provided by the heatmaps, it was decided that the model using 30 Gaussian base functions was the optimum choice. Despite the fact that overfitting could be more pronounced with the 30 Gaussian model, some of the other models including the 12 Gaussian model contained artefacts that resulted in some areas of London having negative rental prices predicted. Taking into consideration that the 30 Gaussian model did not contain such artefacts as well as having the lowest mean RMSE of all the models **(RMSE = 867)** led to it being the definitive version of the model.

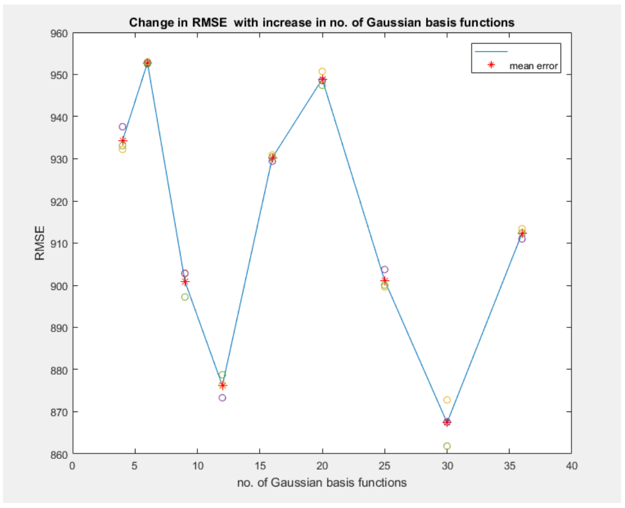


Fig.3(c):Plot showing the change in RMSE with increasing number of Gaussian basis functions.

4 Visualising the heatmap

The heatmap produced for the 30 Gaussian model can be seen below in Fig.4(a). Note that the limits for both Latitude and Longitude were changed in order to better fit the London data.

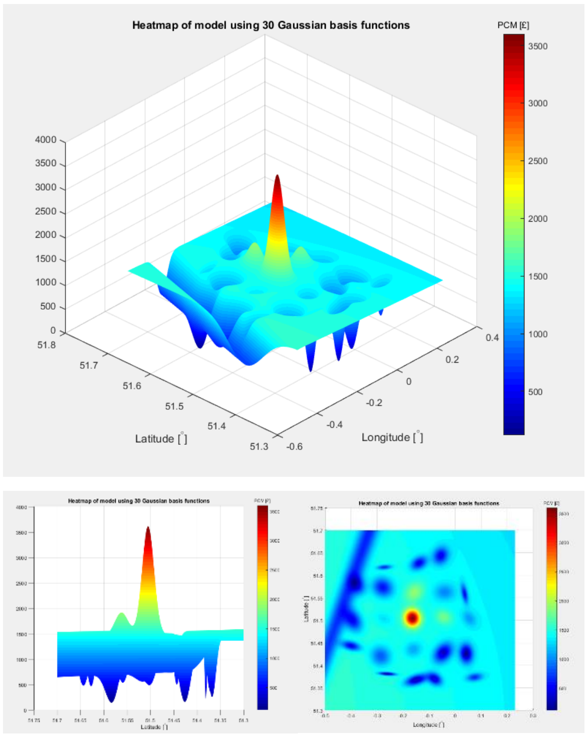


Fig.4(a): Heatmap of model for predicting London rental prices using 30 Gaussian basis functions.

The heatmap seems to follow to some extent the general trend shown in the original 3D plot from section one. There is a plane that runs through the entire heatmap at around £1500, from which we see depressions and ascensions in the price in the form of Gaussian distributions. Further out from the centre of London the areas in which there are a drop down in price are seen more frequently. When we move further towards the centre the drops in price are seen less frequently and increases in the price appear to spike close to the centre. The tallest and most central Gaussian reaches a price limit of around £3500. This is almost a factor of 10 less than the highest price in the original dataset, however due to the scarcity of data in this high price range we do not expect to see much influence on the model. The heatmap also highlights an artefact captured in the model in the form of a depressive valley that runs diagonally from around [-0.5, 51.4] to [-0.3, 51.7].

5 Faux home tube stop

Using the file Matlab file given, *personalisedTube.m*, the name and location of my faux tube stop 'Golders Green' was generated. The coordinates for my faux tube were then run through the test regressor providing the predicted rental price which run over three times averaged at £1152pcm.

To determine whether this was a reasonable prediction an average was taken of the rental prices for properties surrounding the area of my faux tube. The area taken into account was arbitrary, and only took into account roughly 20-30 properties, but still gives a good indication to how reasonable the prediction is. This mean came to £1335, resulting in a difference of £183 between the models prediction and an average of the real data surrounding the tube. Fig.5(a) shows graphically how my faux tube compares with properties in the surrounding area.

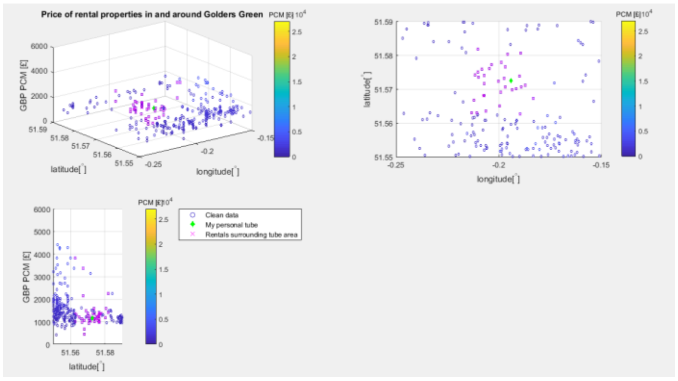


Fig.5(a): Scatter plot of London properties (magenta) surrounding my faux tube (green) and their corresponding prices pcm.

6 Conclusion

Overall the heatmap indicates that the model could be over-fitting the data, and that one would prefer a smoother distribution as opposed to the geometrically complex mapping produced. On the other hand as mentioned before this model provided the lowest RMSE and no negative rental prices. In addition to this the model does seem to follow the same general trend shown in the original data. Finally the predicted price for my faux tube 'Golders Green' was not too dissimilar to the average prices of properties surrounding the area, indicating that in this specific case the model behaved quite effectively.

Appendix

```
%% Load london
load ('london.mat')
%% Clean data
Prices.combined = [Prices.location Prices.rent];
Clean = Prices.combined(Prices.combined(:,1)>=51.3 &
Prices.combined(:,1)<51.7 & Prices.combined(:,2)>=-0.488 &
Prices.combined(:,2)<0.225, :);
%Limits defined from M25

%% Randomise and separate
Clean=Clean(randperm(end),:);%Allows for model to be cross-validated for
each run through
xClean=Clean(:,1);
yClean=Clean(:,2);
pClean=Clean(:,3);
%% 3D plot with clean data
figure
subplot(2,2,1)
scatter3(yClean,xClean,pClean,2 ,pClean);
ylim([51.3 51.7])
xlim([-0.5 0.23])
ylabel('latitude[^\circ]')
xlabel('longitude[^\circ]')
zlabel('GBP PCM [£]')
hcb=colorbar
colormap(jet)
title('Rental price PCM against location for clean London data')
title(hcb,'PCM [£]')

subplot(2,2,2)
scatter3(yClean,xClean,pClean,1 ,pClean);
ylim([51.3 51.7])
xlim([-0.5 0.23])
ylabel('latitude[^\circ]')
xlabel('longitude[^\circ]')
zlabel('GBP PCM [£]')
hcb=colorbar
colormap(jet)
title(hcb,'PCM [£]')
view(2);

subplot(2,2,3)
scatter3(yClean,xClean,pClean,6 ,pClean);
ylim([51.3 51.7])
xlim([-0.5 0.23])
ylabel('latitude[^\circ]')
xlabel('longitude[^\circ]')
zlabel('GBP PCM [£]')
hcb=colorbar
colormap(jet)
title(hcb,'PCM [£]')
view([90 0]);

set(gcf, 'Position', [100, 100, 1500, 700])
%% Cutting data for crossvalidation
```

```
inClean=[xClean yClean];
nrows=round(0.2*length(inClean));
for k=1:5 %k corresponds to number of folds
    if k==5
        data.In{k}=inClean((k-1)*nrows+1:end,:);
        data.Out{k}=pClean((k-1)*nrows+1:end,:);
    else
    data.In{k}=inClean((k-1)*nrows+1:k*nrows,:);
    data.Out{k}=pClean((k-1)*nrows+1:k*nrows,:);
    end
end

%% Training and testing Q2&3
error=zeros(5,1)
for k=1:5
    %Crossvalidation

trainIn=[data.In{mod(k,5)+1};data.In{mod(k+1,5)+1};data.In{mod(k+2,5)+1};
data.In{mod(k+3,5)+1}];

trainOut=[data.Out{mod(k,5)+1};data.Out{mod(k+1,5)+1};data.Out{mod(k+2,5)
+1};data.Out{mod(k+3,5)+1}];
        testIn=data.In{mod(k+4,5)+1};
        testOut=data.Out{mod(k+4,5)+1};
        %Train regressor
        [ params ] = trainRegressor(trainIn, trainOut)
        %Test regressor
        [ val ] = testRegressor(testIn, params)
        %Root mean squared error
        error(k) =sqrt(sum((val-testOut).^2)/numel(val));
end
%Mean of k-fold errors
errorMean=(sum(error(:,1))/5);

%Clusterplot of London separated into grid and corresponding mu's
% figure
% colors=jet(numGaussians);
% for i=1:numGaussians
% plot(temp{1,i}(:,1) , temp{1,i}(:,2),'o','color',colors(i,:))
% plot(param.mu{1,i}(:,1), param.mu{1,i}(:,2),'*','markers',12)
% hold on;
% end
% xlim([-0.5 0.23])
% ylim([51.3 51.7])
% xlabel('longitude[^\circ]')
% ylabel('latitude[^\circ]')
% legend('Mean for each cluster','Clustered clean data')
% title('Clustered data (Clusters=12)')
% grid on


%% Sanity check
sanityCheck(@trainRegressor,@testRegressor)

%% Heatmap
figure
heatmapRent(@testRegressor, params)%For an effective heatmap, limits
should be changed to -0.5<Lat<0.23 and 51.3<Long<51.7

%% Q5 Personalised tube
```

```
%Faux home tube top name and coordinates
[TubeStop, TubeCoor] = personalisedTube(1439118)
%Faux tube stop rental price predicted by model
TubeCoor1=[TubeCoor;TubeCoor];%input needs to be at least a 1x2 matrix
[ val ] = testRegressor(TubeCoor1, params)
%Compare my tube with sample data
MyTube=[TubeCoor(:,1),TubeCoor(:,2),val(1,:)];
MyTubePrice=val(1,:);
TubeArea = Prices.combined(Prices.combined(:,1)>=51.5625 &
Prices.combined(:,1)<51.5825 & Prices.combined(:,2)>=-0.214 &
Prices.combined(:,2)<-0.174, :);

MeanTubePrice = mean(TubeArea(:,3));%average price of property
surrounding local area of my tube
dTube=MyTubePrice-MeanTubePrice;%difference between the price of living
at my tube calculated by the model compared to average prices of
surrounding area

%Plot comparing faux home tube stop wth surrounding area
figure
subplot(2,2,1)
scatter3(yClean,xClean,pClean,9 ,pClean);
hold on
scatter3(MyTube(:,2),MyTube(:,1),MyTube(:,3),24,'gd','filled')
hold on
scatter3(TubeArea(:,2),TubeArea(:,1),TubeArea(:,3),25 ,'x','m')
xlim([-0.25 -0.15])
ylim([51.55 51.59])
zlim([0 6000])
xlabel('longitude[^\circ]')
ylabel('latitude[^\circ]')
zlabel('GBP PCM [£]')
hcb=colorbar
title(hcb,'PCM [£]')
view(3)
title('Price of rental properties in and around Golders Green')
hold off

subplot(2,2,2)
scatter3(yClean,xClean,pClean,9 ,pClean);
hold on
scatter3(MyTube(:,2),MyTube(:,1),MyTube(:,3),24,'gd','filled')
hold on
scatter3(TubeArea(:,2),TubeArea(:,1),TubeArea(:,3),25 ,'x','m')
xlim([-0.25 -0.15])
ylim([51.55 51.59])
zlim([0 6000])
xlabel('longitude[^\circ]')
ylabel('latitude[^\circ]')
zlabel('GBP PCM [£]')
hcb=colorbar
title(hcb,'PCM [£]')
view(2);
hold off

subplot(2,2,3)
Cleanscat=scatter3(yClean,xClean,pClean,9 ,pClean);
hold on
MyTubescat=scatter3(MyTube(:,2),MyTube(:,1),MyTube(:,3),24,'gd','filled')
hold on
```

```
TubeAreascat=scatter3(TubeArea(:,2),TubeArea(:,1),TubeArea(:,3),25
,'x','m')
xlim([-0.25 -0.15])
ylim([51.55 51.59])
zlim([0 6000])
xlabel('longitude[^\circ]')
ylabel('latitude[^\circ]')
zlabel('GBP PCM [£]')
hcb=colorbar
title(hcb,'PCM [£]')
view([90 0]);

h = [Cleanscat(1);MyTubescat;TubeAreascat(1)];
legend(h,'Clean data','My personal tube','Rentals surrounding tube
area');
set(gcf, 'Position', [100, 100, 1500, 700])
hold off
```