Spatial Analysis of central London Rental prices

House prices across central London are highly heterogenous, with prices varying based on the numbers of bedrooms, garden access, size of rooms, and most importantly their geographical location (as well as many other factors). In this document we will investigate how we can use a data analysis approach, called spatial analysis, to understand house prices, and the distribution of houses within central London. Spatial analysis is the integration of geographical information into knowledge discovery through statistics and machine learning strategies

Frequency-based spatial analysis

The frequency of available properties-to-rent in central London indicates not only the availability of housing but also affordability. Housing in central London is roughly spatially segregated, with regions of high population density separated by industrial regions and office spaces. Locations with a high volume of available properties may be indicative of locations wherein houses are unaffordable yet available. By contrast, affordable housing tends to quickly circulate through the market. To investigate the distribution of available properties-to-rent in central London, within our dataset we will implement a kernel density estimation, which gives a visual indication of spatial organisation within a dataset.

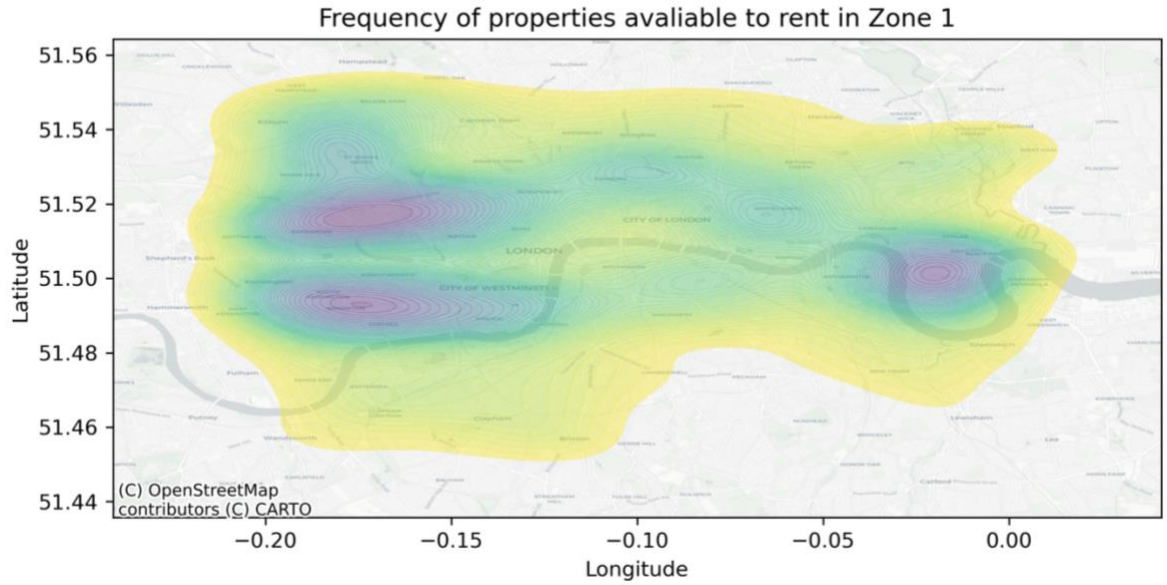

Fig 1. Kernel density estimation of the frequency of properties-to-rent in central London

This same information, represented by the kernel density estimation, can be represented in a hex plot, binning the number of houses available to rent based on their geographical location. This approach creates a far more spatially specific and granular representation of the frequency of properties available to rent in London, as shown in Fig 2 on the following page.

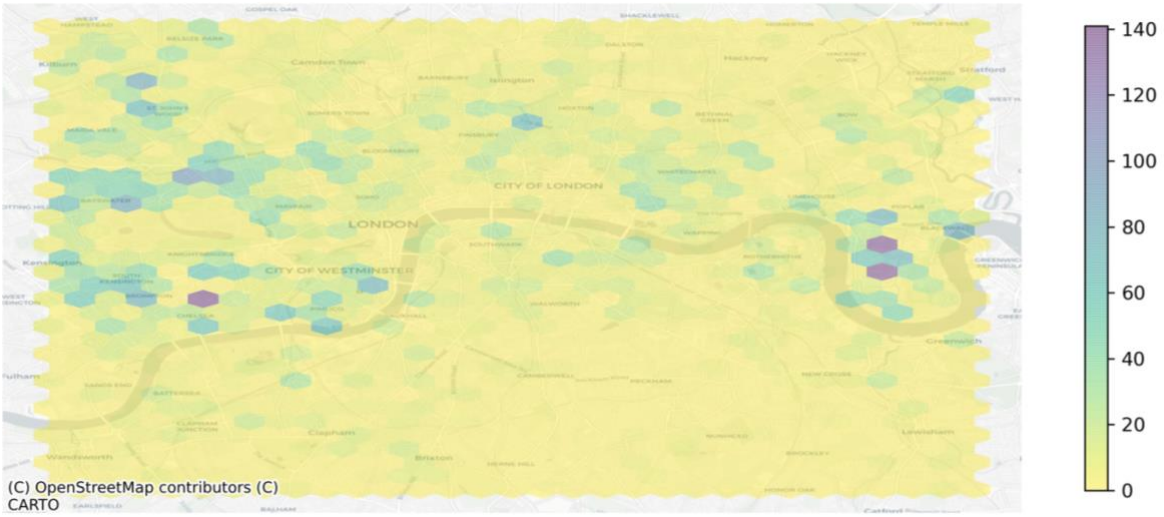

Fig 2. Shows the kernel density estimation of houses to rent in London

From Fig 1-2. we see that the places in London with the most properties to rent are in Canary Wharf, Chelsea, and Bayswater/Queensway. This data however gives us no information on the price of houses in London.

Price-based spatial analysis

To generate a heatmap of house prices across London, we use a 5-Nearest Neighbour interpolation to generate a 1000×1000 surface of house prices across London, based on the information available within the dataset. This heatmap is shown below.

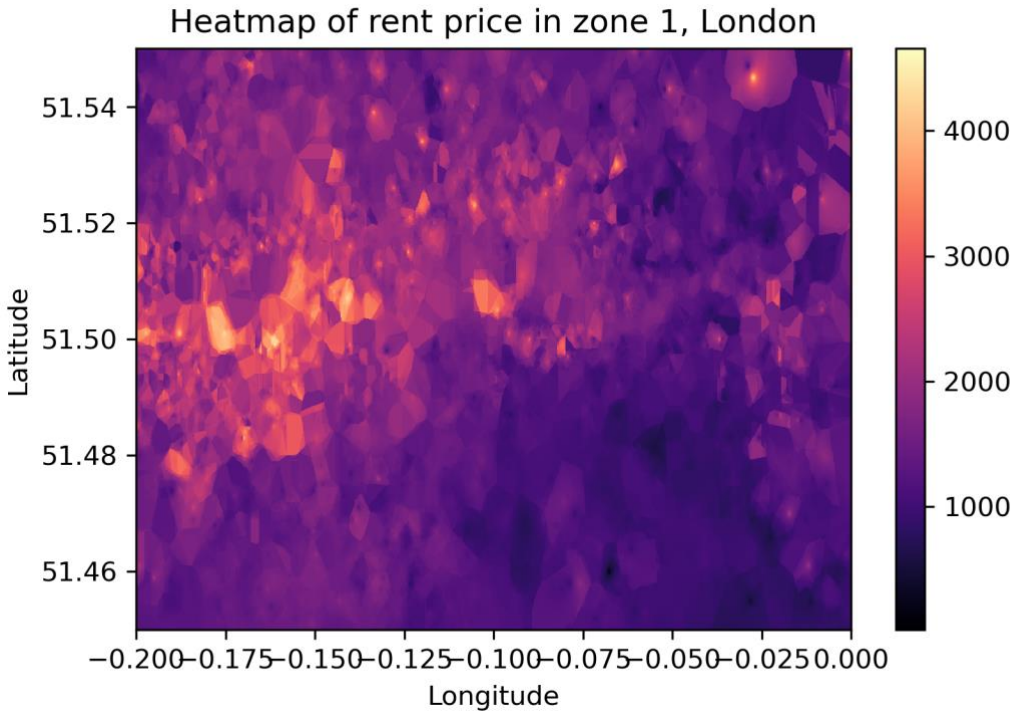

Fig 3. Heatmap of rent price (pcm) <£5000 in zone 1, London

The heatmap plot gives us an understanding of house prices in London, with spikes of high prices scattered all over London. The frequency of properties that cost >£5000 make up 1.7% of those available on the market ($\frac{146}{8541}$). Clearly, whether the property is north or south of the river Thames has a large impact on its property value. To separate the dataset based on the river's location, we use the highly simplified model explained in appendix 1. Using this model, we identify that there are substantially more properties with rent cost >£5000 pcm north of the Thames compared to south of it (Frequency count: 145 v 1). Furthermore, that there is a significant decrease in the mean rent prices in properties south of the river Thames (£1305 pcm), relative to those north of it (£2030 pcm) as seen in Fig 3 ($p < 0.01$, t-test). North of the river, there is a significant reduction in the mean house price in Hackney (μ = £1554 pcm) relative to the mean price of other properties north of the Thames (μ = £2041 pcm) ($p < 0.01$, t-test). However, even the rent price within the London borough of Hackney is highly heterogenous, with a wide variety of house prices across the borough as shown in Fig 4-5.

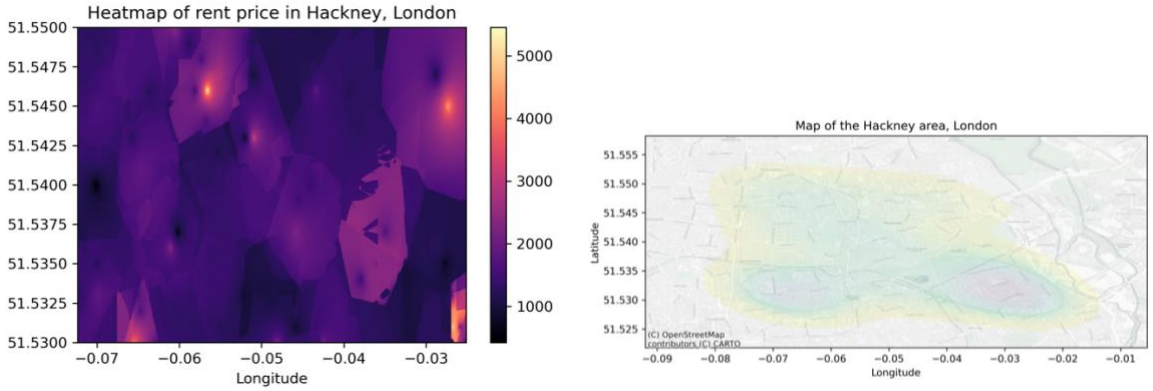

Fig 4a. Heatmap of rental prices across Hackney 4b. distribution of properties to rent across hackney based on frequency

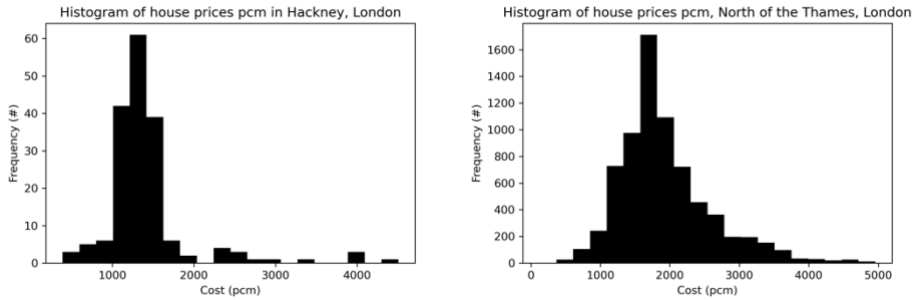

Fig 5a. Histogram of rental prices in Hackney 5b. Histogram of rental prices north of the river Thames and not in Hackney

Fig 5a-b. alludes to the wealth disparity present in Hackney, with far fewer houses in the £2000-3000 pcm range in Hackney, relative to the rest of properties north of the Thames River. This phenomenon however is not isolated to Hackney, with a general trend of an increased frequency of properties with rental value >£2000pcm in west London relative to east London (2456 v 459). This is further shown on the following page in Fig 6a-d.

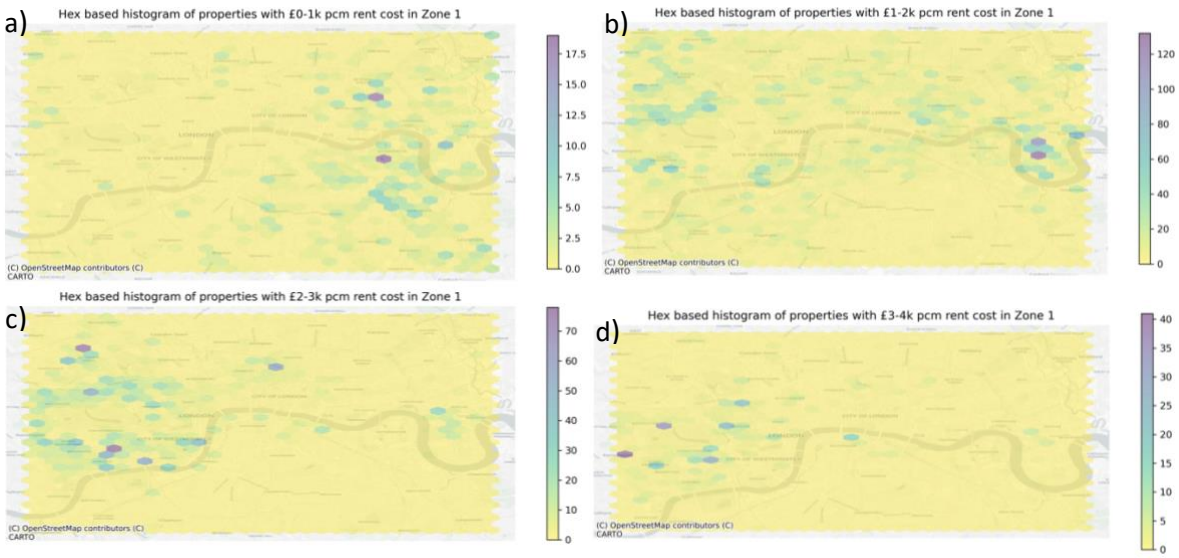

Fig 6a-d. Hex plots of the frequency of properties available at different price ranges.

Conclusion

From house prices and the availability of houses on the market, we have begun to understand the wealth disparity and wealth distribution in London, uncovering areas of high population and low population through the proxy of rental property available in those locations. The results of this document are no doubt unsurprising to someone who has spent a considerable amount of time living in London. However, they indicate the power of this method, as someone with very little prior knowledge of London's property market, can use this dataset to intuit both the distribution of wealth and availability of housing across London. This approach is by no means complete but is instead an exploratory analysis, investigating how interwoven spatial locations and rental prices are within the central London rental market. Whilst location is by no means the only factor determining housing price, this document has shown that it is a large contributing factor in the overall price of a rental property in London.

Appendix 1. North of the River

The river Thames is difficult to model, in this study we use this gross simplification determined by:

$$y = m_i x_i + c_i$$

$$x \leq -0.03 \longrightarrow m_1 = \frac{1}{9}, c_1 = 51.498$$

$$-0.03 < x \leq -0.02 \longrightarrow m_2 = \frac{-3}{2}, c_2 = 51.457$$

$$-0.02 < x \longrightarrow m_3 = \frac{1}{9}, c_3 = 51.485$$

Where x is the Longitude and y is the corresponding Latitude. Using this equation, we can estimate whether a property is north or south of the Thames. Spatially this equation gives rise to the following plot shown in Fig 4.

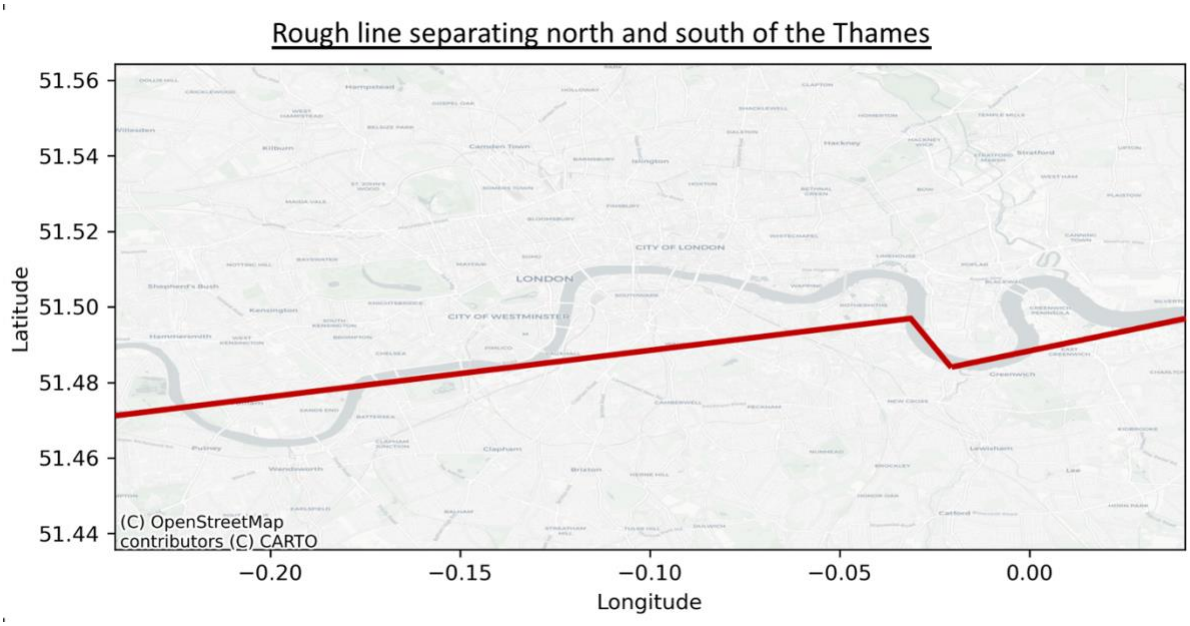

Fig 4. Plot of the curve estimating the Thames

Whilst this is a gross misrepresentation of the actual curvature of the river Thames, it useful whilst remaining low in cost to implement. The exact lines were chosen because the error region (Fulham, Southwark etc,) had relatively low numbers of data samples as shown on Fig 1. and Fig 2.

Appendix 2. Hackney

The outline of the borough of Hackney is smooth and contiguous, making it difficult to model accurately. To save time in this study, it was isolated to a single box as shown on Fig 5.

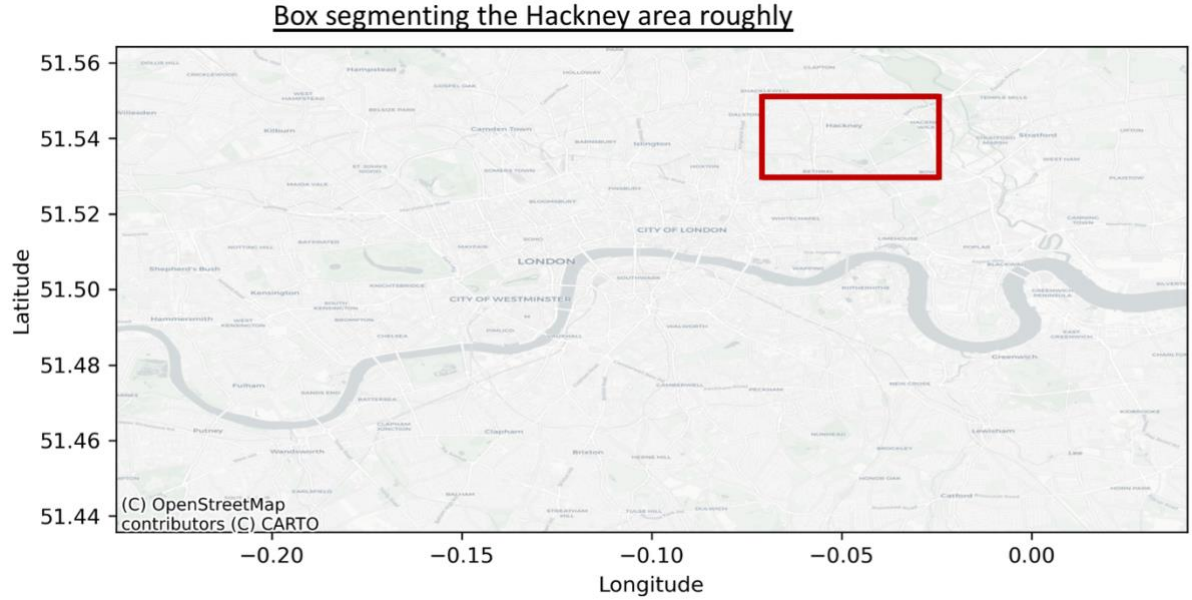

Fig 5. Outline of Hackney used in this study

Appendix 3: Python Code

```
import numpy as np
import pandas as pd
import seaborn as sns
import contextily
import matplotlib.pyplot as plt
import scipy as sci

dat =
pd.read_csv("/Users/sam/Documents/Scripts_Backup/Scripts/PythonScripts/london.csv")


# Clean the data for roughly zone 1, London
dat = dat.drop(dat[dat.Price==0].index)
dat = dat.drop(dat[dat.Latitude<51.45].index)
dat = dat.drop(dat[dat.Latitude>51.55].index)
dat = dat.drop(dat[dat.Longitude<-0.2].index)
dat = dat.drop(dat[dat.Longitude>0].index)


def Joint_histogram(dat):
   # Generate scatterplot + histograms of property long/lat
   joint_axes = sns.jointplot(x=dat['Longitude'],y=dat['Latitude'],kind='scatter',s=2)

contextily.add_basemap(joint_axes.ax_joint,crs="EPSG:4326",source=contextily.providers.C
artoDB.PositronNoLabels);

def Hex_map(dat):
   #Generate 2D Hitstogram
   f, axes = plt.subplots(1, figsize=(12, 9))
   # Generate and add hexbin with 30 hexagons each
   hist = axes.hexbin(
      dat['Longitude'],
      dat['Latitude'],
      gridsize=30,
      linewidths=0,
      alpha=0.4,
      cmap='viridis_r'
      )
   # Add basemap

contextily.add_basemap(axes,crs="EPSG:4326",source=contextily.providers.CartoDB.Positro
n)
   plt.colorbar(hist,shrink=0.5)
   plt.title('Hex based histogram of properties with £3-4k pcm rent cost in Zone 1')
   axes.set_axis_off()
   plt.savefig('/Users/sam/Documents/Hexmap_london_rent_3_4K.png', dpi=300)
```

```
def KDE(dat):
   #Generate kernal density estimation
   f, axes = plt.subplots(1, figsize=(9, 9))
   # Generate and add KDE with a shading of 30 gradients
   sns.kdeplot(
      dat['Longitude'],
      dat['Latitude'],
      n_levels=50,
      shade=True,
      alpha=0.1,
      cmap='viridis_r'
      )
   # Add basemap

contextily.add_basemap(axes,crs="EPSG:4326",source=contextily.providers.CartoDB.Positro
n)
   plt.title('Map of the Hackney area, London')
   plt.savefig('/Users/sam/Documents/KDE_London_Rent.png',dpi=300)

def plot_3d(dat,inte):
   fig = plt.figure()
   X_,Y_,Z_= interp_3d(dat)
   if inte==1:
      ax = fig.add_subplot(111, projection='3d')
      ax.plot_surface(X_, Y_, Z_)
      ax.set_xlabel('Longitude')
      ax.set_ylabel('Latitude')
      ax.set_zlabel('Price')
      plt.title('Surface map of rent price in zone 1, London')
      plt.show()
      plt.savefig('/Users/sam/Documents/surface_London_Rent.png',dpi=300)
   if inte==2:
      fig, ax = plt.subplots()
      heatmap = ax.pcolormesh(X_, Y_, Z_, cmap='magma')
      ax.axis([X_.min(), X_.max(), Y_.min(), Y_.max()])
      fig.colorbar(heatmap, ax=ax)
      ax.set_xlabel('Longitude')
      ax.set_ylabel('Latitude')
      plt.title('Heatmap of rent price in Hackney, London')
      plt.savefig('/Users/sam/Documents/Heatmap_Hackney_Rent.png',dpi=300)

def interp_3d(dat):
   #5-NN to interpolate surface  of rent price
   from sklearn.neighbors import KNeighborsRegressor
   neigh = KNeighborsRegressor(n_neighbors=5,weights='distance',p=2)
   neigh.fit(dat[['Longitude','Latitude']].values, dat['Price'].values)
```

```
    X_synth = np.linspace(min(dat.Longitude),max(dat.Longitude),1000)
    Y_synth = np.linspace(min(dat.Latitude),max(dat.Latitude),1000)
    [X_,Y_] = np.meshgrid(X_synth,Y_synth);
    X_T = np.append(X_.reshape(-1,1),Y_.reshape(-1,1),axis=1)
    Z_ = neigh.predict(X_T)
    Z_ = Z_.reshape(X_.shape)

    return X_,Y_,Z_

def Price_hist(dat):
  #Removing houses that cost greater than 5000  for cleaner results
  dat = dat.drop(dat[dat.Price>5000].index)
  plt.hist(dat['Price'].values,20,color='k')
  plt.xlabel('Cost (pcm)')
  plt.ylabel('Frequency (#)')
  plt.title('Histogram of house prices pcm, North of the Thames, London')
  plt.savefig('/Users/sam/Documents/Histogram_NotT_Hackney.png',dpi=300)

def KM(dat,n_components,inte):
    from sklearn.cluster import KMeans
    K = KMeans(n_clusters = n_components).fit(dat[['Longitude','Latitude','Price']].values);
    X_,Y_,Z_= interp_3d(dat)
    X_T = np.append(X_.reshape(-1,1),Y_.reshape(-1,1),axis=1)
    X_T = np.append(X_T,Z_.reshape(-1,1),axis=1)

    Labels_ = K.predict(X_T)
    Labels_= Labels_.reshape(X_.shape)

    if inte ==0:
       return X_,Y_,Z_;
    if inte ==1:
       fig, ax = plt.subplots()
       heatmap = ax.pcolormesh(X_, Y_, Labels_, cmap='cividis')
       ax.axis([X_.min(), X_.max(), Y_.min(), Y_.max()])
       ax.set_xlabel('Longitude')
       ax.set_ylabel('Latitude')
       plt.title('Segmentation of rent price in zone 1, London by K-means')
       #plt.savefig('/Users/sam/Documents/K-means_London_Rent.png',dpi=300)

       return X_,Y_,Z_;

def Natural_Breaks(dat,Quantiles):

    dat['Breaks'] = pd.qcut(dat['Price'], q=Quantiles, labels=range(0,Quantiles))

    from sklearn.neighbors import KNeighborsRegressor
    neigh = KNeighborsRegressor(n_neighbors=5,weights='distance',p=2)
```

```
    neigh.fit(dat[['Longitude','Latitude']].values, dat['Breaks'].values)

    Break_means = np.zeros([Quantiles])
    for i in range(0,Quantiles):
        Break_means[i] = np.mean(dat[dat.Breaks==i].Price.values)

    X_synth = np.linspace(min(dat.Longitude),max(dat.Longitude),1000)
    Y_synth = np.linspace(min(dat.Latitude),max(dat.Latitude),1000)
    [X_,Y_] = np.meshgrid(X_synth,Y_synth);
    X_T = np.append(X_.reshape(-1,1),Y_.reshape(-1,1),axis=1)
    Z_ = neigh.predict(X_T)
    Z_ = Z_.reshape(X_.shape)

    fig, ax = plt.subplots()
    heatmap = ax.pcolormesh(X_, Y_, Z_, cmap='cividis')
    ax.axis([X_.min(), X_.max(), Y_.min(), Y_.max()])
    ax.set_xlabel('Longitude')
    ax.set_ylabel('Latitude')
    plt.title('Segmentation of rent price in Hackney, London based on natural breaks')

def Hackney(dat):
    #Bethanl Green + Hackney
    #  51.51<lat<51.53
    #  -0.075<long<-0.025
    dat = dat.drop(dat[dat.Latitude<51.53].index)
    dat = dat.drop(dat[dat.Latitude>51.55].index)
    dat = dat.drop(dat[dat.Longitude<-0.075].index)
    dat = dat.drop(dat[dat.Longitude>-0.025].index)

    return dat;

def Not_Hackney(dat):

    dat = dat.drop(dat[
                (dat.Longitude>-0.075) & (dat.Longitude<-0.025) &
                (dat.Latitude<51.55) & (dat.Latitude>51.53)].index);
    return dat;

def North_of_Thames(dat):
    #Define to linear equations to seperate the north of the Thames.
    dat = dat.drop(dat[(dat.Longitude<=-0.03) &
(dat.Latitude<(((1/9)*dat.Longitude)+51.498))].index)
    dat = dat.drop(dat[(dat.Longitude<=-0.02) & (dat.Longitude>-0.03) & (dat.Latitude<(((-
3/2)*dat.Longitude)+51.457))].index)
    dat = dat.drop(dat[(dat.Longitude>-0.02) &
(dat.Latitude<(((1/9)*dat.Longitude)+51.485))].index)
```

```
    return dat;

def South_of_Thames(dat):
    # Define linear equations to seperate the south of the Thames
    dat = dat.drop(dat[(dat.Longitude<=-0.03) &
(dat.Latitude>(((1/9)*dat.Longitude)+51.498))].index)
    dat = dat.drop(dat[(dat.Longitude<=-0.02) & (dat.Longitude>-0.03) & (dat.Latitude>(((-
3/2)*dat.Longitude)+51.457))].index)
    dat = dat.drop(dat[(dat.Longitude>-0.02) &
(dat.Latitude>(((1/9)*dat.Longitude)+51.485))].index)
    return dat;

def T3_4K(dat):
    dat = dat.drop(dat[(dat.Price<3000)].index)
    dat = dat.drop(dat[(dat.Price>=4000)].index)
    return dat;
def T2_3K(dat):
    dat = dat.drop(dat[(dat.Price<2000)].index)
    dat = dat.drop(dat[(dat.Price>=3000)].index)
    return dat;
def T1_2K(dat):
    dat = dat.drop(dat[(dat.Price<1000)].index)
    dat = dat.drop(dat[(dat.Price>=2000)].index)
    return dat;
def T0_1K(dat):
    dat = dat.drop(dat[(dat.Price>=1000)].index)
    return dat;
Hex_map(T3_4K(dat))
#plot_3d(North_of_Thames(Hackney(dat)),2)
#KDE(North_of_Thames(Hackney(dat)))
#Price_hist(North_of_Thames(Not_Hackney(dat)))
#print(np.mean(Hackney(North_of_Thames(dat)).Price.values))
#print(np.mean(Not_Hackney(North_of_Thames(dat)).Price.values))
#print(sci.stats.ttest_ind(Hackney(North_of_Thames(dat)).Price.values,Not_Hackney(North
_of_Thames(dat)).Price.values));

"""
print(np.mean(dat.Price.values))
print(np.mean(North_of_Thames(dat).Price.values))
print(np.mean(South_of_Thames(dat).Price.values))
print(sci.stats.ttest_ind(South_of_Thames(dat),North_of_Thames(dat)));
print(sci.stats.ttest_ind(Hackney(dat),Not_Hackney(dat)));
print(sci.stats.mannwhitneyu(Hackney(dat),Not_Hackney(dat)));
print(np.mean(dat.Price.values))
print(np.mean(Not_Hackney(dat).Price.values))
print(np.mean(Hackney(dat).Price.values))
```

```
print('The number of properties in London that cost >£5000 pcm is ' + str((1- (dat.shape[0] -
sum(dat.Price.values>5000))/dat.shape[0])*100) + '% of all properties in London')
"""
```