Predicting popularity of a NYC apartment listing using renthop data

By Matthew Chung

THE WORST ROOM

A BLOG ABOUT TRYING TO FIND AFFORDABLE HOUSING IN NEW YORK CITY

8 May

Only $300!

Source: TheWorstRoom.com

Washington Heights, Manhattan. $300/month

(The Breakfast Nook)

Outline

- *Renthop action statement:* “Apartment hunting can be overwhelming, but we realized that finding a new home isn't about looking at every apartment listing, it's about looking at the best ones. Quality, not quantity.”
- Quality is measured on a ‘HopScore’
- Each listing includes features, descriptions, bathrooms, bedrooms, pictures, location etc.

Project Summary

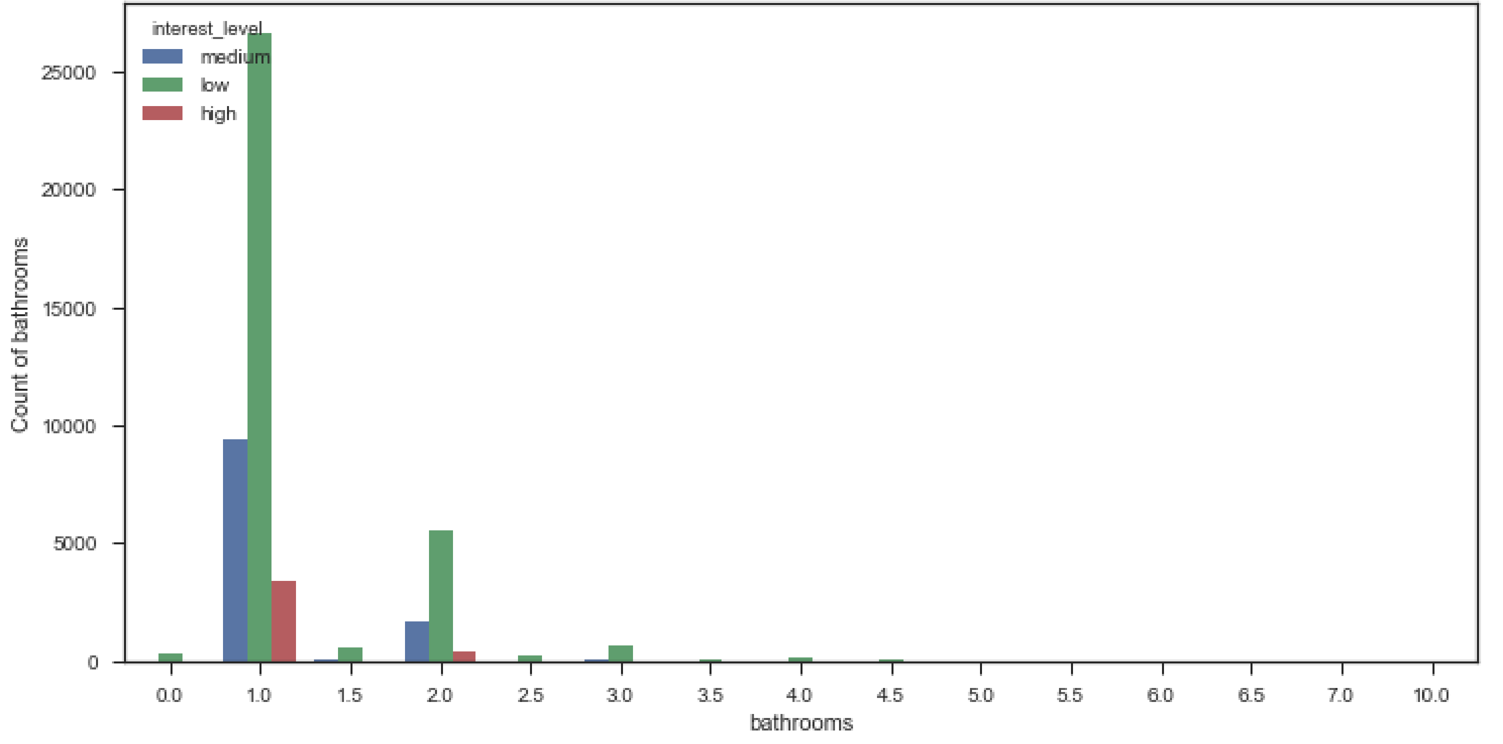
- *Given features can we predict interest level?*
- *Interest level prediction of an apartment in NYC being either High, Medium or Low*
- Using Log Loss formula to evaluate the difference between predicted and actual results
- Data comes from the renthop website's actual listings which are anonymized by building id

Summary continued

- Plan: Using Language Processing algorithms on the features & descriptions to transform categorical variables to continuous to feed into the models
- Use regression and classifier models to predict interest level

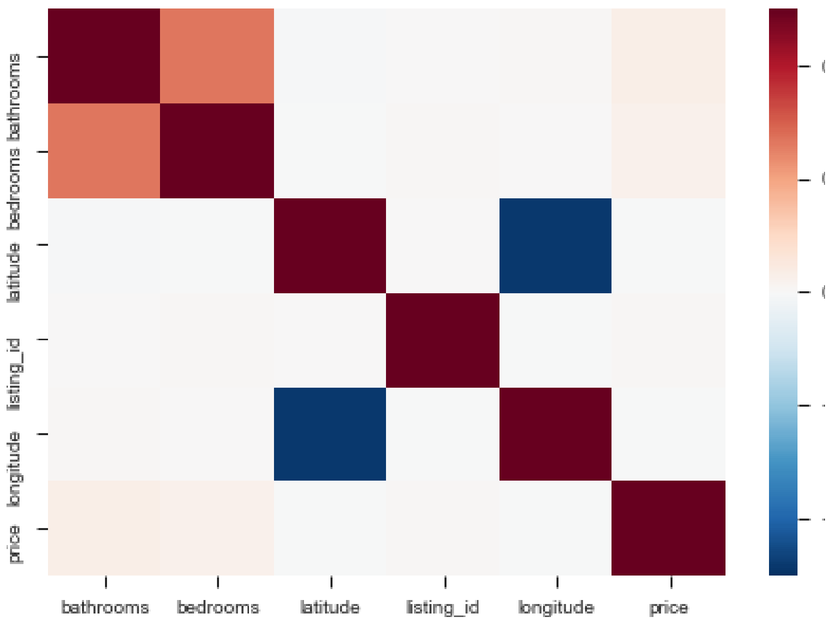
MODELING INSIGHTS

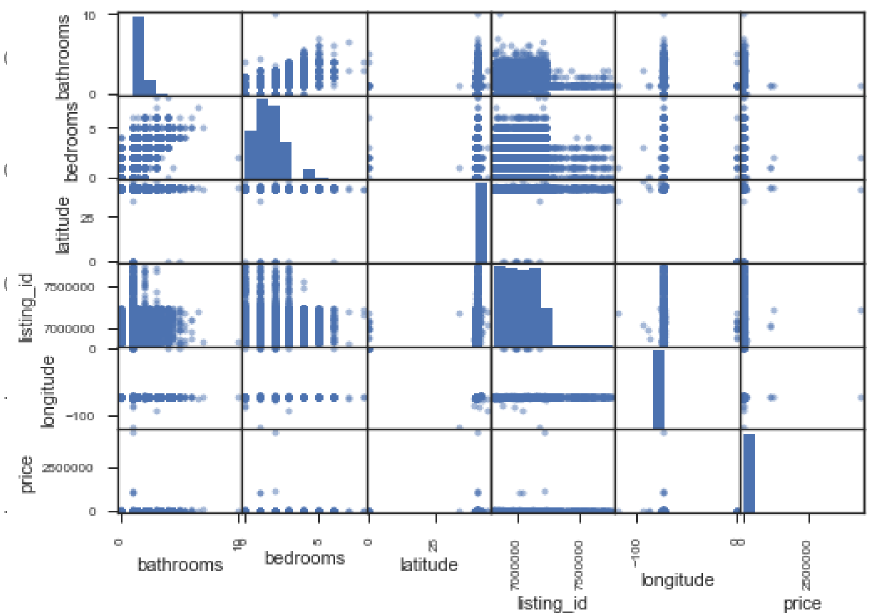
There were 313 listings(out of 49352) with no bathrooms and people were still interested:



Modeling insights continued

- Scatter plots and correlation map of variables;





Data fields: bathrooms, bedrooms, building_id, created, description, display_address, features, latitude, longitude, manager_id, photos(links), price, street_address, interest_level

Modeling

- *Best model for the baseline log loss seems to be RandomForrestClassifier;*

```
model = RandomForestClassifier(n_estimators=1000)
model.fit(X_train, y_train)
y_val_pred = model.predict_proba(X_val)
log_loss(y_val, y_val_pred)
```

```
0.6402424379786356
```

```
model = GradientBoostingClassifier()
model.fit(X_train, y_train)
y_val_pred = model.predict_proba(X_val)
log_loss(y_val, y_val_pred)
```

```
0.6438259822828972
```

```
model = LogisticRegression()
model.fit(X_train, y_train)
y_val_pred = model.predict_proba(X_val)
log_loss(y_val, y_val_pred)
```

```
0.7186543652672564
```

Insights on Features variable

- Wordcloud of the features;

Feature Engineering

Helper function to reduce the amount of unique words in the features columns

```
def clean(a):
    a = str(a)
    a = a.replace('-', ' ')#etc.etc.
    a = a.replace('_', ' ')
    a = a.replace('&', 'and')
    a = a.replace('24/7', '24')
    a = a.replace('24hr', '24')
    a = a.replace('24hour', '24')
    a = a.replace('24 hour', '24')
    a = a.replace('a/c', 'aircon')
    a = a.replace('air conditioner', 'aircon')
    a = a.replace('bicycle', 'bike')
    a = a.replace('concierge', 'doorman')
    a = a.replace('concierge service', 'doorman')
    a = a.replace('counter tops', 'counters')
    a = a.replace('countertops', 'counters')
    a = a.replace('granite kitchen', 'granite counters')
    a = a.replace('dish washer', 'dishwasher')
    a = a.replace('full tie', 'ft')
    a = a.replace('indoor swimming pool', 'indoor pool')
    a = a.replace('laundry on every floor', 'laundry on floor')
    a = a.replace('media screening room', 'media room')
    a = a.replace('one month free rent', 'one month free')
    a = a.replace('prewar', 'pre war')
    a = a.replace('roofdeck', 'roof deck')
    a = a.replace('ss appliance', 'stainless')
    a = a.replace('storage facilities', 'storage')
    a = a.replace('twenty four hour', '24')
    a = a.replace('washer and dryer', 'washer/dryer')
    a = a.replace('wi fi', 'wifi')
    return a
```

Map words into a matrix

```
from sklearn_pandas import DataFrameMapper
mapper = DataFrameMapper([
    ('featured', CountVectorizer(binary=True, ngram_range=(1, 2)))

])
features_sparse=mapper.fit_transform(df)
```

```
X = sparse.hstack([df[new_cols_to_keep], features_sparse]).tocsr()
y = df['interest_level']
X_train, X_val, y_train, y_val = train_test_split(X,y, test_size=0.33)
```

Results

- *Did my Feature Engineering improve results;*

```
model = GradientBoostingClassifier()
model.fit(X_train, y_train)
y_val_pred = model.predict_proba(X_val.toarray())
log_loss(y_val, y_val_pred)
```

```
0.62762756094982386
```

- *By .02!*

Conclusion

- A Log Loss of ~.64 ranks about 1,800 of the 2,488 submissions(72nd percentile!)
- .49194 was the best

- Approaches for future work; more feature engineering on text features(word2vec etc.), image processing, neighborhood classification with lat/longitude, xgboost if I can ever get it to work on my laptop

The end!

Data from: https://www.kaggle.com/c/two-sigma-connect-rental-listing-inquiries

Always looking for work/and or project collaborations!

Email: Mat@KuckChung.com