University of New Haven
TAGLIATELA COLLEGE OF ENGINEERING
Electrical & Computer Engineering and Computer Science

Electrical & Computer Engineering & Computer Science (ECECS)

Data Engineering

Final Project
TECHNICAL REPORT

By:

1. Srijani C. Schall2@unh.newhaven.edu
2. Omkar A. Domkal@unh.newhaven.edu
3. Abhishek K. Akukrl@unh.newhaven.edu
4. Vaishnavi M. Vmalal@unh.newhaven.edu
5. Sai M. Smiya@unh.newhaven.edu

CONTENTS

Title of Project

Rental Price Predictor

GitHub link:

https://github.com/UNH-FantasticFive

Submitted on: December 11th, 2022.

Abstract

The purpose of this project was to develop a way for students to easily access information about rental properties near the campus all in one place. Using data found on Zillow we created a dataset of Connecticut rental property prices in several counties in the New Haven area. We took into account factors like number of bedrooms, time of the year, and proximity to campus, which would be used to train our model to output the approximate rental cost of each property.

Price and distance of apartments are the main factors to consider when looking at rental properties and the most important concern for international students. Therefore, the model will provide students with valuable insights into the properties on the market, which will allow them to compare prices easier and be confident in the rental property they choose.

Introduction

Post Covid-19, after waiting for 1½ years, a lot of International students came to UNH for their studies. Finding accommodation is the biggest problem for them. They are paying a lot of money to the motels until they find a house with a good price to live in. So we wanted to help the students by creating a website which will predict the prices of houses with 1 to 5 bedrooms near the University of New Haven campus. For this we are creating a model using linear regression algorithm and predicting the prices of houses based on previous renting data.

Review of Available Research

Specific brands have sizing charts and applications, but they are very specific to each brand, so we wanted to create a model that can be applicable to all different brands. This will normalize sizing and make it easier for consumers. The more data from various sources that is added to our model the more accurate the end results.

Methodology

The scientific method that was followed during the development process of the model application is CRISP-DM (Cross Industry Standard Process for Data Mining). This analytics method consists of six major steps, which were each used and iterated over as the process developed.

Business Understanding: There are several websites that are related to leasing or buying a house. A newly entered International student to the United States will not know anything about the leasing prices

of the houses nearby UNH. So we thought of developing a website especially for students, by decoding the past prices of the houses, so that they can easily search for the houses, meeting their requirements and budget. We have chosen some cities nearby UNH so that students can easily travel to the University through public and private transports. So, we created a model that trains on the previous year's house price with rent data and displays the rent of the house with the selected month and year along with your selected requirements.

Data Understanding: The dataset we have collected from the online resources has the information of previous months and years rents of 11 cities (Branford, North Branford, East Haven, Hamden, Wallingford, Milford, West Haven, New Haven, Orange, Shelton and Woodbridge) which were near to University of New Haven. The data has columns like location of the House, previous rent of the house, months and years, number of bedrooms and bathrooms (1 bedroom, 2 bedroom, 3 bedroom, 4 bedroom, 5 bedroom and Condo), distance from the university, rank of public transport, rank of private transport, inflation rate and total number of months that house has been given to rent. Up-to-date we have 18,019 rows and 6 columns of data with which we have trained the model.

The process used is as follows:

Intro: One of the machine learning algorithms is linear regression. In this section, we have built a linear regression that predicts the rent of the houses. We have used one-hot encoding to distinguish months. At the end of this section, our model will be saved with pickle and, in the next section, we'll deploy our machine learning model with Flask.

```
# importing dependencies

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error

from sklearn.model_selection import cross_val_score
from sklearn.linear_model import Lasso

plt.style.use('fivethirtyeight')
```

```
# importing AWS dependencies

import sagemaker
import boto3
from sagemaker.amazon.amazon_estimator import get_image_uri
from sagemaker.session import s3_input, Session
```

```
# creating S3 bucket

bucket_name = 'rentprediction'
my_region = boto3.session.Session().region_name
print(my_region)

s3 = boto3.resource('s3')
try:
    if  my_region == 'us-east-1':
        s3.create_bucket(Bucket=bucket_name)
    print('S3 bucket created successfully')
except Exception as e:
    print('S3 error: ',e)
```

```
us-east-1
S3 bucket created successfully
```

First we have download the libraries required and have to create a bucket in S3 server in AWS:

After creating the S3 bucket we have to set an output path where our trained model will be saved

```
# set an output path where the trained model will be saved

prefix = 'linear-regression'
output_path ='s3://{}/{}/output'.format(bucket_name, prefix)
print(output_path)
```

```
s3://rentprediction/linear-regression/output
```

```
# importing raw file

df1=pd.read_excel('Raw Data - v5.xlsx',sheet_name='Data')
df1.head()
```

	Month	Location	Rent	Apartment Type	Distance from University (miles)	Public Transportation Rank	Private Transportation Rank	Inflation	t_month
0	2000-01-31	Branford	0.0	1 Bedroom	16.0	10	11	2.7	1
1	2000-02-29	Branford	0.0	1 Bedroom	16.0	10	11	3.2	2
2	2000-03-31	Branford	0.0	1 Bedroom	16.0	10	11	3.8	3
3	2000-04-30	Branford	0.0	1 Bedroom	16.0	10	11	3.1	4
4	2000-05-31	Branford	0.0	1 Bedroom	16.0	10	11	3.2	5

We have to import the dataset file to the S3 bucket which we have created

```
# set an output path where the trained model will be saved

prefix = 'linear-regression'
output_path ='s3://{}/{}/output'.format(bucket_name, prefix)
print(output_path)
```

```
s3://rentprediction/linear-regression/output
```

```
# importing raw file

df1=pd.read_excel('Raw Data - v5.xlsx',sheet_name='Data')
df1.head()
```

	Month	Location	Rent	Apartment Type	Distance from University (miles)	Public Transportation Rank	Private Transportation Rank	Inflation	t_month
0	2000-01-31	Branford	0.0	1 Bedroom	16.0	10	11	2.7	1
1	2000-02-29	Branford	0.0	1 Bedroom	16.0	10	11	3.2	2
2	2000-03-31	Branford	0.0	1 Bedroom	16.0	10	11	3.8	3
3	2000-04-30	Branford	0.0	1 Bedroom	16.0	10	11	3.1	4
4	2000-05-31	Branford	0.0	1 Bedroom	16.0	10	11	3.2	5

We will look at our dataset that contains 6 columns:

We will look at our dataset using pandas:

```
# reading the file from s3 bucket and creating a dataframe

obj = s3.Bucket('rentprediction').Object('linear-regression/file/rawdatav6.csv').get()
df1 = pd.read_csv(obj['Body'], header=0)
df1.head()
```

	Rent	Month	Location	Apartment Type	Inflation	t_month
0	0.0	2000-01-31	Branford	1 Bedroom	2.7	1
1	0.0	2000-02-29	Branford	1 Bedroom	3.2	2
2	0.0	2000-03-31	Branford	1 Bedroom	3.8	3
3	0.0	2000-04-30	Branford	1 Bedroom	3.1	4
4	0.0	2000-05-31	Branford	1 Bedroom	3.2	5

Data Preparation: The data we used was made up of 18019 rows and 6 columns, which contained relevant information for our project. Some columns had missing data points for rent prices which were marked as 0, however rent prices cannot be 0 so we changed them to NaN values. Here's how we did cleaning: We used df1[cols] = df1[cols].replace({'0':np.nan, 0:np.nan}) to replace all the 0 values in the rent column to NaN values. We also used df.info and df.isnull.sum() to look for missing values and see how many null values there are.

```
1  # Checking null values and datatypes in the dataset
2
3  df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18018 entries, 0 to 18017
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Rent            14804 non-null  float64
 1   Month           18018 non-null  object
 2   Location        18018 non-null  object
 3   Apartment Type  18018 non-null  object
 4   Inflation       18018 non-null  float64
 5   t_month         18018 non-null  int64
dtypes: float64(2), int64(1), object(3)
memory usage: 844.7+ KB
```

```
1  df1.isna().sum()
```

```
Rent              3214
Month                0
Location             0
Apartment Type       0
Inflation            0
t_month              0
dtype: int64
```

Now, we will remove the missing values and our dataset will have 14804 rows and 6 columns.

```
# Removing null values from the dataset

df1.dropna(inplace=True)
df1.shape
```

```
(14804, 6)
```

We created a bhk column with just integers to replace the Apartment Type column. This will allow us to create nonlinearity in the data later and make it easier to work with our time series problem. Then we reformed the dataframe and dropped the Apartment Type column and Inflation column to help us with data modeling. Below is the final dataframe:

```
print(df3.shape)
df3.head()
```

```
(12261, 5)
```

	Rent	Month	Location	t_month	bhk
0	128371.0	2021-09-30	East Haven	261	1
1	129469.0	2021-10-31	East Haven	262	1
2	128843.0	2021-11-30	East Haven	263	1
3	128668.0	2021-12-31	East Haven	264	1
4	130580.0	2022-01-31	East Haven	265	1

Finally we had to adjust the rent information we got from zillow as it gave us the purchase price for the properties when we needed the rental price. We went back to Zillow and took the ZHVI data given which is the purchase price for our chosen city as well as the ZORI data which gives the rental amount ratio.

We then divided the ZHVI purchase price with the ZORI rental ratio to get the final rental price.

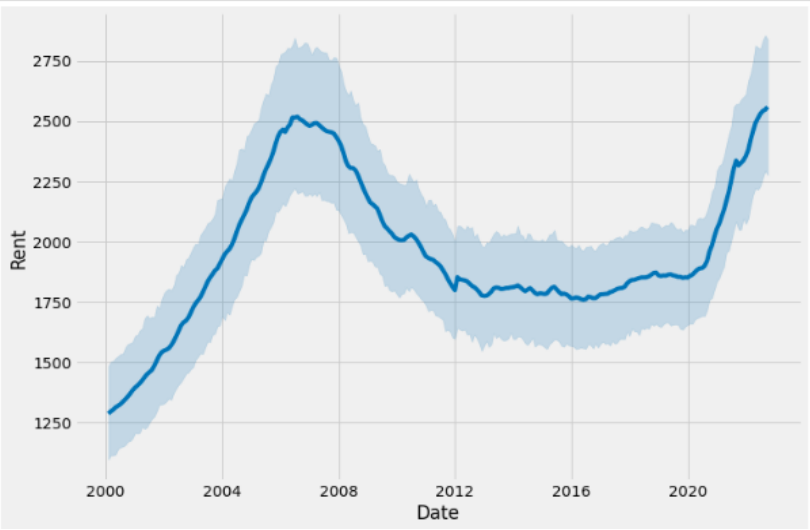
```
# Calculating the rent amount by dividing the column by 150

df4=df3.copy()
df4['Rent'] = df4['Rent'].apply(lambda x: round(x / 150),2)
print(df4.shape)
df4.head()
```

```
(12261, 5)
```

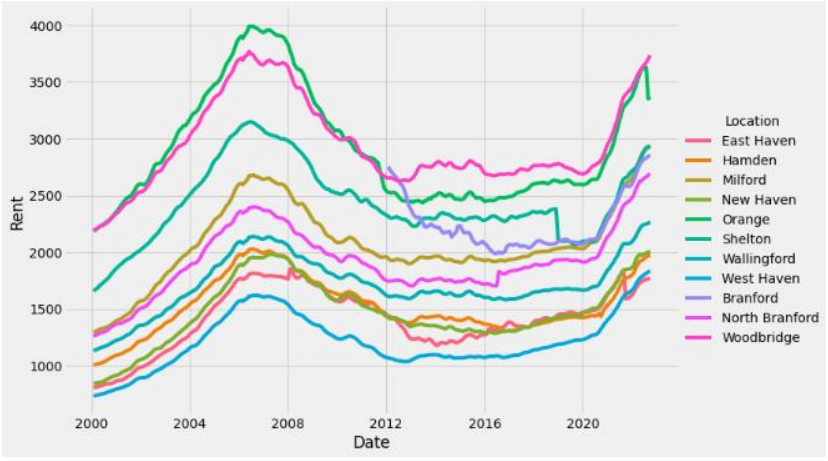
	Rent	Month	Location	t_month	bhk
0	856	2021-09-30	East Haven	261	1
1	863	2021-10-31	East Haven	262	1
2	859	2021-11-30	East Haven	263	1
3	858	2021-12-31	East Haven	264	1
4	871	2022-01-31	East Haven	265	1

Data Analysis and Visualization: Various analysis and visualization tools were used for the steps mentioned above. AWS sagemaker was used to launch and run jupyter notebook and s3 was used to store our data files, json files, and the model. Python and its various libraries have been used, like Pandas, numpy, Matplotlib, Sklearn libraries, pickle, requests, and json. These tools were needed to create line graphs, box plots, and histograms to visualize our cleaned data and to train our tool.

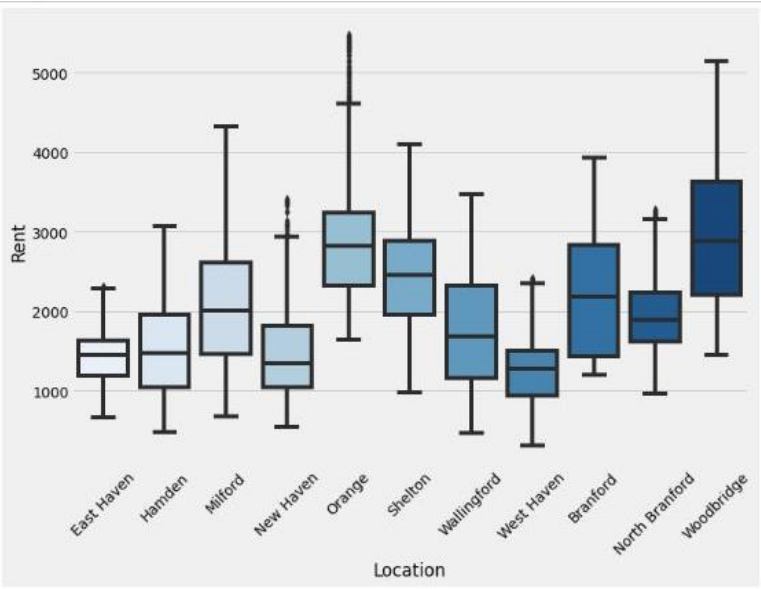
As we are working to predict the rental cost. We will look at the date column carefully and see the rental cost trends from 2000s till now:

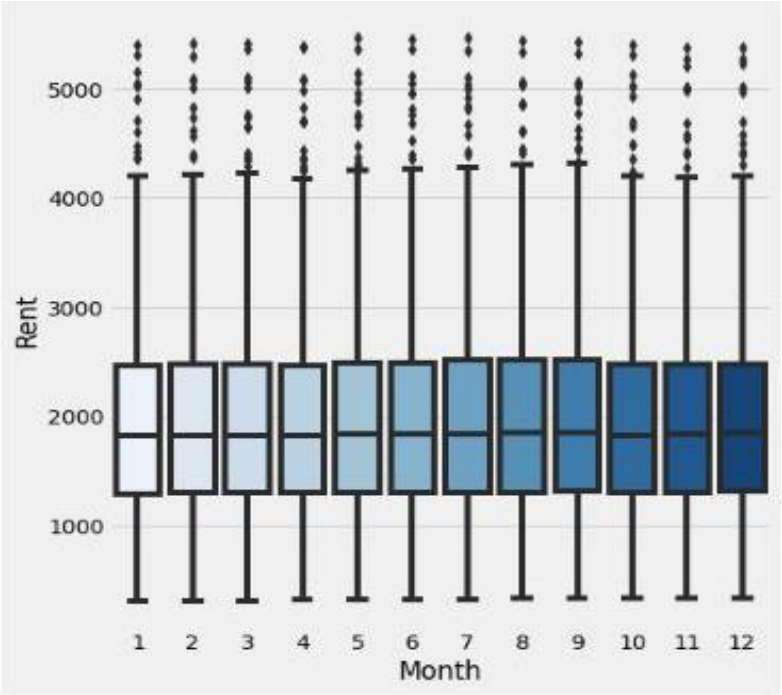


We visualized the rent costs for each location in the data set to find which areas had the highest rents on average. Orange, Woodbridge, and Shelton had the highest rent prices on average.

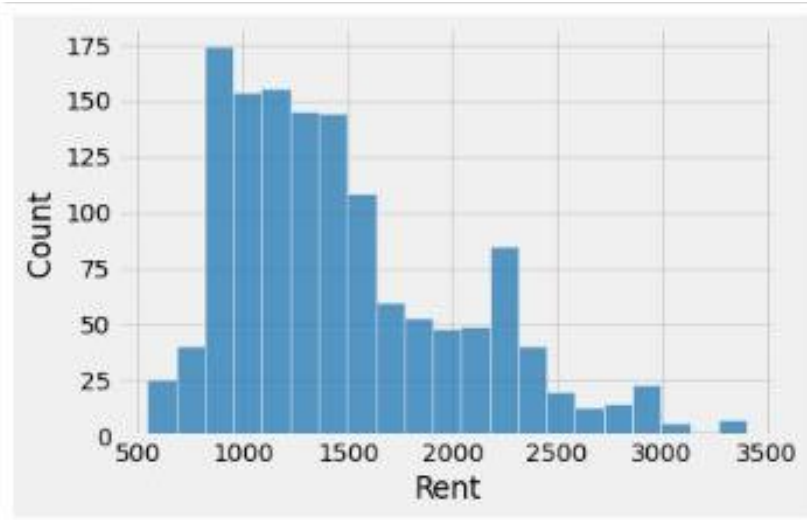


The box plots showed us that West Haven on average has the cheapest rental prices and that rental prices are typically the cheapest in the fall and winter months.





The histograms helped us identify which cities have the higher rental mean values which was Orange, Shelton and Woodbridge.



Modeling: The modeling techniques employed in this project are driven by our aim to train the model to predict the rental prices in several counties near University of New Haven. This involves asking the tenant for their specific interest in choosing the houses based on the bedrooms and distance from the university. Different scikit-learn libraries were used to train our model to predict the prices of the houses. Then the model can be integrated into a website for students to access.

Model Training and Validation: We train the model with the 90% data so that we can predict the current year's housing prices. We will wrangle the data, train, and validate the results. To build our regression model, we need to import train_test_split from sklearn. After training the data, we will validate with the 10% of remaining data.

```
# Splitting the dataset for training and validation

training=df7[df7['Year']<2021]
print(training['Year'].unique())
print(training.shape)
training.head()
```

```
[2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013
 2014 2015 2016 2017 2018 2019 2020]
(11238, 27)
```

	Rent	t_month	BHK	Year	t_month2	t_month3	Apr	Aug	Feb	Jan	...	East Haven	Hamden	Milford	New Haven	North Branford	Orange	Shelton	Wallingford	West Haven
13	516	1	1	2000	1	1	0	0	0	1	...	0	1	0	0	0	0	0	0	0
14	516	2	1	2000	4	8	0	0	1	0	...	0	1	0	0	0	0	0	0	0
15	517	3	1	2000	9	27	0	0	0	0	...	0	1	0	0	0	0	0	0	0
16	516	4	1	2000	16	64	1	0	0	0	...	0	1	0	0	0	0	0	0	0
17	518	5	1	2000	25	125	0	0	0	0	...	0	1	0	0	0	0	0	0	0

5 rows × 27 columns

```
validation=df7[(df7['Year']>=2021)]
print(validation['Year'].unique())
print(validation.shape)
validation.head()
```

```
[2021 2022]
(1023, 27)
```

	Rent	t_month	BHK	Year	t_month2	t_month3	Apr	Aug	Feb	Jan	...	East Haven	Hamden	Milford	New Haven	North Branford	Orange	Shelton	Wallingford	West Haven
0	856	261	1	2021	68121	17779581	0	0	0	0	...	1	0	0	0	0	0	0	0	0
1	863	262	1	2021	68644	17984728	0	0	0	0	...	1	0	0	0	0	0	0	0	0
2	859	263	1	2021	69169	18191447	0	0	0	0	...	1	0	0	0	0	0	0	0	0
3	858	264	1	2021	69696	18399744	0	0	0	0	...	1	0	0	0	0	0	0	0	0
4	871	265	1	2022	70225	18609625	0	0	0	1	...	1	0	0	0	0	0	0	0	0

5 rows × 27 columns

Fitting and saving the model:

Now, we will use Linear Regression to build our model and predict our result by plotting actual and predicted values

```
# Building linear regression model
y_train=training['Rent']
x_train=training.drop(columns=['Rent','Year'])

y_valid=validation['Rent']
x_valid=validation.drop(columns=['Rent','Year'])

mlr=LinearRegression()
mlr.fit(x_train,y_train)
```

```
LinearRegression()
```

```
# Predicting the rental amount on the validation dataset

y_pred=mlr.predict(x_valid)
y_pred
```

```
array([ 919.81787149,  939.60688216,  961.26472341, ..., 3681.08231877,
       3704.75484923, 3729.8067704 ])
```

```
# Plotting the actual vs predicted value

df=pd.DataFrame({'Actual':y_valid,'Predicted':y_pred})
df.sort_index()
```

	Actual	Predicted
0	856	919.817871
1	863	939.606882
2	859	961.264723
3	858	982.659746
4	871	1019.797386
...	...	...
12256	4070	3633.329942
12257	4097	3657.499657
12258	4131	3681.082319
12259	4179	3704.754849
12260	4230	3729.806770

1023 rows × 2 columns

Creating a function to predict the price: We defined predict_price function so that if the student gives location and no.of bedrooms, month and year as input, it will predict the price.

```
: # Creating a function for predicting price

def predict_price(location,t_month,bhk,t_month2,t_month3,month):
    try:
        loc_index_1 = np.where(x_train.columns==location)[0][0]
        loc_index_2 = np.where(x_train.columns==month)[0][0]
    except:
        loc_index_1=-1
        loc_index_2=-1
    x = np.zeros(len(x_train.columns))
    x[0] = t_month
    x[1] = bhk
    x[2]=t_month2
    x[3]=t_month3
    if loc_index_1 >= 0:
        x[loc_index_1] = 1
    if loc_index_2 >= 0:
        x[loc_index_2] = 1
    print(x)

    return mlr.predict([x])[0]
```

Testing the model with different inputs:

Input 1:

```
: # testing the rental amount for different inputs

predict_price('East Haven',236,4,236**2,236**3,'Aug')
```

```
[2.3600000e+02 4.0000000e+00 5.5696000e+04 1.3144256e+07 0.0000000e+00
 1.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00
 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00
 1.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00
 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00]
/home/ec2-user/anaconda3/envs/python3/lib/python3.8/site-packages/sklearn/base.py:445: UserWarning: X does not have valid featu
re names, but LinearRegression was fitted with feature names
  warnings.warn(
```

```
: 1729.7349354827875
```

Input 2:

```
predict_price('Branford',253,3,253**2,253**3,'Jan')
```

```
[2.5300000e+02 3.0000000e+00 6.4009000e+04 1.6194277e+07 0.0000000e+00
 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00
 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00
 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00
 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00]
```

```
/home/ec2-user/anaconda3/envs/python3/lib/python3.8/site-packages/sklearn/bas
re names, but LinearRegression was fitted with feature names
  warnings.warn(
```

```
2450.0556377828525
```

Deployment of model: The model was created in a jupyter notebook and then saved to a disk. Now we want to deploy all of this on a website, so that anyone can access this model. We'll send everything from the backend to the frontend with Flask, but before that we saved our model with pickle and uploaded the pickle file to the S3 bucket.

```
import pickle
with open('ct_home_prices_model.pickle','wb') as f:
    pickle.dump(mlr,f)
```

```
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix, 'model/ct_home_prices_model.pickle')).
upload_file('ct_home_prices_model.pickle')
s3_input_train = sagemaker.TrainingInput(s3_data='s3://{}/{}/file'.format(bucket_name, prefix), content_type='pickle')
```

```
import json
columns = {
    'data_columns' : [col.lower() for col in x_train.columns]
}
with open("columns.json","w") as f:
    f.write(json.dumps(columns))
```

```
boto3.Session().resource('s3').Bucket(bucket_name).Object(os.path.join(prefix, 'model/columns.json')).upload_file('columns.json')
s3_input_train = sagemaker.TrainingInput(s3_data='s3://{}/{}/file'.format(bucket_name, prefix), content_type='json')
```

Flask: We will setup a new virtual environment:

python3 -m venv .venv **source**

.venv/bin/activate

And install flask using–pip install flask. Now to check out everything's working fine follow these steps. Create a new file — I'm naming mine app.py — and paste the following code (that's a minimal app example extracted from the Flask documentation) from flask import Flask app = Flask(__name__)

```
@app.route("/")

def home():

    return "Hello, Flask!"
```

Open up a terminal, use cd to go to the working directory where app.py is located, and run the following command – python app.py. This will show Hello, Flask on the web.

Now let me explain to you what that little script does.

- **app = Flask(__name__): Creates an instance of the Flask class. __name__ is a variable that represents the name of the application's module (this helps Flask know where to look for resources like "templates" which we'll use later)**
- **@app.route("/"): @ represents decorators (they modify the behavior of a function or class). The route() decorator tells Flask what URL should trigger our function. In our example, the homepage (/) should trigger the hello_world() function.**
- **def home(): This function returns a message that will be displayed in the browser.**

Now it's time to start building our web app in the app.py file we created before. First, we import Flask, request, and render_template. Then, we load the linear regression model model.pkl that we previously saved with pickle.

To build the front end we created a folder named "templates" in the folder where our app is located. Inside this "templates" folder we created an HTML file. I named this file index.html.

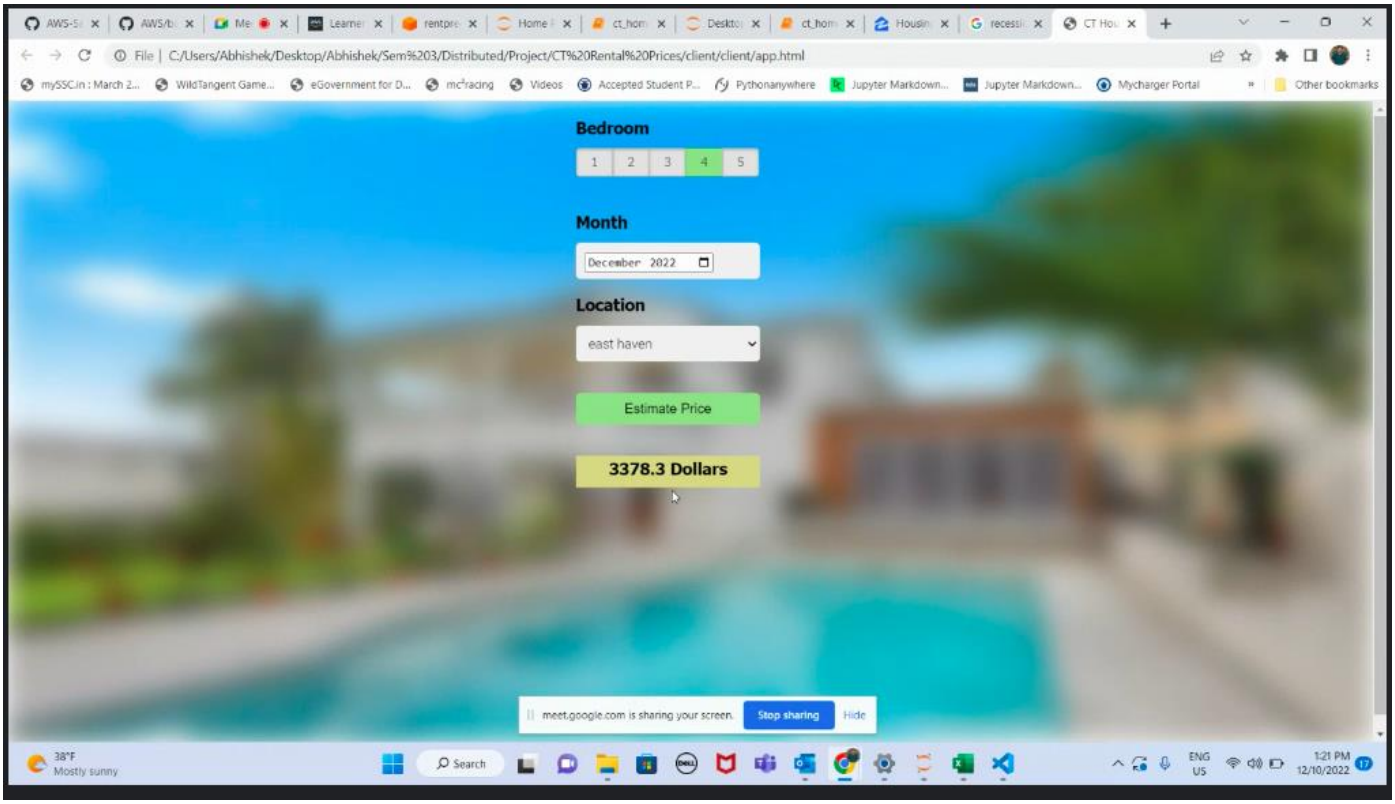
After this we have updated the html file and app.py file to build the front end and predict our result.

Results Section

As mentioned in the introduction, two major problems for international students are finding a rental property that is priced well and close to campus, but looking for a place that suits those criteria can take a very long time and it is a huge problem. We decided to create an application that can do the heavy lifting and tell the students which cities would be perfect for them to rent based on certain features.

This is the website we created with our data and this model can be used by users to find the rental price based on # of bedrooms, the month they are looking, and the location. This website is designed to address students' concerns about rental price and location as well as streamline the process of finding a rental property.

Now, our model will be ready to use on browsers. Our final result of our model is:



Discussion

The purpose of this project was to develop a model that estimates rental prices to give students a better idea of what to expect when renting. There were four features of concern when training our model to output rental prices. Three out of four features are user inputted and help in accessing the rental price based on their criteria.

In our research it seemed that students were concerned with the unaffordability of rental properties. This concern was prevalent among international students as the cost of living in the United States is much higher than they are used to.

Conclusion

One way to improve the accuracy of this model would be to add more data to train our model and to incorporate the inflation rate into our model when predicting rental rates. These factors impact rental prices and by training our model with these values we will be able to provide more accurate prices.

In addition, international students are new to the country and don’t know what the expected rental cost is in different cities so our model provides them with an easy way to figure this out. In order to make the rental process easier for students we created our product which takes into account all the relevant information related to rental cost and location.

Contributions/References

1. Burns, J.B. (2011, May 28). Jean genie, Your number is up…or down. *The Age (Melbourne, Australia)*, P. 7.
2. Catikkas, F. (2011). Physical correlates of college students' body image satisfaction level. *Social Behavior and Personality, 39*(4), 497-502.
3. Clifford, S. (2011, April 25). One size fits nobody: Seeking a steady 4 or 10. *The New York Times*, p. 1.
4. Kim, H., and Damhorst, M (2013). Gauging Concerns with Fit and Size of Garments among Young Consumers in Online Shopping. *Journal of Textile and Apparel, Technology and Management*, vol. 8, no. 3, ojs.cnr.ncsu.edu/index.php/JTATM/article/view/4566.
5. https://www.oberlo.com/blog/online-shopping-statistics