



Predicting Building Energy Usage


w207 section 3 - C. Ilin
Donahoe, Meier, Rosenberg
<https://github.com/bennnyys/w207-sec3-final-proj-DMR>

Sam



Motivation

Sam



What question are we working on? Why is it interesting?

- Can we predict energy usage from building characteristics?
 - Energy use intensity - site
- Current modeling requires many inputs, lots of expensive time, specific people, and computational power
 - EnergyPlus, DOE2, eQUEST, etc.
- Reduced order modeling can more easily inform building system planning for new construction as well as retrofit decision making
- If we change one thing in a building, will it save energy?
- Target - within 20% error

SAM

Question: Can we predict site energy usage from publically available building performance data?

EUI is the energy usage intensity of a building, simply you can think of it as the energy usage normalized by square footage.

This value is measured by a building operator, but is not necessarily shared with everyone who may be interested in it, and it is also useful to estimate EUI when considering energy retrofit work so that you can compare a new estimate with the measured value in the real world

Current modeling practices are time intensive, require specialized people, and is computationally taxing

If reduced order modeling can be used to predict EUI values, energy retrofit planning, utility strategy planning, building code production and commissioning, and other energy efficiency objectives would be significantly easier to complete.

Really, we want to know - if we change something in this building, how much might that impact energy usage. And hopefully, we can get that to within 20% accuracy which would be reasonable for actual implementation, although within 5% would be ideal



Estimated “Best Features”

- year of construction
 - building type
 - window-wall ratio
 - wall r-value
 - floor r-value
 - ceiling r-value
 - window surface area
- climate zone
 - heating type
 - cooling type
 - number of stories
 - number of occupants
 - estimated air leakage

Sam

From research and subject matter expertise, we have identified a few features that would likely be very effective for predicting site EUI, unfortunately due to the proprietary nature of this dataset, we don't have access to all of these, but these data are commonly available in circumstances where access can be granted

Now on to Ben to discuss more about the dataset that we used for this project



Ben



What data are we using?

- U.S. Department of Energy Building Performance Database
 - Energy consumption and building characteristics of actual buildings from 16 states/municipalities in the U.S. (New York, San Francisco, Chicago, D.C., Seattle, Philadelphia, Austin, etc.)
- Up to 10 years of panel data per building
- Number of rows: 296,065
- Number of features: 33
- Main features
 - Site EUI (energy use intensity: kBtu per square feet, 91k NaN)
 - Also: year, electric EUI (95k NaN), fuel EUI (135k NaN), GHG emissions (152k NaN)
 - Climate region (combination of temperature and humidity, 823 NaN)
 - Residential or Commercial (0 NaN)
 - Facility type (office, education, warehouse, industrial, lodging, etc., 156k NaN)
 - Floor Area (square feet, 0 NaN)
 - Year built (4,779 NaN)
 - Energy star rating (206k NaN)

Ben will present

- Data source
- Size
- Main features
- Summary stats
-



State/Municipality Dataset Analysis

- Austin - 8,390 (~1,600 rows with roof/ceiling and window glass data)
- Berkeley - 414
- Boston - 6,835 (~4,500 rows with energy star ratings)
- CA Building Energy Benchmarking Program - 3,881
- CA Prop 39 K-12 Program - 1,501
- Cambridge - 4,122
- Chicago - 11,937
- Fannie Mae - 857
- Gainesville - 154,528 (includes cooling, roof_ceiling type, but only 75k with site_eui)
- NYC Ordinance - 54,298
- NY Residential - 439
- Philadelphia - 6,457
- San Francisco - 9,647
- Seattle - 17,002
- Syracuse - 139
- D.C. - 15,618

Ben will present



Fairness

- Location skew - data collected at state/municipal level mostly large coastal cities
- Income skew - buildings which receive energy audits may be skewed to affluent communities
 - Health and safety usually prioritized over energy efficiency in less affluent communities
- Consistency - Commercial vs Residential
 - Commercial buildings have significant variation in construction compared to residential buildings

Ben

Descriptive Statistics: numeric

	year	floor_area	year_built	energy_star_rating	electric_eui	fuel_eui	site_eui	source_eui	ghg_emissions_int
count	296065.000000	2.960650e+05	291286.000000	89543.000000	200966.000000	160800.000000	204210.000000	196936.000000	143854.000000
mean	2013.978505	7.469098e+04	1971.887276	61.809511	29.602798	23.291331	63.231037	126.836547	4.937401
std	2.609692	1.796745e+05	28.742165	28.910037	28.690731	32.060086	54.413696	110.338840	4.202703
min	2010.000000	5.000000e+02	1649.000000	0.000000	0.000000	0.000000	1.001169	1.075772	0.000000
25%	2012.000000	1.635000e+03	1961.000000	40.000000	16.253904	8.182419	32.009412	72.190118	2.881654
50%	2014.000000	3.260000e+03	1978.000000	68.000000	23.384545	15.138294	50.051977	102.748173	4.026421
75%	2016.000000	8.036400e+04	1993.000000	86.000000	33.968446	26.041885	79.382514	146.732770	5.665176
max	2020.000000	6.385382e+06	2020.000000	100.000000	987.466930	936.379589	997.866120	3133.315574	109.708218

Ben will present
EUI - measured in kBtu per square foot

Descriptive Statistics: categorical

```
CLIMATE - number of distinct vals: 12 -----
2A Hot - Humid (Houston-TX)      162918
4A Mixed - Humid (Baltimore-MD)   76484
5A Cool - Humid (Chicago-IL)     23269
4C Mixed - Marine (Salem-OR)     17039
3C Warm - Marine (San Francisco-CA) 11171
3B Warm - Dry (El Paso-TX)       3957
NaN                               823
6A Cold - Humid (Burlington-VT)  183
4B Mixed - Dry (Albuquerque-NM)   95
5B Cool - Dry (Boise-ID)         67
Name: climate, dtype: int64
```

```
BUILDING_CLASS - number of distinct vals: 2 -----
Residential    223601
Commercial     72464
Name: building_class, dtype: int64
```

```
FACILITY_TYPE - number of distinct vals: 82 -----
NaN          156941
Multifamily - Uncategorized    56268
Office - Uncategorized        22030
Education - Other classroom    8350
2-4 Unit Building             5018
5+ Unit Building              4508
Lodging - Hotel                4327
Retail - Uncategorized         3362
Education - College or university 3118
Commercial - Other             2586
Name: facility_type, dtype: int64
```

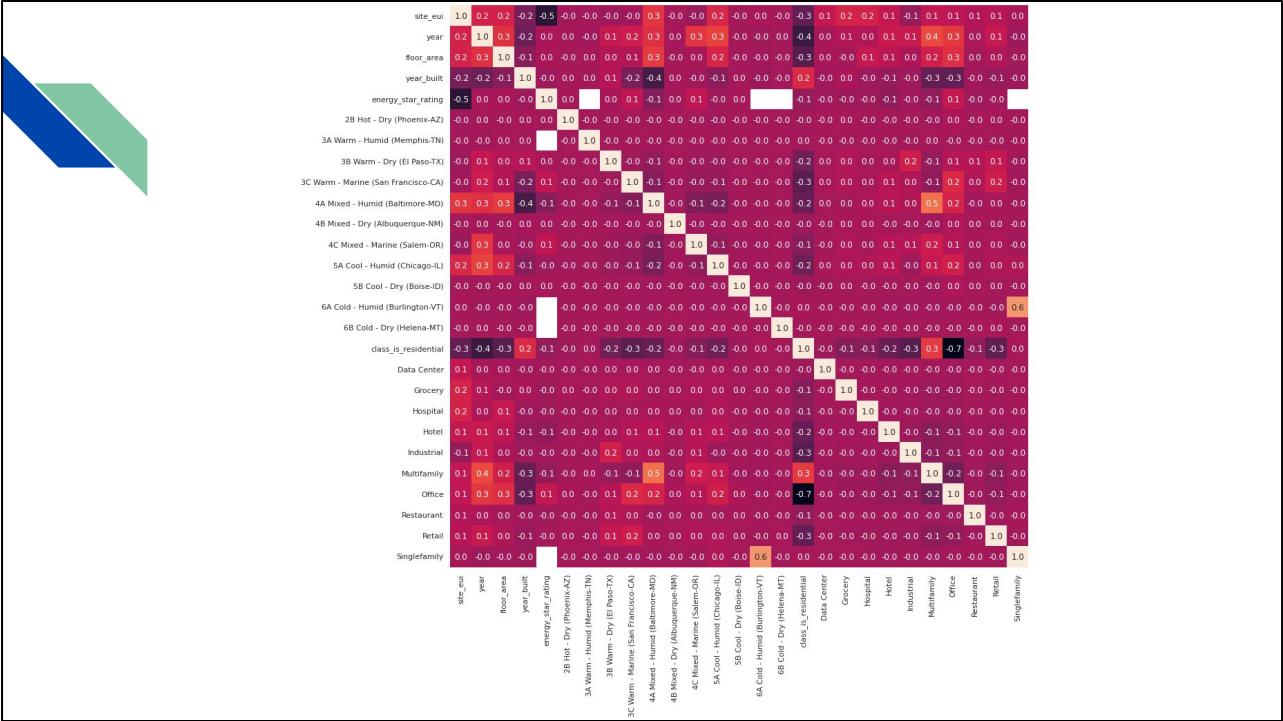
Ben will present



Descriptive Statistics: correlation matrix

- Most promising features
 - Features with low NaN counts
 - Turn categorical features into multi-hot
 - Calculate correlation between features

Ben will present

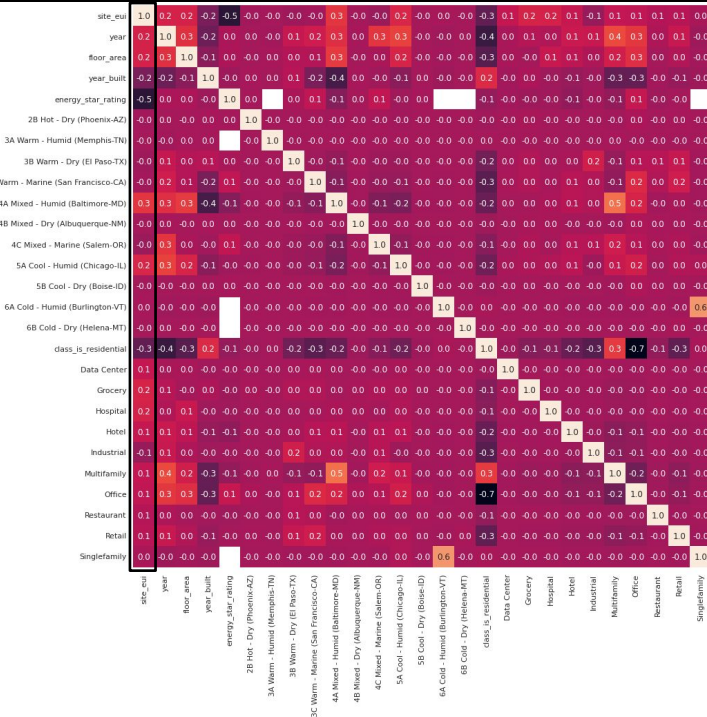


Ben will present - Heatmap

Best predictors:

- Energy_star_rating
- Residential
- Floor_area
- Humid climates

Moving forward want to re-categorize the facility types (grocery, hospital, hotel, etc.) using kmeans

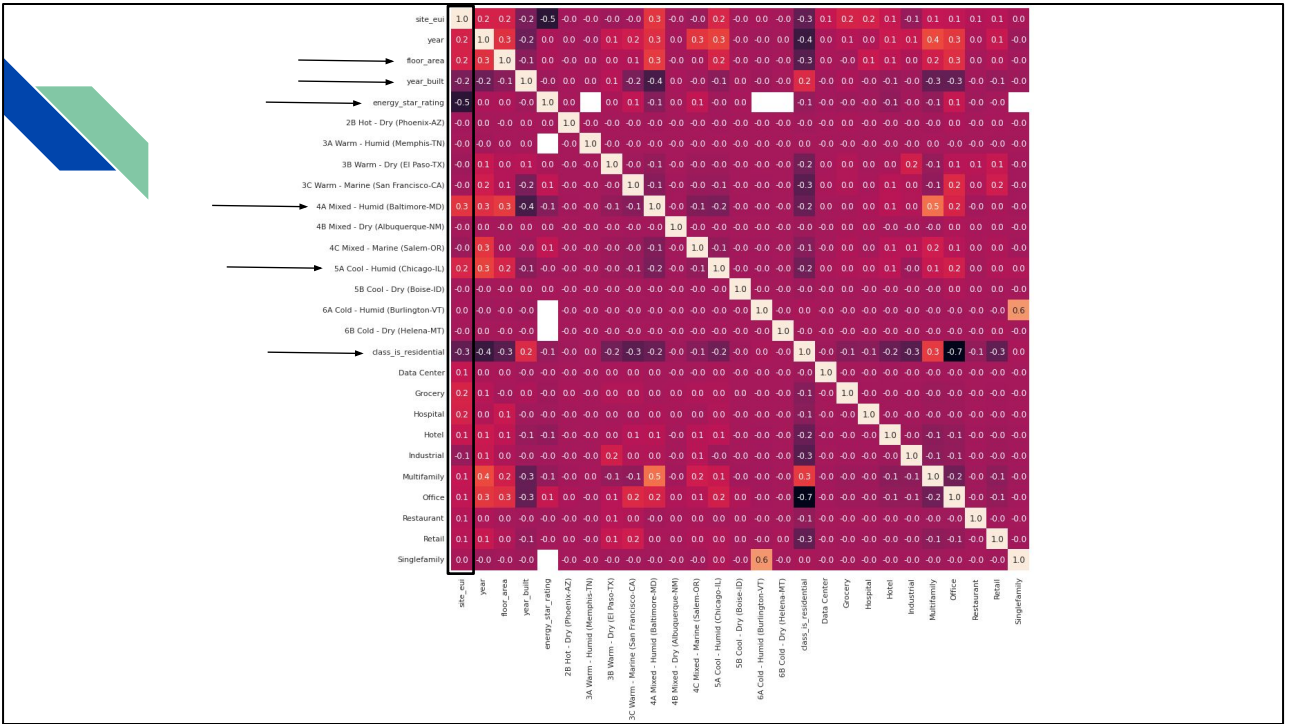


Ben will present - Heatmap

Best predictors:

- Energy_star_rating
- Residential
- Floor_area
- Humid climates

Moving forward want to re-categorize the facility types (grocery, hospital, hotel, etc.) using kmeans



Ben will present - Heatmap

Best predictors:

- Energy_star_rating
- Residential
- Floor_area
- Humid climates

Moving forward want to re-categorize the facility types (grocery, hospital, hotel, etc.) using kmeans



Experiments

CJ



Prediction Algorithm

Linear Regression


- The feature we are trying to predict is a continuous, numerical variable which makes it a good candidate for predicting with a linear model.

Random Forest

- A random forest regression may be best able to handle the number of categorical and missing variables.

Feed Forward Neural Network

- Tried to improve on our initial linear model by building a feed forward neural network with embedding features to try and achieve any available marginal gains.



How will we evaluate our results?

- We chose to use Mean Absolute Error (MAE) as our evaluation metric.
- It is easy to interpret since it tells us how many energy intensity units our predictions were off by.
- MAE is less sensitive to outliers than Mean Squared Error and our data have observations with site EUI values multiple standard deviations above the mean.

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$

CJ

Thoughts on loss function:

- Mean squared error penalizes outliers and we likely have some large ones since the distribution of site_eui values has a long tail (mean = 63.23, std = 54.42, min = 1.00, 75% = 79.38, max = 997.87).
- We may not want the few outliers with very high site_eui to make our predictions for all the other buildings worse.
- Two possible alternatives
 - Use mean absolute error as the loss function since it doesn't penalize large errors
 - Use mean squared error, but winsorize the data by, for example, replacing the site_eui of the top 2% of buildings with the site_eui of the 98th percentile.



Baseline Model

MSE and MAE from the mean

```
MSE Baseline to Train: 1563.4135692561551  
MSE Baseline to Test: 1349.2001965901652
```

```
MAE Baseline to Train: 29.96627005205943  
MAE Baseline to Test: 28.480106853244518
```



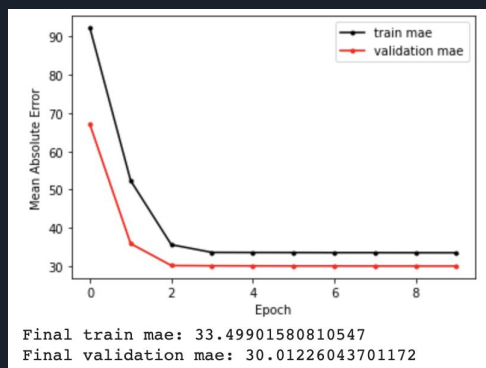
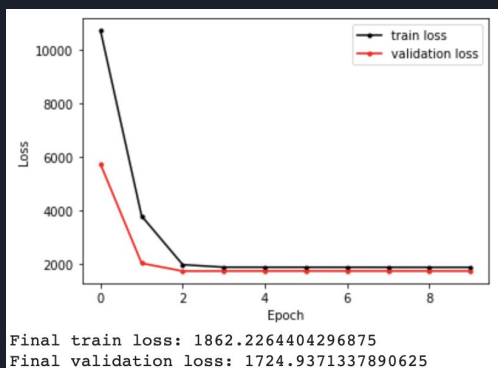
Experiments: Linear Regression

Sam

Thanks! I took on the linear regression portion of this exercise, and historically in the literature and in industry, regression based models are the standard of practice for energy modeling. Explainability is key here, and we are very interested in the weights that come out of the model, so regression is a obvious first step into this type of work.

For these regression models we settled on Adam as our optimizer, and tuned learning rate and epochs as we went

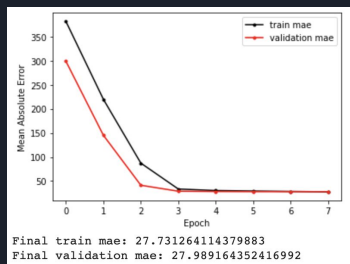
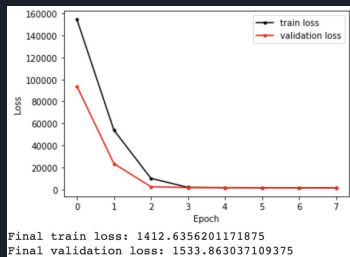
Baseline Linear Model - Year Built



Sam

We started with a baseline model - year built - which was relatively effective, with reasonable loss curves and MAE reaching close to our baseline mean model, but obviously year built is inadequate at describing a building, so we moved on to a more rich feature selection

Linear Model - Full Feature



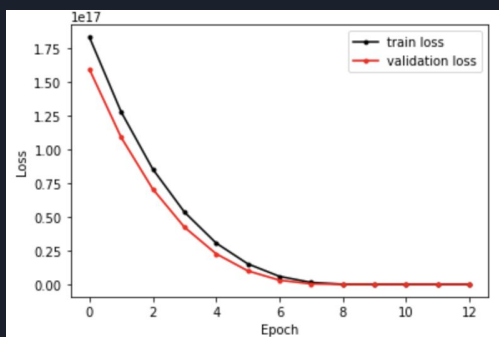
Features include:

- Floor area
- Year of construction
- Year of data collection
- Climate zone - one hot
- Building type - one hot
- Window glass layers - one hot
- Roof type - one hot
- Cooling type - one hot
- Heating type - one hot
- Wall type - one hot

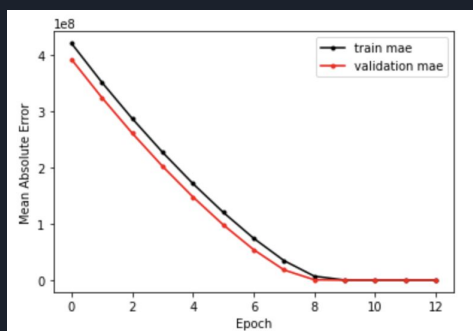
Sam

Here you see our full list of features which we had access to and had populated values in the dataset. We are missing a few potentially important features, but given what was available this was a relatively comprehensive list. With this model, we are actually beating the MAE of our mean baseline model, and again the loss curve converges nicely. We still think there can be some improvement, so it is now time to normalize.

Normalized Baseline Linear Model - Year Built



Final train loss: 1736.6119384765625
Final validation loss: 1636.8787841796875

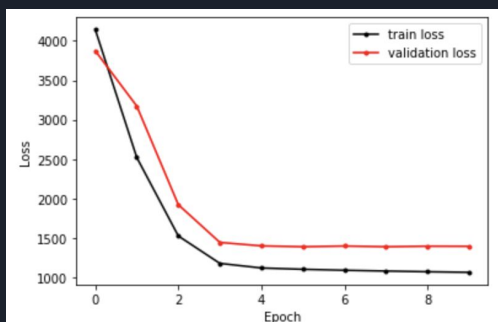


Final train mae: 31.90424919128418
Final validation mae: 28.7767276763916

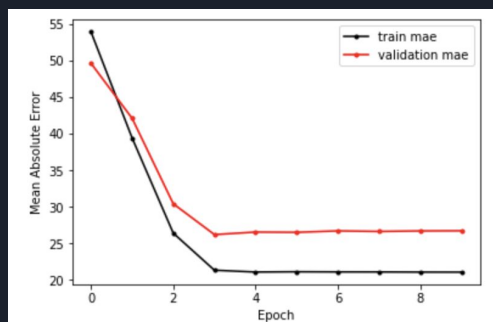
Sam

The normalized baseline regression again has a nice loss curve, and beats the non-normalized baseline, but we run into the same issue of year of construction not describing a building very well at all.

Normalized Linear Model - Full Feature



Final train loss: 1062.92724609375
Final validation loss: 1393.57666015625



Final train mae: 21.0861759185791
Final validation mae: 26.736709594726562

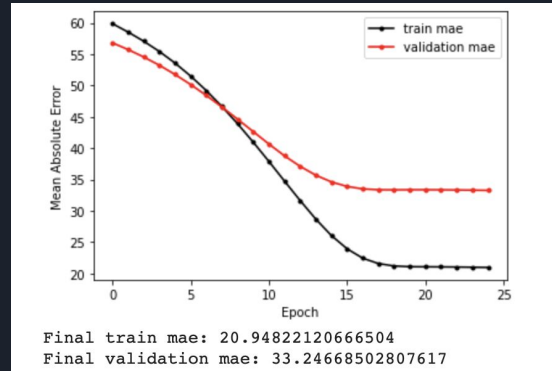
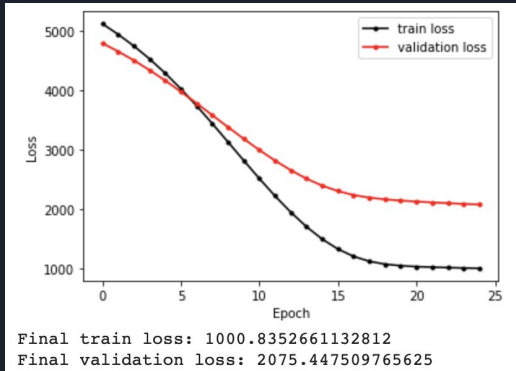
Sam

So, we normalized our full feature model, and here we begin to see pretty good performance. We are right around 21 for our training MAE and 26 for our validation MAE, our lowest yet, which although it isn't shown here, is getting closer to that 20% accuracy, although not quite hitting it yet.

Deep Neural Network Regression

- Normalization layer
- Dense non-linear layer with ReLU activation
- Dense linear single-output layer

```
Model: "sequential_1"
Layer (type)                Output Shape              Param #
-----
normalization (Normalizatio  (None, 55)                111
n)
dense_2 (Dense)              (None, 64)                3584
dense_3 (Dense)              (None, 1)                 65
-----
Total params: 3,760
Trainable params: 3,649
Non-trainable params: 111
```



Sam

Finally we tried a deep neural network with a normalization layer, a non-linear dense layer, and a linear output dense layer, but this model proved difficult in hyperparameter tuning, and didn't end up performing the best. I think this is a very promising method to look at in future studies when more building-descriptive features are available

So... we have 5 models, and it appears the normalized full-feature model is performing best. What do we do with this information?

Best Linear Model (Normalized Full-Feature) Weights

```
Final bias: 6.33
year 222.7343
floor_area 127.4397
year_built 175.1638
clim_2A Hot - Humid 0.3617
clim_3C Warm - Marine 0.0001
clim_5A Cool - Humid 0.0
clim_3B Warm - Dry 0.0184
clim_2B Hot - Dry 0.0415
clim_4C Mixed - Marine 0.361
clim_6B Cold - Dry 0.0002
clim_4B Mixed - Dry 0.0893
clim_5B Cool - Dry 0.1233
clim_4A Mixed - Humid 0.0
clim_6A Cold - Humid 0.0011
clim_3A Warm - Humid 0.0
BC_Residential 0.314
BC_Commercial 0.686
FT_Multifamily 0.0003
FT_Industrial 0.0073
FT_Office 0.0056
FT_Retail 0.0442
FT_Other 0.036
FT_Hotel 0.3233
FT_Restaurant 0.1804
FT_Grocery 0.0113
FT_Hospital 0.0024
FT_Data Center 0.037
FT_Singlefamily 0.0027
```

```
WGL_Single-pane 0.0012
WGL_Double-pane 0.0058
WGT_Low-e 0.0002
roof_Shingles 0.3233
roof_Built-up 0.0168
roof_Slate or tile shingles 0.0
roof_Metal surfacing 0.0023
roof_Other Or Combination 0.0031
roof_Asphalt/fiberglass/other shingles 0.0002
roof_Wood shingles/shakes/other wood 0.0071
roof_Plastic/rubber/synthetic sheeting 0.0006
roof_Green Roof 0.0
cool_Central AC 0.3305
cool_No cooling 0.0
cool_Other 0.0102
cool_Cooling_Heat Pump 0.0
cool_Split AC 0.0
heat_Boiler 0.0002
heat_Resistance Heating 0.0
heat_Heating_Heat pump 0.0
heat_Other 0.0
heat_Heating_Furnace 0.0
wall_Other 0.0013
wall_wall_Wood 0.0004
wall_wall_Metal 0.0
wall_wall_Brick_stone 0.0003
wall_wall_Concrete 0.0007
```

Sam - These weights are the real “meat” of the project. With these weights, an interested party can adjust pieces of a building, like a more efficient window glass type, or heating system, and produce an estimated EUI to compare to the buildings current EUI - without having to deal with thousands of inputs and many hours spent in more complex simulations.

Linear Model Results - Test Set

Full featured normalized model has best performance and had most ideal loss plot

	Year_Built	Full_Feature	Year_Built_Norm	Full_Feature_Norm	DNN_Model
MSE	1617.91	1175.92	1477.95	803.35	965.60
MAE	31.59	25.49	29.90	19.12	20.75

Sam

Finally, here we can see the test data applied to our different linear models, and the one we thought was best seems to have performed best, and we're seeing the MAE under 20 which is actually getting pretty close to under 20% error, so that's good news for the linear model!

Now let's have Ben take a look at random forests



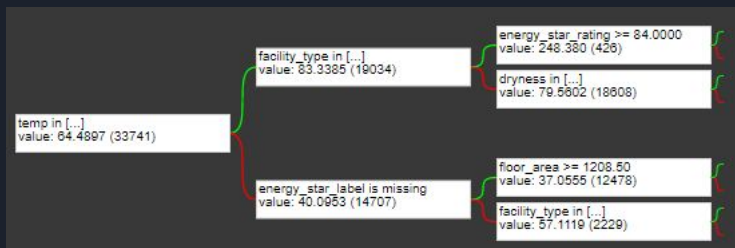
Experiments: Random Forest

Ben

Thanks Sam, I'll now touch on our random forest regression using keras api in tensorflow. Given the large number of categorical and missing values in our dataset and small range of outcome values (site EUI ranges from 1 to 1,000), a random forest regression seemed like a worthwhile experiment.

Random Forest Model

- Features:
 - Facility type, floor area, year_built, roof/ceiling, window glass layers, window glass type, energy star label and rating, cooling ,heating, residential, dryness, temperature
- Data cleaning:
 - Drop rows without site EUI
 - Drop duplicate rows for same building (keep most recent)
 - Train: 34k, Validation: 8k, Test: 19k



Ben

The data cleaning and preprocessing was limited for this experiment. I dropped all rows with null values for site_eui and used only the most recent row for each building.

My first model had a small number of features and some consolidation of categorical variables. This resulted in a mean absolute error of 29 for training and 30 for validation, which seemed promising relative to our initial baseline linear regression model.



Random Forest

Model:

```
# Specify the model.
model = tfdf.keras.RandomForestModel(
    task = tfdf.keras.Task.REGRESSION,
    temp_directory= path + 'output_ben/',
    verbose = 2,
    allow_na_conditions=True,
    max_depth=25,
    min_examples=3,
    num_trees=300,
    random_seed=3000,
    name='Random_Forest_Regressor',
)
```

Train:

```
mse: 540.2357
mae: 10.4744
mape: 32.1941
RMSE: 23.24297004985123
```


Test:

```
mse: 1218.7836
mae: 16.1808
mape: 47.1652
RMSE: 34.9110808961272
```

Ben

- There were several hyperparameters to tune,
 - Number of trees: 100 to 500
 - Maximum tree depth, 10 to 30
 - Minimum examples per leaf: 10 to 1
- After some experimental tuning and evaluation against a validation dataset, the best model had:
 - 300 trees with a maximum tree depth of 25, with each leaf having a minimum of 3 buildings in it
 - Allowing null values to be their own category
 - Sampling with replacement
 - This resulted in a MAE of 10 for training
- Additional models attempted, but difference in MAE between the Train and Validation was more substantial
- The settings for the best model are shown above. Evaluating the model against our test dataset resulted in a MAE of 16.2

I'll now pass it on to CJ to discuss our Neural Network Model.



Experiments: Feed Forward Neural Network

CJ

FFNN

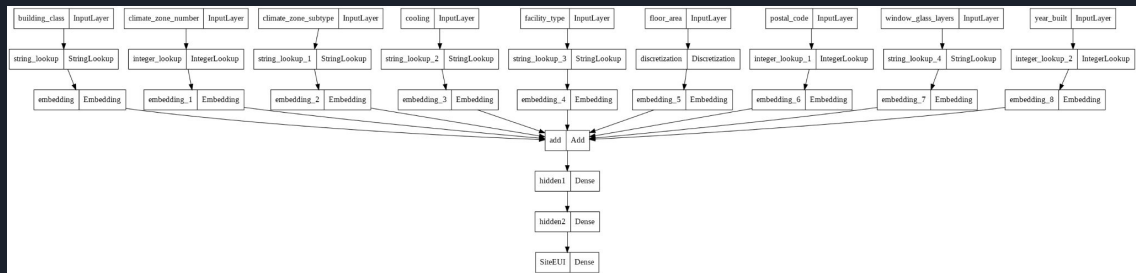
- Extract climate zone number and subtype as individual features
- Convert all features to embeddings
- Use Tensorflow Functional API with a simple model architecture of 2 fully connected layers with 32 nodes each.
- Exponential decay learning rate schedule

```
# Tensorflow input layers
building_class = tf.keras.layers.Input(shape=1), dtype=tf.string, name='building_class')
climate_zone_number = tf.keras.layers.Input(shape=1), dtype=tf.int64, name='climate_zone_number')
climate_zone_subtype = tf.keras.layers.Input(shape=1), dtype=tf.string, name='climate_zone_subtype')
cooling = tf.keras.layers.Input(shape=1), dtype=tf.string, name='cooling')
facility_type = tf.keras.layers.Input(shape=1), dtype=tf.string, name='facility_type')
floor_area = tf.keras.layers.Input(shape=1), dtype=tf.int64, name='floor_area')
postal_code = tf.keras.layers.Input(shape=1), dtype=tf.int64, name='postal_code')
window_glass_layers = tf.keras.layers.Input(shape=1), dtype=tf.string, name='window_glass_layers')
year_built = tf.keras.layers.Input(shape=1), dtype=tf.int64, name='year_built')

# Tensorflow categorical feature layers
building_class_binned = tf.keras.layers.StringLookup(vocabulary=['Commercial', 'Residential'])(building_class)
climate_zone_number_binned = tf.keras.layers.IntegerLookup(vocabulary=[2, 4, 5, 6])(climate_zone_number)
climate_zone_subtype_binned = tf.keras.layers.StringLookup(vocabulary=['A', 'B', 'C'])(climate_zone_subtype)
cooling_vocab = get_feature_vocab(df_train.cooling)
cooling_binned = tf.keras.layers.StringLookup(vocabulary=cooling_vocab)(cooling)
facility_type_vocab = get_feature_vocab(df_train.facility_type)
facility_type_binned = tf.keras.layers.StringLookup(vocabulary=facility_type_vocab)(facility_type)
floor_area_bins = list(np.percentile(df_train['floor_area'], [0, 20, 30, 40, 50, 50, 70, 80, 90]))
floor_area_binned = tf.keras.layers.DiscretizationBin(boundaries=floor_area_bins)(floor_area)
postal_code_vocab = get_feature_vocab(df_train.postal_code)
postal_code_binned = tf.keras.layers.IntegerLookup(vocabulary=postal_code_vocab)(postal_code)
window_glass_layers_binned = tf.keras.layers.StringLookup(vocabulary=['Single-pane', 'Double-pane'])(window_glass_layers)
year_built_vocab = get_feature_vocab(df_train.year_built)
year_built_binned = tf.keras.layers.IntegerLookup(vocabulary=year_built_vocab)(year_built)

# Tensorflow embedding layers
building_class_embed = tf.keras.layers.Embedding(
    input_dim=2, output_dim=embed_dim, input_length=1)(building_class_binned)
climate_zone_number_embed = tf.keras.layers.Embedding(
    input_dim=6, output_dim=embed_dim, input_length=1)(climate_zone_number_binned)
climate_zone_subtype_embed = tf.keras.layers.Embedding(
    input_dim=4, output_dim=embed_dim, input_length=1)(climate_zone_subtype_binned)
cooling_embed = tf.keras.layers.Embedding(
    input_dim=len(cooling_vocab)+1, output_dim=embed_dim, input_length=1)(cooling_binned)
facility_type_embed = tf.keras.layers.Embedding(
    input_dim=len(facility_type_vocab)+1, output_dim=embed_dim, input_length=1)(facility_type_binned)
floor_area_embed = tf.keras.layers.Embedding(
    input_dim=len(floor_area_bins)+1, output_dim=embed_dim, input_length=1)(floor_area_binned)
postal_code_embed = tf.keras.layers.Embedding(
    input_dim=len(postal_code_vocab)+1, output_dim=embed_dim, input_length=1)(postal_code_binned)
window_glass_layers_embed = tf.keras.layers.Embedding(
    input_dim=3, output_dim=embed_dim, input_length=1)(window_glass_layers_binned)
year_built_embed = tf.keras.layers.Embedding(
    input_dim=len(year_built_vocab)+1, output_dim=embed_dim, input_length=1)(year_built_binned)
```

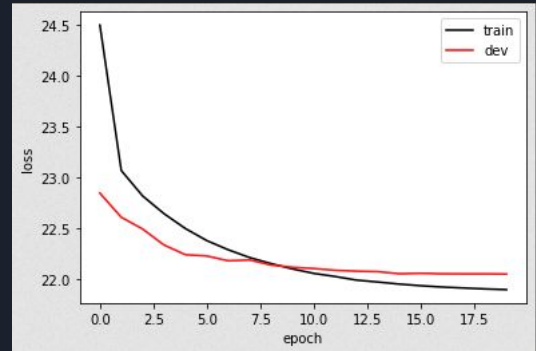
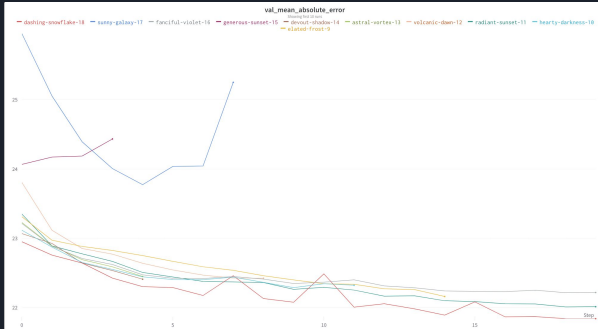
FFNN



CJ

FFNN

It was very difficult to beat the performance of the full-feature, normalized linear regression model
Extensive experimentation and hyperparameter tuning could not return a MAE below 20.



CJ

Experimenting with different NN architectures and numerous hyperparameters quickly became difficult to organize. Weights and Biases is a great tool that will keep track of everything for you so you don't have to worry about forgetting what learning rate was used on that one really good experiment from yesterday for example.




Conclusions

CJ



Conclusions

- More and better data is needed
- Need more representative buildings from the communities this model will be used in
- Residential and commercial buildings should potentially be separately modeled to reduce complexity and more effectively select features



Code and Contributions

CJ



Code and Contributions

Code

- GitHub Repo: <https://github.com/bennyvys/w207-sec3-final-proj-DMR>

	Ben	Sam	CJ
Theoretical Research		X	
Data cleaning	X	X	X
Data splitting	X		
Hyperparameter tuning	X	X	X
Augmentations	(random forest)	(linear regression)	(FFNN)
Presentation Slides	X	X	X

CJ