**Final Project**

Using Azure Machine Learning to predict house prices

**Problem Statement:**

In 2017, around 5.57 million existing homes were sold in the US (Statista, U.S. existing home sales 2005-2018). Determining the optimal sell price is a critical decision of the seller. If priced too low, the seller leaves money on the table. If priced too high, the house will sit on the market unsold. A negative perception can result when a house is on the market for a considerable amount of time, or when the price is reduced often. Using existing sales, how accurately can we predict the selling price of a home before it is put onto the market? We will attempt to build such a system to predict housing prices using Azure Machine Learning.

**Data Source:**

Kaggle House Prices: Advanced Regression Techniques

https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data

I am working with the training data, which allows me to evaluate the effectiveness of the model against known data. The training data consists of 1,460 rows of data of house sales, each with 81 attributes.

* SalePrice - the property's sale price in dollars. This is the target variable that you're trying to predict. **SalePrice is the attribute my system will predict.**
* MSSubClass: The building class
* MSZoning: The general zoning classification
* LotFrontage: Linear feet of street connected to property
* LotArea: Lot size in square feet
* Street: Type of road access
* Alley: Type of alley access
* LotShape: General shape of property
* LandContour: Flatness of the property
* Utilities: Type of utilities available
* LotConfig: Lot configuration
* LandSlope: Slope of property
* Neighborhood: Physical locations within Ames city limits
* Condition1: Proximity to main road or railroad
* Condition2: Proximity to main road or railroad (if a second is present)
* BldgType: Type of dwelling
* HouseStyle: Style of dwelling
* OverallQual: Overall material and finish quality
* OverallCond: Overall condition rating
* YearBuilt: Original construction date
* YearRemodAdd: Remodel date
* RoofStyle: Type of roof
* RoofMatl: Roof material
* Exterior1st: Exterior covering on house
* Exterior2nd: Exterior covering on house (if more than one material)
* MasVnrType: Masonry veneer type
* MasVnrArea: Masonry veneer area in square feet
* ExterQual: Exterior material quality
* ExterCond: Present condition of the material on the exterior
* Foundation: Type of foundation
* BsmtQual: Height of the basement
* BsmtCond: General condition of the basement
* BsmtExposure: Walkout or garden level basement walls
* BsmtFinType1: Quality of basement finished area
* BsmtFinSF1: Type 1 finished square feet
* BsmtFinType2: Quality of second finished area (if present)
* BsmtFinSF2: Type 2 finished square feet
* BsmtUnfSF: Unfinished square feet of basement area
* TotalBsmtSF: Total square feet of basement area
* Heating: Type of heating
* HeatingQC: Heating quality and condition
* CentralAir: Central air conditioning
* Electrical: Electrical system
* 1stFlrSF: First Floor square feet
* 2ndFlrSF: Second floor square feet
* LowQualFinSF: Low quality finished square feet (all floors)
* GrLivArea: Above grade (ground) living area square feet
* BsmtFullBath: Basement full bathrooms
* BsmtHalfBath: Basement half bathrooms
* FullBath: Full bathrooms above grade
* HalfBath: Half baths above grade
* Bedroom: Number of bedrooms above basement level
* Kitchen: Number of kitchens
* KitchenQual: Kitchen quality
* TotRmsAbvGrd: Total rooms above grade (does not include bathrooms)
* Functional: Home functionality rating
* Fireplaces: Number of fireplaces
* FireplaceQu: Fireplace quality
* GarageType: Garage location
* GarageYrBlt: Year garage was built
* GarageFinish: Interior finish of the garage
* GarageCars: Size of garage in car capacity
* GarageArea: Size of garage in square feet
* GarageQual: Garage quality
* GarageCond: Garage condition
* PavedDrive: Paved driveway
* WoodDeckSF: Wood deck area in square feet
* OpenPorchSF: Open porch area in square feet
* EnclosedPorch: Enclosed porch area in square feet
* 3SsnPorch: Three season porch area in square feet
* ScreenPorch: Screen porch area in square feet
* PoolArea: Pool area in square feet
* PoolQC: Pool quality
* Fence: Fence quality
* MiscFeature: Miscellaneous feature not covered in other categories
* MiscVal: $Value of miscellaneous feature
* MoSold: Month Sold
* YrSold: Year Sold
* SaleType: Type of sale
* SaleCondition: Condition of sale

**Hardware Used:**

Windows 10 64 bit processor desktop

**Software Used:**

Microsoft Azure Machine Learning Studio (<https://studio.azureml.net/>)

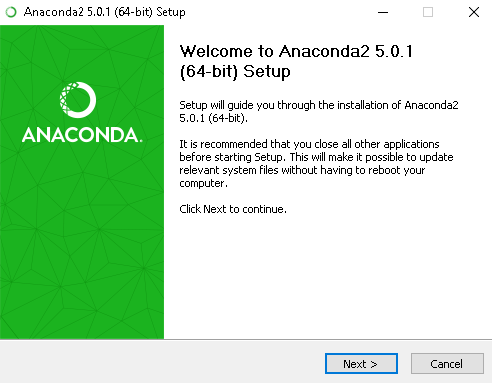
Anaconda 5.0 distribution of Python 64 bit (<https://www.anaconda.com/>), which includes:

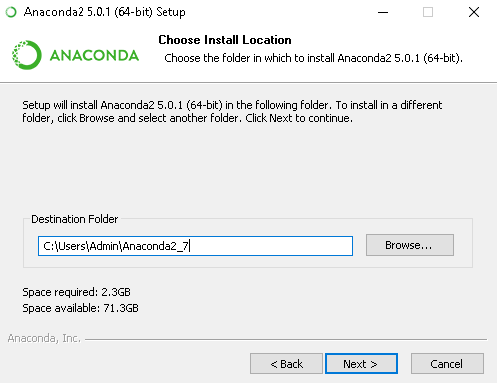
* Python 2.7.14
* Jupyter (visualization tool)

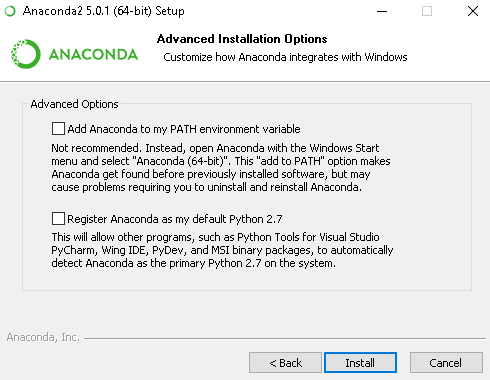
Microsoft Azure Machine Learning Studio is a web based tool, which requires no installation or configuration to use.

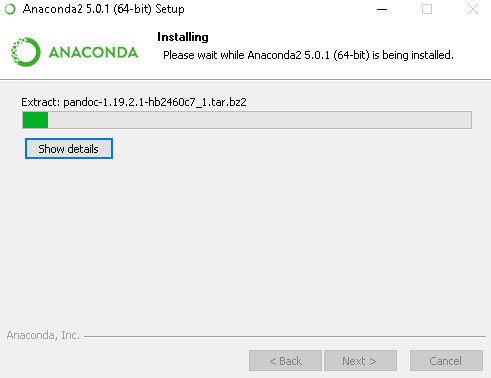
Jupyter Notebooks, using Python 2.7, within Microsoft Azure Machine Learning Studio is a web based tool, which requires no installation or configuration to use.

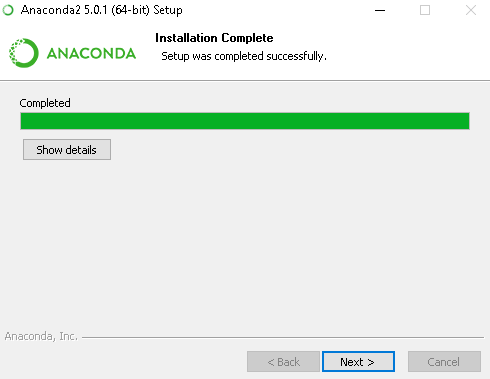
Anaconda installation steps:

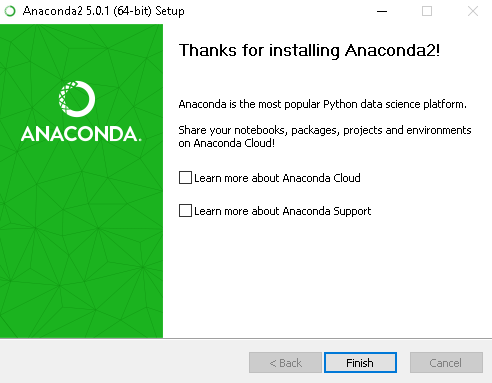




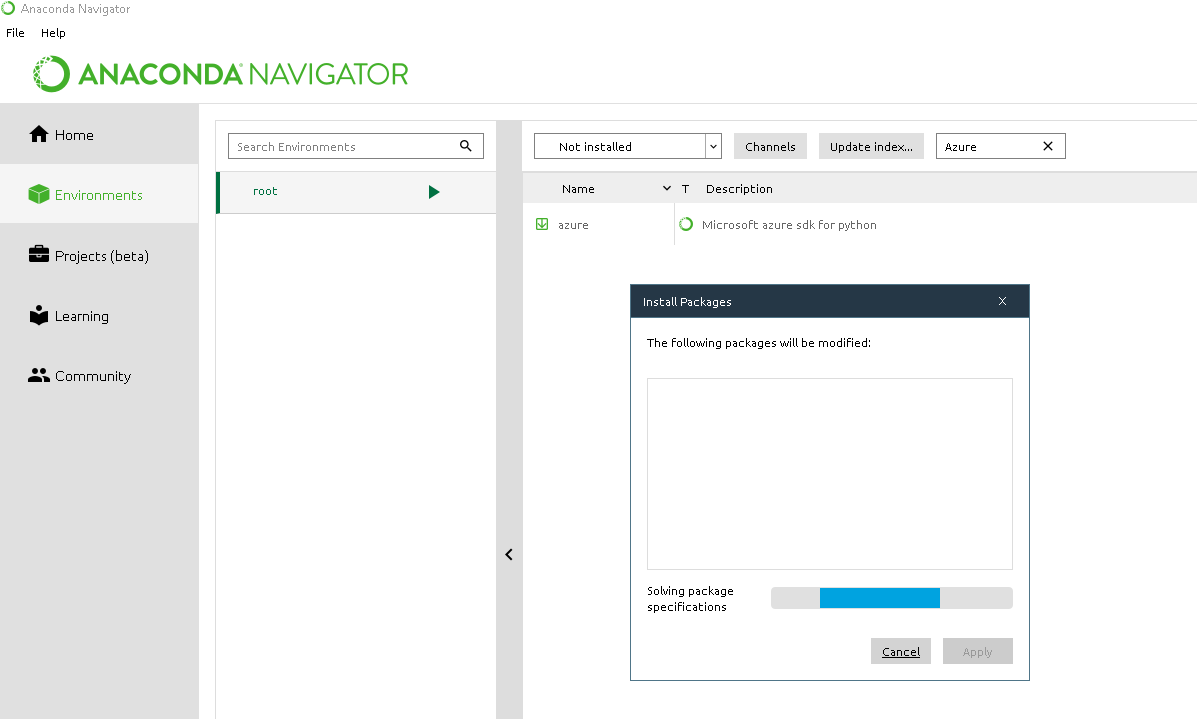








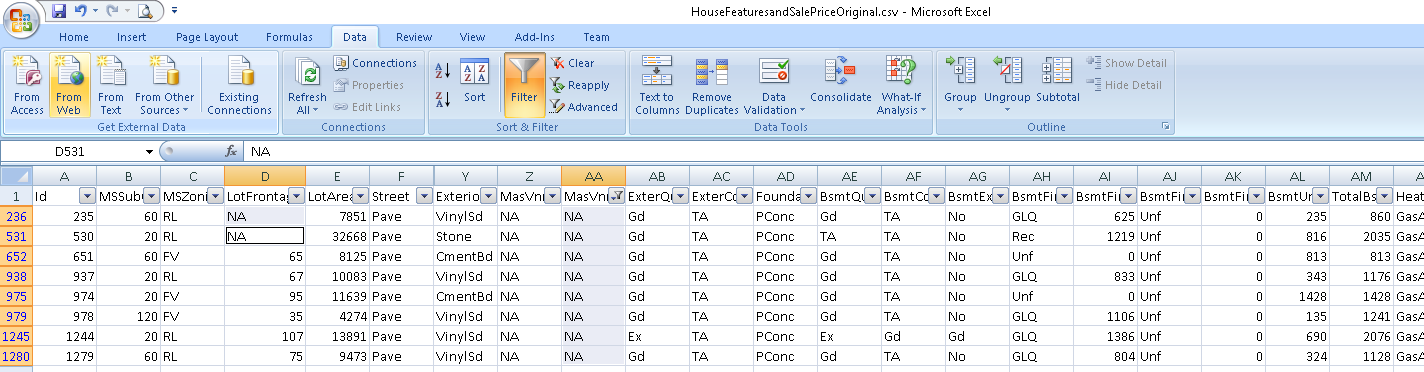
I imported the Azure library, required to run the Azure visualization code in Jupyter.

****

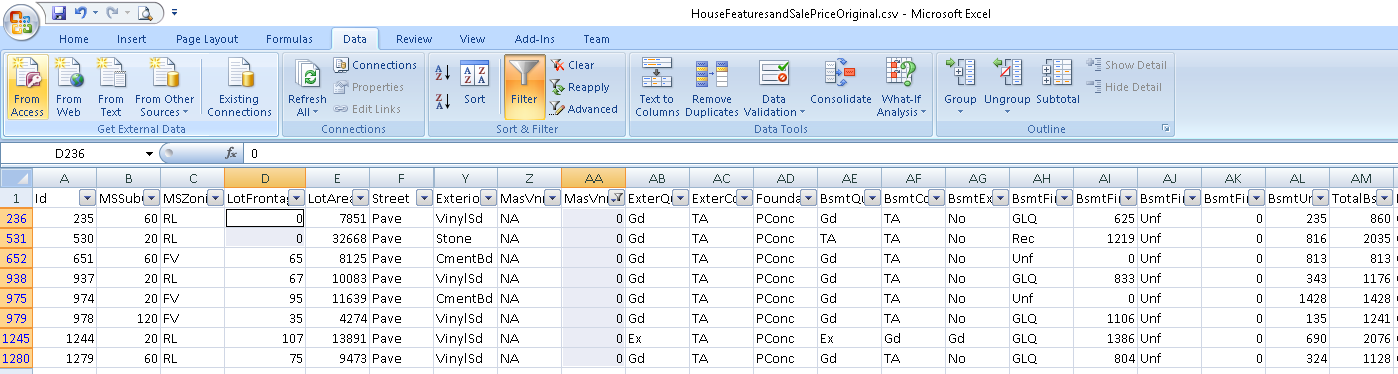
**Data (and cleansing):**

Columns LotFrontage and MasVnrArea are numeric. The creators of the data used the value 'NA' to denote no value. In these cases, I replaced NA with '0'. Cleansing is necessary to supply correct values to the machine learning algorithms.

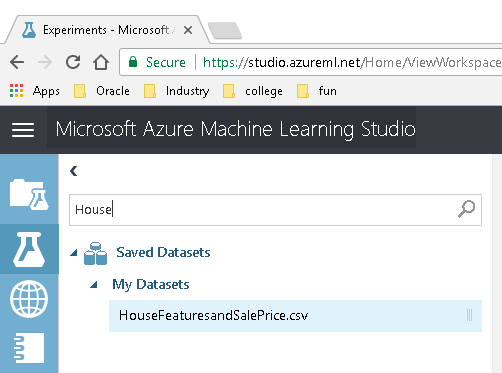
Before cleansing the data.



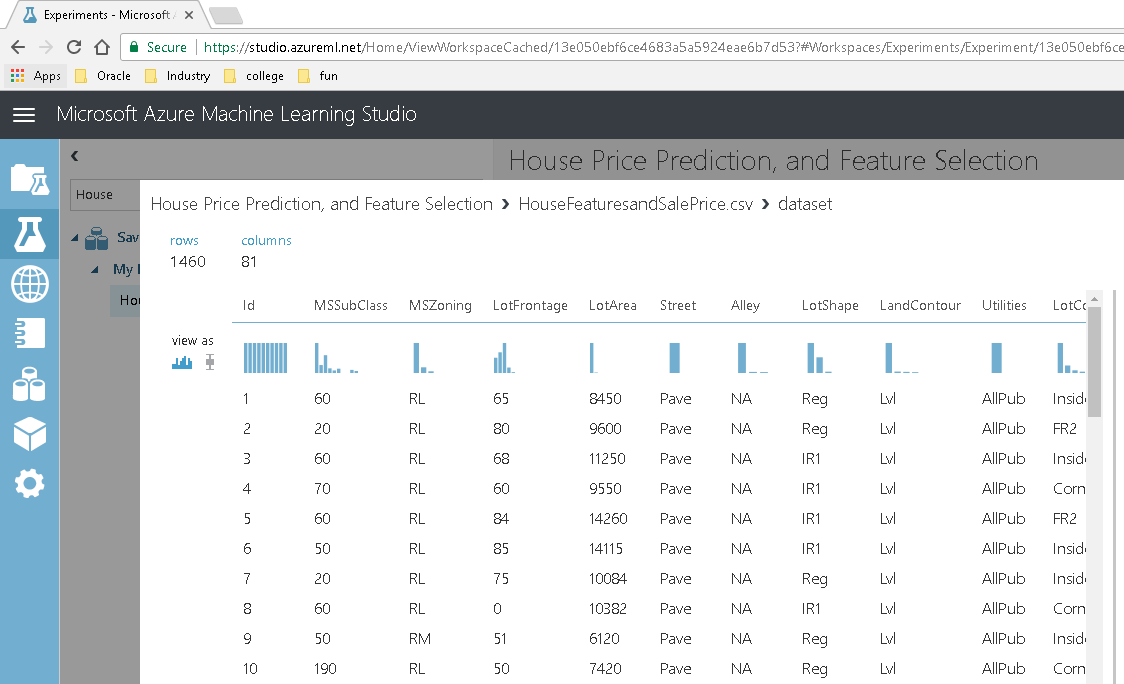
After replacing 'NA' with '0'.



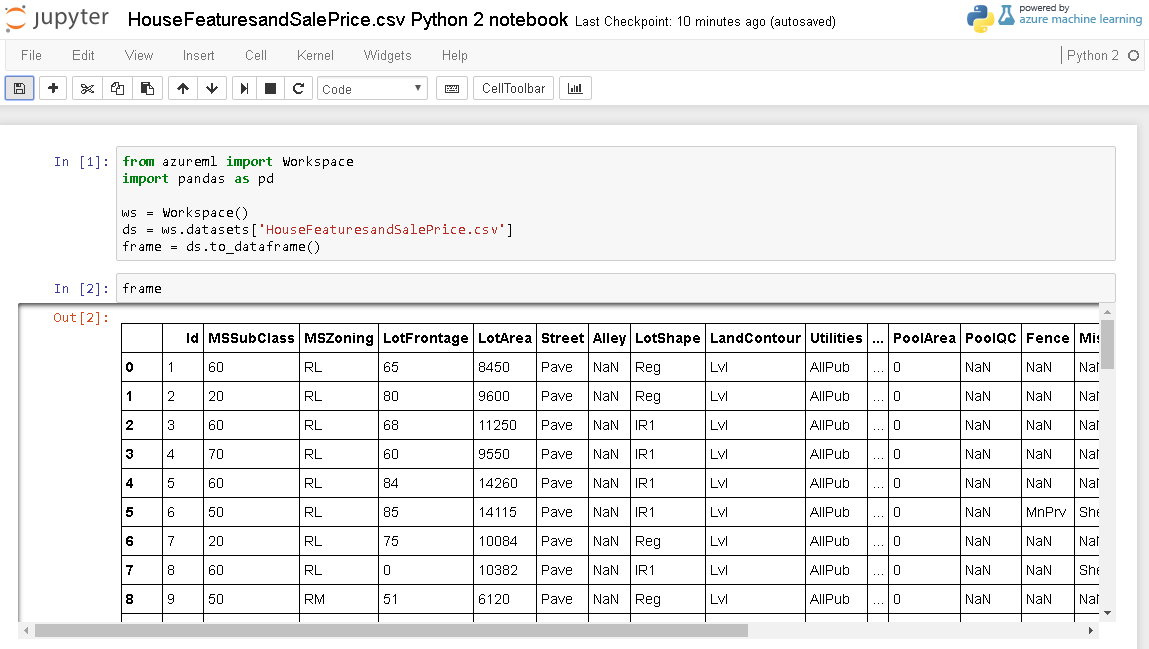
I uploaded the cleansed data to Azure Machine Learning Studio.



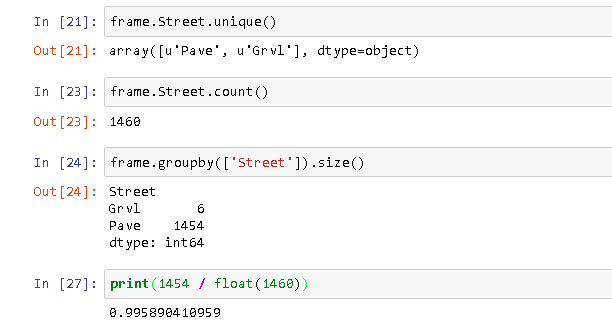
I created a new experiment, House Price Prediction, and Feature Selection, and added the dataset.



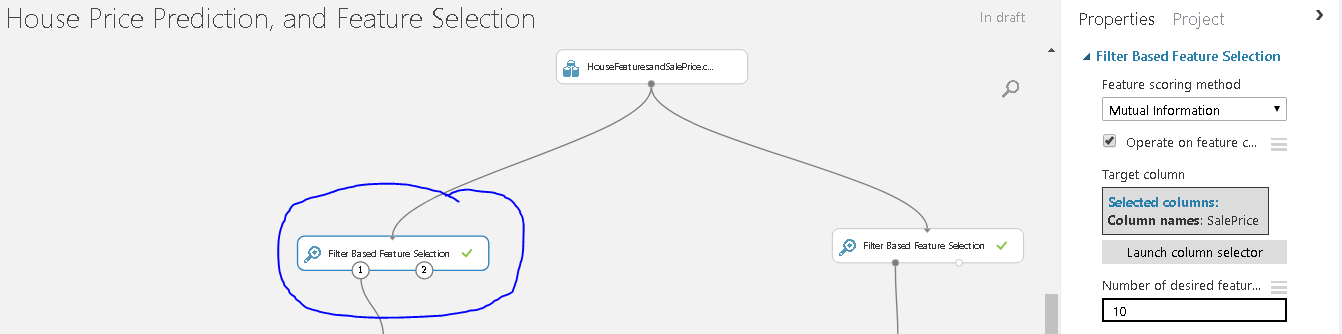
Azure Machine Learning Studio does not support the manipulation of experiments from a programming language. It does support accessing data that is the output of its various modules. I built the experiment using Studio's graphical interface but will visualize the data using Jupyter.



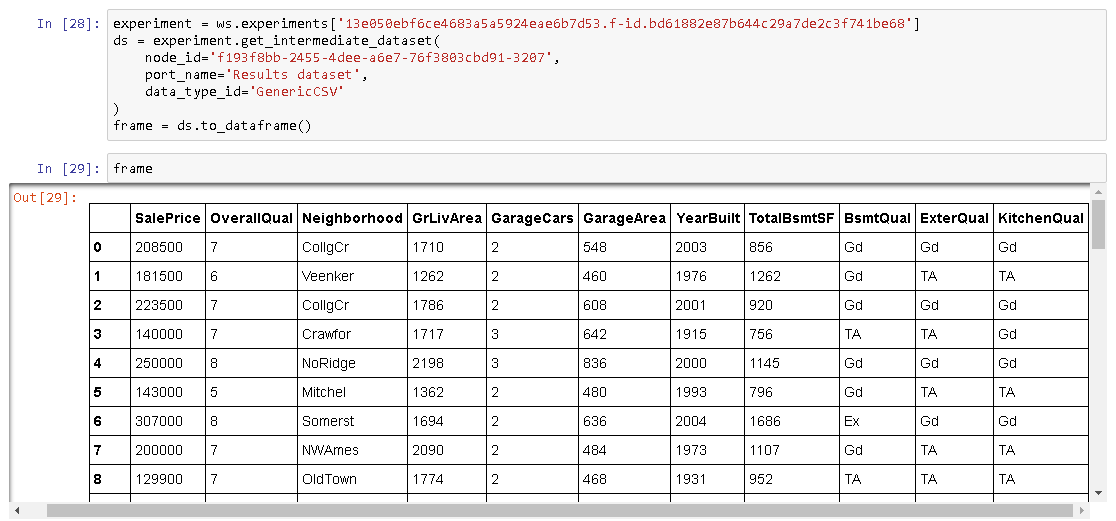
Not all 81 attributes will be useful to model SalesPrice. For example, 'Street' is 'Pave' for 1,454 entries and 'Grvl' for 6 entries. Given that 99.5% of the entries for 'Street' are the same value, it is doubtful that there is a correlation between this attribute and SalesPrice.



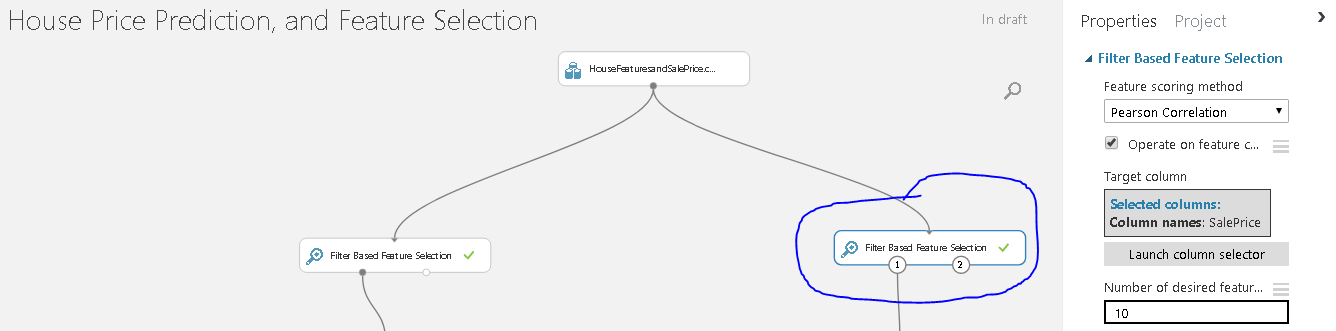
I decided to use the Filter Based Feature Selection module in Studio rather than identifying attributes on my own. On the left hand side, 'SalesPrice' was the target column, and I configured the module to identify the 10 most relevant attributes using the Mutual information scoring method.

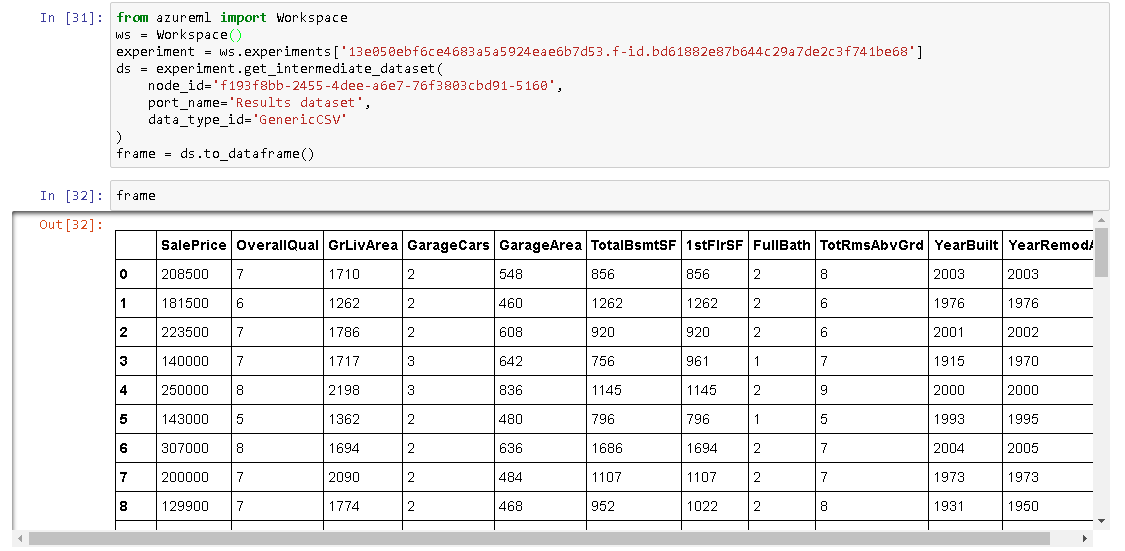


From left to right, in descending order, the Filter Based Feature Selection module selected these columns that are most relevant for predicting SalesPrice.



On the right hand side, 'SalesPrice' was the target column, and I configured the module to identify the 10 most relevant attributes using the Pearson correlation scoring method.

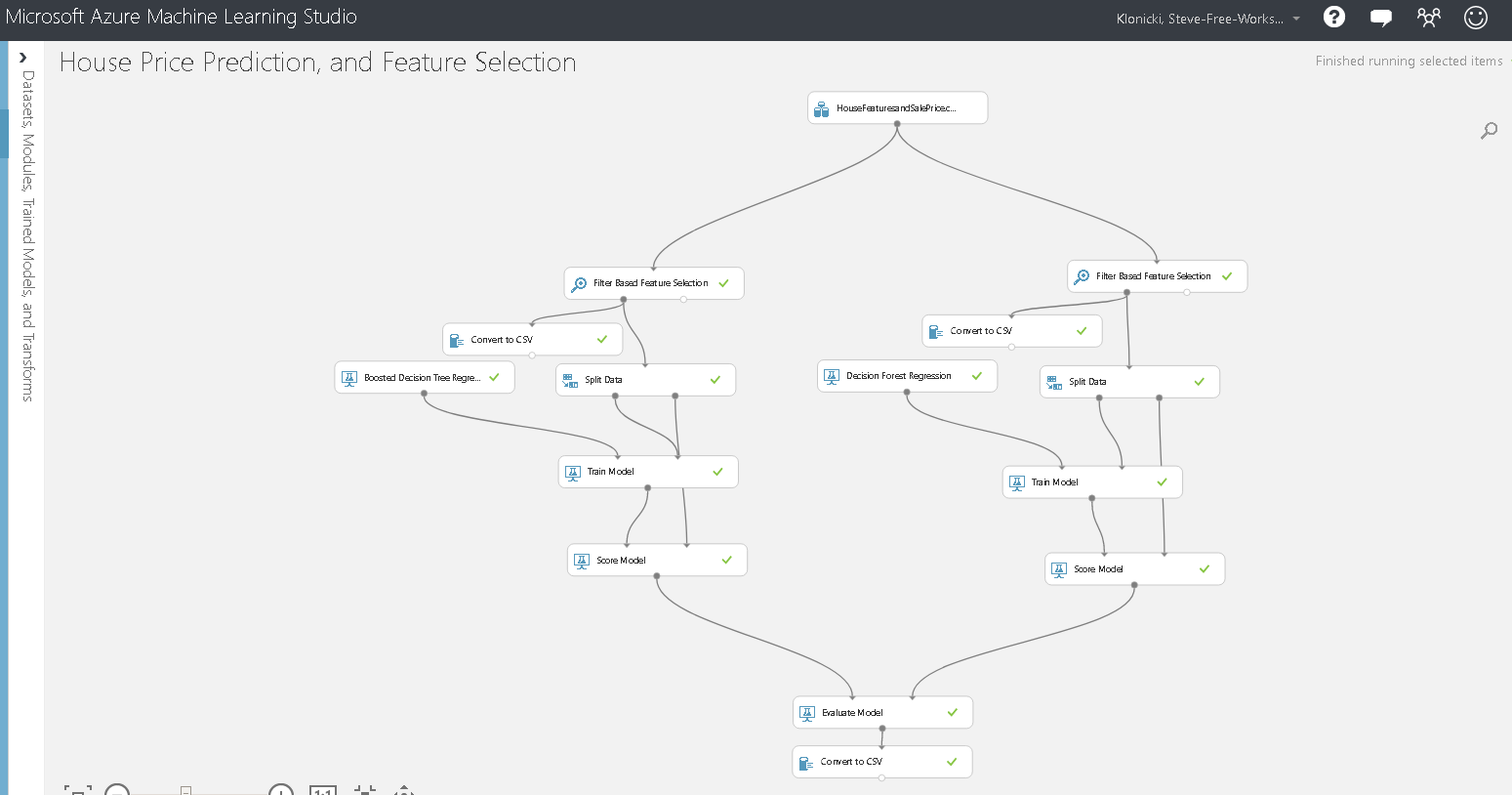




I observed how the 2 Feature Scoring methods came to different conclusions. Listed in descending order of importance

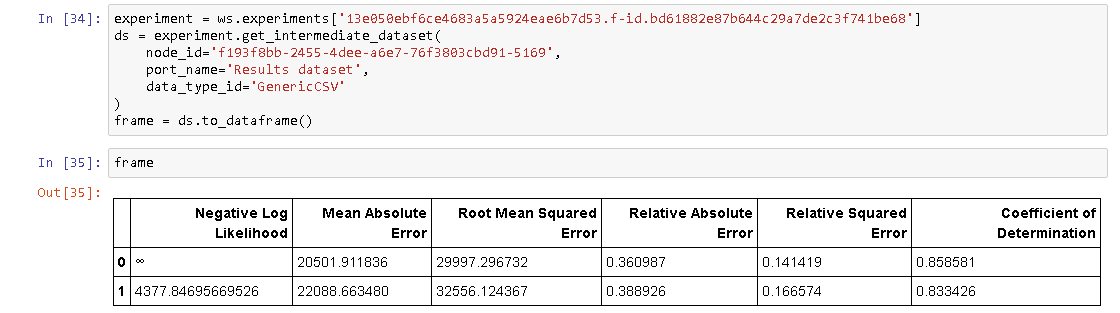
|  |  |
| --- | --- |
| Mutual Information | Pearson Correlation |
| Overall Quality | OverallQuality |
| Neighborhood | GrLivArea |
| GrLivArea | GarageCars |
| GarageCars | GarageArea |
| GarageArea | TotalBsmtSF |
| YearBuilt | 1stFlrSF |
| TotalBsmtSF | FullBath |
| BsmtQual | TotRmsAbvGrd |
| ExterQual | YearBuilt |
| KitchenQual | YearRemodled |

This is my Azure Machine Learning Studio experiment.



After loading the data, and using the Filter Based Feature Selection module to select the features to be used by the Regression algorithms. The input contained SalesPrice for all rows. For the 2 regression algorithms to be evaluated, I used the Split Data module to use 75% of the data to train the algorithm. The remaining 25% of the data was used to score the algorithm, i.e. determine SalesPrice using the trained algorithm. Finally, I evaluated the results of the 2 algorithms.

In a comparison of the 2 algorithms Boosted Decision Tree Regression outperformed Decision Forest Regression. This was determined by their Coefficient of Determination 0.858581 to 0.833426 respectively. In regression, the R2**coefficient of determination** is a statistical measure of how well the regression line approximates the real data points. An R2 of 1 indicates that the regression line perfectly fits the data. (cite https://en.wikipedia.org/wiki/Coefficient\_of\_determination)



**Code:**

# coding: utf-8

# Load housing data.

from azureml import Workspace

import pandas as pd

ws = Workspace()

ds = ws.datasets['HouseFeaturesandSalePrice.csv']

frame = ds.to\_dataframe()

# Print the housing data

frame

# Identify unique values for the Street attribute

frame.Street.unique()

# Print the number of rows with each unique value for the attribute Street

frame.groupby(['Street']).size()

# Calculate percentage of values that equal 'Pave'

print(1454 / float(1460))

# Print the results from the Filter Based Feature Selection, Mutual Information

experiment = ws.experiments['13e050ebf6ce4683a5a5924eae6b7d53.f-id.bd61882e87b644c29a7de2c3f741be68']

ds = experiment.get\_intermediate\_dataset(

node\_id='f193f8bb-2455-4dee-a6e7-76f3803cbd91-3207',

port\_name='Results dataset',

data\_type\_id='GenericCSV'

)

frame = ds.to\_dataframe()

# Print the results from the Filter Based Feature Selection, Pearson Correlation

frame

# Print the results from the Filter Based Feature Selection, Pearson Correlation

from azureml import Workspace

ws = Workspace()

experiment = ws.experiments['13e050ebf6ce4683a5a5924eae6b7d53.f-id.bd61882e87b644c29a7de2c3f741be68']

ds = experiment.get\_intermediate\_dataset(

node\_id='f193f8bb-2455-4dee-a6e7-76f3803cbd91-5160',

port\_name='Results dataset',

data\_type\_id='GenericCSV'

)

frame = ds.to\_dataframe()

# In[32]:

frame

# Print the results from the Evaluate Model module

experiment = ws.experiments['13e050ebf6ce4683a5a5924eae6b7d53.f-id.bd61882e87b644c29a7de2c3f741be68']

ds = experiment.get\_intermediate\_dataset(

node\_id='f193f8bb-2455-4dee-a6e7-76f3803cbd91-5169',

port\_name='Results dataset',

data\_type\_id='GenericCSV'

)

frame = ds.to\_dataframe()

frame

**Summary:**

I concluded that Boosted Decision Tree Regression outperformed Decision Forest Regression.

With Microsoft Azure Machine Learning Studio, I was able to quickly generate an experiment that, in theory, could predict the price of a house with 85% accuracy. Negatives are the limited ways Azure Machine Learning can be used programmatically. It's possible to view the results of a module through Python, but it is not possible to configure or execute the module through a programming language.

If I were to continue, I would look to improve on 85% accuracy. I would research and employ additional statistical techniques such as removing outliers.

**YouTube Links:**

2 Min: https://youtu.be/kb44rs7QztQ

15 Min: https://youtu.be/GQZ3NAmyNxA

**GitHub:**

https://github.com/we814/AMLDeepDive

**References:**

Microsoft Azure Machine Learning Studio (<https://studio.azureml.net/>)

https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data

https://en.wikipedia.org/wiki/Coefficient\_of\_determination