Data Science and Machine Learning Project - House Prices Dataset

Exploratory Data Analysis

+ Understand the problem
+ Explore the data and deal with missing values
+ Select and transform variables, especially categorical ones

+ With 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, this competition challenges you to predict the final price of each home.

+ we are going to explore the dataset, try to get some insights from it, and use some tools to transform the data into formats that make more sense

Importing Libraries

+ The standard math module provides access to the mathematical functions.
+ The NumPy lib is fundamental for any kind of scientific computing with Python.
+ pandas is a must-have tool for data analysis and manipulation.
+ matplotlib is the most complete package in Python when it comes to data visualizations.
+ seaborn is based on matplotlib as a higher-level set of visualization tools, not as powerful as matplotlib, but much easier to work with and delivers a lot with less work.

Loading Data

+ since the format is CSV (Comma-Separated Values), we use the read_csv() function from pandas.
+ we have 1169 rows in the CSV file, but the header that describes the columns doesn’t count.

Looking at the Data

+ Using the head() function from pandas with an argument of 3, we can take a look at the first 3 records.
+ The .T means Transpose, this way we visualize rows as columns and vice-versa.
+ train.head(3).T

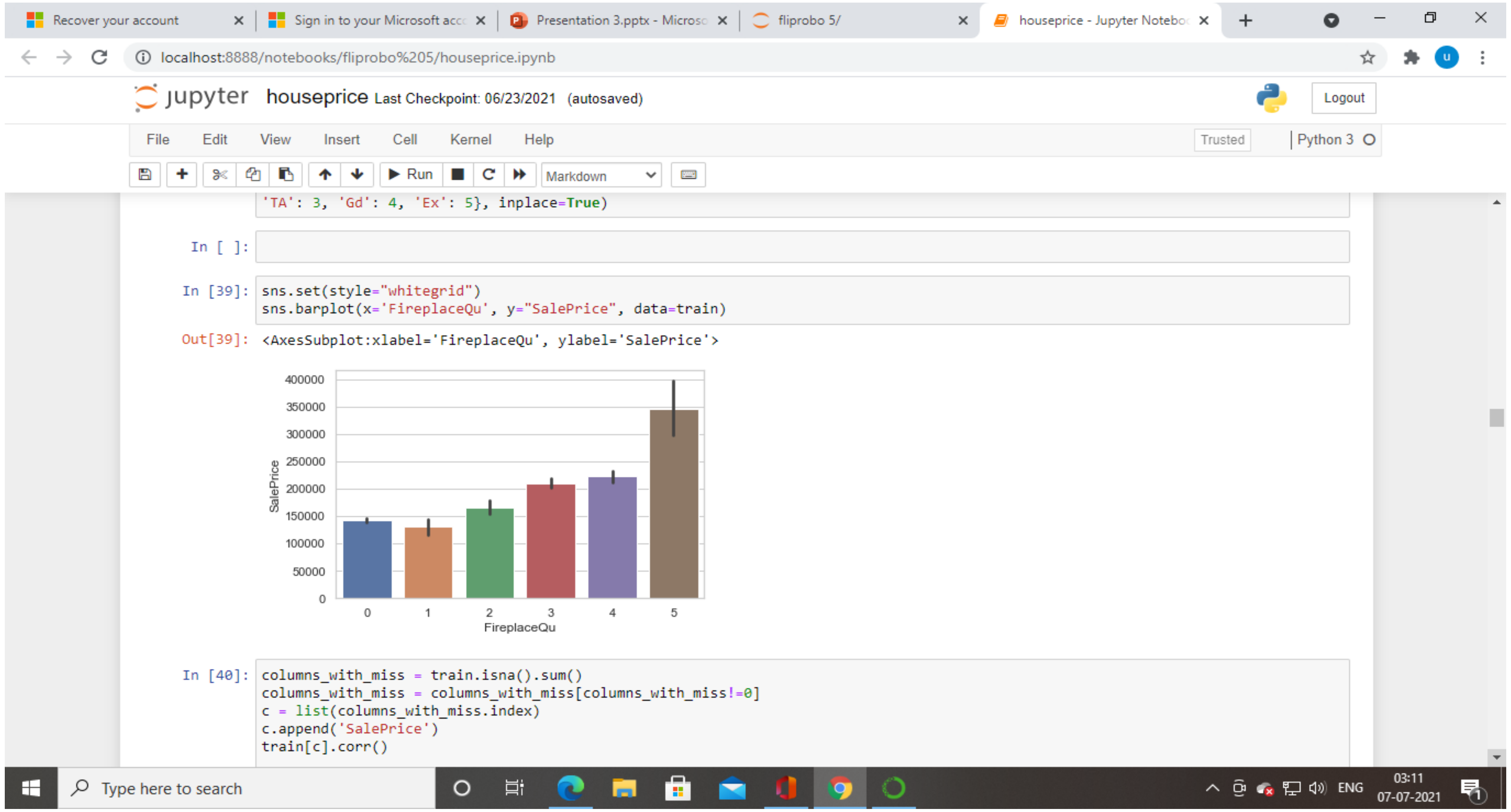
- The describe() method is good to have the first insights of the data.
- It automatically gives you descriptive statistics for each feature: number of non-NA/null observations, mean, standard deviation, the min value, the quartiles, and the max value.
- Note that the calculations don’t take NaN values into consideration.

Data Cleaning

- The id column The id column is only a dumb identification with no correlation to SalePrice.
- So let’s remove the id:
- train.drop(columns=['Id'], inplace=True)

Missing values

- When we used info() to see the data summary, we could see many columns had a bunch of missing data.
- isna() from pandas will return the missing values for each column, then the sum() function will add them up to give you a total.

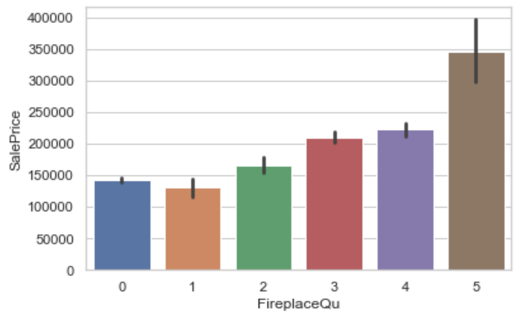

Recover your account
Sign in to your Microsoft acc
Presentation 3.pptx - Microsc
fliprobo 5/
houseprice - Jupyter Notebo
localhost:8888/notebooks/fliprobo%205/houseprice.ipynb
jupyter houseprice Last Checkpoint: 06/23/2021 (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Markdown
Type here to search
ENG
03:11
07-07-2021

```
'TA': 3, 'Gd': 4, 'Ex': 5}, inplace=True)
```

```
In [ ]:
```

```
In [39]: sns.set(style="whitegrid")
         sns.barplot(x='FireplaceQu', y="SalePrice", data=train)
```

```
Out[39]: <AxesSubplot:xlabel='FireplaceQu', ylabel='SalePrice'>
```



```
In [40]: columns_with_miss = train.isna().sum()
         columns_with_miss = columns_with_miss[columns_with_miss!=0]
         c = list(columns_with_miss.index)
         c.append('SalePrice')
         train[c].corr()
```

- we can see how the category of the FirePlace increases the value of SalePrice.
- It is also worth noting how much higher the value is when the house has an Excellent fireplace.
- This means we should keep FireplaceQu as feature.

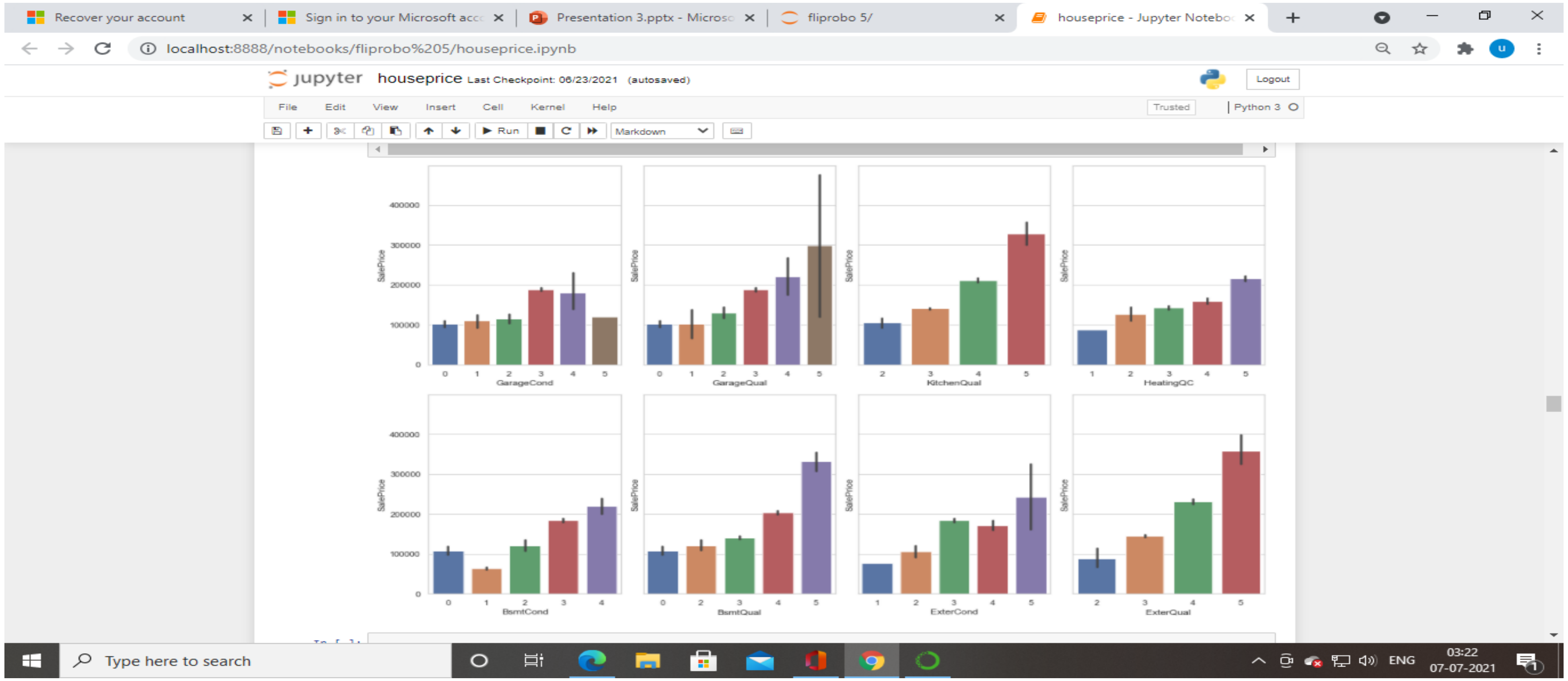
Missing values in numeric columns

- Another feature with a high number of missing values is LotFrontage with a count 204.
- Columns with missing values: 11

Categorical variables

+ Dealing with missing values
+ # Fills NA in place of NaN for c in ['GarageType', 'GarageFinish', 'BsmtFinType2', \ 'BsmtExposure', 'BsmtFinType1']: train[c].fillna('NA', inplace=True) # Fills None in place of NaN train['MasVnrType'].fillna('None', inplace=True)

Ordinal


Recover your account
Sign in to your Microsoft acc
Presentation 3.pptx - Microso
fliprobo 5/
houseprice - Jupyter Notebo
localhost:8888/notebooks/fliprobo%205/houseprice.ipynb
jupyter houseprice Last Checkpoint: 06/23/2021 (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Markdown
SalePrice
GarageCond
GarageQual
KitchenQual
HeatingQC
BsmtCond
BsmtQual
ExterCond
ExterQual
Type here to search
ENG
03:22
07-07-2021

Zero values

+ Another quick check is to see how many columns have lots of data equals to 0.
+ train.isin([0]).sum().sort_values(ascending=False).head(25)
+ , even though there are many 0's, they have meaning.
+ For instance, PoolArea (Pool area in square feet) equals 0 means that the house doesn't have any pool area.
+ This is important information correlated to the house and thus, we are going to keep them

Outliers

- There are a lot of outliers in the dataset. But, if we check the data description file, we see that, actually, some numerical variables, are categorical variables that were saved (codified) as numbers. So, some of these data points that seem to be outliers are, actually, categorical data with only one example of some category. Let’s keep these outliers

Saving cleaned data

+ We have no more missing values:

+
  ```
  columns_with_miss = train.isna().sum()
  columns_with_miss =
  columns_with_miss[columns_with_miss!=0]
  print(f'Columns with missing values:
  {len(columns_with_miss)}')
  columns_with_miss.sort_values(ascending=False)
  ```

+ After cleaning the data, we are left with 73 columns out of the initial 81. (1168, 73) Let's take a look at the first 3 records of the cleaned data.

Conclusions

- We dealt with missing values and removed the following columns: ‘Id’, ‘PoolQC’, ‘MiscFeature’, ‘Alley’, ‘Fence’, ‘LotFrontage’, ‘GarageYrBlt’, ‘MasVnrArea’.
- We also:
- Replaced the NaN with NA in the following columns: ‘GarageType’, ‘GarageFinish’, ‘BsmtFinType2’, ‘BsmtExposure’, ‘BsmtFinType1’. Replaced the NaN with None in ‘MasVnrType’. Imputed the most frequent value in place of NaN in ‘Electrical’.

Data Cleaning Script

+ the final decisions made to clean the data in the Exploratory Data Analysis into a single Python script that will take the data in CSV format and write the cleaned data also as a CSV.

+ Code
+ You can save the script on a file ‘data_cleaning.py’ and execute it directly with python3 data_cleaning.py or python data_cleaning.py, depending on your installation.
+ The script expects the ‘raw_data.csv’.
+ The output will be a file named ‘cleaned_data.csv’.
+ It will also print the shape of the original data and the shape of the new cleaned data
+ Original Data: (1168, 81)
+ After Cleaning: (1168, 73)

Machine Learning Model

+ Now we are going to use the 'cleaned_data.csv' file generated with the data cleaning script to generate the Machine Learning Model.

Training the Machine Learning Model

- You can save the script on a file train_model.py and execute it directly with python3 train_model.py or python train_model.py, depending on your installation.
- It expects you to have a file called ‘cleaned_data.csv’.

+ The script will output three other files:
+ model.pkl: the model in binary format generated by pickle that we can reuse later train.csv:
+ the train data after the split of the original data into train and test test.csv:
+ the test data after the split of the original data into train and test

+ The output on the terminal will be similar to this:
+ Train data for modeling: (934, 74)
+ Test data for predictions: (234, 74)
+ Training the model ...
+ Testing the model ...
+ Average Price Test: 175652.0128205128
+ RMSE: 10552.188828855931
+ Model saved at model.pkl
+ It means the models used 934 data point to train and 234 data points to test.

+ The average Sale Price in the test set is 175k dollars.
+ The RMSE (root-mean-square error) is a good metric to understand the output because in you can read it using the same scale of you dependent variable, which is Sale Price in this case.
+ A RMSE of 10552 means that, on average, we missed the correct Sale Prices by a bit over 10k dollars.
+ Considering an average if 175k, missing the mark, on average, by 10k, is not too bad.