Feature Engineering and Data Preparation

Feature Engineering

- In the real world, not every data set is machine learning ready, we often need to perform data cleaning or try to produce more usable features.
- In this section, we'll work on the large linear regression data set to get it ready for a machine learning project.

Feature Engineering

- What is feature engineering?
  - Is the process of using domain knowledge to **extract features** from raw data via data mining techniques.
- But what does this actually entail?

Feature Engineering

- Three general approaches:
  - Extracting Information
  - Combining Information
  - Transforming Information

Feature Engineering

- Extracting Information
  - Imagine a dataset with visitor expenditure information for a bar.
  - We have a timestamp for each row:
    - 1990-12-01 09:26:03
  - In its current format, its very difficult to pass into a machine learning algorithm.

Feature Engineering

- Extracting Information
  - In its current format, its very difficult to pass into a machine learning algorithm.
  - There is no coefficient we can apply for a non-numeric data point:
    - 1990-12-01 09:26:03
  - In general for most algorithms we need to make sure features are float or int.

Feature Engineering

- Extracting Information
  - Instead we extract information
    - 1990-12-01 09:26:03
      - Year: 1990
      - Month: 12
      - Weekday or Weekend (0/1)
      - Mon:1,Tues:2, ... Sun:7

Feature Engineering

- Extracting Information
  - More complex examples:
    - Text data for deed of house
      - Length of text
      - Number of times certain terms are mentioned

Feature Engineering

- Combining Information
  - We’ve actually already done this with Polynomial Regression!
  - Recall advertising spend could have possible interaction terms to consider, so we could multiply them together.

Feature Engineering

- Combining Information
  - Could also combine extracted information:
    - New Feature:
      - 0 or 1 value indicating:
        - Both weekend and evening?

Feature Engineering

- Transforming Information
  - Very common for string data
  - Most algorithms can not accept string data (can’t multiply a string such as “red” by a numeric coefficient)

Feature Engineering

- Transforming Information
  - Often categorical data is presented as string data.
  - For example a large data set of social network users could have country of origin as a string feature (e.g. USA, UK, MEX, etc…)

Feature Engineering

- Transforming Information
  - We can use two approaches here:
    - Integer Encoding
    - One-hot Encoding (Dummy Variables)

Feature Engineering

- Integer Encoding
  - Directly convert categories into integers 1,2,3…N

Feature Engineering

- Integer Encoding
    - Directly convert categories into integers 1,2,3…N

Country
USA
MEX
CAN
USA

Feature Engineering

- Integer Encoding
  - Directly convert categories into integers 1,2,3…N

Country
USA
MEX
CAN
USA

Country
1
2
3
1

Feature Engineering

- Integer Encoding
  - Possible issue is implied ordering and relationship (ordinal variable)

Country
USA
MEX
CAN
USA

$\Rightarrow$

Country
1
2
3
1

Feature Engineering

- Integer Encoding
    - Here we see the implication MEX is twice the value of USA

Country
USA
MEX
CAN
USA

Country
1
2
3
1

Feature Engineering

- Integer Encoding
    - Here we see the implication CAN is three times the value of USA

Country
USA
MEX
CAN
USA

Country
1
2
3
1

Feature Engineering

- Integer Encoding
    - This may or may not make sense depending on the feature and domain

Country
USA
MEX
CAN
USA

$\Rightarrow$

Country
1
2
3
1

Feature Engineering

- Integer Encoding
    - This may or may not make sense depending on the feature and domain

Spice Level
Mild
Hot
Fire
Mild

Spice Level
1
2
3
1

Feature Engineering

- Integer Encoding
  - Always carefully consider the implication of integer encoding

Spice Level
Mild
Hot
Fire
Mild

Spice Level
1
2
3
1

Feature Engineering

- Integer Encoding
  - Pros:
    - Very easy to do and understand.
    - Does not increase number of features.
  - Cons:
    - Implies ordered relationship between categories.

Feature Engineering

- One Hot Encoding (Dummy Variables)
    - Convert each category into individual features that are either 0 or 1

Feature Engineering

- One Hot Encoding (Dummy Variables)
    - Convert categories into individual features that are either 0 or 1

Country
USA
MEX
CAN
USA

Feature Engineering

- One Hot Encoding (Dummy Variables)
    - Convert categories into individual features that are either 0 or 1

Country
USA
MEX
CAN
USA

USA	MEX	CAN
1	0	0
0	1	0
0	0	1
1	0	0

Feature Engineering

- One Hot Encoding (Dummy Variables)
  - No ordered relationship is implied between categories.

Country
USA
MEX
CAN
USA

USA	MEX	CAN
1	0	0
0	1	0
0	0	1
1	0	0

Feature Engineering

- One Hot Encoding (Dummy Variables)
    - However we greatly expanded our feature set, many more columns.

Country
USA
MEX
CAN
USA

USA	MEX	CAN
1	0	0
0	1	0
0	0	1
1	0	0

Feature Engineering

- One Hot Encoding (Dummy Variables)
  - We can try to reduce this feature column expansion by creating higher level categories.
  - For example, regions or continents instead of countries.

Feature Engineering

- One Hot Encoding (Dummy Variables)
  - Using pandas .map() or .apply() can achieve this.
  - May require a lot of tuning and domain experience to choose reasonable higher level categories or mappings.

Feature Engineering

- One Hot Encoding (Dummy Variables)
    - Also must be aware of the “dummy variable trap”, mathematically known as multi-collinearity.
    - Converting to dummy variables can cause features to be duplicated.
    - Let’s consider the simplest possible example...

Feature Engineering

- One Hot Encoding (Dummy Variables)
  - Consider a binary category (only two options):

Vertical Direction
UP
DOWN
UP
DOWN

Feature Engineering

- One Hot Encoding (Dummy Variables)
  - Consider a binary category (only two options):

Vertical Direction
UP
DOWN
UP
DOWN

$\rightarrow$

UP	DOWN
1	0
0	1
1	0
0	1

Feature Engineering

- One Hot Encoding (Dummy Variables)
  - The new columns are duplicate information with inverted encoding.

Vertical Direction
UP
DOWN
UP
DOWN

UP	DOWN
1	0
0	1
1	0
0	1

Feature Engineering

- One Hot Encoding (Dummy Variables)
  - Easily fixed by simply dropping last column.

Vertical Direction
UP
DOWN
UP
DOWN

UP
1
0
1
0

Feature Engineering

- One Hot Encoding (Dummy Variables)
  - This can be extended to more than 2 categories:

Country
USA
MEX
CAN
USA

→

USA	MEX
1	0
0	1
0	0
1	0

Feature Engineering

- One Hot Encoding (Dummy Variables)
  - Pros:
    - No ordering implied.
  - Cons:
    - Potential to create many more feature columns and coefficients.
    - Dummy variable trap consideration.
    - Not easy to add new categories.

Feature Engineering

- Throughout this section of the course we'll work on addressing the following issues:
    - Outliers in Data
    - Missing Data
    - Categorical Data
- Not every issue here is strictly "feature engineering", but could also be called "data cleaning".

Feature Engineering

- Keep in mind feature engineering in general will always be data and domain dependent.
- There is no one size fits all solution!

Dealing with Outliers

Outliers

- Often a data set will have a few points that are extreme outliers.
- It’s often better to simply remove these few points from the data set in order to have a more generalized model.

Outliers

- Outlier Considerations
    - Definition of an Outlier
        - Range and Limits
        - Percentage of Data
- These are both very domain dependant!

Outliers

- Outlier Considerations
    - Range and Limits
        - We need to decide what will constitute an outlier with some methodology:
            - InterQuartile Range
            - Standard Deviation
            - Visualized or Domain Limit Value

Outliers

- Outlier Considerations
  - Percentage of Data
    - Keep in mind if a large percentage of your data is being labeled as an outlier, then you actually just have a wide distribution, not outliers!
    - Limit outliers to a few percentage points a most.

Outliers

- Outlier Considerations
    - Utilize visualization plots to be able to see and identify outlier points.
    - Keep in mind, this will create caveats for your future model (e.g. Model not suitable for houses priced over $10 Million)

Outliers

- Keep in mind, there is no 100% correct outlier methodology that will apply to every situation.
- Let’s explore the Ames Data Set for outliers!

Dealing with Missing Data

PART ONE: EVALUATING WHAT IS MISSING

Missing Data

- Make sure you've viewed the "Missing Data" lecture in the pandas section **before** continuing with this series of lectures!
- Many concepts and methods referred to here were explained in those lectures.

Missing Data

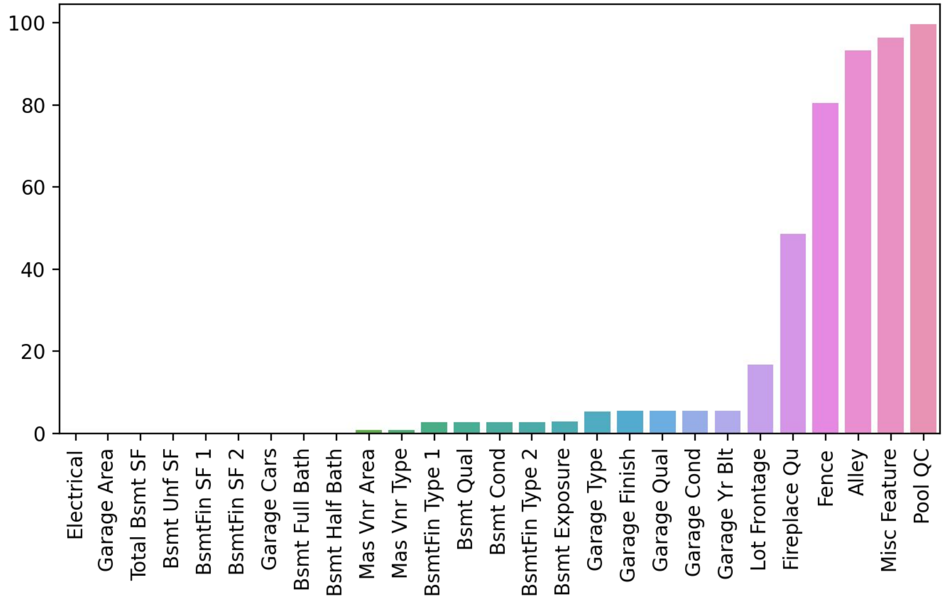
- Working with the Ames data set, in Part One we will focus on evaluating just how much data is missing.

Dealing with Missing Data

PART TWO: FILLING DATA FOR ROWS

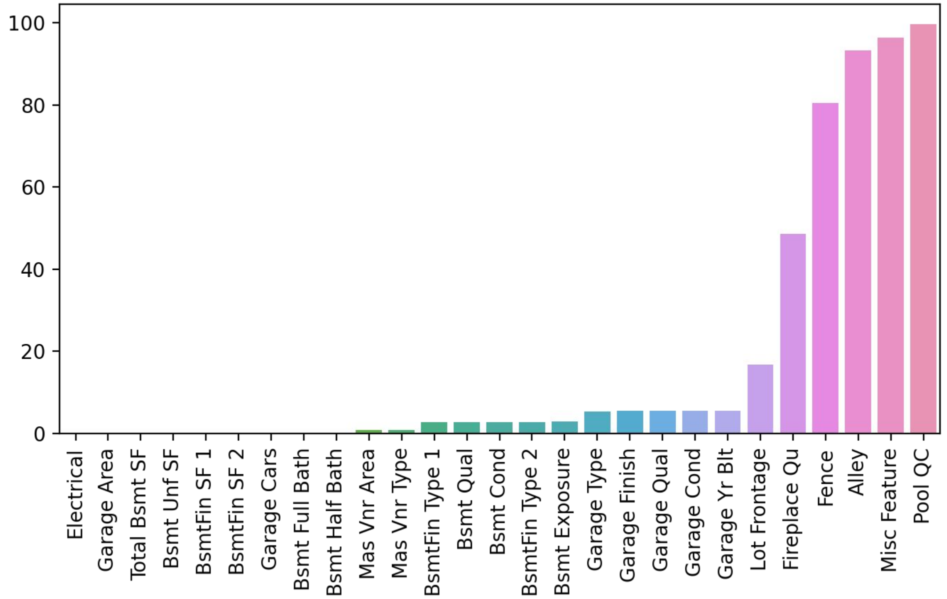
Missing Data

- Recall we just calculated percentage of data missing per feature column:



Missing Data

- Let’s first work on considering features that have a very small percent missing.



Missing Data

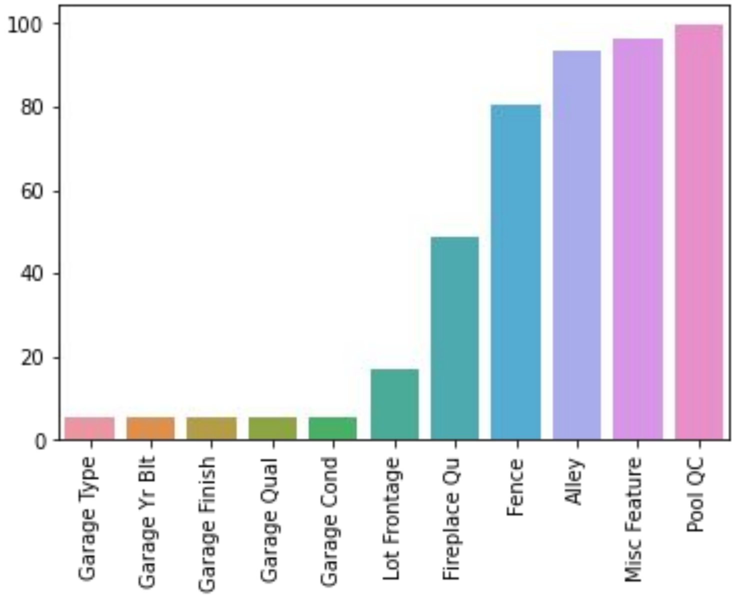
- In the case of just a few rows missing the feature data, we’ll consider either dropping these few rows or filling in with a reasonable assumption based off domain knowledge.
- Let’s jump to the notebook to explore our options!

Dealing with Missing Data

PART THREE: FEATURE COLUMNS

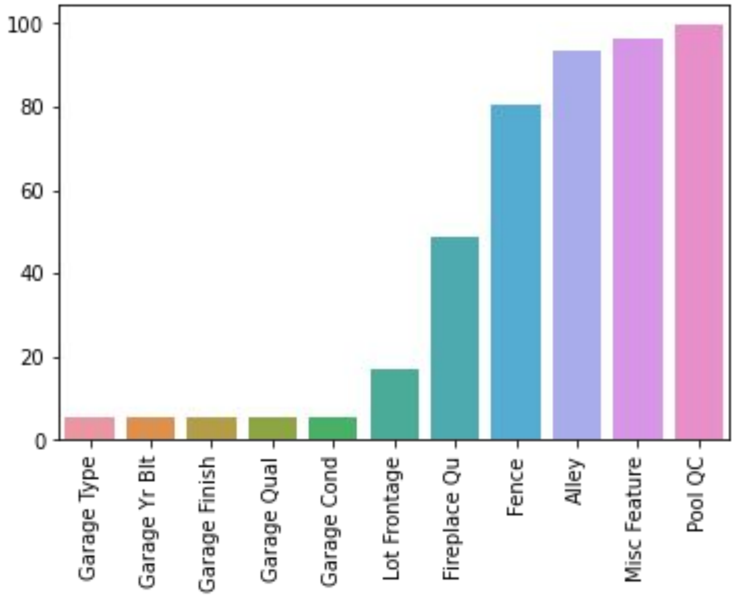
Missing Data

- We are now dealing with missing data that goes beyond our 1% threshold.



Missing Data

- In other words, more than 1% of rows are missing some of these feature values.



Missing Data

- Two main approaches here:
    - Fill in the missing values
    - Drop the feature column
- Let’s consider the pros and cons of each approach…

Missing Data

- Dropping the feature column:
  - Very simple to do.
  - No longer need to worry about that feature in the future.
  - Potential to lose a feature with possible important signal.
  - Should consider drop feature approach when many rows are NaN.

Missing Data

- Filling in the missing feature data:
  - Potentially changing ground truth in data.
  - Must decide on reasonable estimation to filled value.
  - Must apply transformation to all future data for predictions.

Missing Data

- Filling in the missing feature data:
  - Simplest case:
    - Replace all NaN values with a reasonable assumption (e.g. zero if assumed NaN implied zero)
  - Harder cases:
    - Must use statistical methods based on other columns to fill in NaN values.

Missing Data

- Filling in the missing feature data:
    - Statistical Estimation:
        - Dataset about people with some age data missing.
        - Could use current career/education status to fill in data (e.g. people currently in college fill in with 20 yrs)

Missing Data

- Let's explore both approaches!
    - *Important note!*
        - *Realistically on the Ames data set, many NaN values are probably actually correctly "zero". But we want to show the methodology for multiple approaches!*

Dealing with Categorical Data

Categorical Data

- We're going to jump straight to the transformation of the data, but for a detailed intuition on dummy variables and one hot encoding refer the introduction!