SEAS 6414

Spring 2024

Assignment 4, Michael Wacey

Dr. Adewale Akinfaderin

I probably spent too much time on this assignment this week. You may not see that in the results. I actually enjoyed much of it and learned a lot. In many cases, I just printed the dataframe rather than using head. This provided the first five rows and last five rows. That seemed useful to me. Some of the data is very wide. So, I will upload a text file along with this. It is easier to see the wide data on a text file.

Let me know if you want to see the source file. This is the executed file and has everything in it. But I am happy to share the source file. This assignment is in GitHub at <https://github.com/OwlSaver/GWU>.

# Execution

#######################################################################################

# Problem 1

#######################################################################################

Problem:

Dataset: homework4 file1.csv

Data Description: The dataset contains records of merchant transactions, each

with a unique merchant identifier, time of transaction, and amount in cents.

Objective: Analyze merchant transaction data to understand business growth and

health. Preprocess the dataset for future merchant transactions and generate specific

features for each merchant.

Task: Generate the following features for each unique merchant:

- trans amount avg: Average transaction amount for each merchant.

- trans amount volume: Total transaction amount for each merchant.

- trans frequency: Total count of transactions for each merchant.

- trans recency: Recency of the last transaction (in days from 1/1/2035).

- avg time btwn trans: Average time between transactions (in hours).

- avg trans growth rate: Average growth rate in transaction amounts.

Data Dimension: The dataset is N by 3, where N is the number of records.

Final Deliverables:

- Shape of the new dataset.

- The top five rows of the new dataset using new dataset.head().

- Descriptive statistics of the new dataset.

Code:

import pandas as pd

import numpy as np

import datetime as dt

pd.options.display.float\_format = '{:,.2f}'.format

pd.set\_option('display.max\_columns', None)

pd.set\_option('display.width', 2000)

HW4F1 = pd.read\_csv('./gwu/SEAS 6414/homework4\_file1.csv')

# Make the time column a Pandas time rather than a string

HW4F1['time'] = [pd.Timestamp(ts) for ts in HW4F1.time]

HW4F1.sort\_values(by=['merchant', 'time'], inplace=True)

HW4F1New = HW4F1.groupby("merchant").agg(

min\_amount=("amount\_usd\_in\_cents", "min")

, max\_amount=("amount\_usd\_in\_cents", "max")

, trans\_amount\_avg=("amount\_usd\_in\_cents", "mean")

, trans\_amount\_volume=("amount\_usd\_in\_cents", "sum")

, trans\_frequency=("amount\_usd\_in\_cents", "count")

, most\_recent\_date=("time", "max")

, avg\_time\_btwn\_trans=("time", lambda group: group.sort\_values().diff().mean().seconds/(60\*60))

, avg\_trans\_growth\_rate=("amount\_usd\_in\_cents", lambda group: group.sort\_values().pct\_change().mean())

)

# I tried to do this as a Lambda in the agg, but it would not recognize the dt library

# So, in the agg, I find the max and here I calculate the delta

HW4F1New['trans\_recency'] = (HW4F1New['most\_recent\_date'] - dt.datetime(2035, 1, 1)).dt.days

# getting rid of the no longer needed maximum value

HW4F1Final = HW4F1New.drop(columns=['most\_recent\_date'])

print(f"The shape of the original data frame is: {HW4F1.shape}")

print(f"The shape of the new data frame is: {HW4F1Final.shape}")

print("")

print("The top five rows are:")

print(HW4F1Final.head(5))

print("")

print("Descriptive statistics:")

print(HW4F1Final.describe())

Execution:

The shape of the original data frame is: (100000, 3)

The shape of the new data frame is: (7902, 8)

The top five rows are:

min\_amount max\_amount trans\_amount\_avg trans\_amount\_volume trans\_frequency avg\_time\_btwn\_trans avg\_trans\_growth\_rate trans\_recency

merchant

00057d4302 1156 1279 1,217.50 2435 2 1.43 0.11 -581

000ed1585f 21932 35784 28,050.25 112201 4 16.03 0.19 -175

000f8c3297 3455 15047 6,635.56 106169 16 4.47 0.12 -59

0020aefbd9 3589 3589 3,589.00 3589 1 NaN NaN -216

0026f256ac 34880 34880 34,880.00 34880 1 NaN NaN -473

Descriptive statistics:

min\_amount max\_amount trans\_amount\_avg trans\_amount\_volume trans\_frequency avg\_time\_btwn\_trans avg\_trans\_growth\_rate trans\_recency

count 7,902.00 7,902.00 7,902.00 7,902.00 7,902.00 5,253.00 5,253.00 7,902.00

mean 20,390.86 55,609.45 30,733.18 196,354.72 12.66 11.45 1.90 -170.32

std 135,797.59 187,450.07 141,780.27 600,043.78 46.53 7.43 21.23 180.31

min 201.00 209.00 209.00 209.00 1.00 0.00 0.00 -727.00

25% 2,061.25 6,585.75 4,846.18 10,252.00 1.00 4.69 0.13 -265.00

50% 4,226.00 15,442.00 9,053.63 34,840.00 3.00 11.45 0.33 -98.00

75% 10,510.25 40,685.00 21,147.05 138,863.00 8.00 18.04 0.89 -26.00

max 10,385,508.00 10,385,508.00 10,385,508.00 15,499,827.00 1,673.00 24.00 1,224.39 -1.00

#######################################################################################

# Problem 2

#######################################################################################

Problem:

You are provided with two datasets: sales data.csv and product info.csv.

- sales data.csv contains transaction records with columns: 'TransactionID',

'ProductID', 'Date', 'Quantity', and 'Price'.

- product info.csv contains product details with columns: 'ProductID', 'ProductName', 'Category'.

Your task involves multiple steps of data manipulation using Pandas and NumPy to

extract insights from these datasets.

Tasks:

1. Data Loading and Merging:

- Load both datasets using Pandas.

- Merge them into a single DataFrame on 'ProductID'.

2. Data Cleaning:

- Check for and handle any missing values in the merged dataset.

- Convert the 'Date' column to a DateTime object.

3. Data Analysis using Slicing and Indexing:

- Create a new column 'TotalSale', calculated as 'Quantity' \* 'Price'.

- Using slicing, create a subset DataFrame containing only transactions from

the last quarter of the year (October, November, December).

- Using Boolean indexing, find all transactions for a specific 'Category' (e.g.,

'Electronics').

- Extract all transactions where the 'TotalSale' is above the 75th percentile

of the 'TotalSale' column using NumPy functions.

4. Advanced Indexing:

- Using loc and iloc, perform the following:

- Select all rows for 'ProductID' 101 and columns 'ProductName' and

'TotalSale'.

- Select every 10th row from the merged dataset and only the columns

'Date' and 'Category'.

5. Grouping and Aggregation:

- Group the data by 'Category' and calculate the total and average 'TotalSale'

for each category.

6. Time-Series Analysis:

- Resample the data on a monthly basis and calculate the total 'Quantity'

sold per month.

Final Deliverables:

- Provide the code for each step.

- Include comments explaining your approach.

- Display the first 5 rows of the DataFrame after each major step.

Code:

import numpy as np

import pandas as pd

pd.options.display.float\_format = '{:,.2f}'.format

print("Task 1 - Data Loading and Merging")

SalesData = pd.read\_csv("./gwu/SEAS 6414/sales\_data.csv")

Product = pd.read\_csv("./gwu/SEAS 6414/product\_info.csv")

# I checked the row counts and there are no product ids in the Sales Data

# that have product keys that are not in Product data. So, an inner join

# will work for this data.

SalesProductData = pd.merge(SalesData, Product, on="ProductID", how="inner")

print("The merged SalesProductData data frame.")

print(SalesProductData)

print("")

print("Task 2 - Data Cleaning")

# Counting the NAs across the dimensions shows that there is no missing data. I also ran

# dropna and saw that the result had the same shape as the input. So, I am confident that

# there is no missing data. Which worries me. Why would you ask us to address missing data

# if there was none.

print(f"The merged data frame has {SalesProductData.isnull().sum().sum()} missing values.")

print("")

print("The SalesProductData types before converting to a datetime:")

print(SalesProductData.dtypes)

SalesProductData['Date'] = [pd.to\_datetime(aDate) for aDate in SalesProductData.Date]

print("")

print("The SalesProductData types after converting to a datetime:")

print(SalesProductData.dtypes)

print("")

print("Task 3 - Data Analysis using Slicing and Indexing")

SalesProductData['TotalSale'] = SalesProductData['Quantity'] \* SalesProductData['Price']

SalesProductData4Q = SalesProductData.set\_index('Date').sort\_values(by=['Date'])['2023-10-01' : '2023-12-31']

print("")

print("Sales records for the fourth quarter:")

print(SalesProductData4Q)

mask = SalesProductData['Category'] == 'Electronics'

SalesProductElectronics = SalesProductData[mask]

print("")

print("Sales records for Electronics:")

print(SalesProductElectronics)

# First create and index of all records that have a TotalSale value greater than the 75th percentile

SalesProductOver75Index = np.where(SalesProductData['TotalSale']>np.percentile(SalesProductData['TotalSale'],75))

# Next select those values.

SalesProductOver75 = SalesProductData.loc[SalesProductOver75Index]

print("")

print("Sales records for total price over the 75th percentile:")

print(SalesProductOver75)

print("")

print("Task 4 - Advanced Indexing")

SalesProductDataPID = SalesProductData.set\_index('ProductID')

SalesProductData101 = SalesProductDataPID.loc[101,['ProductName','TotalSale']]

print("")

print("Sales records for product 101 with Product Name and Total Sale:")

print(SalesProductData101)

SalesProductDataEvery10th = SalesProductData.iloc[::10,[2,6]]

print("")

print("Sales records for every 10th row with Date and Category:")

print(SalesProductDataEvery10th)

print("")

print("Task 5 - Grouping and Aggregation")

SalesProductDataCatGrp = SalesProductData.groupby("Category").agg(

total\_sale=("TotalSale", "sum")

, average\_sale=("TotalSale", "mean")

)

print("")

print("Sales records grouped by category with total and average sales by category:")

print(SalesProductDataCatGrp)

print("")

print("Task 6 - Time-Series Analysis")

# Get down to just the columns needed. I tried to combine this with the indexing but

# none of my incantations would work.

SalesProductDataSmall = SalesProductData.loc[:,['Date','Quantity']]

# To resample, we need the date to be the index

SalesProductDataDate = SalesProductDataSmall.set\_index('Date')

# Now we can resample down to Month End and calculate the average

SalesProductDataMonth = SalesProductDataDate.resample('ME').mean()

print(SalesProductDataMonth)

Execution:

Task 1 - Data Loading and Merging

The merged SalesProductData data frame.

TransactionID ProductID Date Quantity Price ProductName Category

0 1 136 2023-03-13 8 245.29 pull Toys

1 2 121 2023-06-09 2 355.60 left Home Appliances

2 3 179 2023-04-18 7 25.39 according Books

3 4 142 2023-09-03 10 260.76 hospital Toys

4 5 101 2023-06-21 1 212.49 ready Clothing

... ... ... ... ... ... ... ...

9995 9996 136 2023-01-30 7 29.29 pull Toys

9996 9997 160 2023-05-23 1 96.70 next Electronics

9997 9998 122 2023-07-14 10 175.15 product Toys

9998 9999 116 2023-03-25 10 337.27 carry Home Appliances

9999 10000 186 2024-01-11 5 451.14 increase Home Appliances

[10000 rows x 7 columns]

Task 2 - Data Cleaning

The merged data frame has 0 missing values.

The SalesProductData types before converting to a datetime:

TransactionID int64

ProductID int64

Date object

Quantity int64

Price float64

ProductName object

Category object

dtype: object

The SalesProductData types after converting to a datetime:

TransactionID int64

ProductID int64

Date datetime64[ns]

Quantity int64

Price float64

ProductName object

Category object

dtype: object

Task 3 - Data Analysis using Slicing and Indexing

Sales records for the fourth quarter:

TransactionID ProductID Quantity Price ProductName Category TotalSale

Date

2023-10-01 4495 125 9 155.04 instead Toys 1,395.37

2023-10-01 3289 172 3 334.31 other Books 1,002.94

2023-10-01 5299 140 6 15.47 condition Clothing 92.83

2023-10-01 4878 165 9 271.28 others Home Appliances 2,441.51

2023-10-01 4358 156 6 335.66 create Toys 2,013.94

... ... ... ... ... ... ... ...

2023-12-31 1495 172 2 72.93 other Books 145.86

2023-12-31 7296 158 7 227.04 team Books 1,589.29

2023-12-31 3479 103 2 276.34 avoid Clothing 552.68

2023-12-31 4827 135 3 158.90 candidate Books 476.71

2023-12-31 9126 138 2 174.79 collection Home Appliances 349.58

[2524 rows x 7 columns]

Sales records for Electronics:

TransactionID ProductID Date Quantity Price ProductName Category TotalSale

6 7 134 2023-11-06 3 182.18 interview Electronics 546.54

50 51 164 2023-12-14 9 480.57 energy Electronics 4,325.13

54 55 166 2023-04-25 5 410.22 group Electronics 2,051.08

56 57 145 2023-08-21 5 405.02 market Electronics 2,025.11

66 67 166 2023-04-21 4 447.68 group Electronics 1,790.74

... ... ... ... ... ... ... ... ...

9977 9978 177 2023-09-29 8 399.54 floor Electronics 3,196.35

9980 9981 124 2023-11-11 9 236.24 table Electronics 2,126.18

9981 9982 134 2023-10-04 4 399.44 interview Electronics 1,597.76

9985 9986 175 2023-09-24 2 156.95 true Electronics 313.90

9996 9997 160 2023-05-23 1 96.70 next Electronics 96.70

[1465 rows x 8 columns]

Sales records for total price over the 75th percentile:

TransactionID ProductID Date Quantity Price ProductName Category TotalSale

3 4 142 2023-09-03 10 260.76 hospital Toys 2,607.58

13 14 186 2023-01-31 8 405.01 increase Home Appliances 3,240.12

14 15 143 2023-11-30 10 293.56 cup Clothing 2,935.56

17 18 173 2023-08-08 9 458.96 either Toys 4,130.64

18 19 172 2023-06-03 6 363.44 other Books 2,180.66

... ... ... ... ... ... ... ... ...

9986 9987 154 2023-10-09 9 497.43 could Books 4,476.90

9987 9988 169 2023-04-09 8 341.18 everyone Toys 2,729.46

9988 9989 113 2023-02-17 6 458.28 positive Clothing 2,749.66

9998 9999 116 2023-03-25 10 337.27 carry Home Appliances 3,372.71

9999 10000 186 2024-01-11 5 451.14 increase Home Appliances 2,255.70

[2500 rows x 8 columns]

Task 4 - Advanced Indexing

Sales records for product 101 with Product Name and Total Sale:

ProductName TotalSale

ProductID

101 ready 212.49

101 ready 1,331.01

101 ready 3,311.02

101 ready 1,565.75

101 ready 74.59

... ... ...

101 ready 623.85

101 ready 207.41

101 ready 1,348.78

101 ready 2,056.10

101 ready 1,845.12

[98 rows x 2 columns]

Sales records for every 10th row with Date and Category:

Date Category

0 2023-03-13 Toys

10 2023-05-16 Home Appliances

20 2023-12-18 Home Appliances

30 2023-12-05 Books

40 2023-04-07 Books

... ... ...

9950 2024-01-12 Home Appliances

9960 2023-03-02 Electronics

9970 2023-10-09 Clothing

9980 2023-11-11 Electronics

9990 2023-04-28 Toys

[1000 rows x 2 columns]

Task 5 - Grouping and Aggregation

Sales records grouped by category with total and average sales by category:

total\_sale average\_sale

Category

Books 2,756,942.14 1,405.17

Clothing 2,547,136.81 1,339.89

Electronics 2,151,251.34 1,468.43

Home Appliances 3,339,347.31 1,414.38

Toys 3,320,096.27 1,436.65

Task 6 - Time-Series Analysis

Quantity

Date

2023-01-31 5.40

2023-02-28 5.39

2023-03-31 5.60

2023-04-30 5.56

2023-05-31 5.68

2023-06-30 5.48

2023-07-31 5.57

2023-08-31 5.39

2023-09-30 5.36

2023-10-31 5.59

2023-11-30 5.43

2023-12-31 5.59

2024-01-31 5.40

#######################################################################################

# Problem 3

#######################################################################################

Problem:

Zillow's marketplace offers a data-driven home valuation platform utilized by a diverse

range of users including home buyers, sellers, renters, homeowners, real estate

agents, mortgage providers, property managers, and landlords. The machine learning

and data science team at Zillow employs various tools for predicting home valuations,

such as Zestimate (Zillow Estimate), Zestimate Forecast, Zillow Home Value Index,

Rent Zestimate, Zillow Rent Index, and the Pricing Tool.

Assignment Overview:

You are provided with a dataset named zillow feature sample.csv, containing

various features relevant to Zillow's marketplace. Accompanying the dataset is a

data dictionary titled zillow data dictionary.xlsx, which details the description

of each column.

Tasks:

1. Develop a Missing Data Strategy:

- Assess the zillow feature sample.csv dataset and devise a comprehensive strategy to handle missing data.

2. Quantitative Analysis of Missing Data:

- Calculate and report the percentage of missing data in each feature of the

dataset.

- Analyze and infer the potential mechanism of missing data (e.g., Missing

Completely at Random, Missing at Random, Missing Not at Random).

3. Imputation Strategy:

- Propose and justify an imputation strategy for the missing values in the

dataset. Your rationale should be data-driven and well-explained.

4. Open-Ended Exploration:

- This question is open-ended, allowing you to explore other relevant aspects

of the dataset. Conduct additional analyses or apply data processing techniques as appropriate.

Submission Guidelines:

- Document your analysis and findings in a clear and structured format.

- Ensure that your submission is thorough and well-reasoned.

Code:

import numpy as np

import pandas as pd

pd.options.display.float\_format = '{:,.2f}'.format

print("")

print("Task 1 - Develop a Missing Data Strategy")

ZillowFeatureSample = pd.read\_csv("./gwu/SEAS 6414/zillow\_feature\_sample.csv")

print("The data provided:")

print(ZillowFeatureSample)

print("Descriptive statistics for each feature:")

print(ZillowFeatureSample.describe())

print("")

print("This data is used to predict house prices. Since it does not have actual prices, we cannot")

print("use it for training or testing our models. Therefore, we cannot test the impact of any")

print("missing data strategy with just this data at hand. However, we can look at the data and")

print("determine if any missing data approach would be useful. Below is my strategy based on a")

print("review of the data values and data dictionary.")

print("")

print("From the data dictionary:")

print(" - The data dictionary has eight tabs.")

print(" - The first one is for the data file.")

print(" - The remaining seven are code tables for features that are coded.")

print(" - Eight of feature descriptions had the phrase 'if any' in them, or should.")

print(" - Some features probably should include 'if any' in the description")

print(" - For example, 'airconditioningtypeid' is described as 'Type of cooling system")

print(" present in the home (if any)'")

print(" - For example, 'assessmentyear' is described as 'The year of the property tax assessment'.")

print(" Since a house may never have been assessed, this is similar to 'if any'.")

print(" - In both these cases, any unavailable information could be treated as a No or whatever")

print(" is appropriate.")

print(" - Seventeen of the features have the characters ID at the end of the name.")

print(" - Of these seven have tables on other tabs and ten do not.")

print(" - Assignment of an ID means that a process was followed to code the data.")

print(" - Given this process, I would be reluctant to replace the missing data with a value.")

print(" - Some data is dependant on other data.")

print(" - If 'regionidzip' is available, we could use that to fill in City, State, etc.")

print(" - For each feature, we can look into any dependencies that could help derive the values.")

print(" - We will need to be careful with this. We will have to determine the dependencies, then")

print(" - derive the data, then remove the dependant values so that only one of them remains. This")

print(" - ensures that we are only left with independent variables (features).")

print(" - There appear to be a lot of missing values. We will need to carefully consider these")

print(" features. We may need to drop those that are missing too many values.")

print("")

print("Task 2 - Quantitative Analysis of Missing Data")

missing\_value\_analysis = pd.DataFrame({'count\_missing': ZillowFeatureSample.isna().sum()

, 'percent\_missing': ZillowFeatureSample.isnull().sum() \* 100 / len(ZillowFeatureSample)})

print("")

print("Count and percent missing for each feature, sorted low to high by percent:")

print(missing\_value\_analysis.sort\_values(by=['percent\_missing']))

print("")

print("Searching the web, it looks like a lot of people consider between 10 and 20% missing")

print("a cutoff point -> more than 20% missing, do not use the feature. But this is always followed")

print("with - there is no hard cutoff point. Since we have 9.25% missing and then 34.00% missing")

print("my working assumption for now is that this will be the cutoff point. But I will continue")

print("analyzing the data to see if some of the features with 34.00% or greater missing are useful.")

print("")

print("Trying to infer the mechanism of missing data will be tricky for me. There are several")

print("reasons for this:")

print(" - I do not know how any of the data was collected.")

print(" - This is not an area that I have any expertise in.")

print("")

print("With those caveats in mind, here is my estimation for each feature.")

print(" - For the 23 features that have a missing percent under 4%, I deem them as not")

print(" really missing. If a value is needed for them, it can easily be imputed.")

print(" - For the 26 features with a missing percent over 70%, I deem them as to much")

print(" missing. I would be hard pressed to impute these values. There may be special")

print(" cases as the analysis progresses.")

print(" - The remaining nine features need to be addressed.")

print(" - Based on the information provided, I cannot say if they are MCAR, MAR, or MNAR.")

print(" I would need details about how the information was collected and about housing")

print(" data.")

print("")

print("Based on the above, I created the table below for values that could be imputed:")

print(" finishedsquarefeet12 Impute from Calculated square feet")

print(" lotsizesquarefeet Impute from address")

print(" unitcnt Do not impute - I expect number of units to be unique")

print(" propertyzoningdesc Impute from address")

print(" buildingqualitytypeid Do not impute - an ID")

print(" heatingorsystemtypeid Do not impute - an ID")

print(" regionidneighborhood Impute from address")

print(" garagecarcnt Impute from address")

print(" garagetotalsqft Impute from address")

print("")

print("Task 3 - Imputation strategy")

print("")

print("Let me start by saying that my gut reaction is that using imputation is a really bad")

print("idea. We have data that we are trying to use to predict something and before we do")

print("we are predicting values that are missing from the data. If we use existing values to")

print("impute the values, we are not adding anything to the data we have. I am actually concerned")

print("that people are making decisions based on this. It seems like an incredibly bad idea.")

print("")

print("If I had to impute values for this data set, I would use averages in most cases. I would")

print("try to find a set of the data from the same general area and similar houses. This is based")

print("on the idea that all 3,000 square foot houses built in the same area in the same time period")

print("will essentially be the same. So, if we can get enough records, we can do that. This data set")

print("may be too small to get enough records. But given that Zillow seems to have data for every")

print("house in the US, it should be possible to get more data.")

print("")

print("Based on this, I would be willing to impute values for the 23 features that are missing under")

print("4% of the values and the four features identified above.")

print("")

print("Task 4 - Open-Ended Exploration")

print("")

print("Does year built correlate with size?")

ZillowFeatureSampleSmall = ZillowFeatureSample.loc[:,['yearbuilt','calculatedfinishedsquarefeet']]

print(ZillowFeatureSampleSmall.corr(numeric\_only=True))

print("It appears to have a low correlation.")

print("")

print("Does latitude correlate air conditioning?")

ZillowFeatureSampleSmall = ZillowFeatureSample.loc[:,['latitude','airconditioningtypeid']]

print(ZillowFeatureSampleSmall.corr(numeric\_only=True))

print("This seems to be saying that there is an inverse relation. That makes sense. The higher")

print("the latitude, the less need there is for air conditioning. Note that the values for")

print("air conditioning are not really good for this correlation. To really do it right, I would")

print("need to convert the values. But as a first cut, it makes sense.")

print("I could probably do similar things for pools at lower latitudes and fire places at higher")

print("latitudes. I am not sure it would be worthwhile given the amount of missing data.")

Execution:

Task 1 - Develop a Missing Data Strategy

The data provided:

parcelid airconditioningtypeid architecturalstyletypeid basementsqft bathroomcnt bedroomcnt buildingclasstypeid buildingqualitytypeid calculatedbathnbr decktypeid finishedfloor1squarefeet calculatedfinishedsquarefeet finishedsquarefeet12 finishedsquarefeet13 finishedsquarefeet15 finishedsquarefeet50 finishedsquarefeet6 fips fireplacecnt fullbathcnt garagecarcnt garagetotalsqft hashottuborspa heatingorsystemtypeid latitude longitude lotsizesquarefeet poolcnt poolsizesum pooltypeid10 pooltypeid2 pooltypeid7 propertycountylandusecode propertylandusetypeid propertyzoningdesc rawcensustractandblock regionidcity regionidcounty regionidneighborhood regionidzip roomcnt storytypeid threequarterbathnbr typeconstructiontypeid unitcnt yardbuildingsqft17 yardbuildingsqft26 yearbuilt numberofstories fireplaceflag structuretaxvaluedollarcnt taxvaluedollarcnt assessmentyear landtaxvaluedollarcnt taxamount taxdelinquencyflag taxdelinquencyyear censustractandblock

0 12833975 NaN NaN NaN 3.00 4.00 NaN 6.00 3.00 NaN NaN 1,812.00 1,812.00 NaN NaN NaN NaN 6,037.00 NaN 3.00 NaN NaN NaN 2.00 33,999,334.00 -117,955,651.00 5,419.00 NaN NaN NaN NaN NaN 0100 261.00 LCR106 60,374,086.31 45,602.00 3,101.00 NaN 96,489.00 0.00 NaN NaN NaN 1.00 NaN NaN 1,955.00 NaN NaN 155,403.00 304,592.00 2,016.00 149,189.00 3,708.29 NaN NaN 60,374,086,311,002.00

1 11070096 1.00 NaN NaN 4.00 4.00 NaN 7.00 4.00 NaN NaN 3,134.00 3,134.00 NaN NaN NaN NaN 6,037.00 NaN 4.00 NaN NaN NaN 2.00 34,283,974.00 -118,583,756.00 NaN NaN NaN NaN NaN NaN 010D 269.00 LARE 60,371,082.02 12,447.00 3,101.00 275,078.00 96,356.00 0.00 NaN NaN NaN 1.00 NaN NaN 2,012.00 NaN NaN 493,070.00 821,783.00 2,016.00 328,713.00 10,087.59 NaN NaN 60,371,082,021,002.00

2 12752672 1.00 NaN NaN 2.00 3.00 NaN 6.00 2.00 NaN NaN 1,817.00 1,817.00 NaN NaN NaN NaN 6,037.00 NaN 2.00 NaN NaN NaN 2.00 33,899,765.00 -117,989,887.00 5,953.00 NaN NaN NaN NaN NaN 0100 261.00 LMR1\* 60,375,038.01 5,465.00 3,101.00 NaN 96,190.00 0.00 NaN NaN NaN 1.00 NaN NaN 1,957.00 NaN NaN 126,695.00 247,962.00 2,016.00 121,267.00 3,377.86 NaN NaN 60,375,038,011,001.00

3 11338563 NaN NaN NaN 3.00 4.00 NaN 7.00 3.00 NaN NaN 2,280.00 2,280.00 NaN NaN NaN NaN 6,037.00 NaN 3.00 NaN NaN NaN NaN 34,674,776.00 -118,418,589.00 856,438.00 NaN NaN NaN NaN NaN 0700 263.00 LCRA10000- 60,379,201.02 25,468.00 3,101.00 NaN 97,316.00 0.00 NaN NaN NaN 1.00 NaN NaN 2,006.00 NaN NaN 130,500.00 308,900.00 2,016.00 178,400.00 3,578.92 NaN NaN 60,379,201,022,009.00

4 17098704 NaN NaN NaN 0.00 3.00 NaN NaN NaN NaN 1,200.00 1,200.00 1,200.00 NaN NaN 1,200.00 NaN 6,111.00 NaN NaN 2.00 400.00 NaN NaN 34,364,318.00 -119,055,992.00 6,750.00 NaN NaN NaN NaN NaN 1110 261.00 NaN 61,110,004.00 26,965.00 2,061.00 NaN 97,113.00 5.00 NaN NaN NaN NaN NaN NaN 1,987.00 1.00 NaN 142,271.00 223,101.00 2,016.00 80,830.00 2,564.86 NaN NaN 61,110,004,003,032.00

... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...

9995 11176537 1.00 NaN NaN 3.00 4.00 NaN 8.00 3.00 NaN NaN 3,170.00 3,170.00 NaN NaN NaN NaN 6,037.00 NaN 3.00 NaN NaN NaN 2.00 34,473,832.00 -118,633,360.00 7,341.00 1.00 NaN NaN NaN 1.00 0104 261.00 LCA22\* 60,379,201.16 10,734.00 3,101.00 NaN 96,398.00 0.00 NaN NaN NaN 1.00 NaN NaN 2,000.00 NaN NaN 259,395.00 399,915.00 2,016.00 140,520.00 5,692.80 NaN NaN 60,379,201,162,006.00

9996 12544738 NaN NaN NaN 1.00 1.00 NaN 6.00 1.00 NaN NaN 882.00 882.00 NaN NaN NaN NaN 6,037.00 NaN 1.00 NaN NaN NaN 7.00 33,767,900.00 -118,166,000.00 16,410.00 NaN NaN NaN NaN NaN 010E 266.00 LBR2N 60,375,768.01 46,298.00 3,101.00 416,302.00 96,237.00 0.00 NaN NaN NaN 1.00 NaN NaN 1,957.00 NaN NaN 73,738.00 98,658.00 2,016.00 24,920.00 1,242.21 NaN NaN 60,375,768,013,008.00

9997 12546936 NaN NaN NaN 2.00 2.00 NaN 8.00 2.00 NaN NaN 1,308.00 1,308.00 NaN NaN NaN NaN 6,037.00 NaN 2.00 NaN NaN NaN 2.00 33,765,200.00 -118,177,000.00 28,576.00 NaN NaN NaN NaN NaN 010E 266.00 LBPD5 60,375,766.01 46,298.00 3,101.00 416,302.00 96,236.00 0.00 NaN NaN NaN 1.00 NaN NaN 1,958.00 NaN NaN 130,500.00 520,000.00 2,016.00 389,500.00 6,214.01 NaN NaN 60,375,766,011,005.00

9998 10858309 1.00 NaN NaN 2.00 2.00 NaN 7.00 2.00 NaN NaN 1,250.00 1,250.00 NaN NaN NaN NaN 6,037.00 NaN 2.00 NaN NaN NaN 2.00 34,150,700.00 -118,441,000.00 23,518.00 1.00 NaN NaN NaN 1.00 010C 266.00 LARD1.5 60,371,412.01 12,447.00 3,101.00 27,080.00 96,424.00 0.00 NaN NaN NaN 1.00 NaN NaN 1,979.00 NaN NaN 112,656.00 167,805.00 2,016.00 55,149.00 2,109.45 NaN NaN 60,371,412,012,001.00

9999 12617699 NaN NaN NaN 2.00 4.00 NaN 6.00 2.00 NaN NaN 1,581.00 1,581.00 NaN NaN NaN NaN 6,037.00 NaN 2.00 NaN NaN NaN 2.00 33,800,022.00 -118,218,562.00 6,444.00 NaN NaN NaN NaN NaN 0100 261.00 LBR1N 60,375,727.00 46,298.00 3,101.00 275,989.00 96,244.00 0.00 NaN NaN NaN 1.00 NaN NaN 1,977.00 NaN NaN 101,845.00 129,147.00 2,016.00 27,302.00 1,689.56 NaN NaN 60,375,727,004,003.00

[10000 rows x 58 columns]

Descriptive statistics for each feature:

parcelid airconditioningtypeid architecturalstyletypeid basementsqft bathroomcnt bedroomcnt buildingclasstypeid buildingqualitytypeid calculatedbathnbr decktypeid finishedfloor1squarefeet calculatedfinishedsquarefeet finishedsquarefeet12 finishedsquarefeet13 finishedsquarefeet15 finishedsquarefeet50 finishedsquarefeet6 fips fireplacecnt fullbathcnt garagecarcnt garagetotalsqft heatingorsystemtypeid latitude longitude lotsizesquarefeet poolcnt poolsizesum pooltypeid10 pooltypeid2 pooltypeid7 propertylandusetypeid rawcensustractandblock regionidcity regionidcounty regionidneighborhood regionidzip roomcnt storytypeid threequarterbathnbr typeconstructiontypeid unitcnt yardbuildingsqft17 yardbuildingsqft26 yearbuilt numberofstories structuretaxvaluedollarcnt taxvaluedollarcnt assessmentyear landtaxvaluedollarcnt taxamount taxdelinquencyyear censustractandblock

count 10,000.00 2,781.00 13.00 4.00 9,987.00 9,987.00 39.00 6,470.00 9,612.00 68.00 695.00 9,851.00 9,141.00 26.00 612.00 695.00 72.00 9,987.00 1,047.00 9,612.00 3,022.00 3,022.00 6,243.00 9,987.00 9,987.00 9,075.00 1,838.00 106.00 63.00 110.00 1,725.00 9,987.00 9,987.00 9,790.00 9,987.00 3,922.00 9,958.00 9,987.00 4.00 1,071.00 20.00 6,600.00 254.00 12.00 9,834.00 2,345.00 9,856.00 9,881.00 9,987.00 9,790.00 9,934.00 184.00 9,760.00

mean 13,275,043.02 2.06 7.69 830.75 2.23 3.09 3.67 6.31 2.31 66.00 1,392.47 1,833.44 1,778.31 1,154.62 2,624.70 1,398.83 2,352.69 6,048.19 1.17 2.26 1.82 381.15 4.09 34,000,911.04 -118,202,882.01 23,218.24 1.00 490.61 1.00 1.00 1.00 260.19 60,484,996.91 34,571.57 2,561.21 196,536.71 96,603.31 1.51 7.00 1.01 6.00 1.16 318.82 269.92 1,964.53 1.41 177,411.47 443,317.62 2,016.00 268,830.85 5,391.37 13.61 60,485,717,292,883.38

std 7,303,010.69 3.35 4.23 424.65 1.08 1.26 0.62 1.75 1.01 0.00 601.14 1,098.69 987.65 350.81 1,990.31 609.42 1,237.82 20.30 0.47 1.00 0.66 258.70 3.30 243,729.18 348,059.72 115,331.36 0.00 119.23 0.00 0.00 0.00 15.16 201,540.04 49,474.74 792.08 171,211.13 5,276.70 2.87 0.00 0.10 0.00 0.57 196.35 306.78 23.69 0.54 214,892.76 568,226.73 0.05 408,268.68 6,519.44 1.66 201,940,766,465.50

min 10,711,956.00 1.00 2.00 240.00 0.00 0.00 1.00 1.00 1.00 66.00 69.00 2.00 2.00 520.00 432.00 69.00 432.00 6,037.00 1.00 1.00 0.00 0.00 1.00 33,339,600.00 -119,446,532.00 329.00 1.00 207.00 1.00 1.00 1.00 31.00 60,371,011.10 3,491.00 1,286.00 6,952.00 95,982.00 0.00 7.00 1.00 6.00 1.00 40.00 30.00 1,861.00 1.00 9.00 7.00 2,014.00 7.00 21.44 9.00 60,371,011,101,002.00

25% 11,636,318.25 1.00 7.00 739.50 2.00 2.00 3.00 5.00 2.00 66.00 1,020.00 1,215.00 1,198.00 900.00 1,675.50 1,021.50 1,342.00 6,037.00 1.00 2.00 2.00 300.00 2.00 33,826,478.00 -118,395,579.00 5,722.00 1.00 420.00 1.00 1.00 1.00 261.00 60,373,117.00 12,447.00 1,286.00 46,736.00 96,185.00 0.00 7.00 1.00 6.00 1.00 187.50 89.00 1,950.00 1.00 78,617.50 192,896.00 2,016.00 84,164.75 2,508.37 13.00 60,373,115,001,759.00

50% 12,558,408.00 1.00 7.00 915.50 2.00 3.00 4.00 6.00 2.00 66.00 1,332.00 1,578.00 1,544.00 1,344.00 2,185.50 1,336.00 2,235.50 6,037.00 1.00 2.00 2.00 441.00 2.00 34,008,400.00 -118,173,535.00 7,067.00 1.00 467.50 1.00 1.00 1.00 261.00 60,375,717.04 25,218.00 3,101.00 118,920.00 96,378.00 0.00 7.00 1.00 6.00 1.00 277.00 126.00 1,963.00 1.00 128,198.00 326,192.00 2,016.00 180,417.00 4,059.53 14.00 60,375,719,002,016.00

75% 14,117,709.00 1.00 7.00 1,006.75 3.00 4.00 4.00 8.00 3.00 66.00 1,650.50 2,162.00 2,096.00 1,440.00 3,023.00 1,652.00 3,344.00 6,059.00 1.00 3.00 2.00 493.00 7.00 34,163,792.50 -117,946,339.50 10,044.50 1.00 540.00 1.00 1.00 1.00 261.00 60,590,423.25 45,457.00 3,101.00 274,765.00 96,974.00 0.00 7.00 1.00 6.00 1.00 400.00 360.00 1,981.00 2.00 206,600.75 522,933.00 2,016.00 334,595.75 6,305.05 15.00 60,590,423,303,000.00

max 168,182,628.00 13.00 21.00 1,252.00 12.00 12.00 4.00 12.00 12.00 66.00 5,408.00 35,560.00 35,560.00 1,536.00 32,548.00 5,408.00 5,357.00 6,111.00 6.00 12.00 19.00 5,974.00 24.00 34,762,584.00 -117,559,744.00 6,971,010.00 1.00 840.00 1.00 1.00 1.00 275.00 61,110,091.00 396,556.00 3,101.00 764,167.00 399,675.00 14.00 7.00 2.00 6.00 7.00 1,260.00 1,123.00 2,015.00 4.00 5,275,190.00 19,310,938.00 2,016.00 14,217,944.00 224,000.93 15.00 61,110,091,003,005.00

This data is used to predict house prices. Since it does not have actual prices, we cannot

use it for training or testing our models. Therefore, we cannot test the impact of any

missing data strategy with just this data at hand. However, we can look at the data and

determine if any missing data approach would be useful. Below is my strategy based on a

review of the data values and data dictionary.

From the data dictionary:

- The data dictionary has eight tabs.

- The first one is for the data file.

- The remaining seven are code tables for features that are coded.

- Eight of feature descriptions had the phrase 'if any' in them, or should.

- Some features probably should include 'if any' in the description

- For example, 'airconditioningtypeid' is described as 'Type of cooling system

present in the home (if any)'

- For example, 'assessmentyear' is described as 'The year of the property tax assessment'.

Since a house may never have been assessed, this is similar to 'if any'.

- In both these cases, any unavailable information could be treated as a No or whatever

is appropriate.

- Seventeen of the features have the characters ID at the end of the name.

- Of these seven have tables on other tabs and ten do not.

- Assignment of an ID means that a process was followed to code the data.

- Given this process, I would be reluctant to replace the missing data with a value.

- Some data is dependant on other data.

- If 'regionidzip' is available, we could use that to fill in City, State, etc.

- For each feature, we can look into any dependencies that could help derive the values.

- We will need to be careful with this. We will have to determine the dependencies, then

- derive the data, then remove the dependant values so that only one of them remains. This

- ensures that we are only left with independent variables (features).

- There appear to be a lot of missing values. We will need to carefully consider these

features. We may need to drop those that are missing too many values.

Task 2 - Quantitative Analysis of Missing Data

Count and percent missing for each feature, sorted low to high by percent:

count\_missing percent\_missing

parcelid 0 0.00

fips 13 0.13

propertylandusetypeid 13 0.13

rawcensustractandblock 13 0.13

regionidcounty 13 0.13

longitude 13 0.13

roomcnt 13 0.13

bedroomcnt 13 0.13

bathroomcnt 13 0.13

assessmentyear 13 0.13

latitude 13 0.13

propertycountylandusecode 14 0.14

regionidzip 42 0.42

taxamount 66 0.66

taxvaluedollarcnt 119 1.19

structuretaxvaluedollarcnt 144 1.44

calculatedfinishedsquarefeet 149 1.49

yearbuilt 166 1.66

regionidcity 210 2.10

landtaxvaluedollarcnt 210 2.10

censustractandblock 240 2.40

fullbathcnt 388 3.88

calculatedbathnbr 388 3.88

finishedsquarefeet12 859 8.59

lotsizesquarefeet 925 9.25

unitcnt 3400 34.00

propertyzoningdesc 3411 34.11

buildingqualitytypeid 3530 35.30

heatingorsystemtypeid 3757 37.57

regionidneighborhood 6078 60.78

garagecarcnt 6978 69.78

garagetotalsqft 6978 69.78

airconditioningtypeid 7219 72.19

numberofstories 7655 76.55

poolcnt 8162 81.62

pooltypeid7 8275 82.75

threequarterbathnbr 8929 89.29

fireplacecnt 8953 89.53

finishedfloor1squarefeet 9305 93.05

finishedsquarefeet50 9305 93.05

finishedsquarefeet15 9388 93.88

yardbuildingsqft17 9746 97.46

taxdelinquencyflag 9816 98.16

taxdelinquencyyear 9816 98.16

hashottuborspa 9827 98.27

pooltypeid2 9890 98.90

poolsizesum 9894 98.94

finishedsquarefeet6 9928 99.28

decktypeid 9932 99.32

pooltypeid10 9937 99.37

buildingclasstypeid 9961 99.61

finishedsquarefeet13 9974 99.74

typeconstructiontypeid 9980 99.80

architecturalstyletypeid 9987 99.87

yardbuildingsqft26 9988 99.88

fireplaceflag 9989 99.89

basementsqft 9996 99.96

storytypeid 9996 99.96

Searching the web, it looks like a lot of people consider between 10 and 20% missing

a cutoff point -> more than 20% missing, do not use the feature. But this is always followed

with - there is no hard cutoff point. Since we have 9.25% missing and then 34.00% missing

my working assumption for now is that this will be the cutoff point. But I will continue

analyzing the data to see if some of the features with 34.00% or greater missing are useful.

Trying to infer the mechanism of missing data will be tricky for me. There are several

reasons for this:

- I do not know how any of the data was collected.

- This is not an area that I have any expertise in.

With those caveats in mind, here is my estimation for each feature.

- For the 23 features that have a missing percent under 4%, I deem them as not

really missing. If a value is needed for them, it can easily be imputed.

- For the 26 features with a missing percent over 70%, I deem them as to much

missing. I would be hard pressed to impute these values. There may be special

cases as the analysis progresses.

- The remaining nine features need to be addressed.

- Based on the information provided, I cannot say if they are MCAR, MAR, or MNAR.

I would need details about how the information was collected and about housing

data.

Based on the above, I created the table below for values that could be imputed:

finishedsquarefeet12 Impute from Calculated square feet

lotsizesquarefeet Impute from address

unitcnt Do not impute - I expect number of units to be unique

propertyzoningdesc Impute from address

buildingqualitytypeid Do not impute - an ID

heatingorsystemtypeid Do not impute - an ID

regionidneighborhood Impute from address

garagecarcnt Impute from address

garagetotalsqft Impute from address

Task 3 - Imputation strategy

Let me start by saying that my gut reaction is that using imputation is a really bad

idea. We have data that we are trying to use to predict something and before we do

we are predicting values that are missing from the data. If we use existing values to

impute the values, we are not adding anything to the data we have. I am actually concerned

that people are making decisions based on this. It seems like an incredibly bad idea.

If I had to impute values for this data set, I would use averages in most cases. I would

try to find a set of the data from the same general area and similar houses. This is based

on the idea that all 3,000 square foot houses built in the same area in the same time period

will essentially be the same. So, if we can get enough records, we can do that. This data set

may be too small to get enough records. But given that Zillow seems to have data for every

house in the US, it should be possible to get more data.

Based on this, I would be willing to impute values for the 23 features that are missing under

4% of the values and the four features identified above.

Task 4 - Open-Ended Exploration

Does year built correlate with size?

yearbuilt calculatedfinishedsquarefeet

yearbuilt 1.00 0.17

calculatedfinishedsquarefeet 0.17 1.00

It appears to have a low correlation.

Does latitude correlate air conditioning?

latitude airconditioningtypeid

latitude 1.00 -0.45

airconditioningtypeid -0.45 1.00

This seems to be saying that there is an inverse relation. That makes sense. The higher

the latitude, the less need there is for air conditioning. Note that the values for

air conditioning are not really good for this correlation. To really do it right, I would

need to convert the values. But as a first cut, it makes sense.

I could probably do similar things for pools at lower latitudes and fire places at higher

latitudes. I am not sure it would be worthwhile given the amount of missing data.