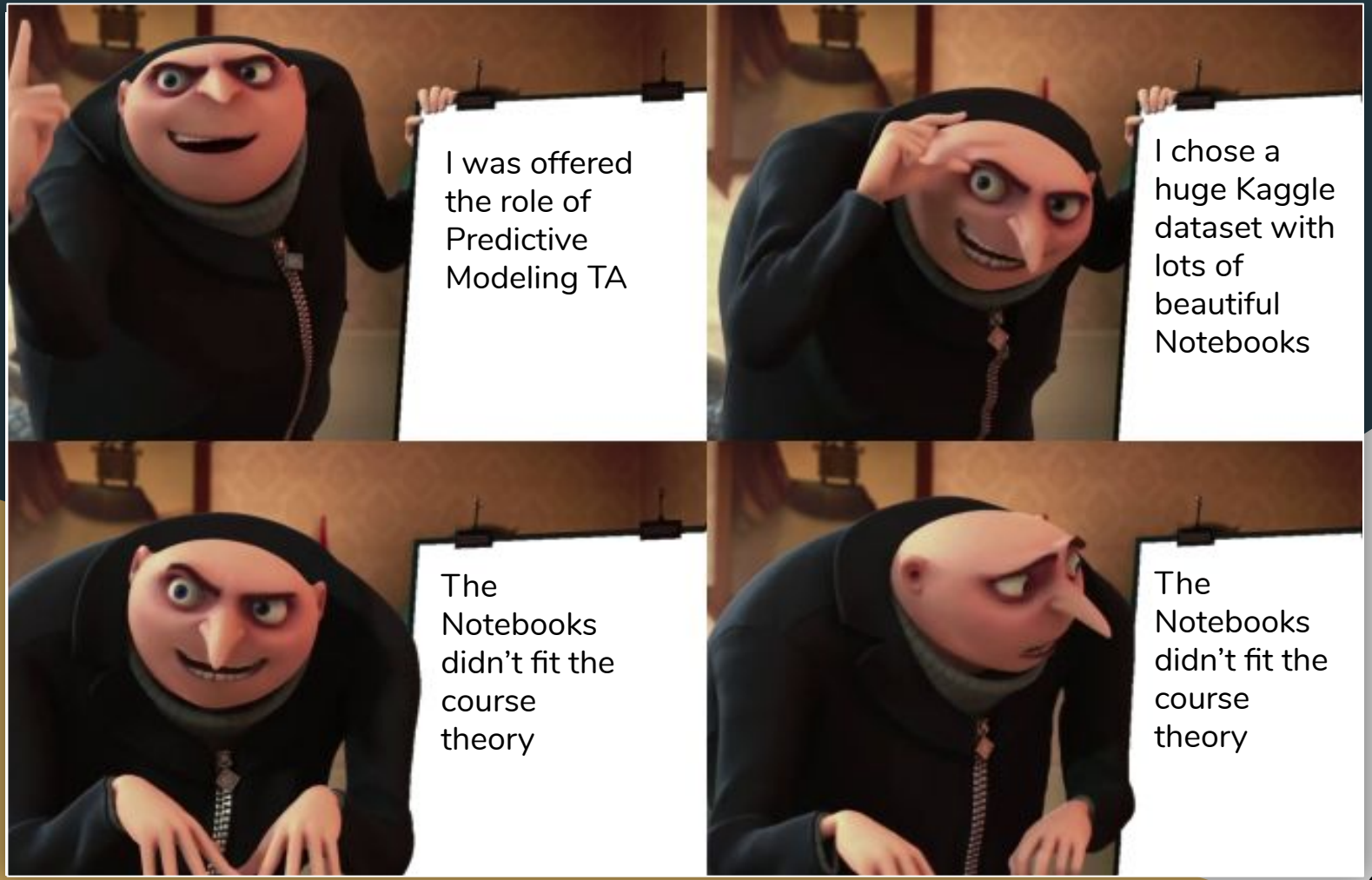
Before You Train

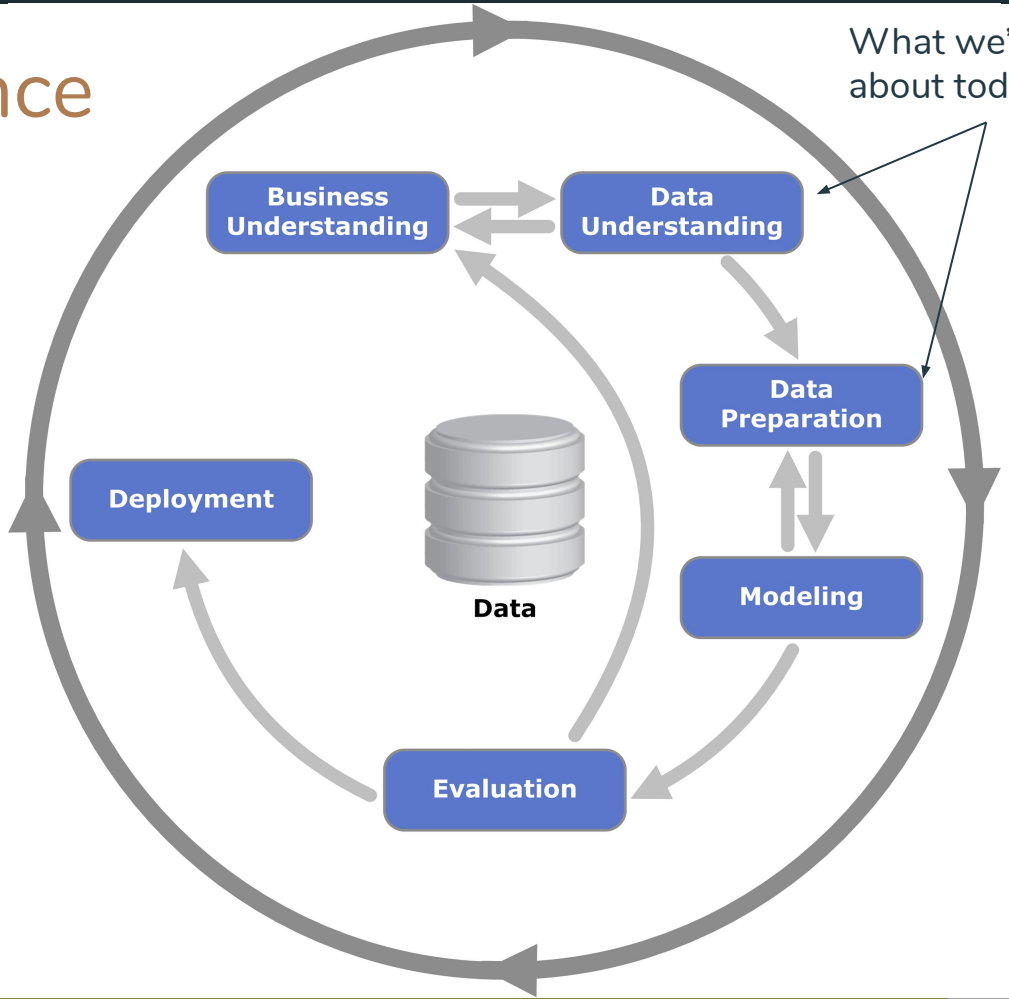
By Dina Bavli



By Dina Bavli

By Dina Bavli

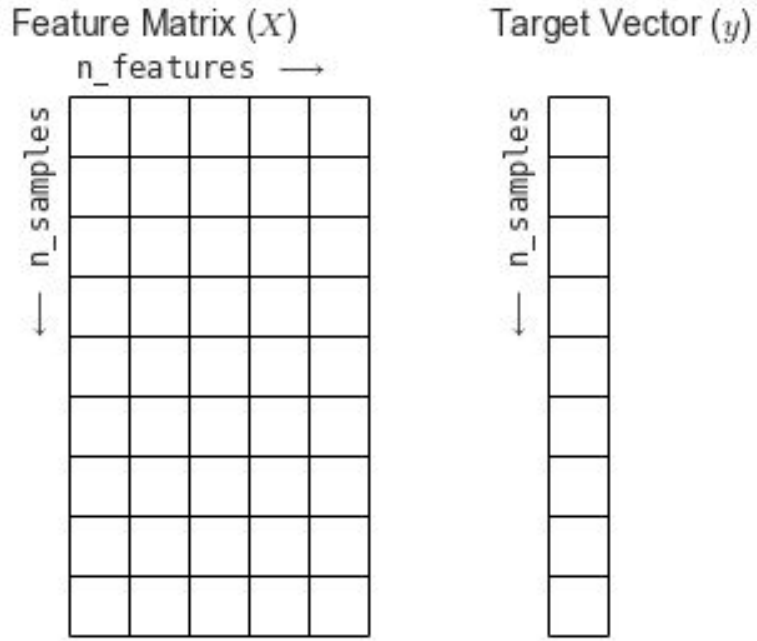
Data Science Pipeline



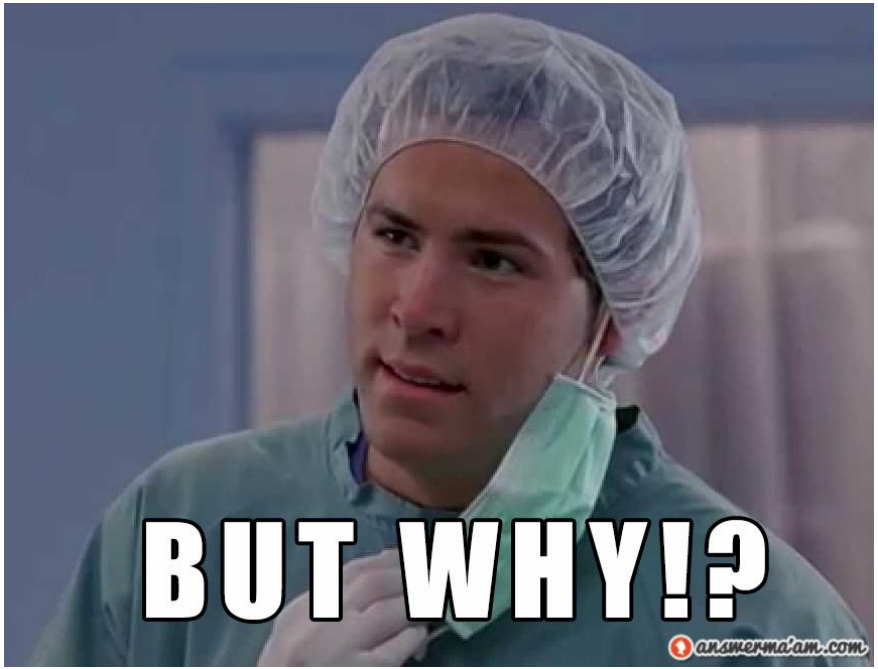
By Dina Bavli

By Dina Bavli

Classic Data Frame for Supervised Machine Learning



Why would you want to do that?



By Dina Bavli

How would you do that?

What do you need for that?

- **Read the F$@#ing description!!!**
- `import numpy as np # linear algebra`
- `import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)`
- `import matplotlib.pyplot as plt # data visualization`
- `import seaborn as sns # data visualization`
- `from numpy import mean #statistics`
- `from numpy import std #statistics`
- `from scipy.stats import stats #statistics`
- `from scipy.stats import kurtosis #statistics`

By Dina Bavli

EDA and Visualization

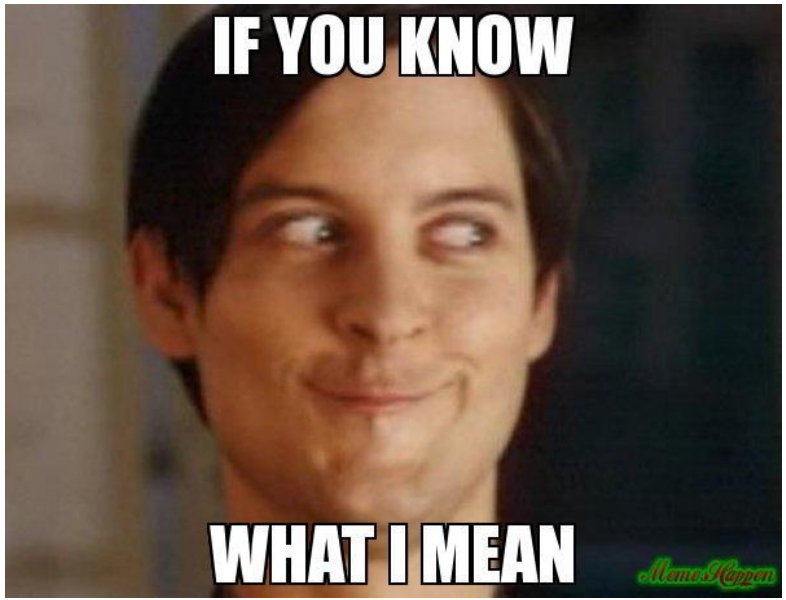
- The feature type is critical.
- Data description is important.
- Categorical features may appear as integers.
- You need to make sure all features are defined correctly.

Recommended visualization guide

By Dina Bavli

But Why?

It make no sense to make decisions based on the mean of a categorical feature.



Data describe

```
data_describe=data.describe()
```

```
data_describe.round(2).head(10)
```

	isFraud	TransactionAmt	card1	card2	card3	card5	addr1	addr2	dist1
count	590540.00	590540.00	590540.00	581607.00	588975.00	586281.00	524834.00	524834.00	238269.00
mean	0.03	135.03	9898.73	362.56	153.19	199.28	290.73	86.80	118.50
std	0.18	239.16	4901.17	157.79	11.34	41.24	101.74	2.69	371.87
min	0.00	0.25	1000.00	100.00	100.00	100.00	100.00	10.00	0.00
25%	0.00	43.32	6019.00	214.00	150.00	166.00	204.00	87.00	3.00
50%	0.00	68.77	9678.00	361.00	150.00	226.00	299.00	87.00	8.00
75%	0.00	125.00	14184.00	512.00	150.00	226.00	330.00	87.00	24.00
max	1.00	31937.39	18396.00	600.00	231.00	237.00	540.00	102.00	10286.00

8 rows × 406 columns

```
data_describe = data_describe.T
```

```
data_describe.head()
```

data_describe=data.describe()

data_describe = data_describe.T

	count	mean	std	min	25%	50%	75%	max
isFraud	590540.0	0.034990	0.183755	0.000	0.000	0.000	0.0	1.000
TransactionAmt	590540.0	135.027176	239.162522	0.251	43.321	68.769	125.0	31937.391
card1	590540.0	9898.734658	4901.170153	1000.000	6019.000	9678.000	14184.0	18396.000
card2	581607.0	362.555488	157.793246	100.000	214.000	361.000	512.0	600.000
card3	588975.0	153.194925	11.336444	100.000	150.000	150.000	150.0	231.000

By Dina Bavli

Let's get to know the data

Up close and personal.

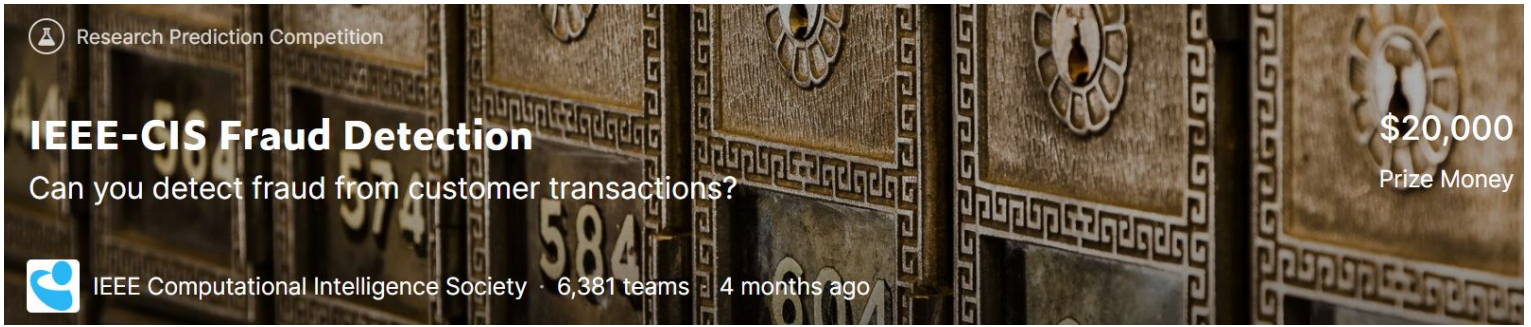
The data you'll handle is based on the IEEE-CIS Fraud Detection Kaggle competition dataset.

By Dina Bavli

The original data

Contains two tables - Transaction and Identity. The tables are associated by transactions.

Your data contains both tables joint by transactionID, and after some alterations.



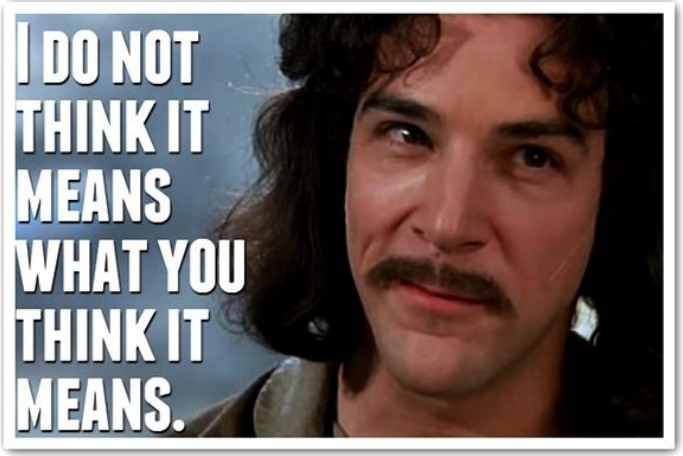
Features information is listed at the top of the notebook. https://bit.ly/2R6SOE6

For more info https://bit.ly/362mqgs .

By Dina Bavli

Defining category features

- Categorical features may appear as integers.
- You need to make sure all features are defined correctly.
- It make no sense to make decisions based on the mean of a categorical feature.



By Dina Bavli

Defining category features

```
cat_cols=['ProductCD', 'card1', 'card2', 'card3', 'card4', 'card5', 'card6',
       'addr1', 'addr2', 'M1', 'M2', 'M3', 'M4', 'M5', 'M6', 'M7', 'M8', 'M9',
       'id_12', 'id_13', 'id_15', 'id_16', 'id_17', 'id_19', 'id_20', 'id_28',
       'id_29', 'id_31', 'id_35', 'id_36', 'id_37', 'id_38', 'DeviceType',
       '_Weekdays', '_Hours', '_Days', 'P_emaildomain_bin',
       'P_emaildomain_suffix', 'R_emaildomain_bin', 'R_emaildomain_suffix',
       'device_name', 'had_id', '_Month']
```

```
for i in range(len(cat_cols)):
  col=cat_cols[i]
  data[col]=data[col].astype(str)
```

```
object_cols = data.select_dtypes(include=['object']).columns
len(object_cols)
```

```
48
```

Handling timedelta

TransactionDT: timedelta from a given reference datetime (not an actual timestamp)

By Dina Bavli

TimeDelta Feature

Converting to Total Days, Weekdays and Hours

```
import datetime

START_DATE = '2017-12-01'
startdate = datetime.datetime.strptime(START_DATE, "%Y-%m-%d")
df["Date"] = df['TransactionDT'].apply(lambda x: (startdate + datetime.timedelta(seconds=x)))

df['_Weekdays'] = df['Date'].dt.dayofweek
df['_Hours'] = df['Date'].dt.hour
df['_Days'] = df['Date'].dt.day
df['_Month'] = df['Date'].dt.month
```

```
df.drop('TransactionDT', axis=1, inplace=True)
```

By Dina Bavli

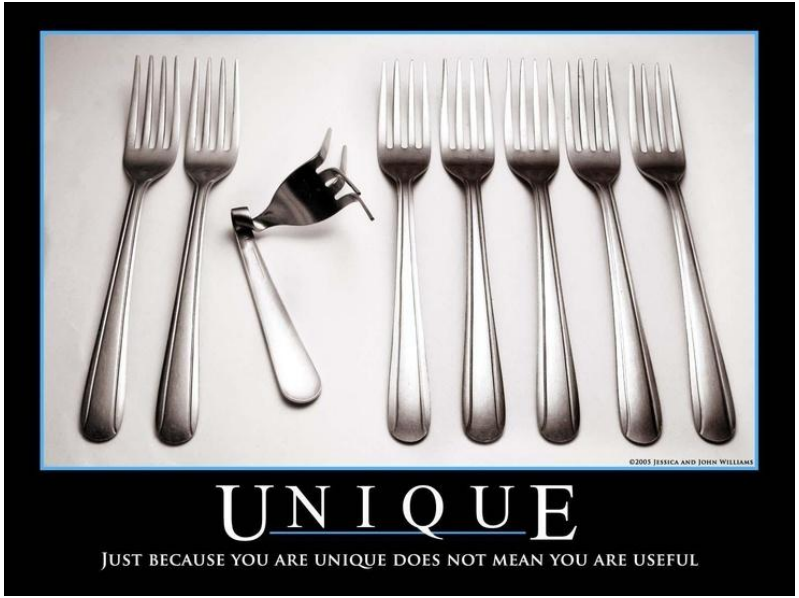
Understanding Cardinality*

.nunique()

.unique()

.value_counts()

*Uniqueness of data values contained in a column



By Dina Bavli

Reducing Cardinality: Why and How Should You Do That?

```
df['device_name'] = df['DeviceInfo'].str.split('/', expand=True)[0]
```

```
print(df["device_name"].nunique())
```

```
1574
```

```
print(set(df['DeviceInfo'].unique()))
```

```
{nan, 'SAMSUNG SM-J700M BUILD/LMY48B', 'KFMEWI', 'SM-A510M BUILD/NRD90M', 'WIN64', 'XT1003', 'SM-G900F BUILD/KOT49H', 'SM-G950W', 'SAMSUNG SM-G950F BUILD/R16NW',
```

By Dina Bavli

Reducing Cardinality- why and how should you do that?

'SAMSUNG SM-J700M BUILD/LMY48B', 'SM-A510M BUILD/NRD90M', 'XT1003', 'SM-G900F BUILD/KOT49H', 'SM-G950W', 'SAMSUNG SM-G950F BUILD/R16NW', 'SM-A710M BUILD/NRD90M'

ALL SM\ SAMSUNG ARE THE SAME DEVICE TYPE- SAMSUNG

By Dina Bavli

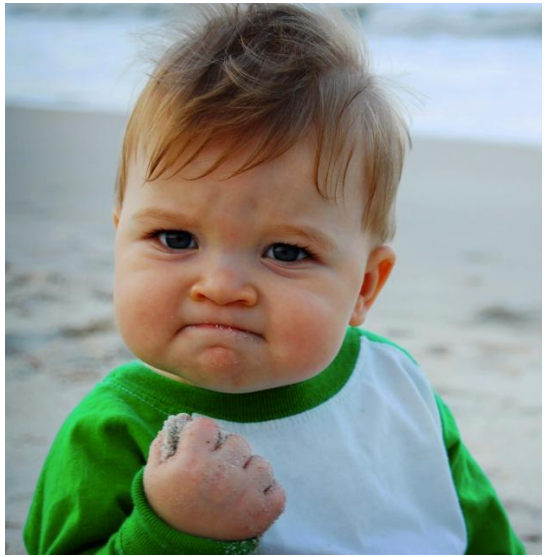
Reducing Cardinality- why and how should you do that?

```
def setDevice(df):

    df['DeviceInfo']=df['DeviceInfo'].str.upper()
    df['device_name'] = df['DeviceInfo'].str.split('/', expand=True)[0]

    df.loc[df['device_name'].str.contains('SM', na=False), 'device_name'] = 'Samsung'
    df.loc[df['device_name'].str.contains('SAMSUNG', na=False), 'device_name'] = 'Samsung'
    df.loc[df['device_name'].str.contains('GT-', na=False), 'device_name'] = 'Samsung'
    df.loc[df['device_name'].str.contains('MOTO G', na=False), 'device_name'] = 'Motorola'
    df.loc[df['device_name'].str.contains('MOTO ', na=False), 'device_name'] = 'Motorola'
    df.loc[df['device_name'].str.contains('LG-', na=False), 'device_name'] = 'LG'
    df.loc[df['device_name'].str.contains('RV:', na=False), 'device_name'] = 'RV'
    df.loc[df['device_name'].str.contains('HUAWEI', na=False), 'device_name'] = 'Huawei'
    df.loc[df['device_name'].str.contains('ALE-', na=False), 'device_name'] = 'Huawei'
    df.loc[df['device_name'].str.contains('-L', na=False), 'device_name'] = 'Huawei'
    df.loc[df['device_name'].str.contains('BLADE', na=False), 'device_name'] = 'ZTE'
    df.loc[df['device_name'].str.contains('LINUX', na=False), 'device_name'] = 'Linux'
    df.loc[df['device_name'].str.contains('XT', na=False), 'device_name'] = 'Sony'
    df.loc[df['device_name'].str.contains('HTC', na=False), 'device_name'] = 'HTC'
    df.loc[df['device_name'].str.contains('ASUS', na=False), 'device_name'] = 'Asus'

    df['had_id'] = 1
    gc.collect()

    return df

df=setDevice(df)
```

Reducing cardinality from 1574 to 687

```
print(df["device_name"].nunique())
```

```
687
```



By Dina Bavli

Reducing Cardinality- why and how should you do that?

```
    df.loc[df['device_name'].str.contains('XT', na=False), 'device_name'] = 'Sony'
    df.loc[df['device_name'].str.contains('HTC', na=False), 'device_name'] = 'HTC'
    df.loc[df['device_name'].str.contains('ASUS', na=False), 'device_name'] = 'Asus'

    df.loc[df.device_name.isin(df.device_name.value_counts()[df.device_name.value_counts() < 200].index), 'device_name'] = "Others"
    df['had_id'] = 1
    gc.collect()

    return df

df=setDevice(df)
```

By adding this line to the function:

```
print(df["device_name"].unique())
```

```
[nan 'Samsung' 'IOS DEVICE' 'WINDOWS' 'MACOS' 'ZTE' 'Sony' 'Others' 'RV'
 'LG' 'TRIDENT' 'Huawei' 'Motorola' 'HTC']
```

```
print(df["device_name"].nunique())
```

```
13
```

We reduce cardinality to 13

By Dina Bavli

Unique Trick

By Dina Bavli

```
data_describe = data.describe()
data_describe = data_describe.T

df_numeric = data._get_numeric_data()

data_describe['corr_with_target'] = df_numeric.drop("isFraud", axis = 1).apply(lambda x : x.corr(df_numeric.isFraud))
data_describe['dtypes'] = df_numeric.dtypes
data_describe['Missing %'] = df_numeric.isnull().sum()/len(data) * 100
data_describe['Cardinality'] = df_numeric.apply(pd.Series.nunique)
data_describe['Skew'] = df_numeric.skew(axis = 0, skipna = True)
Zscore, pvalue = stats.skewtest(df_numeric, axis = 0, nan_policy = 'omit')
data_describe['Z-score'] = Zscore
data_describe['pvalue'] = pvalue
data_describe['kurtosis'] = kurtosis(df_numeric)
```

By Dina Bavli

Data describe

```
data_describe=data.describe()
```

```
data_describe.round(2).head(10)
```

	isFraud	TransactionAmt	card1	card2	card3	card5	addr1	addr2	dist1
count	590540.00	590540.00	590540.00	581607.00	588975.00	586281.00	524834.00	524834.00	238269.00
mean	0.03	135.03	9898.73	362.56	153.19	199.28	290.73	86.80	118.50
std	0.18	239.16	4901.17	157.79	11.34	41.24	101.74	2.69	371.87
min	0.00	0.25	1000.00	100.00	100.00	100.00	100.00	10.00	0.00
25%	0.00	43.32	6019.00	214.00	150.00	166.00	204.00	87.00	3.00
50%	0.00	68.77	9678.00	361.00	150.00	226.00	299.00	87.00	8.00
75%	0.00	125.00	14184.00	512.00	150.00	226.00	330.00	87.00	24.00
max	1.00	31937.39	18396.00	600.00	231.00	237.00	540.00	102.00	10286.00

8 rows × 406 columns

```
data_describe = data_describe.T
```

```
data_describe.head()
```

	count	mean	std	min	25%	50%	75%	max
isFraud	590540.0	0.034990	0.183755	0.000	0.000	0.000	0.0	1.000
TransactionAmt	590540.0	135.027176	239.162522	0.251	43.321	68.769	125.0	31937.391
card1	590540.0	9898.734658	4901.170153	1000.000	6019.000	9678.000	14184.0	18396.000
card2	581607.0	362.555488	157.793246	100.000	214.000	361.000	512.0	600.000
card3	588975.0	153.194925	11.336444	100.000	150.000	150.000	150.0	231.000

data_describe=data.describe()

data_describe = data_describe.T

By Dina Bavli

```
data_describe = data.describe()
data_describe = data_describe.T

df_numeric = data._get_numeric_data()

data_describe['corr_with_target'] = df_numeric.drop("isFraud", axis = 1).apply(lambda x : x.corr(df_numeric.isFraud))
data_describe['dtypes'] = df_numeric.dtypes
data_describe['Missing %'] = df_numeric.isnull().sum()/len(data) * 100
data_describe['Cardinality'] = df_numeric.apply(pd.Series.nunique)
data_describe['Skew'] = df_numeric.skew(axis = 0, skipna = True)
Zscore, pvalue = stats.skewtest(df_numeric, axis = 0, nan_policy = 'omit')
data_describe['Z-score'] = Zscore
data_describe['pvalue'] = pvalue
data_describe['kurtosis'] = kurtosis(df_numeric)
```

By Dina Bavli

Z-score

```
data_describe.round(2)
```

	count	mean	std	min	25%	50%	75%	max	corr_with_target	dtypes	Missing %	Cardinality	Skew	skewness	pvalue	outliers	kurtosis
TransactionID	590540.0	3282269.50	170474.36	2987000.00	3134634.75	3282269.50	3429904.25	3577539.00	0.01	int64	0.00	590540	-0.00	-0.00	1.00	0	-1.20
isFraud	590540.0	0.03	0.18	0.00	0.00	0.00	0.00	1.00	NaN	int64	0.00	2	5.06	646.53	0.00	20663	23.62
TransactionDT	590540.0	7372311.31	4617223.65	86400.00	3027057.75	7306527.50	11246620.00	15811131.00	0.01	int64	0.00	573349	0.13	40.97	0.00	0	-1.23
TransactionAmt	590540.0	135.03	239.16	0.25	43.32	68.77	125.00	31937.39	0.01	float64	0.00	20902	14.37	912.48	0.00	10093	1123.95
card1	590540.0	9898.73	4901.17	1000.00	6019.00	9678.00	14184.00	18396.00	-0.01	int64	0.00	13553	-0.04	-12.84	0.00	0	-1.14
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
id_22	5169.0	16.00	6.90	10.00	14.00	14.00	14.00	44.00	0.12	float64	99.12	25	3.23	50.08	0.00	336	NaN
id_24	4747.0	12.80	2.37	11.00	11.00	11.00	15.00	26.00	-0.00	float64	99.20	12	1.29	28.56	0.00	61	NaN
id_25	5132.0	329.61	97.46	100.00	321.00	321.00	371.00	548.00	0.03	float64	99.13	341	0.05	1.34	0.18	0	NaN
id_26	5163.0	149.07	32.10	100.00	119.00	149.00	169.00	216.00	0.10	float64	99.13	95	0.01	0.22	0.83	0	NaN
id_32	77586.0	26.51	3.74	0.00	24.00	24.00	32.00	32.00	0.07	float64	86.86	4	0.74	75.63	0.00	6	NaN

403 rows × 17 columns

By Dina Bavli

By Dina Bavli

Before_train1.ipynb

https://bit.ly/3aeDhsK

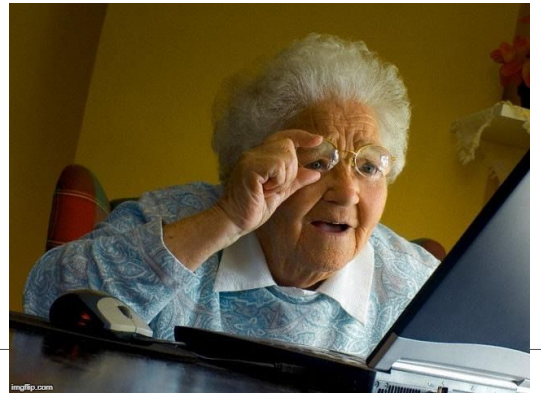

HANDLE
WITH CARE

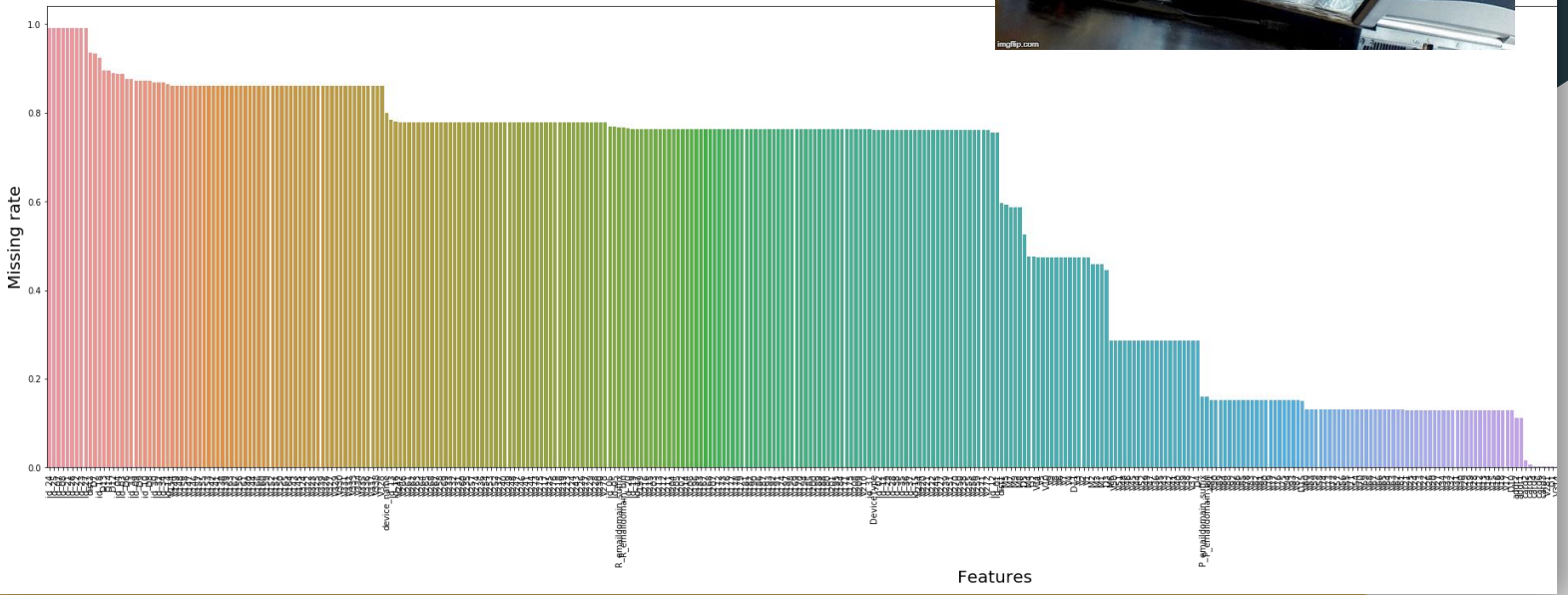
We will cover

Methods for dealing with:

1. Missing data
2. Transformations
3. Outliers

Missing Visualization



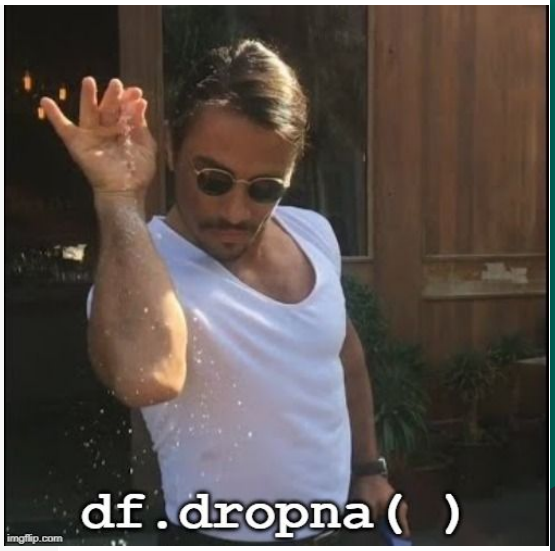


By Dina Bavli

Methods for Handling Missing Data

1. Dropping missing data
2. Knowing it's missing
3. Fill with 'NaN' and '0'
4. Forward and Back -Fill
5. Fill Nulls with Mode and Mean
6. Fill Nulls by Distribution
7. Handle Nulls with Interpolate

```
# Drop the rows where at least one element is missing.
df.dropna()
# Drop the columns where at least one element is missing.
df.dropna(axis='columns')
# Drop the rows where all elements are missing.
df.dropna(how='all')
# Keep only the rows with at least 2 non-NA values.
df.dropna(thresh=2)
# Define in which columns to look for missing values.
df.dropna(subset=['name', 'born'])
# Keep the DataFrame with valid entries in the same variable.
df.dropna(inplace=True)
```




```
missing_rat = [col for col in df.columns if df[col].isnull().sum() /
               df.shape[0] > 0.8]
```

```
cols_to_drop = list(missing_rat)
null_drop = df[df.columns.difference(cols_to_drop)]
```

By Dina Bavli

Initial filtering by criteria

```
corr_with_target = df_numeric.drop("isFraud", axis = 1).apply(lambda x : x.corr(df_numeric.isFraud))

corr_target = abs(corr_with_target[corr_with_target<.1])

missing_rat = [col for col in data.columns if data[col].isnull().sum() / data.shape[0] > 0.7]

a = set(corr_target.keys())

b = set(missing_rat)

result = a.intersection(b)

cols_to_drop = list(result)

new_df = df[df.columns.difference(cols_to_drop)]
```

By Dina Bavli

A house not having a pool will affect the target price.

By Dina Bavli

```
col_name = df.columns #Columns Names
df2 = df.isnull() #New datafram- is null? true or false?

# New names for the new columns:
new_col = [col.replace(col, col+str(' isNAN')) for col in col_name]

df2.columns = new_col #Naming the new columns
```

```
drop_isNAN = df2.join(df, how='outer') #Joining the columns
```

Fill with 'NaN' and '0'

```
for col in object_cols:
    train2[col] = train2[col].fillna('NaN')
```

```
for col in numeric_cols:
    train2[col] = train2[col].fillna(0)
```

By Dina Bavli

Fill Nulls with Mode and Mean

```
for col in object_cols:
  df[col].fillna(df[col].mode()[0], inplace = True)
```

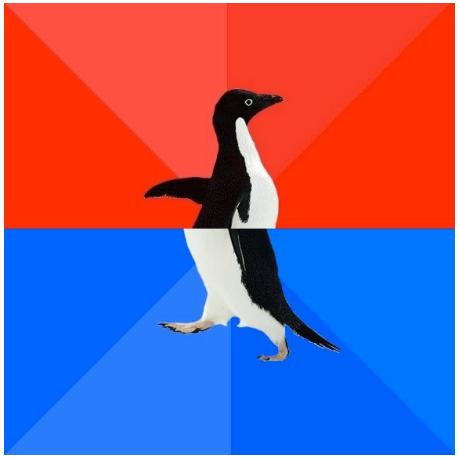
```
for col in int_cols:
  value_to_fill  =round(df[col].mean())
  #print(col,value_to_fill)
  df[col].fillna(value_to_fill, inplace = True)
```

```
for col in float_cols:
  value_to_fill = df[col].mean()
  #print(col,value_to_fill)
  df[col].fillna(value_to_fill,inplace = True)
```

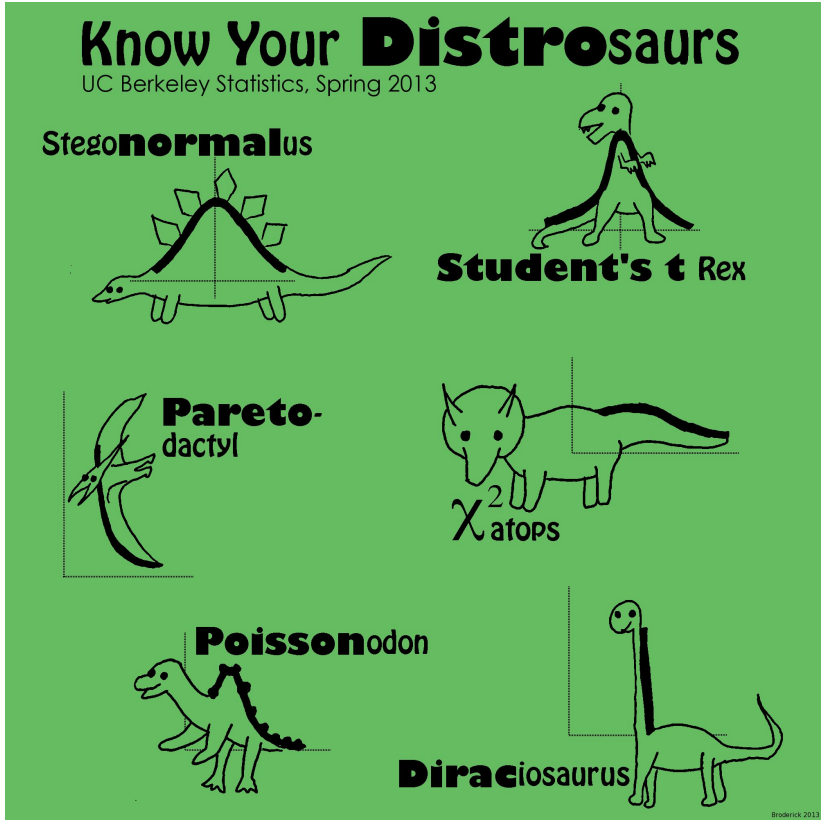
By Dina Bavli

Forward and Back -Fill

```
# forward-fill
df.fillna(method = 'ffill')
# back-fill
df.fillna(method = 'bfill')
```



Fill Nulls by Distribution



By Dina Bavli

Fill Nulls by Distribution

```
for col in int_cols:
  values=df_integers[col].unique().tolist()
  values__probs = df_integers.groupby(col).size().div(len(df_integers))
  randomnumber = choice(values, p=values__probs)
  df[col] = df[col].fillna(randomnumber)
```

```
for col in float_cols:
    vals=pd.Series(df[col][df[col].notnull()])
    density = scipy.stats.gaussian_kde(vals)
    df_nan=df.loc[(df[col].isnull())]
    indexes=set(df_nan.index)
    sample=density.resample(len(indexes)).T[:,0]
    values_list=random.sample(set(pd.Series(sample)),k=len(indexes))
    values_to_fill={}
    n=0;
    for i in indexes:
        values_to_fill[i]=values_list[n]
        n+=1

    df[col] = df[col].fillna(value=values_to_fill)
```

Wider Look on Missing Data

Let's say we are looking at an income report.

The salary column has missing data.

How should we handle it?

Isn’t the salary affected by the education level?

An undergraduate will earn as much as a Ph.D.?

Affected by the field of education? STEM vs social science?

Affected by the profession? High-tech vs education?

Affected by location? And even by sex?

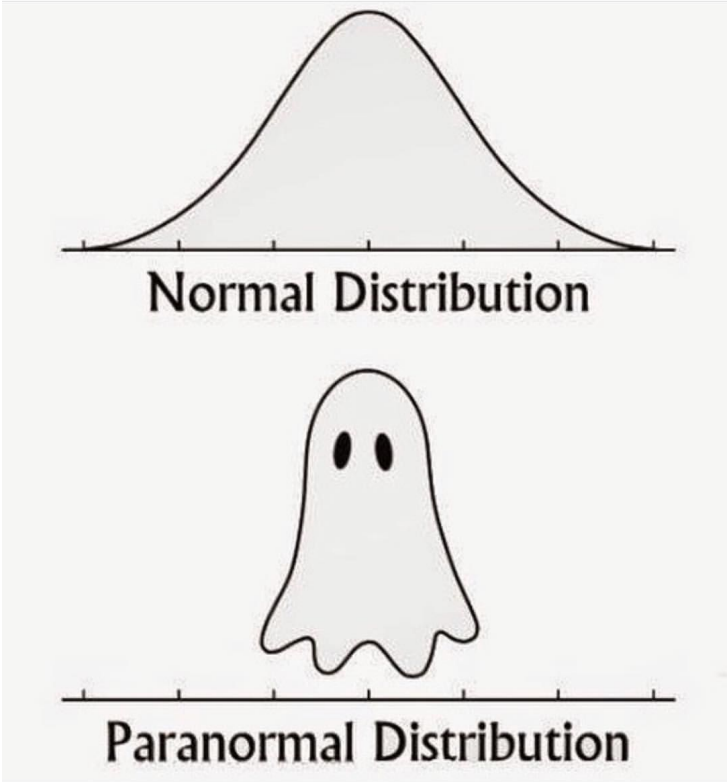
Handle Nulls with Interpolate

	A	B	C	D
0	12.0	NaN	20.0	14.0
1	4.0	2.0	16.0	3.0
2	5.0	54.0	NaN	NaN
3	NaN	3.0	3.0	NaN
4	1.0	NaN	8.0	6.0

	A	B	C	D
0	12.0	NaN	20.0	14.0
1	4.0	2.0	16.0	3.0
2	5.0	54.0	9.5	4.0
3	3.0	3.0	3.0	5.0
4	1.0	3.0	8.0	6.0

By Dina Bavli

Performing Data Transformation



```
# return all columns with cardinality more than threshold and pass the skew test
def get_cols_for_skew_test(df,threshold):
  cols_for_skew_test = df_describe[df_describe['Cardinality']>threshold].index
  print('There are ' + str(len(cols_for_skew_test)) +
        ' columns with cardinality higher than '+ str(threshold))

  is_skew_cols=set()
  for col in cols_for_skew_test:
    skewness, pvalue = stats.skewtest(df_numeric[col],
                                      axis = 0, nan_policy = 'omit')
    if pvalue<0.01:
      is_skew_cols.add(col)

  print('Out of them, there are ' + str(len(is_skew_cols)) +
        ' passed the skew test')
  print(list(is_skew_cols))
  return set(is_skew_cols)
```

Is Skew?

Transform Positive Skew

```
pos_skew_cols = skew_cols.intersection(set(df_describe
                                           [df_describe['Skewtest']>0].index))
print(len(pos_skew_cols))
print(sorted(pos_skew_cols))
```

```
#This loop is taking time... please be patient :)
for col in pos_skew_cols:
  #print(col)
  min_val = df_describe['min'][col]
  #print('min: ' + str(min_val))
  df[col]=df[col].apply(lambda x: np.log(x+abs(min_val)+0.001))
```

Transform Negative Skew

```
neg_skew_cols = skew_cols.intersection(set(df_describe
                                           [df_describe['Skewtest']<0].index))
print(len(neg_skew_cols))
print(sorted(neg_skew_cols))
```

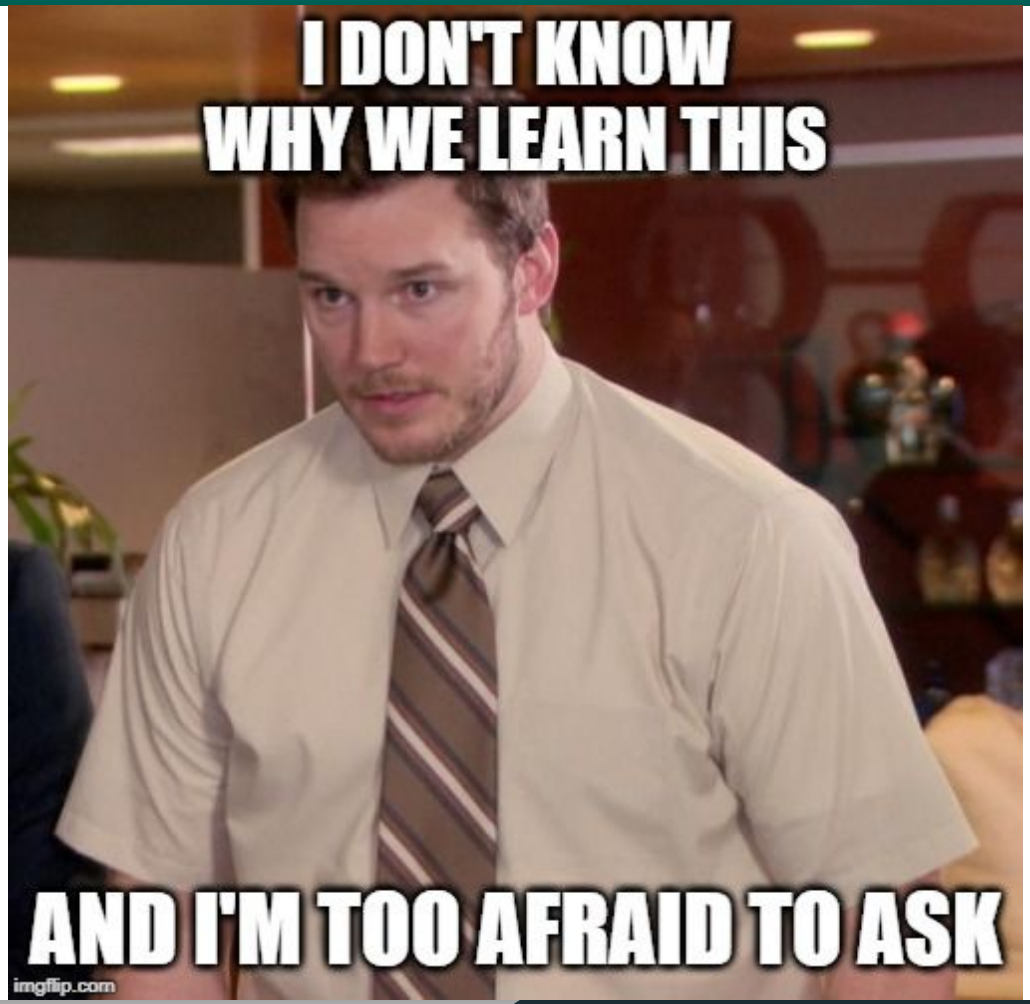
```
df[list(neg_skew_cols)]=df[list(neg_skew_cols)].apply(lambda x: x*x)
```

Removing Outliers

```
def remove_outlier(df_in, col_name):
    q1 = df_in[col_name].quantile(0.05)
    q3 = df_in[col_name].quantile(0.95)
    iqr = q3-q1 #Interquartile range
    fence_low  = q1-1.5*iqr
    fence_high = q3+1.5*iqr

    if iqr>0:
      kurt_value=kurtosis(df_in[col_name])
      if(abs(kurt_value)<4):
        statistic,p_value = kurtosistest(df_in[col_name])
        df_out = df_in.loc[(df_in[col_name] < fence_low) |
                           (df_in[col_name] > fence_high)]

        return set(df_out.index)
    return set()
```

By Dina Bavli

By Dina Bavli



By Dina Bavli

By Dina Bavli

If you torture the data long enough,
it will confess.
Ronald Coase

By Dina Bavli

https://www.linkedin.com/in/dina-bavli-502430158/ https://github.com/dinbav

By Dina Bavli

Start playing the data

Now you :)

Before_train1.ipynb

https://bit.ly/3aeDhsK

Before_train2.ipynb

https://bit.ly/37ZapmE