Machine Learning Useful Tips

Disclaimer: the material is from insights, instructions from multiple DS SMEs

- **How to manipulate Dataframe for Data cleaning**
  change long row into ML friendly multiple columns

Method5: Use melt() built-in function

In [94]:
```
# df1=df['product_type'].apply(lambda x: str(x).split(','))
# df1[1]
df.head(2)
```

Out[94]:

	cost	price	weight	purchase_date	product_type	product_level	maker	ingredient	height	width	depth
0	$333k	$300,492	3 Ton 90 Kg	Dec 19 2008	Q,B	advanced	M14122	IN732052,IN732053	2.76 meters	97 cm	26 cm
1	NaN	$430,570	3 Ton 30 Kg	Sep 10 1997	J,D	basic	NaN	IN732054,IN732055,IN732056,IN732057,IN732058	2.67 meters	98 cm	26 cm

In [102]:
```
df1=df.melt(id_vars=['product_level', 'maker','product_type'],var_name=['allotherfeatures'])
# df_relinc=df_relinc.melt(id_vars=["religion"],var_name=["income"],value_name="frequency")
```

In [107]:
```
df1
```

Out[107]:

	product_level	maker	product_type	allotherfeatures	value
0	advanced	M14122	Q,B	cost	$333k
1	basic	NaN	J,D	cost	NaN
2	basic	NaN	J,D	cost	$270k
3	advanced	M14123	U	cost	NaN
4	advanced	NaN	D,R	cost	$97k
...	...	...	...	...	...
25571	advanced	M14904	D	depth	29 cm
25572	intermediate	M14578	J,B	depth	28 cm
25573	intermediate	M14883,M15011	C	depth	25 cm
25574	advanced	M14341	R	depth	26 cm
25575	super advanced	NaN	N	depth	24 cm

25576 rows × 5 columns

Method 3: Use Explode built-in function

In [83]:
```
df['product_type'].map(lambda x:str(x).split(',')).explode('product_type').value_counts()
# .explode('product_type')
```

Out[83]:
```
D    1100
H     647
A     500
U     395
N     331
R     245
B     218
C     203
J     196
S     179
F     146
K     111
T      93
P      92
O      90
```

- **Difference between Apply/Map/ApplyMap**

https://towardsdatascience.com/introduction-to-pandas-apply-applymap-and-map-5d3e044e93ff

- **Pandas Time Series**

https://pandas.pydata.org/docs/user_guide/timeseries.html#timeseries-offset-aliases

what is rolling (moving) average?

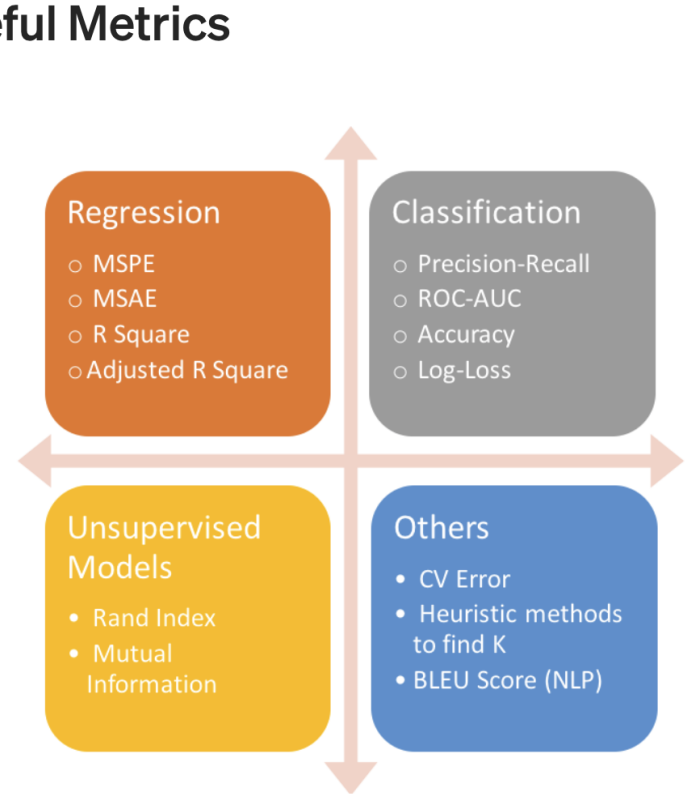
Month	Sales	Moving Average
Jan-08	280	
Feb-08	356	
Mar-08	486	374
Apr-08	603	482
May-08	737	609
Jun-08	815	718
Jul-08	882	811
Aug-08	907	868
Sep-08	952	914
Oct-08	1004	954
Nov-08	1087	1014
Dec-08	1090	1060

two steps:

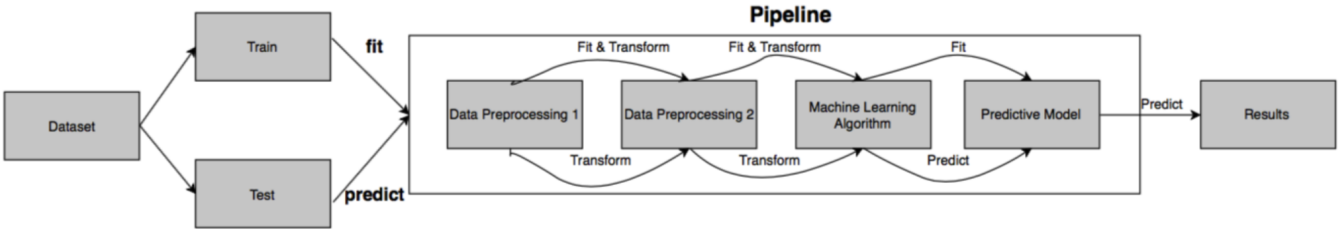
1. list the profit of each day, ordered and aggregated (could have multiple sales in one day) - resample function
2. calculate rolling average - rolling function

- **How to Choose Metrics**

https://medium.com/usf-msds/choosing-the-right-metric-for-machine-learning-models-part-1-a99d7d7414e4
https://towardsdatascience.com/what-are-the-best-metrics-to-evaluate-your-regression-model-418ca481755b



- **About Pipeline Concept**



https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.PolynomialFeatures.html

https://hub.packtpub.com/automl-build-machine-learning-pipeline-tutorial/

- # window = '7D'
  profit.resample('D').sum().rolling('7D').mean()

In [45]:
```
# window = 7
profit.resample('D').sum().rolling(7).mean()
```

Out[45]:
```
purchase_date
1996-08-09              NaN
1996-08-10              NaN
1996-08-11              NaN
1996-08-12              NaN
1996-08-13              NaN
                        ...
2017-10-02     12870.285714
2017-10-03     12870.285714
2017-10-04    -18644.000000
2017-10-05    -18644.000000
2017-10-06    -21481.285714
Length: 7729, dtype: float64
```

In [46]:
```
# window = '7D'
profit.resample('D').sum().rolling('7D').mean()
```

Out[46]:
```
purchase_date
1996-08-09         0.000000
1996-08-10         0.000000
1996-08-11         0.000000
1996-08-12         0.000000
1996-08-13         0.000000
                        ...
2017-10-02     12870.285714
2017-10-03     12870.285714
2017-10-04    -18644.000000
2017-10-05    -18644.000000
2017-10-06    -21481.285714
Length: 7729, dtype: float64
```

Question: ***why rolling 7 and '7D' have different head rows?***

https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.rolling.html

min_periods

- https://www.dataquest.io/blog/settingwithcopywarning/

\

- **How to download Github repo from Github**

1350

Update Apr. 2021: there are a few tools created by the community that can do this for you:

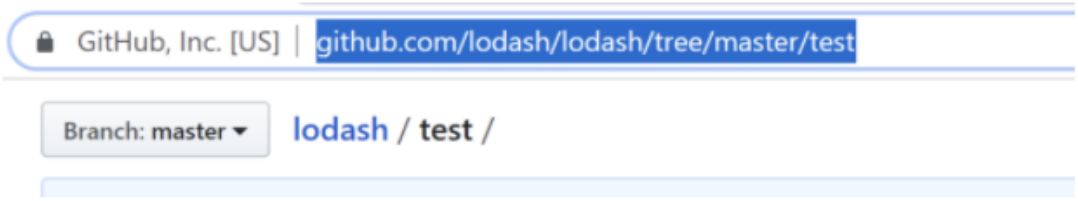
- Download Directory (Credits to fregante)
  - It has also been integrated into the excellent **Refined Github** chrome extension as a button in the Github web UI.
- GitZip (Credits to Kino - see his answer **here**)
- DownGit (Credits to Minhas Kamal - see his answer **here**)

Note: if you're trying to download a large number of files, you may need to provide a token to these tools to avoid rate limiting.

Original (manual) approach: Checking out an individual directory is not supported by `git` natively, but Github can do this via SVN. If you checkout your code with subversion, Github will essentially convert the repo from git to subversion on the backend, then serve up the requested directory.

Here's how you can use this feature to download a specific folder. I'll use the popular javascript library `lodash` as an example.

1. **Navigate to the folder you want to download**. Let's download `/test` from `master` branch.



2. **Modify the URL for subversion**. Replace `tree/master` with `trunk`.

   `https://github.com/lodash/lodash/tree/master/test` ➜

   `https://github.com/lodash/lodash/trunk/test`

3. **Download the folder**. Go to the command line and grab the folder with SVN.

```
svn checkout https://github.com/lodash/lodash/trunk/test
```

You might not see any activity immediately because Github takes up to 30 seconds to convert larger repositories, so be patient.

> *Full URL format explanation:*
>
> - If you're interested in `master` branch, use `trunk` instead. So the full path is `trunk/foldername`
> - If you're interested in `foo` branch, use `branches/foo` instead. The full path looks like `branches/foo/foldername`
> - Protip: You can use `svn ls` to see available tags and branches before downloading if you wish

That's all! Github supports more subversion features as well, including support for committing and pushing changes.