Lead_Magnates

April 25, 2018

```
In [130]: import datetime
          import pandas as pd
          import numpy as np
          import matplotlib.pyplot as plt
          from sklearn.model_selection import train_test_split
          from sklearn.preprocessing import LabelEncoder
          from sklearn.svm import SVC

In [131]: df = pd.read_csv('../../data/Lead_Magnates.csv')
          # df['Pipeline Type'].unique()

In [132]: df = pd.DataFrame({'created':df['Created On'],'updated':df['Updated On'],'assigned':df

In [133]: df['created'] = pd.to_datetime(df['created'])
          df['updated'] = pd.to_datetime(df['updated'])
          df['completed'] = (df['updated'] - df['created']).dt.days
          df.head()

Out[133]:       assigned             created        source stage_category  \
          0       Vivek  2016-07-11 23:47:00  Magic Bricks           Lost
          1       Vivek  2016-08-11 00:02:00  Magic Bricks           Lost
          2  Sweety Jain 2016-07-11 17:50:00  Magic Bricks           Lost
          3  Sweety Jain 2016-09-11 17:22:00  Magic Bricks           Lost
          4  Sweety Jain 2016-09-11 17:24:00  Magic Bricks           Lost

                          updated  completed
          0 2016-07-11 23:48:00        0.0
          1 2016-09-11 16:18:00       31.0
          2 2016-09-11 16:48:00       61.0
          3 2016-09-11 17:22:00        0.0
          4 2016-09-11 17:24:00        0.0

In [134]: # remove data whose with null time and negative days
          df = df.drop(df[df.completed < 0].index)
          df = df.drop(df[df.completed.isna()].index)
          df = df.drop('created',axis=1)
          df = df.drop('updated',axis=1)
```

```
In [135]: le_a = LabelEncoder()
          le_a.fit(df['assigned'])
          df['assigned']=le_a.fit_transform(df['assigned'])

          # le_s = LabelEncoder()
          # le_s.fit(df['source'])
          # df['source']=le_s.fit_transform(df['source'])

          le_a = LabelEncoder()
          le_a.fit(df['stage_category'])
          df['stage_category']=le_a.fit_transform(df['stage_category'])
```

```
In [136]: df.tail()
```

```
Out[136]:       assigned            source  stage_category  completed
          2518         1      Magic Bricks               0      171.0
          2519         3      Magic Bricks               0        4.0
          2520         2      Magic Bricks               0        7.0
          2522         2      Social Media               0        3.0
          2523         0  Gujarat Samachar               0       13.0
```

```
In [168]: group = df.groupby(df['source']).
          group_nm = [x for x in group.groups]
          a = pd.DataFrame(group[0])
```

```
        ---------------------------------------------------------------------------

        KeyError                                  Traceback (most recent call last)

        <ipython-input-168-a58eb06bd381> in <module>()
          1 group = df.groupby(df['source'])
          2 group_nm = [x for x in group.groups]
    ----> 3 a = pd.DataFrame(group[0])


        ~/anaconda3/envs/py3/lib/python3.6/site-packages/pandas/core/base.py in __getitem__(self
        257         else:
        258             if key not in self.obj:
    --> 259                 raise KeyError("Column not found: {key}".format(key=key))
        260             return self._gotitem(key, ndim=1)
        261


        KeyError: 'Column not found: 0'
```

```
In [161]: group_nm[i].head()
```

```
    ---------------------------------------------------------------------------

    TypeError                                 Traceback (most recent call last)

    <ipython-input-161-e55be70cd269> in <module>()
----> 1 group_nm[i].head()


    TypeError: list indices must be integers or slices, not DataFrame
```

```
In [113]: model = SVC()

In [119]: model.fit(df['source'].values.reshape(-1,1),df['assigned'].ravel())

Out[119]: SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
            decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
            max_iter=-1, probability=False, random_state=None, shrinking=True,
            tol=0.001, verbose=False)

In [124]: model.predict(4)

Out[124]: array([3])
```