**[On the plague trail: HackerEarth Machine Learning challenge](https://www.hackerearth.com/challenges/competitive/on-the-plague-trail-hackerearth-machine-learning-challenge/)**

**Tools Used –**

1. **Python 3.7**
   1. **Pandas**
   2. **Sklearn**

Requirement file is attached.

**Data Analytics and Preprocessing.**

The first thing to be done is to get data cleaning and preprocessing. I have done it in the data\_explore.ipynb file.

1. Check the type of data in the pandas data frame. I found that most of them is float and intger and only 4 objects are there. That is ID, datetime, WindDir, HiDir

train.info()

1. Convert the datetime to datetime type.
2. Check the correlation in the dependent variables.

train.corr()

1. Check out null values in the date which I found to be none. But in some variables null is not used but some other indicator is used, example in WinDir, HiDir --- is used to show null value.

print(df['WindDir'].value\_counts())

>

SSE 14523

--- 10479

S 6976

SW 5575

WSW 3843

SE 3768

SSW 3008

W 2534

WNW 2410

NW 1992

N 1768

ESE 1436

NNW 1180

E 1060

ENE 970

NNE 491

NE 433

Name: WindDir, dtype: int64

1. And rest is okay with the data.

After going through the data, I have found that the data is very much clean and we are ready to go.

1. First remove all the object type and lets just proceed with the float and integer data.

train=train.select\_dtypes(exclude='object')

test=test.select\_dtypes(exclude='object')

1. Store PA,PB,PC,PD,PE,PF,PG in different-different variable and From the training dataset just remove them. We are ready to train.

PA=train['PA']

PB=train['PB']

PC=train['PC']

PD=train['PD']

PE=train['PE']

PF=train['PF']

PG=train['PG']

train=train.drop(['PA','PB','PC','PD','PE','PF','PG'],axis=1)

**Algorithm**

1. Take the training set as x values and PA as y values. Train the model and predict the result for the test set.

from sklearn import linear\_model

reg = linear\_model.Lasso(alpha=0.1)

reg.fit(train,PA)

PA\_pred=reg.predict(test).astype(int)

1. After doing it attach the actual value of PA with train set and predicted value with test set.

train['PA']=PA

test['PA']=PA\_pred

1. Now train the model with y as PB and repeat the above two steps.

reg = linear\_model.Lasso(alpha=0.1)

reg.fit(train,PB)

PB\_pred=reg.predict(test).astype(int)

train['PB']=PB

test['PB']=PB\_pred

1. Folow this to train and predict all the P’s and then we have our output ready as the result.

df=pd.DataFrame()

df['ID']=ID

df['PA']=PA\_pred

df['PB']=PB\_pred

df['PC']=PC\_pred

df['PD']=PD\_pred

df['PE']=PE\_pred

df['PF']=PF\_pred

df['PG']=PG\_pred

df.to\_csv('output\_lasso.csv',index=None)

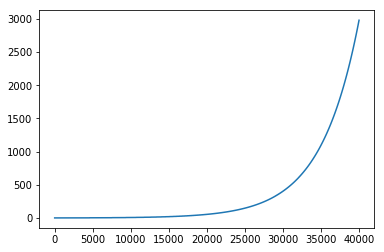
I have tried several models and here are some results.

|  |  |
| --- | --- |
| **Models** | **R2 Value** |
| Decision tree regressor | ~0.80 |
| Linear regression Lasso | ~0.95 |
| Linear regression LassoLars | ~0.94 |
|  |  |

However, I found a simple crack to this problem and it is the pattern the output is following.

Output has the exponential relation with the index and thus participants are able to score 100. But this is not a machine learning approach but just a simple trick with is found just by simple viaualization.

plt.plot(df['PA'])



All the code files are attached and can be found with this doc.

It was a good competition and lot to try over models and algorithms.

Thankyou for giving me and opportunity to participate in competition.