PROJECT REPORT

COURSE NAME: INTRODUCTION TO DATA MINING AND MACHINE LEARNING IN BIOINFORMATICS

COURSE CODE: BIN203

TITLE FOR THE GROUP PROJECT:

EXOPLANET HUNTING USING MACHINE LEARNING

GROUP DETAILS:

Govindhapriya. M – 122013014

Haridha .C - 122013015

Harini. S – 122013016

MOTIVATION AND INTRODUCTION:

All of the planets in our solar system orbit around the Sun. Planets that orbit around other stars are called **exoplanets**.

What are we looking for in anexoplanet:

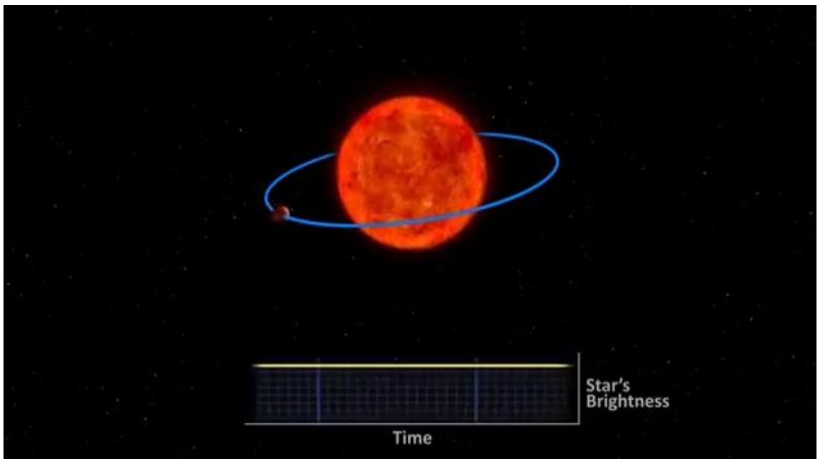
Astronomers are looking for a planet that's the same size as our earth, and which is located at the **habitable zone.**

How do we find them:

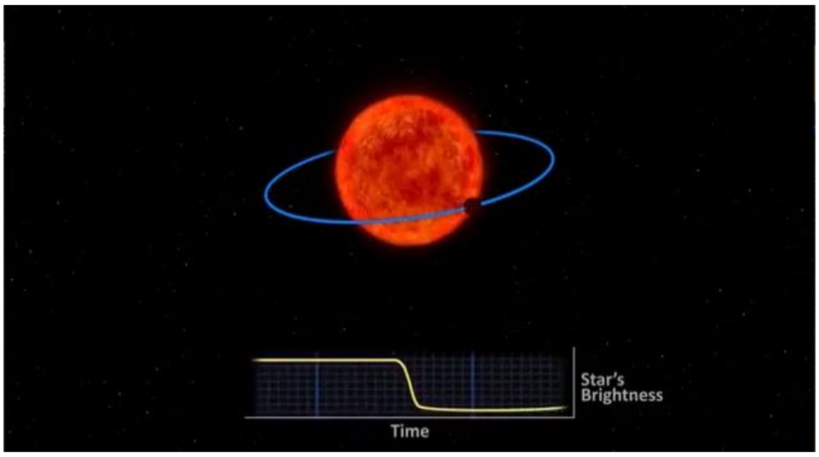
If we were to look at an exoplanet (the nearest one is over 4 light-years away), it would be very close to a brilliantly lit star, making the planet impossible to see.

The transit method:

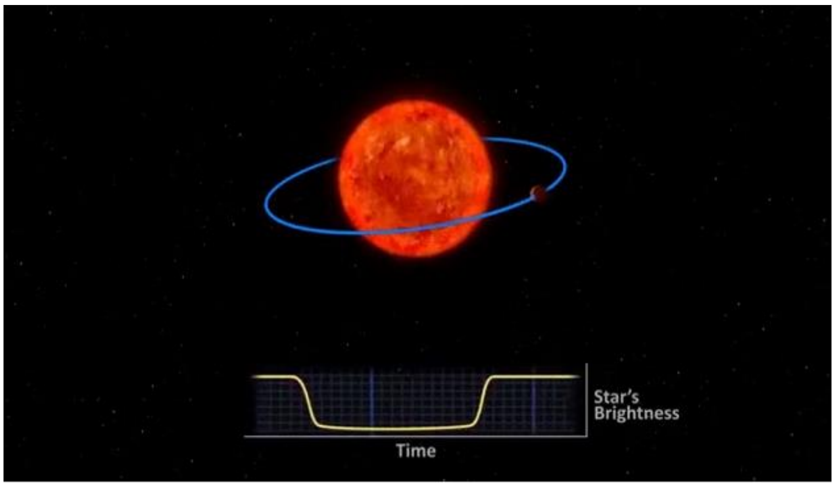
Considering this fact into account scientists at NASA developed a method which they called **Transit method** in which a digital-camera-like technology is used to detect and measure tiny dips in a star's brightness as a planet crosses in front of the star. With observations of transiting planets, astronomers can calculate the ratio of a planet's radius to that of its star — essentially the size of the planet's shadow — and with that ratio, they can calculate the planet's size.



From the graph given in the above diagram,we see that there's no drop in the brightness of the star. This is because the planet is not obscuring the light that's coming from its star.



Here, we can notice a dip in the star's brightness. This is because the starlight is partially obscured by the planet, given our position.



The starlight rises back to its original value once the planets crosses in front of the star.

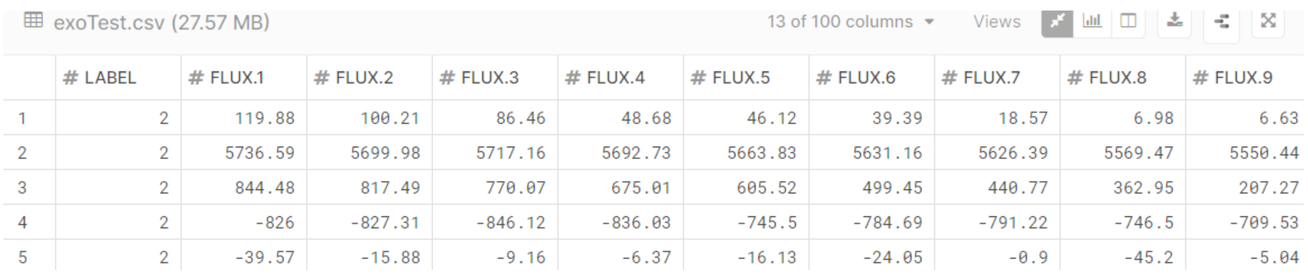
Missions:

Tess missionor transiting exoplanet survey satellite (which replaced the **kepler mission** in 2018)

We were looking for a topic which we all liked in common. That's how we landed in this project. During the phase 1 of this project we did further research on exoplanets and we found this topic very interesting and challenging.

The Dataset:

The dataset was made available by NASA. We got the data from the kaggle website. The first five rows of the dataset can be seen below:


exoTest.csv (27.57 MB)
13 of 100 columns
Views

	# LABEL	# FLUX.1	# FLUX.2	# FLUX.3	# FLUX.4	# FLUX.5	# FLUX.6	# FLUX.7	# FLUX.8	# FLUX.9
1	2	119.88	100.21	86.46	48.68	46.12	39.39	18.57	6.98	6.63
2	2	5736.59	5699.98	5717.16	5692.73	5663.83	5631.16	5626.39	5569.47	5550.44
3	2	844.48	817.49	770.07	675.01	605.52	499.45	440.77	362.95	207.27
4	2	-826	-827.31	-846.12	-836.03	-745.5	-784.69	-791.22	-746.5	-709.53
5	2	-39.57	-15.88	-9.16	-6.37	-16.13	-24.05	-0.9	-45.2	-5.04

Our idea:

We wanted to see if we could look at the available exoplanet data and make predictions about which planets might be hospitable to life. The data made publicly available by NASA is beautiful in that it contains many useful features. The goal is to create a model that can predict the existence of an Exoplanet, utilizing the flux (light intensity) readings from 3198 different stars over time.

OBJECTIVES:

- ✓ To apply the machine learning concepts studied in this course
- ✓ The problem that we have chosen is a **supervised learning** problem, it is a **classification problem**.
- ✓ Our idea is to build model using different classification algorithms and evaluating our models using various evaluation metrics and find the best model.
- ✓ To effectively use preprocessing methods such as **Normalization and PCA-Principle Component Analysis.**
- ✓ To build a proper machine learning model.

METHODS:

Introduction to the dataset:

Description

Trainset:

- 5087 rows or observations.
- 3198 columns or features.
- Column 1 is the label vector. Columns 2 - 3198 are the flux values over time.
- 37 confirmed exoplanet-stars and 5050 non-exoplanet-stars.

Testset:

- 570 rows or observations.
- 3198 columns or features.
- Column 1 is the label vector. Columns 2 - 3198 are the flux values over time.
- 5 confirmed exoplanet-stars and 565 non-exoplanet-stars.

The data describe the change in flux (light intensity) of several thousand stars. *Each star has a binary label of 2 or 1****. 2 indicates that that the star is confirmed to have at least one exoplanet in orbit;*** some observations are in fact multi-planet systems.

As you can imagine, planets themselves do not emit light, but the stars that they orbit do. If said star is watched over several months or years, there may be a regular 'dimming' of the flux (the light intensity). This is evidence that there may be an orbiting body around the star; such a star could be considered to be a 'candidate' system. Further study of our candidate system, for example by a satellite that captures light at a different wavelength, could solidify the belief that the candidate can in fact be 'confirmed'.

Methods used:

We have implemented our model using python programing language

For preprocessing:

i. Normalization
ii. Principle Component Analysis
iii. Synthetic minority oversampling technique(SMOTE)

For building the model:

i. SMV algorithm
ii. Random forest algorithm

For evaluating the model:

i. Accuracy score
ii. Confusion matrix
iii. Precision , recall, F1

1) We started by importing all the necessary libraries:

```
import os
import warnings
import math
warnings.filterwarnings('ignore')
```

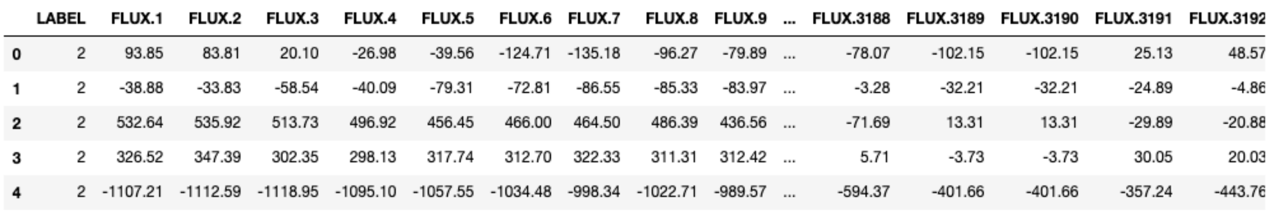
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
from pylab import rcParams
rcParams['figure.figsize'] = 10, 6
from sklearn.metrics import mean_squared_error, mean_absolute_error
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split
from sklearn import linear_model
from sklearn.model_selection import cross_val_score
from sklearn.metrics import precision_score,
recall_score,roc_curve,auc, f1_score, roc_auc_score,confusion_matrix,
accuracy_score, classification_report
from sklearn.preprocessing import StandardScaler, normalize
from scipy import ndimage
import seaborn as sns
```

2) Reading the test and train dataset:

```
test_data = pd.read_csv('exoTest.csv').fillna(0)
train_data = pd.read_csv('exoTrain.csv').fillna(0)
```

3) Displaying the first five rows of the train dataframe:

```
train_data.head()
```



	LABEL	FLUX.1	FLUX.2	FLUX.3	FLUX.4	FLUX.5	FLUX.6	FLUX.7	FLUX.8	FLUX.9	...	FLUX.3188	FLUX.3189	FLUX.3190	FLUX.3191	FLUX.3192
0	2	93.85	83.81	20.10	-26.98	-39.56	-124.71	-135.18	-96.27	-79.89	...	-78.07	-102.15	-102.15	25.13	48.57
1	2	-38.88	-33.83	-58.54	-40.09	-79.31	-72.81	-86.55	-85.33	-83.97	...	-3.28	-32.21	-32.21	-24.89	-4.86
2	2	532.64	535.92	513.73	496.92	456.45	466.00	464.50	486.39	436.56	...	-71.69	13.31	13.31	-29.89	-20.88
3	2	326.52	347.39	302.35	298.13	317.74	312.70	322.33	311.31	312.42	...	5.71	-3.73	-3.73	30.05	20.03
4	2	-1107.21	-1112.59	-1118.95	-1095.10	-1057.55	-1034.48	-998.34	-1022.71	-989.57	...	-594.37	-401.66	-401.66	-357.24	-443.76

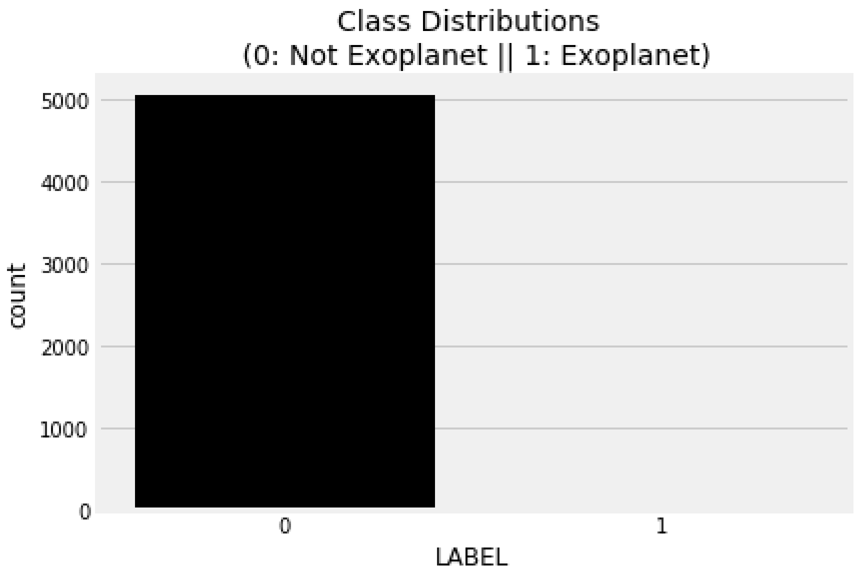
4)Now the target column LABEL consists of two categories **1**(Does not represents exoplanet) and **2**(represents the presence of exoplanet). So, we convert them to binary values for easier processing of data:

```
categ = {2: 1,1: 0}
train_data.LABEL = [categ[item] for item in train_data.LABEL]
test_data.LABEL = [categ[item] for item in test_data.LABEL]
```

5) Since the data is too large, for memory optimization we performed a memory reduction, this step reduced the memory usage of the train_datadataframe by 55.1%

6) We then visualized the target column in the train dataset to get an idea about the class distribution

```
plt.figure(figsize=(6,4))
colors = ["0", "1"]
sns.countplot('LABEL', data=train_data, palette=colors)
plt.title('Class Distributions \n (0: Not Exoplanet || 1: Exoplanet)',
fontsize=14)
```



We can see that the data is highly imbalanced.

So we performed pre-processing steps on the dataset.

7) So, we split the dataset and we normalized them:

```
x_train = train_data.drop(["LABEL"],axis=1)
y_train = train_data["LABEL"]
x_test = test_data.drop(["LABEL"],axis=1)
y_test = test_data["LABEL"]


x_train = normalized = normalize(x_train)
x_test = normalize(x_test)
```

8) The number of columns/features that we have been working with is huge. We have 5087 rows and 3198 columns in our training dataset. Basically we

need to decrease the number of features(Dimensionality Reduction) to remove the possibility of Curse of Dimensionality**.**

For reducing the number of dimensions/features we will use the most popular dimensionality reduction algorithm PCA(Principal Component Analysis)**.**

Choosing the number of features:

```
#Dimentioanlity reduction
from sklearn.decomposition import PCA
pca = PCA()
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)
total=sum(pca.explained_variance_)
k=0
current_variance=0
while current_variance/total < 0.90:
current_variance += pca.explained_variance_[k]
k=k+1
```

The above code gives **k=37.**

We take k=37 and apply PCA on our independent variables.

```
pca = PCA(n_components=37)
x_train = pca.fit_transform(x_train)
x_test = pca.transform(x_test)
plt.figure()
plt.plot(np.cumsum(pca.explained_variance_ratio_))
plt.xlabel('Number of Components')
plt.ylabel('Variance (%)') #for each component
plt.title('Exoplanet Dataset Explained Variance')
plt.show()
```

9) Since the class is not equally distributed, we resampled the data so the target class is equally distributed.

We have used the SMOTE(Synthetic Minority Over-sampling TEchnique)

```
sm = SMOTE(random_state=27, ratio = 1.0)
x_train_res, y_train_res = sm.fit_sample(x_train, y_train.ravel())
```

```
Before OverSampling, counts of label 1: 37
Before OverSampling, counts of label 0:5050

After OverSampling, counts of label 1: 5050
After OverSampling, counts of label 0: 5050
```

10) Building the model:

We created a function model which will:

1. fit the model
2. perform Cross-validation
3. Check the Accuracy of our model
4. generate Classification report
5. generate Confusion matrix

```
def model(classifier,dtrain_x,dtrain_y,dtest_x,dtest_y):
#fit the model
classifier.fit(dtrain_x,dtrain_y)
predictions = classifier.predict(dtest_x)

#Cross validation
accuracies = cross_val_score(estimator = classifier, X = x_train_res, y =
y_train_res, cv = 5, n_jobs = -1)
mean = accuracies.mean()
variance = accuracies.std()
print("Accuracy mean: "+ str(mean))
print("Accuracy variance: "+ str(variance))

#Accuracy
print ("\naccuracy_score :",accuracy_score(dtest_y,predictions))

#Classification report
print ("\nclassification report
:\n",(classification_report(dtest_y,predictions)))

#Confusion matrix
plt.figure(figsize=(13,10))
plt.subplot(221)
sns.heatmap(confusion_matrix(dtest_y,predictions),annot=True,cmap="viridis",
fmt = "d",linecolor="k",linewidths=3)
plt.title("CONFUSION MATRIX",fontsize=20)
```

RESULTS:

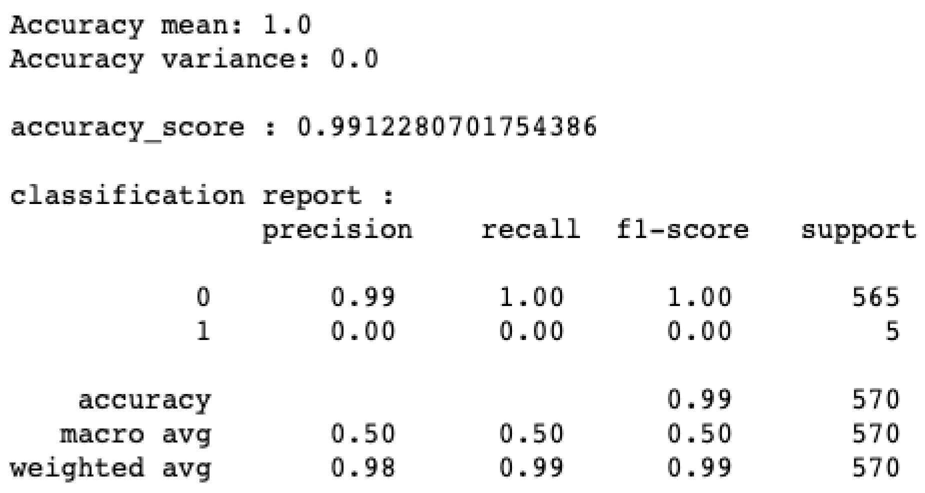
There is always a need to validate the stability of your machine learning model. ***You need some kind of assurance that your model has got most of the patterns from the data correct, and it's not picking up too much on the noise, or in other words its low on bias and variance.***

Since this is a classification problem, we can use SVM and RANDOM FOREST methods

SVM ALGORITHM:

```
fromsklearn.svm import SVC
SVM_model=SVC()
model(SVM_model,x_train_res,y_train_res,x_test,y_test)
```

OUTPUT:



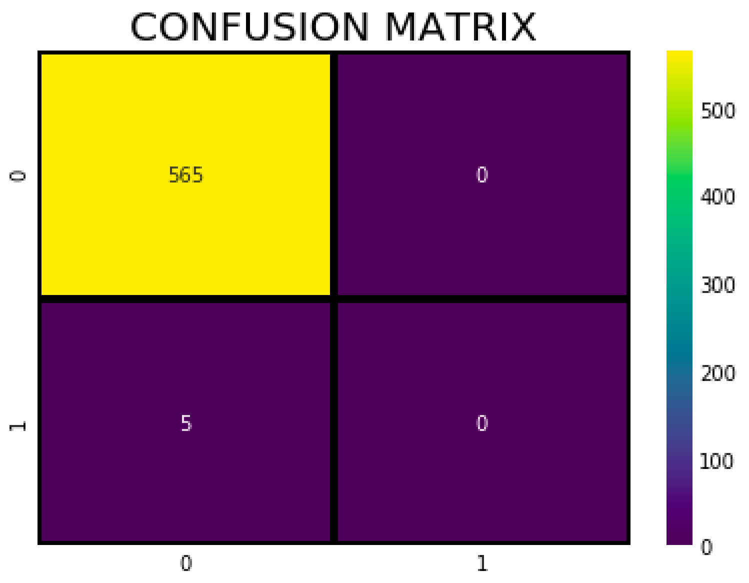
```
Accuracy mean: 1.0
Accuracy variance: 0.0

accuracy_score : 0.9912280701754386

classification report :
              precision    recall  f1-score   support

           0       0.99      1.00      1.00       565
           1       0.00      0.00      0.00         5

    accuracy                           0.99       570
   macro avg       0.50      0.50      0.50       570
weighted avg       0.98      0.99      0.99       570
```



RANDOM FOREST MODEL:

```
fromsklearn.ensemble import RandomForestClassifier
rf_classifier = RandomForestClassifier()
model(rf_classifier,x_train_res,y_train_res,x_test,y_test)
```

OUTPUT:

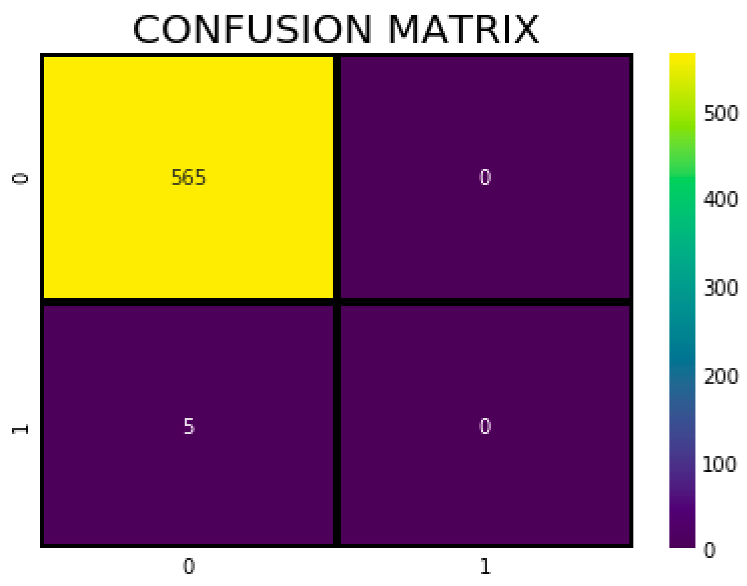
```
Accuracy mean: 1.0
Accuracy variance: 0.0

accuracy_score : 0.9912280701754386

classification report :
               precision    recall  f1-score   support

           0       0.99      1.00      1.00       565
           1       0.00      0.00      0.00         5

    accuracy                           0.99       570
   macro avg       0.50      0.50      0.50       570
weighted avg       0.98      0.99      0.99       570
```



DISCUSSIONS AND CONCLUSION:

Comparing both the methods, we are getting pretty good results from them and t is seem that in SVM there is zero accuracy variance and accuracy mean is 1.0

However the accuracy score is same.

Within the next 10 years, 30 to 40m diameter telescopes will operate from the Earth to detect exoplanets by imaging and velocity variations of the stars. Large Space telescopes are being designed at NASA to detect signs of life on exoplanets by 2050.

In the more distant future, huge space interferometers will make detailed maps of planets. And possibly, interstellar probes will be launched towards the nearest exoplanets to take close-up images. Engineers are already working on propulsion techniques to reach such distant targets.

REFERENCES:

https://exoplanets.nasa.gov/blog/

https://exoplanets.nasa.gov/what-is-an-exoplanet/about-exoplanets/

https://www.analyticsvidhya.com/blog

https://datatofish.com/python-tutorials/

https://towardsdatascience.com/

https://www.kaggle.com/keplersmachines/kepler-labelled-time-series-data