Problem Statement

Most of the astronomical datasets collected from satellites and telescopes are unbalanced or limited, to perform training and get adequate result through machine learning or deep learning classifiers.

But what is Imbalance Data?

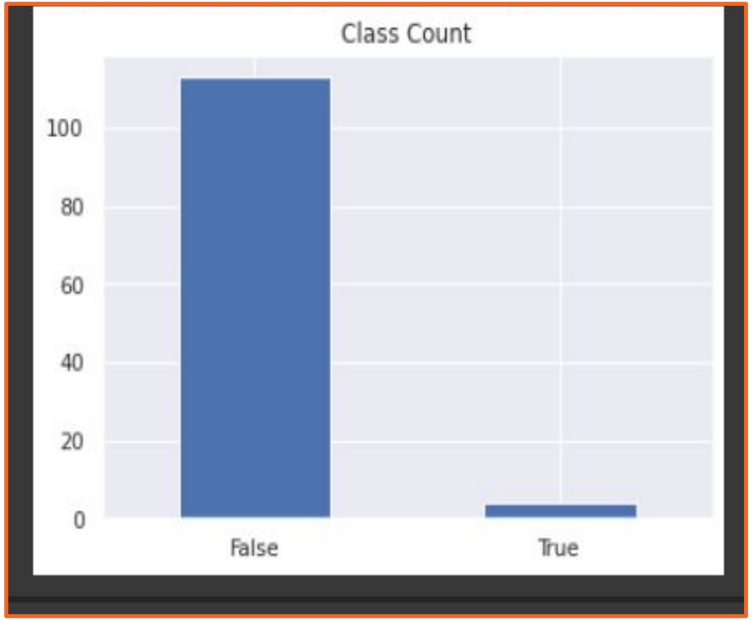
Let's understand this with the help of an example.
Ex: In an exoplanet habitability detection data set has the following data:

Total Observations = 1000

Positive Observations = 10

Negative Observations = 990

Event Rate= 1 % (the probability of finding a habitable exoplanet among all)



Distribution of observations
0 = Negative Class
1 = Positive Class

Problem Statement

Challenges faced by imbalance data with Machine learning

1. Standard classifier algorithms like Decision Tree and Logistic Regression have a bias towards classes which have number of instances.
2. They tend to only predict the majority class data. The features of the minority class are treated as noise and are often ignored. Thus, there is a high probability of misclassification of the minority class as compared to the majority class.
3. While working in an imbalanced domain accuracy is not an appropriate measure to evaluate model performance.
4. For eg: A classifier which achieves an accuracy of 99 % with an event rate of 1 % is not accurate, if it classifies all instances as the majority class. And eliminates the 1 % minority class observations as noise or mis-classifies them.
5. That means the algo would correctly classify majority class but would wrongly classify minority class which makes the model of no use to us in such cases.

Objectives

Analyze | Research | Implement | Compare

Collecting and reviewing astronomical datasets from original sources

Analyzing the datasets for limited or unbalanced data

Research techniques to implement in case of limited data or imbalance dataset

Researching and reviewing research papers to find useful techniques to improve model performance

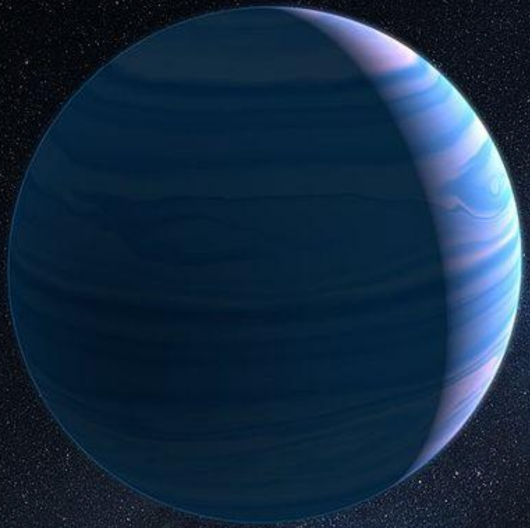
Implement the techniques found useful after the survey

Implementing techniques like Anomaly Detection, SMOTE and Sampling

Compare and check the performance with initial result

Comparing the confusion matrices of different techniques with the models implemented before to check what works better on imbalance data

Analysis using Problem of Habitability Detection



Dataset Description

Taken from [Kaggle]

- ➔ **Main Goal**
  Finding habitable exoplanets in different stellar systems. The dataset contains all modeled parameters for currently confirmed exoplanets along with a few of them which are confirmed to be habitable.
- ➔ **PHL Exoplanet Catalog**
  Exoplanet Habitability Data for different exoplanets
- ➔ **110 Columns**
  Consisting of 110 different physical parameter features for exoplanets like Planet Density, Mass, Temperature, Orbital velocity, Transit duration and many more.

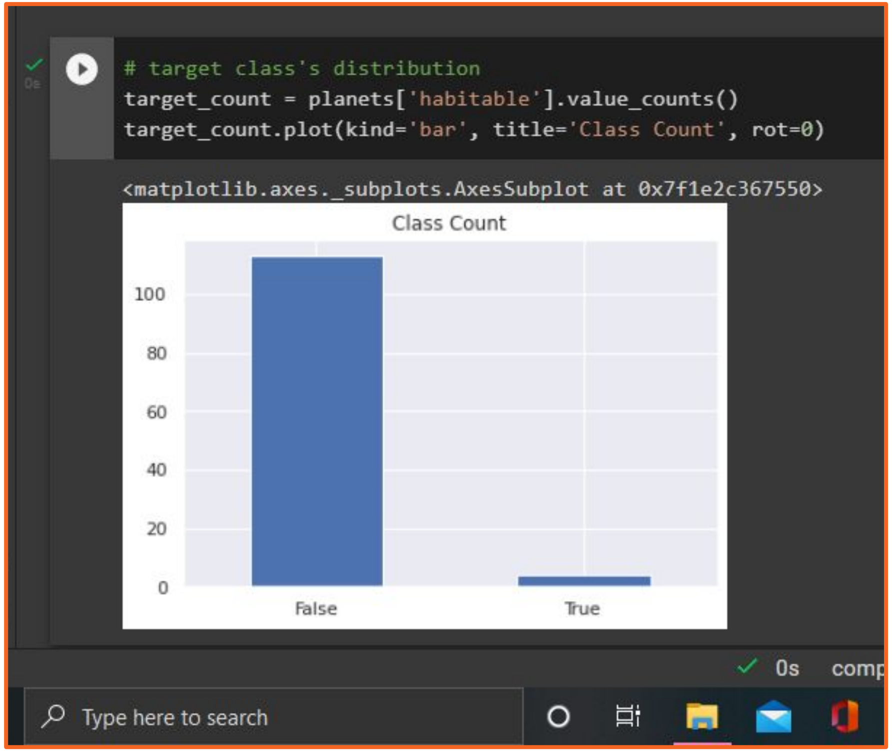
Dataset Preview

A1 P_NAME

	AO	AP	AQ	AR	AS	AT	AU	AV	AW	AX	AY	AZ	BA	BB	BC	BD	BE	BF	BG	BH	BI
1	P_ATMOS	S_NAME	S_RA	S_DEC	S_MAG	S_DISTAN	S_DISTAN	S_DISTAN	S_METALL	S_METALL	S_METALL	S_MASS	S_MASS_E	S_MASS_E	S_RADIUS	S_RADIUS	S_RADIUS	S_TYPE	S_AGE	S_AGE_ER	S_AGE
2		11 Com	185.1793	17.79287	4.74	93.37	-1.92	1.92	-0.35	-0.09	0.09	2.7	-0.3	0.3	19	-2	2	K0 III			
3		11 UMi	229.2745	71.8239	5.016	125.72	-1.97	1.97	-0.02			2.78	-0.69	0.69	29.79	-2.84	2.84	K4 III			
4		14 And	352.8226	39.2362	5.227	75.59	-0.71	0.71	-0.24	-0.03	0.03	2.2	-0.2	0.1	11	-1	1	G8 III			
5		14 Her	242.6013	43.81765	6.61	17.94	-0.01	0.01	0.41			0.9	-0.04	0.04	0.93	-0.01	0.01	K0 V			
6		16 Cyg B	295.4666	50.51753	6.25	21.15	-0.01	0.01	0.06			1.08	-0.04	0.04	1.13	-0.01	0.01	G2.5 V			
7		18 Del	314.6081	10.83929	5.506	76.38	-0.62	0.62	-0.052	-0.023	0.023	2.3			8.5			G6 III			
8		1RXS J160	242.3763	-21.083		145	-14	14				0.85	-0.1	0.2				K7 V	0.005		
9		24 Boo	217.1576	49.84485	5.58	96.25	-0.64	0.64	-0.77	-0.03	0.03	0.99	-0.13	0.19	10.64	-0.59	0.84	G3 IV	6.92	-2.75	
10		24 Sex	155.8682	-0.90224	6.441	72.21	-0.68	0.68	-0.03	-0.04	0.04	1.54	-0.08	0.08	4.9	-0.08	0.08	K0 IV	2.7	-0.4	
11		24 Sex	155.8682	-0.90224	6.441	72.21	-0.68	0.68	-0.03	-0.04	0.04	1.54	-0.08	0.08	4.9	-0.08	0.08	K0 IV	2.7	-0.4	
12		2MASS J01	20.71224	-24.664	14.24	36	-4	4				0.4	-0.05	0.05				M3.5 V	0.12	-0.01	
13		2MASS J02	34.84211	-39.4229		39.4	-2.6	2.6				0.11	-0.01	0.01	0.28	-0.01	0.01	M6	0.035	-0.005	0.
14		2MASS J04	70.43707	23.03094		140						0.02						M8.5	0.001		
15		2MASS J12	181.8895	-39.5483	20.15	64.42	-0.65	0.65				0.02						M8	0.008	-0.003	0.
16		2MASS J19	294.6359	46.06642	12.264	400.77	-6.48	6.48				0.48	-0.03	0.03							
17		2MASS J21	325.1222	16.42176		25	-10	10				0.08	-0.06	0.06	0.12	-0.04	0.05	M8.5			
18		2MASS J22	339.1022	47.86182	12.51	74	-10	10				0.6	-0.05	0.05				K7 V	0.12	-0.01	
19		30 Ari B	39.24059	24.64806	6.962	44.71	-0.1	0.1	0.19			1.93	-0.27	0.27	1.41	-0.05	0.05	F6 V			
20		4 UMa	130.0534	64.32793	4.61	73.58	-1.18	1.18	-0.25	-0.04	0.04	1.23	-0.15	0.15	18.11	-1.47	1.47	K1 III	4.604	-2	
21		42 Dra	276.4964	65.56348	4.82	90.86	-1.32	1.32	-0.46	-0.05	0.05	0.98	-0.05	0.05	22.03	-1	1	K1.5 III	9.49	-1.76	

phl_exoplanet_catalog_2019

EDA



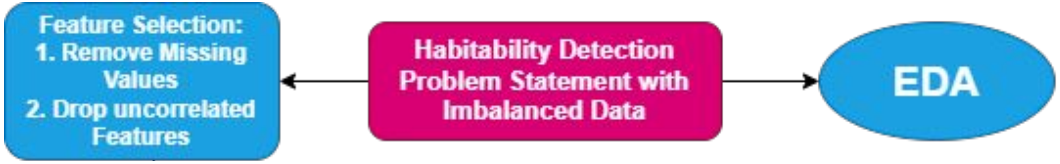
Code with Class Distributions Graph

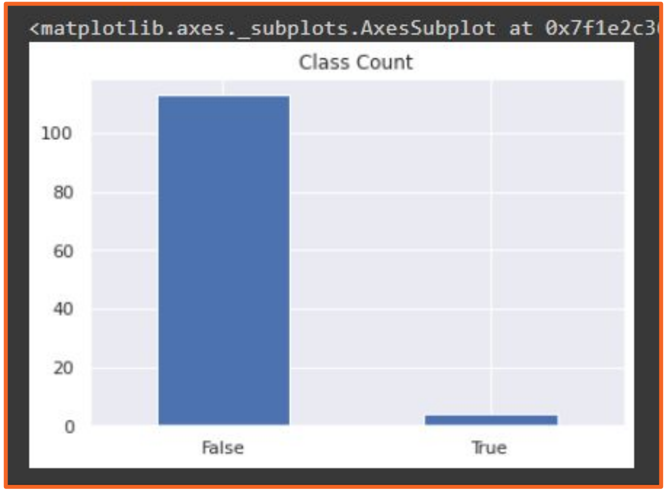
Before starting with training, we need to understand our data and for that we use Exploratory Data Analysis.

We plot class distributions first which show that our data is certainly imbalanced.

The class count distribution for false class represents non habitable exoplanets while True class represents habitable exoplanets.

Summary





- **Main Goal**
  Finding habitable exoplanets in different stellar systems. The dataset contains all modeled parameters for currently confirmed exoplanets along with a few of them which are confirmed to be habitable. Taken from **PHL Exoplanet Catalog**

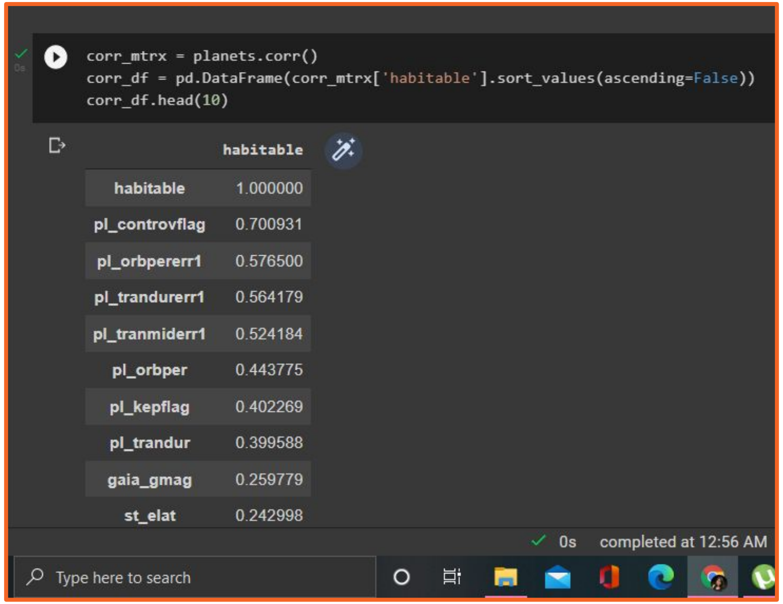
EDA: We plot class distributions first which show that our data is certainly imbalanced.

Feature Selection

Feature Selection is the process of selecting the attributes that can make the predicted variable more accurate or eliminating those attributes that are irrelevant and can decrease the model accuracy and quality.

1) Dropping attributes with >25% missing values.
2) Dropping Low Correlated attributes

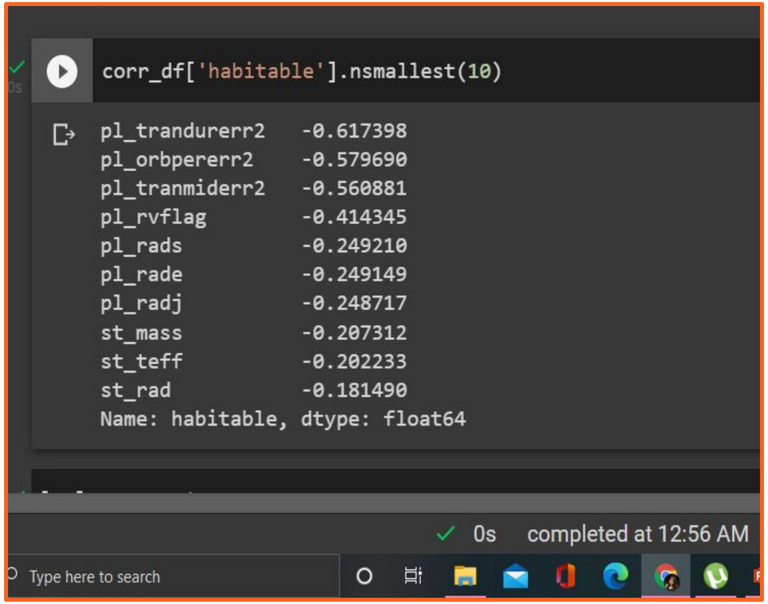
Top 10 Positive Correlated Features



```
corr_mtrx = planets.corr()
corr_df = pd.DataFrame(corr_mtrx['habitable'].sort_values(ascending=False))
corr_df.head(10)
```

	habitable
habitable	1.000000
pl_controvflag	0.700931
pl_orbpererr1	0.576500
pl_trandurerr1	0.564179
pl_tranmiderr1	0.524184
pl_orbper	0.443775
pl_kepflag	0.402269
pl_trandur	0.399588
gaia_gmag	0.259779
st_elat	0.242998

Top 10 Negative Correlated Features



```
corr_df['habitable'].nsmallest(10)
```

```
pl_trandurerr2   -0.617398
pl_orbpererr2    -0.579690
pl_tranmiderr2   -0.560881
pl_rvflag        -0.414345
pl_rads          -0.249210
pl_rade          -0.249149
pl_radj          -0.248717
st_mass          -0.207312
st_teff          -0.202233
st_rad           -0.181490
Name: habitable, dtype: float64
```


- We take the correlation of variables wrt “habitability” i.e. target to get the most important features from the dataset.
- For this we derive Top 10 features with positive correlation to the target feature and Least 10 correlated features.

ML Model Implementation with imbalanced Data

- K Nearest Neighbor algorithm falls under the Supervised Learning category and is used for classification (most commonly) and regression.
- We implement a KNN classifier and the accuracy obtained is *96.67%* on test data.

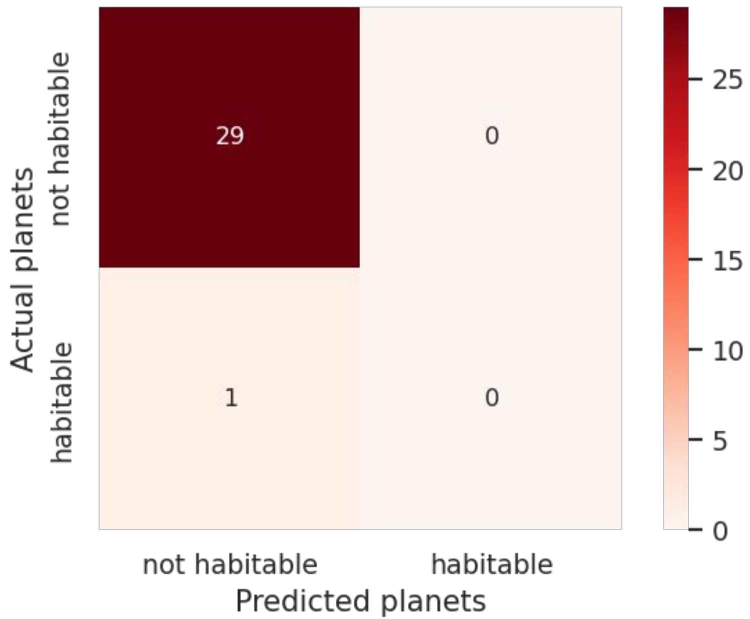
K-Nearest Neighbour

```
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)

knn_acc = round(knn.score(X_train, y_train) * 100, 2)
knn_acc_test = round(accuracy_score(y_test, y_pred) * 100, 2)
print(f'Train Accuracy of KNN: % {knn_acc}')
print(f'Test Accuracy of KNN: % {knn_acc_test}')

# Get precision, recall, and f1
precision, recall, f1, support = score(y_test, y_pred, average = 'macro')
print(f'Precision : {precision}')
print(f'Recall : {recall}')
print(f'F1-score : {f1}')
```

```
Train Accuracy of KNN: % 96.55
Test Accuracy of KNN: % 96.67
Precision : 0.4833333333333334
Recall : 0.5
F1-score : 0.4915254237288135
```

Results: Confusion Matrix

- To finally check the performance, we use a confusion matrix which shows the actual vs predicted values.
- However, as we can see the confusion matrix of a 96% accuracy model still classifies the 1 habitable planet as not habitable.
- This is due to directly feeding imbalance class to our training model.
- Hence, proved that high-accuracy models also mis classify data in case of imbalance datasets.



ML Model Implementation with imbalanced Data

Support Vector Classifier

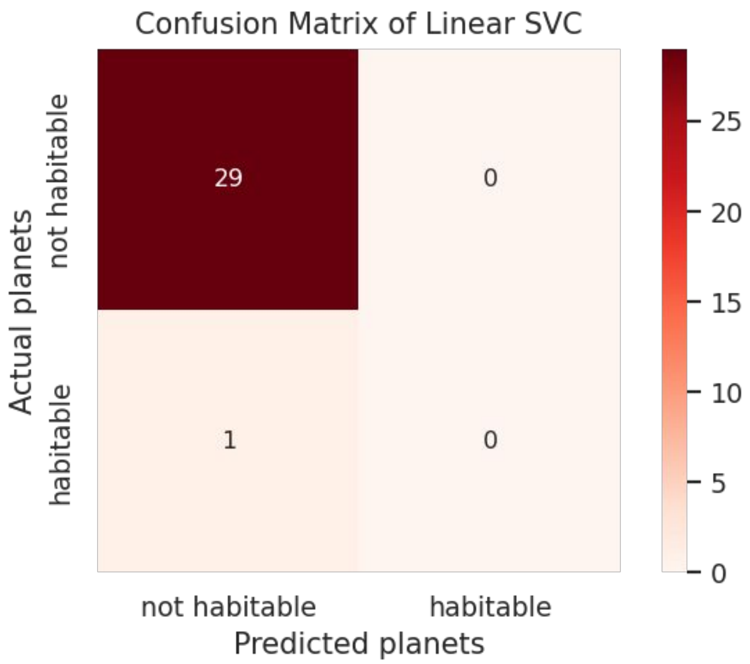
- The objective of a Linear SVC (Support Vector Classifier) is to fit to the data we provide, returning a "best fit" hyperplane that divides, or categorizes, ourdata.
- We implement a SVC classifier and the accuracy obtained is *96.67%* on test data

```
[91] svc = SVC(kernel='linear', gamma=0.001, C=100, probability=True)
     svc.fit(X_train, y_train)
     y_pred_s = svc.predict(X_test)

     svc_acc = round(svc.score(X_train, y_train) * 100, 2)
     svc_acc_test = round(accuracy_score(y_test, y_pred_s) * 100, 2)

     print(f'Train Accuracy Score of LinearSVC: % {svc_acc}')
     print(f'Test Accuracy Score of LinearSVC: % {svc_acc_test}')

     # Get precision, recall, f1 scores
     precision, recall, f1, support = score(y_test, y_pred_s, average='macro')
     print(f'Precision : {precision}')
     print(f'Recall    : {recall}')
     print(f'F1-score   : {f1}')
```

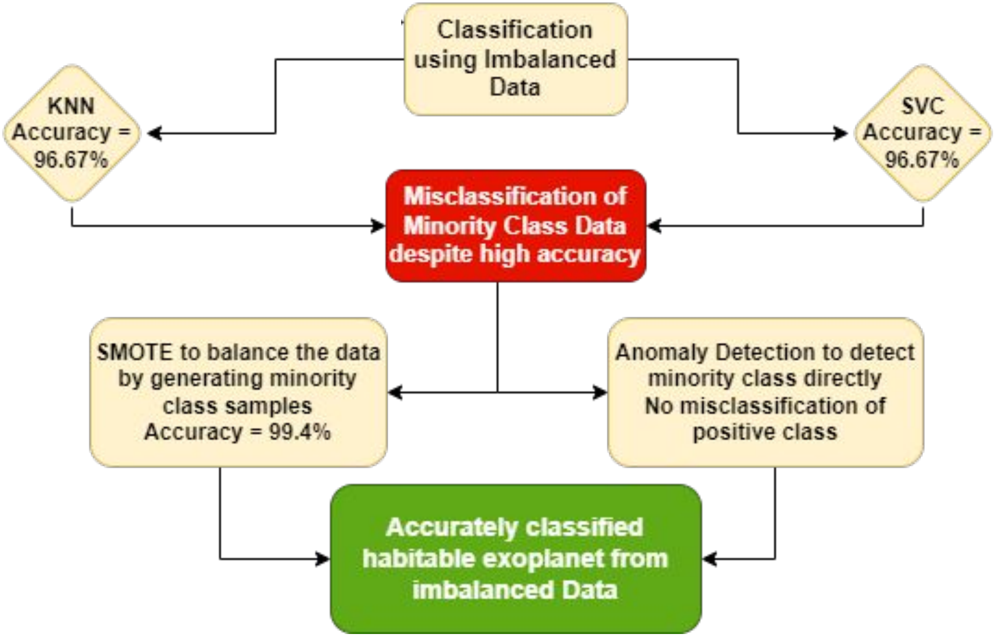
```
Train Accuracy Score of LinearSVC: % 100.0
Test Accuracy Score of LinearSVC: % 96.67
Precision : 0.48333333333333334
Recall    : 0.5
F1-score   : 0.4915254237288135
```

Results: Confusion Matrix

- To finally check the performance, we use a confusion matrix which shows the actual vs predicted values.
- However, as we can see the confusion matrix of a 96% accuracy model still classifies the 1 habitable planet as not habitable.
- This is due to directly feeding imbalance class to our training model.
- Hence, proved that high-accuracy models also mis classify data in case of imbalance datasets.

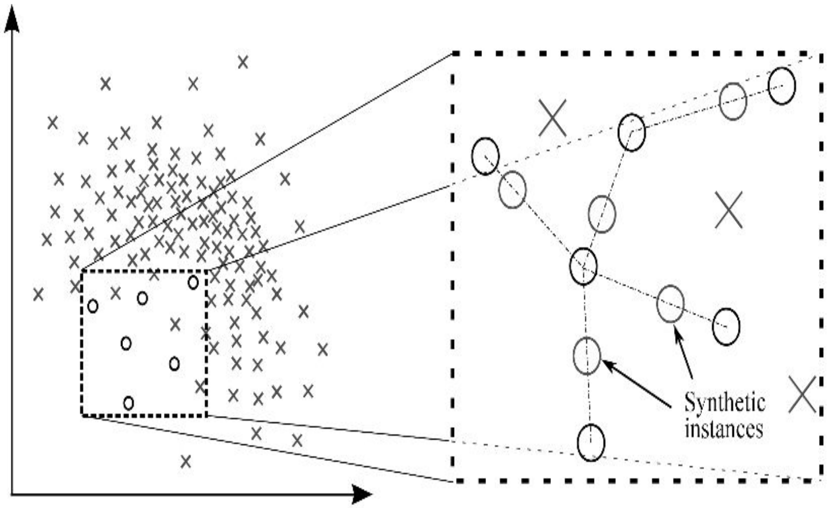


Flowchart Diagram of the steps



SMOTE technique :

- This technique is used to solve the problem of unbalanced data, also called Synthetic Minority Over-sampling Technique.
- In this technique a subset of data is taken from the minority class and then new synthetic similar instances are created.
- Then these synthetic instances are added to the original dataset and this new dataset is used to train the model.
- This technique is used in order to avoid the overfitting.



Lets us suppose at first we have the following data :

Total Observations = 1000

Positive Observations = 10

Negative Observations = 990

Event Rate= 1 % (the probability of finding a habitable exoplanet among all)

Now a sample of 7 instances is taken from minority class and similar synthetic instances are created 20 times

So after this the new dataset created is as follow :

Positive observations = 140

Negative Observations = 990

Event Rate = 140/1130 = 12.3 %

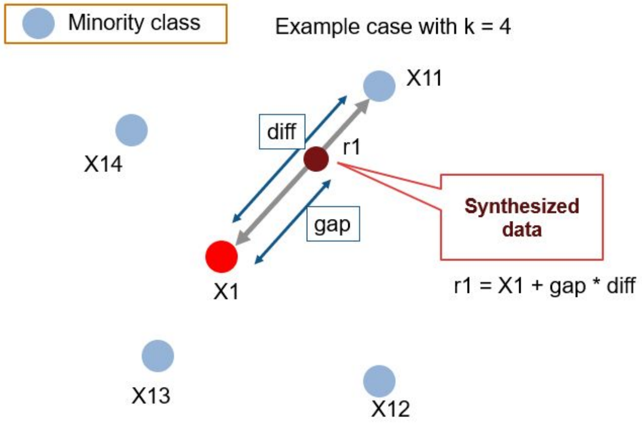
Intuition :

1. Setting the minority class set **A** , for each x € A , the k-nearest neighbors of x are obtained by calculating the Euclidean distance between **x** and every other sample in set **A**.

2. The sampling rate **N** is set according to the imbalanced proportion. For each x € A , **N** examples (i.e x1, x2, …xn) are randomly selected from its k-nearest neighbors, and they construct the set A_1 .

3. For each example x_k € A (k=1, 2, 3…N), the following formula is used to generate a new example:

$$x' = x + rand(0,1) * |x - x_k|$$



in which rand(0, 1) represents the random number between 0 and 1

Example :

Let us consider a sample at position (6,4) and let (4,3) be its nearest neighbor.

Now , (6,4) is the sample for which k-nearest neighbors are being identified and (4,3) is one of its k-nearest neighbors.

Let : f1_1 = 6 , f2_1 = 4   f2_1 - f1_1 = -2

f1_2 = 4 , f2_2 = 3   f2_2 - f1_2 = -1

Then the new samples will be generated as :

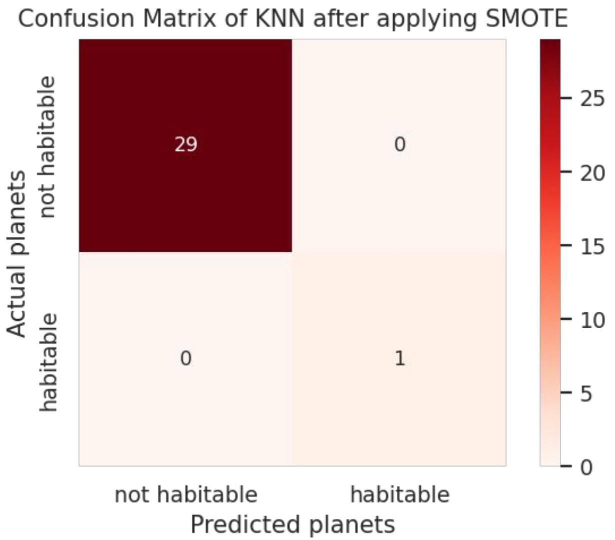
f1',f2' = (6,4) + rand(0-1) * (-2,-1)

where rand(0-1) generates a vector of two random numbers between 0 and 1.

Before SMOTE

After SMOTE

KNN Classification & Confusion Matrix after Applying SMOTE




```python
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_smoted, y_smoted)
y_pred = knn.predict(X_test)

knn_acc = round(knn.score(X_smoted, y_smoted) * 100, 2)
knn_acc_test = round(accuracy_score(y_test, y_pred) * 100, 2)
print(f'Train Accuracy of KNN: % {knn_acc}')
print(f'Test Accuracy of KNN: % {knn_acc_test}')

# Get precision, recall, and f1
precision, recall, f1, support = score(y_test, y_pred, average = 'macro')
print(f'Precision : {precision}')
print(f'Recall : {recall}')
print(f'F1-score : {f1}')
```

```
Train Accuracy of KNN: % 99.4
Test Accuracy of KNN: % 100.0
Precision : 1.0
Recall : 1.0
F1-score : 1.0
```

We can see there is no misclassification of Positive minority class anymore, despite such low frequency of occurrence.

Since, Minority class is now being oversampled and the two classes are fed to classifier after being balanced.

Anomaly Detection

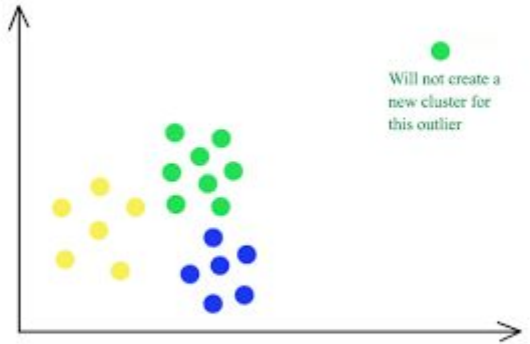
Anomalies are data points in a dataset that stand out from the rest and contradict the data's expected behaviour. These data points or observations differ from the dataset's typical patterns of behaviour.

Anomaly detection is useful for detecting fraudulent transactions, detecting diseases, and handling case studies with a high-class imbalance.

In case of high dimensional data, SMOTE is not much effective but anomaly detection works better.

Anomaly Detection

- Since, a habitable exoplanet would be 1 out of 1000, we can consider such imbalance dataset problems as problems of anomaly detection.
- Anomaly detection, also called outlier detection, is the identification of unexpected events, observations, or items that differ significantly from the pattern.
- This is done by first clustering the data like an unsupervised approach and forming clusters of data which are similar to each other compared to other points.
- The data points which are anomalies happen to be outside and away from such clusters and these outliers can be then detected.



The green point is an outlier in such case.

ANOMALY DETECTION

1. We have used the isolation forest algorithm for outlier detection.
2. An Isolation Forest is an ensemble learning anomaly detection algorithm, that is especially useful at detecting outliers in **high dimensional datasets.**
3. Its working is based on decision trees and it calculates the anomaly score which tells us how likely a point is an outlier. Finally, it filters out the values of outliers as '-1' and others as '+1'.
4. Since, it assigns -1 to anomalies that is our positive class and '+1' to negative class we replace them with 1 and 0 respectively.

```
[9] from sklearn.ensemble import IsolationForest
    clf = IsolationForest(max_samples=100, contamination=0.1, random_state=42)
    clf.fit(X_train)
    # predictions
    y_pred_train = clf.predict(X_train)
    y_pred_test = clf.predict(X_test)
```

```
y_pred_train
np.place(y_pred_train, y_pred_train>0, [0])
np.place(y_pred_train, y_pred_train<0, [1])
```

```
array([ 1, -1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,
        1,  1, -1,  1,  1, -1,  1,  1,  1,  1, -1, -1,  1,  1,  1,  1,  1,
        1,  1])
```

```
y_pred_test
```

```
array([ 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1])
```

```
[17] #To convert '-1' values to 1 as they are the anomaly or poisitive class
     #convert '1' values to 0 since they are the negative class
     np.place(y_pred_test, y_pred_test>0, [0])
     np.place(y_pred_test, y_pred_test<0, [1])
     y_pred_test
```

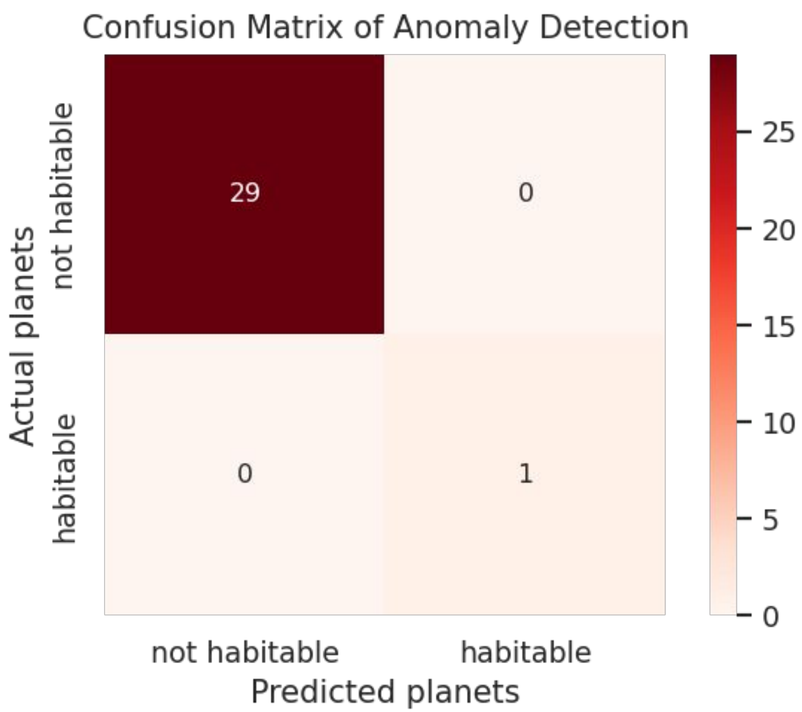
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0])
```

✓ 0s completed at 1:10 AM

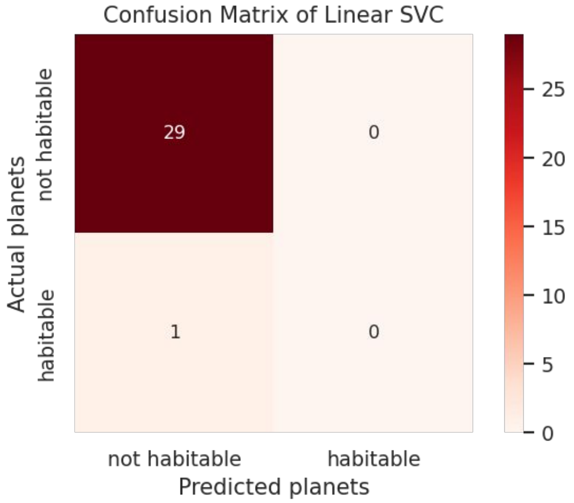
Confusion Matrix for Anomaly Detection

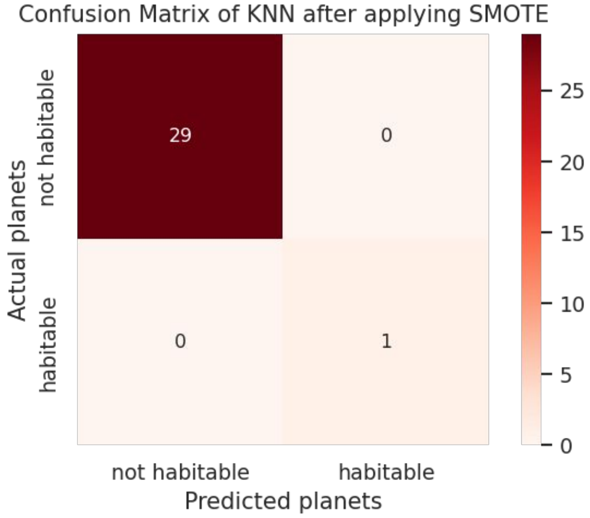
We can see there is no misclassification of Positive minority class anymore, Despite such low frequency of occurrence.

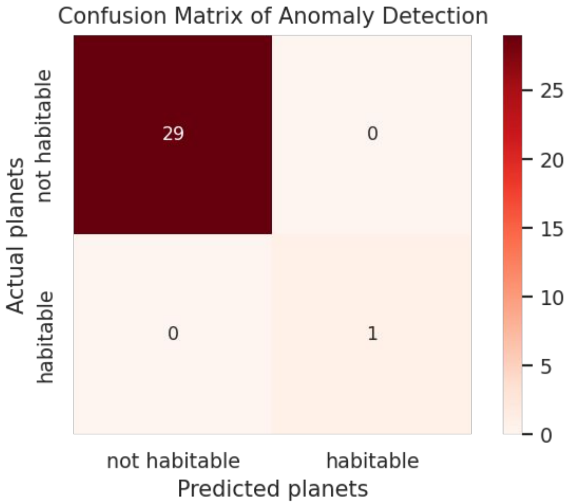
Since, Minority class is now an anomaly which is being detected directly.



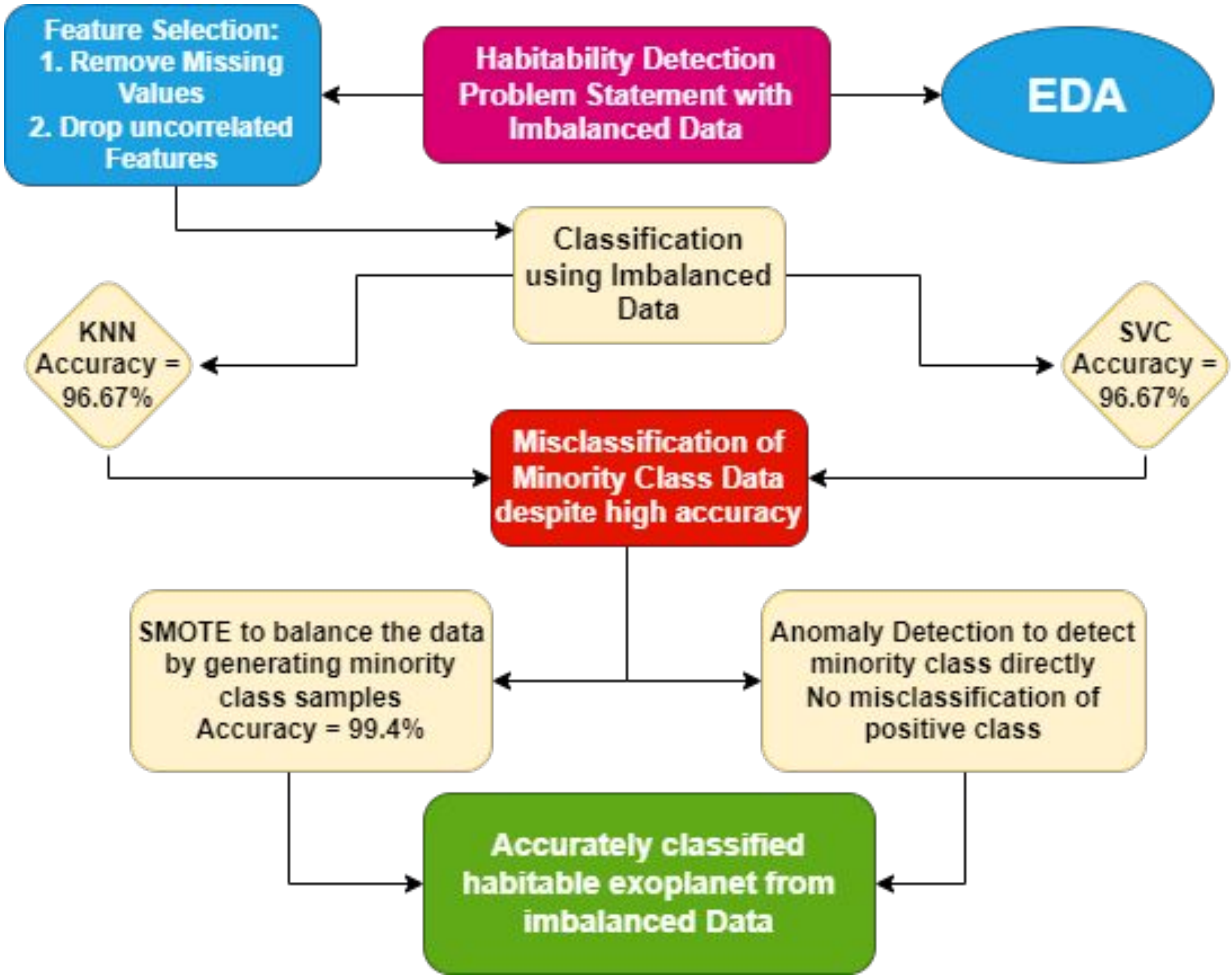
Final Comparison







Conclusion



Feature Selection:
1. Remove Missing Values
2. Drop uncorrelated Features
Habitability Detection Problem Statement with Imbalanced Data
EDA
Classification using Imbalanced Data
KNN Accuracy = 96.67%
SVC Accuracy = 96.67%
Misclassification of Minority Class Data despite high accuracy
SMOTE to balance the data by generating minority class samples
Accuracy = 99.4%
Anomaly Detection to detect minority class directly
No misclassification of positive class
Accurately classified habitable exoplanet from imbalanced Data

References:

1. Sturrock, George Clayton, Brychan Manry, and Sohail Rafiqi. "Machine learning pipeline for exoplanet classification." *SMU Data Science Review* 2.1 (2019): 9.
2. Malik, Abhishek, Benjamin P. Moster, and Christian Obermeier. "Exoplanet detection using machine learning." *Monthly Notices of the Royal Astronomical Society* (2020).
3. Sarkar, Jyotirmoy, et al. "Postulating exoplanetary habitability via a novel anomaly detection method." *Monthly Notices of the Royal Astronomical Society* 510.4 (2022): 6022-6032.
4. Jagtap, Rutuja, et al. "Habitability of Exoplanets using Deep Learning." *2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*. IEEE, 2021.
5. Girimonte, Daniela, and Dario Izzo. "Artificial intelligence for space applications." *Intelligent Computing Everywhere*. Springer, London, 2007. 235-253.
6. Block, Jessica, et al. "An unsupervised deep learning approach for satellite image analysis with applications in demographic analysis." *2017 IEEE 13th International Conference on e-Science (e-Science)*. IEEE, 2017.
7. Seager, Sara. "The future of spectroscopic life detection on exoplanets." *Proceedings of the National Academy of Sciences* 111.35 (2014): 12634-12640.
8. Seager, Sara. "Exoplanet habitability." *Science* 340.6132 (2013): 577-581.
9. https://www.analyticsvidhya.com/blog/2017/03/imbalanced-data-classification/

Thank You!