The idea behind this classification task is to identify different types of forest cover types in undisturbed forests.

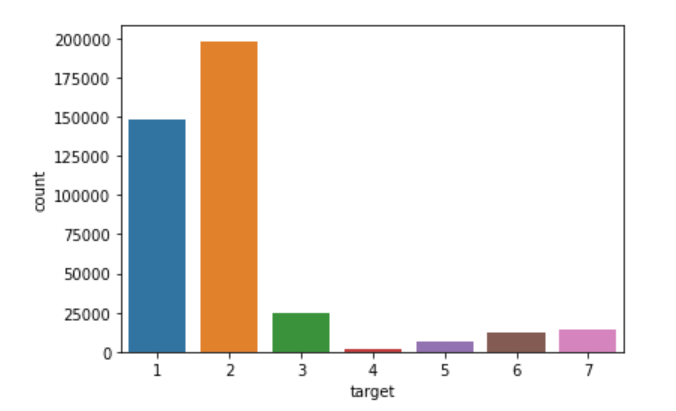
The data is complete or doesn’t need any type of data interpolation since there is not data missing.

1. **Exploratory Data Analysis – EDA**

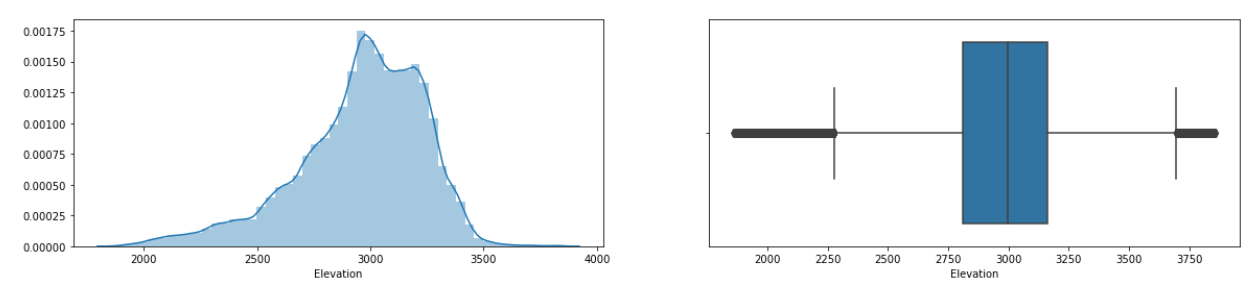
Conducting EDA to understand our data, the first thing to do is to check whether there is or not missing data,

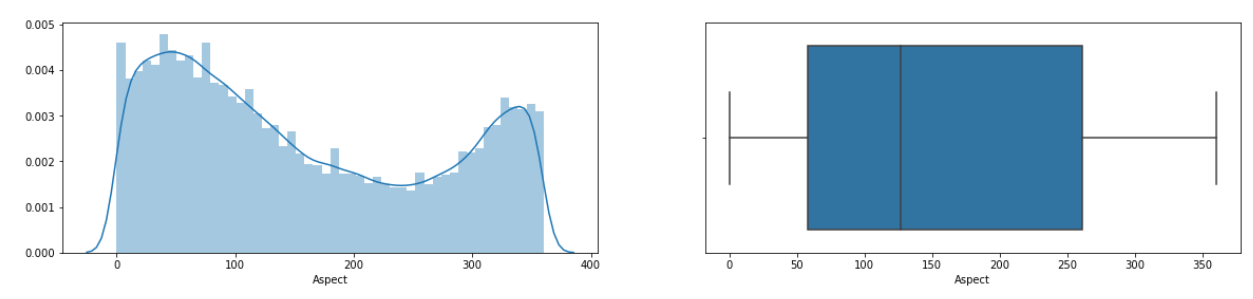
lets join our target labeling or classification into the training data to try to find patterns or relationships between the variables. From the image we can see that the cover type 1 and 2 are the most predominant features over the dataset, while labelings from 3 to 7 are much lower.

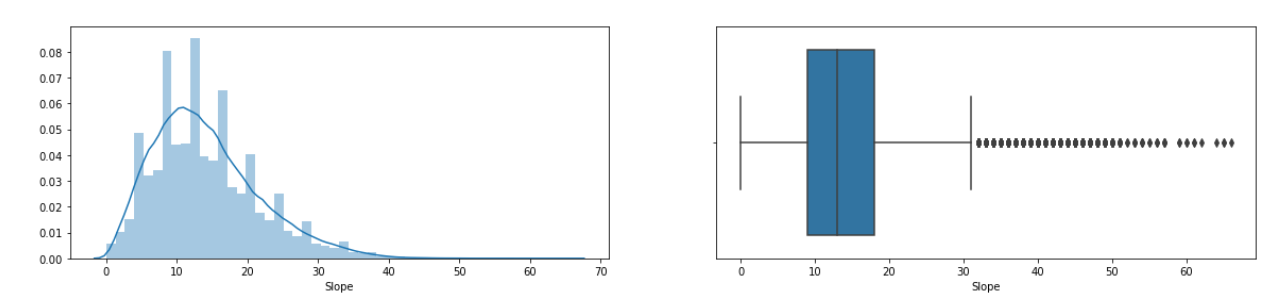
From the Figure below, we can see that the dataset is not completely balanced, there are more counted classes over class 1 and 2 in comparison to classes 3 to 7.

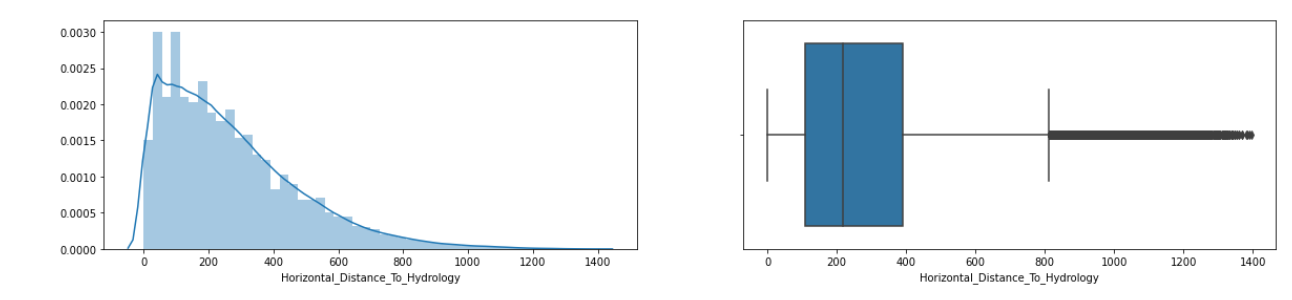


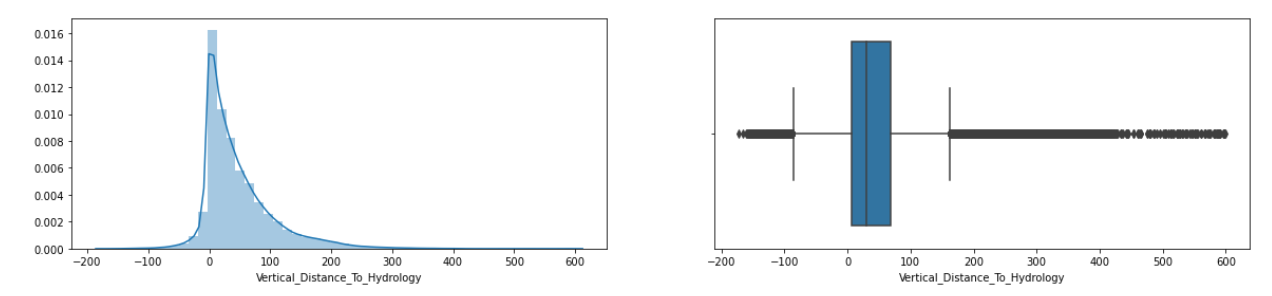
Plotting all the variables using the seaborn library, we can see depicted below the distribution of all the variables one by one, next to its boxplot. From this we can infer that most of the territories have an elevation of 3000 meters, a mean aspect of 120, a horizontal distance of roughly 200 m, a slope of 15, horizontal distance to hydrology of 210, vertical distance to hydrology of 30, a horizontal distance to roadways of 2000 m, a hillshade at 9 am of 220m, a hillshade at noon of 125m, a hillshade at 3 pm of 140, a horizontal distance to fire points of 1800 m.

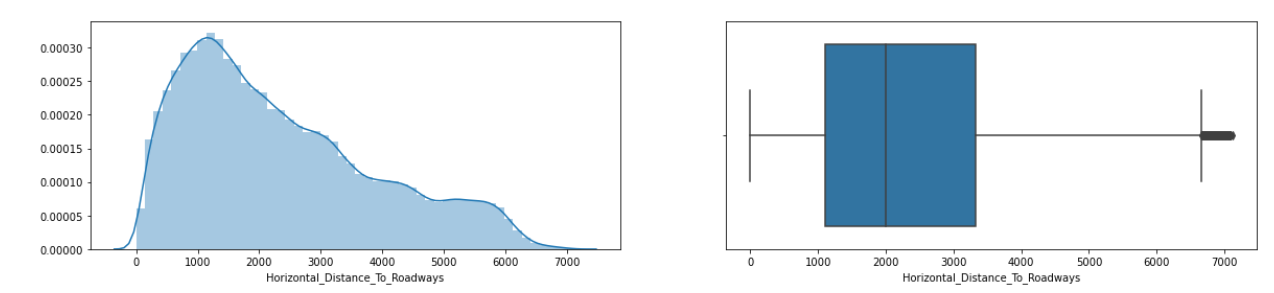


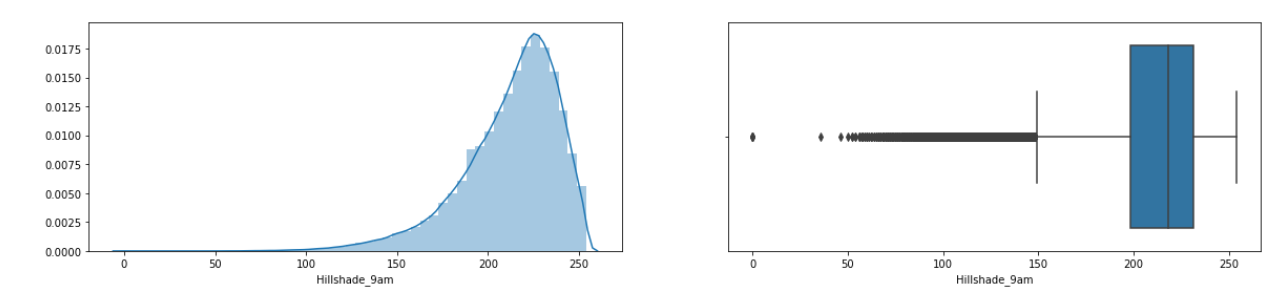


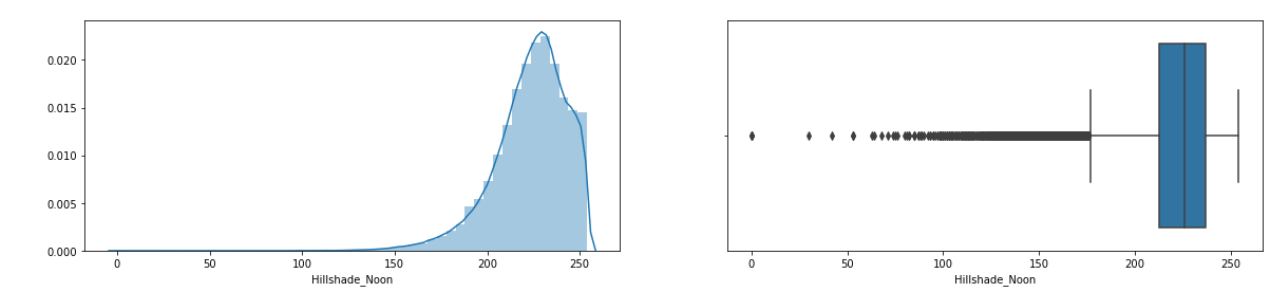


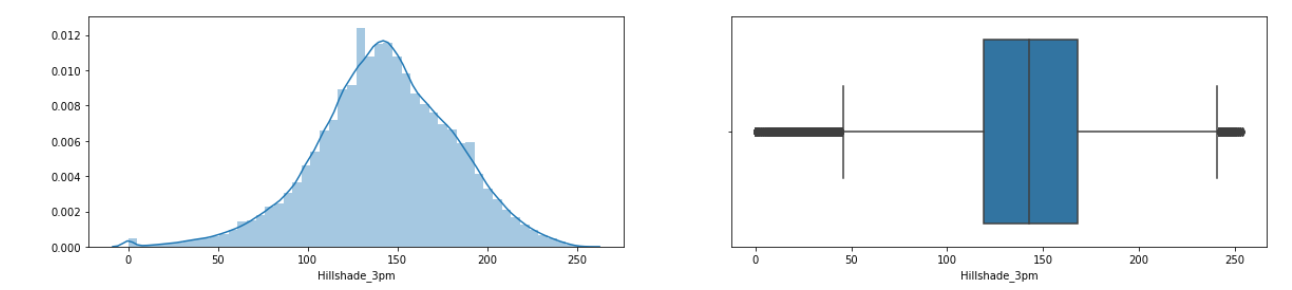


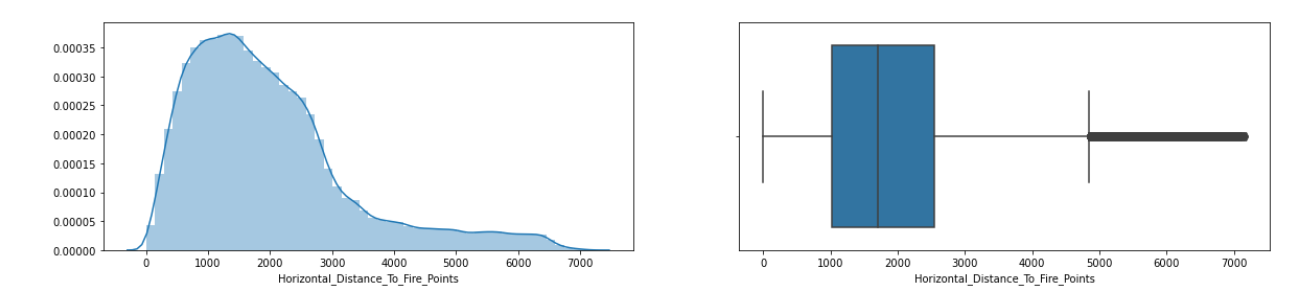


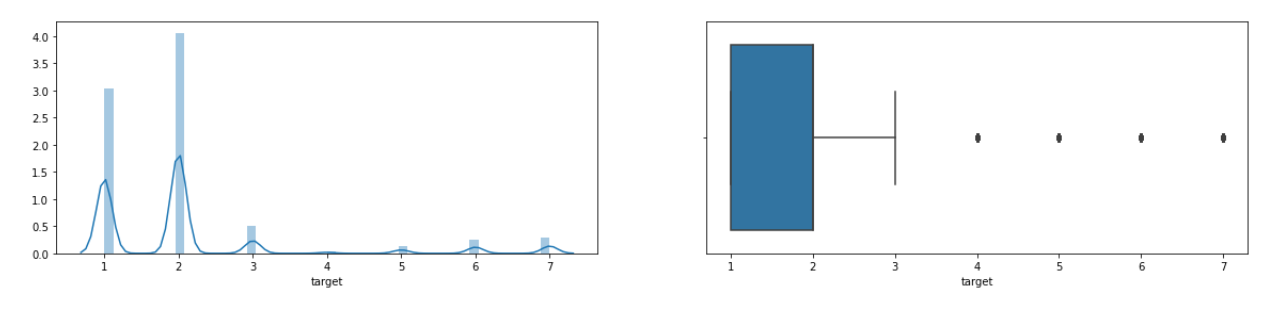




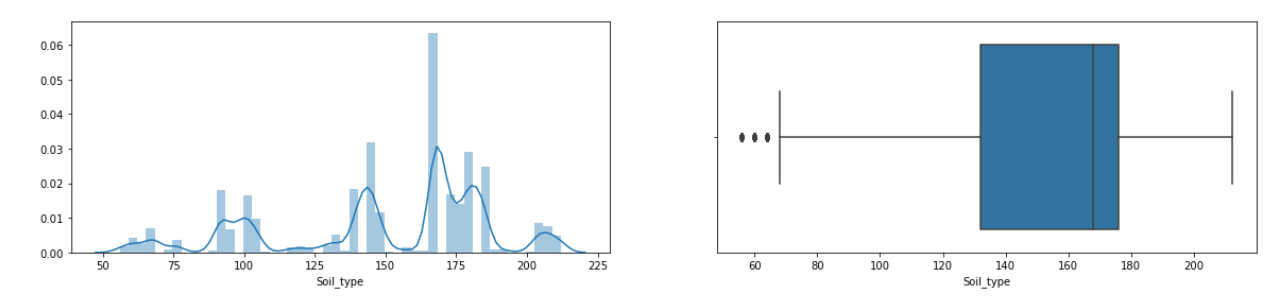




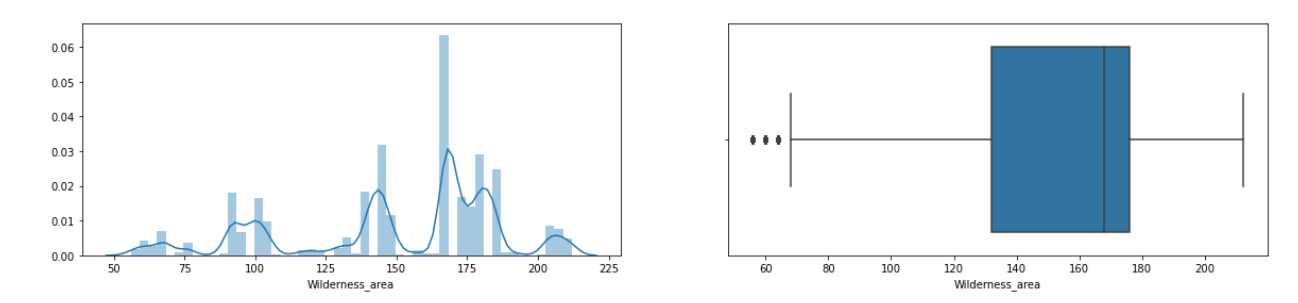




1. **Data preprocessing**
   * Convert soil type 1-40 from binary type, to number type and merged them into one single column called soil\_type.



* + Convert Wilderness areas 1-4 from binary type, to number type and merged them into one single column called wildernas\_Area.



1. **Algorithms**
   * **KNN Model**
2. **Tuning**
3. **KNN Model:** This model was initially tuned by iterating over a for-loop between 1 and 30 neighbors, showing its best performance with 3 neighbors. Afterwards, the leaf\_size and the Minkowski distance formula (p) parameters were stored in a dictionary and added to GridSearchCV function, which optimizes by cross-validated grid-search over KNN model trained with the preprocessed dataset alongside with 10-fold cross validation, the results are in section V, table 1.
4. **Results**
5. Using the KNN model the accuracy improved to 95.4554 %, when considering the pre-preprocessed data.

Changing number of neighbors to n=3, the accuracy improved to 96.0553%

1. 10-fold Cross-validation using the model\_selection mododule from sklearn and using the original dataset, yielded a precision on the leaderboard of 0.96989. Another detail about this analysis was that I used the the entire dataset, that is, since I had already tried splitting it using split at 70% of the data (for keeping the validationset appart), I tried with the whole dataset to check whether more details with the entire data were possible to detect using it as a whole.

|  |  |  |
| --- | --- | --- |
| **Model** | **Accuracy test – set using 10-fold cross-validation** | **Accuracy – leaderboard (50% of testing data)** |
| KNN model hypertuned withGridSearch, with CV=10 and 3 neighbors and over the initial preprocessed dataset |  |  |
| KNN model hypertuned withGridSearch, with CV=10 and 3 neighbors and over the original dataset. |  |  |
|  |  |  |

1. **Overview of relevant results**
2. **Conclusions**