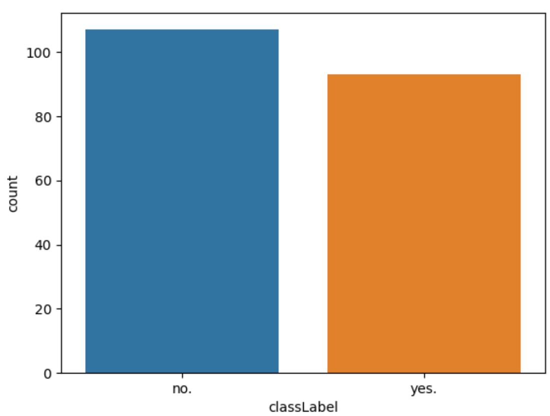
Binary Classification - Comparing Multiple Methods

On the data set: It is not known what the data provided is about or what each feature means. Column names are arbitrary, making it difficult to perform feature selection.

	variable2	variable3	variable8	variable11	variable14	variable15	variable17	variable19
count	191.000000	191.000000	191.000000	191.000000	191.000000	191.000000	1.910000e+02	191.000000
mean	32.206230	0.000473	2.026545	2.921466	194.036649	688.942408	1.940366e+06	0.513089
std	12.332321	0.000499	2.675925	4.290640	204.266811	1635.537311	2.042668e+06	0.501142
min	15.920000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000e+00	0.000000
25%	22.710000	0.000092	0.250000	0.000000	71.500000	0.000000	7.150000e+05	0.000000
50%	29.670000	0.000275	1.085000	0.000000	160.000000	5.000000	1.600000e+06	1.000000
75%	39.170000	0.000752	2.687500	5.000000	270.000000	540.000000	2.700000e+06	1.000000
max	76.750000	0.002508	20.000000	20.000000	2000.000000	10000.000000	2.000000e+07	1.000000

Stats on the validation data after cleaning

About the Data:



Two separate files were provided one containing training data and the other validation.

As shown this is the number of no to yes. The set is not balanced perfectly, however not warranting dedicated solutions for imbalanced cases.

Data Set preparation:

. There was also a lot of missing values, dropped the column with a number of missing values exceeding 2000, which was more than half the number rows.

```
variable1      39
variable2      39
variable3       0
variable4      64
variable5      64
variable6      66
variable7      66
variable8       0
variable9       0
variable10      0
variable11      0
variable12      0
variable13      0
variable14    100
variable15      0
variable17    100
variable19      0
classLabel      0
dtype: int64
```

Number of NaNs after dropping column 18

I then proceeded to drop the rows with missing values, their count insignificant to the total number of rows.

The data had to be encoded in order to be fed to the model, label encoding was less messy, easier to implement and gave a better accuracy.

Testing Algorithms:

Classifiers:

- Logistic Regression
- Decision Tree
- Support Vector Machine
- Random Forest
- K-Nearest Neighbors

Scoring:

- accuracy score

Why use accuracy score?

Accuracy looks at correctly classified observations **both positive and negative** according to "F1 Score vs ROC AUC vs Accuracy vs PR AUC: Which Evaluation Metric Should You Choose?" by Jakub Czakon. Because true negatives and true positives are equally important then accuracy is the metric that is most suitable.

Conclusion:

From my observation the accuracy of the classification on the training data is very high compared to the test data. This might be because of *overfitting*.

References in this Project:

- https://medium.com/@rrfd/cleaning-and-prepping-data-with-python-for-data-science-best-practices-and-helpful-package
- https://mclguide.readthedocs.io/en/latest/sklearn/binary.html
- https://www.kaggle.com/klaudiajankowska/binary-classification-multiple-method-comparisons-af1edfbe2a3
- https://hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008f
- https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.get_dummies.html
- https://neptune.ai/blog/f1-score-accuracy-roc-auc-pr-auc
- https://machinelearningmastery.com/save-load-machine-learning-models-python-scikit-learn/
- https://medium.com/datadriveninvestor/deploy-your-machine-learning-model-using-flask-made-easy-now-635d2f12c50c
- https://towardsdatascience.com/categorical-encoding-using-label-encoding-and-one-hot-encoder-911ef77fb5bd
- https://medium.com/@contactsunny/label-encoder-vs-one-hot-encoder-in-machine-learning-3fc273365621