Machine-Learning - Exoplanet Exploration

Over a period of nine years in deep space, the NASA Kepler space telescope has been out on a planet-hunting mission to discover hidden planets outside of our solar system.

Use Jupyter Notebook, Pandas, Matplotlib, and Scikit-Learn to create machine learning models capable of classifying candidate exoplanets from the raw dataset.

Preprocess the Data

- Preprocess the dataset prior to fitting the model.
- Perform feature selection and remove unnecessary features.
- Use MinMaxScaler to scale the numerical data.
- Separate the data into training and testing data.

Tune Model Parameters

- Use GridSearch to tune model parameters.
- Tune and compare at least two different classifiers.

Assumptions

In this assignment, Machine Learning models, capable of classifying candidate exoplanets, were created.

The data set used for the machine learning model is available at Exoplanet Data Source, but the csv located in the folder Notebook was used.

The algorithms used for the models were:

- Logistic Regression
- Support Vector Machines (SVM) with Linear, Gaussian, and Polynomial kernels. The SVM was suggested in the starter code, but without kernel specification, so I tried three kernels.
- Random Forests. A third classifier was proposed, based on the results of the SVM.

- Deep Learning and Decision Tree Models were used to compare with other models.

Also, a hyper-parameter tuning with the *meta-estimator* function GridSearchCV() was used.

Evaluate Model Performance Results:

1. Model-1: Logistic Regression

    - Logistic Regression - Training Data Score: 0.8565706656494373
    - Logistic Regression - Testing Data Score: 0.8569794050343249
    - Logistic Regression – After Tuning: 0.8809842525414971

2. Model-2: SVM Model (Kernels: Linear, Gaussian, and Polynomial)
    - SVM Linear - Training Data Score: 0.8445546442876216
    - SVM Linear - Testing Data Score: 0.8323798627002288
    - SVM Linear – After Tuning: Best Training Score: 0.8815571354761715

	precision	recall	f1-score	support
FALSE POSITIVE	0.84	0.68	0.75	422
CONFIRMED	0.74	0.86	0.80	450
CANDIDATE	0.99	1.00	0.99	876
accuracy			0.89	1748
macro avg	0.86	0.85	0.85	1748
weighted avg	0.89	0.89	0.88	1748

    - SVM Gaussian - Training Data Score: 0.8306313179477398
    - SVM Gaussian - Testing Data Score: 0.8569794050343249
    - SVM Gaussian – After Tuning: Best Training Score: 0.8319684686979238

	precision	recall	f1-score	support
FALSE POSITIVE	0.74	0.56	0.64	422
CONFIRMED	0.66	0.80	0.72	450
CANDIDATE	0.99	1.00	0.99	876
accuracy			0.84	1748
macro avg	0.80	0.79	0.79	1748
weighted avg	0.84	0.84	0.84	1748

- SVM Polynomial - Training Data Score: 0.845317566278848
- SVM Polynomial - Testing Data Score: 0.8535469107551488
- SVM Polynomial – After Tuning: Best Training Score: 0.5954607805325318

	precision	recall	f1-score	support
FALSE POSITIVE	0.00	0.00	0.00	422
CONFIRMED	0.51	0.98	0.67	450
CANDIDATE	0.98	1.00	0.99	876
accuracy			0.75	1748
macro avg	0.50	0.66	0.55	1748
weighted avg	0.62	0.75	0.67	1748

3. Model-3: Random Forest
    - Random Forest - Training Data Score: 0.9927522410833493
    - Random Forest - Testing Data Score: 0.8724256292906178
    - Random Forest – After Tuning: Best Training Score: 0.8909010035002437

    - Random Forest with feature importance – Top 5
        - (0.10853029754294186, 'koi_fpflag_co'),
        - (0.0879774956366661, 'koi_fpflag_nt'),
        - (0.057873736655589224, 'koi_prad'),
        - (0.05621385985556862, 'koi_fpflag_ss'),
        - (0.05552266135021724, 'koi_model_snr'),
        - (0.03760885434958448, 'koi_fpflag_ec'),

- Random Forest - Accuracy Score 40 features: 0.8930205949656751
- Random Forest - Accuracy Score 10 features: 0.8975972540045767

4. Model-4: Deep Learning Model
    - Deep Learning - Accuracy Score: 0.8878718614578247
    - Loss: 0.23645108938217163

5. Model-5: Decision Tree Model
    - Decision Tree - Testing Data Score: 0.8501144164759725

Conclusion:

The Logistic Regression Model preformed well at the 85.70%, however even using GridSearchCV to tune the model's parameters, changing C-values, and increasing the number of iterations did not improve score significantly. The improvement was only about 2.4%. That is overall after tuning, logistic regression model performed at 88.1%.

Amongst the SVM Linear kernel, the SVM Gaussian kernel (Radial Basis Function), and the SVM Polynomial kernel, the Linear kernel performed better at 88.2% when predicting classes False Positive and confirmed. Even after tuning the model, Linear kernel improved by 5% while other two kernels did not perform well after tuning the SVM models.

The Random Forest model performed accurately at 89.09%, and for deeper analysis the feature importances were obtained in this model at the 0.03 threshold. In this analysis, accuracy scores were calculated for the original model (40 features - 0.8930205949656751) and for a model that contained only those features with threshold=0.03 (10 features - 0.8975972540045767). Therefore, for a small difference in accuracy (89.30 % vs. 89.80%) the number of features could be considerably reduced in similar models.

In the Deep Learning model analysis, the accuracy score is 88.79% and the Decision Tree model the performed with accuracy at 85.01%.

Comparing all the above models, the Random Forest Model (89.09) performed a little better than Deep Learning (88.79%), SVM Linear Model (88.15%), and Logistic Regression model (88.10%) when predicting classes. Although the Random Forest model was quite successful since there was no significant difference in the score even after

hyper-tuning compared to other models. It would be interesting and important to train and test additional models for accuracy and reliability with StandardScaler to scale the data might results better numbers for the training and testing scores.